Intelligent Airbag Deployment

by

Alan Kenyon
Dominic Nicoletti
Lindsay Mosbrucker
Hiram Lew
Steven Arms

College of Engineering
California Polytechnic State University
San Luis Obispo
2012
Statement of Disclaimer

Since this project is a result of a class assignment, it has been graded and accepted as fulfillment of the course requirements. Acceptance does not imply technical accuracy or reliability. Any use of information in this report is done at the risk of the user. These risks may include catastrophic failure of the device or infringement of patent or copyright laws. California Polytechnic State University at San Luis Obispo and its staff cannot be held liable for any use or misuse of the project.
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Tables</td>
<td>5</td>
</tr>
<tr>
<td>Table of Figures</td>
<td>Error! Bookmark not defined.</td>
</tr>
<tr>
<td>Executive Summary</td>
<td>7</td>
</tr>
<tr>
<td>Chapter 1: Introduction</td>
<td>8</td>
</tr>
<tr>
<td>Sponsor Background and Needs</td>
<td>8</td>
</tr>
<tr>
<td>Formal Problem Definition</td>
<td>8</td>
</tr>
<tr>
<td>Objective/Specification Development</td>
<td>8</td>
</tr>
<tr>
<td>Project Management</td>
<td>9</td>
</tr>
<tr>
<td>Chapter 2: Background</td>
<td>10</td>
</tr>
<tr>
<td>Existing Products</td>
<td>10</td>
</tr>
<tr>
<td>Current State of the Art</td>
<td>11</td>
</tr>
<tr>
<td>Delphi Occupant Classification Systems (OCS)</td>
<td>11</td>
</tr>
<tr>
<td>Nissan Facial Monitoring System</td>
<td>11</td>
</tr>
<tr>
<td>Ford Smart Car</td>
<td>11</td>
</tr>
<tr>
<td>Chapter 3: Design Development</td>
<td>13</td>
</tr>
<tr>
<td>Discussion of Conceptual Designs</td>
<td>13</td>
</tr>
<tr>
<td>Research</td>
<td>13</td>
</tr>
<tr>
<td>Design Requirements and Specifications</td>
<td>13</td>
</tr>
<tr>
<td>Ideation</td>
<td>13</td>
</tr>
<tr>
<td>Concept selection &amp; justification</td>
<td>14</td>
</tr>
<tr>
<td>Supporting Preliminary Analysis</td>
<td>14</td>
</tr>
<tr>
<td>Concept #1 Steering Wheel Monitoring</td>
<td>14</td>
</tr>
<tr>
<td>Concept #2 Red Eye Detection</td>
<td>15</td>
</tr>
<tr>
<td>Concept #3 Xbox Kinect Biometrics Scan</td>
<td>16</td>
</tr>
<tr>
<td>Proof-of-Concept Analysis Initial Testing</td>
<td>17</td>
</tr>
<tr>
<td>Chapter 4 Description of the Final Design</td>
<td>24</td>
</tr>
<tr>
<td>Detailed Design Description</td>
<td>24</td>
</tr>
<tr>
<td>Testbed</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>26</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Monitoring System</td>
<td>26</td>
</tr>
<tr>
<td>System Architecture</td>
<td></td>
</tr>
<tr>
<td>Main Program</td>
<td>29</td>
</tr>
<tr>
<td>Biometric Measurements</td>
<td>29</td>
</tr>
<tr>
<td>Cost breakdown</td>
<td>32</td>
</tr>
<tr>
<td>Any special safety considerations</td>
<td>33</td>
</tr>
<tr>
<td>Any maintenance and repair considerations</td>
<td>33</td>
</tr>
<tr>
<td>Chapter 5 Product Realization</td>
<td>34</td>
</tr>
<tr>
<td>Chapter 6 Design Verification Testing</td>
<td>36</td>
</tr>
<tr>
<td>Test Descriptions with photos</td>
<td>36</td>
</tr>
<tr>
<td>Detailed Results</td>
<td>37</td>
</tr>
<tr>
<td>Specification Verification Checklist</td>
<td>39</td>
</tr>
<tr>
<td>Relating Working Model and Kinect: Injury Scales</td>
<td>39</td>
</tr>
<tr>
<td>Chapter 7 Conclusions and Recommendations</td>
<td>53</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>54</td>
</tr>
<tr>
<td>References</td>
<td>55</td>
</tr>
<tr>
<td>Appendices</td>
<td>57</td>
</tr>
<tr>
<td>Appendix A QFD, Decision Matrices etc.</td>
<td>57</td>
</tr>
<tr>
<td>Appendix B: Bill of Materials, Detailed Part Drawing</td>
<td>58</td>
</tr>
<tr>
<td>Appendix C Detailed Supporting Analysis</td>
<td>61</td>
</tr>
<tr>
<td>Appendix D Gantt Chart</td>
<td>24</td>
</tr>
<tr>
<td>Appendix E Code Listing</td>
<td>244</td>
</tr>
</tbody>
</table>
List of Tables
Table 1.......................................................................................................................... 12
Table 2.......................................................................................................................... 13
Table 3.......................................................................................................................... 14
Table 4.......................................................................................................................... 18
Table 5.......................................................................................................................... 20
Table 6.......................................................................................................................... 21
Table 7.......................................................................................................................... 30
Table 8.......................................................................................................................... 33
Table 9.......................................................................................................................... 39
Table 10......................................................................................................................... 40
Table 11......................................................................................................................... 40
Table 12......................................................................................................................... 41
Table 13......................................................................................................................... 42
Table 14......................................................................................................................... 43
Table 15......................................................................................................................... 43
Table 16......................................................................................................................... 45
Table 17......................................................................................................................... 46
Table 18......................................................................................................................... 47
Table 19......................................................................................................................... 48
Table 20......................................................................................................................... 49
Table 21......................................................................................................................... 50
Table 22......................................................................................................................... 50
Table 23......................................................................................................................... 51
List of Figures

Figure 1 ............................................................................................................. 10
Figure 2 ............................................................................................................. 11
Figure 3 ............................................................................................................. 11
Figure 4 ............................................................................................................. 11
Figure 5 ............................................................................................................. 15
Figure 6 ............................................................................................................. 16
Figure 7 ............................................................................................................. 17
Figure 8 ............................................................................................................. 19
Figure 9 ............................................................................................................. 20
Figure 10 .......................................................................................................... 22
Figure 11 .......................................................................................................... 22
Figure 12 .......................................................................................................... 23
Figure 13 .......................................................................................................... 24
Figure 14 .......................................................................................................... 25
Figure 15 .......................................................................................................... 26
Figure 16 .......................................................................................................... 27
Figure 17 .......................................................................................................... 29
Figure 18 .......................................................................................................... 29
Figure 19 .......................................................................................................... 31
Figure 20 .......................................................................................................... 32
Figure 21 .......................................................................................................... 34
Figure 22 .......................................................................................................... 36
Figure 23 .......................................................................................................... 36
Figure 24 .......................................................................................................... 37
Figure 25 .......................................................................................................... 39
Figure 26 .......................................................................................................... 41
Figure 27 .......................................................................................................... 42
Figure 28 .......................................................................................................... 44
Figure 29 .......................................................................................................... 46
Figure 30 .......................................................................................................... 48
Figure 31 .......................................................................................................... 51
Executive Summary

The purpose of the Raytheon Innovation Challenge was to develop an innovative method of monitoring human physiology with an emphasis on automotive applications. The design specifications required that the system needs to be non-invasive, innovative, practical. The problem identified is airbags can cause significant injuries if deployed under inappropriate conditions. The design of the solution is to build a monitoring system that can identify the driver size and position relative to steering wheel to determine optimal airbag deployment. This design is composed of three parts. The **Microsoft Kinect** Near-Depth Skeleton tracking software performs real-time measurements of passenger height and deployment depth. **Working Model 2D** software tests the effects of varying driver’s size, drivers’ position, and airbag stiffness. The simulation provides data on the acceleration of the drivers' head and force on the drivers' face during impact. The **Injury Severity Scale** analysis used the data from Working Model to classify the severity of injuries. The analysis determined the probability of head injury, neck injury, and torso injury, and provided the probability of death/ cost of car crash. The conclusions suggest that implementing the intelligent airbag deployment system into a real automotive airbag safety system would prove to successfully decrease the number of avoidable airbag injuries. The system has the potential to decrease a drivers probability of dying by up to 90% in some situations and save on average $78,122 over the lifetime of a personal injury claim.
Chapter 1: Introduction

Sponsor Background and Needs
Raytheon Company is a technology and innovation leader specializing in defense, security and civil markets throughout the world. Noel Ellis, Deputy Director of Electronics at Raytheon, proposed a project to develop an innovative method of monitoring human physiology with an emphasis on automotive applications. It was essential that the system be non-invasive and supply useful information that could be used for vehicle control systems. Ideally, the concept would also present future applications outside the automotive industry. The primary goal of the project is to provide a proof of concept of an innovative vehicle Human Machine Interface (HMI) and create a prototype version.

Formal Problem Definition
With growing system complexity, and increasing computing content, the Human-Machine-Interface (HMI) plays an increasing role in applications spanning from gaming to control systems to simulation environments. While the HMI has generally been understood to consider the interface of the human operator with a computer, this has been expanded to include the interface of the operator with various computer rich systems. With the intent of enhancing the user’s experience and enabling new system capabilities, there is interest in expanding the HMI to include monitoring the operator’s physiology (e.g., heart rate, skin temperature, eye dilation, EKG, etc.). For example, modern automobiles are incorporating physiology monitoring of the driver (e.g., retinal cameras) for pre-collision avoidance systems and there is interest in expanding this monitoring to other aspects of the driver’s physiology which might correlate with operational decisions or the overall user’s experience. With the growing complexity of defense systems (e.g., UAV control, flight control, network monitors, command and control centers, etc.), the ability to monitor the operator’s physiology and correlate the measured data to preferred operations or outcomes may enable new system capabilities.

This challenge seeks new ways to monitor an operator’s physiology in a non-obtrusive fashion, in real time, so that the results can be input into a control system. Of particular interest are ideas for sensor and control systems that can operate within the environment of an automobile, be incorporated into an immersive training environment or be applied to the operator of a defense system.

Objective/Specification Development
The challenge of our project expanded past developing the actual system; in order to develop a system to develop, we needed to decide on a current problem we wanted our system to eliminate. The initial instructions were to develop an innovative, noninvasive way to monitor human physiology in an automotive setting. The first challenge was to determine what we would be gaining from monitoring driver physiology. Unlike many other projects, ours did not start with a problem to solve. Therefore, our project requirements, and therefore project specifications, were far from typical. Since the only initial customer/sponsor requirements included being innovative, noninvasive, and practical, quantitative specifications were nearly nonexistent. Later on in the process once our problem was clearly defined, the team was able to determine our own personal specifications such as needing to decrease the forces on varying regions of the torso, head, and neck during a crash test simulation, as well as a number of others such as having to decrease the probability of death according to the AIS and Injury Scale.
Project Management

During the fall quarter of the project, we rotated leadership roles among members each week. This approach allowed the team to evaluate the leadership abilities of each individual. In the first week of the Winter 2013 quarter, we will appoint a team leader and all other roles.

The team leader is responsible for managing the conversation, preparing meeting agendas. The team scribe will be responsible for taking notes of each meeting and maintaining an organized binder of all documents. The team contact is responsible for talking to the sponsor and any outside parties.

A Google group and an inter-team phone number have been established to facilitate communication. The Google group allows documents to be uploaded and viewed by the entire team. The mutual phone number allows members to contact one another at any point in time in a group format.

Besides the required class lecture and labs, M5 will meet with Professor MacCarley every week. Outside research will be conducted as well to insure the members are prepared for meeting discussions. This adds up to a minimum of eight hours every week spent working on the project.
Chapter 2: Background

Existing Products

Air bag injuries have been a prevalent issue in car crashes since they were implemented in the early 1970’s. From 1994 to 2010, The National Highway Traffic Safety Administration (NHTSA) has calculated that there have been 443,810 fatal crash by restraints in the United States [1]. Also, there are numerous records of serious injuries due to airbag deployments which target the head, neck and thorax.

Many injuries from the head and neck include facial trauma, decapitation, cervical spine fractures and lacerations. Facial contact has led to orbital fractures, retinal detachment and lens fracture. Data has also shown injuries related to rib fractures, spleen lacerations, aortic rupture, and lung contusions. Most of these are serious injuries that are caused by incorrect angles of deployment and distances of airbags.

Today airbags are controlled by an airbag control unit (ACU) which determines the impact, force of the crash, and other variables to release different stage airbags. When the ACU determines if it is to deploy the airbag, it sends an electrical pulse to the inflator unit which ignites a chemical reaction to inflate the airbag. Currently, there are two and some three stage deployment systems in higher-end models that deploy different intensities to help reduce these injuries.

Airbag controls are becoming a lot more advanced in order to characterize drivers and passengers and deploy different size airbags and with less force. In 2006, GM and Ford introduced these designs based on the location and weight of the occupant. Currently GM is using a dual depth system which is able to analyze the force the detection of the crash in real time along with the occupant’s weight, seat position, and safety belt usage in order to deploy a smart airbag to lessen injuries [2]. Figure 1 is an image depicting the dual depth airbag by GM. The yellow lines represent the deep stage deployment for rearward seat position and blue lines represent shallow deployment based on forward seat deployment. Ford dual depth system is using the seating position to change the airbag deployment and drop the steering column based on the occupant’s size.
Current State of the Art

**Delphi Occupant Classification Systems (OCS)**
According to law, vehicles must be able to classify the passenger of a vehicle sitting in the passenger side seat. The Delphi occupant classification system (OCS) characterizes the weight of the vehicle operator and adjusts airbag deployment. The current system identifies three passenger states; no passenger, infant/child, and adult (George).

![Figure 2: Delphi OCS exploded diagram](image)

**Nissan Facial Monitoring System**
Nissan is designing facial monitor systems using an instrument cluster mounted camera. The system monitors the driver's state of conscience through specific facial cues. If the system detects that the driver is drowsy the seat belt tightens to grab the driver’s attention (Drunk Driving Prevention Concept Car).

![Figure 3: Nissan driver consciousness monitor](image)

**Ford Smart Car**
The future Ford smart car includes biometric sensors located on the steering wheel and seat. Measuring ambient temperature, driver temperature, heart rate, and perspiration as well as traffic around the car the Ford smart car uses a program to estimate driver workload. If the driver is stressed and at high workload then the car locks the drivers cell phone to prevent further distractions.

![Figure 4: Ford Car Interior](image)
## Kinect Background

**Table 1. Software Framework Options**

<table>
<thead>
<tr>
<th>Framework</th>
<th>Skeleton Tracking Middleware</th>
<th>Facial Tracking</th>
<th>Compatible Hardware</th>
<th>Compatible Programming Languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenNI 2.0</td>
<td>NITE 2.0 (Open Source)</td>
<td>No (Face detector only)</td>
<td>Asus Xtion Pro, PrimeSense Carmine, Kinect for Windows</td>
<td>C#, C++</td>
</tr>
<tr>
<td>Microsoft Kinect for Windows SDK</td>
<td>Kinect for Windows SDK</td>
<td>Yes</td>
<td>Kinect for Windows</td>
<td>Visual Basic, C#, C++</td>
</tr>
<tr>
<td>OpenKinect</td>
<td>Skeltrack (Open Source)</td>
<td>No (Face detector only)</td>
<td>Kinect for Windows, Kinect for Xbox 360</td>
<td>Java, C#, C++, Actionscript, Python</td>
</tr>
</tbody>
</table>
Chapter 3: Design Development

Discussion of Conceptual Designs
The goal of the project is to integrate a monitoring system into a car that can observe the operator’s physiology in a non-invasive manner in order to increase safety.

Research
The first step in the ideation process was conducting background research. To obtain a better understanding of Human Machine Interaction encompasses and the available automotive physiology monitoring products already on the market, our team started with a broad internet research.

Design Requirements and Specifications
To better understand the problem, design requirements and specifications were made and agreed upon by our group, Mr. Ellis, and Dr. MacCarley. Appendix B shows the Quality Function Deployment Chart. This chart fully defines the problem, which allows us to create the most appropriate solution. The main tasks for this QFD were to identify the customers, determine the customer requirements, weighing the customer requirements, benchmarking the competition, defining the engineering requirements, and relating customer requirements to the engineering specifications. Through this process, our group bridged the gap between project requirements, which were given to our group by Mr. Ellis, and project specifications, which were important in brainstorming ideas and narrowing down our concept list.

Ideation
To begin generating our own ideas, we started with a brainstorming session. During this phase of ideation, all ideas were accepted and feasibility was disregarded. Our main priority at this point of the brainstorming process was coming up with ideas that followed our sponsors most significant requirements; innovative and noninvasive. The broad scope of the project resulted in a wide range of ideas generated, as can be seen below in Table 2 which shows a list of ideas conjured at our initial brainstorming sessions

<table>
<thead>
<tr>
<th></th>
<th>Initial brainstorming list from early concept meeting</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Finger print scanner on seatbelt buckle</td>
</tr>
<tr>
<td>2</td>
<td>Back of headrest</td>
</tr>
<tr>
<td>3</td>
<td>Optical camera mounted in front</td>
</tr>
<tr>
<td>4</td>
<td>Outside of car</td>
</tr>
<tr>
<td>5</td>
<td>On pedals</td>
</tr>
<tr>
<td>6</td>
<td>Center console</td>
</tr>
<tr>
<td>7</td>
<td>Handle of door</td>
</tr>
<tr>
<td>8</td>
<td>Sensor on car key</td>
</tr>
<tr>
<td>9</td>
<td>Breath rate by seatbelt coil</td>
</tr>
<tr>
<td>10</td>
<td>Determine mood by music you play</td>
</tr>
<tr>
<td>11</td>
<td>Rearview mirror, especially for dilation</td>
</tr>
<tr>
<td>12</td>
<td>Stick shift</td>
</tr>
<tr>
<td>13</td>
<td>Infrared temperature</td>
</tr>
</tbody>
</table>
Concept selection & Justification

After our initial brainstorming sessions, we were left with a large number of ideas. It was necessary to make realistic reductions to narrow our list of potential concepts. Additional background research was conducted to insure our ideas were truly innovative as well as eliminate ideas that might already have been on the market. Additionally, we employed the use of a Pugh Chart to help organize the physiological responses that could be measured at a variety of locations. The Pugh Chart, seen below in Table 3, shows the possibilities of sensor location and the physiological stimuli.

Table 3: Pugh Chart to determine the relationship between sensor function and sensor location

<table>
<thead>
<tr>
<th>Function</th>
<th>Optical Sensor</th>
<th>Infrared</th>
<th>Steering Wheel</th>
<th>Key</th>
<th>Door Handle</th>
<th>Seatbelt</th>
<th>Chair</th>
<th>Stick Shifter</th>
<th>Headrest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heart Rate</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Respiratory</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pupil Dilation</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O2</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Sweat</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Temperature</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>BP</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>Inebriation</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Rental Scan</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Once the list of concepts was narrowed down to around ten ideas, we evaluated the overall potential of each concept by using a Decision Matrix. The Decision Matrix compared each of our concepts to a datum. The datum that was chosen was the Kinect Biometrics, since it was considered to be our most promised design at the time. A variety of factors were used to grade each concept. Among the many factors used were the following: cost, the value of the physiological information being obtained, level of innovation, marketability, future application capability, and the ability to increase. The Decision Matrix can be seen below in Table 3. These Decision Matrices, along with group discussion, lead us to narrow our list of concepts down to three; Steering Wheel Monitoring, Red Eye Detection, and Xbox Kinect Biometrics Scan.

It was decided that basic feasibility testing and detailed research was necessary to further evaluate each of the top three ideas and narrow down our list to one final concept. Each concept was explored further and results and decisions are listed below for each of the remaining three ideas.

Supporting Preliminary Analysis

Concept #1 Steering Wheel Monitoring

A sensor mounted on the steering wheel was chosen as the first feasible idea. Sensors on the steering wheel could potentially measure: heart rate, heartbeat, heart variability, blood pressure, oxygen saturation, and blood glucose. A number of different sensors would be needed to measure the various physiological attributes.

The research we conducted and advice provided by professors suggested that the concept was not feasible. Obtaining usable ECG data would require the use of multiple electrical leads placed on the
subject. Oxygen saturation was found to be a feasible attribute to monitor, but difficult because it requires conductance with both hands to function. Monitoring glucose was found to be possible, but would be inaccurate and expensive. Ultimately, we chose not to pursue the steering wheel idea further because it was not creative and could not provide accurate or useful data.

Figure 5: Schematic of layout of the steering wheel and its possible sensor locations

Concept #2 Red Eye Detection
A distracted driver monitoring system that is based on the phenomena of the red eye effect (commonly seen in photography) was the second best concept. The red eye effect occurs when a bright light is illuminated into the eyes and is reflected off the retina back towards the camera. The concept of the monitoring system utilizes near infrared light, which is invisible to the human eye, in order to monitor the direction of the vehicle operators’ eyes. Multiple infrared (IR) light sources would be placed across the dashboard of the car and reflect off the windshield towards the driver. The IR light would reflect off the driver’s eyes and back toward the windshield, where it would be reflected toward a number of IR detectors across the dashboard. See Figure 6 for details.
Figure 6: Schematic layout of the driver monitoring system using the red-eye effect

The drivers’ gaze would be measured by the red eye intensity returned to the detector as well as the relative shape of the pupil. A microprocessor would be used to analyze the red eye intensity and pupil shape at each camera. Using the intensity of the red eye, shape of the pupil, and positioning of the infrared cameras, it could be determined if the driver was not looking at the road. In the case where distracted driving occurs, a buzzer would be set off to warn the driver.

We chose not to pursue this idea further because the project does not suit the team skills very well. This project would require extensive electrical and software development, but does not require much mechanical or biomedical work.

Concept #3 Xbox Kinect Biometrics Scan

The third concept was to use an Xbox Kinect (Appendix) sensor to biometrically characterize the vehicle driver. The Kinect sensor provides access to valuable data including, RGB color, infrared, environment depth, skeletal tracking, and face tracking. The combination of such data streams would provide biometric characteristics about the driver. Measurements taken from the Kinect would allow for the system to adjust the seat, mirrors, and steering wheel into the optimum driving position. The same information used to classify the driver can be used as a security measure to prevent operators from using the vehicle if their detected size and shape mismatches that of the owner. The face tracking information can be utilized to track the driver’s head position to monitor where the driver is looking and possibly if the driver is falling asleep.

Additional pressure sensors within the seat can be used to accurately measure the weight and weight distribution of the occupant. This would provide further informational depth about the driver and enhance the comfort of the seat as well as the reliability of the vehicle security.

In order to accurately monitor the driver’s biomechanical makeup, the location of the cameras must be carefully considered. The necessary measurements might include:

1. The angle of the knee, between upper and lower leg
2. Head pitch, and yaw
3. The torso length of the driver
4. The weight and weight distribution of the driver using pressure sensors

Ideal points of interest could include the dots in the following image. The blue dot is where the weight sensor could be located, (in the seat). The green dot could measure height of the person in respect to
the ceiling. The distance between the seat and the ceiling is a known value, so if you subtract the distance between the ceiling and drivers head, you will have the driver’s upper body length. Lastly, the angle between the tibia and femur could be measured at the red dot.

**Proof-of-Concept Analysis Initial Testing**

Through online research and simple feasibility testing, we discovered that the Kinect offered skeletal tracking, infrared range finding, and face tracking -- all features that could be valuable for the project. One idea is that this information could be used for automatic adjustment of mirrors and seating position in the vehicle based on a full body scan. This system could also collect data about the facial features of the drivers, which could be used to detect distracted drivers.

---

**Figure 7: Schematic showing proper vs improper seating positioning**

- **Bad car seat posture**
  - Knees higher than bottom
  - Lumbar spine rounded/flexed
  - Reaching for wheel
  - Rounded shoulders
  - Chin forward

- **Good car seat posture**
  - Bottom as high as knees
  - Lumbar spine held in curve
  - Easy reach for wheel
  - Shoulders resting back on seat
  - Chin down

---
Table 4: The different capabilities of the Kinect sensor

<table>
<thead>
<tr>
<th>Capability</th>
<th>Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>Head angle (yaw) with respect to a 0 degrees reference towards the car's forward direction</td>
<td><img src="image1" alt="Diagram" /></td>
</tr>
<tr>
<td>Head angle (pitch) with 0 degrees forward reference</td>
<td><img src="image2" alt="Diagram" /></td>
</tr>
<tr>
<td>Knee joint angle</td>
<td><img src="image3" alt="Diagram" /></td>
</tr>
<tr>
<td>Hip-to-spine angle</td>
<td><img src="image4" alt="Diagram" /></td>
</tr>
<tr>
<td>Distance between top eyelid and bottom eyelid</td>
<td><img src="image5" alt="Image" /></td>
</tr>
</tbody>
</table>

A simple, life-size mock-up of a car interior was created. This mockup serves as a simple prototype for us to learn from when we design our final test stand.
Using our test bed, we simulated various Kinect placements and recorded whether or not the skeletal tracking would function. Testing showed that the Kinect sensor could not accurately recreate the skeletal structure of an individual seated in the driver seat of a car because there are too many objects blocking the path of the infrared beam.

It was discovered that a similar CPE project to monitor distracted drivers had also implemented a Kinect sensor. The goal of the project was to detect a person’s head orientation, where they were eyes were directed, and if they’re eyes were open or closed. The result of the project proved that while head position and orientation were obtainable, the Kinect was unable to monitor the driver’s eyes or eye lids.

With the potential for a body scan eliminated and face-tracking for distracted drivers having limited success the project directed its primary focus into developing an intelligent airbag system. Background research on airbag injuries showed a number of flaws within the design of airbag systems. Modern vehicles employ a small handful of sensors within the vehicle to determine vague parameters about the driver. These parameters are used to determine what type of airbag is deployed in the event of a crash. Many injuries, sometimes fatal, that occurred due to the airbag could have been prevented with the expansion of in vehicle sensors and a more intelligent airbag deployment system.

Examining injuries caused by airbag deployment the majority of serious injuries affected the drivers head and arms. Using this knowledge we began designing an advanced data acquisition system to prevent severe airbag injuries. The final system design included three key devices to acquire data about the vehicle and driver; the devices being a Kinect sensor, Bluetooth OBD device, and a capacitive touch steering wheel. The chart below breaks down the data acquired by each system.
Table 5: Data received from various acquisition devices

<table>
<thead>
<tr>
<th>Data Acquisition Device</th>
<th>Relevant Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kinect Sensor</td>
<td>-Driver head position in reference to the airbag</td>
</tr>
<tr>
<td></td>
<td>-Distance between the driver and airbag</td>
</tr>
<tr>
<td></td>
<td>-Driver head rotation in reference to the airbag</td>
</tr>
<tr>
<td>OBD Bluetooth</td>
<td>-Vehicle Speed</td>
</tr>
<tr>
<td></td>
<td>-Driver Weight</td>
</tr>
<tr>
<td></td>
<td>-Seat Belt Status</td>
</tr>
<tr>
<td>Capacitive Touch Steering Wheel</td>
<td>-Position of hands on the steering wheel</td>
</tr>
</tbody>
</table>

Creating an intelligent airbag system that helps prevent unnecessary airbag injuries satisfies the project requirements completely. The final design is an innovative solution that provides a purposeful feedback loop between the driver and vehicle. It is non-intrusive because all sensors are contact-free devices. Finally, it has applications beyond the automobile in places like factories, airplanes and heavy equipment operators.

Figure 9: Solidworks model of the final test stand
The software prototype will provide a C# graphical user interface that reports real time data acquired by the sensor systems as well as the airbag deployment for the interpreted data.

**Software Framework**

Among the available frameworks available for the project, the Microsoft Kinect for Windows SDK was chosen because of the relative robustness of the software package. This SDK provides a toolkit designed to accelerate the development of Kinect-based applications by providing a large library of source-provided applications which can be used to build new applications on top of. The framework and associated hardware, middle, and programming language choice are shown in 6.

<table>
<thead>
<tr>
<th>Framework</th>
<th>Skeleton Tracking Middleware</th>
<th>Facial Tracking Middleware</th>
<th>Chosen Hardware</th>
<th>Chosen Programming Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Kinect for Windows SDK</td>
<td>Kinect for Windows SDK</td>
<td>Kinect for Windows SDK</td>
<td>Kinect for Windows</td>
<td>C#</td>
</tr>
</tbody>
</table>

**Software System Architecture**

The C# language and .NET Framework were chosen to provide a foundation for application development. Since there are no limitations on processing power nor memory restrictions for the final project deliverable, using compiled code with large dynamically linked libraries are suitable. The graphical user interface developed through C# using Windows Presentation Foundation (WPF) provides an easy way to create a well-designed, attractive application under the Windows platform.
At the module level, the monitoring system was designed with a GUI module and a biometric measurement module as shown in Figure 10. The GUI module provides the code that creates a program for the user including text fields, infrared camera bitmaps (images), and RGB camera bitmaps. The biometrics module performs calculations based on sensor data and provides this information to be displayed to the application user in the text fields provided by the GUI.
The Software Program Flow is described in the flow chart provided in Figure 12: Software System flowchart describes GUI and UpdateBiometrics Modules.
Chapter 4 Description of the Final Design

Detailed Design Description

Testbed

A testbed was built to test the Kinect monitoring system. As seen in Figure 13, the testbed was dimensioned to match a typical car interior. This included (among other things) the position of the steering wheel, the height of the dashboard, and the size of the seat and steering wheel. The dashboard was built with sliders so that the placement of the Kinect device could be easily and accurately adjusted (see part #2 and #3 of Figure 14). The testbed was built on wheels so that it could easily be transported around campus. An exploded view of the testbed can be seen in below.

![Image of testbed]

Figure 13: An overall drawing of the testbed and some basic dimensions.
Monitoring System
Development of the monitoring system began by building the framework that the system would run within. Expanding upon Kinect code examples provided by Microsoft the program was designed around two event triggers. The first being an event to signal that the Kinect had new data to transmit and the second being a timer to refresh the Graphical User Interface (GUI). Before event triggers were created the system would initialize the GUI and Kinect sensor before monitoring would began. Once the Kinect was initialized monitoring and data processing would begin.

The monitoring system GUI provides real-time measurements of the driver, a depth field image, and a color image with visual interpretations of some system measurements. The final GUI layout can be seen in Figure 16 below. The GUI includes large images of both the depth field and composite bitmaps provided by the Kinect sensor. Real-time measurements provide the user with the distance between the drivers head and the top of the steering wheel, the distance between the drivers head and center of the steering wheel, the measured and estimated height of the driver, the orientation of the drivers head (roll, pitch, and yaw), the distance between the steering wheel and driver the airbag as to deploy, and the distance between the drivers head and the line of airbag deployment.

![Figure 15: Actual testbed that was constructed](image-url)
Once the application has finished initializing the GUI the program creates a timer and event delegate to refresh the GUI every 50 milliseconds. Large blocks of memory are then allocated for both the Kinect depth and color image. The size of each block is calculated using the image size and pixel format. The Kinect outputs each pixel color in 32-bit Red-Green-Blue-Alpha (RGBA) format, requiring 4 bytes per pixel. Each image is 640 pixels wide by 480 pixels tall, resulting in 1,228,800 bytes required to hold all the image data.

With the refresh timer created and memory allocated the program then preps and starts the Kinect sensor. The Microsoft Kinect API provides routines to connect the program to a Kinect sensor and declare specific sensor data streams. When the program finds a Kinect sensor connected a USB port the program creates a sensor object to and begins to enable Kinect data streams. The Kinect color camera is set to 640 x 480 resolution at 30 frames per second (fps). The Kinect depth stream is enabled to work in the near field with an image resolution of 640 x 480 at 30 fps. The Kinect skeletal stream is set to track skeletons in the near range and in the seated position. The skeletal stream is also given a number of smoothing parameters to adjust for error between each skeletal frame. The system then initializes the required global variables and delegates a system event to signify that the Kinect is ready to transmit all data streams. Once the sensor is started the final FaceTracker and CoordinateMapper objects required throughout the program are initialized. The refresh timer is started and the system begins the monitoring process.

When the Kinect is ready to transmit the color, depth, and skeletal data the sensor object raises the AllFramesReady event. During the AllFramesReady subroutine each Kinect data stream is opened and transferred into global memory. After all data is transferred the routine uses the FaceTracker object to find the roll, pitch, and yaw of the face as well as the wireframe triangles representing the face.

The refresh timer event handles all the image processing and system measurements. At the beginning of the routine the data collected by the AllFramesReady routine is copied again. This step allows the system to process the data without interference from the AllFramesReady routine. Once the data is
safely copied the refresh timer routine processes the depth and color data into bitmaps before continuing on to draw the skeleton, draw the face, and measure distances.

The color data returned from the Kinect is in the form of a single dimensional byte array, this data is copied directly into the memory space allocated during program startup. Microsoft’s Bitmap class provides constructors that transfer the allocated memory into a bitmap object. The color bitmap is later superimposed with measurements to create the final composite image seen in the GUI.

The processing of the depth data is much more complex than that of the color image processing. The Kinect outputs the depth data as a single dimensional short array, each value representing depth in millimeters. The Kinect returns values between 200mm and 8000mm in distance. To provide the user with a visual interpretation each depth value is mapped across the color spectrum. To eliminate complexity the depth value was mapped to color using the Hue, Saturation, and Lightness (HSL) color space instead of the typical RGB color space. The hue was determined by dividing the depth value by the maximum depth value, while the saturation and lightness were predetermined values. This method resulted in objects close to the Kinect to appear red while moving away from the Kinect resulted in a smooth transition between shades of orange, yellow, green, blue, and violet. Using a HSL to RGB conversion function we then filled a byte array with RGBA data. Once all the depth pixels were processed we used the same method as described in the color data processing. The byte array was transferred into the memory allocated at startup and used in the constructor of a bitmap object to produce the final bitmap.

![Figure 127: The RGB and HSL color spaces](image-url)
Main Program

The highest level of the software architecture is the Main Program which provides the user with a real-time interface of data from each subsystem. An example of the user interface can be seen in Figure 17 below. The interface provides all the data coming from the Kinect, OBD, and Hand Position Sensors. Along with providing the user with real-time data the Main Program computes the proper airbag deployment scheme produced by an algorithm.

![Graphical user interface of the intelligent airbag deployment system](image)

**Figure 14: Graphical user interface of the intelligent airbag deployment system**

A program flow diagram of the Main Program can be seen in Figure 17 to the right. Upon startup the program builds the GUI and starts three threads; Kinect, OBD, and Arduino. Designing the program to be multithreaded allows the program to accomplish multiple tasks in parallel. Having each subsystem run on an external thread allows each device to operate continuously. After the threads are started the program delays to allow the threads to finish their initialization processes. Once the delay is finished the main program begins collecting data from the threads. To safely grab data from another thread the individual thread is paused, data is collected, and then the thread is restarted. The process is repeated for all three threads and insures that data is not being written and read simultaneously. After data from the three threads is grabbed the data is then processed in the airbag deployment algorithm. The algorithm produces the final result to if what stage airbag would be deployed. Once this airbag deployment is chosen the main program updates all values in the GUI and returns to data collection.

Biometric Measurements

The module `UpdateBiometrics()` is the module where measurements are performed and is updated every time the frames (both depth and RGB) are refreshed. Because the software depends on the Kinect software to detect the skeleton structure, `UpdateBiometrics()` will only run if the Kinect has locked onto a skeleton for tracking. Upon loss of skeleton tracking, measurements become invalid – the loss of the skeleton tracking means the Skeleton.Joints[] array does not update thereby invalidating any
calculations done. For this reason, `UpdateBiometrics()` does not run when the Kinect API fails to lock onto a skeleton for the frame of interest.

The decision on what data about the driver to collect was based on the capabilities of the Kinect API. The Kinect API provides skeleton points in the `Skeleton.Joints` array which correspond to body parts of the driver. This data is used to determine positional measurements within the driver’s seat cabin. With the Kinect mounted in a fixed position, arbitrary points of reference were chosen to represent the center of the steering wheel, the top of the steering wheel, and the base of the driver’s seat. These points were manually determined with the Kinect System Explorer after mounting the Kinect on fixed location on the test set. A summary of the points of interest and how the Kinect was used to track the point is shown in Table 7.

Table 7: Points of Interest Obtained with the Kinect API

<table>
<thead>
<tr>
<th>Point of Interest</th>
<th>Corresponding point in Error! Reference source not found.</th>
<th>How the point was obtained</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center of Head</td>
<td>P3</td>
<td>Kinect skeleton tracking</td>
<td>The Kinect API provides a <code>JointType.Head</code> coordinate estimating the head centroid</td>
</tr>
<tr>
<td>Chest</td>
<td>P1</td>
<td>Kinect depth sensor and software collision detection</td>
<td>The RGB coordinates where the depth map matches the expected depth stream distance of the line of deployment vector.</td>
</tr>
<tr>
<td>Left Hand, Right Hand</td>
<td>P (Right Hand only)</td>
<td>Kinect skeleton tracking</td>
<td>The Kinect API provides a <code>JointType.HandLeft, JointType.HandRight</code> coordinate</td>
</tr>
<tr>
<td>Center of steering wheel</td>
<td>P0</td>
<td>Kinect depth sensor with manual estimation</td>
<td>A point chosen in the RGB bitmap is mapped to a coordinate in the infrared depth bitmap.</td>
</tr>
</tbody>
</table>
The Kinect uses two frames of reference – one for the depth camera and one for the RGB camera. In order to switch from one frame of reference to another. The API provides two functions to shift between the two reference frames called CoordinateMapper.MapSkeletonPointToDepthPoint() and CoordinateMapper.MapDepthPointToSkeletonPoint(). Since a 3-dimensional Cartesian coordinate system is used, all coordinates are mapped from depth frame to the skeleton frame.

\[
d = \sqrt{(x_3 - x_0)^2 + (y_3 - y_0)^2 + (z_3 - z_0)^2}
\]

Equation 1: Distance between two points \( P_3, P_0 \) in Cartesian Space \( \mathbb{R}^3 \)

The three dimensional distance formula between two points in Equation 1: Distance between two points \( P_3, P_0 \) in Cartesian Space \( \mathbb{R}^3 \) is used for measuring the distance between the head and the center of the steering wheel, the head and the top of the wheel, the head and the base of the seat, and the center of the chest and the wheel. Since coordinates are provided by the API in meters with an accuracy \( 1/1000^{th} \) meter, the distances are converted to English units later in the GUI display for readability. The green line in Error! Reference source not found. 9 represents the “line-of-deployment” vector defined by the two points corresponding to the center of the steering wheel and the chest. When drawn on the RGB bitmap, the vector visually describes the estimated minimum impact distance during airbag deployment.

One of the features of the biometrics module is to detect whether or not the driver’s hands are in the way of the airbag. Using skeleton tracking, the DetectHandCollision() sub-module provides the collision detection necessary to provide this information. The point-to-line distance formula in Equation 2: Distance between a point \( P \) and a line described by vector \( P_0, P_1 \) in Cartesian Space \( \mathbb{R}^3 \) was used obtain the minimum distance between the hand point and the line-of-deployment vector. Any distances under
a threshold equal to the radius of the steering wheel, in this case 5 inches, changed the color of the skeleton points drawn on the bitmap to blue, indicating that a collision would occur between the driver’s hand and the airbag. Figure 16: Visual Representation of the area of hand collision detection.0 shows the area that the software detects hand collisions represented by the shaded blue cylindrical region.

\[
\frac{||\sigma^T \cdot x||}{||\sigma^T||} = \frac{||[(x_1 - x_0) (y_1 - y_0) (z_1 - z_0)] \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}||}{||[(x_1 - x_0) (y_1 - y_0) (z_1 - z_0)]||}
\]

*Equation 2: Distance between a point P and a line described by vector P0, P1 in Cartesian Space \( \mathbb{R}^3 \)*

Cost breakdown
The budget provided to us from our sponsor was $3,000. In the initial stages of our project, money was used mainly for baseline testing. An infrared camera was purchased in order to do early stage testing to eliminate potential project ideas. Additionally, money went into building a test stand used to get a better visual of how our design would be incorporated into an automotive type setup as well to get an initial idea of the Kinect distance capabilities. In the table provided below, you can see that throughout initial testing we realized we would need to upgrade our testing platform. The Kinect works in a way that requires a minimum distance for face detection. In order to accommodate the necessary location of the Kinect in a vehicle, a sliding platform was added onto our platform requiring additional purchases. A
cost analysis of manufacturing (that is, fully implemented into a working automobile) has not been performed because it is outside the scope of this project.

Table 8: Cost Analysis

<table>
<thead>
<tr>
<th>PROJECT CATEGORY</th>
<th>Part</th>
<th>Amount</th>
<th>Vendor</th>
</tr>
</thead>
<tbody>
<tr>
<td>KINECT</td>
<td>Kinect for Windows</td>
<td>$149.99</td>
<td>microsoft.com</td>
</tr>
<tr>
<td></td>
<td>Kinect Zoom Lense</td>
<td>$29.99</td>
<td>Nyko.com</td>
</tr>
<tr>
<td>IR CAMERA</td>
<td>B/W 380 TVL Flat Pinhole Board Camera</td>
<td>$17.99</td>
<td>supercircuits.com</td>
</tr>
<tr>
<td>MOCK MATERIALS 1.0</td>
<td>BMW Steering Wheel</td>
<td>$150.00</td>
<td>Junk Yard</td>
</tr>
<tr>
<td></td>
<td>Suburban Seat</td>
<td></td>
<td>Junk Yard</td>
</tr>
<tr>
<td></td>
<td>Wheels (x4)</td>
<td>$12.00</td>
<td>Home Depot</td>
</tr>
<tr>
<td></td>
<td>2x4 wood (x6)</td>
<td>$18.00</td>
<td>Home Depot</td>
</tr>
<tr>
<td></td>
<td>Screws</td>
<td>$4.99</td>
<td>Home Depot</td>
</tr>
<tr>
<td></td>
<td>Plywood</td>
<td></td>
<td>donated</td>
</tr>
<tr>
<td>UPGRADED MOCK</td>
<td>flat plywood</td>
<td>$7.99</td>
<td>Home Depot</td>
</tr>
<tr>
<td></td>
<td>Screws</td>
<td>$4.99</td>
<td>Home Depot</td>
</tr>
<tr>
<td></td>
<td>Slider</td>
<td>$29.99</td>
<td>Home Depot</td>
</tr>
<tr>
<td></td>
<td>Wheels</td>
<td>$32.15</td>
<td>Home Depot</td>
</tr>
<tr>
<td>FUTURE TESTING</td>
<td>Working Model 2d Simulation</td>
<td>$29.95</td>
<td>Design Simulation Technologies</td>
</tr>
</tbody>
</table>

TOTAL: $487.63

Any special safety considerations
Constructing the testbed required the use of power tools and sharp saws. Extra caution was used with these tools. The power tool institute and the circular saw safety manual were used as safety references.

Any maintenance and repair considerations
The testbed did need some maintenance over the lifetime of the project. The steering wheel did come loose the more that the testbed was used. Additionally, the wheels needed to be replaced because the original set deteriorated. It is recommended that these components be checked if the testbed is used in the future.

Chapter 5 Product Realization

The final design for the project was determined to be a proof-of-concept system as a technology demonstration. The final demonstration included a hardware test-bed and software-demo. The final deliverable did not specify a manufacturable product for a number of reasons -- the main reason being that a working intelligent airbag deployment system would be vehicle dependent. In other words, the specific vehicle that the sensor system integrates with will determine the necessary sensor system specifications such as sensor placement, infrared projector intensity, infrared detector sensitivity, etc.

However, the results of our in-vehicle testing can be used to make general recommendations for the usage of the Kinect based sensor system in a vehicle. It was found through preliminary testing that a standard passenger vehicle would not support a Kinect mounted on the dashboard directly in front of the driver, nor mounted directly above the steering wheel where the windshield meets the ceiling. Further testing showed that the Kinect would not operate correctly in locations on the left side near the passenger door because of limitations of the hardware. For this reason, the recommended placement of the Kinect should be to the right of the passenger on the dashboard above the center console. Although the center rear-view mirror has been suggested as a possible location, the test-bed does not address this possibility, so further testing would need to be done to validate placement on the rear-view mirror.

![Figure 17: Kinect Sensor Tracking ranges limit the placement of the Kinect Sensor hardware within a vehicle [6]](image)

A reference to this suggested sensor placement was found during further research. The Microsoft patent discussing a sensor-based intelligent airbag concept suggests that “main sensor system ... [be] mounted overhead right above the rear view mirror.” [3] The suggested placement can be seen in Figure 18:
Microsoft patent Figure 13 shows suggested sensor placement which refers to Figure 13 in the patent. According to the preferred embodiment of the patent, multiple sensors could be in this location with multiple light sources in different locations in the cabin to illuminate the entire cabin. Although this patent describes a different technology than the Kinect – namely the patent describes a time-of-flight based infrared tracking system as compared to the “structured light” tracking system used by the Kinect – at the time of writing the second-generation Kinect has been announced which will use time-of-flight technology, the underlying technology behind radar and sonar. This change in technology will decrease the minimum distance required for placement. [18] Time-of-flight cameras can track distances up to four times shorter than the Kinect currently can (0.1m vs. 0.4m).

In light of this, it becomes evident that future implementations details, including the sensor placement depend on the current technological limitations. Again, any implementation of the technology demonstrated by the project would require a thorough design in a real vehicle. Specifications would be matched to the vehicle used, and prototyping and testing in real-world conditions would be required.
Chapter 6 Design Verification Testing

Test Descriptions with photos
As previously stated, the purpose of using Working Model simulation was to study the effectiveness of a smart airbag on a driver in a car crash.

The simulation was built to model a front end car crash. The crash simulations were run at 55 mph to simulate a high speed car crash. A large rectangular box with a weight of 4000 lbs acted as the object that the car impacts. Two dampers were used with a dampening coefficient of 1000 lb-sec/ft to mimic the car bumper. See Figure 23 for the layout of the simulation.

The simulation included a two dimensional model of a car and test dummy. A simplified version of a car was used for modeling purposes, but the interior dimensions (seat and steering wheel) were modeled accurately. The car had a weight of 4000 lbs. The airbag was simulated with four spring-dampers placed on the dummies face in parallel. The spring constant and damping coefficient of the airbag was manipulated to study how using a smaller or larger airbag in a car crash could be beneficial. A seatbelt was modeled with two springs with a high spring constant of 10000 lb/ft.

Two versions of a dummy were modeled for testing purposes. The first model was 5’ 7”, with a weight of 170 lbs, and the second model was 6’ 4” with a weight of 210 lbs. The average size model was used in the control tests and the ‘leaning forward’ tests. The tall model was used to study the effects of height in a car crash. Figure 24 shows a more detailed image of the simulation setup.

The Working Model software had the capabilities of measuring the acceleration of the drivers and the forces applied to the driver’s face. This information was used in the ISS analysis.

![Figure 19. A screenshot of a typical Working Model test.](image-url)
Seven crash scenarios were tested to study the effects of 1) driver height and 2) the distance from the drivers’ chest to steering wheel. The first three tests included the average size dummy seated fully back in the seat. The coefficients of the airbag were changed to study the effects of airbag size. The fourth and fifth test used the large dummy seated fully back in the seat. Again, the coefficients of the airbag were changed to study the effects of airbag size. The sixth and seventh tests used the average sized dummy seated leaning in towards the steering wheel. The airbag coefficients were also varied in this test.

Detailed Results
The results of the Working Model simulation testing suggest that a smart airbag deployment system would help to reduce the peak accelerations and forces experienced by the driver in a car crash.

The first three tests (which included the average size dummy seated fully back in the seat) confirm this result. The smart airbag produced lower peak head acceleration than both the normal airbag and the test without an airbag. The test without an airbag should have even higher peak acceleration because Working Model is not accurate enough to measure the acceleration when the drivers’ head impacts the steering wheel. This is one limitation of the software. The original graphs of the drivers’ head acceleration can be seen below. There was little change in the peak head force in these tests.

Figure 20: Detailed image of a car crash simulation. Note the car interior is modeled with dimensional accuracy, the dummy is built proportionally, and the airbag/seatbelt are modeled with springs and dampers.
The fourth and fifth tests (which used the large dummy seated fully back in the seat), suggest that height may have a huge affect on the effectiveness of an airbag. The peak acceleration of a normal airbag was 3600 ft/s\(^2\), which was significantly larger than the smart airbag, which had a value of 900 ft/s\(^2\). This could be attributed to the fact that the taller driver has a larger torso, and ‘wrapped around’ the airbag when it was deployed. The smart airbag could prevent this since it can take into account the drivers’ height. The peak head force was somewhat lower on the smart airbag test than the normal airbag test. Table 9 shows the results of these tests.

The sixth and seventh tests (which used the average sized dummy seated leaning in towards the steering wheel), both produced higher peak accelerations than the other tests. This suggests that leaning forward in a car crash is adverse under any airbag deployment scenario. Since there is a smaller distance for the airbag to deploy, and the head is already closer to the steering wheel, this makes intuitive sense. The smart airbag reduced the peak acceleration of the drivers head by almost 50%. Like the other tests, the peak force was somewhat smaller. Table 9 shows the results of these tests.

Figure 21: Original graphs of the drivers' head acceleration for the first 3 tests. The upper graph simulated a normal airbag deployment, the middle graph simulated no airbag deployment, and the bottom graph simulated a smart airbag deployment.
Table 9: Results of the Working Model simulation for the 7 test scenarios.

<table>
<thead>
<tr>
<th>Test No.</th>
<th>Test Dummy Description</th>
<th>Airbag</th>
<th>Peak Accel (ft/s²)</th>
<th>Peak Head Force (lbf)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5'7&quot; @ 170lbs</td>
<td>Normal</td>
<td>900</td>
<td>336</td>
</tr>
<tr>
<td>2</td>
<td>5'7&quot; @ 170lbs</td>
<td>None</td>
<td>3400</td>
<td>N/A</td>
</tr>
<tr>
<td>3</td>
<td>5'7&quot; @ 170lbs</td>
<td>Smart</td>
<td>800</td>
<td>341</td>
</tr>
<tr>
<td>4</td>
<td>6'5&quot; @ 210lbs</td>
<td>Normal</td>
<td>3600</td>
<td>580</td>
</tr>
<tr>
<td>5</td>
<td>6'5&quot; @ 210lbs</td>
<td>Smart</td>
<td>900</td>
<td>418</td>
</tr>
<tr>
<td>6</td>
<td>5'7&quot; @ 170lbs (leaning)</td>
<td>Normal</td>
<td>1900</td>
<td>571</td>
</tr>
<tr>
<td>7</td>
<td>5'7&quot; @ 170lbs (leaning)</td>
<td>Smart</td>
<td>1000</td>
<td>503</td>
</tr>
</tbody>
</table>

Specification Verification Checklist
Since this project was a proof of concept, a verification checklist was outside of the scope. The purpose of the project was to see if an intelligent airbag deployment system could provide additional driver safety. The results suggest that an intelligent airbag would in fact decrease the peak acceleration of the driver in an impact. Observing any reduction in acceleration was the goal of the testing. The magnitude or accuracy of that reduction was not the purpose of the project, so a verification checklist was not required.

Relating Working Model and Kinect: Injury Scales
A system needed to be developed in order to relate Working Model 2D and the Xbox Kinect to a normal airbag deployment as well as our smart airbag deployment. Many obstacles stood in the way from developing a system based on the parameters Working Model 2D provided. An injury criteria was used to set up a system to obtain injury probabilities, mortality rates, and medical costs. The injury criteria is defined as a function of several physical parameters which correlates well with the injury severity of the body region under consideration [Injury and Tolerance Levels PDF]. They are based on an engineering principle that states that the internal responses of a mechanical structure, no matter how big or small, or from what material it is composed, are uniquely governed by the structure’s geometric and material properties and the forces and motions applied to its surface [12]. The development of these criteria were created from multiple studies conducted on real life scenarios and experimental tests from human cadavers. The model was split up into three main criteria including the Head Injury Criteria (HIC), Neck Injury Criteria (NIC), and Thoracic Injury Criteria (TIC) which were based on the 50 percentile human. Once these Injury criteria were modeled they were converted into The Abbreviated Injury Scale (AIS). After the AIS scales were proposed then they could be combined to form the Injury Severity Scale (ISS) which gave mortality probabilities and average medical costs.

The injury scales were broken down into the three main types of injuries during frontal impact collisions [15] which include head injury, neck injury and chest injury. These were found from the values of Working Model shown in Table 10.
Table 10: Values from Working model 2D

<table>
<thead>
<tr>
<th>Test Number</th>
<th>Test Dummy</th>
<th>Airbag</th>
<th>Peak Acceleration (ft/s²)</th>
<th>Peak Head Force (N)</th>
<th>Moment Arm (m)</th>
<th>Thoracic Peak Forces (KN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5'7&quot; @ 170lbs</td>
<td>Normal</td>
<td>900</td>
<td>1494.60</td>
<td>0.203</td>
<td>2.362</td>
</tr>
<tr>
<td>2</td>
<td>5'7&quot; @ 170lbs</td>
<td>None</td>
<td>3400</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>3</td>
<td>5'7&quot; @ 170lbs</td>
<td>Smart</td>
<td>800</td>
<td>1516.84</td>
<td>0.101</td>
<td>0.329</td>
</tr>
<tr>
<td>4</td>
<td>6'5&quot; @210lbs</td>
<td>Normal</td>
<td>3600</td>
<td>2579.96</td>
<td>0.101</td>
<td>2.633</td>
</tr>
<tr>
<td>5</td>
<td>6'5&quot; @210lbs</td>
<td>Smart</td>
<td>900</td>
<td>1859.35</td>
<td>0.152</td>
<td>0.983</td>
</tr>
<tr>
<td>6</td>
<td>5'7&quot; @ 170lbs (leaning forward)</td>
<td>Normal</td>
<td>1900</td>
<td>2539.93</td>
<td>0.203</td>
<td>6.401</td>
</tr>
<tr>
<td>7</td>
<td>5'7&quot; @ 170lbs (leaning forward)</td>
<td>Smart</td>
<td>1000</td>
<td>2237.45</td>
<td>0.1015</td>
<td>1.975</td>
</tr>
</tbody>
</table>

**Head Injury Criteria**

The Head Injury Criteria is based on the average translational head acceleration and time. It was recently adopted by the National Highway Traffic Safety Administration to more accurately model injuries. The HIC calculated in our project used the following equation:

\[
HIC = \left[ \frac{1}{t_2-t_1} \int_{t_1}^{t_2} a(t) dt \right]^{2.5} (t_2 - t_1)
\]

This equation was simplified into the following equation by replacing \( t_1 \) with zero and then calculated by inserting the time duration and acceleration (g) using the values in Table 11.

\[
HIC = \left[ \left( \frac{1}{t_2} \right)^{1.5} \left( \frac{a(t) \cdot t_2}{2} \right)^{2.5} \right]
\]

Table11. Shows acceleration and time values

<table>
<thead>
<tr>
<th>Test Number</th>
<th>Test Dummy</th>
<th>Airbag</th>
<th>Peak Acceleration (ft/s²)</th>
<th>Acceleration (g)</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5'7&quot; @ 170lbs</td>
<td>Normal</td>
<td>900</td>
<td>27.97281</td>
<td>0.1</td>
</tr>
<tr>
<td>2</td>
<td>5'7&quot; @ 170lbs</td>
<td>None</td>
<td>3400</td>
<td>105.67506</td>
<td>0.15</td>
</tr>
<tr>
<td>3</td>
<td>5'7&quot; @ 170lbs</td>
<td>Smart</td>
<td>800</td>
<td>24.86472</td>
<td>0.2</td>
</tr>
<tr>
<td>4</td>
<td>6'5&quot; @210lbs</td>
<td>Normal</td>
<td>3600</td>
<td>111.89124</td>
<td>0.1</td>
</tr>
<tr>
<td>5</td>
<td>6'5&quot; @210lbs</td>
<td>Smart</td>
<td>900</td>
<td>27.97281</td>
<td>0.1</td>
</tr>
<tr>
<td>6</td>
<td>5'7&quot; @ 170lbs (leaning forward)</td>
<td>Normal</td>
<td>1900</td>
<td>59.05371</td>
<td>0.15</td>
</tr>
<tr>
<td>7</td>
<td>5'7&quot; @ 170lbs (leaning forward)</td>
<td>Smart</td>
<td>1000</td>
<td>31.0809</td>
<td>0.1</td>
</tr>
</tbody>
</table>
From these values in Table 12 the following HIC values were calculated:

<table>
<thead>
<tr>
<th>Test Number</th>
<th>HIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>73.16</td>
</tr>
<tr>
<td>2</td>
<td>304.02</td>
</tr>
<tr>
<td>3</td>
<td>109.00</td>
</tr>
<tr>
<td>4</td>
<td>2341.07</td>
</tr>
<tr>
<td>5</td>
<td>73.16</td>
</tr>
<tr>
<td>6</td>
<td>710.61</td>
</tr>
<tr>
<td>7</td>
<td>95.20</td>
</tr>
</tbody>
</table>

**Table 12: Calculated NIC values**

**Neck Injury Criteria (\(N_{ij}\))**

The current model used to calculate the NIC values incorporate axial loading along with flexion and extension moments. The tolerances used in determining the \(N_{ij}\) were 4500 N in tension/compression, and 125 Nm in extension. These tolerances are strictly for frontal impacts to restrain loading in a signal direction.

The Criteria was then normalized into the \(N_{ij}\) from the following equation:

\[
N_{ij} = \frac{F_x}{F_{int}} + \frac{M_z}{M_{int}}
\]

Where \(F_x\) represents the loading on the neck in the x direction, \(F_{int}\) is the critical intercept tolerance, \(M_z\) is the extension moment and \(M_{int}\) is the moment intercept tolerance. The \(N_{ij}\) value for the second test could not be calculated from an error reading from Working Model 2D.
From the previous values, \( N_{ij} \) values were calculated:

**Table 13: Calculated Nij values**

<table>
<thead>
<tr>
<th>Test Number</th>
<th>( N_{ij} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.622</td>
</tr>
<tr>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>3</td>
<td>2.762</td>
</tr>
<tr>
<td>4</td>
<td>1.570</td>
</tr>
<tr>
<td>5</td>
<td>2.670</td>
</tr>
<tr>
<td>6</td>
<td>4.693</td>
</tr>
<tr>
<td>7</td>
<td>2.316</td>
</tr>
</tbody>
</table>

**Thoracic Injury Criteria**

In order to find the AIS values for the Chest, the max deflection of the chest needed to be calculated. The depression of the sternum can cause the ribs to bend inward, fracture, and injure other vital organs. Also, the depression of the skeleton can push on vital organs causing contusions, ruptures and other injuries. In order to find the maximum deflection, the chest was modeled as a spring with a spring constant of 2.81 kN/cm [17]. Figure 2 shows a lumped mass model of the thorax in blunt frontal impacts.

The deflections of the chest were simplified using the basic spring equation:

\[
\Delta x_{\text{max}} = \frac{F_{\text{max}}}{K_{12}}
\]
Each of the max deflections are shown in Table 14.

<table>
<thead>
<tr>
<th>Test Number</th>
<th>Thoracic Peak Forces (KN)</th>
<th>Δx (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.362</td>
<td>8.405</td>
</tr>
<tr>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>3</td>
<td>0.329</td>
<td>1.171</td>
</tr>
<tr>
<td>4</td>
<td>2.633</td>
<td>9.371</td>
</tr>
<tr>
<td>5</td>
<td>0.983</td>
<td>3.498</td>
</tr>
<tr>
<td>6</td>
<td>6.401</td>
<td>22.779</td>
</tr>
<tr>
<td>7</td>
<td>1.975</td>
<td>7.029</td>
</tr>
</tbody>
</table>

**Abbreviated Injury Scale (AIS)**

After the injury criteria’s were calculated, they needed to be converted into an AIS value to be further transferred into an ISS value. The Abbreviated Injury Scale is an anatomical scoring system that provides a reasonably accurate ranking of the severity of injury [14]. This scoring system classifies each injury in multiple body regions based on a scale of 1 to 6. Table 15 shows the correspondence of AIS number to injury. For this analysis, only the head, neck and chest for AIS values were found. Lognormal equations were used for each criteria in order to find the probability of an AIS value. The probability was determined to cutoff at 50%, meaning an AIS value has to have to have a probability of 50% or greater to occur.

<table>
<thead>
<tr>
<th>Injury</th>
<th>AIS Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Minor</td>
</tr>
<tr>
<td>2</td>
<td>Moderate</td>
</tr>
<tr>
<td>3</td>
<td>Serious</td>
</tr>
<tr>
<td>4</td>
<td>Severe</td>
</tr>
<tr>
<td>5</td>
<td>Critical</td>
</tr>
<tr>
<td>6</td>
<td>Unsurvivable</td>
</tr>
</tbody>
</table>

**Head AIS Value**

Currently The NHTSA uses brain injury risk functions to find various levels of head AIS levels shown in Figure 28.
Lognormal equations were found to find the probabilities of AIS head values. These lognormal equations are shown below [13]:

\[
P(AIS \geq 1) = \frac{1}{1 + e^{\left(\frac{1.54 + 200}{HIC} - 0.0065 \cdot HIC\right)}}
\]

\[
P(AIS \geq 2) = \frac{1}{1 + e^{\left(\frac{2.49 + 200}{HIC} - 0.00483 \cdot HIC\right)}}
\]

\[
P(AIS \geq 3) = \frac{1}{1 + e^{\left(\frac{3.39 + 200}{HIC} - 0.00372 \cdot HIC\right)}}
\]

\[
P(AIS \geq 4) = \frac{1}{1 + e^{\left(\frac{4.9 + 200}{HIC} - 0.00351 \cdot HIC\right)}}
\]

\[
P(AIS \geq 5) = \frac{1}{1 + e^{\left(\frac{7.82 + 200}{HIC} - 0.00429 \cdot HIC\right)}}
\]

\[
P(AIS \geq 6) = \frac{1}{1 + e^{\left(\frac{12.24 + 200}{HIC} - 0.00565 \cdot HIC\right)}}
\]

The HIC values for each test were inserted into the previous equations to find probabilities of each AIS level. These can be seen on Table 16.
Table 26. Shows all of the AIS probabilities from each test

<table>
<thead>
<tr>
<th>Head AIS Values</th>
<th>Test 1</th>
<th>0.0219</th>
<th>Test 4</th>
<th>1.0000</th>
<th>Test 7</th>
<th>0.0464</th>
</tr>
</thead>
<tbody>
<tr>
<td>P(AIS ≥1)</td>
<td>0.0219</td>
<td></td>
<td>P(AIS ≥1)</td>
<td>1.0000</td>
<td>P(AIS ≥1)</td>
<td>0.0464</td>
</tr>
<tr>
<td>P(AIS ≥2)</td>
<td>0.0076</td>
<td></td>
<td>P(AIS ≥2)</td>
<td>0.9998</td>
<td>P(AIS ≥2)</td>
<td>0.0158</td>
</tr>
<tr>
<td>P(AIS ≥3)</td>
<td>0.0029</td>
<td></td>
<td>P(AIS ≥3)</td>
<td>0.9947</td>
<td>P(AIS ≥3)</td>
<td>0.0058</td>
</tr>
<tr>
<td>P(AIS ≥4)</td>
<td>0.0006</td>
<td></td>
<td>P(AIS ≥4)</td>
<td>0.9620</td>
<td>P(AIS ≥4)</td>
<td>0.0013</td>
</tr>
<tr>
<td>P(AIS ≥5)</td>
<td>0.0000</td>
<td></td>
<td>P(AIS ≥5)</td>
<td>0.8945</td>
<td>P(AIS ≥5)</td>
<td>7.393E-05</td>
</tr>
<tr>
<td>P(AIS ≥6)</td>
<td>0.0000</td>
<td></td>
<td>P(AIS ≥6)</td>
<td>0.7113</td>
<td>P(AIS ≥6)</td>
<td>1.013E-06</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test 2</th>
<th>P(AIS ≥1)</th>
<th>1.0000</th>
<th>Test 5</th>
<th>P(AIS ≥1)</th>
<th>0.0219</th>
</tr>
</thead>
<tbody>
<tr>
<td>P(AIS ≥2)</td>
<td>1.0000</td>
<td></td>
<td>P(AIS ≥2)</td>
<td>0.0076</td>
<td></td>
</tr>
<tr>
<td>P(AIS ≥3)</td>
<td>0.9996</td>
<td></td>
<td>P(AIS ≥3)</td>
<td>0.0029</td>
<td></td>
</tr>
<tr>
<td>P(AIS ≥4)</td>
<td>0.9967</td>
<td></td>
<td>P(AIS ≥4)</td>
<td>0.0006</td>
<td></td>
</tr>
<tr>
<td>P(AIS ≥5)</td>
<td>0.9944</td>
<td></td>
<td>P(AIS ≥5)</td>
<td>3.571E-05</td>
<td></td>
</tr>
<tr>
<td>P(AIS ≥6)</td>
<td>0.9926</td>
<td></td>
<td>P(AIS ≥6)</td>
<td>4.748E-07</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test 3</th>
<th>P(AIS ≥1)</th>
<th>0.0650</th>
<th>Test 6</th>
<th>P(AIS ≥1)</th>
<th>0.9425</th>
</tr>
</thead>
<tbody>
<tr>
<td>P(AIS ≥2)</td>
<td>0.0219</td>
<td></td>
<td>P(AIS ≥2)</td>
<td>0.6594</td>
<td></td>
</tr>
<tr>
<td>P(AIS ≥3)</td>
<td>0.0080</td>
<td></td>
<td>P(AIS ≥3)</td>
<td>0.2635</td>
<td></td>
</tr>
<tr>
<td>P(AIS ≥4)</td>
<td>0.0017</td>
<td></td>
<td>P(AIS ≥4)</td>
<td>0.0637</td>
<td></td>
</tr>
<tr>
<td>P(AIS ≥5)</td>
<td>0.0001</td>
<td></td>
<td>P(AIS ≥5)</td>
<td>0.0064</td>
<td></td>
</tr>
<tr>
<td>P(AIS ≥6)</td>
<td>1.428E-06</td>
<td></td>
<td>P(AIS ≥6)</td>
<td>0.0002</td>
<td></td>
</tr>
</tbody>
</table>
The determined AIS values are shown in Table 17. It was noted that the Head AIS for test number 2 and 4 were 6, meaning that the dummy would have not survived.

**Table 17. Shows the chosen AIS values**

<table>
<thead>
<tr>
<th>Test Number</th>
<th>Test Dummy</th>
<th>Head (AIS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5'7&quot; @ 170lbs</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>5'7&quot; @ 170lbs</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>5'7&quot; @ 170lbs</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>6'5&quot; @ 210lbs</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>6'5&quot; @ 210lbs</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>5'7&quot; @ 170lbs (leaning forward)</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>5'7&quot; @ 170lbs (leaning forward)</td>
<td>1</td>
</tr>
</tbody>
</table>

**Neck AIS Value**

Neck AIS values use the Nij injury criteria to find the probability of AIS Values. The risk curve created for the Neck Injury AIS was found by predicting the $N_{ij}$ calculations from experimental dummy test data and real world injury rates estimated from the National Automotive Sampling System database and then expanded from Mertz and Prasad [Development of Improved Injury Criteria for the Assessment of Advanced Automotive Restraint Systems]. The risk probability for $N_{ij}$ values is shown in Figure 29.
The Lognormal equations found from NHTSA’s database are similar to the Head AIS values. The equations are shown below:

\[
P(AIS \geq 2) = \frac{1}{1 + e^{(2.054 + 1.195N_{ij})}}
\]

\[
P(AIS \geq 3) = \frac{1}{1 + e^{(3.227 + 1.969N_{ij})}}
\]

\[
P(AIS \geq 4) = \frac{1}{1 + e^{(2.693 + 1.195N_{ij})}}
\]

\[
P(AIS \geq 5) = \frac{1}{1 + e^{(3.817 + 1.195N_{ij})}}
\]

The results from these equations are shown in Table 18.

<table>
<thead>
<tr>
<th>Test</th>
<th>P(AIS ≥2)</th>
<th>Test</th>
<th>P(AIS ≥2)</th>
<th>Test</th>
<th>P(AIS ≥2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.9672</td>
<td>4</td>
<td>0.9610</td>
<td>6</td>
<td>0.9992</td>
</tr>
<tr>
<td>2</td>
<td>0.9012</td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.6473</td>
<td>5</td>
<td>0.8840</td>
<td></td>
<td>0.9976</td>
</tr>
<tr>
<td>4</td>
<td>0.3736</td>
<td>6</td>
<td>0.6220</td>
<td>6</td>
<td>0.9486</td>
</tr>
<tr>
<td>5</td>
<td>0.7383</td>
<td>7</td>
<td>0.3484</td>
<td>7</td>
<td>0.8571</td>
</tr>
</tbody>
</table>

Test number 2 could not be completed because the value output from Working Model 2D was unobtainable. From these tests, it can be seen that the majority of these injuries have an AIS value of 4 with 50 or greater percent probability. The final determined AIS values are found on Table 19.

<table>
<thead>
<tr>
<th>Test Number</th>
<th>Test Dummy</th>
<th>Neck (AIS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5’7” @ 170lbs</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>5’7” @ 170lbs</td>
<td>N/A</td>
</tr>
<tr>
<td>3</td>
<td>5’7” @ 170lbs</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>6’5” @210lbs</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>6’5” @210lbs</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>5’7” @ 170lbs (leaning forward)</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>5’7” @ 170lbs (leaning forward)</td>
<td>4</td>
</tr>
</tbody>
</table>
Combined Thoracic Index

The Combined Thoracic Index is used in order to take the max deflection that the driver would encounter with the deployment of an airbag. A probability curve was created through various testing of surrogate humans and animals.

From this, like the Neck and Head AIS, Lognormal equations were created to show probability of each AIS value. These are shown below:

\[
P(AIS \geq 2) = \frac{1}{1 + e^{(1.8706 - 0.94439 \cdot D_{\text{max}})}}
\]

\[
P(AIS \geq 3) = \frac{1}{1 + e^{(3.7124 - 0.0475 \cdot D_{\text{max}})}}
\]

\[
P(AIS \geq 4) = \frac{1}{1 + e^{(5.0952 - 0.0475 \cdot D_{\text{max}})}}
\]

\[
P(AIS \geq 5) = \frac{1}{1 + e^{(8.8274 - 0.0459 \cdot D_{\text{max}})}}
\]
The results are shown in Table 20.

**Table 20.** Shows all the AIS possibilities of each test

<table>
<thead>
<tr>
<th>Test 1</th>
<th>Neck AIS Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>P(AIS ≥2)</td>
<td>0.1828</td>
</tr>
<tr>
<td>P(AIS ≥3)</td>
<td>0.0351</td>
</tr>
<tr>
<td>P(AIS ≥4)</td>
<td>0.0091</td>
</tr>
<tr>
<td>P(AIS ≥5)</td>
<td>0.0002</td>
</tr>
</tbody>
</table>

As with the previous AIS chart, test number 2 was unable to be ran.

**Table 21.** Shows the determined AIS values for the chest

<table>
<thead>
<tr>
<th>Test Number</th>
<th>Test Dummy</th>
<th>Chest (AIS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5'7&quot; @ 170lbs</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>5'7&quot; @ 170lbs</td>
<td>N/A</td>
</tr>
<tr>
<td>3</td>
<td>5'7&quot; @ 170lbs</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>6'5&quot; @210lbs</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>6'5&quot; @210lbs</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>5'7&quot; @ 170lbs (leaning forward)</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>5'7&quot; @ 170lbs (leaning forward)</td>
<td>1</td>
</tr>
</tbody>
</table>

**Injury Severity Scale (ISS)**

The Injury Severity Scores were the last calculations found for the analysis. The ISS is also an anatomical scoring system which uses the summation of squares of AIS values. The score is taken from three of the most severe injuries on the body. The use of this scoring system facilitates comparison of the mortality experience of varied groups of trauma patients [2]. The ISS ranges in scores between 0 and 75. If one of the AIS values of the three regions obtains a 6 then the ISS value instantly is 75. Since our testing only used three body parts, each of these were used in analysis. The following equation was used to find the ISS value:

\[ ISS = AIS^2_{\text{head}} + AIS^2_{\text{neck}} + AIS^2_{\text{chest}} \]

The summation of these values are shown in Table 21.
Table 21: Summation of AIS values

<table>
<thead>
<tr>
<th>Test Number</th>
<th>Test Dummy</th>
<th>Airbag</th>
<th>Head (AIS)</th>
<th>Neck (AIS)</th>
<th>Thoracic (AIS)</th>
<th>ISS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5'7&quot; @ 170lbs</td>
<td>Normal</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>18</td>
</tr>
<tr>
<td>2</td>
<td>5'7&quot; @ 170lbs</td>
<td>None</td>
<td>6</td>
<td>N/A</td>
<td>N/A</td>
<td>75</td>
</tr>
<tr>
<td>3</td>
<td>5'7&quot; @ 170lbs</td>
<td>Smart</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>6'5&quot; @210lbs</td>
<td>Normal</td>
<td>6</td>
<td>4</td>
<td>1</td>
<td>75</td>
</tr>
<tr>
<td>5</td>
<td>6'5&quot; @210lbs</td>
<td>Smart</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td>5'7&quot; @ 170lbs (leaning forward)</td>
<td>Normal</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>30</td>
</tr>
<tr>
<td>7</td>
<td>5'7&quot; @ 170lbs (leaning forward)</td>
<td>Smart</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>18</td>
</tr>
</tbody>
</table>

From these injury severity scores, we can now relate them to previous models that have been created through various experiments and previous analysis. The following probabilities in Table 22 have been found from the previous models in “Injury Severity Score Revisited.”

Table 22. Shows the probability of mortality

<table>
<thead>
<tr>
<th>Test Number</th>
<th>Mortality Probability ( &lt;50 Yrs)</th>
<th>Mortality Probability ( &gt;50 Yrs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10%</td>
<td>22%</td>
</tr>
<tr>
<td>2</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>3</td>
<td>2%</td>
<td>7%</td>
</tr>
<tr>
<td>4</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>5</td>
<td>10%</td>
<td>22%</td>
</tr>
<tr>
<td>6</td>
<td>22%</td>
<td>48%</td>
</tr>
<tr>
<td>7</td>
<td>10%</td>
<td>22%</td>
</tr>
</tbody>
</table>
Statistical Analysis of ISS

Figure 31 shows the ISS values with normal airbags and smart airbags. Visually, it seems that the smart airbag has a lower overall ISS value than the normal airbag.

![Airbag ISS Value](image)

Figure 31. Shows the ISS values of ISS values

Further analysis was completed to determine if a smart airbag system would be beneficial in a motor vehicle. A one way ANOVA test was run to determine if there is a significantly lower effect of injuries with a smart airbag. This is shown in Table 23. The p value obtained from the test was .2214 which is greater than the 95% confidence level of .05. This test concludes that there is a significant affect in both types of injury sets.

Table 23. Shows the ANOVA test results

<table>
<thead>
<tr>
<th>Source of Variation</th>
<th>SS</th>
<th>df</th>
<th>MS</th>
<th>F</th>
<th>P-value</th>
<th>F crit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Between Groups</td>
<td>962.6667</td>
<td>1</td>
<td>962.6667</td>
<td>2.094271</td>
<td>0.221405</td>
<td>7.708647</td>
</tr>
<tr>
<td>Within Groups</td>
<td>1838.667</td>
<td>4</td>
<td>459.6667</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>2801.333</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Limitations**

There are many limitations during the analysis of Working Model 2D. The injury criteria values and AIS values are based on engineering principle that states that the internal responses of a mechanical structure, no matter how big or small, or from what material it is composed, are uniquely governed by the structure’s geometric and material properties and the forces and motions applied to its surface [7]. Because of the variability in injury tolerance levels between humans, this model will never be 100% accurate. This system of models is based on probabilities of injuries which have been found through various testing and experiments with multiple subject. To account for this uncertainty a high 50% probability cutoff was used to choose the AIS values.

Another limitation includes the statistical analysis with the ANOVA test. Since there were only three tests that were ran for each scenario, the results could have been skewed. In order to confidently say that the ANOVA proves that there is a lower difference in ISS values, multiple more Working Model 2D tests need to be ran. With a higher number sampling rate, the results will more accurately show differences if any in the two types of airbags.
Chapter 7 Conclusions and Recommendations

As initially stated in the introduction, the goal of the challenge was to develop a noninvasive monitoring system that would monitor operator physiology in real time that could then be input into a control system. Particularly, the goal was to develop a system that could be incorporated into an automotive setting. Upon defining our focus into the realm of airbags, we later included our own requirements of decreasing the number of avoidable airbag injuries by means of dynamically monitoring the driver of an automobile. Upon completion, our team managed to develop a prototype that was validated through simulation testing. Implementing our monitoring system into a real automotive airbag safety system would prove to successfully decrease the number of avoidable airbag injuries. Decreasing the number of avoidable airbag injuries will also, it was discovered, prove to decrease medical costs and reduce annual automotive insurance costs. Through in depth analysis of previous studies, it was proven that implementing our monitoring system has the potential to decrease a drivers probability of dying by up to 90% in some situations [17]. Additionally, this would correlate to saving an average of roughly $72,000 over the lifetime of a personal injuries claim. Through testing validation and correlation to literature review, we successfully developed a proof of concept design that could further be integrated into an automobile.
Acknowledgements

This project would have not have been successful without the advisement, dedication, and extensive knowledge shared with the team by our advisor, Art MacCarley. Additional acknowledgement goes out to our sponsor representative, Noel Ellis, Deputy Director of Electronics at Raytheon, who presented us with this challenge and monetary support which made this project possible. Lastly, we would like to thank the professors of this year’s Interdisciplinary Senior Design Class, who throughout the year have taught us how to successfully plan for and succeed in completing a design project.
References


# Appendices

## Appendix A QFD, Decision Matrices etc.

Below, please find a Decision Matrix used in the ideation process to narrow our initial ideas down to three.

<table>
<thead>
<tr>
<th>Concept → Criteria↓</th>
<th>Steering Wheel Sensor</th>
<th>Red Eye Tracking</th>
<th>Kinect Biometrics</th>
<th>Alcohol Monitor</th>
<th>Glucose</th>
<th>Infrared Temperature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Innovative</td>
<td>-</td>
<td>+</td>
<td>DATUM</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Non-intrusive</td>
<td>S</td>
<td>S</td>
<td>DATUM</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>Broad Applications</td>
<td>-</td>
<td>+</td>
<td>DATUM</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Increase Safety</td>
<td>-</td>
<td>-</td>
<td>DATUM</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Cost</td>
<td>+</td>
<td>+</td>
<td>DATUM</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Valuable</td>
<td>-</td>
<td>-</td>
<td>DATUM</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Physiological</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Information</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accurate</td>
<td>-</td>
<td>S</td>
<td>DATUM</td>
<td>+</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Lifetime</td>
<td>-</td>
<td>S</td>
<td>DATUM</td>
<td>+</td>
<td>-</td>
<td>S</td>
</tr>
<tr>
<td>Marketable</td>
<td>-</td>
<td>-</td>
<td>DATUM</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Σ-</td>
<td>7</td>
<td>3</td>
<td>DATUM</td>
<td>4</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td>Σ+</td>
<td>1</td>
<td>2</td>
<td>DATUM</td>
<td>4</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>ΣS</td>
<td>1</td>
<td>3</td>
<td>DATUM</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
Appendix B: Bill of Materials, Detailed Part Drawing, Engineering Specifications

### Bill of Materials

Project: Raytheon Innovation Challenge  
Team: M5  
Date: 2/7/13  
Revision: 1

<table>
<thead>
<tr>
<th>Part #</th>
<th>QTY</th>
<th>Name</th>
<th>Material</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>1</td>
<td>Kinect</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>002</td>
<td>2</td>
<td>Platform</td>
<td>Plywood</td>
<td></td>
</tr>
<tr>
<td>003</td>
<td>2</td>
<td>Slider</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>004</td>
<td>2</td>
<td>Flatteners</td>
<td>Wood (2X4)</td>
<td></td>
</tr>
<tr>
<td>005</td>
<td>2</td>
<td>Steering Wheel Mount Longs</td>
<td>Wood (2X4)</td>
<td></td>
</tr>
<tr>
<td>006</td>
<td>2</td>
<td>Steering Wheel Mount Shorts</td>
<td>Wood (2X4)</td>
<td></td>
</tr>
<tr>
<td>007</td>
<td>1</td>
<td>Steering Wheel Structure</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>008</td>
<td>1</td>
<td>Steering Wheel</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>009</td>
<td>2</td>
<td>Vertical Longs</td>
<td>Wood (2X4)</td>
<td></td>
</tr>
<tr>
<td>010</td>
<td>2</td>
<td>Vertical Shorts</td>
<td>Wood (2X4)</td>
<td></td>
</tr>
<tr>
<td>011</td>
<td>1</td>
<td>Stabilizer</td>
<td>Wood (2X4)</td>
<td></td>
</tr>
<tr>
<td>012</td>
<td>2</td>
<td>Seat Base</td>
<td>Wood (2X4)</td>
<td></td>
</tr>
<tr>
<td>013</td>
<td>1</td>
<td>Plywood Base</td>
<td>Plywood</td>
<td></td>
</tr>
<tr>
<td>014</td>
<td>2</td>
<td>Frame Longs</td>
<td>Wood (2X4)</td>
<td></td>
</tr>
<tr>
<td>015</td>
<td>4</td>
<td>Frame Shorts</td>
<td>Wood (2X4)</td>
<td></td>
</tr>
<tr>
<td>016</td>
<td>2</td>
<td>Front Wheels</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>017</td>
<td>2</td>
<td>Rear Wheels</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td></td>
<td>Wood Screws</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>OBD Reader</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Drivers Seat</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Arduino Microcontroller</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>50FT</td>
<td></td>
<td>Wire</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td></td>
<td>470 Ohm Resistors</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>Azoteq IQS158 Capacitive Proximity Sensors</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>Multiplexors</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>100 microF Capacitors</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>1 pF capacitors</td>
<td>Unspecified</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Steering Wheel Cover</td>
<td>Unspecified</td>
<td></td>
</tr>
</tbody>
</table>
## Engineering Specifications

**Project:** M5  
**Customer:** Noel Ellis (Raytheon)  
**Drivers & Car Companies**

<table>
<thead>
<tr>
<th>Requirements (Whats)</th>
<th>Weighting (Total 100)</th>
<th>No Change</th>
<th>Technology never used</th>
<th>Significant Advancement</th>
<th>Shown in history</th>
<th>$2.5M</th>
<th>Incidence of $1-$99</th>
<th>Incidence of $100-$299</th>
<th>Incidence of $300+</th>
<th>Improves in 1% or more</th>
<th>Improves by 1%</th>
<th>Miscellaneous</th>
<th>Applications to 50% of people</th>
<th>Applications to &gt;50% of people</th>
<th>Applications to &gt;90% of people</th>
<th>Cost of $100 or less</th>
<th>Cost of $100-$1,000</th>
<th>Cost of $1,000-$10,000</th>
<th>Cost of $10,000-$100,000</th>
<th>Cost of &gt;$100,000</th>
<th>Cost of $10,000 per month</th>
<th>Cost of $50,000 per month</th>
<th>Cost of $100,000 per month</th>
<th>Benchmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-Invasive</td>
<td>9.5</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Innovative</td>
<td>9.6</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Broad Applicability</td>
<td>9.6</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Increase Safety</td>
<td>7.6</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Increase Performance</td>
<td>7.1</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Cost</td>
<td>7.1</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Marketable</td>
<td>0.5</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>9</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Reliable</td>
<td>8.1</td>
<td>1</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Broad Audience</td>
<td>3.8</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>9</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Valuable Phys. Data</td>
<td>6.2</td>
<td>1</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Real-Time Changes</td>
<td>5.2</td>
<td>1</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Simple to Learn</td>
<td>4.8</td>
<td>1</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Long Lifetime</td>
<td>4.3</td>
<td>1</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Unobtrusive</td>
<td>6.7</td>
<td>1</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Small Power Req.</td>
<td>1.0</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Integratable</td>
<td>5.7</td>
<td>1</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Temperature Change</td>
<td>1.4</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Light Changes</td>
<td>1.9</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Manufacturable</td>
<td>2.4</td>
<td>3</td>
<td>1</td>
<td>9</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Low Maintenance</td>
<td>2.9</td>
<td>3</td>
<td>1</td>
<td>9</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9 Strong Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 Medium Correlation</td>
</tr>
<tr>
<td>1 Small Correlation</td>
</tr>
<tr>
<td>Blank No Correlation</td>
</tr>
</tbody>
</table>

**Units**  
**Targets**  
Benchmark #1  
Benchmark #2
Appendix C Detailed Supporting Analysis

The Kinect Sensor is a video game console accessory developed by Microsoft, which uses proprietary IR range tracking from a company called PrimeSense. The Kinect is uses a combination of an infrared depth-finding and structured light technology for edge detection (shape detection). Since the original Kinect’s release, Microsoft has released a version of the Kinect which interfaces directly with a PC along with a software-development kit to allow for custom Kinect applications to be developed on Windows.

PrimeSense has made an open-source library available called OpenNI which “has made available an open source framework – the OpenNI framework – which provides an application programming interface (API) for writing applications utilizing natural interaction.[4]” Using these open-source tools, it would be possible to use the Kinect Sensor to determine the body orientation of an individual sitting on a seat. Whether the OpenNI or standard Microsoft API is used, data such as joint orientation, body orientation and skeletal tracking can be obtained through the Kinect system.

The Kinect Sensor provides data about object depth. The application-programming interface (API) provided in Kinect for Windows provides tools for determining the depth of an object based on their X-Y coordinates in the image. Extracting this data for biometric use would not be a difficult task using the Kinect hardware and software development kit (SDK). In addition, recent advances in the Kinect allow for the tracking of the user’s head, neck, arm positions, and facial expressions. With access to this data, an intelligent automobile system could potentially determine if the operator is distracted while driving.
Appendix D Gantt Chart

The Gantt chart detailing the project schedule has a relevant critical path beginning midway through the Winter 2013 quarter and extending through Spring 2013’s project completion. The chart shows the critical with critical tasks and critical links through the College of Engineering’s Senior Project Showcase. The Gantt chart tasks focuses on developing technical and performance specifications and extend through fabrication, testing, and production. These critical tasks will be dependent on the completion of previous tasks. Because of this dependency, the project has a critical path through the later testing and production stages. For example, component testing on the Kinect Sensor portion will depend on the completion of the development of the Kinect software components.
Appendix E Code Listing

/* M5 - Intelligent Airbag Software System
 * Alan Kenyon, Dominic Nicoletti, Hiram Riley Lew, Lindsay Mosbrucker, Steven Arms
 * EE 470 Senior Project
 *
 * Author: Alan Kenyon
 * Contributors: Hiram Riley Lew
 *
 * June 7, 2013
 *
 * This C# program uses the Microsoft Kinect API, skeleton tracking, and face tracking
 * to determine driver biometrics.
 *
 * Notes:
 *
 * - Both RGB and infrared stream are converted to bitmap.
 * - Kinect Range Mode set to Near Mode
 * - Kinect skeleton tracking tracks head, center of shoulder, left shoulder,
 *    right shoulder, right elbow, left elbow, right hand and left hand.
 * - Calculations are made between skeleton tracking points and arbitrary points
 *    Arbitrary points include: Center of Steering wheel depth frame
 *    Top of Steering wheel in depth frame
 *    Bottom of Passenger Seat
 *    These arbitrary points are determined visually from a fixed Kinect position
 *    on the test bed.
 * - Facial tracking shows pitch, yaw, and roll during if face tracking succeeds
 * - Program set to refresh window and calculations every 50ms.
 * - Microsoft XNA Library used for Vector calculations.
 */

using System;
using System.IO;
using System.Diagnostics;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading;
using Microsoft.Kinect;
using Microsoft.Kinect.Toolkit;
using Microsoft.Kinect.Toolkit.FaceTracking;
using Microsoft.Xna.Framework;
using System.Windows;
using System.Windows.Forms;
using System.Windows.Media;
using System.Drawing;
using System.Drawing.Imaging;
using System.Runtime.InteropServices;

namespace WindowsFormsApplication1
{
    public partial class MSIAS : Form
    {
        public MSIAS()
        {
            //Load Splash Screen while Kinect and Main screen load
            Form2.ShowSplashScreen();

            InitializeComponent();

            //Timer to Process new data
            RefreshTimer.Interval = 50;
            RefreshTimer.Tick += RefreshWindow;

            // Memory Allocation for Bitmaps
            Marshal.FreeHGlobal(colorPtr);
            colorPtr = Marshal.AllocHGlobal(1228800);
            Marshal.FreeHGlobal(depthPtr);
            depthPtr = Marshal.AllocHGlobal(1228800);
            Marshal.FreeHGlobal(compositePtr);
            compositePtr = Marshal.AllocHGlobal(1228800);

            // Start Kinect and timer. Close splash screen
            Kinect();
            RefreshTimer.Start();
            Form2.CloseForm();
        }

        // SubModule Handles GUI Refresh and Kinect Data Refresh
        public void RefreshWindow(object sender, EventArgs e)
        {

        }
    }
}
//Begin Data Processing
//If Kinect data exists copy into local arrays for processing
Stopwatch stopwatch = new Stopwatch();
stopwatch.Start();
if (colorDataTemp != null && depthDataTemp != null) {
    KinectLock.WaitOne();
    colorDataTemp.CopyTo(colorData, 0);
    depthDataTemp.CopyTo(depthData, 0);
    skeletonData = Skeleton_Clone(skeletonDataTemp);
    KinectLock.ReleaseMutex();
}

//Image Processing
if (colorData != null && depthData != null) {
    //Convert Kinect image arrays into Bitmaps
    kinectDepthBitmap = DepthDataFormatter(depthData);
    kinectCompositeBitmap = PixelDataToBitmap(colorData, compositePtr);
    GetSkeleton();
    //Check Skeleton and Face Tracking status
    if (skeleton != null && driver == true) {
        SkelTrackVal.Text = "Tracking";
        SkeletonToComposite();
    } else {
        SkelTrackVal.Text = "Untracked";
        this.LOFHeadProxVal.Text = "0.00";
        this.LOFDistanceVal.Text = "0.00";
        this.HeadOrienRollVal.Text = "0.00";
        this.HeadOrienYawVal.Text = "0.00";
        this.HeadOrienPitchVal.Text = "0.00";
        this.textBox1.Text = "0.00";
        this.textBox2.Text = "0.00";
        //this.textBox3.Text = "0.00";
        this.textBox4.Text = "0.00";
        this.textBox5.Text = "0.00";
        this.textBox6.Text = "0.00";
        this.textBox7.Text = "0.00";
        //this.textBox8.Text = "0.00";
        //this.textBox9.Text = "0.00";
        this.textBox10.Text = "0.00";
    }
    // Check for Face Tracking
    if (facePoints != null && faceTriangles != null) {
        FaceMaskToComposite();
        FaceTrackingStatus.Text = "Tracking";
        HeadOrienRollVal.Text = headRoll.ToString("F");
        HeadOrienYawVal.Text = headYaw.ToString("F");
        HeadOrienPitchVal.Text = headPitch.ToString("F");
    } else {
        FaceTrackingStatus.Text = "Untracked";
        try {
            headX = skeleton.Joints[JointType.Head].Position.X * 1000;
            headY = skeleton.Joints[JointType.Head].Position.Y * 1000;
            shoulderCenterX = skeleton.Joints[JointType.ShoulderCenter].Position.X * 1000;
            shoulderCenterY = skeleton.Joints[JointType.ShoulderCenter].Position.Y * 1000;
            shoulderCenterZ = skeleton.Joints[JointType.ShoulderCenter].Position.Z * 1000;
        } catch (Exception error) {
            //
        }
    }
}

//Update Displays, Calculate Biometrics if Skeleton found.
if (colorData != null && depthData != null) {
    kinectDepthBitmap.RotateFlip(RotateFlipType.RotateNoneFlipX);
kinectCompositeBitmap.RotateFlip(RotateFlipType.RotateNoneFlipX);

KinectDepth.Image = kinectDepthBitmap;
KinectComposite.Image = kinectCompositeBitmap;

if(CordMapper != null)
{
    UpdateBiometrics();
}

stopwatch.Stop();
textBox10.Text = stopwatch.ElapsedMilliseconds.ToString();

// Sub-module Handles all Biometric calculations
private void UpdateBiometrics()
{
    // Depth Point manually found using Kinect System Explorer
    DepthImagePoint dpToW = new DepthImagePoint();
    dpToW.Depth = 708;
    dpToW.X = 9;
    dpToW.Y = 184;

    // Arbitrary Depth Point A for finding center of wheel
    DepthImagePoint dpPtA = new DepthImagePoint(); // Point A
dpPtA.Depth = 936;
dpPtA.X = 194;
dpPtA.Y = 280;

    // Arbitrary Depth Point B for finding center of wheel
    DepthImagePoint dpPtB = new DepthImagePoint(); // Point B
dpPtB.Depth = 1073;
dpPtB.X = 283;
dpPtB.Y = 245;

    // Convert the three points from Depth Reference to Reference
    SkeletonPoint skToW = CordMapper.MapDepthPointToSkeletonPoint(
        DepthImageFormat.Resolution640x480Fp30, dpToW);
    SkeletonPoint skpa = CordMapper.MapDepthPointToSkeletonPoint(
        DepthImageFormat.Resolution640x480Fps30, dpPtA);
    SkeletonPoint skpb = CordMapper.MapDepthPointToSkeletonPoint(
        DepthImageFormat.Resolution640x480Fps30, dpPtB);

    // Skeleton Point for Center of Wheel
    SkeletonPoint skCoW = new SkeletonPoint();
    skCoW.X = (float)(skpa.X+skpa.X-skpb.X);
    skCoW.Y = (float)(skpa.Y+skpa.Y-skpb.Y);
    skCoW.Z = (float)(skpa.Z+skpa.Z-skpb.Z);
    CalculateDeploymentVectors(skCoW);

    if (driver == true)
    {
        FindUserHeight();
        DetectHandCollision(dpPtA, dpPtB);

        // Distance from Top of Wheel
        textBox4.Text = ((Math.Sqrt(Math.Pow((headX - (skToW.X * 1000)), 2) +
            Math.Pow((headY - (skToW.Y * 1000)), 2) +
            Math.Pow((headZ - (skToW.Z * 1000)), 2)) / 25.4)*1.0216 -
            5.2369).ToString("F");

        // Distance from center of wheel
        textBox5.Text = ((Math.Sqrt(Math.Pow((headX - (skCoW.X * 1000)), 2) +
            Math.Pow((headY - (skCoW.Y * 1000)), 2) +
            Math.Pow((headZ - (skCoW.Z * 1000)), 2)) / 25.4)*1.014 -
            4.4682).ToString("F");
    }
}

private void DetectHandCollision(DepthImagePoint dpPtA, DepthImagePoint dpPtB)
{
    SkeletonPoint skLHand;
    SkeletonPoint skRHand;

    if (skeleton == null)
    {
        return;
    }
    jointCollection = skeleton.Joints;
    skLHand = jointCollection[JointType.HandLeft].Position;
    skRHand = jointCollection[JointType.HandRight].Position;
private void DrawHandCollision(DepthImagePoint dpPtA, DepthImagePoint dpPtB, SkeletonPoint skHandPt)
{
    // Center of the chest
    SkeletonPoint skptA = CordMapper.MapDepthPointToSkeletonPoint(DepthImageFormat.Resolution640x480Fps30, dpPtA);
    // Center of steering wheel
    SkeletonPoint skptB = CordMapper.MapDepthPointToSkeletonPoint(DepthImageFormat.Resolution640x480Fps30, dpPtB);
    // Line of Deployment Vector Points
    float x1 = skptA.X;
    float y1 = skptA.Y;
    float z1 = skptA.Z;
    float x0 = skptB.X;
    float y0 = skptB.Y;
    float z0 = skptB.Z;
    // Hand Points
    float x3;
    float y3;
    float z3;

    x3 = skHandPt.X - x0;
    y3 = skHandPt.Y - y0;
    z3 = skHandPt.Z - z0;

    // Invert the X skeleton point value so it displays correctly (not sure why)
    skHandPt.X = skHandPt.X * -1;

    Graphics g = Graphics.FromImage(kinectCompositeBitmap);
    Vector3 u = new Vector3(x1 - x0, y1 - y0, z1 - z0);
    Vector3 pq = new Vector3(x3, y3, z3);
    float distance = Vector3.Cross(pq, u).Length() / u.Length();
    //textBox4.Text = distance.ToString();
    if (distance < 0.203) // 203mm or 8 inches
    {
        DrawSkeletonDot(g, bluBrush, skHandPt);
    }
}

private void FindUserHeight()
{
    // Bottom of Seat Depth Point
    DepthImagePoint dpSeat = new DepthImagePoint();
    dpSeat.Depth = 1136;
    dpSeat.X = 306;
    dpSeat.Y = 478;
    SkeletonPoint skSeat = CordMapper.MapDepthPointToSkeletonPoint(DepthImageFormat.Resolution640x480Fps30, dpSeat);

    textBox1.Text = (HeadToNeck + NeckToSeat).ToString("F");
    textBox2.Text = ((HeadToNeck + NeckToSeat) / 0.43).ToString("F");
    textBox6.Text = Math.Floor(((HeadToNeck + NeckToSeat) / 0.43) / 12).ToString();
    textBox7.Text = (((HeadToNeck + NeckToSeat) / 0.43) % 12).ToString("F1");
}

private void CalculateDeploymentVectors(SkeletonPoint CenterOfWheel)
{
    BestX = 0;
    BestY = 0;
    float Y = 0;
    short depth;
    float LoFDepth = 0;

    DepthImagePoint dpCenterOfWheel = CordMapper.MapSkeletonPointToDepthPoint(CenterOfWheel, DepthImageFormat.Resolution640x480Fps30);

    // Torso Collision Detection
    // Search for collision point between line of deployment
    // and person’s torso from color camera X-coord 128-351
    for (int X = 128; X < 351; X++)
\[
Y = (\text{float})((-0.39167 \times X) + 356);
\]
\[
\text{depth} = \text{depthPixelData}[\text{int}((\text{Math.Round}(Y) \times 640) + X)];
\]

// F(X) = Y -- where Y is LOFDepth
// F(X) = 0.6928X + 736.3
\[
\text{LOFDepth} = (\text{float})(1.14167 \times X) + 714.5;
\]

if (Math.Round((\text{double})(\text{depth})) == Math.Round((\text{double})(\text{LOFDepth})))
{
    \text{BestX} = X;
    \text{BestY} = (\text{int})(Y);
    \text{driver} = \text{true};
    \text{break};
}
else if ((Math.Round((\text{double})(\text{depth})) - Math.Round((\text{double})(\text{LOFDepth})) <= 10) &&
    (Math.Round((\text{double})(\text{depth})) - Math.Round((\text{double})(\text{LOFDepth})) >= -10))
{
    \text{BestX} = X;
    \text{BestY} = (\text{int})(Y);
    \text{driver} = \text{true};
}

// No collision found
else if (X == 350 && \text{BestX} == 0)
{
    \text{BestX} = 0;
    \text{BestY} = 0;
    \text{driver} = \text{false};
    \text{break};
}

// If collision found draw deployment vector
if (\text{BestX} != 0)
{
    \text{DepthImagePoint dpLOF} = \text{new DepthImagePoint}();
    \text{dpLOF.Depth} = (\text{int})(\text{LOFDepth});
    \text{dpLOF.X} = 640 - \text{BestX};
    \text{dpLOF.Y} = \text{BestY};
    \text{CenterOfWheel.X} = 0 - \text{CenterOfWheel.X};
    \text{SkeletonPoint skLOF} = \text{CordMapper.MapDepthPointToSkeletonPoint(\text{DepthImageFormat.Resolution640x480Fps30, dpLOF});}
    \text{Graphics g} = \text{Graphics.FromImage(kinectCompositeBitmap)};
    \text{if (skLOF.X != 0)}
    {
        \text{DrawSkeletonLine(g, greenPen, skLOF, CenterOfWheel);}
    }
    \text{LOFDistanceVal.Text} = (\text{Math.Sqrt(\text{Math.Pow((CenterOfWheel.X * 1000) - (skLOF.X * 1000), 2) +
        Math.Pow((CenterOfWheel.Y * 1000) - (skLOF.Y * 1000), 2) +
    \text{LOFHeadProxVal.Text} = (\text{Math.Sqrt(\text{Math.Pow(headX - (skLOF.X * 1000), 2) +
        Math.Pow(headY - (skLOF.Y * 1000), 2) +
        Math.Pow(headZ - (skLOF.Z * 1000), 2) / 25.4).ToString("F");}

private void Form_FormClosing(object sender, FormClosingEventArgs e)
{
    //Stop the sensor
    StopKinect(sensor);
}

private void Kinect()
{
    //Set up the Kinect Sensor and Variables
    if (KinectSensor.KinectSensors.Count == 0)
    {
        return;
    }
    else
    {
        //Create sensor, enable streams, set resolutions, setup skeletal
        sensor = KinectSensor.KinectSensors[0];
        sensor.ColorStream.Enable(ColorImageFormat.RgbResolution640x480Fps30);
        sensor.DepthStream.Range = DepthRange.Near;
        sensor.DepthStream.Enable(DepthImageFormat.Resolution640x480Fps30);
        sensor.SkeletonStream.EnableTrackingInRangeNearRange = true;
    }
sensor.SkeletonStream.TrackingMode = SkeletonTrackingMode.Seated;
sensor.SkeletonStream.Enable(
    new TransformSmoothParameters() {
        Correction = 0.5f,
        JitterRadius = 0.05f,
        MaxDeviationRadius = 0.05f,
        Prediction = 0.5f,
        Smoothing = 0.5f
    });

//Create New Event for Kinect data
sensor.AllFramesReady += new EventHandler<AllFramesReadyEventArgs>(SensorAllFramesReady);

//Initialize data structures
KinectLock.WaitOne();
skeletonDataTemp = new Skeleton[6];
skeletonData = new Skeleton[6];
colorDataTemp = new byte[sensor.ColorStream.FramePixelDataLength];
colorData = new byte[sensor.ColorStream.FramePixelDataLength];
depthData = new DepthImagePixel[sensor.DepthStream.FramePixelDataLength];
depthDataTemp = new DepthImagePixel[sensor.DepthStream.FramePixelDataLength];
depthColorData = new byte[sensor.ColorStream.FramePixelDataLength];
KinectLock.ReleaseMutex();

//Start the Kinect sensor
sensor.Start();
faceTracker = new FaceTracker(sensor);
CoordMapper = new CoordinateMapper(sensor);

private void StopKinect(KinectSensor sensor)
{
    //Stop the Kinect sensor
    if (sensor != null)
    {
        sensor.Dispose();
        sensor.Stop();
    }
}

private void SensorAllFramesReady(object sender, AllFramesReadyEventArgs e)
{
    //Lock Mutex before data exchange
    KinectLock.WaitOne();

    //Get color frame
    using (ColorImageFrame colorImageFrame = e.OpenColorImageFrame())
    {
        if (colorImageFrame == null)
        {
            return;
        }
        colorImageFrame.CopyPixelDataTo(colorDataTemp);
    }

    //Get depth frame
    using (DepthImageFrame depthImageFrame = e.OpenDepthImageFrame())
    {
        if (depthImageFrame == null)
        {
            return;
        }
        depthImageFrame.CopyDepthImagePixelDataTo(depthDataTemp);
    }

    //Get skeletal frame
    using (SkeletonFrame skeletonFrame = e.OpenSkeletonFrame())
    {
        if (skeletonFrame == null)
        {
            return;
        }
        if (skeletonDataTemp == null || skeletonDataTemp.Length != skeletonFrame.SkeletonArrayLength)
        {
            skeletonDataTemp = new Skeleton[skeletonFrame.SkeletonArrayLength];
        }
        skeletonFrame.CopySkeletonDataTo(skeletonDataTemp);
    }

    //Process Skeletal for facial tracking data
    if (colorDataTemp != null && depthDataTemp != null && skeleton != null)
short[] temp = new short[depthDataTemp.Length];
for (int i = 0; i < depthDataTemp.Length; i++)
{
    temp[i] = (short)(depthDataTemp[i].Depth);
}

FaceTrackFrame faceFrame = faceTracker.Track(ColorImageFormat.RgbResolution640x480Fps30, colorDataTemp,
DepthImageFormat.Resolution640x480Fps30, temp, skeleton);
if (faceFrame.TrackSuccessful)
{
    facePoints = faceFrame.GetProjected3DShape();
    faceTriangles = faceFrame.GetTriangles();
    headRoll = faceFrame.Rotation.Z;
    headPitch = faceFrame.Rotation.X;
    headYaw = faceFrame.Rotation.Y;
}
else
{
    facePoints = null;
    faceTriangles = null;
}

private static Skeleton[] Skeleton_Clone(Skeleton[] skOrigin)
{
    //Copy skeletal data
    MemoryStream ms = new MemoryStream();
    BinaryFormatter bf = new BinaryFormatter();
    bf.Serialize(ms, skOrigin);
    ms.Position = 0;
    object obj = bf.Deserialize(ms);
    ms.Close();
    return obj as Skeleton[];
}

private static Bitmap PixelDataToBitmap(byte[] inputData, IntPtr Ptr)
{
    //Convert Color byte array into bitmap
    Marshal.Copy(inputData, 0, Ptr, inputData.Length);
    return new Bitmap(640, 480, 2560, System.Drawing.Imaging.PixelFormat.Format32bppRgb, Ptr);
}

private static Bitmap DepthDataFormatter(DepthImagePixel[] inputData)
{
    //Format depth short array into bitmap
    int colorPixelIndex = 0;
    DepthPixelData = new short[inputData.Length];
    for (int i = 0; i < inputData.Length; ++i)
    {
        short depth = inputData[i].Depth;
        ColorRGB c = HSL2RGB(((double)((depth-200)/8192),0.66,0.5));
        if (depth == 0)
        {
            depthColorData[colorPixelIndex++] = (byte)(0); //Blue
            depthColorData[colorPixelIndex++] = (byte)(0); //Green
            depthColorData[colorPixelIndex++] = (byte)(0); //Red
            depthColorData[colorPixelIndex++] = (byte)(255);
        }
        else
        {
            depthColorData[colorPixelIndex++] = (byte)(c.B); //Blue
            depthColorData[colorPixelIndex++] = (byte)(c.G); //Green
            depthColorData[colorPixelIndex++] = (byte)(c.R); //Red
            depthColorData[colorPixelIndex++] = (byte)(255);
        }
    }
    try
    {
        depthPixelData[i] = depth;
    }
    catch (Exception error)
private static void SkeletonToComposite()
{
    // Draw Skeleton to Bitmap
    jointCollection = skeleton.Joints;
    Graphics g = Graphics.FromImage(kinectCompositeBitmap);

    // Draw Dots
    DrawSkeletonDot(g, redBrush, jointCollection[JointType.Head].Position);
    DrawSkeletonDot(g, redBrush, jointCollection[JointType.ShoulderCenter].Position);
    DrawSkeletonDot(g, redBrush, jointCollection[JointType.ShoulderLeft].Position);
    DrawSkeletonDot(g, redBrush, jointCollection[JointType.ShoulderRight].Position);
    DrawSkeletonDot(g, redBrush, jointCollection[JointType.ElbowLeft].Position);
    DrawSkeletonDot(g, redBrush, jointCollection[JointType.ElbowRight].Position);
    DrawSkeletonDot(g, redBrush, jointCollection[JointType.WristLeft].Position);
    DrawSkeletonDot(g, redBrush, jointCollection[JointType.WristRight].Position);
    DrawSkeletonDot(g, redBrush, jointCollection[JointType.HandLeft].Position);
    DrawSkeletonDot(g, redBrush, jointCollection[JointType.HandRight].Position);

    // Draw Lines
    DrawSkeletonLine(g, redPen, jointCollection[JointType.Head].Position,
                     jointCollection[JointType.ShoulderCenter].Position);
    DrawSkeletonLine(g, redPen, jointCollection[JointType.ShoulderCenter].Position,
                     jointCollection[JointType.ShoulderLeft].Position);
    DrawSkeletonLine(g, redPen, jointCollection[JointType.ElbowLeft].Position,
                     jointCollection[JointType.WristLeft].Position);
    DrawSkeletonLine(g, redPen, jointCollection[JointType.ElbowLeft].Position,
                     jointCollection[JointType.ShoulderRight].Position);
    DrawSkeletonLine(g, redPen, jointCollection[JointType.ElbowRight].Position,
                     jointCollection[JointType.WristRight].Position);
    DrawSkeletonLine(g, redPen, jointCollection[JointType.WristRight].Position,
                     jointCollection[JointType.HandLeft].Position);
    DrawSkeletonLine(g, redPen, jointCollection[JointType.WristRight].Position,
                     jointCollection[JointType.HandRight].Position);
}

private static void DrawSkeletonDot(Graphics g, System.Drawing.SolidBrush brush, SkeletonPoint skPt)
{
    // Convert skeleton point into X,Y and draw dot
    DepthImagePoint dpPt = CordMapper.MapSkeletonPointToDepthPoint(skPt, DepthImageFormat.Resolution640x480Fps30);
    g.FillEllipse(brush, dpPt.X-5, dpPt.Y-5, 10, 10);
}

private static void DrawSkeletonLine(Graphics g, System.Drawing.Pen pen, SkeletonPoint skPt1, SkeletonPoint skPt2)
{
    // Draw line between two skeleton points
    DepthImagePoint dpPt1 = CordMapper.MapSkeletonPointToDepthPoint(skPt1, DepthImageFormat.Resolution640x480Fps30);
    DepthImagePoint dpPt2 = CordMapper.MapSkeletonPointToDepthPoint(skPt2, DepthImageFormat.Resolution640x480Fps30);
    g.DrawLine(pen, dpPt1.X, dpPt1.Y, dpPt2.X, dpPt2.Y);
}

private static void FaceMaskToComposite()
{
    // Draw face mask points onto bitmap and build triangles on bitmap
    Graphics g = Graphics.FromImage(kinectCompositeBitmap);
    foreach (FaceTriangle triangle in faceTriangles)
    {
        try
        {
            DrawFaceTriangle(g, bluPen,
                             facePoints[(FeaturePoint)triangle.First],
                             facePoints[(FeaturePoint)triangle.Second],
                             facePoints[(FeaturePoint)triangle.Third]);
        }
        catch (Exception error)
        {
            continue;
        }
    }
}

private static void DrawFaceTriangle(Graphics g, System.Drawing.Pen pen,
private static void GetSkeleton()
{
    //Find useable skeleton
    if (skeletonData != null)
    {
        skeleton = null;
        foreach (Skeleton skel in skeletonData)
        {
            if (skel != null)
            {
                if (skel.TrackingState == SkeletonTrackingState.Tracked)
                {
                    skeleton = skel;
                }
            }
        }
    }
}

public struct ColorRGB
{
    public byte R;
    public byte G;
    public byte B;

    public ColorRGB(System.Drawing.Color value)
    {
        this.R = value.R;
        this.G = value.G;
        this.B = value.B;
    }

    public static implicit operator System.Drawing.Color(ColorRGB rgb)
    {
        return c;
    }

    public static explicit operator ColorRGB(System.Drawing.Color c)
    {
        return new ColorRGB(c);
    }
}

public static ColorRGB HSL2RGB(double h, double s, double l)
{
    // Given H,S,L in range of 0-1
    // Returns a Color (RGB struct) in range of 0-255
    double r, g, b;
    r = l; // default to gray
    g = l;
    b = l;
    v = (l <= 0.5) ? (l * (1.0 + s)) : (l + s - l * s);
    if (v > 0)
    {
        double m;
        double sv;
        int sextant;
        double fract, vsf, mid1, mid2;
        m = l + l - v;
        sv = (v - m) / v;
        h *= 6.0;
        sextant = (int)h;
        fract = h - sextant;
        vsf = v * sv * fract;
        mid1 = m + vsf;
        mid2 = v - vsf;
        switch (sextant)
        {
            case 0:
                r = v;
                g = mid1;
                b = m;
                break;
            case 1:
                r = mid2;
g = v;
b = m;
break;
case 2:
r = m;
g = v;
b = mid1;
break;
case 3:
r = m;
g = mid2;
b = v;
break;
case 4:
r = mid1;
g = m;
b = v;
break;
case 5:
r = v;
g = m;
b = mid2;
break;
}
ColorRGB rgb;
rgb.R = Convert.ToByte(r * 255.0f);
rgb.G = Convert.ToByte(g * 255.0f);
rgb.B = Convert.ToByte(b * 255.0f);
return rgb;
private static Mutex KinectLock = new Mutex();
private static KinectSensor sensor;
private static FaceTracker faceTracker = null;
private static EnumIndexableCollection<FeaturePoint, Microsoft.Kinect.Toolkit.FaceTracking.Point> facePoints = null;
private static Microsoft.Kinect.Toolkit.FaceTracking.FaceTriangle[] faceTriangles = null;
private static FaceTrackFrame faceFrame = null;
private static Skeleton skeleton = null;
private static Skeleton[] skeletonData = null;
private static DepthImagePixel[] depthDataTemp = null;
private static DepthImagePixel[] depthData = null;
private static byte[] colorDataTemp = null;
private static byte[] colorData = null;
private static short[] depthPixelData = null;
private static JointCollection jointCollection = null;
private static CoordinateMapper CordMapper = null;
// Bitmaps
private static Bitmap kinectCompositeBitmap = null;
private static Bitmap kinectDepthBitmap = null;
// Pointers
private static IntPtr colorPtr;
private static IntPtr depthPtr;
private static IntPtr compositePtr;
// Brushes
// Joint points and orientations
private static float headX = 0;
private static float headY = 0;
private static float headZ = 0;
private static float shoulderCenterX = 0;
private static float shoulderCenterY = 0;
private static float shoulderCenterZ = 0;
private static float headRoll = 0;
private static float headPitch = 0;
private static int BestX = 0;
private static int BestY = 0;
private static bool driver = true;
// Timer event