Team 504: The Examination of Occupant and Vehicle Responses to Low Speed Rear-End Crashes

Michael Small, Joseph Godio, Robert Montuoro, Vincent Grimes, William Abraira

FAMU-FSU College of Engineering

Operations Manual

**Table of Contents**

[Project Overview 3](#_Toc4781436)

[Component/Model Description 3](#_Toc4781437)

[Mechanical Systems 3](#_Toc4781438)

[Experiment Design 3](#_Toc4781439)

[Visual Tracking 4](#_Toc4781440)

[Analytical Tracking 5](#_Toc4781441)

[Assembly 7](#_Toc4781442)

[The Bumper Mount 7](#_Toc4781443)

[Experimental Setup 9](#_Toc4781444)

[Requirements Pre-Testing 9](#_Toc4781445)

[Testing Procedure 9](#_Toc4781446)

[Post-Testing Check 9](#_Toc4781447)

[Safety 9](#_Toc4781448)

[Arduino Wiring 10](#_Toc4781449)

[Troubleshooting 10](#_Toc4781450)

[Electronic Components 10](#_Toc4781451)

[Mounting the Bumper 10](#_Toc4781452)

[Arduino Code 10](#_Toc4781453)

# Project Overview

The overall objectives of the project are to create a general empirical model for low speed collisions below 7mph. In order to do this, live crash tests are done to examine the forces sustained by the struck occupant and vehicle. After ample trials are run, the data will be imported into Mathematical Dynamic Modeling (MADYMO), a software that helps to create models based on an input of forces or values. In this context, the forces examined on the occupant will be added to MADYMO a relationship will be determined between the occupant and vehicle responses. Along with furthering the minimal knowledge of low speed collisions, the results of this project is to also validate expert witness claims in court when the case is based upon these similar conditions.

# Component/Model Description

The following section lists the major components in our order sheet with brief explanations stating the reasoning and importance of each part when looking at our experiment.

## Mechanical Systems

Struck Car – The struck car is the car that is stationary at the start of the experiment and is then hit from behind by the striking car. This struck car contains the laptop, the wiring, the accelerometers, the mount, the camera, and the occupant. Looking at the specifics, the occupant is seated in the driver’s seat with the accelerometers attached to his head and chest. While four accelerometers are placed on the body, two accelerometers are attached to the struck car in order to capture its movement. The computer, Arduino, and wiring is placed in the passenger’s side seat and floorboard in order to not get in the way of testing. The camera and mount are also seated in the car during testing in order to capture the occupant motion.

Striking Car– The striking car hits the struck car from behind at a speed of 7mph. Traveling in a street line, the striking car is used to be the stimulus for the experiment.

Bumpers– Bumpers are used to switch in and out of experimentation if needed. Because there are a wide variety of bumpers based on the car shape, size, and model, testing a multitude of bumpers can help when speaking about specific car accidents. Although not necessary, additional bumpers can be used to test these varying hypothesis.

## Experiment Design

Traffic Cones– Traffic cones help to set up the path that both cars will travel. These eliminate variation in the experiment by allowing the start, end, and width components of the track to be set before and after each run.



*Figure 1. Lakeside Plastics 12” Traffic Construction Safety Cones. Quantity: 6*

Additional Components – In order to draw a clearer path/set a clearer route for the truck to draw, additional markers could be used such as chalk (depending on testing grounds), Kaizan Tape for additional floor marking, etc.

## Visual Tracking

Gimbal– The gimbal is used to house the camera during experimentation. Along with using accelerometers to track the acceleration of the occupant the vehicle, the camera will be used to add a visual tracking component to the project. In order to decrease the vibrational effects of the experiment, the gimbal is used to adjust for the crash impulse felt by the truck.



*Figure 2. Zhiyun-Tech Crane v2 3-Axis Handheld Gimbal Stabilizer. Quantity: 1*

Camera– The camera, in partnership with the gimbal, is used to add the visual tracking for the experiment.



*Figure 3. Casio Exilim EX F1. Quantity: 1*

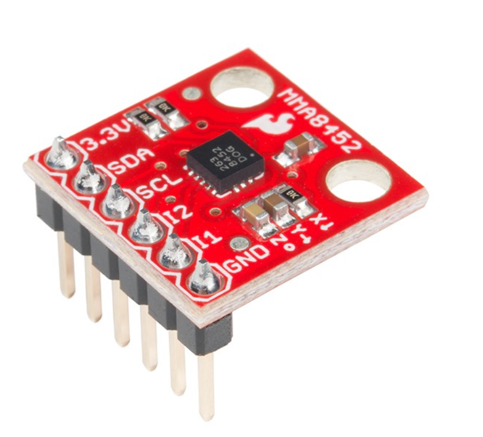
Ratchet Straps – To assure the camera mount does not move during experimentation, necessary tiedowns are required in order to fix the mount to the car. In doing so, ratchet straps are used to extend across the stems of the gimbal so it will be fixed in the car yet allow the mounting part to move in order to track the occupant during experimentation.



*Figure 4. 1” x 10’ Ratchet Straps. Quantity: 1*

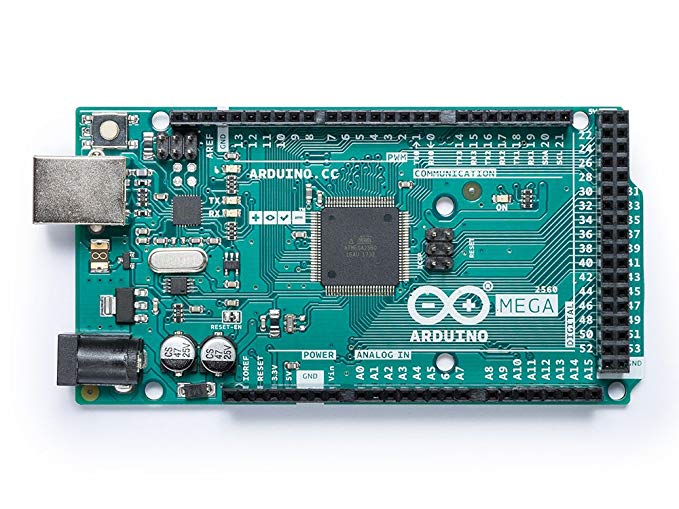
## Analytical Tracking

Accelerometers– In order to capture the accelerations of the occupant and vehicles, accelerometers are used to capture the momentary accelerations due to the forces obtained by the crash pulse. As discussed above, there are 6 accelerometers used that are placed on the occupant and vehicle at strategic points.



*Figure 5. SparkFun Triple Axis Accelerometer Breakout – MMA8452Q. Quantity: 6*

Arduino– For the accelerometers to properly work, a code must be implemented into them in order to have the proper outcome. Using the code written in Arduino, the accelerometers will read the X, Y, and Z directions along with displaying the code in an Excel spreadsheet.



*Figure 6. Arduino MEGA 2560 REV3. Quantity: 1*

Male Dupont Wire – In order to connect the accelerometers to the Arduino board, male-to-male wires must be used on each accelerometer. Each accelerometer has an X, Y, Z, ground, and 3V connection that needs to be connected in order to have the proper connection for testing. Given the 6 accelerometers that will be used in testing, 30 male deponent wires total will be needed for testing. The diagram for connecting the wires will be found in the later section titled ‘Arduino Schematic’.



*Figure 7. SUNKEE 20CM Male to Male Dupont Wire Color Jumper Cable. Quantity: 30*

Stranded Copper Wire – To account for the length of the Arduino connection, excess wire is used to make the total connection longer. Copper wire is used in order to split the dupont cable and lengthen it to the desired measurements given the car and setup type.



*Figure 8. 50ft Primary Stranded Copper Wire. Quantity: 1*

Headwear– In order to set up the accelerometers on the individual, a headwear mount must be used in order to house the accelerometers and let them be worn by the occupant. The headwear used is a simple beanie with the three accelerometers stitched into the inside. With an ease of adjustability, the beanie allows for a durability told hold through testing along with changes to be made to the design if needed.



*Figure 9. Top Level Unisex Cuffed Plain Beanie. Quantity: 1*

## Software Needed

For this experiment, a basic understanding of Arduino and its components are needed along with an understanding of Mathematical Dynamic Modeling (MADYMO). The two programs are used to collective and analyze the data gathered respectively.

# Assembly

## The Bumper Mount

In order to set up the experiment, the first piece needed to be set up is the bumper mount. In a brief summary, the old bumper needs to be taken off and replaced with the mount, which involves placing on the driver and passenger’s side frame, installing the foam core and piston isolators, and screwing in the middle connection piece in order to fit all of the pieces together. The final bumper pieces screw into the bumper that is to be tested. While our experiment is more towards the experimental testing rather than the bumper mount, the procedure is less fleshed out. The attachments are as followed:

*Figure 10. From left to right: driver’s side frame, passenger’s side frame, piston isolators*



*Figure 11. Middle connection piece*

When looking at the *Figure 10.*, the two pieces that connect to the car are seen in the left and center. In order to connect to the replaceable bumper, the image on the right is used to attach to both ends of the bumper. In order to change the width to adjust for specific bumpers, the piece from *Figure 11.* is used as a buffer between the aforementioned pieces. With the total pieces screwed together, the total bumper mount can be seen below in *Figure 12.*



*Figure 12.* The complete bumper mount

# Experimental Setup

## Requirements Pre-Testing

In order for testing to begin, the following steps must be completed:

1. The proper wiring has been done to the Arduino
2. The code for the accelerometers has been uploaded from the computer
3. The wires are in such a condition to where testing will not tamper with their position
4. All the accelerometers must be mounted onto the occupant and car
5. The bumper mount has been properly installed to the struck vehicle
6. The bumper with which to be tested has been installed onto the mount
7. The camera mount has been positioned in the struck car with the camera set in place
8. The proper tires PSI has been checked in order to make sure the car is level
9. The testing grounds have been examined and all necessary debris has been removed
10. The vehicles are aligned in a way so that they will hit with no angle
11. The necessary medical equipment has been obtained in case of an accident
12. All parties are aware that the testing is to begin

## Testing Procedure

The set up for this experiment requires that the struck vehicle was parked and secured in place at a distance of 20 feet from the striking vehicle. This allows for the striking vehicle to reach and maintain a speed of 7 mph (and reach the desired delta v of 7 mph) before colliding with the struck vehicle. The path for the vehicle was guided using traffic cones and chalk lines ensuring that the striking vehicle impacted the rear bumper of the struck vehicle in the same spot. The camera gimbal was secured in the door handle of the passenger door and firmly secured using ratchet straps. At this position in the struck vehicle, the camera can capture the motion of the occupant before, during, after the impact and as a second form of acceleration measurement. The primary form of acceleration measurement were the accelerometers that were in the beanie worn by the occupant and placed on the rear bumper. For each trial the camera was turned on and recording with the Arduino plugged into the laptop with excel open and the code activated so the data from the collision could be fed live into the spreadsheet and saved. This process is repeated for each trial, resetting the electronics, returning the striking vehicle to the start position, and ensuring that all the equipment was returned to its original set up to keep measurements consistent.

## Post-Testing Check

After every run, it is necessary to check and make sure the data was properly recorded. Once the data is deemed to be imported, resetting the Excel spreadsheet is needed so further data can be collected. Aside from the accelerometer data, it is necessary to make sure the camera is reset for the next trial, the cars are moved back to the original starting positions, and all other testing parameters are kept constant. Once all parties are notified, the next run can begin.

## Post-Testing Analysis

After all testing is complete, the data must then be analyzed in Mathematical Dynamics Modeling. The program features different tools that can take the accelerometer data and format it in a way that can be used to model dynamic changes in motion. In this scenario, we are modeling how an individual responds in a low speed rear-end impact collision.

## Safety

If an issue involving the safety of any persons involved, all other persons will use the medical kit to tend to the injured. If the problem persists, contact the necessary emergency/medical personnel. While this may be calling 911 or using tools found in the safety kit, it is important to ensure the wellbeing of all involved before testing is to resume.

# Arduino Wiring

To wire up the connections, each accelerometer is to be hooked up to the Arduino as followed:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Power | Ground | X | Y | Z |
| Accelerometer 1 | 3V | GND | A0 | A1 | A2 |
| Accelerometer 2 | 3V | GND | A3 | A4 | A5 |
| Accelerometer 3 | 3V | GND | A6 | A7 | A8 |
| Accelerometer 4 | 3V | GND | A9 | A10 | A11 |
| Accelerometer 5 | 3V | GND | A12 | A13 | A14 |

# Troubleshooting

## Electronic Components

If issues involving the accelerometers or data acquisition arise, consider the following:

* Check the soldered joints. It is likely that one of the soldering joints is wrong and thus cannot give the proper data to the computer.
* Examine the Arduino Schematic and ensure that all wires are connected properly.
* Ensure the code is both correct and has been uploaded to the accelerometers.
* Check the computer to make sure no other programs are interfering with the data collection process.
* Recalibration of the accelerometers is a necessary component that should be done if ‘odd’ readings are being obtained. This link has a useful guide to properly calibrating said accelerometers: https://learn.adafruit.com/adafruit-analog-accelerometer-breakouts/calibration-and-programming

## Mounting the Bumper

Ensure that the bumper is properly aligned and the holes are accurate to what is shown in *Figure 12.* While it may look simple, the bumper design is rather technical and involves a decent understanding in order to properly install. If the bumper appears to be loose or angled, contact Beau Biller, Cummings Scientific Representative, or Michael Small, Team 504 Systems Integrator, for an in-depth walkthrough.

# Arduino Code

int scale = 200;

boolean micro\_is\_5V = true;

void setup() {

// put your setup code here, to run once:

Serial.begin(128000);

Serial.println("CLEARDATA"); //clears up any data left from previous projects

Serial.println("LABEL,TIME,X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3,X4,Y4,Z4,X5,Y5,Z5"); //always write LABEL, so excel knows the next things will be the names of the columns (instead of Acolumn you could write Time for instance)

Serial.println("RESETTIMER"); //resets timer to 0

}

void loop() {

// put your main code here, to run repeatedly:

int rawX1 = analogRead(A0);

int rawY1 = analogRead(A1);

int rawZ1 = analogRead(A2);

int rawX2 = analogRead(A3);

int rawY2 = analogRead(A4);

int rawZ2 = analogRead(A5);

int rawX3 = analogRead(A6);

int rawY3 = analogRead(A7);

int rawZ3 = analogRead(A8);

int rawX4 = analogRead(A9);

int rawY4 = analogRead(A10);

int rawZ4 = analogRead(A11);

int rawX5 = analogRead(A12);

int rawY5 = analogRead(A13);

int rawZ5 = analogRead(A14);

double X1,Y1,Z1, scaledX1, scaledY1, scaledZ1,X2,Y2,Z2, scaledX2, scaledY2, scaledZ2,X3,Y3,Z3, scaledX3, scaledY3, scaledZ3,X4,Y4,Z4, scaledX4, scaledY4, scaledZ4,X5,Y5,Z5, scaledX5, scaledY5, scaledZ5; // Scaled values for each axis

scaledX1 = map(rawX1, 0, 675, -scale, scale); // 3.3/5 \* 1023 =~ 675

scaledY1 = map(rawY1, 0, 675, -scale, scale);

scaledZ1 = map(rawZ1, 0, 675, -scale, scale);

scaledX2 = map(rawX2, 0, 675, -scale, scale); // 3.3/5 \* 1023 =~ 675

scaledY2 = map(rawY2, 0, 675, -scale, scale);

scaledZ2 = map(rawZ2, 0, 675, -scale, scale);

scaledX3 = map(rawX3, 0, 675, -scale, scale); // 3.3/5 \* 1023 =~ 675

scaledY3 = map(rawY3, 0, 675, -scale, scale);

scaledZ3 = map(rawZ3, 0, 675, -scale, scale);

scaledX4 = map(rawX4, 0, 675, -scale, scale); // 3.3/5 \* 1023 =~ 675

scaledY4 = map(rawY4, 0, 675, -scale, scale);

scaledZ4 = map(rawZ4, 0, 675, -scale, scale);

scaledX5 = map(rawX5, 0, 675, -scale, scale); // 3.3/5 \* 1023 =~ 675

scaledY5 = map(rawY5, 0, 675, -scale, scale);

scaledZ5 = map(rawZ5, 0, 675, -scale, scale);

X1 = scaledX1 + 7;

Y1 = scaledY1 + 8;

Z1 = scaledZ1 + 7;

X2 = scaledX2 + 7;

Y2 = scaledY2 + 8;

Z2 = scaledZ2 + 8;

X3 = scaledX3 + 7;

Y3 = scaledY3 + 7;

Z3 = scaledZ3 + 7;

X4 = scaledX4 + 7;

Y4 = scaledY4 + 8;

Z4 = scaledZ4 + 7;

X5 = scaledX5 + 7;

Y5 = scaledY5 + 8;

Z5 = scaledZ5 + 7;

Serial.print("DATA,TIME,"); //writes the time in the first column A and the time since the measurements started in column B

Serial.print(X1);

Serial.print(',');

Serial.print(Y1);

Serial.print(',');

Serial.print(Z1);

Serial.print(',');

Serial.print(X2);

Serial.print(',');

Serial.print(Y2);

Serial.print(',');

Serial.print(Z2);

Serial.print(',');

Serial.print(X3);

Serial.print(',');

Serial.print(Y3);

Serial.print(',');

Serial.print(Z3);

Serial.print(',');

Serial.print(X4);

Serial.print(',');

Serial.print(Y4);

Serial.print(',');

Serial.print(Z4);

Serial.print(',');

Serial.print(X5);

Serial.print(',');

Serial.print(Y5);

Serial.print(',');

Serial.println(Z5);

delay(100);

}