High-Speed Photography

Isaac Lambing

1 Abstract

The goal of this project is to produce a system using an Arduino microcontroller that allows the user to capture events on photographic film that happen too quickly to be viewed by the human eye. The system consists of multiple sensors (laser, sound, and infrared) that all serve to trigger a camera strobe. The final product includes the ability to switch between these sensors and set a variable delay time through a physical interface (potentiometers and switches).

2 Introduction

High-speed photography doesn't need much of an introduction; we've all seen the pictures of water balloons mid-burst and apples immediately after a bullet has cruised through their center. What does need some clarification, however, is how these pictures are obtained. Trial and error? Huge expensive rigs? Hardly. While trial and error does come into play every so often with some of the slower moving collisions (like drops of water), it is very time consuming and becomes nearly impossible when attempting to capture an object approaching the speed of a bullet. Expensive rigs, while sometimes used, are hardly necessary to begin capturing high-speed events.

The two main methods used in high-speed photography involve either an extremely fast shutter or a strobe [1]. Both of these methods use some sort of sensor to trigger them, as human reactions are too slow to capture most instances. The fast shutter method does not require any special circumstances, although the strobe method does. If a strobe is used, the subject must be placed in a pitch-black room. The camera is manually set to a long exposure time (anywhere from three to ten sec-

---

This paper was written for Dr. James Dann's Applied Science Research class in the spring of 2010.

onds) and the flash triggers for a split second. The split second the flash is on is the only time when the film is being exposed, which allows for the capturing of a precise moment.

This project will utilize the strobe method, allowing the user to replicate high-speed photography using a microcontroller. It includes multiple sensing options that are used in conjunction with a microcontroller to trigger a flash. These sensing options include a microphone, laser, and infrared sensor. These will be explained in detail further in the paper, but ultimately all the sensors serve to trigger the flash unit.

The idea for the project came from a website [2] that listed the top 40 projects that use the Arduino (a cheap, widely used microcontroller) [3]. The article that ultimately led to this project was entitled "How to: High-speed Photography using the Arduino" [4].

The system created for this project is interesting not only for the amazing pictures it can produce, but for its successor: high-speed videography. This allows the whole process of whatever is happening (a bulb shattering, for example) to be played back frame by frame and viewed in excruciating detail. It has huge potential beyond simple entertainment, as it can be used to understand occurrences that happen much too fast to be taken in by the human eye. Videos played back in slow motion can be used to analyze everything from surface tension to missile accuracy [5]. The techniques for high-speed videography are completely different than for high-speed photography, however, as the cameras utilize a rotating prism rather than a shutter to capture images [6]. As a result my work is limited to photography, which has less scientific use but high entertainment value.

While the definitive start to high-speed photography is speculative, some consider it to be the result of experiments done in 1851 by William Henry Fox Talbot [7]. He attached a newspaper article to a rotating wheel (rotating fast enough that the article could not be read) and then, in a dark room, exposed the newspaper to light for a fraction of a second using a Leyden Jar [8] (an early form of what we know as a capacitor) to generate a spark. The resulting image that was captured on a wet plate was sharp enough to be legible.

The first practical use of this branch of photography came about in the 1870s, when Eadweard Muybridge questioned whether or not the feet of a galloping horse were all off the ground simultaneously [9]. Muybridge used a set of tripwires, each set to trigger a different camera in order to capture the full range of a horse's gallop and settle the question in the affirmative.

A huge breakthrough for high-speed photography came with invention of the Stroboscopic Flash System developed by Dr. Harold Edgerton [10]. This system allowed for extremely short bursts of light to be produced, which allowed William Talbot's method to work with much faster occurrences. Edgerton (1903-1990) is well known for his iconic images such as the bullet through the apple (Figure 1) and the photo of a milk drop, which "was featured in the New York Museum of Modern Art's first photography exhibit" in 1937 (Figure 2). This invention paved the way for high-speed photography and benefited everyone from physicists to army generals.

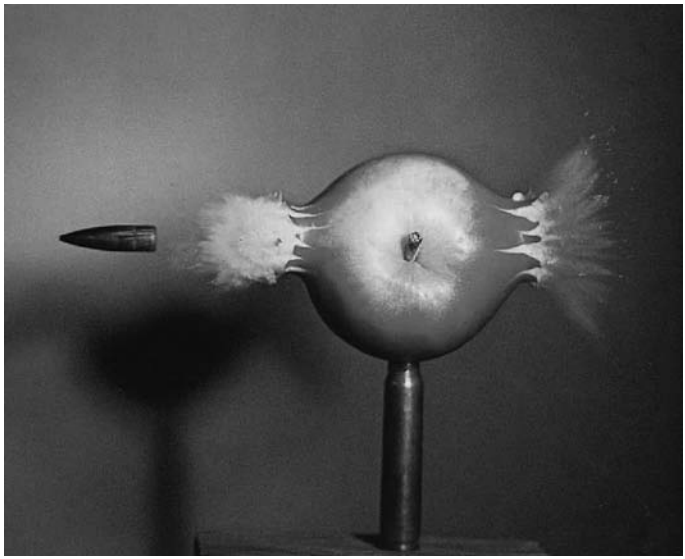

Figure 1: *Edgerton's bullet through the apple photo.*

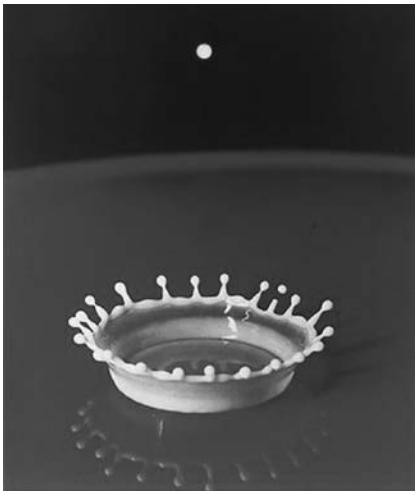

Figure 2: *Edgerton's milk drop photo.*

3 Design and Construction

The duration of the flash from a typical flash unit hacked from a digital camera is about 1/1700th of a second, or approximately 590 microseconds [11]. Given the relatively low velocity of a falling water droplet, the disposable flash is able to capture a splash with relative sharpness. However, this ultimately means that it is necessary to utilize a higher quality flash unit so that shorter flash durations can be achieved. This allows for other sorts of pictures to be taken, like projectiles firing from an air gun or a Christmas ornament being shattered with a baseball bat. If I were to try to take a picture of a projectile from an airsoft gun traveling, say, 300 feet per second (a relatively slow speed) with a disposable camera flash, the BB would travel approximately: 2.1 in: (300 ft)*(12 ft/in)/(1700) during the time light is being expelled from the flash. The photo taken from such an endeavor would be no more than a streak of white.

The three different sensors, (laser/tripwire, sound, and infrared) all ultimately work in the same way. They take an input (movement of a physical object or a sound wave) and send a signal to the Arduino to trigger the flash circuit. This concept is outlined in Figure 3.

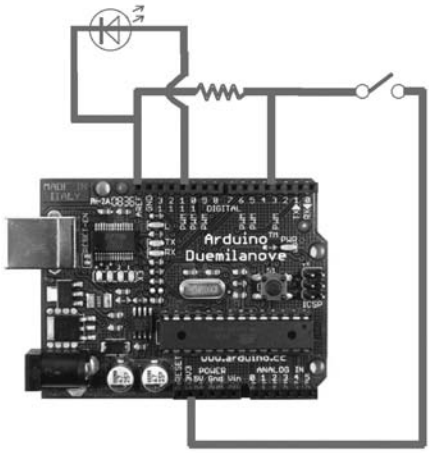

Figure 3: *The basic circuit. To illustrate the basics, the flash circuit is represented by an LED and the sensor by a switch. See Figure 4 for the completed circuit.*

This circuit is used to show how the final setup will work. The switch serves in place of a sensor (either laser trip-wire, infrared, or microphone), and the LED saves the place of what will be the high-voltage flash unit. When the switch closes (or the sensor is triggered) a signal is sent to the Arduino, which then lights the LED (or triggers the flash). An optoisolator must also be used in conjunction with the flash. By isolating the flash from the Arduino, this serves to eliminate any damage to the chip from a large voltage spike. The resistance of the resistor is 10,000 Ω, which limits the amount of current that travels to the digital input. The basic coding for the Arduino is as follows:

```
const int buttonPin = 2;     // the number of the pushbutton pin
 const int ledPin =  13;      // the number of the LED pin
 // variables (named as integers) will change:
 int buttonState = 0;         // variable for reading the pushbutton status
 void setup() {
   // initialize the LED pin as an output:
   pinMode(ledPin, OUTPUT);
   // initialize the pushbutton pin as an input:
   pinMode(buttonPin, INPUT);
 }
 void loop(){
```

```
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);
  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  }
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

(This coding is adapted from a tutorial on the Arduino website, found here: http://Arduino.cc/en/Tutorial/Button.)

The laser sensor (Figure 4) will function as a tripwire. The laser will be powered by one of the Arduino's digital pins and will be focused on a photoresistor. The photoresistor will serve as a break in a circuit between +5 V and an analog input on the Arduino. An analog input is used so that the system can be fine-tuned; a 'cutoff' value can be set so that the code triggers only when the current drops below a certain level. When an object is dropped and breaks the line of the laser, the photoresistor increases in resistance and decreases the flow of current into the analog input. This drop in current is sensed by the Arduino and used to do two things. First, the power to the laser is turned off so that it will not be captured in the picture. Second, after a manually set delay (normally no more than 10 milliseconds), the Arduino triggers the flash circuit to light the scene and expose the film in the camera for a split second.

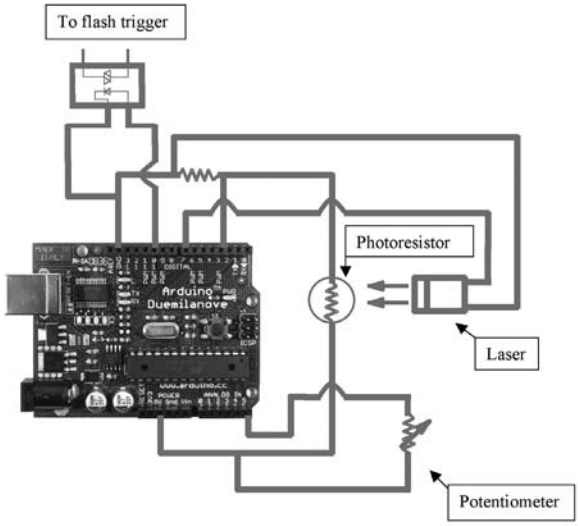

Figure 4: *Completed circuit with laser sensor and potentiometer. Laser sensor is comprised of laser and photoresistor, while potentiometer controls variable delay.*

This coding was modified slightly when the laser sensor was added (Figure 4), as an analog pin was used to increase the adjustability of the trigger and a potentiometer was added to introduce variable delay. The coding for the laser tripwire is as follows:

```
const int sensorPin = 0; // set input pin for tripwire
const int laser = 7;  // set output pin to power laser
const int ledPin =  8;  // set output pin to optoisolator (to trigger flash)
int sensorVal = 0; // defines variable for measuring analog input

void setup() {
   pinMode(ledPin, OUTPUT); // defines pin as output
  pinMode(laser, OUTPUT); // defines pin as input
  digitalWrite(laser, HIGH);  // turns laser on upon startup (so
photoresistor will sense light and not immediately trigger flash)
  delay(1000); // delay to make sure laser is aligned
```

```
}

void loop(){

  sensorVal=analogRead(sensorPin); // set variable equal to value
from analog input
  if (sensorVal<500) // if laser is broken, causing analog reading to
drop below predetermined value, then:
{
    digitalWrite(laser, LOW); // turn laser off (so it won't be in picture)
    digitalWrite(ledPin, HIGH); // trigger flash
    delay(10);
    digitalWrite(ledPin, LOW); // stop voltage to flash circuit
    delay(5000);  //wait 5 seconds so that flash doesn't trigger again
while shutter is open
  }
  else {
    digitalWrite(ledPin, LOW); // keep flash off
   digitalWrite(laser, HIGH);  // keep laser on
  }
}
```

The sound sensor (Figure 5) works in a slightly different way. A regular studio microphone was hooked up to a guitar amplifier to provide a large enough signal even when very faint sounds (e.g., a water drop) needed to be heard. The signal from the amplifier was constantly fed into a digital pin on the Arduino through a diode in order to prevent the Arduino from receiving the negative portion of the sound wave, which could cause damage. The microcontroller checks this signal to see if the signal passes a threshold range determined in the programming. If it is above the predetermined range, then the Arduino will trigger the flash circuit and expose the film in the camera.

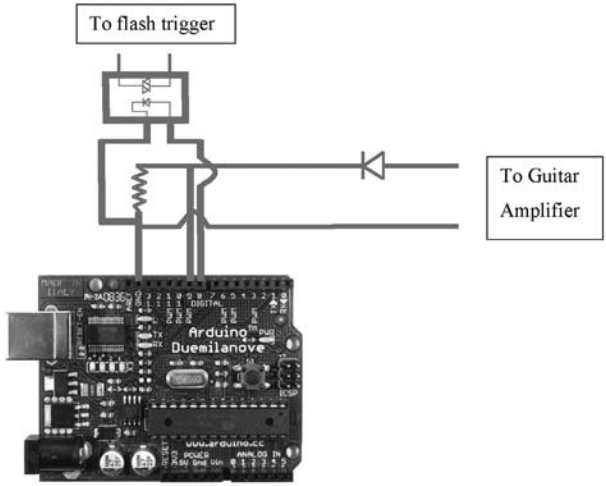

Figure 5: *Sound sensor. The diode is used to prevent negative voltage from potentially damaging the Arduino.*

Coding:

```
const int led = 3;
const int input = 2;
int soundVal = 0;

void setup() {
  pinMode (led, OUTPUT);
  Serial.begin(19200);
}

void loop(){
  soundVal=analogRead(input);
  Serial.println(soundVal);

  if (soundVal > 10) {
  digitalWrite(led, HIGH);
  delay(100);
  digitalWrite(led, LOW);
  }
  else {
    digitalWrite(led, LOW);
  }
}
```

The infrared sensor (Figure 6) can be thought of as a modified tripwire. Two infrared sensors are placed two inches apart. Each sensor is made up of an infrared LED and an infrared transistor facing each other. When voltage is applied to the LED, infrared light triggers the base of the photo-transistor, which allows voltage to flow into an input in the Arduino. However, when an object passes through a sensor, the infrared light path is broken and voltage is abruptly stopped. The Arduino pin senses this and starts a timer. When the second sensor is tripped, the timer stops and the elapsed time is stored as a variable. This variable is then multiplied by a certain number based on when you want the flash to trigger (e.g., 10 for 20 inches past the sensor, 15 for 30 inches past the sensor), and added as a delay. This allows the user to set the Arduino to trigger when the projectile has traveled a specific distance without the need for trial and error.

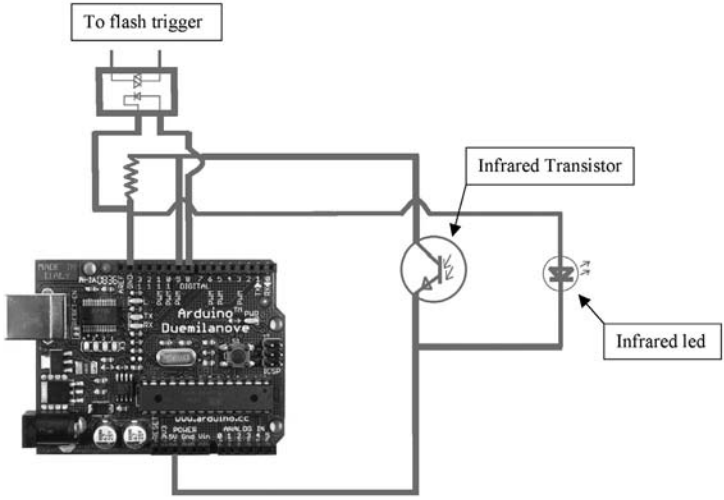

Figure 6: *Infrared projectile sensor. Two infrared sensors are used to determine a projectile's speed and set the delay for the flash trigger appropriately. (To decrease clutter in the diagram only one sensor is shown.)*

The infrared sensor uses infrared LEDs and transistors to detect the speed of a moving object and delay the flash an appropriate amount of time so that the moving object is frozen in a pre-determined spot. The coding and diagram (see Figure 4) are presented separately from the main code and diagram for ease of understanding, but they will

ultimately be combined together in the final product. The coding is as follows:

```
const int led = 8; // set led pin
const int first = 9; // set first infrared pin
const int second = 10; // set second infrared pin
const int delayPot = 3; // set delay potentiometer pin
int firstState = 0; //variable for state of first infrared sensor
int secondState = 0; // variable for state of second infrared sensor
int val; // variable used for delay
long startTime; // variable used for time
long elapsedTime; // variable used for time

void setup () {
  pinMode (led, OUTPUT); // establish led pin as output
  pinMode (first, INPUT); // establish first pin as input
  pinMode (second, INPUT); // establish second pin as output
}

Void loop () {
  val=analogRead(delayPot); // set val equal to reading from potenti-
ometer
  val=map(val, 0, 1023, 30, 0); map potentiometer values to 0-30
  firstState = digitalRead(first); // variable = reading from first sensor
  secondState = digitalRead(second); // variable = reading from sec-
ond sensor
    if (firstState == LOW) { // if first sensor is tripped
      startTime = millis(); // set startTime = time elapsed since start
    }
    if (secondState == LOW) { // when second sensor is tripped
      elapsedTime = millis() - startTime; // determine elapsed time
      delay (elapsedTime*val/2); // delay time for object to pass preset
dist
      digitalWrite(led, HIGH); // trigger flash
      delay(10); // delay
      digitalWrite(led, LOW); // turn flash off
    }
  }
```

The transition from an LED to a high-voltage flash unit is relatively simple, as shown through the depiction of an optoisolator in Figure 7 below.

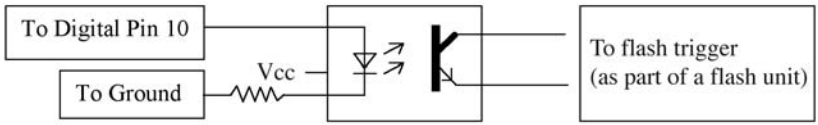

Figure 7: *Optoisolator*

The resistor used is 50 Ω. The optoisolator serves to separate the high voltage of the flash from the Arduino so that any spikes of voltage do not damage the microcontroller. When a relatively small voltage is sent from digital pin 10 through the optoisolator, a tiny LED lights up. This triggers a phototransistor, completing the circuit in the flash trigger. This method allows the two circuits to be completely isolated from one another but still trigger within a tiny fraction of a second [12].

The first optoisolator I used wasn't powerful enough to handle the voltage demands of the flash unit. This was determined by discharging different capacitors with the optoisolator. As long as the current wasn't extremely high (in which case the current limitations of the optoisolator would be exceeded), the unit would fail to discharge at voltages higher than 150 V. This is lower than both the written specifications (200 V) and the voltage of the flash (190 V). As a result I have had to replace the optoisolator with a more powerful unit that uses a triac in place of a transistor.

The original flash unit I used was salvaged from a disposable camera. The circuit consisted of an energy source (AA battery), a switch to trigger the charging of a capacitor, and a switch to complete the circuit, which discharges the capacitor and powers the flash. Menlo's photography teacher, Pete Zivkov, generously agreed to lend an external flash unit for use in the project. This made possible a shorter flash duration for clearer pictures.

4 Schematic

The final product not only adds the ability to switch between all three sensors on the fly, but includes an LCD display so that the user can view which sensor is currently selected and the variable delay time or distance (depending on the sensor). Two potentiometers are used, one for selecting the delay/distance (val1) and one for selecting the sensor to be used (val2). Depending upon the value of the second potentiometer (used to select sensor), the program skips to one of three sections, each of which houses the code not only for that sensor but for the variable delay and the LCD display.

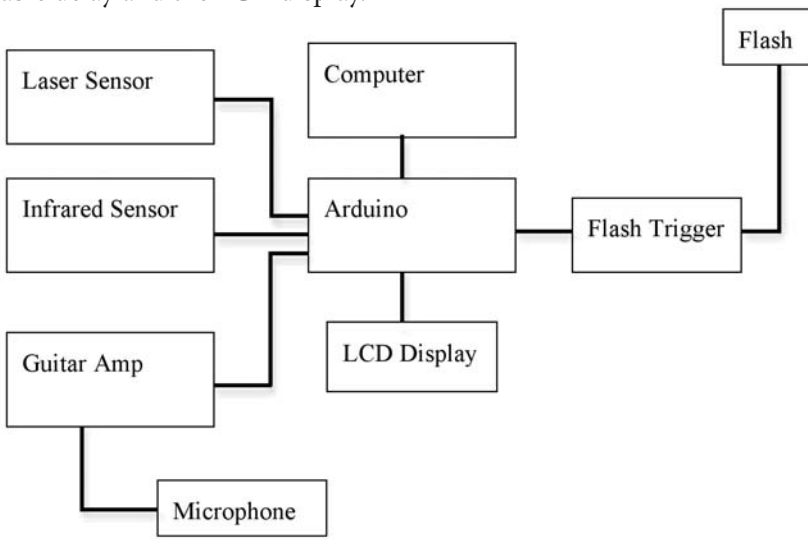

Figure 8: *Flowchart of final set-up.*

```
#include <LiquidCrystal.h> // include the library code:
const int laserSensor = 5;
const int laser = 9;
const int flash = 8;
const int soundSensor = 2;
const int infraredFirst = 10;
const int infraredSecond = 11;
int sensorVal = 0;
int soundVal = 0;
int firstVal = 0;
```

```
int secondVal = 0;
const int sensorPot = 4;
const int delayPot = 3;
int val;
int val2;
long startTime;
long elapsedTime;
LiquidCrystal lcd(7, 6, 5, 4, 3, 2); // initialize the library with the
numbers of the interface pins

void setup() {
 pinMode(flash, OUTPUT);
 pinMode(laser, OUTPUT);
 lcd.begin(16, 2); // set cursor for lcd
}

void loop() {
 val2=analogRead(sensorPot);
 val2=map(val2, 0, 1023, 3, 1);

 if (val2 == 1) {
  val=analogRead(delayPot);
  val=map(val, 0, 1023, 30, 0);
  lcd.setCursor(0,0);
  lcd.print("Sensor    Delay");
  lcd.setCursor(0,1);
  lcd.print("laser   ");
  lcd.setCursor(9, 1);
  lcd.print("  ");
  lcd.print(val);
  lcd.print(" ");
  sensorVal=analogRead(laserSensor);
  digitalWrite(laser, HIGH);
  if (sensorVal<300) {
   delay(val);
   digitalWrite(laser, LOW);
   digitalWrite(flash, HIGH);
   delay(10);
```

```
    digitalWrite(flash, LOW);
    delay(1000);
    digitalWrite(laser, HIGH);
    delay(5000);
   }
   else {
    digitalWrite(flash, LOW);
    digitalWrite(laser, HIGH);
   }
}

if (val2 == 2) {
  digitalWrite(laser, LOW);
  val=analogRead(delayPot);
  val=map(val, 0, 1023, 30, 0);
  lcd.setCursor(0,0);
  lcd.print("Sensor    Delay");
  lcd.setCursor(0,1);
  lcd.print("sound   ");
  lcd.setCursor(9, 1);
  lcd.print("  ");
  lcd.print(val);
  lcd.print(" ");
  soundVal=analogRead(soundSensor);
  if (soundVal > 200) {
    digitalWrite(flash, HIGH);
    delay(10);
    digitalWrite(flash, LOW);
  }
  else {
    digitalWrite(flash, LOW);
  }
}

if (val2 == 3) {
  val=analogRead(delayPot);
  val=map(val, 0, 1023, 30, 0);
  lcd.setCursor(0,0);
```

```
    lcd.print("Sensor  Dist(in)");
    lcd.setCursor(0,1);
    lcd.print("infrared");
    lcd.setCursor(9, 1);
    lcd.print(val);
    lcd.print("  ");
    firstVal = digitalRead(infraredFirst);
    secondVal = digitalRead(infraredSecond);
    if (firstVal == HIGH) {
      startTime = millis();
    }
    if (secondVal == HIGH) {
      elapsedTime = millis() - startTime;
      delay (elapsedTime);
      digitalWrite(flash, HIGH);
      delay(10);
      digitalWrite(flash, LOW);
    }
  }
}
```

5 Results

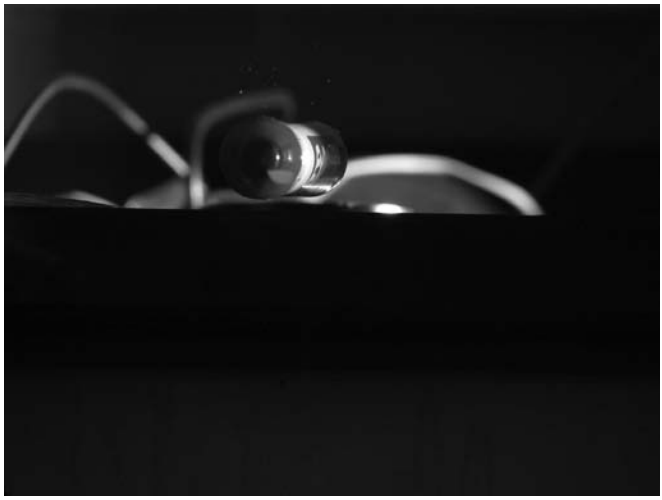

Figure 9: *Picture taken in the first testing of the laser tripwire.*

In the first testing of the laser tripwire, as seen in Figure 9, the laser was set slightly above the plane of the table (the beam itself was approximately 8 mm above the table) and the pen was dropped from a height of about 2 ft above the surface of the table. The camera was positioned slightly below the plane of the table. The flash triggered while the pen was still millimeters above the surface, producing a relatively clear image given the amount of ambient light that was impossible to eliminate from the room.

Figure 10: *Another picture taken in the first testing of the tripwire.*

In the picture in Figure 10, the falling object is a roll of electrical tape.

Figure 11: *Picture taken with laser tripwire.*

Figure 11 depicts a Christmas ornament immediately after its initial impact with the ground. (The object was thrown at the ground from a height of approximately 4 ft). The laser used for the flash trigger can be seen because it came on again before the shutter closed.

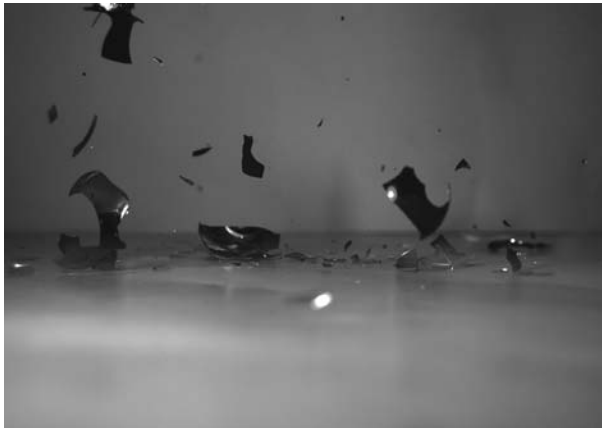

Figure 12: *Picture taken with laser tripwire; delay time adjusted.*

The photo in Figure 12 was taken with the same method as the shot in Figure 11, but the delay time was adjusted so the flash would occur later in the breaking process.

Figure 13: *First slap picture.*

In the photo in Figure 13, the sound sensor was used, although it was not sensitive enough to capture the sound immediately, and as a result the shot occurs too late. There is some ghosting as well: to the right of the face you can faintly see the subject before he was slapped. The room was not completely dark and so the film was lightly exposed before the triggering of the sound sensor.

Figure 14: *Another slap picture.*

In the photo in Figure 14, results are still not as hoped for. Image clarity would have benefited from the use of multiple flash units. Even so, a slight ripple can be seen in the subject's lips. Given the limited patience of the participants, this proved to be the best shot attainable.

Figure 15: *Picture taken with sound sensor.*

The photo in Figure 15 was triggered via sound: a metal weight (invisible behind the frontmost fragment) was dropped from above. There is slight ghosting in this image as well; the faint image of the ornament sitting on the ground is visible in the background. ●

6 Appendix A

Part Description:	Use:	Cost:	Place of Purchase:
Arduino	Control the interaction between triggers and flash & allow for delay	$30	Already purchased from Sparkfun
Flash unit from disposable camera	Expose camera film	$10	Already purchased from Walgreens
Camera Flash	Upgrade from disposable flash unit	n/a	Borrowed from Pete Zivkov
5 V laser pointer	Tripwire sensor	$5	Dealextreme.com
Phototransistor	Sense laser for tripwire sensor	n/a	ASR part cabinet
Microphone	Initial sound sensor	n/a	Already owned
Guitar amp	Amplify signal from microphone	n/a	Already owned
Small microphone	For use in final sound sensor	$2	Sparkfun
Amplification Circuit	For use in final sound sensor	$5-$10	Sparkfun
Optoisolator	Trigger flash unit	$1 each	Already purchased from Digikey
Various resistors	Circuitry	n/a	ASR part cabinet
Diode	Prevent negative voltage from entering Arduino input	n/a	ASR part cabinet

7 Citations

1. http://en.wikipedia.org/wiki/High_speed_photography
2. http://hacknmod.com/hack/top-40-arduino-projects-of-the-web/
3. http://www.Arduino.cc/
4. http://www.glacialwanderer.com/hobbyrobotics/?p=11
5. http://en.wikipedia.org/wiki/High_speed_camera
6. http://electronics.howstuffworks.com/high-speed-photography.htm
7. http://people.rit.edu/andpph/text-hs-history.html
8. http://en.wikipedia.org/wiki/Leyden_jar
9. http://en.wikipedia.org/wiki/High_speed_photography
10. http://web.mit.edu/invent/iow/edgerton.html
11. http://www.flickr.com/groups/highspeed/discuss/7205759130908461/
12. http://homepages.which.net/~paul.hills/SpeedControl/Optos.html

8 Acknowledgments

Mr. Pete Zivkov, for lending a Vivitar strobe for use in the project and allowing it to be modified in ways seen fit.

Dr. James Dann, for teaching the Applied Science Research class and allowing his students to pursue individual projects such as this during second semester, as well as helping with troubleshooting the countless problems that arose throughout the course of this project.

Jordan Jadallah and Charles Lewis, for allowing me to slap them in the face repeatedly in an attempt to capture the distortion in their faces mid-slap. You guys are troupers.