Joseph Borrego & Henry Daly
Advisor: Liang Cheng

Dueling Droids

Signature: ______________________________ Date: ______________

Problem Statement

Develop a quick, cheap boxing robot with extended life, and interface with Bluetooth to allow communication with an Android phone.

Project Overview & Background Research

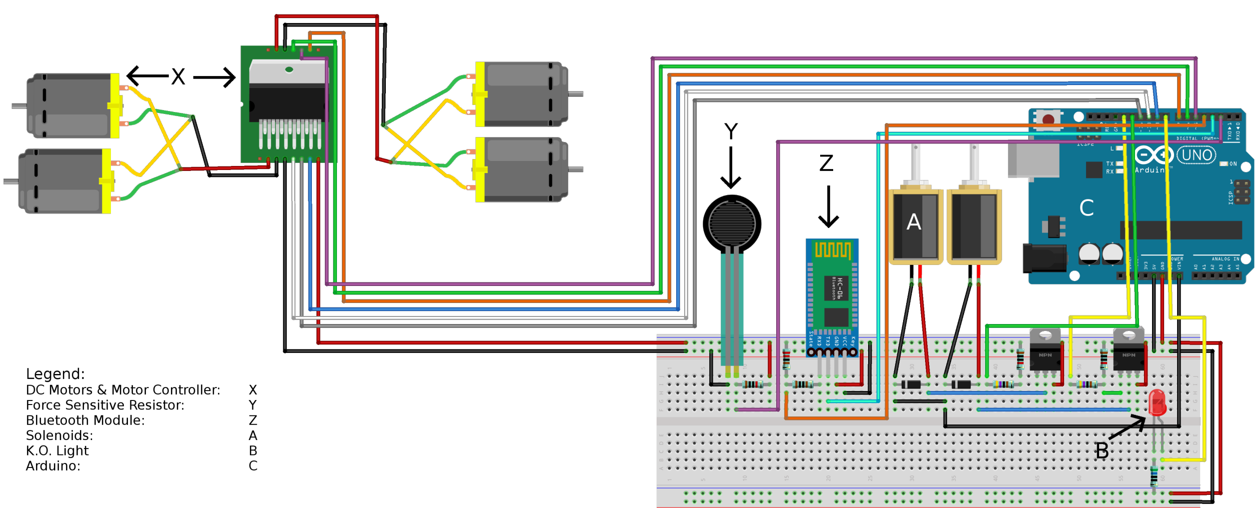
The goal of Dueling Droids is to improve our understanding of robotics through the design of boxing robots. We were inspired by the popular game Rock ‘Em Sock ‘Em Robots and the movie Real Steel. In Real Steel, people participate in a futuristic form of boxing in which boxers control robots through motion detection[1]. Boxers’ movements are tracked and replicated by the robots. Rock ‘Em Sock ‘Em, our other inspiration, was a game in which players control an android through two levers with the goal of knocking the opponent’s droid’s head off[2]. With this information in mind, we researched hardware and software products and designed a robot that would allow maximum mobility.

User & Technical Specifications

The primary specifications for our robots are responsiveness, extended battery life, and inexpensive. We wanted our bots to have a minimal (if not unnoticeable) delay in executing commands sent from the control App. For extended battery life, we chose twelve hours as our goal, for this should allow a full day of use. Finally, we wanted our bots to be affordable; we believe $100 is a reasonable amount users would spend. Furthermore, using a freely available phone App would reduce the cost of needing a controller. Having laid out the goals for the project, we began exploring options for meeting the goals.

In order to implement the above specifications, we decided to make our prototypes with an Arduino Uno. As a programmable microcontroller, the Arduino has a wide variety of uses. We decided to use it to actuate commands sent over Bluetooth from our Android control App. This allows users familiar with smartphones to apply their mobile gaming experience to the control of the robots; thus, the App was designed with a horizontal layout. The movement for the robot is on the left side while both fist movement is controlled on the right side. A figure can be seen in the in the Software Design section below. Upon the completion of generic system architecture, we initiated the design of the hardware to support the architecture.

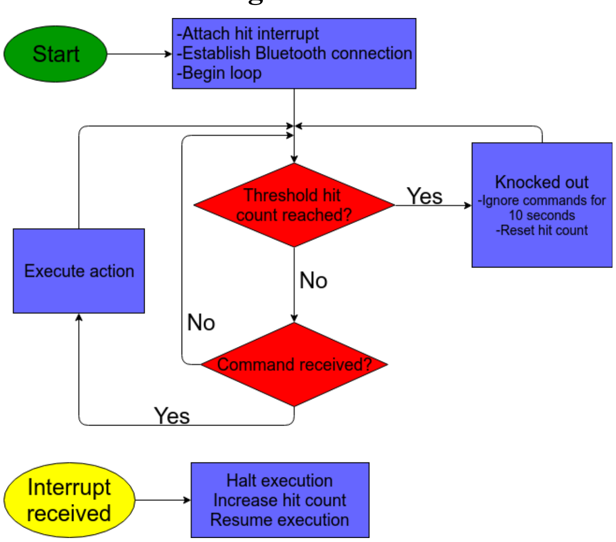
Hardware Design

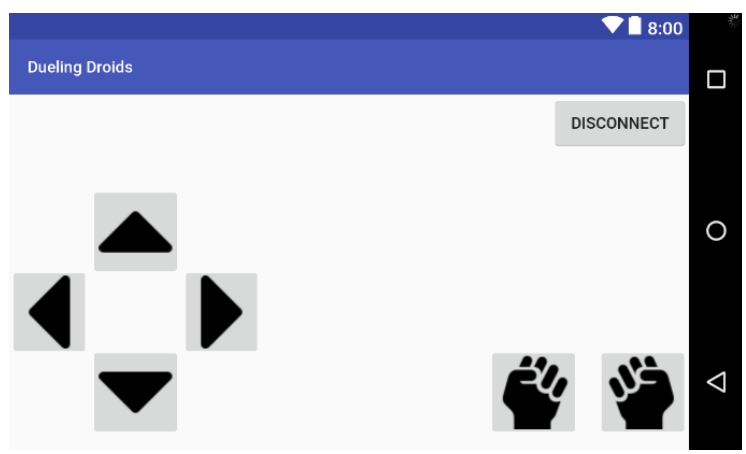


The figure above is a schematic layout of the internal circuitry of our boxing robots. Powered by an unseen 12V 6000mAh battery, the Arduino acts as a motherboard, providing the glue for the other components. In the top left one can see the four DC motors used to power the wheels; these DC motors can move in either direction, depending on the voltage difference across their leads. The L298N motor controller, which directs the DC motors, can also be seen providing the interface between the Arduino and DC motors. It is powered by the same 12V battery as the Arduino. Together these provide movement through the chassis. The next most left item is the Force Sensitive Resistor (FSR), which is tied to Vcc and a 10kΩ resistor, which is tied to ground. Punches are registered through the pressure registered on the FSR, which decreases resistance with increased pressure. Voltage level between the 10kΩ resistor and Ground is checked to register such hits; thus, this hardware implements a sense of touch for our bots.

Following along to the right is the Bluetooth HC-06 module. The module has four pins: Vcc, Ground, Transmit and Receive. Interestingly, the receive pin cannot work with the typical Vcc of five volts, and therefore has voltage cut through 1kΩ and 2.2kΩ resistors. Using this module, the Arduino can connect and communicate with our Android control app. The hardware used to implement the fists of our bots can be seen to the right of the HC-06 module. Powered by the same 12V battery as the Arduino and L298N, these large push-pull solenoids require a number of interesting solutions to function properly. Upon discharge of electricity, a spike of current exits the solenoids, so the 1N4001 diode between the positive and negative leads to the solenoids exists to prevent MOSFET destruction. The MOSFETs serve as power switches for the solenoids, and voltage applied to the gate is split between 1kΩ & 10kΩ resistors. The drain of the MOSFET connects to the negative lead of the solenoid while the source connects to ground. Finally, the LED on the bottom right represents the display of a robot being "knocked out," and it is wired with a 560Ω resistor.

Software Design





This project requires both the robot's internal software as well as the controller App program. The embedded software was developed in Arduino's language (a subset of C/C++). Source code is attached in the appendix, and a flow chart can be seen above on the left. The software is rather straightforward; after attaching an interrupt for processing hits and establishing a Bluetooth connection with a controller, the robot enters a loop. At the beginning of the loop the bot checks if it has reached the hit count threshold, and if so enters the knocked out state for ten seconds. In this state the LED blinks to notify that the droid is knocked out, and all commands sent are ignored. Otherwise, it checks if any commands have been received and executes the command if it has received one. Upon a hit, the robot execution is halted to increase the hit count.

The control App's software was implemented for Android in Java, and source code is not included. A screenshot of the control screen can be seen above on the right. On opening the App, one finds a list of Bluetooth devices wherein one can choose a droid and initiate a connection. After the connection is established, the control screen opens. On the left side are the movement buttons, where the droid can be controlled similarly to an RC car. On the right are the two punch buttons and a disconnect button on the top for ending a session. The embedded software and the control App work well together and allow seamless control of the droids.

Ethical Consequences

Our project does not breach any ethical rules. Small boxing robots do not present any more violence than the average user will see on the news or in video games. No aspect of our project presents a threat to a user. We also find no ethical consequences. Our robots are much less dangerous than those presented in one of our inspirations, Real Steel; thus, we believe our project to be well within all ethical boundaries set in the field of robotics.

Intellectual Property Considerations

Similar projects were discovered while searching for patents, but enough differences exist in our project that we do not expect patent infringement. The two patents found similar to our project are the Fighter Robot System (US 8,758,077 B2) and the Fighting Toy (US 8,764,510 B1). The Fighter Robot System describes two robots being supplied by the same power supply and connected to a rotary system such that the robots will always face each other. This patent covers the fighting robots and rotary system. The concept of the robot fighting system is similar to our project; however, the difference lies in the implementation. The primary differences include connected robots, robots implemented with legs, a central power supply for both robots, and a rotary member attaching the two robots. Our design includes none of these; therefore, our design should not infringe on this patent. The Fighting Toy patent details a robot fighting system relying on a lever with a shaft for robot handling. The robot, connected physically to the handle, directly responds to handle movement. Electrical supply to the robot powers a motor which actuates the movement of the handle. This seems to be a more advanced version of Rock Em' Sock Em', and again the differences in our project and this patent stem from the controls, robot design, and power supply. The size of the ring and robot are limited by the handle and controller. While robots in this system can move along the X & Y axes, they are limited by the bounds of the boxing ring. The main difference between our design and this patent is the limit of size and mobility. Our design will not infringe on this patent due to an unlimited boxing ring size as well as our controller not being directly connected to the robots.

Alternate Design

At the inception of our project we considered an alternate design in which we would more closely mimic Real Steel and control our robots with Wii controller nun chucks. Using the nun chuck we could synchronize our punches with the robots, and we believed that after designing the initial robot we'd migrate from an Xbox 360 controller to the nun chuck. However, facing a multitude of difficulties along the way, we were led to the conclusion of scrapping the handheld controller for Bluetooth communication with the Android App.

Cost Analysis and Bill of Material

As can be seen below, the majority of included parts in our project are not expensive. The total spent on the project is $418.12. While this may seem like a large amount, it is important that we note the actual cost of the robot prototypes is much less ($125 per robot). A variety of these parts were faulty (such as DC motor, solenoid, L298N motor controller) and had to be ordered again. A few parts, such as the USB host shield, was not be used in our project (the loss of IR communication renders it obsolete), and thus can be considered a sunk cost due to experimentation. Mass production of these bots would significantly reduce cost per robot. The Android App is free, and thus would be cheaper than a competitor's physical controller. Though the cost per robot is greater than the initial goal of $100, we still believe the mass production and future revisions will bring the price down below the goal.

Distributor	part #	manufacturer rep	manufacturer	man part #	part	price	quantity
Amazon	N/A	Amazon	Amazon	N/A	HC-06 Bluetooth Wireless Module	8.99	2
Copy	B00LUJYN55	" "	3DMakerWorld	A000004	Arduino USB Host Shield	39.45	2
" "	46170000	" "	Talentcell	FBA_YB1206000	Rechargeable 6000mAh Li-ion battery	29.99	2
" "	B00DX6ZUBM	" "	Valley Ent.	DC-MM-24-18	M to M plug DC Power Adapter Cable	6.68	2
" "	B06XFZHN2	" "	Aloway	LYSB06XFZHN2-CMPTRACCS	2pcs plastic tire wheel	8.99	3
" "	N/A	" "	LGDehome	4333FRAFALCR1	L298n H- bridge motor drive controller	11.99	1
" "	B014KMHSW6	" "	Qunqi C	MK-050	L298n motor drive controller	6.89	3
" "	B072L1XMUR	" "	RexQuails	N/A	120pcs Multicolored Dupont Wire Kit	6.66	1
" "	B0150LWDL	" "	Wallyshine	N/A	12 pcs 12V 23A Battery Spring clip	7.87	1
" "	B00DNGTUAE	" "	mybatterysupplier	KL-JTW0-BZD9	GP 23AE 12V Batteries	5.9	1
Adafruit	N/A	Adafruit	Adafruit	413	Large push pull solenoid	14.95	4
N/A	N/A	N/A	Wilbur Powerhouse	n/A	3D printed top	4.3	4
N/A	N/A	N/A	Wilbur Powerhouse	n/A	3D printed body	18.8	2
n/A	n/A	n/A	Wilbur Powerhouse	n/A	3D printed arms	3.7	2
n/A	B00TGSPV6	Velcro	Velcro	90075	velcro	2.98	3
Amazon	B018B2EEFK	Amazon	Transformer	B7064	Transformer head	3.46	2
Amazon	B00B887DBC	Amazon	InkBright	1645	Force Sensitive resistor	14.99	2

Government Regulations Impact Study

Two government regulations affecting our project deal with transmission signals (found in Title 47), but our project does not break either of these regulations. The first regulation, Title 47 chapter 1 Subchapter A Part 15 §15.15, covers general technical requirements for a radio frequency device. It states that the radiator should be constructed with good engineering design and manufacturing practice. It also details that the parties responsible for equipment compliance should note that the limits specified in this part will not prevent harmful interference under all circumstances. Both requirements are satisfied in our project. The second regulation, Title 47 chapter 1 Subchapter A Part 15 §15.33, deals with the frequency range of radiated measurements. Given that our Bluetooth module operates at around 2.5GHz, we do not believe our project breaks this regulation.

Health and Safety Study

Considering some of the small parts used in this project as well as the fact that some may consider our boxing robots as toys, under the Consumer Product Safety Act of 2008 (CPSIA) Federal Hazardous Substances Act a choking hazard warning could be added to our product. If marketed as a toy, the target demographic would be children roughly ten years of age and above.

Targeting this age group requires adding permanent distinguishing marks with scope for tracking labels to enhance recall effectiveness according to the CPSIA.

Environmental Impact

No modification of our design was made with regard to the environment. Furthermore, our project does not affect the environment in any way post construction, as it is battery charged and gives off no emissions. Thus, our project has little impact on the environment.

Project Progress

Task Name	Duration	Start	Finish	
Chassis: Construction	-	-	-	
Chassis: Software	-	-	-	
3D Print Arms	-	-	-	
Arms: Software	-	-		
3D Print Body & Attach Arms	14d	22-Jan	5-Feb	
Bluetooth Communication	21d	22-Jan	12-Feb	
3D Print Head & Fists	14d	5-Feb	19-Feb	
FSR: Software	21d	12-Feb	5-Mar	
K.O. Functionality	14d	5-Mar	26-Mar	(Spring Break)
Construct Second Robot	21d	19-Feb	19-Mar	(Spring Break)
Enhancements	-	26-Mar	Finish	(Discuss options)

Overall, the project proceeded on track and was completed on time. However, a functional end product meeting all specifications does not mean difficulty was not encountered. Faulty parts, noise interference, and communication issues desynchronized our project with original plans. We underestimated the time it would take to accomplish tasks; the inherent difficulties arising from dealing with unfamiliar hardware and latencies from troubleshooting slowed progress.

The first semester primarily concerned the base chassis, the solenoids, and communication. The initial chassis was constructed in a week, needing only a motor controller to be fully operational. After acquiring an L298N and attaching it, only the wheels on the left side of the chassis moved properly. Initially we checked the software, believing this to be the issue; however, this proved unhelpful, so next the motor controller was singled out and investigated. This investigation also proved unhelpful, and eventually we began testing every component. One of the motors on the right side turned out to be faulty, and fixing this resolved our initial delay.

Our second main delay spawned from the use of a USB host shield to communicate with Xbox 360 and Playstation 4 controllers. Connections could be made, but sending commands disconnected the controllers. Once again we assumed this to be the result of faulty code, but were unsuccessful at finding the issue. Eventually, we realized that our problem was noise interference between the host shield pin-outs and our actuators. Thus, we shifted gears and decided to implement Bluetooth communication through an Android App the following semester. The Gantt chart created at the midway point of the project can be seen above. It was expected that the initial goals for the project would be completed by March 26th; however, this was not to be the case. Below one can see the revised Gantt chart for the second half of the semester.

Arms & Body: Design & 3D Printing	25d	22-Jan	16-Feb
Bluetooth Communication	16d	22-Jan	14-Feb
Head & Fist Implementation	14d	16-Feb	2-Mar
Control App Development	14d	14-Feb	28-Feb
Force Sensitive Resistor: Software	7d	28-Feb	7-Mar
K.O. Functionality	7d	19-Mar	26-Mar
Construct Second Robot	21d	19-Mar	9-Apr
Body & Circuitry Integration	10d	9-Apr	19-Apr
Final Testing, Enhancements	8d	19-Apr	27-Apr

The second semester again provided frustrations to plans; however, this was to a much lesser degree. The implementation of the FSR and Android App went smoothly. 3D printing proved to take a great deal longer than anticipated due to unfamiliarity with modeling software as well as print time. Despite some issues, the project continued along relatively on progress, and by the end of the semester all original goals were completed. The final product was tested, and the robot passed initial specifications with flying colors.

Experimental Results

After completing the construction of the first prototype, tests were conducted to verify that specifications were met. First, battery life was calculated through measuring components' current draw. The top three contributors were the DC motors, the motor controller, and the Arduino itself. Interestingly, though the solenoids require a large current draw to discharge, they do not fire frequently and therefore have a low average current draw. Total current draw was roughly 265mA; therefore, with a 6000mAh battery the droids have a battery life of over 22 hours. The solenoids claim 10N of force upon discharge, and tests proved the claim's validity. Testing speed, we measured the time it took the droid to drive 25 meters and found it to be under 36 seconds. General use of the robot proved that delay between sending a command and the robot executing said command was unnoticeable.

Conclusion

Through this project we have greatly improved our technical as well as executive skills. By working with new hardware, we learned the importance of testing parts comprehensively to assure successful integration. Executive skills were improved by describing the project to both peers and judges as well as through progress tracking and management. Ultimately, Dueling Droids provided a necessary challenge; integrating much of the theoretical knowledge of class and working on a schedule prepared us for industry. Excluding cost, all original specifications were met or exceeded, yet even so improvements were considered. Future plans include converting the Arduino and breadboard implementation into a PCB, enhancing the UI of the App, and adding programmable combo moves to the App. Through these improvements, cost can be decreased and the product as a whole will improve.

Appendix

Arduino Source Code:

```
/* ----------------------------------------------------------------------------------------------
 * DUELING DROIDS
 * Partners: Joseph Borrego, Henry Daly
 * ----------------------------------------------------------------------------------------------
 */

/* INCLUDES */
#include <SoftwareSerial.h>

/* DEFINES */
// The following are the pins for the solenoids
#define   armR    13
#define   armL    12
// The following are pins for the motor controller
#define   enA     11     // blue    - ena
#define   in1     10     // purple  - in1
#define   in2     9      // grey    - in2
#define   in3     7      // white   - in3
#define   in4     6      // black   - in4
#define   enB     5      // brown   - enb
// The following are pins for tracking hits and displaying K.O.
#define   FSR_PIN   2 // actual input pin
#define   FSR_INTPT 0 // the "interrupt" pin
#define   DEAD_LITE 8
volatile  int num_hits = 0;

/* Uncomment to have debug information printed to Serial Monitor */
//#define DEBUG

/* BLUETOOTH MODULE */
SoftwareSerial BT(3,4); // pins 3 & 4 used for Bluetooth communication
char  received_command;

/* The following is our interrupt function */
void punched() {
  num_hits++;
}

/* The following are our actuator functions */
/// display K.O.
void knocked_out() {
  num_hits = 0;
  for(int i = 0; i < 10; i++) {
    digitalWrite(DEAD_LITE, HIGH);
    delay(500);
    digitalWrite(DEAD_LITE, LOW);
    delay(500);
```

```
  }
}

/// move bot backward
void all_backward() {
#ifdef DEBUG
  Serial.println("All Backward!");
#endif
    // Left Backward
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
    // Right Backward
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
  delay(200);
}

/// move bot forward
void all_forward() {
#ifdef DEBUG
  Serial.println("All Forward!");
#endif
    // Left Forward
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
    // Right Forward
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  delay(200);
}

/// turn bot left
void turn_left() {
#ifdef DEBUG
  Serial.println("Turn Left!");
#endif
    // Right Forward
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  delay(200);
}

/// turn bot right
void turn_right() {
#ifdef DEBUG
  Serial.println("Turn Right!");
#endif
    // Left Forward
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  delay(200);
```

```
}

/// stop bot movement
void center() {
#ifdef DEBUG
  Serial.println("Stop!");
#endif

    // Left Stop
  digitalWrite(in1, HIGH);
  digitalWrite(in2, HIGH);
    // Right Stop
  digitalWrite(in3, HIGH);
  digitalWrite(in4, HIGH);
  delay(200);
}

/// throw right hook
void right_punch() {
#ifdef DEBUG
  Serial.println("Right Punch!");
#endif

  digitalWrite(armR, HIGH);
  delay(500);
  digitalWrite(armR, LOW);
}

/// throw left hook
void left_punch() {
#ifdef DEBUG
  Serial.println("Left Punch!");
#endif

  digitalWrite(armL, HIGH);
  delay(500);
  digitalWrite(armL, LOW);
}

void setup() {
#ifdef DEBUG
  Serial.begin(9600);
  Serial.print("DUELING DROIDS\n\tInitializing......\n");
#endif

  /* Enable pin inputs */
  pinMode(FSR_PIN, INPUT);

  /* Enable pin outputs */
  pinMode(DEAD_LITE, OUTPUT);
  pinMode(enA, OUTPUT);
```

```
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
  pinMode(enB, OUTPUT);
  pinMode(armR, OUTPUT);
  pinMode(armL, OUTPUT);

  /* Initialize default pin outputs */
  digitalWrite(enA, HIGH);
  digitalWrite(enB, HIGH);
  digitalWrite(armR, LOW);
  digitalWrite(armL, LOW);
  digitalWrite(DEAD_LITE, LOW);

  /* Initialize Bluetooth communication baud rate */
  BT.begin(9600);

  /* Attach interrupt */
  attachInterrupt(FSR_INTPT, punched, RISING);

#ifdef DEBUG
  Serial.print("\tInitialization Complete.\n");
#endif
}

void loop() {
  if(num_hits >= 3) {
    knocked_out();
    while(BT.available()){BT.read();}
  }
  if(BT.available()) {
    received_command = (BT.read());

#ifdef DEBUG
    Serial.print("Received command: '");
    Serial.print(received_command);
    Serial.print("'\n");
#endif

    switch (received_command) {
      case 'F': all_forward();    break;
      case 'B': all_backward();   break;
      case 'R': turn_right();     break;
      case 'L': turn_left();      break;
      case 'P': right_punch();    break;
      case 'O': left_punch();     break;
      case 'S':
      default:  center();
    }
  }
```

```
  delay(50);
}
```

Sources

"Real Steel Plot." *IMDb,* IMDB.com, www.imdb.com/title/tt0433035/plotsummary.

Siegel, Alan. "Machines Don't Fall Down Dead: How Rock'em Sock'em Robots Came to Be." *Deadspin*, Deadspin.com, 20 Dec, 2011, deadspin.com/5869804/its-ok-when-machines-get-hurt-how-rockem-sockem-robots-came-to-be.