Final Report

Stephen Walden

Joshua Castor

Bennett Maciorowski

Vig Tarush

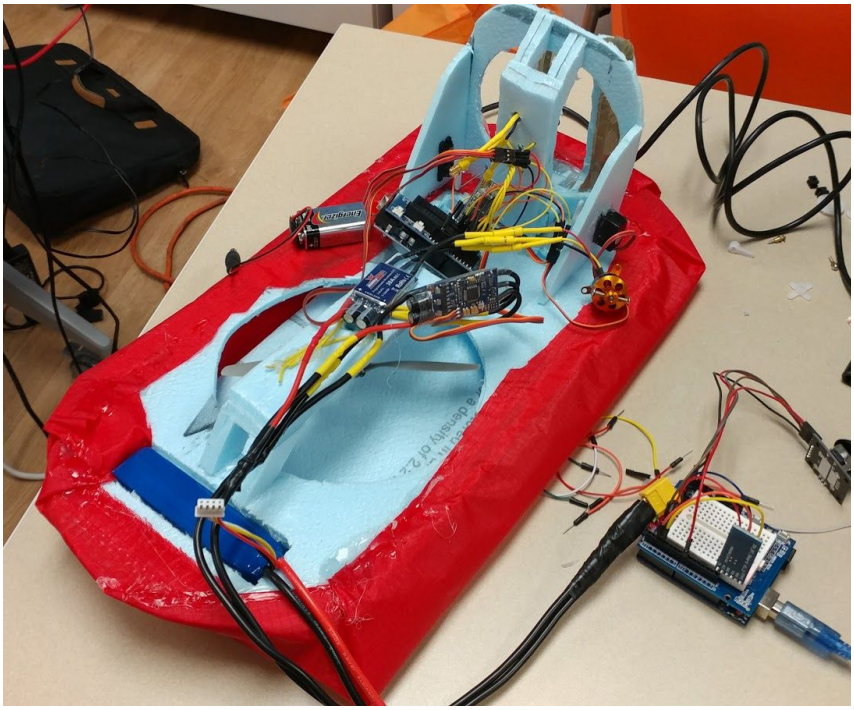
UIC Undergraduates

CS 362

Fall 2017

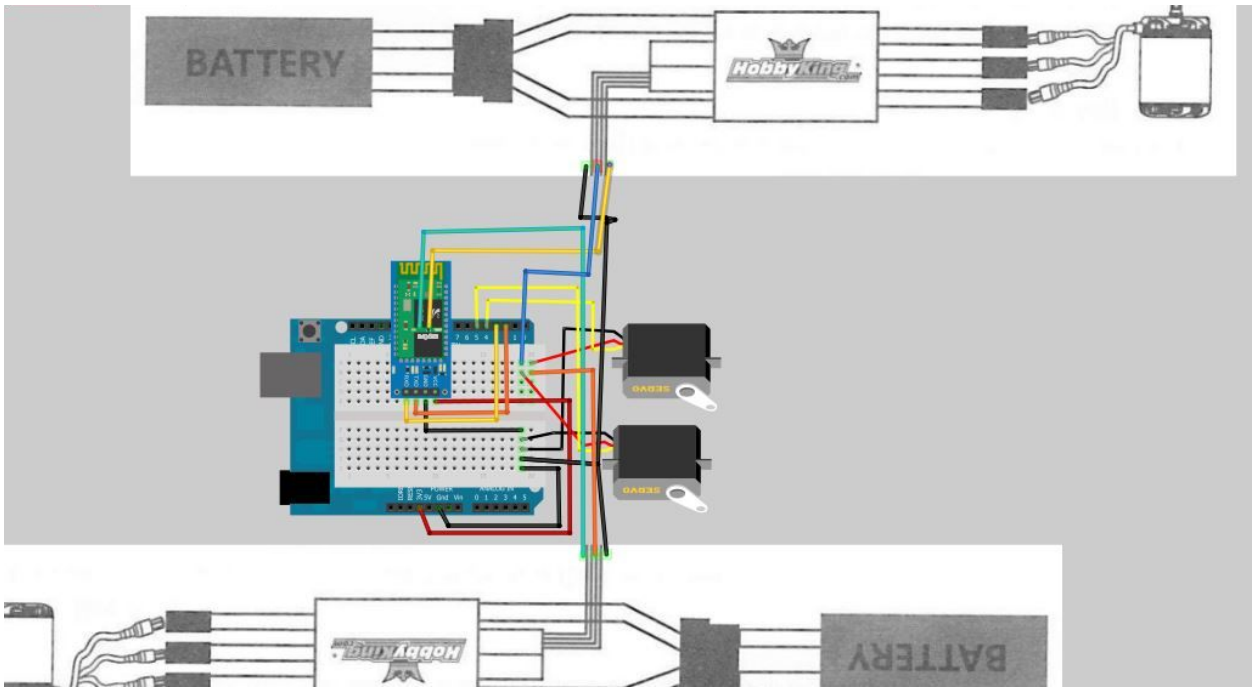
Project Description

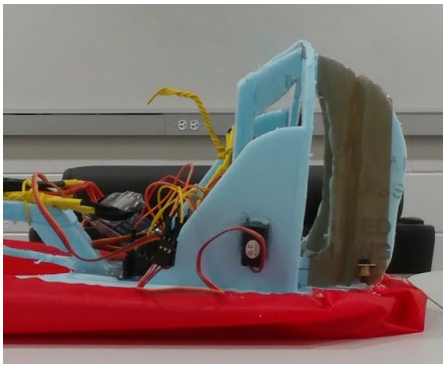
The original theme/concept for our project was to build two hovercrafts. Each hovercraft set will have a hovercraft and a controller for that hovercraft. Using two sets of hovercrafts, a pair of people can play a game along a surface, where each hovercraft will try shooting each other using infrared beams and the last one standing wins. Because of the issues with inaccurate hardware lists on our kits, we were unable to pull together the infrared transmitters. We thought about dissassembly of our Infrared remotes that were included in our kit, but due to time constraints we decided to scrap the second hovercraft, and instead replaced it with a pair of communicating lighthouses. This was done in order to allow to complete the objectives of our class prompt within the original time constraint that was given.



Each hovercraft contains two separate engines which are controlled by HobbyKing 30A Electronic Speed Controller (ESC); one engine would be on the bottom to inflate the skirt and give the hovercraft life, and one on the back of the craft to provide thrust. To get the ESC's to work, they first have to be programmed. The ESC's utilize a sound style menu, and as such the

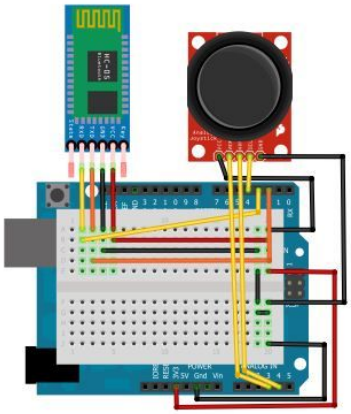
PDF for the ESC's must be referenced to program them The process takes about 30 minutes from scratch.



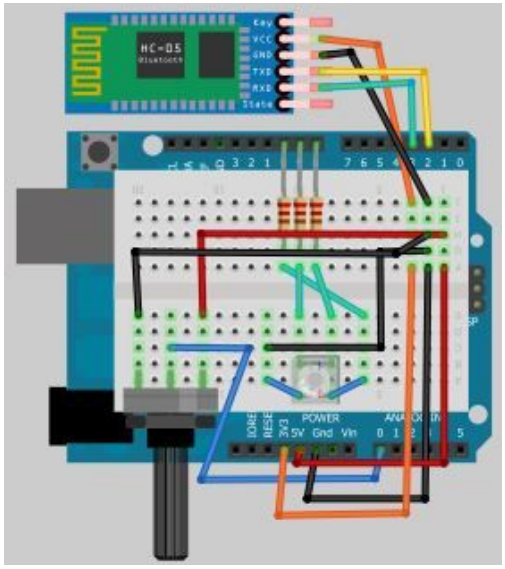


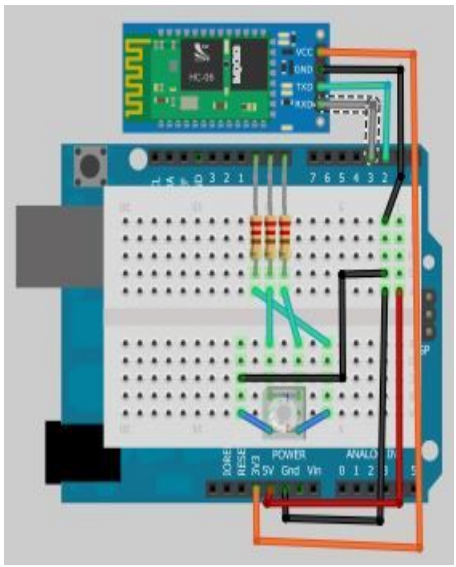
There are also two servos which draw their external power from the ESC's 5V power which is normally the middle wire (due to the way that most Radio-Control receivers are set up). These servos are physically mirrored on the hovercraft. The push-rods are attached to the servo arms by a z-style bend. The rudders are attached to the push-rods through usage of control horns, which in our case was an expired credit card that was cut to proper size. The rods link through special quick connectors utilized by the mass majority of Radio-Control enthusiasts. The purpose of the servos is to control the direction of the rear engine's thrust.

To control the servos and motors, there is an Arduino on the hovercraft. For wireless communication, the hovercraft will use an HC-06 bluetooth module configured in “servant” mode that is paired up to a controller’s HC-05 bluetooth module. To power all the components, there is a model airplane battery for the motors and servos and a 9-volt battery for the Arduino board on the hovercraft. For the construction of the hovercraft, we used ⅛” insulation foam for the structure of the craft and nylon fabric for the skirt.



The controller of the hovercraft isn’t too fancy. There’s an Arduino board for registering all the inputs and outputs. Connected to the Arduino is a single analog stick, an HC-05 bluetooth module, and a 9-volt battery. The analog stick provides the directional input as well as the throttle input for the hovercraft. The HC-05 bluetooth module acts as a transmitter for all the controls the user inputs, meaning that the HC-05 module will be configured in “master” mode. Finally, the 9-volt battery powers the Arduino board.





LightHouse Master LightHouse Servant

The two lighthouses utilize bluetooth communication, in the same manner as the hovercraft and controller. Each lighthouse has an RGB LED that can be configured to either output a specific color, or randomize the color that is output. The main lighthouse has a potentiometer to increase and decrease the pulse modulation; turning the potentiometer to the max will increase the delay between LED state changes, and turning the potentiometer to the minimum will decrease the delay between LED state changes (i.e. very quick flashing). The main lighthouse uses the bluetooth module to send information to the servant lighthouse based on whether its LED is on or off. The servant lighthouse receives this information via its own bluetooth module, then decodes it. It is set up so that the lights will alternate between the two lighthouses; if the main Lighthouse is not emitting light, then the Servant Lighthouse emits light. In this way both lighthouses can be controlled by one input.

Lessons Learned

It goes without saying that it is not quite as simple to build a hovercraft as it may seem. One of the largest physical issues that presented itself throughout the three weeks it took to figure out the physical positions of the hardware was the center of gravity. Originally the plan was to have the batter in the front of the hovercraft and the arduino in the rear. This eventually morphed into a design with the battery in between the blades, with the arduino positioned towards the front.

What we learned about the physical build was that the center of gravity doesn't matter quite as much as the overall weight ratios of the components (such as not needing such an

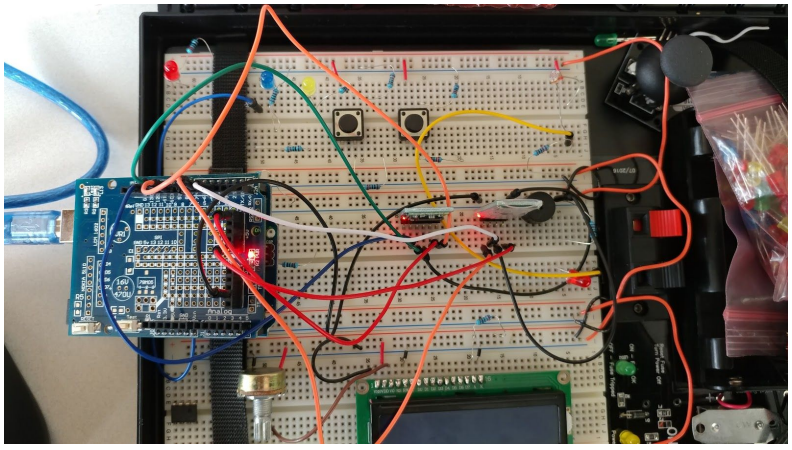
oversized battery for the tiny engines. In the final design, the battery was moved back to the front of the hovercraft so that the overall length of the hovercraft could be shortened. Given enough time, we would have designed a more robust chassis, where the battery would reside in between the engines, and the arduino would rest on top of the lift motor mount. The engines also played a large role in the hovercraft build. In fact, it may have worked better if we had utilized a slightly larger lift motor. The fact that we were using the lift motor at a constant 80% of it's full speed was not optimal, as these engines are only really supposed to be run at a consistent 50-60% total speed.

In relation to the motors, Stephen and Josh struggled to get the motors to engage in any consistent manner. One of the struggles that Josh had when he attempted tackling the motors the first time was getting the motors to understand the commands that were being given. Stephen speculates, but hasn't confirmed what the original problem was. One thing that Stephen learned was that the ESC's don't generally react to PWM values that are any less than 700. In other words, the servo.writemicrosecond(var-here) command doesn't work for the ESC's without a value less than 700. Stephen also, after several google searches, found and referenced the HobbyKing ESC manual. Although not for the 30amp ESC's, Stephen did remember that most of the HobbyKing ESC's are programmed pretty much the same way. He had also searched a found an easy way to initially program the ESC's.

But the ESC's were more tricky than simple programming. One of the trickiest problems about the code was the fact that the initial PWM start value for the ESC's are different. One started at 700, and the other at 750. It took several dozen motor tests for Stephen to figure that issue out though. Once he managed to fix that particular problem the physical tests, such as hovering and servo control, could begin.

Then there are the servos. Josh had initially wired the single servo to the arduino. Which worked, but only temporarily. The issue started happening after the first few tests, and it had to do with how the servo and arduino interacted. It is not common knowledge, however servos tend to pull 5v power in order to move. The problem with that is when 2 servo's are wired to the same arduino, as soon as noise is encountered, and the servos jump to life, the arduino would immediately shut down. After looking into it some more, Stephen realized what was going on and tried coming up with a solution. Know from past experience Stephen understood how ESC's worked and was able to figure out how to wire the external power from the ESC to the servo. After that the servos no longer killed the arduino every time they encountered noise.

We also had a few struggles with the bluetooth modules throughout the project. The first step was to find a way to pair them together. After about a week's worth of research, Josh was eventually able to have them pair automatically after looking through multiple tutorials (described in the resource section below). The circuit setup just for pairing an HC-06 module (servant) and an HC-05 module (master) is shown to the right; there is an Arduino on the left, an HC-05 module to its immediate right, and an HC-06 module to the right of the HC-05 module.



After figuring out how to have them connect automatically, the next challenge we encountered was sending data between the modules. We had two pairs of HC-05 and HC-06 modules, but one pair was a bit buggy. By that, we mean the full alphabet couldn't be sent

between the pair; some letters only work if they're lowercase, some letters only work if they're uppercase, and some numbers were missing. This was a major issue, as we needed to send numerical data from the hovercraft controller to the hovercraft for direction and throttle information. To get around this limitation, Josh encoded the numbers based on the index of the number on the controller, and decoded those numbers on the hovercraft, as shown below.

```
//excerpt from Controller.ino
const char scrambledAlphabet[26] = "abCdEFghIJkLmnOpQRsTuvWXyz";

//encode string from regular numbers to scrambled alphabet
String encodeNumbers(String input){
  String s = "";
  const int len = input.length();
  for(int i = 0; i < len; ++i){
      char currentChar = input.charAt(i);

      //replace number with character at index
      if(isDigit(currentChar)){
            currentChar = scrambledAlphabet[(int)currentChar - '0'];
      }

      Serial.print(String(input.charAt(i)) + "->" + currentChar + "| ");

      s += currentChar;
  }
  return s;
}
```

```
//excerpt from Hovercraft.ino
const String numberAlphabet = "abCdEFghIJ";

//decode string from scrambled alphabet to regular numbers
String decodeNumbers(String input){
  String s = "";
  const int len = input.length();
  for(int i = 0; i < len; ++i){
      char currentChar = input.charAt(i);
      int index = numberAlphabet.indexOf(currentChar);
      //  convert input to numbers
      if(index > -1){
            currentChar = (char)index + '0';
```

```
      }
      s += currentChar;
  }
  return s;
}
```

By using this encoding/decoding format for the bluetooth communication, we were able to successfully send data from the controller to the hovercraft without issue.

Resources

Online Resources

- How to connect and use Analog Joystick with Arduino
    - Referenced how to connect analog stick and read it with Arduino code
    - Knowledge was used to connect the analog stick to the Hovercraft Controller
- Arduino Reference - map()
    - Referenced how the map function works
    - Knowledge was used for scaling inputs from the analog stick and from the bluetooth module
- Programming the ESC - ESC Manual
    - Took programming code for the ESC, and utilized it to program the HobbyKing 30 amp BEC ESC's.
    - Used the PDF to reference the menu tones for the ESC programming.
- How to interface Servo motor with PIC (specifically this schematic)
    - Referenced how to connect a servo to a circuit
    - Used to connect rudder servos and ESCs to Arduino.

- Arduino - Knob
    - Referenced how to communicate with servo through Arduino code
    - Used to program rudder servos to move based on analog input
- ESC Programming on Arduino (Hobbyking ESC)
    - Referenced how to send signals to ESCs
    - Used to send signals to ESCs
- Fritzing Components: hc06 and HC05 (from this repo)
    - Used to create schematics for the build
- Forum for reference and inspiration.
    - Used for reference and inspiration
- Sending a series of commands to the HC-05 Module via .ino
    - Used to figure out how to send commands to the HC-05 module via .ino for pairing
- Sending a series of commands to the HC-06 module via .ino
    - Used to figure out how to send commands to the HC-06 module via .ino for pairing
- Commands to pair HC-06 and HC-05 together
    - Used in conjunction with the links to send commands to the modules via .ino in order to help automate the pairing process
- HC06 Connection Tutorial and HC05 Connection Tutorial
    - Used to figure out usage, communication, and circuit connections for the two bluetooth modules

Parts List

- C2222 Micro brushless Outrunner 2850kv (15g)
- Dubro E/Z Connector Heavy Duty 4-40 (12 Pack)
- HobbyKing 30AMP-BEC ESC's 2x
- HobbyKing 928BB Servo 2.0kg / 0.13sec / 9g 2x
- DSD TECH HC-05 Bluetooth Serial Pass-through Module Wireless Serial Communication with Button for Arduino
- KEDSUM Arduino HC-06 Serial Bluetooth Slave Wireless RF Transceiver Module 4 Pin with DuPont Cable
- EMMAKITES 60"x36"(WxL) 40D Ripstop Nylon Fabric by the yard Pre-cut PU Coating Light Waterproof UV Resistant
- Expired Credit Card (in lieu of control horns)
- Glue Gun Ammunition for Glue Gun
- 1/8th inch insulation foam
- 1mm Music Wire (Pushrods)
- 4 Arduinos
- Analog Joystick - Arduino
- RGB LED’s (resistors if not a module light)
- Potentiometer
- Various gauge wires (both control and harness for hovercraft)
- RioRand XT60 Drone Connectors
- Vktech Gold-plated Bullet Connector
- Electronic Solder & flux