Advanced Manufacturing Academy - 2016

ROBOTICS – 4
More Outputs

College of Engineering and Technology

East Carolina University

Improved Output Controls

- Tried Simple digital controls
- Multiple devices
- Controlled speed of lights
- Did anybody get the light to DIM?
  - How would you control the brightness?
  - Pulse Width Modulation (PWM)
- Introduce PWM
- The Servo

What is PWM?

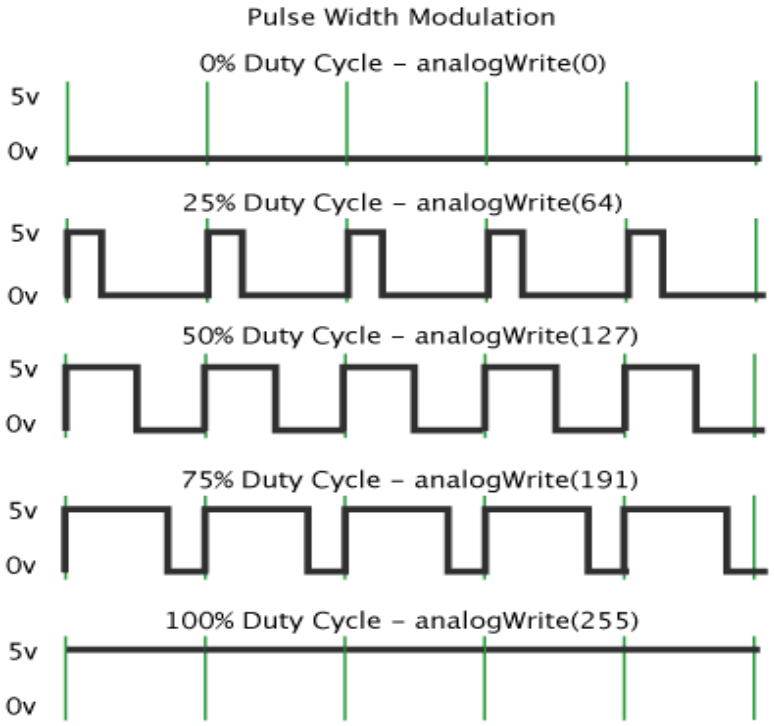
- Pulse Width Modulation
- Digital way to vary power to device

Digital control is used to create a square wave, a signal switched between on and off. This on-off pattern can simulate voltages in between full on (5 Volts) and off (0 Volts) by changing the portion of the time the signal spends on versus the time that the signal spends off. The duration of "on time" is called the pulse width. To get varying analog values, you change, or modulate, that pulse width. If you repeat this on-off pattern fast enough with an LED for example, the result is as if the signal is a steady voltage between 0 and 5v controlling the brightness of the LED.

http://www.arduino.cc/en/Tutorial/PWM

What is PWM?

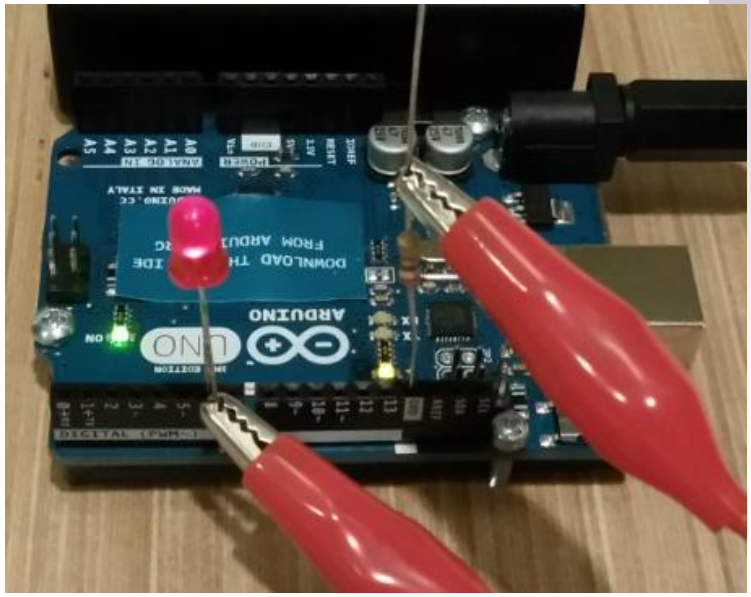
In the graphic below, the green lines represent a regular time period. This duration or period is the inverse of the PWM frequency. In other words, with Arduino's PWM frequency at about 500Hz, the green lines would measure 2 milliseconds each. A call to analogWrite() is on a scale of 0 - 255, such that analogWrite(255) requests a 100% duty cycle (always on), and analogWrite(127) is a 50% duty cycle (on half the time) for example.



http://www.arduino.cc/en/Tutorial/PWM

Dimmable LED

- You will need:
  - Robot, Laptop, External LED/Resistor
- Is there any way to vary the power to the LED?
  - Change/remove resistor
  - Change the blinks?
- Install LED in pin 6
- Install resistor in GND



All info available

- www.roboxsumo.com

- Bill McClung
- juhling@suddenlink.net
- 252-347-3498

Dimmable LED - Code

- Load code from desktop –
  - Test_2_camp_062016

```
// This program increases the intensity of light from a LED light over several steps

int ledPin = 6;      // LED connected to digital pin 9

void setup()
{
  pinMode(ledPin, OUTPUT);   // sets the pin as output
}

void loop()
{
  analogWrite(ledPin, 0);  // analogWrite values from 0 to 255
  delay(1500); // Wait 1.5 seconds
  analogWrite(ledPin, 50);  // analogWrite values from 0 to 255
  delay(1500); // Wait 1.5 seconds
  analogWrite(ledPin, 100);  // analogWrite values from 0 to 255
  delay(1500); // Wait 1.5 seconds
  analogWrite(ledPin, 150);  // analogWrite values from 0 to 255
  delay(1500); // Wait 1.5 seconds
  analogWrite(ledPin, 200);  // analogWrite values from 0 to 255
  delay(1500); // Wait 1.5 seconds
  analogWrite(ledPin, 255);  // analogWrite values from 0 to 255
  delay(1500); // Wait 1.5 seconds
}
```

analogWrite

- Used to Send PWM signal to output
  - Only on designated pins 3,5,6,9,10,11
  - 0 – 255: 0 = off, 255 = max
- SYNTAX
  - analogWrite(pin #,value)
  - Don’t have to declare pin an output to use this comand…

Dimmable LED - Code

- Load code from desktop –
  - Test_3_camp_062016

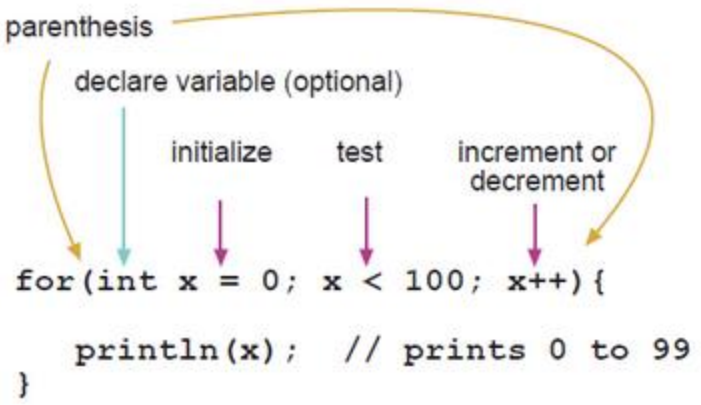
```
// This program increases the intensity of light from a LED smoothly
// using a for loop
int ledPin = 6;      // LED connected to digital pin 6

void setup()
{
  pinMode(ledPin, OUTPUT);   // sets the pin as output
}

void loop()
{
  // use for loop to smoothly increase value of i in the analogWrite statement
  for (int i=0; i <= 255; i++){
      analogWrite(ledPin, i);
      delay(15);
   }
}
```

For Loop

- The For loop
- Repeat a command set over and over till a specific parameter is met.
- SYNTAX
  - for (initialization; condition; increment) {

    //statement(s);

    }

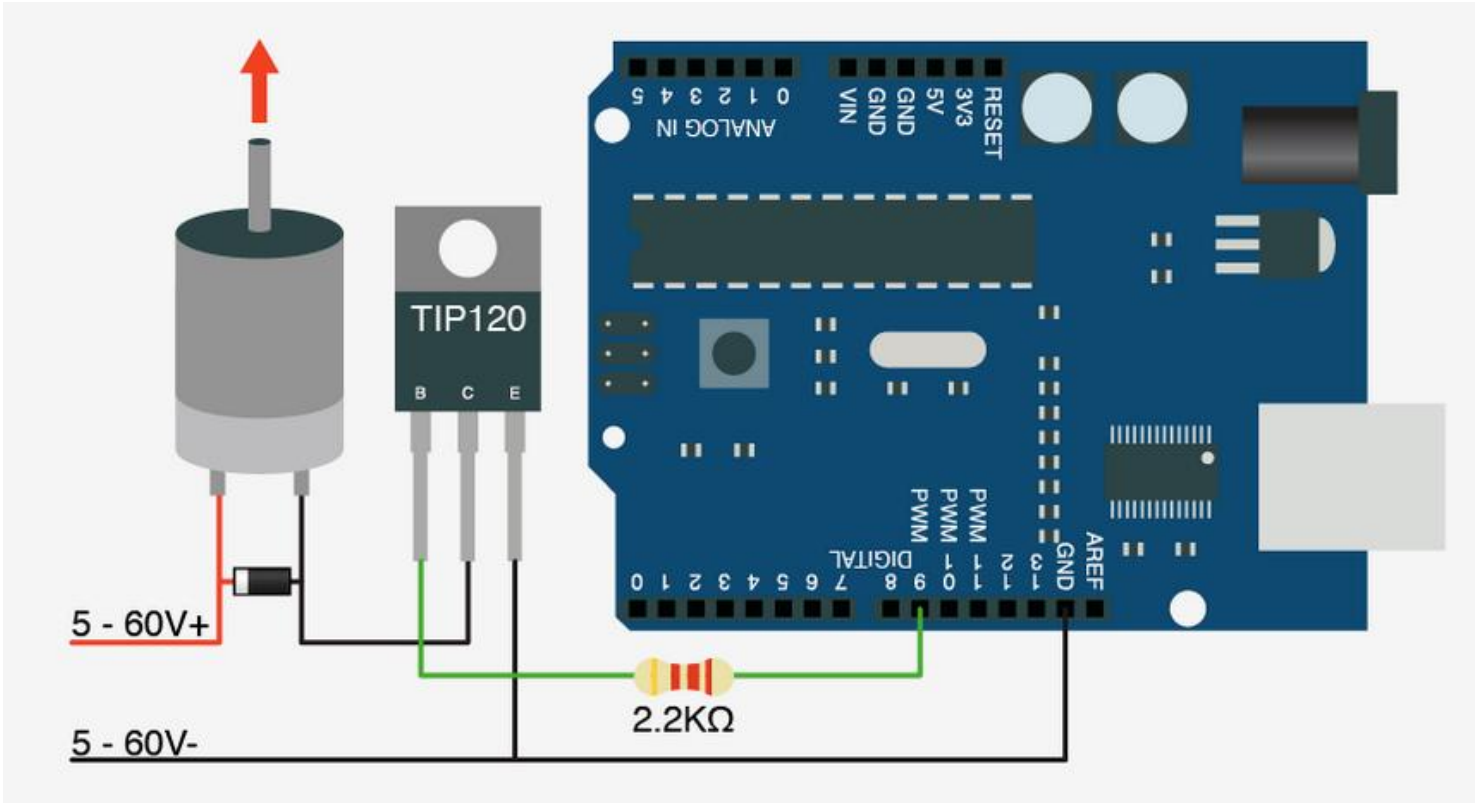


Motor Control

- We can vary output on a PWM Pin
- Now we can control motor speed, right?
  - Nope
  - Power problem - .040A x 5v = .2 watts max
  - OK for small device like LED but not good for big device like motor
- Use the small signal from the Arduino to adjust the power from a larger source
  - Motor Controller Circuit between the Arduino and the motor

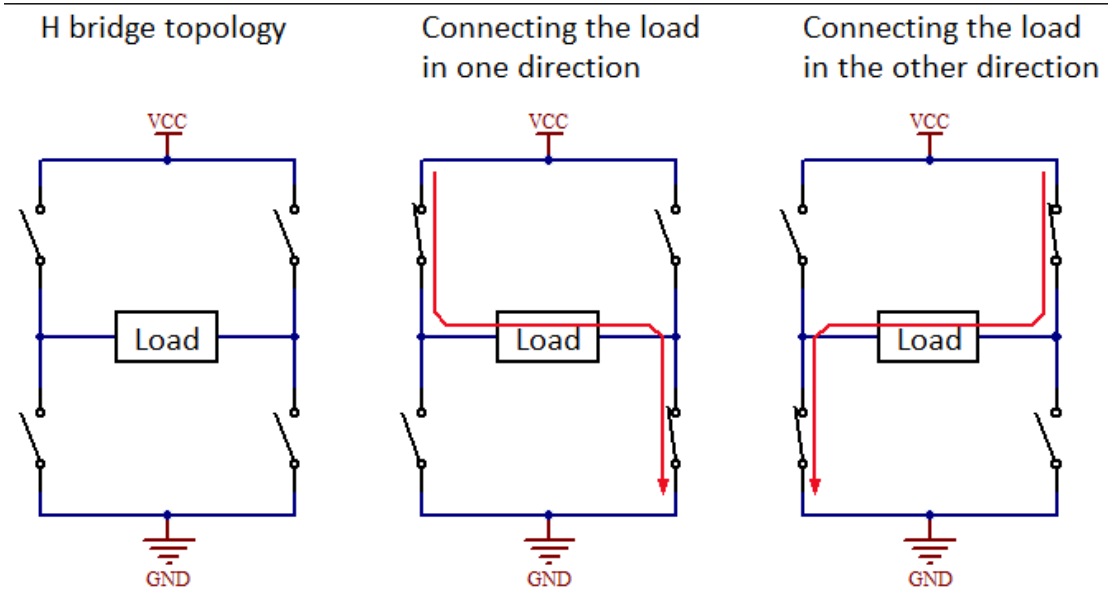
Motor Control - Example

- Use a Transistor
  - Small current from Arduino controls large current to motor



Motor Control - Problems

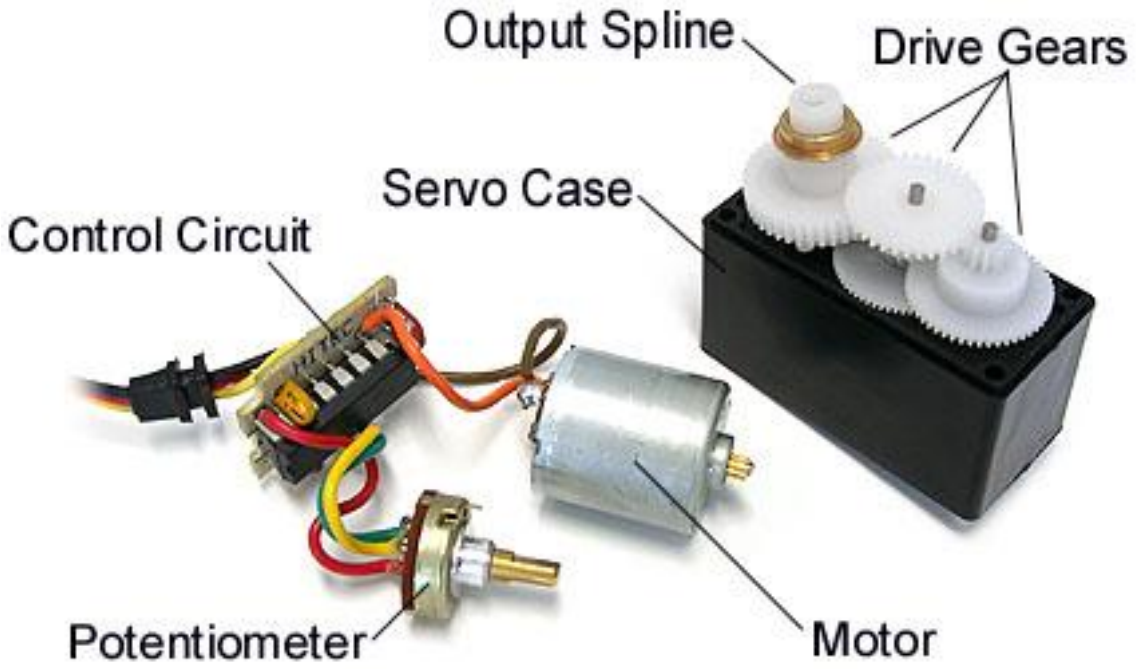
- Motor only turns one way?
  - How to reverse?
- Add an “H” bridge or relay
- Power + Direction control = Motor Ctrl



http://axotron.se/index_en.php?page=34

Combine Parts - SERVO

- Servo combines all parts in one.
  - Motor, gearbox, controls and sensor



https://www.servocity.com/html/how_do_servos_work_.html#.VYEvWvlViko

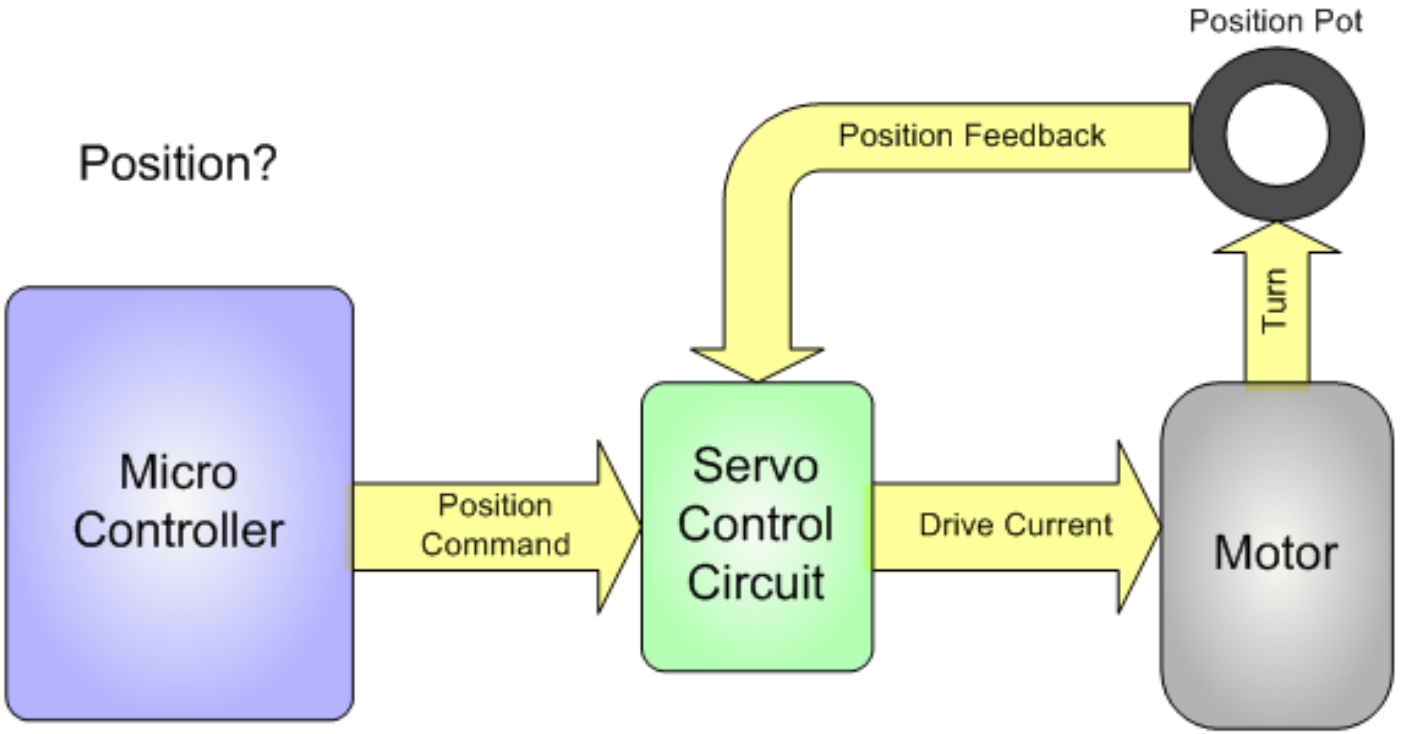
Servo – Feedback Control

- Typical servo used for positioning
- Servo moves to desired position and holds that position
- Internal feedback loop
  - Potentiometer in servo measures position
  - Control circuit compares input request to signal from the potentiometer
  - If servo is not in correct position, it will move till the input and output signal match

http://gentles.ltd.uk/control/index.htm

Servo Control Loop

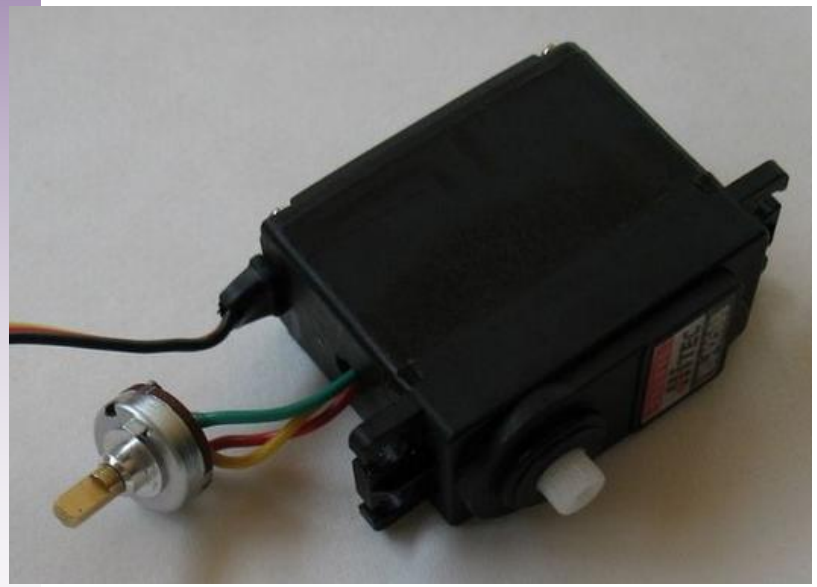
- Normal Servo looks like this.

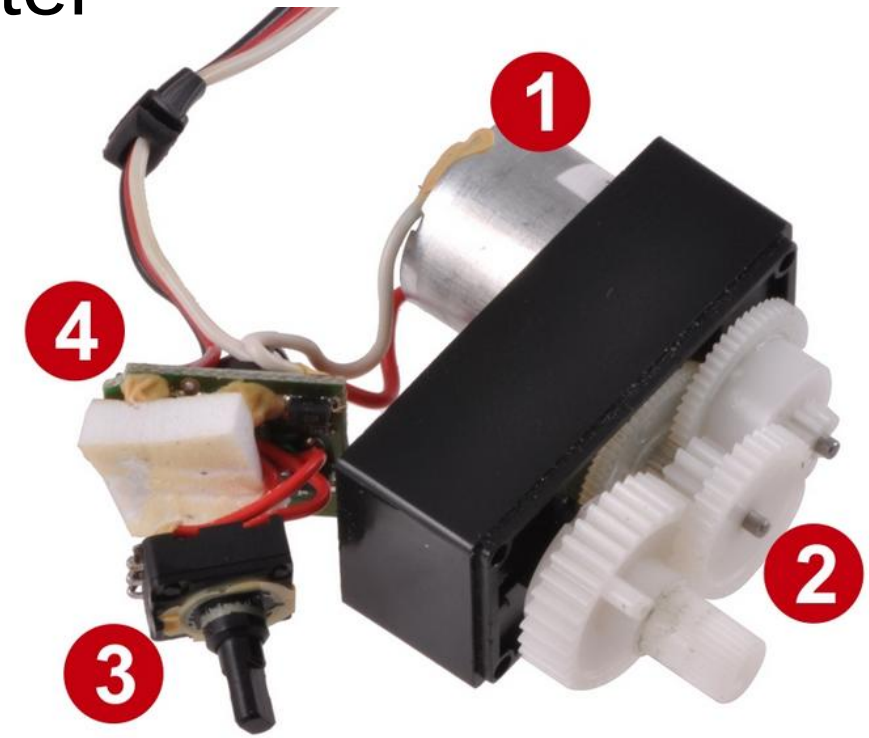


https://learn.adafruit.com/analog-feedback-servos/about-servos-and-feedback

Continuous rotation - Difference

- Break connection at position sensor.
- Modify Potentiometer





http://www.instructables.com/id/Low-Cost-Hobby-Servo-XY-Table/step6/Modify-the-Servos-for-Continuous-Rotation-and-Remo/

http://www.pololu.com

Normal Servo

- Normal servo = Pololu #1503
  - Sub-Micro Servo 3.7g (Generic)
  - Selection of servo adapters

Sub-micro servo 3.7 g on a US quarter for size reference.

Dimensions

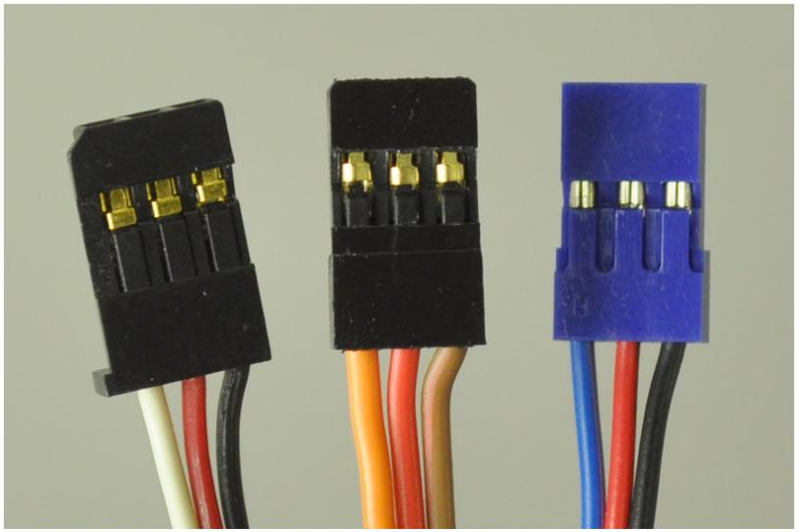
Size:	20.2 x 8.5 x 20.2 mm
Weight:	3.7 g

General specifications

Digital?:	N
Speed @ 6V:	0.07 sec/60°
Stall torque @ 6V:	6 oz·in
Speed @ 4.8V:	0.09 sec/60°
Stall torque @ 4.8V:	4 oz·in
Lead length:	6 in
Hardware included?:	Y

Servo Control - Wiring

- What do we need to connect our servo?
  - Servo needs 3 electrical connectors
    - ✓ Power – Jumper to bread board - row
    - ✓ Ground – Jumper to bread board + row
    - ✓ Signal – Jumper to Arduino PWM port
      - 3,5,6,11 still available
  - Typical servo wires
    - ✓ Color different
    - ✓ Same function/location
  - Let's borrow an existing circuit.



Common RC servo connectors. From left to right: Futaba, JR, Airtronics Z.

Servo Control

- Let's try out a "normal" servo
- You will need:
  - Robot
  - Existing servo wiring
  - Normal servo from parts
- Disconnect the right wheel servo
- Connect the test servo
- Install a servo adapter
- Watch installation on overhead

Sub-micro servo 3.7 g on a US quarter for size reference.

Test the Servo

- Open test_4_camp_062016
  - In desktop folder
  - First part

```
// Controlling a servo position using set values
// modified code from Michal Rinott <http://people.interaction-ivrea.it/m.rinott>
// modified by William McClung

#include <Servo.h>

Servo myservo;  // create servo object to control a servo

int val;    // variable 0 to read the value for the servo
int val1;   // variable 1 to read the value for the servo
int val2;   // variable 2 to read the value for the servo
int val3;   // variable 3 to read the value for the servo

void setup()
```

Test the Servo

- Open test_4_camp_062016
  - Second part – void setup

```
void setup()
{
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
  val = 20;
  val1 = 60;
  val2 = 120;
  val3 = 170;

 myservo.write(val);                  // sets the servo position according to the value
  delay(2000);                          // waits for the servo to get there
  myservo.write(val1);                 // sets the servo position according to the value
  delay(2000);                          // waits for the servo to get there
  myservo.write(val2);                 // sets the servo position according to the value
  delay(2000);                          // waits for the servo to get there
  myservo.write(val3);                 // sets the servo position according to the value
  delay(2000);                          // waits for the servo to get there
  }
```

Test the Servo

- Open: test_4_camp_062016
- Third section – Void loop

```
void loop()
{
 // nothing here
}
```

- Load and run the program
- What does it do?
- Change the values for val, val1, val2, val3. What happens? Limits?

Servo Library – The Rules

- Important notes:
  - Using “Servo” library
  - Must load library at beginning
  - Special command sets
  - Takes the place of analogWrite
- RULES for Servo library
  - Must call library at start
  - Create a servo “object”
  - Must attach it to a pin
    - ✓ Must be a PWM Pin

Now Continuous Rotation

- What happens if you:
  - Connect right wheel servo and run Test4?
  - Run away servos… Why?
  - How to stop the servo?
- If 0 is full left and 180 is full right
  - What is the middle?
  - Try setting the servo for 90 on last line
  - What happens?
- How do we use this?

Load New Code for Servos

- Let's Load a new code to control servos
- Open file: Test_5_camp_062016
  - Review code – Init section and void setup

```
// Code for camp robots 062016 - William McClung

#include <Servo.h>   // get servo library for our robot

Servo rmyservo;  // create a right servo
Servo lmyservo;  // create a left servo

void setup()
{
  rmyservo.attach(9);  // attach right servo to pin 9
  lmyservo.attach(10); // attach left servo to pin 10
  rmyservo.write(90);  // set right servo to mid-point (stop)
  lmyservo.write(90);  // set left servo to mid-point (stop)
  delay(1000); // wait one second
```

New Code for Servos

Continue Review

```
rmyservo.write(45);  // set right servo to drive forward 1/2 speed
lmyservo.write(135);  // set left servo to drive forward 1/2 speed
delay(2000);  // wait 2 seconds

rmyservo.write(90);  // set right servo to mid-point
lmyservo.write(90);  // set left servo to mid-point
delay(1000); // wait 1 second

rmyservo.write(135);  // set right servo to drive backward 1/2 speed
lmyservo.write(45);  // set left servo to drive backward 1/2 speed
delay(2000);  // wait 2 seconds

rmyservo.write(90);  // set right servo to mid-point
lmyservo.write(90);  // set left servo to mid-point
delay(1000);  // wait 1 second

rmyservo.write(135);  // set right servo to drive backwards 1/2 speed
lmyservo.write(135);  // set left servo to drive forwards 1/2 speed
delay(2000);  // wait 2 seconds

rmyservo.write(90); // set right servo to mid-point
lmyservo.write(90); // set left servo to mid point
delay(1000); // wait 1 second
```

New Code for Servos

Continue Review

```
  rmyservo.write(45); // set right servo to drive forward 1/2 speed
  lmyservo.write(45); // set left servo to drive backwards 1/2 speed
  delay(2000); // wait 2 seconds

  rmyservo.write(90); // set right servo to mid-point
  lmyservo.write(90); // set left servo to mid point
  delay(1000); // wait 1 second

  rmyservo.write(45);  // set right servo to drive forward 1/2 speed
  lmyservo.write(135);  // set left servo to drive forward 1/2 speed
  delay(2000);  // wait 2 seconds

  rmyservo.write(90);  // set right servo to mid-point
  lmyservo.write(90);  // set left servo to mid-point
  delay(1000); // wait 1 second

  rmyservo.write(135);  // set right servo to drive backward 1/2 speed
  lmyservo.write(45);  // set left servo to drive backward 1/2 speed
  delay(2000);  // wait 2 seconds

  rmyservo.write(90); // set right servo to mid-point
  lmyservo.write(90); // set left servo to mid-point

}

void loop() {}
```

Test_5 Code Questions?

- Run Test_5_Code
- Help me figure out these questions
  - What is full power forward? R&L same?
  - What is full power backward? R&L same?
  - Why are the speeds different for the left and right servo when going forward or backward?
  - How do you make a right turn?
  - How do you make a left turn?

Test_5 Code Answers

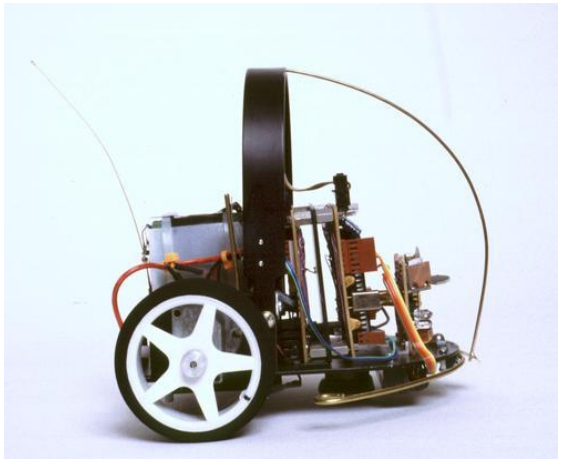
- Full speed forward?
  - Rmyservo = 180, lmyservo = 0
- Full speed backward?
  - Rmyservo = 0, lmyservo = 180
- Speeds different for the left and right?
  - Servos mounted backward of each other.
- Make a right turn?
  - Left wheel fwd, Right wheel back
- How do you make a left turn?
  - Right wheel fwd, Left wheel back

Three Challenges!

- Use “Dead Reckoning”
- Use the course provided
- Make your robot move using any combination of code
- SAVE YOUR WORK
  - Your results may be useful later

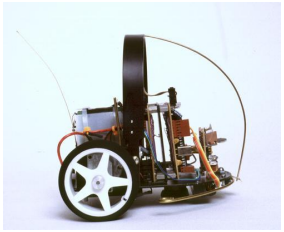
Challenge 1

- Challenge 1
  - Start 3 feet from the wall
  - Stop as close as possible without touching



Challenge 2

- Follow along divider
- Stop without hitting wall
- Turn
- Move close to wall without touching



Challenge 3

- Follow along divider
- Stop without hitting wall
- Turn
- Move close to wall without touching
- **Return to where you started!**