CS281: Computer Systems

Lab 05: Sensors and Servo Motor Control

The main purpose of this lab is to build a servo motor control circuit. Like the last lab, where we used a sensor (the distance sonar) as the "input", and then used that to drive the "output" of the actuator (the buzzer) or the "output" of the RampLight, we want to define a behavior based on at least two input sensors and then have that affect the actuator of the servo motor.

In contrast to the last lab, this lab will involve even more self-definition ... both in choice of the input sensors, in the way those input sensors manifest themselves into behavior of the actuator, and the exploratory process. But we should again divide the work into the exploratory process and the design and execution of a solution to the general problem.

During the exploratory process, we experiment with a single element, with one of the sensors, or with the servo motor. Such exploration includes learning about how the device works, how it interacts with the Arduino, and some basic Arduino IDE "sketch" programs that can either get information from the device (for sensors) or control the device (for actuators).

During the design process, you incorporate and integrate what you learned in the exploratory process to conceive of a solution, and then iterate through putting the pieces together and debugging the Arduino sketch that incorporates multiple sensors and the servo motor actuator.

Sensors

Button

One of the simplest sensor/input devices is a push button. From the perspective of a digital logic circuit, a push button makes a connection (as though attaching a wire between two points on the breadboard) when the button is depressed, and no connection exists when the button is not depressed. When wired with a resistor connected to ground on one side of the push button, and wired to +5V on the other side of the push button, we can digitally read between the resistor and the button and get a "1" or "0" and have an Arduino sketch look accordingly.

So to explore the Button, you would look (via a Google search) for Arduino tutorials using the button. For sites with multiple tutorials, this will often be one of the first tutorials. Such tutorials will give you pictures and schematics for how to wire the button on the breadboard and connect it to the Arduino, and will almost always also give an Arduino Sketch showing how it works. The arduino.cc site has a good tutorial.

Potentiometer

The potentiometer is an analog device that we wire to +5V and GND and take the third (analog) signal and wire it to one of the Analog pins of the Arduino. The Arduino sketch allows us to read the variable resistance of the potentiometer and returns a value between 0 and 1023.

Search for tutorials showing how to use the Potentiometer with the Arduino. You can also look at our previous lab to see how we used the breadboard potentiometer. From these, experiment with and test the potentiometer in isolation. The arduino.cc site has an early lesson on "Analog In Out Serial" that uses a potentiometer.

Human/PIR Sensor

The HC-SR501 is an infrared motion detector that is used to detect human movement. It does so by measuring IR from two different points and, if the IR return moves from one point to the other, it determines that motion has occurred. Such human motion detection is used in a wide range of applications, from automatic opening of doors to home security applications to controlling air conditioning and heating based on human presence.

Like the potentiometer, the HC-SR501 has just three wires used for interfacing with the Arduino. It needs both +5V and GND, and then one wire for the signal itself. Unlike the potentiometer, the signal wire is digital, not analog. So it can be read on a digital input pin with a digitalRead on the Arduino and a HIGH return indicates human motion (within the last interval of time) whereas a LOW return indicates no motion.

Using a tutorial, explore the PIR sensor. Search for the part number. I found a really basic tutorial at arduinobasics.blogspot.com.

Joystick

A joystick is a device that allows a user, through manipulation of the joystick in a particular direction, to specify to a program a 2D direction as well as an "amount" or amplitude in that direction. Interestingly, we can get the two axis of dimension with a very simple joystick that is made up of two potentiometers, one for each axis of dimension. Pushing the joystick north or south (Y axis) yields variation in resistance from no resistance to maximum resistance. Likewise with east or west for the X axis. By combining the two, we get points throughout the 2D space.

The joystick has five wire connections. One for GND, shared between the two potentiometers, a +5V connection shared between the two potentiometers, and two analog output connection for the signal line for the X axis and the signal line for

the Y axis. The two analog output connections get wired to two different analog input pins on the Arduino. There is also a connection marked 'SW' for the ability to push a button on the joystick.

Actuator

Servo Motor

A servo motor is used for positioning a directional element (arm) in a specific angle range, from 0 to 180 degrees. It is a three wire device, with +5 V (the red/orange wire for our servos), GND (the brown or black wire for our servos) and a signal wire (the orange wire). As an actuator, the signal wire is an input to the device and is output from one of the analog pins on the Arduino.

On the analog side, the control required for requesting a particular angle of the servo motor is accomplished through a mechanism known as Pulse Width Modulation (PWM), where the ratio of time at +5V to the time at 0V may be varied within a fixed oscillation period of a square wave giving a relative value. This value is then interpreted as the desired angle.

Fortunately for us, the Servo library provided in the Arduino software takes care of all of the details. All we have to do is wire up +5V and GND as always, and connect the signal pin to one of the Analog pins of the Arduino. The library allows us to create a Servo object, which we `attach` to in the setup function, and then in the loop function we can `write` a value in the range from 0 to 180, and the arm positions to that angle.

Again look for a tutorial for example code. If may help to know that our servo motors are model SG90.

Design: Use multiple input sensors to define a behavior and control a servo motor

The main purpose of this design portion of the lab is to construct a robotic device that uses two to four inputs and maps the state of those inputs into a behavior for the servo motor.

I encourage you to be creative. More points will be awarded for nontrivial designs.

Make sure your servo controller does not try to push the servo too far left (beyond 0) or too far right (beyond 179) as this will damage the servo motor internals. Make sure that the mapping is user-expectation friendly. Linear mappings are more likely to be what the user expects than non-linear mappings.

Demonstrate the correct operation of your device to the class professor.

Your lab report should focus exclusively on this design portion of the lab. Describe fully the problem given to you. Describe fully your hardware design and construction. Describe fully your software solution (include your code in the lab report). Pay special attention to the design challenges that you overcame and that makes your solution unique/better than other solutions. Include graphs, pictures, charts, tables or other visuals that will help convey your design.