Sensors & Motor Control Lab

Objectives

1. Demonstrate the use of a GUI and sensor input to a microcontroller to move motors or otherwise change motor states and, if possible, mechanical degrees of freedom coupled to the motors. Discuss potential and probable uses of sensors, motors, and motor control in your system.
2. Demonstrate ability to read a datasheet and basic knowledge of signal conditioning and PID control.

Content

1. In-lab presentation & demonstration by your team of a unified sensor-motor-microcontroller system as described in the Specification below.
2. Individual Lab Report (ILR) describing your recent work on the lab and project. Your first ILR should focus on your contribution to the Sensors & Motor Control lab and any additional progress made on your project since submission of the Conceptual Design Review Report.
3. Code for the sensor interfacing or GUI, depending on which individual task was performed.
4. Sensors & Motor Control Quiz – The quiz is attached to the assignment and should be completed independently by each student.

Presentation/Submission

Content element 1 above is presented during a 15-minute lab (NSH B506) period according to the schedule posted on Canvas under Course Information. Remember to designate one team member to present for 5-7 minutes, followed by general discussion. The Sensors & Motor Control Lab is on the Wednesday of week 4.

Content elements 2-4 above are individual, independently performed work and should be submitted by midnight on the day after the lab on Canvas in a single pdf document per student. Use the naming convention TeamA_andrewid_ILR01.pdf (replace "A" with your own team letter and "andrewid" with your own Andrew ID).

Detailed guidelines for Progress Reviews (i.e., lab presentations), Individual Lab Reports (ILR), and Team Meetings are under Course Documents-->Assignment Guidance and also linked at the bottom of this assignment. Make sure to read the PR and ILR guidance in preparing for the current task!

In addition to your lab demonstration and ILR/quiz submission, there are two more things to do for this assignment:

1. By midnight on the day the ILR is due, one person from each team should send an email with subject "Team A PR #1 Goals" (replace “A” with your team letter) to cmumrsdproject@gmail.com (and cc to your teammates) with your goals for the next Progress Review.
2. Post on your website the names of the presenters for each of the five lab presentations this semester, using round-robin scheduling. The Spring Validation Demonstration (SVD) and SVD Encore do not require a presenter to be named.

Examples
Good prior ILR examples (minus the code) are attached.

Specification
General

- Use three (or four if you are a five-person team) different sensors to control three different motors (RC servomotor, DC motor with encoder, and stepper motor) using a microcontroller of your choice. In addition to the sensors, to demonstrate your ability to interface the microcontroller to a physical switch, use at least one pushbutton switch (like those used in the Microcontroller Familiarization task) to control one or more of the motor control functions and implement switch debouncing.
- If possible, use sensors and actuators that are relevant to your project. If we don't stock the motors or sensors you want to use, order them ASAP!

Sensors

- Sensors acceptable for this lab include: Sharp IR proximity sensor, ultrasonic rangefinder, potentiometer, temperature sensor, ambient light sensor, force resistive sensor, IR emitter-receiver pair, Omron slot sensor, electret microphone, thermistor, flex sensor. The DC motor encoder ***does not*** count as one of the sensors. At least one of the sensors you choose should be an analog sensor.
- Use the Arduino (or microcontroller of your choice) and breadboard you received for the Microcontroller Familiarization task to work with the sensor.

Software

- All but one of the team members should select and write code for one sensor each, and the final team member should write the GUI.
- The end result should be a single program that runs on your microcontroller, not multiple separate programs. The code for each of the individual sensors from each of the individual team members will need to be integrated with the motor control code to form the single microcontroller program required by the lab. Don't provide or demonstrate as the end product multiple separate programs that have to be separately uploaded to the microcontroller.

Graphical User Interface

- Develop a standalone GUI application that runs on a laptop and: 1) interfaces to your embedded controller (e.g., via serial port); 2) reports the state of your motors and sensor inputs via meaningful GUI controls, and 3) provides a means to manipulate the state of the motors. There is no specific requirement of language, toolkit, etc. for developing the GUI, but suggestions include:
    - "Processing" (http://processing.org), which is a domain-specific language designed to "promote software literacy within the visual arts and visual literacy within technology." Many people have used Processing with the Arduino (see http://playground.arduino.cc/Interfacing/Processing).

    - "Qt" (http://qt.nokia.com), which is a free, open-source, cross-platform GUI toolkit. Qt provides bindings to multiple languages, including C++ and Python, and is broadly used in robotics research and development, including ROS.
    - "ROS" (http://www.ros.org), which is a highly popular open-source software system that enables rapid development of robotic systems. ROS is especially useful for research and development of advanced robotic algorithms and includes a wide variety of examples and a centralized GUI tool, called "rviz".

Presentation

- Since it's hard to see motor shafts spinning and reaching different positions, mark motor shafts that are not connected to anything else with something (e.g. colored tape) that makes it easy to see position changes. Stabilize the motors so they don't writhe around on the benchtop when moving. This should be through some type of provisional mounting. If possible, connect the motor shaft to some early facsimile or prototype of one of your system DOFs.

Grading

This task is worth the usual 10 points for lab demonstrations (*5 points ILR; 5 points lab presentation/demo*), but includes 5 points for the Sensors & Motor Control quiz, for a total of 15 points. We will grade the lab demonstration on the basis of the following:

GUI & display

1. The ability to implement a screen-based GUI to handle and display user commands and sensor feedback to control and interact with the system. The simplest form of displaying real-time sensor data is printing streaming data to the screen; a fancier example is plotting data, either statically (by collecting data for some time, then displaying it) or dynamically. It is fine to have text-based I/O as part of your GUI, but also include some graphical components for receiving input and displaying output.

Sensor processing

2. The ability to accurately process sensor data through as much of its range as is feasible (e.g.: convert the voltage on an IR rangefinder to a distance) for each of the sensors. This means that, to the extent possible (some datasheets may be sketchy in providing information), the output you display should be in physical units, not in volts or binary numbers.
3. Presentation of a transfer function plot (that means electrical input units on the x-axis and physical output units on the y-axis) for at least one of the sensors, on screen or printed out, ***based on physical measurements that you have made yourself.***
4. The ability to filter (via software or hardware) sensor data for at least one sensor to reduce noise.

Motor control

5. The ability to use both the GUI and sensor data to control the motors as follows:
    - RC servo motor: Be able to move the motor to either of its extreme limit positions and to any position (in degrees) in between. If you choose to use a continuous rotation RC servo, be able to move the motor at any desired velocity (in RPM, rev/sec, degrees/sec, etc.) within the achievable range in either direction.

- DC motor: Using an encoder for position feedback and PID control, be able to move the motor a user-selectable a) number of degrees from an initial position and b) desired velocity in either direction.
- Stepper motor: Be able to move the motor a user-selectable number of degrees in either direction.

6. The ability to implement a user-operated switch with debouncing to provide some kind of input to control some motor state(s).

Project mechanisms progress

7. Demonstration/discussion of tangible progress in the mechanisms area of your project, such as more complete or detailed drawings than for the Conceptual Design Review, detailed parts identification, partial parts ordering, or partial parts fabrication.

Parts information

Notes

Parts distribution

1. Each team will receive one DC motor, one stepper motor, one servo motor, one H-bridge (Solarbotics) for the DC motor, and one stepper motor driver. Other needed components should be in the lab.
2. **Each student (except the team member writing the GUI) will need to obtain one sensor from TAs during office hours.**

Power

3. In general, do not plug or unplug boards or chips when they are powered. This includes wiring a board into the Arduino while the Arduino is powered, and wiring a motor into (or out of) the board when the board is powered.
4. Don't power the stepper (or DC) motor from the 5V output of your Arduino – it doesn't provide enough current, and this could lead to the Arduino's resetting. Use a separate power supply for your motors.

Encoders

5. You will need to use a pull-up resistor for the Hall Effect encoder sensors on the Robot Shop Cytron DC motors in order to get a sharp, correct, and usable signal. The difference is visually dramatic if you check the encoder output on the oscilloscope.

PWM

6. The DC motors have friction that causes a low-percentage PWM duty cycle to be unable to move the motor. That nonlinearity poses a challenge to PID motor control. One way to reduce this "deadband" is by increasing the voltage used to power the motor. You can do this within reason (e.g. by powering a 12V motor at 15V), but don't go overboard - running motors at well over their rated voltage for long periods of time eventually causes thermal damage.
7. When you use the Arduino servo library with the Arduino Uno, the PWM capability on pins 9 & 10 is disabled. You therefore can't use both simultaneously. Knowing this will keep you from wasting a lot of debugging time.

Stepper motors

8. An earlier student noticed that, at least for some shipped versions of the SM-42BYG011-25 stepper motor (see pdf below), there is apparently a mistake in the datasheet wiring indication. Instead of A=red, C=grn, B=yul (yellow?), D=blue, it should be A=red, C=blue, B=black, D=green.
9. Please be careful using (at least the Pololu A4988) stepper motor drivers, since students have blown out many of them in the past. A common problem is failing to use current-limiting on either the power supply or by adjusting the potentiometer on the stepper motor driver chip.

Useful links

1. Important information for high-speed I/O on Arduino: http://arduino.cc/en/Reference/PortManipulation
2. Solarbotics L298 Compact Motor Driver: http://www.solarbotics.com/product/k_cmd/
3. Encoder Information:
   http://arduino.cc/playground/Main/RotaryEncoders
   http://en.wikipedia.org/wiki/Rotary_encoder#Incremental_rotary_encoder
4. Stepper Driver & Motor Information:
   http://www.pololu.com/product/1182/
   https://www.sparkfun.com/products/10267/
   https://www.sparkfun.com/datasheets/Robotics/SM-42BYG011-25.pdf
5. Stepper acceleration information:
   http://www.atmel.com/Images/doc8017.pdf
6. Sensor links:
   Infrared Proximity Sensor (10-80 cm): http://www.sparkfun.com/products/242
   Infrared Sensor Pigtail: http://www.sparkfun.com/products/8733
   TSOP4038 infrared receiver: https://solarbotics.com/product/TSOP4038
   Ultrasonic Range Finder: http://www.sparkfun.com/products/639
   Potentiometer: http://www.sparkfun.com/products/9939
   Omron slot sensor: Documentation attached to assignment
   Analog temperature sensor: http://www.sparkfun.com/products/245
   Ambient light sensor: http://www.sparkfun.com/products/8348
   Force-sensitive resistor: http://www.sparkfun.com/products/9673