Lab 3 (part 3): Steering and Speed Control

Preparation

Reading

Lab Manual

Chapter 4 - The Silicon Labs C8051F020 and the EVB (sections on the *C8051 timer functions, interrupts, and the programmable counter array*)

Chapter 5 - Circuitry Basics and Components (The buffer)

Chapter 6 - Motor Control (DC Motors)

Objectives

Use the sensor readings to control the hardware. The steering control developer will use the compass direction readings to control the steering. The speed control developer will use the ranger distance readings to control the car speed. The LED control developer will use the ranger light sensor readings to control the LED brightness. This laboratory has individual assignments that should **not** be combined into a single file.

Motivation

In the previous lab, pulse width modulation was introduced to change the speed of the drive-motor, the steering of the servomotor, and the brightness of an LED. In this lab, the compass and ranger readings will be used to control the steering, the speed of the car, and LED brightness respectively.

Lab Description & Activities – Speed Controller Task

1. Use the ranger distance reading to control the drive motor speed in a continuous loop. Inside the loop, poll a run/stop switch connected to Pin 6 of Port 3 (P3.6). If the switch is ON, the drive motor speed should be running. If the switch is OFF, the motor should stop running. (The gondola will also have switches on the pins.) Create a simple control program where

   – The motor has full power forward if an object is 10 cm or less above the car.
   – The motor is neutral when the object is ~45 cm above the car.
   – The motor has full power reverse if the closest object is more than 80 cm from the car.
   – The pulse width varies linearly for distances from 10 cm to 80 cm, between max forward and max reverse pulsewidths.

```
error = neutral_height - actual_height;
temp_servo_pw = k*(error) + center_pw;
```

2. Have a TA verify the speed performance of your car.

- Print the distance and the motor pulse width as you move any object near the ranger.

Lab Description & Activities – Steering Servo Task

1. Use the compass reading (heading) to control the steering servo in a continuous loop. Inside the loop, poll a run/stop switch connected to Pin 7 of Port 3 (P3.7). If the switch is ON, the steering servo should turn the car towards desired heading. If it is OFF, the wheels should be parallel to the car. (The gondola will also have a switch on P3.7).
   - Install a slide switch on the protoboard and connect it to P3.7.
   - Assume a desired heading, make this a variable so it can be changed later, but fix the value for now. For example:

     ```
     unsigned int desired_heading = 900;
     ```

   - This example sets the heading to 90°, assuming that the units used are 1/10 of a degree.
   - Read the actual heading
   - Use the difference between the actual and the desired heading to set the servo pulsewidth using proportional control.

     Notes: consider the following control algorithm:

     ```
     error = desired_heading - actual_heading;
     temp_servo_pw = k*(error) + center_pw;
     ```

   - In the above equation, use a value for `k` that yields a maximum change about center_pw of 750.
   - This algorithm fails if the desired heading is in the range of 270° to 360° and the actual is in the range of 0° to 90°, and vice versa. Develop an algorithm that handles these cases.
   - Center_pw is the value obtained using your Lab3 Part 1 code.
   - The variable temp_servo_pw must be checked and adjusted to be within the range determined using your Lab 3 Part 1 code.
   - The CCMn (Capture/Compare Module n) value can then be set by a line of code such as:

     ```
     SERVO_PW = temp_servo_pw;
     //Check to insure SERVO_PW is withing max and min limits for motor
     cex0_lo_to_hi = 0xFFFF - SERVO_PW; //Implement current PWM
     ```

   - There is not an effective way to determine the best value of k at this point. Adjust the value of k so that steering will respond as the car is turned.
2. Have a TA verify the speed performance of your car.
   - Print the desired heading, the actual heading and the steering pulse width as you rotate to the desired heading.

Lab Description & Activities – LED Task

1. Use the ranger light sensor reading to control the brightness of the LED in a continuous loop. Inside the loop, poll a run/stop switch connected to Pin 5 of Port 3 (P3.5). If the switch is ON, the LED should be on, with light intensity dependent on the ranger light sensor. If the switch is OFF, the LED should be off. (The gondola will also have a switch on P3.5). Create a simple control program where

   – The LED is brightest if an object if the light sensor returns a value less than 40.
   – The LED is dimmest if an object if the light sensor returns a value less than 215.
   – Use your calibrated brightest and dimmest pulsewidth values from laboratory 3.1.
   – The pulse width varies linearly from brightest (PWLEDmax) to dimmest (PWLEDmin) as the light sensor varies from 40 to 215

2. Have a TA verify the changing brightness of the LED as the ranger light sensor changes.
   – Print the light sensor value and the LED pulse width as you cover the ultrasonic ranger.

Lab Check-Off: Demonstration and Verification

1. Note your results and present them to your TA.
2. Your TA may ask you to explain how sections of the C code or circuitry you developed for this exercise work. To do this, you will need to understand the entire system.
3. Be able to explain how the PCA is used to generate the PWM signals.
4. Capture the Hyper-terminal screen showing the following output for at least 5 different values:

   Ultrasonic ranger developer: Print the distance and the motor pulse width.

   Electric compass developer: Print the desired heading, the actual heading and the steering pulse width.

When completing this exercise make sure the compass and ranger remain mounted on the car to be available for use in the next class or open shop. **The compass and ranger should never be mounted on your own protoboard.**