DL8 — 6.01 – Fall 2011

Turning Heads

Goals: Design Lab 8 focuses on designing and demonstrating circuits to control the speed of a motor. It builds on the model of the motor presented in **Homework 2** and the proportional controller studied in **Design Lab 6** (and earlier), culminating in a simple feedback system which steers the motor in the robot head to point its photoresistive eyes toward an incident light source.

- Characterizing the Lego motor used in the robot head
- Buffering the voltage used to drive the motor, using an op-amp
- Designing a simple bi-directional speed controller
- Demonstrating a feedback system to turn the robot head towards light, using a robot brain to control the motor

Resources: This lab should be done with a partner. Each partnership should have a **robot** and lab **laptop** or a personal laptop that reliably runs soar. In addition, you will need some lab equipment and components, as specified in the **Resources** sections below.

Do `athrun 6.01 getFiles` to get the following files (in `Desktop/6.01/designLab08`):

- `noInput.py`, `oneStep.py`, `threeSteps.py`: Input signals for the CMax simulation.
- `turnHeadToLightBrain.py`: A brain with a state machine acting as a proportional controller to provide feedback and steer the motor in the head towards a light.

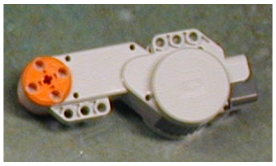
1 Lego Motor

Objective: Characterize the Lego **DC motor**, by determining its resistance and learning how to drive it.

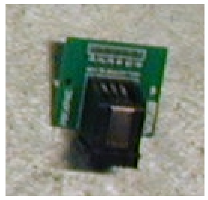
We use a Lego motor for the "neck" of the robot head. As described in **Homework 2**, the angular acceleration of the motor is proportional to the current flowing through it. And the current is inversely proportional to the resistance r_m of the motor. We shall determine r_m, and investigate simple circuits with voltage sources, resistors, and op-amps, for driving the motor.

Some of the software and design labs contain the command athrun 6.01 getFiles. Please disregard this instruction; the same files are available on the 6.01 OCW Scholar site as a .zip file, labeled Code for [Design or Software Lab number].

Resources: Obtain the following equipment and components



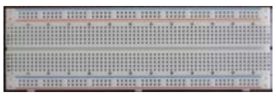
Lego motor



Six-pin connector

Multimeter

Two 1 KΩ Resistors



Proto board

Power supply

Four clip leads

Black motor cable

Images © source unknown. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/fairuse.

The motor attaches to a **6-pin** proto board connector via a short black cable; notice that the two ends of the cable are different: the locking clip is centered on one end and offset on the other. The end with the centered clip goes into the connector, and the end with the offset clip goes into the motor. The motor is driven by the voltage difference between pins 5 and 6 of the connector.

1.1 Driving the motor

Step 1. The motor is designed to be driven with a voltage difference between 0 and 10 V across its terminals. Try it out as follows.

- Connect the power supply terminals labeled +15 V and **GND** to the power rails of your separate proto board using clip leads; **do not build your circuit directly on your power supply**. Adjust the power supply voltage to 0 V. (Yes, really 0).
- Plug a **6-pin** connector into the proto board and connect it to a standalone motor; **do not use a pre-built head**.
- Turn off the power supply; wire pins 5 and 6 of the connector to the power and ground rails, respectively, of the proto board.
- Turn on the power supply.
- Connect a multimeter to measure the voltage from the power supply.
- Adjust the power supply voltage between 0 and 10 V and note the relation between motor speed and applied voltage.
- Swap the connections to power and ground. What happens?
- What is the minimum voltage required to make the motor turn?

$V_{min} = 0.38$

Step 2.

- Remove the connection between the motor and power rails of the proto board.
- Re-adjust the power supply back to +10 V (in preparation for next part), then turn it off.
- Measure the resistance r_m of the disconnected motor using the multimeter.

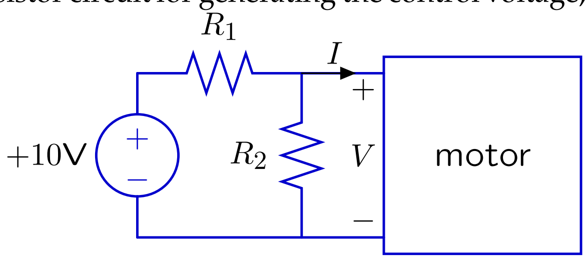
$r_m =$ 4.6

1.2 Controlling the motor with resistors

Our goal is to control the motor electronically. We will ultimately mount the motor on the robot and use the robot's power supply, which is constant at 10 V. The point of this section is to find a way to use a constant-voltage power supply to get a range of motor speeds.

First, think about how we might control the velocity with resistors. One way might be to use a voltage divider to generate a control voltage between 0 and 10 V, and then use this control voltage to drive the motor.

Consider the following resistor circuit for generating the control voltage, where $R_1 = R_2 = 1000\Omega$.



Step 3. Build the circuit on your proto board. Turn the power supply back on and measure the voltage across the motor and observe the motor's behavior.

$V_{motor} =$ 0.04

Check Yourself 1. Does the motor turn? Explain.

Step 4. Use circuit theory, treating the motor as a resistor, to determine the voltage across the motor. Use the resistance value you measured in **step 2**.

$V_{motor} =$ 0.045

Check Yourself 2. Does the theory match the measurement in the previous part? Explain.

2 Buffering the motor voltage

Objective: Design and build a circuit which can turn the Lego motor in one direction, at a controlled speed.

Resources: Obtain the following equipment and components

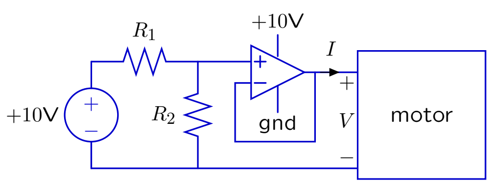
Poten-
tiometer

Op-amp in
an 8-pin DIP
package

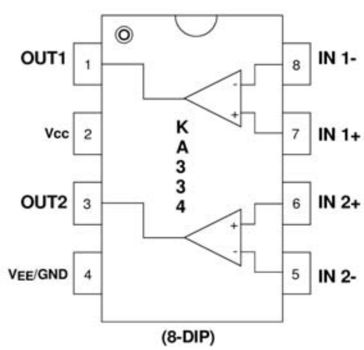
Resistors
as needed

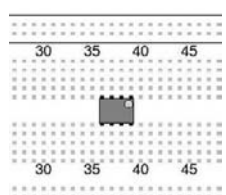
- `CMax.py` - CMax simulation program (from the `designLab08` folder)

We can add an op amp to *buffer* the output of the resistor network so that the resistors function as a voltage divider while the resulting voltage drives the motor. A simple buffer circuit is shown below.



We use **op amps (KA334)** that are packaged so that two op amps fit in an 8-pin **(dual inline) package**, as shown below.





KA334 internal block diagram (middle image) © Fairchild Semiconductor. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/fairuse.

The spacing of the pins is such that the package can be conveniently inserted into a breadboard as shown above (notice that a dot marks pin 1). V_{CC} (pin 2) should be connected to the positive supply voltage (+10 V), and V_{EE} (pin 4) should be connected to the negative supply voltage (ground).

Step 5. Start CMax by running `python CMax.py` from the `designLab08` folder (for additional instructions see the **CMax documentation** found in Software and Tools on the OCW Scholar 6.01 site.) Make sure you are running the CMax distribution contained in the `designLab08` folder (hitting `ctrl-a` should bring up a window saying you are running CMax version 1.6), and make sure that you only use simulation files contained within the `designLab08` directory. If you see an error message relating to an argument called "meter" or "display," you are using an outdated simulation file.

Add a **Motor Connector** to your layout; the motor will be driven by the voltage difference between pins 5 and 6 of this connector.

Use CMax to lay out the buffered divider circuit. In your layouts, **use only short vertical or horizontal wires**, and **do not cross wires**. Crossing wires, as well as diagonal or excessively long wires, make your circuit harder to debug and build.

Also, double-check that you are using the correct pins on the op-amp package. You can check the accuracy of your circuit by running the `noInput.py` simulation file.

Running the simulation will produce several graphs, all of which have time on the x axis, and some other quantity on the y axis. Each signal is sampled at intervals of 0.02 seconds.

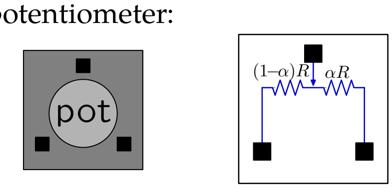
- **Probe** (in green): When there is a probe in the circuit, this graph shows the voltage measured across the probes.
- **Motor** (in red): When there is a motor in the circuit, then simulations **assume that the motor is attached to a potentiometer**, which turns as the motor turns, as in the robot head. One of the motor graphs is the α value of the motor potentiometer. The other motor graph shows the motor's rotational velocity (rad/s). Remember that the potentiometer has a finite range of rotation (0 to $3\pi/2$ radians), so with a constant voltage, the motor will quickly reach the end of the range and stop (and the rotational velocity will go to zero).

  When you're using a real robot head, driving it up against the end of the range in this way risks tearing the head apart.
- **Input** (in blue): When there is an external input to the simulation, such as a potentiometer, this graph shows the input value, for example, the value α for a potentiometer, which goes between 0 and 1.

Step 6. Build the buffered divider circuit on your proto board (exactly like your CMax layout). Measure the voltage across the motor and observe the motor's behavior.

Check Yourself 3. Compare the behaviors of the circuit with and without the buffer.

Step 7. Now, in CMax, **replace the two resistors in the voltage divider with a potentiometer**, so that when $\alpha = 0$ (the pot is turned as far counter-clockwise as possible), the voltage across the motor is 0, and when $\alpha = 1$, the voltage across the motor is 10. The diagram below shows the resistances between the three nodes of the potentiometer:



Load the simulation file `oneStep.py`, which specifies the input to the potentiometer (`oneStep` simulates a potentiometer that starts at $\alpha = 0$ and changes abruptly to $\alpha = 0.1$ after some time).

Check Yourself 4. Make sure you understand the meaning of the motor rotational velocity and motor pot alpha graphs. **Save your CMax circuit and resulting plots.**

Step 8. Replace the resistors in your physical circuit with a potentiometer, as in your CMax layout.

Step the potentiometer through various settings (1/4 turn, 1/2 turn, 3/4 turn). Observe the behavior of the motor, and compare this behavior to your CMax simulation.

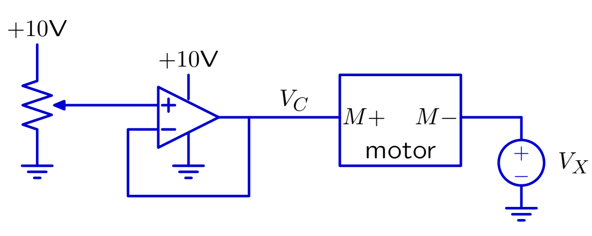
Checkoff 1. **Wk.8.2.1**: Explain to a staff member the results of your experiments, with and without buffering. Compare the simulated behavior to the actual behavior of the circuit you built.

3 Bidirectional Speed Controller

Objective: Design and simulate a circuit which can turn the Lego motor both clockwise and counterclockwise.

The circuit you built for Checkoff 1 controls the speed of a motor. That circuit allows the motor to turn fast or slow (depending on the choice of resistors or the pot setting), but only in one direction. To make our robot head turn both left and right, we need to design a bidirectional speed controller; when the α value of the pot is near zero, the motor should spin quickly in one direction; when α is 0.5, the motor should be stopped, and when α is near 1, the motor should spin quickly in the other direction.

The circuit in the previous parts of this lab only turns in one direction because the op-amp operates from a single +10 V power supply (and the op-amp can only produce an output voltage that lies between V_{ee} (ground) and V_{cc} (power)). We are limited to a single +10 V power supply, because it is the only power supply available from the robots for which we are building the "head." A simple approach to this problem (using a potentiometer) is to connect the motor as follows:

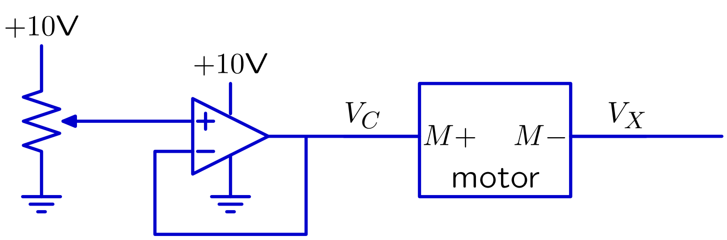


Step 9. The key new component in the bidirectional speed controller is the voltage source V_X. What value of V_X gives the most symmetric (around 0) range of speeds for the motor?

$V_X =$

Design a circuit to implement this voltage source (using only a fixed 10 V supply, which is all that's available from the robot). Modify the circuit diagram below to include your design of a circuit for supplying V_X.

Check Yourself 5. Can you implement V_X with just a voltage divider? Explain.



Step 10. Use CMax to lay out your bidirectional control circuit. Again, make sure to use only short horizontal or vertical wires, and avoid crossing wires.

This circuit can be tested with the `threeSteps.py` simulation file, which simulates turning the potentiometer first to $\alpha = 0.25$ and holding it there, then turning it to $\alpha = 0.5$ and holding it there, and finally turning it to $\alpha = 0.75$ and holding it there.

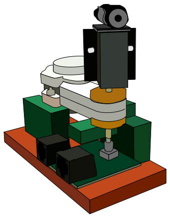
Checkoff 2. **Wk.8.2.2**: Demonstrate your working simulation. Explain the relation between motor speed and potentiometer angle. Demonstrate that you can generate both positive and negative speeds. Explain how your circuit accomplishes bidirectional speed control.

Save your plots , as well as your CMax circuit. Mail these results to your partner. We will discuss them at your interview.

4 Show me the light!

Objective: Add feedback via a robot brain to control the motor in a robot head and steer its photoresistive eyes to point towards a light.

Resources: In addition to the equipment needed for the previous section, you will use:



Robot head

Robot

Two eight-pin connectors

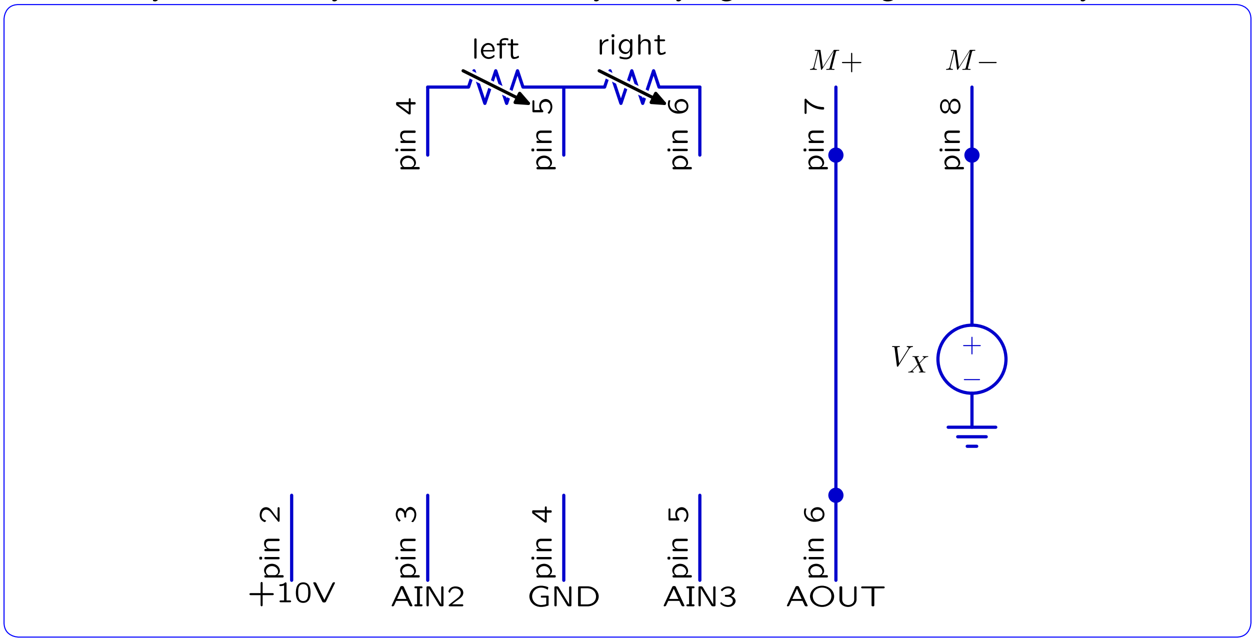
Red cable

- Additional resistors and wires as needed
- `turnHeadToLightBrain.py` - Robot brain file with a proportional controller for providing the motor feedback

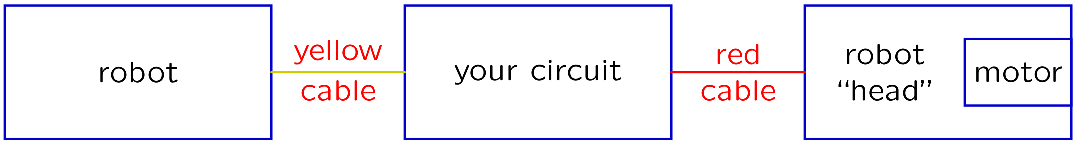
We now construct a system which moves the Lego motor to steer the eyes (two photoresistors) on the robot head to locate and track a light. This combines the circuit you designed for **Design Lab 7** to sense light, the proportional controller ideas studied in **Design Lab 6** (and earlier), and ideas from the bi-directional controller (above).

The circuit to construct is shown in the following diagram. Note that the voltage source V_X should be implemented with your circuit from the section above. The bottom pins show the **robot connector**, and the top pins show the **head connector**. Add to this diagram your eye circuit from Step 16 of Design Lab 7.

This is not yet a CMax layout! Think of a way of laying it out that gives a clean layout.



Just as in Design Lab 7, here is how the whole system should be configured:



It is convenient to mount the head on the robot. The two voltages derived from the photoresistors (using your circuit from Design Lab 7), V_L and V_R, should be connected to analog input #2 (pin 3) on the **robot** connector and connect V_R to analog input #3 (pin 5), respectively. The robot will process these signals in its brain, then produce a voltage through its analog output port (**AOUT**), using a **D-to-A (digital to analog) converter** within the robot.

The **AOUT** signal is a voltage roughly in the range of zero to +10 volts, and is already buffered sufficiently to drive the motor. The **AOUT** signal should be wired up to the + terminal of the motor, and your V_X voltage source's output should be wired up to the − terminal of the motor. You **must** use the +10V provided by the robot (through the **robot** connector instead of the bulky power supply), to power your circuit implementing the V_X voltage source.

Step 11.

- Make sure that both the yellow cable and the read cable are connected to your circuit and head.
- Connect the black cable to motor connector on the head but **do not connect it to the motor on the head yet.**
- Turn the robot on.

- Start soar, select the `turnHeadToLightBrain.py` brain, and click Start.
- Turn on the light.
- Connect the black cable to the motor. If the head slams against the side, disconnect the cable immediately!
- Verify that the robot head follows as you direct the light from different incident angles.

Step 12. When you stop the brain it will show a plot of the difference between the two eye voltages. You can use this plot to measure how quickly the head responds (count how many dots before it reaches the target). Test the behavior of the system (see below) by quickly moving the light in one direction, pausing and then quickly moving it back. Repeat this a couple of times. Then, stop the brain and examine the resulting plot.

- Start with the light at approximately 0.5 meters from the head. Change `gain` in the brain file to get the head to track the light as fast as possible without excessive oscillations. Keep track of the gain and save the plot.
- Move the light to about 1 meter away. What's the best gain/speed now? Save the plot.
- Modify the brain to have an additional delay.
- Find the gain with the fastest non-oscillatory response at 0.5 meters. Keep track of the gain and save the resulting plot.
- Repeat at 1 meter.

Save your plots, labelled with the gain and distance. Mail these results to your partner. We will discuss them at your next interview.

Checkoff 3. **Wk.8.2.3**: Demonstrate your working light tracker. Show how fast you can make the motor smoothly track the position of the light. Explain the observed motor behavior in response to changes in gain, distance and delay. What limits the speed with which you can track the light?

Turn off your meter, disassemble your board, and put the wires, op-amp, pot, and connectors back in the appropriate places. Throw away the resistors.

MIT OpenCourseWare
http://ocw.mit.edu

6.01SC Introduction to Electrical Engineering and Computer Science
Spring 2011

For information about citing these materials or our Terms of Use, visit: http://ocw.mit.edu/terms.