Hookup to drive motors & encoders 1/12/07

Motors

The motors can be configured to drive either a conventional sign-magnitude PWM h-bridge, or to drive a motor controller which accepts RC type signals. The configuration is selected with jumpers on the microcontroller board which are described in the assembly directions and/or in the microcontroller board modifications page.

Sign-magnitude PWM:

The microcontroller board currently provides motor drive commands to MOT0 and MOT1. These commands consist of:

A PWM signal with a period of 15 Khz and a duty cycle varying from 0 to 100%. The PWM signal is ON when the signal is high.

A Direction signal which is of opposite polarity for MOT0 and MOT1 since the motor installations are usually mirror images. The dir signal is LOW to provide forward motion on the left wheel and HIGH to provide forward on the right wheel.

A Brake signal which is currently not used in the motor control logic, but which is set LOW. Hence, use of this signal is optional in your motor driver design.

The PWM control method used is sign/magnitude in which the PWM signal is always 0 duty cycle for no motor command and varies up to 100% to command forward motion either in the forward or reverse direction depending on the state of the dir signal.

Hence, to interface these signals to your motors, you need to supply an h-bridge or other controller which can accept the above signals. This controller will convert the low power logic signals into high power motor drive signals. For low current motors, such as the Pittmans, an LM18200 (or 18201) h-bridge will do the job. The 18200 will handle a continuous 3 amps with peaks of up to 6 amps for less than a second. A suitable 18200 board can be found at www.picobotics.com as their PicoHB part.

For larger motors, appropriate H bridges can be found. If anyone finds something that works, please let me know and I'll list it on this page. Scott Mummert is planning on using the board to drive a robot with wheelchair motors. This should generate a large motor option.

While the direction signal described above is the convention used by the controller board, ultimately the motor direction is determined by the order in which the wires from the h-bridge/controller to the motor are attached. If the motor is turning in the wrong direction, just reverse the two motor wires.

Maybe a few comments on selecting motors would be appropriate. The current robots weigh in between 55 lbs and maybe 75. They don't expect to get a lot heavier. Adding an arm someday might add another 25 or 30 lbs. Your robot can be whatever weight you like.

The motors have to be selected with enough power to run in the environment you plan to run in. The current robots are designed to run indoors on hard floors and rugs and on level surfaces. They really don't need very large motors.

The torque available depends on the gear ratio between the motor and the distance traveled. I feel that robots that are large, heavy and maybe a bit dangerous should move at relatively slow speeds. Hence, the current robots are geared down to achieve a top speed of about 500 mm/sec (1.5 feet/second). Gearing them down this much (a total reduction of about 36:1 to a six inch diameter wheel), provides sufficient torque. A large gear reduction also provides more precision in moving the robot.

RC motor control:

This configuration requires two jumpers as described in the assembly instructions. In addition, MOT2 is used rather than MOT1 to connect to the right motor.

For this operation, only the PWM output signal is used which will have a positive pulse every 16 milliseconds which varys in width from 1.0 to 2.0 milliseconds.

Since there are a lot of RC motor controllers available and only two have been tested so far, the following is a general hookup which should usually work.

You will need two cables, each of which has two wires connected to the MOT0 and MOT2 connector to the ground and PWM contacts. The other end of the cables should be in an RC style 3 pin connector and should go to the ground and signal inputs. If your motor controller requires an external 5 volts on the third RC connector pin, you will have to provide it yourself. It is probably better if you get it from the power source coming into your controller; as the microcontroller board has a limited amount of spare power available.

Encoders

The motor control equations in the microcontroller require position feedback from a quadrature encoder to work. The ENC0,1 ports provide +5vdc, ground and two input signals. The two signals should vary in the range of 0 to 5 vdc.

The encoder should be fairly high resolution. Our current robots with the Pittman motors with built-in encoders are providing about 40 clicks on each encoder channel per millimeter of wheel motion. If you can arrange comparable resolution, the system should work fine. (actual calibration of the encoders is done separately, you don't have to be exactly the same).

The robot provides commands and controls in millimeters. Hence, you really should try to have a resolution of at least 1 click/ mm. Below that, performance may deteriorate. Of course, you are always free to adapt the microcontroller code to work with your setup.

The connection of the two encoder signals to the microcontroller board determines the whether the board thinks the wheel is moving forward or backward for a given motor command. You could probably go to great lengths to figure out the proper wiring of these two signals (and have, at least, a 50/50 chance of being right ;-) or you just hook up the two wires temporarily, roll the wheel forward a bit and see if the distance measured by the microcontroller is becoming more positive (forward) or negative (reverse). If it is in the wrong direction, just swap the two signal wires.