Introduction to Motion Control

Arthur Richards, February 2010

This guide offers a very brief survey of motion control methods, including options for interfacing. Only electrical systems are considered, focussing on methods that may be of use for the EMAT20002 Engineering Design Project and relating experience from past years of that course. This is not an exhaustive list. The guide is divided into two parts: **sensing** and **actuation**.

1 Actuation

1.1 DC Motors

Direct Current (DC) brushed motors are, in terms of the electrical connections, the simplest of all actuators – see Figure 1. Connect the two wires to either end of a DC power supply and the motor will turn. Reverse the polarity and the motor will turn the opposite way. Loosely, the voltage is proportional to the speed and the current is proportional to the torque, although this tends to break down at low speeds. Inside the motor, there are permanent magnets fixed to the casing, and a rotating shaft with multiple electric coils. As the shaft rotates, electrical contacts or "brushes" fixed to the casing cause different coils to be energized.

You can buy purpose built DC motor drivers that interface to computers (Figure 2), such as the Phidget Motor LV available. These tend to offer variable speed control, and alter the voltage by switching the full supply on and off in different time ratios. For example, if 9v is on for 66% time and off for 34%, the motor sees roughly 6v and responds accordingly. It hums a bit, but offers good performance even at low speeds. Furthermore, these drivers often include some digital inputs, intended to be used as limit switches *e.g.* to detect an automatic door reaching fully open or fully closed position.

If you only want simple run/stop control at a fixed speed, you could just use a relay, which is an electrically operated switch. This allows the small current from a digital output to control a large current driving a motor – digital outputs alone don't typically provide enough current themselves. Figure 3 shows a simple one-way circuit, and Figure 5 shows an extended circuit for bidirectional running. Note that the arrangement in Figure 5 short circuits the motor if its not commanded either way: this causes it to stop abruptly, rather than just slowing gently. Off-the-shelf motor driver

chips like the L298 could be used instead of the relays if you prefer. Finally, if you want to save some money and do your own electronics for variable speed drive, Figure 4 shows a circuit for a manual controller that will work, but won't be very efficient or powerful. Replace the potentiometer with an analog output and the switch by a relay to automate it.

Figure 1: A Typical DC Motor

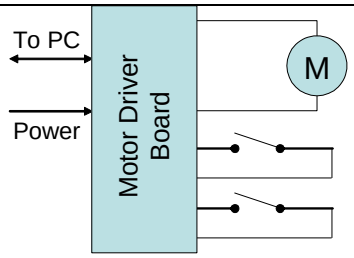


Figure 2: General Connections for Typical DC Motor Driver, including Limit Switches

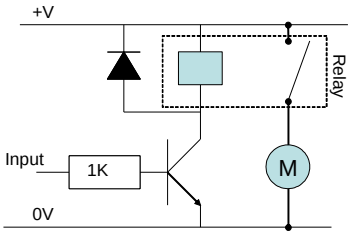


Figure 3: Simple Relay-based On/Off Motor Drive Circuit

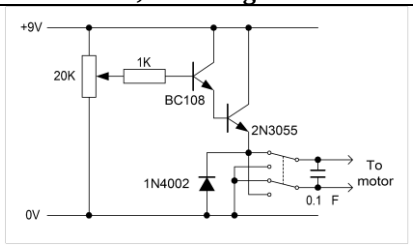


Figure 4: Manually Controlled Variable Speed Motor Drive Circuit

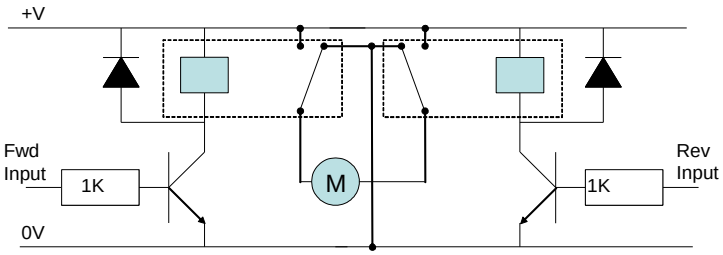


Figure 5: Relay-based Bidirectional Motor Drive Circuit

DC motors are available in an enormous variety of sizes, speed and torque ratings, and are relatively cheap. The downside for motion control is their unpredictability. The response typically varies significantly from motor to motor, with different loading conditions and different driving circuits. As a result, they are nearly always combined with some sensors for feedback. Also, the contacts from the body to the shaft that switch the polarity of the coils inside (the "brushes") tend to cause electrical noise, so be careful combining DC motors with sensitive circuits. The capacitor across the motor connections in Figure 4 is intended to filter out some of this noise.

1.2 Stepper Motors

Unlike brushed DC motors, stepper motors let you control the coils yourself. This allows you very fine control over the movement in terms of small steps or rotation – by clever use of multiple coils, typical step sizes are between 1^o and 4^o. Sometimes a motor is specified in terms of number of steps: hence a 128 step motor has steps of $360^o/128 = 2.8^o$. A very simplified schematic of an 8-step motor and its coil control sequence are shown. Happily, even for a motor with many more steps, the sequence and number of connections stays similar.

Figure 6 shows a small selection of stepper motors. There's not quite such a variety available as DC motors, and they tend to be more expensive and a bit bigger for the same job. The other difference is that there are more wires, and the control sequencing clearly requires more complicated driving circuitry. Relays, as used for the very simple DC motor control above, generally cannot switch fast enough for practical stepper motor control. Happily, there are some generally-available semiconductor products that do the job, e.g. the L298 chip for driving the coils and the L297 for generating the coil command sequences – see Figure 9 for the general arrangement. Alternatively you could generate the sequence yourself (*e.g.* in software driving digital output pins from a microcontroller) but you'd still need an L298 or similar to amplify the current – see Figure 8. Finally, you can buy controllers (e.g. the Phidget Stepper Driver) that do everything, taking commands by USB or serial connection and producing drive outputs direct to the motor (Figure 7).

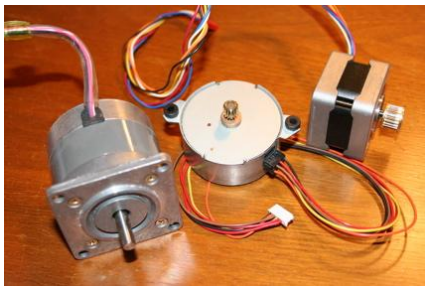


Figure 6: Selection of Typical Stepper Motors

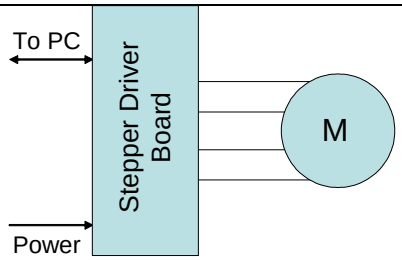


Figure 7: General Connections of a Commercial Stepper Motor Driver

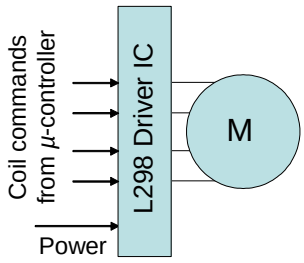


Figure 8: General Connections for Driving using a Microcontroller for Sequence Generation

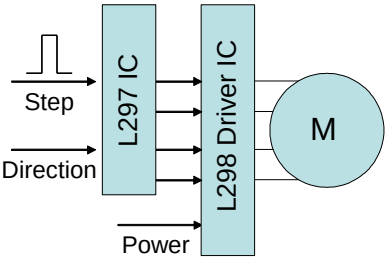


Figure 9: General Connections for Driving a Stepper Motor using the L297 for Sequence Generation

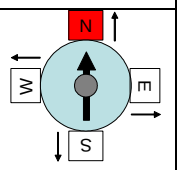
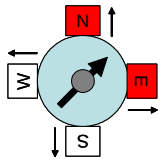
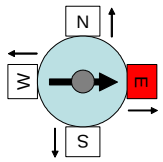
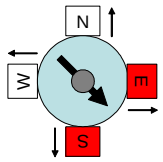
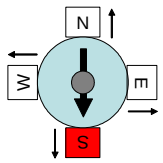
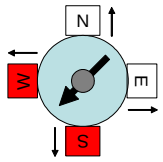
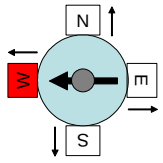
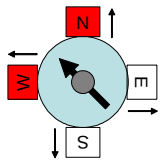
Step No.	Coils				Angle
	N	E	S	W	
1	ON	OFF	OFF	OFF	0° 
2	ON	ON	OFF	OFF	45° 
3	OFF	ON	OFF	OFF	90° 
4	OFF	ON	ON	OFF	135° 
5	OFF	OFF	ON	OFF	180° 
6	OFF	OFF	ON	ON	225° 
7	OFF	OFF	OFF	ON	270° 
8	ON	OFF	OFF	ON	315° 

Figure 10: Stepper Motor Operating Sequence and Simplified Schematic

1.3 Radio Control Servos

RC Servos combine actuation, sensing and control into one standardised package (right). The mechanical output is typically a rotation between 0 and 180^o, transmitted through a wheel or crank. Servos can be readily purchased in a wide variety of sizes and torque ratings. The interface is simple: the red wire carries 5V power; black is ground; and white is the command signal. The latter consists of a 5V pulse sent at most every 20ms. The pulse length encodes the desired rotation: 1ms for 0, 2ms for 180^o, or anything in between. Electronic modules to drive servos from computers are readily available, and many robotics-aimed computing like the Arduino or PICs have standard programming libraries for servo control. Two things to watch out for: servos can draw quite a lot of current, so find a decent power source for the red lead; and sometimes, if you ask two much of them, they can burn out.

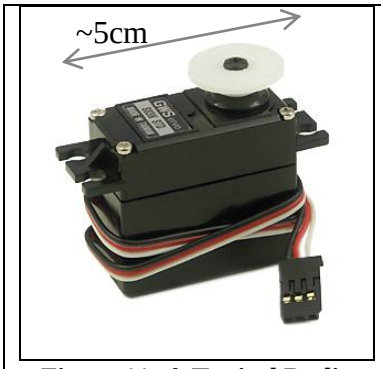


Figure 11: A Typical Radio Control Servo

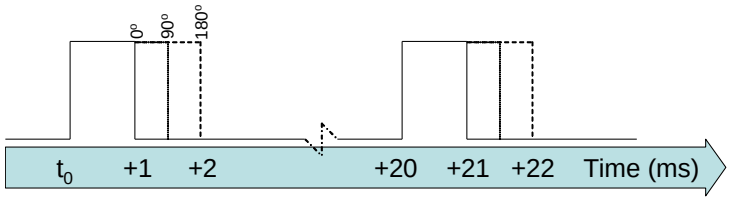


Figure 12: Command Signal Layout for Radio Control Servo

2 Sensing

2.1 Open-Loop Prediction

In theory, you could determine the position of an axis[1] by recording how long the motor runs, and using knowledge of the speed of running to convert that to position. In practice, the behaviour of DC motors is too unpredictable for this to be very useful: their speed varies considerably with load. Stepper motors can be used in this way, but use caution: they can "slip" if too much speed, torque or power is requested. Also, if adopting this route, consider the implications of losing track of the position due to a failure of software or hardware – there may be no way back.

2.2 Microswitches

These are small, sensitive switches requiring very little actuation force. The picture below right is an enlarged image showing a switch fitted with a roller for motion control. Typically, a tab is fixed to the moving part to depress the roller as it passes a particular location (below left), providing a digital signal that the axis is in a particular location. The circuit below right shows how to interface a microswitch with a typical 5V digital input, *e.g.* a TTL chip input or Arduino digital input.

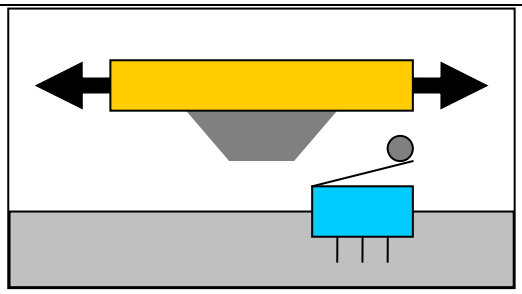


Figure 13: Example of Microswitch Installation

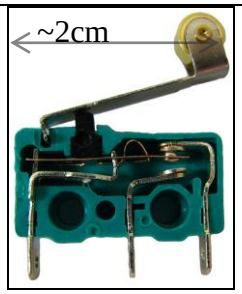


Figure 14: A Typical Microswitch

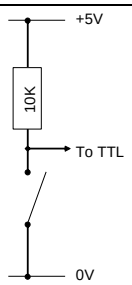


Figure 15: Circuit for Digital Input from Microswitch

Microswitches are cheap and simple to interface, and particularly efficient when accuracy is required at only a few key locations. However, the activating tab needs to be carefully positioned – one group in a previous year were still adjusting microswitches during their test to get reliable indication, and a switch failure led to severe problems. The size of the tab it also tricky – too long, and accuracy suffers as the tab is depressed across a large region; too short, and the axis can overshoot. Finally, beware contact bounce in the switch. In practice, mechanical switches do not close cleanly, but they "bounce" off and on for a few microseconds. This can be long enough to fool the electronics into thinking that the axis has passed and moved on. As a simple fix, try a capacitor across the contacts. If you prefer software, wait for 3 (or so) successive "on" readings from the switch before triggering the event.

[1] For brevity, the term "axis" is used throughout to refer to an independent controlled degree of freedom, *e.g.* each joint of a robot arm.

2.3 Light Gates

A light gate is an LED, often infra-red, and a matched phototransistor or photodiode, mounted in a plastic package and separated by a gap – see enlarged figure, below left. When there is something opaque (to IR) in the gap, the light path is blocked and the phototransistor switches off. When the gap is empty (and the LED is on, with current flowing from anode (a) to cathode (k)) then the phototransistor is turned on and current flows from the collector (c) to emitter (e). As with the microswitch, a small tab shutter is attached to the moving axis to provide a digital indication when at a key location.

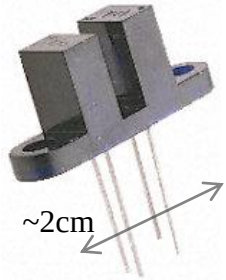


Figure 16: A Typical Light Gate

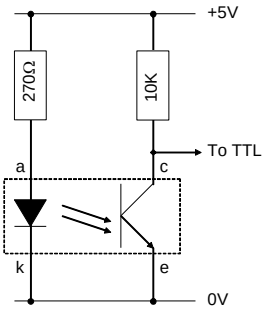


Figure 17: Circuit for Digital Input from a Light Gate

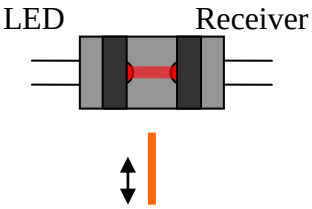


Figure 18: Schematic of Light Gate Operation

Light gates are comparable in price to microswitches and only slightly less simple to use (although noone has tried in EMAT20002). You need an extra wire to power the LED, but otherwise the interface is the same. The method could be less sensitive to misalignment than a microswitch and it is free from contact bounce. Light gates are also better suited to faster pulsing, *e.g.* roller ball mice used light gates with serated wheels to measure movement.

2.4 Potentiometers

Familiar from electronics as variable resistors (below left), you could attach a potentiometer to an axis to determine its position – as is done in servos. They have the advantage of offering continuous monitoring of position, not just at a few chosen locations. However, they need analog interfacing, can give noisy signals when moving, and can introduce friction. You can also buy linear potentiometers (below middle).

Figure 19: A Typical Potentiometer

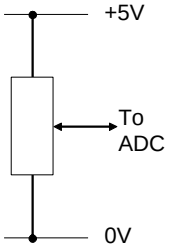


Figure 20: Circuit for Analog Input from Potentiometer

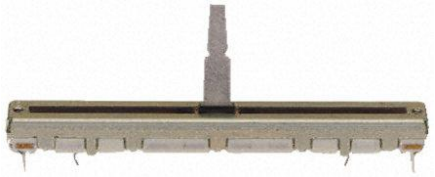


Figure 21: A Linear Potentiometer

Figure 22: A Typical Rotary Encoder

2.5 Digital Encoders

Similar in size, shape, and mechanical operation to a potentiometer (above left), an encoder uses a different operating principle. Inside, it has multiple light gates and a patterned wheel: by reading which lights are blocked, you can determine which sector of the wheel is aligned with the gates. The output is typically a digital signal. They offer less friction and noise than potentiometers, but require more complex electronics and they're generally more expensive. In particular, an *incremental encoder* only has two segments on the wheel, and you have to do your own counting. As with potentiometers, linear versions are available.