

DC Motor Guide

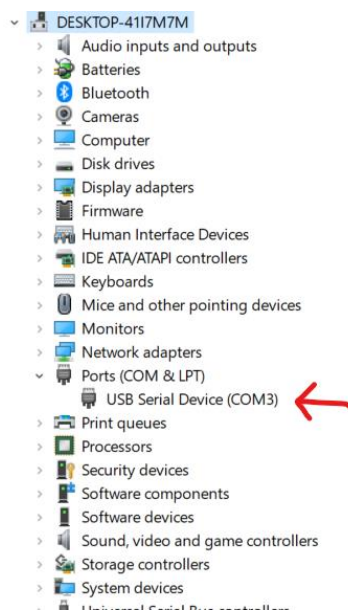
Setup

Plug in the Arduino to the PC with a USB cable. Plug the power adaptor into the 12V barrel (not the one on the Arduino board).

Finding COM Port

- On Windows

With the Arduino connected to the computer, open Device Manager (from search bar or Windows settings), and look for the USB Serial Device under the Ports tab. This will show you the COM port that the Arduino is currently assigned.



- On Linux

Open Matlab and run the 'seriallist' command. This will return a list of available serial ports. The Arduino is usually assigned to /dev/ttyACM0 on Linux.

Using The run Function

Calling the run function will run the DC Motor system. It takes 3 parameters:

- u – The input vector, containing an input value between -1 and 1 (-100% and 100% of max torque) for each sample.
- portNum – A string containing the COM port number.
- Ts – (Optional) The sampling time (in seconds) to run the system with. This is set by default to 5ms, so you can omit this if you don't need to change the settling time.

The function returns 3 vectors:

- vel – A vector containing the velocity of the motor in RPM.
- alpha – A vector containing the angle of the motor shaft in degrees.

- t – A time vector

These 3 vectors have the same length as u.

Here's an example of how the function is used:

```
u = [zeros(10, 1); 0.5*ones(1000, 1)];  
[vel, alpha, t] = run(u, '4');  
plot(t, vel);
```

Occasionally, the system will not run when calling the run function. If this occurs, a second attempt at running the system will usually function correctly. If the second attempt fails, ask a teaching assistant for help.

Considerations for u and Ts

The run function clears the USB read buffer when it first runs, so in order to let it do so, it's best to prepend your u vector with about 10 zeros.

I added a few helper functions for generating input vectors:

ustep(n) – generates a step signal with n – 10 samples. The first 10 samples are 0.

urand(n) – generates a random staircase input vector. Contains steps of random values with a width of 100 samples each.

Ex:

```
u1 = 0.5*ustep(1000);  
u2 = urand(1000);
```

The output vectors (vel and alpha) are only as long as u, so if you want to run an impulse, you need to append enough zeros to the end of u to see the entire impulse response.

The minimum Ts (for the windows version) is 2.5ms. Any lower than that and the true sampling time won't be stable. Here's an example of setting a 10ms sampling time:

```
[vel, alpha, t] = run(u, '4', 10e-3);
```