EM Matlab Exercises

The EM functionality is controlled via a COM port connection to the computer. We have provided a library of functions to allow the various aspects of the EM to be investigated and controlled. The various EM function scrips can be found inside the Primitives folder. The scripts require a global variable to be declared in your project to set the COM port number.

*global EM\_USB;  
% Find the COM port on your PC where the Matrix EM control unit is plugged  
% in and use it here instead of 'COM17'  
EM\_USB = serial('COM17','Baudrate',115200,'Databits',8);  
fopen(EM\_USB);*

*% Main program resides here*

*fclose(EM\_USB) % Close the COM port*

# Exercise 1

Using Matlab create a program that will control the DC motor.

Run the motor at a set voltage then create a loop to vary the load to the motor logging the speed and torque.

Plot the results to a graph.

Perform similar plots for speed vs current, torque vs current.

Repeat the tests for different voltages.

Repeat the tests for other types of motor.

Here is some example code.

*global EM\_USB;  
% Find the COM port on your PC where the Matrix EM control unit is plugged  
% in and use it here instead of 'COM17'  
EM\_USB = serial('COM17','Baudrate',115200,'Databits',8);  
fopen(EM\_USB);*

*SetMotorType (0); % DC Motor*

*pause(0.1);*

*motor\_voltage = 10; % 10V DC*

*% This just tests the comms to the control unit before we start*

*api\_version = GetAPIVersion();*

*disp(['api\_version = ' , num2str(api\_version)]);*

*% Check the Dynamometer 'motor located' switches*

*switch\_state = GetSwitches ();*

*if switch\_state >= 1*

*disp('==== DC MOTOR TEST START ====')*

*SetDynoLoad(0);*

*index = 1;*

*SetActive(1); % Motor ON*

*pause(0.3); % Wait for the motor to spin up*

*% Perform the sampled measurements in a loop*

*for loop = 0:255*

*load\_cell\_torque\_range(index) = GetLoadCell(1);*

*encoder\_speed\_range(index) = GetRPM();*

*index = index + 1;*

*SetDynoLoad(loop);*

*pause(0.1);*

*end*

*% Sampling over, you can turn off the motor and close the COM port*

*SetActive(0); % Motor OFF*

*fclose(EM\_USB)*

*% Plot the RPM against Time*

*curent\_torque\_plot = figure;*

*figure(curent\_torque\_plot);*

*movegui(curent\_torque\_plot , 'northwest');*

*hold on*

*p2 = polyfit(encoder\_speed\_range, load\_cell\_torque\_range , 1); % line of best fit*

*y2 = polyval(p2, encoder\_speed\_range);*

*plot(encoder\_speed\_range, load\_cell\_torque\_range, 'ro', encoder\_speed\_range, y2, '-r' );*

*current\_speed\_polynomial = poly2sym(p2,sym('T'));*

*current\_speed\_equation = sprintf('Speed = %s',current\_speed\_polynomial);*

*disp(current\_speed\_equation);*

*grid on*

*legend('Samples','Best fit line','Location','southeast');*

*xlabel('Speed, RPM');*

*ylabel('Torque, mNm');*

*title(['PM DC Motor Speed v Torque @ ', num2str(motor\_voltage) , 'V']);*

*hold off*

*else*

*disp('==== Motor not fitted in Dynomometer ====');*

*end*

# Exercise 2

In an industrial context you will often need a motor to be running at a constant speed. Using a DC motor, the output motor voltage as your control signal and the speed RPM as your feedback create a closed loop system script to run the motor at a defined speed.

Vary the load to the motor using the SetDynoLoad API function and confirm that the speed remains consistent.

Can the speed be made consistent for all values of load?

Repeat for other types of motor.

# Exercise 3

A motor cannot go from not moving to moving at a certain speed in an instant. Create a script to allow the motor speed to be ramped up and ramped down to achieve the desired speed and reduce excess current usage.

How much of a difference does ramping make on the efficiency of the motor?

Which is better a linear ramp profile or an exponential ramp profile?

Repeat for other types of motor.