Dynamics and motion control

Workshop A, Parameter identification and control of a DC motor.

1. Introduction

Written report

The report should not be very thick. A good report includes exactly the information needed for the reader to understand the results and nothing more. You do not have to include all the steps in the report when you derive controllers. The important parts are the results in terms of figures with plots of *appropriate* signals for each design phase, experiment, etc. Appropriate signals can be all or some of the signals, position, velocity, voltage and command signals. If necessary, you may also include a Bode plot of frequency response, pole-zero map, and other relevant figures. A short discussion of the results shown in the plots should also be included. **All plots must have** a **white background**.

There are two levels of the problems below. If you only do the level 1, you cannot get the highest grade on the workshop. Solving both level 1 and level 2 problems, however, is not equal to getting an A. The quality has to be high throughout the whole report.

Tools

In the workshops of this course, you are going to run the Simulink in "**External Mode**". The external mode simulation establishes communication between the Simulink on the computer and the target hardware. The target hardware runs the executable file created by the code generation and build process. In external mode, you can tune the parameters in real-time, monitor the signal data from the hardware and record the data.

For the target hardware, you are going to use a powerful microcontroller board: LAUNCHXL-F28379D. This board is a real-time control MCU from Texas Instrument. It has 2 CPUs and the frequency is 200MHz.

2. Learning how to work with the tools

Work through the document "Workshop Tutorial".

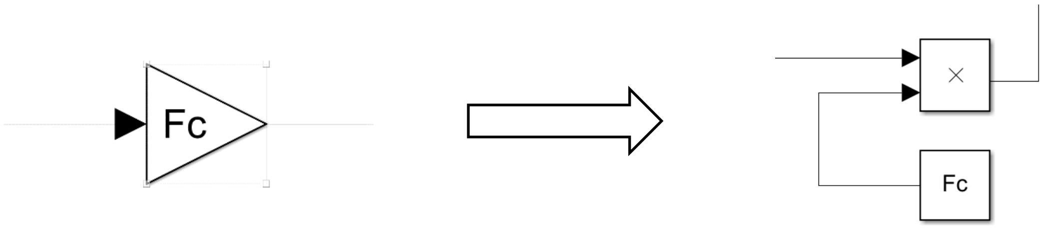
3. Parameter identification

The simplest (not the best though) way to identify the model parameters is to make a step response of the model and the real motor and compare the results. A step excites all frequencies of the process. Two main behaviours can be used to find the parameters. They are,

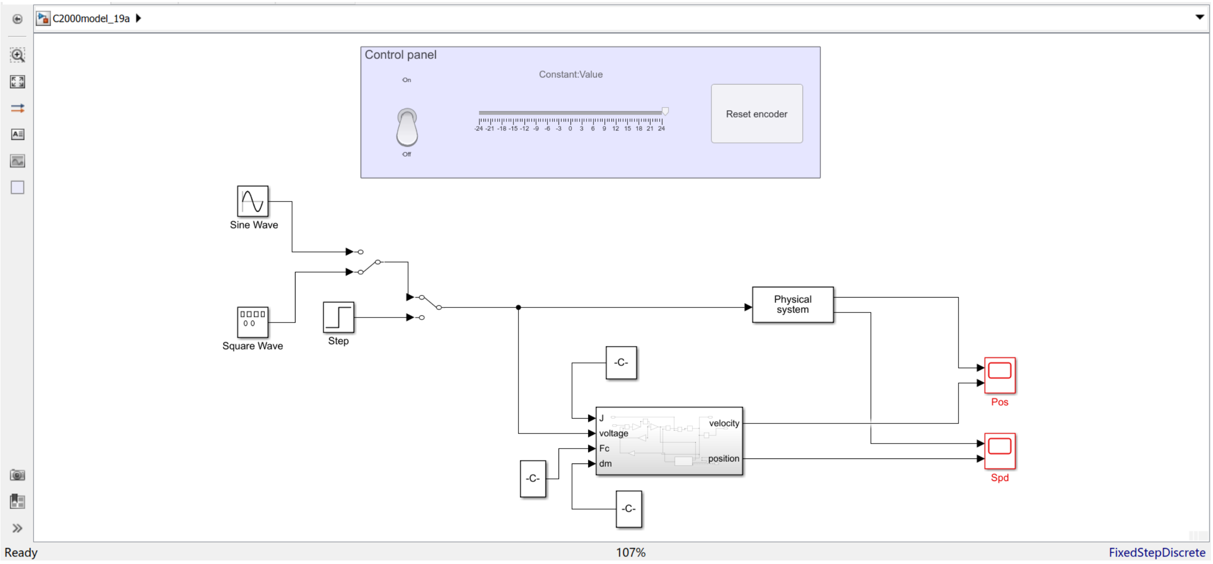
- Steady state velocity gain
- Time constant

The electric parameters can be taken from the datasheet, they should be correct. The unknown model parameters are the *inertia* and the *friction*.

Copy the Simulink blocks of your dc-motor model (without the inductance) into the Simulink file that you already have used in Section 2. You may need to change something before you run the file. When running in the external model, sometimes it is not possible to change the value of “Gain”. Thus, you need to replace the “Gain” with a multiplication. An example is shown below:



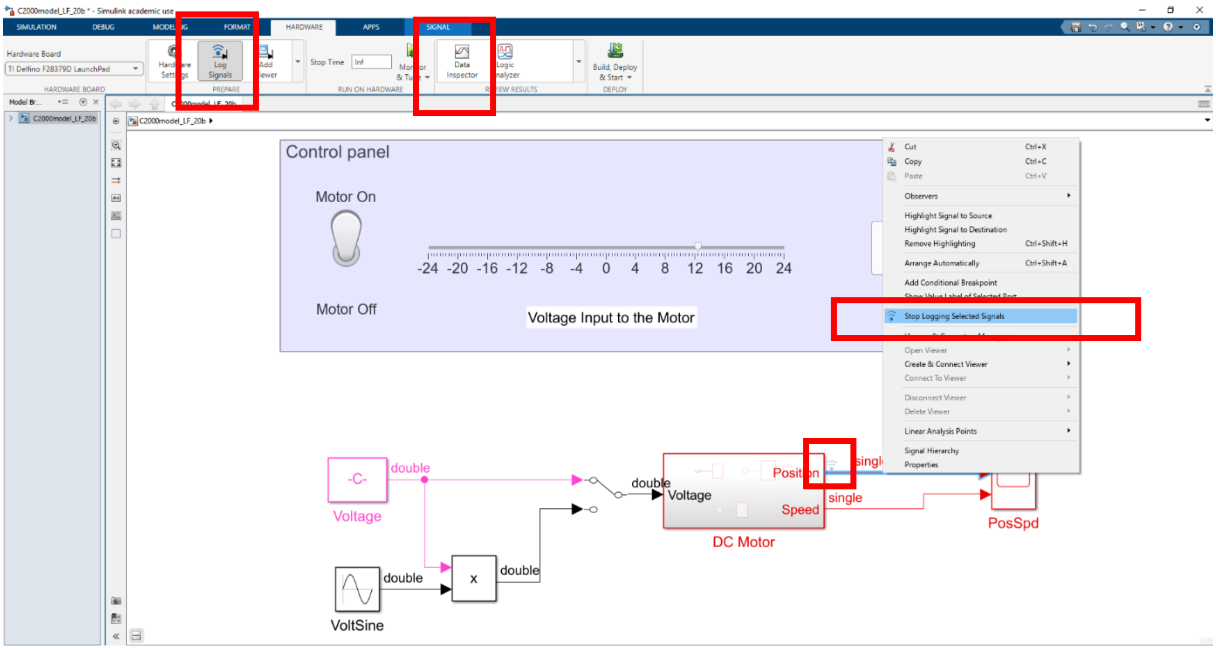
Furthermore, to make it easier, combining blocks into subsystems is highly recommended. You should have different kinds of signals as input and switches to change the input signal. You should also connect the velocity signals from the motor model and the encoder to one scope and compare them. An example is shown below:



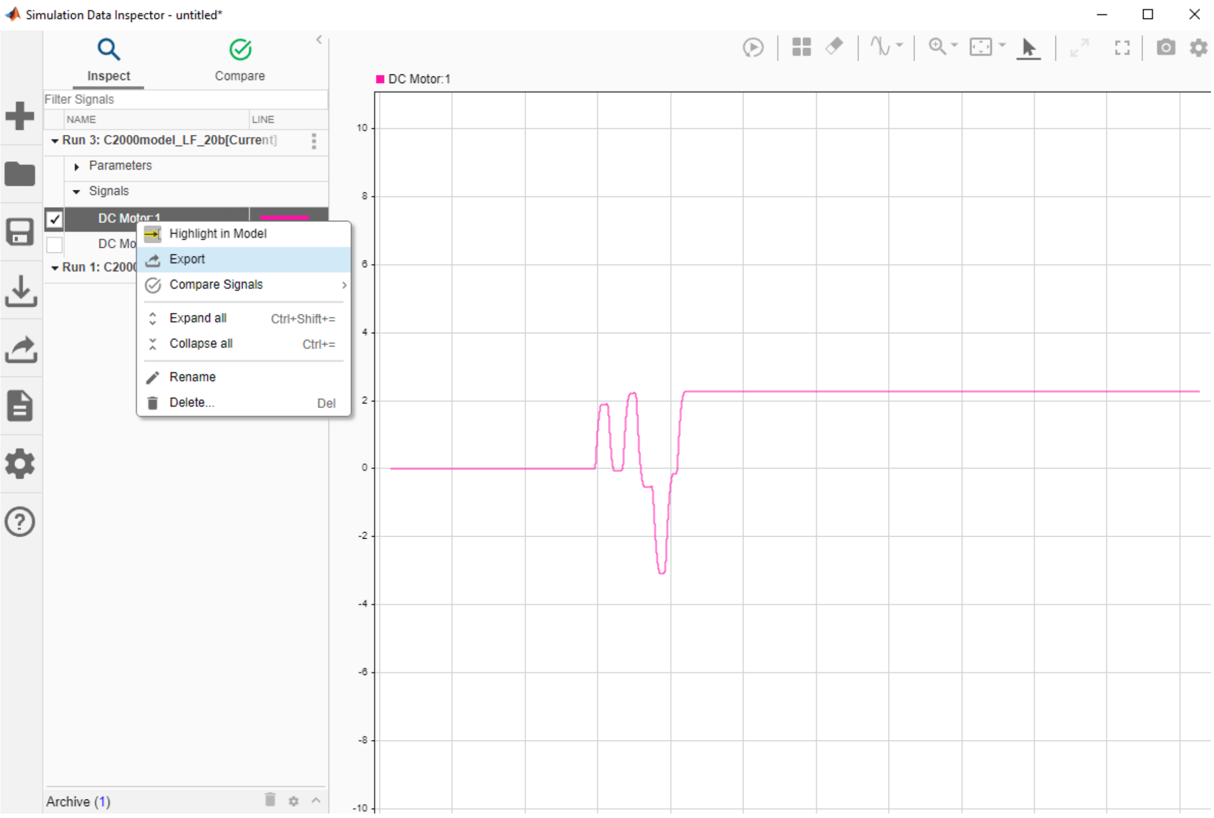
Run the “initPar.m” script to initialize encoder parameters.

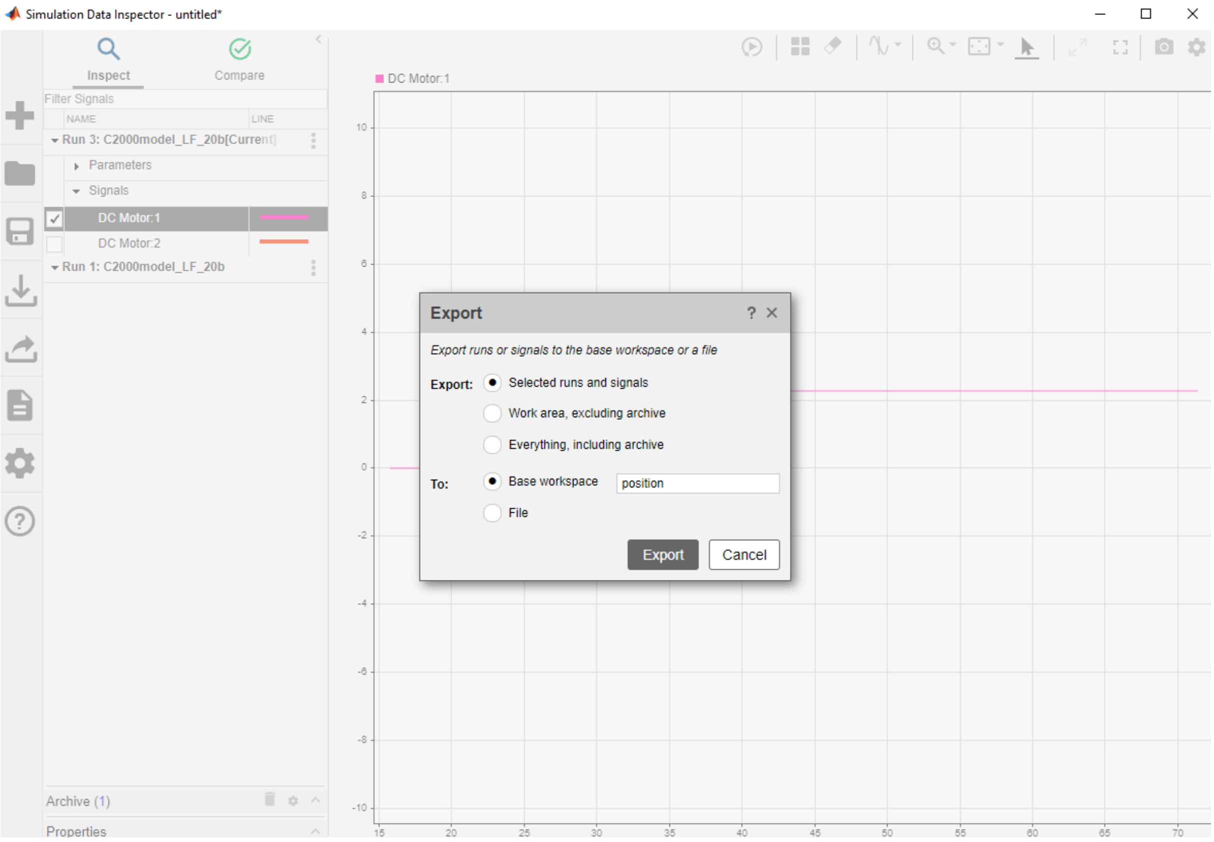
Work with the motor parameters until a good fit between the model speed and the measurement speed from the encoder is achieved. Save the experiment and the identified parameters. Observe that the polarity of the motors can vary. If the rotation direction of a motor is wrong, you can fix it by inverting the polarity in the Simulink model, i.e., a gain of -1 on the voltage signal.

For recording the signal, you should make some adjustments before running the file. One way is to log the data. Right-click on the output line. Click the “**data inspector”** or just click the **Wi-Fi** button.

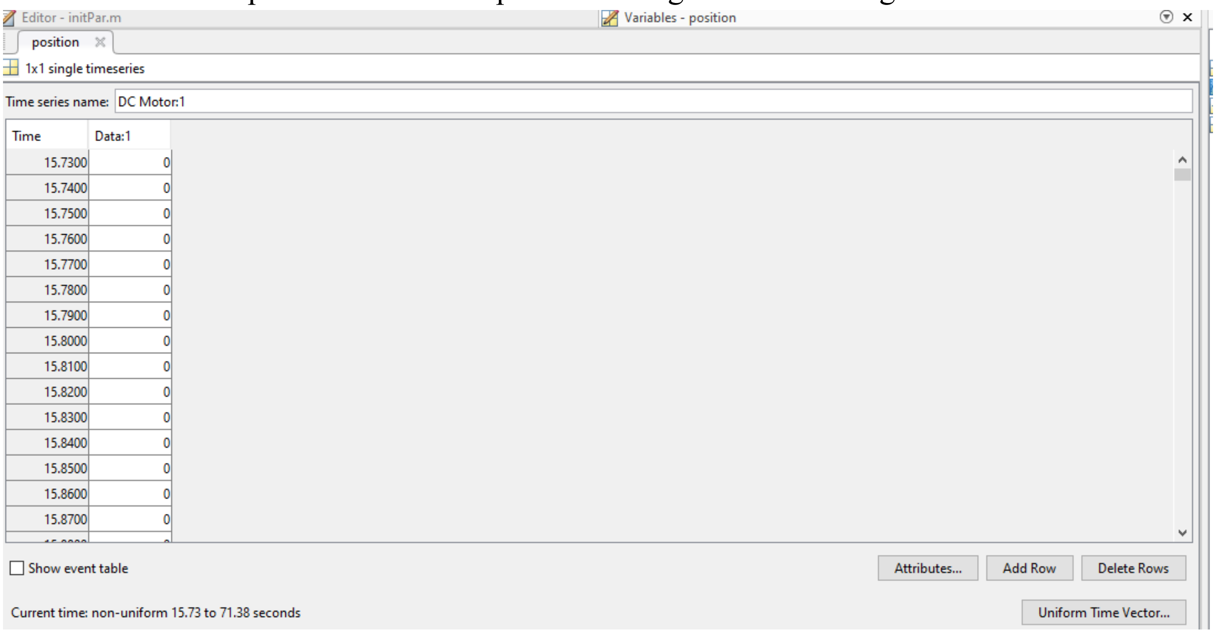


In the data inspector, choose the signal you want to use. Right-click and choose “**Export**”.

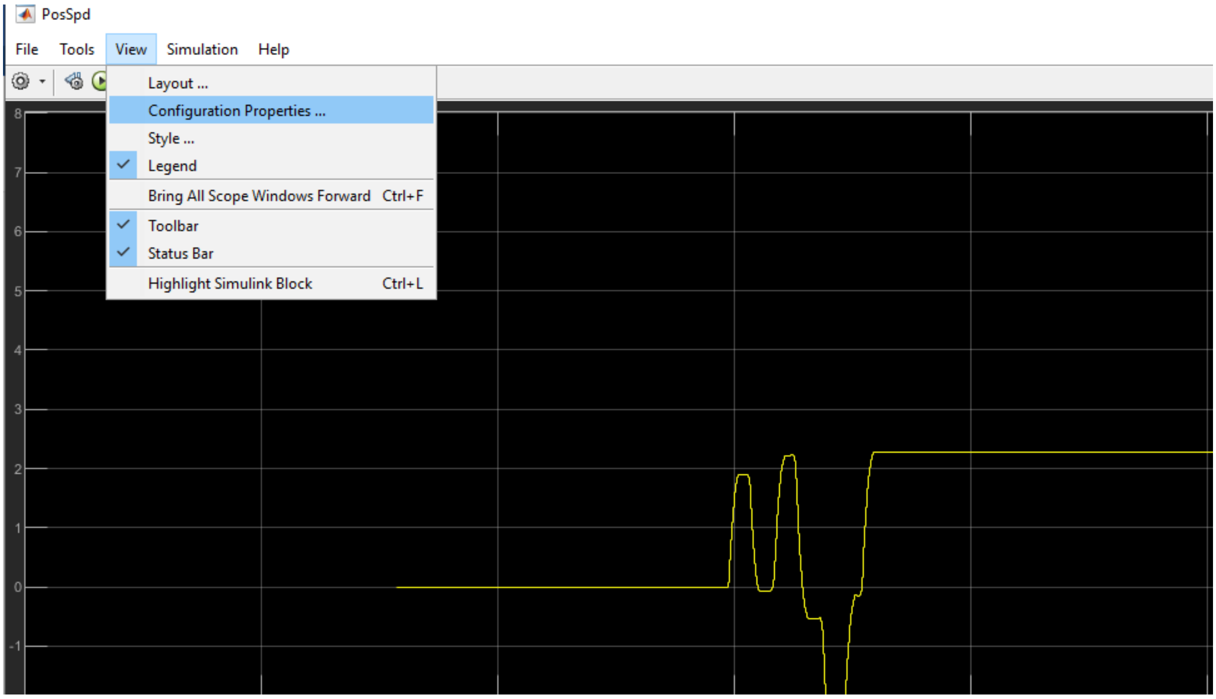




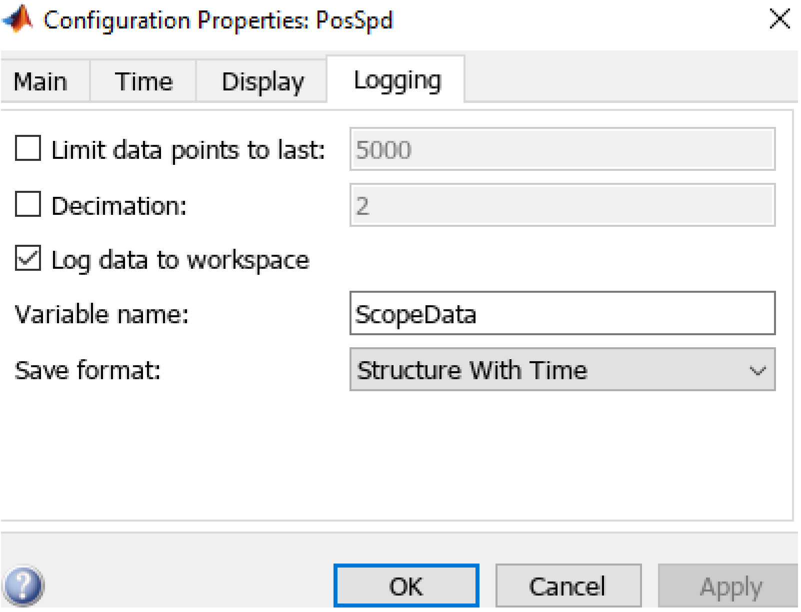
The data will be exported to the workspace including the time and signals.



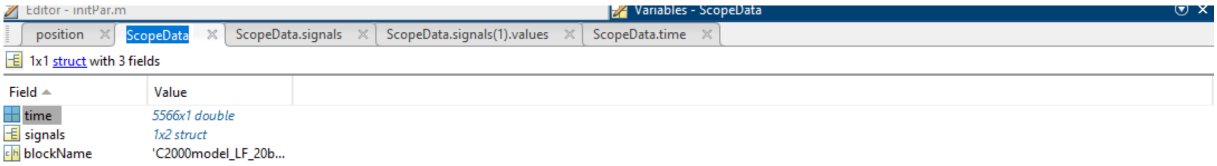
Another method is to log data in the scope. Click **View** and **Configuration Properties**.



In the logging, select **Log data to workspace** and name the variable.



The time and signals in the workspace are separate as the type of the scope data is structure.



Level 1.

Identify the parameters: linear friction and inertia.

List the two identified parameters in a table. Show the velocity signals from the model and the real motor in the same plot for step response and any other input signal that makes sense to show.

Level 2.

Use the nonlinear friction model $M_f = d\omega + F_c sgn(\omega)$ in your Simulink model. Identify the key parameters: d and F_c through motor experiment.

List the identified parameters in a table. Show the velocity signals from both model and motor from at least two step inputs with different amplitudes and a sinewave input with low amplitude. Discuss any improvement from the linear model in Level 1.

4. Controlling the motor

Velocity control

Design and implement a controller that controls the velocity of the real motor. The denominator and the numerator of the controller must have the same order. Note that you need the good parameters, linear friction coefficient and inertia from the identification part first. The controller may be designed in continuous-time or discrete-time but must be implemented as a discrete-time controller (Z transfer function). You may select any appropriate sampling period. In the report, you must show that the specifications are fulfilled on the real motor.

List the closed-loop poles and sampling period in a table. The poles can be given in either the S-domain or Z-domain. Write the control law in the Z transfer function.

Level 1.

Test the velocity controller on the real motor with the following three cases. All three cases must use the same controller with the same parameters. For each case, plot the speed reference signal, model velocity, and the motor measurement velocity in one figure, and plot the voltage in another figure.

1.)

- Step response to 50 *rad/s* with as fast as possible response time.
- No overshot
- Maximum steady-state error 1 *rad/s*.

2.)

- Sine wave reference signal $\omega_{ref} = 50\sin(2\pi \cdot 0.2t)$.

3.)

- Same sine wave reference signal $\omega_{ref} = 50\sin(2\pi \cdot 0.2t)$ and at the same time brake the motor slightly by applying a force with your finger on the flywheel. Apply a force so that the voltage saturates.

Position control

Design a proper controller that controls the position of the real motor. The position controller may be designed in either continuous time or discrete time but must be implemented in discrete time (Z transfer function). In the report, you should give the control law and show that the specifications are fulfilled on the real motor. Depending on the design, it may be necessary to include an anti-windup in the controller.

For both questions below, list the closed-loop poles and the sampling period in a table. Write the control law as a Z transfer function, plot the position reference signal, model position, and the motor measurement position in one figure, and plot the voltage in another figure.

Level 1.

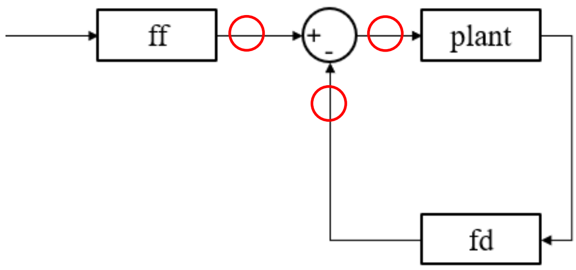
Position control system specifications

- Step response from 0 to 10 *rad* with rise time $t_r = 0.5s \pm 0.05\,s$. Rise time is here defined as 95% of the reference step.
- Maximum overshot 2% of the step amplitude.
- Maximum 0.5% steady-state error
- Any sampling period can be used.

Level 2.

With the same specifications as in Level 1, design the controller by *discrete-time design method* and choose an as-long-as-possible sampling period. Compare the results with a controller that is approximated to discrete time from a continuous-time design. Include a discussion on what limits the sampling period.

Hint: Before you do level 2, make sure that everything works fine in level 1. Do not only check the velocity/position signal when doing level 1. You should also check these points in the control loop (see the circled parts in the picture below). Check whether the signals in these points are reasonable. (Why?) If in one of these points, the signal is not reasonable, it may also achieve a good control performance in simulation, but it will be a problem when implementing the controller in the hardware. (Why?)



Furthermore, when you are doing level 2, it is **not necessary** to put a motor model in the Simulink file.