# Exercises belonging to: Trajectory Planning, Setpoint Generation and Feedforward for Motion Systems

Paul Lambrechts

*Senior Application Engineer, The MathWorks Benelux.*

September 27, 2006

# make4 en profile4

1. Use make4 to plan a trajectory with the following limitations:
   d=1000, j=45, a=5, v=1, p=1.
   See help make4, do not specify any optional parameters.
   In what time will the maximum velocity be achieved?
2. Use profile4 to calculate and plot this trajectory: specify acc = 0.01.
3. Build a Simulink model using a 'From Workspace' block with parameter dj (calculated with profile4.)
   Use 4 integrator blocks to determine jerk, acceleration, velocity and position; use a scope block with 5 separate inputs to show all profiles.
   What goes wrong if you switch off zero crossing detection in the 'From Workspace' block?
   Try to improve results by adjusting the simulation configuration parameters.
4. Discretize the model using the forward Euler method with sampling time Ts = 0.001s: in Simulink you can do this in two ways!? (try both.)
   What happens with the profiles?
5. Use make4 to plan a discrete time trajectory with Ts = 0.001s.
   In what time will the maximum speed be achieved?
   Use profile4 to calculate and plot this trajectory: note that you must specify the bound on djerk as calculated by make4, also note that you should take acc = 0.001.
   What are now the results in Simulink?

## 4th order feedforward

1. Open the library ‘motion;’ open ‘General;’ open ’example 1.’
   Redo previous questions 1 and 5 using this Simulink scheme by filling in parameters (double click the ’profile generator’ block and the ’feedforward’ block.)
2. Select the ‘4th order continuous’ block for feedforward calculation and ‘look under mask.’
   Where are the q parameters calculated?
3. Open ‘example 2’ and examine the motion system block.
   Specify the trajectory as given above in the ‘profile generator’ block
4. Using only feedback: what is the error response.
   Also compare the feedback force with the calculated feedforward force: This is a check to be certain that the Feedforward controller is implemented correctly.
5. Now add the feedforward force to the actuator input and run the simulation.
   Impressed? (remember: it is only a simulation.)
6. Compare with the results of Rigid Body feedforward. Also consider the open loop responses.
7. You can analyse the sensitivity to parameter variations by changing the parameters in the motion system block. Also consider the results with a 6th order motion system (see ‘block choice.’)

## discrete time and real-time

1. In ‘example 2:’ change the profile generator and feedforward calculation to discrete time with sampling time Ts = 0.001 s.
   Run an open loop simulation with the parametrized 4th order motion system.
   What goes wrong here, and how do you fix it?
2. Open ‘example 3.’ With this example you can examine several aspects of discrete time implementation.
3. Open ‘example 2:’ the discrete time version of the ‘Profile Generator’ block together with the ‘Feedforward’ block could be used to generate a trajectory and feedforward in real-time (using Real-Time Workshop.)
   When considering using this for a pick-and-place machine in an industrial environment, name at least three reasons why this would be unsuitable.
4. Open ‘Real Time’ block in library ‘motion’ and open example.
   Define Ts = 0.001 and start simulation.
   Look what happens if you change step size and/or bounds.
5. What could happen if the step size or a bound is changed during execution of a move, and how is this prevented?
6. Open the model ‘bad_idea.’ Why is it?

## answers

For professors and teachers on request:

email: Paul.Lambrechts@MathWorks.NL