Guide for Simulation Code

Auto-Tuning of Model-based Feedforward Controller by Feedback Control Signal in Ultraprecision Motion Systems

Table of Contents

1 Introduction

1.1 Simulation in continuous time domain V.S. simulation in discrete time domain

As it has been pointed out in the manuscript, with the transfer function of zero order holder considered, results in continuous time domain are the same with those in discrete time domain. Therefore, the simulation model is constructed in continuous time domain. Moreover, instead of directly using the transfer function of zero order holder, half sampling period time delay is used to approximate zero order hold effect, see section II-A in the manuscript.

1.2 Trajectory planning

The trajectory planning algorithm is from the paper ***Lambrechts P, Boerlage M, Steinbuch M. Trajectory planning and feedforward design for electromechanical motion systems[J]. Control Engineering Practice, 2005, 13(2): 145-157.*** The authors have generously opensourced the MATLAB implementation of the trajectory planning algorithm, which can be found at:

https://ww2.mathworks.cn/matlabcentral/fileexchange/16352-advanced-setpoints-for-motion-systems?s_tid=prof_contriblnk

1.3 Simulation environment

The simulation model is constructed in MATLAB 2017Rb.

2 Step by step guide for running the simulation code

2.1 Initialization

(1) First, run script **RunThisFirst.m**

(2) Second, run script **initiateSimulation.m**

2.2 Feedforward tuning

2.2.1 Tuning of acceleration feedforward

(3) Run script **RunSim.m**, the initial feedforward coefficients should be all zero.

(4) Run **code section A** in **feedforwardAutoTuning.m**, tuned acceleration feedforward can be attained.

2.2.2 Tuning of jerk feedforward

(5) Set acceleration feedforward coefficient with the value attained in (4), then run script **RunSim.m**

(6) Run **code section B** in **feedforwardAutoTuning.m**, tuned jerk feedforward can be attained.

2.2.3 Joint tuning of acceleration and jerk feedforward

(7) Set jerk feedforward coefficient with the value attained in (6), then run script **RunSim.m**

(8) Run **code section C** in **feedforwardAutoTuning.m**, jointly tuned acceleration and jerk feedforward can be attained.

2.2.4 Joint tuning of acceleration, jerk and snap feedforward

(9) Set acceleration and jerk feedforward coefficients with the values attained in (8), then run script **RunSim.m**

(10) Run script **preprocessingUfb.m** to preprocess feedback control signal.

(11) Run **code section D** in **feedforwardAutoTuning.m**,

(12) Tuning finished.

2.3 Post-processing

(13) In the MATLAB command line, run command

```
>> edit postprocessing\postProcessing.m
```

The **code section A** of **postProcessing.m** contains code to plot tracking error together with derivatives

of reference trajectory; The **code section B** contains code to plot feedback control signal together with derivatives of reference trajectory. Every time **RunSim.m** is executed, **postProcessing.m** can be executed to plot corresponding tracking error and feedback control signal.

References

[1] Lambrechts P, Boerlage M, Steinbuch M. Trajectory planning and feedforward design for electromechanical motion systems[J]. Control Engineering Practice, 2005, 13(2): 145-157.