# Discretization of Continuous Controllers - TP correction

**Thao Dang**

**VERIMAG, CNRS (France)**

# Exercise: Application to LEGO robots

1. Compute the continuous-time transfer function of the open loop $F_{BO}(s)$

2. Estimate the crossover frequency $\omega_c$ of $F_{BO}(s)$ using the function "margin" in matlab.

3. Choose sampling period $T_e$ according to the rule: $\omega_c T_e$ **is between** $0.05$ **and** $0.14$

4. Discretize the controllers using one of the approximation methods and the chosen sampling period. (Note that instead of replacing the whole continuous-time controller with its discretized version, we can replace only its components containing continuous-time blocs, such as the integrators $\frac{1}{s}$)

5 Add the digital anti-aliasing filter

# Determining a sampling period (1)

The transfer function of the orientation $\theta$ is:

$$H_\theta(s) = 1/(ls)$$

where $l$ is the distence between two wheels The transfer function of the orientation

$$C(s) = ki_\theta/s + kp_\theta$$

The open loop transfer function:

$$H_OP(s) = (kp_\theta s + ki_\theta)/(ls^2)$$

Note: from the crossover frequency $wc$ of the open loop, we can deduce the bandwidth $wb$ of the closed loop $wc \leq wb \leq 2wc$.

# Determining a sampling period (2)

```
%Matlab code
%transfer function C(s) = ki_teta/s + kp_teta
%transfer function H_teta(s) = 1/(ls)
%transfer function
%H_OP(s) = (kp_teta*s + ki_teta)/(ls^2)

sysbo = tf([kp_teta ki_teta], [ 1 0 0])

[g p f wc] = margin(sysbo)
%wc  "crossover frequency" (rad/s)

%Te = sampling period
Te=0.05/wc;

%we = sampling frequency
we=2*pi/Te
```

# Adding an Anti-Aliasing Filter

Specifications of the filter:

- passband corner frequency $\omega_p$ and ripple $R$ (in decibels) for the frequencies in $[0, \omega_p]$
- stopband corner frequency $\omega_s$ and attenuation $A(dB)$ at the frequency $\omega_s$

Note that the Nyquist frequency $\omega_N = 2\omega_e$ ($\omega_e$ is the sampling frequency)

```
%Matlab code
[n,wo] = buttord(w_p/w_N,w_s/w_N, R, A);
% Returns n = order of the filter; wo=cut-off frequency;
[b,a] = butter(n,wo);
freqz(b,a,512,1000);
title('Butterworth Lowpass Filter')
```

# Some rules of thumb

- The stopband frequency is generally at most $\frac{1}{2}$ the sampling frequency, i.e. $\frac{\omega_e}{2}$
- Because of the approximate nature of the method, to assure good performance, one can choose faster sampling (but the resulting implementation is more costly).
- 