INTO-CPS

20-sim Tutorial

Ken Pierce, Newcastle University

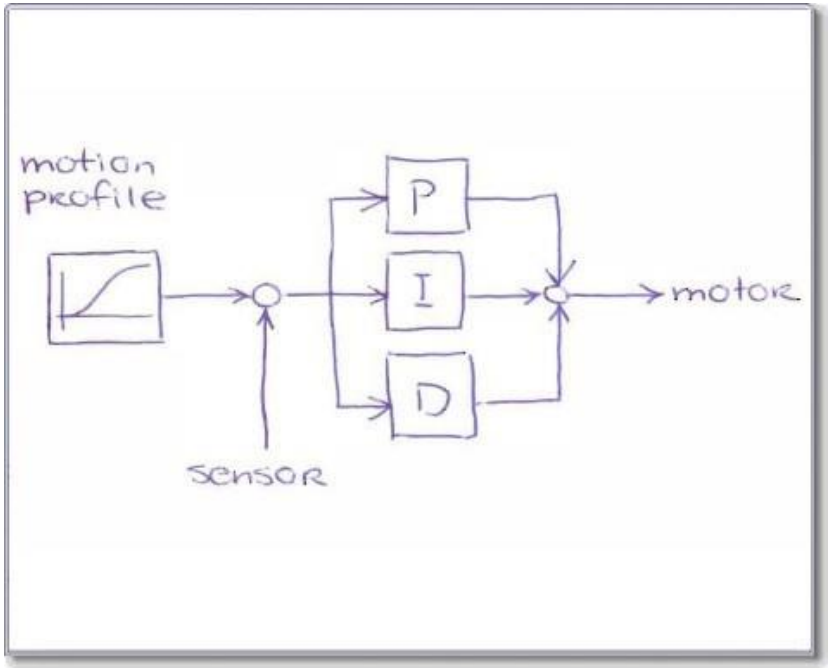
NII 大学共同利用機関法人 情報・システム研究機構 国立情報学研究所 National Institute of Informatics

AARHUS UNIVERSITY

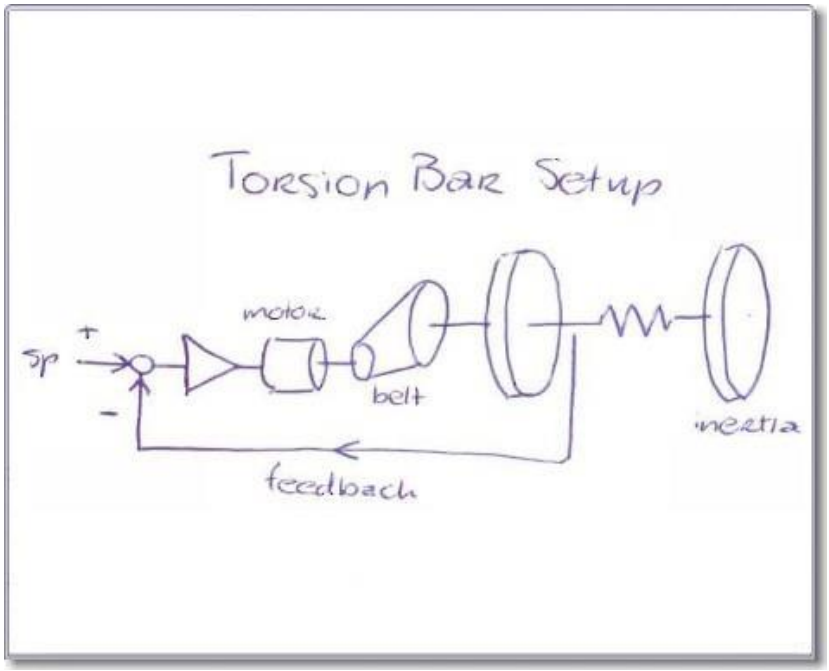
Newcastle University

Horizon 2020 Programme

From Sketch to ...

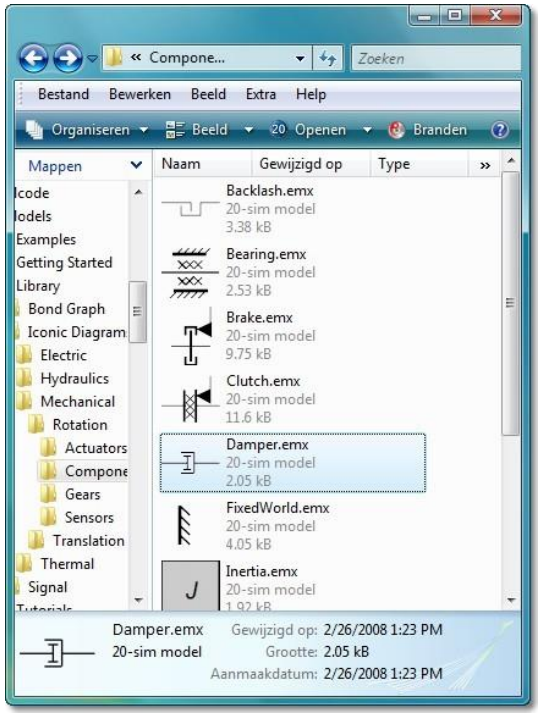


Controller

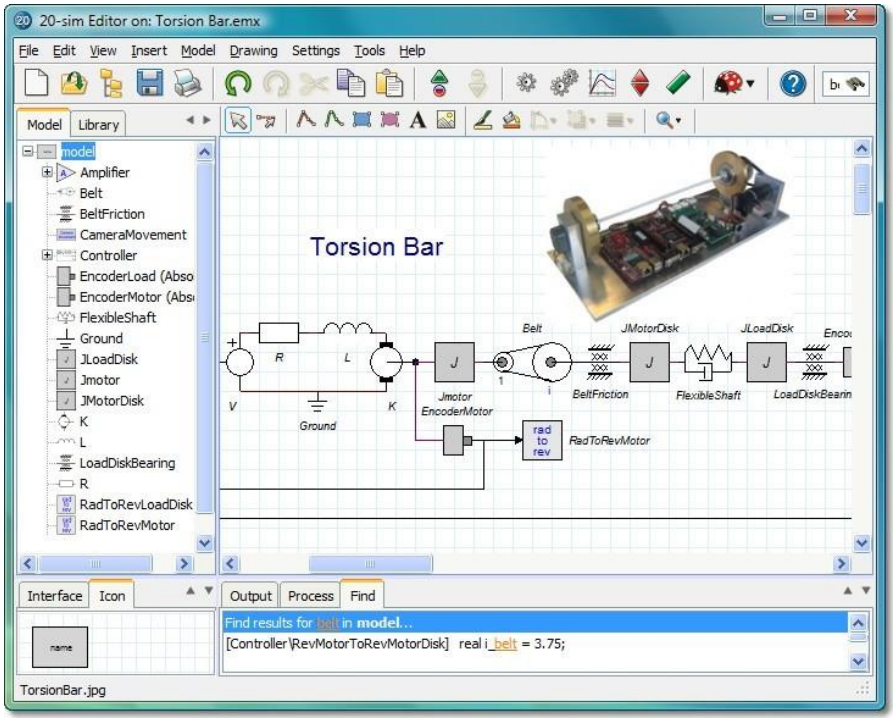


Physical System

Modeling

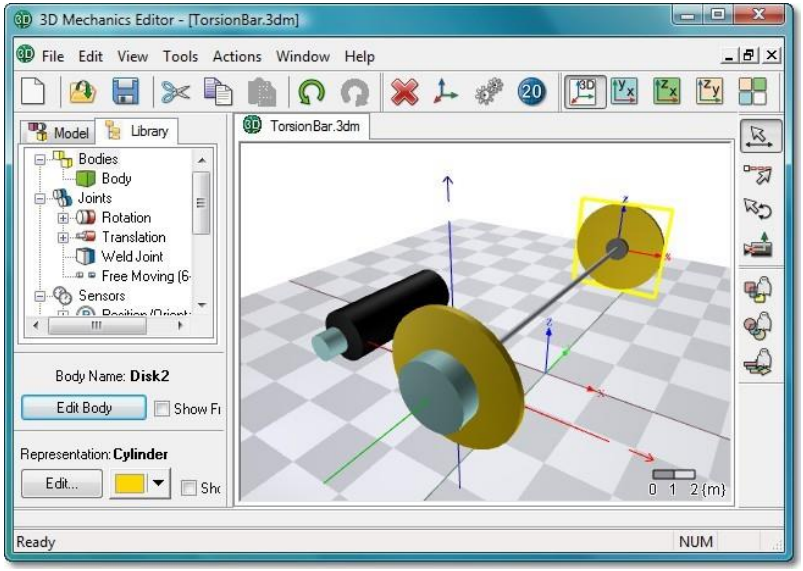


Library

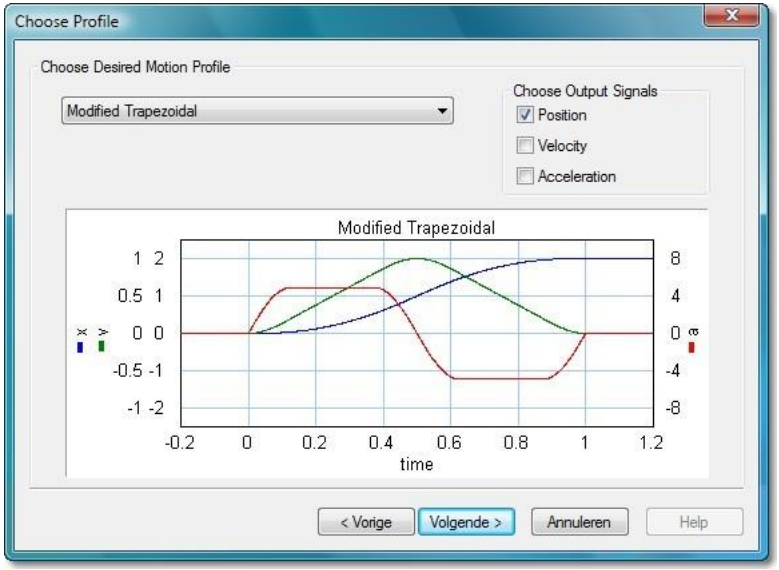


Editor

Modeling Tools

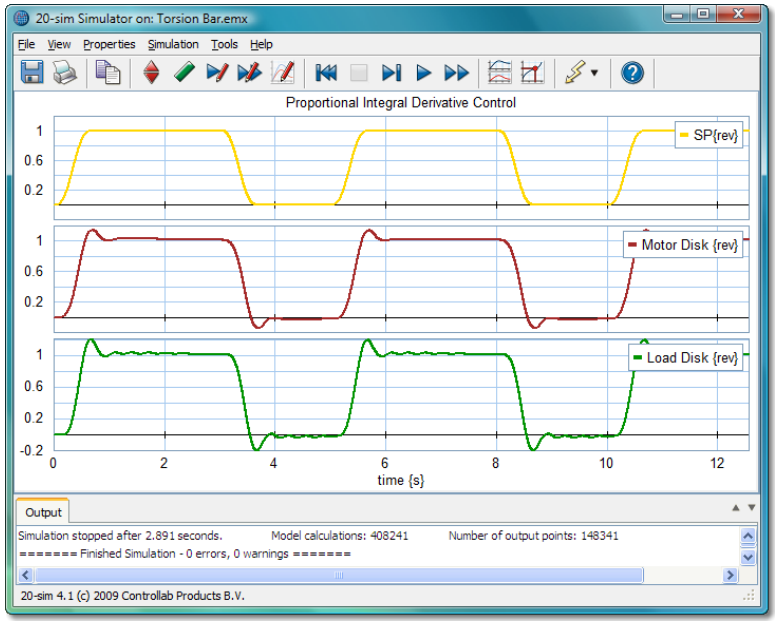


3D Mechanics Editor

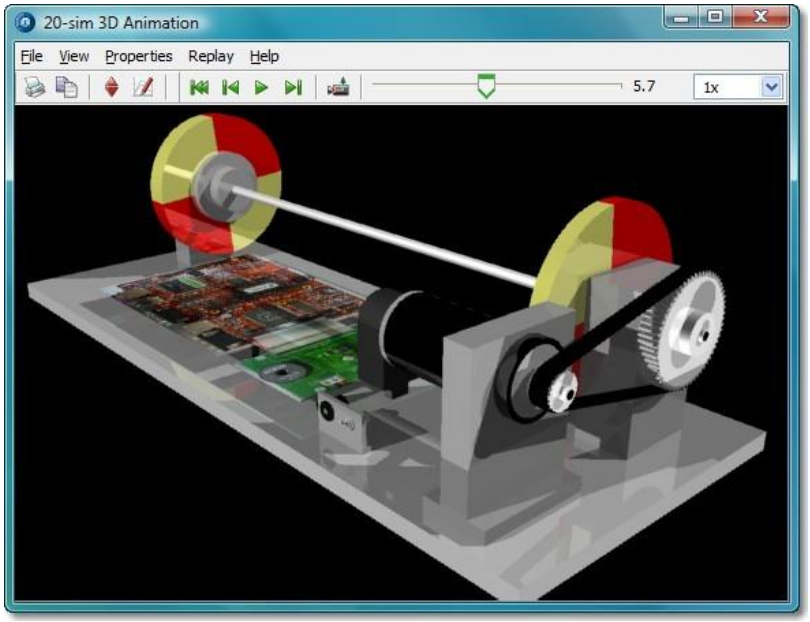


Motion Profile Wizard

Simulation

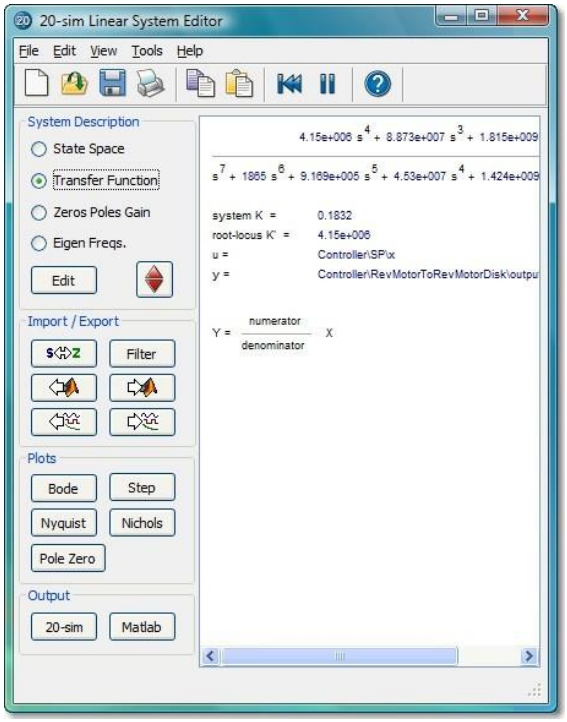


Simulator

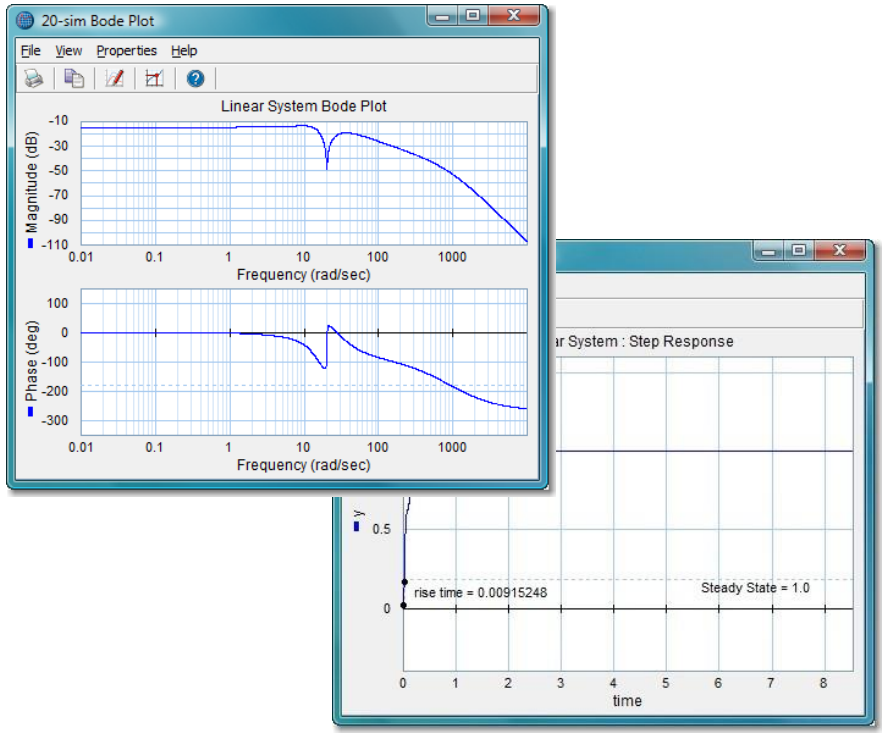


3D Animation

Frequency Domain

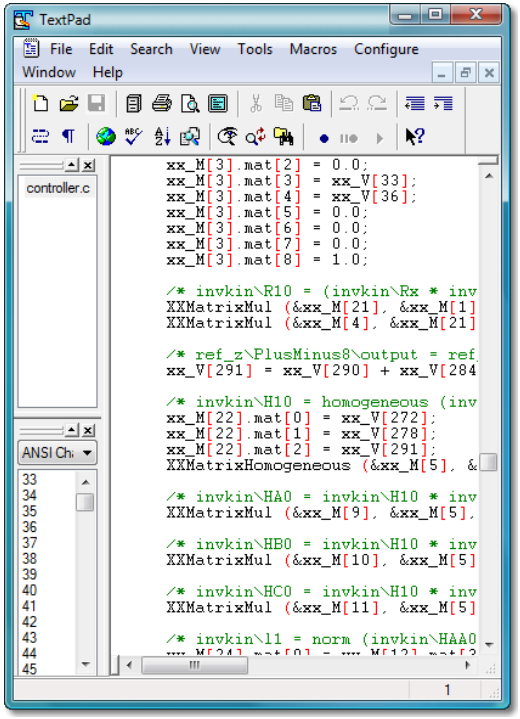


Linearized model

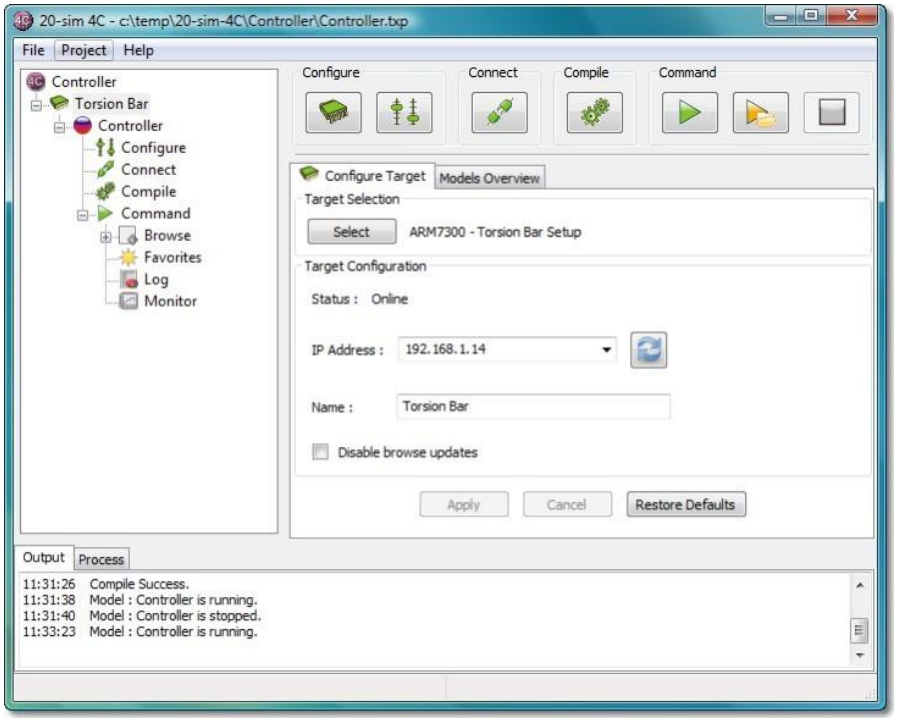


Bode plot and step response

C-code Generation

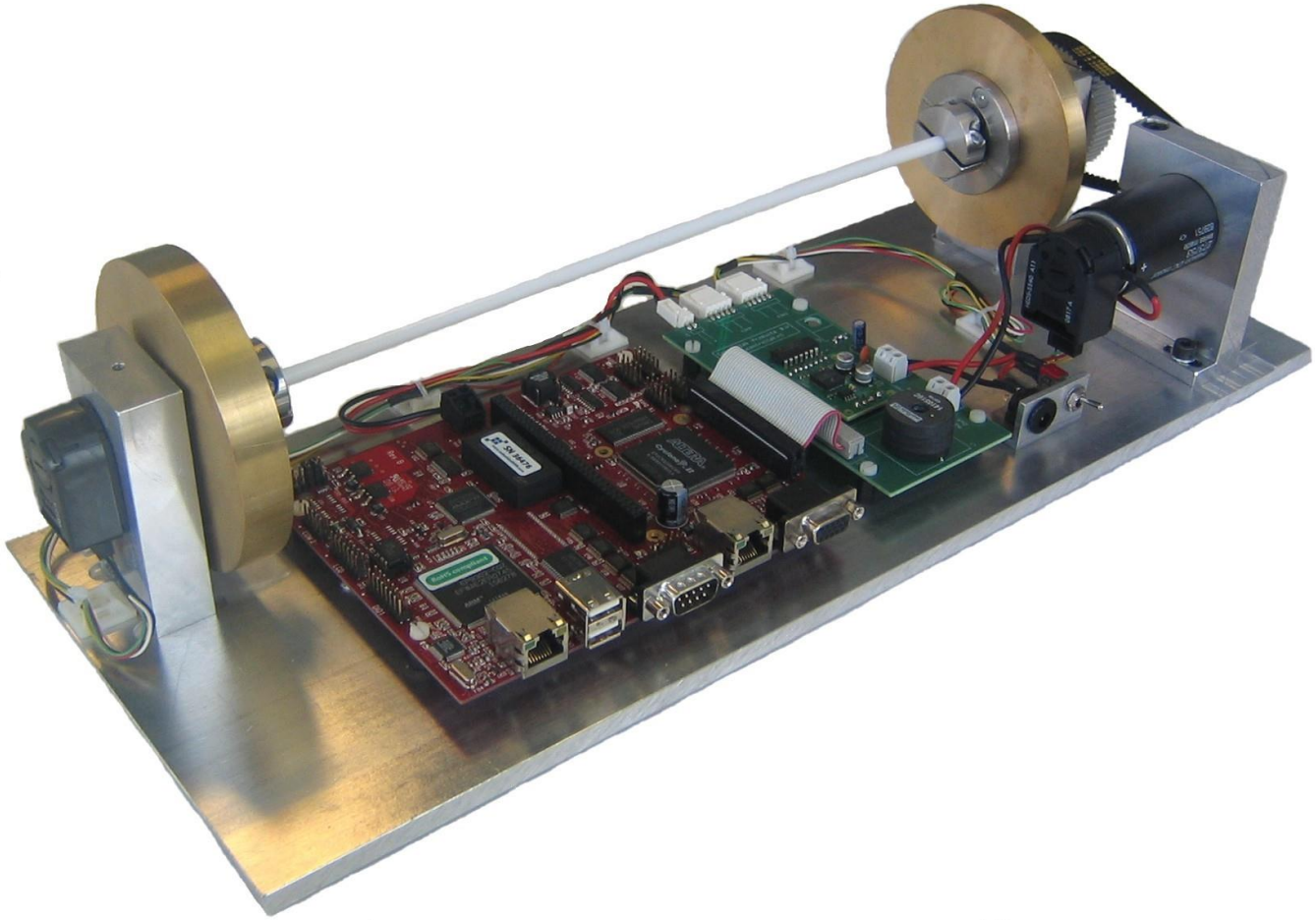


C-code



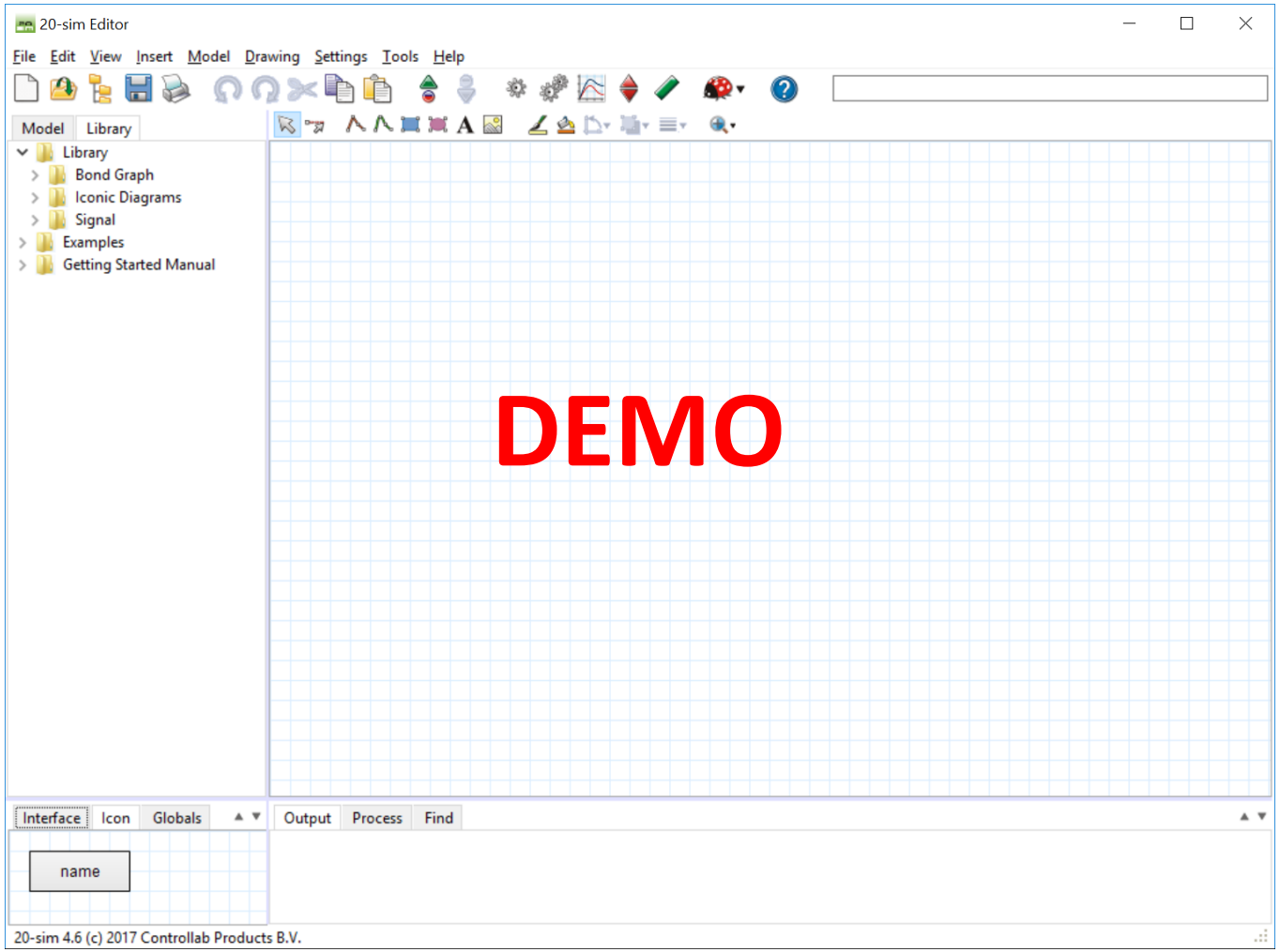
20-sim 4C

.. to Prototype

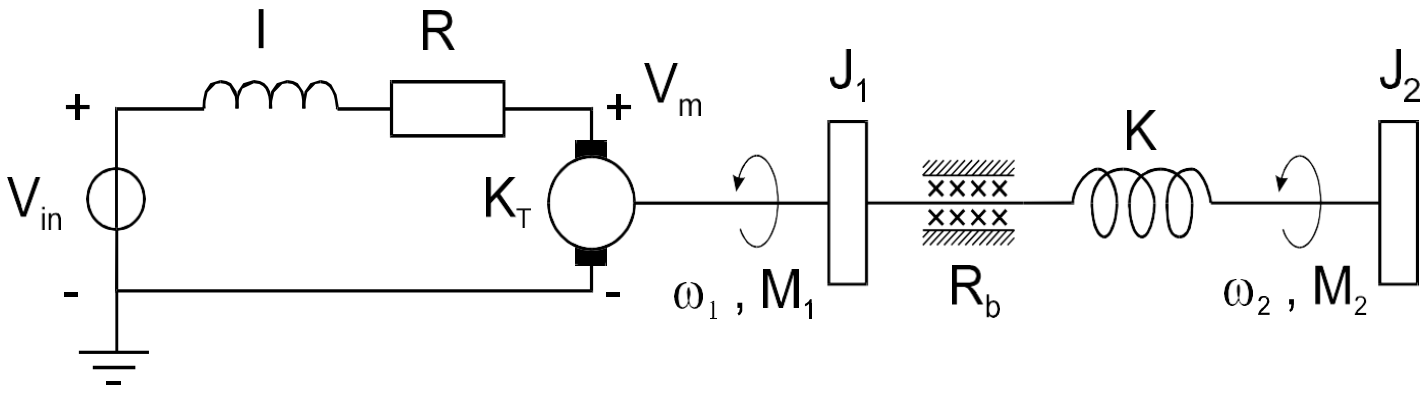


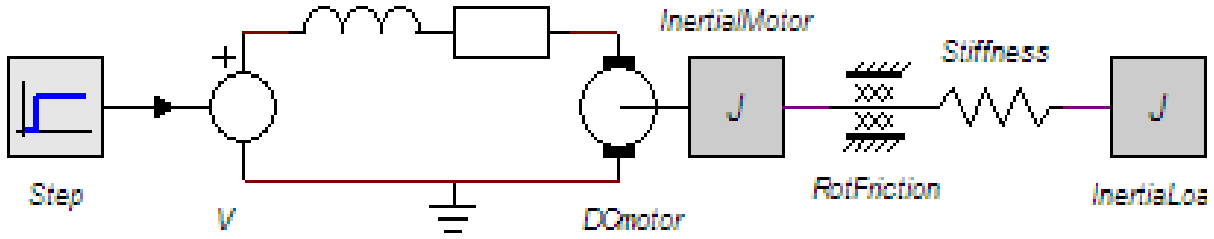
Demonstration

INTO-CPS

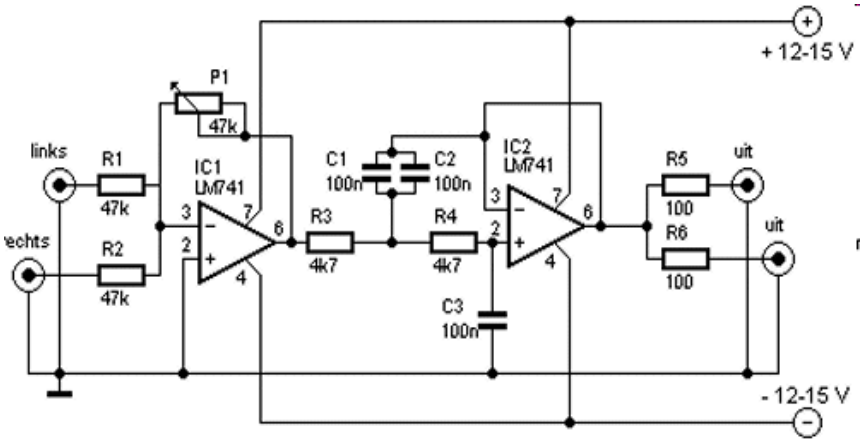


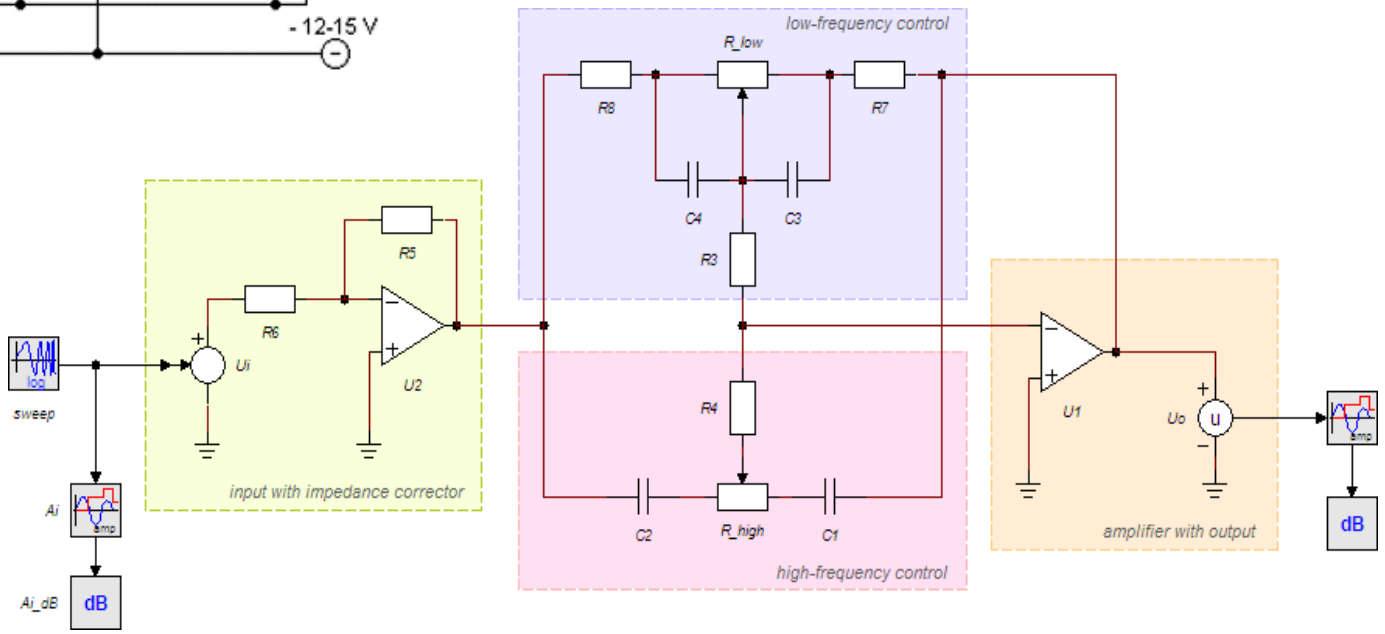
Mechanics


I
R
+
V_{in}
-
+
V_m
-
K_T
J_1
ω_1 , M_1
R_b
K
J_2
ω_2 , M_2

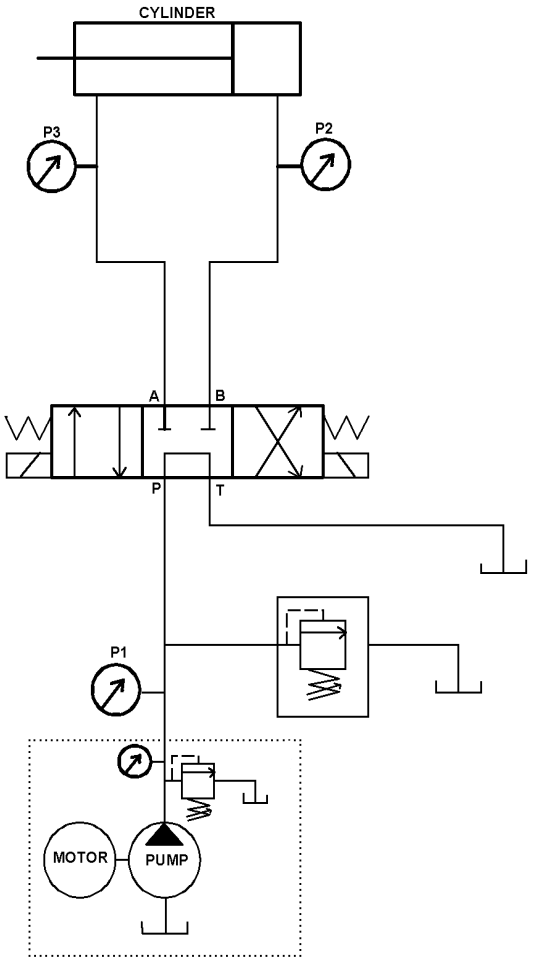

InertialMotor
Stiffness
J
J
Step
V
DCmotor
RotFriction
InertiaLoa

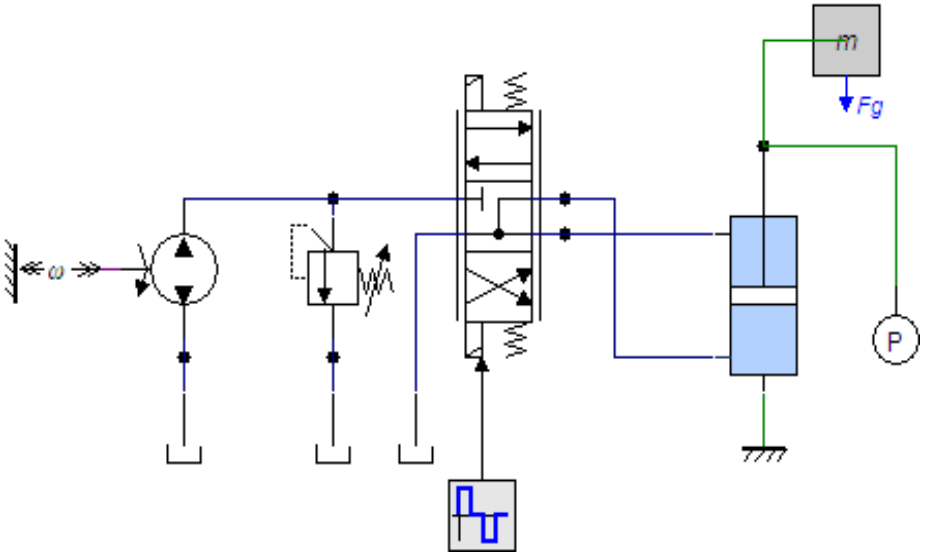
Electronics



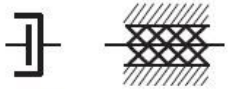


Hydraulics

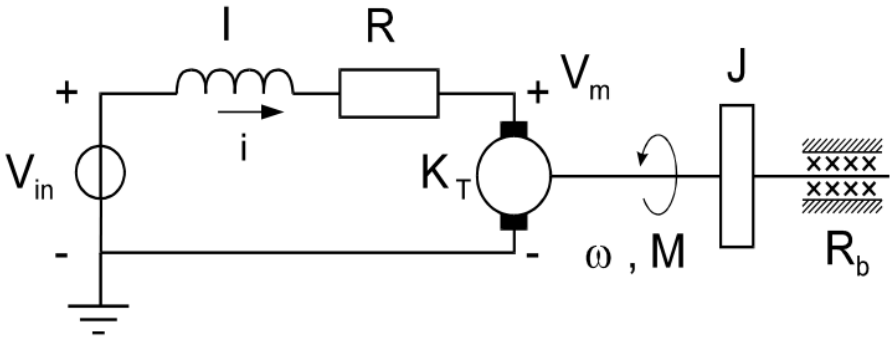




Domains

domain	*behaviours*		
mechanical (translation)	spring: 	damper / friction: 	mass: 
$P = Fv$	$F = \frac{1}{c}\int v dt$	$F\text{-}dv = 0$	$v = \frac{1}{m}\int F dt$
mechanical (rotation)	spring: 	damper/friction: 	inertia: 
$P = T\omega$	$T = \frac{1}{c}\int \omega dt$	$T\text{-}d\omega = 0$	$\omega = \frac{1}{J}\int T dt$
electrical	capacitor: 	resistor: 	inductance: 
$P = ui$	$u = \frac{1}{C}\int i dt$	$u\text{-}Ri = 0$	$i = \frac{1}{L}\int u dt$
hydraulical $P = p\phi$	tank: $p = \frac{1}{c}\int \phi dt$	resistance: $p\text{-}r\phi = 0$	inertia: $\phi = \frac{1}{I}\int p dt$

Differential Equations



$$i = \int \frac{V_{in}}{I} - \frac{R \times i}{I} - \frac{K_T \times \omega}{I} dt$$

$$\omega = \int \frac{K_T \times i}{J} + \frac{R_b \times \omega}{J} dt$$

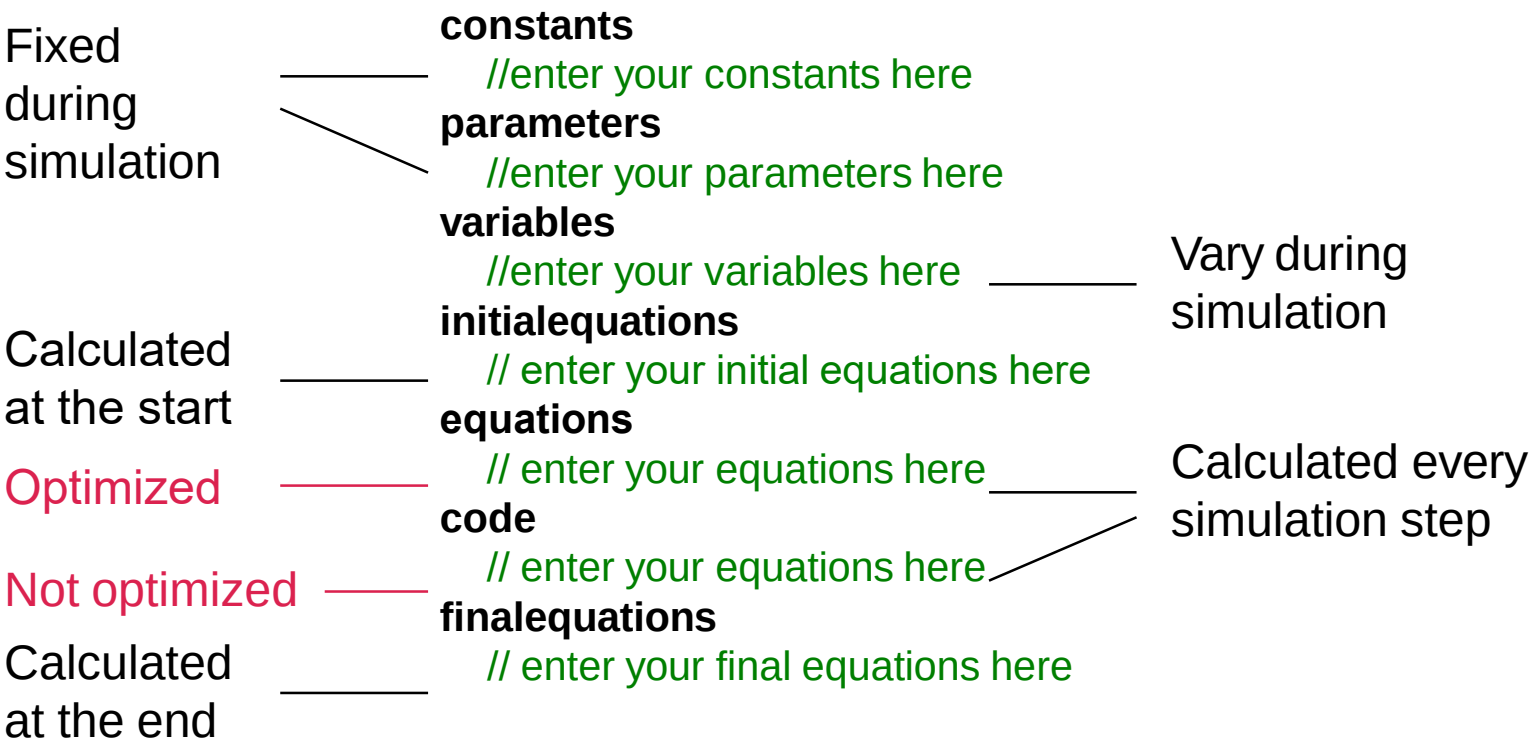
General simulation model:

$$x(t) = x(0) + \int_0^t f(x, u, t)\, dt$$

set of first-order equations and output equation

$$y(t) = g(x, u, t)$$

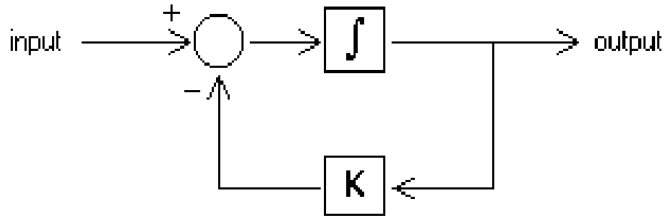
Language Definition



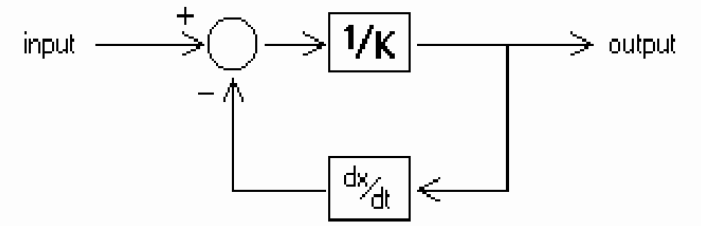
Language Example

```
constants
    real g = 9.81 {m/s2};          // gravity: free fall
parameters
    real start_time = 10 {s};    // start time of the fall
    string out_string = 'run 1';
variables
    real half_g {m/s2};
initialequations
    half_g = -0.5*g;
equations
    if time < start_time then
        output = 0;
    else
        output = half_g*( (time-start_time)^2 );
    end;
finalequations
    toMatlab ( output,out_string);              // send output value to Matlab
```

Integral Form & Differential Form



$$output = \int input - K \times output$$



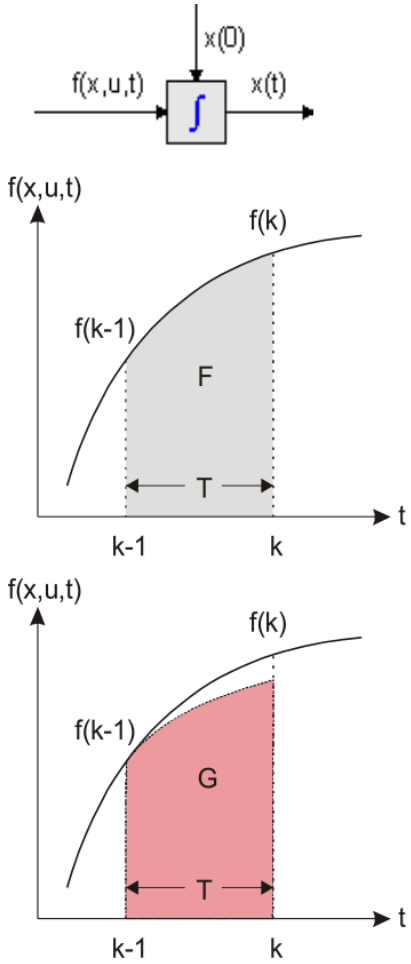
$$output = \frac{input - \frac{d\,output}{dt}}{K}$$

- Integral form is preferred: much easier to simulate
- 20-sim will rewrite differential forms to integral form if possible
- Write your own equations in integral form if possible!

Numerical Integration

$$x(t) = x(0) + \int_0^t f(x, u, t)\, dt$$

$$x(t) = x(k-1) + \int_{(k-1)T}^{kT} f(x, u, t)\, dt$$



Numerical Integration: approximate integral by analytical function G:

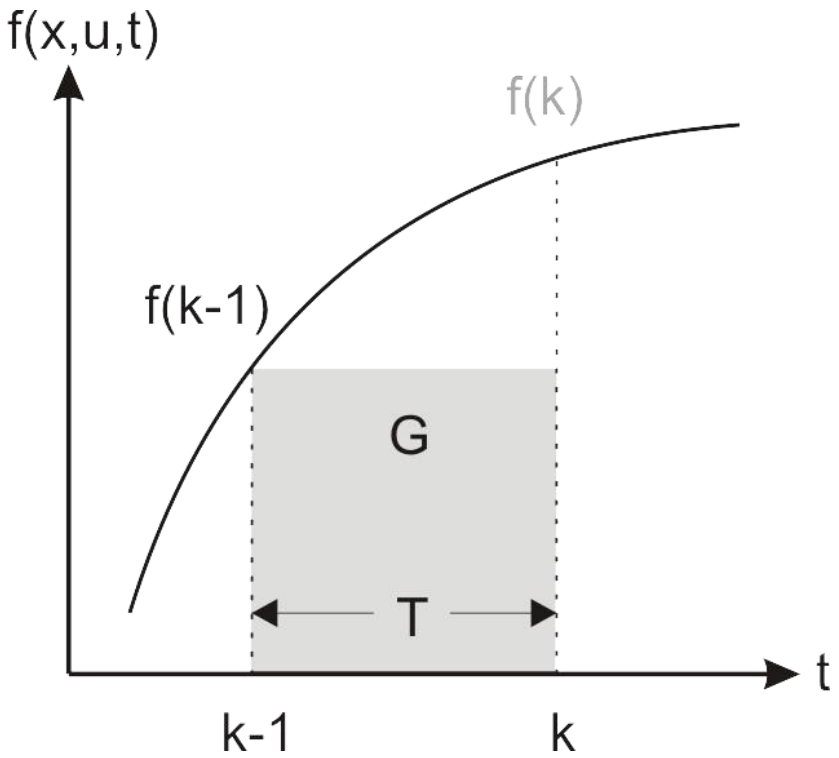
$$\tilde{x}(k) = \tilde{x}(k-1) + G \qquad G \approx \int_{(k-1)T}^{kT} f(x, u, t)\, dt$$

Euler

$$x(t) = x(k-1) + \int_{(k-1)T}^{kT} f(x,u,t)\,dt$$

$$G = T \cdot f(k-1)$$

$$\tilde{x}(k) = \tilde{x}(k-1) + T \cdot f(k-1)$$



Runge-Kutta 4

$$x(k) = x(k-1) + \int_{(k-1)T}^{kT} f(x,u,t)dt$$

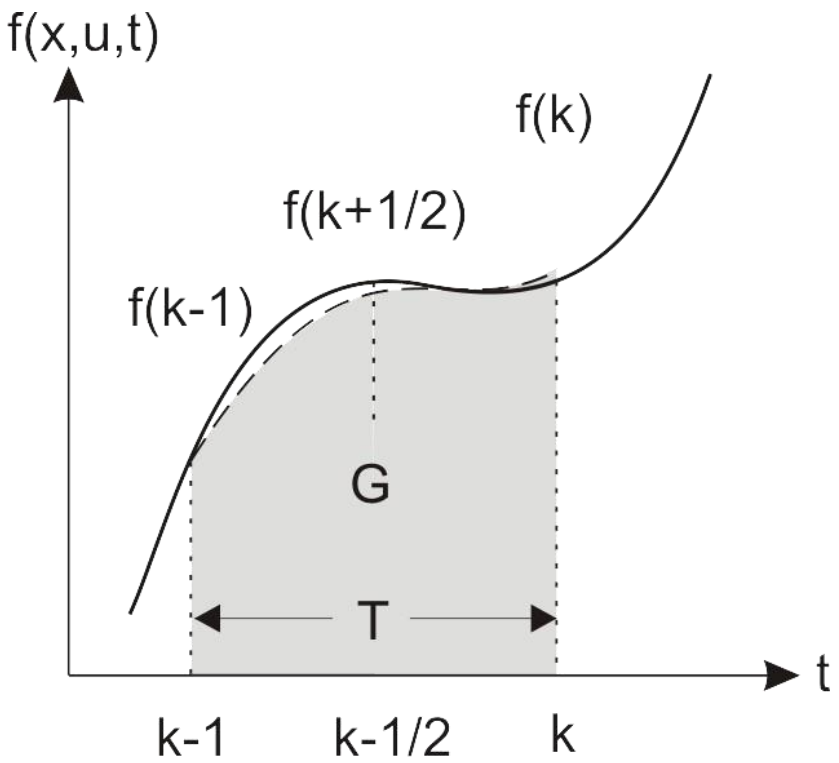
$$G = \frac{T}{6} \times (k_1 + 2 \times k_2 + 2 \times k_3 + k_4)$$

$$k_1 = f(x(k-1),u(k-1),k-1)$$

$$k_2 = f(x(k-1) + \frac{T}{2} k_1, u(k-\frac{1}{2}), k-\frac{1}{2})$$

$$k_3 = f(x(k-1) + \frac{T}{2} k_2, u(k-\frac{1}{2}), k-\frac{1}{2})$$

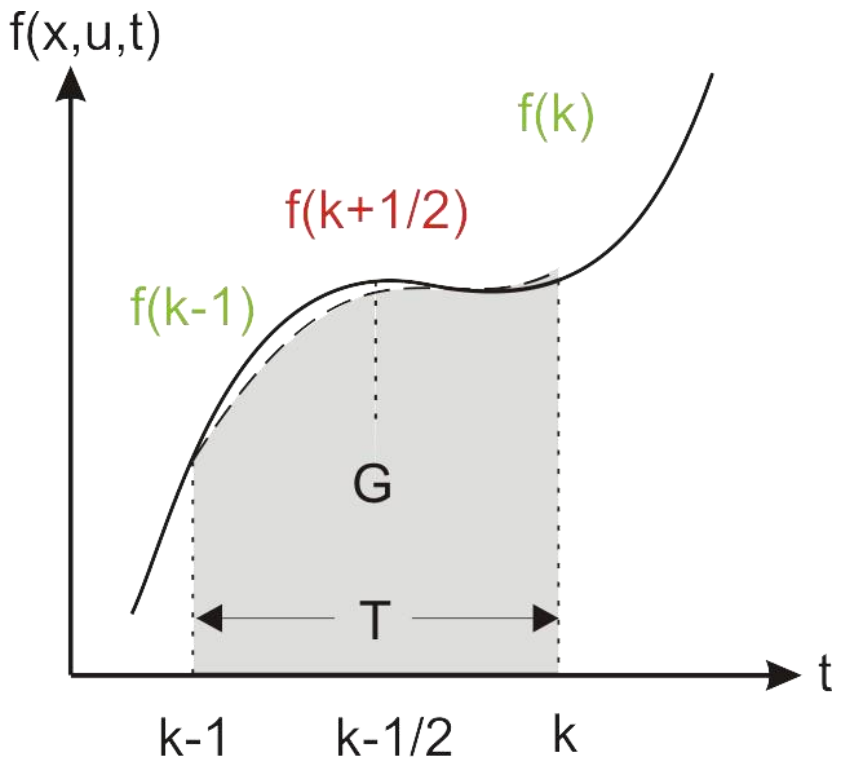
$$k_4 = f(x(k-1) + T \times k_3, u(k), k)$$



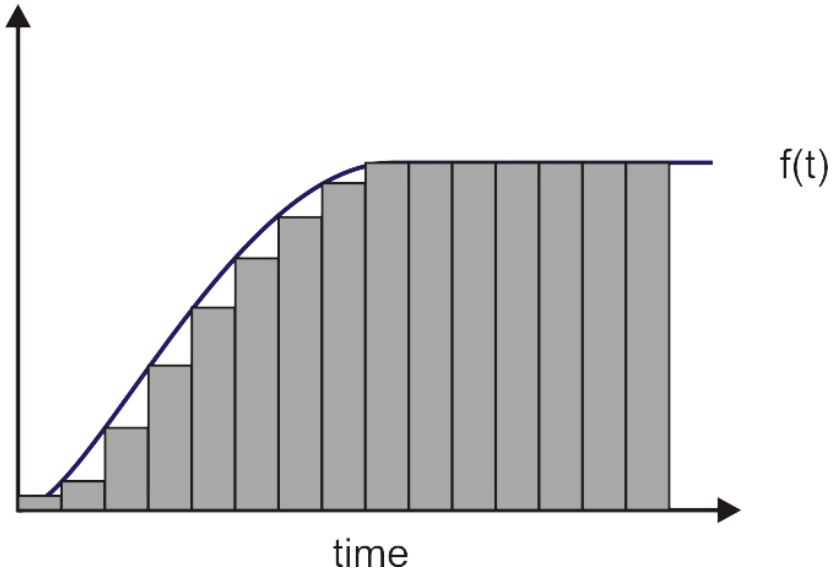
Major and Minor Steps

- Major steps: f(k-1), f(k)
- Minor steps: f(k+1/2)
- Don't use code like:

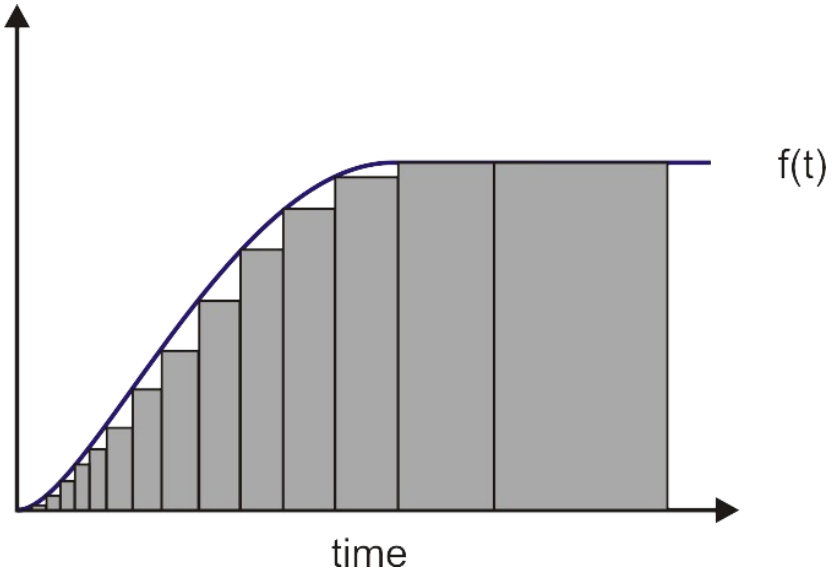
```
if (time > 0) then
x = x+1;
end;
```



Fixed and Variable Step



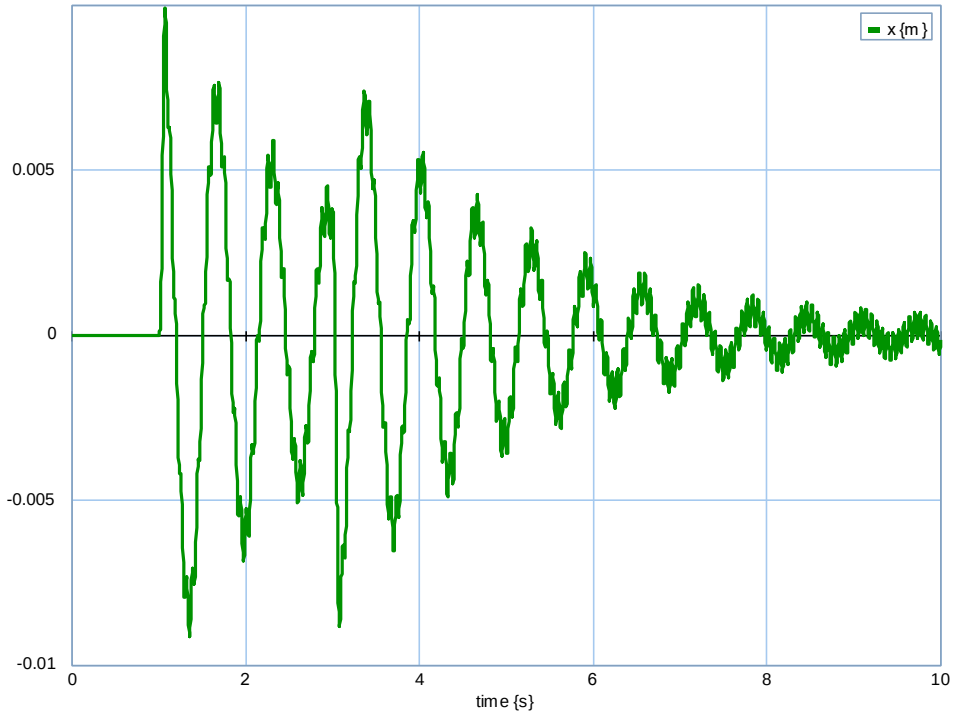
Fixed Step



Variable Step

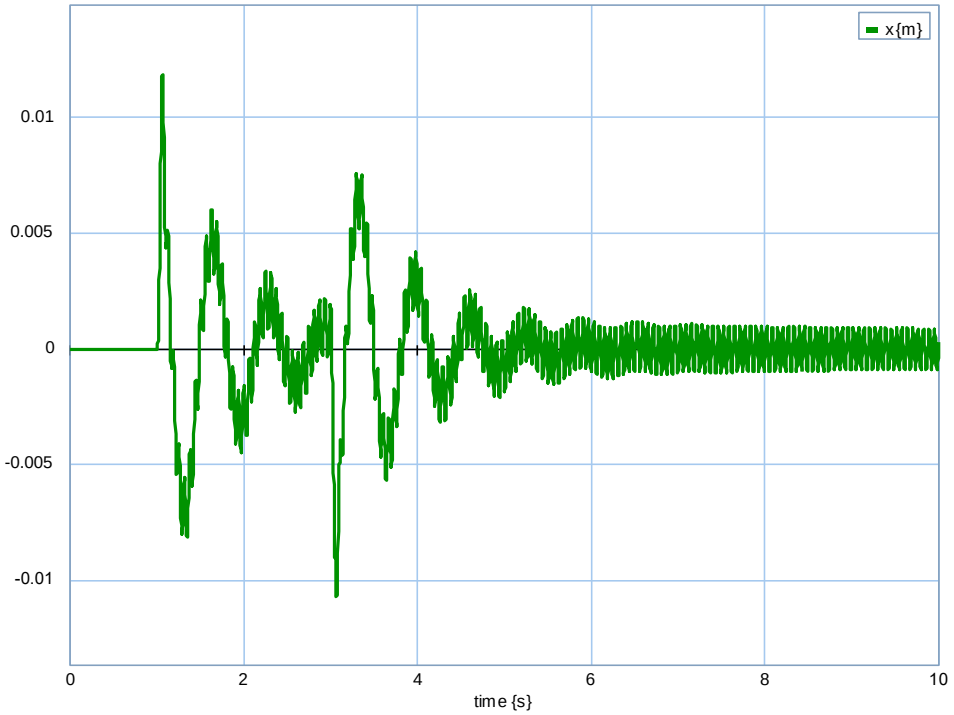
Stiff Models

- Model has low frequency resonance high frequency resonances
- Hard to simulate with fixed step methods

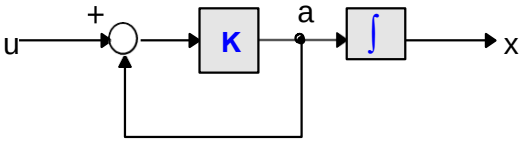


No Damping

- Model has resonances which damp out poorly
- Hard to simulate with some methods



Algebraic Loops



$$x(t)=x(k-1)+\int_{(k-1)T}^{kT} f(x,u,t)\,dt$$

$$f(x,u,t)=a$$

$$a=k\cdot u-a$$

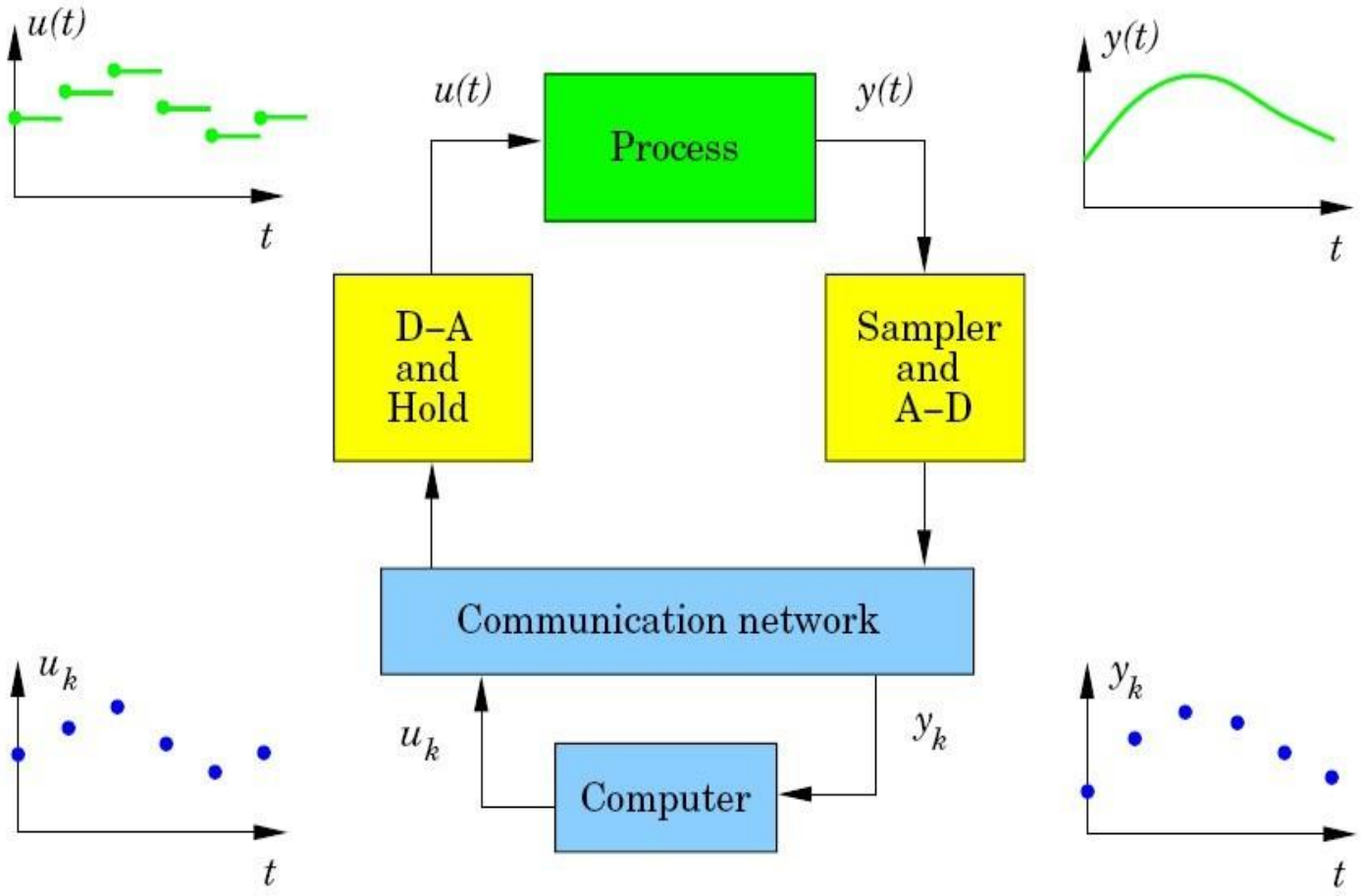
↓

algebraic loop

Algebraic Loops:

- function that is a function of itself
- Require iteration to solve
- Reduces simulation speed

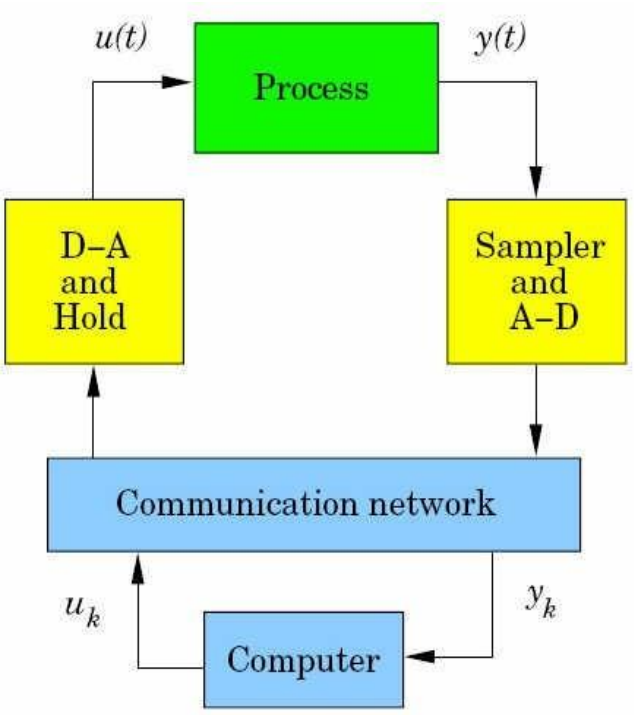
Computer Controlled Systems



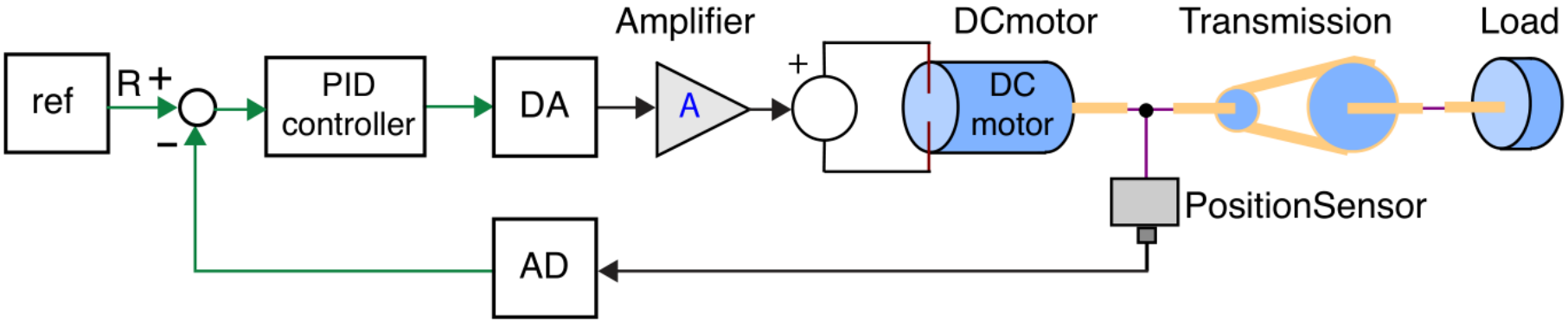
Discrete

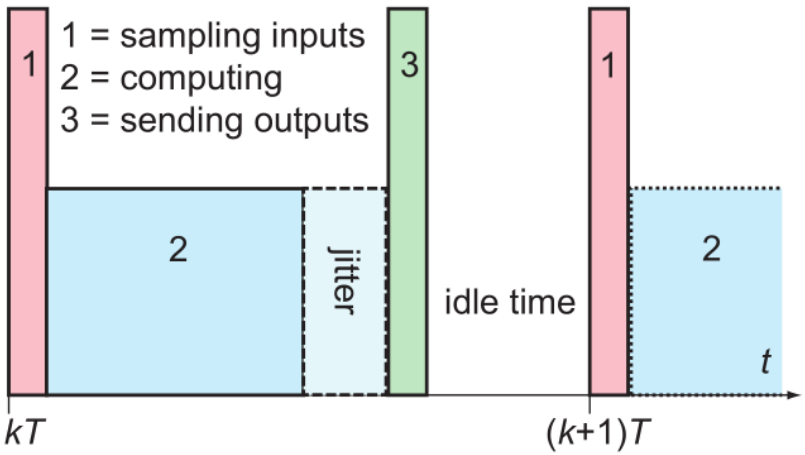
Using certain functions will make a block to run in discrete-time.

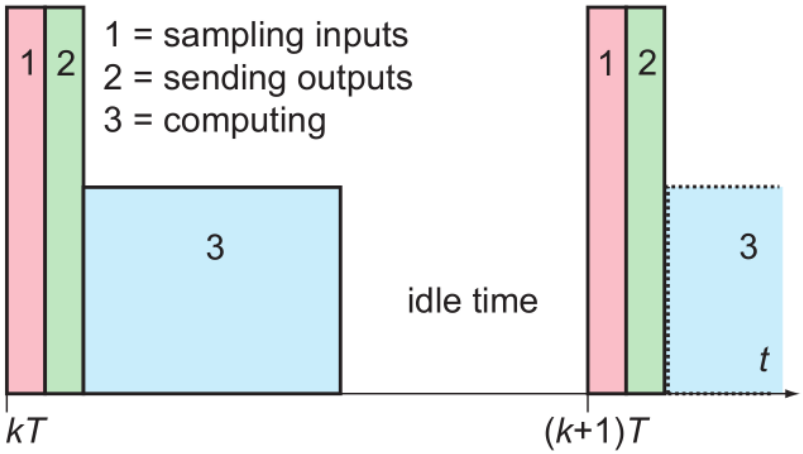
- sample: output of the function is discrete time, the input is continuous time
- hold: the output is continuous time, the input of the function is discrete time



Sampling


ref
R
+
−
PID controller
DA
Amplifier
A
+
DCmotor
DC motor
Transmission
Load
PositionSensor
AD

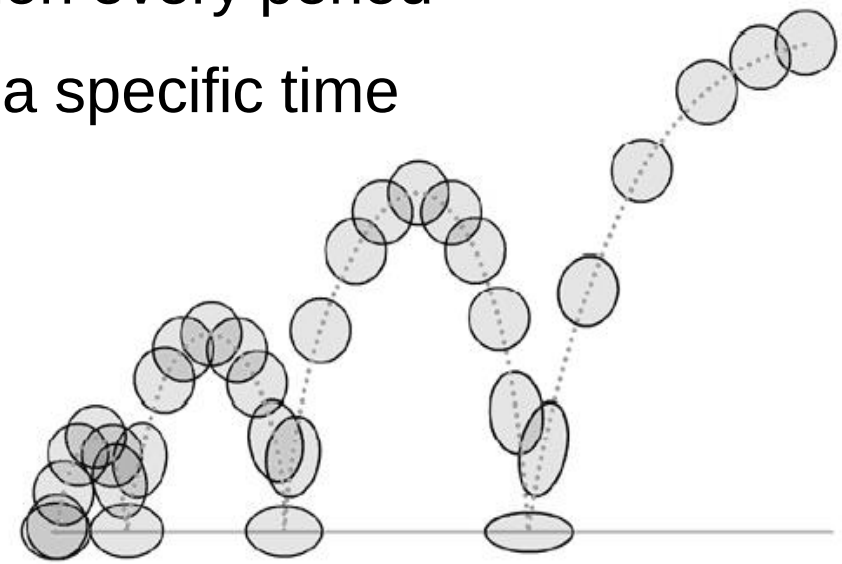

1
1 = sampling inputs
2 = computing
3 = sending outputs
3
2
jitter
idle time
1
2
t
kT
(k+1)T


1
2
1 = sampling inputs
2 = sending outputs
3 = computing
3
idle time
1
2
3
t
kT
(k+1)T

Events

Force integration method to do a calculation at a specific point:

- event: find a zero crossing
- eventup: find a zero crossing from negative to positive
- eventdown: find a zero crossing from positive to negative
- frequencyevent: do a calculation every period
- timeevent: do a calculation at a specific time



INTO-CPS

20-sim Tutorial

Ken Pierce, Newcastle University

NII 大学共同利用機関法人 情報・システム研究機構 国立情報学研究所 National Institute of Informatics

AARHUS UNIVERSITY

Newcastle University

Horizon 2020 Programme