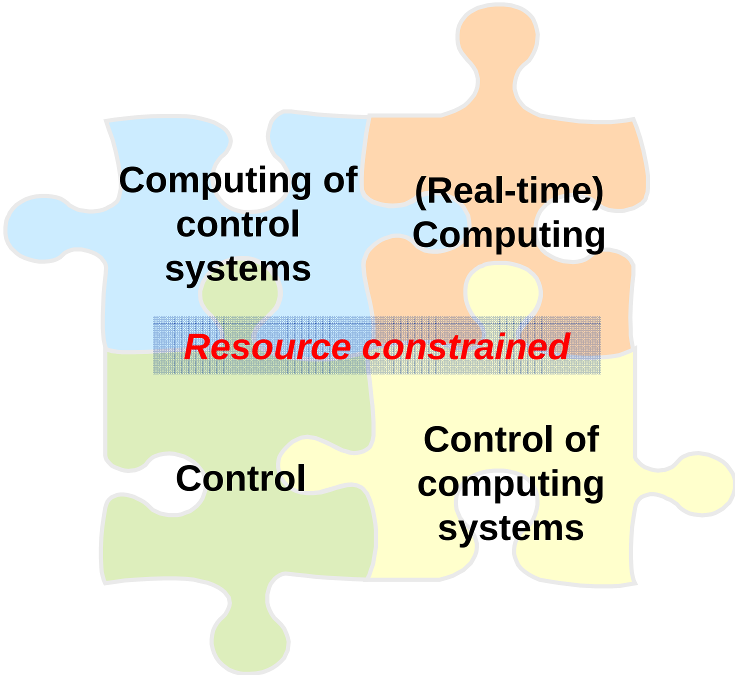
# Real-time control systems

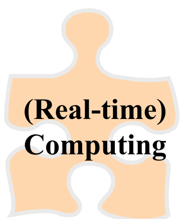





ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

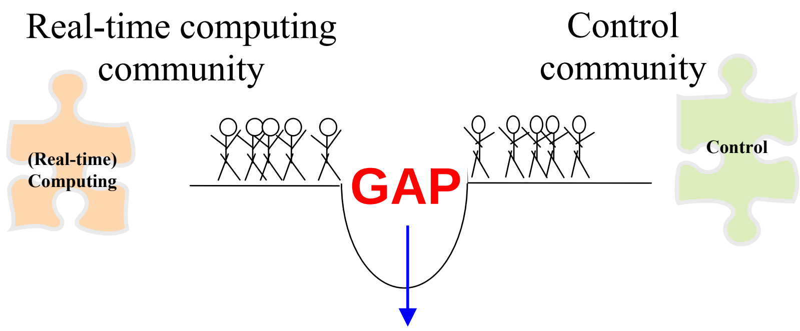
# Real-time control systems


- We do not focus only on control issues … too difficult ☺
- ... but we analyse control problems and offer solutions when required




- We do not focus only on real-time issues… too difficult ☺
- ... but we analyse real-time problems and offer solutions when required

# Real-time control systems




## Misconceptions, e.g.

Real-Time Computing
*assumes hard deadlines for control algorithms*

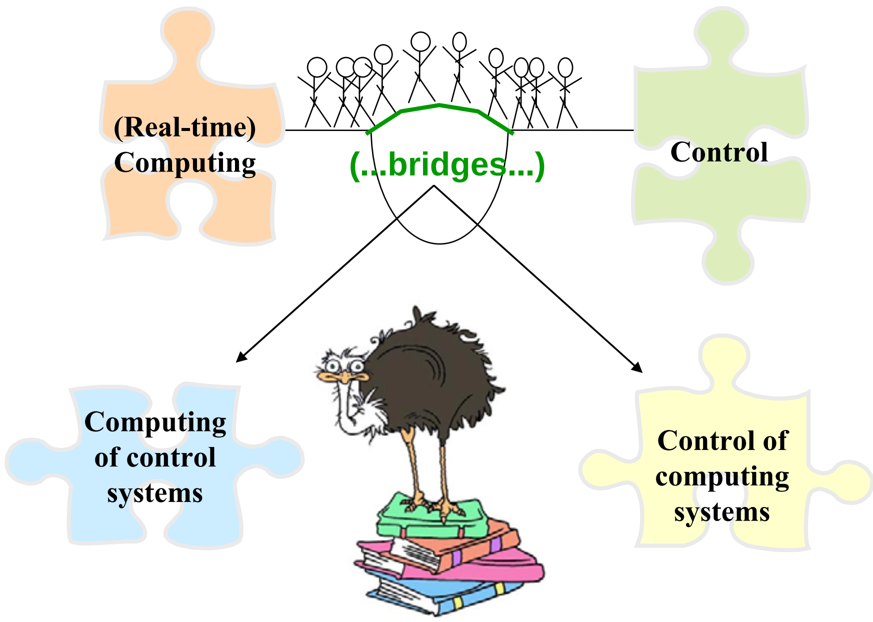
Control
*assumes determinism in the computing platform*



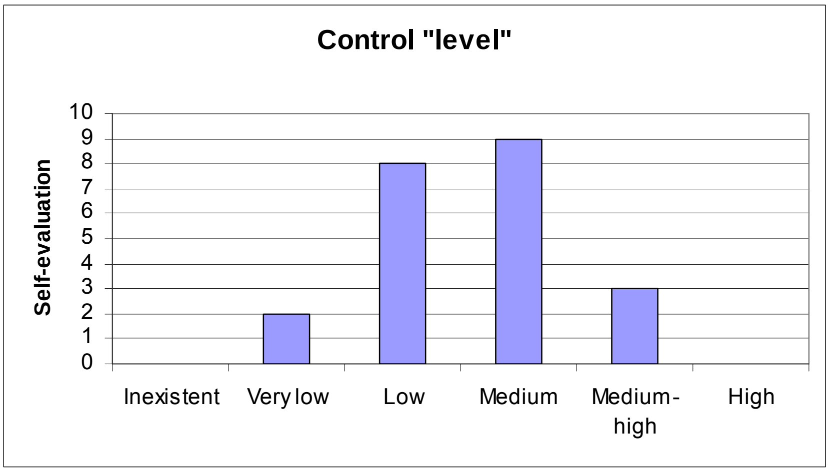


# Real-time control systems

**Bridging the GAP**



# Real-time control systems




(20-25 participants)



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time control systems

**1. Control systems: the basics**

... focuss on timing sensitive features

**2. Real-time computing of control systems**

... do resource constraints affect control systems performance?

**3. Control of computing systems**

... can control theory improve computing systems performance?

**4. Stability issues**

... at least, everything has to be stable

# Control systems: the basics

1. Concepts
2. Introductory examples
3. System poles
4. State Space Models
5. Discrete time systems, models and poles
6. Simulation
7. Controller design



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control systems: the basics

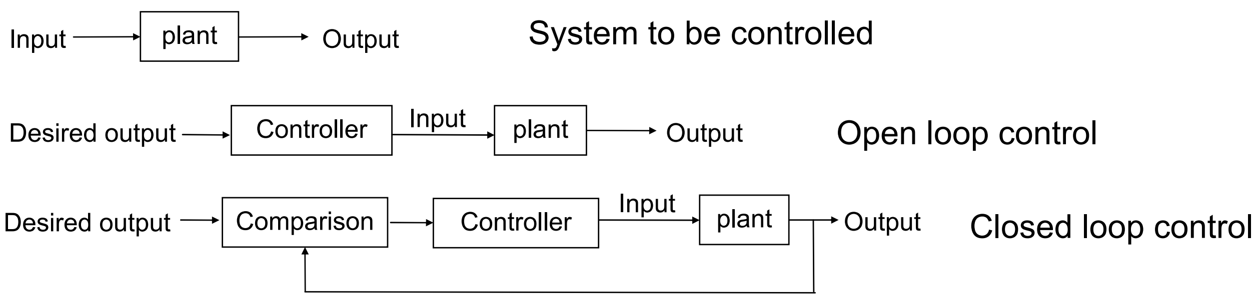
## Concepts

The **goal of control** is to apply inputs to a system in order to have the system behaving as expected/desired.

• Cruise-control car: f_engine(t)=? → speed=60 mph

• Utilization control in a video server : Admitted_requests(h)? → utilization=80%

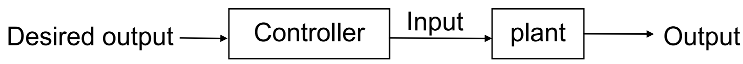
### Configurations



# Control systems: the basics

## Concepts

**Open loop control**




**Idea:** Compute control input without continuous* variable measurement

- Simple
- Need to know EVERYTHING ACCURATELY to work right

  Cruise-control car: friction(t), ramp_angle(t)

  video server: workload(h) (request arrival rate? decoding times?);

**Open-loop control fails when**

- We don't know everything
- We make errors in estimation/modeling
- Things change

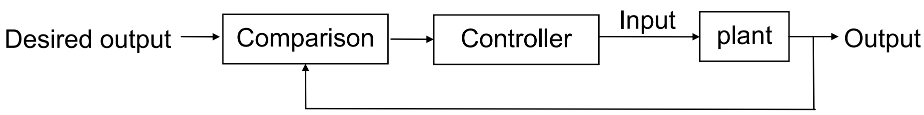




# Control systems: the basics

## Concepts

**Closed loop control**




**Idea:** Compute control input as a function of continuous* variable measurement

- More complicated (so we need control theory)
- Continuously measure & correct
  - Cruise-control car: measure speed & change engine force
  - Video server: measure utilization and completion rate & change admission rate

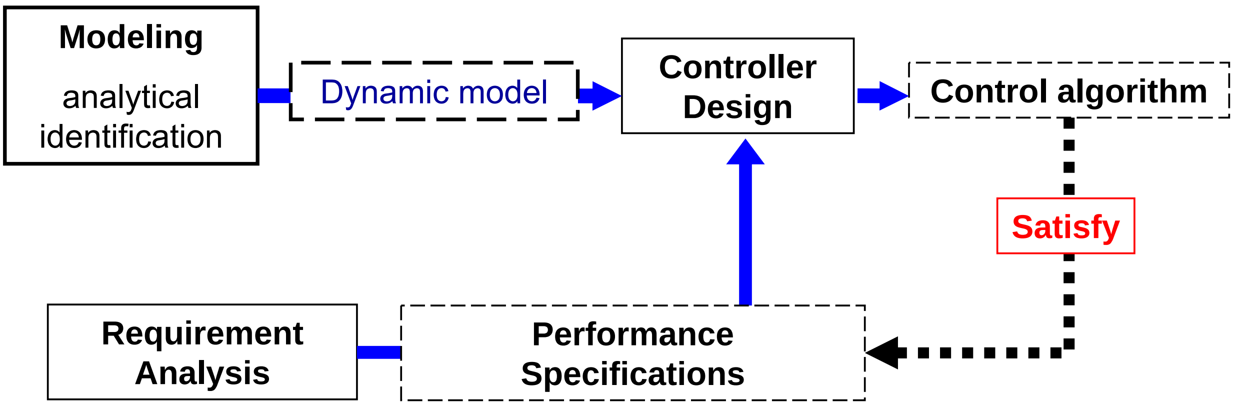
**Feedback control theory makes it possible to control well even if**

- We don't know everything
- We make errors in estimation/modeling
- Things change

# Control systems: the basics

## Concepts

**Control design methodology**






ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control systems: the basics

## Introductory examples

**Let’s look at cruise control**



- The **set/accel** button tells the car to maintain the speed you are currently driving at. Holding down the set/accel button will make the car accelerate

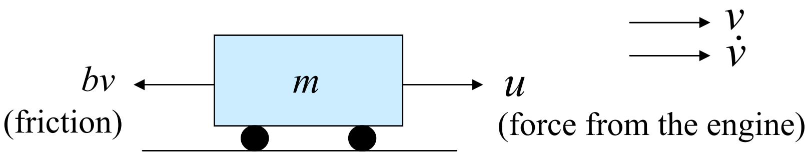
- Holding down the **coast** button will cause the car to decelerate

**Objective:** to accelerate to the desired speed without overshooting, and then to maintain that speed with little deviation no matter how much weight is in the car, or how steep the hill you drive up.

# Control systems: the basics

## Introductory examples

**Let's look at cruise control: simple model**




System dynamics (Newton's law): $\sum f_i = ma \longrightarrow u - bv = m\dot{v}$

Desired output *y*: a desired velocity $\Longrightarrow y = v$

| | |
|---|---|
| Model parameters | m = 1000kg<br>b = 50Ns/m |

What's the behavior of our car?



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

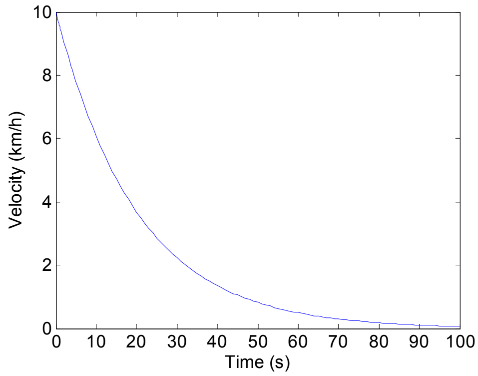
# Control systems: the basics

## Introductory examples

**Let's look at cruise control: stable dynamics**

First order differential equation $\quad u - bv = m\dot{v} \qquad y = v$ ?

Going at 10km/h, if no force (u=0) is applied...




... the car slows down $\Rightarrow$ the car is **stable**

# Control systems: the basics

## Introductory examples

**Let's look at cruise control: stable dynamics**

First order differential equation $u - bv = m\dot{v}$ $\qquad y = v$ ?

Going at 10km/h, if no force (u=0) is applied...

Solution: $v(t) = C \cdot e^{\frac{-b}{m}\cdot t}$

Proof: $0 - b \cdot C \cdot e^{\frac{-b}{m}\cdot t} = m \cdot C \cdot \frac{-b}{m} \cdot e^{\frac{-b}{m}\cdot t}$

(if $v(0)=10 \Rightarrow C=v_0=10$)

The shape of the response depends on the exponential coeficient



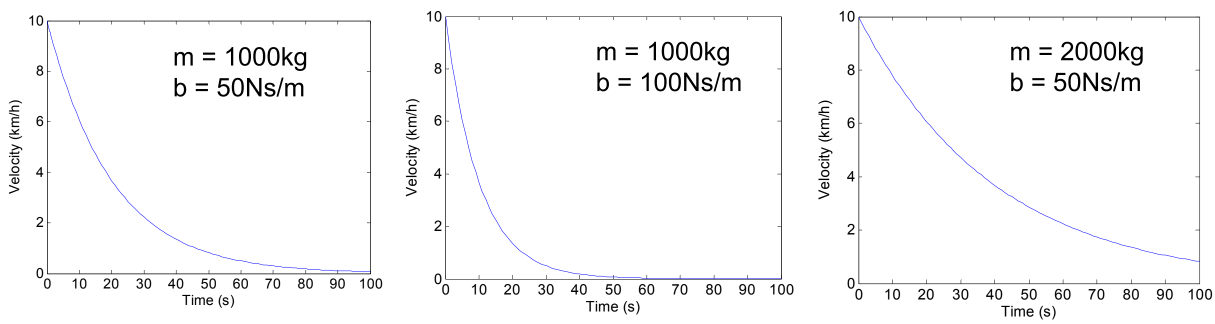
ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control systems: the basics

## Introductory examples

**Let's look at cruise control: dynamics** $\qquad v(t) = v_0 \cdot e^{\frac{-b}{m}\cdot t}$

If the friction increases (b↑), the response will be faster
If the mass increases (m↑), the response will be slower



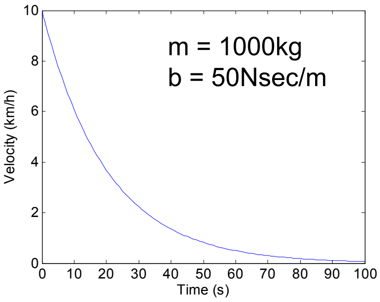

Note that the exponential coefficient is < 0

# Control systems: the basics

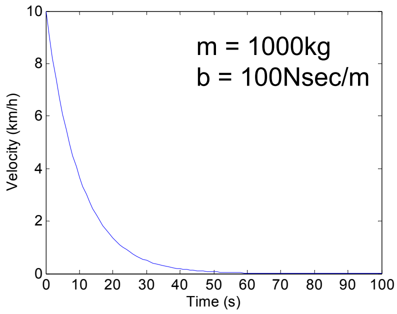
## Introductory examples

**Summary: the exponent's coefficient determines the response shape.**

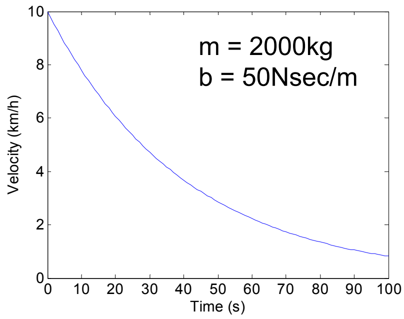
The smaller is the coefficient the faster is the response $v(t) = v_0 \cdot e^{\frac{-b}{m} \cdot t}$



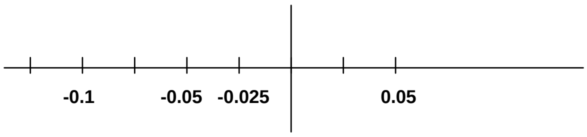

$\lambda = -b / m = -0.05$




$\lambda = -b / m = -0.1$




$\lambda = -b / m = -0.025$








# Control systems: the basics

## Introductory examples

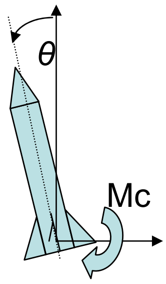
**Let's look at rocket control**

**Objective:** to keep a desired angle

# Control systems: the basics

## Introductory examples

**Let's look at rocket control: model**




$M_c$ control torque (Nm)
J rotational inertia (kg·m²)
$\theta$ angle

System dynamics (Newton's law): $\sum f_i = ma \longrightarrow J\ddot{\theta} - b\dot{\theta} - m\theta = M_c$

Desired output *y*: a desired angle $\longrightarrow y = \theta = 0$

| | |
|---|---|
| Model parameters (to simplify) | J = 10kg·m² <br> b=1 Nsm <br> m=5 Nm |

What's the behavior of the rocket?



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

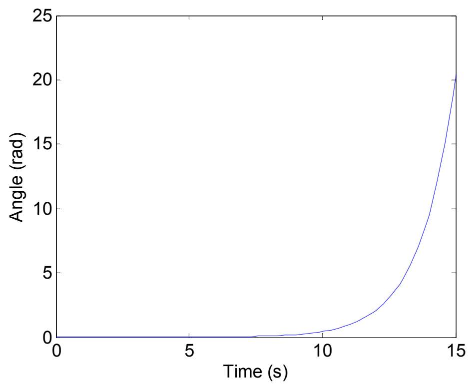
# Control systems: the basics

## Introductory examples

**Let's look at cruise control: unstable dynamics**

Second order differential equation $J\ddot{\theta} - b\dot{\theta} - m\theta = M_c \qquad y = \theta?$

Initial angle of 0.0005 rad, if no force ($M_c$=0) is applied...




... the rocket starts rotating $\Rightarrow$ the rocket is **unstable**

# Control systems: the basics

## Introductory examples

**Let's look at cruise control: unstable dynamics**

First order differential equation $J\ddot{\theta} - b\dot{\theta} - m\theta = M_c$ $\qquad y = \theta$?

With an initial angle of 0.0005 rad, if no force ($M_c$=0) is applied...

Solution: $\theta(t) = C_1 \cdot e^{\frac{b+\sqrt{b^2+4Jm}}{2J}t} + C_2 \cdot e^{\frac{b-\sqrt{b^2+4Jm}}{2J}t} = C_1 \cdot e^{\lambda_1 t} + C_2 \cdot e^{\lambda_2 t}$

If inital conditons are:
*θ(0)=0.0005*
*dθ/dt (0)=0*
$\Rightarrow$
*C1=0.0002323*
*C2=0.0002676*

If parameters are:
J = 10kg·m2
b=1 Nsm
m=5 Nm
$\Rightarrow$
$\lambda_1$=0.7589
$\lambda_2$=-0.6589

Once again, the shape of the response depends on the exponent's coefficients



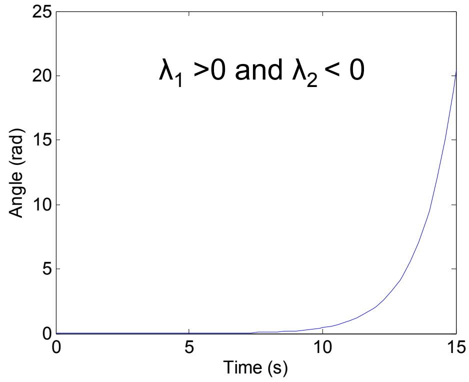


# Control systems: the basics

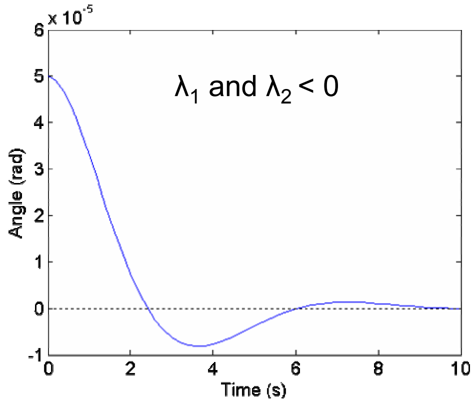
## Introductory examples

**Let's look at rocket control: dynamics** $\theta(t) = C_1 \cdot e^{\lambda_1 t} + C_2 \cdot e^{\lambda_2 t}$

If coefficients λ are positive, the system is unstable (one would be enough)
If coefficients λ are negative, the system is stable




unstable




stable

# Control systems: the basics

## System poles

**General Case:** differential equation or order n

$$a_n y^{n)} + a_{n-1} y^{n-1)} + \ldots + a_0 y = u$$

**Solution:**

$$y(t) = C_1 \cdot e^{\lambda_1 t} + C_2 \cdot e^{\lambda_2 t} + \ldots + C_n \cdot e^{\lambda_n t}$$

The shape of the response depends on the exponent's coefficients λ

**Stability:** If any of these coefficients is positive the system will be unstable

These coefficients are called "System Poles"



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control systems: the basics

## State-space models

- We do not work with high order differential equations … too difficult ☺
- There are simplified working methods based on "models" which are equivalent to the differential equations, but simplify operations

**(Easy to understand) Available models**

**Differential equations**

**State-space models** | **Input-Output models**

# Control systems: the basics

## State-space models

**Example:** Cruise control

$$u - bv = m\dot{v}$$

$$\dot{v}(t) = \left[\frac{-b}{m}\right] v(t) + \left[\frac{1}{m}\right] u(t)$$

$$y(t) = [1] v(t)$$

**Example:** rocket control

$$J\ddot{\theta} - b\dot{\theta} - m\theta = M_c$$

$$\begin{bmatrix} \dot{\theta}_1(t) \\ \dot{\theta}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{m}{J} & \frac{b}{J} \end{bmatrix} \begin{bmatrix} \theta_1(t) \\ \theta_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} M_C(t)$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \theta_1(t) \\ \theta_2(t) \end{bmatrix}$$

**General state space model (SISO)**

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t)$$
$$y(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}u(t)$$



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control systems: the basics

## State-space models

**Example:** rocket control $\quad J\ddot{\theta} - b\dot{\theta} - m\theta = M_c$

Define variables

$$\theta = x_1$$
$$\dot{\theta} = x_2$$

$\longrightarrow$

Substitute variables

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = \frac{b}{J}\dot{\theta} + \frac{m}{J}\theta + M_c = \frac{b}{J}x_2 + \frac{m}{J}x_1 + M_c$$

$$\begin{bmatrix} \dot{\theta}_1(t) \\ \dot{\theta}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{m}{J} & \frac{b}{J} \end{bmatrix} \begin{bmatrix} \theta_1(t) \\ \theta_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} M_C(t)$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \theta_1(t) \\ \theta_2(t) \end{bmatrix}$$

# Control systems: the basics

## State-space models

$$\dot{\mathbf{x}}(t)=\mathbf{A}\mathbf{x}(t)+\mathbf{B}u(t)$$
$$y(t)=\mathbf{C}\mathbf{x}(t)+\mathbf{D}u(t)$$

Where are the system poles? $|\lambda\mathbf{I}-\mathbf{A}|=0$

Characteristic equation

**Example: rocket control**

Model

$$\begin{bmatrix}\dot{\theta}_1(t)\\ \dot{\theta}_2(t)\end{bmatrix}=\begin{bmatrix}0 & 1\\ \frac{m}{J} & \frac{b}{J}\end{bmatrix}\begin{bmatrix}\theta_1(t)\\ \theta_2(t)\end{bmatrix}+\begin{bmatrix}0\\ 1\end{bmatrix}M_C(t)$$

$$y(t)=\begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}\theta_1(t)\\ \theta_2(t)\end{bmatrix}$$

$$|\lambda I + A| = \left|\begin{bmatrix}\lambda & 0\\ 0 & \lambda\end{bmatrix}-\begin{bmatrix}0 & 1\\ \frac{m}{J} & \frac{b}{J}\end{bmatrix}\right| = 0$$

$$\left|\begin{bmatrix}\lambda & -1\\ -\frac{m}{J} & \lambda-\frac{b}{J}\end{bmatrix}\right| = 0$$

$$\lambda(\lambda-\frac{b}{J})-\frac{m}{J}=0$$

$$\lambda^2-\frac{b}{J}\lambda-\frac{m}{J}=0$$

**Solution:**

$$\lambda=\frac{b\pm\sqrt{b^2+4Jm}}{2J}$$





# Control systems: the basics

## Summary

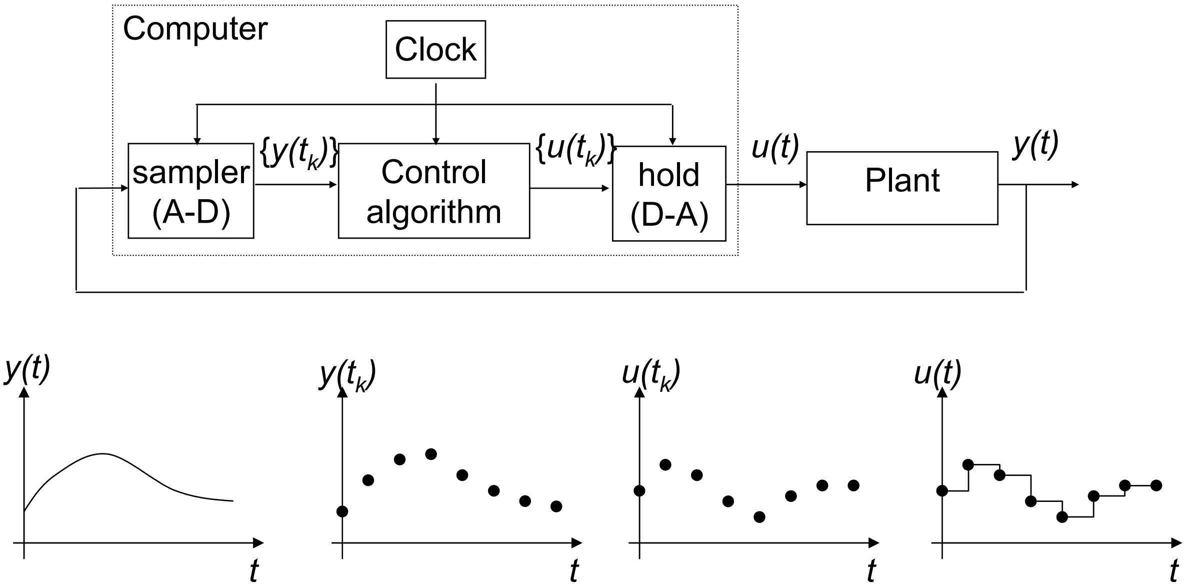
System poles determine stability
shape of the response

State space models ... continuous

Computers have a discrete view of the world !!!!

# Control systems: the basics

## Discrete time systems, models and poles







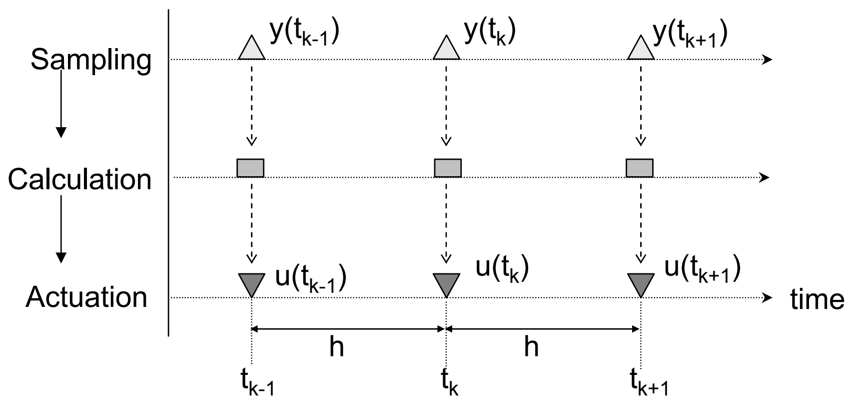

# Control systems: the basics

## Discrete time systems, models and poles

The computer has a discrete view of the plant ⇒ the control algorithm has to be designed taking into account that it will “deal with” sequences of numbers ⇒ a discrete model of the plant is required.

The most commonly used model assumes:

- equidistant sampling: **h sampling period** (seconds)
- instantaneous input-to-output latency



# Control systems: the basics

## Discrete time systems, models and poles

Continuous time model

$$\dot{\mathbf{x}}(t)=\mathbf{A}\mathbf{x}(t)+\mathbf{B}u(t)$$
$$y(t)=\mathbf{C}\mathbf{x}(t)+\mathbf{D}u(t)$$

Discrete time model

$$\mathbf{x}_{kh+h}=\mathbf{\Phi}\mathbf{x}_{kh}+\mathbf{\Gamma}u_{kh}$$
$$y_{kh}=\mathbf{C}\mathbf{x}_{kh}+\mathbf{D}u_{kh}$$

$$\mathbf{\Phi}=e^{\mathbf{A}h}$$
$$\mathbf{\Gamma}=\int_0^h e^{\mathbf{A}s}ds\mathbf{B}$$

Computing $\Phi$ and $\Gamma$:

- Symbolically
- Series expansion of the exponential matrix
- Numerically (Matlab)





# Control systems: the basics

## Discrete time systems, models and poles

$$\mathbf{\Phi}=e^{\mathbf{A}h}$$
$$\mathbf{\Gamma}=\int_0^h e^{\mathbf{A}s}ds\mathbf{B}$$

Symbolically
Rocket model
(simple model)

```
syms h;
syms s;
A=[0 1;1 1];
B=[0;1];
Phi=expm(A*h)
Gamma=int(expm(A*s),s,0,h)*B
```

$$\Phi=\begin{bmatrix} 0.2764e^{1.6180h}+0.7236e^{-0.6180h} & 0.4472e^{-0.6180h}+0.4472e^{1.6180*h} \\ -0.4472e^{-0.6180h}+0.4472e^{1.6180*h} & 0.7236e^{1.6180*h}+0.2764e^{-0.6180h} \end{bmatrix}$$

$$\Gamma=\begin{bmatrix} 0.2764e^{1.6180h}-1+0.7236e^{-0.6180h} \\ 0.4472e^{0.5h}(e^{1.1180h}-e^{-1.1180h}) \end{bmatrix}$$

h=1 $\longrightarrow$ $\Phi=\begin{bmatrix} 1.7839 & 2.0143 \\ 2.0143 & 3.7982 \end{bmatrix}$ $\Gamma=\begin{bmatrix} 0.7839 \\ 2.0143 \end{bmatrix}$

# Control systems: the basics

## Discrete time systems, models and poles

$$\mathbf{\Phi} = e^{\mathbf{A}h}$$
$$\mathbf{\Gamma} = \int_0^h e^{\mathbf{A}s} ds \mathbf{B}$$

Matlab
Rocket model
(simple model)

```
A=[0 1;1 1];
B=[0;1];
C=[1 0];
D=[0];
sys=ss(A,B,C,D);
dsys=c2d(sys,1,'zoh');
[Phi,Gamma,dC,dD]=ssdata(dsys)
```

$$\Phi = \begin{bmatrix} 1.7839 & 2.0143 \\ 2.0143 & 3.7982 \end{bmatrix}$$

$$\Gamma = \begin{bmatrix} 0.7839 \\ 2.0143 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 \end{bmatrix}$$





# Control systems: the basics

## Discrete time systems, models and poles

$$\mathbf{\Phi} = e^{\mathbf{A}h}$$
$$\mathbf{\Gamma} = \int_0^h e^{\mathbf{A}s} ds \mathbf{B}$$

Series expansion

$$\mathbf{\Phi} = \mathbf{I} + \mathbf{A}\mathbf{\Psi}$$
$$\mathbf{\Gamma} = \mathbf{\Psi}\mathbf{B}$$

$$\mathbf{\Psi} = \int_0^h e^{\mathbf{A}s} ds = \mathbf{I}h + \frac{\mathbf{A}h^2}{2!} + \frac{\mathbf{A}^2 h^3}{3!} + \ldots + \frac{\mathbf{A}^i h^{i+1}}{(i+1)!} + \ldots$$

Rocket model: we have to operate until we obtain the desired accuracy

- For the rocket model (simple model), with i=5, we obtain 2 correct decimals
- depending on the model (e.g., stable plants), the expansion works very well

# Control systems: the basics

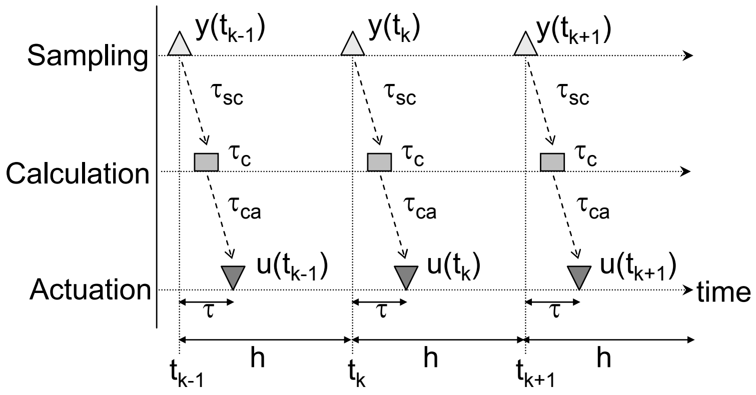
## Discrete time systems, models and poles

Note that the most commonly used model assumptions:

- equidistant sampling: **h sampling period**
- instantaneous input-to-output latency

are not real

Computations (or network) take time (input-to-output latency≠0)



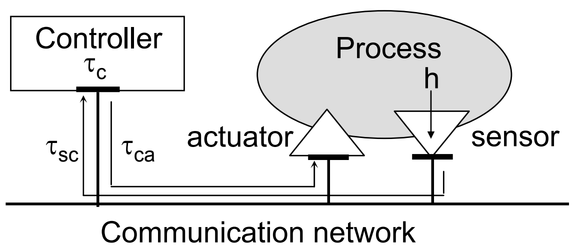



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control systems: the basics

## Discrete time systems, models and poles

**NCS (networked control systems)**



# Control systems: the basics

## Discrete time systems, models and poles

Continuous time model with delay $\tau$ ($\tau$<h)

$$\dot{\mathbf{x}}(t)=\mathbf{A}\mathbf{x}(t)+\mathbf{B}u(t-\tau)$$
$$y(t)=\mathbf{C}\mathbf{x}(t)$$

Discrete time model with delay $\tau$ ($\tau$<h)

$$\mathbf{x}_{kh+h}=\mathbf{\Phi}\mathbf{x}_{kh}+\mathbf{\Gamma}_0 u_{kh}+\mathbf{\Gamma}_1 u_{kh-h}$$
$$y_{kh}=\mathbf{C}\mathbf{x}_{kh}$$

$$\mathbf{\Phi}=e^{\mathbf{A}h}$$
$$\mathbf{\Gamma}_0=\int_0^{h-\tau} e^{\mathbf{A}s}ds\mathbf{B}$$
$$\mathbf{\Gamma}_1=e^{\mathbf{A}(h-\tau)}\int_0^{\tau} e^{\mathbf{A}s}ds\mathbf{B}$$

Computing $\Phi$, $\Gamma_0$ and $\Gamma_1$ is similar to the method already seen

State space model
z(kh)=u(kh-h)

$$\begin{bmatrix}\mathbf{x}_{kh+h}\\ \mathbf{z}_{kh+h}\end{bmatrix}=\begin{bmatrix}\mathbf{\Phi} & \mathbf{\Gamma}_1\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{x}_{kh}\\ \mathbf{z}_{kh}\end{bmatrix}+\begin{bmatrix}\mathbf{\Gamma}_0\\ \mathbf{I}\end{bmatrix}u_{kh}$$



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control systems: the basics

## State-space models

$$\mathbf{x}_{kh+h}=\mathbf{\Phi}\mathbf{x}_{kh}+\mathbf{\Gamma}u_{kh}$$
$$y_{kh}=\mathbf{C}\mathbf{x}_{kh}+\mathbf{D}u_{kh}$$

Where are the system poles? $|\lambda\mathbf{I}-\mathbf{\Phi}|=0$

Characteristic equation

**Example: rocket control**

Simple model

$$\Phi=\begin{bmatrix}1.7839 & 2.0143\\ 2.0143 & 3.7982\end{bmatrix}$$

$$|\lambda I+\Phi|=\left|\begin{bmatrix}\lambda & 0\\ 0 & \lambda\end{bmatrix}-\begin{bmatrix}1.7839 & 2.0143\\ 2.0143 & 3.7982\end{bmatrix}\right|=0$$

$$\left|\begin{bmatrix}\lambda-1.7839 & -2.0143\\ -2.0143 & \lambda-3.7982\end{bmatrix}\right|=0$$

$$(\lambda-1.7839)(\lambda-3.7982)-(-2.0143)^2=0$$

$$\lambda_1=0.5390$$

$$\lambda_2=5.0432$$

# Control systems: the basics

## Discrete time systems, models and poles

**Correspondence between continuous time poles vs. discrete time poles: Cruise control example**

$$\dot{v}(t)=[-0.05]v(t)+[0.001]u(t)$$
$$y=[1]v(t)$$

if h=1 $\longrightarrow$

$$v_{k+1}=[0.9512]v_k+[0.0009754]u_k$$
$$y=[1]v_k$$

$$\left[\begin{array}{l}\mathbf{\Phi}=e^{\mathbf{A}h}\\ \mathbf{\Phi}=e^{-0.05}=0.9512\end{array}\right]$$

$$|\lambda\mathbf{I}-\mathbf{A}|=0$$
$$|\lambda-(-0.05)|=0$$
$$\lambda=-0.05$$

$\longrightarrow$

$$|\lambda\mathbf{I}-\Phi|=0$$
$$|\lambda-(0.9512)|=0$$
$$\lambda=0.9512$$

**Mapping**

$$\lambda_d=e^{\lambda_c h}$$

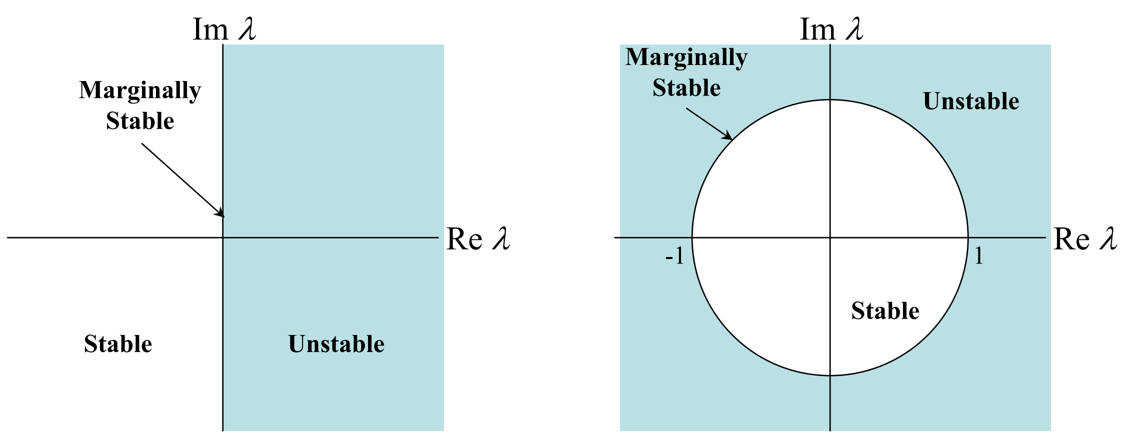




# Control systems: the basics

## Discrete time systems, models and poles

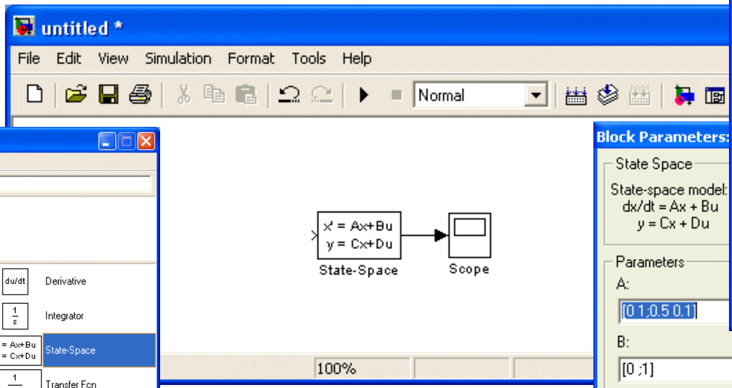
**Correspondence between continuous time poles vs. discrete time poles**

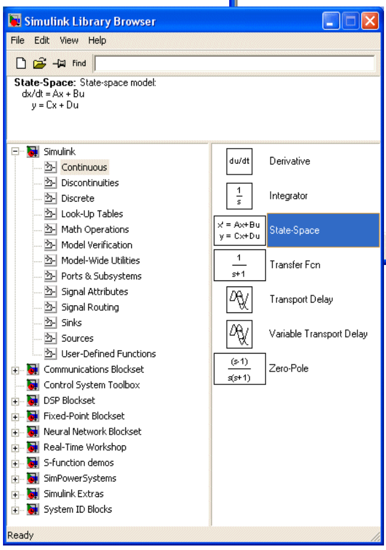


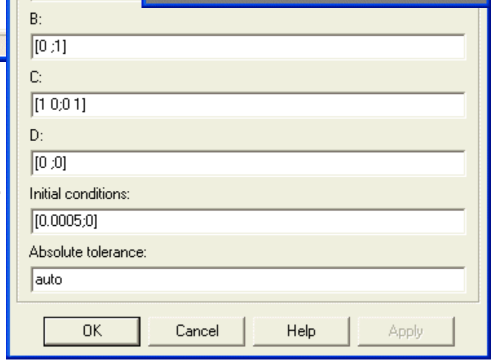
# Control systems: the basics

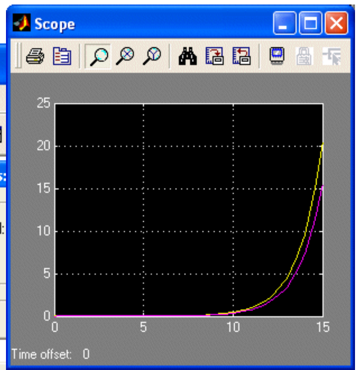
## Simulation

**Simulink**




**continuous**

$$\begin{bmatrix}\dot{\theta}_1(t)\\ \dot{\theta}_2(t)\end{bmatrix}=\begin{bmatrix}0 & 1\\ 0.5 & 0.1\end{bmatrix}\begin{bmatrix}\theta_1(t)\\ \theta_2(t)\end{bmatrix}+\begin{bmatrix}0\\ 1\end{bmatrix}M_C(t)$$

$$y(t)=\begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}\theta_1(t)\\ \theta_2(t)\end{bmatrix}$$

ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control systems: the basics

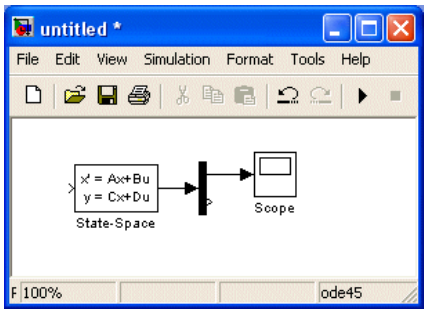
## Simulation

**Observation:**
System response is a solution

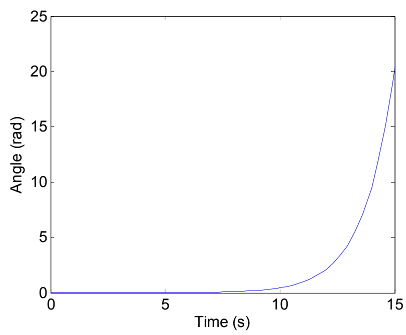
**continuous**

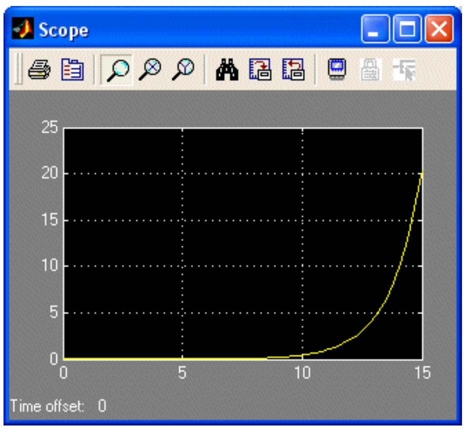
$$\begin{bmatrix}\dot{\theta}_1(t)\\ \dot{\theta}_2(t)\end{bmatrix}=\begin{bmatrix}0 & 1\\ 0.5 & 0.1\end{bmatrix}\begin{bmatrix}\theta_1(t)\\ \theta_2(t)\end{bmatrix}+\begin{bmatrix}0\\ 1\end{bmatrix}M_C(t)$$

$$y(t)=\begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}\theta_1(t)\\ \theta_2(t)\end{bmatrix}$$




t=0:15
plot(t,0.0002323*exp(0.7589*t)+0.0002676*exp(-0.6589*t))

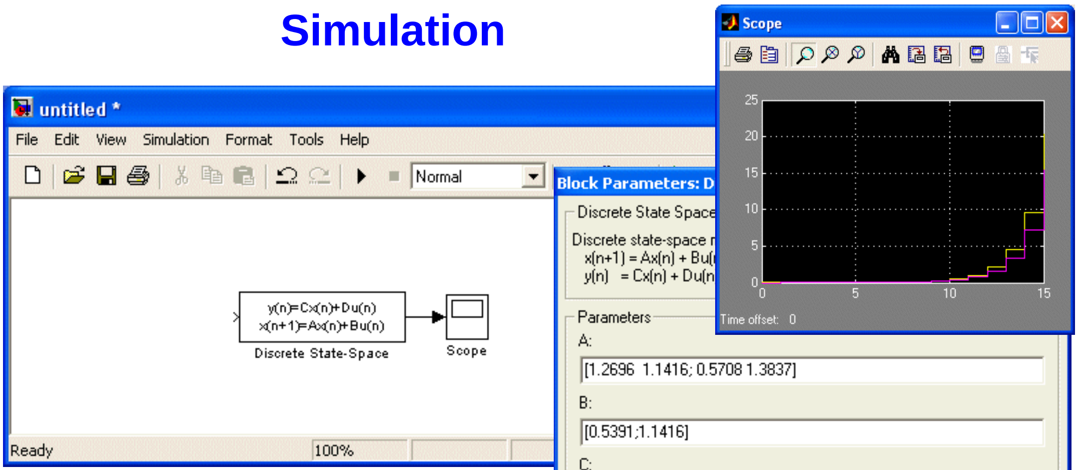


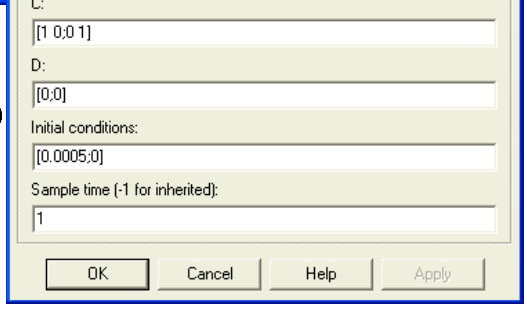



# Control systems: the basics

## Simulation

**Simulink**




**discrete**

$$\begin{bmatrix}\theta_1(k+1)\\ \theta_2(k+1)\end{bmatrix} = \begin{bmatrix}1.2696 & 1.1416\\ 0.5708 & 1.3837\end{bmatrix}\begin{bmatrix}\theta_1(k)\\ \theta_2(k)\end{bmatrix} + \begin{bmatrix}0.5391\\ 1.1416\end{bmatrix} M_C(k)$$

$$y(k) = \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}\theta_1(k)\\ \theta_2(k)\end{bmatrix}$$

ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control systems: the basics

## Simulation

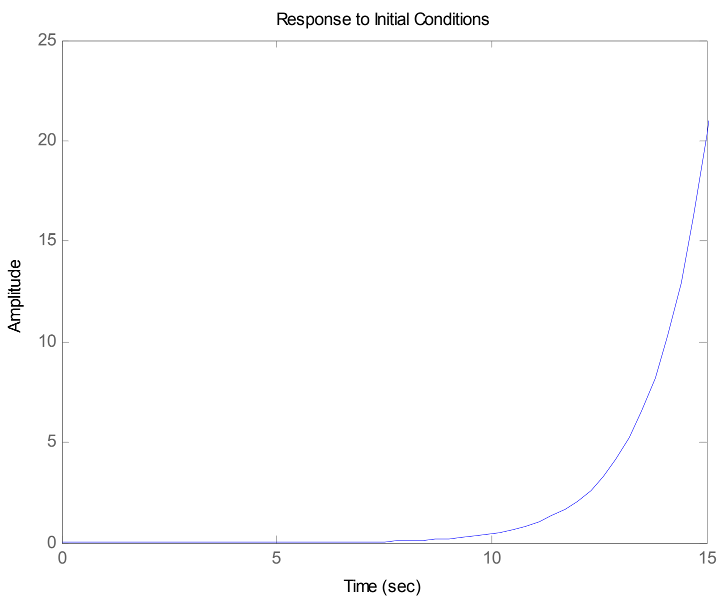
**Matlab**

**continuous**

$$\begin{bmatrix}\dot\theta_1(t)\\ \dot\theta_2(t)\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0.5 & 0.1\end{bmatrix}\begin{bmatrix}\theta_1(t)\\ \theta_2(t)\end{bmatrix} + \begin{bmatrix}0\\ 1\end{bmatrix} M_C(t)$$

$$y(t) = \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}\theta_1(t)\\ \theta_2(t)\end{bmatrix}$$

```
A=[0 1;0.5 0.1];
B=[0;1];
C=[1 0];
D=[0];
sys=ss(A,B,C,D);
initial(sys,[0.0005;0],15)
```



# Control systems: the basics

## Simulation

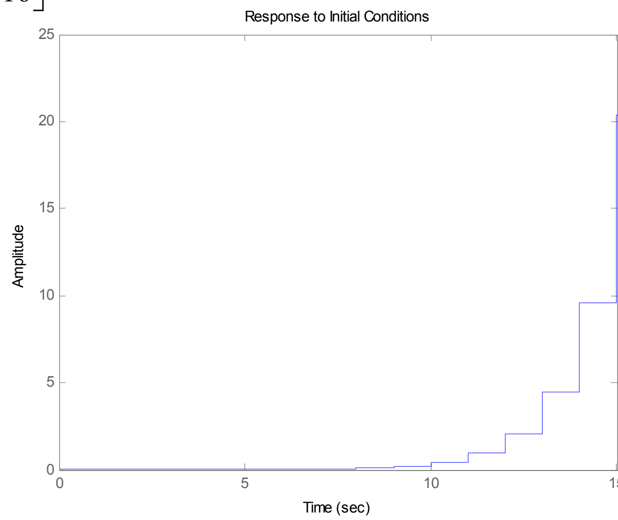
**Matlab**

**discrete**

$$\begin{bmatrix}\theta_1(k+1)\\ \theta_2(k+1)\end{bmatrix}=\begin{bmatrix}1.2696 & 1.1416\\ 0.5708 & 1.3837\end{bmatrix}\begin{bmatrix}\theta_1(k)\\ \theta_2(k)\end{bmatrix}+\begin{bmatrix}0.5391\\ 1.1416\end{bmatrix}M_C(k)$$

$$y(k)=\begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}\theta_1(k)\\ \theta_2(k)\end{bmatrix}$$

```
A=[0 1;0.5 0.1];
B=[0;1];
C=[1 0];
D=[0];
sys=ss(A,B,C,D);
dsys=c2d(sys,1,'zoh');
initial(dsys,[0.0005;0],15)
```






ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control systems: the basics

## Control design

The objective of control is to shape the system response

Pole placement by state feedback allows to specify arbitrary pole locations

**Example** (Open loop) plant (simple model)

$$\begin{bmatrix}\theta_1(k+1)\\ \theta_2(k+1)\end{bmatrix}=\begin{bmatrix}1.7893 & 2.0143\\ 2.0143 & 3.7982\end{bmatrix}\begin{bmatrix}\theta_1(k)\\ \theta_2(k)\end{bmatrix}+\begin{bmatrix}0.7839\\ 2.0143\end{bmatrix}M_C(k)$$

A linear feeback can be described by

$$M_C(k)=-l_1\theta_1(k)-l_1\theta_1(k)=-\begin{bmatrix}l_1 & l_2\end{bmatrix}\begin{bmatrix}\theta_1(k)\\ \theta_2(k)\end{bmatrix}$$

With this feedback, the closed loop system becomes

$$\begin{bmatrix}\theta_1(k+1)\\ \theta_2(k+1)\end{bmatrix}=\begin{bmatrix}1.7893 & 2.0143\\ 2.0143 & 3.7982\end{bmatrix}\begin{bmatrix}\theta_1(k)\\ \theta_2(k)\end{bmatrix}+\begin{bmatrix}0.7839\\ 2.0143\end{bmatrix}\left(-\begin{bmatrix}l_1 & l_2\end{bmatrix}\begin{bmatrix}\theta_1(k)\\ \theta_2(k)\end{bmatrix}\right)$$

# Control systems: the basics

## Controller design

(example continuation)

$$\begin{bmatrix} \theta_1(k+1) \\ \theta_2(k+1) \end{bmatrix} = \begin{bmatrix} 1.7893 - 0.7839 l_1 & 2.0143 - 0.7839 l_2 \\ 2.0143 - 2.0143 l_1 & 3.7982 - 2.0143 l_2 \end{bmatrix} \begin{bmatrix} \theta_1(k) \\ \theta_2(k) \end{bmatrix}$$

Characteristic equation $|\lambda \mathbf{I} - \mathbf{\Phi}| = 0$

$$\left| \lambda \mathbf{I} - \begin{bmatrix} 1.7893 - 0.7839 l_1 & 2.0143 - 0.7839 l_2 \\ 2.0143 - 2.0143 l_1 & 3.7982 - 2.0143 l_2 \end{bmatrix} \right| = 0$$

$$\lambda^2 + (0.7839 l_1 - 5.5821 + 2.0143 l_2)\lambda + 2.7182 + 1.0799 l_1 - 2.0143 l_2 = 0$$

We know that stability requires to have system poles $|p_i|<0$. Let's arbitrarely choose

$$p_1 = 0.4 + 0.4i \quad \text{and} \quad p_2 = 0.4 - 0.4i$$

The desired characteristic equation is $(\lambda - (0.4+0.4i))(\lambda - (0.4-0.4i)) = \lambda^2 - 0.8\lambda + 0.32 = 0$





# Control systems: the basics

## Controller design

(example continuation)

Characteristic equation

$$\lambda^2 + (0.7839 l_1 - 5.5821 + 2.0143 l_2)\lambda + 2.7182 + 1.0799 l_1 - 2.0143 l_2 = 0$$

The desired characteristic equation is

$$\lambda^2 - 0.8\lambda + 0.32 = 0$$

Equating equations coefficients $\left.\begin{aligned} 0.7839 l_1 - 5.5821 + 2.0143 l_2 &= -0.8 \\ 2.7182 + 1.0799 l_1 - 2.0143 l_2 &= 0.32 \end{aligned}\right\} \quad \begin{aligned} l_1 &= 1.2790 \\ l_2 &= 1.8763 \end{aligned}$

Closed loop system equation

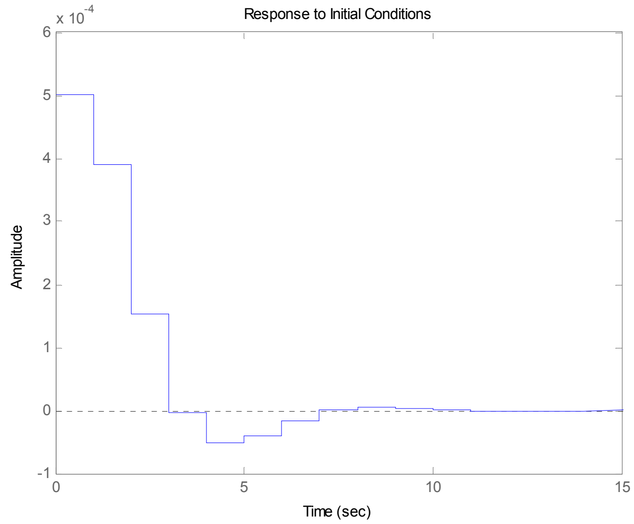
$$\begin{bmatrix} \theta_1(k+1) \\ \theta_2(k+1) \end{bmatrix} = [\Phi - \Gamma L] \begin{bmatrix} \theta_1(k) \\ \theta_2(k) \end{bmatrix} = \begin{bmatrix} 0.7813 & 0.5434 \\ -0.5620 & 0.0187 \end{bmatrix} \begin{bmatrix} \theta_1(k) \\ \theta_2(k) \end{bmatrix}$$

# Control systems: the basics

## Controller design

Matlab (simple model)

```
A=[0 1;1 1]; %Simple model
B=[0;1];
C=[1 0];
D=[0];
sys=ss(A,B,C,D);
initial(sys,[0.0005;0],10);
dsys=c2d(sys,1,'zoh');
[Phi,Gamma,dC,dD]=ssdata(dsys);
poles=[0.4 + 0.4i,0.4 - 0.4i];
Ld=acker(Phi,Gamma,poles);
cldsys=ss(Phi-Gamma*Ld,Gamma,dC,dD,1);
initial(cldsys,[0.0005;0])    //WRONG//
```





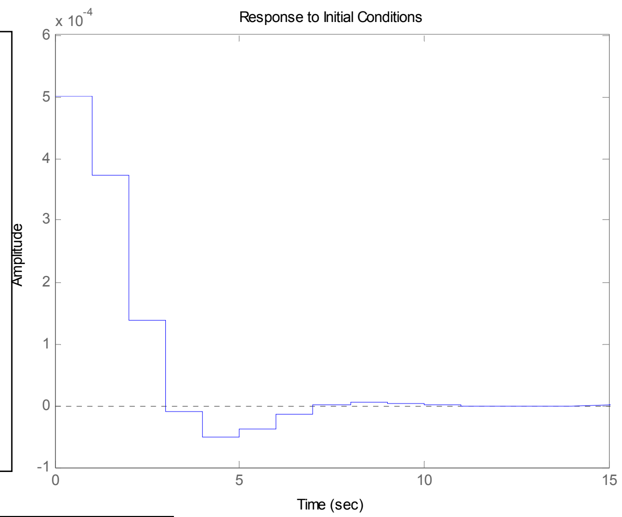

ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control systems: the basics

## Controller design

Matlab

```
A=[0 1;0.5 0.1];
B=[0;1];
C=[1 0];
D=[0];
sys=ss(A,B,C,D);
initial(sys,[0.0005;0],10);
dsys=c2d(sys,1,'zoh');
[Phi,Gamma,dC,dD]=ssdata(dsys);
poles=[0.4 + 0.4i,0.4 - 0.4i];
Ld=acker(Phi,Gamma,poles)
cldsys=ss(Phi-Gamma*Ld,Gamma,dC,dD,1);
initial(cldsys,[0.0005;0])    //WRONG//
```

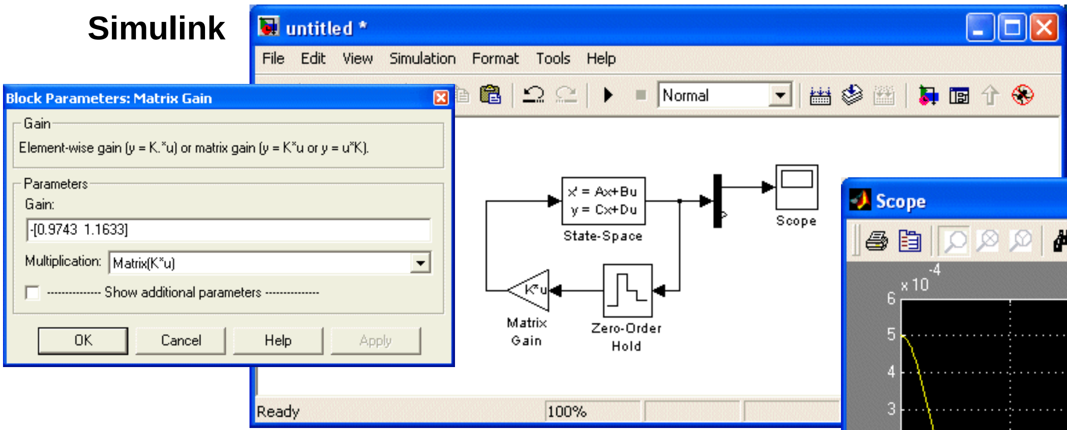


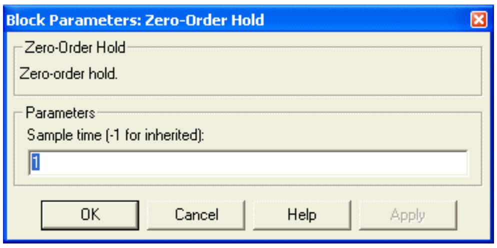

Ld=[0.9743  1.1633]

# Control systems: the basics

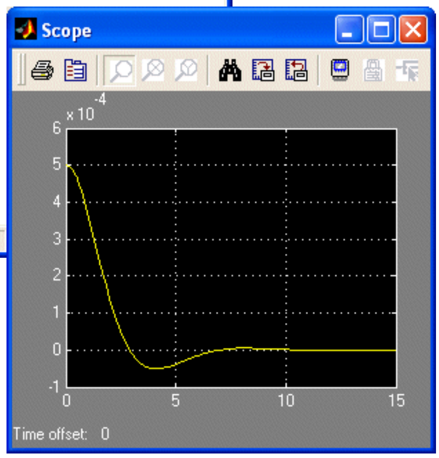
## Controller design

**Simulink**




State-space with intial conditions $x_0$=[0.0005;0]



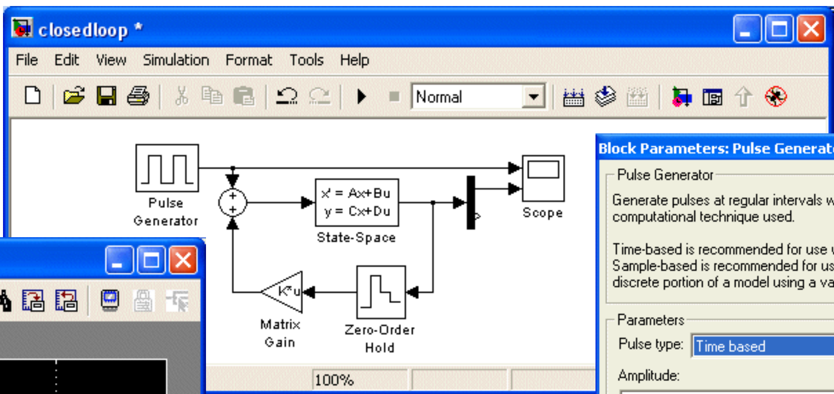


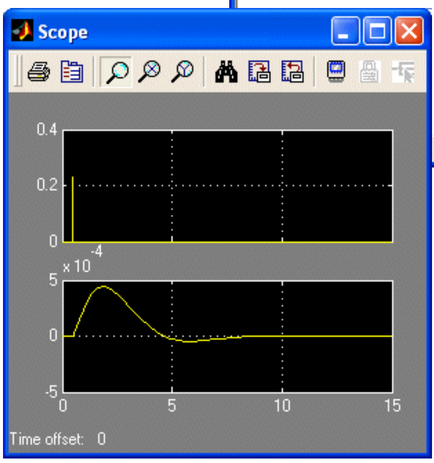

ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control systems: the basics

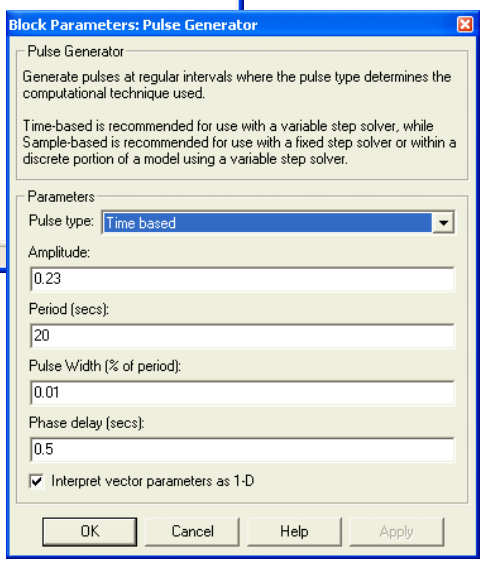
## Controller design

**Simulink**




State-space with intial conditions $x_0$=[0;0]



# Control systems: the basics

## Controller design

**Summary**

- Controller design     L=acker($\Phi$,$\Gamma$,[$p_1$, $p_2$, ..., $p_n$])
- $p_i$ are the desired pole locations of the closed-loop system

- Questions:     Where do we place the desired poles?

  What about the sampling period or execution time (i.e., control tasks timing constraints)?





# Real-time computing of control systems

1. Controller implementation
2. Selection of the control task period
3. Trading off performance vs. schedulability
4. The jitter problem
5. Flexible timing constraints

# Real-time computing of control systems

## Controller implementation

The controller is an algorithm to be executet at each sampling period *h*

***Controller task***

{ *read_input* ($y_k$) *//assuming that* $\mathbf{y_k} = \mathbf{x}_k$
$u_k = -\mathbf{L}\mathbf{x}_k$
*write_output* ($u_k$)
}

Which is the right code?





# Real-time computing of control systems

## Controller implementation

The controller is an algorithm to be executet at each sampling period *h*

**Attempt 1:**

```
LOOP
        PeriodicActivity;
        WaitTime(h);
END;
```

- Does not work.
- Both, Period *h* and PeriodicActivity, time-varying.
- The execution time of PeriodicActivity is not accounted for.

# Real-time computing of control systems

## Controller implementation

The controller is an algorithm to be executed at each sampling period *h*

**Attempt 2:**

```
LOOP
        Start = CurrentTime();
        PeriodicActivity;
        Stop = CurrentTime();
        C := Stop - Start;
        WaitTime(h - C);
END;
```

- Does not work. An interrupt causing suspension may occur between the assignment and WaitTime.
- In general, a WaitTime (Delay) primitive is not enough to implement periodic processes correctly.
- A WaitUntil (DelayUntil) primitive is needed.



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## Controller implementation

The controller is an algorithm to be executed at each sampling period *h*

**Attempt 3:**

```
t = CurrentTime();
LOOP
    PeriodicActivity;
    t = t + h;
    WaitUntil(t);
END;
```

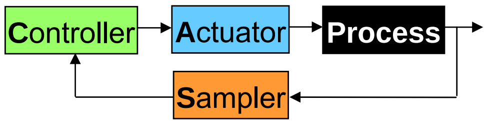
- Will try to catch up if the actual execution time of PeriodicActivity occasionally becomes larger than the period (a too long period is followed by a shorter one to make the average correct)

- Reasonable for alarm clocks, but perhaps not for controllers.

# Real-time computing of control systems

## Controller implementation

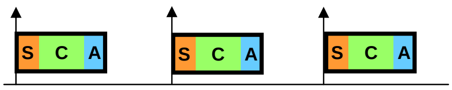
**Computer-control theory timing assumptions:**




- Equidistant sampling (constant *sampling period*, h)
- Small and constant *time delay* ($\tau$)

### Control task model (1/2)

*Sampling*, *control algorithm computation,* and *actuation* are implemented sequentially within a single periodic task.



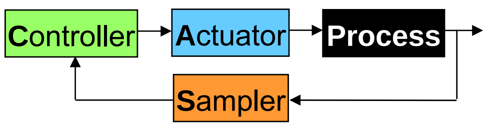





# Real-time computing of control systems

## Controller implementation

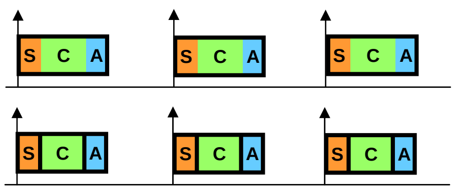
**Computer-control theory timing assumptions:**




- Equidistant sampling (constant *sampling period*, h)
- Small and constant *time delay* ($\tau$)

### Control task model (2/2)

*Sampling*, *control algorithm computation,* and *actuation* are implemented in several periodic tasks.



# Real-time computing of control systems

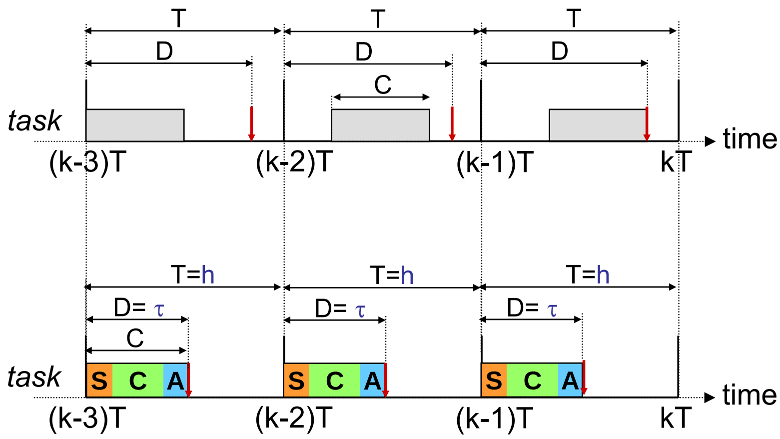
## Controller implementation

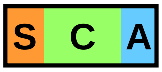
Keeping control timing assumptions with real-time task timing constraints

### Standard timing constraints for periodic tasks

- Period, T
- Deadline, D

(Computation time, C)




**Timing constraints for control tasks**

Period T = *Sampling period*, h

Deadline D = *Time delay*, $\tau$ (=C)

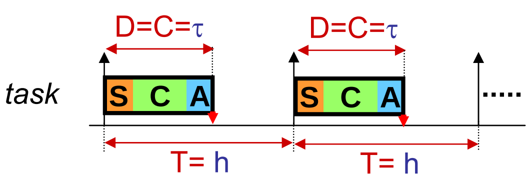


ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## Controller implementation

### Limits control task scheduling




$\Rightarrow$

**Scheduling *n* control tasks**

$n = 1$ ok

$n > 1$ in the general case, not possible

## Problem

Computer-control theory timing assumptions

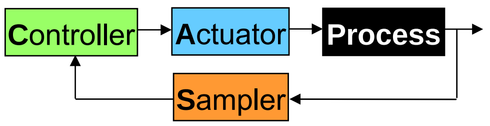
Over-constrain control task specification

Poor (or infeasible) system schedulability

# Real-time computing of control systems

## Controller implementation

**Computer-control theory timing assumptions:**




- Equidistant sampling (constant *sampling period*, h)
- Small and constant *time delay* ($\tau$)

**Control task model (new)**

*Sampling* and *actuation are performed by hardware interrupts*, *control algorithm computation* is implemented in a periodic tasks.

At the end, it will present the same problems as the previous models

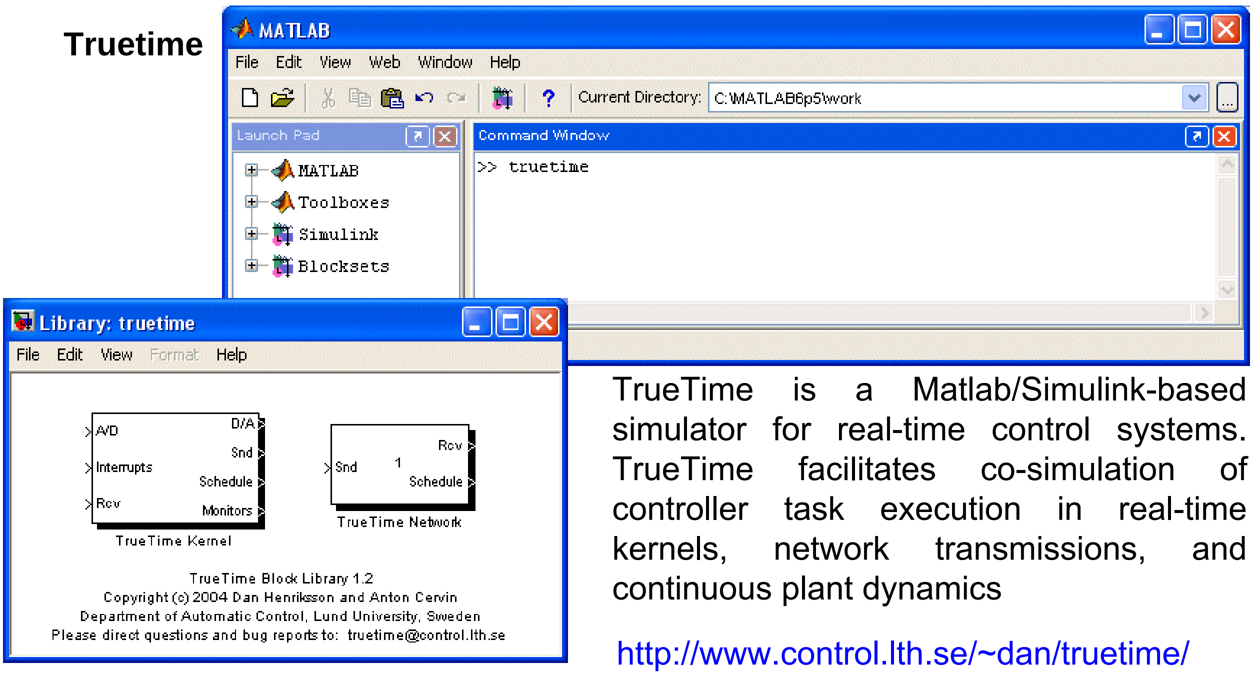


ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## Controller implementation

**Truetime**



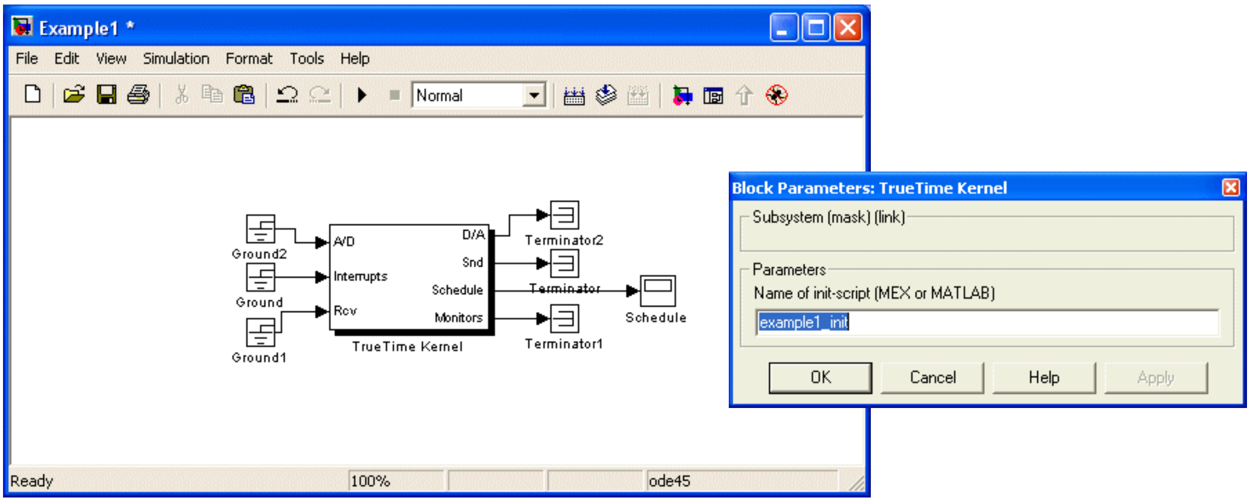

TrueTime is a Matlab/Simulink-based simulator for real-time control systems. TrueTime facilitates co-simulation of controller task execution in real-time kernels, network transmissions, and continuous plant dynamics

http://www.control.lth.se/~dan/truetime/

# Real-time computing of control systems

## Controller implementation

**Truetime *Example1*:** a periodic task with period 10ms, execution time 2ms, under RM



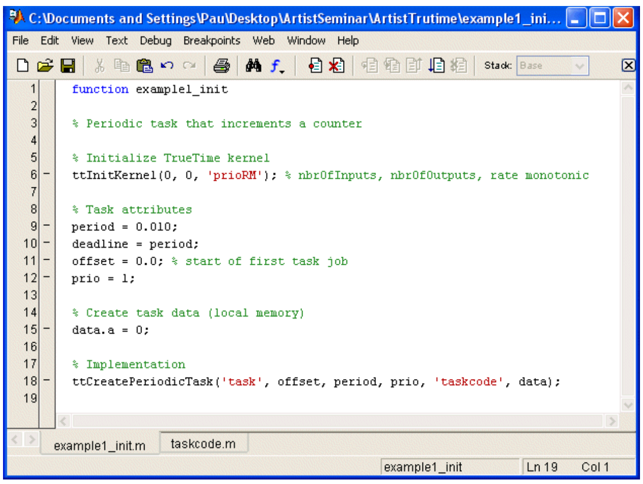



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

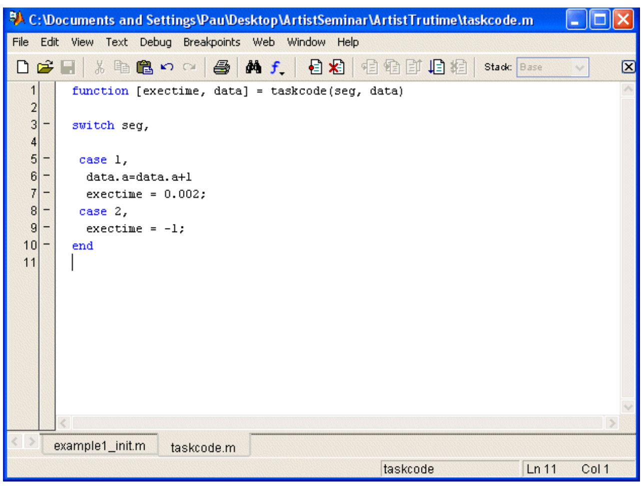
# Real-time computing of control systems

## Controller implementation

**Truetime: *Example1*:** a periodic task with period 10ms, execution time 2ms, under RM, that increments a counter




example1_init.m




taskcode.m

# Real-time computing of control systems

## Controller implementation

**Truetime: *Example1*:** a periodic task with period 10ms, execution time 2ms, under RM, that increments a counter

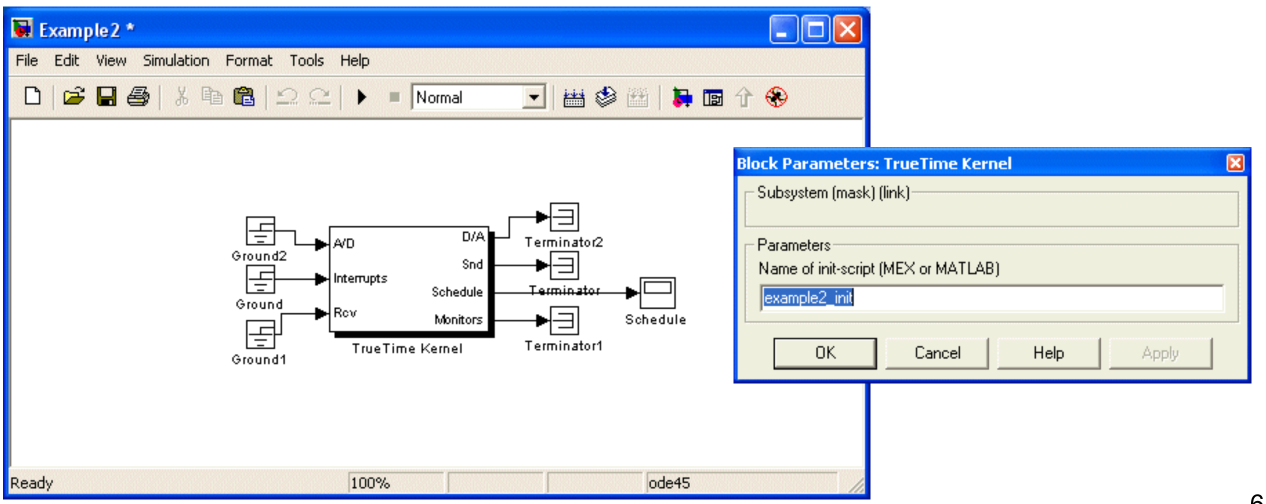



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## Controller implementation

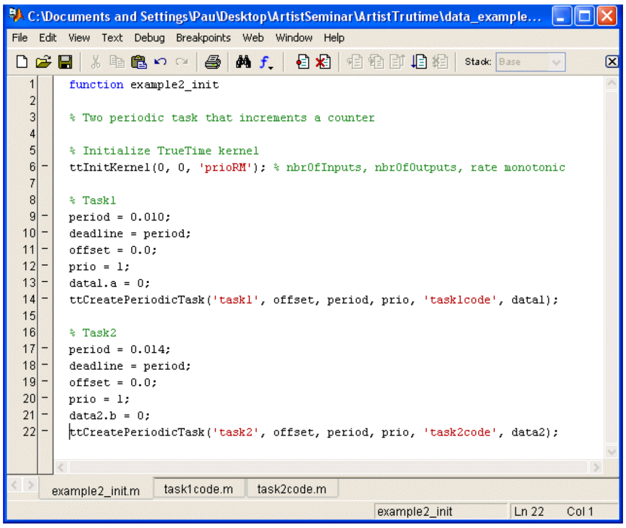
**Truetime: *Example2*:** two periodic tasks with periods 10 and 14ms, execution times 2 and 4ms, under RM, each one increments a counter

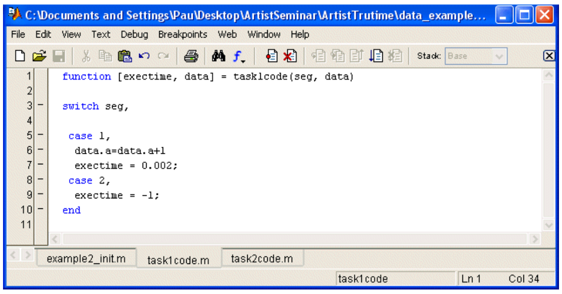


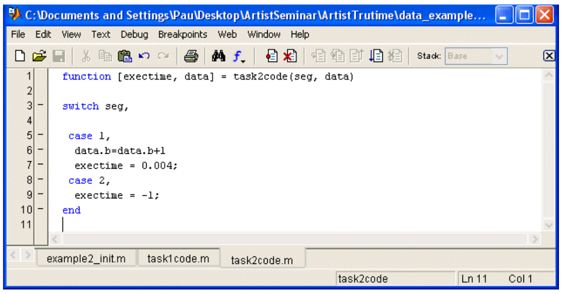
# Real-time computing of control systems

## Controller implementation

**Truetime: *Example2*:** two periodic tasks with periods 10 and 14ms, execution times 2 and 4ms, under RM, each one increments a counter



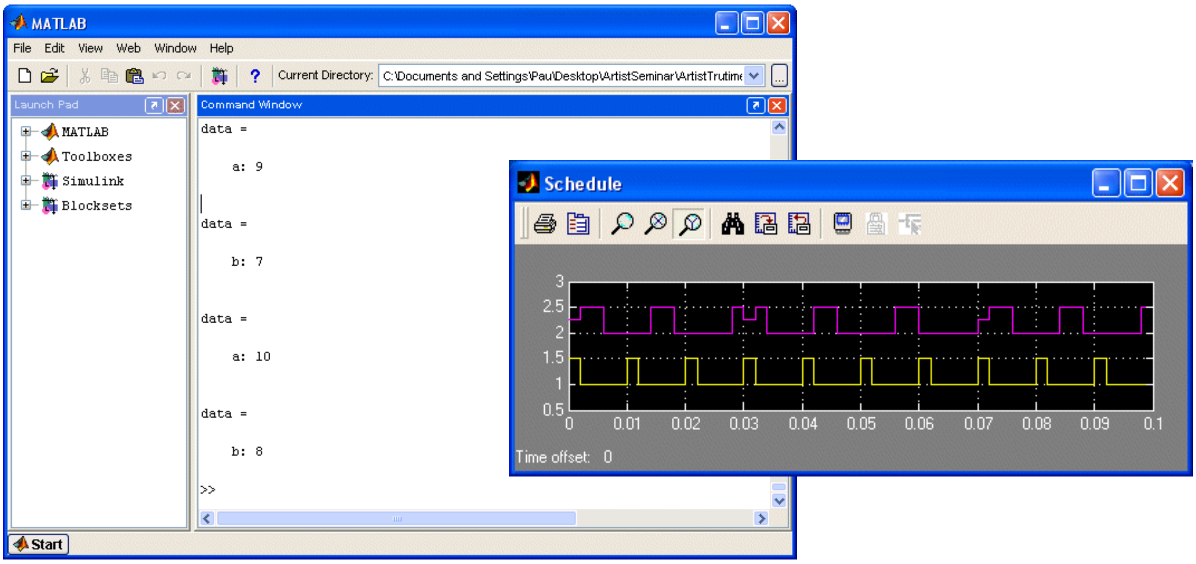



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## Controller implementation

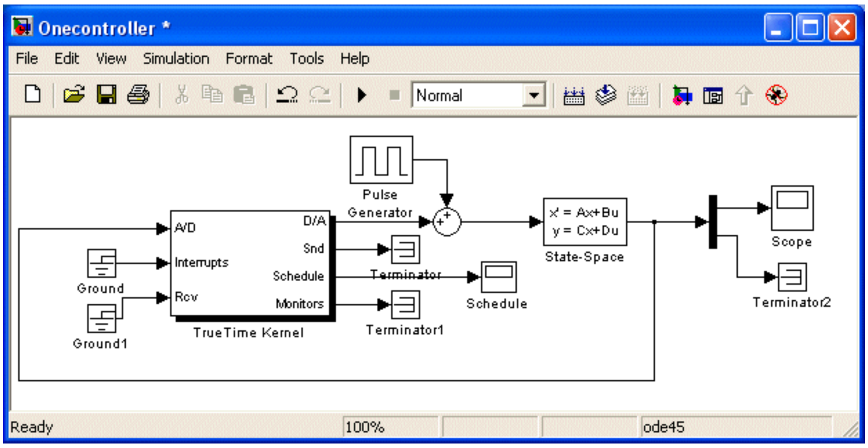
**Truetime: *Example2*:** two periodic tasks with periods 10 and 14ms, execution times 2 and 4ms, under RM, each one increments a counter



# Real-time computing of control systems

## Controller implementation

**Truetime: *Example3*:** the rocket controller task



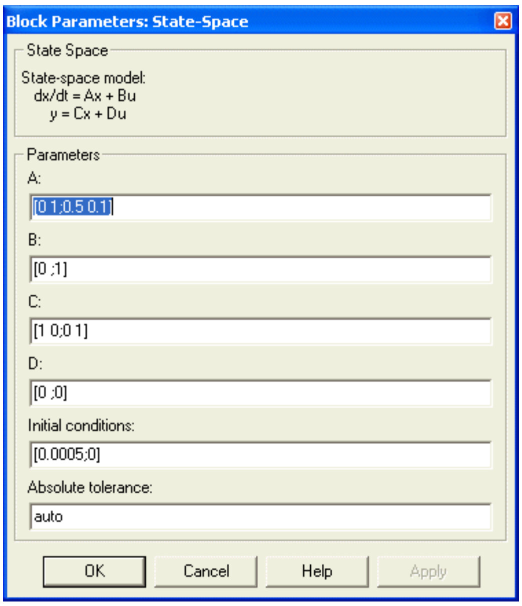


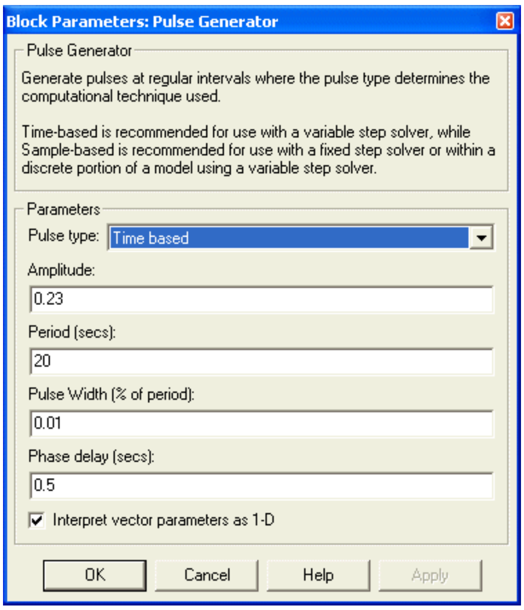



# Real-time computing of control systems

## Controller implementation

**Truetime: *Example3*:** the rocket controller task

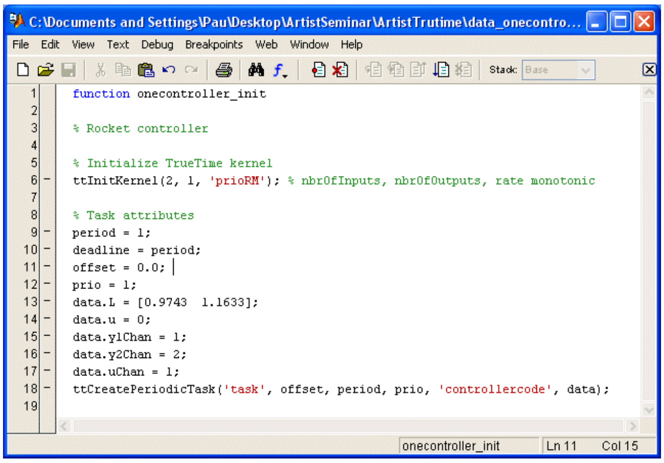


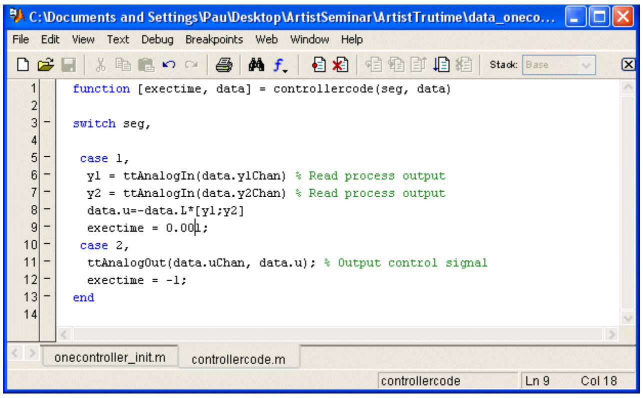



# Real-time computing of control systems

## Controller implementation

**Truetime: *Example3*:** the rocket controller task



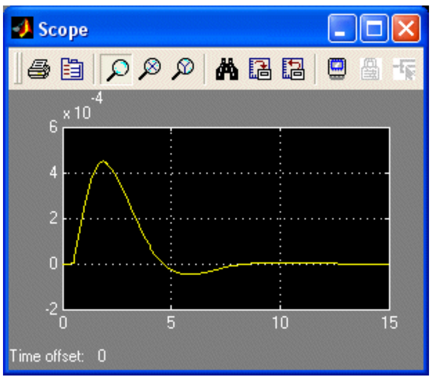


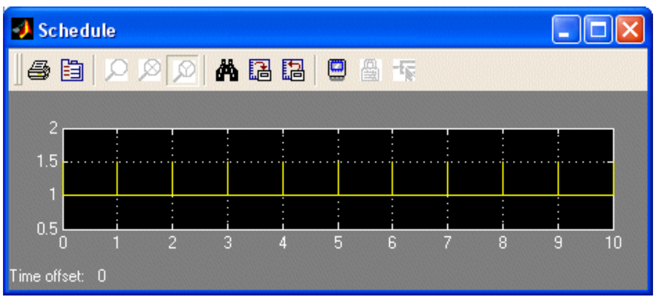

ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## Controller implementation

**Truetime: *Example3*:** the rocket controller task



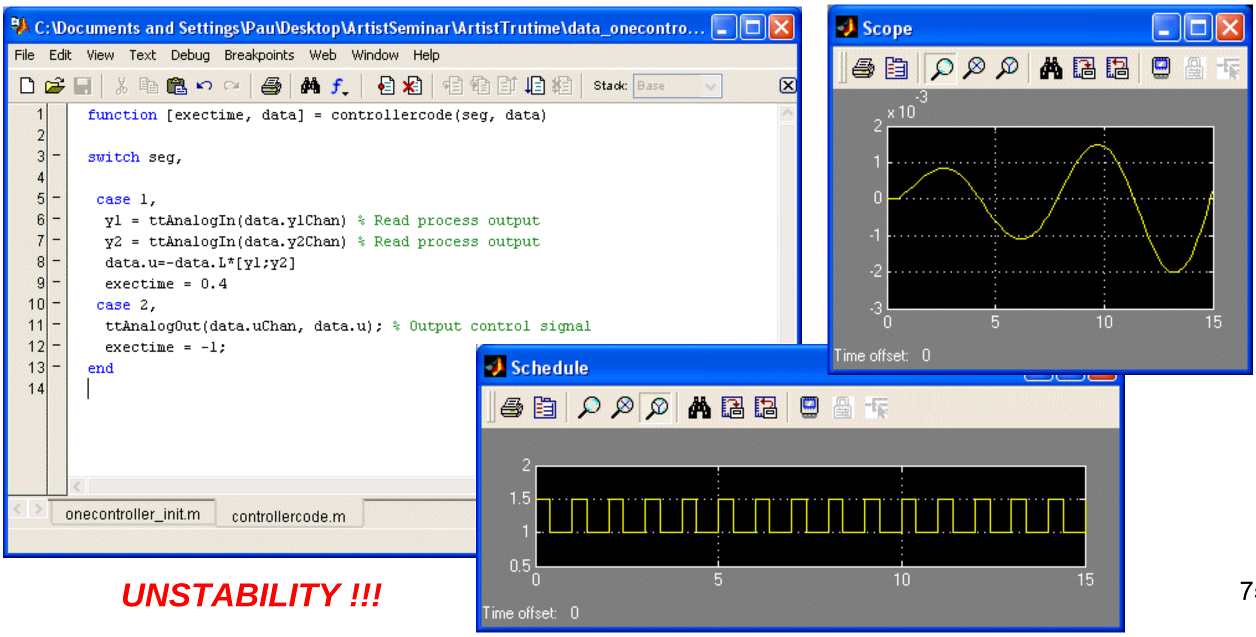




- What happens when the delay increases? Why (check poles)?

# Real-time computing of control systems

## Controller implementation

**Truetime: *Example3*:** the rocket controller task (with delay of 0.4s)




**UNSTABILITY !!!**



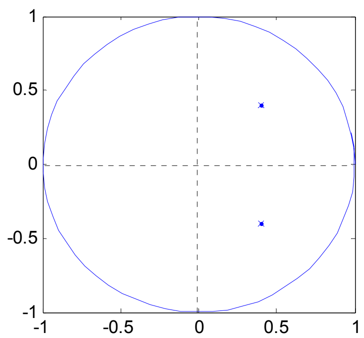


# Real-time computing of control systems

## Controller implementation

**Analysis: *Example3*:** the rocket controller task (with delay of 0.4s)

Poles of the closed loop system without delays



```
A=[0 1;0.5 0.1];B=[0;1];C=[1 0];D=[0];
sys=ss(A,B,C,D)
dsys=c2d(sys,1,'zoh');
[Phi,Gamma,dC,dD]=ssdata(dsys)
poles=[0.4 + 0.4i,0.4 - 0.4i]
Ld=acker(Phi,Gamma,poles)
cldsys=ss(Phi-Gamma*Ld,Gamma,dC,dD,1);
p=pole(cldsys);
t=0:1/10:6.5;
y=sin(t);
x=cos(t);
plot(x,y)
hold on
plot(real(p(1)),imag(p(1)),'Marker','x')
plot(real(p(2)),imag(p(2)),'Marker','x')
hold off
```

# Real-time computing of control systems

## Controller implementation

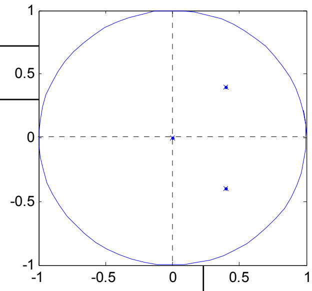
**Analysis : *Example3*:** the rocket controller task (with delay of 0.4s)

Poles of the closed loop system without delays: system with delay

$$\begin{bmatrix} \mathbf{x}_{kh+h} \\ \mathbf{z}_{kh+h} \end{bmatrix} = \left( \begin{bmatrix} \mathbf{\Phi} & \mathbf{\Gamma}_1 \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{\Gamma}_0 \\ \mathbf{I} \end{bmatrix} \begin{bmatrix} -\mathbf{L} & \mathbf{0} \end{bmatrix} \right) \begin{bmatrix} \mathbf{x}_{kh} \\ \mathbf{z}_{kh} \end{bmatrix}$$

```
h=1
t=0   % delay
syms s;
Ad=expm(A*h)
Bd0=int(expm(A*s),s,0,h-t)*B
Bd1=exp(A*(h-t))*int(expm(A*s),s,0,t)*B
Phiorg=subs(Ad,{h,t},{1,0})
Gamma0=subs(Bd0,{h,t},{1,0})
Gamma1=subs(Bd1,{h,t},{1,0})
Phi=[Phiorg Gamma1;[0 0] [0]]
Gamma=[Gamma0 ;[1]]
dC=[1 0 0];
dD=[0];
cldsysdelay=ss(Phi-Gamma*[0.9743 1.1633 0],Gamma,dC,dD,1)
```

```
p=pole(cldsysdelay);
t=0:1/10:6.5;
y=sin(t);
x=cos(t);
plot(x,y)
hold on
plot(real(p(1)),imag(p(1)),'Marker','x')
plot(real(p(2)),imag(p(2)),'Marker','x')
plot(real(p(2)),imag(p(3)),'Marker','x')
hold off
```






ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

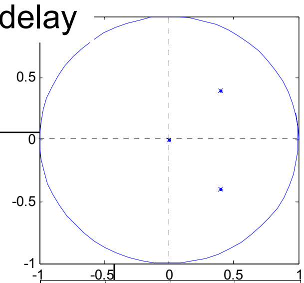
## Controller implementation

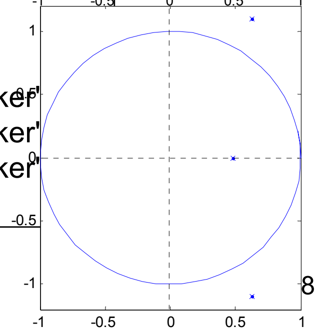
**Analysis : *Example3*:** the rocket controller task (with delay of 0.4s)

Poles of the closed loop system without delays: system with delay

```
h=1
t=0.4   % delay
syms s;
Ad=expm(A*h)
Bd0=int(expm(A*s),s,0,h-t)*B
Bd1=exp(A*(h-t))*int(expm(A*s),s,0,t)*B
Phiorg=subs(Ad,{h,t},{1,0.4})
Gamma0=subs(Bd0,{h,t},{1,0.4})
Gamma1=subs(Bd1,{h,t},{1,0.4})
Phi=[Phiorg Gamma1;[0 0] [0]]
Gamma=[Gamma0 ;[1]]
dC=[1 0 0];
dD=[0];
cldsysdelay=ss(Phi-Gamma*[0.9743 1.1633 0],Gamma,dC,dD,1)
```

```
p=pole(cldsysdelay);
t=0:1/10:6.5;
y=sin(t);
x=cos(t);
plot(x,y)
hold on
plot(real(p(1)),imag(p(1)),'Marker
plot(real(p(2)),imag(p(2)),'Marker
plot(real(p(2)),imag(p(3)),'Marker
hold off
```



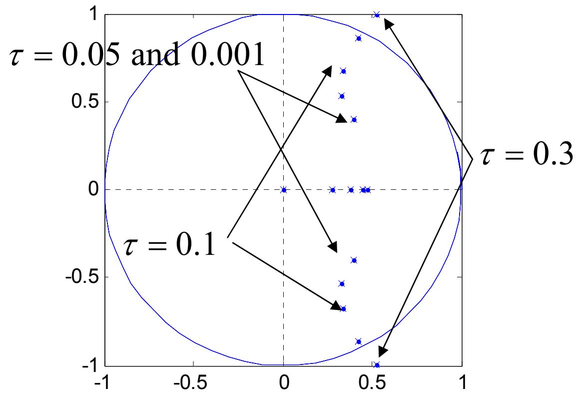
# Real-time computing of control systems

## Controller implementation

**Analysis : *Example3*:** the rocket controller task (with delay up to 0.3s)

Poles of the closed loop system without delays: system with delay ($z_{kh}=u_{kh-h}$)

$$\begin{bmatrix} \mathbf{x}_{kh+h} \\ \mathbf{z}_{kh+h} \end{bmatrix} = \left( \begin{bmatrix} \mathbf{\Phi} & \mathbf{\Gamma}_1 \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{\Gamma}_0 \\ \mathbf{I} \end{bmatrix} \begin{bmatrix} -\mathbf{L} & \mathbf{0} \end{bmatrix} \right) \begin{bmatrix} \mathbf{x}_{kh} \\ \mathbf{z}_{kh} \end{bmatrix}$$




Poles movement (h=1s, $\tau$=0.001, 0.05, 0.1, 0.2, 0.3s)



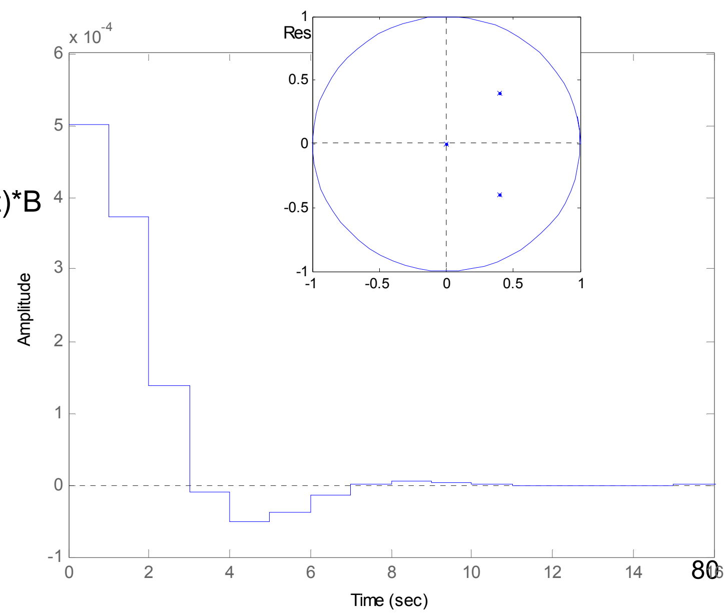


# Real-time computing of control systems

## Controller implementation

**Analysis : *Example3*:** the rocket controller task (with delay of 0.4s)

```
h=1
t=0
syms s;
Ad=expm(A*h)
Bd0=int(expm(A*s),s,0,h-t)*B
Bd1=exp(A*(h-t))*int(expm(A*s),s,0,t)*B
Phiorg=subs(Ad,{h,t},{1,0})
Gamma0=subs(Bd0,{h,t},{1,0})
Gamma1=subs(Bd1,{h,t},{1,0})
Phi=[Phiorg Gamma1;[0 0] [0]]
Gamma=[Gamma0 ;[1]]
dC=[1 0 0];
dD=[0];
cldsysdelay=ss(Phi-Gamma*
[0.9743 1.1633 0],Gamma,dC,dD,1)
initial(cldsysdelay,[0.0005;0;0])
```

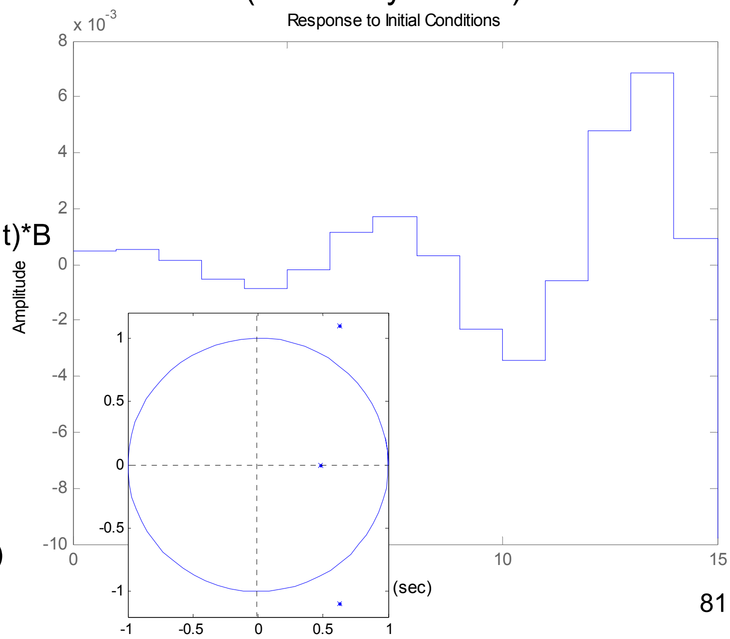


# Real-time computing of control systems

## Controller implementation

**Analysis : *Example3*:** the rocket controller task (with delay of 0.4s)

```
h=1
t=0.4
syms s;
Ad=expm(A*h)
Bd0=int(expm(A*s),s,0,h-t)*B
Bd1=exp(A*(h-t))*int(expm(A*s),s,0,t)*B
Phiorg=subs(Ad,{h,t},{1,0.4})
Gamma0=subs(Bd0,{h,t},{1,0.4})
Gamma1=subs(Bd1,{h,t},{1,0.4})
Phi=[Phiorg Gamma1;[0 0] [0]]
Gamma=[Gamma0 ;[1]]
dC=[1 0 0];
dD=[0];
cldsysdelay=ss(Phi-Gamma*
[0.9743 1.1633 0],Gamma,dC,dD,1)
initial(cldsysdelay,[0.0005;0;0])
```





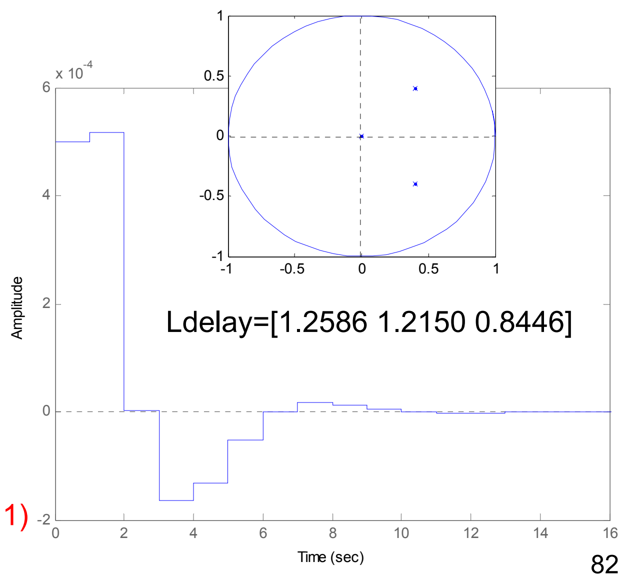

ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## Controller implementation

**Analysis : *Example3*:** the rocket controller task (with delay of 0.4s)

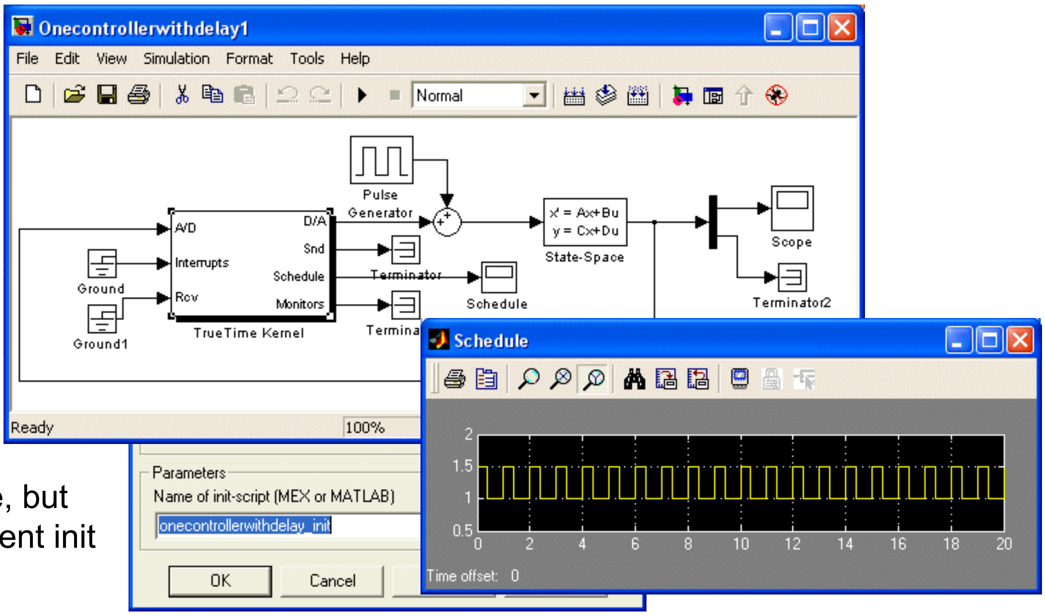
```
h=1
t=0.4
syms s;
Ad=expm(A*h)
Bd0=int(expm(A*s),s,0,h-t)*B
Bd1=exp(A*(h-t))*int(expm(A*s),s,0,t)*B
Phiorg=subs(Ad,{h,t},{1,0.4})
Gamma0=subs(Bd0,{h,t},{1,0.4})
Gamma1=subs(Bd1,{h,t},{1,0.4})
Phi=[Phiorg Gamma1;[0 0] [0]]
Gamma=[Gamma0 ;[1]]
dC=[1 0 0];
dD=[0];
Ldelay=acker(Phi,Gamma,[0.4 + 0.4i,0.4 - 0.4i,0])
cldsysdelay=ss(Phi-Gamma*Ldelay,Gamma,dC,dD,1)
initial(cldsysdelay,[0.0005;0;0])
```



# Real-time computing of control systems

## Controller implementation

**Truetime : *Example3*:** the rocket controller task (with delay of 0.4s)




Model as before, but calling a diferenent init

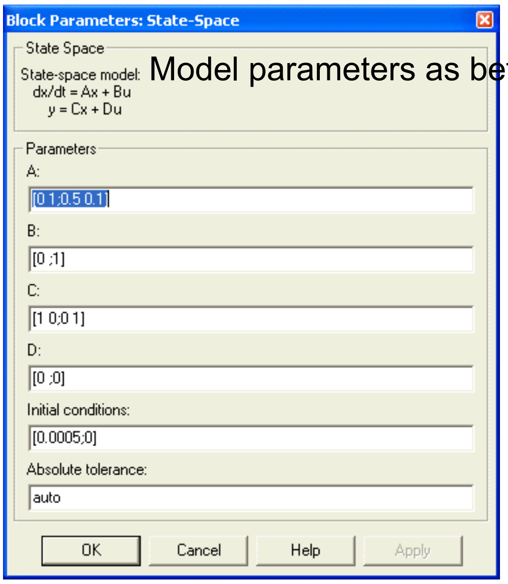
ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

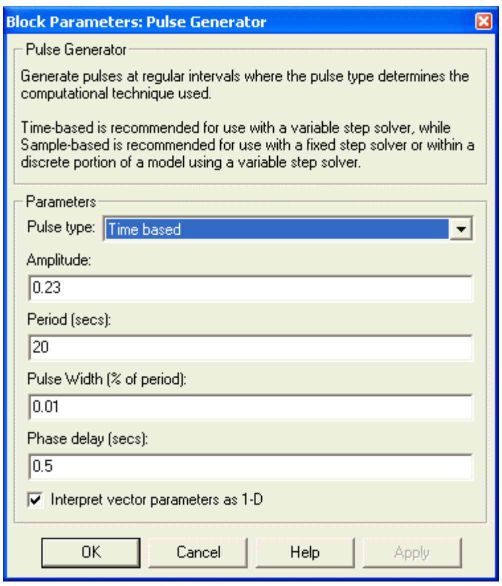
# Real-time computing of control systems

## Controller implementation

**Truetime : *Example3*:** the rocket controller task (with delay of 0.4s)

Model parameters as before



# Real-time computing of control systems

## Controller implementation

**Truetime : *Example3*:** the rocket controller task (with delay of 0.4s)

```
function onecontrollerwithdelay_init

% Rocket controller
ttInitKernel(2, 1, 'prioRM'); % nbrOfInputs, nbrOfOutputs, rate monotonic
% Task attributes
period = 1;
deadline = period;
offset = 0.0;
prio = 1;
data.L=[1.2586    1.2150    0.8446];
data.u = 0;
data.y1Chan = 1;
data.y2Chan = 2;
data.y3=0;
data.uChan = 1;
ttCreatePeriodicTask('task', offset, period, prio, 'controllercodewithdelay', data);
```



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## Controller implementation

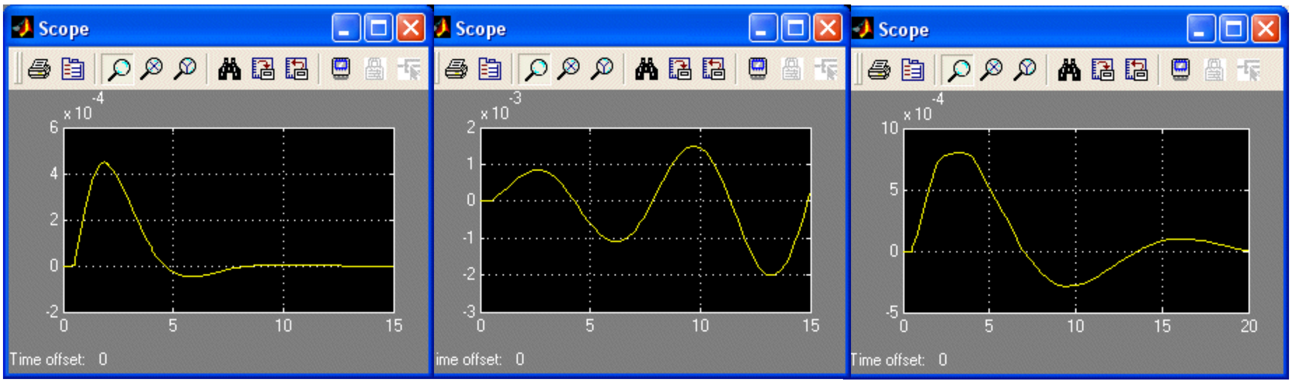
**Truetime : *Example3*:** the rocket controller task (with delay of 0.4s)

```
function [exectime, data] = controllercodewithdelay(seg, data)

switch seg,

 case 1,
  y1 = ttAnalogIn(data.y1Chan) % Read process output
  y2 = ttAnalogIn(data.y2Chan) % Read process output
  data.u=-data.L*[y1;y2;data.y3]
  data.y3=data.u
  exectime = 0.4
 case 2,
  ttAnalogOut(data.uChan, data.u); % Output control signal
  exectime = -1;
end
```

# Real-time computing of control systems

## Controller implementation

**Truetime : *Example3*:** the rocket controller task (with delay of 0.4s)




Without delay

**With delay**

Without accounting for the delay

**With delay**

Accounting for the delay

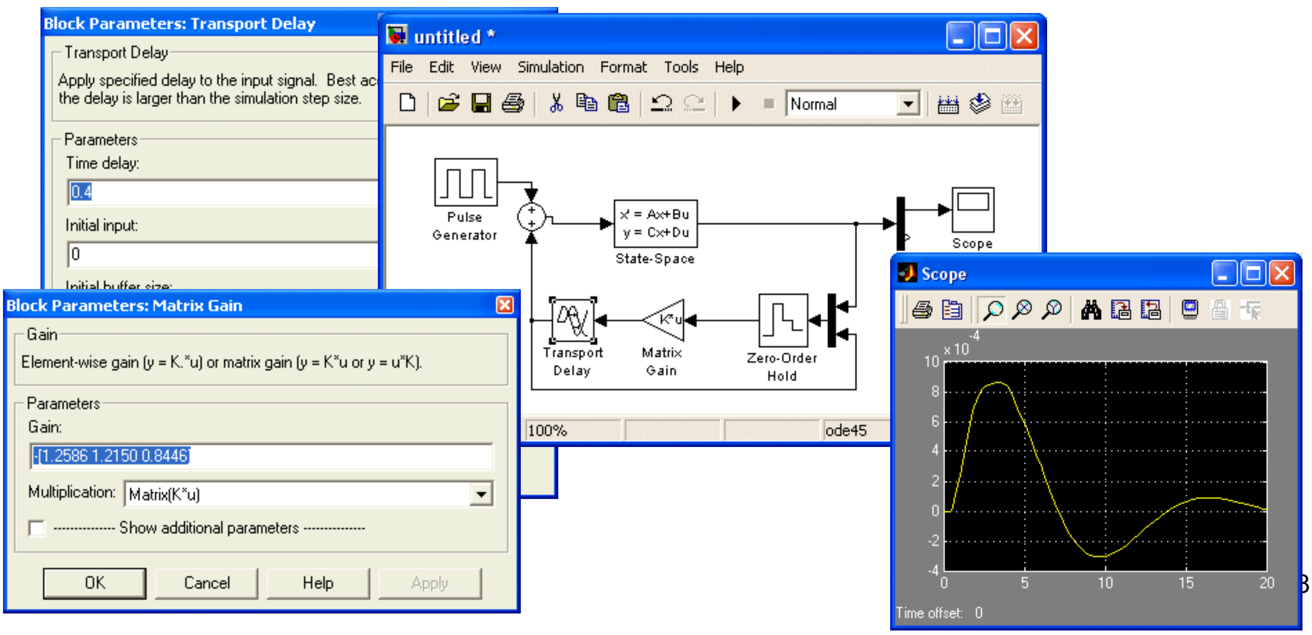




# Real-time computing of control systems

## Controller implementation

**Truetime : *Example3*:** the rocket controller task (with delay of 0.4s) with simulink



# Real-time computing of control systems

## Controller implementation

**Summary**

- care must be taken with the effects of the controller execution time (delay)
- task deadline should be equal to task worst case execution time (to bound delays)

Let's look at the sampling period (= task period)





# Real-time computing of control systems

## Selection of the control task period

**Closed loop system poles and sampling period (i.e., control task period) determine performance**

Correct affirmations:

- Poles determine system response
- Poles determine the choice of the sampling period
- Sampling period determines poles
- Sampling period determines system response

From where do we start? Poles

# Real-time computing of control systems

## Selection of the control task period

Where do we place closed-loop system poles?

Classical design rules:

1.- Reduce the expected behavior of high order systems to 2nd order systems by placing two "slow" poles (in continuous time domain). Place the remaining (continuous time) poles in such way that the output behaves as expected (like the two slow poles would determine)

2.- Choose a sampling period according to the continuous time pole position

3.- Compute the discrete feedback law

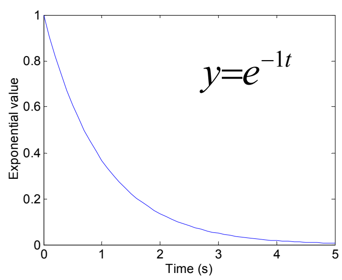


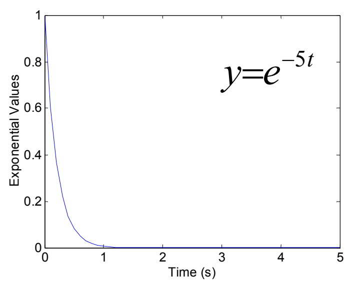


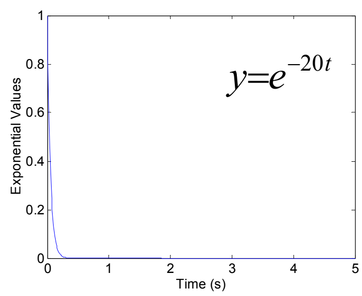
# Real-time computing of control systems

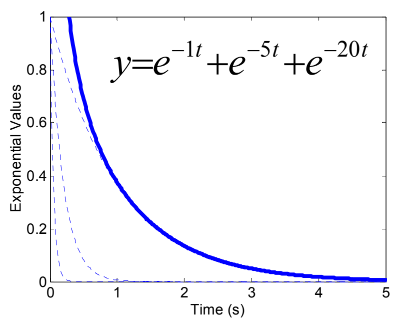
## Selection of the control task period

**Reduce the expected behavior of high order systems to 2nd order systems by placing two "slow" poles (in continuous time domain). Why?**



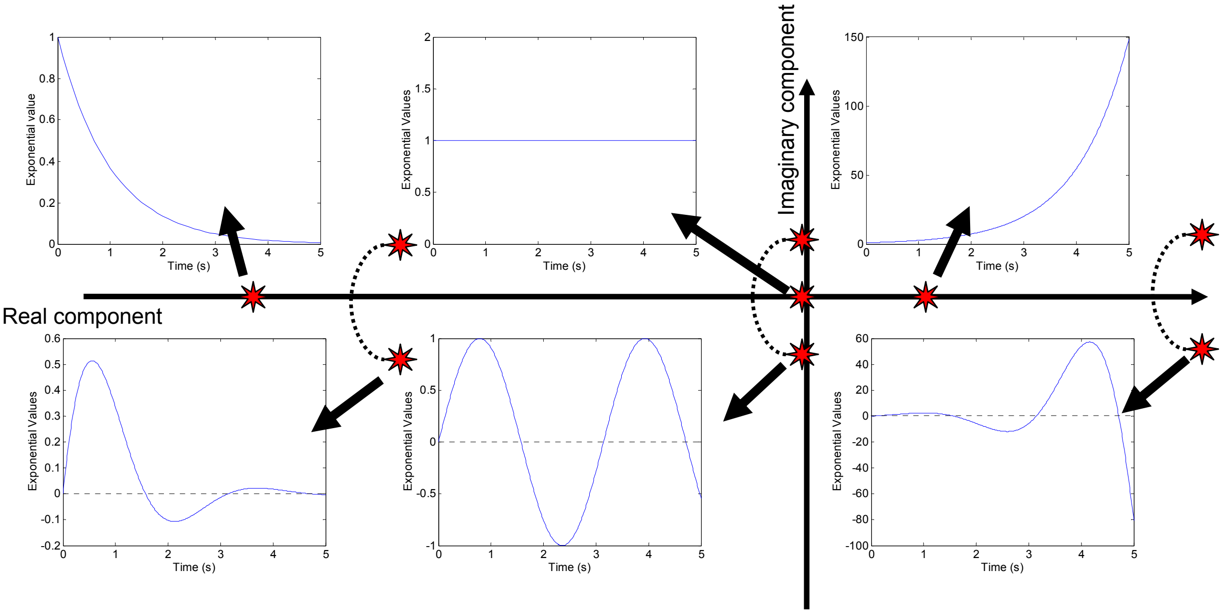

• Fast poles (the smallest) don't affect too much. Any high order system behavior is seen as a second order system behavior

• Slow poles (also called dominant) determine the output

# Real-time computing of control systems

## Selection of the control task period

**Reduce the expected behavior of high order systems to 2nd order systems by placing two "slow" poles (in continuous time domain). Why?**








# Real-time computing of control systems

## Selection of the control task period

**Choose a sampling period (if possible) according to the continuous time pole position**

- There are several rules-of-the-thumb for selecting the sampling period
- …and extended practice tells that the sampling period should accomplish that the number of samples (N) should be at least 10 to 20

$$N = \frac{2\pi}{\omega_n h \sqrt{1-\zeta^2}}$$

where $\omega_n$ is the natural (oscillation) frequency and $\zeta$ is the damping coefficient of the desired closed-loop respose (given by the two slow poles)

The relation betwen these parameters, $\omega_n$ and $\zeta$, and continuos time poles $s_{1,2}$ is:

$$(\lambda - s_1)(\lambda - s_2) = 0$$
$$\lambda^2 + 2\zeta\omega_n\lambda + \omega_n^{\,2} = 0$$

# Real-time computing of control systems

## Selection of the control task period

**Choose a sampling period (if possible) according to the continuous time pole position**

Also fast poles determine sampling period

An extended rule-of-the-thumb for fast poles is that the sampling period should be 10 to 20 times smaller than the period at the natural frequency

$$T_n = \frac{2\pi}{w_n} \quad \Longrightarrow \quad h < \frac{T_n}{10}$$





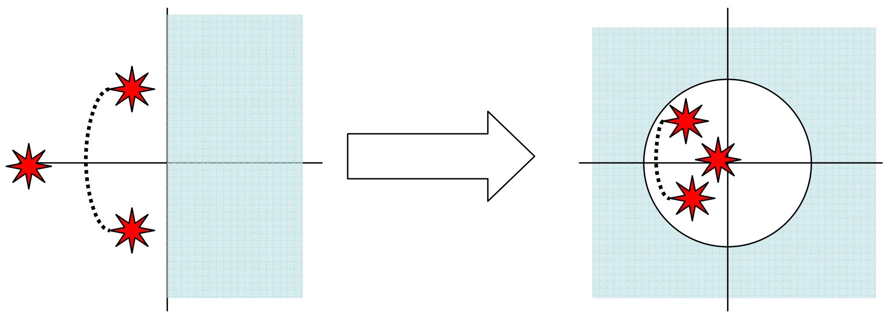
# Real-time computing of control systems

## Selection of the control task period

**Compute the discrete feedback law**

Use the technique shown before:

- Convert the selected continuous poles to discrete ones



- Compute the gains (using ackerman's formula)

# Real-time computing of control systems

## Selection of the control task period

### Servo positioning

**System model**

$$\dot{x} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{b}{J} & \frac{k}{J} \\ 0 & -\frac{k}{L} & \frac{R}{L} \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x$$

**Pole positions**

$p1 = -100 + 300i$

$p2 = -100 - 300i$

$p3 = -500$

**Discrete model**

$$x_{k+1} = \begin{bmatrix} 1 & 0.0075 & 0 \\ 0 & 0.55 & 0.0032 \\ 0 & 0.0038 & 0 \end{bmatrix} x_k + \begin{bmatrix} 0.087 \\ 16.01 \\ 0.14 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x$$

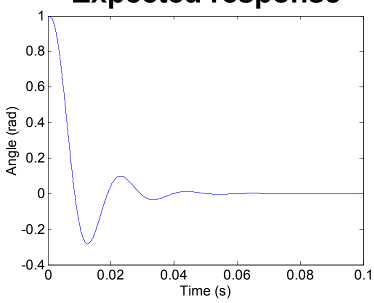
**Discrete Poles**

z1=0.9896 + 0.0297i

z2=0.9896 - 0.0297i

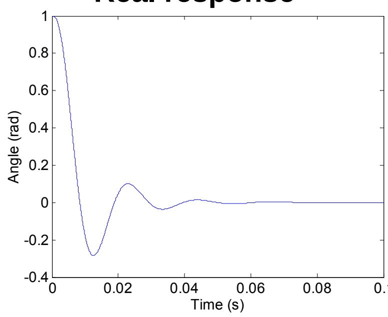
z3=0.8187

**Expected response**




$$h = 0.0001s$$

**Real response**






ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## Selection of the control task period

**Period selection:**

**Pole positions**

$z1 = -100 + 300i$

$z2 = -100 - 300i$

$z3 = -500$

**Characteristic equation**

$$(\lambda + 100 + 300i)(\lambda + 100 - 300i) = 0$$

$$\lambda^2 + 200\lambda + 100000 = 0$$

$$\lambda^2 + 2\xi\omega_n\lambda + \omega_n{}^2 = 0$$

$$\omega_n = 100\sqrt{10}$$

$$\xi = \frac{1}{\sqrt{10}}$$

**Discrete model**

*Slow poles*

*Assume* $N = 10$

$$h = \frac{2\pi}{\omega_n N\sqrt{1-\xi^2}}$$

$$h = \frac{2\pi}{100 * \sqrt{10} * 10\sqrt{1 - \frac{1}{10}}}$$

$$h = 0.0021$$

*fast poles*

*Assume* $N = 10$

$$h = \frac{2\pi}{\omega_n N}$$

$$h = \frac{2\pi}{100 * \sqrt{10} * 10}$$

$$h = 0.002$$

$$h = 0.0001s$$

# Real-time computing of control systems

## Selection of the control task period

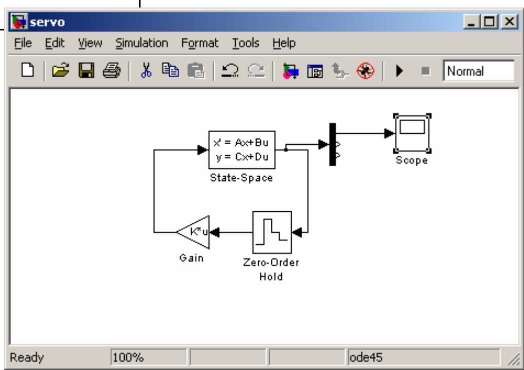
**Matlab Code:**

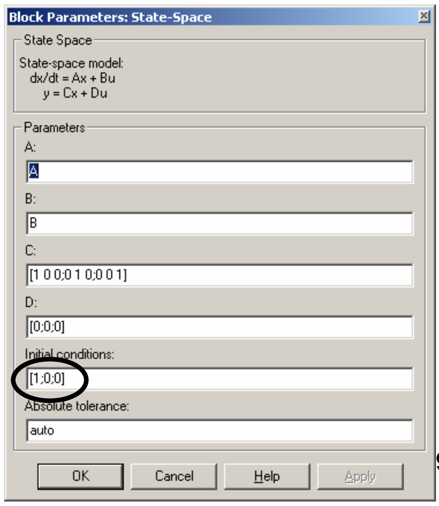
**System definition:**

```
J=3.2284E-6;
b=3.5077E-6;
K=0.0274;
R=4;
L=2.75E-6;
A=[0 1 0; 0 -b/J K/J; 0 -K/L -R/L];
B=[0 ; 0 ; 1/L];
C=[1 0 0;0 1 0;0 0 1];
D=[0;0;0];
```

**Controller design:**

```
Poles=[-1+3*i -1-3*i -20];
h=0.01;
[Ad,Bd]=C2D(A,B,h);
Dpoles=exp(Poles*h);
K=acker(Ad,Bd,Dpoles);
```






ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## Trading off performance vs. schedulability

The sampling period usually has restrictions, e.g, in EDF, ∑ci/Ti<1

The processor in charge of the rocket angle control, may be also in charge of:

- Trajectory
- Nozzle aperture
- Engine temperature
- Power
- Gyroscopes
- …

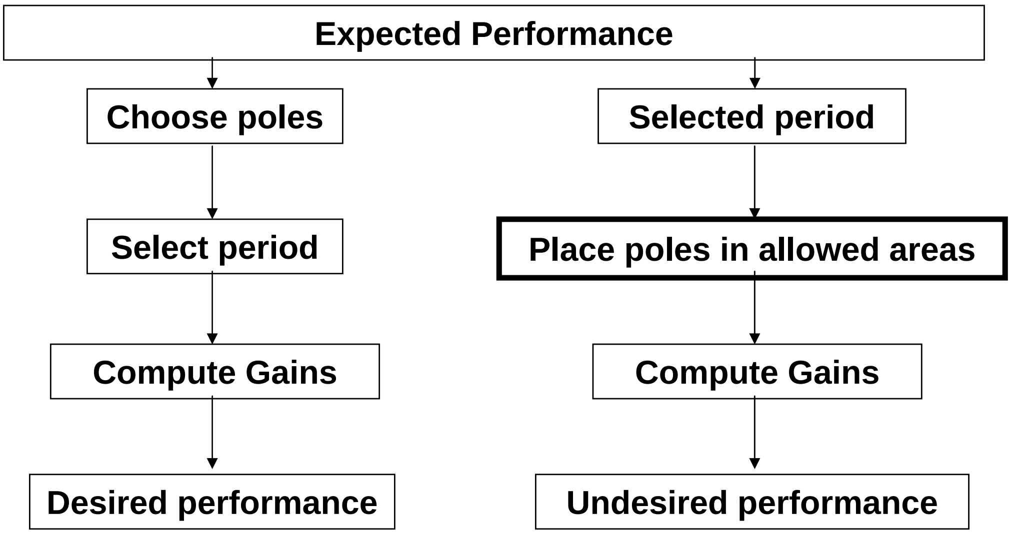
There are many tasks to be executed, and limited resources (processors, network…)

This problem becomes more critical for mass-marked products ( i.e. mobile phones, automotive components, etc…) subject to hard economic constraints

Usually there is a minimum sampling period

# Real-time computing of control systems

## Trading off performance vs. schedulability






ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## Trading off performance vs. schedulability

**Place poles in allowed areas**

With a given h, we can only choose or frequency or damping coefficient: It is not possible to completely specify the closed loop dynamics. $\Rightarrow$

$$N = \frac{2\pi}{w_n h \sqrt{1-\zeta^2}}$$

Also fast poles have limitations….
A rule-of-the-thumb for fast poles is $\Rightarrow$

$$N = \frac{2\pi}{w_n h}$$

# Real-time computing of control systems

## Trading off performance vs. schedulability

**Servo positioning when minimum sampling is fixed:**

**System model**

$$\dot{x} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{b}{J} & \frac{k}{J} \\ 0 & -\frac{k}{L} & \frac{R}{L} \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x$$

**Pole positions**

$$p1 = -100 + 300i$$
$$p2 = -100 - 300i$$
$$p3 = -500$$

**Discrete model**

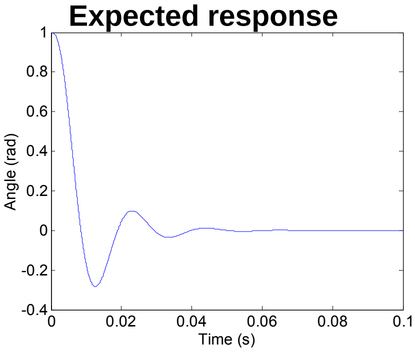
$$x_{k+1} = \begin{bmatrix} 1 & 0.0075 & 0 \\ 0 & 0.55 & 0.0032 \\ 0 & 0.0038 & 0 \end{bmatrix} x_k + \begin{bmatrix} 0.087 \\ 16.01 \\ 0.14 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x$$

**Discrete Poles**

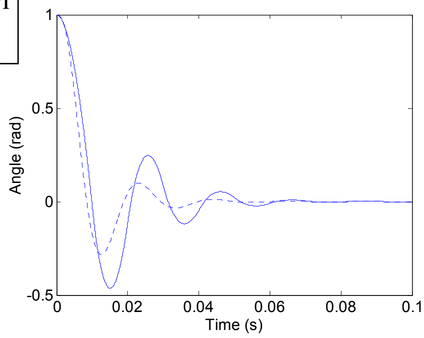
$$z1 = -0.3642 + 0.0519i$$
$$z2 = -0.3642 - 0.0519i$$
$$z3 = 0.0067$$

**Expected response**




$$h = 0.01s$$

**Real response**






ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

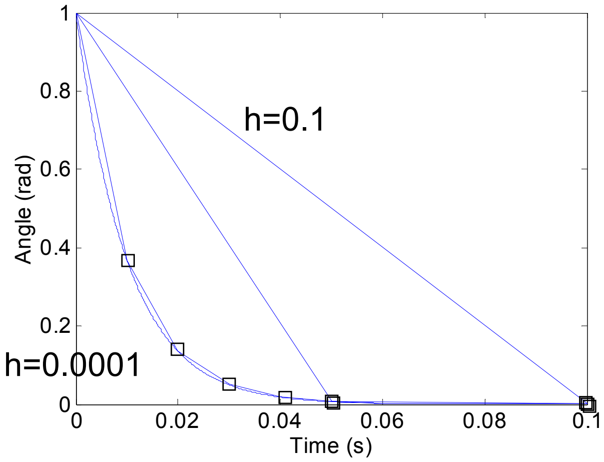
## Trading off performance vs. schedulability

**Degradation due to sampling period selection:**

For a first order system, the solution is exact at the sampling points, but there is a degradation between them.

$$\dot{x} = x + u$$
$$p = -100$$
$$h_1 = 0.0001$$
$$h_2 = 0.01$$
$$h_3 = 0.05$$
$$h_4 = 0.1$$

**Z1=0.99   z2=0.9048   z3=0.0067   z4=0**

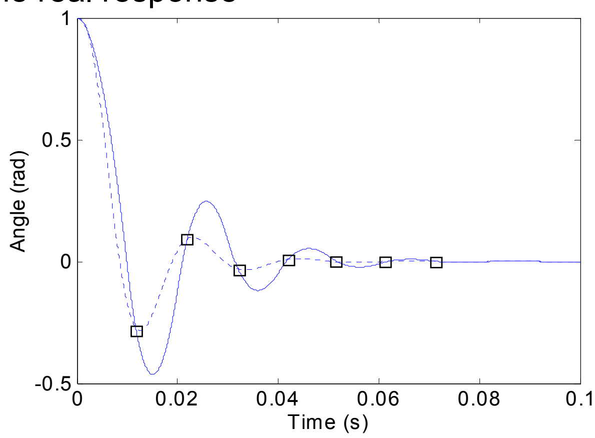


# Real-time computing of control systems

## Trading off performance vs. schedulability

**Degradation due to sampling period selection:**

For higher order systems, the degradation also implies exact solutions at the sampling points. The system behaves as expected only at the sampling points, and thus we obtain worst responses.
The lower the density of sampling points, the higher the difference between the desired and the real response







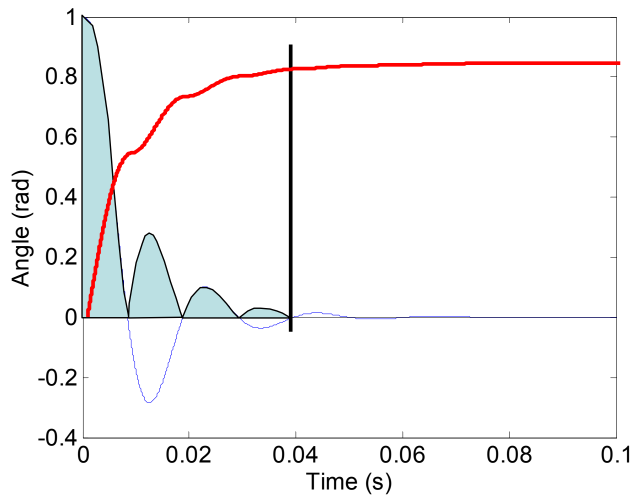

# Real-time computing of control systems

## Trading off performance vs. schedulability

To evaluate the degradation we use indexes that reflect how much time the system needs to converge towards the setpoint and how far it is from the set point during this time. A commonly used index is the IAE (Integral of the Absolute value of the Error)

$$IAE = \int_0^\infty |y(t)| dt$$

$$IAE = \int_0^\infty |Y_d(t) - y(t)| dt$$



# Real-time computing of control systems

## The jitter problem

General purpose scheduling policies ⟶ Inherent jitter in task instance execution

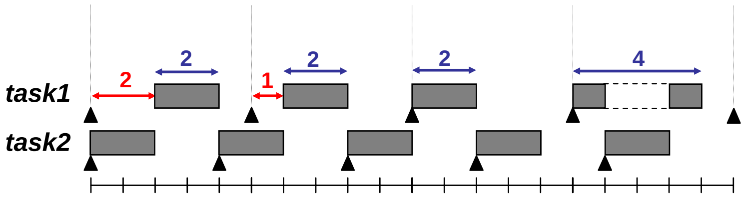
**Real-time interpretation:**

- varying start time delays
- varying response times

| | T | C |
|---|---|---|
| *task1* | 5 | 2 |
| *task2* | 4 | 2 |

**(D = T)**

**Rate Monotonic Scheduling**






ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## The jitter problem

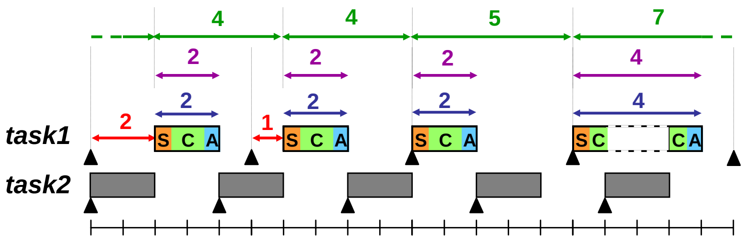
**Control interpretation:**

- ***Sampling jitter:*** different values for the sampling period
- ***Sampling-actuation jitter*:** different values for the time delay

| | T | C |
|---|---|---|
| *Control task1* | 5 | 2 |
| *task2* | 4 | 2 |

**(D = T)**

**Rate Monotonic Scheduling**




| | | | | |
|---|---|---|---|---|
| **Sampling period values** | 4 | 4 | 5 | 7 |
| **Time delay values** | 2 | 2 | 2 | 4 |

# Real-time computing of control systems

## The jitter problem

**Design stage**

Computer-control timing assumptions

- **Constant** *sampling period*
- **Constant** *time delay*

**System run time**

Scheduling inherent jitters

- **Varying** *sampling periods*
- **Varying** *time delays*

Run time **violation** of the closed-loop timing assumptions

**Problem**

Standard scheduling schemes

Variability in task instance execution

Closed-loop performance degradation



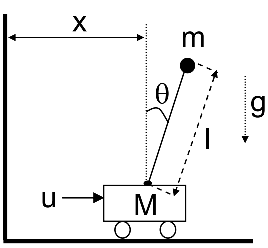
ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## The jitter problem

**Example**

**System**

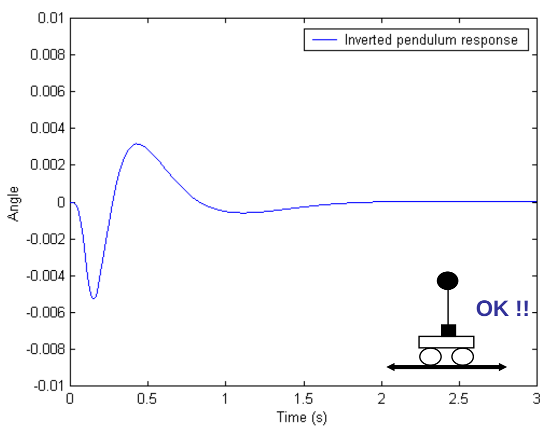



Inverted pendulum

**Controller**

- Design: classical pole placement (to recover from a perturbation in less than 2s)
- Implementation: periodic task

**Response** (task executed in **isolation**)



# Real-time computing of control systems

## The jitter problem

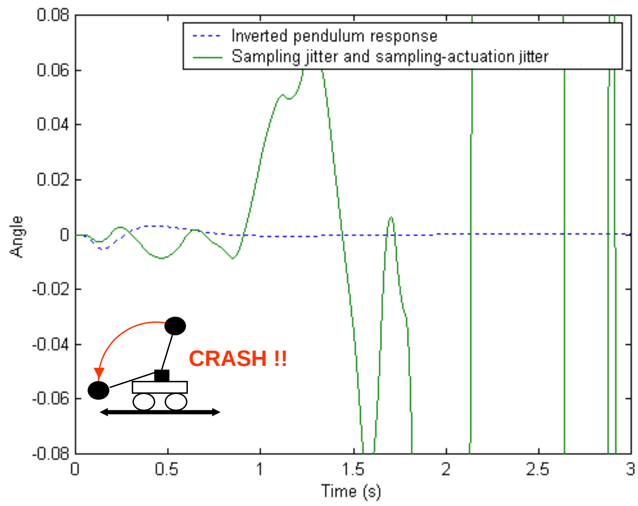
**Example**

**Context: limited resources**

Task set

| | T | C |
|---|---|---|
| *task1* | 50ms | 10ms |
| *task2* | 60ms | 10ms |
| *controller* | 80ms | 20ms |
| *task3* | 100ms | 20ms |

+ arbitrary offline scheduling scheme




**Task set feasible schedule** → **Controlled system unstability**



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## The jitter problem

**Solution approach**

Offline schedule analysis + Offline control analysis + scheduling with compensation approach

↓

Analysis of scheduling inherent jitter for control tasks

**Compensation approach:**
**accept jitters** in the control design

- **problem formulation**
- **stability analysis**
- **computational overhead**
- **memory requirements**

**Improved**
- control performance and
- system schedulability

↓

**Flexible timing constraints** for control task scheduling

↓

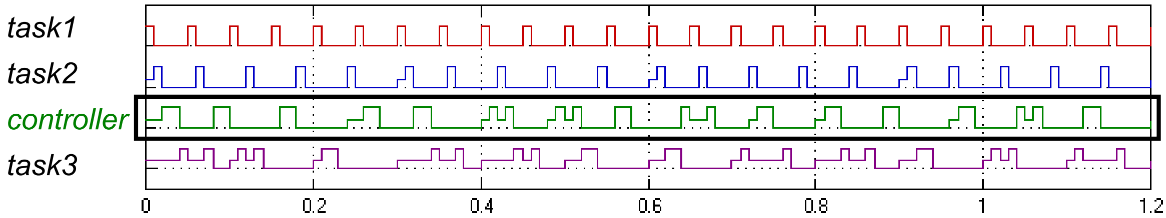
**New schedulability analysis**

Run-time controller **parameters adjustment**

# Real-time computing of control systems

## The jitter problem

### Offline schedule analysis




### Jitter variability

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Sampling intervals | 60 | 80 | 100 | 60 | 90 | 80 | 70 | 80 | 90 | 80 | 70 | 90 | 70 | 80 | 100 |
| Sampling-actuation delays | 20 | 20 | 20 | 20 | 20 | 30 | 30 | 20 | 40 | 20 | 20 | 20 | 20 | 30 | 20 |

$h=\{60,70,80,90,100\}$ (ms)

$\tau=\{20,30,40\}$ (ms)



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## The jitter problem

### Compensation idea:

**Classical scenario**

Controller: -L($\mathbf{h}$,$\tau$)

**h,** $\tau$ are **constant** values

***Regularly*** **sampled discrete-time systems with *constant* time delays**

**Run time controller parameters adjustment**

**New scenario**

Controller: -L($\mathbf{h}_j$,$\tau_j$)

$\mathbf{h}_j$,$\tau_j$ are **varying** values

***Irregularly*** **sampled discrete-time systems with *varying* time delays**

- ***set*** *of sampling intervals,* $\mathbf{h}_j$
- ***set*** *of sampling-actuation delays,* $\tau_j \in$

# Real-time computing of control systems

## The jitter problem

### Offline control analysis

h={60,70,80,90,100} (ms)
τ={20,30,40} (ms)

- Classically designed controller, **-L(h,τ)**
- Stability analysis **-L(h,τ)**

### Practical aspects

- Computational overhead (for the run time recalculation strategy)

| | Traditional design | Compensation approach design |
|---|---|---|
| **Number of Flops** | **60** | **2000** |

- Memory requirements (for the run time table look-up strategy)

**10Kb** approx.





# Real-time computing of control systems

## The jitter problem

### Offline control analysis

**_Regularly_ sampled discrete-time systems with _constant_ time delays**

$$x(kh+h)=\Phi(h)x(kh)+\Gamma_0(h,\tau)u(kh)+\Gamma_1(h,\tau)u(kh-h)$$
$$y(kh)=Cx(kh)+Du(kh)$$
$$u(kh)=\boxed{-L(h,\tau)}x(kh)$$

**Closed-loop matrix:**

$$\boxed{\Phi_{cl}}=\begin{bmatrix}\Phi(h)-\Gamma_0(h,\tau)\cdot L(h,\tau) & \Gamma_1(h,\tau)\\ -L(h,\tau) & 0\end{bmatrix}$$

**System evolution** (k^th iteration):

$$x(kh+h)=\Phi_{cl}x(kh)=\Phi_{cl}\Phi_{cl}^{\;k-1}x(kh-h)=\boxed{\Phi_{cl}^{\;k}}x(0)$$

**_Irregularly_ sampled discrete-time systems with _varying_ time delays**

$$x(t_{k+1})=\Phi(h_j)x(t_k)+\Gamma_0(h_j,\tau_j)u(t_k)+\Gamma_1(h_j,\tau_j)u(t_{k-1})$$
$$y(t_k)=Cx(t_k)+Du(t_k)$$
$$u(t_k)=\boxed{-L(h_j,\tau_j)}x(t_k)$$

**Closed-loop matrix:**

$$\boxed{\Phi_{cl\,k}}=\begin{bmatrix}\Phi(h_j)-\Gamma_0(h_j,\tau_j)\cdot L(h_j,\tau_j) & \Gamma_1(h_j,\tau_j)\\ -L(h_j,\tau_j) & 0\end{bmatrix}$$

**System evolution** (k^th iteration):

$$x(t_{k+1})=\Phi_{cl\,k}x(t_k)=\boxed{\Phi_{cl\,k}\Phi_{cl\,k-1}\Phi_{cl\,k-2}\dots\Phi_{cl\,1}}x(0)$$

# Real-time computing of control systems

## The jitter problem

### Offline control analysis

$$x(t_{k+1}) = \Phi_{cl\,k} x(t_k) = \Phi_{cl\,k} \Phi_{cl\,k-1} \Phi_{cl\,k-2} \ldots \Phi_{cl\,1} x(0)$$

**Stability analysis** (depends on the implementation)

- one closed-loop matrix

stable $\Leftrightarrow$ $\rho(\Phi_{cl}) < 1$

- an infinite product of closed-loop matrices that follow a periodic pattern

stable $\Leftrightarrow$ $\rho(\Phi_{cl_1} \cdot \Phi_{cl_2} \cdot \ldots \cdot \Phi_{cl_n}) < 1$

- an infinite product of closed-loop matrices taken randomly from a finite set, $\Omega$

Necessary and sufficient stability condition: $\Omega$ stable $\Leftrightarrow \exists$ P>0: $\Phi^T_{cl_k} \cdot P \cdot \Phi_{cl_k} - P < 0$, $\forall \Phi_{cl_k} \in \Omega^k$, $k \geq 1$

Sufficient stability condition (I) : if $\exists$ P>0 : $\forall\ \Phi_{cl_k} \in \Omega$ , $\Phi^T_{cl_k} \cdot P \cdot \Phi_{cl_k} - P < 0 \Rightarrow$ stable

Sufficient stability condition (II): if $\forall\ \Phi_{cl_k} \in \Omega$ : $\Phi^T_{cl_k} \cdot \Phi_{cl_k} - I < 0 \Rightarrow$ stable





# Real-time computing of control systems

## The jitter problem

### Compensation approach basic idea

Controller **parameters adjustment**
at each control task instance execution
**according** to both actual
- **sampling interval** and
- **samplng-acutation delay**

$\Bigg\}$ **-L($h_k$,$\tau_k$)**

**Key point: online parameters adjustment**

- Through online parameters recalculation (if negligible overhead)
- Through online access to controller parameters look-up table (if not negligible overhead)

# Real-time computing of control systems

## The jitter problem

### Computational overhead analysis

Depends on { the system to be controlled
the controller design method

### Examples

- Pole placement observer state feedback controller design for the inverted pendulum: **$O(n^4)$**

| | Number of Flops (without extra calculations) | Number of Flops (with extra calculations) |
|---|---|---|
| simplified pendulum (*2x2 matrix*) | 25 | 250 |
| complete pendulum (*4x4 matrix*) | 60 | 2000 |

- PID controller design for a DC servo: **insignificant**



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## The jitter problem

### Memory requirements analysis

Size(table)= $(h_{max} - h_{min})*h_{max}$*clockticksize * size(controller parameters)

Sampling interval variability: $h_{min} \leq h_k \leq h_{max}$

Sampling-actuation delays variability: $0 \leq \tau_k \leq h_{max}$

Clockticksize (granularity)

**Inverted pendulum example:**

| $h_k$ (ms) | $\tau_k$ (ms) | $-L(h_k, \tau_k)$ |
|---|---|---|
| 60 | 20 | -L(60, 20) |
| 70 | 20 | -L(70, 20) |
| | 30 | -L(70, 30) |
| 80 | 20 | -L(80, 20) |
| | 30 | -L(80, 30) |
| 90 | 20 | -L(90, 20) |
| | 40 | -L(90, 40) |
| 100 | 20 | -L(100, 20) |

# Real-time computing of control systems

## The jitter problem

**Key point: run-time controller parameters adjustment**

**Implementation issues:** Slightly modification in the controller code

***Generic controller***

{ *read_inputs* $(y(t_k), r(t_k))$;
  $e(t_k) = r(t_k) - y(t_k)$;
  $u(t_k)$ = *calculate_output* $(L(h,\tau), e(t_k))$;
  *write_output* $(u(t_k))$;
}

***Compensation approach controller***

{ *read_inputs* $(y(t_k), r(t_k))$;
  ***obtain*** $(h_j, \tau_j)$;
  $e(t_k) = r(t_k) - y(t_k)$;
  ***obtain*** $(L(h_j, \tau_j))$;
  $u(t_k)$ = *calculate_output* $(L(h_j, \tau_j), e(t_k))$;
  *write_output* $(u(t_k))$;
}



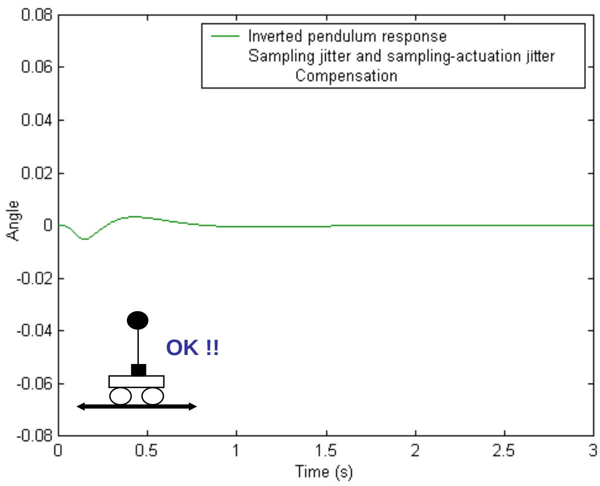


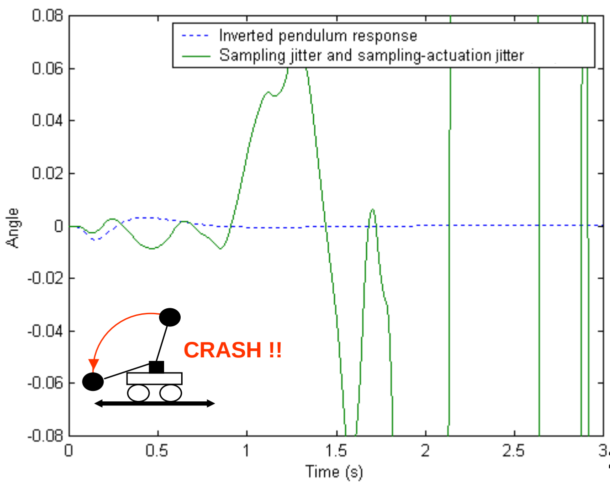
# Real-time computing of control systems

## The jitter problem

**Results**

**Elimination** of the closed-loop system performance degradation (caused by standard scheduling policies inherent jitters)



# Real-time computing of control systems

## Flexible timing constraints

### Problem formulation

- Standard scheduling schemes are based on **fixed timing constraints** (T, D)
- ***Fixed*** means that a **single value** for a constraint holds for **all instances of a task**

↓ **Flexibility**

We do **not want to set single values** for the timing constraints,
we want provide **ranges** of values to choose from

### Compensation approach timing assumptions:

- Set of sampling intervals
- Set of sampling-actuation delays

→ **Idea: to derive new timing constraints for control tasks**



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

## Flexible timing constraints

Flexible timing constraints for control tasks designed with compensations

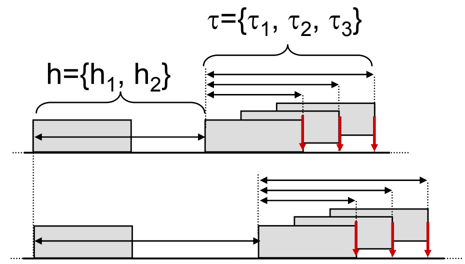
| **Compensation approach** | | **Flexible timing constraints** |
|---|---|---|
| Set of sampling periods: $h_j$ | $\Rightarrow$ | Instance separations: $\forall$ $task_k$, $task_{k+1}$: s($task_{k+1}$) - s($task_k$) $\in$set |
| Set of delays: $\tau_j$ | | Response times: $\forall$ $task_k$: f($task_k$) - s($task_k$) $\in$set |

- **Several values** for a constraint are eligible at each **task instance execution**

**Example**

- two instances separations
- three response times

can be selected at each instance execution



# Real-time computing of control systems

## Flexible timing constraints

**Solved problem:** Poor system schedulability due to the application of fixed timing constraints for control tasks

### Idea

- characterize control tasks with flexible timing constraints, and
- exploit their flexibility to improve system schedulability



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Real-time computing of control systems

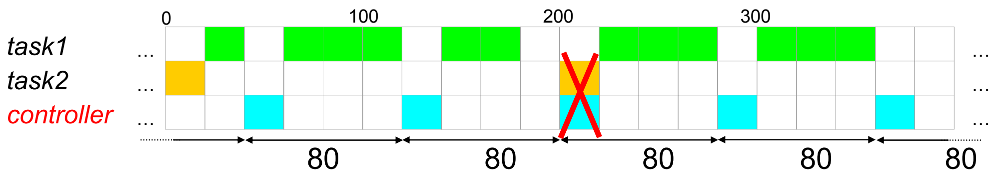
## Flexible timing constraints

**Example**

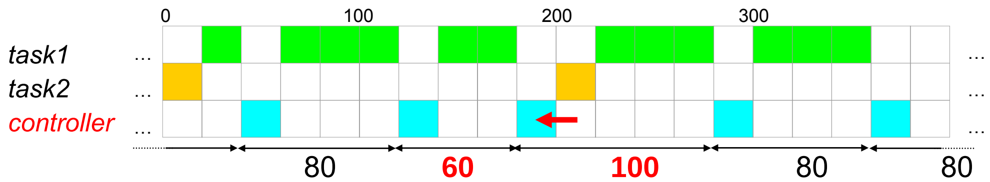
| | T | C | D | O |
|---|---|---|---|---|
| *task1* | 100ms | 60ms | | |
| *task2* | 200ms | 20ms | 20ms | |
| *controller* | h | 20ms | 20ms | 40ms |

(h=sampling period)

- Fixed timing constraints (h = 80ms)




- Flexible timing constraints (h ∈ {60ms, 80ms, 100ms})



# Real-time computing of control systems

## Flexible timing constraints

From **infeasible schedules** of task sets (that include **control tasks**) based on **fixed timing contraints** → We obtain **feasible schedules** and **stable control systems**

↑

**Flexible timing constraints**
**+**
**Compensation approach**





# Real-time computing of control systems

## Summary

- relation between control task timing constraints and control system properties
- impact of scheduling methods on control performance: closed-loop systems degradation
- impact of classic control requirements on schedulabiliy: poor system schedulability
- compensation approach as a control-based solution
- flexible timing constraints to improve task set schedulability

# Control of computing systems

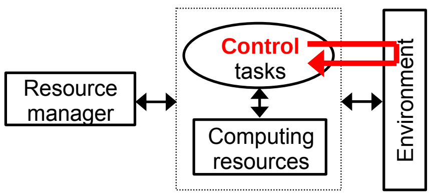
1. Introduction
2. Key aspects in “control of RT control systems”
3. Slack reclaiming: optimal state feedback
4. Overload conditions



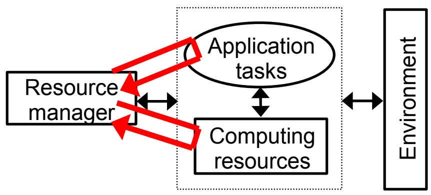


# Control of computing systems

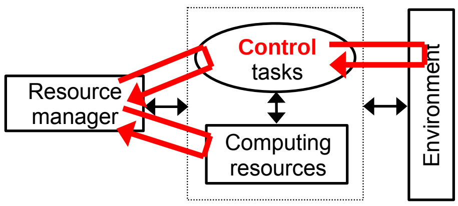
## Introduction




Natural feedback




Control of QoS
“Feedback control of computing systems”




Control of Quality of Control
“Control of Real-Time Control Systems”

# Control of computing systems

## Key aspects in "control of RT control systems"

- Controller design
- Type of control desing
- Possibility of controller adaptation
- Intuitive guidelines for co-design



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control of computing systems

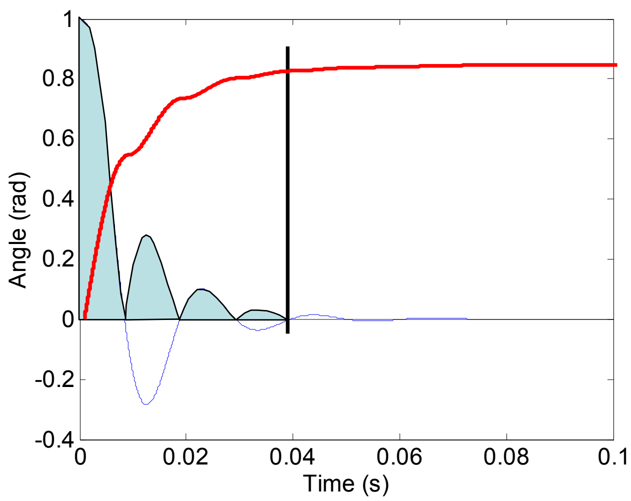
## Key aspects in "control of RT control systems"

To control (improve) a RT control system, we need a means of evaluation

**IAE (Integral of the Absolute value of the Error)**

$$IAE = \int_0^\infty |y(t)|\,dt$$

$$IAE = \int_0^\infty |Y_d(t) - y(t)|\,dt$$



# Control of computing systems

## Key aspects in “control of RT control systems”

IAE can be evaluated for a set of sampling periods
This can be either applied to:

- adapted controllers
- non adapted controllers

**For adapted controllers**

For(h=0 ; h<limit ;h=h+increment)
{
    Compute $\Phi$ and $\Gamma$ for h
    Compute discrete poles for h
    Compute gains for h
    Compute IAE
}

**For non adapted controllers**

Precompute nominal gains for $h_i$
For(h=0 ; h<limit ;h=h+increment)
{
    Compute Ad,Bd for h
    Compute discrete poles for h
    ~~Compute gains for h~~
    Compute IAE
}

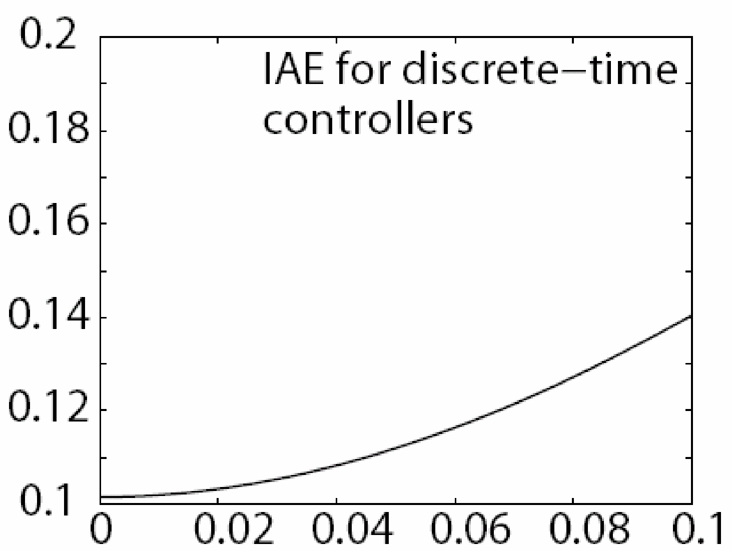


ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control of computing systems

## Key aspects in “control of RT control systems”

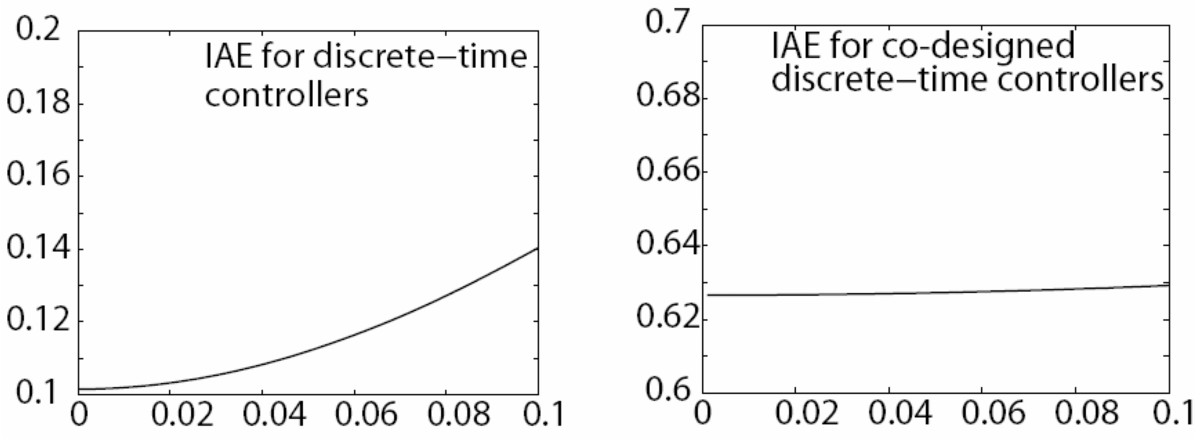
**IAE degradation diagram for adapted controllers, general aspect:**



# Control of computing systems

## Key aspects in “control of RT control systems”

**IAE degradation diagram for adapted controllers, general aspect:**




(a) Approach 1

(b) Approach 2

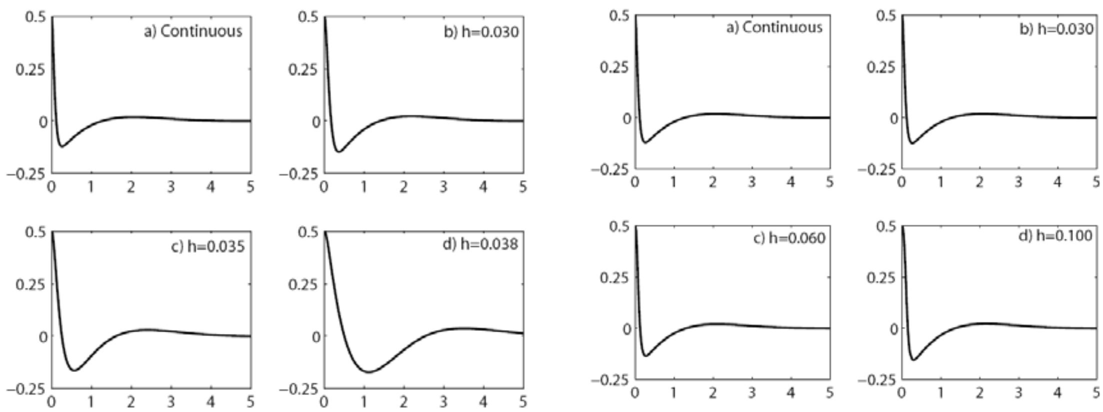


ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control of computing systems

## Key aspects in “control of RT control systems”

- Is the degradation caused by platform constraints always the same regardless of the control design methodology? NO

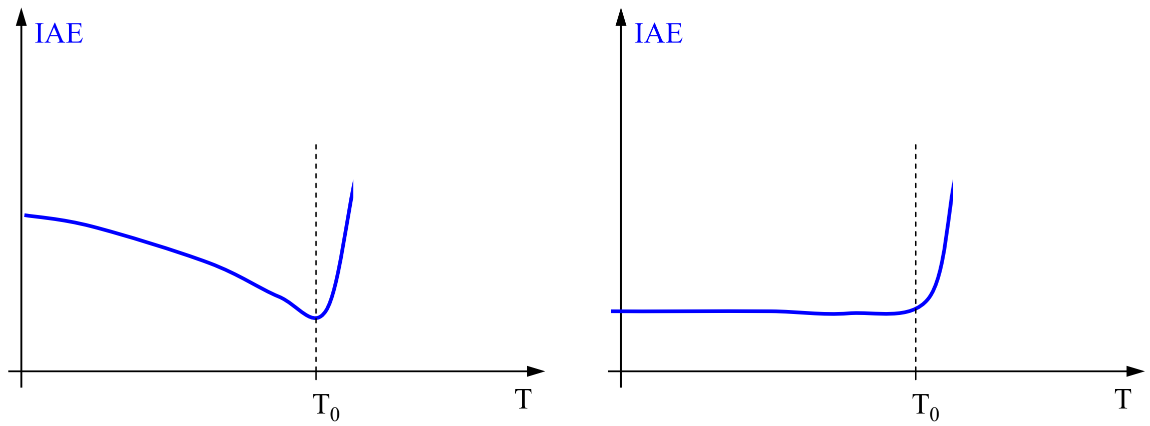



Discrete appox. of cont. design (dacd) Direct discrete design (ddd)

# Control of computing systems

## Key aspects in “control of RT control systems”

**IAE degradation diagrams (approximation) for non-adapted controllers**






ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control of computing systems

## Key aspects in “control of RT control systems”

**Guidelines for co-design**

- They give an idea of how near to a correct design we are. If the curve (for non adapted controllers) goes up quickly, then the system is near to the limits (i.e., it degrades quickly) and, probably, jitters effects will be more noticeable. If curve is flat (almost flat) then we have some guaranties that the response will be closer to the one we expect. Useful for designing control of QoC

- They are useful for designing a set of controllers for overload conditions. A flat curve (for adapted controllers) indicates that dynamics will be invariant under processor overloads. We can allow the kernel to change sampling periods at execution time and then select the correct controller knowing that the response will not be affected

# Control of computing systems

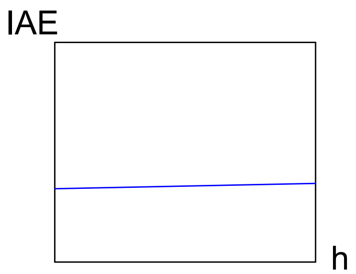
## Key aspects in "control of RT control systems"

### Guidelines for co-design

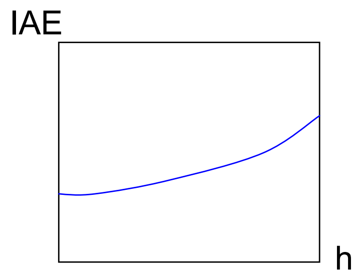
… useful for slack reclaiming approaches for improving control in RTCS

It is well accepted than running a controller faster means to obtain a better response

Not always !!!




Not really




Probably



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

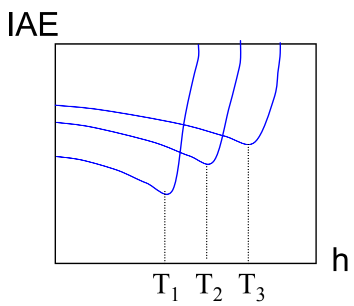
# Control of computing systems

## Key aspects in "control of RT control systems"

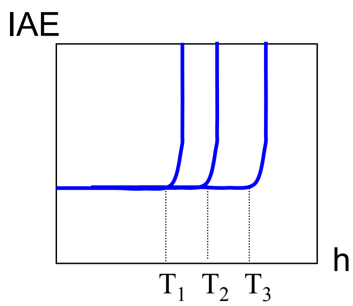
### Guidelines for co-design

… useful in overload conditions for controlling QoC

Intuitively



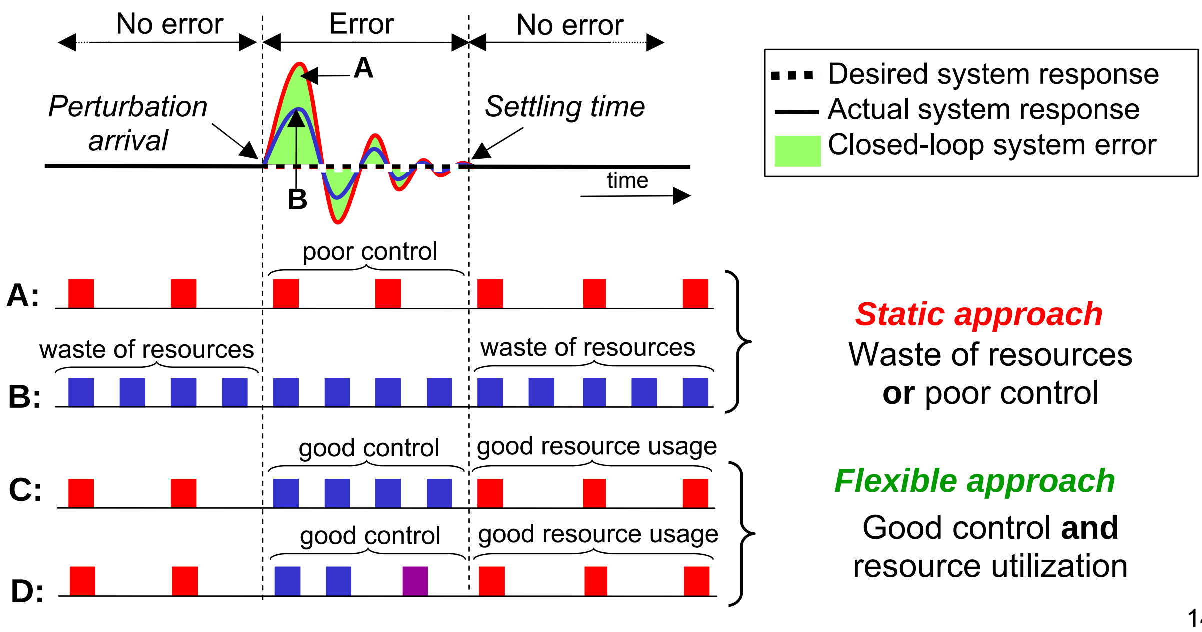

It makes sense to switch controllers




It does not

# Control of computing systems

## Slack reclaiming: optimal state feedback




***Static approach***
Waste of resources
**or** poor control

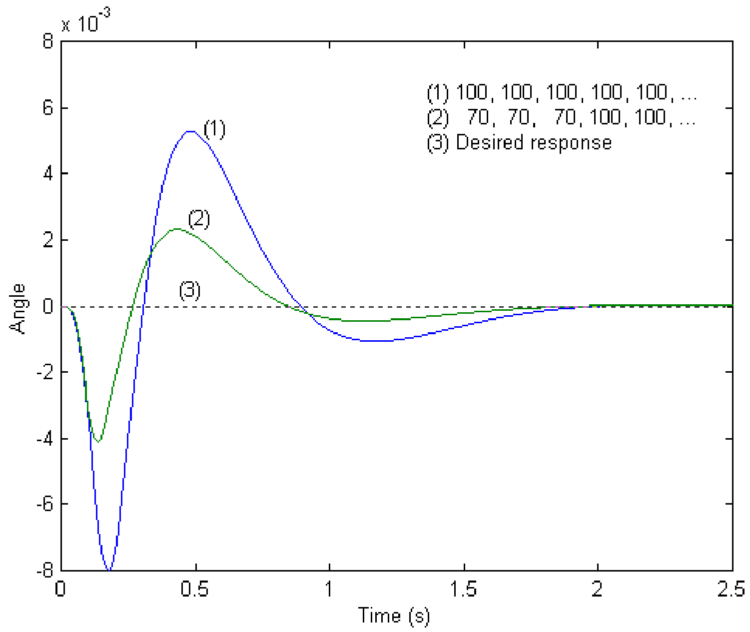
***Flexible approach***
Good control **and** resource utilization





# Control of computing systems

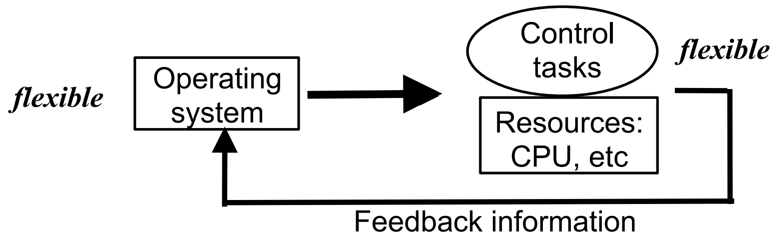
## Slack reclaiming: optimal state feedback




Example on an inverted pendulum

# Control of computing systems

## Slack reclaiming: optimal state feedback




- Control System:
  - Flexible controller: capable of running with different frequencies, i.e. h in [$h_{min}$…$h_{max}$]
  - Provide useful feedback information to OS
- Operating System:
  - Real-Time: guarantee time constraints
  - Dynamic and flexible resource allocation (e.g., RBED, Elastic)



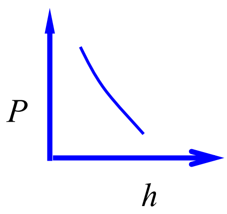
ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control of computing systems

## Slack reclaiming: optimal state feedback

### Static and dynamic feedback

- **Static**: performance (*P*) vs period (*h*):




$$\Rightarrow \quad P(h) = \frac{\beta}{h} \qquad \oplus \qquad r = \frac{c}{h}$$

$$\Downarrow$$

$$P(r) = \frac{\beta}{c} r = \alpha \cdot r$$

- **Dynamic**: States of controlled system $x_i(t)$

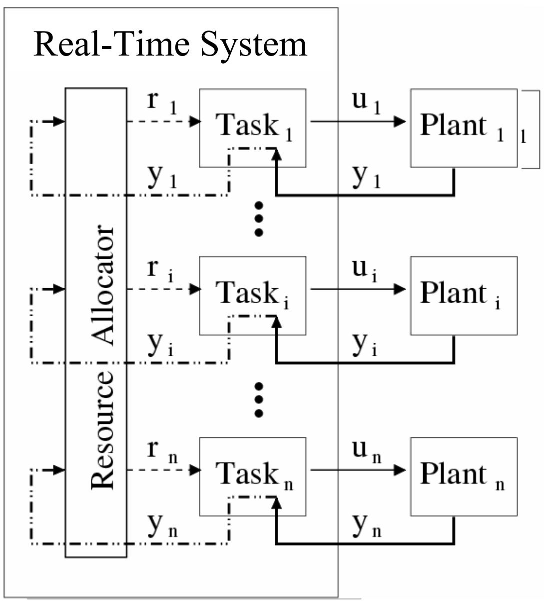
$$E(t) = |x(t)| = \sqrt{x_1^2(t) + \cdots + x_n^2(t)}$$

$$\alpha \cdot E$$ **State Feedback**

# Control of computing systems

## Slack reclaiming: optimal state feedback

### Resource allocation problem




$y_1 = \alpha_1 E_1$

$y_i = \alpha_i \cdot E_i$

$y_n = \alpha_n \cdot E_n$

?



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control of computing systems

## Slack reclaiming: optimal state feedback

- **Problem**: Find $[r_1, .., r_n]$ such as the sum of all control performance benefits is maximized, subject to the schedulability constraints $U_d$

$$\text{Maximize} \quad \sum_{i=1}^{n} P_i(r_i) E_i$$

$$\text{subject to} \quad \sum_{i=1}^{n} r_i \le U_d$$

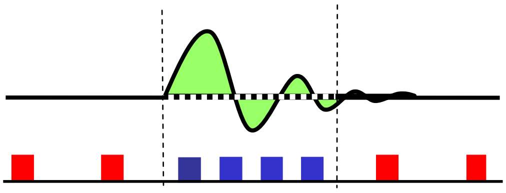
- **Solution**: $r_i = 0 (1 \le i \le n, i \ne j), \text{ and } r_j = U_d$

(assign all desired CPU to one task)

# Control of computing systems

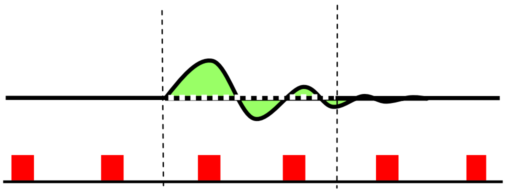
## Slack reclaiming: optimal state feedback

Task1



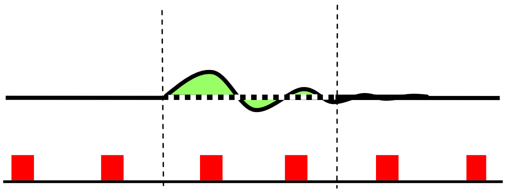
Run at **maximum** frequency if possible

Task2



Run at **minimum** frequency

Task3



Run at **minimum** frequency



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control of computing systems

## Slack reclaiming: optimal state feedback

- Optimal

$$r_i = \begin{cases} U_d - \sum_{1 \le j \le n, j \ne i} r_{j,\min}, & \text{if } \alpha_i E_i \text{ is maximum} \\ r_{i,\min}, & \text{otherwise} \end{cases}$$

- Proportional

$$r_i = \frac{\alpha_i E_i}{\sum_{1 \le j \le n} \alpha_j E_j} \cdot U_d$$

- Discrete

$$\alpha_i E_i \text{ is maximum} \Rightarrow r_i \text{ increases}$$

- Static

$$r_i = \frac{\alpha_i}{\sum_{1 \le j \le n} \alpha_j} \cdot U_d$$

# Control of computing systems

## Slack reclaiming: optimal state feedback

- System Configurations
  - RBED: Resource-rate Based Earliest Deadline scheduling
  - Linux 2.4.20
- Workloads
  - 3 inverted pendulums, each one with the same controller
    - ***worst execution time*** = 13.5ms, ***period*** = [30ms … 50ms]
    - Discrete: {30ms, 40ms, 50ms}
  - Random perturbations with different intervals
- Performance metrics
  - Control: cumulative error (IAE) $= \int_0^t \sum_{i=1}^4 |x_i(t)| dt$
  - CPU usage and overhead





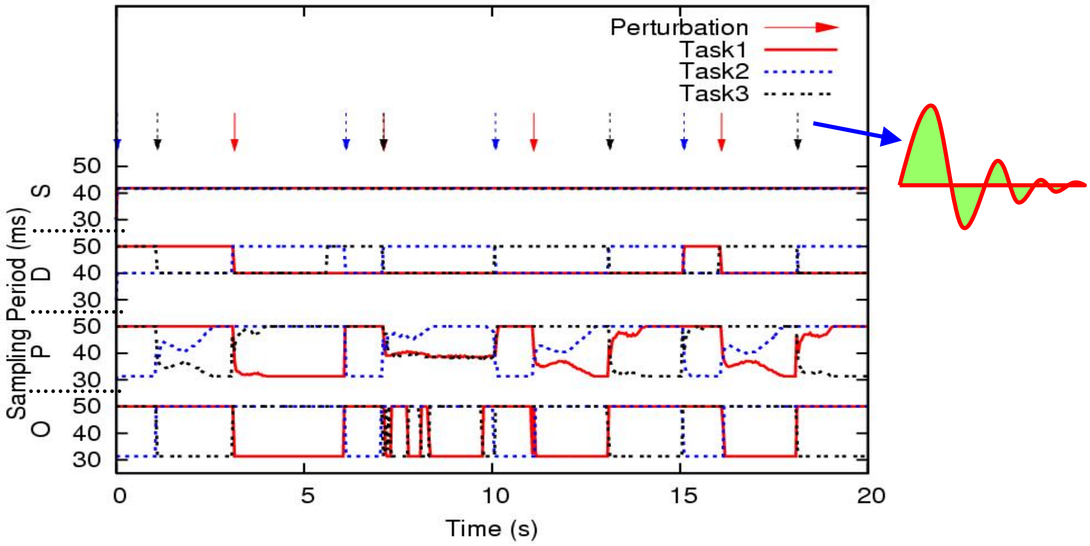
# Control of computing systems

## Slack reclaiming: optimal state feedback

```
Controller
{
    h_i := h_i^next
    x_i := read_input()
    u_i := calculate_output(h_i); send_output(u_i)
    x_i^next := update_state(x_i, h_i); e_i := |x_i^next|
    r_i := rate_adjust(i, e_i)
    h_i^next := c_i / r_i
}

rate_adjust(i, e_i)
{
    if (e_i != e_i^old)
    {
        b_i := w_i e_i p_i
        r_i^next := resource_allocation(policy, b_i)
        e_i^old := e_i
        return r_i^next
    }
}
```

# Control of computing systems

## Slack reclaiming: optimal state feedback




Mean interval between perturbations: 4s

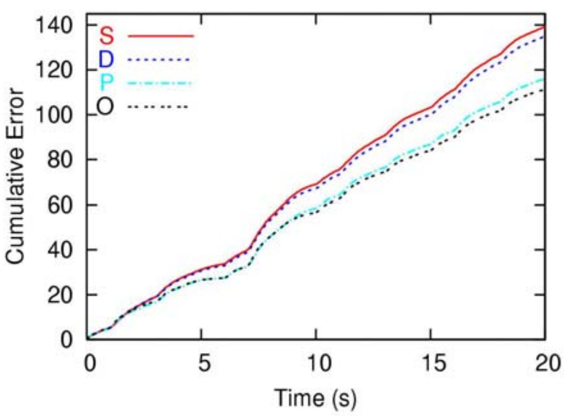


ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
UPC
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

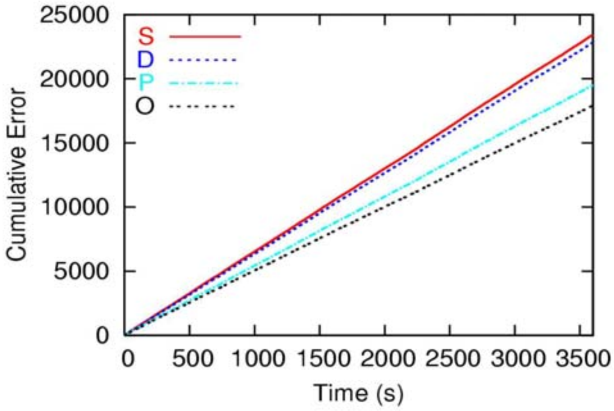
# Control of computing systems

## Slack reclaiming: optimal state feedback

Optimal is the **best** !!!



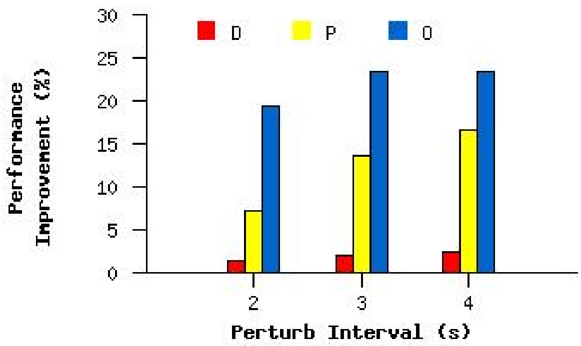

(a) 20 seconds



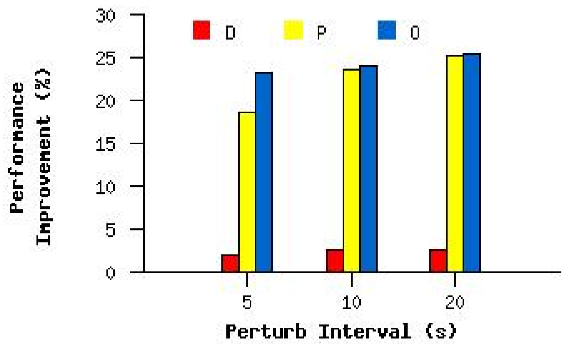

(b) 1 Hour

Mean interval between perturbations: 4s

# Control of computing systems




(a) More frequent perturbation




(b) Less frequent perturbation

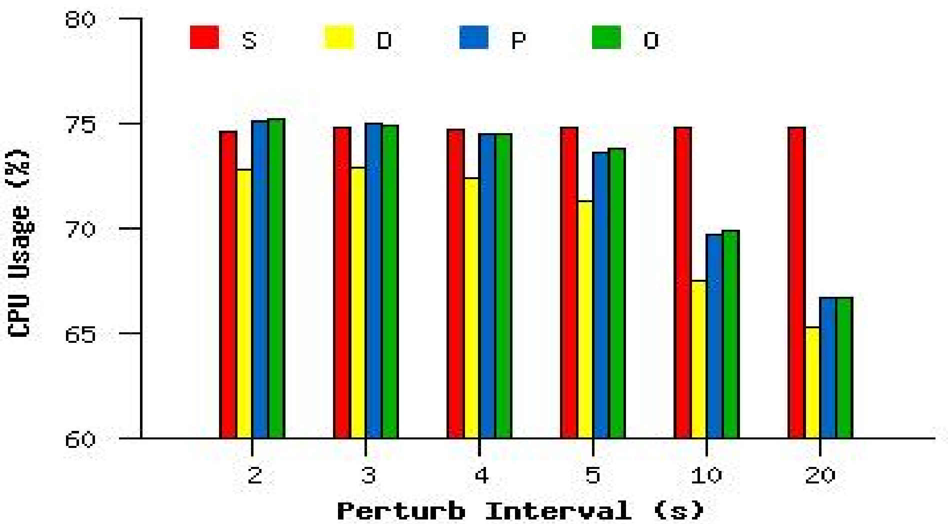
- adaptation works
- optimal is the best
- less frequent perturbation, better optimal
- more frequent, better optimal compared to proportional



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control of computing systems

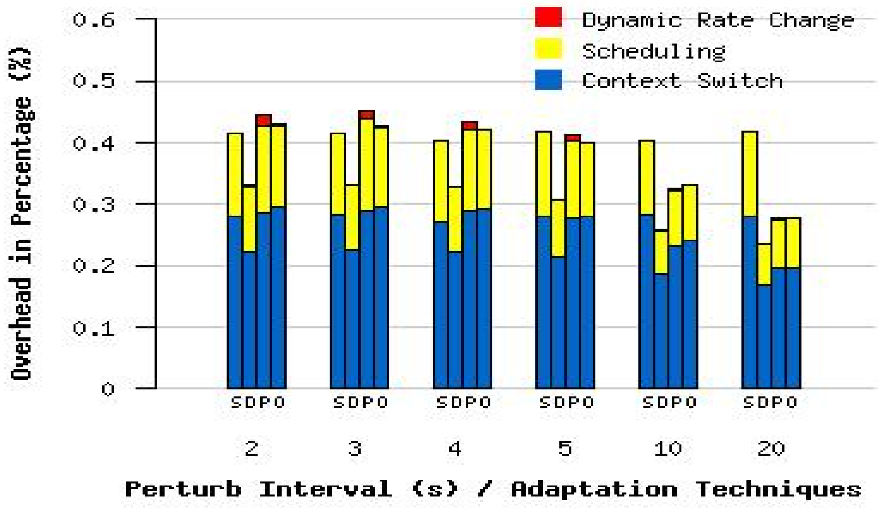
## Slack reclaiming: optimal state feedback




more frequent perturbations, optimal comparable to static
less frequent, better optimal compared to static

# Control of computing systems

## Slack reclaiming: optimal state feedback




Context switch dominates
Global and dynamic rate change: **negligible**





# Control of computing systems

## Slack reclaiming: optimal state feedback

- Feedback policies improve control performance + save resources.
- Optimal outperforms others and improves by 25% in the best scenario.
- Overhead of optimal policy is negligible

Can it be done distributed?

# Control of computing systems

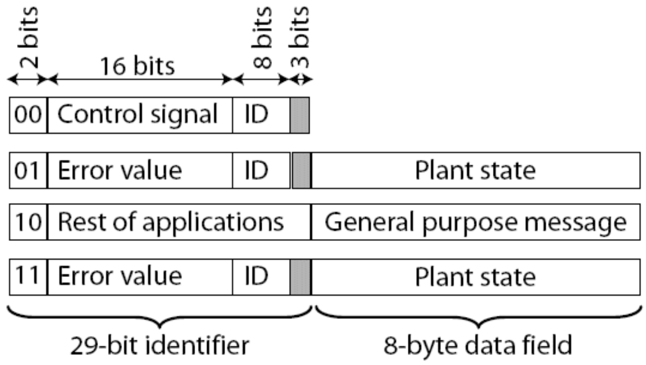
## Slack reclaiming: optimal state feedback

Can it be done distributed? Yes

Problem: it requires having centralized the errors of all plants

Solution: using CAN network and its bitwise arbitration mechanism

Idea: each message identifier is inversely proportional to each plant error

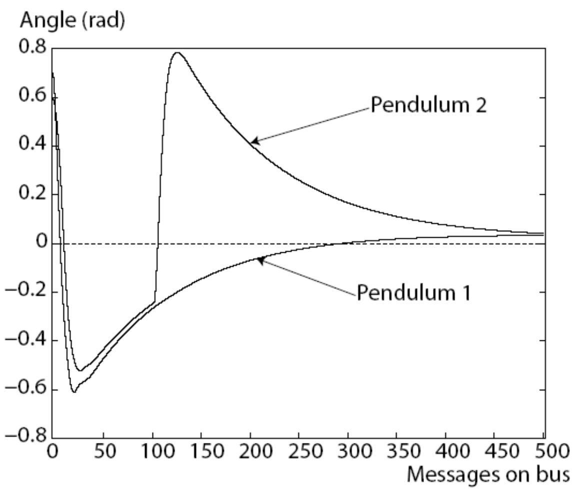


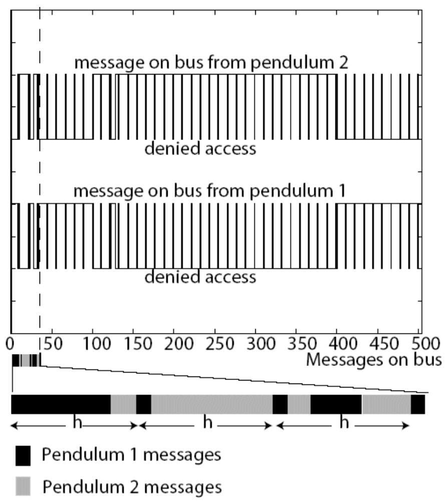



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control of computing systems

## Slack reclaiming: optimal state feedback



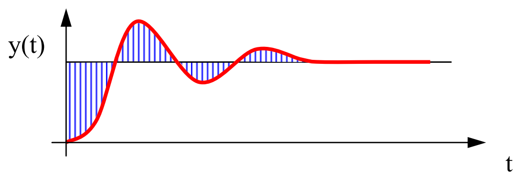
# Control of computing systems

## Overload conditions

Control in overload conditions (cpu, bandwidth): which is the minimum number of controllers required to keep a desired QoC?

Integral of the Absolute Error (**IAE**):

$$IAE = \int_0^\infty |e(t)|\,dt$$



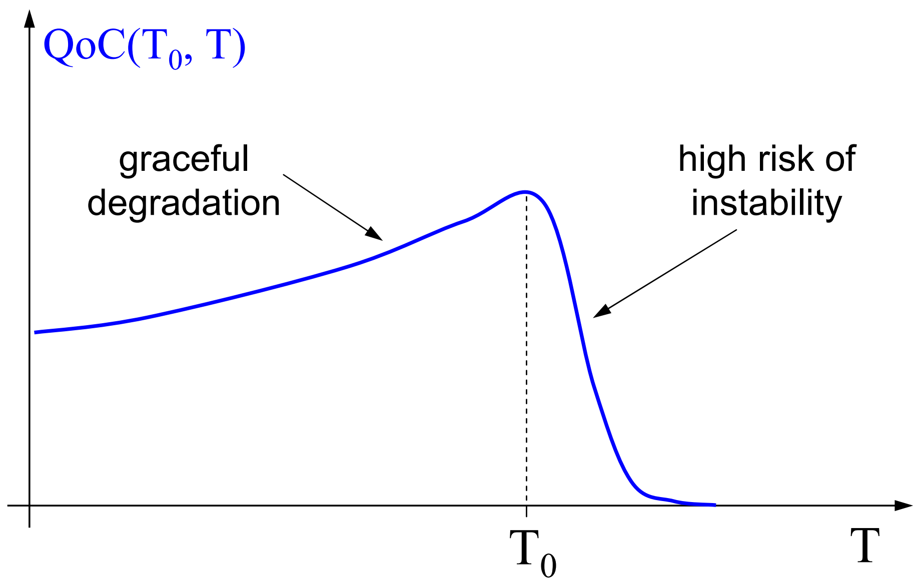

$$QoC = \frac{1}{IAE}$$



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

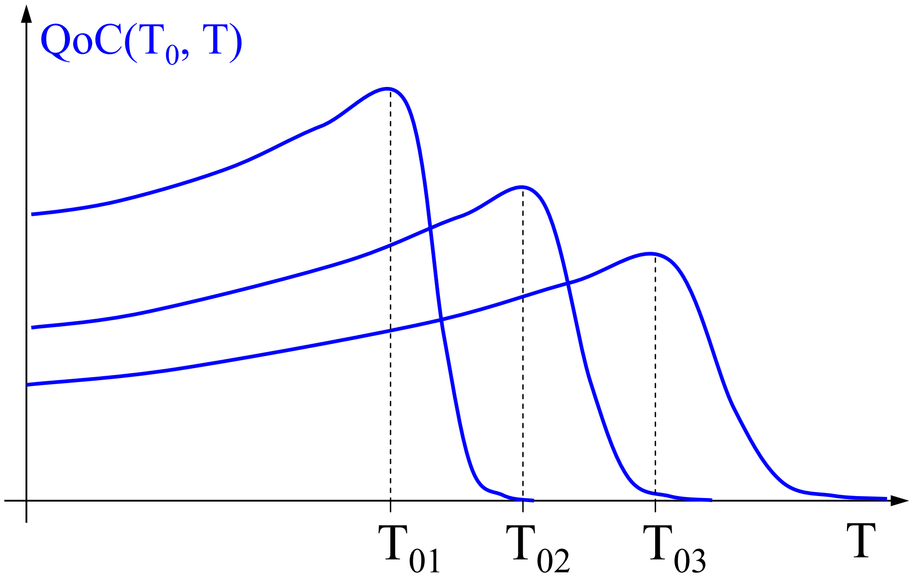
# Control of computing systems

## Overload conditions



# Control of computing systems

## Overload conditions

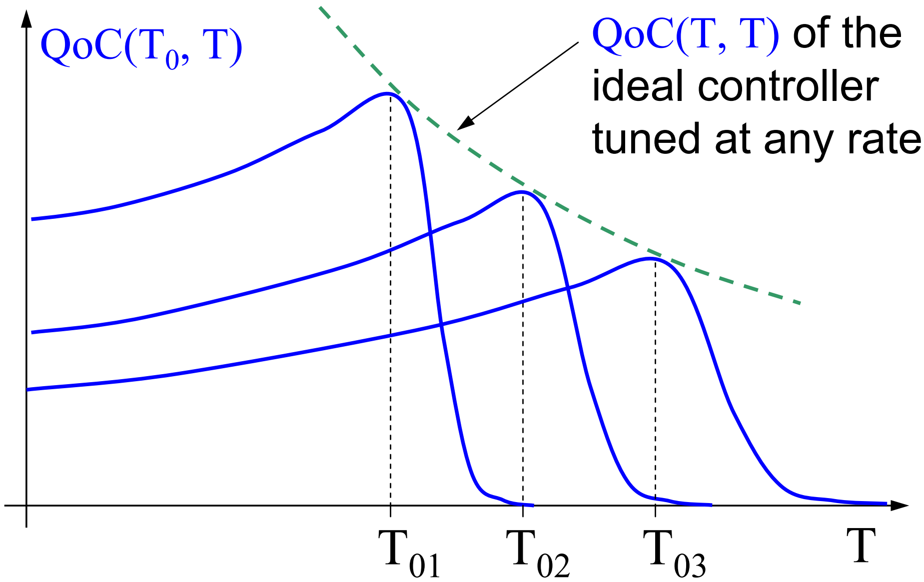






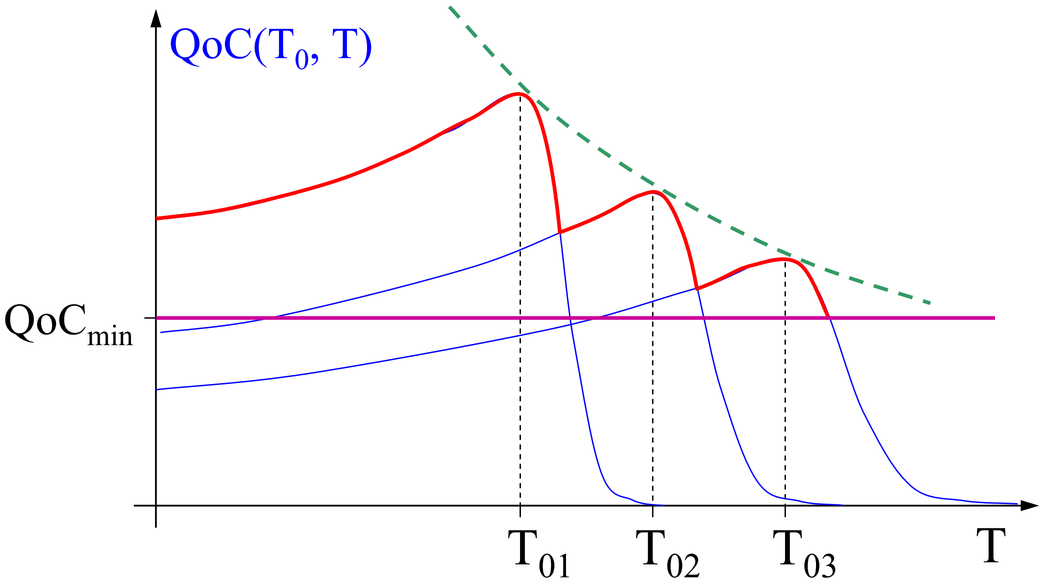

# Control of computing systems

## Overload conditions



# Control of computing systems

## Overload conditions




Switch controller to keep the QoC above a minimum

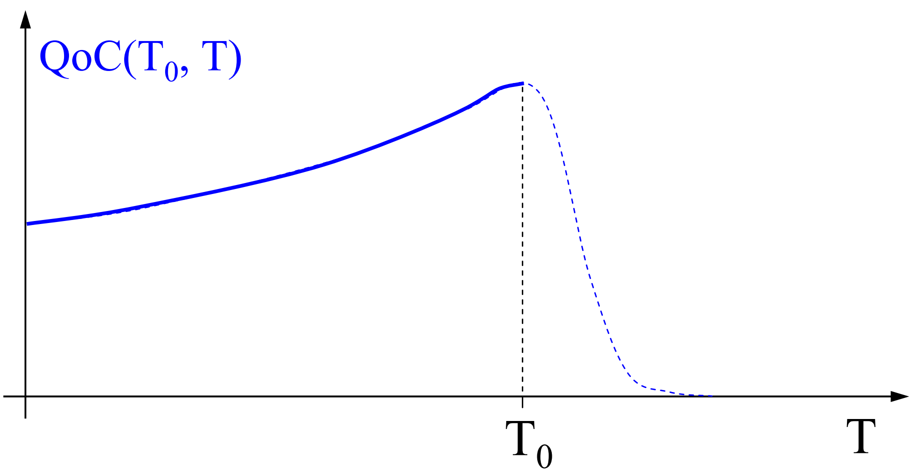




# Control of computing systems

## Overload conditions

To prevent instability, we decided to run a controller designed for $T_0$ only with $T_i \leq T_0$



# Control of computing systems

## Overload conditions

Minimize the number of controllers to be designed off line to keep the QoC within a given bound.

**The user specifies:**

| | |
|---|---|
| $QoC_{min}$ | minimum QoC level |
| $\varepsilon_{max}$ | maximum error (in %) w.r.t. the ideal envelop curve |

**Feasiblity analysis imposes:**

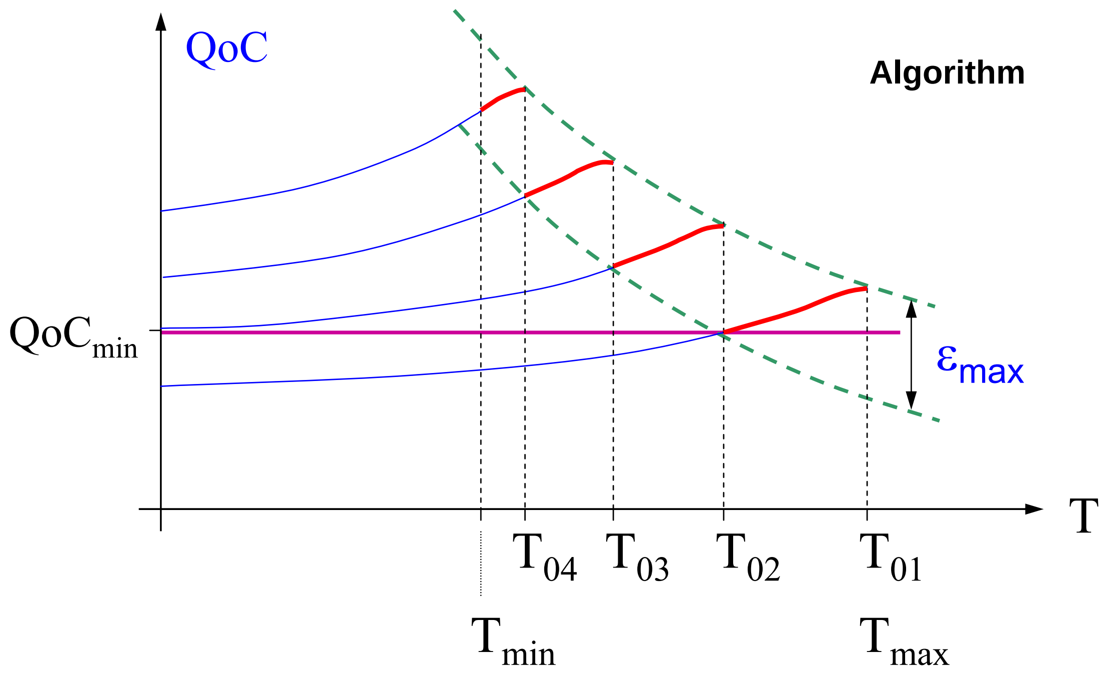
| | |
|---|---|
| $T_{min}$ | minimum period for a controller |



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control of computing systems

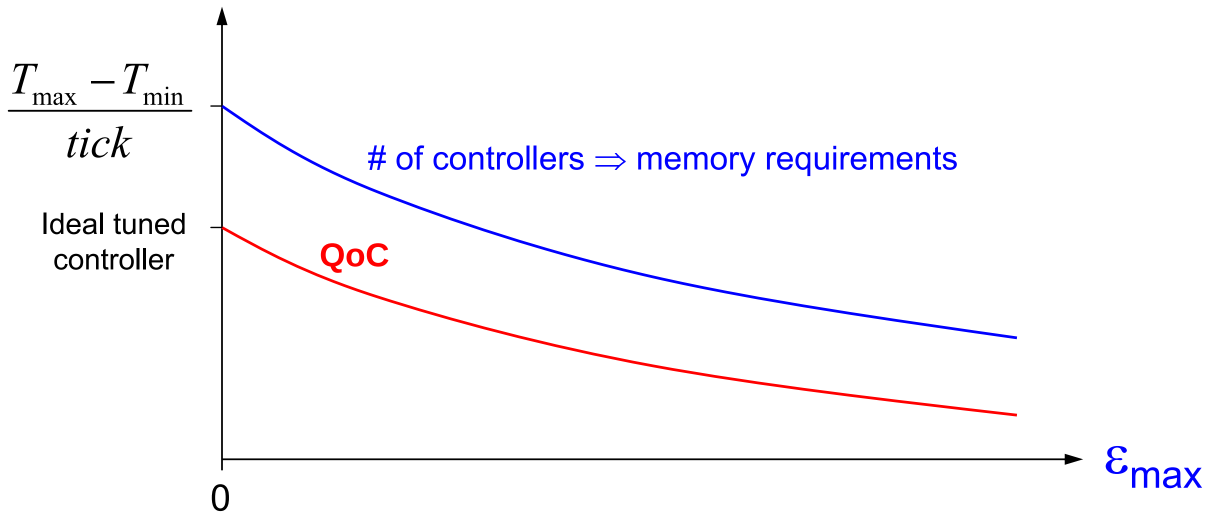
## Overload conditions



# Control of computing systems

## Overload conditions

The error $\varepsilon_{max}$ allows to balance control performance with memory requirements:






ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Control of computing systems

## Overload conditions

We tested our approach on an inverted pendulum.
Three strategies have been compared.

$$\begin{cases} \text{Period range (ms) = [250, 540]} \\ \text{System tick = 10 ms} \end{cases}$$

**Optimal task** ($\varepsilon_{max} = 0$). It requires 30 controllers.

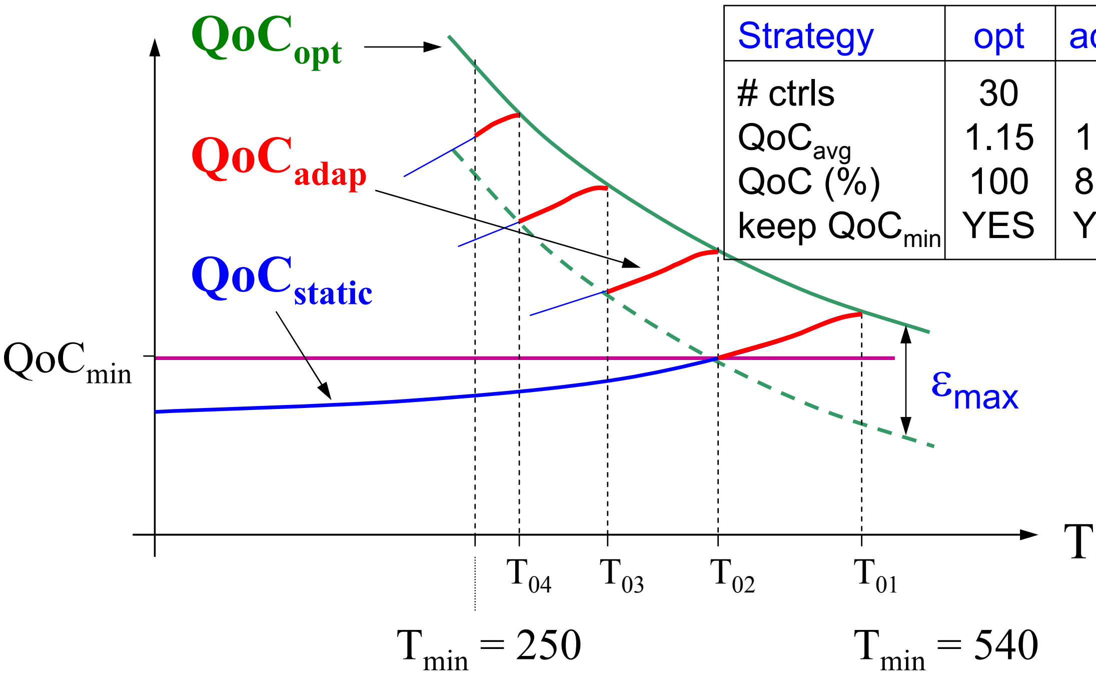
**Adaptive task** ($\varepsilon_{max} = 0.3$). It requires 4 controllers.

**Static task**. Single controller running at T = 540 ms.

# Control of computing systems

## Overload conditions

| Strategy | opt | adap | static |
|---|---|---|---|
| # ctrls | 30 | 4 | 1 |
| $QoC_{avg}$ | 1.15 | 1.09 | 0.67 |
| QoC (%) | 100 | 87.8 | 58.2 |
| keep $QoC_{min}$ | YES | YES | NO |








# Control of computing systems

## Overload conditions

- method to cope with overload conditions in real-time control systems.
- The method allows using continuous rate adaptation mechanisms (e.g. elastic scheduling) for efficient resource exploitation.
- It allows minimizing the number of controllers to be designed off line (and memory requirements), while keeping a desired QoC performance.

# Stability issues



ARTIST2
IST NoE on Embedded Systems Design
Seminar on Adaptive Real-Time Systems, with emphasis on Real-Time Control Systems
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Automatic Control Dept. (ESAII), Technical University of Catalonia (UPC), Barcelona, June 20-23, 2005

# Conclusions

# The end