Real-Time and Control Issues

Dr. Pau Marti and Dr. Manel Velasco

Technical University of Catalonia
Barcelona, Spain

Email: pau.marti@upc.edu, manel.velasco@upc.edu

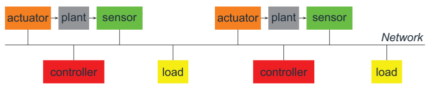
Networkd and Embedded Control Systems

From Theory to Practice

Manel Velasco and Pau Martí

Automatic Control Deptartment, Technical University of Catalonia
manel.velasco,pau.marti@upc.edu

Networked and Embedded Control Systems

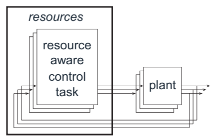


- Sensors, controllers and actuators physically distributed
- Control (and non-control) activities and data
- Real-time technology in nodes and network
- Resource limitations

Tentative contents

- Crash course on control
- Control design and resource management issues
  - Timing problems and solutions
  - Can resource management policies maximize control performance? ⟶ Feedback scheduling
  - Can control design strategies minimize resource utilization? ⟶ Event-driven control
- Simulation and implementation of NECS

Understanding the course scope



Event-driven control

Laboratory experiments:

- Simulation tools: Matlab/Simulink (or Scilab/Scicos)
- Executing platform: Erika/Flex/CAN
- Goal: CAN-based control of a double integrator electronic circuit

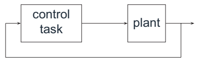
Reference books

- Control books, e.g. Åström *et al.* [1] or Hespanha [2]
- Real-time books, e.g. Burns *et al.* [3] or Buttazzo [4]
- Control applied to computing systems, e.g. Hellerstein *et al.* [5]

Tentative course outline

- Crash course on control
- Timing and implementation
- Feedback scheduling
- Event-driven control systems
- Conclusions

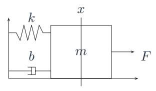
What is control?



It is the art of making systems to behave as we want instead of how they would behave.

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Simple example: mass-spring-damper (m-s-d)



- System dynamics (Newton's law): $\sum f_i = ma \rightarrow F - kx - b\dot{x} = m\ddot{x}$
- Desired output (y): desired position $\rightarrow y = x$
- Input (u): force $\rightarrow u = F$
- Model parameters: $m = 1$kg, $k = 1$N/m, $b = 1$Ns/m

What's the mass behavior?

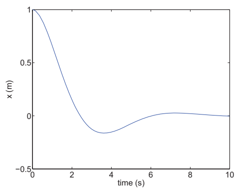
Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Simple example: The dynamics

Second order differential equation

$F = m\ddot{x} + b\dot{x} + kx \rightarrow y = x?$

With initial condition $x = 1$m, if no force is applied ($F = 0$):

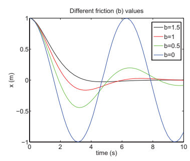


Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Simple example: The dynamics

Differential equation: $F = m\ddot{x} + b\dot{x} + kx$

- Solution ($F = 0$):
  $x(t) = C_1 e^{\frac{-b+\sqrt{b^2-4mk}}{2m}t} + C_2 e^{\frac{-b-\sqrt{b^2-4mk}}{2m}t} = C_1 e^{\lambda_1 t} + C_2 e^{\lambda_2 t}$



The coefficients λ_i are the system poles

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Simple example: Modeling $\rightarrow F = m\ddot{x} + b\dot{x} + kx$

- Simple input / simple output (SISO) state-space model:

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t)$$

  with $x \in \mathbb{R}^n$ and $u, y \in \mathbb{R}$, and n the system order
- ... in our case

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{-k}{m} & \frac{-b}{m} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Simple example: Modeling $F = m\ddot{x} + b\dot{x} + kx$

- Define state variables: $x_1 = x$ and $x_2 = \dot{x}$
- Substitute variables
  $\dot{x}_1 = x_2$
  $\dot{x}_2 = \ddot{x} = -\frac{k}{m} - \frac{b}{m} + \frac{F}{m}$

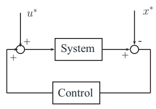
If $u = F$ and $y = x_1$, the resulting model is

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{-k}{m} & \frac{-b}{m} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

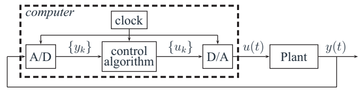
Learned topics

- System's dynamics may be changed easily, just use $u = k_1 x_1 + k_2 x_2 + \ldots$
- The system can only converge to its "natural" equilibrium point (most cases is 0). Equilibrium points are computed using $0 = Ax$
- If we want the system go to some other point it can be only a forced equilibrium point. Forced equilibrium points are computed using $0 = Ax + Bu$, and u is left as a parameter.
- The right way to make feedback is to use next diagram



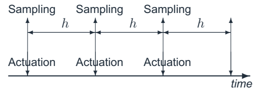
Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Models for discrete-time systems



Simplest mathematical model with

- constant sampling period
- instantaneous input-output latency



Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Basic discrete-time models

Linear time-invariant continuous-time system state-space

$$\begin{aligned} \frac{dx(t)}{dt} &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{1}$$

Discrete form, with zero-order-hold sampling period h [1]

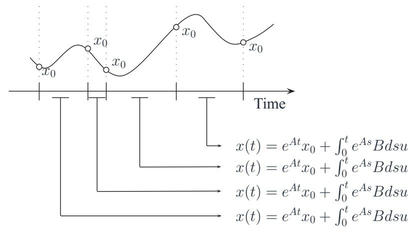
$$\begin{aligned} x_{k+1} &= \Phi(h)x_k + \Gamma(h)u_k \\ y_k &= Cx_k, \end{aligned} \tag{2}$$

where $\Phi(t)$ and $\Gamma(t)$ are obtained using the following

$$\Phi(t) = e^{At}, \quad \Gamma(t) = \int_0^t e^{As} B ds, \tag{3}$$

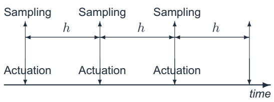
Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

The idea: where does the model come from?

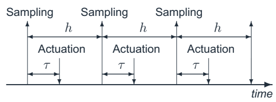


Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Timing: Timing of the basic model is not realistic

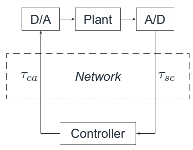


Adding a time delay to model an input/output latency due to the algorithm computation or the insertion of a network



Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

The network case



Delays controller-to-actuator τ_{ca} and sensor-to-controller τ_{sc} can be integrated into a single delay τ.

The network case

Continuous-time system with time delay τ

$$\begin{aligned}\frac{dx(t)}{dt} &= Ax(t) + Bu(t-\tau)\\ y(t) &= Cx(t)\end{aligned} \tag{4}$$

Discrete form, with $\tau \leq h$

$$\begin{aligned}x_{k+1} &= \Phi(h)x_k + \Phi(h-\tau)\Gamma(\tau)u_{k-1} + \Gamma(h-\tau)u_k.\\ y_k &= Cx_k,\end{aligned} \tag{5}$$

where $\Phi(t)$ and $\Gamma(t)$ are also obtained using (3).

The network case

State-space form for (5), extended model:

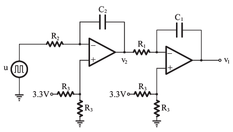
$$\begin{bmatrix} x_{k+1} \\ z_{k+1} \end{bmatrix} = \begin{bmatrix} \Phi(h) & \Phi(h-\tau)\Gamma(\tau) \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ z_k \end{bmatrix} + \begin{bmatrix} \Gamma(h-\tau) \\ I \end{bmatrix} u_k \tag{6}$$

where $z_k \in \mathbb{R}^{m\times 1}$ represent past control signals ($z_k = u_{k-1}$).

- This notation slightly differs from conventional notation [1] to stress dependencies on h and τ.
- The notation may be still misleading: u_k is applied τ time units after x_k is taken.
- Simplified notation of (6):

$$x_{k+1} = \Phi(h,\tau)x_k + \Gamma(h,\tau)u_k \tag{7}$$

Playing with Matlab/Simulink with our circuit [6]



Model ($R_1 = R_2 = 100\text{k}\Omega$, $C_1 = C_2 = 470n\text{F}$, $R_3 = 1\text{k}\Omega$)

$$\begin{aligned}\begin{bmatrix} \dot v_1(t) \\ \dot v_2(t) \end{bmatrix} &= \begin{bmatrix} 0 & -21.2766 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v_1(t) \\ v_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ -21.2766 \end{bmatrix} u(t)\\ y(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} v_1(t) \\ v_2(t) \end{bmatrix}\end{aligned}$$

Playing with Matlab/Simulink with our circuit [6]

Computing $\Phi(h)$ and $\Gamma(h)$ in (2):

A=[0 -21.2766;0 0]; B=[0;-21.2766]; C=[1 0]; D=[0];

- symbolically:
  syms h; syms s;
  Phi=expm(A*h); Gamma=int(expm(A*s),s,0,h)*B
- numerically:
  h=1; sys=ss(A,B,C,D); dsys=c2d(sys,h,'zoh');
  [Phi,Gamma,dC,dD]=ssdata(dsys)

And $\Phi(h,\tau)$ and $\Gamma(h,\tau)$ in (7)?

- symbolically: do it
- numerically:
  h=1; tau=0.5; sys=ss(A,B,C,D, 'iodelay', tau);
  dsys=c2d(sys,h,'zoh'); [Phi,Gamma,dC,dD]=ssdata(dsys)

Systems' properties (1)

Stability: A system is stable if its state variables always decrease from an initial value and do not show permanent oscillations.

- CLTI: The system is stable if all eigenvalues of A have strictly negative real parts: $\forall\lambda_i[A],\ \mathbb{R}\lambda_i[A] < 0$
- DLTI: The system is stable if all eigenvalues of Φ have magnitude strictly smaller than 1: $\forall\lambda_i[\Phi],\ |\lambda_i[\Phi]| < 1$

Note that the relation between the s and z plane: $z_i = e^{s_i h}$

Related topics: equilibrium points, Lyapunov stability [2]

Systems' properties (2)

Controllability: a system is controllable (reachable) if it is possible to find a control sequence such that the origin (an arbitrary state) can be reached from any initial state in finite time.

- Controllability matrix: $W_c = \begin{bmatrix}\Gamma & \Phi\Gamma & \cdots & \Phi^{n-1}\Gamma\end{bmatrix}$
- Controllability test: $rank(W_c) = n$

If a system is not controllable, re-think actuators...

Systems' properties (3)

Observability: a system is observable if there is a finite k such that the knowledge of the inputs $u(0), \ldots, u(k-1)$ and the outputs $y(0), \ldots, y(k-1)$ is sufficient to determine the initial state of the system.

- Observability matrix: $W_o = \begin{bmatrix} C \\ \vdots \\ C\Phi^{n-1} \end{bmatrix}$
- Observability test: $rank(W_o) = n$

If a system is not observable, re-think sensors...

Controller design: the goal

To shape the system response by *State Feedback*

Pole placement:

- system response depends on the system poles
- *pole placement* allows to specify arbitrary pole locations

Optimal control: to minimize a cost function on states and inputs

- widely used technique
- it can also be treated by specifying pole locations

Controller design: standard model (2)

For closed loop operation of (2), given

$$u_k = -K(h)x_k \tag{8}$$

where $K(h)$ is the state feedback gain, the closed loop system evolution is

$$\begin{aligned} x_{k+1} &= \Phi(h)x_k + \Gamma(h)u_k = \\ &= \Phi(h)x_k - \Gamma(h)K(h)x_k \\ &= (\Phi(h) - \Gamma(h)K(h))\, x_k = \Phi_{cl}(h)x_k \end{aligned} \tag{9}$$

Controller design: model with delay (6)

For closed loop operation of (6), given

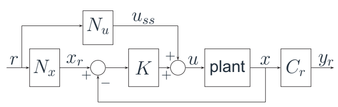
$$u_k = -K(h,\tau) \begin{bmatrix} x_k \\ z_k \end{bmatrix} \tag{10}$$

where $K(h,\tau)$ is the state feedback gain, the closed-loop system evolution is

$$\begin{aligned} \begin{bmatrix} x_{k+1} \\ z_{k+1} \end{bmatrix} &= \left(\begin{bmatrix} \Phi(h) & \Phi(h-\tau)\Gamma(\tau) \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} \Gamma(h-\tau) \\ I \end{bmatrix} k(h,\tau) \right) \begin{bmatrix} x_k \\ z_k \end{bmatrix} = \\ &= \Phi_{cl}(h,\tau) \begin{bmatrix} x_k \\ z_k \end{bmatrix} \end{aligned} \tag{11}$$

($K(h,\tau)$ of appropriate dimensions)

Controller design: tracking



- N_u is the feedforwad matrix to eliminate steady-state errors
- N_x translates the reference signal into a reference vector

$$\begin{pmatrix} \mathbf{N_x} \\ \mathbf{N}_u \end{pmatrix} = \begin{pmatrix} \mathbf{\Phi - I} & \mathbf{\Gamma} \\ \mathbf{C}_r & 0 \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Playing with Matlab/Simulink with our circuit [6]

Controller design: *acker* for pole placement

```
A=[0 -21.2766;0 0]; B=[0;-21.2766]; C=[1 0]; D=[0];
sys=ss(A,B,C,D);
h=1;
dsys=c2d(sys,1,'zoh');
contpoles=[-3,-2];
discpoles=exp(contpoles*h)
Kd=acker(Phi,Gamma,discpoles);
closedloop=ss(dsys.a-dsys.b*Kd,dsys.b,dsys.c,dsys.d,h);
initialconditions=[1;0];
initial(closedloop,initialconditions,10)
```

Play with different poles, periods, initial cond., and Simulink

Sampling period (h) selection:

$h \longrightarrow$ tradeoff control performance and resource utilization

- From control to resources
  - pole location determines system response/performance
  - pole location determines the choice of the sampling period h
  - h determines resource utilization
- From resources to control
  - resource limitations restricts h
  - the choice of h determines feasible pole locations
  - pole location determines system response/performance

Sampling period (h) selection:

- Schedulability bounds affect tasks/messages periods

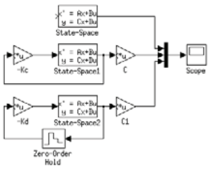
$$\text{For } n \text{ tasks, } \sum_{i=1}^{n} \frac{c_i}{h_i} \leq U^{alg}$$

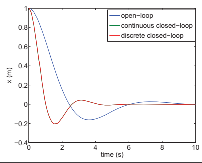
- Rules of thumb for sampling period selection according to the continuous-time poles allow different choices
  - If fastest pole $s_i \in \mathbb{R} \rightarrow h \leq \frac{t_r}{N}$, with $N = 4$ to 10
  - If fastest poles $s_{1,2} \in \mathbb{C} \rightarrow h \leq \frac{2\pi}{N\omega_d}$, with $N = 10$ to 20

where $t_r = \frac{1}{s_i}$ is the rise time, and $\omega_d = \omega_n\sqrt{1-\xi^2}$ is the response oscillation where $s_{1,2} = -\omega_n\xi \pm j\omega_n\sqrt{1-\xi^2}$

Sampling period (h) selection: m-s-d control

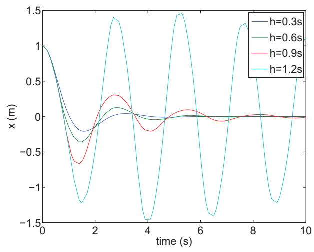
```
b=1;m=1;k=1
A=[0 1;-k/m -b/m];
B=[0;1/m];
C=[1 0];
D=[0];
cpoles=[-1+2i -1-2i];
Kc=acker(A,B,cpoles);
h=0.3 % note that h=2*pi/(10*2);
dpoles=exp(cpoles*h);
csys=ss(A,B,C,D);
dsys=c2d(csys,h,'zoh');
Kd=acker(dsys.a,dsys.b,dpoles);
```





Sampling period (h) selection: m-s-d control

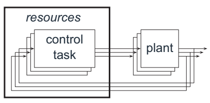
Choosing different periods



Tentative course outline

- Crash course on control
- Timing and implementation
- Feedback scheduling
- Event-driven control systems
- Conclusions

Timing and implementation of control algorithms



Several loops sharing a processor or network

Double integrator: differential equation and model

$$\frac{d^2y}{dt^2} = u \tag{12}$$

If y and $\dot{y}$ are x_1 and x_2, a state space form is given by

$$\begin{aligned} \dot{x}(t) &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) \end{aligned} \tag{13}$$

Discrete-time model, with period h and delay τ

$$x_{k+1} = \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} \tau\left(h - \frac{\tau}{2}\right) \\ \tau \end{bmatrix} u_{k-1} + \begin{bmatrix} \frac{(h-\tau)^2}{2} \\ h - \tau \end{bmatrix} u_k \tag{14}$$

Double integrator: processing speed

If $h = 0.1$s and $\tau = 0.01$s, the state space form is given by

$$x_{k+1} = \begin{bmatrix} 1 & 0.1 & 0.001 \\ 0 & 1 & 0.01 \\ 0 & 0 & 0 \end{bmatrix} x_k + \begin{bmatrix} 0.004 \\ 0.09 \\ 1 \end{bmatrix} u_k \tag{15}$$

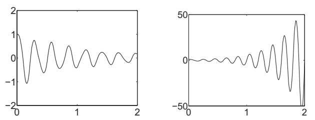
Closing the loop with $u_k = -\begin{bmatrix} 271.7 & 21.86 & 0.23 \end{bmatrix} x_k$, the discrete closed loop poles are

$\lambda_1 = -0.3,\ \lambda_2 = -0.1,\ \lambda_3 = -0.9$

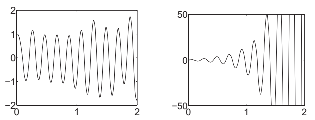
- With a faster micro $\tau = 0.005$, poles go at $\lambda_1 = 0.0082 - 0.2850i,\ \lambda_2 = 0.0082 + 0.2850i,\ \lambda_3 = -1.5513$
- With a slower micro $\tau = 0.02,\ \lambda_1 = -0.4910 + 0.9516i,\ \lambda_2 = -0.4910 - 0.9516i,\ \lambda_3 = 0.1316,$ with $|\lambda_1| = |\lambda_2| = 1.0708$

Double integrator: Random periods/delays

Random delay (with $\tau \in \begin{bmatrix} 0.005 & 0.015 \end{bmatrix}$, where $\tau_d = 0.01$)



Random period (with $h \in \begin{bmatrix} 0.05 & 0.15 \end{bmatrix}$, where $h_d = 0.1$)



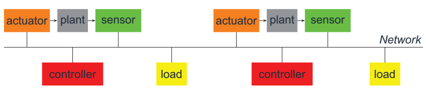
When does timing affect control loop performance?

- When the selected dynamics for the closed loop system are as fast as the sampling sampling
- When the natural system dynamics are as fast as the sampling
- When placed poles are near -1
- When the computing platform does not guarantee the assumed timing!
- When the computing platform introduces timing uncertainty

"How much" control performance is affected by timing issues?

- it depends on how performance is measured.
- A standard approach is to use a cost function.

$$J_c = \int_0^\infty x(t)^T Q_{c1} x(t) + u(t)^T Q_{c2} u(t) dt. \tag{16}$$



Network

- schedule + medium access control
- message transmission
- packet dropouts

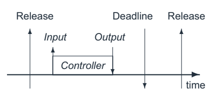
Nodes

- schedule
- task execution

Non-equidistant sampling/actuation and varying time delays

First controller design, then implementation [7]

Control task as a hard real time task under an arbitrary schedule



Naïf control task model

- task period equal to sampling period T=h
- sampling and actuation occur at the beginning and termination of each job execution
- and task deadline bounding the time delay, D$\geq \tau$

Sources of uncertainty: separation of concerns

Example: controller design

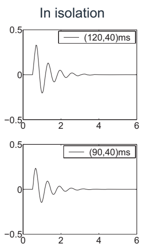
$$\frac{d^2y}{dt^2} = u$$

Let's locate the continuous closed loop poles at $\lambda_{1,2} = -1.5 \pm 10i$.

task	h	C=τ
A	120 ms	40 ms
B	90 ms	40 ms

$K_{120,40} = \begin{bmatrix} 75.8572 & 10.1051 & 0.3435 \end{bmatrix}$
$K_{90,40} = \begin{bmatrix} 83.5998 & 9.7351 & 0.3225 \end{bmatrix}$

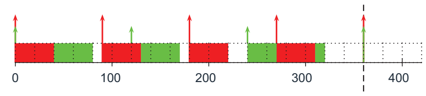
In isolation

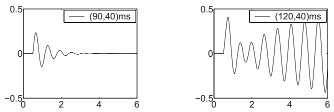


Sources of uncertainty: separation of concerns

Example: real-time computing implementation.

Offline schedule (in ms) (the two tasks sharing the CPU)





Simple example of co-design

Analyze timing and design accordingly:

If h and τ vary at each job execution, the model is given by

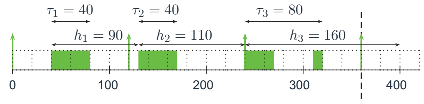
$$\begin{bmatrix} x_{k+1} \\ z_{k+1} \end{bmatrix} = \begin{bmatrix} \Phi(h_k) & \Phi(h_k-\tau_k)\Gamma(\tau_k) \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ z_k \end{bmatrix} + \begin{bmatrix} \Gamma(h_k-\tau_k) \\ I \end{bmatrix} u_k$$

where $x_{k+1} = x(\sum_{i=0}^{k+1} h_i)$ and $z_{k+1} = z(\sum_{i=0}^{k+1} h_i)$, which is a family of models. Observations:

- Given the sequence $\{h_k, \tau_k\}$ values, a specific system is obtained and it can be analyzed
- Controllability and observability
- Stability

Simple example of co-design

Task A was affected by jitters in the offline schedule, and the response became unstable.

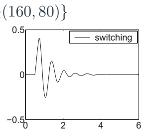


Let's approach A as a switched system:

$\Phi_{cl}(h,\tau) \in \{\Phi_{cl}(90,40), \Phi_{cl}(110,40), \Phi_{cl}(160,80)\}$

with

$K_{90,40} = \begin{bmatrix} 83.5998 & 9.7351 & 0.3225 \end{bmatrix}$
$K_{110,40} = \begin{bmatrix} 78.4894 & 10.0117 & 0.3377 \end{bmatrix}$
$K_{160,80} = \begin{bmatrix} 65.0282 & 12.7871 & 0.8149 \end{bmatrix}$



Simple example of co-design

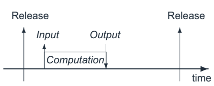
Is the previous approach enough?

- Stability, controlability and observability not checked !
- No performance guarantees !
- Computing overheard if gains are calculated online?
- Memory overhead for storing the gains calculated offline?
- Can we reduce the number of gains and still obtain acceptable control performance? See [8]

Yet another simple example of co-design

Apply a more effective task/message and controller model

Naïf approach [7]



One-sample approach [9], [10]

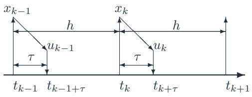
One-shot approach [11]

Yet another simple example of co-design

One-shot: Model (6) in closed-loop form is based on two synchronization points, on a time reference given by the sampling instants.

$$\begin{bmatrix} x_{k+1} \\ z_{k+1} \end{bmatrix} = \begin{bmatrix} \Phi(h) & \Phi(h-\tau)\Gamma(\tau) \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ z_k \end{bmatrix} + \begin{bmatrix} \Gamma(h-\tau) \\ I \end{bmatrix} u_k$$

$$u_k = \begin{bmatrix} K_1 & K_2 \end{bmatrix} \begin{bmatrix} x_k \\ z_k \end{bmatrix} = K_1 x_k + K_2 z_k : K_1 \in \mathbb{R}^{1\times n}, \quad K_2 \in \mathbb{R}^{1\times m}$$



Yet another simple example of co-design

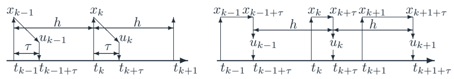
Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

One-shot: constructing the model by changing time coordinates to a time reference given by the actuation instants.

$$x_{k+\tau+1} = \Phi(h)x_{k+\tau} + \Gamma(h)u_k, \quad \text{with} \quad u_k = Kx_{k+\tau} \qquad (17)$$

$x_{k+\tau}$ has to be predicted from x_k:

$$x_{k+\tau} = \Phi(\tau)x_k + \Gamma(\tau)u_{k-1}. \qquad (18)$$



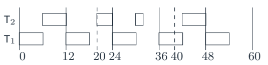
It admits irregular sampling

Yet another simple example of co-design

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

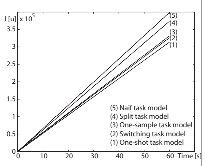
One-shot: Simulation of naif, one-sample, switching, one-shot, split [12]

	h	C
T_1	12	6
T_2	20	6



Voltage stabilizer (RCRC circuit)

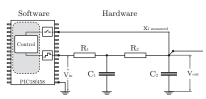
$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -918.2 & -90.9 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 918.2 \end{bmatrix} u(t)$$

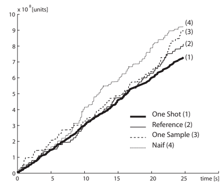


Yet another simple example of co-design

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

One-shot: Implementation on PIC18





Also implemented in Erika/Flex and over CAN

Contents

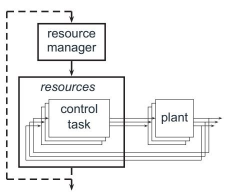
Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Tentative course outline

Feedback scheduling

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Can resource management improve control performance?



Feedback scheduling
... or "clever" sampling period selection ...

Feedback scheduling: motivation

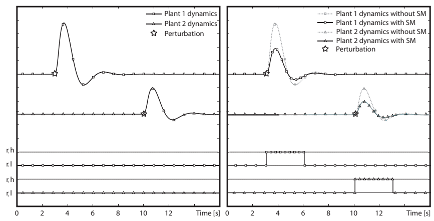
- Common approach to NECS: to assume the periodic execution of control algorithms.
- Advantages
  - maturity of digital control theory
  - simplicity of the schedulability analysis for periodic workloads
- Disadvantages
  - the selection of fixed rates of execution is not an easy task: low rates ↔ low resource usage ↔ low control performance
  - enforcing a fixed rate can be inappropriate in front of changes in the CPU/network load and in the controlled plants.

To overcome this periodicity limitation ⟶ FBS

Feedback scheduling: intuitive example

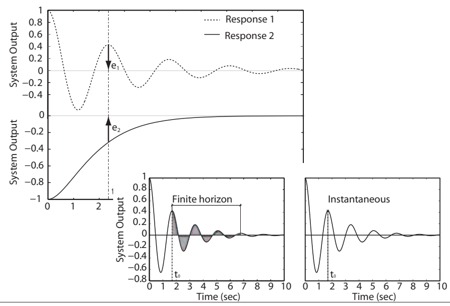
A Policy: Assign higher rates to "worse" control loops, s.t.

- not all loops can execute at their fastest rate
- higher rates imply better control performance



Feedback scheduling: intuitive example

What is "worse"?



Feedback scheduling: goal and questions

Apply efficient sampling period selection techniques that account for load and plants dynamics in such a way that the aggregated control performance delivered by the set of control loops is improved.

Questions [13]:

- How to formulate the problem?
- When to solve it (offline/online)?
- Which dynamics are included (workload/plants)?
- Target architecture (processor/network)?
- Implementation feasibility?

Feedback scheduling: processor systems

Common formulation: optimization problem

minimize (maximize):	penalty (benefit) on control performance
with respect to:	sampling periods / job execution / controllers
subject to:	closed loop stability
	task set schedulability

Two type of results

- Optimal sampling periods (e.g., [14], [15],[16],[17],[18],[19],[20])
- Optimal job sequence (e.g., [21], [22], [23])

Remark: mature discipline

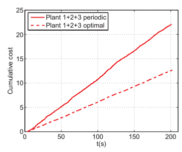
Feedback scheduling: processor systems

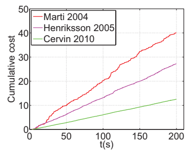
Approach	Opt.	When	Dynamics	PI	Sol.
[14]	Yes	Off			periods
[15]	Yes	On	kernel		periods
[16]	Yes	On	kernel/plant	Inst.	periods
[18]	Yes	On	kernel/plant	FH	periods
[21]	Yes	Off/On	kernel/plant	FH	seq.

Three tasks controlling RCRC circuits (Simulation)

Approach	Cost
Static approach	105.82
Off-line RM [14]	96.59
On-line FS [15]	98.74
On-line instantaneous FS [16]	64.41
On-line finite horizon FS [18]	86.99
Heuristic on-line cyclic scheduling [21]	62.48

Latest results: “Optimal on-line sampling assignment” [20]





- Periods are assigned according to plant states and noise while meeting a given processor utilization bound
- Implementation on Erika/Flex with 3 tasks controlling 3 double integrators

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Remarks

- mature discipline
- finding feasible solutions to the optimization problems is not easy ⟶ should the problem be formulated using alternative methods?
- still good performance numbers are obtained

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Can the same results be transferred to the networked case?

NO ! .. in the general case

- It's feasible for off-line approaches, e.g. [24],[25]
- Not feasible for on-line approaches because the information to solve the optimization problem is physically distributed

... but the particular case of CAN some on-line approaches can be adapted or developed... e.g. [26],[27],[28]

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Why in CAN? Intuitive example:

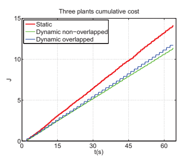
Recall that policy [16]: “always execute the task with biggest plant error”

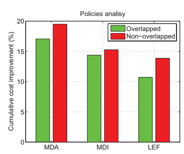
- Observation: CAN schedules messages on a priority based semantics
- Key idea: to encode plants errors into each message identifier in order to have the solution implemented at the CAN arbitration.

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Latest results: “Run-time allocation of optional jobs” [28]

- standard periodic messaging is guaranteed to ensure stability
- non-periodic additional messaging is added whenever the bus is idle to improve control performance





Implemented in Erika/Flex/CAN with 3 double integrator loops

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

Remarks

- not a mature discipline
- in CAN, solutions require to be specified as MAX or MIN functions
- for other networks, novel problem formulations are required
- again, good performance numbers are obtained

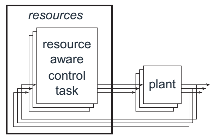
Contents

Tentative course outline

- Crash course on control
- Timing and implementation
- Feedback scheduling
- Event-driven control systems
- Conclusions

Event-driven control

Can control design strategies minimize resource usage?



Event-driven control
... or actuating only “when needed” ...

Event-driven control: the same motivation!!

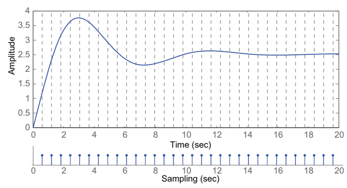
- Common approach to NECS: to assume the periodic execution of control algorithms.
- Advantages
  - maturity of digital control theory
  - simplicity of the schedulability analysis for periodic workloads
- Disadvantages
  - the selection of fixed rates of execution is not an easy task: low rates ↔ low resource usage ↔ low control performance
  - enforcing a fixed rate can be inappropriate in front of changes in the CPU/network load and in the controlled plants.

To overcome this periodicity limitation → Event-driven control

Event-driven control: Intuitive example

Periodic controller

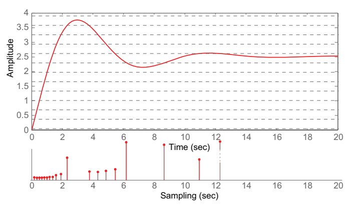
- control jobs are triggered by the progression of time
- constant control task period



Event-driven control: Intuitive example

Event-driven controller

- control jobs are triggered upon some control-related condition
- non-constant control task period



Event-driven control: different goal and questions

Execute control jobs only when required (to bound the intersampling dynamics or to ensure stability) in order to minimize resource utilization.

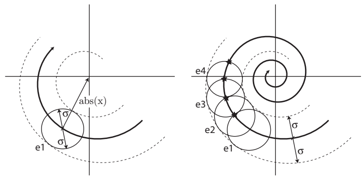
How to detect when to execute a control job?

- Include an analog event detector, e.g. [29],[30],[31],[32]
- The measurement method itself is event-based, e.g. [33]
- Assume a coordinator aware of all plant states, e.g. [34]
- Enforce a minimum inter-execution time, e.g. [35]
- Observe the occurrence of the event (self-triggered), e.g. [36], [37]

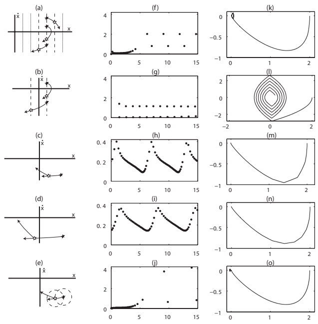
Event-driven control: common formulation

From the sampled state

1. a boundary in terms of the tolerated error (e_i, σ), is set,
2. a control action is taken, and
3. the system trajectory evolves.
4. When the controlled system trajectory reaches the boundary, a new sample is taken, and go to 1

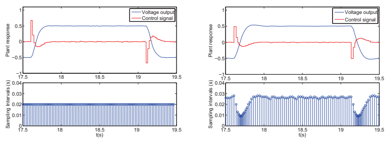


Event-driven control: examples



Event-driven control: a recent result

“Self-triggered Networked control Systems” [38]



- Control messages are triggered according to restrictions on the plan dynamics
- self-triggered: similar cost and less messages !
- On Erika/Flex/CAN with 1 loop (double integrator)

Event-driven control: remarks

non-mature discipline for either control and real-time

- for control
  - specifications of event-conditions (and finding techniques for achieving self-triggering)
  - ... and associated stability and performance guarantees
  - design of sensors
  - the networked case, e.g. distributed event-detection
- for real-time
  - characterization of sampling sequences [39]
  - schedulability analysis for processor [40] and networks [27]

Contents

Tentative course outline

- Crash course on control
- Timing and implementation
- Feedback scheduling
- Event-driven control systems
- Conclusions

Conclusions

- Impact of computing on control: done $\longrightarrow$ control engineers did (and are doing) the job
- Feedback scheduling: almost done $\longrightarrow$ the network case is open, and requires both control and real-time engineers.
- Event-driven control: open $\longrightarrow$ control engineers are progressing. Real-time engineers need to help!

References

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

[1] Åström, K.J., Wittenmark, B., *Computer controlled systems*. Prentice Hall, 1997
[2] Hespanha, J.P, *Linear Systems Theory*. Princeton Press, 2009
[3] Burns, A. and Wellings, A. J., *Real-Time Systems and Programming Languages: ADA 95, Real-Time Java, and Real-Time POSIX*. 3rd. Edition, Addison-Wesley, 2001
[4] Buttazzo, G., *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*. Second Edition, Springer, 2005
[5] Hellerstein, J.L., Diao, Y., Parekh, S., and Tilbury, D.M., *Feedback Control of Computing Systems*, Wiley-IEEE Press, 2004
[6] Velasco, M., Martí, P., Fuertes, J.M., Lozoya, C., Brandt, S., "Experimental Evaluation of Slack Management in Real-Time Control Systems: Coordinated vs. Self-Triggered Approach," *ournal of Systems Architecture*, Vol. 56, N. 1, January 2010
[7] Årzén, K.-E., Cervin, A., Eker, J., Sha, L., "An introduction to control and scheduling co-design,", 39th IEEE Conference on Decision and Control, 2000
[8] Buttazzo, G., Velasco, M., and Martí, P., "Quality-of-Control Management in Overloaded Real-Time Systems," *IEEE Transactions on Computers,* vol. 56, No. 2, pp. 253-266, 2007
[9] Liu, C.L., and Layland, J.W., "Scheduling algorithms for multiprogramming in a hard real-time environment," *Journal of the ACM* 20(1), pp. 40–61, 1973

References (1)

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

[10] Henzinger, T.A., Horowitz, B., and Kirsch, C.M., "Giotto: a time-triggered language for embedded programming", First International Workshop on Embedded Software, LNCS 2211, pp. 166-184, 2001
[11] Lozoya, C., Velasco, M., and Martí, P., "The one-shot task model for robust real-time embedded control systems," *IEEE Transactions on Industrial Informatics*, 4(3):164–174, Aug. 2008
[12] Balbastre, P., Ripoll, I., Vidal, J., Crespo, A., "A task model to reduce control delays," *Journal of Real-Time Systems*, vol.27, n.3, pp. 215–236, 2004
[13] Lozoya, C., Martí, P., Velasco, M., "Minimizing Control Cost in Resource-Constrained Control Systems: from Feedback Scheduling to Event-driven Control," in 18th Mediterranean Conference on Control and Automation, Marrakech, Morocco, June 2010
[14] Seto, D., Lehoczky, J. P., Sha, L., Shin, K. G., "On task schedulability in real-time control systems," 17th IEEE Real-Time Systems Symposium, December 1996
[15] Eker, J., Hagander, P., Årzén, K.-E.: "A Feedback Scheduler for Real-time Control Tasks," *Control Engineering Practice*, Vol. 8 N. 12, December 2000
[16] Martí, P., Lin, C., Brandt, S., Velasco, M., Fuertes, J.M. "Optimal State Feedback Based Resource Allocation for Resource-Constrained Control Tasks", 25th IEEE Real-Time Systems Symposium, December 2004

References (2)

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

[17] Palopoli, L., Pinello, C., Bicchi, A., Sangiovanni-Vincentelli, A., "Maximizing the Stability Radius of a Set of Systems Under Real-Time Scheduling Constraints," *IEEE Transactions on Automatic Control*, Vol. 50, N. 11, pp. 1790-1795, Nov. 2005
[18] Henriksson, D. and Cervin, A., "Optimal On-line Sampling Period Assignment for Real-Time Control Tasks Based on Plant State Information", 44th IEEE Conference on Decision and Control and European Control Conference 2005, December 2005
[19] Martí, P., Lin, C., Brandt, S., Velasco, M., Fuertes, J.M., "Draco: Efficient resource management for resource-constrained control tasks," *IEEE Transactions on Computers*, vol. 58, no. 1, pp. 90–105, January 2009
[20] Cervin, A., Velasco, M., Martí, P., Camacho, A., "Optimal On-Line Sampling Period Assignment," Research Report ESAII-RR-09-04, December 2009
[21] Rehbinder, H., Sanfridson, M., "Integration of off-line scheduling and optimal control", 12th Euromicro Conference on Real-Time Systems, 2000
[22] Ben Gaid, M-M. B., Çela, A., Hamam, Y., Ionete, C., "Optimal Scheduling of Control Tasks with State Feedback Resource Allocation," 2006 American Control Conference, June 2006
[23] Gaid, M.-M. B., Çela, A., Hamam, Y., "Optimal real-time scheduling of control tasks with state feedback resource allocation," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 2, pp. 309 – 326, March 2009

References (3)

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

[24] Hristu-Varsakelis, D., "Feedback control systems as users of a shared network: communication sequences that guarantee stability", 40th IEEE Conference on Decision and Control, 2001
[25] Gaid, M.-M. B., Çela, A., Hamam, Y., "Optimal Integrated Control and Scheduling of Networked Control Systems with Communication Constraints: Application to a Car Suspension System," *IEEE Transactions on Control Systems Technology*, 14 (4), pp. 776-787, 2006
[26] Walsh, G.C., Hong, Y., "Scheduling of networked control systems," *IEEE Control Systems Magazine,* vol. 21, n. 1, pp. 57-65, Feb. 2001
[27] Anta, A., Tabuada, P., "On the benefits of relaxing the periodicity assumption for networked control systems over CAN,", in Real Time Systems Symposium, December 2009
[28] Martí, P., Camacho, A., Velasco, M., Gaid, M.-M. B., "Run-time allocation of optional control jobs to a set of CAN-based networked control systems," *IEEE Transactions on Industrial Informatics*, in submission, 2010
[29] Årzén, K.-E., "A Simple Event-Based PID Controller," in 14th World Congress of IFAC, January, 1999
[30] P. Tabuada, "Event-triggered real-time scheduling of stabilizing control tasks," *IEEE Trans. on Automatic Control*, 52(9), 2007, pp. 1680-1685

References (4)

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

[32] Heemels, W.P.M.H., Sandee, J.H., van den Bosch, P.P.J, "Analysis of event-driven controllers for linear systems," *International Journal of Control*, 81(4), 2008, pp. 571-590
[31] Nguyen, V.H., Suh, Y.S., "Networked Estimation with an Area-triggered Transmission Method," *Sensors*, Vol. 8, No. 2, pp. 897-909, 2008
[33] Heemels, W.P.M.H., Gorter, R.J.A., van Zijl, A., van den Bosch, P.P.J., Weiland, S., Hendrix, W.H.A., Vonder, M.R.,"Asynchronous measurement and control: a case study on motor synchronization," *Control Engineering Practice*, Vol. 7, pp. 1467–1482, 1999
[34] Zhao, Q.C., Zheng, D.Z., "Stable and Real-Time Scheduling of a Class of Hybrid Dynamic Systems," *Discrete Event Dynamic Systems,* Vol. 9, N. 1, pp. 45-64, 1999
[35] Henningsson, T., Johannesson, E., Cervin, A., "Sporadic Event-Based Control of First-Order Linear Stochastic Systems," *Automatica*, vol. 44, n. 11, pp. 2890–2895, Nov. 2008
[36] Anta, A., Tabuada, P., "Space-time scaling laws for self-triggered control," in Proceedings of the 47th IEEE Conference on Decision and Control, December 2008
[37] Wang, X., Lemmon, M., "Self-triggered Feedback Control Systems with Finite-Gain L2 Stability," *IEEE Transactions on Automatic Control*, v. 45, n. 3, pp. 452–467, March 2009

References (4)

Course scope
Basics on control
Timing issues
Feedback scheduling
Event-driven control
Conclusions
References

[38] Camacho, A., Martí, P., Velasco, M., Lozoya, C., Villá, R., Fuertes, J.M., Griful, E., "Self-Triggered Networked Control Systems: an Experimental Case Study," in IEEE 2010 International Conference on Industrial Technology, March 2010
[39] Velasco, M., Martí, P., Bini, E., "Equilibrium sampling sequences for event-driven controllers," in European Control Conference, 2009
[40] Velasco, M., Martí, P., Bini, E., "Control-driven Tasks: Modeling and Analysis," *in 29th IEEE Real-Time Systems Symposium*, 2008