Real-Time Systems

Introduction

Hermann Härtig

TECHNISCHE UNIVERSITÄT DRESDEN

WS 2017/18

first exercise

- in 2 weeks:
  26. Oktober 2016

- please register for mailing list !!

- some words on exams

Examples of Real-Time-Systems

- video decoding and other multimedia
- haptic (force feed) back devices
- planes, trains, cars, e-bikes
- control systems (see later slide)
- alarm systems
- automatic trading systems (?)
- ABS, ESP, … autonomous driving
- “Tactile Internet”
- robot: balancing (short term) and movement (longer term)
- breaks (recent ICE problem)
- microscope control

Other Terminology

Embedded Systems

- Computers as part of something else
- Sometimes Real-Time

Safety-Critical Systems

- **mission** (humans) in danger

General Purpose (Interactive) Systems

- smoothness of UI
- …

Overview of class

Basics:

* Definitions, Models and Terminology
* Time (Synchronisation etc)
* Time-Driven + Event-Driven: Basic Scheduling

Advanced Scheduling:

* Resource Sharing
* Multiprocessor
* probabilistic
* mixed criticality

More than just scheduling:

* Hardware
* Progr. Languages
* Communication
* Operating Systems

Course Material

Textbooks (available in library):

- [Kopetz]
  Hermann Kopetz
  Real-Time Systems (Kluwer)
- [Liu]
  Jane Liu
  Real-Time Systems (Prentice Hall)

In addition:

- papers provided in lectures

Real-Time Systems

Definition (strict)

Systems, whose correctness depends

- (not only) on the correct logical results of computations
- (but also) on meeting **all** deadlines.

Deadlines are dictated by the environment of the system.

Results, **deadlines** must be specified.

Definition (weaker)

Systems, whose quality depend

- (not only) on the logical results of computations
- (but also) on the time these results are produced.

Requested timing characteristics originate from the environment of the system.

Weakness Flavors

- Some deadlines are more important/critical than others (imprecise computations, mixed criticality, …)
- Occasional misses of deadlines are ok. (e.g., 3 in 10)
- The value of a result depends on the time it becomes available:
  - An imperfect result early may be better than a perfect result (too) late.
  - The more results can be obtained before a given deadline the better.
  - Explicit mapping of time to value

Weakness Flavors

Specification needed for:

- Results, deadlines AND
- “Importance” of certain deadlines OR
- How many deadlines per time period may be missed OR
- Mapping of time to values of results OR …

A saying by Doug Jensen (?):
Hard real-time systems are hard to build,
soft real-time systems even harder.

Hard, Firm, Soft

hard real-time systems

- deadlines are strict: missing has fatal consequences for the controlled object or humans
- must work under peak load

firm real-time systems

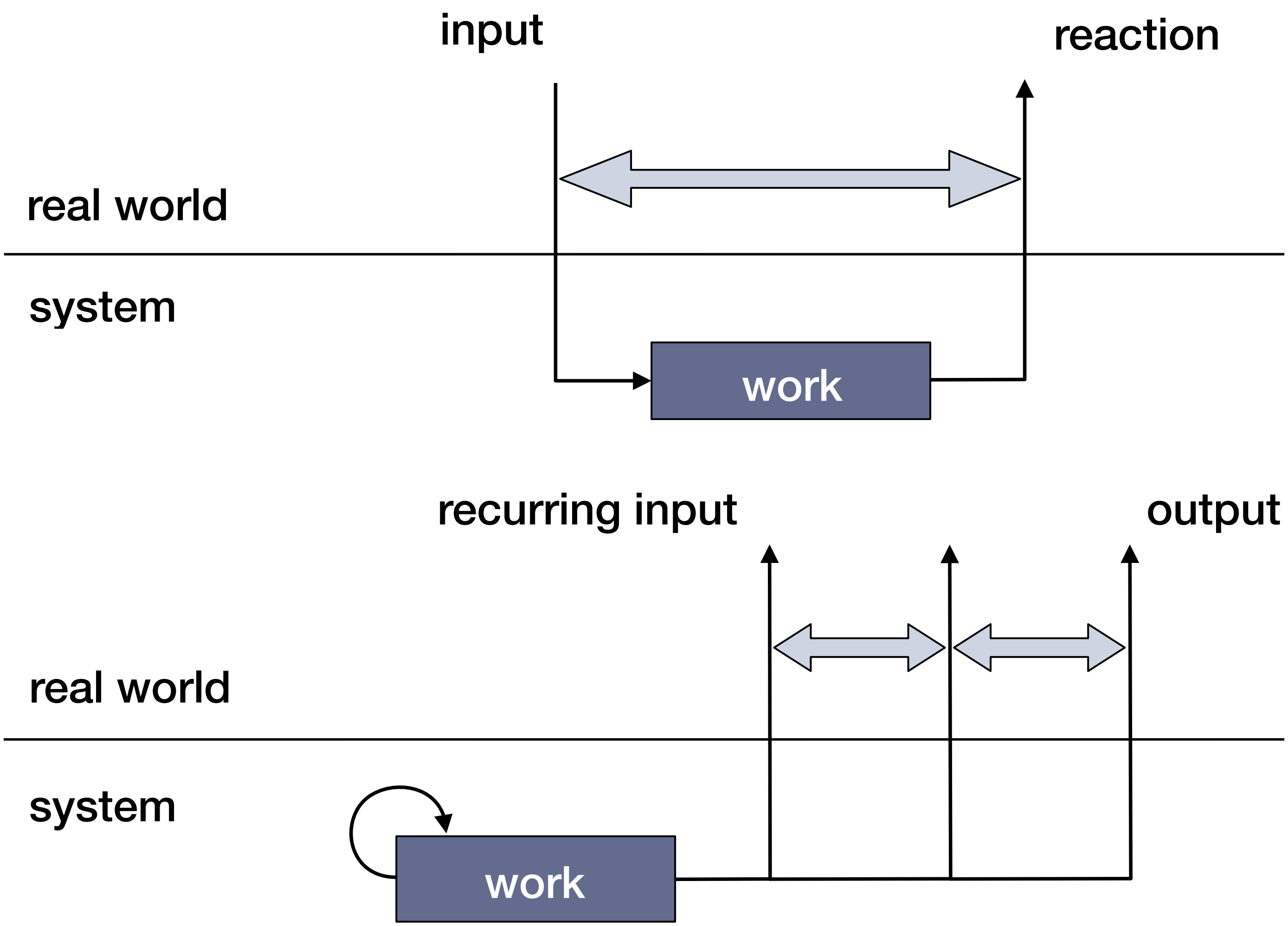
- deadlines are strict: late results have no benefit

soft real-time systems

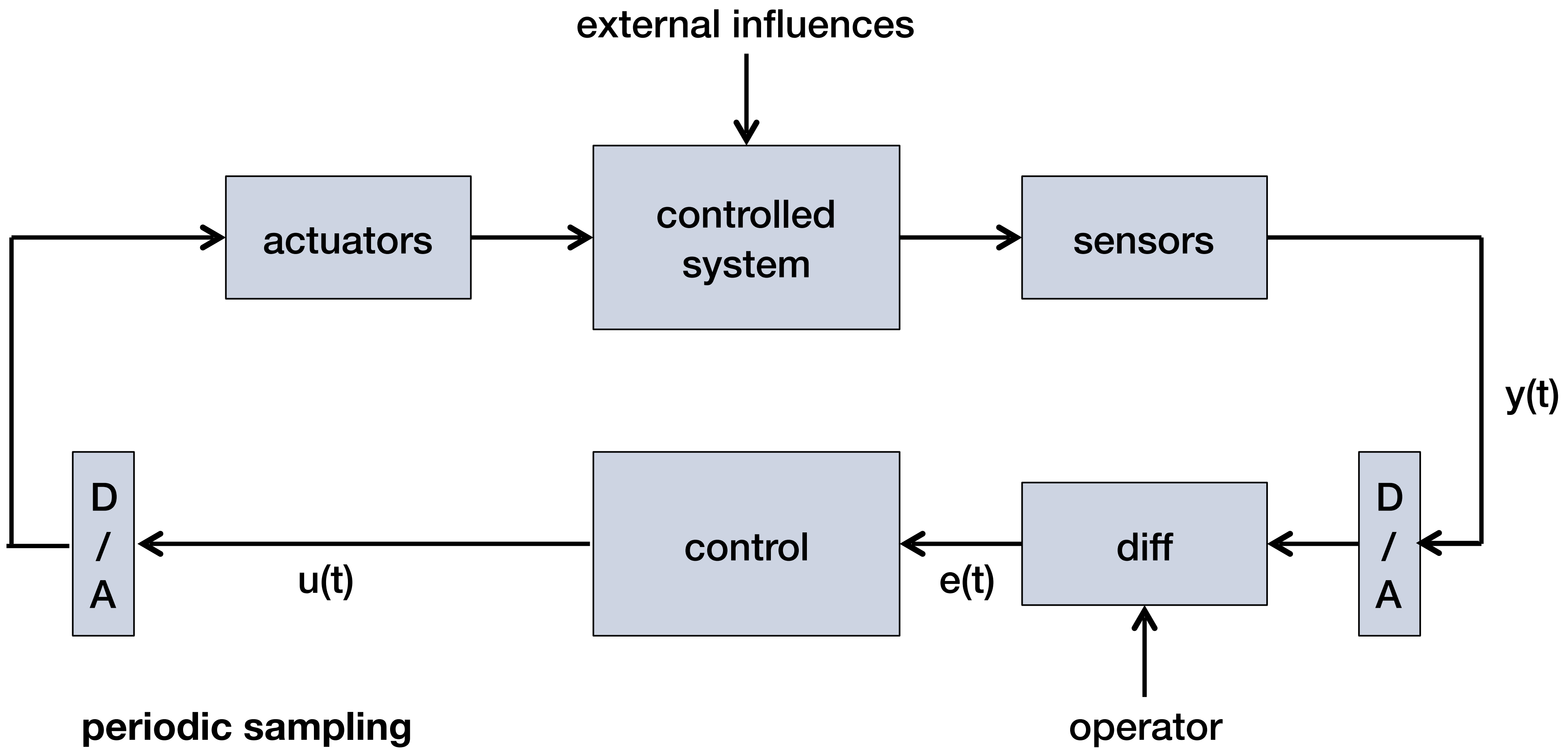
- deadlines should be met
- value of results decreases with time
- graceful degradation under peak load is acceptable

“**approximate** computing” (energy saving)

Real-World Context



Simple Digital Control System



“PID” controller

- Continuous formula:

$$u = k_p e + k_i \int_{\tau=0}^{t} e(\tau)d\tau + T_d \frac{de}{dt}$$

- Approximation by periodic sampling (rate T)

Integral via Simpson's Rule:

$$\frac{T}{3} * (e_{k-2} + 4e_{k-1} + e_k)$$

Differential:

$$\frac{e_k - e_{k-1}}{T}$$

Then:

$$u_k := u_{k-2} + a * e_k + b * e_{k-1} + c * e_{k-2}$$

With

$$a = k_p + \frac{k_i T}{3} + \frac{T_d}{T} \qquad b = \frac{4k_i T}{3} - \frac{T_d}{T} \qquad c = \frac{k_i T}{3}$$

Digital Controllers

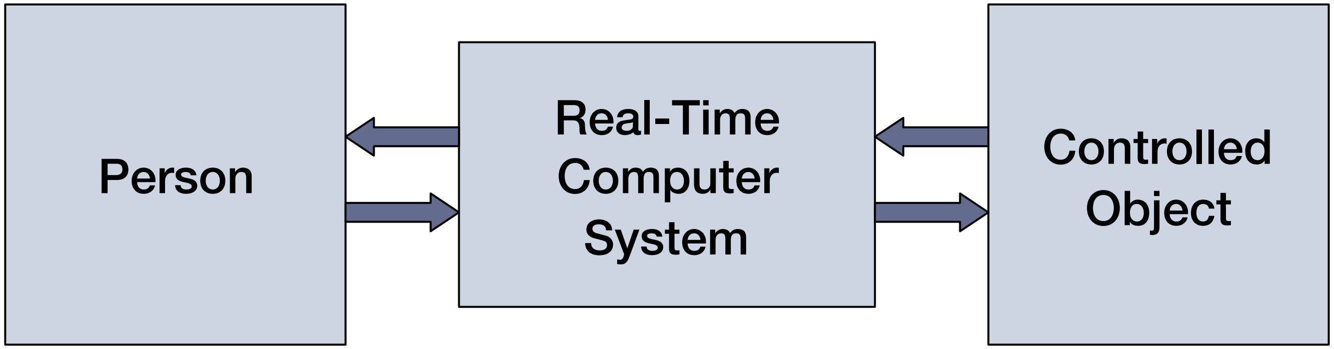
```
at every xx time units do
    read y and compute e
    u_k :=u_k-2 + a*e_k + b*e_k-1 + c*e_k-2
    write u
done
```

$$u_k := u_{k-2} + a * e_k + b * e_{k-1} + c * e_{k-2}$$

sample period xx depends on:

- reactivity of person (<100 ms)
- reactivity of controlled object

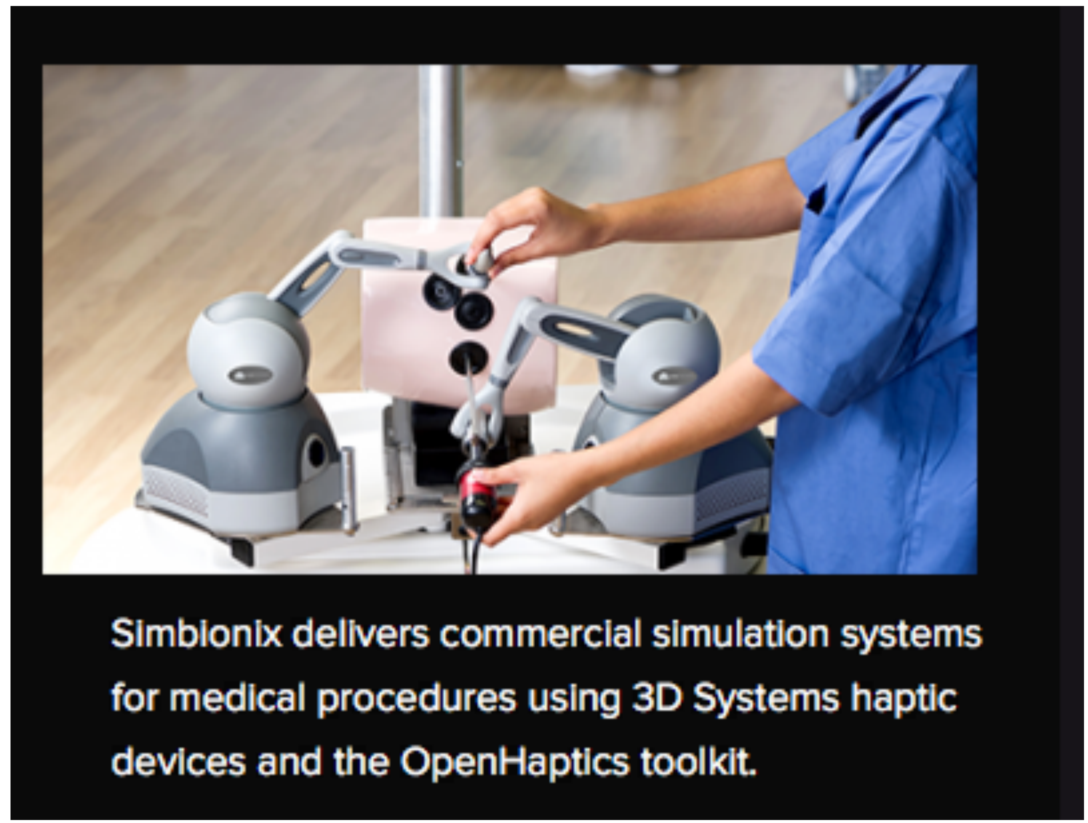
Interfaces



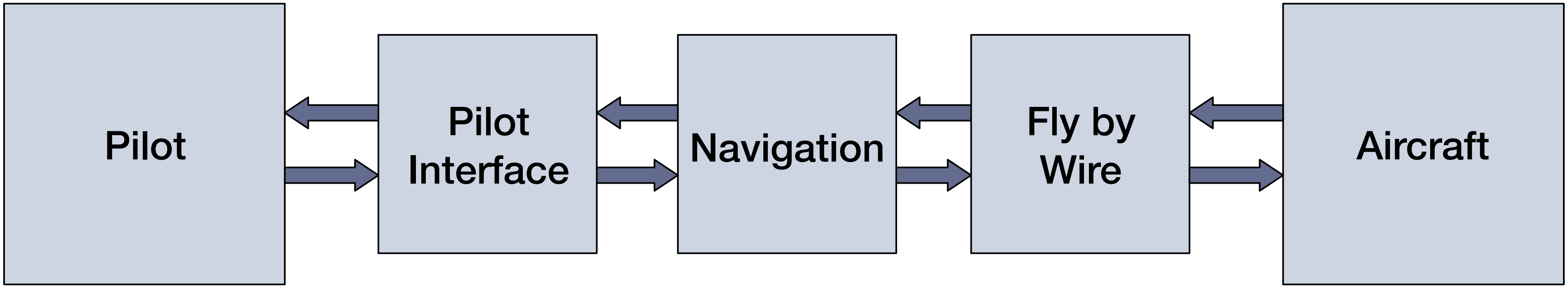
Timing requirements on both interfaces

Example for sources of timing requirements

source geomagic.com

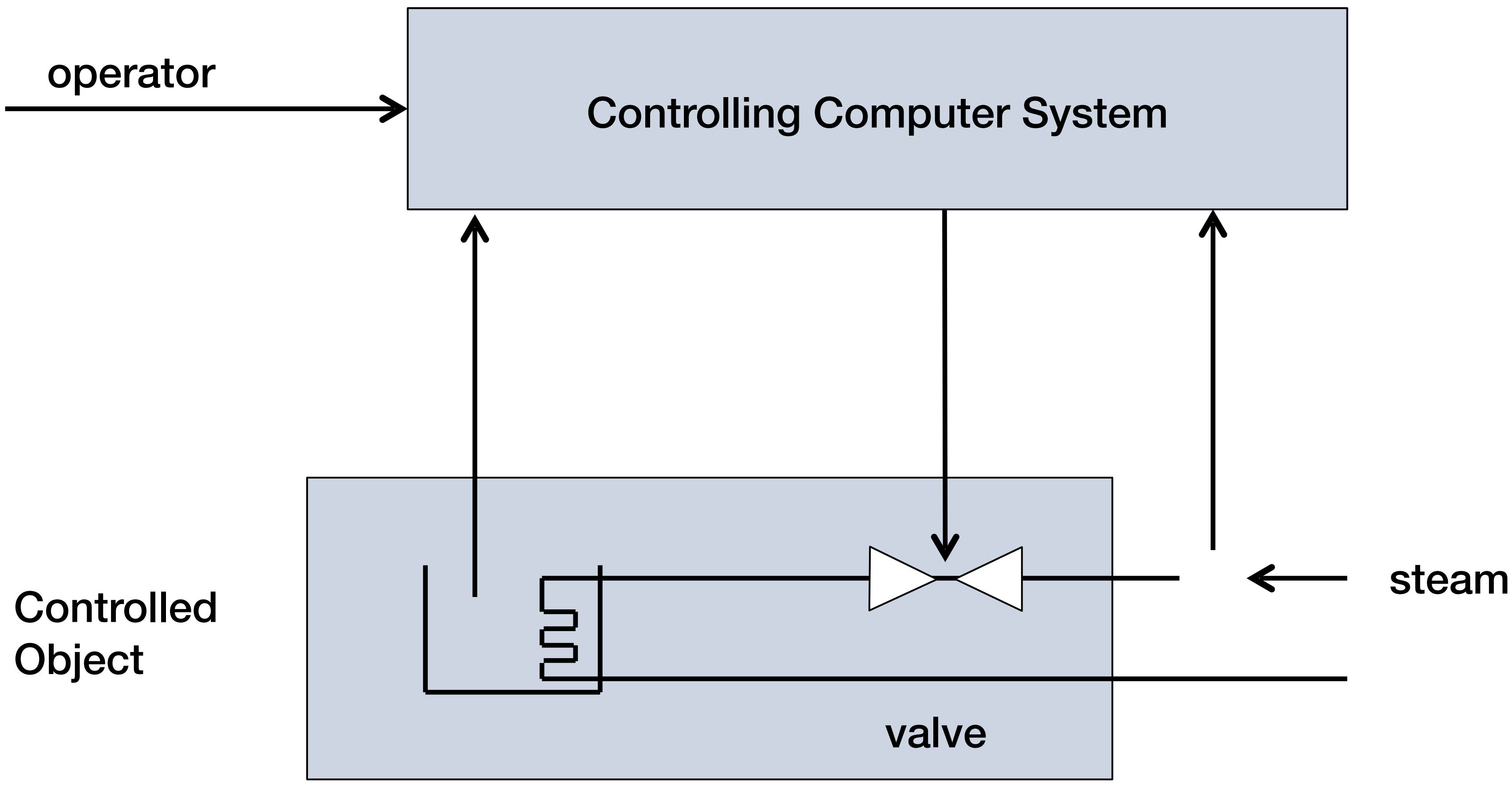


Layers of Control



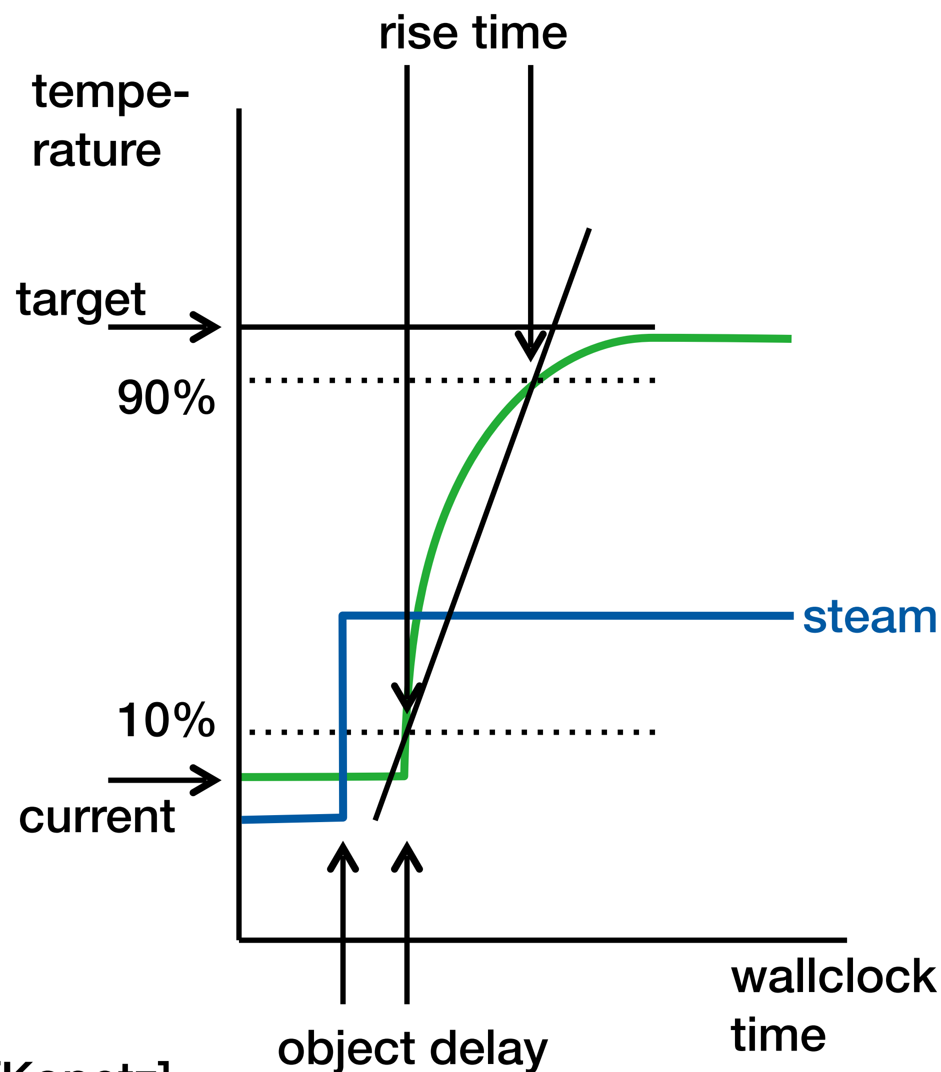
Multiple stages may induce different times

Control Example



Example following [Kopetz]

Times



[Kopetz]

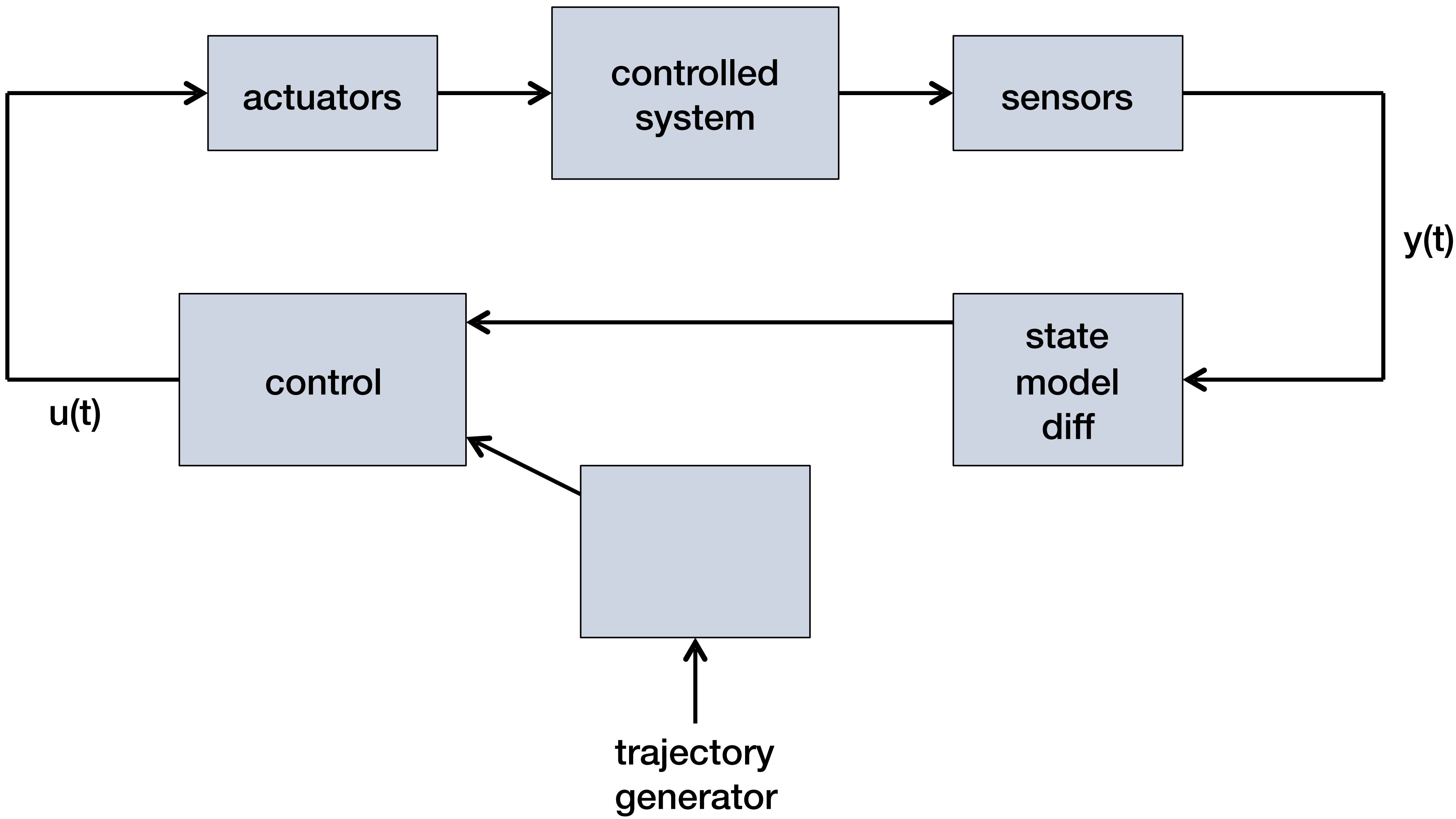
- rise time: 10% or other small neighborhood
- object delay: inertia of control process
- computation delay and jitter: < sample period
- deadtime: object delay + computation delay
- sampling period: rule of thumb < 1/10 to 1/20 rise time
- shorter sampling periods result in: smoother operation, less oscillation, more resources used

Complications of Simple Model: Internal State

```
state ...
at every xx timeunits do
  read ...
  compute
    output and new state
    Use: samples and current state
  write ...
done
```

- complete state of controlled object is not represented in sampled data, example: robot arm
- dangerous situations when internal and real-world state disagree

Stateful Control System



Modifications of Simple Model

- multiple sensors, actuators, and state variables
- different sampling rates: multi-rate controller
- often the larger are integer multiples of smaller rates: *harmonic rates*
- example: rotation, temperature (engine control)
- method (successive loop closure):
  - start with highest rate sensor
  - integrate it in system and consider it part of the controlled object
  - determine next rates (as multiples of fastest)

Summary

- real-time system: **time** matters
- hard, firm, soft …
- system context: contact with the real world
- control systems