Real-Time Systems

Introduction

Hermann Härtig

TECHNISCHE UNIVERSITÄT DRESDEN

WS 2018/19

Examples of Real-Time-Systems

- video decoding and other multimedia
- touch and 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” (5G)
- robot: balancing (short term) and movement (longer term)

Surrounding Terminology

Embedded Systems

- Computers as part of something else
- Can be Real-Time Systems

Cyber-Physical Systems

- Interaction with physical environment
- Often Real-Time Systems

Safety-Critical Systems

- Humans in danger, mission in danger

General Purpose Systems

- Interactive Systems
- Smoothness and Responsiveness of UI

Computer Science Areas

- Computer architecture
- Low Level I/O
- Communication
- Modeling techniques (Scheduling)
- Computer science theory
- Operations research
- Programming languages
- Parallel programming
- Fault tolerance
- Software engineering

Overview of Class

Basics

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

Advanced Scheduling

- Resource Sharing
- Multiprocessor Scheduling
- Probabilistic and Mixed Criticality Scheduling

More than just scheduling

- Hardware & Communication
- Programming Languages & Operating Systems

Course Material

Textbooks (available in library)

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

Additional 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 and deadlines must be specified.

Real-Time Systems

Definition (weaker)

Systems, whose quality depends

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

Required timing characteristics originate from the environment of the system.

Weakness Flavors

- Some deadlines are more important than others
  (Later: imprecise computations, mixed criticality)
- Occasional misses of deadlines are OK. (e.g., 3 in 10)
- Approximate values may be sufficient
  (approximate computing, energy saving)

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

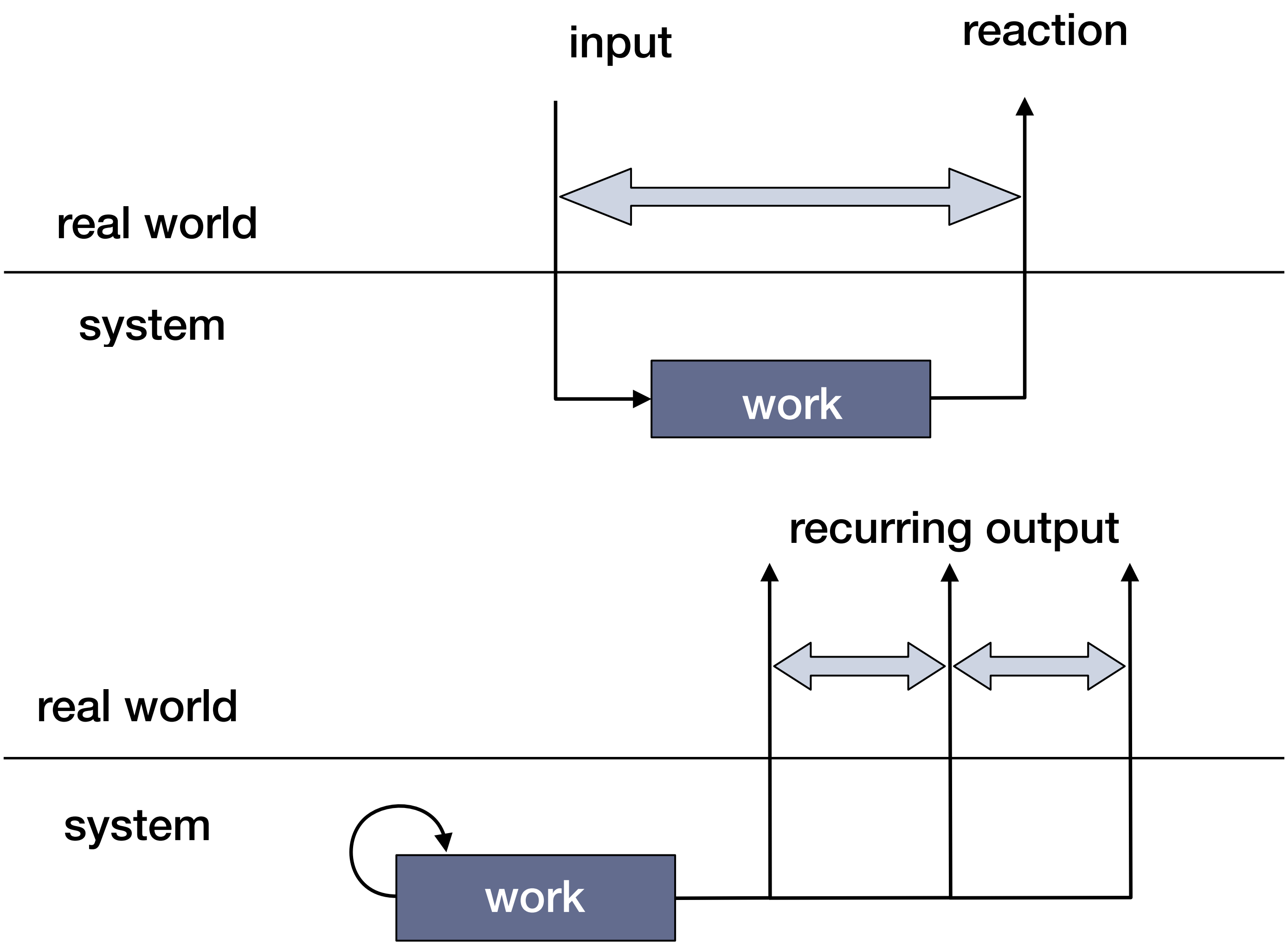
soft real-time systems

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

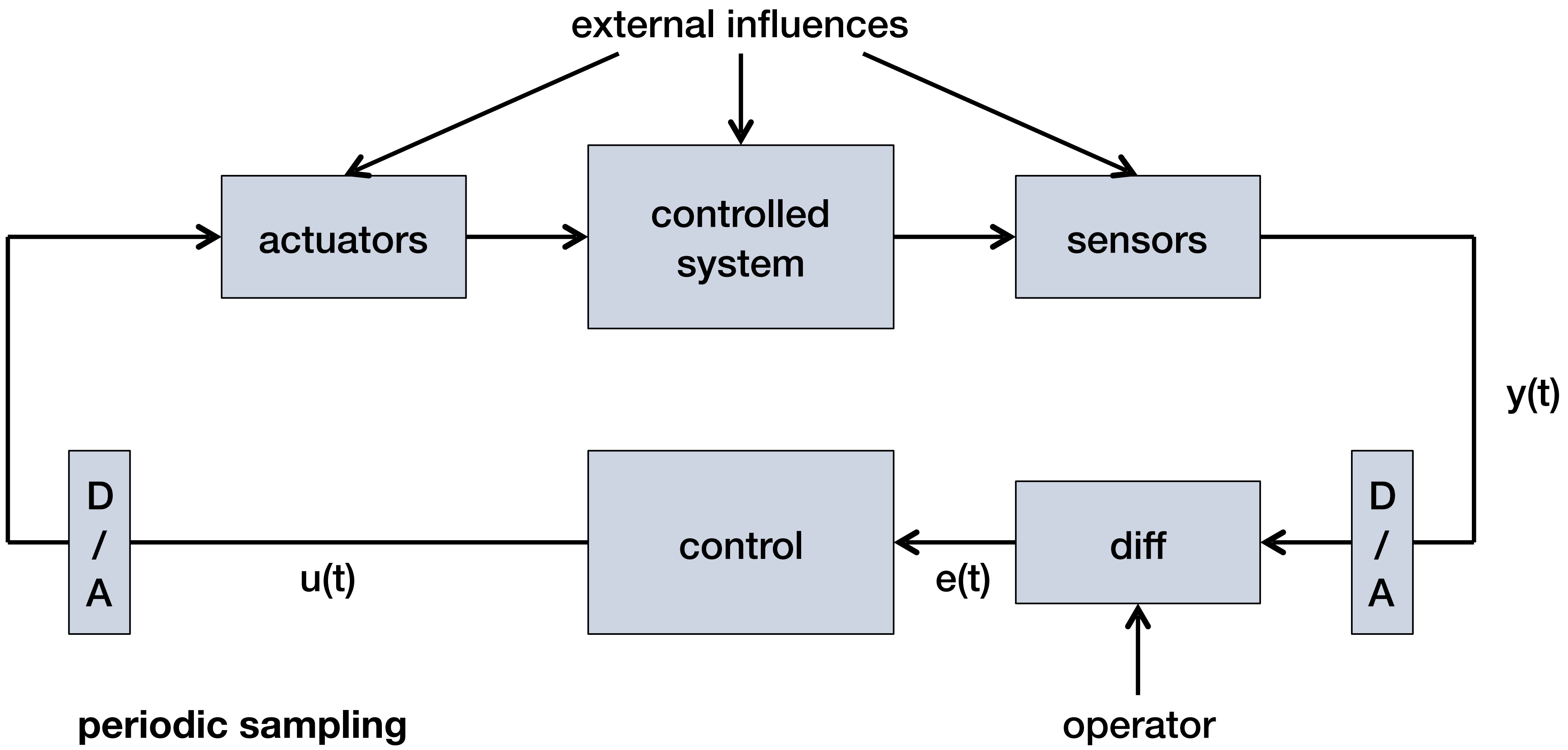
Design Process

- Model load and desired outcome
  - Load: resource requirements like CPU
  - Outcome: deadlines or other (hard/firm/soft) timing requirements
- Load depends on software (algorithm) and hardware (CPU speed)
  - Compiler maps high-level code to machine code
  - May involve algorithmic flexibility (approximation)
  - May involve other (shared) resources
- Scheduling
  - Test for feasibility (mathematical analysis)
  - allocate resources (OS scheduling API)
  - enforce allocations

Context


input
reaction
real world
system
work
recurring output
real world
system
work

Simple Digital Control System



PID Controller

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

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: $$u_k = u_{k-2} + ae_k + be_{k-1} + ce_{k-2}$$

With $$a = k_p + \frac{k_i T}{3} + \frac{k_d}{T}, \quad b = \frac{4k_i T}{3} - \frac{k_d}{T}, \quad c = \frac{k_i T}{3}$$

Digital Controllers

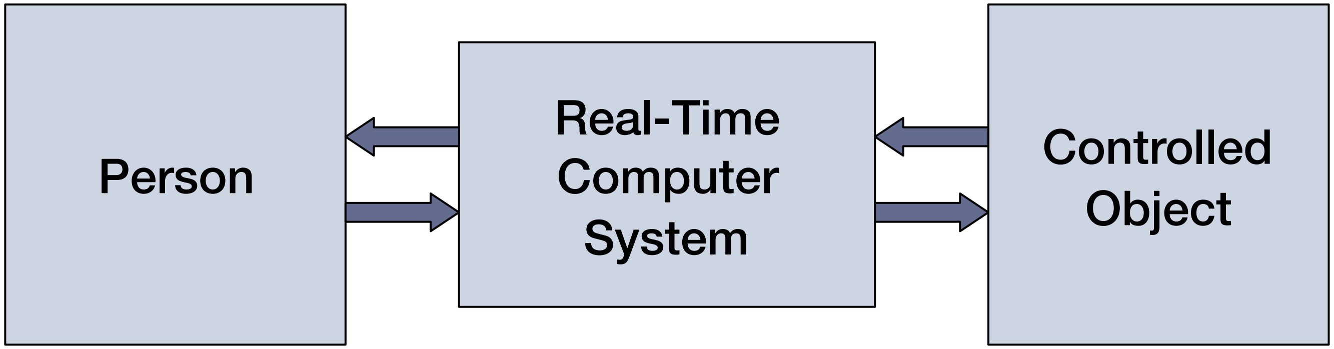
```
at every period time units do
    read y and compute e
    u_k := u_k-2 + a*e_k + b*e_k-1 + c*e_k-2
    write u
done
```

sample period depends on:

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

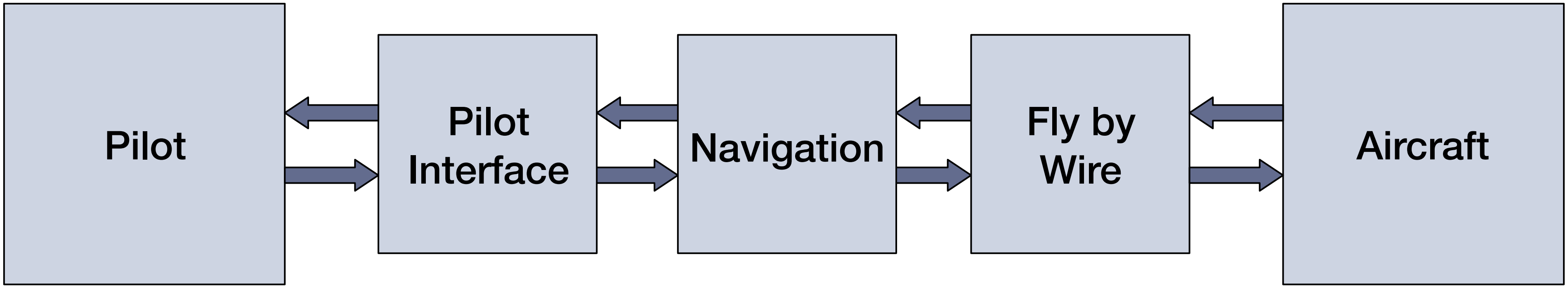
Interfaces

Timing requirements on both interfaces

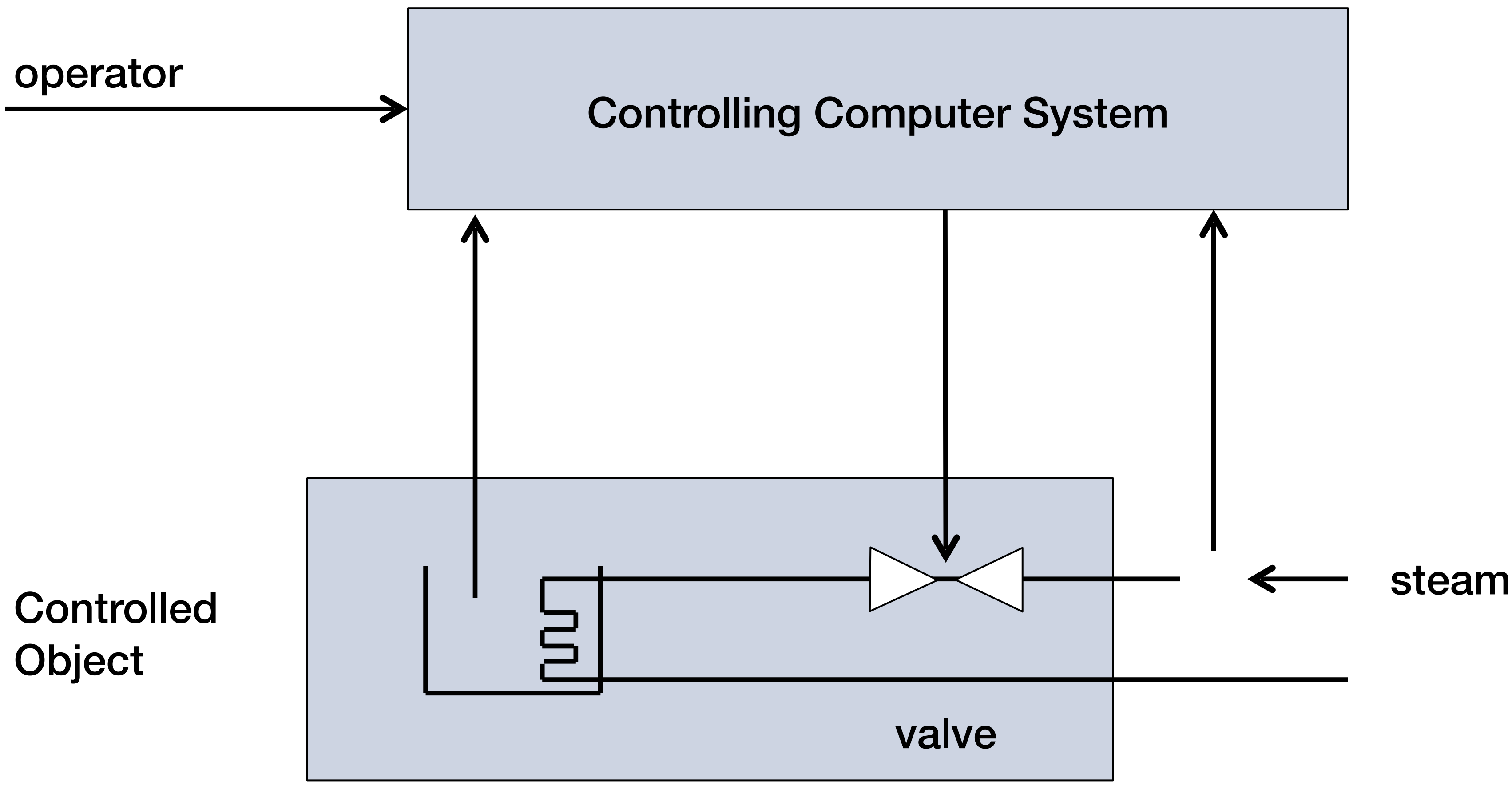


Layers of Control

Multiple stages may induce different times



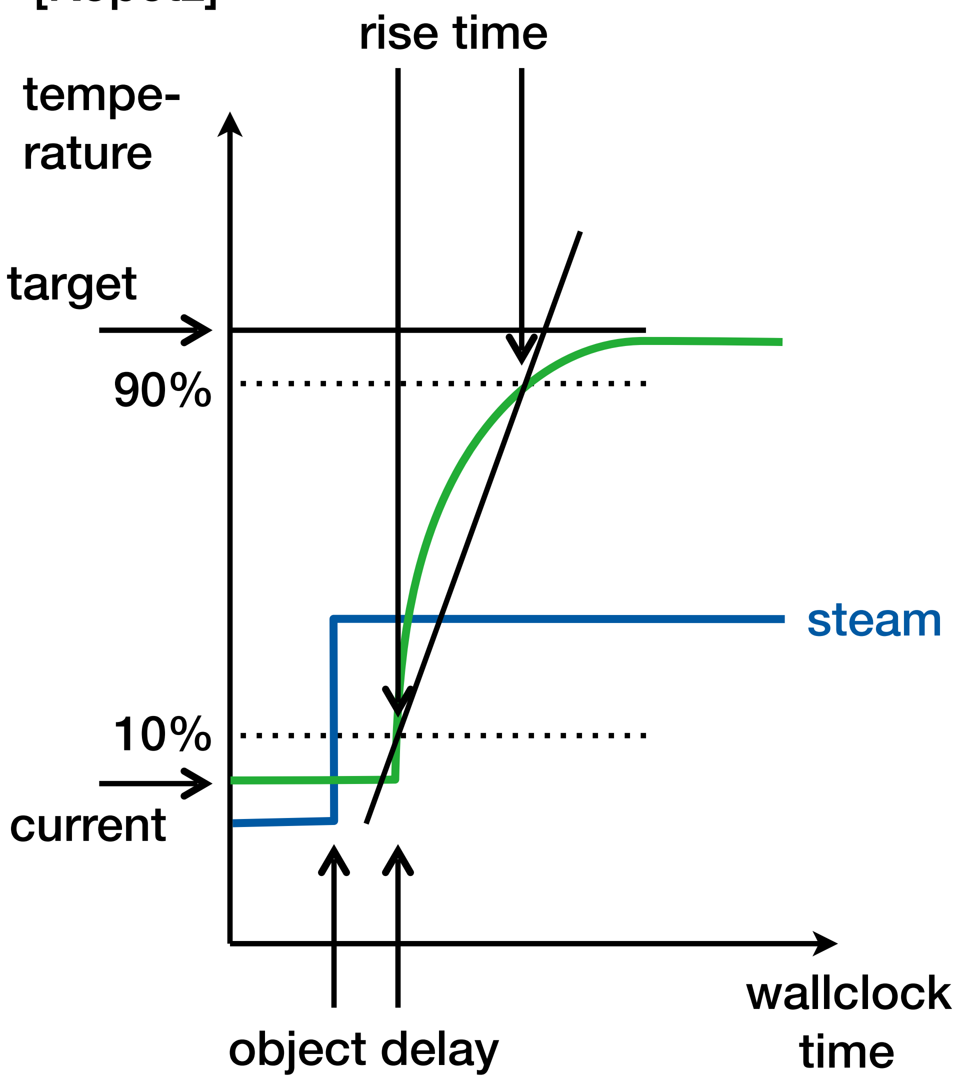
Control Example



Example following [Kopetz]

Times

[Kopetz]



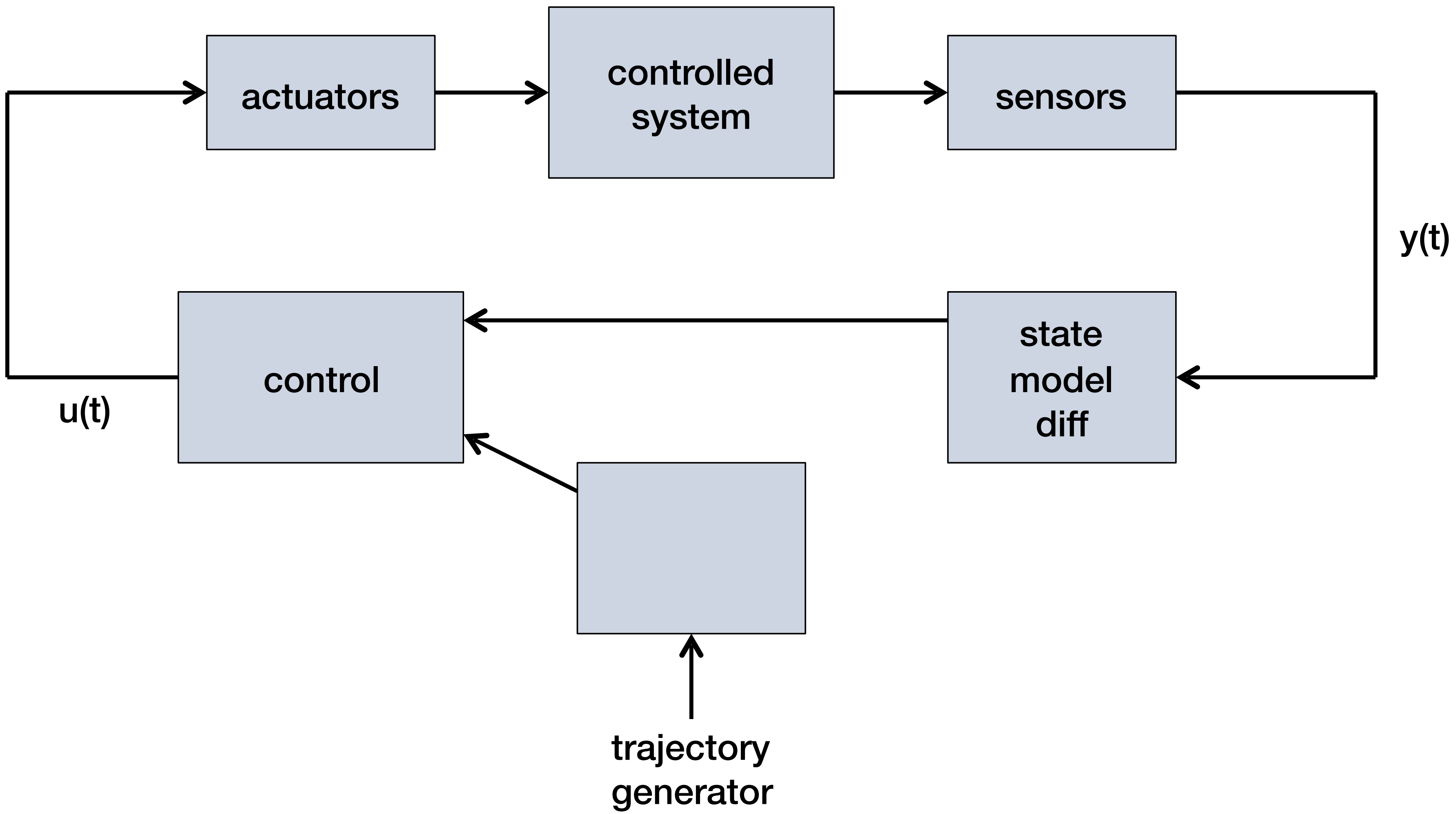
- rise time: 10% or other small neighborhood
- object delay: inertia of control process
- computation delay and jitter: < sample period
- deadtime: object delay + computation delay
- sampling period: rule of thumb < 1/10 to 1/20 rise time
- shorter sampling periods result in: smoother operation, less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

Complications of Simple Model: Internal State

```
initialize state
at every period timeunits do
    read input
    compute
        output and new state
        Use: samples and current state
    write output
done
```

- Complete state of controlled object is not represented in sampled data, example: robot arm
- Keep internal copy (“digital twin”) of believed state
- 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, and other delineating attributes
- system context: contact with the real world
- control systems