EL2450 Hybrid and Embedded Control

Lecture 14: Summary

- Course summary
- What's on the exam?

Course Summary

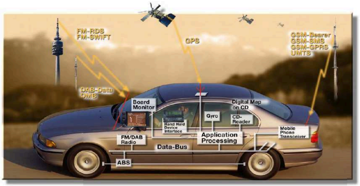
- **Time-triggered control**
  - Models of sampled systems: sampling, ZOH, delays, poles, zeros
  - Analysis of sampled control systems: stability, observers
  - Computer implementation of controllers: approximation of continuous-time designs, computer code, sampling time
  - Implementation platform aspects: modeling and compensation for jitter, delay, quantization
- **Event-triggered control**
  - Networked control systems: real-time systems, quantization, event-based control, packet losses, time-delays

- Real-time scheduling: periodic and aperiodic tasks, schedulability analysis
- Models of computations: discrete-event systems and transition systems

- **Hybrid control**
  - Time- and event-triggered systems: hybrid dynamics
  - Models of hybrid systems: hybrid automata, solutions, Zeno
  - Control of hybrid systems: stability, supervisory control
  - Verification of hybrid systems: reachability, verification

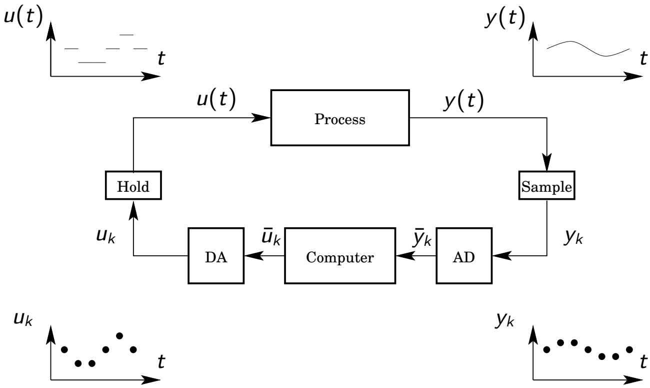
Hybrid and Embedded Control Systems

- Computational systems (but not a computer)
- Integrated with physical world via sensors and actuators
- Reactive (at the speed of the environment)
- Heterogeneous (mixed hw/sw architectures)
- Networked (share data and resources)



Sampled Systems

Consider mapping from u_k to y_k:



For all $t \in [t_k, t_{k+1}]$

$$
\begin{aligned}
x(t) &= e^{A(t-t_k)}x(t_k) + \int_{t_k}^{t} e^{A(t-\tau)}Bu(\tau)d\tau \\
&= e^{A(t-t_k)}x(t_k) + \int_{t_k}^{t} e^{A(t-\tau)}d\tau Bu(t_k) \\
&= e^{A(t-t_k)}x(t_k) + \int_{0}^{t-t_k} e^{As}dsBu(t_k),
\end{aligned}
$$

$$
\begin{aligned}
x(t_{k+1}) &= \Phi(t_{k+1}, t_k)x(t_k) + \Gamma(t_{k+1}, t_k)u(t_k) \\
y(t_k) &= Cx(t_k) + Du(t_k) \\
\Phi(t_{k+1}, t_k) &= e^{A(t_{k+1}-t_k)}, \quad \Gamma(t_{k+1}, t_k) = \int_0^{t_{k+1}-t_k} e^{As}dsB
\end{aligned}
$$

- Time-varying linear system!
- When does it become time-invariant?

Analysis of sampled systems

A continuous function $V : \mathbb{R}^n \to \mathbb{R}$ is a *Lyapunov function* for $x(k+1) = f(x(k)), f(0) = 0$ if

1. $V(0) = 0$ and $V(x) > 0$, $\forall x \neq 0$
2. $\Delta V(x) := V(f(x)) - V(x) \leq 0$, $\forall x \neq 0$

Linear Systems $V(x) = x^T P x$, P positive definite, is a Lyapunov function for $x(k+1) = \Phi x(k)$, if (and only if)

$$\Phi^T P \Phi - P = -Q, \qquad Q \text{ positive semidefinite}$$

because
$\Delta V(x) = V(\Phi x) - V(x) = x^T(\Phi^T P \Phi - P)x = -x^T Q x \leq 0.$

Observers, Feedback

What is a reduced-order observer?
The regular observer has a unit delay from y to $\hat{x}$, which can be avoided by considering

$$\begin{aligned}\hat{x}(k+1|k+1) = \Phi\hat{x}(k|k) + \Gamma u(k) \\ + K[y(k+1) - C(\Phi\hat{x}(k|k) + \Gamma u(k))]\end{aligned}$$

where $y(k+1)$ is the current measurement.
The reconstruction error $\tilde{x} = x - \hat{x}$ fulfills

$$\tilde{x}(k+1|k+1) = (I - KC)\Phi\tilde{x}(k|k)$$

so K should be chosen such that $|\lambda_i[(I - KC)\Phi]| < 1$.

Computer implementation of controllers

Discretized PID control

$$u(kh) = P(kh) + I(kh) + D(kh)$$

where

$$P(kh) = K\left[br(kh) - y(kh)\right]$$

$$I(kh+h) = I(kh) + \frac{Kh}{T_i}e(kh)$$

after forward approximation and

$$D(kh) = \frac{T_d}{T_d + Nh}D(kh-h) - \frac{KT_dN}{T_d+Nh}[y(kh) - y(kh-h)]$$

after backward approximation.

Computer implementation of controllers

How implement sampled-data controller?

```
nexttime = getCurrentTime();
while (true) {
  AD_conversion();
  calculateOutput();
  DA_conversion();
  updateState();
  nexttime = nexttime + h;
  sleepUntil(nexttime);
}
```

Implementation platform aspects

Compensating delays in output feedback

$$\dot{x}(t) = Ax(t) + Bu(t), \qquad y(t) = Cx(t)$$

Sensor node sends $y(kh)$ to control node with transmission delay $\tau_k = \tau(kh) < h$. Estimation and control performed in following order:

$$\begin{aligned}
\bar{x}(kh) &= \hat{x}(kh) + K[y(kh) - C\bar{x}(kh)] \\
\bar{x}(kh+\tau_k) &= e^{A\tau_k}\bar{x}(kh) + \int_{kh}^{kh+\tau_k} e^{A(kh+\tau_k-s)}Bu(s)ds \\
u(kh+\tau_k) &= -L\bar{x}(kh+\tau_k) \\
\hat{x}(kh+h) &= e^{A(h-\tau_k)}\bar{x}(kh+\tau_k) + \int_{kh+\tau_k}^{kh+h} e^{A(kh+h-s)}Bu(s)ds
\end{aligned}$$

Implementation platform aspects

State feedback stability under quantization.

- Plant: $\dot{x} = Ax + Bu$. Assume that $u = Kx$ is a stabilizing controller.
- Controller with quantization: $u = Kq(x)$. ("certainty equivalence controller").

Theorem Suppose that the closed-loop system without quantization is asymptotically stable. Then there exist sufficiently small δ_u, δ_l for which the quantized closed-loop system (i)is asymptotically stable, for the case of logarithmic quantizers, (ii) converges to a region around the equilibrium point, whose size depends on δ_u, for the case of uniform quantizers.

Event-based control

- Event-based control takes into account state or output feedback in order to sample as less as possible in an aperiodic fashion.
- Controller with aperiodic sampling and ZOH: $u(t) = Kx(t_k), t \in [t_k, t_{k+1})$.
- Question: how can we choose the sequence $\{t_k\}, k = 0, \ldots$ so that the stability properties are maintained and the sampling is as rare as possible?
- Answer: this is held by using *feedback* information from the plant in the sampling process!

Event-based control

Theorem The event-based rule

$$t_{k+1} = \inf_t \{t > t_k \mid ||e(t)|| = \sigma ||x(t)||\}$$

maintains the asymptotic stability of the original system for appropriate choice of $\sigma > 0$. Furthermore, the inter-sampling times $t_{k+1} - t_k$ are lower bounded by a positive constant τ^* for all k,$\sigma > 0$.

Real Time Scheduling

What scheduling algorithms should we know?

- Earliest deadline first
  - Dynamic priorities: task J_i with shortest time to deadline d_i is executed
- Rate monotonic scheduling
  - Fixed priorities: $T_i < T_j$ implies that J_i gets higher priority than J_j
- Deadline monotonic scheduling
  - Fixed priorities: $D_i < D_j$ implies that J_i gets higher priority than J_j
- Scheduling with polling server (mixing aperiodic and period tasks)

Discrete-Event Systems

A deterministic automaton A is a five-tuple

$$A = (Q, E, \delta, q_0, Q_m)$$

- Q is a finite set of states
- E is a finite set of events
- $\delta : Q \times E \mapsto Q$ is a transition function
- $q_0 \in Q$ is the initial state
- $Q_m \subseteq Q$ is the marked (or final) states

$q' = \delta(q, e)$ means that there is a transition labeled by event e from state q to q'.

Transition Systems

Safety For a transition system $T = (S, \Sigma, \rightarrow)$ with initial state in S_0, let $B \subset S$ denote a "bad" set, i.e., a set of states that we don't want the system to enter. T is **safe** if

$$\mathrm{Reach}(S_0) \cap B = \emptyset$$

Remark

- B encodes the property to verify
- Verification is about verifying that the system fulfills its specification, i.e., $\mathrm{Reach}(S_0) \cap B = \emptyset$

Transition Systems

Reachability Algorithm to compute $\mathrm{Reach}(S_0)$

$\mathrm{Reach}_{-1} := \emptyset$, $\mathrm{Reach}_0 := S_0$, $i := 0$

`while` $\mathrm{Reach}_i \neq \mathrm{Reach}_{i-1}$ `do`

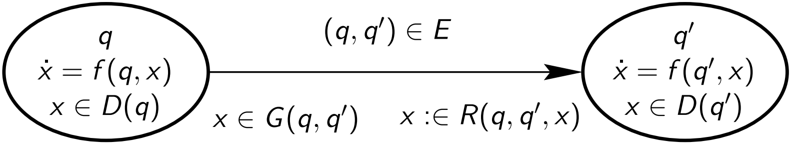
  $\mathrm{Reach}_{i+1} := \mathrm{Reach}_i \cup \mathrm{Post}(\mathrm{Reach}_i)$

  $i := i + 1$

`end`

- If the algorithm terminates, then $\mathrm{Reach}(S_0) := \mathrm{Reach}_i$
- If the state space is finite, the algorithm terminates in a finite number of steps
- The algorithm does not necessarily terminate for general transition systems
- $(q, x) \xrightarrow{e} (q_p, x_p)$ is simple, but $(q, x) \xrightarrow{t} (q_p, x_p)$ in general hard

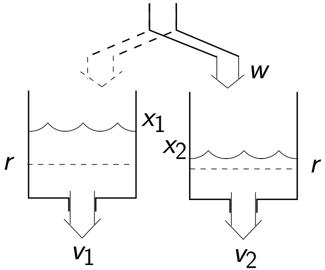
Hybrid Automata $H = (Q, X, \text{Init}, f, D, E, G, R)$

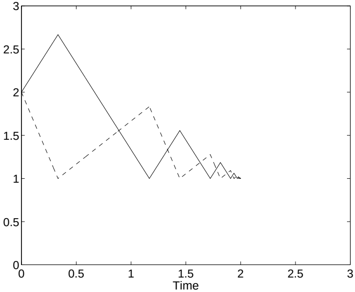


- Q discrete state space and X continuous state space
- $\text{Init} \subseteq Q \times X$ initial states
- $f : Q \times X \to X$ vector fields
- $D : Q \to 2^X$ domains
- $E \subset Q \times Q$ edges
- $G : E \to 2^X$ guards
- $R : E \times X \to 2^X$ resets

Zeno Solution of Hybrid Automaton

A solution $\chi = (\tau, q, x)$ is Zeno if $\tau_\infty = \sum_{i=1}^{\infty}(\tau_i' - \tau_i) < \infty$
Example—Water tank system: If $\max\{v_1, v_2\} < w < v_1 + v_2$ then
$\tau_\infty = (x_1(0) + x_2(0) - 2r)/(v_1 + v_2 - w) < \infty$





Execution is not defined for $t > \tau_\infty$

Control of Hybrid Systems

Suppose $x^* = 0$ is an equilibrium of each mode $q = 1, \ldots, m$ of the switched system

$$\dot{x} = f_q(x), \qquad x \in \Omega_q$$

If there exist functions $V_1, \ldots, V_m$ such that

$$\begin{aligned} V_q(0) = 0, \quad V_q(x) > 0, \;\; \forall x \in \mathbb{R}^n \setminus \{0\} \\ \dot{V}_q(x(t)) \leq 0, \quad \text{whenever } x(t) \in \Omega_q \end{aligned}$$

and the sequences $\{V_q(x(\tau_{i_q}))\}$, $q = 1, \ldots, m$ are non-increasing, where τ_{i_q} are the time instances when mode q becomes active, then x^* is stable.

Verification of Hybrid Systems

If $\sim$ is a simulation relation from T to T' and $\sim' = \{(s', s) : (s, s') \in \sim\}$ is a simulation relations from T' to T, then $\sim$ is a **bisimulation relation**.

- The existence of a bisimulation relation between two transition systems indicates that they are equivalent in some sense
- We say that T and T' are bisimular

Reachability for Bisimilar Transition Systems

Given $T = (S, \Sigma, \rightarrow, S_0, S_F)$, the question whether

$$\text{Reach}(S_0) \cap S_F = \emptyset$$

in T is equivalent to the question whether

$$\text{Reach}(\hat{S_0}) \cap \hat{S_F} = \emptyset$$

in the bisimulation quotient transition system $\hat{T}$.

Hybrid Systems Bisimulations

Timed automata are a subclass of hybrid automata
$TA = (Q, X, \mathrm{Init}, f, D, E, G, R)$

- $Q = \{q_1, \ldots, q_m\}$, $X = \mathbb{R}_+^n$, $\mathrm{Init} \subseteq Q \times \{0\}^n$
- $f(q, x) = (1, \ldots, 1)$: "clock" dynamics
- $E \subseteq Q \times Q$
- $D(q), G(e)$ are *rectangular* sets, i.e., they are finite boolean combinations of constraints of the form $x_i \bowtie a_i$, where $\bowtie \in \{<, \leq, =, \geq, >\}$, and a_i is a positive integer.
- $R(e, x) = \{y\}$, where $y_i = 0$ or $y_i = x_i$ for all $1 \leq i \leq n$.

Why do we do Hybrid Systems?

- **Embedded systems are hybrid**
  - Real-time software interacting with physical processes, a huge pool of CS tools to work with
- **Abstractions in design lead to hybridness**
  - Time-scale separation, timing abstractions, hierarchical modeling, state-space minimization
- **Control strategies are hybrid**
  - Optimal control, model predictive control, control constraints, supervisory control
- **Nature is hybrid**
  - Relays, impact mechanics, state constraints
- **Control Objectives are hybrid**
  - Safety, invariance, formal languages based objectives

Doing Master Thesis Project in Automatic Control

- Theory <u>and</u> practice
- Cross-disciplinary
- The research edge
- Collaboration with leading industry and universities
- Get insight in research and development

Hints:

- The topic and the results of your thesis are up to you
- Discuss with faculty, PhD and other MSc students
- Check old online theses at DIVA

Master Thesis Project Examples in Automatic Control

- Distributed Intelligent Sampling and Control
- Distributed Multi-Agent Coordination (UAVs, AUVs, multi-robot systems, sensor networks)
- Formal methods for distributed control
- Collaborative manipulation, Human-Robot interaction
- Finite-state representation of multi-agent control systems
- Contact person: Dimos Dimarogonas (dimos@kth.se)
- . . .

Doing PhD in Automatic Control

- Intellectual stimuli
- Get paid for studying
- International collaborations and travel
- Competitive
- World-wide job market

- Research (60%), courses (30%), teaching (10%), fun (100%)
- 4-5 yr's to PhD (Lic after 2-3 yr's)

Last frame

- Thank you for attending this course!