5. Hybride Simulation

Bisher strenge Trennung in

- kontinuierliche Systeme
  Darstellung kontinuierlicher Zustandsänderungen in
  - vielen naturwissenschaftlichen Anwendungen (Physik, Chemie, ...)
  - populationsbasierten Systemen mit sehr großer Population (Straßenverkehr, Bevölkerungsentwicklung)
- diskrete Systeme
  Darstellung im wesentlichen ereignisorientierter Zustandsänderungen in
  - vielen technischen Systemen (Computer, Kommunikation, Fertigung, ...)
  - diskretisierten kontinuierlichen Systemen

Einzelne Simulationsansätze

- etabliert und in vielen Systemen realisiert
- aber grundsätzlich verschieden
  - numerische Analyse von Differentialgleichungen im kontinuierlichen Fall
  - stochastische ereignisdiskrete Simulation im diskreten Fall

Kombination von kontinuierlicher und diskreter Simulation:
Wird diese benötigt?
Wie simuliert man?
Wie modelliert und spezifiziert man?
$\Rightarrow$ Hybride Simulation: Inhalt dieses Kapitels

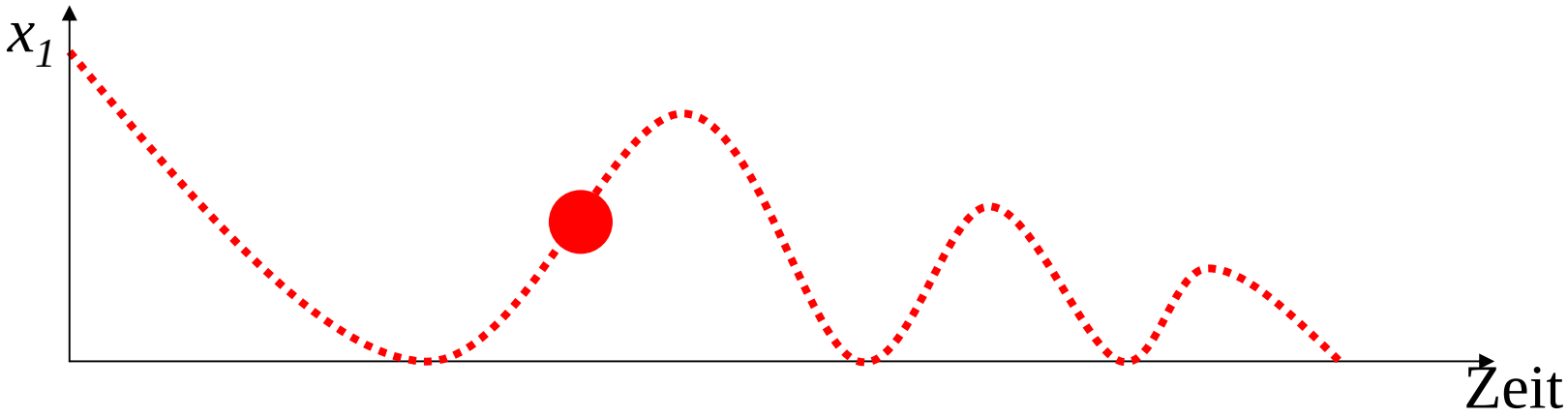
5.1 Beispiele für hybride Systeme
5.2 Analyse kontinuierlicher Systeme (eine kurze Zusammenfassung)
5.3 Hybride Simulationsansätze
5.4 Spezifikation hybrider Systeme

5.1 Beispiele für hybride Systeme

Im wesentlichen zwei Klassen von Beispielen:

- Kontinuierliche Modelle, bei denen Diskontinuitäten auftreten
  - Überlast in elektronischen Schaltungen
  - Kollision von Körpern
  - sehr steife Differentialgleichungssysteme
- Diskrete Steuerung kontinuierlicher Prozesse
  - digitale Steuerung
    - generiert zeitgesteuerte Ereignisse
    - reagiert auf Zustandsänderungen im kontinuierlichen Teil
  - kontinuierlicher Prozesse
    - schreitet fort
    - wird durch Steuerungsimpulse beeinflusst u.U. auch durch abrupte Zustandsänderungen

Aufspringender Ball:



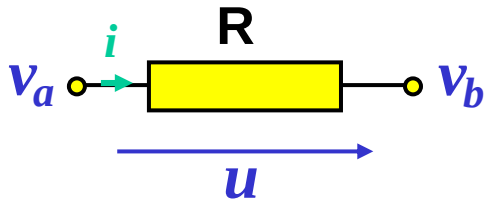
Beschreibung:

- x_1 sei die vertikale Position des Balls ($x_1 \geq 0$)
  uns interessiert nur die Höhe, nicht die horizontale Position
- x_2 Geschwindigkeit des Balls, also $\dot{x}_1 = x_2$
- Geschwindigkeitsänderung durch Erdbeschleunigung vorgegeben
  also $\dot{x}_2 = -g$
- Richtungsänderung beim Aufprall des Balls
  falls $x_1 = 0$ dann $x_2 = -c \cdot x_2$ (wobei $0 \leq c \leq 1$)

Elektronische Schaltungen mit Dioden

Elemente in der Schaltung und zugehörige Gleichungen/Differentialgl.

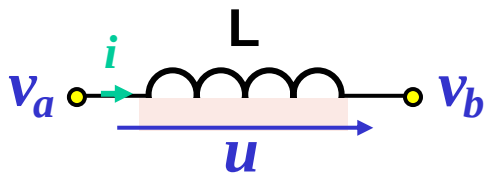
Widerstand



$u = v_a - v_b$

$u = R \cdot i$

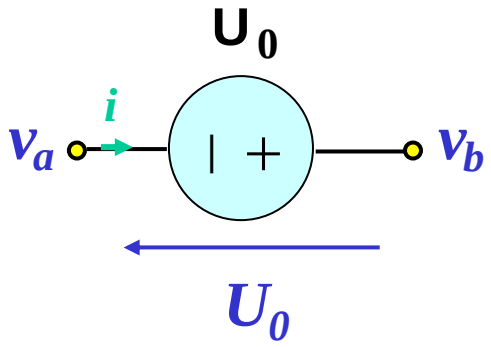
Spule



$u = v_a - v_b$

$u = L \cdot \frac{di}{dt}$

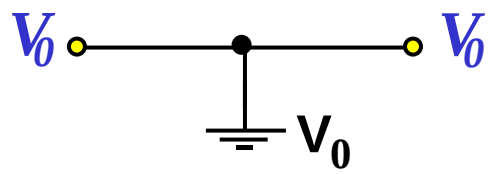
Spannungsquelle



$U_0 = v_b - v_a$

$U_0 = f(t)$

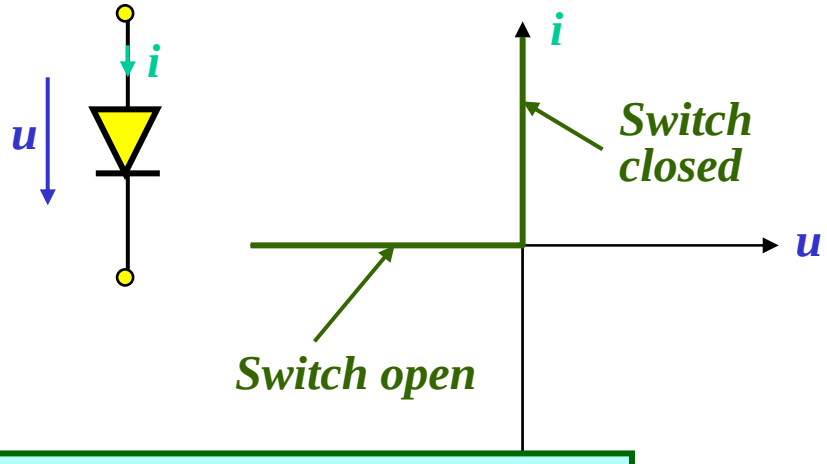
Erdung



$V_0 = 0$

Rein kontinuierliches Verhalten

Verhalten einer idealen Diode:

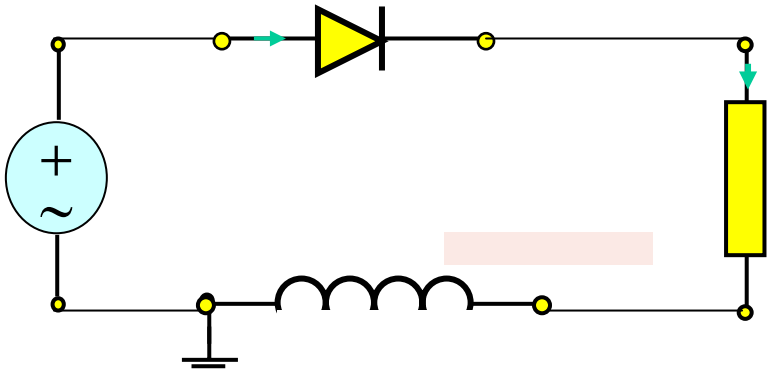


open = u < 0 ;
0 = if open then i else u ;

When ***u < 0***, the switch is open. No current flows through.

When ***u > 0***, the switch is closed. Current may flow. The ideal diode behaves like a short circuit.

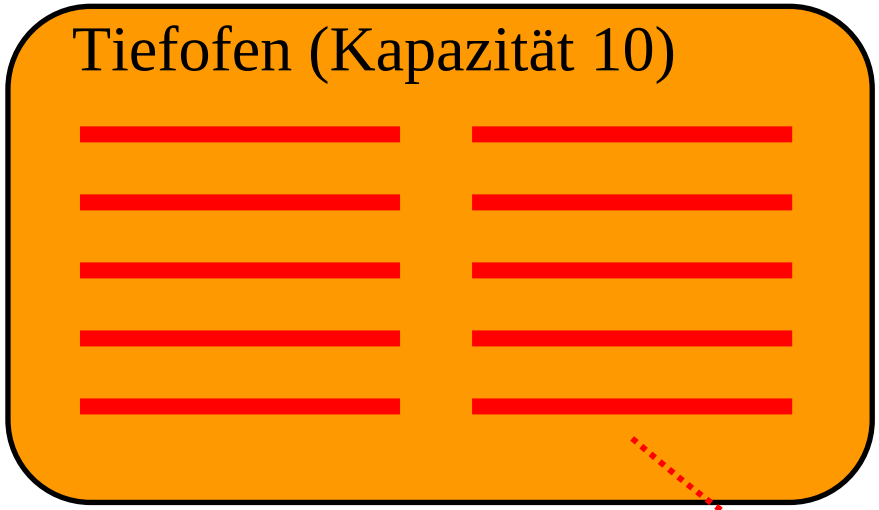
Beispiel:



Tiefofen in einem Walzwerk

Entfernen eines Barrens aus dem Ofen, falls seine Temperatur ≥ 380 ^{0}C

Stahlbarren im Warteraum



Ankunftsstrom
Poisson mit Rate λ
Ankunftstemperatur:
200 ^{0}C

Temperatur im Warteraum
25 ^{0}C

Temperatur im Ofen y(t) ≤ 500 ^{0}C

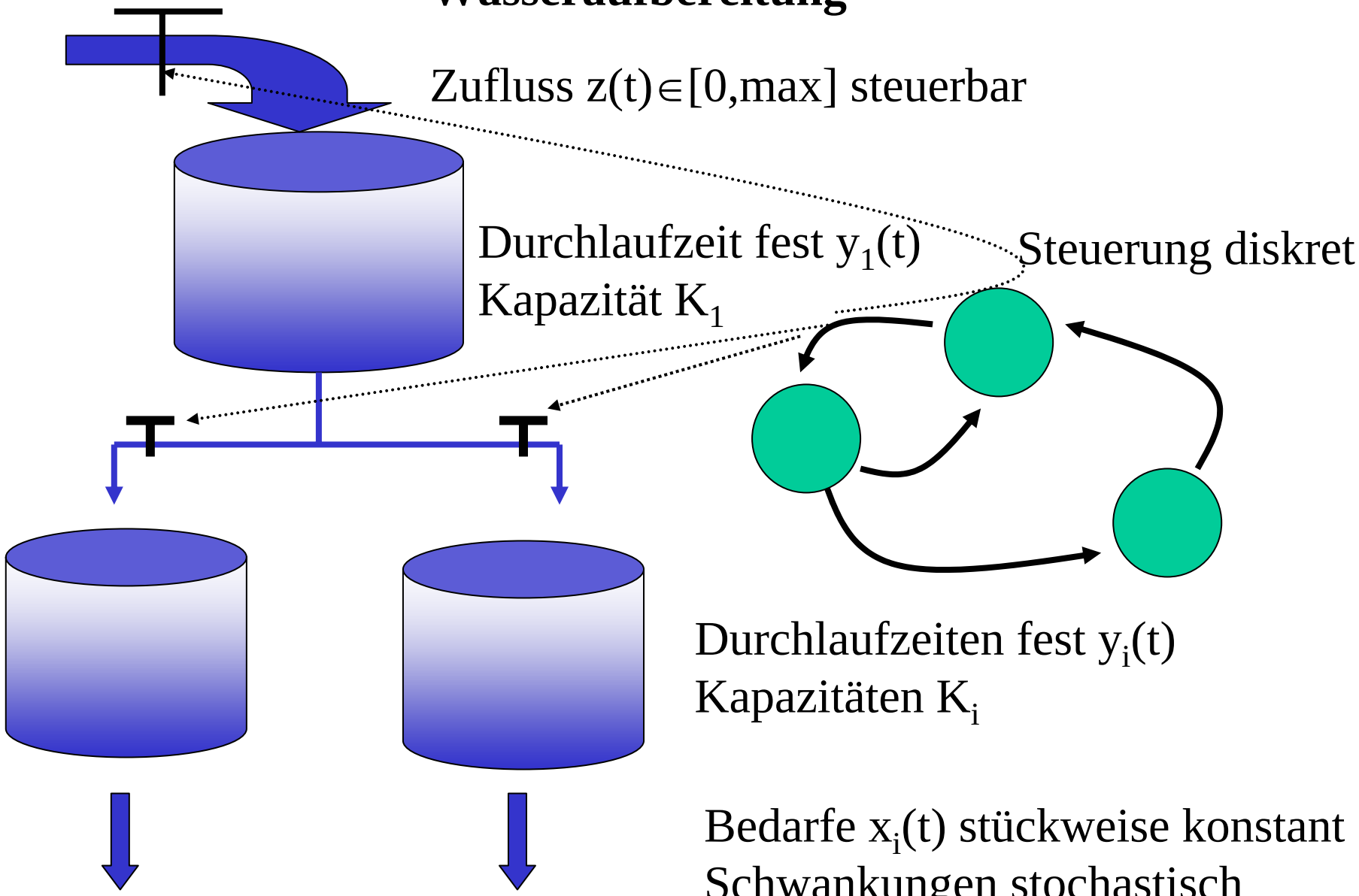
Temperaturänderungen:

$x_i(t)$ Temperatur Barren i
$u_i(t)$ Umgebungstemperatur Barren i (25 oder y(t))

$$\dot{x}_i(t) = (u(t) - x_i(t))/7$$

$$\dot{y}(t) = 500 - \sum_{i \text{ im Ofen}} (y(t) - x_i(t))/3$$

Wasseraufbereitung



5.2 Analyse kontinuierlicher Systeme

Allgemeine Form eines kontinuierlichen Systems: $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, t)$

Wir betrachten oft den einfachen Sonderfall: $\dot{\mathbf{x}} = f(\mathbf{x})$

- Vektor **u** kann durch Erweiterung während der Lösung berücksichtigt werden
- Oft gilt für den Lösungsvektor **p** = h(**x**), so dass Beobachtung von **x** ausreicht

Beschreibung des Modells kann (und wird oft) allgemeiner sein, da

1. höhere Ableitungen vorkommen
2. die Zeit t explizit vorkommt
3. nur Beziehungen zwischen Variablen beschrieben werden

In diesen Fällen muss zuerst die Modellbeschreibung (möglichst automatisch) so transformiert werden, dass die obige Darstellung entsteht (siehe MAO)

Ausgangslage

(bei den meisten kontinuierlichen Simulationsmodellen):

- eine Menge von Gleichungen über die Systemvariablen
- eine Menge von Differentialgleichungen

Vor der eigentlichen Simulation:

1. Zahl der Gleichungen reduzieren
   (Elimination durch Substitution)
2. Gleichungen sortieren
   (je eine Gleichung pro Variable)
3. Gleichungslösung
   (möglichst symbolisch, falls dies nicht möglich ist, siehe nächste Folie)

Falls Schritte 1-3 erfolgreich liegt ein Differentialgleichungssystem der gewünschten Form vor und die numerische Simulation kann durchgeführt werden

Differential-algebraische Gleichungssysteme (DAEs)

Allgemeine Form: $\dot{x} = f(x, y)$ *und* $0 = g(x, y)$

für den Anfangszustand gilt $0 = g(x_0, y_0)$

x differential Variablen
y algebraische Var.

Simultane Lösung eines (nicht notwendigerweise linearen) Gleichungssystems und eines Differentialgleichungssystems

Lineare DAEs: $\dot{x} = A \cdot x + B \cdot y + v$ *und* $0 = D \cdot x + E \cdot y + w$

Annahme: E sei quadratische reguläre Matrix
(kann oft durch Streichung von redundanten Gleichungen erreicht werden)

Damit gilt $y = -E^{-1}(Dx + w)$ und

$$\begin{aligned}\dot{x} &= Ax - E^{-1}(Dx + w) + v \\ &= (A - E^{-1}D)x + (v - E^{-1}w)\end{aligned}$$

Kann mit einem „normalen“ Verfahren für DGLs analysiert werden!

Allgemeine DAEs:

Simultane Lösung:

Verwendung eines impliziten Verfahrens zur DGL-Lösung

z.B. implizites Euler Verfahren: $x(t+\Delta t) = x(t) + \Delta t \cdot f(x(t+\Delta t))$

$$\begin{aligned} \frac{1}{\Delta t}\left(x(t+\Delta t) - x(t)\right) &= f(x(t+\Delta t), y(t+\Delta t)) \\ 0 &= g(x(t+\Delta t), y(t+\Delta t)) \end{aligned}$$

Umwandlung in ein Fixpunktproblem und Lösung mittels Newton-Verf.

$$\begin{aligned} x(t+\Delta t) &= \Delta t \cdot f(x(t+\Delta t), y(t+\Delta t)) + x(t) \\ y(t+\Delta t) &= y(t+\Delta t) - g(x(t+\Delta t), y(t+\Delta t)) \end{aligned}$$

Newton-Verfahren:

$$x^{(k+1)} = x^{(k)} - \Delta x^{(k)}$$
$$\Delta x^{(k)} = \left(H\left(x^{(k)}\right)\right)^{-1} \cdot f\left(x^{(k)}\right)$$
$$H(x) = \frac{df}{dx}$$

Falls Newton nicht konvergiert $\Rightarrow$ Δt verkleinern

Kombination ODE-Lösung und Gleichungslöser:

1. Starte mit $x^{(0)}(t+\Delta t)$ und $y^{(0)}(t+\Delta t)$, setze k=1
2. Berechne $x^{(k)}(t+\Delta t)$ mit ODE Löser
3. Berechne $y^{(k)}(t+\Delta t)$, so dass $g(x^{(k)}(t+\Delta t), x^{(k)}(t+\Delta t))=0$ mittels Newton-Verfahren
   - falls keine Konvergenz verkleinere Δt und fahre bei 2 fort
4. Falls $x^{(k)}(t+\Delta t) \approx x^{(k-1)}(t+\Delta t)$ und $y^{(k)}(t+\Delta t) \approx y^{(k-1)}(t+\Delta t)$ stop, sonst k=k+1 und fahre bei 2. fort

Lösungsverfahren für DGLs hier Runge-Kutta

Zur Erinnerung: Lösung durch Approximation der Taylor-Reihe am Punkt t

$\Rightarrow$ Koeffizienten der Taylorreihe bis zu einer vorgegebenen Potenz müssen durch entsprechende Wahl der α_{ij} und β_i wiedergegeben werden

Beispiele:

n=1

Taylorreihe:

$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \Delta t \cdot f(\mathbf{x}(t)) + O(\Delta t^2)$

RK-Verfahren:

$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \beta_1 \cdot \Delta t \cdot f(\mathbf{x}(t))$

$\Rightarrow \beta_1 = 1$ muss gelten

(eindeutiges Verfahren)

n=2

Taylorreihe:

$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \Delta t \cdot f(\mathbf{x}(t)) + \Delta t^2/2 \cdot f(\mathbf{x}(t)) + O(\Delta t^3)$

RK-Verfahren:

$\mathbf{x}(t+\Delta t) =$

$\mathbf{x}(t) + \beta_1 \cdot \Delta t \cdot f(\mathbf{x}(t)) + \beta_2 \cdot \Delta t \cdot f(\mathbf{x}(t) + \alpha_{21} \cdot \Delta t \cdot f(\mathbf{x}(t))$

$\Rightarrow \beta_1 + \beta_2 = 1$ und $\beta_2 \cdot \alpha_{21} = 0.5$ muss gelten

(kein eindeutiges Verfahren)

Mögliche Realisierungen

0	
1	0
$\frac{1}{2}$	$\frac{1}{2}$

0	
$\frac{2}{3}$	0
$\frac{1}{4}$	$\frac{3}{4}$

n=3

Taylorreihe:

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \Delta t \cdot f(\mathbf{x}(t)) + \Delta t^2/2 \cdot \dot{f}(\mathbf{x}(t)) + \Delta t^3/6 \cdot \ddot{f}(\mathbf{x}(t)) + O(\Delta t^3)$$

RK-Verfahren:

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \beta_1 \cdot \Delta t \cdot f(\mathbf{x}(t) + \beta_2 \cdot \Delta t \cdot f(\mathbf{x}(t) + \alpha_{21} \cdot \Delta t \cdot f(\mathbf{x}(t))) + \beta_3 \cdot \Delta t \cdot f(\mathbf{x}(t) + \alpha_{31} \cdot \Delta t \cdot f(\mathbf{x}(t)) + \alpha_{32} \cdot \Delta t \cdot f(\mathbf{x}(t) + \alpha_{21} \cdot \Delta t \cdot f(\mathbf{x}(t))))$$

$\beta_1 + \beta_2 + \beta_3 = 1$ und $\beta_2 \cdot \alpha_{21} + \beta_3 \cdot \alpha_{31} + \beta_3 \cdot \alpha_{32} = 1/2$ und $\beta_3 \cdot \alpha_{21} \cdot \alpha_{32} = 1/6$ muss gelten (Lösung eines nichtlinearen Gleichungssystems)

Mögliche Realisierungen

Heun

0			
1/3	0		
0	2/3	0	
1/4	0	3/4	

Kutta

0			
1/2	0		
−1	2	0	
1/6	4/6	1/6	

Vorgehen bei der Analyse kontinuierlicher Systeme
(wie bisher beschrieben):

(Approximative) Berechnung der Funktionswerte zu äquidistanten Zeitpunkten mit Abstand Δt

- kleinere Werte von Δt führen zu genaueren Resultaten
  (zumindest so lange bis Rundungsfehler auftreten)
- kleinere Werte von Δt erhöhen den Aufwand
- tatsächlicher Fehler der Berechnung hängt von der Struktur des zu analysierenden Systems ab
  (Δt muss entsprechend der Zeitkonstanten des Systems gewählt werden)

Auftretende Probleme:

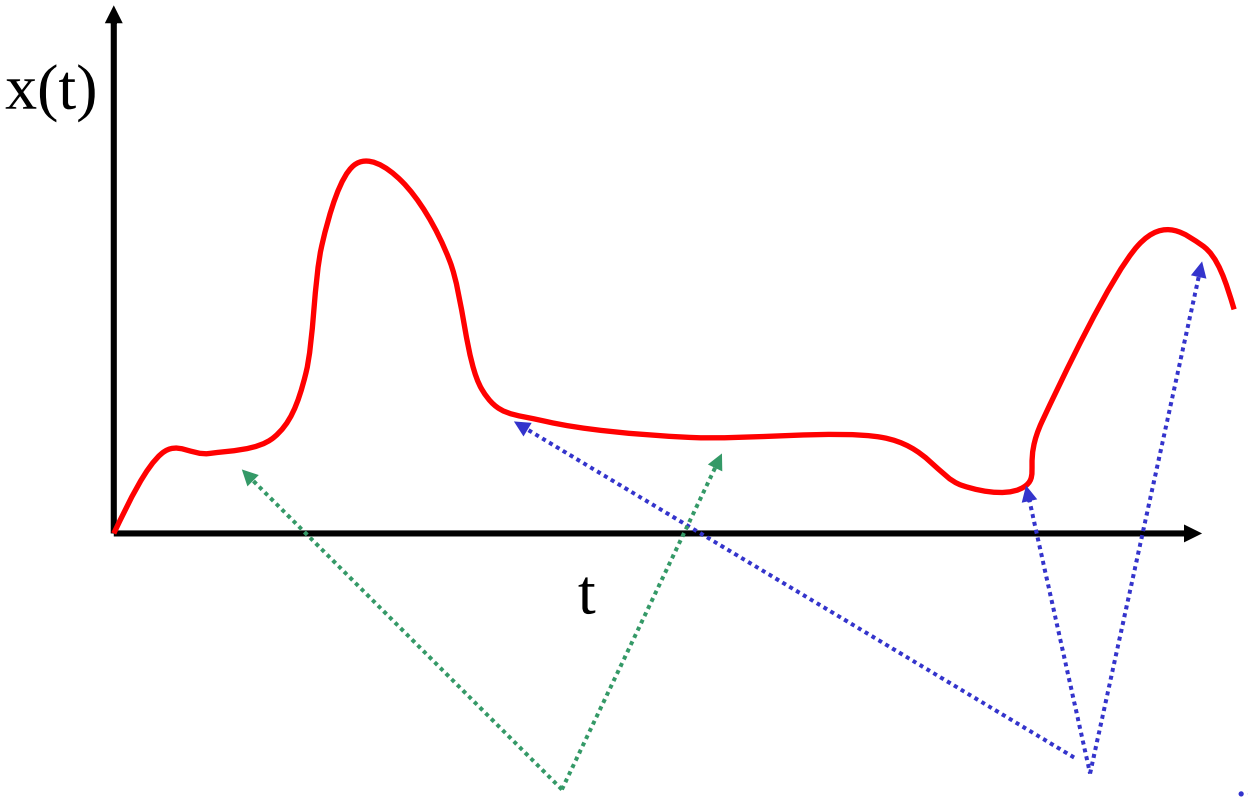
- Zeitkonstanten eines Systems sind nur schwer ablesbar und variieren in Abhängigkeit vom Zustand

$\Rightarrow$ die Festlegung einer adäquaten Schrittweite ist schwieirg

$\Rightarrow$ die adäquate Schrittweite kann sich während der Simulation ändern

$\Rightarrow$ naheliegender Ansatz:
Adaptive Schrittweitensteuerung während der Analyse
(heute in fast allen Werkzeugen zur kontinuierlichen Simulation verwendet)

Darstellung des Problems am einfachen Beispiel



Falls Δt zu

- groß gewählt wird, wird die Berechnung zu ungenau
- klein gewählt wird, wird der Berechnungs-aufwand zu hoch

➢ es existiert kein festes Δt, welches genügend genaue Resultate mit vertretbarem Aufwand liefert

Fast lineares Verhalten, auch bei relativ großem Δt genau genug approximierbar

Starke Änderung der Ableitung, nur bei kleinem Δt genau genug approximierbar

Zur sinnvollen Berechnung sollte Δt so klein sein, dass

$|\varepsilon_{ED}(\Delta t)| \approx C \cdot \Delta t^{n+1}$ für ein Verfahren der Ordnung n

Bei bekanntem C und vorgegebener Fehlertoleranz ε_{Tol} gilt:

$$
\begin{aligned}
& |\varepsilon_{ED}(\Delta t)| && \leq && \varepsilon_{Tol} \\
\Leftrightarrow \quad & C \cdot \Delta t^{n+1} && \leq && \varepsilon_{Tol} \\
\Leftrightarrow \quad & \Delta t && \leq && \left(\frac{\varepsilon_{Tol}}{C} \right)^{1/(n+1)}
\end{aligned}
$$

$\varepsilon_{ED}(Dt)$ und ε_{Tol} können als absolute oder relative Fehler berechnet werden:
Praktisch verwendete Definition des relativen Fehlers
($\mathbf{x}_e$ exakt, $\mathbf{x}_a$ approximativ)

$$\varepsilon_{rel} = \frac{\|\mathbf{x}_e - \mathbf{x}_a\|}{\max(\|\mathbf{x}_e\|, \|\mathbf{x}_a\|, \delta)}$$

Da C nicht bekannt ist, muss der Wert geschätzt werden
Eine Schätzung kann nur auf Basis numerisch berechneter Resultate erfolgen $\Rightarrow$

- Berechne Resultate auf unterschiedliche Weise
- Benutze Differenz der Ergebnisse zur Schätzung von C

Methode der Schrittweitenhalbierung

In jedem Schritt werden 3 Werte berechnet:

- Wert zum Zeitpunkt t + Δt: $\mathbf{x}_{\Delta t}(t+\Delta t)$
- Wert zum Zeitpunkt t + 0.5·Δt: $\mathbf{x}_{0.5,\Delta t}(t+0.5\cdot\Delta t)$ und davon ausgehend Wert zum Zeitpunkt t + Δt: $\mathbf{x}_{0.5,\Delta t}(t+\Delta t)$

Approximation des auftretenden Fehlers:

$$\begin{aligned}\left|\varepsilon_{ED}(\Delta t)\right| &\approx \left\|\mathbf{x}_{\Delta t}(t+\Delta t)-\mathbf{x}_{0.5,\Delta t}(t+\Delta t)\right\| \\ &\approx C\cdot\Delta t^{n+1}\end{aligned}$$

Womit folgende Abschätzung gilt: $C \approx \dfrac{\left\|\mathbf{x}_{\Delta t}(t+\Delta t)-\mathbf{x}_{0.5,\Delta t}(t+\Delta t)\right\|}{\Delta t^{n+1}}$

Anschließend wird Δt verändert, um die vorgegeben Fehlerschranke zu erreichen: $\Delta t_{opt} = \alpha \cdot \sqrt[n+1]{\varepsilon_{Tol}/C}$ mit $0<\alpha<1$

Vierfacher Rechenaufwand pro Schritt !!
(3 Schritte zur Bestimmung von Δt_{opt} + 1 Verfahrensschritt)

Methode der eingebetteten Verfahren

Die Lösung $\mathbf{x}(t+\Delta t)$ wird, ausgehend von $\mathbf{x}(t)$, mit zwei unterschiedlichen Verfahren berechnet, die Lösungsvektoren $\mathbf{x}_1(t+\Delta t)$ und $\mathbf{x}_2(t+\Delta t)$ liefern (z.B. RK 2/3 oder RK 4/5)

Anschließend schätzt man $C \approx \dfrac{\left\|\mathbf{x}_1(t+\Delta t) - \mathbf{x}_2(t+\Delta t)\right\|}{\Delta t^{n+1}}$

(wobei n die kleinere der beiden Fehlerordnungen ist)

Δt_{opt} wird wie auf der vorherigen Folie beschrieben berechnet

Aufwand pro Schritt: 3 Berechnungsschritte
(2 zur Bestimmung von Δt_{opt} und 1 Verfahrensschritt)

Aufwand lässt sich auf 2 Berechnungsschritte reduzieren, falls Runge-Kutta Verfahren so konstruiert werden, dass Zwischenergebnisse wiederverwendbar sind (eingebettete Verfahren).

Weitere Bemerkungen:

- C wird nur geschätzt und die Schätzung kann falsch sein (deshalb eher etwas kleinere Werte für α verwenden)
- Für die meisten Modelle wird der Aufwand zusätzlicher Berechnungen durch die dynamische Anpassung der Schrittweite mehr als ausgeglichen
- Eine a priori Schätzung von Δt ist nicht notwendig
- Falls das aktuell verwendet Δt kleiner als das ermittelte Δt_{opt} ist, so kann auf den abschließenden Verfahrensschritt verzichtet werden und mit dem „genaueren“ der ermittelten Vektoren $x(t+\Delta t)$ fortgefahren werden
- Da durch eine Verkleinerung der Schrittweite die Zahl der Schritte steigt, steigt auch der Gesamtfehler bei konstantem Einzelschrittfehler
  Berücksichtigung des Gesamtfehlers liefert:
  Schrittzahl $n = \tau/\Delta t$ und $\Delta t_{opt} = \sqrt[n]{\varepsilon_{DTol} \cdot \tau / C}$
  (ε_{DTol} Gesamtfehlerschranke)

Vorgehen i.d.R. nicht bei hybriden Modellen einsetzbar, da

- Änderungen abrupt stattfinden und damit nur approximativ durch einen starken Anstieg beschrieben werden
- der diskrete Teil sich nicht adäquat beschreiben lässt (Ereignisorientiertheit, Stochastik etc.)

Verfahren zur kontinuierlichen Simulation
(inkl. Schrittweitensteuerung) nutzbar,
muss aber mit Algorithmus zur diskreten Simulation gekoppelt werden
$\Rightarrow$ hybride Simulation

Einfacher Spezialfall:

- alle Ableitung von der Form $\dot{x} = k$ für konstantes k
  $\Rightarrow$ Darstellung durch diskrete Ereignisse, da Wert von $x(t)=x(0+k \cdot t)$
- keine Ableitung von einer kontinuierlichen Variable abhängt
  $\Rightarrow$ Darstellung durch diskrete Ereignisse, Ereigniszeit kann aus Berechnung der bestimmten Integrale gewonnen werden

5.3 Hybride Simulationsansätze

- Hybride Simulation als Erweiterung diskreter Simulation, falls kontinuierlicher Teil einfach diskretisierbar
- Hybride Simulation als Erweiterung kontinuierlicher Simulation, falls diskontinuierliche Ereignisse spezifizierbar und ohne Ereignisliste (u.U. auch ohne Stochastik) realisierbar

Für allgemeine hybride Modelle sind spezielle Simulationsansätze notwendig (Kombination diskret & kontinuierlich)

Solange keine Interaktion zwischen diskretem und kontinuierlichem Teil auftritt, können die jeweiligen Simulationskonzepte Verwendung finden

$\Rightarrow$ zentraler Aspekt Interaktion zwischen den unterschiedlichen Teilen

Mögliche Interaktionsszenarien:

Zeitgesteuerte Ereignisse

- Ereignisse werden in der Ereignisliste gehalten und sind nach der Zeit ihres Eintretens geordnet
- Ereignisse können zu Zustandsänderungen im diskreten Teil führen
- Ereignisse können zu Diskontinuitäten im kontinuierlichen Teil führen
- Ereignisse werden zu Beginn der Simulation und während der Simulation durch andere Ereignisse generiert (Zeit eines neuen Ereignisses kann nicht in der Vergangenheit liegen!)

Zustandsgesteuerte Ereignisse

- Bei Über- oder Unterscheiten vorgegebener Schranken wird ein diskretes Ereignis ausgelöst
- Auswirkung des Ereignisses siehe oben
- Ereigniszeitpunkte ergeben sich implizit aus der Zustandstrajektorie (nicht vorhersehbar bzw. speicherbar)

Simulation zeitgesteuerter Ereignisse:

Sei t der aktuelle Zeitpunkt und t_e der Zeitpunkt des nächsten Ereignisses in der Ereignisliste

- Integrationsalgorithmus wird benutzt, um aus bekanntem x(t) den Wert $x(t_e)$ zu berechnen,
- falls im Intervall $[t, t_e)$ keine zustandsgesteuerten Ereignisse auftreten
  - normaler Ablauf des Algorithmus,
  - im letzten Schritt ist Δt so zu wählen, dass genau t_e erreicht wird
- diskretes Ereignis zum Zeitpunkt t_e wird ausgeführt und liefert neuen Zustand $x'(t_e)$
- $x'(t_e)$ wird als Startwert der Simulation bis zum nächsten Ereignis verwendet

Simulation eines Systems mit d Ereignissen entspricht d+1 separaten Simulationsläufen

Zustandsgesteuerte Ereignisse
Algorithmus für den eindimensionalen Fall, Ereignis falls x(t)=y

1. Ausführung eines Simulationsschrittes mit Schrittweite Δt
2. Falls $((x(t)<y \wedge x(t+\Delta t)>y) \vee (x(t)>y \wedge x(t+\Delta t)<y))$
   a. $\delta t=(|x(t)-y|)/(|x(t)-x(t+\Delta t)|) \cdot \Delta t$
   b. berechne $x(t+\delta t)$
   c. Falls $((x(t)<y \wedge x(t+\delta t)<y) \vee (x(t)>y \wedge x(t+\delta t)>y))$
      $t=t+\delta t$ und $\Delta t=\Delta t-\delta t$
   d. Falls $((x(t)<y \wedge x(t+\delta t)>y) \vee (x(t)>y \wedge x(t+\delta t)>y))$
      $\Delta t=\delta t$
   e. falls $|x(t+\Delta t)-y|<\varepsilon$
      $t = t + \Delta t$ und gehe zu 3.
   f. falls $|x(t)-y|<\varepsilon$
      gehe zu 3.
   g. gehe zu 2a
3. führe das diskrete Ereignis aus
4. Falls $t+\Delta t >$ Simzeit dann stop, sonst $t=t+\Delta t$ und gehe zu 1.

- Algorithmus ist für mehrere Dimensionen einfach erweiterbar
- x(t)=y kann nur bis auf eine Toleranz ε bestimmt werden
- es ist nicht feststellbar, ob x(t‘)=y für ein $t<t‘<t+\Delta t$ erreicht wird, wenn sich das Vorzeichen der Differenz nicht ändert
- Automatische Schrittweitensteuerung sollte Abstand |x(t)-y| mit einbeziehen

5.4 Spezifikation hybrider Systeme

- Zahlreiche kontinuierliche Simulatoren erlauben Integration diskreter Ereignisse, oft aber keine vollständige ereignisdiskrete Modellierung
- Einzelne ereignisdiskrete Simulatoren erlauben Integration kontinuierlicher Anteile, oft muss das kontinuierliche Lösungsverfahren aber manuell integriert werden
- Einzelne Simulationsbibliotheken/-sprachen beinhalten Komponenten für diskrete und kontinuierliche Simulation, Kopplung ist aber in der Regel manuell zur realisieren

Wie betrachten hier

- Modelica als objektorientierte Sprache für kontinuierliche Systeme mit Erweiterungen zur hybriden Modellierung
- Hybride Petri Netze
- Hybride Automaten

Modelica:

Modelica ist eine objektorientierte Sprache primär zur Beschreibung kontinuierlicher Systeme.

Beispiel Schaltkreis:

```
connector Pin
  Voltage v ;
  flow Current i ;
end Pin
```

```
connect (Pin1, Pin2)
```

definiert implizit die Gleichungen

Pin1.v = Pin2.v und Pin1.i + Pin2.i = 0

```
partial class TwoPin
   Pin p, n ;
   Voltage v ;
   Current i ;
 equation
  v = p.v – n.v ;
  0 = p.i + n.i ;
  i  = p.i ;
end TwoPin ;
```

Definition eines Widerstandes:

```
class Resistor
   extends TwoPin ;
   parameter Real R(unit = „Ohm“) ;
 equation
   R · i = v ;
end Resistor ;
```

In ähnlicher Form werden weitere Elemente definiert

Definition von diskreten Ereignissen über if .. then .. else ...

Beispiel Diode:

```
class Diode
  extends TwoPin ;
equation
  open = u < 0 ;
  0 = if open then i else v ;
end Diode ;
```

Konstrukte sind ausreichend, um Diskontinuitäten in kontinuierlichen Modellen zu beschreiben, aber nicht, um komplexe diskrete Modelle zu beschreiben!

Bibliotheken existieren um z.B. einfache Petri-Netze oder State-Charts zu beschreiben

Nicht realisiert sind

- stochastische Funktionen
- Ereignisliste
- dynamische Objekterzeugung

Hybride Petri-Netze

Petri Netze sind ursprünglich ein Modell mit diskreter Zustandsbeschreibung und explizit ohne Zeitbegriff.
In der Folgezeit in unterschiedliche Richtungen weiterentwickelt, u.a. zu zeitbehafteten, stochastischen kontinuierlichen und hybriden Petri Netze

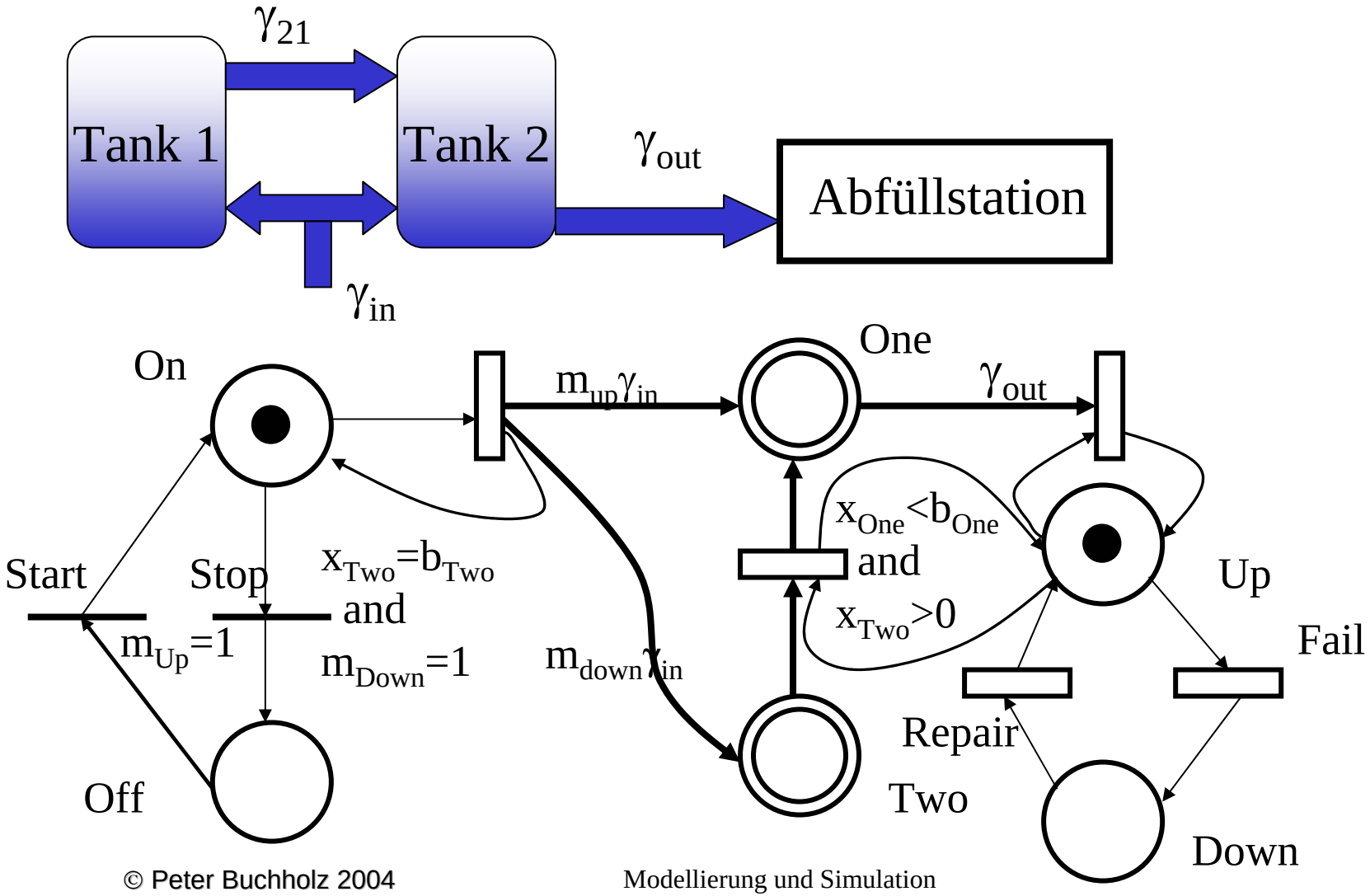
$PN=(S,T,F,W,M_0)$ mit

- S endliche Stellenmenge
- T endliche Transitionsmenge
- $F \subseteq S \times T \cup T \times S$
- M_0 initiale Markierung

Für spezielle Klassen Erweiterung der Definition!

Beispiel: Fluide stochastische Petri-Netze (andere Klassen existieren)

Einfaches Beispiel zwei Tanks



Hybride Automaten:

- Automatemodelle sind in der Informatik in unterschiedlichen Varianten wohl etabliert (zeitlos, stochastisch, probabilistisch, timed, ...)
- hybride Automaten entstanden als Erweiterung von timed automata
  sie vereinen
  - kontinuierliches Verhalten in den Zuständen
  - Ereignisse durch Zustandsübergänge

Def. hybrider Automat (eine von vielen Varianten):

$H=(Q, X, Init, Inv, f, E, G, J, \Sigma)$ mit

- Q endliche Zustandsmenge
- $X \subseteq \mathbb{R}^n$ kontinuierlicher Variablenraum
- $Init \subseteq L \times L$ initiale Zustandsmenge
- $Inv: L \rightarrow 2^X$ Zustandsinvariante
- $f: Q \rightarrow (X \rightarrow X)$ Änderungsfunktion kontinuierlicher Variablen
- $E \subseteq L \times L$ Transitionsmenge
- $G: E \rightarrow L^X$ Guard-Menge
- $J: E \rightarrow (X \rightarrow X)$ Diskontinuitätsfunktion
- S endliche Menge von Transitionslabeln

Beispiel: Pumpe

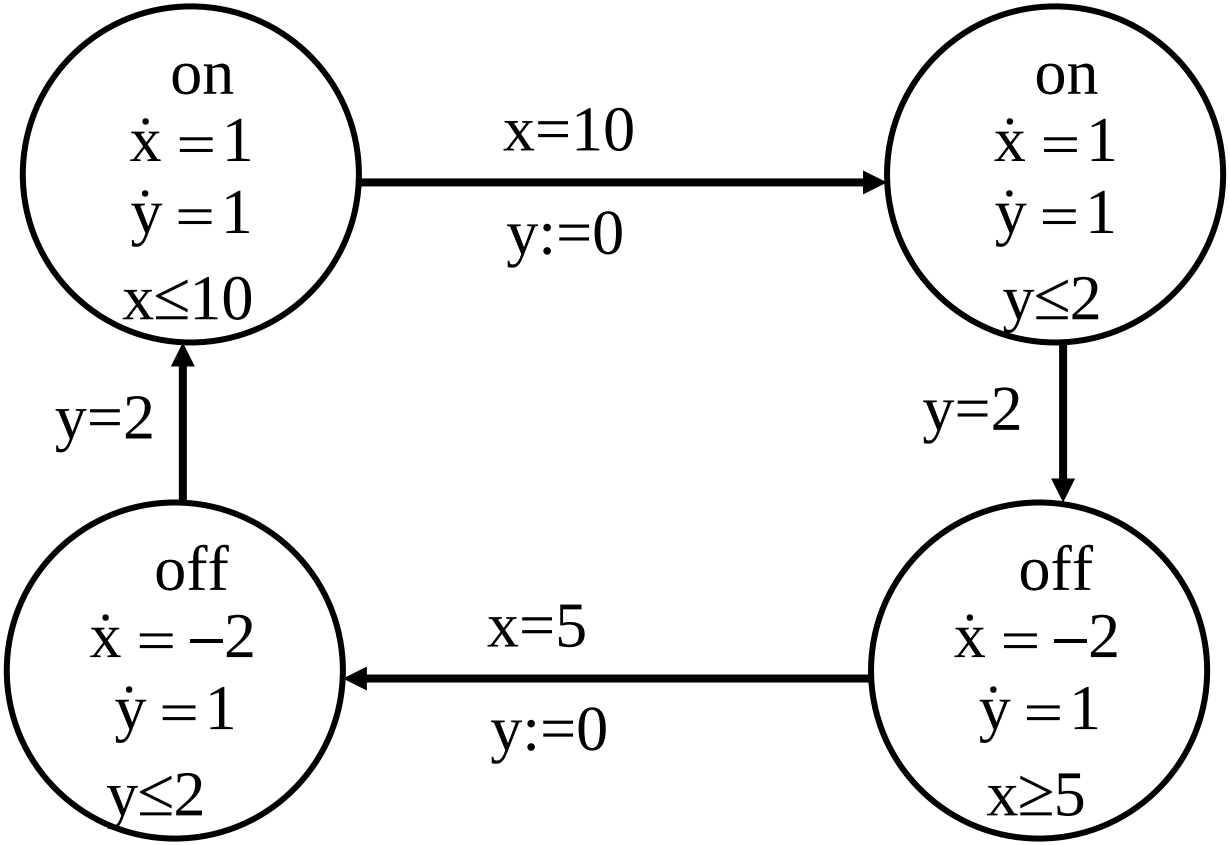
Pumpe on Wasser steigt um 1cm pro Sek.,

off Wasser fällt um 2cm pro Sek.

Signalübertragung zur Pumpe dauert 2 Sek.

x Wasserhöhe, y Zeit seit letzter Sendung

Ziel: Wasserspiegel zwischen 2 und 12 cm



Hybride Automaten

- existieren auch mit Erweiterungen zur stochastischen Modellierung
- erlauben in einigen Varianten auch die Komposition zu Automatennetzen (Varianten, in denen dynamisch neue Automaten erzeugt werden, existieren aber bisher nicht)
- wurden oftmals in eingeschränkten Versionen untersucht, so dass analytische zustandsbasierte Analysetechniken anwendbar sind, Ziele dann Nachweis, dass bestimmte Zustände nicht eintreten können bzw. alle erreichbaren Zustände bestimmte Invarianten erfüllen
- lassen sich zwar in einigen Werkzeugen spezifizieren (Simulink, Matlab u.a.) werden aber nicht primär unter dem Gesichtspunkt der Spezifikation von Simulationsmodellen untersucht