Wie wird ein Regler implementiert?

Thierry Prud'homme

Hochschule Luzern
Technik & Architektur

Outline

1. Direkter / Indirekter Reglerentwurf

Outline

1. Direkter / Indirekter Reglerentwurf
2. Implementierung eines digitalen Reglers

Outline

1. Direkter / Indirekter Reglerentwurf
2. Implementierung eines digitalen Reglers
3. Diskretisierung des PID Reglers

Outline

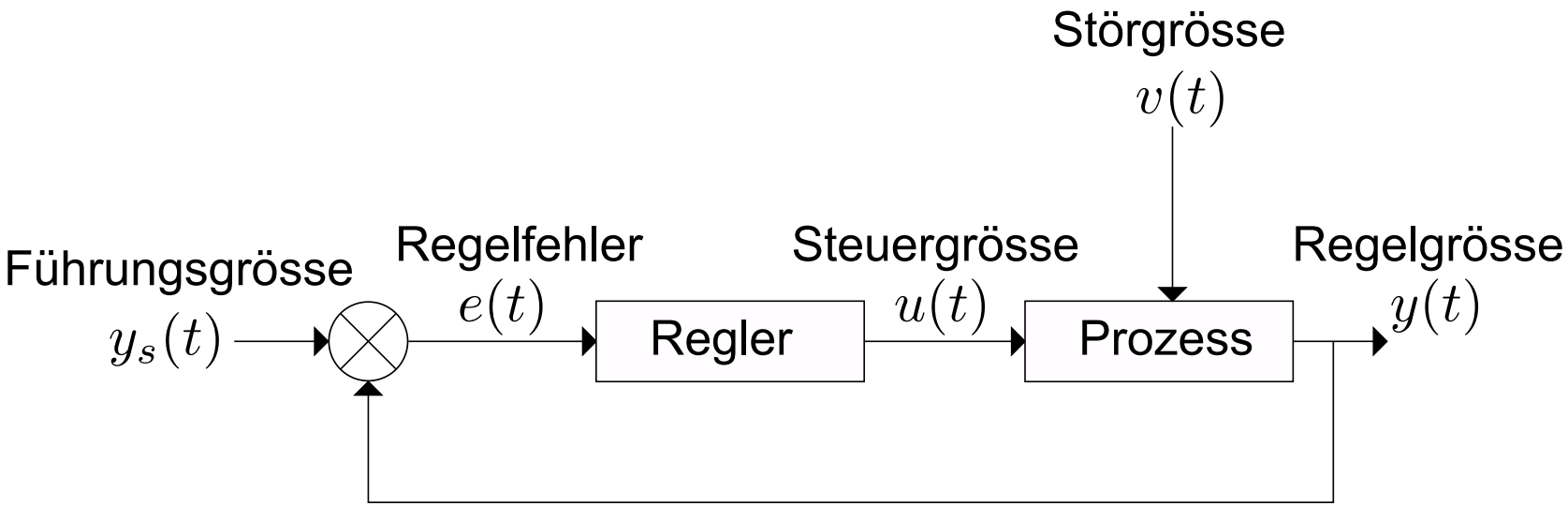
1. Direkter / Indirekter Reglerentwurf
2. Implementierung eines digitalen Reglers
3. Diskretisierung des PID Reglers
4. Anti-Reset Windup

Lernziele

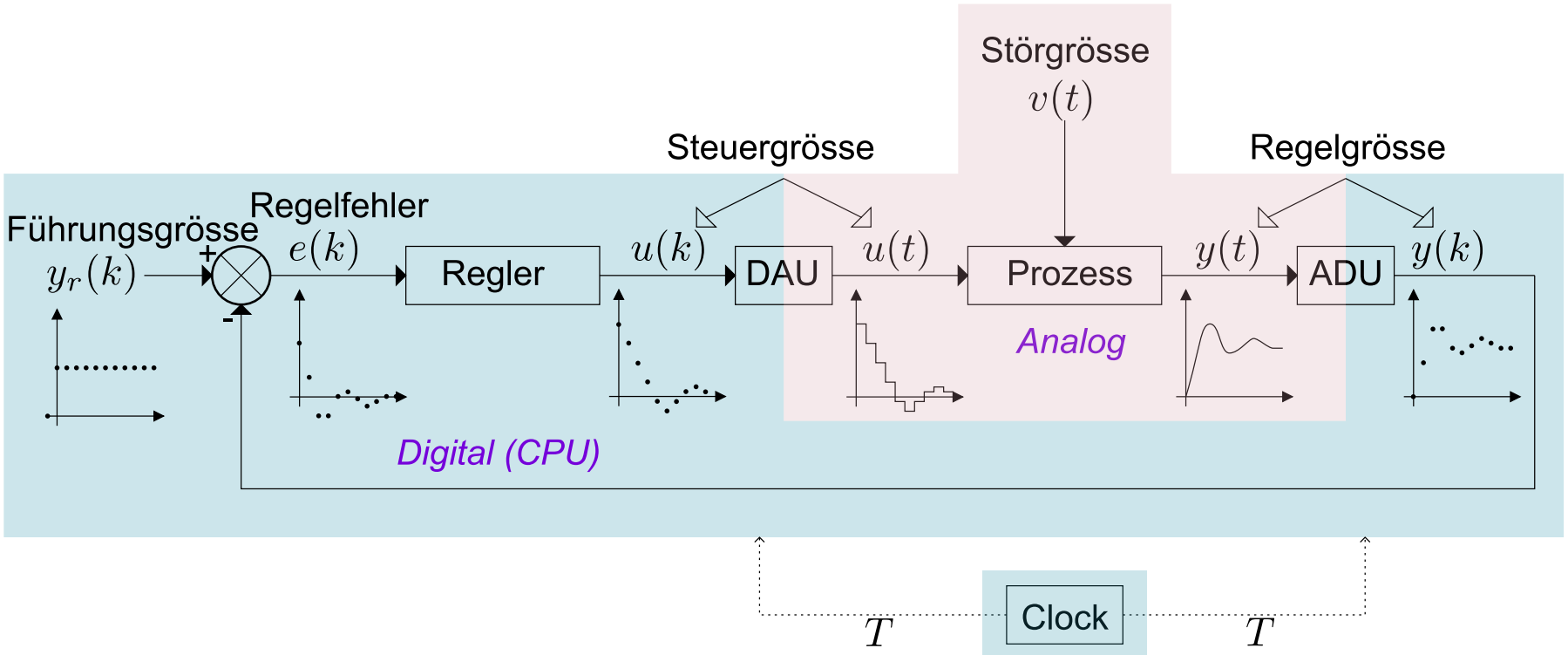
- Die Studierende können den Unterschied zwischen direkten und indirekten Reglerentwurf erklären.
- Die Studierende können einen PID Regler diskretisieren.
- Die Studierende können einen PID Regler mit und ohne ARW implementieren.

1. Direkter / Indirekter Reglerentwurf
   - Analoger / Digitaler Geschlossener Regelkreis
   - Direkter / Indirekter Reglerentwurf
2. Implementierung eines digitalen Reglers
3. Diskretisierung des PID Reglers
4. Anti-Reset Windup

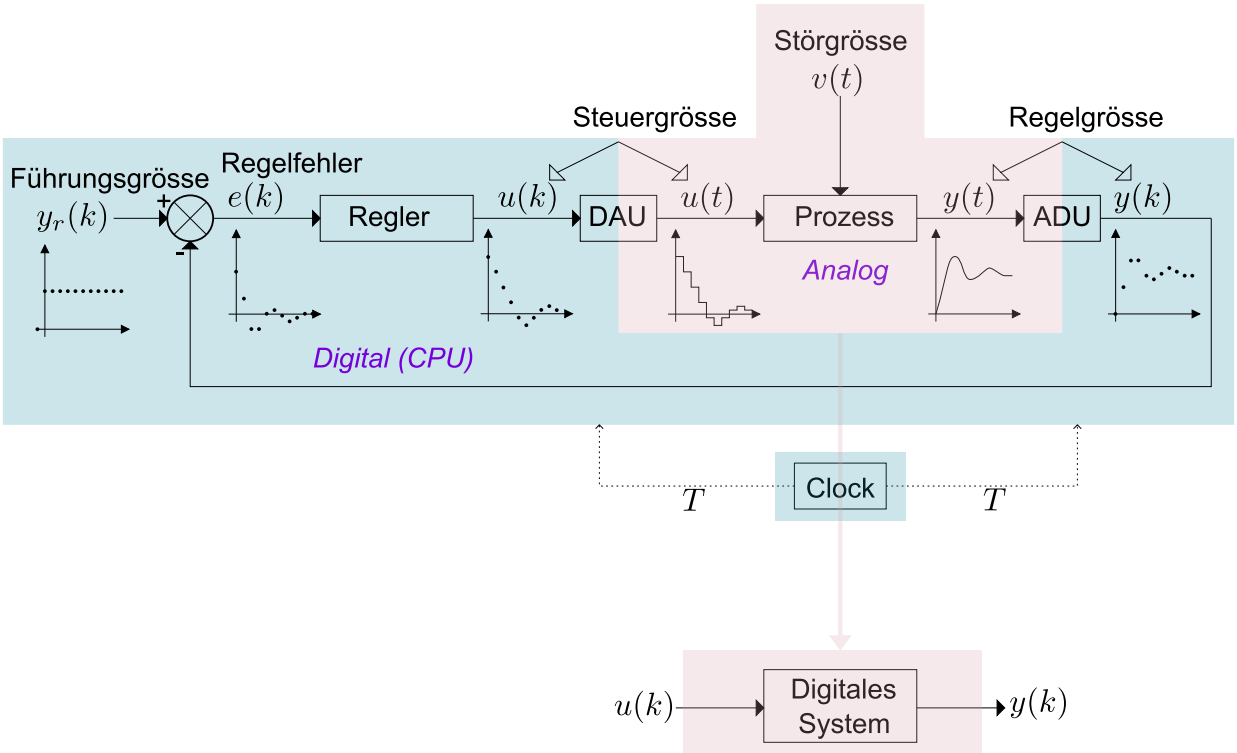
Analoger Geschlossener Regelkreis



Digitaler Geschlossener Regelkreis

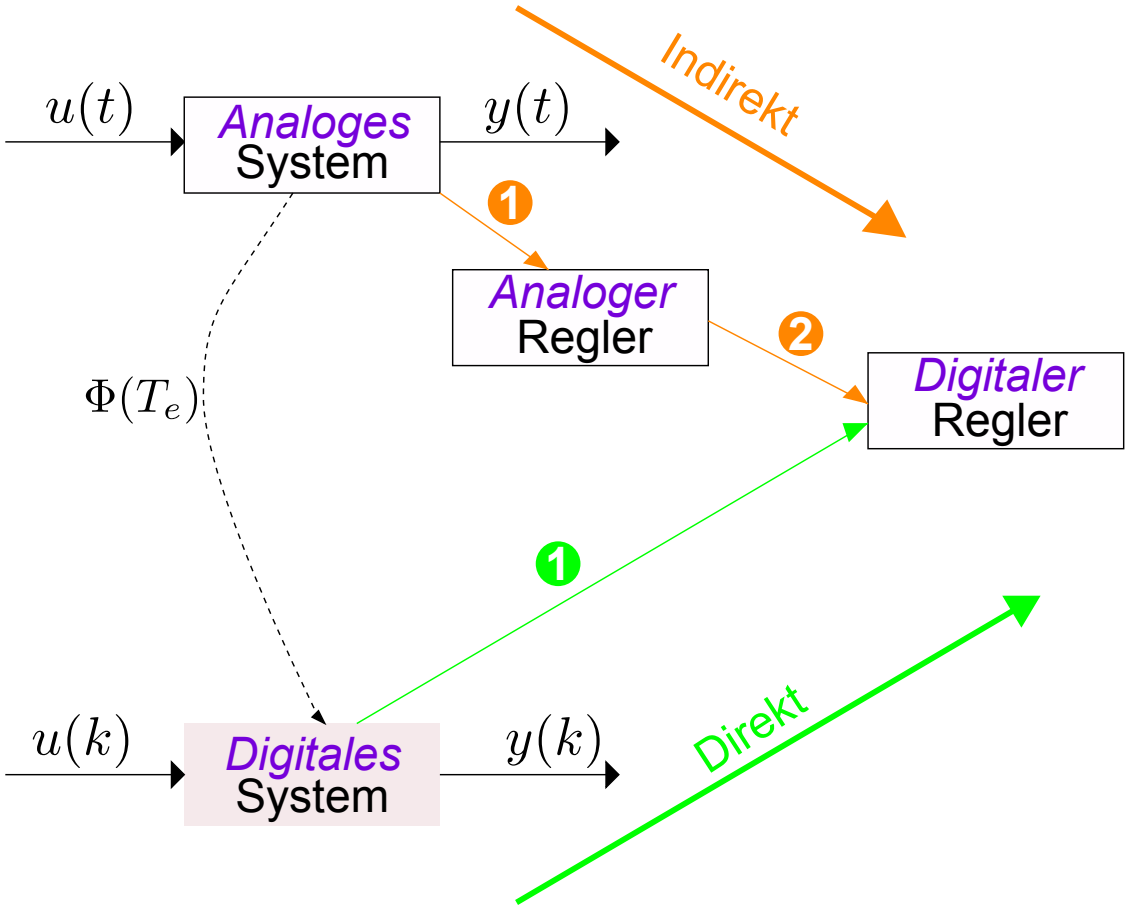


Digitale Regelstrecke / Digitales System



$$\begin{aligned} \{u(t), T\} &\rightarrow u(k) \\ \{u(t), T\} &\rightarrow u(k) \\ G(s) = \frac{Y(s)}{U(s)} &\rightarrow H(z) = \frac{Y(z)}{U(z)} \end{aligned}$$

Direkter / Indirekter Reglerentwurf



Fokus auf Indirekt - Vorteile / Nachteile

Vorteile

- Keine neuen Methode notwendig
- Intuitivität, Intepretation

Nachteile

- Auf analogen Regler beschränkt (PID)
- Keine explizite Berücksichtigung der Discretizierung

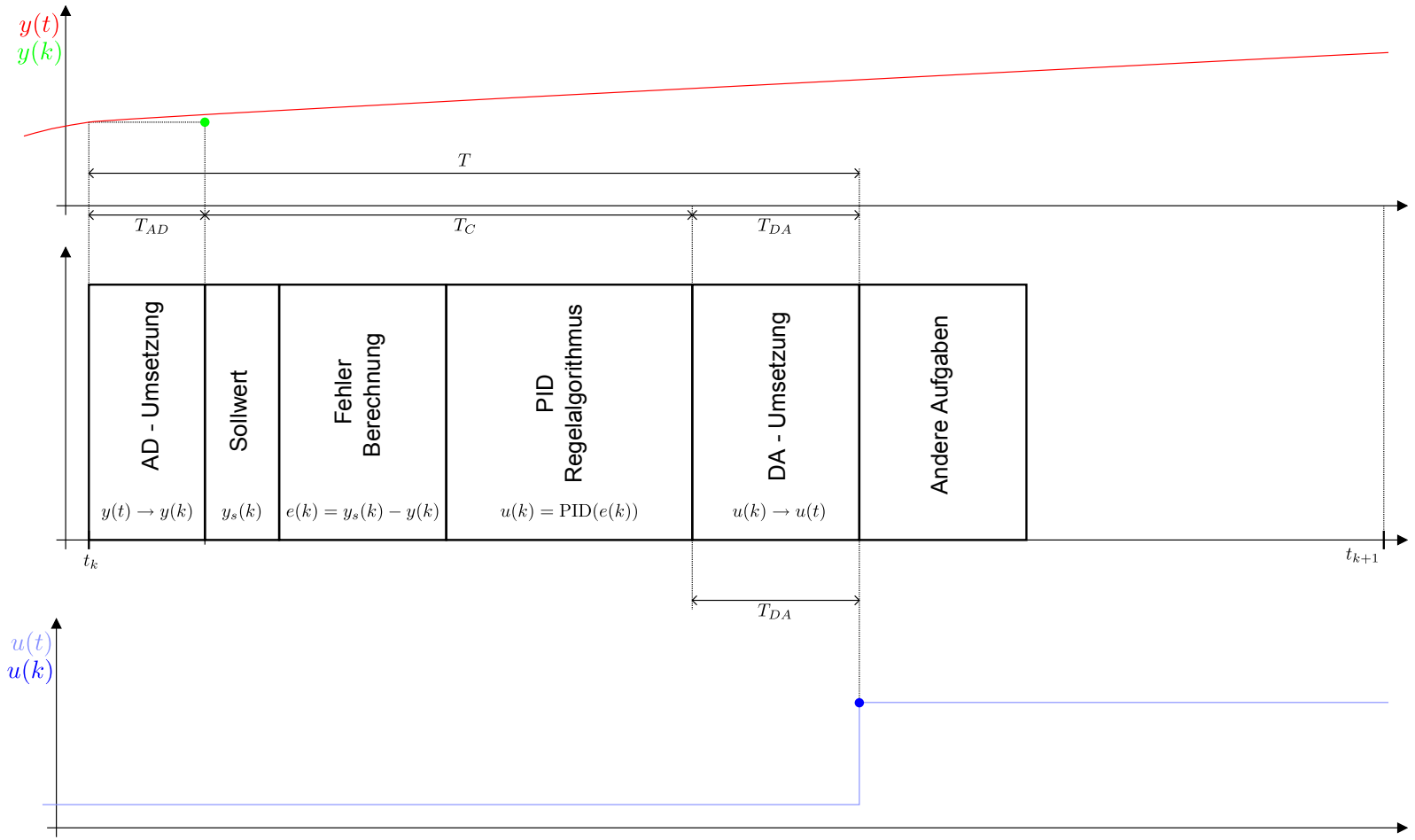
1 Direkter / Indirekter Reglerentwurf

2 Implementierung eines digitalen Reglers
- Regelungsaufgabe
- Pseudocode

3 Diskretisierung des PID Reglers

4 Anti-Reset Windup

Regelungsaufgabe



$$(T_{AD} + T_C + T_{DA}) \ll T \ ?$$

Pseudocode

```
start_control_task
// Trigger the AD conversion and get the results
// in y_m_k
y_m_k = AD(Y_M_AD_ID);
// Get the reference point
y_s_k = GetReference();
// Compute the error
e_k = y_s_k - y_m_k;
// Compute the control command u_k
u_k = PID(e_k);
// Trigger the DA conversion
DA(U_K_DA_ID, u_k);
end_control_task
```

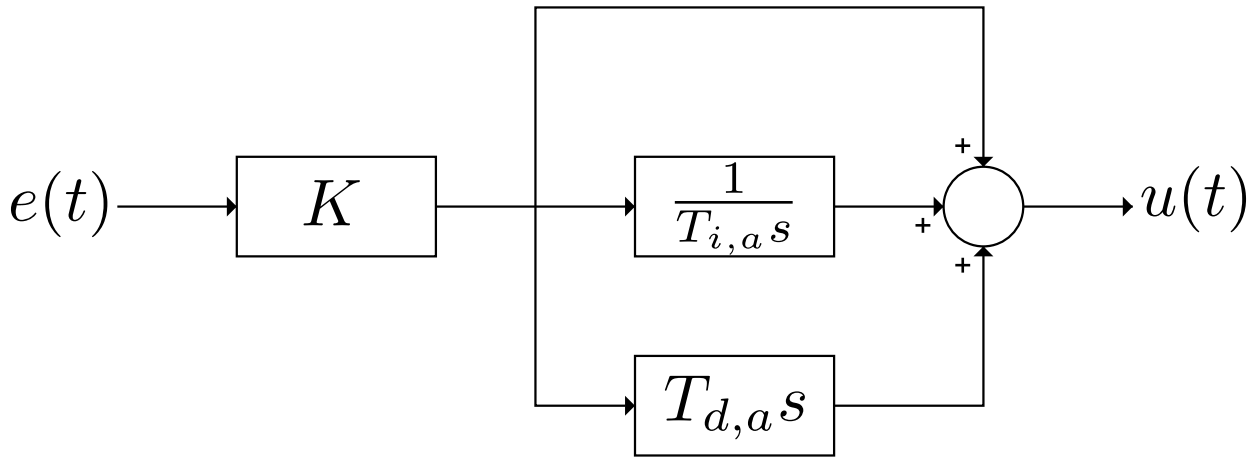
1 Direkter / Indirekter Reglerentwurf

2 Implementierung eines digitalen Reglers

3 Diskretisierung des PID Reglers
- Analoger/Digitaler PID
- Diskretisierung
- Pseudocode

4 Anti-Reset Windup

Analoger PID



$$
\begin{aligned}
u(t) &= K_a \left(e(t) + \frac{1}{T_{i,a}} \int_0^t e(\tau) d\tau + T_{d,a} \dot{e}(t) \right) \\
u(t) &= K_a e(t) + \frac{K_a}{T_{i,a}} \int_0^t e(\tau) d\tau + K_a T_{d,a} \dot{e}(t) \\
u(t) &= u_{k,a}(e(t)) + u_{i,a}(e(t)) + u_{d,a}(e(t))
\end{aligned}
$$

Digitaler PID
Zielsetzung

Wir haben...

$$u(t) = u_{k,a}(e(t)) + u_{i,a}(e(t)) + u_{d,a}(e(t))$$

Wir wollen...

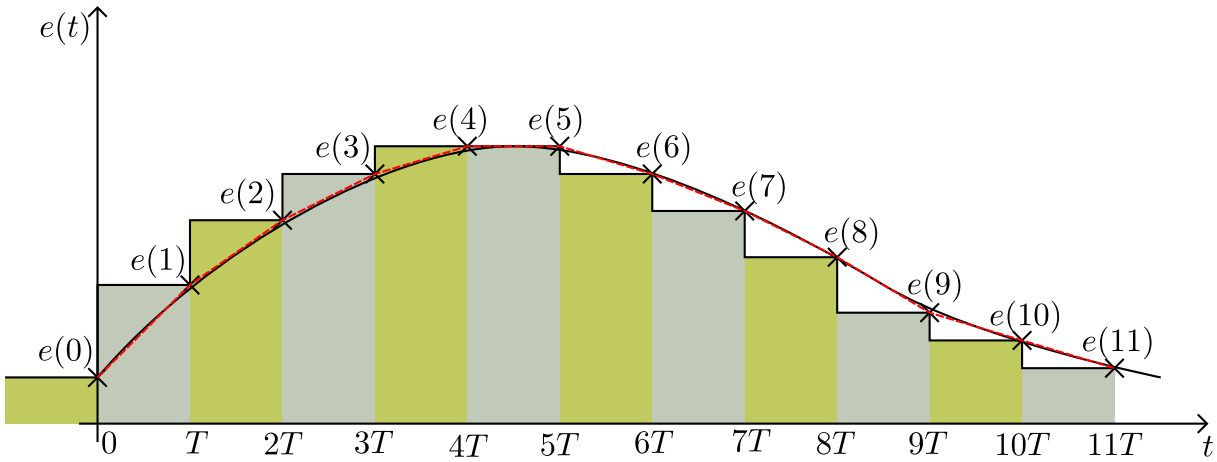
$$u(k) = u_{k,d}(e(k)) + u_{i,d}(e(k)) + u_{d,d}(e(k))$$

P-Anteil

$$
\begin{aligned}
u_{k,a}(e(t)) &= K_a(e(t)) \\
u_{k,d}(e(k)) &= K_d(e(k)) \\
K_d &= K_a
\end{aligned}
$$

I-Anteil

Digitale Berechnung des Integrals



$$\int_0^{11T} e(\tau)d\tau \approx \begin{cases} \sum_{i=0}^{11} e(i)\,T & \text{Rückwärts-Rechteckregel} \\ \sum_{i=0}^{11} \frac{e(i)+e(i-1)}{2}\,T & \text{Trapezregel} \end{cases}$$

I-Anteil

Unbeschränkt Anzahl von Summanden

$\sum_0^k = \sum_0^{k-1} + \alpha(k)$

$$\sum_{i=0}^{11} e(i)\,T = \sum_{i=0}^{10} e(i)\,T + e(11)\,T$$

$$\sum_{i=0}^{11} \frac{e(i)+e(i-1)}{2}\,T = \sum_{i=0}^{10} \frac{e(i)+e(i-1)}{2}\,T + \frac{e(11)+e(10)}{2}\,T$$

I-Anteil

Von Summe zu Differenzengleichung

Implementierbare Differenzgleichungen für I-Anteil

Rückwärts-Rechteckregel:

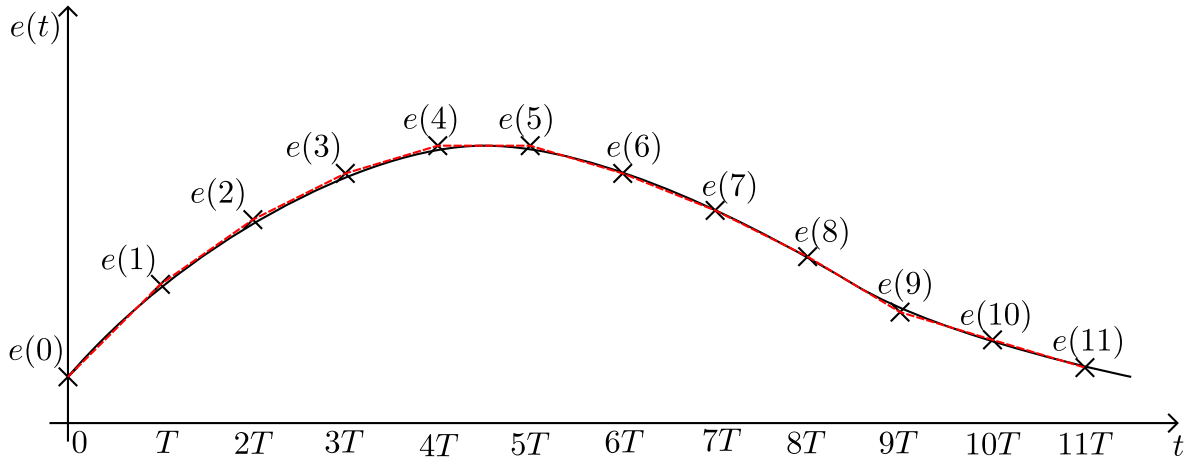
$$u_{i,d,r}(e(k)) = u_{i,d,r}(e(k-1)) + \frac{K_a}{T_{i,a}} e(k)\, T$$

Trapezregel:

$$u_{i,d,t}(e(k)) = u_{i,d,t}(e(k-1)) + \frac{K_a}{T_{i,a}} \frac{e(k) + e(k-1)}{2} T$$

D-Anteil

Digitale Berechnung der Ableitung



$$\dot{e}(t) = \frac{de}{dt}(t) = \lim_{\epsilon \to 0} \frac{e(t) - e(t-\epsilon)}{\epsilon} \approx \frac{e(k) - e(k-1)}{T}$$

D-Anteil

Implementierbare Gleichung für den D-Anteil

$$u_{d,d}(e(k)) = K_a T_{d,a} \frac{e(k) - e(k-1)}{T}$$

Pseudocode

```
// Parameters
K_a T_i T_d T
// Global variables
e_k_1 = 0.0;
u_i_k_i = 0.0;
start_control_task
// Trigger the AD conversion and get the results
// in y_m_k
y_m_k = AD(Y_M_AD_ID);
// Get the reference point
y_s_k = GetReference();
```

Pseudocode

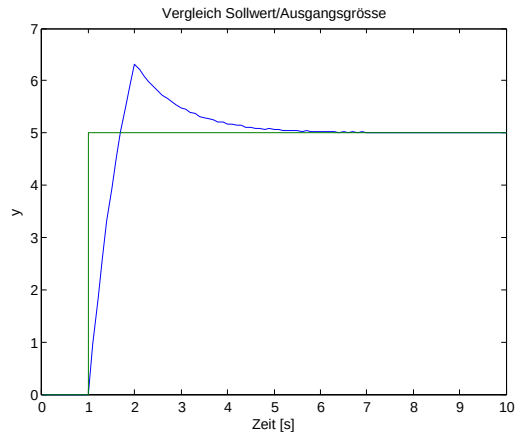
```
// Compute the error
e_k = y_s_k - y_m_k;
// Compute the control command u_k
// Proportional term
u_p_k = K_a * e_k;
// Integral term (trapez)
u_i_k = u_i_k_1 + K_a/T_i*(e_k+e_k_1) / 2.0 * T;
// Derivative term
u_d_k = K_a * T_d * (e_k-e_k_1) / T;
// Total command
u_k = u_p_k + u_i_k + u_d_k;
```

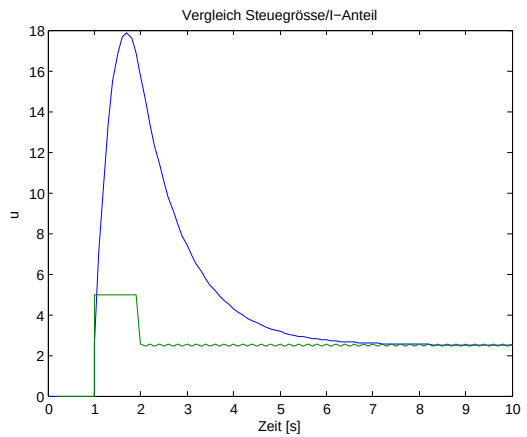
Pseudocode

```
// Save variables for next step
u_i_k_1 = u_i_k; e_k_1 = e_k;
// Trigger the DA conversion
DA(U_K_DA_ID, u_k);
end_control_task
```

1. Direkter / Indirekter Reglerentwurf
2. Implementierung eines digitalen Reglers
3. Diskretisierung des PID Reglers
4. Anti-Reset Windup
   - Einschränkung der Steuergrösse
   - Lösung

Realität der Einschränkung der Steuergrösse $u(t)/u(k)$





Ohne Antireset-Windup

Ohne Antireset-Windup

$$u_i(k) = u_i(k-1) + K_a \frac{T}{T_i} \frac{e(k) + e(k-1)}{2}$$

$$u_{nosat}(k) = u_p(k) + u_i(k) + u_d(k)$$

Ohne Antireset-Windup

Ohne Antireset-Windup

if $u_{nosat}(k) > u_{sat,max}$

$u\_k = u_{sat,max}$

else if $u_{nosat}(k) < u_{sat,min}$

$u(k) = u_{sat,min}$

else

$u(k) = u_{nosat}(k)$

endif

Mit Antireset-Windup

Mit Antireset-Windup (Typ Backcalculation)

$$u_{nosat}(k) = u_p(k) + u_i(k-1) + u_d(k)$$

if $u_{nosat}(k) > u_{sat,max}$
$u(k) = u_{sat,max}$
else if $u_{nosat}(k) < u_{sat,min}$
$u(k) = u_{sat,min}$
else
$u(k) = u_{nosat}(k)$
endif

Mit Antireset-Windup

Mit Antireset-Windup (Typ Backcalculation)

$$u_i(k) = u_i(k-1) + K_a \frac{T}{T_i} \frac{e(k) + e(k-1)}{2} + \frac{T}{T_r}(u(k) - u(k)_{nosat})$$