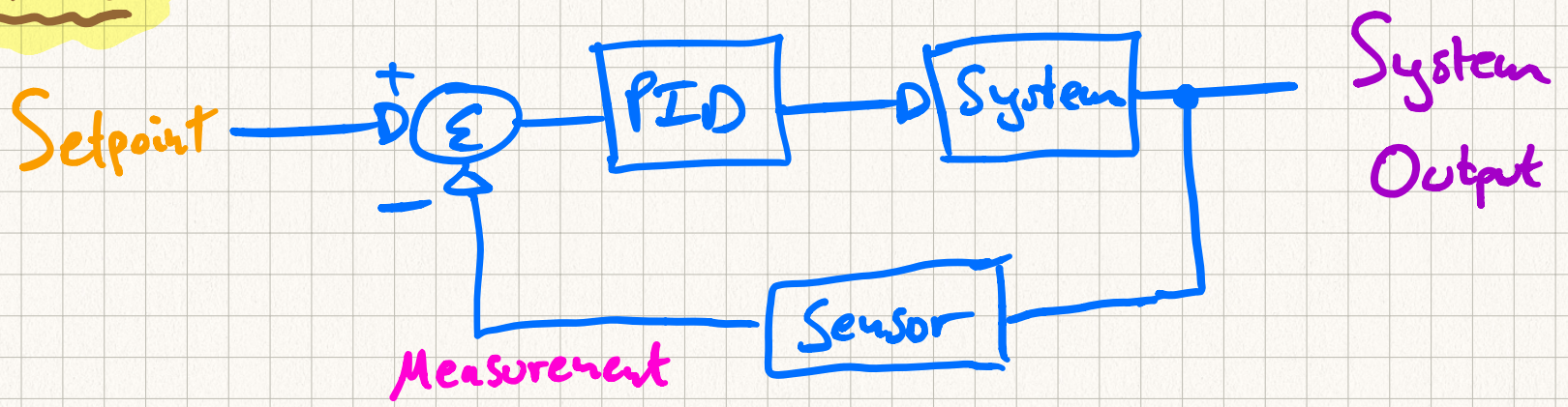
PID Controller Implementation in Software

- Overview.
- Converting from continuous to discrete time.
- Practical considerations.
- PID code in C (for embedded systems).
- Example application.

Overview



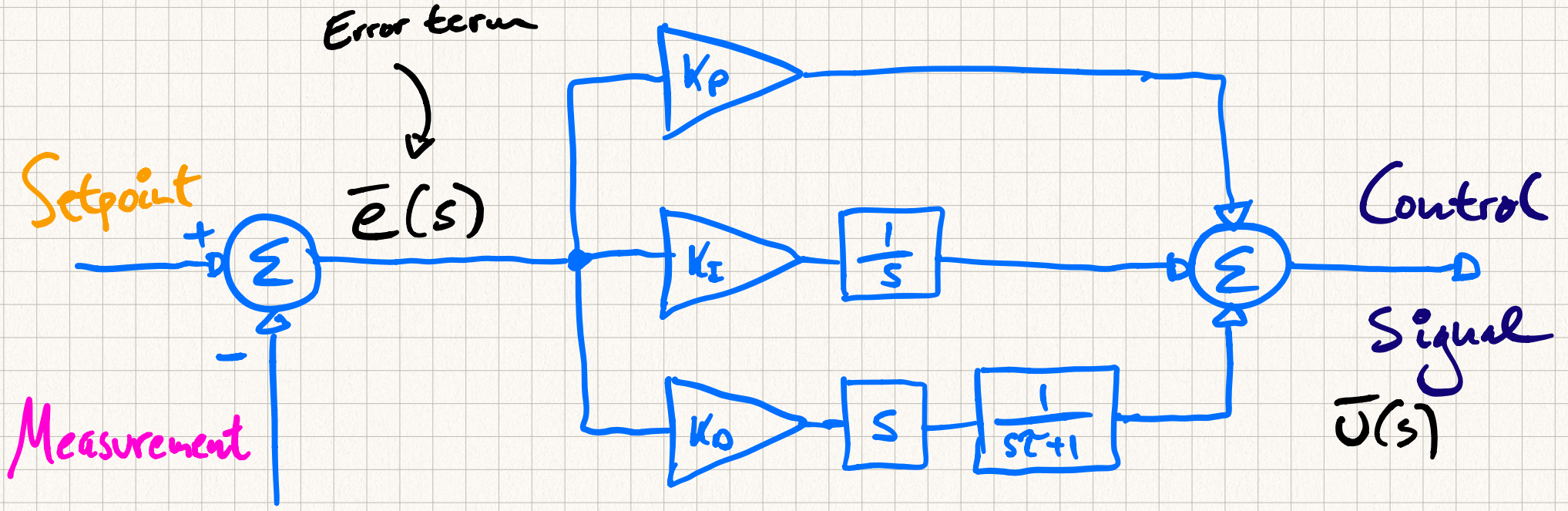
→ Want to make system output track desired setpoint via feedback!

→ PID controller takes <u>two</u> inputs (setpoint, measurement) and produces a control signal.

→ System usually in continuous time, PID controller usually implemented digitally.

⇒ How can we write the controller in code?

PID in Continuous Domain



$$G(s) = \frac{\bar{u}(s)}{\bar{e}(s)} = K_P + K_I \cdot \frac{1}{s} + K_D \cdot \frac{s}{s\tau+1}$$

Proportional Integral (Filtered) Derivative

Continuous to Discrete

Can implement this in code!

→ I.e. S-Domain to z-Domain to difference equation!

→ Use Tustin transform (best frequency domain match)

① Substitute $s \rightarrow \frac{2}{T}\frac{z-1}{z+1}$

② Recall: $\bar{Y}(z) = \bar{X}(z) \cdot z^{-1} \Rightarrow y[n] = x[n-1]$

($T \hat{=}$ sampling time of discrete controller in seconds)

After a lot of substituting and rearranging...

$$p[n] = K_p \cdot e[n] \qquad \text{P}$$

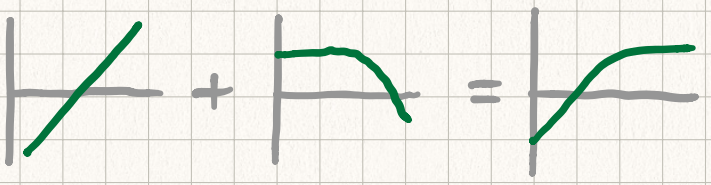
$$i[n] = \frac{K_I T}{2}\left(e[n] + e[n-1]\right) + i[n-1] \qquad \text{I}$$

$$d[n] = \frac{2K_D}{2\tau + T}\left(e[n] - e[n-1]\right) + \frac{2\tau - T}{2\tau + T} d[n-1] \qquad \text{D}$$

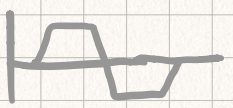
$$\Rightarrow u[n] = p[n] + i[n] + d[n]$$

↑ Controller output

Practical Considerations



- Derivative amplifies HF noise. → Derivative filter!
- Derivative "kick" during setpoint change. → Derivative-on-measurement!
- Integrator can saturate output. → Integrator anti-windup!



- Limits on system input amplitude. → Clamp controller output!
- Choosing a sample time T. → $T_{controller} < \frac{T_{BW,system}}{10}$

Code Structure

- Header-file "library". (#include "pid.h")
- PID controller struct. (Contains gains, storage variables, ...)
- Initialisation function. (set gains, sample time, ...)
- Update function. (provide setpoint and measurement, returns controller output)