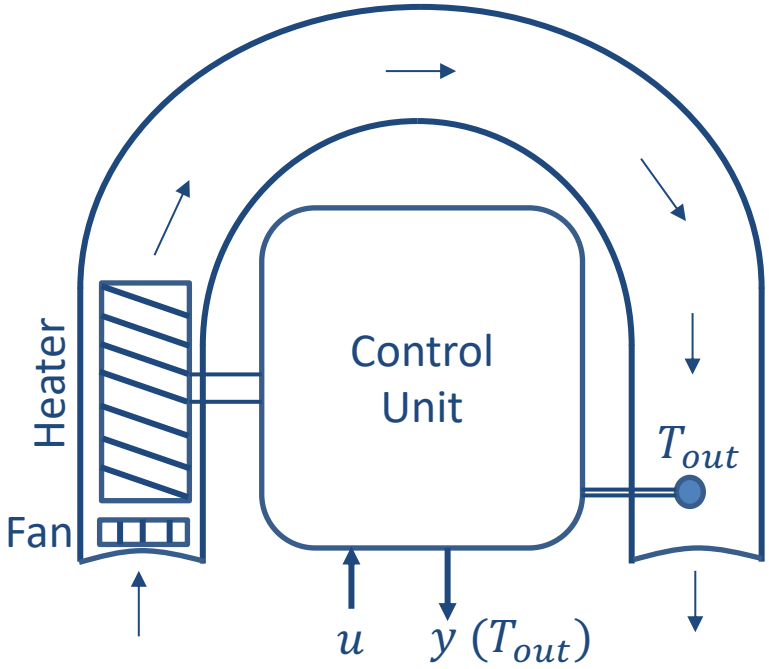
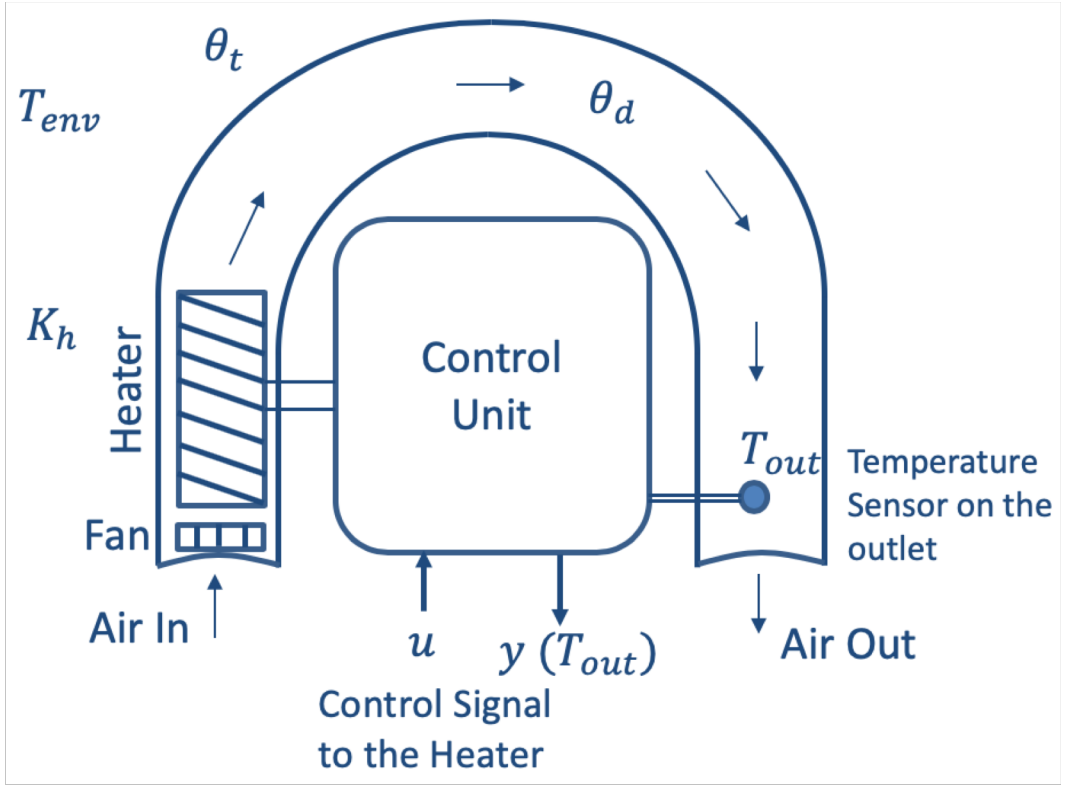
https://www.halvorsen.blog

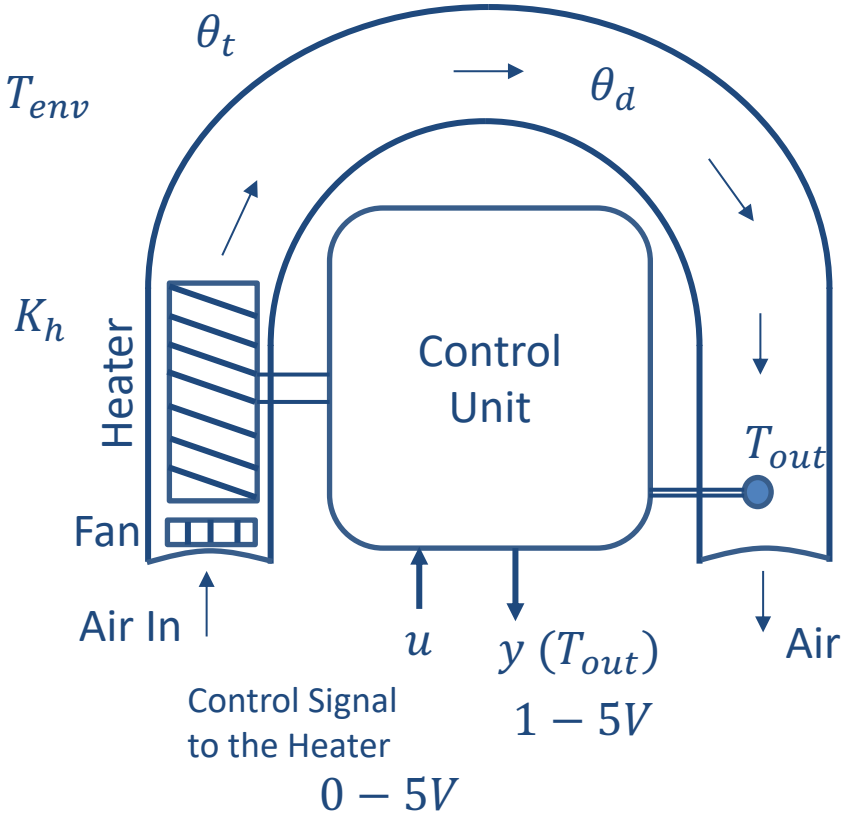
Air Heater System

Hans-Petter Halvorsen


Heater
Fan
Control
Unit
T_{out}
u
$y\ (T_{out})$


θ_t
T_{env}
θ_d
K_h
Heater
Control
Unit
T_{out}
Temperature
Sensor on the
outlet
Fan
Air In
u
$y\ (T_{out})$
Air Out
Control Signal
to the Heater

Air Heater System

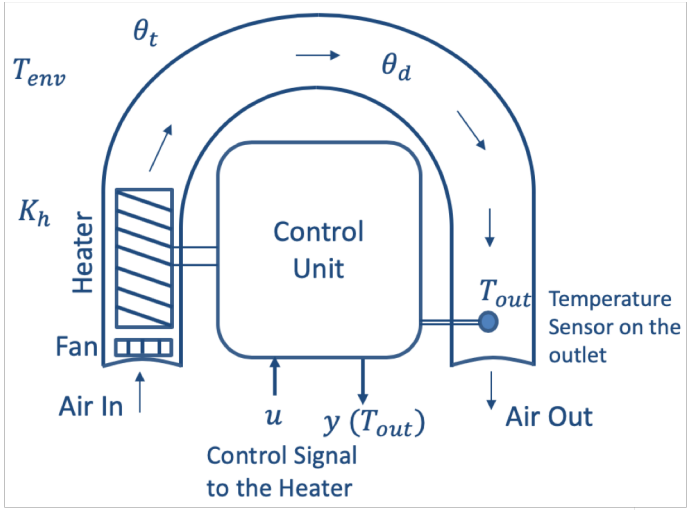


Aim:
Control the Temperature on the outlet (T_{out})

Air Heater System

The system can be modelled as a 1. order system with time-delay

$$\dot{T}_{out} = \frac{1}{\theta_t}\{-T_{out} + [K_h u(t-\theta_d) + T_{env}]\}$$



- T_{out} is the air temperature at the tube outlet
- $u\ [V]$ is the control signal to the heater
- $\theta_t\ [s]$ is the time-constant
- $K_h\ [deg\ C\ /\ V]$ is the heater gain
- $\theta_d\ [s]$ is the time-delay representing air transportation and sluggishness in the heater
- T_{env} is the environmental (room) temperature. It is the temperature in the outlet air of the air tube when the control signal to the heater has been set to zero for relatively long time (some minutes)

Model Values

You can assume the following values in your simulations:

$$\theta_t = 22\ sec$$

$$\theta_d = 2\ sec$$

$$K_h = 3.5\ \frac{°C}{V}$$

$$T_{env} = 21.5\ °C$$

The range for T_{out} could, e.g., be $20°C \leq T_{out} \leq 50°C$

https://www.halvorsen.blog

Air Heater in LabVIEW

Hans-Petter Halvorsen

Air Heater in LabVIEW

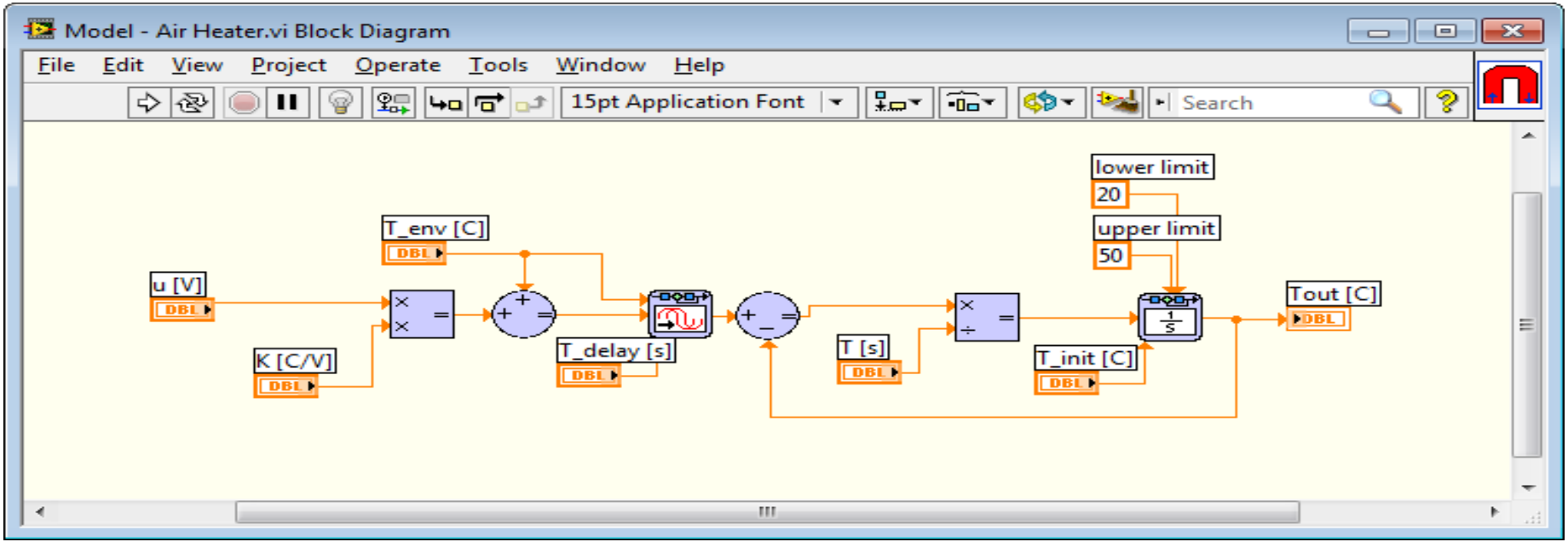
You can implement the Air Heater in LabVIEW in different ways:

- The model can be implemented using the blocks (Integrator, Summation, Multiplication, etc.) from the Simulation palette in LabVIEW (LabVIEW Control Design and Simulation Module)
- Create a Discrete version of the differential equation (use e.g., Euler Forward). Then use the Formula Node, MathScript Node or MATLAB Node inside LabVIEW

You should test both these alternatives

Air Heater Model in LabVIEW

LabVIEW Control Design and Simulation Module

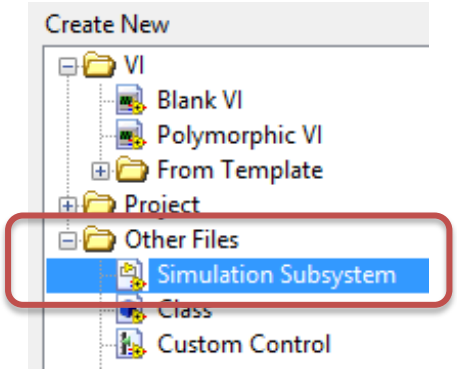


Note! This model is implemented in a so-called "**Simulation Subsystem**" (which is recommended!!!)

Simulation Subsystem

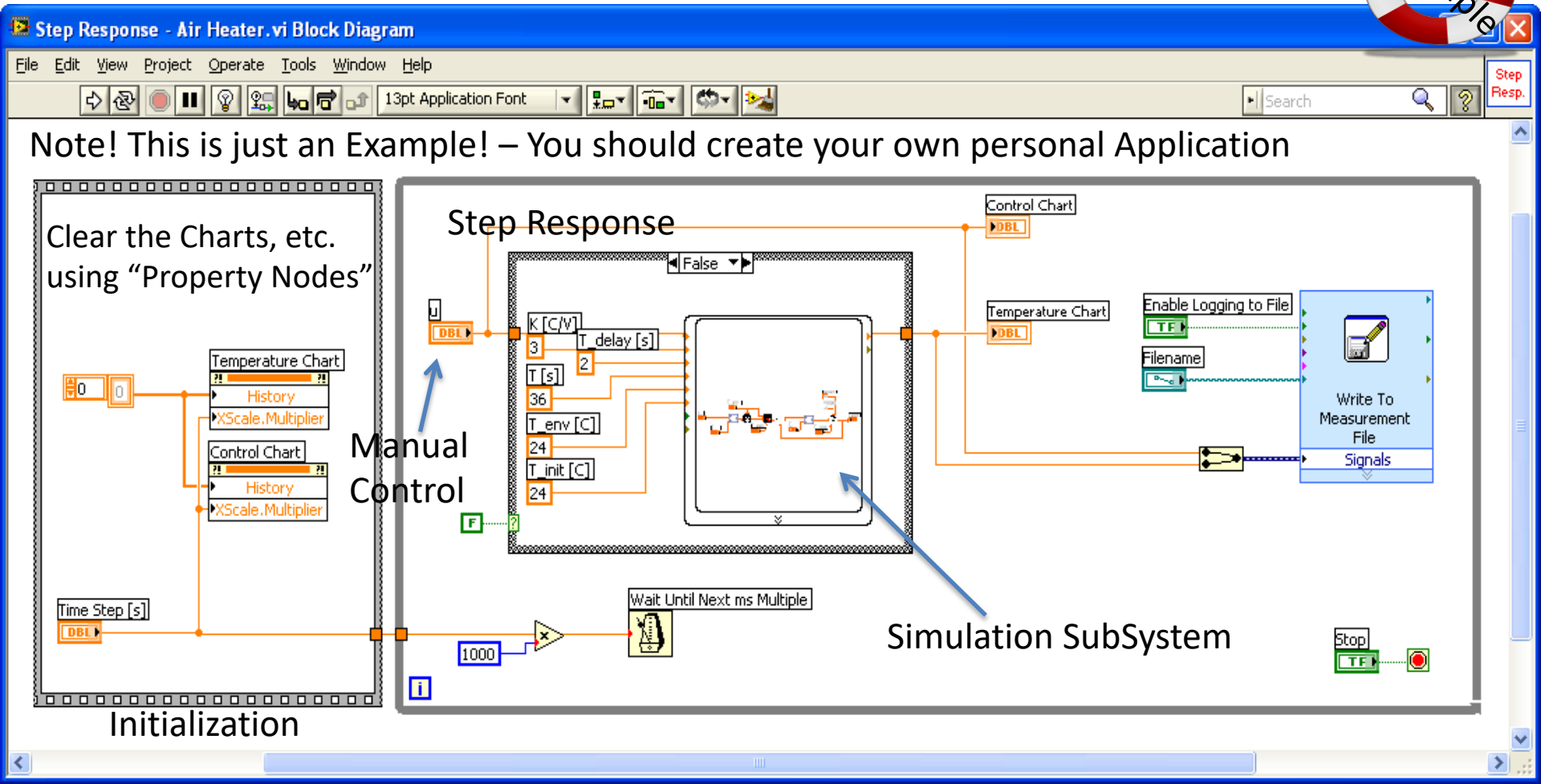
A Way to structure your code, similar to SubVIs

This is the recommended way to do it! – You can easily reuse your Subsystems in different VIs and your code becomes more structured!



Select File -> New ..., Then choose “Simulation Subsystem”.
Create your Model within the Simulation Subsystem

Model Simulation Example (Note! No PID Control in this Example)



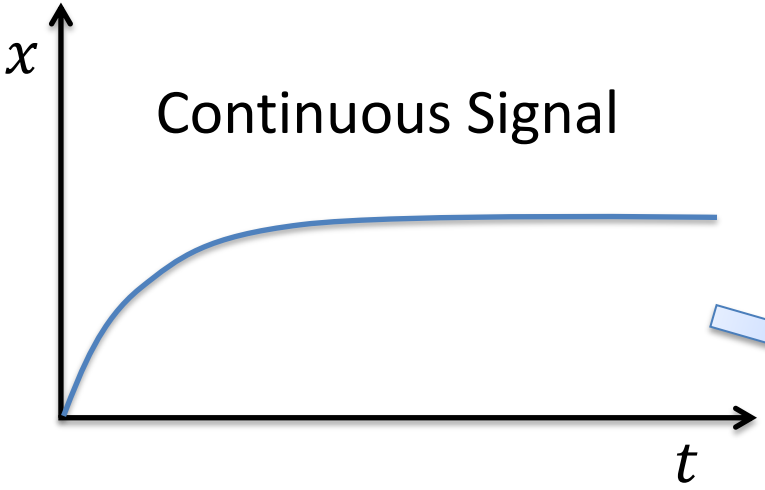
https://www.halvorsen.blog

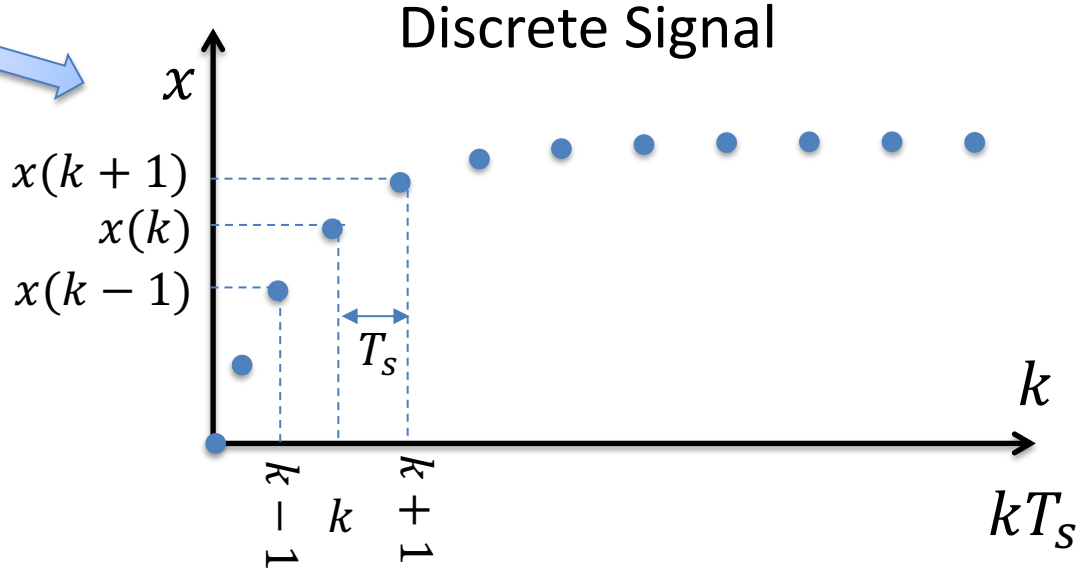
Discretization

Hans-Petter Halvorsen

Continuous vs. Discrete Systems

A computer can only deal with discrete signals





T_s - Sampling Interval

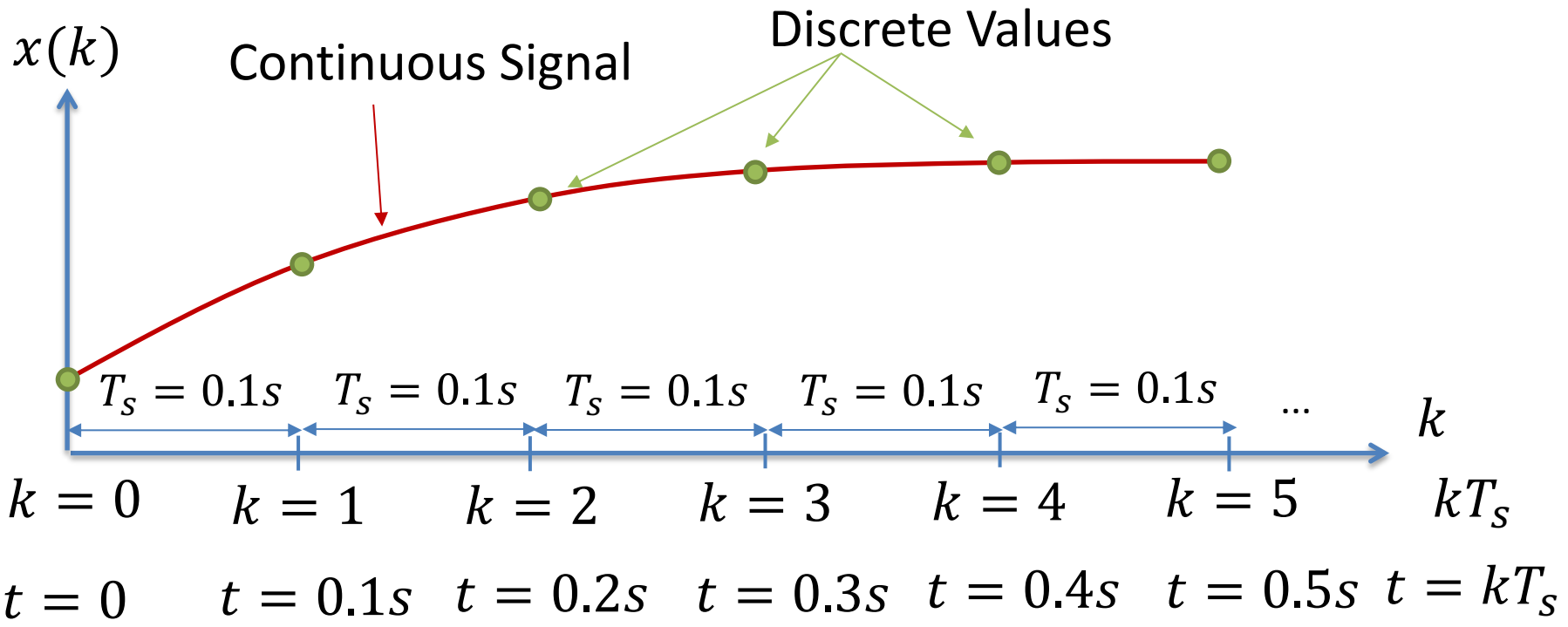
$x(k-1)$ - Previous Value

$x(k)$ - Current Value

$x(k-1)$ - Next Value

Continuous vs. Discrete Systems - Example

In this Example we have used Sampling Interval $T_s = 0.1s$



Discretization of Air Heater

Continuous Model:

$$\dot{T}_{out} = \frac{1}{\theta_t}\{-T_{out} + [K_h u(t-\theta_d) + T_{env}]\}$$

We can use e.g., the Euler Approximation in order to find the discrete Model:

$$\dot{x} \approx \frac{x(k+1) - x(k)}{T_s}$$

T_s - Sampling Time

$x(k)$ - Present value

$x(k+1)$ - Next (future) value

The discrete Model will then be on the form:

$$x(k+1) = x(k) + \ \ldots$$

We can then implement the discrete model in any programming language

Discretization of Air Heater

We make a discrete version:

$$\dot{T}_{out} = \frac{1}{\theta_t}\{-T_{out} + [K_h u(t-\theta_d) + T_{env}]\}$$

$$\frac{T_{out}(k+1) - T_{out}(k)}{T_s} = \frac{1}{\theta_t}\{-T_{out}(k) + [K_h u(k-\theta_d) + T_{env}]\}$$

This gives the following discrete system:

$$T_{out}(k+1) = T_{out}(k) + \frac{T_s}{\theta_t}\{-T_{out}(k) + [K_h u(k-\theta_d) + T_{env}]\}$$

The Time delay θ_d makes it a little complicated. We can simply by setting $\theta_d = 0$

$$T_{out}(k+1) = T_{out}(k) + \frac{T_s}{\theta_t}\{-T_{out}(k) + [K_h u(k) + T_{env}]\}$$

Discretization of Air Heater

Discrete version with Time delay $\theta_d = 0$

$$T_{out}(k+1) = T_{out}(k) + \frac{T_s}{\theta_t}\{-T_{out}(k) + [K_h u(k) + T_{env}]\}$$

We can use the following values in the simulation:

$\theta_t = 22$
$K_h = 3.5$
$T_{env} = 21.5$

We can set the Sampling Time $T_s = 0.1s$

Discretization of Air Heater with Time Delay

We have the following discrete system:

$$T_{out}(k+1) = T_{out}(k) + \frac{T_s}{\theta_t}\{-T_{out}(k) + [K_h u(k-\theta_d) + T_{env}]\}$$

The Time delay θ_d makes it more complicated to implement

θ_d is in seconds and we need to convert it to discrete intervals in forms of k

The discrete version of θ_d is: $\frac{\theta_d}{T_s}$

Then we get:

$$T_{out}(k+1) = T_{out}(k) + \frac{T_s}{\theta_t}\left\{-T_{out}(k) + \left[K_h u\left(k - \frac{\theta_d}{T_s}\right) + T_{env}\right]\right\}$$

Assuming $\theta_d = 2s$ and $T_s = 0.1s$ we get $u(k-20)$

This mean that we have to remember the 20 previous samples of $u(k)$

https://www.halvorsen.blog

Control System in LabVIEW

Hans-Petter Halvorsen

The PID Algorithm

$$u(t) = K_p e + \frac{K_p}{T_i}\int_0^t e d\tau + K_p T_d \dot{e}$$

Where u is the controller output and e is the control error:

$$e(t) = r(t) - y(t)$$

r is the Reference Signal or Set-point

y is the Process value, i.e., the Measured value

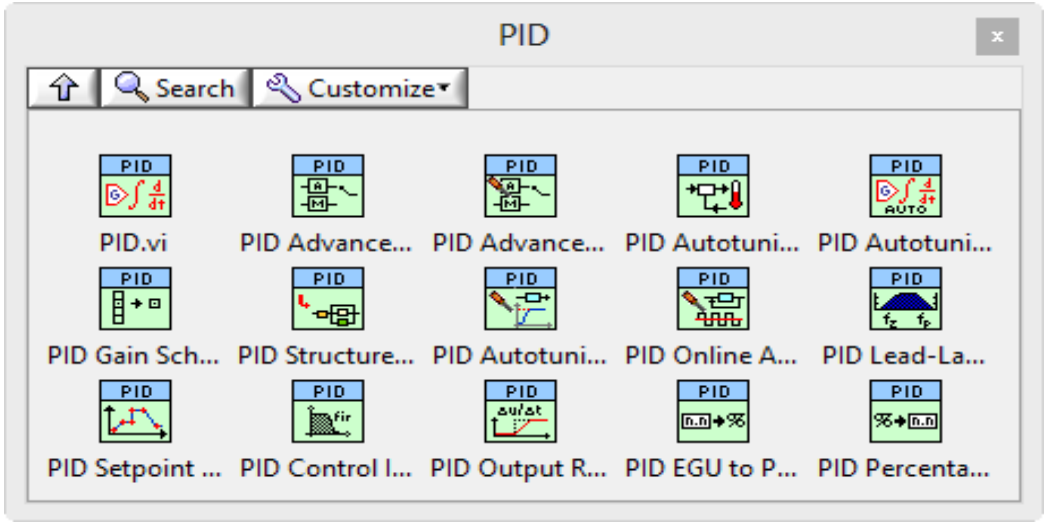
Tuning Parameters:

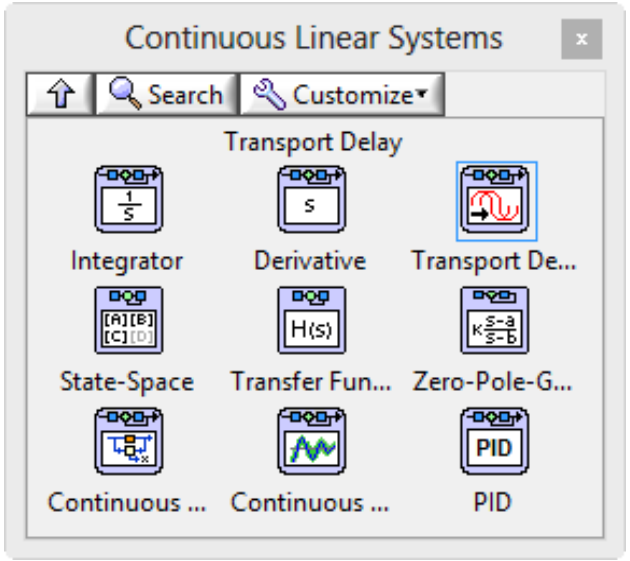
K_p Proportional Gain

T_i Integral Time [sec.]

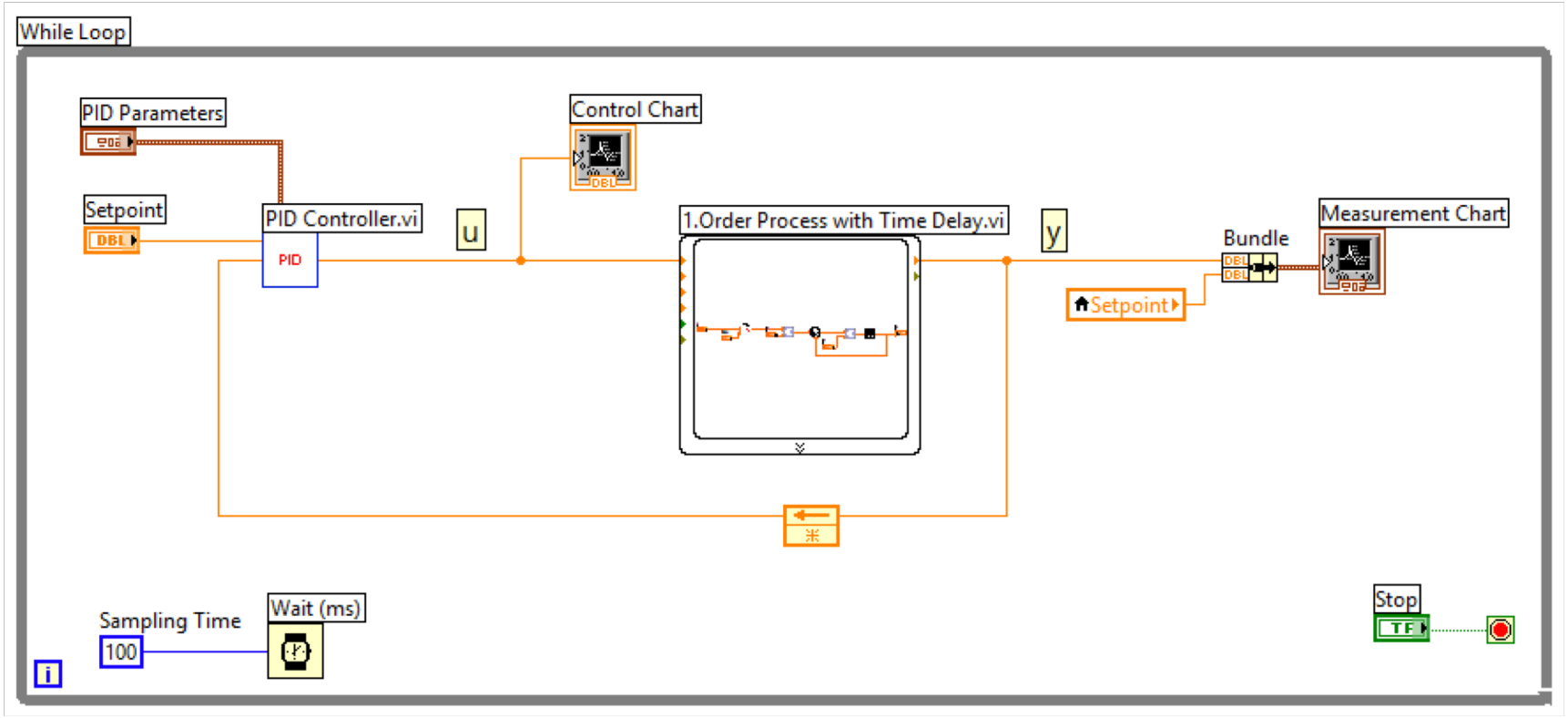
T_d Derivative Time [sec.]

Built-in PID in LabVIEW



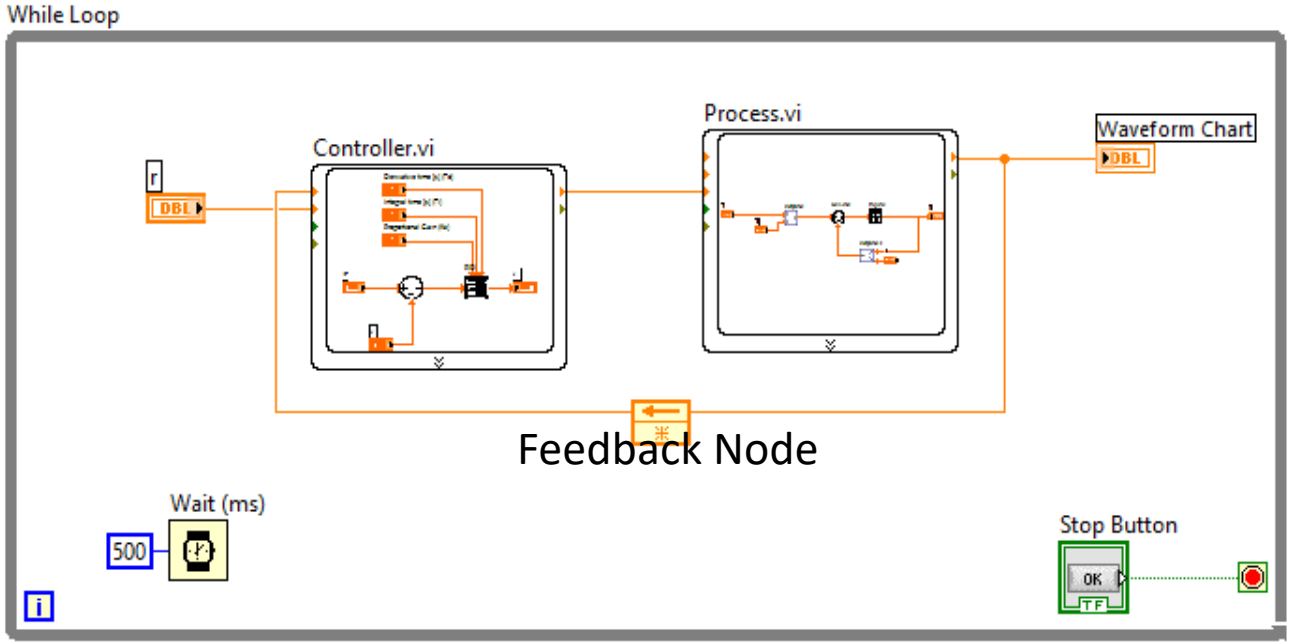


Example of Control System in LabVIEW

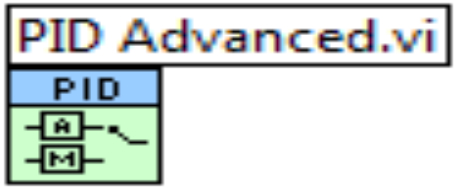


PID Control of Model in LabVIEW

The Simulation Loop has some drawbacks/is more complicated to use than an ordinary While Loop. If we use Simulatation Subsystems, we can use them inside a While Loop instead! - which becomes very handy!

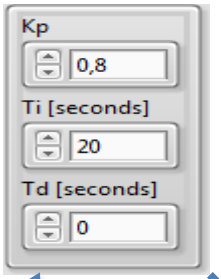


For real applications that involves more than just simulations (such as DAQ, File Logging, PID control of the real process, etc.), I recommend to use a While Loop instead of a Simulation Loop.



PID in LabVIEW

Front Panel

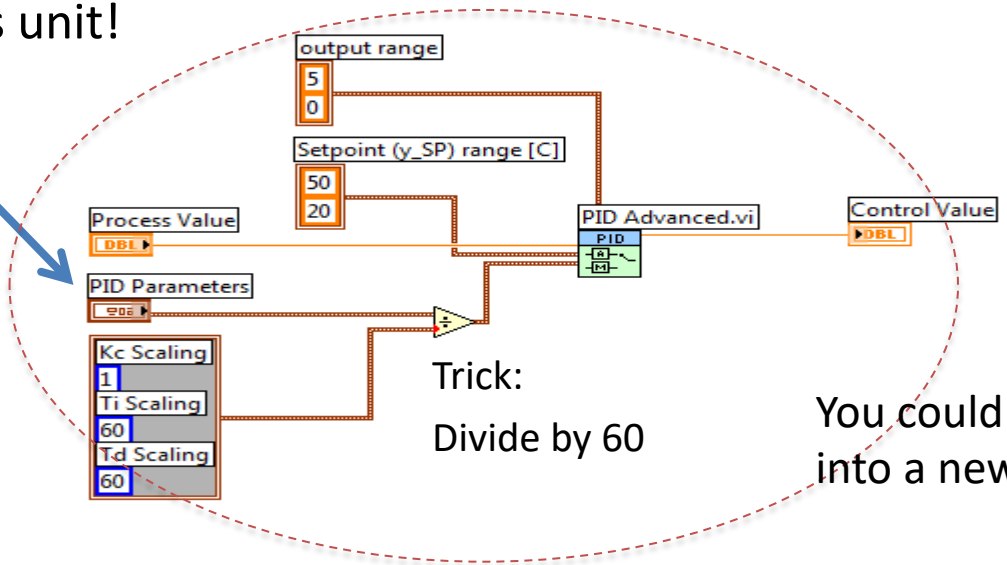


Cluster

Normally we use <u>seconds</u> as unit for Ti and Td (which is recommended!)

But the built-in PID algorithm in LabVIEW uses <u>minutes</u> as unit!

Block Diagram:



Trick:

Divide by 60

You could also put this code into a new SubVI

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog