Design and Implementation of Control System

HANS-PETTER HALVORSEN, 2013.08.05
# Table of Contents

Table of Contents ......................................................................................................................... ii
1  Introduction................................................................................................................................. 3
2  Modeling and Simulation .............................................................................................................. 4
   2.1  Introduction ............................................................................................................................ 4
   2.2  Transfer function .................................................................................................................... 5
   2.3  Model Adaptation in LabVIEW ............................................................................................ 5
   2.4  MathScript ............................................................................................................................. 7
   2.5  LabVIEW ................................................................................................................................ 7
3  Frequency Response .................................................................................................................... 10
   3.1  Frequency Response from sinusoidal input and output signals ........................................ 11
4  Design and Analysis .................................................................................................................... 13
   4.1  PI Controller Design .............................................................................................................. 13
      4.1.1  Skogestad’s method ........................................................................................................ 13
   4.2  Frequency Response and Stability Analysis ........................................................................ 14
      4.2.1  Analysis of the Feedback System .................................................................................. 14
      4.2.2  System Bandwidths ......................................................................................................... 16
      4.2.3  Stability Analysis ............................................................................................................ 16
      4.2.4  Stable vs. Unstable System ............................................................................................. 17
5  Control System ............................................................................................................................ 18
   5.1  Discrete Low-pass Filter ........................................................................................................ 18
   5.2  Discrete PI Controller .......................................................................................................... 19
   5.3  Control System Implementation .......................................................................................... 19
Introduction

In this project we will design and implement a control system for a small scale industrial process, called Air Heater.

Below we see a sketch of the final system we will end up with. The control system shall be implemented in LabVIEW.

Before we can start using the real air heater process, we need to create a simulator of the system. We start by creating a mathematical model of the system which we will implement in the simulator.

We will also analyze and design the control system using MathScript. We will find the transfer function for the process and for the feedback system (including PID controller), which we will use in the design and analysis phase.

Based on frequency response and stability analysis, we will find proper PID parameters for the system.

When we have found proper PID parameters, we will simulate the system and check if the performance is as expected.

Finally, we will implement our control system for the real Air Heater process to verify the performance and stability of the system.
Modeling and Simulation

2.1 Introduction

In order to design our control system we need to start with a mathematical model of an Air Heater system.

A simple mathematical model of the system could be:

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

Where:

- \(T_{out}[^\circ C]\) 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
- \(T_{env}[^\circ C]\) 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)
**Heater**: The air is heated by an electrical heater. The supplied power is controlled by an external voltage signal in the range \(0 - 5 \text{ V}\) (min power, max power).

**Temperature sensors**: Two Pt100 temperature elements are available (some of the Air Heaters have only one). You can use Temperature sensor 1 in this assignment. The range is \(1 - 5 \text{ V}\), and this voltage range corresponds to the temperature range \(20 - 50^\circ \text{C}\) (with a linear relation).

### 2.2 Transfer function

Since much of the control design theory is based on transfer functions, we need to find the transfer function \(H(s)\) for the Air Heater process based on the given differential equation.

**Tip!** Use **Laplace** transformation on the differential equation for the Air Heater and find the transfer function from \(u(s)\) to \(T_{\text{out}}(s)\).

\[
T_{\text{out}} = \frac{1}{\theta_t} \{ -T_{\text{out}} + [K_h (t - \theta_d) + T_{\text{env}}] \}
\]

The Air Heater process is a 1.\text{order} process with time-delay, so a transfer function on the following general form should be expected:

\[
H(s) = \frac{y(s)}{u(s)} = \frac{K}{T s + 1} e^{-\frac{t}{T}}
\]

### 2.3 Model Adaptation in LabVIEW

In the assignment we will use a **USB-6008 DAQ** unit in order to read data from the process \(y\) to the PC, and write data \(u\) from the PC to the process.

**DAQ Assistant:**

In order to communicate with the USB-6008 DAQ device within LabVIEW, we can use the DAQ Assistant. Note! The **NI-DAQmx** software needs to be installed.
The DAQ Assistant is located from the Functions palette: “Measurement I/O → NI-DAQmx → DAQ Assist”.

**Analog In Example:**
Below we see a typical example in LabVIEW where we read from the “Analog Input” port of the DAQ device:

![Analog In Example Image]

**Analog Out Example:**
Below we see a typical example in LabVIEW where we read from the “Analog Output” port of the DAQ device:

![Analog Out Example Image]

The Air Heater processes available are slightly different, so we need to do some practical experiments in order to find the unknown model parameters \((\theta_t, K_h, \theta_d)\) for your Air Heater process.

To do that we need the following equipment:

- **Air Heater Process**
- **USB-6008 DAQ Device**
There are several ways to find the unknown model parameters ($\theta_t, K_n, \theta_d$). Running a step response on the real process using LabVIEW is a simple and effective method.

The procedure is as follows:

Based on the transfer function found in the previous task and by plotting the step response for the temperature $T_{out}$, we can find the unknown model parameters ($\theta_t, K_n, \theta_d$) directly from the plot as illustrated above.

Note! Don’t spend too much time on this, but go back an fine-tune the parameters later if necessary.

### 2.4 MathScript

Since we are going to use MathScript in the analysis phase, create frequency responses, etc, we need to create and implement the transfer function of the Air Heater in MathScript as well.

You should also perform a step response to find our more about the dynamics in the system.

You may implement the transfer function using the `sys_order1()` function or/and a Pade’ approximation (e.g., use the built-in `pade()` function together with the `tf()` function).

Use the model parameters ($\theta_t, K_n, \theta_d$) found previously.

You should also find the Poles and the Zeros for the transfer function in MathScript.

Do you get the same results here compared to the previous task?

Compare and discuss the results.

### 2.5 LabVIEW
We need also to implement the model in LabVIEW as well. In LabVIEW we can easily implement a model based on the differential equation. In LabVIEW we can create a block diagram of the differential equation.

The model should be implemented using the blocks (Integrator, Transport Delay, Summation, Multiplication, etc.) from the Simulation palette in LabVIEW:

**Tip!** Draw a block diagram of the system using pen and paper before you start to implement the system in LabVIEW.

Simulate the model and show the output temperature $T_{out}$ in a plot after a step in the control signal $u$.

Use the model parameters found in a previous task in the simulations.

You should also validate the model by running the model in parallel with the real system as shown below:
By plotting the output $T_{out}$ for both the real process and the simulated process in the same plot, we can easily see if the model is good or not.

You should discuss the results.
3 Frequency Response

The frequency response of a system is a frequency dependent function which expresses how a sinusoidal signal of a given frequency on the system input is transferred through the system. Each frequency component is a sinusoidal signal having certain amplitude and a certain frequency.

The frequency response is an important tool for analysis and design of signal filters and for analysis and design of control systems. The frequency response can be found experimentally or from a transfer function model.

The frequency response of a system is defined as the steady-state response of the system to a sinusoidal input signal. When the system is in steady-state, it differs from the input signal only in amplitude/gain ($A$) and phase lag ($\phi$).

Given the model of the Air Heater:

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

In a previous task you have found the transfer function for the Air Heater on the following form:

$$H(s) = \frac{T_{\text{out}}(s)}{u(s)} = \frac{K}{Ts + 1} e^{-\tau s}$$

Use values for $K_h, \theta_d, \theta_t$ from a previous task.

We need to plot the Frequency Response in a Bode plot (use, e.g., the bode() function in MathScript). What is the break frequency?

Find $A(\omega)$ and $\phi(\omega)$ for the frequencies given below using MathScript code (use the bode() function).

<table>
<thead>
<tr>
<th>$\omega$ [rad/s]</th>
<th>$A(\omega)$</th>
<th>$\phi(\omega)$ [degrees]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
We also need to find the mathematical expressions for $A(\omega)$ [$dB$] and $\phi(\omega)$. You typically use pen and paper for this.

Find $A(\omega)$ and $\phi(\omega)$ for the same frequencies above using the mathematical expressions for $A(\omega)$ and $\phi(\omega)$. Tip: Use a For Loop or/and define a vector, e.g., $w = [0.01, 0.1, ...]$.

It is recommended to use the `semilogx()` function in order to plot the Bode diagram based on these values.

You should compare and discuss the results.

### 3.1 Frequency Response from sinusoidal input and output signals

We can find the frequency response of a system by exciting the system with a sinusoidal signal of amplitude $A$ and frequency $\omega$ [rad/s] (Note: $\omega = 2\pi f$) and observing the response in the output variable of the system.

In a previous task you have found the transfer function for the Air Heater on the following form:

$$ H(s) = \frac{T_{out}(s)}{u(s)} = \frac{K}{Ts + 1} e^{-ts} $$

Use values for $K_h, \theta_d, \theta_t$ from a previous task.

We will use MathScript to find the Frequency Response of the model. You may e.g., use the `lsim()` function in MathScript.

The input signal is given by:

$$ u(t) = U \sin \omega t $$

The steady-state output signal will then be:

$$ y(t) = UA \sin (\omega t + \phi) $$

The gain is given by:

$$ A = \frac{Y}{U} $$

The phase lag is given by:
\[ \phi = -\omega \Delta t \, [rad] \]

Plot \( u \) and \( y \) in the same plot and find \( Y \) and \( \Delta t \) from the plots and find \( A \) and \( \phi \) using the formulas above.

You will get plots like this for each frequency:

![Plot of input and output signals](image)

From the plot we can find \( Y \) and \( \Delta t \) then we use the formulas above to find \( A \) and \( \phi \) for the specific frequency.

→ You should select 2-3 frequencies (\( \omega \)) where you find \( A \) and \( \phi \) for these frequencies.

Fill in your results in a table like this:

<table>
<thead>
<tr>
<th>( \omega )</th>
<th>( A , [dB] )</th>
<th>( \phi , [^\circ] )</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

You should plot the values found in a Bode diagram.

In MathScript you can use the `semilogx()` function in combination with the `subplot()` function.

**Note!** You shall not use the `bode()` function here!

You should compare and discuss the results.
4 Design and Analysis

4.1 PI Controller Design

Using Skogestad’s method in order to find proper PI parameters for the Air Heater system based on the model of the system is recommended, but other methods may be used as well.

4.1.1 Skogestad’s method

In this task we assume the following process:

\[ H(s) = \frac{T_{out}(s)}{u(s)} = \frac{K}{Ts + 1} e^{-\tau s} \]

Use values for \( K_h, \theta_d, \theta_t \) from a previous task.

The Skogestad’s method assumes you apply a step on the input \((u)\) and then observe the response and the output \((y)\), as shown below:

If we have a model of the system (which we have in our case), we can use the following Skogestad’s formulas for finding the PI(D) parameters directly:

<table>
<thead>
<tr>
<th>Process type</th>
<th>( H_{psf}(s) ) (process)</th>
<th>( K_p )</th>
<th>( T_i )</th>
<th>( T_d )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrator + delay</td>
<td>( \frac{K}{s} e^{-\tau s} )</td>
<td>( \frac{1}{K(T_C + \tau)} )</td>
<td>( c(T_C + \tau) )</td>
<td>0</td>
</tr>
<tr>
<td>Time-constant + delay</td>
<td>( \frac{K}{Ts+1} e^{-\tau s} )</td>
<td>( \frac{1}{K(T_C + \tau)} )</td>
<td>( \min{T, c(T_C + \tau)} )</td>
<td>0</td>
</tr>
<tr>
<td>Integr + time-const + del.</td>
<td>( \frac{K}{(Ts+1)s} e^{-\tau s} )</td>
<td>( \frac{1}{K(T_C + \tau)} )</td>
<td>( c(T_C + \tau) )</td>
<td>( T )</td>
</tr>
<tr>
<td>Two time-const + delay</td>
<td>( \frac{K}{(Ts+1)(Ts+1)} e^{-\tau s} )</td>
<td>( \frac{1}{K(T_C + \tau)} )</td>
<td>( \min{T_1, c(T_C + \tau)} )</td>
<td>( T_2 )</td>
</tr>
<tr>
<td>Double integrator + delay</td>
<td>( \frac{K}{s^2} e^{-\tau s} )</td>
<td>( \frac{1}{4K(T_C + \tau)^2} )</td>
<td>( 4(T_C + \tau) )</td>
<td>( 4(T_C + \tau) )</td>
</tr>
</tbody>
</table>
**Tip!** We can e.g., set \( T_C = 10 \text{ s} \) and \( c = 1.5 \) (or try with other values if you get poor PI parameters).

For more details about the Skogestads method, please read this article: “[Model-based PID tuning with Skogestad’s method](#)”.  

## 4.2 Frequency Response and Stability Analysis

Here we will analyze the stability of the system using frequency response methods. We will use MathScript for this purpose.

### 4.2.1 Analysis of the Feedback System

Below we see the block diagram of the feedback system:

![Block Diagram of Feedback System](image)

**Process (Air Heater):**

The transfer function for the process is as follows:

\[
H_p(s) = \frac{T(s)}{u(s)} = \frac{K}{Ts + 1} e^{-Ts}
\]

Use values for \( K, \theta_d, \theta_t \) from a previous task.

**PI controller:**

The PI controller is defined as:

\[
u(t) = K_p e + \frac{K_p}{T_i} \int_0^t e \, dt\]

We need to find the transfer function for the PI Controller:
Design and Analysis

\[ H_c(s) = \frac{u(s)}{e(s)} \]

**Tip!** Use Laplace on the equation above.

You should also plot the Frequency Response for the PI controller \((H_c(s))\) in a Bode plot.

Use values for \(K_p\) and \(T_i\) found previously.

**Loop transfer function: \(L(s)\)**

We also need to find the Loop transfer function \(L(s)\) (Pen & Paper) and define \(L(s)\) using MathScript.

The Loop transfer function is defined as:

\[ L(s) = H_c H_p \]

**Tip!** Use the built-in function `series` in MathScript.

**Tracking transfer function: \(T(s)\)**

We also need to find the Tracking transfer function \(T(s)\) (Pen & Paper) and define \(T(s)\) using MathScript.

The Tracking transfer function is defined as:

\[ T(s) = \frac{y(s)}{r(s)} = \frac{L(s)}{1 + L(s)} \]

**Tip!** Use the built-in function `feedback` in MathScript.

**Sensitivity transfer function: \(S(s)\)**

We also need to find the Sensitivity transfer function \(S(s)\) (Pen & Paper) and define \(S(s)\) using MathScript.

The Sensitivity transfer function is defined as:

\[ S(s) = \frac{e(s)}{r(s)} = \frac{1}{1 + L(s)} = 1 - T(s) \]
4.2.2 System Bandwidths

You should plot the Loop transfer function $L(s)$, the Tracking transfer function $T(s)$ and the Sensitivity transfer function $S(s)$ in the same Bode diagram. Use, e.g., the `bodemag()` function in MathScript (only the gain diagram is of interest in this case, not the phase diagram).

Use the values for $K_p$ and $T_i$ found in a previous Task.

You need to find the different bandwidths $\omega_p, \omega_c, \omega_s$ (see the sketch below).

Discuss the results.

4.2.3 Stability Analysis

You also need to find the crossover-frequencies ($\omega_{180}, \omega_c$) and stability margins GM ($A(\omega)$), PM ($\phi(\omega)$) of the system ($L(s)$) from the Bode plot. Use the `bode()` function in MathScript and find the values from the Bode plot as illustrated below:
Plot also Bode diagram where the crossover-frequencies, GM and PM are illustrated. **Tip!** Use the `margin()` function in **MathScript**.

Use also the `margin()` function in order to find values for $\omega_{180}, \omega_c, A(\omega), \phi(\omega)$ directly.

You should compare and discuss the results.

**How much can you increase $K_p$ before the system becomes unstable?**

### 4.2.4 Stable vs. Unstable System

You should find and use different values of $K_p$ where you get a **marginally stable system**, an **asymptotically stable system** and an **unstable system**.

Plot the time response for the tracking function using, e.g., use the `step()` function in MathScript for all these 3 categories. How can we use the step response to determine the stability of the system?

Find $\omega_{180}, \omega_c, A(\omega)$ and $\phi(\omega)$ in all 3 cases. How can we use $\omega_c$ and $\omega_{180}$ to determine the stability of the system?

Find and plot the poles and zeros for the system for all the 3 categories mentioned above. How can we use the poles to determine the stability of the system?

Plot the Loop transfer function $L(s)$, the Tracking transfer function $T(s)$ and the Sensitivity transfer function $S(s)$ in a Bode diagram for the system for all the 3 categories mentioned above.

You should discuss the results.
5 Control System

Here we will create our own discrete low-pass filter and our own discrete PI controller. We will test the low-pass filter and the discrete controller on a model of the Air Heater system created in a previous task.

This is a typical block diagram of the system:

![Block Diagram](image)

5.1 Discrete Low-pass Filter

Transfer function for a first-order low-pass filter may be written:

\[ H(s) = \frac{1}{T_f s + 1} \]

Where \( T_f \) is the time-constant of the filter.

We need to create a **discrete low-pass filter** in LabVIEW using the **Formula Node** in LabVIEW. Create a **SubVI** of the code. The user needs to be able to set the time constant of the filter \( T_f \) from the outside, i.e., it should be an input to the SubVI. The simulation Time-step \( T_s \) needs also to be set from the outside.

Use the **Euler Backward** method:
Perform simulations to make sure the filter works as expected. Explain/Show how you do this. Why do we use a low-pass filter?

5.2 Discrete PI Controller

A continuous time PI controller may be written:

\[ u(t) = u_0 + K_p e(t) + \frac{K_p}{T_i} \int_0^t e \, dt \]

Where \( u \) is the controller output and \( e \) is the control error:

\[ e(t) = r(t) - y(t) \]

For the control system we need a discrete PI controller in LabVIEW using the, e.g., the Formula Node. You should create a SubVI of the code.

- Typical Inputs to the controller: \( K_p, T_i, y, r \)
- Typical Outputs from the controller: \( u \)

Use the Euler Backward method:

\[ \dot{x} = \frac{x_k - x_{k-1}}{T_s} \]

Make sure the PI controller works as expected.

5.3 Control System Implementation

We need to implement a temperature control system of the Air Heater in LabVIEW using the discrete PI controller and a discrete Low-pass filter created in previous tasks. Test the system with the PI parameters found in a previous task. Tune the parameters if necessary. Document and discuss the results. Use the mathematical model of the Air Heater system created in a previous task in the simulations.

The implementation should be according to the following specifications:

- A PI controller, implemented from scratch with C-code in Formula Node in LabVIEW
• The control signal (the controller output) shall be represented in unit of voltage (0 – 5V).
• The measurement signal, being connected to the controller, shall be represented in unit of degree Celsius (20 – 50°C).
• The temperature setpoint shall be in degree Celsius (20 – 50°C).
• The time-step (sampling time, $T_s$) of the system can be set to, e.g., 0.1 sec.
• Plot the control signal, measurement signal and the setpoint.

When you have tested the system using a mathematical model, it is necessary implement the control system on the real Air Heater process.

For this we will need the real Air Heater process and an USB-6008 DAQ device:

![Air Heater Process](image)

![USB-6008 DAQ Device](image)

**Note!** You need to install the NI-DAQ-mx driver that makes it possible to use the USB-6008 device together with LabVIEW.

Set up hardware and software according to the sketch below:

![Air Heater Control System](image)

Extend your application from the previous task so that you can switch between the simulator and the real process.
Use the same PI parameters that you used in the previous task. Test and document the performance of the control system (both for changes in the set point and for disturbances).

Make sure to use your low-pass measurement filter created in a previous task.

You should discuss the results.
Telemark University College
Faculty of Technology
Kjølnes Ring 56
N-3918 Porsgrunn, Norway
www.hit.no

Hans-Petter Halvorsen, M.Sc.
Telemark University College
Department of Electrical Engineering, Information Technology and Cybernetics

E-mail: hans.p.halvorsen@hit.no
Blog: http://home.hit.no/~hansha/