

Basic
DYNAMICS and CONTROL

Finn Haugen
TechTeach

August 2010
ISBN 978-82-91748-13-9

Contents

1	Introduction to control	1
1.1	The principle of error-driven control, or feedback control . . .	1
1.2	A case study: Level control of wood-chip tank	4
1.2.1	Description of the control system with P&I diagram and block diagram	4
1.2.2	How the level control system works	8
1.3	The importance of control	10
1.4	Some things to think about	13
1.5	About the contents and organization of this book	16
I	PROCESS MODELS AND DYNAMICS	19
2	Representation of differential equations with block diagrams and state-space models	21
2.1	Introduction	21
2.2	What is a dynamic system?	22
2.3	Mathematical block diagrams	23
2.3.1	Commonly used blocks in block diagrams	23
2.3.2	How to draw a block diagram	24

4		
	2.3.3	Simulators based on block diagram models 26
	2.4	State-space models 28
	2.5	How to calculate static responses 31
3	Mathematical modeling	33
	3.1	Introduction 33
	3.2	A procedure for mathematical modeling 34
	3.3	Mathematical modeling of material systems 36
	3.4	Mathematical modeling of thermal systems 41
	3.5	Mathematical modeling of motion systems 45
	3.5.1	Systems with linear motion 45
	3.5.2	Systems with rotational motion 46
	3.6	Mathematical modeling of electrical systems 49
4	The Laplace transform	55
	4.1	Introduction 55
	4.2	Definition of the Laplace transform 55
	4.3	Laplace transform pairs 57
	4.4	Laplace transform properties 58
5	Transfer functions	61
	5.1	Introduction 61
	5.2	Definition of the transfer function 62
	5.3	Characteristics of transfer functions 64
	5.4	Combining transfer functions blocks in block diagrams 65
	5.5	How to calculate responses from transfer function models . . . 65

	5
5.6 Static transfer function and static response	67
6 Dynamic characteristics	69
6.1 Introduction	69
6.2 Integrators	69
6.3 Time-constant systems	71
6.4 Time-delays	75
6.5 Higher order systems	75
II FEEDBACK AND FEEDFORWARD CONTROL	79
7 Feedback control	81
7.1 Introduction	81
7.2 Function blocks in the control loop	81
7.2.1 Automatic and manual mode	81
7.2.2 Measurement lowpass (smoothing) filter	82
7.2.3 Scaling with percentage values	84
7.2.4 Scaling with physical (engineering) values	87
7.3 The PID controller	89
7.3.1 The ideal PID controller function	89
7.3.2 How the PID controller works	90
7.3.3 Positive or negative controller gain? Or: Reverse or direct action?	94
7.4 Practical modifications of the ideal PID controller	96
7.4.1 Lowpass filter in the D-term	96
7.4.2 Reducing P-kick and D-kick caused by setpoint changes	97

7.4.3	Integrator anti wind-up	98
7.4.4	Bumpless transfer between manual/auto mode	102
7.5	Control loop stability	102
8	Feedforward control	105
8.1	Introduction	105
8.2	Designing feedforward control from differential equation models	107
8.3	Designing feedforward control from experimental data	111
9	Controller equipment	117
9.1	Process controllers	117
9.2	Programmable logical controller (PLC)	121
9.3	Programmable Automation Controller (PAC)	123
9.4	SCADA systems	123
9.5	DCS systems	125
9.6	Embedded controllers in motors etc.	126
10	Tuning of PID controllers	129
10.1	Introduction	129
10.2	The Good Gain method	130
10.3	Skogestad's PID tuning method	135
10.3.1	The background of Skogestad's method	135
10.3.2	The tuning formulas in Skogestad's method	137
10.3.3	How to find model parameters from experiments	140
10.3.4	Transformation from serial to parallel PID settings	141
10.3.5	When the process has no time-delay	141

10.4	Auto-tuning	142
10.5	PID tuning when process dynamics varies	144
10.5.1	Introduction	144
10.5.2	PID parameter adjustment with Skogestad's method	146
10.5.3	Gain scheduling of PID parameters	147
10.5.4	Adaptive controller	153
11	Various control methods and control structures	157
11.1	Cascade control	157
11.1.1	The principle of cascade control	157
11.1.2	Benefits of cascade control	158
11.1.3	Controller selection and controller tuning	160
11.1.4	Cascade control and state feedback	161
11.2	Ratio control	168
11.3	Split-range control	169
11.4	Flow smoothing with sluggish level control	171
11.4.1	The control task	171
11.4.2	Controller tuning	172
11.5	Plantwide control	176
12	Sequential control	185
A	Codes and symbols used in Process & Instrumentation Diagrams	191
A.1	Letter codes	191
A.2	Instrumentation symbols used in P&IDs	191

Preface

This book is about on automatic control using the industry-standard PID controller, and control structures based on the PID controller.¹

This book is based on a mathematical description – mathematical models – of the processes to be controlled. Models are very useful for several reasons:

- To represent processes, and other components of control systems, in a clear, conceptual way.
- To create simulators. Simulation is very convenient for design and testing of processes and control systems.
- To characterize the dynamic properties of processes.
- To tune a PID feedback controller from the process model, as with Skogestad's tuning method.
- To design a feedforward controller.

Despite the use of models as mentioned above, this book does not contain theoretical analysis of stability and dynamics of control systems. Also, frequency response analysis is omitted. After many years of teaching basic dynamics and control, I have found that omitting these topics releases valuable time which can be used on practical control topics, as experimental methods for controller tuning and control structures. Furthermore, the practical control tasks that I have been working with have taught me some lessons about what knowledge is needed to solve a control problem. In more advanced courses about control, theoretical analysis of stability and dynamics, including frequency response analysis, is relevant, of course. (A reference for these topics is [2].)

¹PID = proportional + integral + derivative, expressing the mathematical functions of the controller.

The theoretical parts of the book assumes basic knowledge about differential equations. A minimal introduction to the Laplace transform, which is the basis of transfer function models which are used in several sections of the book, is given in a separate chapter of the book.

Supplementary material is available from <http://techteach.no>:

- **Tutorials** for LabVIEW, MATLAB/SIMULINK, Octave, and Scilab/Scicos.
- **SimView** which is a collection of ready-to-run simulators.
- **TechVids** which is a collection of instructional streaming videos, together with the simulators that are played and explained in the videos.
- **An English-Norwegian glossary** of a number of terms used in the book is available at the home page of the book at <http://techteach.no>.

This book is available for sale only via <http://techteach.no>.

It is not allowed to make copies of the book.

About my background: I graduated from the Norwegian Institute of Technology in 1986. Since then I have been teaching control courses in the bachelor and the master studies and for industry in Norway. I have developed simulators for educational purposes, and video lectures, and I have been writing text-books for a couple of decades. I have been engaged in industrial projects about modeling, simulation and control. (More information is on <http://techteach.no/adm/fh>.)

What motivates me mostly is a fascination about using computers to model, simulate and control physical systems, and to bring theoretical solutions into actions using numerical algorithms programmed in a computer. National Instruments LabVIEW has become my favourite software tool for implementing this

Finn Haugen, MSc

TechTeach

Skien, Norway, August 2010

Chapter 1

Introduction to control

Automatic control is a fascinating and practically important field. In short, it is about the methods and techniques used in technical systems which have the ability to automatically correcting their own behaviour so that specifications for this behaviour are satisfied.

In this chapter the basic principles of automatic control and its importance are explained, to give you a good taste of the core of this book! Chapter 7 (and the chapters following that chapter) continues the description of control topics.

1.1 The principle of error-driven control, or feedback control

The basic principle of automatic control is actually something you (probably) are familiar with! Think about the following:

- **How do you control the water temperature of your shower?** I guess that you adjust the water taps with you hand until the difference between the desired temperature – called the reference or setpoint – and the temperature measured by your body is sufficiently small, and you readjust the taps if the difference for some reason becomes too large (water is too hot or too cold). This difference between setpoint and measurement is denoted the *control error*. Thus, the temperature control is *error-driven*.
- **How is the speed of a car controlled?** The gas pedal position is

adjusted by the driver's foot until the difference between the reference or setpoint (desired) speed and measured speed indicated by the speedometer is sufficiently small, and the pedal position is readjusted if necessary. This difference is the control error. Thus, the speed control is *error-driven*.

The principle of *automatic control* is the same as in the examples above – except a technical controller, and not a human being, executes the control action: The control signal is adjusted automatically, by the controller, until the difference between the reference or setpoint (desired) value and the actual, measured value of the process variable is sufficiently small. This difference is denoted the *control error*. Hence, automatic control is obtained with *error-driven control*.

Real measurements always contains some more or less randomly varying noise. Of course, you would not base your control action on such noise. Consequently, you make some smoothing or *filtering* of the measurement to make it less noisy before you use it for control.

Figure 1.1 shows a block diagram of a general control system based on an error-driven control system.

The system contains the following four main blocks:

- Process
- Sensor
- Filter
- Controller

(You will get a deeper understanding of these blocks during the subsequent section which is about a case study of an industrial control system, namely a level control system of a wood-chip tank.)

Although “error-driven control” is a very descriptive term about the principle of an automatic control system, it is more common to use the term *feedback control* about this principle. Look at the block diagram shown in Figure 1.1. The control error is calculated as the difference between the reference (or setpoint) and the process measurement, and the control signal, which is the input to the process to be controlled, is adjusted by the controller as a function of the control error. Consequently, the process output (its measurement) is *fed back* to the process input

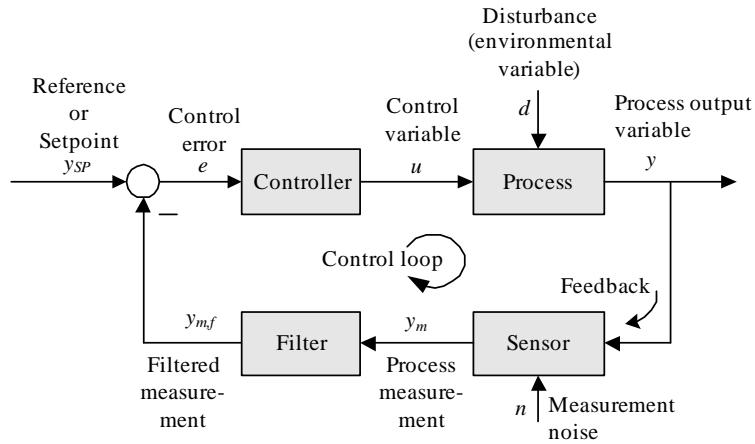


Figure 1.1: Block diagram of an error-driven control system

variable (control signal) via the controller. Hence, error-driven control implies *feedback* control.

Feedback control is not the only solution to the control problem! Although it is the most important and the most frequently used control principle, control can be improved by also including *feedforward control* in the control system. Feedforward control is based on measuring process disturbances, and adjusting the control signal as a function of the disturbance to obtain a direct and momentary compensation of the disturbance, keeping the process variable closer to the setpoint than with only feedback control. (Feedforward control can also include a direct coupling from the setpoint or reference to the control signal to improve reference tracking.) For example, in a position control system for keeping a ship (e.g. an oil tanker) at a specified position at the sea, more accurate position control can be obtained by measuring the wind speed and direction and using this measurement to directly adjust the thruster force to compensate for the wind force. We will not study feedforward control in this introductory chapter, but you can look forward to Chapter 8 which is all about feedforward control.

1.2 A case study: Level control of wood-chip tank

1.2.1 Description of the control system with P&I diagram and block diagram

We will study an example of a real industrial control system. Figure 1.2 shows a level control system for a wood-chip tank with feed screw and conveyor belt which runs with constant speed. Wood-chip is consumed via an outlet screw in the bottom of the tank.^{1 2 3} The purpose of the control system is to keep the measured chip level y_m equal to a level setpoint y_{SP} , despite variations of the outflow, which is a process disturbance d .

The level control system works as follows (a more detailed description of how the control system works is given in Section 1.2.2): The controller tries to keep the measured level equal to the level setpoint by adjusting the rotational speed – and thereby the chip flow – of the feed screw as a function of the control error (which is the difference between the level setpoint and the measured level).

A few words about the need for a level control system for this chip tank: Hydrogene sulphate gas from the pulping process later in the production line is used to preheat the wood chip. If the chip level in the tank is too low, too much (stinking) gas is emitted to the atmosphere, causing pollution. With level control the level is kept close to a desired value (set-point) at which only a small amount of gas is expired. The level must not be too high, either, to avoid overflow and reduced preheating.

In Figure 1.2 the control system is documented in two ways:

- **Process and instrumentation diagram or P&I diagram** which is a common way to document control systems in the industry. This diagram contains easily recognizable drawings and symbols of the process to be controlled, together with symbols for the controllers and the sensors and the signals in the control system. Appendix A

¹This example is based on an existing system in the paper pulp factory Södra Cell Tofte in Norway. The tank with conveyor belt is in the beginning of the paper pulp production line.

²The tank height is 15 m. The diameter is 4.13 m, and the cross-sectional area is 13.4 m². The nominal wood-chip outflow (and inflow at steady state) is 1500 kg/min. The conveyor belt is 200 m long, and runs with fixed speed. The transportation time (time-delay) of the belt is 250 sec = 4.17 min.

³A simulator of the system is available at <http://techteach.no/simview>.

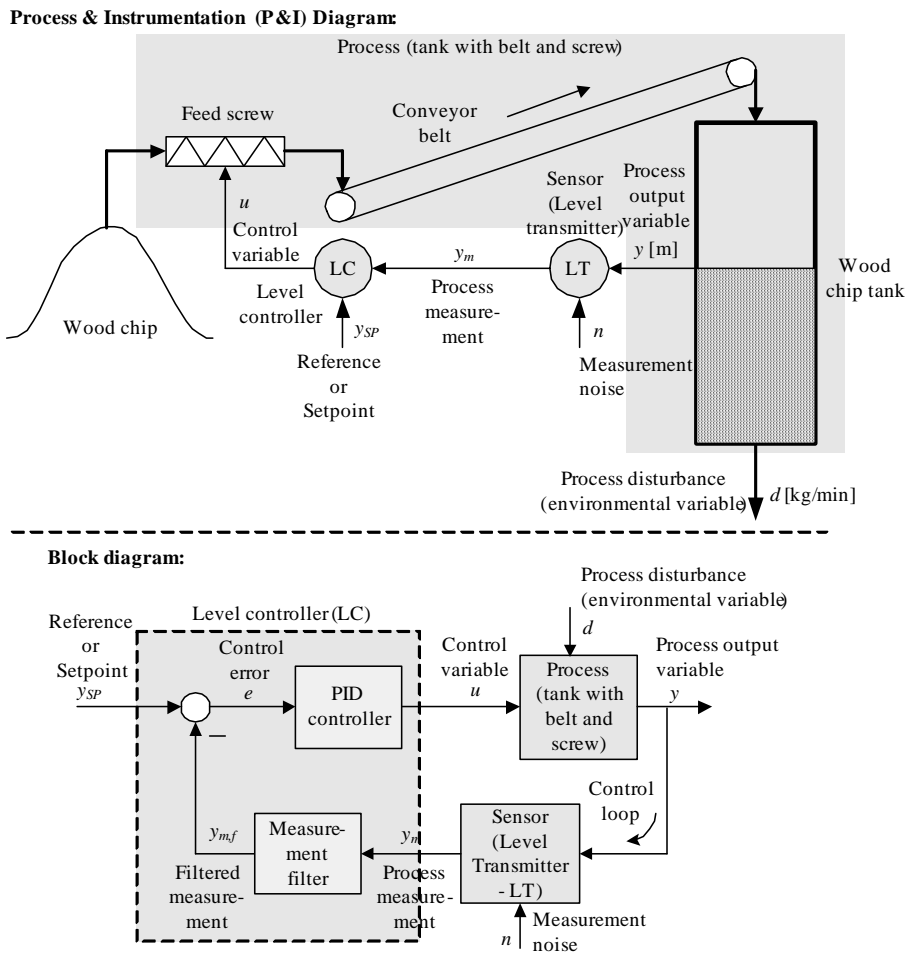


Figure 1.2: P&I (Process and Instrumentation) diagram and block diagram of a level control system for a wood-chip tank in a pulp factory

presents some of the most frequently used symbols in P&I diagrams. There are international standards for instrumentation diagrams, but you can expect that “local”, company standards are used.

In the example the LT (Level Transmitter) represents the level sensor. (The level sensor is based on ultrasound: The level is calculated from the reflection time for a sound signal emitted from the sound transmitter to the sound receiver.)

LC is the Level Controller. The setpoint is usually not shown explicitly in an P&I diagram (it is included in the LC block), but it is shown in Figure 1.2 to make the diagram more illustrative. The controller (here LC) includes a measurement filter used to smooth

noisy measurement. It is not common to show the filter explicitly in P&I diagrams.

- **Block diagram** which is useful in principal and conceptual description of a control system.

Below are comments about the systems and variables (signals) in the block diagram shown in Figure 1.2.

Systems in Figure 1.2:

- **The process** is the physical system which is to be controlled. Included in the process is the actuator, which is the equipment with which (the rest of) the process is controlled.
In the example the process consists of the tank with the feed screw and the conveyor belt.
- **The controller** is typically in the form of a computer program implemented in the control equipment. The controller adjusts the control signal used to control or manipulate the process. The controller calculates the control signal according some mathematical formula defining the controller function. The controller function defines how to adjust the control signal as a function of the control error (which is the difference between the setpoint and the process measurement).
- **The sensor** measures the process variable to be controlled. The physical signal from the sensor is an electrical signal, voltage or current. In industry 4–20 mA is the most common signal range of sensor signals. (In the example the sensor is an ultrasound level sensor, as mentioned earlier.)
- **The measurement filter** is a function block which is available in most computer-based automation systems. The filter attenuates or smooths out the inevitable random noise which exists in the measurement signal. The measurement filter is described in detail in Section 7.2.2.
- **The control loop or feedback loop** is the closed loop consisting of the process, the sensor, the measurement filter, and the controller connected in a series connection.

Variables (signals) in Figure 1.2:

- **The control variable or the manipulating variable** is the variable which the controller uses to control or manipulate the process. In this book u is used as a general symbol of the control variable. In commercial equipment you may see the symbol MV (manipulating variable).

In the example the control variable (or control signal) adjust the flow through the feed screw.

- **The process output variable** is the variable to be controlled so that it becomes equal to or sufficiently close to the setpoint. In this book y is used as a general symbol of the process output variable. In commercial control equipment PV (process variable or process value) may be used as a symbol.

In the example the wood chip level in the tank is the process output variable.

Note: The process output variable is not necessarily a physical output from the process! In our example the chip outflow is *not* the process output variable. The chip outflow is actually a process disturbance, see below.

- **The disturbance** is a non-controlled input variable to the process which affects the process output variable. From the control system's perspective, this influence on the process output variable is undesirable, and the controller will adjust the control variable to compensate for the influence. In this book, d is used as a general symbol for the disturbance. Typically, there are more than one disturbances acting on a process.

In the example the chip outflow from the bottom of the tank is the (main) disturbance as it tends to bring the process variable (level) away from the level setpoint. Other disturbances are variations in the inlet flow due to e.g. chip density variations.

- **The setpoint or the reference** is the desired or specified value of the process output variable. The general symbol y_{SP} will be used in this book.

In the example the desired level is the setpoint. A typical value for this tank is 10 meters.

- **The measurement signal** is the output signal from the sensor which measures the process variable.

In the example the measurement signal is a current signal in the range 4 – 20 mA corresponding to level 0 – 15 m.

- **The measurement noise** is typically a random component in the measurement signal. In practice all sensors produce noise measurements. The noise is propagated via the controller to the control signal, which therefore can have abrupt variations, which may be a serious problem for a control system. The measurement noise may have various sources:
 - It can be electronic (induced) noise.
 - It can be noise due to the measurement principle, as when a ultrasound sensor is used to measure the level of a wavy liquid surface.
 - It can be noise due to the limited resolution of the analog-to-digital (AD) converter which converts the voltage or current measurement signal into a number to be used in the automation computer.

In the example the measurement noise stems from the AD converter and from the irregular surface of the chip surface in the tank.

- **The control error** is the difference between the setpoint and the process output variable:

$$e = y_{SP} - y \quad (1.1)$$

Since the process output variable is known via its measurement, the control error is actually calculated as the difference between the setpoint and the measurement, and it is expressed in a proper unit, e.g. meter or percent.

1.2.2 How the level control system works

Figure 1.3 shows simulated responses in various variables of the system.⁴ Initially, the control signal is $u = 45\%$ (it can be shown that this control signal is producing a chip inflow of 1500 kg/min which equals the initial outflow), and the level is at the setpoint value of 10 m, so the initial control error is zero. Random measurement noise is added to the level measurement.

The controller is a PI controller (proportional plus integral) which is the most commonly used controller function in industry. The PI controller, which is a special case of the more general PID controller (proportional plus integral plus derivative), is described in detail in Section 7.3.

⁴Parameter values of the system can be found on the front panel of the simulator of the system which is available at <http://techt teach.no/simview>.

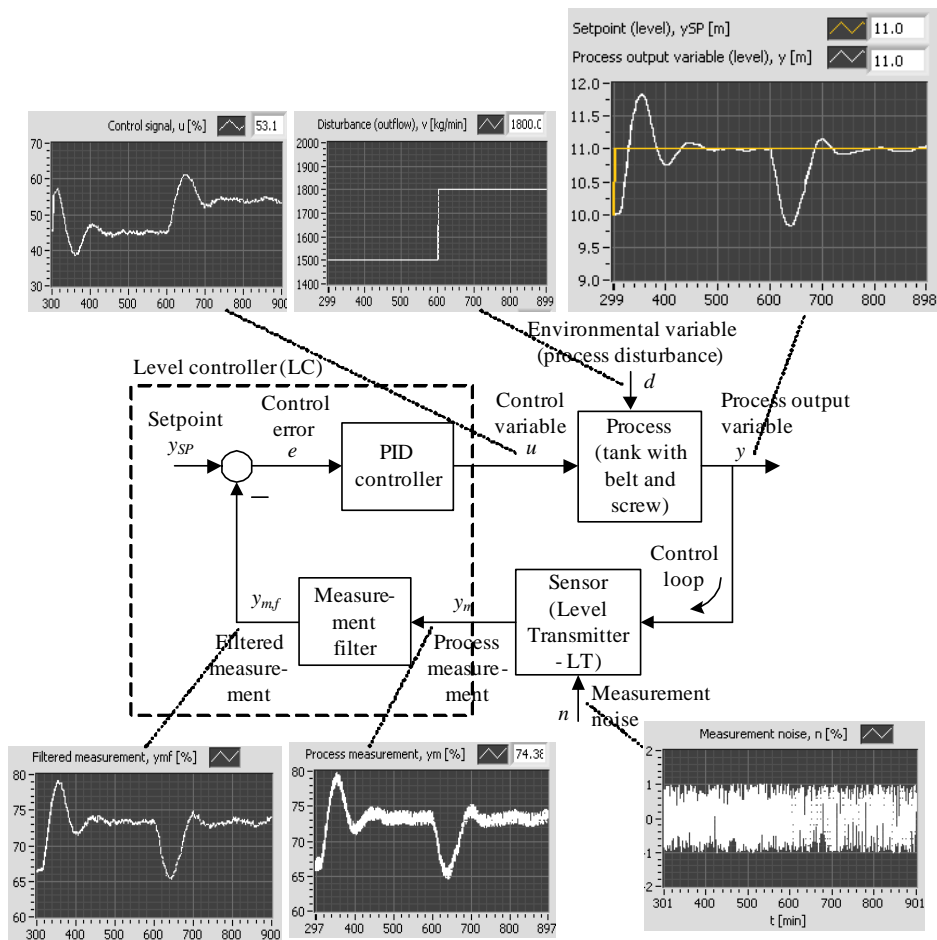


Figure 1.3: Simulated responses in various variables of the level control system of the wood chip tank

The control system works as follows:

- At time 300 min the setpoint is changed as a step from 10 to 11 m (see the upper right plot in Figure 1.3), and hence, the control error is suddenly increased from 0 to 1 m. As a consequence of this non-zero error, the controller starts increasing the control signal to the inlet sciew (upper left plot), causing an increased chip inflow, thereby increasing the level, thereby reducing the control error. This continues until the level reaches the new setpoint of 11 m, so that control error becomes zero (upper right plot).
- At time 600 min the disturbance (outflow) is changed as a step from 1500 to 1800 kg/min (upper middle plot), causing the level to

decrease, and hence the control error becomes different from zero. The operation of the control system is as after the setpoint change: Because of the non-zero error, the controller starts increasing the control signal to the inlet scrow, causing an increased chip inflow, thereby increasing the level, thereby reducing the control error. This continues until the level is at the setpoint of 11 m again.

- The measurement noise is smoothed by the measurement filter, but the noise is not completely removed (lower plots).
- The remaining measurement noise (what remains despite the filtering) is propagated through the controller, causing the control signal to be somewhat noisy (upper left plot).

1.3 The importance of control

In the previous sections we studied a level control system of a wood-chip tank. In general, the following process *variables* are controlled in industrial and other kinds of technical applications:

- Level or mass (of e.g. a storage tank)
- Pressure (in a chemical reactor)
- Temperature (in a room; in the fluid passing a heat exchanger; in a reactor; in a greenhouse)
- Flow (of feeds into a reactor)
- pH (of a reactor)
- Chemical composition (of nitric acid; fertilizers, polypropylene)
- Speed (of a motor; a car)
- Position (of a ship; a painting robot arm; the tool of a cutting machine; a rocket)

Application of control may be of crucial importance to obtain the following aims:

- **Good product quality:** A product will have acceptable quality only if the difference between certain process variables and their setpoint values – this difference is called the *control error* – are kept less than specified values. Proper use of control engineering may be necessary to achieve a sufficiently small control error, see Figure 1.4.

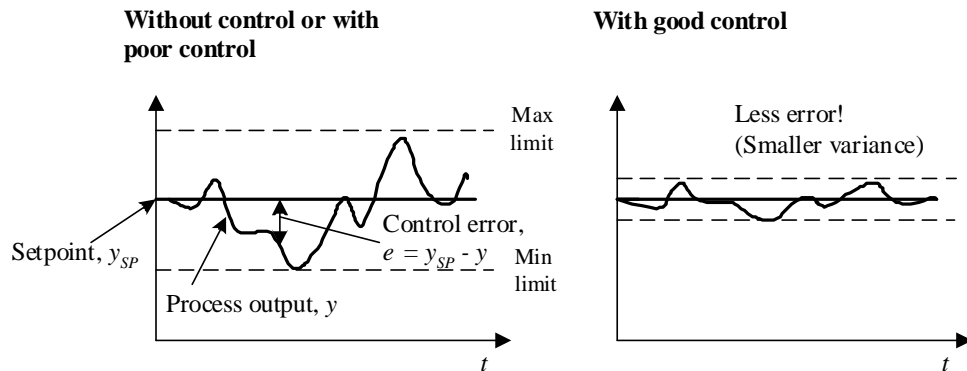


Figure 1.4: Good control reduces the control error

One example: In fertilizers the pH value and the composition of Nitrogen, Phosphate and Potassium are factors which express the quality of the fertilizer (for example, too low pH value is not good for the soil). Therefore the pH value and the compositions must be controlled.

- **Good production economy:** The production economy will become worse if part of the products has unacceptable quality so that it can not be sold. Good control may maintain the good product quality, and hence, contribute to good production economy. Further, by good control it may be possible to tighten the limits of the quality so that a higher price may be taken for the product!
- **Safety:** To guarantee the security both for humans and equipment, it may be required to keep variables like pressure, temperature, level, and others within certain limits– that is, these variables must be controlled. Some examples:
 - An aircraft with an autopilot (an autopilot is a positional control system).
 - A chemical reactor where pressure and temperature must be controlled.

- **Environmental care:** The amount of poisons to be emitted from a factory is regulated through laws and directions. The application of control engineering may help to keep the limits. Some examples:
 - In a wood chip tank in a paper pulp factory, hydrogen sulfate gas from the pulp process is used to preheat the wood chip. If the chip level in the tank is too low, too much (stinking) gas is emitted to the atmosphere, causing pollution. With level control the level is kept close to a desired value (set-point) at which only a small amount of gas is expired.
 - In the so-called washing tower nitric acid is added to the intermediate product to neutralize exhaust gases from the production. This is accomplished by controlling the pH value of the product by means of a pH control system. Hence, the pH control system ensures that the amount of emitted ammonia is between specified limits.
 - Automatically controlled spray painting robots avoid humans working in dangerous areas. See Figure 1.5.



Figure 1.5: Spray painting robot (IRB580, ABB)

- **Comfort:**
 - The automatic positional control which is performed by the autopilot of an aircraft to keep a steady course contributes to the comfort of the journey.
 - Automatic control of indoor temperature may give better comfort.
- **Feasibility:** Numerous technical systems could not work or would even not be possible without the use of control engineering. Some examples:

- An exothermal reactor operating in an unstable (but optimal) operating point
- Launching a space vessel (the course is stabilized)
- A dynamic positioning system holds a ship at a given position without an anchor despite the influence of waves, wind and current on the ship. The heart of a dynamic positioning system is the positional control system which controls the thrusters which are capable of moving the ship in all directions. See Figure 1.6.

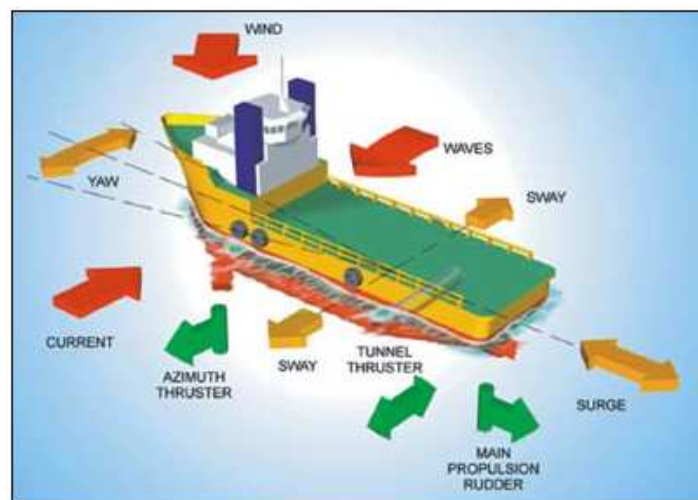


Figure 1.6: A dynamic positioning system holds a ship at a given position without an anchor despite the influence of waves, wind and current on the ship (Kongsberg Simrad, Norway)

- **Automation:** Computers and other kinds of hardware and software implementing control solutions can accomplish tedious and dangerous operations for the benefit of human operators. Also, automation may reduce costs in a factory, thereby indirectly reducing product prices, to customer's benefit.

1.4 Some things to think about

Below are some good questions you should pose to yourself if you get involved in control system design. Many terms in the answers below are explained during this book, so don't expect to understand all of it at this

moment. You may read this section again after you have completed the book!

- **Is there really a need for control?** Yes, there is a need for control if there is a chance that the process output variable will drift too far away from its desired value. Such a drift can be caused by severe variations of environmental variables (process disturbances). For *unstable* processes, like water tanks and exothermal reactors and motion systems like robots and ships which must be positioned, there will always be a need for control to keep the process output variable (level; temperature; position, respectively) at a setpoint or reference value.
- **Which process variable(s) needs to be controlled?** (To make it become equal to a setpoint or reference value.) Liquid level in a given tank? Pressure of vapour in the tank? Temperature? Flow? Composition?
- **How to measure that process variable?** Select an appropriate sensor, and make sure it detects the value of the process variable with as little time-delay and sluggishness as possible! In other words, measure as directly as you can.
- **Is the measurement signal noisy?** It probably is. Use a lowpass filter to filter or smooth out the noise, but don't make the filtering too strong – or you will also filter out significant contents of the measurement signal, causing the controller to react on erroneous information.
- **How to manipulate the process variable?** Select an actuator that gives a strong impact on the process variable (to be controlled)! Avoid time-delays if possible, because time-delays in a control loop will limit the speed of the control, and you may get a sluggish control loop, causing the control error to become large after disturbance variations.
- **Which controller function (in the feedback controller)?** Try the standard PID controller. Note that most PID controllers actually operate as PI controllers since the derivative (D) term is deactivated because it amplifies measurement noise through the controller, causing noisy control signal which may cause excessive wear of a mechanical actuator.
- **How to tune the controller?**

If you don't have mathematical model of the process to be controlled, try the Good Gain method, which is a simple, experimental tuning method.

If you do have a mathematical process model, try Skogestad's model-based tuning method to get the controller parameters directly from the process model parameters. Alternatively, with a model, you can create a simulator of your control system in e.g. LabVIEW or Simulink or Scicos, and then apply the Good Gain method on the simulator.

If your controller has an Auto-tune button, try it!

- **Is the stability of the control system good?** The most important requirement to a control system is that it has good stability. In other words, the responses should show good damping (well damped oscillations). The responses in the level control system which are shown in Figure 1.3 indicate good stability!

If you don't have a mathematical model of the system, apply a (small) step change of the setpoint and observe whether the stability of the control system is ok. If you think that the stability is not good enough (too little damping of oscillations), try reducing the controller gain and increasing the integral time somewhat, say by a factor of two. Also, derivative control action can improve stability of the control loop, but remember the drawback of the D-term related to amplification of random measurement noise.

If you do have a mathematical model of the control system, create a simulator, and simulate responses in the process variable and the control variable due to step changes in the setpoint and the disturbances (load variables), e.g. environmental temperature, feed composition, external forces, etc. With a simulator, you may also want to make realistic changes of certain parameters of the process model, for example time-delays or some other physical parameters, to see if the control system behaves well despite these parameter changes (assuming the controller is not tuned again). In this way you can test the robustness of the control system against process variations. If the control system does not behave well, for example having too poor stability, after the parameter changes, consider Gain scheduling which is based on changing the PID parameters automatically as functions of certain process parameters.

- **Is the control error small enough after disturbance changes, and after setpoint changes?**

If you do not have a simulator of the control system, it may be difficult to generate disturbance changes yourself, but setpoint

changes can of course be applied easily.

If you have a simulator, you can apply both setpoint changes and disturbance changes.

If – using either experiments or simulator – the control error is too large after the changes of the setpoint and the disturbance, try to tune the controller again to obtain faster control.

- **Still not happy with control system performance after retuning the controller?** Then, look for other control structures or methods based on exploiting *more process information*:
 - *Feedforward control*: This requires that you measure one or more of the disturbances (load variables) acting on the process, and using these measurements to directly adjust the control signal to compensate for the disturbance(s).
 - *Cascade control*: This requires that you measure some internal process variable which is influenced by the disturbance, and construct an inner control loop (inside the main control loop) based on this internal measurement to quickly compensate for the disturbance.
 - *Model-based control*: Consider for example optimal control with state-variable feedback (LQ (Linear Quadratic) optimal control), or model-based predictive control (MPC). [2]

1.5 About the contents and organization of this book

This book contains two main parts:

- **Part One: Process models and dynamics** which presents the basic theory of mathematical process models and dynamics. This systems theory is useful for practical process control mainly because it makes it possible to implement simulators of control systems. With simulators you can test a virtual or a real (practical) control system. The purpose of such tests can be design, analysis, or operator (and student) training. Industry uses simulator-based design and training more and more. The systems theory is useful also because it gives you tools to represent and to characterize various components of a control system.

- **Part Two: Feedback and feedforward control** which presents practical control methods with emphasis on feedback control with PID (proportional + integral + derivative) controllers which are commonly used in the industry. Feedforward control is also described. Furthermore, several control structures based on the PID controller are described, as cascade control, ratio control, and plantwide control principles. Also, sequential control using Sequential function charts is described.

Part I

**PROCESS MODELS AND
DYNAMICS**

Chapter 2

Representation of differential equations with block diagrams and state-space models

2.1 Introduction

The basic mathematical model form for dynamic systems is the *differential equation*. This is because the basic modeling principle – denoted the Balance Law or Conservation principle – used to develop a model results in one or more differential equations describing the system. These modeling principles are described in Chapter 3. The present chapter shows you how to represent a given differential equation as *block diagrams* and *state-space models*:

- Block diagrams give good information of the structure of the model, e.g. how subsystems are connected. And what is more important: With block diagrams you can build simulators in graphical simulation tools as SIMULINK and LabVIEW Simulation Module.
- State-space models are a standardized form of writing the differential equations. All the time-derivatives in the model are of first order, and they appear on the left-hand side of the differential equations. Various tools for analysis and design and simulation of dynamic systems and control systems assume state models. However, this book does not cover state-space analysis and design (a reference is

[2]), and concerning simulation it is my view that it is more useful to use block diagrams than state-space models as the basis for creating simulation models. Still, the concept of state-space models is introduced here because you may meet it in certain contexts (e.g. literature) related to basic dynamics and control.

2.2 What is a dynamic system?

Dynamic means “which has to do with the movement and change”. Dynamic systems are systems where the variables can vary or develop with time. We say that dynamic systems have dynamic responses. Figure 2.1

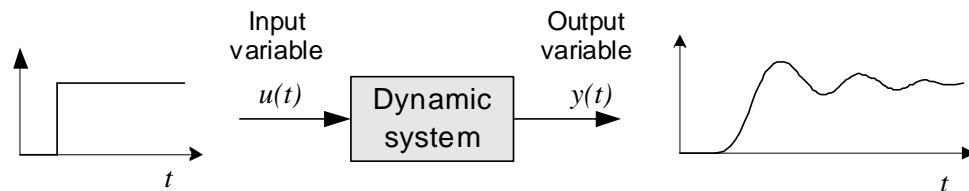


Figure 2.1: Dynamic systems are systems where the variables can vary or develop as functions of time.

gives an illustration. The figure shows a block diagram of a dynamic system. The input variable is here a step function, and the response in the output variable is dynamic since it changes with time. In general, dynamic systems may have more than one input variable and more than one output variable.

Here are some examples of dynamic systems:

- A liquid tank. Input (variable): Inflow. Output (variable): Level.
- A motor. Input: Motor control voltage. Output: Speed.
- A heated water tank. Input: Supplied heat. Output: Temperature in the water in the tank.
- A robot manipulator. Input: Control signal to motor. Output: Arm position.
- A ship. Input: Thruster force. Output: Ship position.
- A signal filter: Input: Filter input to be filtered (smoothed). Output: Filter output signal.

- A control system for a physical process: Input: Setpoint. Output: Process output variable.

2.3 Mathematical block diagrams

2.3.1 Commonly used blocks in block diagrams

A *mathematical block diagram* gives a graphical representation of a mathematical model. Figure 2.2 shows the most frequently used blocks, which we can call the *elementary blocks*, which we can use for drawing block diagrams.¹

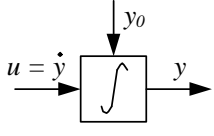
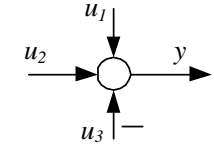
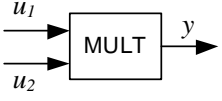
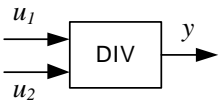
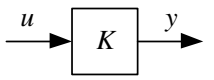

Name:	Symbol:	Function:
Integrator		$y(t) = y_0 + \int_0^t u(t)dt$
Sum (incl. subtraction) (No sign means plus.)		$y = u_1 + u_2 - u_3$
Multiplication		$y = u_1 u_2$
Division		$y = u_1 / u_2$
Gain		$y = Ku$
Time delay		$y(t) = Ku(t-\tau)$

Figure 2.2: Elementary blocks for drawing block diagrams

¹You may wonder what is the difference between the Multiplication block and the Gain block. Mathematically, there is no difference. It is mostly a matter of personal habit and desire which one you use for multiplication. Some people use the Gain block when one of the factors is a constant parameter. Personally, I like to use the Multiplication block in all cases.

Other blocks than the elementary blocks shown in Figure 2.2 must be used to represent non-linear functions. Figure 2.3 shows a few such blocks. Also, there is a Formula block in which you can write any formula. The Formula block can be used to reduce the number of blocks in the block diagram. You can also define blocks yourself.

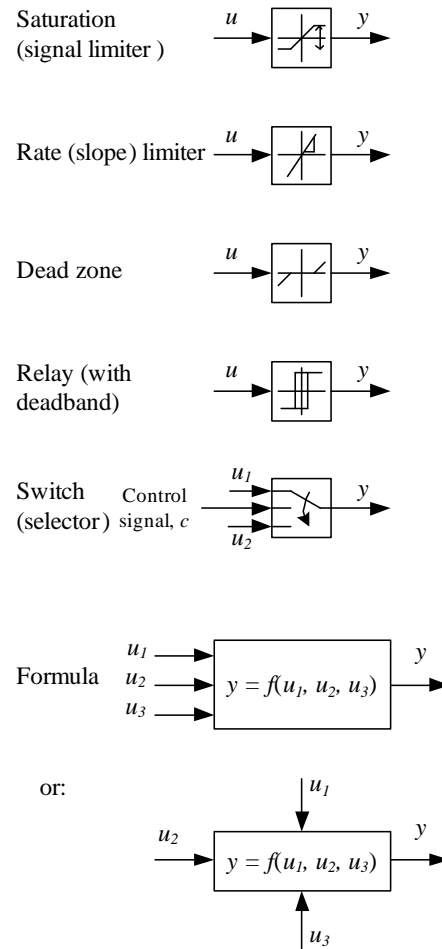


Figure 2.3: Miscellaneous blocks, including blocks for nonlinear functions

2.3.2 How to draw a block diagram

A systematic way of drawing the block diagram of a given differential equation is as follows:

1. Write the differential equations so that the variable of the highest time-derivative order appears alone on the left part of the equations.
2. For *first order* differential equations (first order time-derivatives): Draw one integrator for each of the variables that appear with its time-derivative in the model, so that the time-derivative is at the integrator input and the variable itself is at the integrator output.

For *second order* differential equations (second order time-derivatives): Draw two integrators (from left to right) and connect them in series. The second order time-derivative is then the input to the leftmost integrator.

You will probably not see third or higher order differential equations, but if you do, the above procedure is naturally extended.

The variables at the integrator outputs can be regarded as the *state-variables* of the system because their values at any instant of time represent the *state* of the system. The general names of the state-variables are x_1, x_2 etc. The initial values of the state-variables are the initial outputs of the integrator. (More about state-space models in Section 2.4.)

3. Connect the integrators together according to the differential equation using proper blocks, cf. Figures 2.2 and 2.3. Alternatively, to use less blocks, you can include one

The following example demonstrates the above procedure.

Example 2.1 *Block diagram of mass-spring-damper system*

Figure 2.4 shows a mass-spring-damper-system.² y is position. F is applied force. D is damping constant. K is spring constant. It is assumed that the damping force F_d is proportional to the velocity:

$$F_d(t) = D\dot{y}(t) \quad (2.1)$$

and that the spring force F_s is proportional to the position of the mass:

$$F_s(t) = Ky(t) \quad (2.2)$$

²The mass-spring-damper system is not a typical system found in process control. It is chosen here because it is easy to develop a mathematical model using well known physical principles, here the Newton's second law. Examples of more relevance to process control are described in Chapter 3.

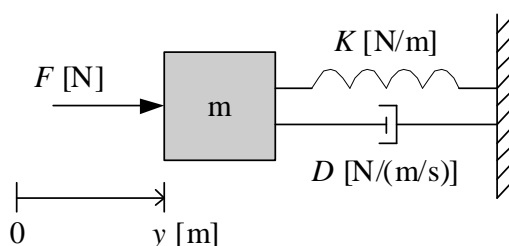


Figure 2.4: Mass-spring-damper

The spring force is assumed to be zero when y is zero. Force balance (Newton's 2. Law) yields³

$$\begin{aligned} m\ddot{y}(t) &= F(t) - F_d(t) - F_s(t) \\ &= F(t) - D\dot{y}(t) - Ky(t) \end{aligned} \quad (2.3)$$

which is a second order differential equation.

We isolate \ddot{y} at the left side:

$$\ddot{y}(t) = \frac{1}{m} [F(t) - D\dot{y}(t) - Ky(t)] \quad (2.4)$$

Then we draw two integrators and connect them in series. \ddot{y} is the input to the first (leftmost) integrator. Finally, we complete the block diagram according to the model. The resulting block diagram is shown in Figure 2.5. (Here I have used Multiplication blocks, and not Gain blocks, to represent multiplication.)

Figure 2.6 shows an alternative block diagram where a Formula block has been used.

[End of Example 2.1]

2.3.3 Simulators based on block diagram models

LabVIEW, SIMULINK, and Scicos are examples of simulation tools for block diagram models.⁴ Figure 2.7 shows the block diagram of the mass-spring-damper system in LabVIEW, and Figure 2.8 shows the front panel (the user interface) of the simulator. The front panel contains adjustable elements (controls), and indicators displaying resulting values and plots.

³Double-dot represents second order time-derivative: $\ddot{y}(t) \equiv d^2y(t)/dt^2$

⁴Tutorials are available at <http://teach.no>.

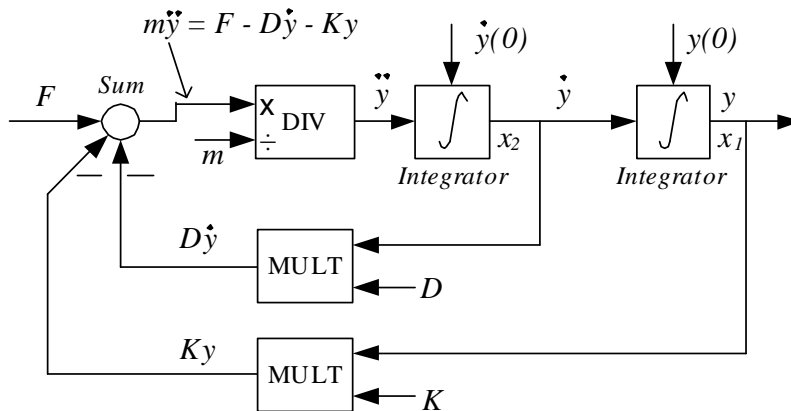


Figure 2.5: Block diagram of (2.4)

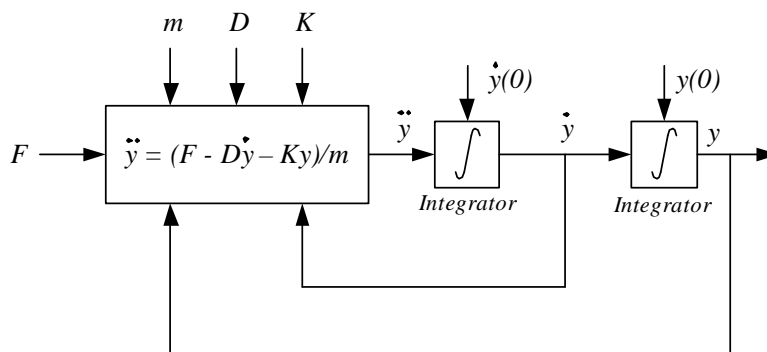


Figure 2.6: An alternative block diagram of the mass-spring-damper system where a Formula block has been used

Figure 2.9 shows an alternative LabVIEW block diagram where a Formula Node (containing programming code in the C-language) has been used in stead of elementary blocks.

Figure 2.10 shows a SIMULINK block diagram of the mass-spring-damper system. The simulated force F and the response in the position y due to a step in the force will be shown in respective scopes. (Since the responses are the same as shown in Figure 2.8, the scopes with the responses are not shown here.)

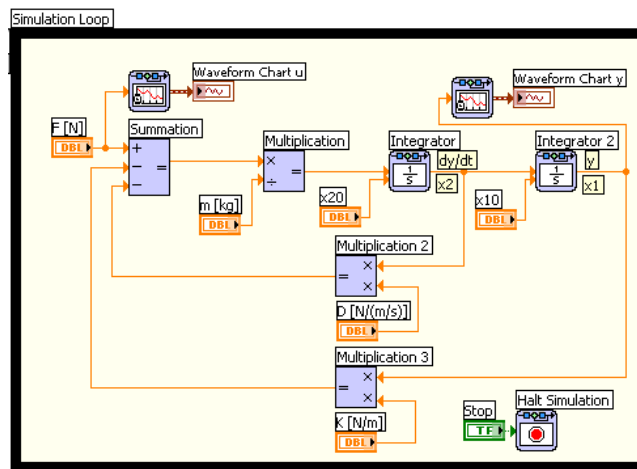


Figure 2.7: The block diagram of the mass-spring-damper system simulator in LabVIEW

2.4 State-space models

A *state-space model* is just a structured form or representation of the differential equations for a system. Some applications of state-space models are:

- **Linearization of non-linear models**
- **Calculation of time-responses** – both analytically and numerically
- **Simulation:** MATLAB, LabVIEW, Octave, and Scilab have simulation functions that assume state-space models.⁵
- **Analysis of dynamic systems**, e.g. stability analysis
- **Analysis and design of advanced controllers and estimators:** Controllability and observability analysis; Design of LQ optimal controllers, Model-based predictive control, and Feedback linearization control; Design of state estimators (Kalman filters).

(The above topics (except simulation) are covered by [2].)

⁵For example the ODE functions in MATLAB. (ODE = Ordinary Differential Equation)

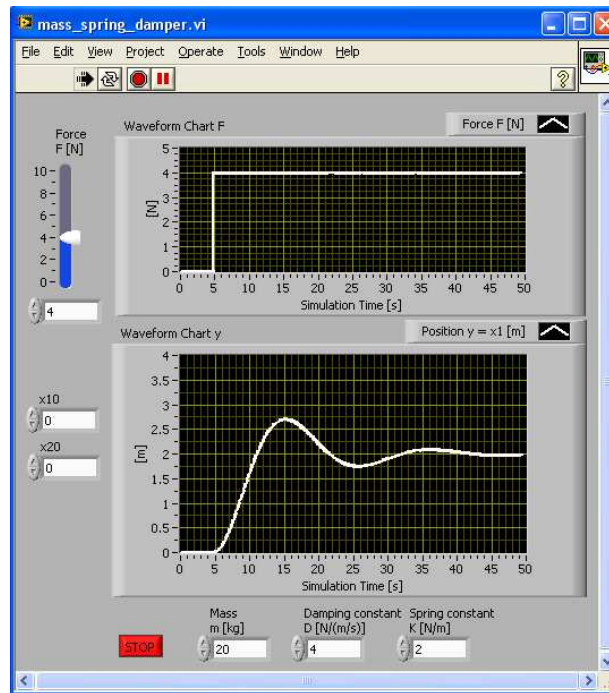


Figure 2.8: The front panel of the mass-spring-damper system simulator in LabVIEW

An n 'th order state-space model consists of n *first order* differential equations characterized by the having *only the time-derivatives on the left side*:

$$\dot{x}_1 = f_1(x_1, x_2, \dots, u_1, u_2, \dots) \quad (2.5)$$

$$\vdots$$

$$\dot{x}_n = f_n(x_1, x_2, \dots, u_1, u_2, \dots) \quad (2.6)$$

where f_1, \dots, f_n are functions. The u -variables are here input variables (independent variables). The list of arguments of the functions can be any, of course, depending on the actual model. The variables which have their time-derivatives in the state-space model are the *state-variables* of the model.⁶ Thus, in the model above the x -variables are state-variables. x is a common name for state-variables, but you can use any name. The initial state (at $t = 0$) are defined by the values $x_1(0), \dots, x_n(0)$.

⁶The name *state-space* model is because the values of the state-variables $x_1(t), \dots, x_n(t)$ defines the *state* of the at any instant of time. These values can be regarded as points in the *state-space*.

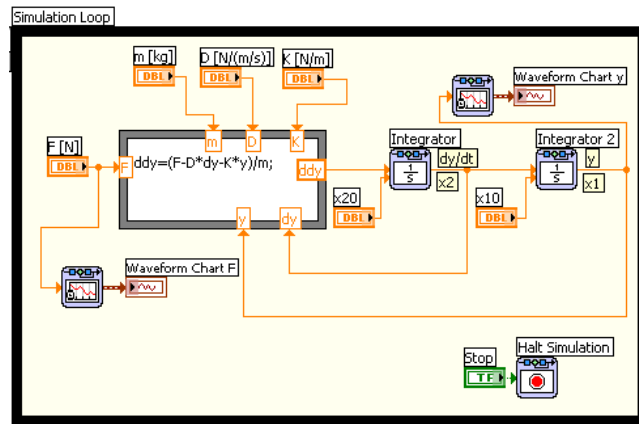


Figure 2.9: An alternative LabVIEW block diagram of the mass-spring-damper system with a Formula Node containing programming code in the C-language (in stead of elementary blocks)

Some times you want to define the *output variables* of a state-space model. y is a common name for output variable. Assuming m output variables, the output equations are

$$y_1 = g_1(x_1, x_2, \dots, u_1, u_2, \dots) \quad (2.7)$$

$$\vdots$$

$$y_m = g_m(x_1, x_2, \dots, u_1, u_2, \dots) \quad (2.8)$$

where g_1, \dots, g_m are functions.

The term state-space model can be used for only the differential equations (2.5) – (2.6), and for the total model (2.5) – (2.8).

Actually, for a given mathematical model, the state variables are not unique. That is, there are an infinite number ways to define the state variables of the system. For example, in a mass-spring-damper system, any linear combination of position and velocity can be a state variable.

However, it is usually natural and most useful to select state variables as physical variables, as position, velocity, level, temperature, etc. In a model consisting of higher order differential equations, it may be a little difficult to define the state variables, but one way is to draw a block diagram of the model and select the integrator outputs as state variables.

Example 2.2 *Mass-spring-damper-model written as a state-space model*

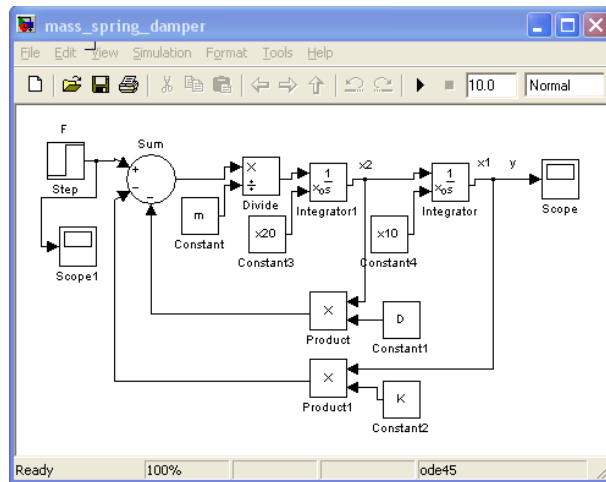


Figure 2.10: SIMULINK block diagram of the mass-spring-damper system

See the block diagram shown in Figure 2.5. We select the integrator outputs as state variables, that is, x_1 (position) and x_2 (speed). From the block diagram we can directly write the state-space model:

$$\dot{x}_1 = \underbrace{x_2}_{f_1} \quad (2.9)$$

$$\dot{x}_2 = \underbrace{\frac{1}{m} (-Dx_2 - K_f x_1 + F)}_{f_2} \quad (2.10)$$

Let us say that position x_1 is the output variable y :

$$y = \underbrace{x_1}_g \quad (2.11)$$

The initial position, $x_1(0)$, and the initial speed, $x_2(0)$, define the initial state of the system.

(2.9) – (2.11) constitute a second order state-space model which is equivalent to the original second order differential equation (2.3).

[End of Example 2.2]

2.5 How to calculate static responses

In some situations it is useful to calculate by hand the *static response* in a dynamic system, e.g. to check that the simulated response is correct. The

static response is the steady-state value of the output variable of the model when the input variables have constant values. This response can be calculated directly from the model after the *time-derivatives have been set equal to zero*, since then the variables have constant values, their time-derivatives equal zero.

Example 2.3 *Calculation of static response for mass-spring-damper*

The mass-spring-damper system described in Example 2.1 has the following model:

$$m\ddot{y} = -D\dot{y} - Ky + F \quad (2.12)$$

Suppose the force F is constant of value F_s . The corresponding static response in the position y can be found by setting the time-derivatives equal to zero and solving with respect to y . The result is

$$y_s = \frac{F_s}{K} \quad (2.13)$$

Let us use (2.13) to check the simulated response shown in Figure 2.8: On the front panel we see that $F_s = 4$ N, and $K = 2$ N/m. Thus,

$$y_s = \frac{F_s}{K} = \frac{4 \text{ N}}{2 \text{ N/m}} = 2 \text{ m} \quad (2.14)$$

which is the same as the simulated static value of y , cf. Figure 2.8.

[End of Example 2.3]

Chapter 3

Mathematical modeling

3.1 Introduction

This chapter describes basic principles of mathematical modeling. A mathematical model is the set of equations which describe the behavior of the system. The chapter focuses on how to *develop* dynamic models, and you will see that the models are *differential equations*. (How to represent such differential equations as block diagrams, ready for implementation in block diagram based simulators, was explained in Section 2.)

Unfortunately we can never make a completely precise model of a physical system. There are always phenomena which we will not be able to model. Thus, there will always be model errors or model uncertainties. But even if a model describes just a part of the reality it can be very useful for analysis and design – if it describes the dominating dynamic properties of the system.¹

This chapter describes modeling based on physical principles. Models can also be developed from experimental (historical) data. This way of mathematical modeling is called system identification. It is not covered by the present book (an introduction is given in [2]).

¹This is expressed as follows in [5]: “All models are wrong, but some are useful.”

3.2 A procedure for mathematical modeling

Below is described a procedure for developing dynamic mathematical models for physical systems:

1. **Define systems boundaries.** All physical systems works in interaction with other systems. Therefore it is necessary to define the boundaries of the system before we can begin developing a mathematical model for the system, but in most cases defining the boundaries is done quite naturally.
2. **Make simplifying assumptions.** One example is to assume that the temperature in a tank is the same everywhere in the tank, that is, there are homogeneous conditions in the tank.
3. **Use the Balance law for the physical balances in the system, and define eventual additional conditions.** The Balance law is as follows:

The rate of change of inventory in the system is equal to inflows minus outflows plus rate of generated inventory.

See Figure 3.1. Here “inventory” is a general term. It can be

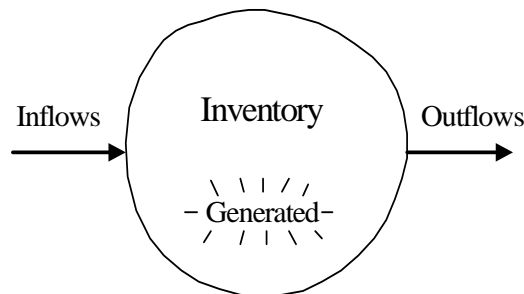


Figure 3.1: Illustration of the Balance law

(accumulated) mass, mole, energy, momentum, or electrical charge in a system. “Generated inventory” can be material generated by certain chemical reactions, or it can be generated energy in an exothermal reactor.

The Balance law can be expressed mathematically as follows:

$$\frac{d(\text{Inventory})}{dt} = \text{Inflows} - \text{Outflows} + \text{Generated} \quad (3.1)$$

The Balance law results in one or more *differential equations* due to the term d/dt . So, the model consists of differential equations.

You can define additional conditions to the model, like the requirement that the mass of liquid in a tank is not negative.

To actually calculate the amount of inventory in the system from the model, you just integrate the differential equation (3.1), probably using a computer program (simulation tool). The inventory at time t is

$$\text{Inventory}(t) = \text{Inventory}(0) + \int_0^t (\text{Inflows} - \text{Outflows} + \text{Generated}) d\theta \quad (3.2)$$

where $t = 0$ is the initial time.

4. **Draw an overall block diagram showing inputs, outputs and parameters.** A block diagram makes the model appear more clearly. Figure 3.2 shows an overall block diagram of an example system having two *manipulated variables*, two *environmental variables*, two *parameters*, and one *output variable*. The output variable is typically the inventory. The input variables are here separated into manipulating (adjustable) variables which you can use to manipulate or control the system (like power from an heater), and environmental variables which you can not manipulate (like environmental temperature). In the context of control systems, environmental variables are often denoted disturbance variables.

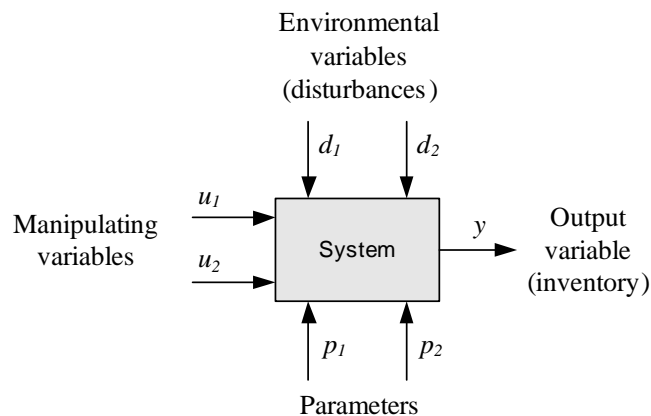


Figure 3.2: Overall block diagram

Mathematically, the input variables are *independent variables*, and the output variables are *dependent variables*.

The parameters of the model are quantities in the model which typically (but not necessarily) have constant values, like liquid density and spring constant. If a parameter have a varying value, it may alternatively be regarded as an environmental variable – you decide. To simplify the overall block diagram you can skip drawing the parameters.

5. **Present the model on a proper form.** The most common model forms are block diagrams (cf. Section 2.3), state models (cf. Section 2.4), and transfer functions (cf. Section 5). The choice of model form depends on the purpose of the model. For example, to tune a PID controller using Skogestad’s method (Sec. 10.3) you need a transfer function model of the process.

The following sections contains several examples of mathematical modeling. In the examples points 1 and 2 above are applied more or less implicitly.

3.3 Mathematical modeling of material systems

In a system where the mass may vary, mass is the “inventory” in the Balance law (3.1) which now becomes a *mass balance*:

$$\frac{dm(t)}{dt} = \sum_i F_i(t) \quad (3.3)$$

where m [kg] is the mass, and F_i [kg/s] is mass inflow (no. i).

Example 3.1 *Mass balance of a liquid tank*

Figure 3.3 shows a liquid tank with inflow and outflow. Assume that the inflow can be manipulated (controlled) with e.g. a pump, while the outflow is not manipulated. The density is the same all over, in the inlet, the outlet, and in the tank. We assume that the tank has straight, vertical walls. The symbols in Figure 3.3 are as follows: q_i is volumetric inflow. q_o is volumetric outflow. h is liquid level. A is cross sectional area. V is liquid volume. m is mass. ρ is density. The flows q_i and q_o and the mass m in the tank are variables. The parameters A and ρ are assumed to be constant.

We will develop a mathematical model which expresses how the mass m varies (as a function of time). We write the following mass balance for the mass in the tank:

$$\frac{dm(t)}{dt} = \rho q_i(t) - \rho q_o(t) \quad (3.4)$$

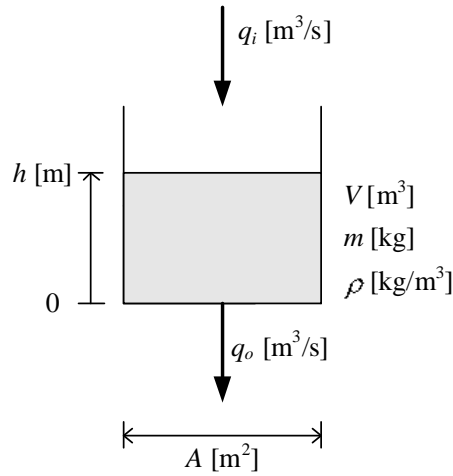


Figure 3.3: Example 3.1: Liquid tank

which is a differential equation for m . An additional condition for the differential equation is $m \geq 0$. (3.4) is a *mathematical model* for the system. ρ is a parameter in the model. Parameters are quantities which usually have a constant values and which characterizes the model.

(3.4) is a differential equation for the mass $m(t)$. Perhaps you are more interested in how level h will vary? The correspondence between h and m is given by

$$m(t) = \rho V(t) = \rho A h(t) \quad (3.5)$$

We insert this into the mass balance (3.4), which then becomes

$$\frac{dm(t)}{dt} = \frac{d[\rho V(t)]}{dt} = \frac{d[\rho A h(t)]}{dt} = \rho A \frac{dh(t)}{dt} = \rho q_i(t) - \rho q_o(t) \quad (3.6)$$

where ρ and A the parameters are moved outside the derivation (these are assumed to be constant). By cancelling ρ and dividing by A we get the following differential equation for $h(t)$:

$$\frac{dh(t)}{dt} = \dot{h}(t) = \frac{1}{A} [q_i(t) - q_o(t)] \quad (3.7)$$

with the condition $h_{\min} \leq h \leq h_{\max}$.

Figure 3.4 shows an *overall block diagram* for the model (3.7). q_i and q_o are *input variables*, which generally are variables which drive the system (we assume here that q_o is independent of the level, as when it is manipulated using a pump). h is an *output variable*, which generally is the variable which expresses the response or a state of the system. Note that q_o is an

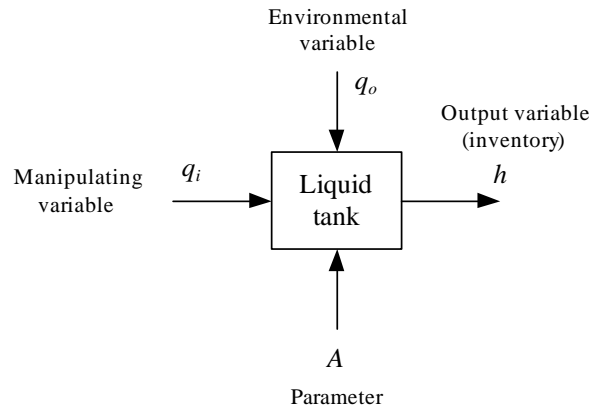


Figure 3.4: Example 3.1: Block diagram of a liquid tank

input variable despite it represents a physical output (outflow) from the tank!

Figure 3.5 shows a mathematical block diagram representing the model (3.7) using elementary blocks. The limits h_{\min} and h_{\max} are shown

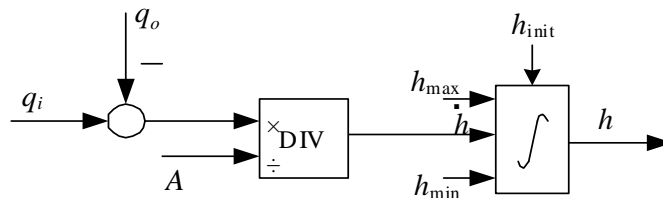


Figure 3.5: Example 3.1: Block diagram for the model (3.7)

explicitly as parameters to the integrator.

Figure 3.6 shows an alternative mathematical block diagram using a formula block in stead of the elementary blocks. Here, q_i enters the formula block at the left because it is a manipulated variable. q_o enter the block at the top because it is an environmental variable, and A enters at the bottom because it is a parameter.

Let us look at a simulation of the tank. The simulator is based on the model (3.7) and is implemented in LabVIEW. Figure 3.7 shows the input signals $q_i(t)$ and $q_o(t)$ and the corresponding time response in the output variable, $h(t)$. The model parameters are shown on the front panel of the simulator (see the figure). The time response is as expected: The level is

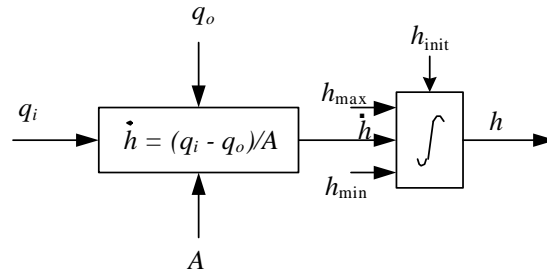


Figure 3.6: Example 3.1: Block diagram for the model (3.7)

steadily increasing when the inflow is larger than the outflow, it is constant when the inflow and the outflow are equal, and the level is decreasing when the inflow is smaller than the outflow.

[End of Example 3.1]

Material balance in the form of mole balance

The material balance can be in the form of a *mole balance*, as illustrated in Example 3.2 below.

Example 3.2 Mole balance

Figure 3.8 shows a stirred blending tank where the material A is fed into a tank for blending with a raw material.

The symbols in Figure 3.8 are as follows: V is the liquid volume in the tank. q is the volumetric inflow of the raw material. q is also the volumetric outflow. c_A is the mole density or concentration of material A in the tank. w_A is the mole flow of material A.

We will now develop a mathematical model which expresses how the concentration c_A varies. We make the following assumptions:

- The blending in the tank has constant volume.²
- The volumetric flow of material A is very small (negligible) compared to the volumetric flow of the raw material.

²This can be accomplished with for example a level control system.

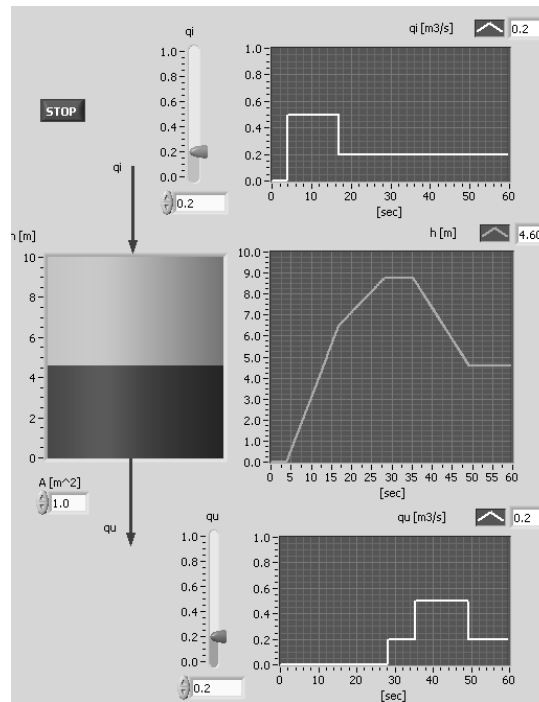


Figure 3.7: Example 3.1: Simulator of the level $h(t)$ in a liquid tank

- There are homogenous conditions (perfect stirring) in the tank.
- The raw material does not contain A.

Mole balance (the total mole number is Vc_A) yields

$$\frac{d[Vc_A(t)]}{dt} = w_A(t) - c_A(t)q(t) \quad (3.8)$$

By taking V (which is constant) outside the differentiation and then dividing by V on both sides of the equation, we get the following differential equation for c_A :

$$\frac{dc_A(t)}{dt} = \dot{c}_A(t) = \frac{1}{V} [w_A(t) - c_A(t)q(t)] \quad (3.9)$$

with the condition $c_A \geq 0$.

Figure 3.9 shows an overall block diagram of the model (3.9). w_A and q are input variables, and c_A is the output variable.

Figure 3.10 shows a simulation of the blending tank. The simulator is based on the model (3.9). The model parameters are shown on the front panel of the simulator (implemented in LabVIEW).

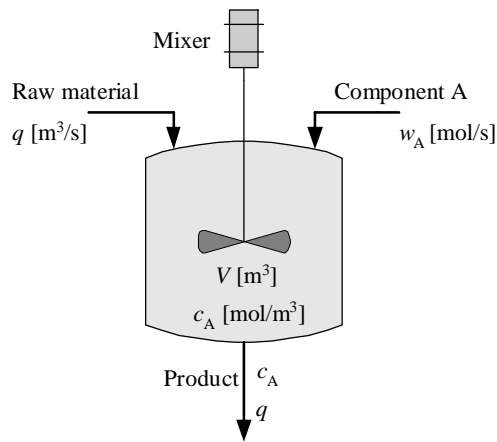


Figure 3.8: Example 3.2: Blending tank

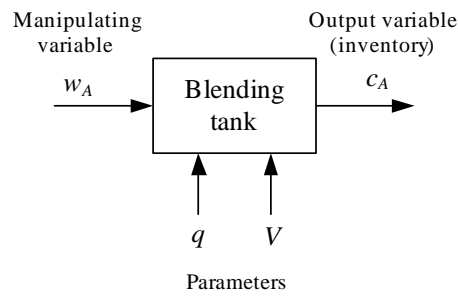


Figure 3.9: Example 3.2: Overall block diagram for stirred blending tank

[End of Example 3.2]

3.4 Mathematical modeling of thermal systems

Mathematical modeling of thermal systems is based on the Balance law to set up energy balances. The term *energy* covers temperature-dependent energy, which we can call thermal energy, and kinetic and potential energy. In general we must assume that there is a transformation from one energy form to another within a given system. For example, kinetic energy can be transformed to thermal energy via friction. For many thermal systems we can assume that the energy consists of only thermal energy and we can neglect the transformation from kinetic and potential energy to thermal energy.

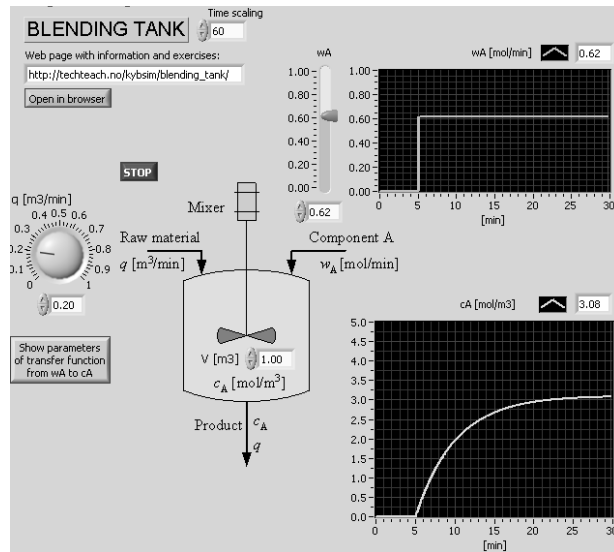


Figure 3.10: Example 3.2: Simulator of a blending tank

The Balance law (3.1) applied to a thermal system becomes an *energy balance*:

$$\frac{dE(t)}{dt} = \sum_i Q_i(t) \quad (3.10)$$

where E [J] is the thermal energy, and Q_i [J/s] is energy inflow no. i . The energy E is often assumed to be proportional to the temperature and the mass (or volume):

$$E = cmT = c\rho VT = CT \quad (3.11)$$

where T [K] is the temperature, c [J/(kg K)] is specific heat capacity, m [kg] is mass, V [m³] volume, ρ [kg/m³] is density, C [J/K] is total heat capacity.

Example 3.3 *Heated liquid tank*³

Figure 3.11 shows a liquid tank with continuous liquid inflow and outflow. There is heat transfer with the environment through the walls. the liquid delivers power through a heating element. P is power from the heating element. T is temperature in the tank and in the outlet flow. T_i is the temperature in the inlet flow. F is mass flow. m is mass of liquid (constant). c is specific heat capacity. U_h is total heat transfer coefficient.

³This example is used in several sections in later chapters.

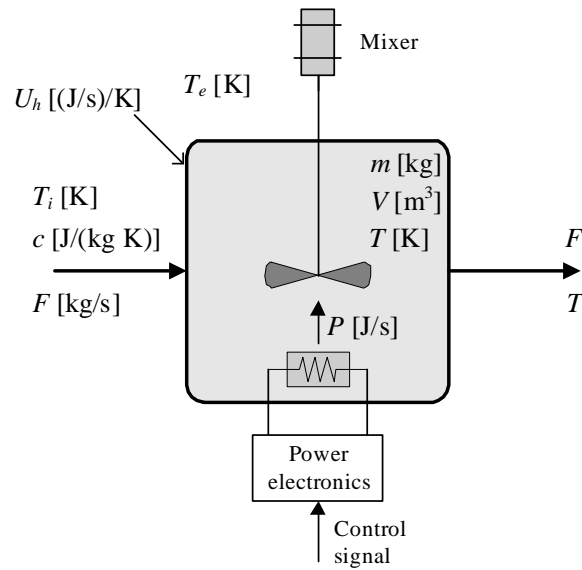


Figure 3.11: Example 3.3: Heated liquid tank

We will now set up an energy balance for the liquid in the tank to find the differential equation which describes the temperature $T(t)$. We will then make the following assumptions:

- The temperature in the liquid in the tank is homogeneous (due to the stirring machine).
- The inflow and in the outflow are equal, and the tank is filled by liquid.
- There is no storage of thermal energy in the heating element itself. This means that all of the supplied power to the heating element is supplied (immediately) to the liquid. (Thus, we do not write an energy balance for the heating element.)

The energy balance is based on the following energy transports (power):

1. Power from the heating element:

$$P(t) = Q_1 \quad (3.12)$$

2. Power from the inflow:

$$cF(t)T_i(t) = Q_2 \quad (3.13)$$

3. Power removed via the outflow:

$$-cF(t)T(t) = Q_3 \quad (3.14)$$

4. Power via heat transfer from (or to) the environment:

$$U_h [T_e(t) - T(t)] = Q_4 \quad (3.15)$$

The energy balance is

$$\frac{dE(t)}{dt} = Q_1 + Q_2 + Q_3 + Q_4 \quad (3.16)$$

where the energy is given by

$$E(t) = cmT(t)$$

The energy balance can then be written as (here the time argument t is dropped for simplicity):

$$\frac{d(cmT)}{dt} = P + cFT_i - cFT + U_h (T_e - T) \quad (3.17)$$

If we assume that c and m are constant, we can move cm outside the derivative term. Furthermore, we can combine the the terms on the right side. The result is

$$cm \frac{dT}{dt} = cm\dot{T} = P + cF (T_i - T) + U_h (T_e - T) \quad (3.18)$$

which alternatively can be written

$$\frac{dT}{dt} = \dot{T} = \frac{1}{cm} [P + cF (T_i - T) + U_h (T_e - T)] \quad (3.19)$$

Figure 3.12 shows an overall block diagram of the model (3.19). P and T_i are input variables, and T is an output variable.

Figure 3.13 shows a simulation of the tank. At time 40 min there is a positive step in the supplied power P , and at time 120 min there is a negative step in the inlet temperature T_i . The model parameters are shown on the front panel of the simulator which is implemented in LabVIEW. The simulator is based on the model (3.19).

[End of Example 3.3]

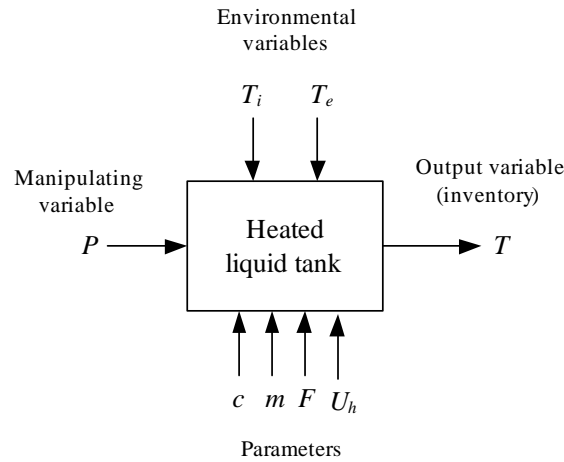


Figure 3.12: Example 3.3: Block diagram of heated tank

3.5 Mathematical modeling of motion systems

3.5.1 Systems with linear motion

The Balance law (3.1) applied to a body with linear motion (we will soon study rotational motion) is a *momentum balance*, which is often denoted *force balance*:

$$\frac{d[I(t)]}{dt} = \frac{d[m(t)v(t)]}{dt} = \sum_i F_i(t) \quad (3.20)$$

where I [Ns] is the momentum (mass times speed), and F_i is force (no. i). I is

$$I = mv = m\dot{x} \quad (3.21)$$

where m [kg] is mass, v [m/s] is speed, and x [m] is position.

If m is constant, m can be moved outside the derivative term in (3.20), which then becomes

$$m\dot{v}(t) = m\ddot{x}(t) = ma(t) = \sum_i F_i(t) \quad (3.22)$$

where $\dot{v} = \ddot{x} = a$ is acceleration. (3.22) is the well-known Newton's second law (the sum of forces is equal to mass times acceleration).

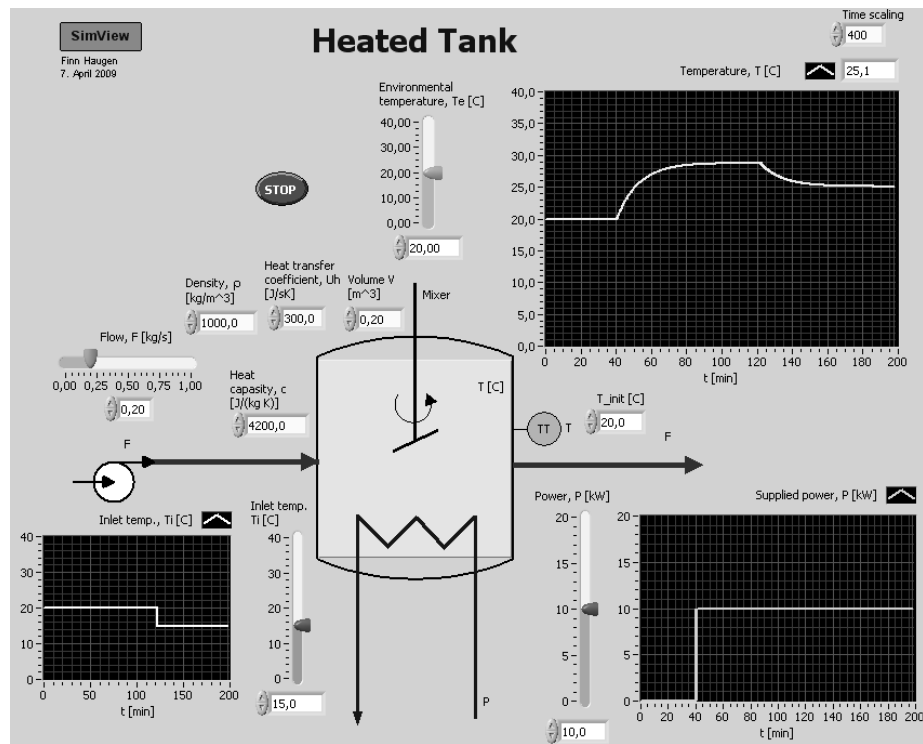


Figure 3.13: Example 3.3: Simulation of a heated tank

Often the mass m is constant. Then (3.22) can be used for mathematical modeling. But if m is time-varying, (3.20) must be used. One example of a system with *varying* mass is a conveyor belt where the mass on the belt is varying.

Example 2.1 (page 25) about the mass-spring-damper system is an example of a system with linear motion.

3.5.2 Systems with rotational motion

Systems with rotational motion can be modelled in the same way as systems with linear motion (see above), but we must use *momentum balance*, which is often denoted *torque balance* for rotational systems:

$$\frac{d[S(t)]}{dt} = \frac{d[J(t)\omega(t)]}{dt} = \sum_i T_i(t) \quad (3.23)$$

Here, S [Nms] is momentum, J [kgm²] is inertia, ω [rad/s] is rotational

speed, and T_i is torque (no. i). If J is constant, (3.23) can be written

$$J\dot{\omega}(t) = J\ddot{\theta}(t) = \sum_i T_i(t) \quad (3.24)$$

where $\dot{\omega} = \ddot{\theta}$ is angular acceleration, and θ [rad] is angular position.

Relations between rotational and linear motion

In mathematical modeling of mechanical systems which consists of a combination of rotational and linear systems, the following relations are useful: Torque T is force F times arm l :

$$T = Fl \quad (3.25)$$

Arc b is angle θ (in radians) times radius r :

$$b = \theta r \quad (3.26)$$

Coupled mechanical systems

Mechanical systems often consist of coupled (sub)systems. Each system can have linear and/or rotational motion. Some examples: (1) A robot manipulator where the arms are coupled. (2) A traverse crane where a wagon moves a pending load. (3) A motor which moves a load with linear motion, as in a lathe machine.

A procedure for mathematical modeling of such coupled systems is as follows:

1. The force or torque balance is put up for each of the (sub)systems, and internal forces and torques acting between the systems are defined.
2. The final model is derived by eliminating the internal forces and torques.

This procedure is demonstrated in Example 3.4. An alternative way of modeling coupled systems is to use *Lagrange mechanics* where the model (the equations of motion) are derived from an expression which contains kinetic and potential energy for the whole system (this method is not described here).

Example 3.4 Modeling coupled rotational and linear motion systems

Figure 3.14 shows an electro-motor (which can be a current-controlled DC-motor) which moves a load with linear motion via a gear and a rod.

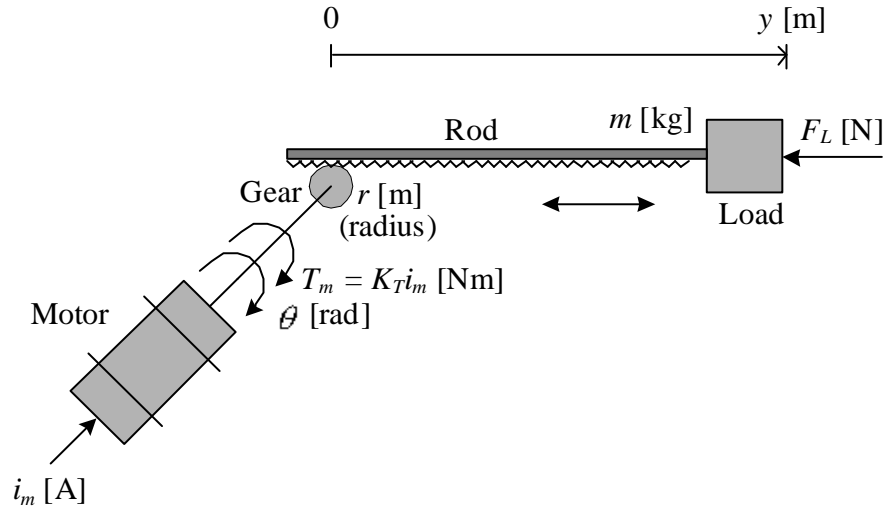


Figure 3.14: Example 3.4: Motor moving a linear load via a gear and a rod

We set up a torque balance for the rotational part of the system and a force balance for the linear part, and then combines the derived equations. We shall finally have model which expresses the position y of the tool as a function of the signal i . (For simplicity the time argument t is excluded in the expressions below.)

1. **Torque and force balance:** The torque balance for the motor becomes

$$J\ddot{\theta} = K_T i_m - T_1 \quad (3.27)$$

where T_1 is the torque which acts on the motor from the rod and the load via the gear. The force balance for rod and load becomes

$$m\ddot{y} = F_1 - F_L \quad (3.28)$$

where F_1 is the force which acts on the rod and the load from the motor via the gear. The relation between T_1 and F_1 is given by

$$T_1 = F_1 r \quad (3.29)$$

The relation between y and θ is given by

$$y = \theta r \quad (3.30)$$

which yields

$$\ddot{\theta} = \frac{\ddot{y}}{r} \quad (3.31)$$

By setting (3.31) and (3.29) into (3.27), (3.27) can be written

$$J \frac{\ddot{y}}{r} = K_T i_m - F_1 r \quad (3.32)$$

2. **Elimination of internal force:** By eliminating the internal force F_1 between (3.28) and (3.32), we get

$$\left(m + \frac{J}{r^2} \right) \ddot{y}(t) = \frac{K_T}{r} i_m(t) - F_L(t) \quad (3.33)$$

which is a mathematical model for the coupled system.

Figure 3.15 shows a block diagram for the model (3.33). i_m and F_L are input variables, and y is the output variable.

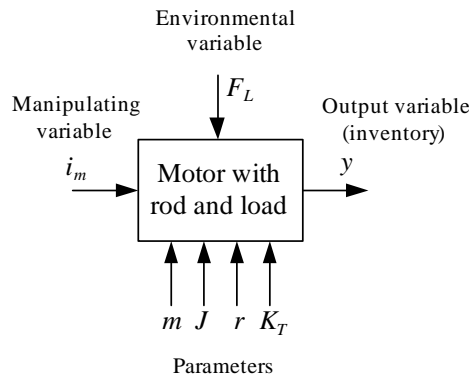


Figure 3.15: Block diagram of motor with rod and load

[End of Example 3.4]

3.6 Mathematical modeling of electrical systems

Here is a summary of some fundamental formulas for electrical systems which you will probably use in mathematical modeling of electrical systems:

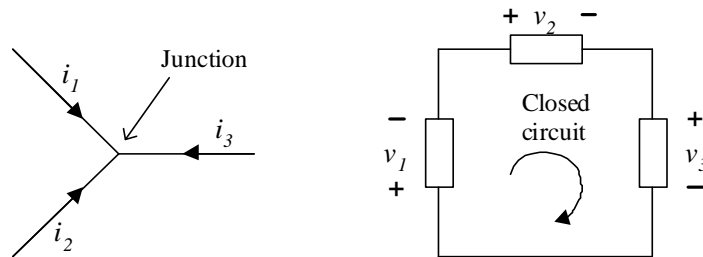


Figure 3.16: Kirchhoff's laws

- **Kirchhoff's laws:**

- *Kirchhoff's current law:* See the left part of Figure 3.16. The sum of currents into a junction in an electrical circuit is zero:

$$i_1 + i_2 + i_3 + \dots = 0 \quad (3.34)$$

- *Kirchhoff's voltage law:* See the right part of Figure 3.16. The sum of voltage drops over the components on a closed electrical loop is equal to zero:

$$v_1 + v_2 + v_3 + \dots = 0 \quad (3.35)$$

- **Resulting resistance for series and parallel combination of resistors:** See Figure 3.17.

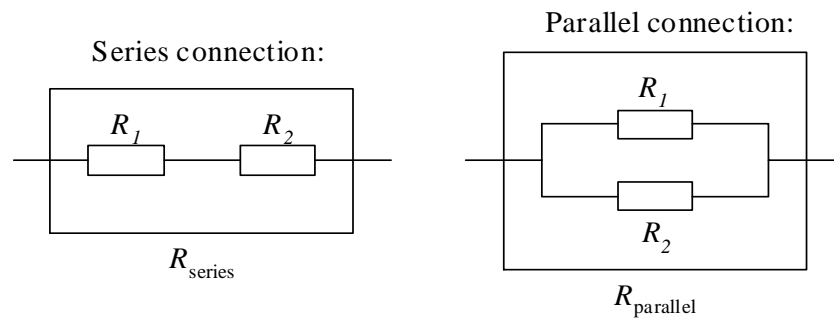


Figure 3.17: Series and parallel combination of resistors

- *Resistors in series:* Given two resistors R_1 and R_2 [Ω] in a series combination. The resulting resistance is

$$R_{\text{series}} = R_1 + R_2 \quad (3.36)$$

- *Resistors in parallel*: Given two resistors R_1 and R_2 [Ω] in a parallel combination. The resulting resistance is

$$R_{\text{parallel}} = \frac{R_1 \cdot R_2}{R_1 + R_2} \quad (3.37)$$

- **Relation between current and voltage for resistor, capacitor and inductor**: See Figure 3.18. Suppose that the current through a

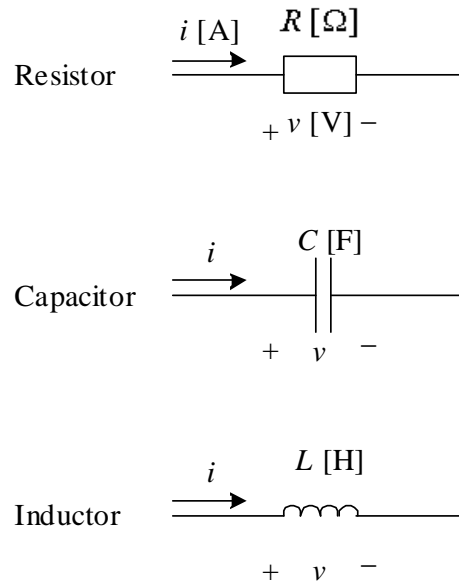


Figure 3.18: Resistor, capacitor and inductor

component is I [A] and that the corresponding voltage drop over the component v [V]. Current and voltage are then related as follows:

- *Resistor*:

$$v(t) = Ri(t) \quad (\text{Ohm's law}) \quad (3.38)$$

- *Capacitor*:

$$i(t) = C \frac{dv(t)}{dt} \quad (3.39)$$

- *Inductor*:

$$v(t) = L \frac{di(t)}{dt} \quad (3.40)$$

- **Power**:

- *Instantaneous power*: When a current i flows through a resistor R , the power delivered to the resistor is

$$P(t) = u(t)i(t) \quad (3.41)$$

where $u(t) = Ri(t)$ is the voltage drop across the resistor.

- *Mean power*: When an alternating (sinusoidal) current of amplitude I flows through a resistor R (for example a heating element), the mean or average value of the power delivered to the resistor is

$$\bar{P} = \frac{1}{2}UI = \frac{1}{2}RI^2 = \frac{1}{2}\frac{U^2}{R} \quad (3.42)$$

where U is the amplitude of the alternating voltage drop across the resistor. (3.42) is independent of the frequency.

Example 3.5 *Mathematical modeling of an RC-circuit*

Figure 3.19 shows an RC-circuit (the circuit contains the resistor R and the capacitor C). The RC-circuit is frequently used as an analog lowpass

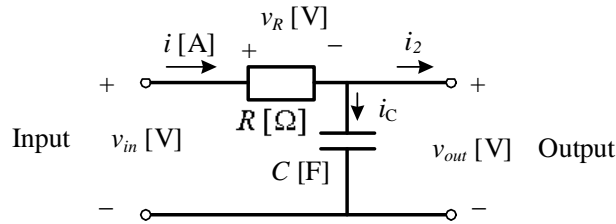


Figure 3.19: RC-circuit

filter: Signals of *low* frequencies *passes* approximately unchanged through the filter, while signals of high frequencies are approximately filtered out (stopped).

We will now find a mathematical model relating v_{out} to v_{in} . First we apply the Kirchhoff's voltage law in the circuit which consists the input voltage terminals, the resistor, and the capacitor (we consider the voltage drops to be positive clockwise direction):

$$-v_{in} + v_R + v_{out} = 0 \quad (3.43)$$

(v_{out} equals the voltage drop over the capacitor.) In (3.43) v_R is given by

$$v_R = Ri \quad (3.44)$$

We assume that there is no current going through the output terminals. (This is a common assumption, and not unrealistic, since it is typical that the output terminals are connected to a subsequent circuit which has approximately infinite input impedance, causing the current into it to be approximately zero. An operational amplifier is an example of such a load-circuit.) Thus, jf. (3.39),

$$i = i_C = C\dot{v}_{out} \quad (3.45)$$

The final model is achieved by using i as given by (3.45) in (3.44) and then using v_R as given by (3.44) for v_R in (3.43). The model becomes

$$RC\dot{v}_{out}(t) = v_{in}(t) - v_{out}(t) \quad (3.46)$$

Figure 3.20 shows a block diagram for the model (3.46). v_{in} is the input variable, and v_{out} is the output variable.

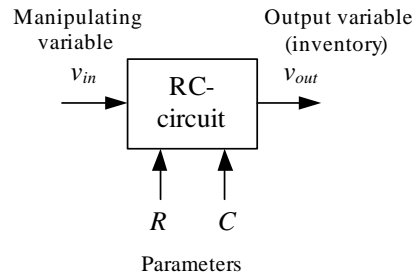


Figure 3.20: Block diagram of an RC-circuit

Figure 3.21 shows the front panel of a simulator of the RC-circuit.

[End of Example 3.5]

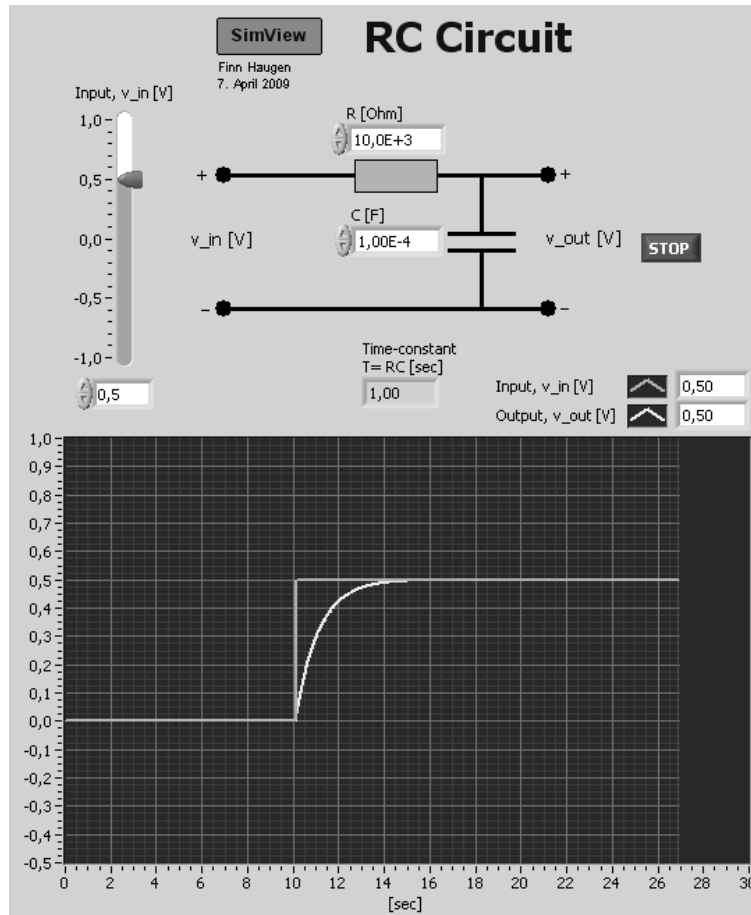


Figure 3.21: Example 3.5: Simulator for an RC-circuit. The input voltage is changed as a step, and the step response in the output voltage is simulated.

Chapter 4

The Laplace transform

4.1 Introduction

The Laplace transform is a mathematical tool which is useful in systems theory. It is the foundation of transfer functions which is a standard model form of dynamic systems. Transfer functions are described in Chapter 5. Furthermore, with the Laplace transform you relatively easily calculate responses in dynamic systems by hand.¹

In this chapter I present the Laplace transform at a minimum level. You can find much more information in a mathematics text-book.

4.2 Definition of the Laplace transform

Given a time-evaluated function $f(t)$ – that is, $f(t)$ is a function of time t . It can be a sinusoid, a ramp, an impulse, a step, a sum of such functions, or any other function of time. The Laplace transform of $f(t)$ can be denoted $F(s)$, and is given by the following integral:

$$F(s) = \mathcal{L}\{f(t)\} = \int_0^{\infty} e^{-st} f(t) dt \quad (4.1)$$

¹It turns out that we rarely need to perform manual calculations of the responses. When we need to know the responses, it is in most situations more convenient to obtain them by simulating the system. With the Laplace transform you can calculate responses only for *linear systems*, that is, systems having a model which can be expressed as a linear differential equation.

Expressed with words, $f(t)$ is multiplied by the weight function e^{-st} , and the resulting product $e^{-st}f(t)$ is integrated from start time $t = 0$ to end time $t = \infty$. The Laplace transform does not care about any value that $f(t)$ might have at negative values of time t . In other words, you can think of $f(t)$ as being “switched on” at $t = 0$. (The so-called two-sided Laplace transform is defined also for negative time, but it is not relevant for our applications.)

s is the Laplace variable.² $F(s)$ is a function of s . The time t is not a variable in $F(s)$ – it disappeared through the time integration. $F(s)$ will look completely different from $f(t)$, cf. the following example.

Example 4.1 *Laplace transform of a step*

Given the function

$$f(t) = 1 \text{ (for } t \geq 0) \quad (4.2)$$

which is a step of amplitude 1 at time $t = 0$. Using (4.1), its Laplace transform becomes

$$F(s) = \int_0^{\infty} e^{-st} \cdot 1 \cdot dt = \left[-\frac{1}{s} e^{-st} \right]_{t=0}^{t=\infty} = \frac{1}{s} \quad (4.3)$$

[End of Example 4.1]

Calculating the time function $f(t)$ from its Laplace transform $F(s)$ – in other words: going from s to t – is denoted *inverse Laplace transform*. This can be expressed as

$$f(t) = \mathcal{L}^{-1} \{F(s)\} \quad (4.4)$$

The inverse Laplace transform is actually defined by a complicated complex integral.³ If you *really* want to calculate this integral, you should use the Residue Theorem in mathematics. However, I suggest you in stead try the simplest method, namely to find $f(t)$ from the precalculated Laplace transform pairs, cf. Section 4.3, possibly combined with one or more of the Laplace transform properties, cf. Section 4.4.

²You may wonder what is the physical meaning of s . It can be interpreted as a complex frequency, but I think the best answer is that there is no meaningful physical meaning.

³ $f(t) = \frac{1}{2\pi j} \int_{\sigma-j\infty}^{\sigma+j\infty} F(s)e^{st} ds$

4.3 Laplace transform pairs

$F(s)$ and $f(t)$ in Example 4.1 can be denoted a *Laplace transform pair*:

$$F(s) = \frac{1}{s} \iff f(t) = 1 \quad (\text{step of amplitude 1}) \quad (4.5)$$

Below are some additional Laplace transform pairs. Each time function, $f(t)$, is defined for $t \geq 0$. $F(s)$ can be derived from (4.1). In the expressions below, k is some constant. For example, (4.5) is (4.7) with $k = 1$.

Laplace transform pairs:

$$F(s) = k \iff f(t) = k\delta(t) \quad (\text{impulse of strength or area } k) \quad (4.6)$$

$$\frac{k}{s} \iff k \quad (\text{step of amplitude } k) \quad (4.7)$$

$$\frac{k}{s^2} \iff kt \quad (\text{ramp of slope } k) \quad (4.8)$$

$$k \frac{n!}{s^{n+1}} \iff kt^n \quad (4.9)$$

$$\frac{k}{Ts + 1} \iff \frac{ke^{-t/T}}{T} \quad (4.10)$$

$$\frac{k}{(Ts + 1)s} \iff k(1 - e^{-t/T}) \quad (4.11)$$

$$\frac{k}{(T_1s + 1)(T_2s + 1)} \iff \frac{k}{T_1 - T_2} (e^{-t/T_1} - e^{-t/T_2}) \quad (4.12)$$

$$\frac{k}{(T_1s + 1)(T_2s + 1)s} \iff k \left[1 + \frac{1}{T_2 - T_1} (T_1e^{-t/T_1} - T_2e^{-t/T_2}) \right] \quad (4.13)$$

4.4 Laplace transform properties

In calculations with the Laplace transform you will probably need one or more of the Laplace transform *properties* presented below.⁴ We will definitely use some of them for deriving transfer functions, cf. Chapter 5. Each of these properties can be derived from the Laplace transform definition (4.1).

Linear combination:

$$k_1 F_1(s) + k_2 F_2(s) \iff k_1 f_1(t) + k_2 f_2(t) \quad (4.14)$$

Special case: Multiplication by a constant:

$$kF(s) \iff kf(t) \quad (4.15)$$

Time delay:

$$F(s)e^{-\tau s} \iff f(t - \tau) \quad (4.16)$$

Time derivative:

$$s^n F(s) - s^{n-1} f(0) - s^{n-2} \dot{f}(0) - \dots - \binom{n-1}{f} f(0) \iff \binom{n}{f}(t) \quad (4.17)$$

Special case: Time derivative with zero initial conditions:

$$s^n F(s) \iff \binom{n}{f}(t) \quad (4.18)$$

Special case: Time derivative with non-zero initial condition:

$$sF(s) - f_0 \iff \dot{f}(t) \quad (4.19)$$

Special case: First order time derivative with zero initial condition:

$$sF(s) \iff \dot{f}(t) \quad (4.20)$$

(So, differentiation corresponds to multiplication by s .)

⁴Additional properties could have been given here, too, but the ones presented are the most useful.

Integration:

$$\frac{1}{s}F(s) \iff \int_0^t f(\tau)d\tau \quad (4.21)$$

(So, integration corresponds to division by s .)

Final Value Theorem:

$$\lim_{s \rightarrow 0} sF(s) \iff \lim_{t \rightarrow \infty} f(t) \quad (4.22)$$

Example 4.2 Calculation of time response (inverse Laplace transform)

Given the following differential equation:

$$\dot{y}(t) = -2y(t) \quad (4.23)$$

with initial value $y(0) = 4$. Calculate $y(t)$ using the Laplace transform.

To calculate $y(t)$ we start by taking the Laplace transform of both sides of the differential equation (4.23):

$$\mathcal{L}\{\dot{y}(t)\} = \mathcal{L}\{-2y(t)\} \quad (4.24)$$

Here, we apply the time derivative property (4.19) at the left side, and the linear combination property (4.15) to the right side, to get

$$sY(s) - 4 = -2Y(s) \quad (4.25)$$

Solving for $Y(s)$ gives

$$Y(s) = \frac{4}{s+2} \quad (4.26)$$

To get the corresponding $y(t)$ from this $Y(s)$ we look for a proper Laplace transform pair. (4.10) fits. We have to write our $Y(s)$ on the form of (4.10). Dividing both the numerator and the denominator by 2 gives

$$Y(s) = \frac{4}{s+2} = \frac{2}{0.5s+1} = \frac{k}{Ts+1} \quad (4.27)$$

Hence, $k = 2$ and $T = 0.5$. Finally, according to (4.10) $y(t)$ becomes

$$\underline{\underline{y(t)}} = \frac{ke^{-t/T}}{T} = \frac{2e^{-t/0.5}}{0.5} = \underline{\underline{4e^{-2t}}} \quad (4.28)$$

[End of Example 4.2]

Example 4.3 *Calculation of steady-state value using the Final Value Theorem*

See Example 4.2. The steady-state value of y in (4.28) is

$$y_s = \lim_{t \rightarrow \infty} y(t) = \lim_{t \rightarrow \infty} 4e^{-2t} = 0 \quad (4.29)$$

Using the Final Value Theorem we get

$$y_s = \lim_{s \rightarrow 0} sY(s) = \lim_{s \rightarrow 0} s \frac{4}{s+2} = 0 \quad (4.30)$$

So, the results are the same.

[End of Example 4.3]

Chapter 5

Transfer functions

5.1 Introduction

Transfer functions is a model form based on the Laplace transform, cf. Chapter 4. Transfer functions are very useful in analysis and design of linear dynamic systems, in particular controller functions and signal filters. The main reasons why transfer functions are useful are as follows:

- **Compact model form:** If the original model is a higher order differential equation, or a set of first order differential equations, the relation between the input variable and the output variable can be described by one transfer function, which is a rational function of the Laplace variable s , without any time-derivatives.
- **Representation of standard models:** Transfer functions are often used to represent standard models of controllers and signal filters.
- **Simple to combine systems:** For example, the transfer function for a combined system which consists of two systems in a series combination, is just the product of the transfer functions of each system.
- **Simple to calculate time responses:** The calculations will be made using the Laplace transform, and the necessary mathematical operations are usually much simpler than solving differential equations. Calculation of time-responses for transfer function models is described in Chapter 5.5.
- **Simple to find the frequency response:** The frequency response is a function which expresses how sinusoid signals are transferred

through a dynamic system. Frequency response is an important tool in analysis and design of signal filters and control systems. The frequency response can be found from the transfer function of the system. However, frequency response theory is not a part of this book (a reference is [2]).

Before we start, I must say something about the mathematical notation: In the following sections, and in the remainder of the book, I use the *same symbol* (letter) for the time function, say $y(t)$, as for the Laplace transform of $y(t)$, here $y(s)$ – although it is mathematically incorrect to do it. The reason is to simplify the presentation. Now, only one variable name (symbol) is needed for both the Laplace domain and the time domain. For example, assume that $y(t)$ is the time function of the level y in a water tank. Then I write $y(s)$, although I formally should have written $Y(s)$ or $y^*(s)$ or $\bar{y}(s)$ (or something else that is different from $y(s)$) for $\mathcal{L}\{y(t)\}$. It is my experience (from many years together with transfer functions) that this simplifying notation causes no problems.

5.2 Definition of the transfer function

The first step in deriving the transfer function of a system is taking the Laplace transform of the differential equation (which must be linear). Let us go on with an example, but the results are general. Given the following mathematical model having two input variables u_1 and u_2 and one output variable y . (I think you will understand from this example how to find the transfer function for systems with different number of inputs and outputs.)

$$\dot{y}(t) = ay(t) + b_1u_1(t) + b_2u_2(t) \quad (5.1)$$

a , b_1 and b_2 are model parameters (coefficients). Let the initial state (at time $t = 0$) be y_0 . We start by taking the Laplace transform of both sides of the differential equation:

$$\mathcal{L}\{\dot{y}(t)\} = \mathcal{L}\{ay(t) + b_1u_1(t) + b_2u_2(t)\} \quad (5.2)$$

By using the linearity property of the Laplace transform, cf. (4.14), the right side of (5.2) can be written as

$$\mathcal{L}\{ay(t)\} + \mathcal{L}\{b_1u_1(t)\} + \mathcal{L}\{b_2u_2(t)\} \quad (5.3)$$

$$= a\mathcal{L}\{y(t)\} + b_1\mathcal{L}\{u_1(t)\} + b_2\mathcal{L}\{u_2(t)\} \quad (5.4)$$

$$= ay(s) + b_1u_1(s) + b_2u_2(s) \quad (5.5)$$

The left side of (5.2) will be Laplace transformed using the differentiation rule, cf. (4.17), on $\mathcal{L}\{\dot{y}(t)\}$:

$$\mathcal{L}\{\dot{y}(t)\} = sy(s) - y_0 \quad (5.6)$$

Thus, we have found that the Laplace transformed (5.2) is

$$sy(s) - y_0 = ay(s) + b_1u_1(s) + b_2u_2(s) \quad (5.7)$$

Solving for the output variable $y(s)$ gives

$$y(s) = \underbrace{\frac{1}{s-a}}_{\frac{y_{\text{init}}(s)}{y_0}} y_0 + \underbrace{\frac{b_1}{s-a}}_{H_1(s)} u_1(s) + \underbrace{\frac{b_2}{s-a}}_{H_2(s)} u_2(s) \quad (5.8)$$

In (5.8),

- y_1 is the contribution from input u_1 to the total response y ,
- y_2 is the contribution from input u_2 to the total response y ,
- y_{init} is the contribution from the initial state y_0 to the total response y .

Of course, these contributions to the total response are in the Laplace domain. The corresponding responses in the time domain are found by calculating the inverse Laplace transforms.

Now we have the following two *transfer functions* for our system:

- The transfer function from u_1 to y is

$$H_1(s) = \frac{b_1}{s-a} \quad (5.9)$$

- The transfer function from u_2 to y is

$$H_2(s) = \frac{b_2}{s-a} \quad (5.10)$$

Thus, *the transfer function from a given input variable to a given output variable is the s -valued function with which the Laplace transformed input variable is multiplied to get its contribution to the response in the output*

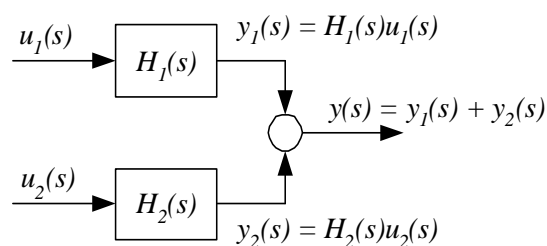


Figure 5.1: Block diagram representing the transfer functions $H_1(s)$ and $H_2(s)$ in (5.8)

variable. In other words: The transfer function expresses how the input variable is transferred through the system.

The transfer functions derived above can be illustrated with the block diagram shown in Figure 5.1.

One alternative way to express the definition of transfer function

From (5.8) we have

$$H_1(s) = \frac{b_1}{s - a} = \frac{y_1(s)}{u_1(s)} \quad (5.11)$$

So, we can define the transfer functions as the *ratio* between the Laplace transformed contribution to the total response in the output variable, here $y_1(s)$, and the Laplace transformed input variable, here $u_1(s)$. We may also say that the transfer functions is the ratio between the Laplace transformed total response in the output variable, here $y(s)$, and the Laplace transformed input variable, here $u_1(s)$, when all other inputs are set to zero and the initial state is zero.

5.3 Characteristics of transfer functions

A transfer function can be written on a factorized form – often called a *zero-pole form*:

$$H(s) = K \frac{(s - z_1)(s - z_2) \cdots (s - z_r)}{(s - p_1)(s - p_2) \cdots (s - p_n)} = \frac{b(s)}{a(s)} \quad (5.12)$$

Here z_1, \dots, z_r are the *zeros* and p_1, \dots, p_n are the *poles* of the transfer

function. For example, the transfer function

$$H(s) = \frac{4s - 4}{s^2 + 5s + 6} = 4 \frac{s - 1}{(s + 3)(s + 2)} \quad (5.13)$$

have two poles, -3 and -2 , one zero, 1 , and the gain is 4 . (As shown in e.g. [2] the values of the poles determines the stability property of a system. The system is stable only if all the poles have negative real parts, in other words if all the poles lie in the left half part of the complex plane. But we will not go into any stability analysis here.)

The s -polynomial in the denominator of $H(s)$, which is $a(s)$ in (5.12), is denoted the *characteristic polynomial*. The poles are the roots of the characteristic polynomial, that is

$$a(s) = 0 \text{ for } s = s_1, s_2, \dots, s_n \text{ (the poles)} \quad (5.14)$$

The *order* of a transfer function is the order of the characteristic polynomial. For example, the transfer function (5.13) has order 2.

5.4 Combining transfer functions blocks in block diagrams

Transfer function blocks may be combined in a block diagram according to the rules shown in Figure 5.2. One possible purpose of such a combination is to simplify the block diagram, or to calculate the resulting or overall transfer function. For example, the combined transfer function of two transfer functions connected in series is equal to the product of the individual transfer functions, i.e. the Series connection rule in Figure 5.2.

5.5 How to calculate responses from transfer function models

It is quite easy to calculate responses in transfer function models manually (by hand). Assume given the following transfer function model:

$$y(s) = H(s)u(s) \quad (5.15)$$

To calculate the time-response $y(t)$ for a given input signal $u(t)$, we can do as follows:

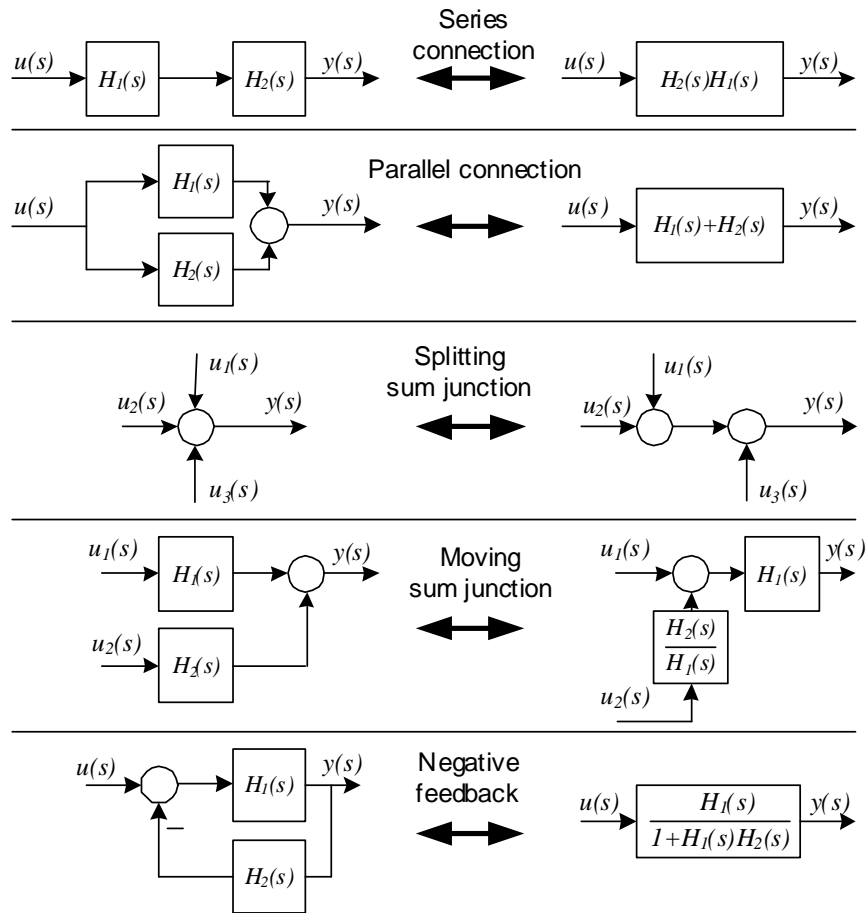


Figure 5.2: Rules for combining transfer function blocks

1. First, find $u(s)$ – the Laplace transform of the input signal. $u(s)$ can be found from precalculated Laplace transform pairs, cf. Section 4.3, possibly combined with one or more of the Laplace transform properties, cf. Section 4.4, where particularly the linearity property (4.14) is useful.

2. The Laplace transform of the output signal, $y(s)$, is calculated from (5.15), that is,

$$y(s) = H(s)u(s) \quad (5.16)$$

where $u(s)$ is found as explained above.

3. The time-function $y(t)$ is calculated as the inverse Laplace transform of $y(s)$, cf. Chapter 4.

Example 5.1 *Calculation of time-response for transfer function model*

Given the transfer function model

$$y(s) = \underbrace{\frac{3}{s+0.5}}_{H(s)} u(s) \quad (5.17)$$

Suppose that $u(t)$ is a step from 0 to 2 at $t = 0$. We shall find an expression for the time-response $y(t)$. The Laplace transform of $u(t)$ is, cf. (4.7),

$$u(s) = \frac{2}{s} \quad (5.18)$$

Inserting this into (5.17) gives

$$y(s) = \frac{3}{s+0.5} \cdot \frac{2}{s} = \frac{6}{(s+0.5)s} = \frac{12}{(2s+1)s} \quad (5.19)$$

(5.19) has the same form as the Laplace transform pair (4.11) which is repeated here:

$$\frac{k}{(Ts+1)s} \iff k \left[1 - e^{-t/T} \right] \quad (5.20)$$

Here $k = 12$ and $T = 2$. The time-response becomes

$$y(t) = 12 \left[1 - e^{-t/2} \right] \quad (5.21)$$

Figure 5.3 shows $y(t)$. The steady-state response is 12, which can be calculated from $y(t)$ by setting $t = \infty$.

[End of Example 5.1]

5.6 Static transfer function and static response

Suppose that the input signal to a system is a step of amplitude u_s . The corresponding static time-response can be found from the Final Value Theorem:

$$y_s = \lim_{s \rightarrow 0} s \cdot y(s) = \lim_{s \rightarrow 0} s \cdot H(s) \frac{u_s}{s} = \lim_{s \rightarrow 0} \underbrace{H(s)}_{H_s} u_s \quad (5.22)$$

where H_s is the *static transfer function*. That is,

$$H_s = \lim_{s \rightarrow 0} H(s) \quad (5.23)$$

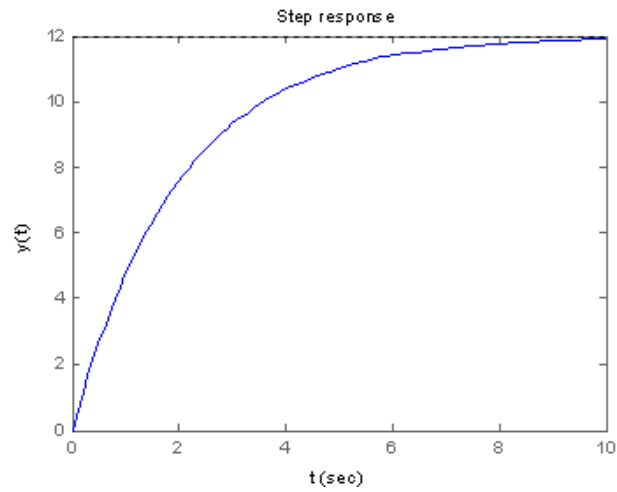


Figure 5.3: Example 5.1: The time-response $y(t)$ given by (5.21)

Thus, the static transfer function, H_s , is found by letting s approach zero in the transfer function, $H(s)$.

Once we know the static transfer function H_s the static (steady-state) response y_s due to a constant input of value u_s , is

$$y_s = H_s u_s \quad (5.24)$$

Example 5.2 *Static transfer function and static response*

See Example ???. The transfer function is

$$H(s) = \frac{3}{s + 0.5} \quad (5.25)$$

The corresponding static transfer function becomes

$$H_s = \lim_{s \rightarrow 0} H(s) = \lim_{s \rightarrow 0} \frac{3}{s + 0.5} = 6 \quad (5.26)$$

Assume that the input u has the constant value of $u_s = 2$. What is the corresponding static response y_s in the output? It can be calculated from the static transfer function as

$$y_s = H_s u_s = 6 \cdot 2 = 12 \quad (5.27)$$

which is the same result as found in Example 5.1.

[End of Example 5.2]

Chapter 6

Dynamic characteristics

6.1 Introduction

In this chapter a number of standard dynamic models in the form of transfer functions will be defined. With such standard models you can characterize the dynamic properties of a physical system in terms of for example gain, time-constant, and time-delay. These terms are also useful for controller tuning, as in the Skogestad's tuning method which is described in Section 10.3.

This chapter covers integrators, time-constant systems and time-delays. Second order systems, which may show oscillatory responses, are not covered.¹

6.2 Integrators

An integrator is a system where *the output variable y is the time integral of the input variable u* , multiplied with some gain K :

$$y(t) = K \int_0^t u d\theta \quad (6.1)$$

¹Because I have found that the theory about second order systems is not important in most applications. However, an article about second order systems is available at <http://techteach.no>.

Taking the time derivative of both sides of (6.1) yields the following differential equation describing an integrator:

$$\dot{y}(t) = Ku(t) \quad (6.2)$$

Taking the Laplace transform using (4.20) gives

$$sy(s) = Ku(s) \quad (6.3)$$

which gives the following *transfer function* of an integrator:

$$H(s) = \frac{y(s)}{u(s)} = \frac{K}{s} \quad (6.4)$$

Let us now find the *step response* of the integrator. We assume that $u(t)$ is a step of amplitude U at $t = 0$. From (4.7) $u(s) = \frac{U}{s}$. Thus,

$$y(s) = H(s)u(s) = \frac{K}{s} \cdot \frac{U}{s} = \frac{KU}{s^2} \quad (6.5)$$

which, inverse Laplace transformed using (4.8), is

$$y(t) = KUt \quad (6.6)$$

Thus, the step response of an integrator is a *ramp* with rate KU . Figure 6.1 shows simulated response of an integrator.

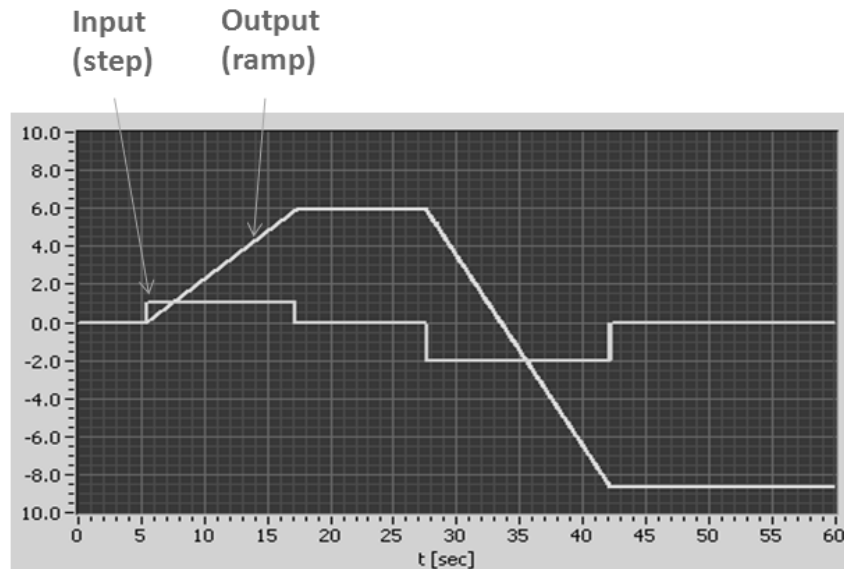


Figure 6.1: Simulated response of an integrator

Example 6.1 An integrator: A liquid tank

See Example 3.1 on page 36 which describes a liquid tank. Assume for simplicity that there is no outflow from the tank. The mathematical model of this system is then

$$\dot{h}(t) = \frac{1}{A}q_i(t) \quad (6.7)$$

Taking the Laplace transform of (6.7) gives

$$sh(s) - h_0 = \frac{1}{A}q_i(s) \quad (6.8)$$

which gives

$$h(s) = \frac{h_0}{s} + \underbrace{\frac{1}{As}}_{H(s)} q_i(s) \quad (6.9)$$

So, the transfer function is

$$H(s) = \frac{h(s)}{q_i(s)} = \frac{1}{A} \cdot \frac{1}{s} \quad (6.10)$$

The system is an integrator!

It is actually quite naturally that the liquid tank is an integrator, since the level is proportional to the integral of the inflow. This can be seen by integrating (6.7), which gives

$$h(t) = h(0) + \int_0^t \frac{1}{A}q_i(\theta) d\theta \quad (6.11)$$

Whenever you need a concrete example of an integrator, recall the tank!

[End of Example 6.1]

6.3 Time-constant systems

In *time-constant* system – also denoted *first order systems* – the output variable y and the input variable u are related according to the following differential equation:

$$Ty(t) + y(t) = Ku(t) \quad (6.12)$$

Here K is the *gain*, and T is the *time-constant*.

Taking the Laplace transform of both sides of (6.12) gives

$$y(s) = \underbrace{\frac{K}{Ts + 1}}_{H(s)} u(s) \quad (6.13)$$

where $H(s)$ is the transfer function. Here is an example:

$$H(s) = \frac{3}{4s + 2} = \frac{1.5}{2s + 1} \quad (6.14)$$

The gain is $K = 1.5$, and the time-constant is $T = 2$ (in a proper time unit, e.g. seconds).

Let us study the *step response* of a first order system. We assume that the input signal $u(t)$ is a step of amplitude U at time $t = 0$. From (4.7) $u(s) = \frac{U}{s}$. The Laplace transformed response becomes

$$y(s) = H(s)u(s) = \frac{K}{Ts + 1} \cdot \frac{U}{s} \quad (6.15)$$

Taking the inverse Laplace transform using (4.11) gives

$$y(t) = KU(1 - e^{-\frac{t}{T}}) \quad (6.16)$$

Let us study the importance of the parameters K and T for the step response. It is assumed that the system is in an operating point where the input is u_0 and the output is y_0 before the step in the input u . Figure 6.2 shows a simulation of a first order system. The parameters used in the simulation are $U = 1$, $K = 2$, and $T = 5$.

- **Importance of K :** The steady-state response due to the input step is

$$y_s = \lim_{t \rightarrow \infty} y(t) = KU \quad (6.17)$$

which is found from (6.16) with $t \rightarrow \infty$. Thus, the step is amplified with the gain K in steady-state. This is confirmed in Figure 6.2.

In Section 5.6 the static transfer function H_s was defined. What is the H_s of a time-constant system? We get

$$H_s = \frac{y_s}{u_s} = \frac{KU}{U} = K \quad (6.18)$$

So, the H_s is equal to the gain parameter, K .

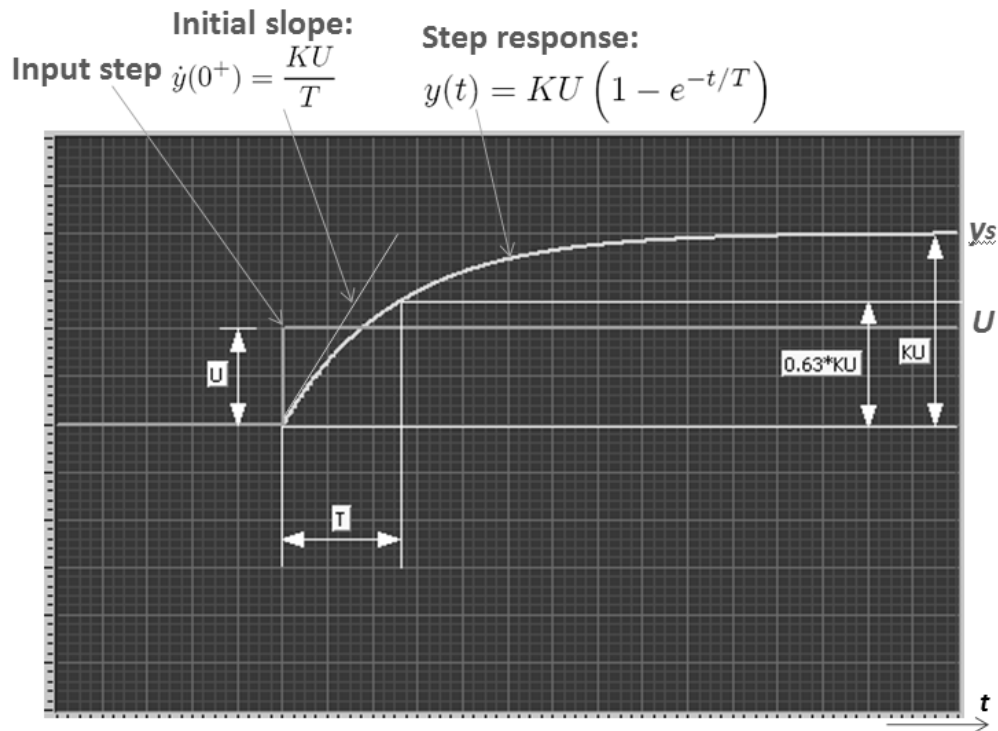


Figure 6.2: Step response of a time-constant system

- **Importance of T :** Let us set $t = T$ in (6.16):

$$y(T) = KU(1 - e^{-\frac{T}{T}}) \quad (6.19)$$

$$= KU(1 - e^{-1}) \quad (6.20)$$

$$= 0.63 \cdot KU \quad (6.21)$$

Thus, at time $t = T$ the step response has increased to 63% of the total increase which is KU . This is confirmed in Figure 6.2. This suggests a practical way to read off the time-constant from a step response curve.

Qualitatively, we can state the importance of the time-constant as follows: *The less T , the faster the system.*

Does the steady-state response depend on the time-constant? No, because the steady-state response is equal to $y_s = KU$ which is not dependent of T .

Example 6.2 *First order system: Heated liquid tank*

In Example 3.3 on page 42 we developed a mathematical model of heated liquid tank (a thermal system). The model is repeated here:

$$cm\dot{T} = P + cF(T_i - T) + U_h(T_e - T) \quad (6.22)$$

Let's for simplicity assume that the tank is well isolated so that

$$U_h \approx 0 \quad (6.23)$$

We will now calculate the transfer functions from P to T and from T_i to T . Taking the Laplace transform of (6.22) gives

$$cm[sT(s) - T_0] = P(s) + cF[T_i(s) - T(s)] \quad (6.24)$$

Since we are to find the transfer function, we may set the initial value to zero:

$$T_0 = 0 \quad (6.25)$$

From (6.24) we will find

$$T(s) = \underbrace{\frac{K_1}{T_1s + 1}}_{H_1(s)} P(s) + \underbrace{\frac{K_2}{T_1s + 1}}_{H_2(s)} T_i(s) \quad (6.26)$$

The gains and the time-constant of each of the two transfer functions are

$$K_1 = \frac{1}{cF} \quad (6.27)$$

$$K_2 = 1 \quad (6.28)$$

$$T_1 = \frac{m}{F} = \frac{\text{Mass}}{\text{Flow}} \quad (6.29)$$

Comments:

- The time-constant, which represents the “dynamics”, is the same for both transfer functions $H_1(s)$ and $H_2(s)$.
- In many applications the flow F may change. Assume that the flow is decreased. The dynamic properties of the system then change:
 - According to (6.27) the gain from P to T increases, and hence the T becomes more sensitive to P , giving higher value of T for a given change of P .
 - According to (6.29) the time-constant increases, causing a more sluggish response in T to a change in P .

[End of Example 6.2]

6.4 Time-delays

In many systems there is a *time-delay* or dead-time in the signal flow, for example with material transport on a conveyor belt, see Figure 6.3. In this application, the relation between the input variable F_{in} and the output variable F_{out} is

$$F_{out}(t) = F_{in}(t - \tau) \quad (6.30)$$

where τ is the time-delay which is the transportation time on the belt. In other words: The outflow at time t is equal to the inflow τ time units ago.

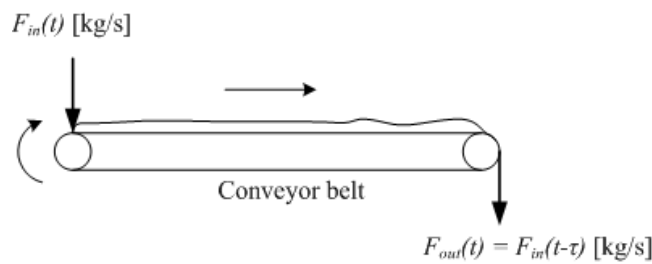


Figure 6.3: Time-delay on a conveyor belt

What is the transfer function of a time-delay? Taking the Laplace transform of (6.30) using (4.16):

$$F_{out}(s) = \underbrace{e^{-\tau s}}_{H(s)} F_{in}(s) \quad (6.31)$$

Thus, the transfer function of a time-delay of τ [time unit] is

$$H(s) = e^{-\tau s} \quad (6.32)$$

Figure 6.4 shows a simulation of a time-delay. The time-delay is $\tau = 1$ sec.

6.5 Higher order systems

Systems having higher order of the denominator polynomial of the transfer function than one, are so-called higher order systems, or more specifically, second order systems, third order systems and so on. A serial connection of first order systems results in a higher order system. (But not all possible higher order systems can be constructed by serial connection of first order systems.) When transfer functions are connected in series, the resulting

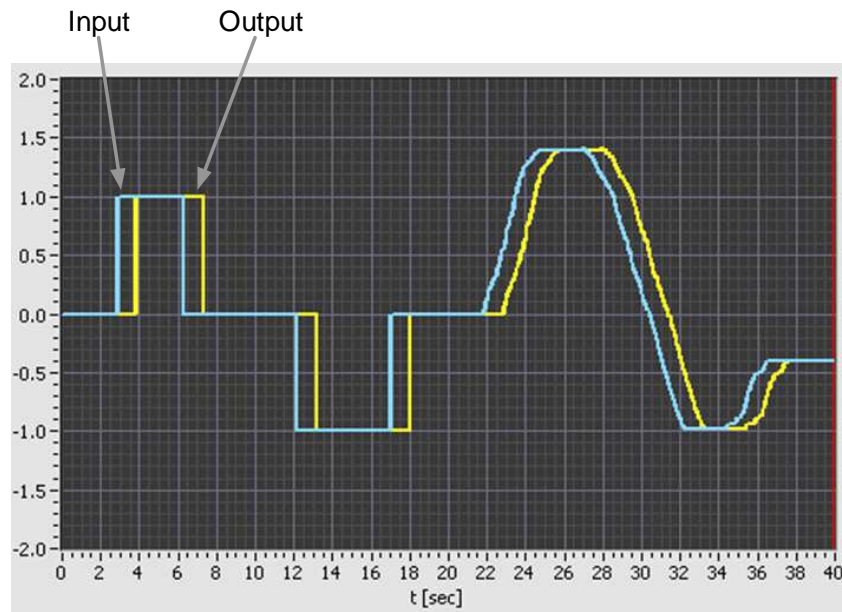


Figure 6.4: Output is equal to input, time delayed 1 sec.

transfer function is the product of the individual transfer functions, cf. Figure 5.2. As an example, Figure 6.5 shows a second order system consisting of “two time-constants” connected in series. The combined transfer function becomes

$$H(s) = \frac{1}{(T_1s + 1)(T_2s + 1)} = \frac{y_2(s)}{u(s)} \quad (6.33)$$

The figure also shows the step responses in the system. It is assumed that $T_1 = 1$, $T_2 = 1$ and $K = 1$. Observe that each first order systems makes the response become more sluggish, as it has a smoothing effect.

Let us define the *response-time* T_r as the *time it takes for a step response to reach 63% of its steady-state value*. For time-constant systems, the response-time is equal to the time-constant:

$$T_r = T \quad (6.34)$$

For higher order systems (order larger than one) it turns out that the response-time can be roughly estimated as the sum of the time-constants of the assumed serial subsystems that make up the higher order system:

$$T_r \approx \sum_i T_i \quad (6.35)$$

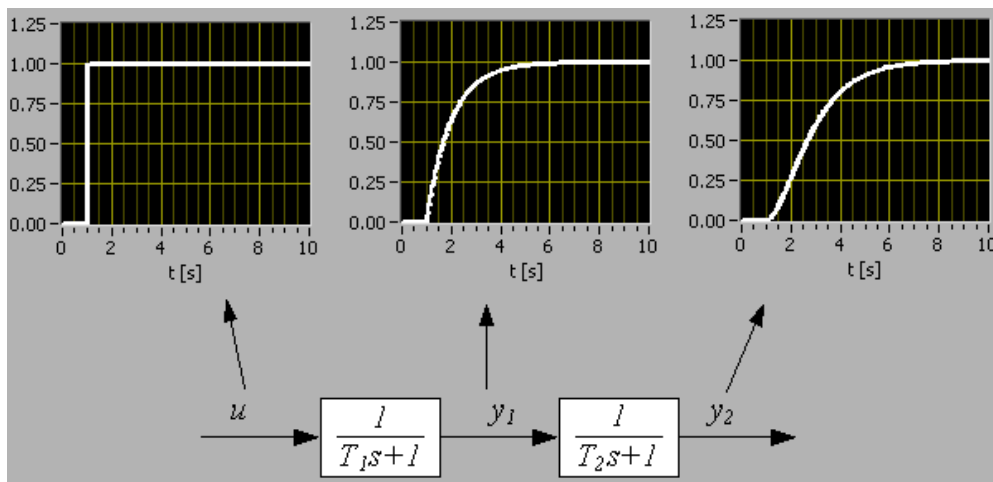


Figure 6.5: Step responses in a second order system

As an example, the response-time of the system shown in Figure 6.5 is

$$T_r \approx 1 + 1 = 2 \text{ s} \quad (6.36)$$

Does the simulation shown in Figure 6.5 confirm this?²

²Yes

Part II

FEEDBACK AND FEEDFORWARD CONTROL

Chapter 7

Feedback control

7.1 Introduction

The principle of feedback control was introduced back in Section 1.1. Now, we will study feedback control in more detail. The sections which now follow continues the description given in Section 1.1.

7.2 Function blocks in the control loop

We will study the function blocks in a typical control loop. If you are to create a simulator of a given control system, these blocks must be included in the simulator (in a simplified simulator you may combine some of the blocks, of course). The level control system for the wood-chip tank, cf. Section 1.2.1, is used as a concrete example.

7.2.1 Automatic and manual mode

Figure 7.1 shows a detailed block diagram of the level control system. The block diagram contains a switch between *automatic mode* and *manual mode* of the controller. On industrial controllers the operator can switch between automatic and manual mode with a button or menu selection on the user interface (front panel, or screen) of the controller.

- **Automatic mode:** The controller calculates the control signal using the control function (typically a PID control function).

- **Manual mode:** A human operator may then adjust the control variable manually on the control equipment, with the control (PID) function being inactive. The control signal can be denoted u_0 – the nominal or manual control value .

Typically, u_0 can not be adjusted in automatic mode. Its value is however included in the control signal in switching from manual to automatic mode to ensure bumpless transfer between the control modes.

The operator will set the controller into manual mode during controller tuning or maintenance.

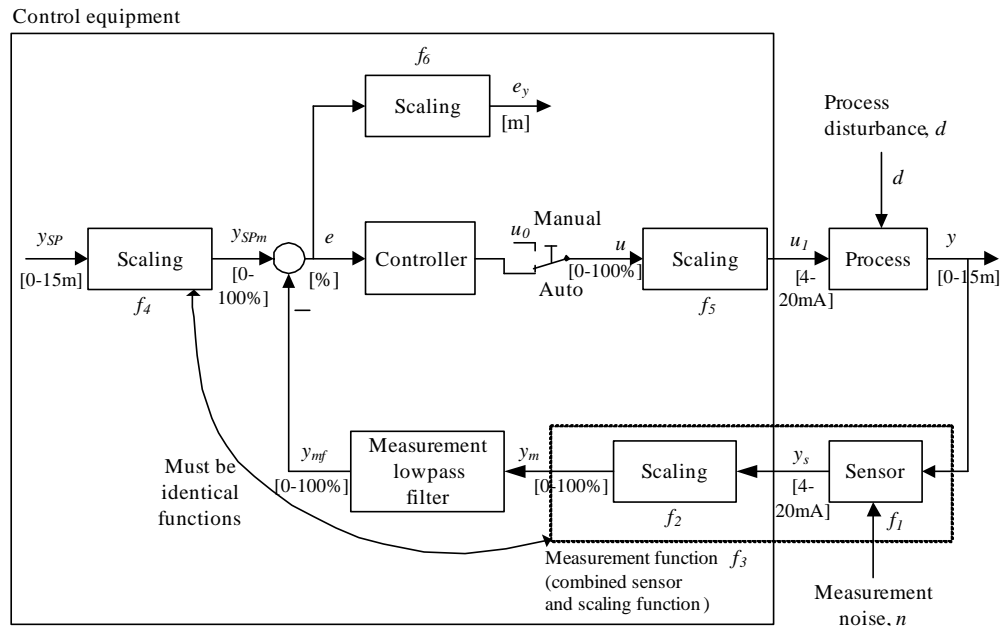


Figure 7.1: Detailed block diagram of the level control system

7.2.2 Measurement lowpass (smoothing) filter

The filter (see down left in Figure 7.1) filters out, or attenuates, the more or less random measurement noise from the measurement signal, so that the measurement signal used by the controller is more smooth, again causing a smoother control signal.

There are many different types of filters that are in use, but the probably most commonly used filter function is the first order filter given by this

transfer function:

$$\frac{y_{mf}(s)}{y_m(s)} = \frac{1}{T_f s + 1} = \frac{1}{\frac{s}{\omega_b} + 1} = \frac{1}{\frac{s}{2\pi f_b} + 1} = H_f(s) \quad (7.1)$$

where T_f is the filter time-constant. ω_b [rad/s] or f_b [Hz] are alternative bandwidths of the filter which are parameters related to the frequency response of the filter. This book does not cover frequency response. In computer-based automation systems the filter is available as a function block containing program code that implements the transfer function. An alternative implementation is with an RC-circuit, cf. Example 3.5.

Let us take a look at how the filtering action of a filter given by (7.1) depends on the time-constant. Figure 7.2 shows the response at the filter

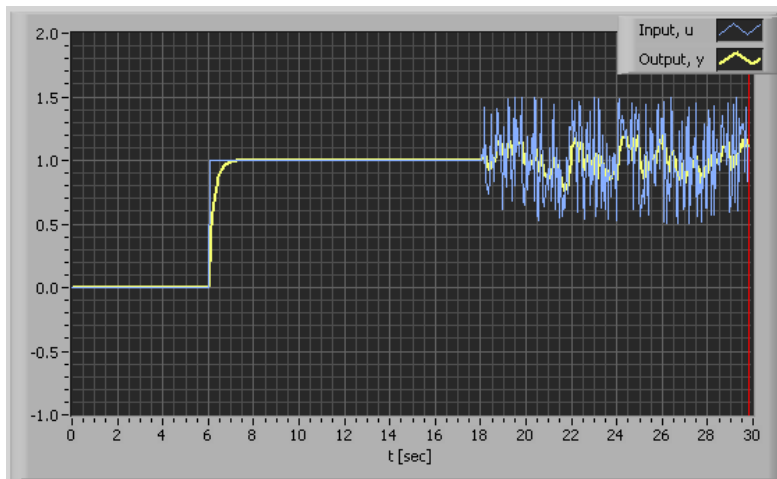


Figure 7.2: Response in filter output due to a input step change at time 6 sec and random noise at the input from time 18 sec for time-constant $T_f = 0.2$ s.

output due to a input step change at time 6 sec and random noise at the input from time 18 sec for time-constant $T_f = 0.2$ s. Figure 7.3 shows the response when the time-constant is ten times larger, i.e. $T_f = 2$ sec. We can here assume that the noise-free part of the input signal (which is changed as a step, which by the way is an unrealistic quick change, but it is proper for illustration) is real process information contained in the measurement, while the random noise is measurement noise.

From these figures we observe the following: *The larger time-constant, the more sluggish filter, and the stronger filtering action, i.e. better noise smoothing.* The benefit of this is the better noise smoothing, and the drawback is that also the real process information is filtered out – more or less.

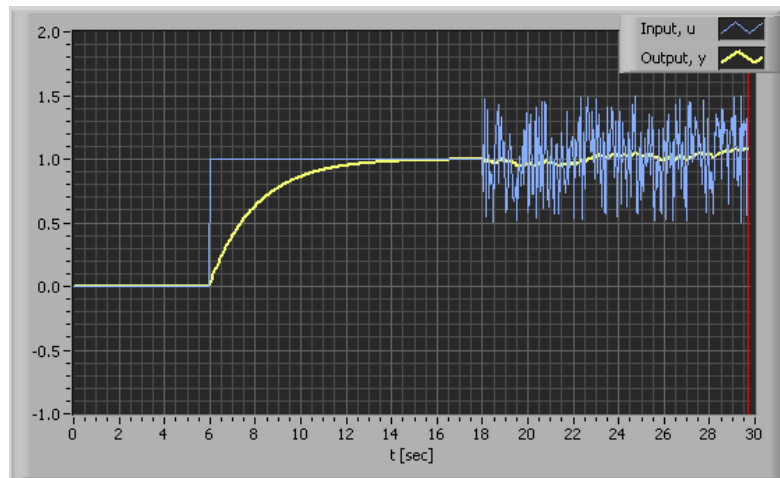


Figure 7.3: Response in filter output due to a input step change at time 6 sec and random noise at the input from time 18 sec for a larger time-constant, $T_f = 2$ s.

It is important to be aware that a measurement filter is a dynamic system in the control loop, and it has impact on the dynamics and stability of the control loop. The controller should therefore be tuned¹ *with* the filter in the loop. If you insert a filter in the loop after the controller has been tuned (without the filter), the stability of the control system will certainly be worse. In the extreme case, instability may occur.

What is a proper value of the filter time-constant, T_f ? It depends on how much noise smoothing you want and the dynamic properties of the process. If you do not have any other requirements, you can initially set it equal to one tenth of the time-constant of the process to avoid the filter adding too much sluggishness to the control loop. It turns out that a time-constant of a few seconds is a typical value in industrial control loops (of e.g. temperature loops).²

7.2.3 Scaling with percentage values

In industrial control systems it is typical that both the process measurement signal and the control signal are scaled in percent. Figure 7.1 shows – as a concrete example – a detailed block diagram of the level

¹Controller tuning is finding proper values of the controller parameters.

²In one of the Fuji PID temperature controllers the preset value of the filter time-constant is 4 sec.

control system with percent values.

The block diagram in Figure 7.1 contains scaling functions for calculating the signals in proper units. In general, these functions are linear, i.e. they are on the following form:

$$z = ax + b \quad (7.2)$$

where a and b are constant parameters, and x is the function input and z the function output. We can say that (7.2) scales or maps x into z .

In (7.2) a (the slope or the gain) can be calculated as

$$a = \frac{z_{\max} - z_{\min}}{x_{\max} - x_{\min}} \quad (7.3)$$

and b (the ordinate axis intersection) can be calculated from (7.2) as

$$b = z_{\min} - ax_{\min} \quad (7.4)$$

or alternatively as

$$b = z_{\max} - ax_{\max} \quad (7.5)$$

The scaling function (7.2) is illustrated in Figure 7.4.

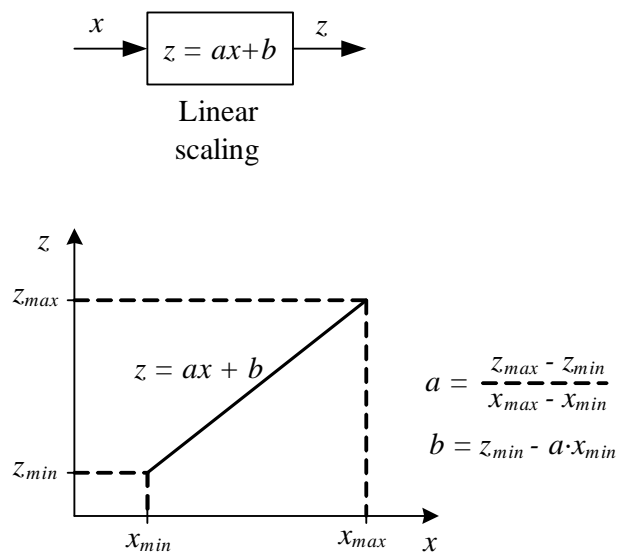


Figure 7.4: Illustration of the scaling function (7.2)

For example, the function f_5 in Figure 7.1 maps the range [0 %, 100 %] into [4 mA, 20 mA] linearly with

$$u_1 = f_5(u) = au + b \quad (7.6)$$

where

$$a = \frac{u_{1\max} - u_{1\min}}{u_{\max} - u_{\min}} = \frac{20 \text{ mA} - 4 \text{ mA}}{100 \% - 0 \%} = 0.16 \text{ mA}/\% \quad (7.7)$$

and

$$b = u_{1\min} - au_{\min} = 4 \text{ mA} - a \cdot 0 \% = 4 \text{ mA} \quad (7.8)$$

Let us look at some of the blocks.

- **Sensor or measurement function, f_1 :** represents the relation between the process output variable y and the physical sensor signal y_s . For industrial applications a number of standard signal ranges of measurement signals are defined. Common standard ranges are 0–5 V, 1–5 V, and 4–20 mA, the latter being the most common.

Wood-chip tank example: The level range [0 m, 15 m] is mapped into a sensor signal range [4 mA, 20 mA] with a linear relation between the ranges. (Of course this scaling is not implemented mathematically, but is inherent in the sensor.) If you need a mathematical function that represents this mapping e.g. in a simulator, you can assume it is on the form (7.2).

- **Measurement signal scaling, f_2 :** It is common that the physical sensor signal is a current between 4 mA and 20 mA, and that the unit of the scaled measurement signal is % (percent). The %-value is then used in the control function. Commercial control equipment contains functions for scaling. During configuration of the controller, the user typically gives information about the minimum and the maximum values of the physical sensor signal, e.g. 4 mA and 20 mA, and the minimum and the maximum values of the scaled measurement signal, e.g. 0 % and 100 %. From this information the controller automatically configures the scaling function, f_2 .

Wood-chip tank example: The sensor signal range [4 mA, 20 mA] is mapped linearly into the measurement signal range [0 %, 100 %]. The mapping is on the form (7.2).

- **Combined measurement scaling and sensor function:** The function f_3 in Figure 7.1 is the combined function of the sensor and the subsequent scaling.

Wood-chip tank example: The level range [0 m, 15 m] is mapped linearly into the measurement signal range [0 %, 100 %]. The mapping is on the form (7.2).

- **Setpoint scaling, f_4 :** The setpoint and the measurement signal must be represented with the same engineering unit, otherwise

subtracting the measurement signal from the setpoint is meaningless. The setpoint scaling must be the same as the combined measurement scaling and sensor function, f_3 .

Wood-chip tank example: The level setpoint range [0 m, 15 m] is mapped linearly into the setpoint range [0 %, 100 %]. The mapping is on the form (7.2).

- **Control variable scaling, f_5 :** If the controller calculates the control variable u in % (which is quite common), then a scaling of the %-value to the physical unit used by the actuator, is necessary.

Wood-chip tank example: The control signal range [0 %, 100 %] is mapped linearly into the range [4 mA, 20 mA], cf. (7.6).

- **Control error scaling, f_6 :** The control error being used as the input to the controller function is commonly in unit %. If you need to map this percentage value into a physical engineering unit, a linear function on the form (7.6) can be used, but the parameter b will be zero.

Wood-chip tank example: The control error range [-100 %, 100 %] is mapped linearly into the range [-15 m, 15 m]. The mapping is on the form (7.2).

- **Unit of the controller gain:** It may be useful, e.g. in controller tuning, to be aware the unit of the (PID) controller gain. The general formula of the controller gain is as follows:

$$\text{Unit of controller gain } (K_p) = \frac{\text{Unit of control signal } (u)}{\text{Unit of measurement signal } (y_{mf})} \quad (7.9)$$

Assume that both the unit of the control signal and of the measurement signal have unit of percent. Then the unit of controller gain (K_p) becomes %/% = 1.

7.2.4 Scaling with physical (engineering) values

The control loop will contain fewer blocks if you scale the process measurement signal and the control signal in physical (engineering) units in stead of in percent. Because the system will appear simpler, you may want to use physical scaling in stead of percent scaling unless there are important reasons for using percent scaling.³

³When I construct control systems from scratch I use physical scaling.

Figure 7.5 shows as an example a detailed block diagram of the level control system with physical scaling. The system is now simpler, compared to the block diagram in Figure 7.1.

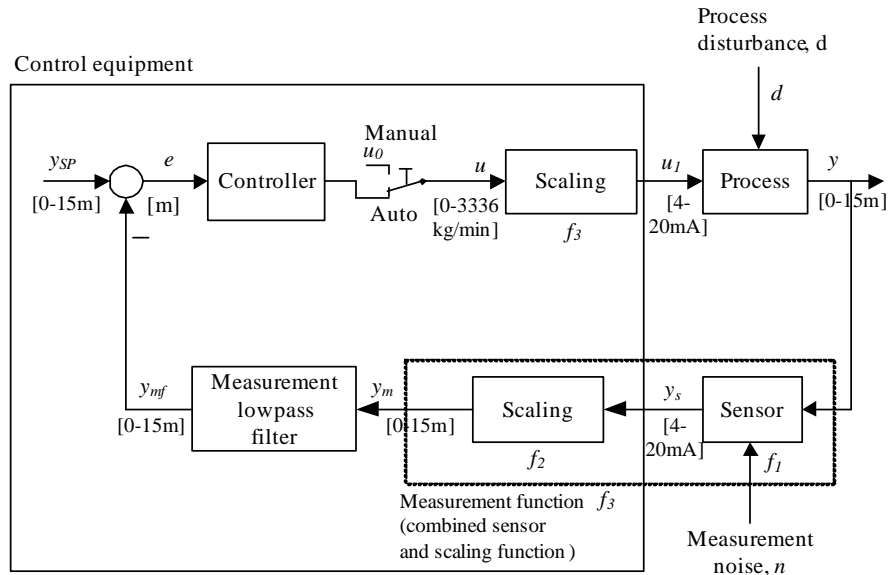


Figure 7.5: Detailed block diagram of the level control system with physical scaling

Comments:

1. The unit of the control signal is kg/min, but this does not mean that the mass flow out of the feed screw (the actuator) is momentarily equal to the control signal value. In general you must expect some sluggishness in the actuator. This sluggishness may be modeled as a time-constant transfer function.
2. The unit of the controller gain K_p is no longer unity, as in the block diagram in Figure 7.1. In stead, the controller gain is

$$\text{Unit of controller gain } (K_p) = \frac{\text{Unit of control signal}}{\text{Unit of measurement signal}} \quad (7.10)$$

In Figure 7.5 the unit of the controller gain is (kg/min)/m.

Of course, the numerical value of the controller gain depends on the scaling. Thus, the controller gain in Figure 7.1 will have a different value from the value of the controller gain shown in Figure 7.5.

7.3 The PID controller

7.3.1 The ideal PID controller function

Figure 7.1 shows where the control function – usually denoted simply “the controller” – is placed in a control loop. The *PID controller* is the standard controller function in industry. The ideal PID controller is as follows (some important modifications are described below):

$$u = u_0 + \underbrace{K_p e}_{u_p} + \underbrace{\frac{K_p}{T_i} \int_0^t e \, d\tau}_{u_i} + \underbrace{K_p T_d \frac{de}{dt}}_{u_d} \quad (7.11)$$

The controller parameters are as follows:

- K_p is the *proportional gain*. (Other symbols of this gain are K_c (c for controller) and K .)
- T_i [s] or [min] is the *integral time*. In some controllers the value of $1/T_i$ is used in stead of the value of T_i . The unit of $1/T_i$ is *repeats per minute*. For example, 5 repeats per minute means that T_i is equal to $1/5 = 0.2$ min.⁴
- T_d [s] or [min] is the *derivative time*.
- u_0 is the nominal value of the control variable. It is the control signal available for adjusted by the operator while the controller is in manual mode. While the controller is in automatic mode, u_0 can usually not be adjusted.

In (7.11) u_p is the P-term. u_i is the I-term. u_d is the D-term.

In some commercial controllers the *proportional band* P_B , is used in stead of the proportional gain, K_p . The proportional band is given by

$$P_B = \frac{100\%}{K_p} \quad (7.12)$$

where K_p is the gain, which here is assumed to be dimensionless. (It will be dimensionless if the control error e and the control variable u have the

⁴The background of the term repeats per minute is as follows: Assume that the control error e is constant, say E . The P-term has value $u_p = K_p E$. During a time interval of 1 minute the I-term equals $\frac{K_p}{T_i} \int_0^1 E \, d\tau = K_p E \cdot 1[\text{min}]/T_i = u_p \cdot 1/T_i$. Thus, the I-term has repeated the P-term $1/T_i$ times.

same unit, typically percent). It is typical that P_B has a value in the range of $10\% \leq P_B \leq 500\%$, which corresponds to K_p being in the range of $0.2 \leq K_p \leq 10$. It is important to note that P_B is inversely proportional to K_p . Thus, a small P_B corresponds to a large K_p , and vice versa.⁵

The *P controller* and the *PI controller* are special cases of the PID controller:

- A P controller is achieved by setting $T_i = \infty$ (or to a very large value) and $T_d = 0$.⁶
- A PI controller is achieved by setting $T_d = 0$.

7.3.2 How the PID controller works

The integral term

The *integral term of the PID controller is essential*. The I-term is

$$u_i = \frac{K_p}{T_i} \int_0^t e \, d\tau \quad (7.13)$$

as calculated as the time integral of the control error e from an initial time, say $t = 0$, to the present point of time, thus the integral is being calculated continuously. Let us think about the level control of the wood-chip tank. Review Figure 1.3. Assume that e initially is greater than zero (the level is then less than the level setpoint). As long as e is positive, u_i and therefore the total control variable u will get a steadily increasing value, since the time integral of a positive value increases with time. The increasing u gives an increasing wood-chip inflow. Consequently, the chip level in the tank increases. Due to this level increase the control error eventually becomes less positive. The increase of the integral term (the inflow) continues until the error has become zero. The conclusion is that the *integral term ensures zero steady-state control error*. This is illustrated in Figure 7.6.

⁵What does *proportional band* actually mean? One explanation is that P_B is the change of the control error interval Δe (or the size of the measurement signal interval) which gives a change of the control signal interval Δu equal to 100% (i.e. a full range change of u): From the P-term $u = K_p e$ we see that $\Delta e = \Delta u / K_p = 100\% / K_p = P_B$.

⁶In some commercial controllers you can set T_i to 0 (zero), which is a code expressing that the integral term is de-activated.

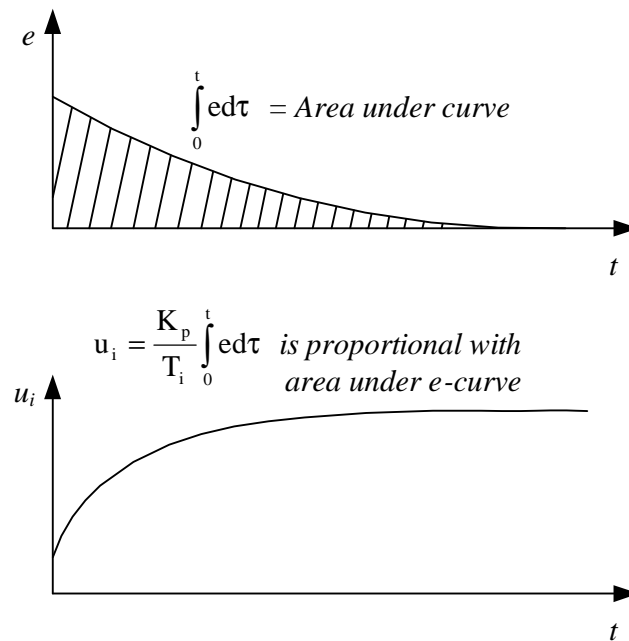


Figure 7.6: The behaviour of the integral term of the PID controller

The proportional term

What is the contribution of the P-term in the control action? The P-term is

$$u_p = K_p e \quad (7.14)$$

The P-term u_p reacts faster to a change in the error than the I-term since the P-term is proportional to the error and reacts instantly, while the I-term reacts little in the beginning and increases, until the error is brought to zero. Therefore, the P-term contributes to faster control compared to not having any P-term. The P-term goes to zero as the control error goes to zero, so the P-term can not by itself ensure zero steady-state control error (it is the I-term alone that ensures zero steady-state error).

The derivative term

What is the contribution of the D-term in the control action? The D-term is

$$u_d = K_p T_d \frac{de}{dt} \quad (7.15)$$

Firstly, assume that the control error e is increasing, so that the time derivative de/dt is positive. In this situation the D-term contributes with a positive value to the total control signal u (it is assumed that K_p and T_d are positive). This will in general give *faster control*. Secondly, assume that the control error e is decreasing so that de/dt is negative. In this situation the derivative contributes with a negative value, i.e. a reduction, to the total control signal u . This will in general give a *breaking or stabilizing control action*. So, the D-term contributes with faster control, and it can also stabilize the control system. So far, so good. But there is one serious practical problem with the D-term: It amplifies the random measurement noise, causing large variations in the control signal. These variations will be reduced with a lowpass filter acting on the process measurement, cf. Section 7.2.2.

From the above it can be concluded that the controller should have an I-term, i.e. the controller should be either a PID controller or a PI controller, to get zero steady-state control error. The D-term should be omitted if the control signal is too noisy even with a measurement lowpass filter. However there are processes where the D-term is essential for obtaining a stable control system (one example is a double-integrator⁷).

Example 7.1 *PID controller in the wood-chip level control system*

Let us see how the P-term, I-term and the D-term works in a simulation of the level control system of the wood-chip tank described in Section 1.2.1. Figure 7.7 shows the time-responses due to a step in the outflow v (the disturbance) from the initial value of 1500 kg/min to 1800 kg/min. The controller parameters are tuned to proper stability. The PID settings are $K_p = 1.9$, $T_i = 540$ s, and $T_d = 135$ s. We can observe the following:

- The I-term is relatively sluggish. Its value changes as long as the control error is different from zero. u_i goes to a (new) constant value after the step in v , large enough to compensate for the increased outflow.
- The P-term u_p is quicker than the I-term, but more sluggish than the D-term. In steady-state, while the control error is zero (thanks to the I-term), the mean value of the P-term is zero.

⁷An example of such a control system is position control of a mechanical system (motor, ship, robot, space-vehicle) where the control signal generates a force or torque, and the damping of the system or process is small.

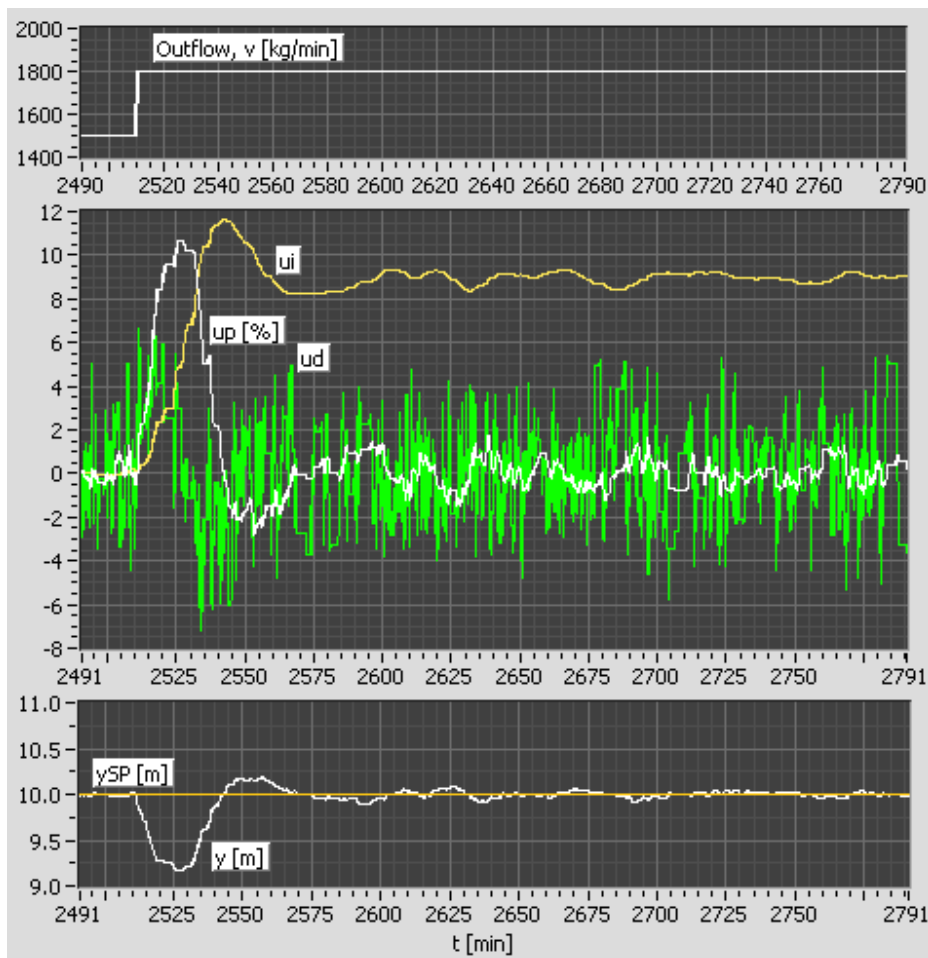


Figure 7.7: Behaviour of the various terms of the PID-controller terms after a step in the outflow v (disturbance)

- The D-term u_d is very noisy. The noisy behaviour is because the time-derivative of the quickly changing noisy control error has large amplitude. In steady-state, while the control error is zero (thanks to the I-term), the mean value of the D-term is zero.

[End of Example 7.1]

7.3.3 Positive or negative controller gain? Or: Reverse or direct action?

On industrial controllers you can choose whether the controller gain of the PID controller shall have positive or negative sign, or in other words: whether the controller shall have reverse or direct action mode. *It is crucial that you set the sign of the controller gain correctly. If set incorrectly, the control system becomes unstable!*

Remember:

- *Positive controller gain* is the same as *Reverse action mode*.
- *Negative controller gain* is the same as *Direct action mode*.

The rules of selecting the correct controller action mode are as follows. Assume that the process measurement is equal to the setpoint initially, and that for any reason the process measurement increases to become *larger* than the setpoint (this change may for example have been caused by a change of the process disturbance). The sign of the change of the control signal needed to bring the process measurement back to the setpoint determines between Reverse action mode and Direct action mode:

- **Reverse action mode:** If the controller must *decrease* the control signal to bring the *increased* process measurement back to the setpoint, the controller shall have *Reverse action mode*.
- **Direct action mode:** If the controller must *increase* the control signal to bring the *increased* process measurement back to the setpoint, the controller shall have *Direct action mode*.

Note that for industrial controllers it is common that the user always enters a positive value for the controller gain parameter (this is actually the absolute value of the controller gain.) To set the sign of the resulting gain the user must in addition select either Reverse action (positive resulting controller gain) or Direct action (negative resulting controller gain) in a dedicated parameter. This is illustrated in Figure 7.8.

Example 7.2 *Reverse or Direct action in the level controller?*

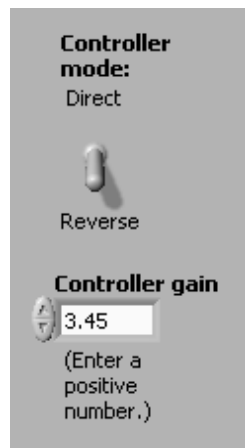


Figure 7.8: Setting the controller mode and the (absolute value of) the controller gain

Figure 7.9 shows a level control system for a liquid tank where the control variable *controls the outflow* of the tank, and it is assumed that increasing control signal gives increasing flow. Shall the level controller have Reverse

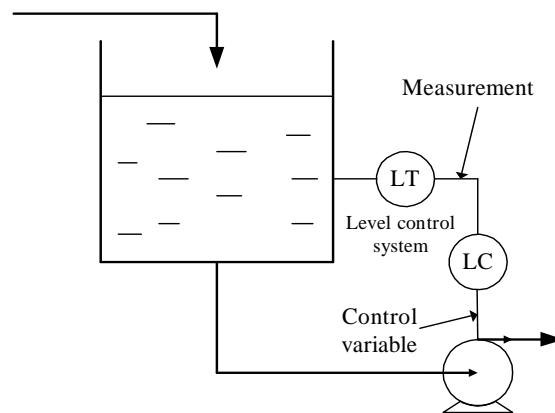


Figure 7.9: Example 7.2: Tank with level manipulated by a pump in the outlet.

or Direct action mode? To answer the question, assume that the level is at the setpoint, and that the level then (for some reason) increases. How should the control signal acting on the pump be adjusted to get the level back to the setpoint? Obviously, the control signal must be increased. Hence, the controller must have *Direct action*.

What if the pump was in the inlet instead? Then the controller must have

Reverse action.

[End of Example 7.2]

7.4 Practical modifications of the ideal PID controller

This Section describes briefly several modifications of the ideal PID controller, (7.11). Some of these modifications are necessary to have a controller that actually works.

7.4.1 Lowpass filter in the D-term

As we have seen above, the D-term amplifies random measurement noise, causing variations in the control signal out from the controller. These variations may have too large amplitude, even if there is a measurement lowpass filter in the feedback path, cf. Figure 1.2. To reduce the problem of the noisy D-term, a *lowpass filter* can be inserted in series with the time-differentiation, so that the control error is lowpass filtered before it is differentiated. This is a common feature of commercial controllers, but there are controllers that do not have it. Let us use the symbol e_f for the filtered error. The modified PID controller is then

$$u = u_0 + \underbrace{K_p e}_{u_p} + \underbrace{\frac{K_p}{T_i} \int_0^t e \, d\tau}_{u_i} + \underbrace{K_p T_d \frac{de_f}{dt}}_{u_d} \quad (7.16)$$

The filter is typically a first order lowpass filter. The relation between e_f and e can be expressed with the following transfer function:⁸

$$e_f(s) = \frac{1}{T_f s + 1} e(s) \quad (7.17)$$

where T_f is the filter time-constant which usually is expressed as a fraction of the derivative time T_d :

$$T_f = a T_d \quad (7.18)$$

a is a constant which typically is chosen between 0.05 and 0.2. If no special requirements exist, we can set $a = 0.1$.

⁸Although it is not mathematically correct, it is convenient to use the same symbol for the time function and the Laplace transform of the function.

7.4.2 Reducing P-kick and D-kick caused by setpoint changes

Abrupt changes of the setpoint y_{SP} , for example step changes, creates an abrupt change of the control error, which in turn may cause unfortunate kicks in the control variable. The problem is related to the P-term and the D-term of the controller function (7.16) which are terms that react quick to changes in the control error. These kicks are denoted proportional kick or *P-kick* and derivative kick or *D-kick*, respectively. Such kicks may cause mechanical actuators to move abruptly, resulting in excessive wear.

Why does these kicks come? Let's first look at the P-term:

$$u_p = K_p (w_p y_{SP} - y) \quad (7.19)$$

In the standard PID controller the weight w_p is 1. Assume that y_{SP} changes as a step. Then there is also a step in u_p , causing a proportional kick, a P-kick, in the total control signal (in which u_p is one additive term).

Let's look at the D-term:

$$u_d = K_p T_d \frac{d(w_d y_{SP} - y)}{dt} \quad (7.20)$$

In the standard PID controller the weight w_d is 1. Assume that y_{SP} changes as a step. Since the time-derivative of a step is an impulse, the step causes an impulse in u_d , causing a derivative kick (impulse), a D-kick, in the total control signal (in which u_d is one additive term).

How to solve the problems about P-kick and D-kick? Perhaps the best solution to this problem is to allow only *smooth setpoint changes*.

Commercial controllers have functions to implement rampwise changes between two setpoint values. This is denoted ramping the setpoint, see Figure 7.10.

Another solution is to *reduce the setpoint weights* w_p and w_d . Regarding w_p , it is not so common to reduce it, but if it is, $w_p = 0.3$ is suggested [9]. Regarding w_d , it is actually quite common to set $w_d = 0$, causing the setpoint to be removed completely from the derivative term. In many commercial controllers $w_d = 0$ is a fixed factory setting.

Example 7.3 *Avoiding controller kicks during setpoint step change*

Figure 7.11 shows responses in simulated PID control system with and without reduction of the setpoint weight in the P-term and the D-term,

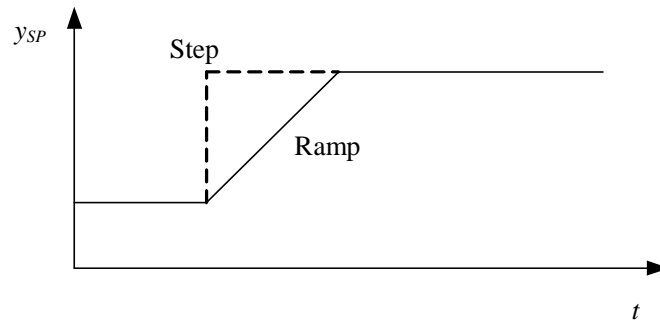


Figure 7.10: The change of the setpoint from one value to another may follow a ramp in stead of a step to avoid kicks in the control signal.

and with setpoint ramping. The reduced weights are $w_p = 0.3$ and $w_d = 0$. The process model is a serial combination of two time-constants of 1.0 s and 2.0 s and a time-delay of 0.5 s. The steady-state process gain is 1. The PID parameters are $K_p = 3.6$, $T_i = 2.0$ s, $T_d = 0.5$ s. The simulations demonstrates that the control signal is smoother with reduced weights and with setpoint ramping.

[End of Example 7.3]

7.4.3 Integrator anti wind-up

All actuators have saturation limits, i.e. a maximum limit and a minimum limit. For example, a power amplifier (for a heater or a motor) can not deliver an infinitely amount of power, and a valve can not have an infinitely large opening and can not be more closed than closed(!). Under normal process operation the control variable should not reach the saturation limits. But everything is not normal all the time. Assume that in a time interval a large process disturbance acts on the process so that the process output variable is reduced. The control error then becomes large and positive, and the control variable will increase steadily during this time interval (because of the integral term of the PID controller) until the control signal limits at its maximum value, u_{\max} . Assume that the disturbance is so large that u_{\max} is not large enough to compensate for the large disturbance. Because of this, the control error remains large, and the integral of the control error continues to increase, which means that the calculated integral term u_i continues to increase. This is *integrator wind-up*.

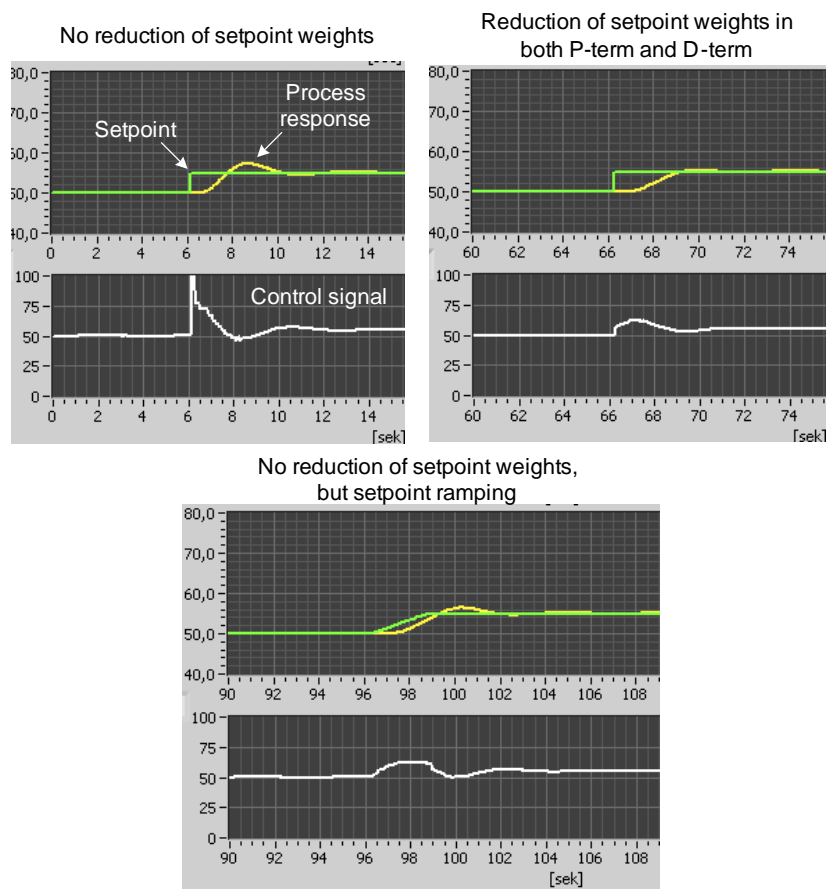


Figure 7.11: Example 7.3: Responses in PID control system with and without reduction of setpoint weight and with setpoint ramping

Assume that the process disturbance after a while goes back to its normal value. This causes the process output variable to increase since the disturbance is reduced (the load is removed), and the error will now change sign (it becomes negative). Consequently the integral term starts to integrate downwards (its value is continuously being reduced), so that the calculated u_i is reduced, which is ok, since the smaller disturbance requires a smaller control signal. However, the problem is that it may take *a long time* until the large value of the calculated u_i is reduced (via the down-integration) to a normal (reasonable) value. During this long time the control variable is larger than what is required to compensate for the disturbance, causing the process output variable to be larger than the setpoint during this time.

A practical PID controller must be able to cope with the possibility of

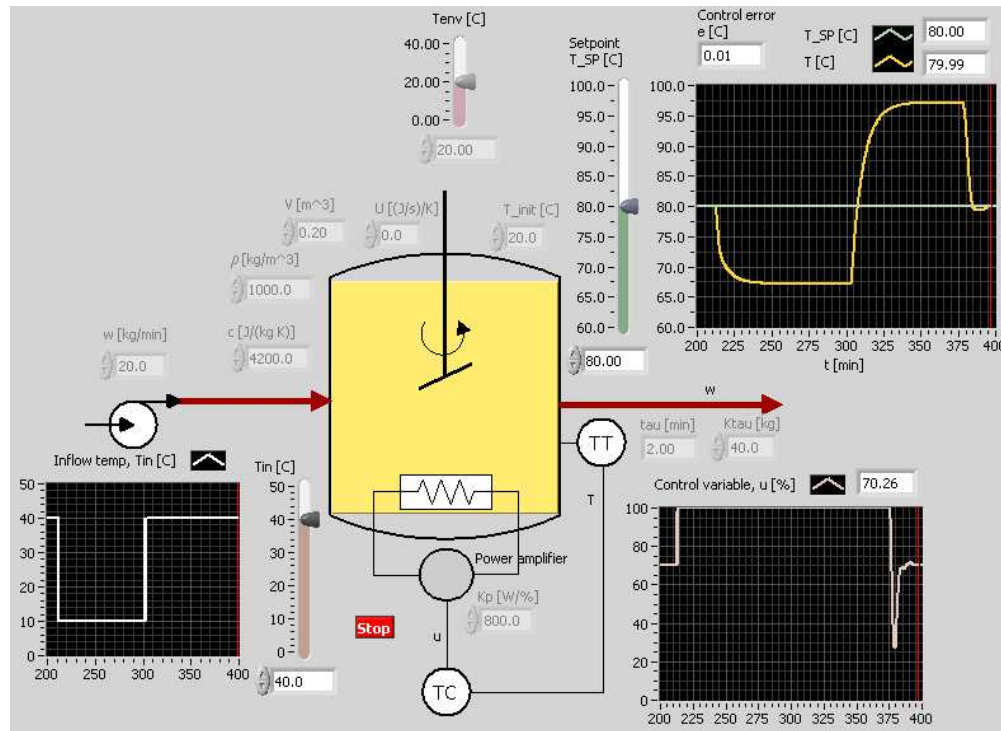


Figure 7.12: Example 7.4: Temperature control *without* anti wind-up

integrator wind-up, that is, it must have some *integral anti wind-up* mechanism. The principle of an anti wind-up mechanism is simple: Since the problem is that the integral term increases continuously during actuator saturation, the solution is to *halt the integration of the control error when the control signal reaches either its maximum or its minimum limit*.

Fortunately, you can assume that anti wind-up is implemented in commercial controllers.

Example 7.4 *Integral anti wind-up in a temperature control system*

Figure 7.12 shows the front panel of a simulator for a temperature control system for a liquid tank with continuously mass flow. The disturbance is here the inlet temperature T_{in} , which is is changed as a step from 40 °C to 10 °C at approx. 210 min and back to 40 °C at approx. 300 min. The temperature setpoint T_{SP} is 70 °C (constant). The parameters of the PID controller are $K_p = 6.7$, $T_i = 252 \text{ s} = 42 \text{ min}$ and $T_d = 63 \text{ s} = 10.5 \text{ min}$

(found using Ziegler-Nichols' closed loop method). The maximum value of the control variable is 100 % and the minimum value is 0 %. When T_{in} is reduced to 10 °C, the actuator (heating element) goes into saturation (100 %), trying to compensate for the (cold) disturbance. It can be shown that the control variable u should have a value of 122.5 % (which corresponds to more heat power than what is available) to be able to compensate for $T_{in} = 10$ °C.

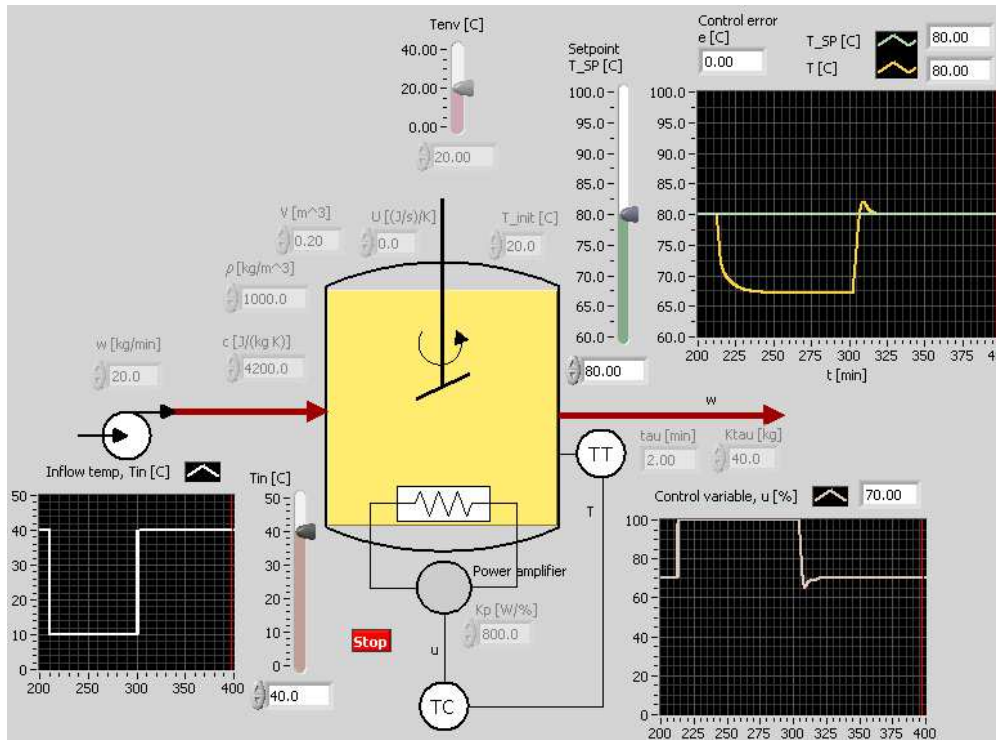


Figure 7.13: Example 7.4: Temperature control *with* anti wind-up

Figure 7.12 shows the simulated responses in the control system *without* using integrator anti wind-up, and Figure 7.13 shows the responses *with* integrator anti wind-up. In the case of no anti wind-up, it was observed (but this is not indicated in Figure 7.12) that the integral term u_i in the PID controller reached a maximum value of approximately 2200 %! The simulations clearly show that it is beneficial to use integrator anti wind-up (as the temperature returns much quicker to the setpoint after the disturbance has changed back to its normal value).

[End of Example 7.4]

7.4.4 Bumpless transfer between manual/auto mode

Figure 7.1 shows a block diagram of a control loop. It is important that the control signal does not jump (too much) when the controller is switched from automatic to manual mode, or from manual to automatic mode. In other words, the transfer between the automatic and manual modes should be *bumpless*. Bumpless transfer is implemented in commercial controllers.

7.5 Control loop stability

It is important to be aware that there may be stability problems in a control loop. It is a basic requirement to a control loop that it is stable. Simply stated this means that the response in any signal in control loop converges towards a finite value after a limited change (with respect to the amplitude) of the setpoint or the disturbance or any other input signal to the loop. For example, the setpoint step response shown in Figure 1.3 (up, right in the figure) indicates that the chip level control system is stable.

There is always a possibility that a feedback control system which is originally stable, *may become unstable* due to parameter changes in the loop. Instability implies that signals in the loop starts to increase in amplitude until some saturation limit is reached (for example, a valve have a limited opening).

As a start of the discussion of how a feedback control system can become unstable, let's repeat the block diagram of a feedback control system from the beginning of this book, see Figure 7.14. Instability in a feedback system can be explained in two ways:

- **Too large gain in the loop:** The loop gain is the product of the gains in each of the subsystems (controller, process, sensor) in the loop. If the present signal one place in the loop is amplified too much through the loop, it comes back amplified. Then, this signal is again amplified through the loop, and eventually the amplitude just increases. In other words, the control loop is unstable.
- **Too large time delay in the loop.** The effect or response due to the controller action is fed back to the controller with too much time delay so that the information in the response is out-of-date and does not reflect the state of the process good enough. Based on this poor information, the controller may generate too much or too little control action. This repeats through the loop, causing larger and

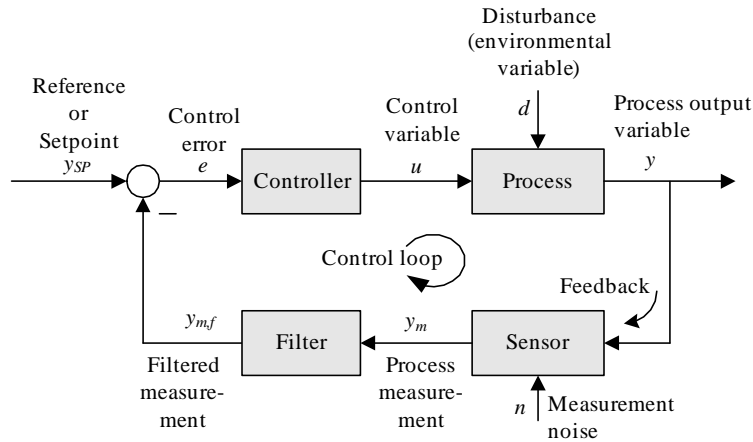


Figure 7.14: Block diagram of a feedback control system

larger deviation from the setpoint. In other words, the control loop is unstable.

[2] describes ways to analyze control loop stability theoretically (from a mathematical model of the control system).

Example 7.5 *Instability in the wood-chip tank level control system*

Initially the controller is a PI controller with proper settings:

$$K_p = 1.35; T_i = 917 \text{ s}; T_d = 0 \text{ s} \quad (7.21)$$

Too large gain: Figure 7.15 shows the response in the level when the controller gain K_p is increased from 1.35 to 5.00 at time 1000 s. The control system becomes unstable. (The amplitude of the oscillations are limited due to the limits of the control variable. The maximum value is 100%, and the minimum value is 0% are reached during the simulations.)

Too large time delay: Figure 7.16 shows the response in the level when the time delay of the conveyor belt in is increased from the nominal value of 250 s to 600 s at time time 11000 s. (The controller gain has its nominal value, 1.35.) The control system becomes unstable.

[End of Example 7.5]

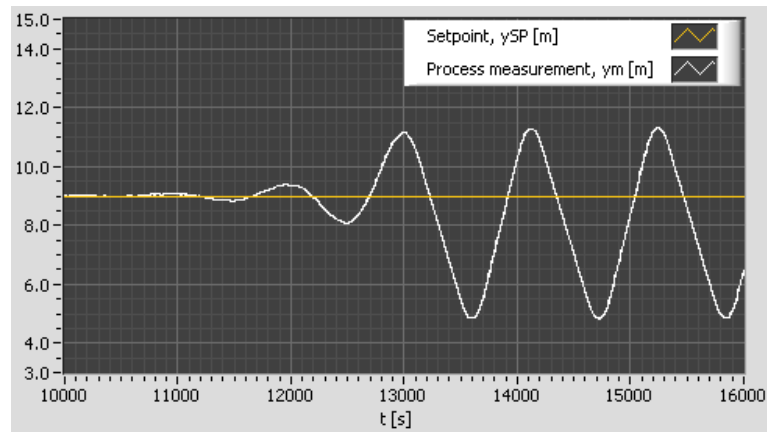


Figure 7.15: The response in the level when the controller gain K_p is increased from 1.35 to 5.00 at time 1000 s. The control system has become unstable.

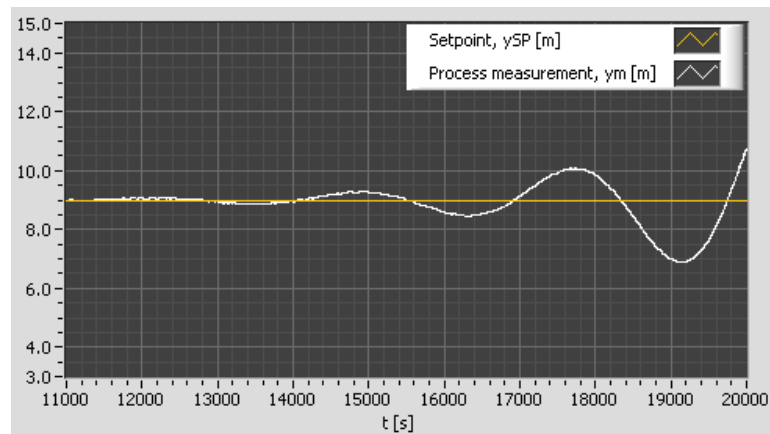


Figure 7.16: The response in the level when the time delay of the conveyor belt in is increased from the nominal value of 250 s to 600 s at time 11000 s. The control system has become unstable.

Chapter 8

Feedforward control

8.1 Introduction

We know from Chapters 1.1 and 7 that feedback control – or error-driven control – can bring the process output variable to or close to the setpoint. Feedback control is in most cases a sufficiently good control method. But improvements can be made, if required. A problem with feedback is that there is no adjustment of the control variable before the control error is different from zero, since the control variable is adjusted as a function of the control error. This problem does not exist in *feedforward control*, which may be used as the only control method, or, more typically, as a supplement to feedback control.

In feedforward control there is a *coupling from the setpoint and/or from the disturbance directly to the control variable*, that is, a coupling from an input signal to the control variable. The control variable adjustment is not error-based. In stead it is based on knowledge about the process in the form of a mathematical model of the process and knowledge about or measurements of the process disturbances.

Perfect or ideal feedforward control gives zero control error for *all types of signals* (e.g. a sinusoid and a ramp) in the setpoint and in the disturbance. This sounds good, but feedforward control may be *difficult to implement* since it assumes or is based on a mathematical process model and that all variables of the model at any instant of time must have known values through measurements or in some other way. These requirements are never completely satisfied, and therefore in practice the control error becomes different from zero. We can however assume that the control error becomes

smaller with imperfect feedforward control than without feedforward control.

If feedforward control is used, it is typically used together with feedback control. Figure 8.1 shows the structure of a control system with both feedforward and feedback control. The purpose of feedback control is to

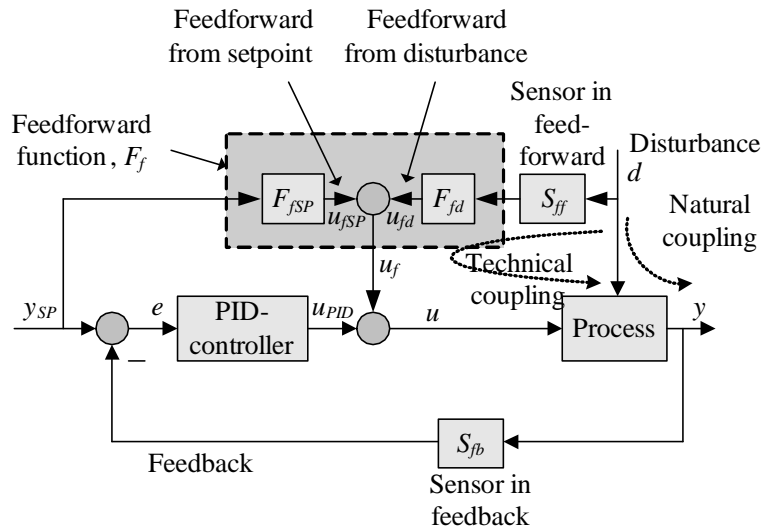


Figure 8.1: Control system with both feedforward and feedback control

reduce the control error due to the inevitable imperfect feedforward control. Practical feedforward control can never be perfect, because of model errors and imprecise measurements.

In Figure 8.1, the feedforward from disturbance can be interpreted as a technical coupling from the disturbance to the process output variable which is supposed to counteract the natural coupling so that the net effect that the disturbance has on the process output y is zero, which is what we want.

The feedforward function F_f , which usually consists of a sum of partial functions F_{fSP} and F_{fd} as shown in Figure 8.1, that is

$$F_f = F_{fSP} + F_{fd} \quad (8.1)$$

can be developed in several ways:

- From a differential equations process model
- From a transfer functions process model

- From experimental data. This method is model-free, and should be regarded as an approximative method, which still may give substantial improvement of control system performance.

Among these, the first and the third method are described in the following sections as they are assumed to be the most useful in practical cases.

Using feedforward together with feedback does not influence the stability of the feedback loop because the feedforward does not introduce new dynamics in the loop.

8.2 Designing feedforward control from differential equation models

The feedforward function F_f can be derived quite easily from a differential equations model of the process to be controlled. The design method is to *solve for the control output variable in the process model with the setpoint substituted for the process output variable* (which is the variable to be controlled). The model may be linear or non-linear.

One practical issue: Typically, the feedforward from setpoint, F_{fSP} , will contain time-derivatives of the setpoint. You should however not implement a pure time-differentiation because the time-derivative is very sensitive to noise and abrupt changes of the signal to be differentiated. Instead, you should implement a smoothed derivative by letting the signal pass through a lowpass filter before the differentiation. This is demonstrated in the following example.

Example 8.1 *Feedforward control of a liquid tank*

Figure 8.2 shows a liquid tank with inflow and outflow. The level h is to be controlled using feedback with PID controller in combination with feedforward control. (The responses are explained later in this example.) From a mass balance of the liquid in the tank we get the following process model:

$$\rho A \dot{h}(t) = F_{in}(t) - F_{out}(t) \quad (8.2)$$

where h [m] is liquid level, F_{in} [kg/s] is mass inflow, F_{out} [kg/s] is mass outflow, A [m²] is cross sectional area of the tank, ρ [kg/m³] is the liquid density. F_{in} is assumed to be equal in value to the applied control signal u .

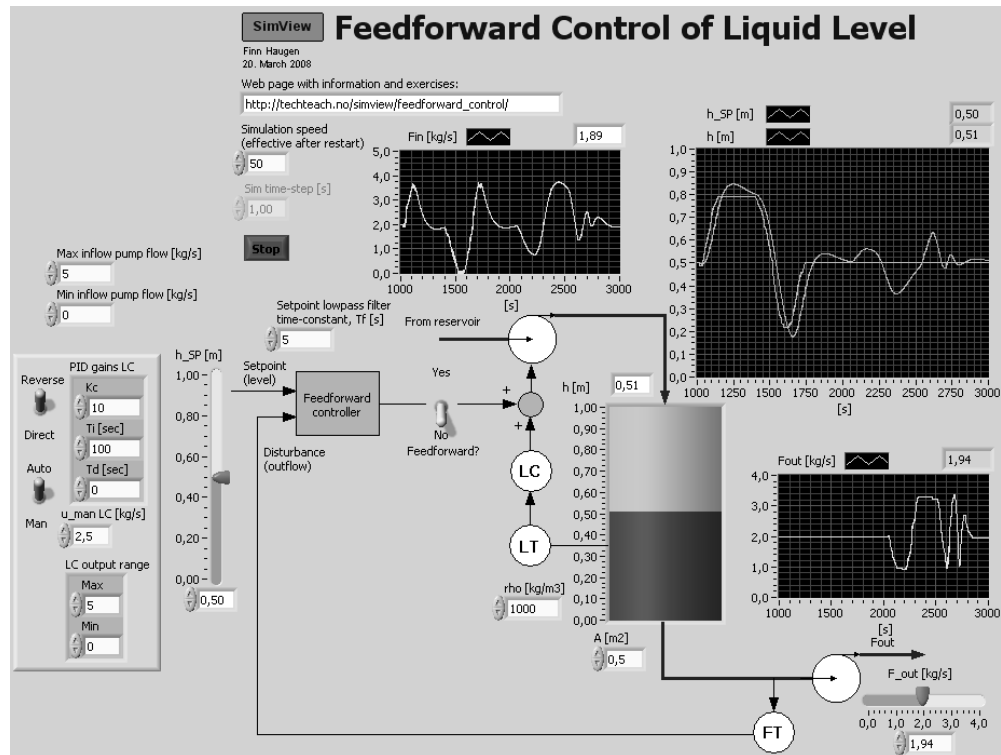


Figure 8.2: Example 8.1: Liquid tank where the level h is controlled with feedback control. Here, feedforward control is *not* applied.

By using $F_{in} = u$ the model becomes:

$$\rho A \dot{h}(t) = u(t) - F_{out}(t) \quad (8.3)$$

Now, let us derive the feedforward control function from the process model (8.3). First, we substitute the level h by its setpoint h_{SP} :

$$\rho A \dot{h}_{SP}(t) = u(t) - F_{out}(t) \quad (8.4)$$

Then we solve (8.4) for the control variable u to get the feedforward control variable u_f :

$$u_f(t) = \underbrace{\rho A \dot{h}_{SP}(t)}_{u_{fSP}} + \underbrace{F_{out}(t)}_{u_{fd}} \quad (8.5)$$

which is the ideal feedforward controller. u_{fSP} represents feedforward from setpoint and u_{fd} represents feedforward from disturbance. We see that calculation of feedforward control signal u_f requires measurement or knowledge of the following three quantities: A , F_{out} , and \dot{h}_{SP} . (Figure 8.2 indicates that flow F_{out} is measured.)

Does the feedforward controller (8.5) make sense?

- The term

$$u_{fd}(t) = F_{out}(t) \quad (8.6)$$

tells that the inflow should be equal to the outflow at any instant of time to cancel the impact of the outflow in the level. Makes sense?¹

- The term

$$u_{fSP}(t) = \rho A \dot{h}_{SP}(t) \quad (8.7)$$

tells that if the setpoint is changing, the inflow should be increased, or decreased – depending on the sign of \dot{h}_{SP} , equal to the specified rate of change of the mass in the tank, which is $\rho A \dot{h}_{SP}$.

In (8.5) the time-derivate \dot{h}_{SP} should be implemented as a *limited time-derivative*: $\dot{h}_{SP_{filt}}$ where $h_{SP_{filt}}$ is lowpass filtered setpoint. So, it is the lowpass filtered setpoint that is time-differentiated. The reason for the lowpass filtering is that the time-derivative is very sensitive to noisy variations of the value to be differentiated. Any non-smooth variation of the setpoint is noise in this respect.

Since the time-derivative is of first order, the filter can be a first order lowpass filter in the form of a "time-constant" filter:

$$h_{SP_{filt}}(s) = \frac{1}{T_f s + 1} h_{SP}(s) \quad (8.8)$$

where T_f is the filter time-constant which can be tuned by trial-and-error.

Two cases are simulated: Level control *without* and *with feedforward control*. In both cases there is feedback control with PI controller with parameters $K_p = 10$ and $T_i = 100$ s. In the first part of the simulation which is the first 1000 seconds, the level setpoint h_{SP} is varied while the outflow (disturbance) is constant, while in the last part of the simulation which is the last 1000 seconds, the level setpoint is kept constant while the outflow F_{out} is varied.

- **Without feedforward control** (only feedback control): Figure 8.2 shows the responses in the level h as the setpoint is varied and as the disturbance (outflow) is varies. The level deviates clearly from the setpoint in both situations.

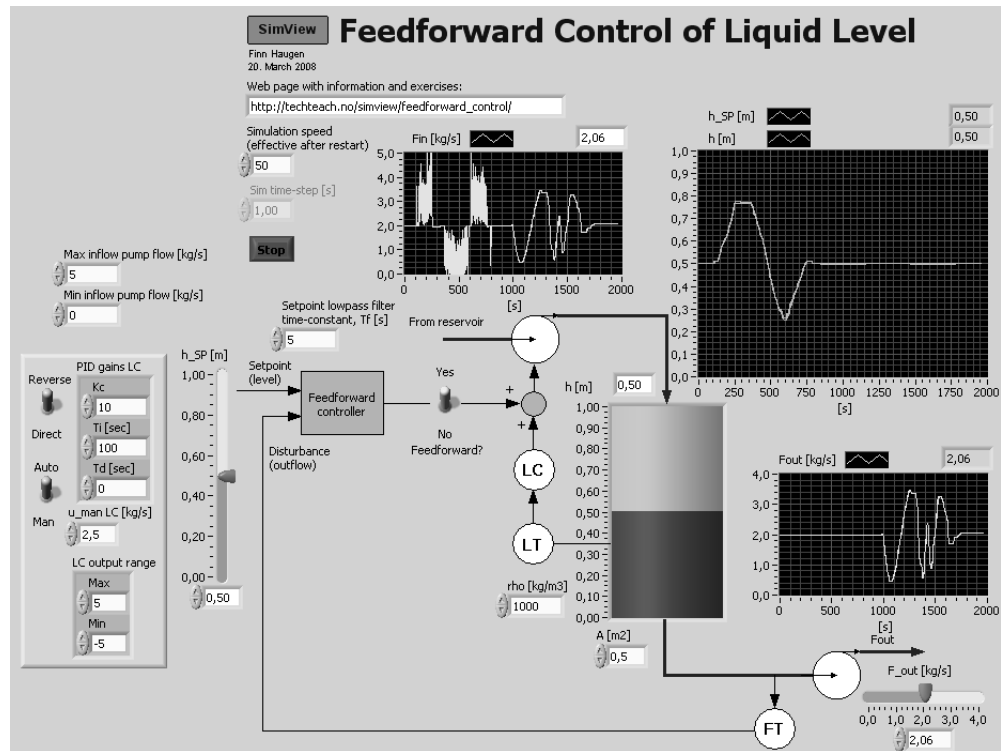


Figure 8.3: Example 8.1: Liquid tank where the level h is controlled with feedback control. Here, feedforward control is applied.

- **With feedforward control** (in addition to feedback control):
Figure 8.3 shows the responses in the level with almost the same variations of the setpoint and the disturbance. We see that the level now deviates very little from the setpoint. The control performance is substantially improved.

Note: Without feedforward control the control signal range of the PID controller is $[0, 5]$ (in unit of m^3/s). With feedforward the output signal range of the PID controller was set to $[-5, +5]$ so that the contribution, u_{PID} , from the PID controller can be negative. If u_{PID} can not become negative, the total control signal, which is

$$u = u_{PID} + u_f \quad (8.9)$$

may not get small enough value to give proper control when the outflow is small. (This was confirmed by a simulation, but the responses are not shown here.)

¹Yes.

[End of Example 8.1]

8.3 Designing feedforward control from experimental data

Feedforward control can be designed from experimental data as follows:

- Decide a proper set of N different values of the disturbance d on which the feedforward control will be based, for example $N = 6$ different values of the disturbance.
- For each of these N distinct disturbance values, find (experimentally or by simulation) the value of the control signal u which corresponds to zero steady state control error. This can (should) be done with PI or PID feedback control. (Typically feedback control is used together with feedforward control, so no extra effort is needed to run the feedback control here.)
- The set of N corresponding values of d and u can be represented by a table, cf. Table 8.1.

u	d
u_1	d_1
u_2	d_2
u_3	d_3
u_4	d_4
u_5	d_5
u_6	d_6

Table 8.1: N corresponding values of v and u

or in a coordinate system, cf. Figure 8.4.

- For any given (measured) value of the disturbance, the feedforward control signal u_f is calculated using interpolation, for example linear interpolation as shown in Figure 8.4. In practice, this linear interpolation can be implemented using a table lookup function.²

Note: This feedforward design method is based on *steady state* data.

Therefore, the feedforward control will not be ideal or perfect. However, it

²Both MATLAB/SIMULINK and LabVIEW have functions that implement linear interpolation between tabular data.

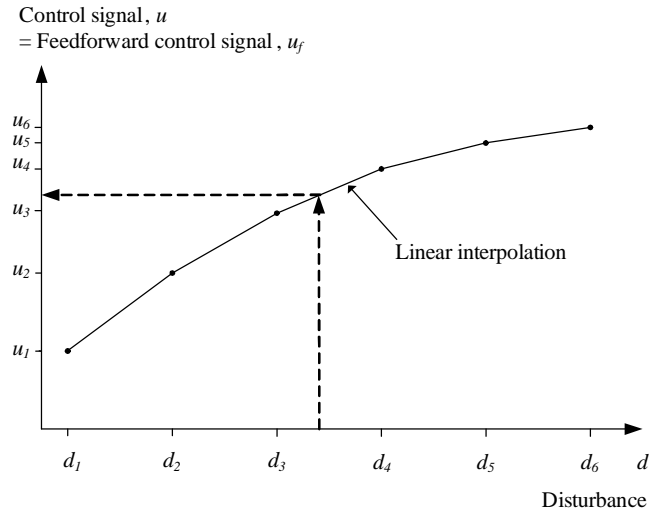


Figure 8.4: Calculation of feedforward control signal from known disturbance value

is easy to implement, and it may give substantial better control compared to only feedback control.

Example 8.2 *Temperature control with feedforward from flow*

Figure 8.5 shows a lab process consisting of a heated air tube where the air temperature will be controlled. The control signal adjusts the power to the heater. The air temperature is measured by the primary Pt100 element. The feedback PID control is based on this temperature measurement. (The control system is implemented with a LabVIEW program running on a laptop PC.)

Variations of the air flow act as disturbances to the process. The feedback controller tries to compensate for such variations using the temperature measurement. Can we obtain improved control by also basing the control signal on measured air flow, which is here available as the fan speed indication? First, ordinary PID control without feedforward is tried. The fan speed was changed from minimum to maximum, and then back again. The temperature setpoint was 40 %. Figure 8.6 shows the fan speed and the response in the temperature (which is represented in % with the range [0–100%] corresponding to [20–70°C]). The maximum control error was 1.0 %.

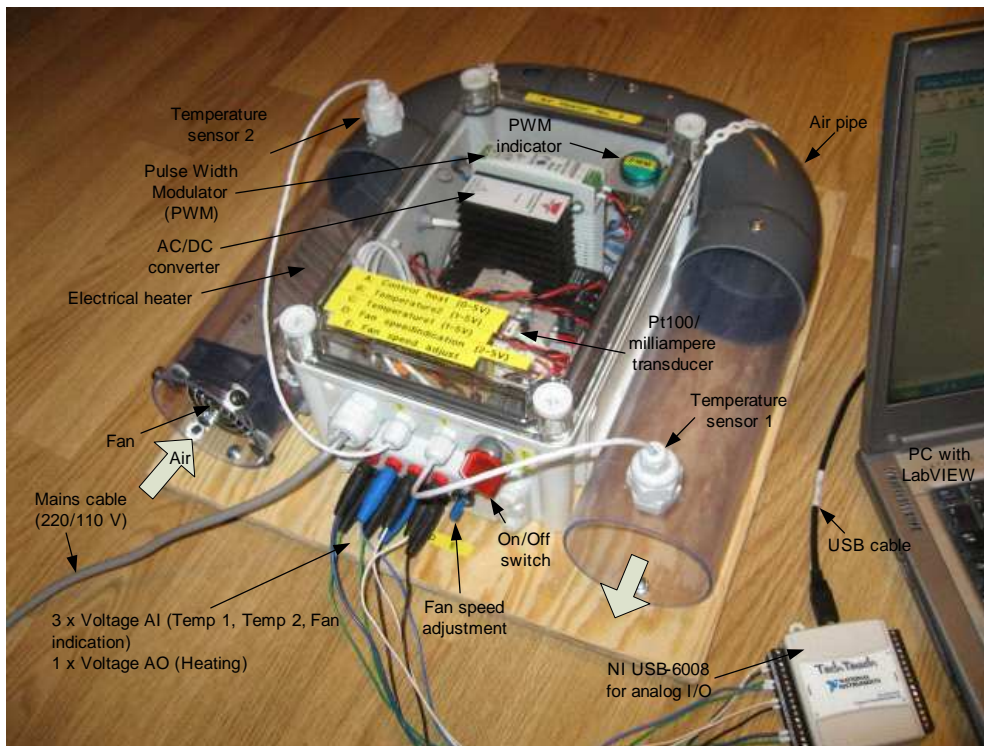


Figure 8.5: Example 8.2: A lab process consisting of a heated air tube where the air temperature will be controlled.

Will there be any improvement by using feedforward control from the fan speed (air flow)? A number of corresponding values of fan speed and control signal was found experimentally. The feedforward control signal, u_f , was calculated by linear interpolation with `Interpolate 1D Array` function in LabVIEW, and was added to the PID control signal to make up the total control signal: $u = u_{PID} + u_f$. Figure 8.7 shows the the fan speed and the response in the temperature. Also, the set of 6 corresponding values of control signal and fan speed, on which the feedforward control signal is based, is shown. In this case the maximum control error was 0.27, which is a large improvement compared to using no feedforward control!

[End of Example 8.2]

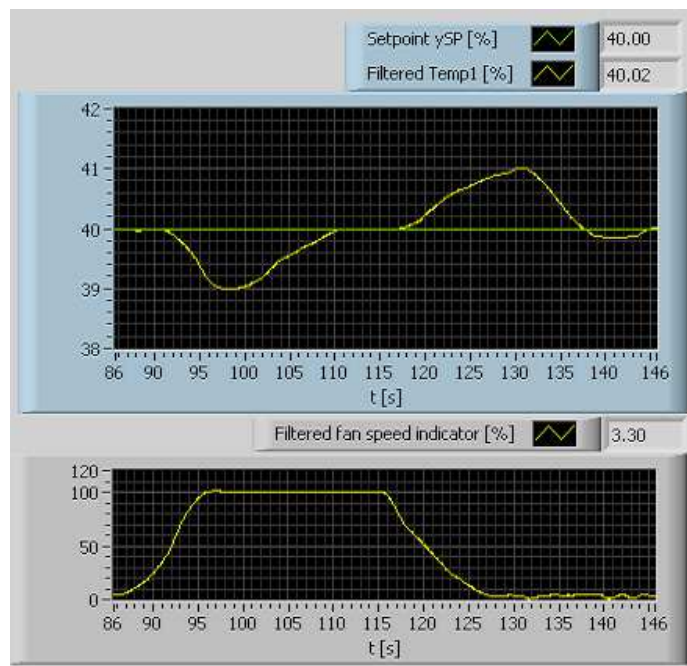


Figure 8.6: Example 8.2: The response in the temperature after a change in the fan speed. Only feedback control (no feedforward) control is used.

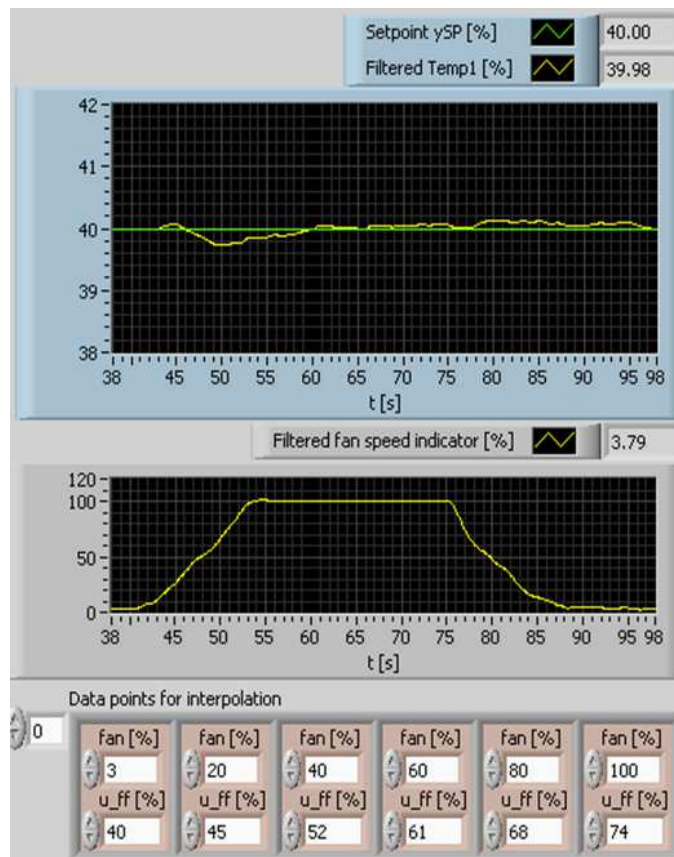


Figure 8.7: Example 8.2: The response in the temperature after a change in the fan speed. Feedforward from fan speed (air flow) is used together with feedback control. u_{ff} is the feedforward control signal, u_f .

Chapter 9

Controller equipment

This chapter gives an overview over various kinds of commercial controller equipment.

9.1 Process controllers

A *process controller* is a single controller unit which can be used to control one process output variable. Figure 9.1 shows an example of a process controller (ABB's ECA600). Figure 9.2 shows the backplane of the ECA600 controller. Both analog and digital sensors (inputs, AI and DI) and actuators (outputs, AO and DO) can be connected to terminals at the backplane, which also has connectors for digital communication (Modbus with RS485 and serial with RS232) with computers or other devices.

Today new process controllers are implemented digitally with a microprocessor. The PID controller function is in the form of a program which runs cyclically, e.g. each 0.1s, which then is called the time step of the controller. Earlier, controllers were build using analog electronics, and even earlier, pneumatic and mechanical components were used. Such controllers are still in use today in some factories since they work well and are safe in dangerous (explosive) areas.

Here is a list of typical characteristics of process controllers:

- The front panel of the controller has vertical bar *indicators and/or numeric displays* showing, see Figure 9.1,
 - the process measurement signal (a common symbol is PV –

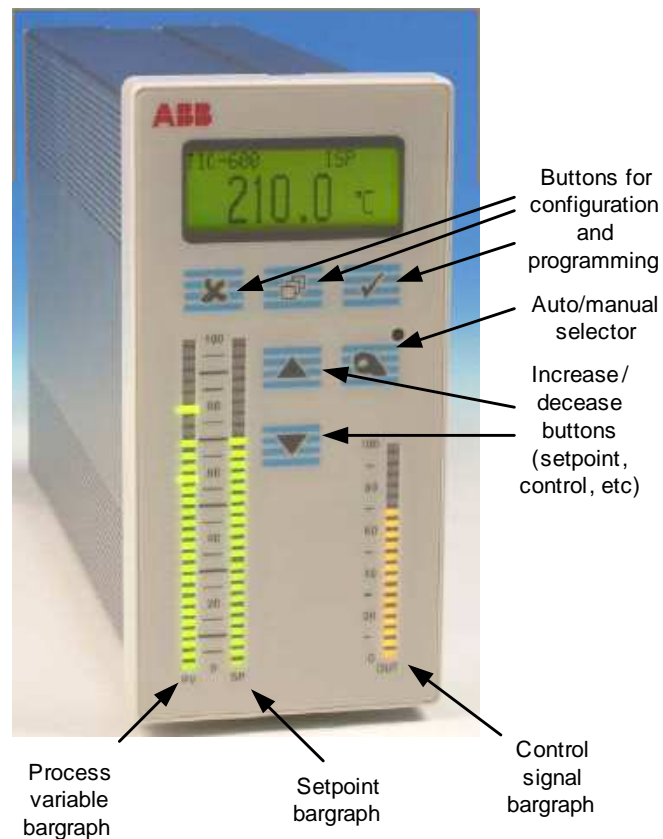


Figure 9.1: A process controller (ABB ECA600)

Process Value),

- the setpoint (symbol SP),
- the control variable (MV – control variable).

- The controller has *analog input* (AI) (for measurement signals) and *analog output* (AO) (for the control variable). The input signal is a voltage signal (in volts) or a current signal (in amperes). The output signal is a current signal. In general, current signals are preferred before voltage signals, because:
 - A current signal is more robust against electrical noise.
 - With long conductors the voltage drop along the conductor may become relatively large due to the internal resistance in the conductor.
 - A current signal of 0 mA is abnormal and indicates a break of the conductor.

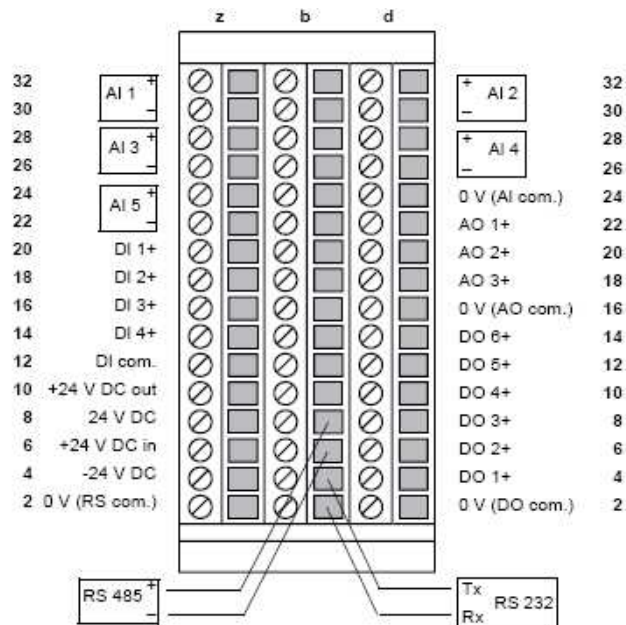


Figure 9.2: The backplane of the ECA600 controller

The standard current range is 4–20 mA (also 0–20 mA is used). There are several standard voltage ranges, as 1–5V¹ and 0–10V. The physical measurement signal in A or V is usually transformed to a percent value using a linear function.

The most important reasons to use 4 mA and not 0 mA as the lower current value, is that the chance that the actual measurement signal is drowned in noise is reduced and that the base signal of 4 mA can be used as an energy source for the sensor or other equipment.

- The controller may have *pulse output*, which may be used to obtain an approximately continuous or smooth control signal using a binary actuator, typically a relay. This technique is called *PWM - Pulse Width Modulation*. The PWM-signal is a sequence of binary signals or on/off-signals used to control the binary actuator. See Figure 9.3. The PWM-signal is kept in the on-state for a certain time-interval of a fixed cycle period. This time-interval (for the on-state) is called the duty cycle, and it is expressed as a number in unit percent. For example, 60% duty cycle means that the PWM-signal is in on-state in 60% of the period (and in the off-state the rest of the period). The duty cycle is equal to the specified analog control signal which is calculated by the PID controller function. In the mean the

¹4–20 mA may be transformed to 1–5 V using a resistor of 250 Ω.

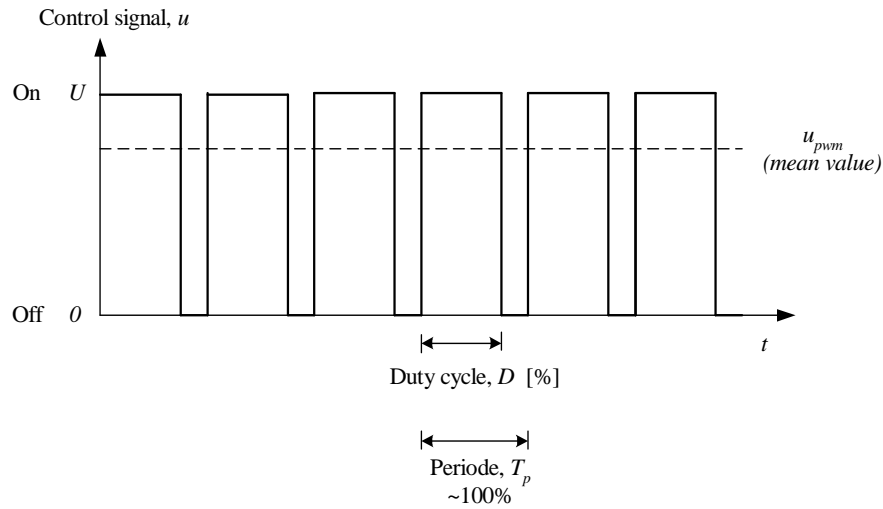


Figure 9.3: Pulse Width Modulation

PWM-signal will become equal to the specified analog control signal, if the cycle period is small compared to the time-constant of the process to be controlled.

- In addition to the analog inputs and outputs the process controller typically have *digital inputs* (on/off-signals) to detect signals from switches, buttons, etc., and *digital outputs* which can be used to control relays, lamps, motors etc.
- The process controller typically have ports for serial (RS-232) *communication* with another controller or a computer.
- The controller can be programmed from a panel on the front or on the side of the unit or from a connected PC. The programming includes combining function modules (see below).
- One of the function modules is the PID controller. You will have to enter values for the PID parameters, typically being K_p , T_i and T_d , but other parameters may be used in stead, as the proportional band, PB, see (7.12).
- Other function modules include logical functions as AND, OR, SR-flipflop, etc., and arithmetic functions, as addition, multiplication, square root calculation, etc.
- In addition to the common single-loop PID control, the process controller may implement more advanced control methods, as

feedforward control (cf. Chapter 8) *gain scheduling PID control* (10.5.3), *cascade control* (11.1) and *ratio control* (11.2).

- The operator can adjust the setpoint *internally* (locally) on the controller, but it is usually possible to use an *external* setpoint, too, which may come from another controller (as in cascade control) or from a sensor (as in ratio control) or from a computer which calculates the setpoint as a result of a plantwide control strategy.
- The operator can take over the control by switching the process controller from *automatic mode* to *manual mode*. This may be necessary if the control program is to be modified, and at first-time start-up. In manual mode the operator adjusts the controller output (which is used as the manipulating variable of the process) manually or directly, while in automatic mode the control output is calculated according to the control function (typically PID control function).
- Many controllers have the possibility of *auto-tuning*, which means that the process controller – initialized by the operator – calculates proper values of the controller parameters based on some automatically conducted experiment on the process to be controlled. An even more advanced option which may be implemented is an *adaptive PID controller* where the PID parameters are calculated continuously from an estimated process model.
- The operator can define alarm limits of the measurement signal. The alarm can be indicated on the front panel of the controller unit, and the alarm limits can be used by the control program to set the value of a digital output of the process controller to turn on e.g. an alarm lamp.

Figure 9.4 shows a section of the data sheet of the controller ECA600 shown in Figure 9.1.

9.2 Programmable logical controller (PLC)

PLCs are common in industrial automation. Figure 9.5 shows a PLC system (Mitsubishi FX2N). PLC is short for Programmable Logical Controller. PLC-systems are modular systems for logical (binary) and sequential control of valves, motors, lamps etc. Modern PLCs includes function modules for PID control. The programming is usually executed on a PC, and the PLC-program is then downloaded (transferred) to the CPU in the PLC-system which then can be disconnected from the PC. The

Controller		Digital Inputs	
Control functions	P, PD, PI, PID, pPI	Type	24 V DC, common digital input ground, current sink, opto-isolated.
Gain	0.01–99.99	Voltage	Max. 35 V, min. -0.5 V.
Integral time	0.1–9999.9 seconds	Logic levels	0 < 3 V (IEC 1131-2, type 1) 1 > 15 V (IEC 1131-2, type 1).
Derivative time	0.0–9999.9 seconds	Digital Outputs	
Control action	Direct, reversed	Type	24 VDC, current source.
Set point	Internal, external, ramp	Load current	Max. 250 mA per output, max. 500 mA total.
Control output	Analogue, pulse	Short-circuit current	Max. 500 mA transient current during 1 μ s.
Alarms	Process value, deviation.	Power supply	
Sample time	30–500 ms	AC	115/230 V AC \pm 10%, 50–60 Hz, 20 VA or 19 V AC \pm 10%, 50–60 Hz, 1 A.
Analogue Inputs		DC	24 V DC \pm 10%
Input ranges	0–20 mA, 4–20 mA, 0–5 V, 1–5 V, 0–10 V, 2–10 V.	Protection	Secondary side of transformer and direct supply fused via thermo type fuse.
Input types	Differential or single ended (jumper selectable).	Transmitter	Max. 24 V DC/150 mA.
Input impedance	Current 250 Ω Voltage 200 k Ω	Environmental specifications	
Alarm function for out-of-range signal	Yes, for 4–20 mA, 1–5 V and 2–10 V, when the signal drops below the lower limit.	Operating temperature	+5 to +55°C (IEC 68-2-1/2).
Functions	First-order software filter, linear / square root.	Non-operating temperature	-25 to +70°C (IEC 68-2-1/2).
Resolution	12 bits	Non-operating damp heat steady state	93% relative humidity at +40°C (IEC 68-2-3).
Inaccuracy	Max. \pm 0.2% of FS within 5–55°C.	Protection class	IP20 generally. IP65 for front. IP65 for front against IP65 compliant panel with panel mounting kit.
Temperature stability	0.01% FS per °C within 5–55°C.	Electrical environment	Fulfills ElectroMagnetic Compatibility, EMC, directive 89/336/EEC
Analogue Outputs		Order codes	ECA 06–0000 ECA 60–0000 ECA 600–0000 EMA 60–0000
Output ranges	0–20 mA, 4–20 mA.		
Type	Current source		
Max. output current	22 mA		
Load resistance on current output	Max. 650 Ω		
Short circuit protection	Yes		
Resolution	12 bits		
Output signal break detection	Yes		
Inaccuracy	Max. \pm 0.2% of FS within 0–50°C.		
Communication			

Figure 9.4: A section of the datasheet of the controller ECA600 shown in Figure 9.1.

program languages are standardized (the IEC 61131-3 standard), but the actual languages which are implemented on commercial PLCs may differ somewhat from the standard.

Figure 9.6 shows the main parts of a PLC.

The PLC operates as follows: The control program executes the program code which is stored in the program storage (memory). The executable program is typically developed on a PC and then downloaded from the PC to the PLC. The data storage contains data used by the program.

Typically the program uses data from the physical inputs, together with data in the data storage, to calculate output values. The program may execute periodically. The cycle time or period is typically in the range of 10ms. The program may also execute if a certain event occurs, e.g. if a program interrupt signal is generated by a digital (boolean) input or by some other external or internal signal source.

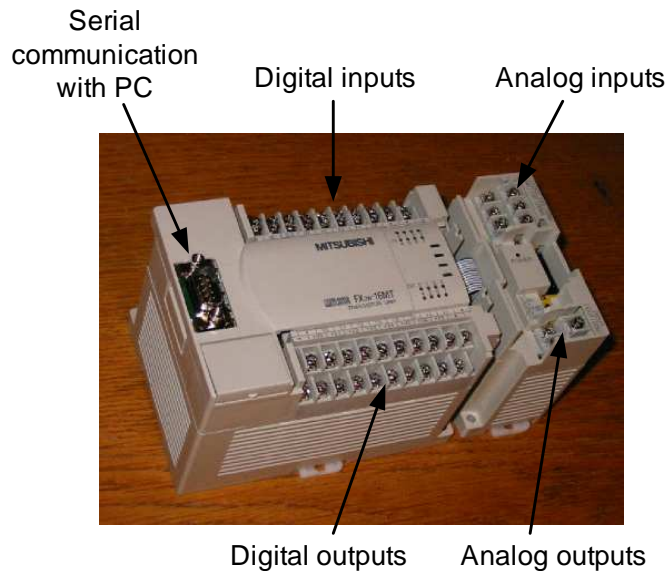


Figure 9.5: A compact PLC (Programmable Logical Controller). (Mitsubishi FX2N)

9.3 Programmable Automation Controller (PAC)

PACs constitute an alternative to PLCs as control hardware. One example of a PAC is National Instruments' (Compact) FieldPoint, see Figure 9.7. It is here assumed that the FieldPoint rack contains a RT unit (Real-time) which contains a microprocessor running a real-time operating system which can run LabVIEW programs downloaded from a PC where the program was developed. To download LabVIEW code, the LabVIEW Real-Time Module must be installed on the PC. Once downloaded, the PAC can be run independently of the PC. National Instruments denotes this equipment a *PAC* – Programmable Automation Controller. The PAC is a modular system similar to PLCs in several aspects. Logical and sequential control and PID control can be realized in the PAC. In fact, all kinds of applications can be developed with LabVIEW and downloaded to the PAC.

9.4 SCADA systems

SCADA systems (Supervisory Control and Data Acquisition) are automation systems where typically PCs are used for supervisory control,

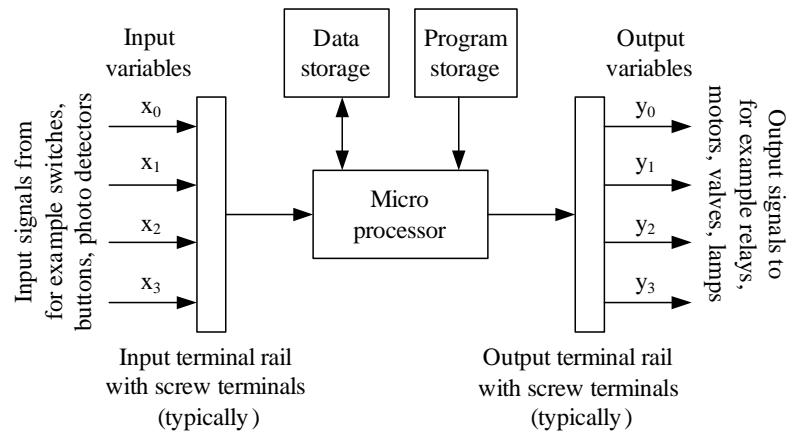


Figure 9.6: The main parts of a PLC

but the execution of the control program takes place in a PLC or some other control equipment. In this way the SCADA system implements a distributed control system architecture. Figure 9.8 shows an example of a SCADA-system (by Siemens).

Here are some characteristics of PC-based control systems:

- A set of function modules are available in the SCADA program on the PC, as arithmetic and logical functions, and signal processing functions. The setpoint to be used by the connected PLC or other control equipment may be calculated by the SCADA program according to some optimal control strategy.
- The user can build the screen contents containing process images of tanks, vessels, valves, etc., and bars, diagrams, numeric displays, alarm indicators etc.
- The PCs can communicate with other PCs or other kinds of computers via standard communication means, as RS-232 cables or Ethernet etc.
- The SCADA system has driver programs for a number of PLC-systems (from different vendors) and other I/O-systems (systems for analog and digital Input/Output). The number of drivers may exceed 100.
- Data can be exchanged between programs in real-time. The OPC

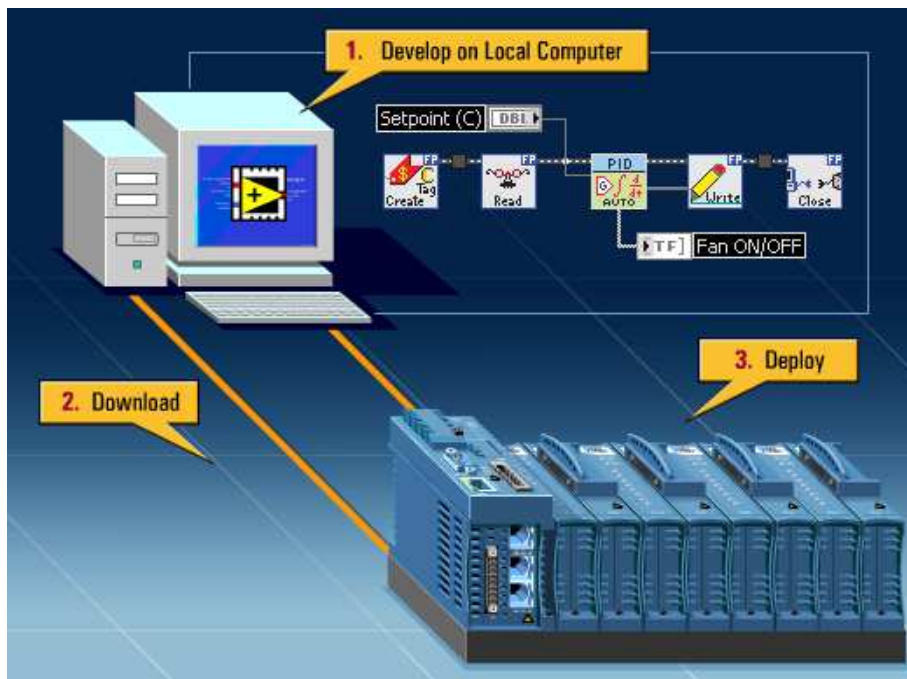


Figure 9.7: Modular control equipment: Compact Fieldpoint, denoted *PAC* - Programmable Automation Controller. (National Instruments)

standard (OLE for Process Control)² has become an important standard for this.

9.5 DCS systems

DCS (Distributed Control Systems) are similar to SCADA systems in that the control equipment is distributed (not centralized) throughout the plant. Special process stations – not standard PLCs or process controllers – executes the control. DCSs can however communicate with PLCs etc. The process stations are mounted in special rooms close to process. The whole plant can be supervised and controlled from control rooms, where the operators communicate with the distributed control equipment, see Figure 9.9.

²OLE = Object Linking and Embedding, which is a technology developed by Microsoft for distributing objects and data between Windows applications.

9.6 Embedded controllers in motors etc.

Producers of electrical and hydraulic servo motors also offers controllers for the motors. These controllers are usually embedded in the motor drive system, as a separate physical unit connected to the motor via cables. The controllers implement speed control and/or positional control. Figure 9.10 shows an example of a servo amplifier for DC-motor. The servo amplifier have an embedded PI controller which the user can tune via screws on the servo amplifier card.

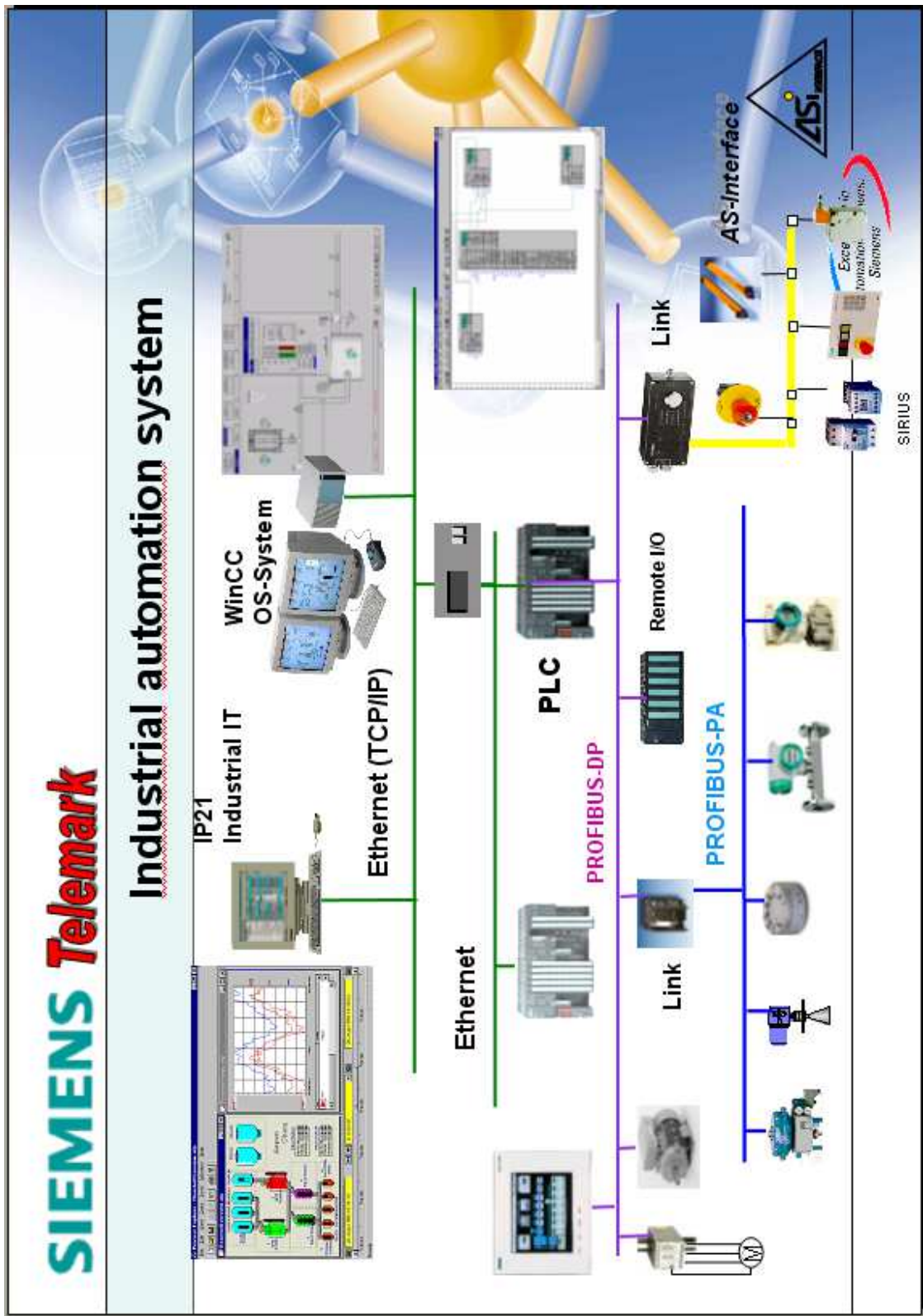


Figure 9.8: SCADA-system (by Siemens). (Reprinted by permission.)



Figure 9.9: Control room of a distributed control system (DCS)

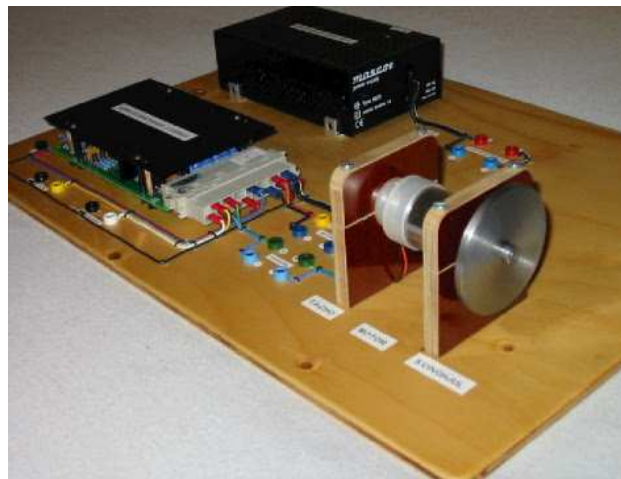


Figure 9.10: DC-motor with servo amplifier (shown behind the motor) implementing a PI-controller

Chapter 10

Tuning of PID controllers

10.1 Introduction

This chapter describes two methods for calculating proper values of the PID parameters K_p , T_i and T_d , i.e. controller tuning. These two methods are:

- **The Good Gain method** [3] which is a simple experimental method which can be used without any knowledge about the process to be controlled. (Of course, if you have a process model, you can use the Good Gain method on a simulator in stead of on the physical process.)
- **Skogestad's method** [7] which is a model-based method. It is assumed that you have mathematical model of the process (a transfer function model). It does not matter how you have derived the transfer function – it can stem from a model derived from physical principles (as described in Ch. 3), or from calculation of model parameters (e.g. gain, time-constant and time-delay) from an experimental response, typically a step response experiment with the process (step on the process input).

A large number of tuning methods exists [1], but it is my view that the above methods will cover most practical cases. What about the famous Ziegler-Nichols' methods – the Ultimate Gain method (or Closed-Loop method) and the Process Reaction curve method (the Open-Loop method)?[8] The Good Gain method has actually many similarities with the Ultimate Gain method, but the latter method has one serious

drawback, namely it requires the control loop to be brought to the limit of stability during the tuning, while the Good Gain method requires a stable loop during the tuning. The Ziegler-Nichols' Open-Loop method is similar to a special case of Skogestad's method, and Skogestad's method is more applicable. Furthermore, the resulting control loop stability with the Ziegler-Nichols' methods is in many cases worse than with other tuning methods. So, I have decided not to include the Ziegler-Nichols' methods in this book.¹

10.2 The Good Gain method

Before going into the procedure of controller tuning with the Good Gain method [3], let's look at what is the aim of the controller tuning. If possible, we would like to obtain both of the following for the control system:

- Fast responses, and
- Good stability

Unfortunately, for most practical processes being controlled with a PID controller, these two wishes can not be achieved simultaneously. In other words:

- The faster responses, the worse stability, and
- The better stability, the slower responses.

For a control system, it is more important that it has good stability than being fast. So, we specify:

Acceptable stability (good stability, but not too good as it gives too slow responses)

Figure 10.1 illustrates the above. It shows the response in the process output variable due to a step change of the setpoint. (The responses correspond to three different controller gains in a simulated control system.)

¹However, both Ziegler-Nichols' methods are described in articles available at <http://techteach.no>.

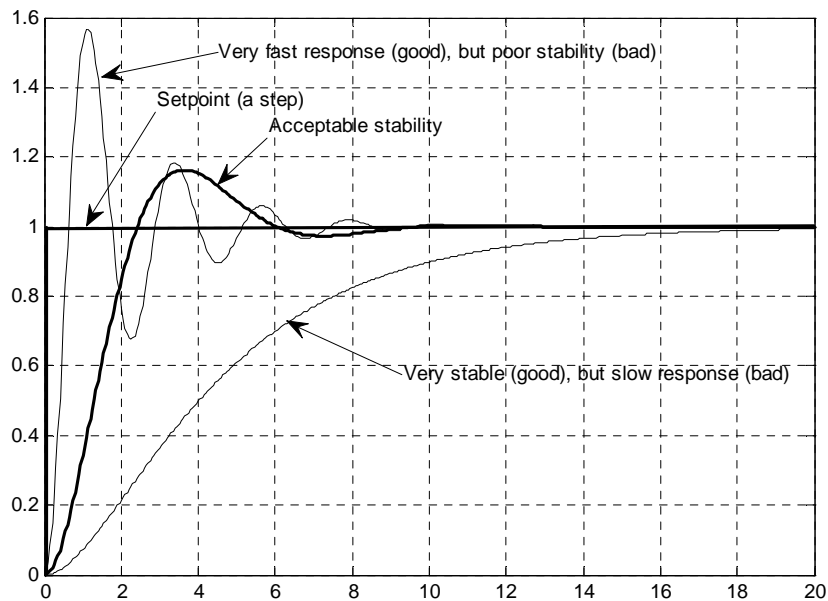


Figure 10.1: In controller tuning we want to obtain acceptable stability of the control system.

What is “acceptable stability” more specifically? There exists no single definition. One simple yet useful definition is as follows. Assume a positive step change of the setpoint. *Acceptable stability is when the undershoot that follows the first overshoot of the response is small, or barely observable.* See Figure 10.1. (If the step change is negative, the terms undershoot and overshoot are interchanged, of course.)

As an alternative to observing the response after a step change of the setpoint, you can regard the response after a step change of the process disturbance. The definition of acceptable stability is the same as for the setpoint change, i.e. that the undershoot (or overshoot – depending on the sign of the disturbance step change) that follows the first overshoot (or undershoot) is small, or barely observable.

The Good Gain method aims at obtaining acceptable stability as explained above. It is a simple method which has proven to give good results on laboratory processes and on simulators. The method is based on experiments on a real or simulated control system, see Figure 10.2. The procedure described below assumes a PI controller, which is the most commonly used controller function. However, a comment about how to

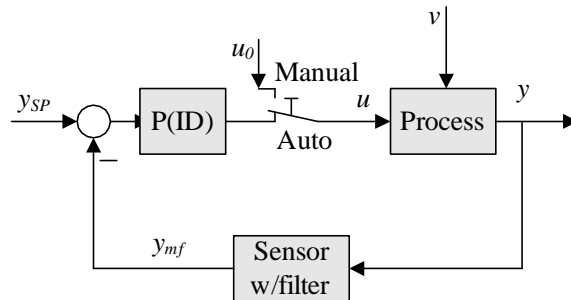


Figure 10.2: The Good Gain method for PID tuning is applied to the established control system.

include the D-term, so that the controller becomes a PID controller, is also given.

1. Bring the process to or close to the normal or specified operation point by adjusting the nominal control signal u_0 (with the controller in manual mode).
2. Ensure that the controller is a P controller with $K_p = 0$ (set $T_i = \infty$ and $T_d = 0$). Increase K_p until the control loop gets good (satisfactory) stability as seen in the response in the measurement signal after e.g. a step in the setpoint or in the disturbance (exciting with a step in the disturbance may be impossible on a real system, but it is possible in a simulator). If you do not want to start with $K_p = 0$, you can try $K_p = 1$ (which is a good initial guess in many cases) and then increase or decrease the K_p value until you observe some overshoot and a barely observable undershoot (or vice versa if you apply a setpoint step change the opposite way, i.e. a negative step change), see Figure 10.3. This kind of response is assumed to represent good stability of the control system. This gain value is denoted K_{pGG} .

It is important that *the control signal is not driven to any saturation limit* (maximum or minimum value) during the experiment. If such limits are reached the K_p value may not be a good one – probably too large to provide good stability when the control system is in normal operation. So, you should apply a relatively small step change of the setpoint (e.g. 5% of the setpoint range), but not so small that the response drowns in noise.

3. Set the integral time T_i equal to

$$\underline{T_i = 1.5T_{ou}} \quad (10.1)$$

where T_{ou} is the time between the overshoot and the undershoot of the step response (a step in the setpoint) with the P controller, see Figure 10.3.² Note that for most systems (those which does not contain a pure integrator) there will be offset from setpoint because the controller during the tuning is just a P controller.

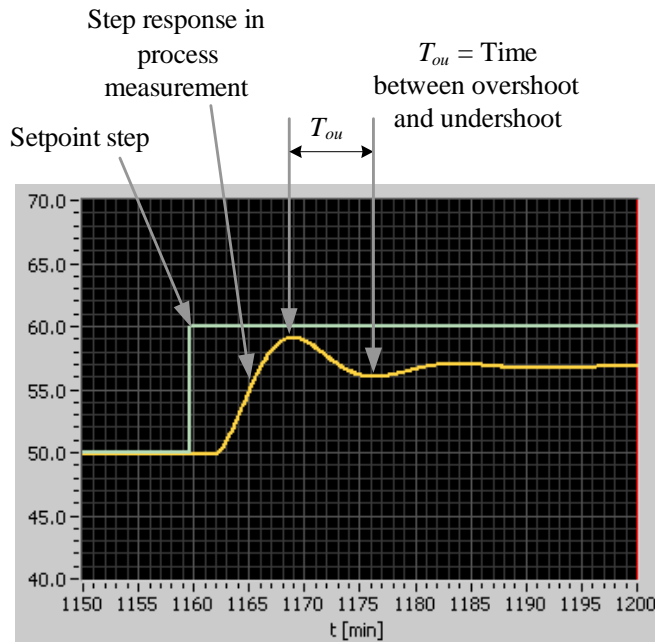


Figure 10.3: The Good Gain method: Reading off the time between the overshoot and the undershoot of the step response with P controller

4. Because of the introduction of the I-term, the loop with the PI controller in action will probably have somewhat reduced stability than with the P controller only. To compensate for this, the K_p can be reduced somewhat, e.g. to 80% of the original value. Hence,

$$\underline{K_p = 0.8K_{pGG}} \quad (10.2)$$

5. If you want to include the D-term, so that the controller becomes a PID controller³, you can try setting T_d as follows:

$$T_d = \frac{T_i}{4} \quad (10.3)$$

²Alternatively, you may apply a negative setpoint step, giving a similar response but downwards. In this case T_{ou} is time between the undershoot and the overshoot.

³But remember the drawbacks about the D-term, namely that it amplifies the measurement noise, causing a more noisy controller signal than with a PI controller.

which is the T_d - T_i relation that was used by Ziegler and Nichols [8].

6. You should check the stability of the control system with the above controller settings by applying a step change of the setpoint. If the stability is poor, try reducing the controller gain somewhat, possibly in combination with increasing the integral time.

Example 10.1 *PI controller tuning of a wood-chip level control system with the Good Gain Method*

I have used the Good Gain method to tune a PI controller on a simulator of the level control system for the wood-chip tank, cf. Figure 1.2. During the tuning I found

$$K_{pGG} = 1.5 \quad (10.4)$$

and

$$T_{ou} = 12 \text{ min} \quad (10.5)$$

The PI parameter values are

$$K_p = 0.8K_{pGG} = 0.8 \cdot 1.5 = 1.2 \quad (10.6)$$

$$T_i = 1.5T_{ou} = 1.5 \cdot 12 \text{ min} = 18 \text{ min} = 1080 \text{ s} \quad (10.7)$$

Figure 10.4 shows the resulting responses with a setpoint step at time 20 min and a disturbance step (outflow step from 1500 to 1800 kg/min) at time 120 min. The control system has good stability.

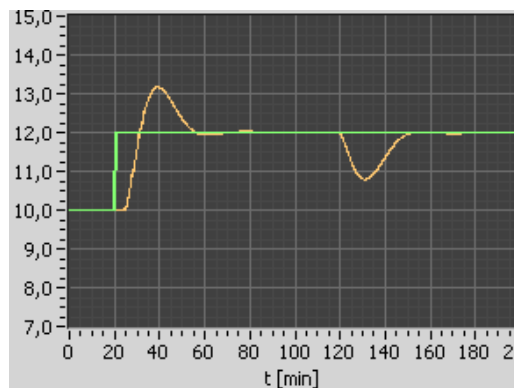


Figure 10.4: Example 10.1: Level control of the wood-chip tank with a PI controller.

[End of Example 10.1]

10.3 Skogestad's PID tuning method

10.3.1 The background of Skogestad's method

Skogestad's PID tuning method [7]⁴ is a model-based tuning method where *the controller parameters are expressed as functions of the process model parameters*. It is assumed that the control system has a transfer function block diagram as shown in Figure 10.5.

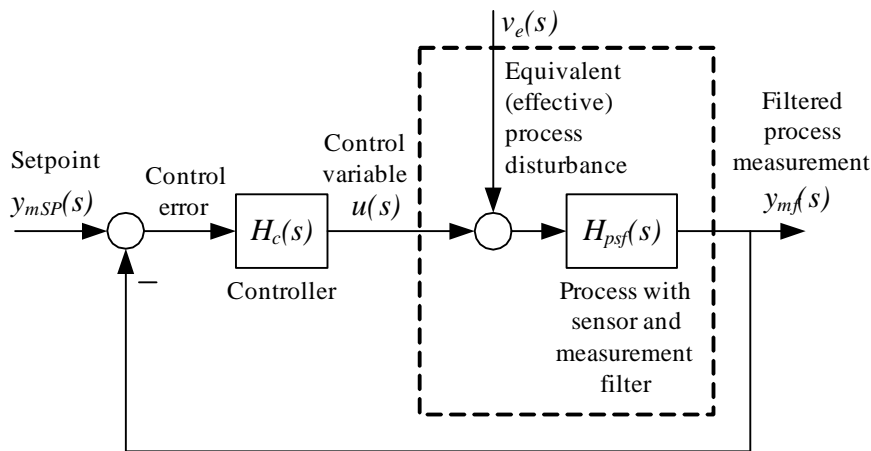


Figure 10.5: Block diagram of the control system in PID tuning with Skogestad's method

Comments to this block diagram:

- The transfer function $H_{psf}(s)$ is a combined transfer function of the process, the sensor, and the measurement lowpass filter. Thus, $H_{psf}(s)$ represents all the dynamics that the controller “feels”. For simplicity we may denote this transfer function the “process transfer function”, although it is a combined transfer function.
- The process transfer function can stem from a simple step-response experiment with the process. This is explained in Sec. 10.3.3.
- The block diagram shows a disturbance acting on the process. Information about this disturbance is not used in the tuning, but if you are going to test the tuning on a simulator to see how the control system compensates for a process disturbance, you should add a

⁴Named after the originator Prof. Sigurd Skogestad

disturbance at the point indicated in the block diagram, which is at the process input. It turns out that in most processes the dominating disturbance influences the process dynamically at the “same” point as the control variable. Such a disturbance is called an *input disturbance*. Here are a few examples:

- Liquid tank: The control variable controls the inflow. The outflow is a disturbance.
- Motor: The control variable controls the motor torque. The load torque is a disturbance.
- Thermal process: The control variable controls the power supply via a heating element. The power loss via heat transfer through the walls and heat outflow through the outlet are disturbances.

The design principle of Skogestad’s method is as follows. The control system *tracking transfer function* $T(s)$, which is the transfer function from the setpoint to the (filtered) process measurement, is *specified* as a first order transfer function with time delay:

$$T(s) = \frac{y_{mf}(s)}{y_{mSP}(s)} = \frac{1}{T_C s + 1} e^{-\tau s} \quad (10.8)$$

where T_C is the time-constant of the control system which *the user must specify*, and τ is the process time delay which is *given* by the process model (the method can however be used for processes without time delay, too). Figure 10.6 shows as an illustration the response in y_{mf} after a step in the setpoint y_{mSP} for (10.8).

From the block diagram shown in Figure 10.5 the tracking transfer function is, cf. the Feedback rule in Figure 5.2,

$$T(s) = \frac{H_c(s)H_{psf}(s)}{1 + H_c(s)H_{psf}(s)} \quad (10.9)$$

Setting (10.9) equal to (10.8) gives

$$\frac{H_c(s)H_{psf}(s)}{1 + H_c(s)H_{psf}(s)} = \frac{1}{T_C s + 1} e^{-\tau s} \quad (10.10)$$

Here, the only unknown is the controller transfer function, $H_c(s)$. By making some proper simplifying approximations to the time delay term, the controller becomes a PID controller or a PI controller for the process transfer function assumed.

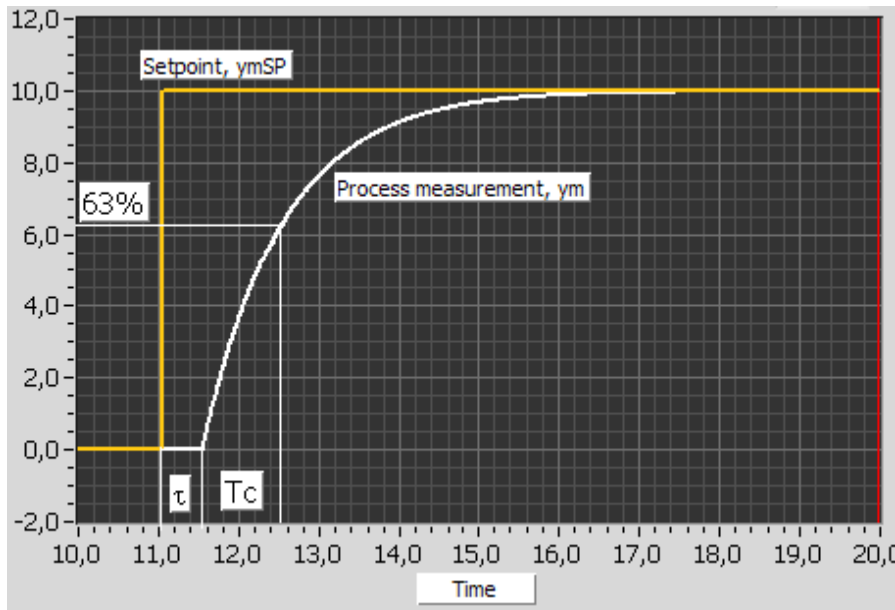


Figure 10.6: Step response of the specified tracking transfer function (10.8) in Skogestad's PID tuning method

10.3.2 The tuning formulas in Skogestad's method

Skogestad's tuning formulas for a number of processes are shown in Table 10.1.⁵

Process type	$H_{psf}(s)$ (process)	K_p	T_i	T_d
Integrator + delay	$\frac{K}{s} e^{-\tau s}$	$\frac{1}{K(T_C + \tau)}$	$c(T_C + \tau)$	0
Time-constant + delay	$\frac{K}{Ts+1} e^{-\tau s}$	$\frac{T}{K(T_C + \tau)}$	$\min [T, c(T_C + \tau)]$	0
Integr + time-const + del.	$\frac{K}{(Ts+1)s} e^{-\tau s}$	$\frac{1}{K(T_C + \tau)}$	$c(T_C + \tau)$	T
Two time-const + delay	$\frac{K}{(T_1s+1)(T_2s+1)} e^{-\tau s}$	$\frac{T_1}{K(T_C + \tau)}$	$\min [T_1, c(T_C + \tau)]$	T_2
Double integrator + delay	$\frac{K}{s^2} e^{-\tau s}$	$\frac{1}{4K(T_C + \tau)^2}$	$4(T_C + \tau)$	$4(T_C + \tau)$

Table 10.1: Skogestad's formulas for PI(D) tuning.

For the "Two time-constant + delay" process in Table 10.1 T_1 is the largest and T_2 is the smallest time-constant.⁶

Originally, Skogestad defined the factor c in Table 10.1 as 4. This gives

⁵In the table, "min" means the minimum value (of the two alternative values).

⁶[7] also describes methods for model reduction so that more complicated models can be approximated with one of the models shown in Table 10.1.

good setpoint tracking. But the disturbance compensation may become quite sluggish. To obtain faster disturbance compensation, I suggest [3]

$$c = 2 \quad (10.11)$$

The drawback of such a reduction of c is that there will be somewhat more overshoot in the setpoint step respons, and that the stability of the control loop will be somewhat reduced. Also, the robustness against changes of process parameters (e.g. increase of process gain and increase of process time-delay) will be somewhat reduced.

Skogestad suggests using

$$T_C = \tau \quad (10.12)$$

for T_C in Table 10.1 – unless you have reasons for a different specification of T_C .

Example 10.2 Control of first order system with time delay

Let us try Skogestad's method for tuning a PI controller for the (combined) process transfer function

$$H_{psf}(s) = \frac{K}{Ts + 1} e^{-\tau s} \quad (10.13)$$

(time-constant with time-delay) where

$$K = 1; T = 1 \text{ s}; \tau = 0.5 \text{ s} \quad (10.14)$$

We use (10.12):

$$T_C = \tau = 0.5 \text{ s} \quad (10.15)$$

The controller parameters are as follows, cf. Table 10.1:

$$K_p = \frac{T}{K(T_C + \tau)} = \frac{1}{1 \cdot (0.5 + 0.5)} = 1 \quad (10.16)$$

$$T_i = \min [T, c(T_C + \tau)] \quad (10.17)$$

$$= \min [1, 2(0.5 + 0.5)] \quad (10.18)$$

$$= \min [1, 2] \quad (10.19)$$

$$= 1 \text{ s} \quad (10.20)$$

$$T_d = 0 \quad (10.21)$$

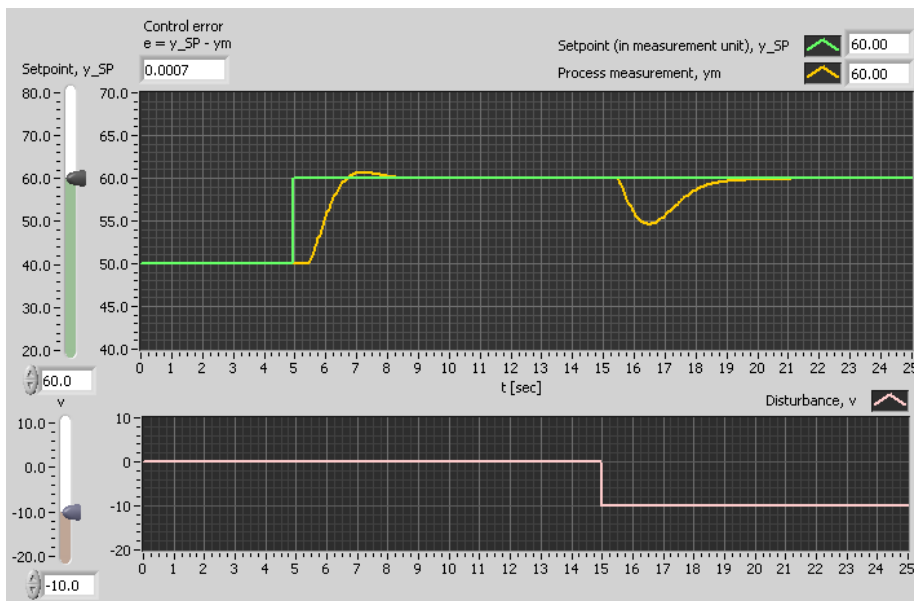


Figure 10.7: Example 10.2: Simulated responses in the control system with Skogestad's controller tuning

Figure 10.7 shows control system responses with the above PID settings. At time 5 sec the setpoint is changed as a step, and at time 15 sec the disturbance is changed as a step. The responses, and in particular the stability of the control systems, seem ok.

[End of Example 10.2]

You may wonder: Given a process model as in Table 10.1. Does Skogestad's method give better control than if the controller was tuned with some other method, e.g. the Good Gain method? There is no unique answer to that question, but my impression is that Skogestad's method in general works fine. If you have a mathematical model of the process to be controlled, you should always simulate the system with alternative controller tunings. The benefit of Skogestad's method is that you do not have to perform trial-and-error simulations to tune the controller. The parameters come directly from the process model and the specified control system response time. Still, you should run simulations to check the performance.

10.3.3 How to find model parameters from experiments

The values of the parameters of the transfer functions in Table 10.1 can be found from a mathematical model based on physical principles, cf. Chapter 3. The parameter values can also be found from a step-response experiment with the process. This is shown for the model *Integrator with time-delay* and *Time-constant with time-delay* in the following respective figures. (The theory of calculating these responses is covered by Chapter 6.)

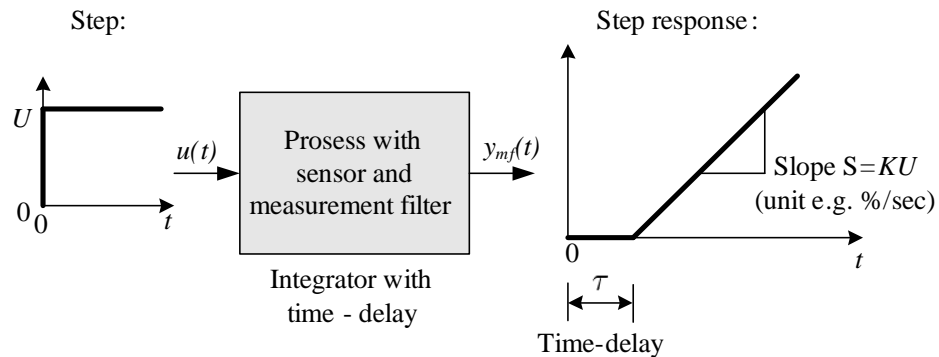


Figure 10.8: How the transfer function parameters K and τ appear in the step response of an *Integrator with time-delay* process

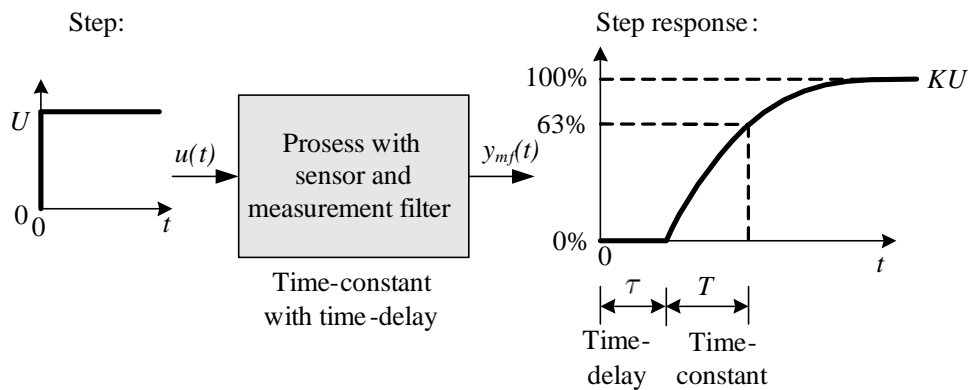


Figure 10.9: How the transfer function parameters K , T , and τ appear in the step response of a *Time-constant with time-delay* process

10.3.4 Transformation from serial to parallel PID settings

Skogestad's formulas assumes a *serial* PID controller function (alternatively denoted cascade PID controller) which has the following transfer function:

$$u(s) = K_{ps} \frac{(T_{is}s + 1)(T_{ds}s + 1)}{T_{is}s} e(s) \quad (10.22)$$

where K_{ps} , T_{is} , and T_{ds} are the controller parameters. If your controller actually implements a *parallel* PID controller (as in the PID controllers in LabVIEW PID Control Toolkit and in the Matlab/Simulink PID controllers), which has the following transfer function:

$$u(s) = \left[K_{pp} + \frac{K_{ip}}{T_{ip}s} + K_{pd}s \right] e(s) \quad (10.23)$$

then you should transform from serial PID settings to parallel PID settings. If you do not implement these transformations, the control system may behave unnecessarily different from the specified response. The serial-to-parallel transformations are as follows:

$$K_{pp} = K_{ps} \left(1 + \frac{T_{ds}}{T_{is}} \right) \quad (10.24)$$

$$T_{ip} = T_{is} \left(1 + \frac{T_{ds}}{T_{is}} \right) \quad (10.25)$$

$$T_{dp} = T_{ds} \frac{1}{1 + \frac{T_{ds}}{T_{is}}} \quad (10.26)$$

Note: The parallel and serial PI controllers are identical (since $T_d = 0$ in a PI controller). Therefore, the above transformations are not relevant for PI controller, only for PID controllers.

10.3.5 When the process has no time-delay

What if the process $H_p(s)$ is *without time-delay*? Then you can not specify T_C according to (10.12) since that would give $T_C = 0$ (zero response time of the control system). You must specify T_C to some reasonable value larger than zero. If you do not know what could be a reasonable value, you can simulate the control system for various values of T_C . If the control signal (controller output signal) is changing too quickly, or often reaches the maximum and minimum values for reasonable changes of the setpoint

or the disturbance, the controller is too aggressive, and you can try increasing T_C . If you don't want to simulate, then just try setting $T_C = T/2$ where T is the dominating (largest) time-constant of the process (assuming the process is a time-constant system, of course).

For the double integrator (without time-delay) I have seen in simulations that the actual response-time (or 63% rise-time) of the closed-loop system may be about twice the specified time-constant T_C . Consequently, you can set T_C to about half of the response-time you actually want to obtain.

10.4 Auto-tuning

Auto-tuning is automatic tuning of controller parameters in one experiment. It is common that commercial controllers offers auto-tuning. The operator starts the auto-tuning via some button or menu choice on the controller. The controller then executes automatically a pre-planned experiment on the uncontrolled process or on the control system depending on the auto-tuning method implemented. Below are described a couple of auto-tuning methods.

Auto-tuning based on relay tuning

The relay method for tuning PID controllers is used as the basis of auto-tuning in some commercial controllers.⁷ The principle of this method is as follows: When the auto-tuning phase is started, a relay controller is used as the controller in the control loop, see Figure 10.10. The relay controller is simply an On/Off controller. It sets the control signal to a high (On) value when the control error is positive, and to a low (Off) value when the control error is negative. This controller creates automatically sustained oscillations in control loop, and from the amplitude and the period of these oscillations proper PID controller parameters are calculated by an algorithm in the controller. Only a few periods are needed for the autotuner to have enough information to accomplish the tuning. The autotuner activates the tuned PID controller automatically after the tuning has finished.

⁷For example the ABB ECA600 PID controller and the Fuji PGX PID controller.

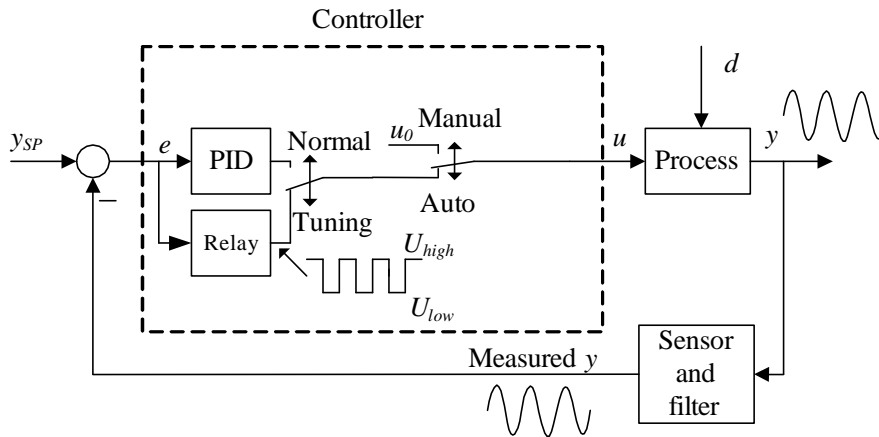


Figure 10.10: Relay control (On/Off control) used in autotuning

Model-based auto-tuning

Commercial software tools⁸ exist for auto-tuning based on an estimated process model developed from a sequence of logged data – or time-series – of the control variable u and process measurement y_m . The process model is a “black-box” input-output model in the form of a transfer function. The controller parameters are calculated automatically on the basis of the estimated process model. The time-series of u and y_m may be logged from the system being in closed loop or in open loop:

- **Closed loop**, with the control system being excited via the setpoint, y_{SP} , see Figure 10.11. The closed loop experiment may be used when the controller should be re-tuned, that is, the parameters should be optimized.
- **Open loop**, with the process, which in this case is *not* under control, being excited via the control variable, u , see Figure 10.12. This option must be made if there are no initial values of the controller parameters.

⁸E.g. MultiTune (Norwegian) and ExperTune

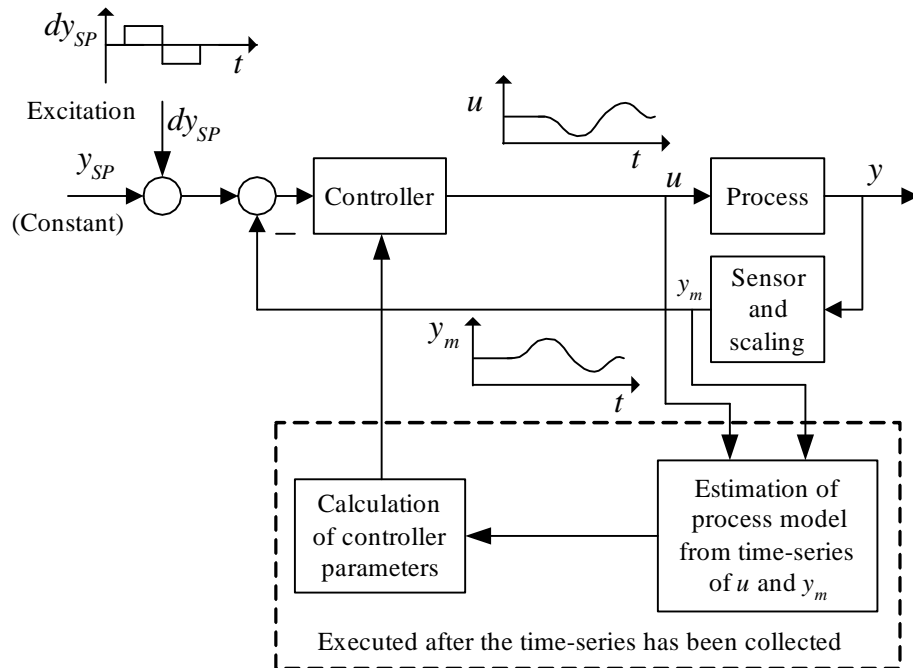


Figure 10.11: Auto-tuning based on closed loop excitation via the setpoint

10.5 PID tuning when process dynamics varies

10.5.1 Introduction

A well tuned PID controller has parameters which are adapted to the dynamic properties to the process, so that the control system becomes fast and stable. If the process dynamic properties varies without re-tuning the controller, the control system

- gets *reduced stability*, or
- becomes *more sluggish*.

Problems with variable process dynamics can be solved in the following alternative ways:

- **The controller is tuned in the most critical operation point**, so that when the process operates in a different operation point, the

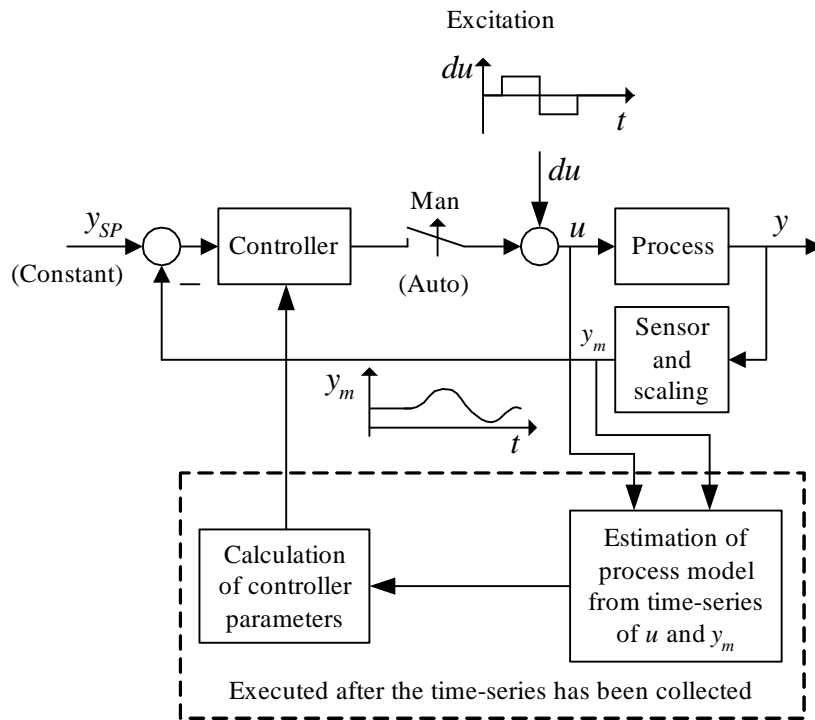


Figure 10.12: Auto-tuning based on open loop excitation via the control variable

stability of the control system is just better — at least the stability is not reduced. However, if the stability is too good the tracking quickness is reduced, giving more sluggish control.

- **The controller parameters are varied in the “opposite” direction of the variations of the process dynamics**, so that the performance of the control system is maintained, independent of the operation point. Some ways to vary the controller parameters are:
 - *Model-based PID parameter adjustment*, cf. Section 10.5.2.
 - *PID controller with gain scheduling*, cf. Section 10.5.3.
 - *Model-based adaptive controller*, cf. Section 10.5.4.

Commercial control equipment is available with options for gain scheduling and/or adaptive control.

10.5.2 PID parameter adjustment with Skogestad's method

Assume that you have tuned a PID or a PI controller for some process that has a transfer function equal to one of the transfer functions $H_{psf}(s)$ shown in Table 10.1. Assume then that some of the parameters of the process transfer function changes. How should the controller parameters be adjusted? The answer is given by Table 10.1 because it gives the controller parameters as functions of the process parameters. You can actually use this table as the basis for adjusting the PID parameters even if you used some other method than Skogestad's method for the initial tuning.

From Table 10.1 we can draw a number of general rules for adjusting the PID parameters:

Example 10.3 *Adjustment of PI controller parameters for integrator with time delay process*

Assume that the process transfer function is

$$H_{psf}(s) = \frac{K}{s} e^{-\tau s} \quad (10.27)$$

(integrator with time delay). According to Table 10.1, with the suggested specification $T_C = \tau$, the PI controller parameters are

$$K_p = \frac{1}{2K\tau} \quad (10.28)$$

$$T_i = k_1 2\tau \quad (10.29)$$

As an example, assume that the process gain K is increased to, say, twice its original value. How should the PI parameters be adjusted to maintain good behaviour of the control system? From (10.28) we see that K_p should be halved, and from (10.29) we see that T_i should remain unchanged.

As another example, assume that the process time delay τ is increased to, say, twice its original value. From (10.28) we see that K_p should be halved, and from (10.29) we see that T_i should get a doubled value. One concrete example of such a process change is the wood-chip tank. If the speed of the conveyor belt is halved, the time delay (transport delay) is doubled. And now you know how to quickly adjust the PI controller parameters if such a change of the conveyor belt speed should occur.⁹

[End of Example 10.3]

⁹What may happen if you do not adjust the controller parameters? The control system may get poor stability, or it may even become unstable.

10.5.3 Gain scheduling of PID parameters

Figure 10.13 shows the structure of a control system for a process which may have varying dynamic properties, for example a varying gain. The

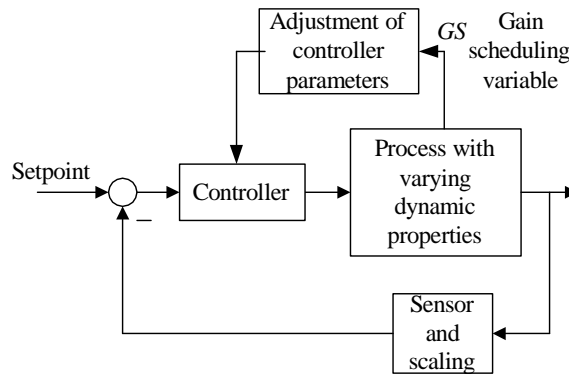


Figure 10.13: Control system for a process having varying dynamic properties. The GS variable expresses or represents the dynamic properties of the process.

Gain scheduling variable GS is some measured process variable which at every instant of time expresses or represents the dynamic properties of the process. As you will see in Example 10.4, GS may be the mass flow through a liquid tank.

Assume that proper values of the PID parameters K_p , T_i and T_d are found (using e.g. the Good Gain method) for a set of values of the GS variable. These PID parameter values can be stored in a parameter table – the gain schedule – as shown in Table 10.2. From this table proper PID parameters are given as functions of the gain scheduling variable, GS .

GS	K_p	T_i	T_d
GS_1	K_{p1}	T_{i1}	T_{d1}
GS_2	K_{p2}	T_{i2}	T_{d2}
GS_3	K_{p3}	T_{i3}	T_{d3}

Table 10.2: Gain schedule or parameter table of PID controller parameters.

There are several ways to express the PID parameters as functions of the GS variable:

- **Piecewise constant:** An interval is defined around each GS value in the parameter table. The controller parameters are kept constant as long as the GS value is within the interval. This is a simple

solution, but it seems nonetheless to be the most common solution in commercial controllers.

When the GS variable changes from one interval to another, the controller parameters are changed abruptly, see Figure 10.14 which illustrates this for K_p , but the situation is the same for T_i and T_d . In Figure 10.14 it is assumed that GS values toward the left are critical with respect to the stability of the control system. In other words: It is assumed that it is safe to keep K_p constant and equal to the K_p value in the left part of the interval.

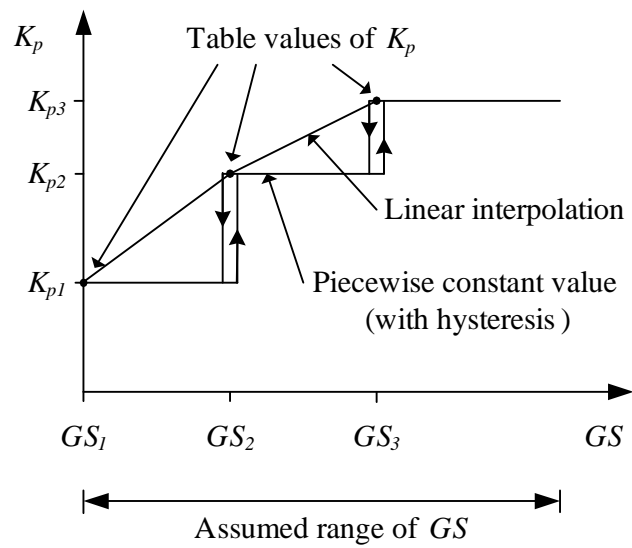


Figure 10.14: Two different ways to interpolate in a PID parameter table: Using piecewise constant values and linear interpolation

Using this solution there will be a disturbance in the form of a step in the control variable when the GS variable shifts from one interval to another, but this disturbance is probably of negligible practical importance for the process output variable. Noise in the GS variable may cause frequent changes of the PID parameters. This can be prevented by using a hysteresis, as shown in Figure 10.14.

- **Piecewise linear**, which means that a linear function is found relating the controller parameter (output variable) and the GS variable (input variable) between to adjacent sets of data in the table. The linear function is on the form

$$K_p = a \cdot GS + b \quad (10.30)$$

where a and b are found from the two corresponding data sets:

$$K_{p_1} = a \cdot GS_1 + b \quad (10.31)$$

$$K_{p_2} = a \cdot GS_2 + b \quad (10.32)$$

(Similar equations applies to the T_i parameter and the T_d parameter.) (10.31) and (10.32) constitute a set of two equations with two unknown variables, a and b (the solution is left to you).¹⁰

- **Other interpolations** may be used, too, for example a polynomial function fitted exactly to the data or fitted using the least squares method.

Example 10.4 *PID temperature control with gain scheduling during variable mass flow*

Figure 10.16 shows the front panel of a simulator for a temperature control system for a liquid tank with variable mass flow, w , through the tank. The control variable u controls the power to heating element. The temperature T is measured by a sensor which is placed some distance away from the heating element. There is a time delay from the control variable to measurement due to imperfect blending in the tank.

The process dynamics

We will initially, both in simulations and from analytical expressions, that the dynamic properties of the process *varies with the mass flow w* . The response in the temperature T after a step change in the control signal (which is proportional to the supplied power) is simulated for a large mass flow and a small mass flow. (Feedback temperature control is not active, thus open loop responses are shown.) The responses are shown in Figure 10.15. The simulations show that the following happens when the mass flow w is reduced (from 24 to 12 kg/min): *The gain process K is larger*. It can be shown that in addition, the time-constant T_t is larger, and the time delay τ is larger. (These terms assumes that system is a first order system with time delay. The simulator is based on such a model. The model is described below.)

Let us see if the way the process dynamics seems to depend on the mass flow w as seen from the simulations, can be confirmed from a

¹⁰Note that both MATLAB/SIMULINK and LabVIEW have functions that implement linear interpolation between tabular data. Therefore gain scheduling can be easily implemented in these environments.

Responses in temperature T [°C] after step amplitude of 10% in control signal, u

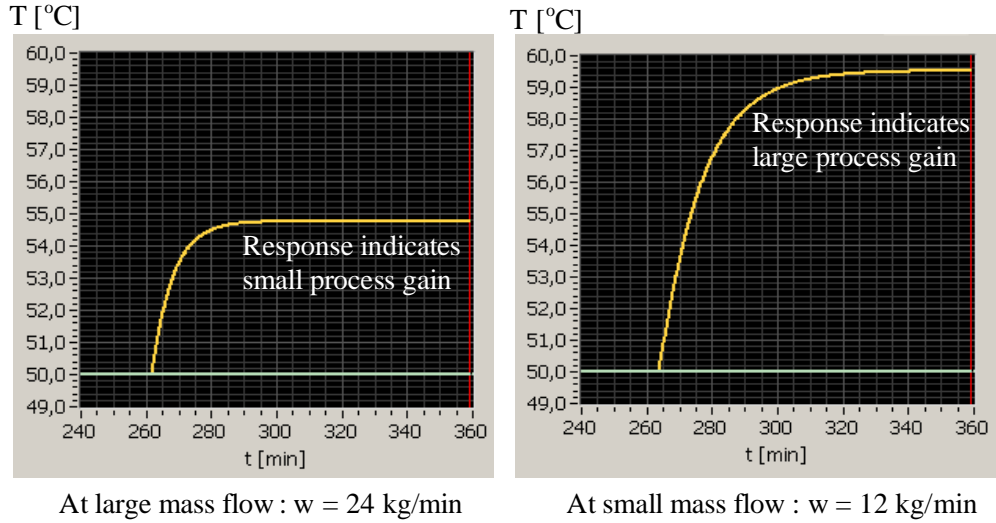


Figure 10.15: Responses in temperature T after a step in u of amplitude 10% at large mass flow and small mass flow

mathematical process model.¹¹ Assuming perfect stirring in the tank to have homogeneous conditions in the tank, we can set up the following energy balance for the liquid in the tank:

$$c\rho VT_1\dot{T}_1(t) = K_P u(t) + cw [T_{in}(t) - T_t(t)] \quad (10.33)$$

where T_1 [K] is the liquid temperature in the tank, T_{in} [K] is the inlet temperature, c [J/(kg K)] is the specific heat capacity, V [m³] is the liquid volume, ρ [kg/m³] is the density, w [kg/s] is the mass flow (same out as in), K_P [W/%] is the gain of the power amplifier, u [%] is the control variable, $c\rho VT_1$ is (the temperature dependent) energy in the tank. It is assumed that the tank is isolated, that is, there is no heat transfer through the walls to the environment. To make the model a little more realistic, we will include a time delay τ [s] to represent inhomogeneous conditions in the tank. Let us for simplicity assume that the time delay is inversely proportional to the mass flow. Thus, the temperature T at the sensor is

$$T(t) = T_1 \left(t - \frac{K_\tau}{w} \right) = T_1 (t - \tau) \quad (10.34)$$

where τ is the time delay and K_τ is a constant. It can be shown that the

¹¹Well, it would be strange if not. After all, we will be analyzing the same model as used in the simulator.

transfer function from control signal u to process variable T is

$$T(s) = \frac{K}{\underbrace{T_t s + 1}_{H_u(s)}} e^{-\tau s} u(s) \quad (10.35)$$

where

$$\text{Gain } K = \frac{K_P}{cw} \quad (10.36)$$

$$\text{Time-constant } T_t = \frac{\rho V}{w} \quad (10.37)$$

$$\text{Time delay } \tau = \frac{K_\tau}{w} \quad (10.38)$$

This confirms the observations in the simulations shown in Figure 10.15: Reduced mass flow w implies *larger process gain*, and larger time-constant and larger time delay.

Heat exchangers and blending tanks in a process line where the production rate or mass flow varies, have similar dynamic properties as the tank in this example.

Control without gain scheduling (with fixed parameters)

Let us look at temperature control of the tank. The mass flow w varies. In which operating point should the controller be tuned if we want to be sure that the stability of the control system is not reduced when w varies? In general the stability of a control loop is reduced if the gain increases and/or if the time delay of the loop increases. (10.36) and (10.38) show how the gain and time delay depends on the mass flow w . According to (10.36) and (10.38) the PID controller should be tuned at minimal w . If we do the opposite, that is, tune the controller at the maximum w , the control system may actually become unstable if w decreases.

Let us see if a simulation confirms the above analysis. Figure 10.16 shows a temperature control system. The PID controller is in the example tuned at the maximum w value, which here is assumed 24 kg/min.¹² The PID parameters are

$$K_p = 7.8; T_i = 3.8 \text{ min}; T_d = 0.9 \text{ min} \quad (10.39)$$

¹² Actually, the controller was tuned with the Ziegler-Nichols' Ultimate Gain method. This method is however not described in this book. The Good Gain method could have been used in stead.

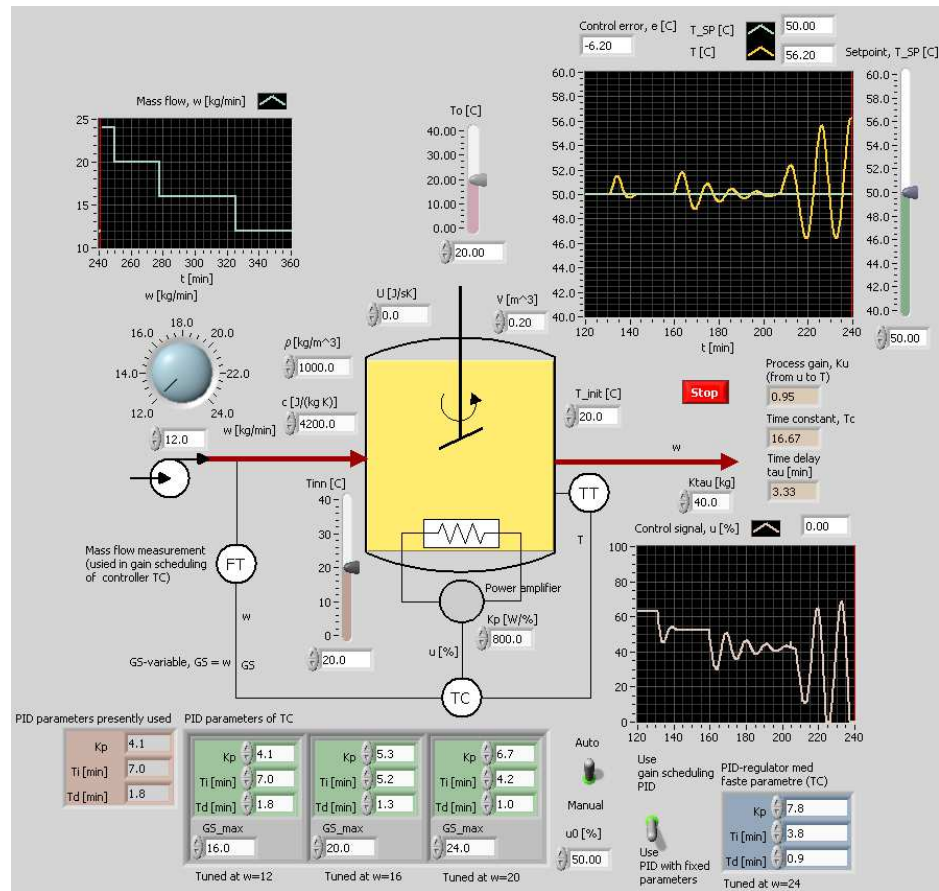


Figure 10.16: Example 10.4: Simulation of temperature control system with PID controller with fixed parameters tuned at maximum mass flow, which is $w = 24\text{kg/min}$

Figure 10.16 shows what happens at a stepwise reduction of w : The stability becomes worse, and the control system becomes *unstable* at the minimal w value, which is 12kg/min .

Instead of using the PID parameters tuned at maximum w value, we can tune the PID controller at minimum w value, which is 12 kg/min . The parameters are then

$$K_p = 4.1; T_i = 7.0\text{ min}; T_d = 1.8\text{ min} \quad (10.40)$$

The control system will now be stable for all w values, but the system behaves sluggish at large w values. (Responses for this case is however not shown here.)

Control with gain scheduling

Let us see if gain scheduling maintains the stability for varying mass flow w . The PID parameters will be adjusted as a function of a measurement of w since the process dynamics varies with w . Thus, w is the gain scheduling variable, GS :

$$GS = w \quad (10.41)$$

A gain schedule consisting of three PID parameter value sets will be used. The PID controller are tuned at the following GS or w values: 12, 16 and 20 kg/min. These three PID parameter sets are shown down to the left in Figure 10.16. The PID parameters are held piecewise constant in the GS intervals. In each interval, the PID parameters are held fixed for an increasing $GS = w$ value, cf. Figure 10.14.¹³ Figure 10.17 shows the response in the temperature for decreasing values of w . The simulation shows that the *stability of the control system is maintained even if w decreases*.

Finally, assume that you have decided not to use gain scheduling, but instead a PID controller with fixed parameter settings. What is the most critical operating point, at which the controller should be tuned? Is it at maximum flow or at minimum flow?¹⁴

[End of Example 10.4]

10.5.4 Adaptive controller

In an adaptive control system, see Figure 10.18, a mathematical model of the process to be controlled is continuously estimated from samples of the control signal (u) and the process measurement (y_m). The model is typically a transfer function model. Typically, the structure of the model is fixed. The model parameters are estimated continuously using e.g. the least squares method. From the estimated process model the parameters of a PID controller (or of some other control function) are continuously calculated so that the control system achieves specified performance in form of for example stability margins, poles, bandwidth, or minimum variance of the process output variable[10]. Adaptive controllers are commercially available, for example the ECA60 controller (ABB).

¹³The simulator uses the inbuilt gain schedule in LabVIEW's PID Control Toolkit.

¹⁴The answer is minimum flow, because at minimum flow the process gain is at maximum, and also the time-delay (transport delay) is at maximum.

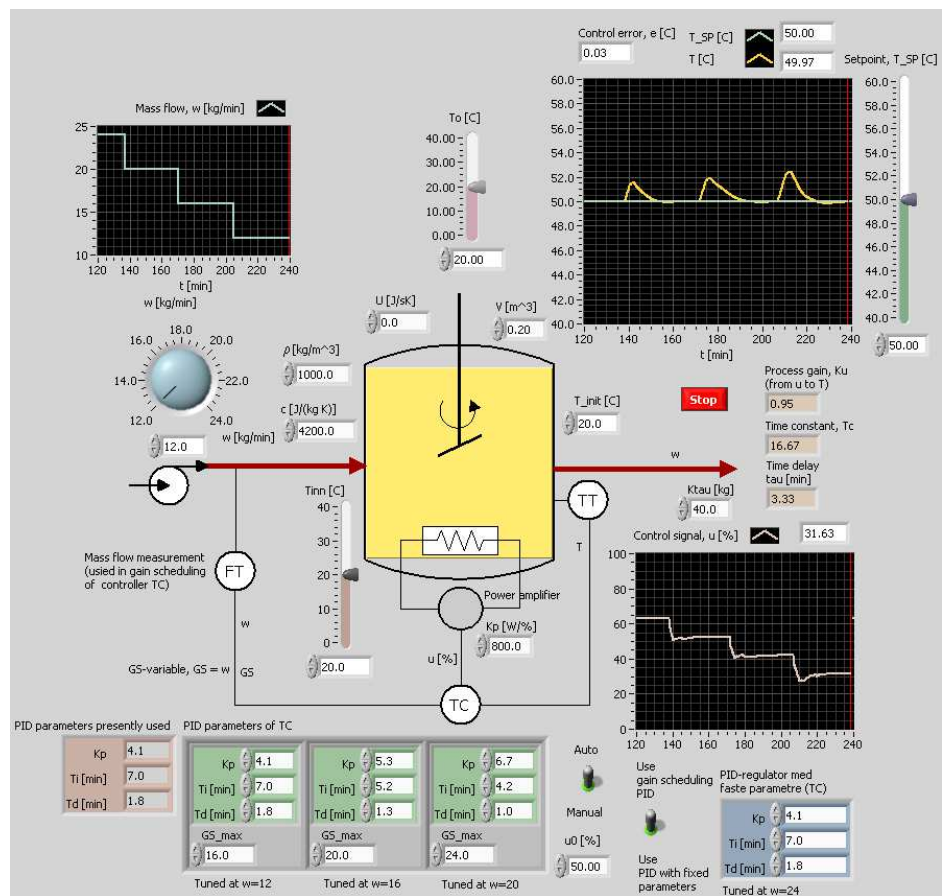


Figure 10.17: Example 10.4: Simulation of temperature control system with a gain schedule based PID controller

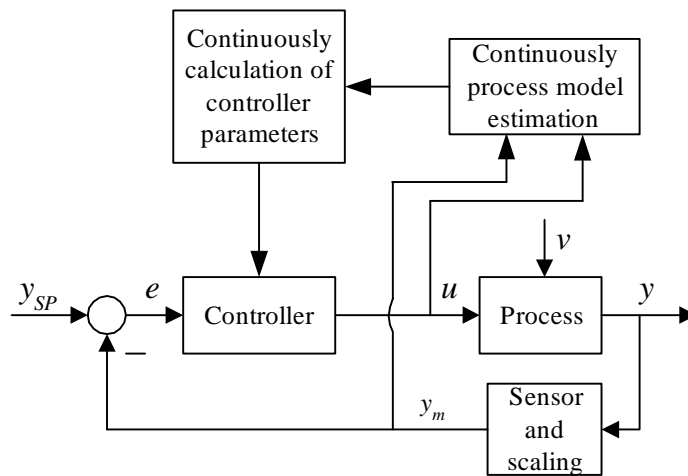


Figure 10.18: Adaptive control system

Chapter 11

Various control methods and control structures

This chapter describes a number of control methods and control structures based on the PID controller that are quite common in industry.

Also, sequential control is described. Sequential control is used to implement batch process control and automated mechanical operations. In a sequential control system PID control loops typically play a minor role, because the overall control task is not about continuous control. For example, PID control may be used to make the temperature of a batch reactor track a ramped temperature profile during the heating step of the control sequence.

11.1 Cascade control

11.1.1 The principle of cascade control

From earlier chapters we know that a control loop compensates for disturbances. If the controller has integral action the steady-state control error is zero despite a change in the disturbance. What more can we wish? We may want the compensation to happen faster, so that the control error goes to zero faster. This can be achieved by *cascade control*, see Figure 11.1.

In a cascade control system there is one or more control loops inside the *primary loop*, and the controllers are in cascade. There is usually one, but

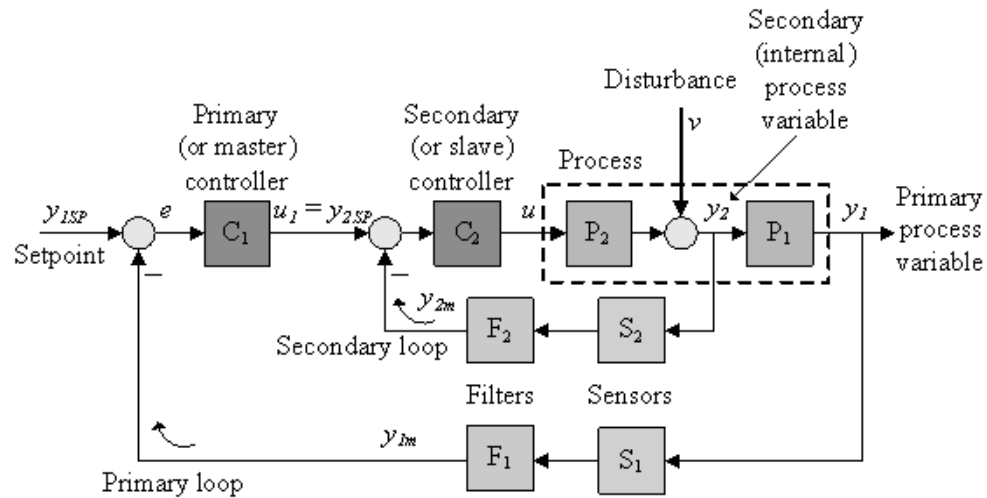


Figure 11.1: Cascade control system

there may be two and even three internal loops inside the primary loop. The (first) loop inside the primary loop is called the *secondary loop*, and the controller in this loop is called the *secondary controller* (or slave controller). The outer loop is called the *primary loop*, and the controller in this loop is called the *primary controller* (or master-controller). The control signal calculated by the primary controller is used as the setpoint of the secondary controller.

11.1.2 Benefits of cascade control

To repeat: One important reason for using cascade control is that it *gives faster (better) disturbance compensation*. In most applications the purpose of the secondary loop is to compensate quickly for the disturbance so that the response of the disturbance in the primary output variable is small. For this to happen the secondary loop must register the disturbance. This is done with the sensor S_2 in Figure 11.1.

One additional benefit of cascade control is that *the internal process variable (y_2) becomes more directly controllable, making it easier to tune the primary controller to obtain satisfactorily good overall stability and fastness of the whole control system*. In many applications process part 2 (P_2 in Figure 11.1) is the actuator (valve, pump, motor), or is closely related to the actuator. The secondary control loop can be regarded as a *new actuator* which can be more directly controlled or manipulated by the

primary controller. See Figure 11.2. The internal process variable y_2 is controlled or manipulated directly by the primary controller: The output of the primary controller, u_1 , demands a value of y_2 as it is the setpoint of y_2 , and the secondary control loop makes y_2 track this setpoint.

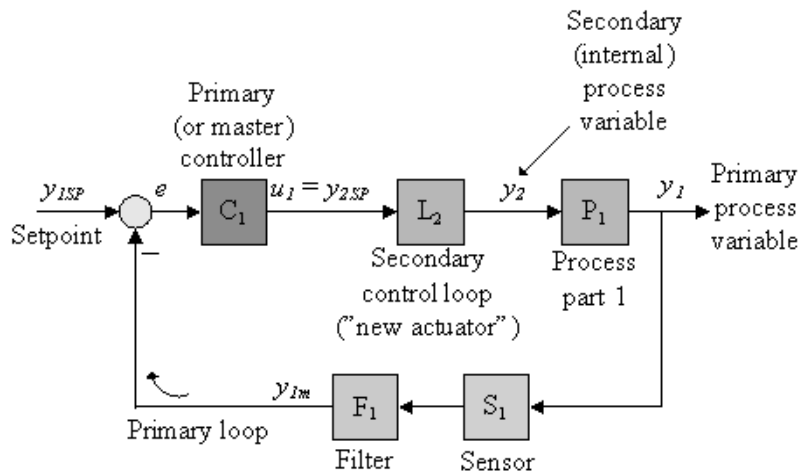


Figure 11.2:

Examples:

- The internal process variable (y_2) is valve flow. The secondary loop is then a flow control loop.
- The internal process variable (y_2) is pressure after (downstream) the valve. The secondary loop is then a pressure control loop.
- The internal process variable (y_2) is valve position (valve opening). The secondary loop is then a position control loop.

The generally improved control with cascade control can be explained by the increased information about the process – there is at least one more measurement. It is a general principle that the more information you have about the process to be controlled, the better it can be controlled. Note however, that there is still only one control variable to the process, but it is based on two or more measurements.

As explained above cascade control can give substantial disturbance compensation improvement. Cascade control can also give improved tracking of a varying setpoint, but only if the secondary loop has faster

dynamics than the process part P_2 itself, cf. Figure 11.1, so that the primary controller “sees” a faster process. But, if there is a time delay in P_2 , the secondary loop will not be faster than P_2 (this is demonstrated in Example 11.1), and then faster setpoint tracking can not be expected.

Are there any drawbacks of cascade control? Since cascade control requires at least two sensors a cascade control system is somewhat more expensive than a single loop control system. Except for cheap control equipment, commercial control equipment are typically prepared for cascade control, so no extra control hardware or software is required.

11.1.3 Controller selection and controller tuning

The primary controller is typically a PID controller or a PI controller. The secondary controller is typically a P controller or a PI controller. The derivative action is usually not needed to speed up the secondary loop since process part 2 anyway has faster dynamics than process part 1, so the secondary loop becomes fast enough. And in general the noise sensitive derivative term is a drawback.

In the secondary controller the P- and the D-term should not have reduced setpoint weights, cf. Section 7.4.2. Why?¹

How do you *tune* the controllers of a cascade control? You can follow this procedure:

- First the secondary controller is tuned, with the primary controller in manual mode.
- Then the primary controller is tuned, the secondary controller in automatic mode.

So, you start with the inner loop, and then move outwards.

The tuning of the controllers (finding good PID settings) can be made using any of the tuning methods described in Section ??.

¹Because attenuating or removing the time-varying setpoint (which is equal to the control signal produced by the primary controller) of the secondary loop will reduce the ability of the secondary loop to track these setpoint changes, causing slower tracking of the total control system.

11.1.4 Cascade control and state feedback

Cascade control is quite similar to *state feedback control*, which is a control principle where the controller generates the control signal as a function of measurements (or estimates) of *all the state variables* of the process to be controlled, see Figure 11.3. It can be shown that with state feedback, you

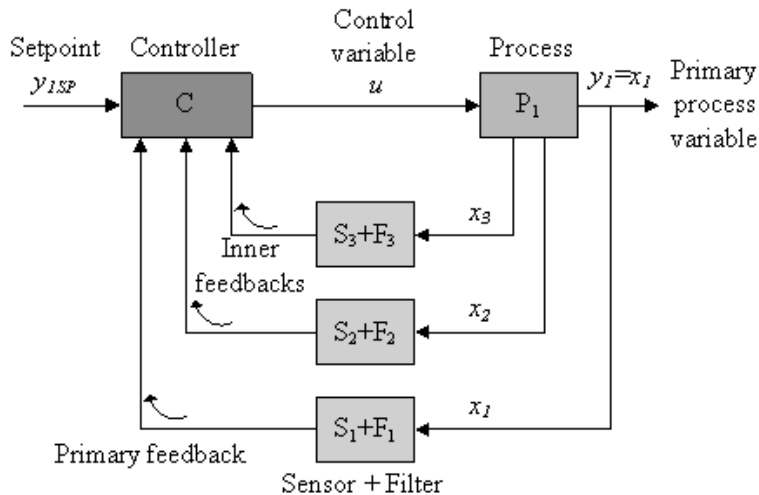


Figure 11.3: State feedback control

can design a control system with specified performance regarding stability and fastness for virtually any stable or unstable process. Hence, state feedback is a very powerful control principle. To design a state feedback controller you need the mathematical model of the process. There are several control methods – linear and nonlinear – that are based on state feedback control, see e.g. [2]. The most basic method is linear state feedback where the control variable is a linear combination of the state variables.

Example 11.1 *Temperature control using cascade control*

This example is about temperature control system of a liquid tank with continuous product flow heated with a heating liquid which flows continuously through a jacket around the tank. The tank can represent a reactor. In the simulator that is used in this example both the product and the heating liquid are water. A cascade control system and single-loop control system are simulated simultaneously (i.e. in parallel) so that the

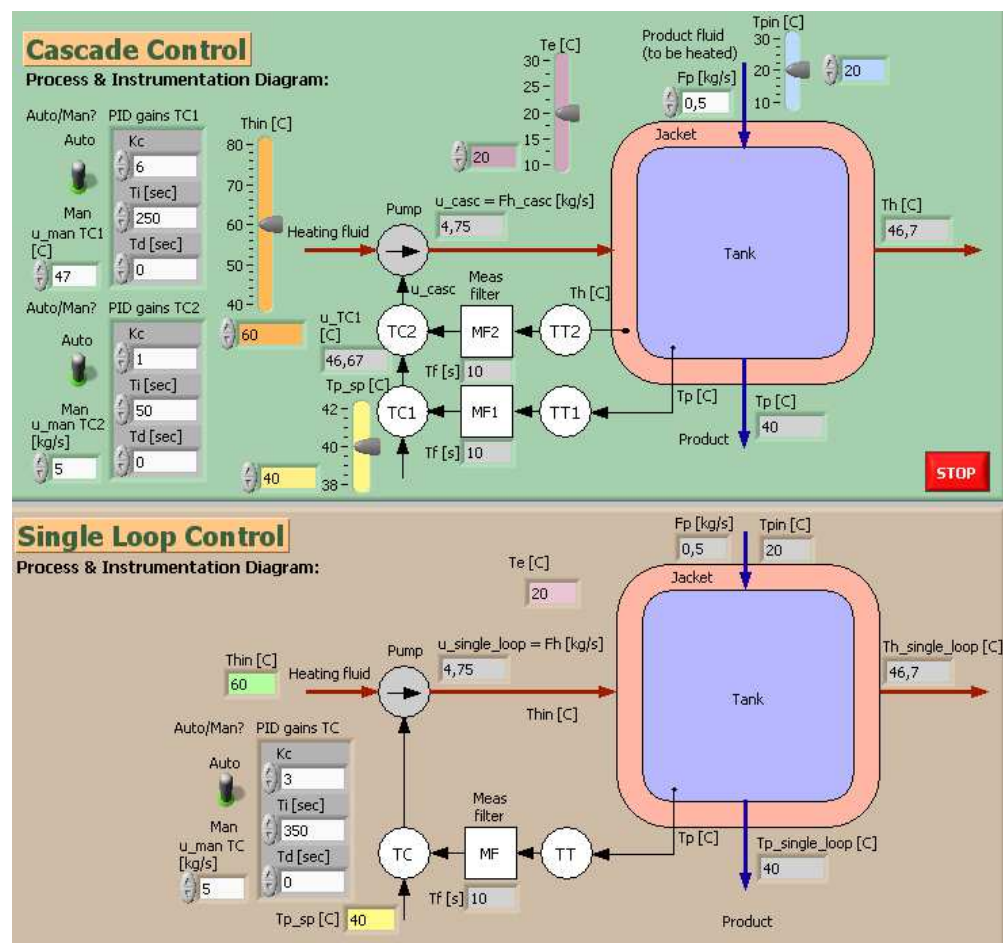


Figure 11.4: Example 11.1: Cascade and single-loop control systems of the product temperature of a tank

differences between these two control structures are easily demonstrated. The two control systems are as follows:

- **A cascade control system** with temperature control of both the tank and the jacket.
- **A single-loop control system** with temperature control of the tank only.

Figure 11.4 shows the P&IDs (Process & Instrumentation Diagrams) of the two control systems.

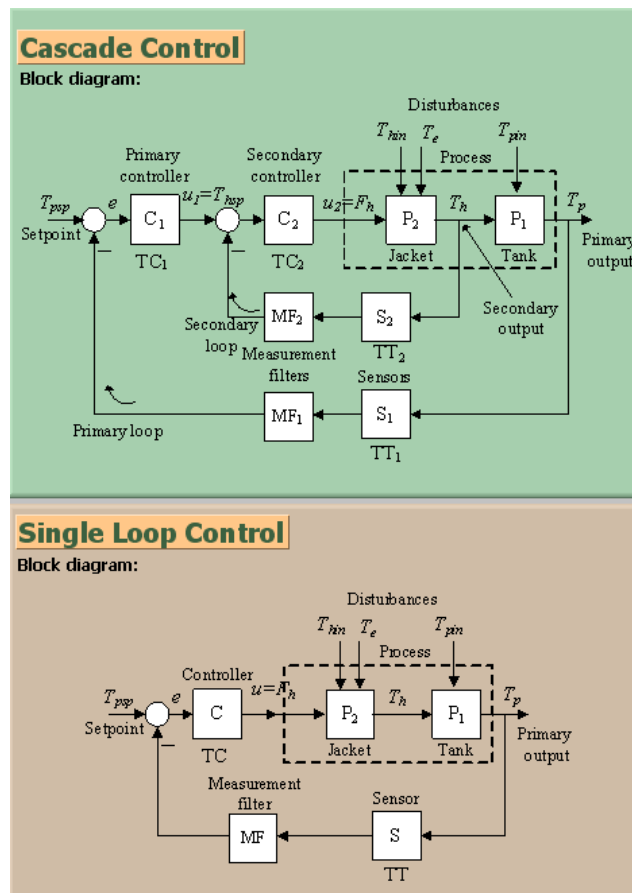


Figure 11.5: Example 11.1: Block diagrams of the cascade and single-loop control systems

Both control systems are excited by the same temperature setpoint and the same disturbances. The controller parameter settings are shown in Figure 11.4.

Figure 11.5 shows block diagrams of the cascade and single-loop control systems.

The mathematical model on which the simulator is based contains the following energy balances (assuming homogenous conditions in both the tank and the jacket):

$$\text{Tank: } c_p d_p V_p \dot{T}_p = [c_p F_p (T_{pin} - T_p) + U_{hp} (T_h - T_p)] \quad (11.1)$$

$$\text{Jacket: } c_h d_h V_h \dot{T}_h = [c_h F_h (T_{hin} - T_h) - U_{hp} (T_h - T_p) + U_{eh} (T_e - T_h)] \quad (11.2)$$

where T_p is product liquid temperature, T_h is heating liquid temperature, c_p and c_h are specific heat capacities, F_p and F_h are flows, $T_{p_{in}}$ and $T_{h_{in}}$ are inlet temperatures, V_p and V_h are volumes of tank and jacket respectively, T_e is environmental temperature, d_p and d_h are liquid densities, U_{hp} and U_{eh} are heat transfer coefficients between jacket and tank, and between jacket and environment (air) respectively. (All parameter values are available from the front panel of the simulator.) To account for unmodelled dynamics, a pure time delay is added to the measured temperatures T_p and T_h . The measurements of T_p and T_h are fed to measurement lowpass filters with time-constants T_f .

Figure 11.6 shows simulated responses due to the following excitations (applied at different points of time):

- **Disturbance change:** The inflow temperature $T_{h_{in}}$ was changed as a step from 60 to 55 °C.
- **Disturbance change:** The environmental (ambient) temperature T_e was changed as a step from 20 to 25 °C.
- **Disturbance change:** The product inlet temperature $T_{p_{in}}$ was changed as a step from 20 to 25 °C.
- **Setpoint change:** The setpoint T_{SP} of the product temperature was changed as a step from 40 to 40.5 °C.

From the simulations we observe the following:

1. The disturbance compensations regarding changes in the jacket inflow temperature $T_{h_{in}}$ and in the environmental (ambient) temperature T_e are substantially better with the cascade control system (see the responses in the upper plot in Figure 11.6). This is because the secondary process variable T_h (the jacket temperature) is directly influenced by these disturbances, and hence the secondary control loop can compensate for their changes.
2. The disturbance compensation regarding the change in the product inflow temperature $T_{p_{in}}$ is *not* substantially better with the cascade control system. This is because the secondary process variable T_h (the jacket temperature) is not directly influenced by this disturbances, and hence the secondary control loop can not compensate immediately for its change.
3. The setpoint tracking is somewhat better – but *not* substantially better – with the cascade control system.

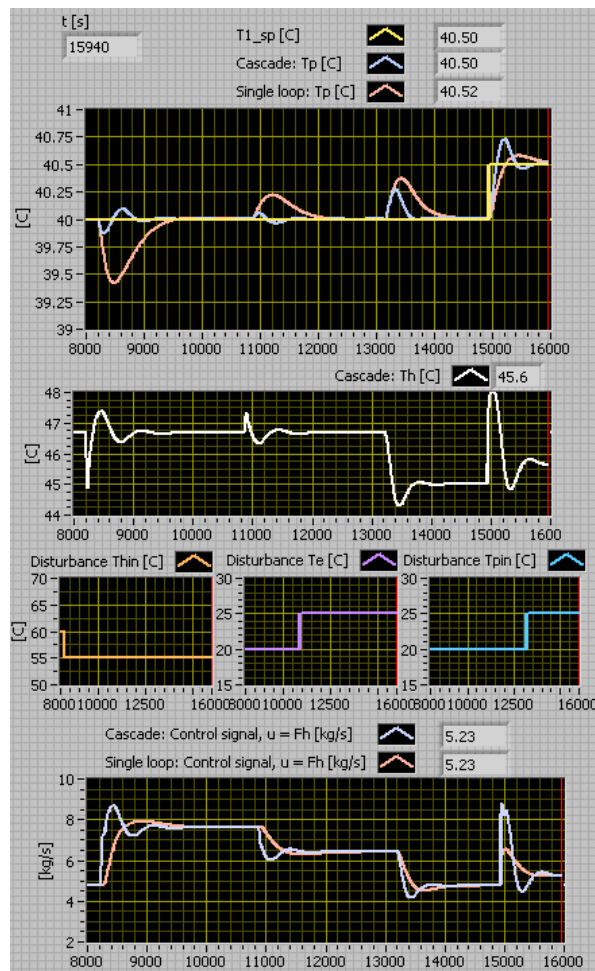


Figure 11.6: Example 11.1: Simulated responses of the cascade and single-loop control systems. The excitations are described in the text.

4. The control signal in the cascade control system works more aggressively than in the single-loop control system, which is due to the relatively quick secondary loop.

[End of Example 11.1]

Cascade control is frequently used in the industry. A few examples are described in the following.

Example 11.2 *Cascade control of the level in wood-chip tank*

Level control of a wood-chip tank has been a frequent example in this book. In the real level control system² cascade control is used, although not described in the previous examples. The primary loop performs level control. The secondary loop is a control loop for the mass flow on the conveyor belt, see Figure 11.7. The mass flow is measured by a flow sensor (which actually is based on a weight measurement of the belt with chip between two rollers). The purpose of the secondary loop is to give a quick compensation for disturbances, d , in the chip flow due to variations in the chip density since the production is switched between spruce, pine and eucalyptus. (In Figure 11.7 the disturbance d actually has a negative value, hence, it represents a reduction of the inflow to the belt.)

In addition to this compensation the secondary flow control loop is a “new feed screw” with flow that is more directly controllable by the level controller.

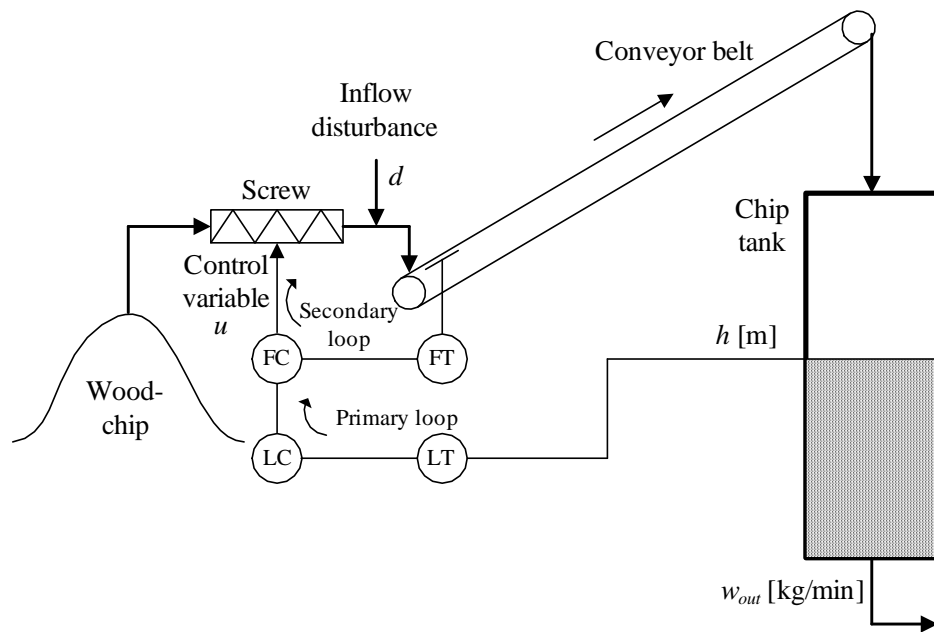


Figure 11.7: Example 11.2: Level control system of a wood-chip tank where the primary loop performs level control, and the secondary loop performs mass flow control. (FT = Flow Transmitter. FC = Flow Controller. LT = Level Transmitter. LC = Level Controller.)

[End of Example 11.2]

²at Södra Cell Tofte in Norway

Example 11.3 *Cascade control of a heat exchanger*

Figure 11.8 shows a temperature control system of a heat exchanger. The control variable controls the opening of the hot water valve. The primary loop controls the product temperature. The secondary loop controls the heat flow to compensate for flow variations (disturbances). The valve with flow control system can be regarded a new valve with an approximate proportional relation between the control variable and the heat flow.

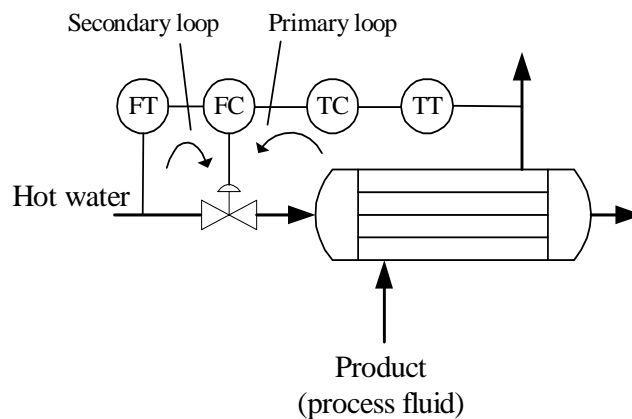


Figure 11.8: Example 11.3: Cascade control of the product temperature of a heat exchanger. (TC = Temperature Controller. TT = Temperature Transmitter. FC = Flow Controller. FT = Flow Transmitter.)

[End of Example 11.3]

There are many other examples of cascade control, e.g.:

- **DC-motor:**

- Primary loop: Speed control based on measurement of the rotational speed using a tachometer as speed sensor.
- Secondary loop: Control of armature current which compensates for nonlinearities of the motor, so that the speed controller can manipulate the current more directly. This may improve the speed control of the motor.

- **Hydraulic motor:**

- Primary loop: Positional control of the cylinder

- Secondary loop: Control of the servo valve position (the servo valve controls the direction of oil flow into the cylinder), so that the primary position controller can manipulate the servo valve position more directly. This may improve the position control of the cylinder.
- **Control valve:**
 - The primary loop: Flow control of the liquid or the gas through the valve.
 - Secondary loop: Positional control of the valve stem, so that the flow controller can manipulate the valve position more directly. This may improve the flow control. Such an internal positional control system is called *positioner*.

11.2 Ratio control

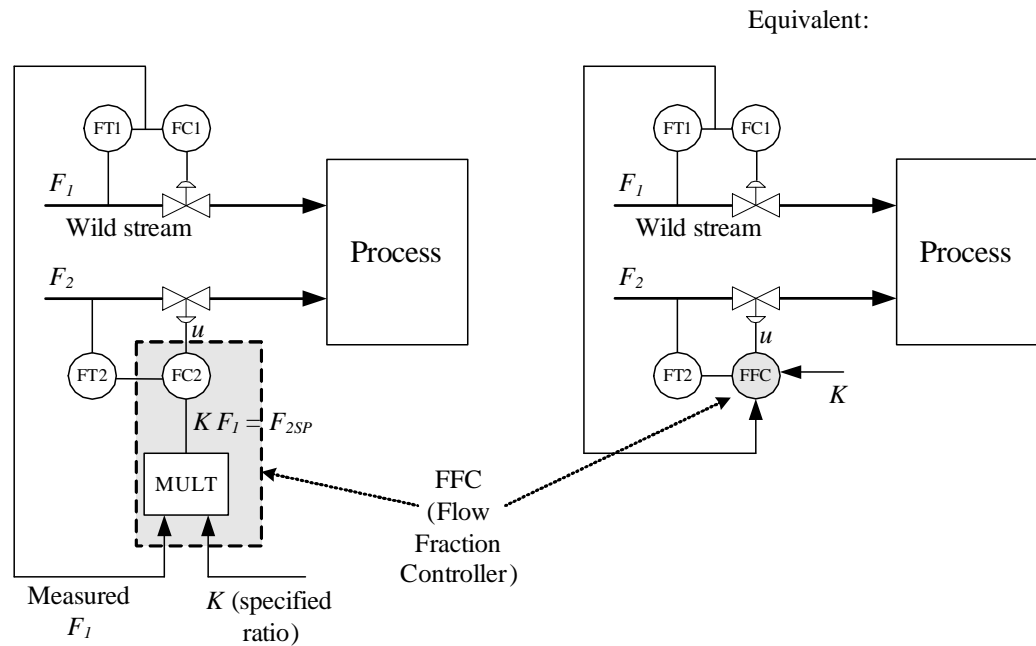


Figure 11.9: Ratio control

The purpose of *ratio control* is to control a mass flow, say F_2 , so that the ratio between this flow and another flow, say F_1 , is

$$F_2 = KF_1 \quad (11.3)$$

where K is a specified ratio which may have been calculated as an optimal ratio from a process model. One example is the calculation of the ratio between oil inflow and air inflow to a burner to obtain optimal operating condition for the burner. Another example is the nitric acid factory where ammonia and air must be fed to the reactor in a given ratio.

Figure 11.9 shows the most common structure of ratio control. As shown with the left diagram in the figure, the setpoint of the flow F_2 is calculated as K times the measured value of F_1 , which is denoted the “wild stream” (this name is used, even if this stream is controlled, as shown in Figure 11.9). The diagram to the right in the figure shows a compact but equivalent representation of ratio control with the symbol FFC (Flow Fraction Controller).

11.3 Split-range control

In *split-range control* one controller controls two actuators in different ranges of the control signal span, which here is assumed to be 0 – 100%. See Figure 11.10. Figure 11.11 shows an example of split-range

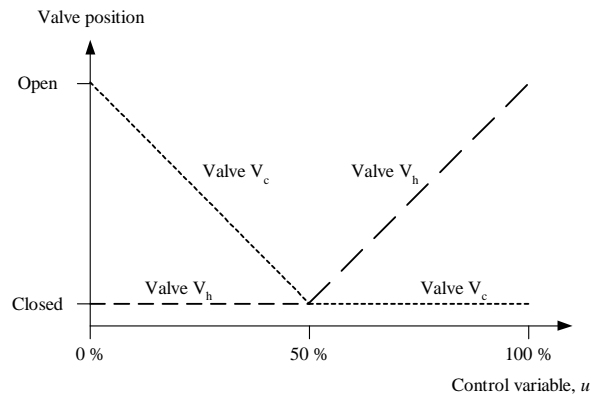


Figure 11.10: Split-range control of two valves

temperature control of a thermal process. Two valves are controlled – one for cooling and one for heating, as in a reactor. The temperature controller controls the cold water valve for control signals in the range 0 – 50 %, and it controls the hot water valve for control signals in the range 50 – 100 %, cf. Figure 11.10.

In Figure 11.10 it is indicated that one of the valves are open while the other is active. However in certain applications one valve can still be open

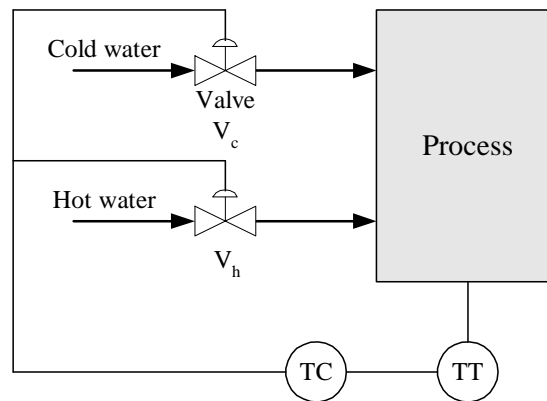


Figure 11.11: Split-range temperature control using two control valves

while the other is active, see Figure 11.12. One application is pressure

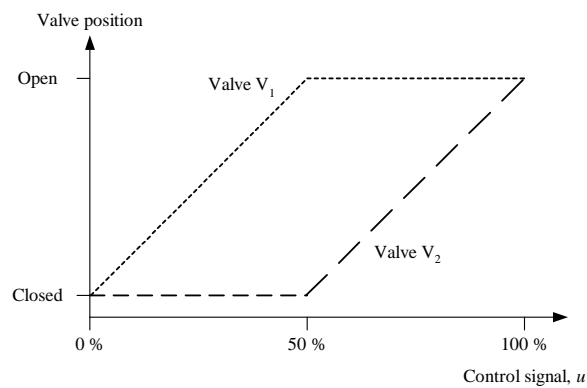


Figure 11.12: In split-range control one valve can be active while another valve is open simultaneously.

control of a process: When the pressure drop compensation is small (as when the process load is small), valve V_1 is *active* and valve V_2 is *closed*. And when the pressure drop compensation is large (as when the process load is large), valve V_1 is *open* and valve V_2 is *still active*.

11.4 Flow smoothing with sluggish level control

11.4.1 The control task

Figure 11.13 shows a flow smoothing system consisting of a tank with a level control system. One important application is in the oil & gas

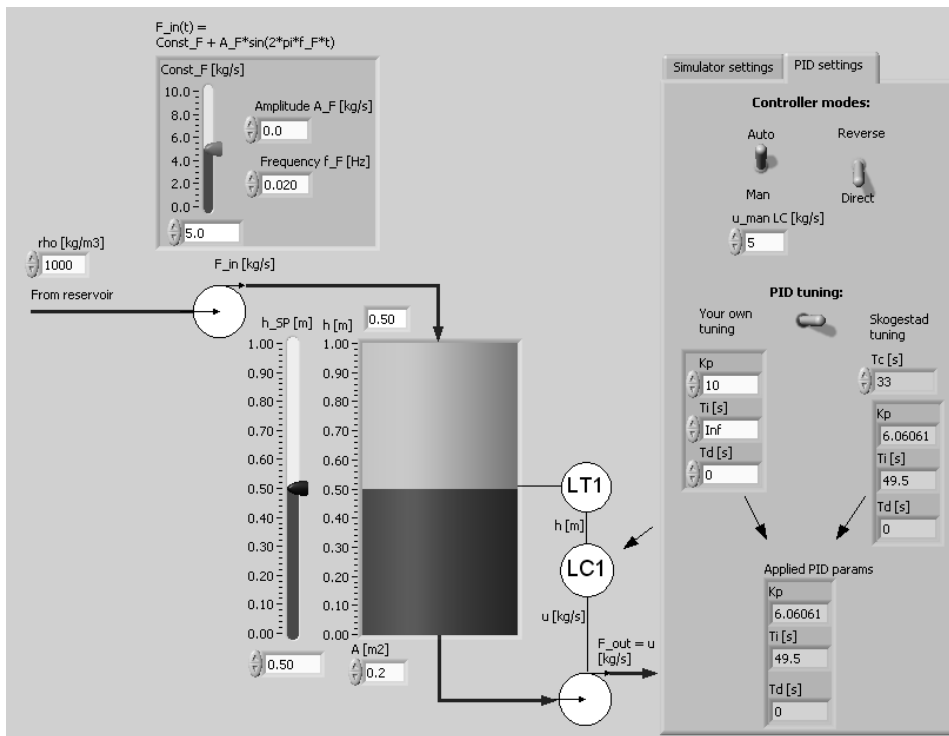


Figure 11.13: A flow smoothing system consisting of a tank with a level control system

production where separators are used to separate oil and gas and water from a three-phase oil/water/gas flow coming from reservoirs.³ Separators are level-controlled to maintain the mass balance in the separator. It is important that *slugs*, which are sudden changes in the inflow to the separator, are attenuated through the separator, otherwise the slug will propagate through the subsequent production line. This attenuation will not take place if the level control works too fast, because in a fast level control system, the outflow is nearly the same as the input! So, slugs require sluggish (slow) level control. On the other hand, the slugs must

³The residence in the separator causes the separation to take place due to the different densities of oil, water and gas.

not cause too large level variations inside the separator, otherwise level alarms will be triggered.

This kind of control is sometimes denoted *averaging control* as the inflow is averaged through the *buffer* tank, to give smooth outflow.

Example 11.4 *Control of a water and oil (and gas) separation system*

This example is a realistic example from the oil and gas production field in Norway (the Vigdis plant).

Figure 11.14 shows the front panel of a simulator of a water, oil and gas separator system. The main flow rate is given by the inflow from the reservoir. So, in this system there is no flow control loop. In stead, the flow is given by the flow coming from the reservoir. The level controllers must maintain the mass balances of the separators. (Mass of gas is maintained using pressure control, but the pressure control system is not shown in the figure.) In the first stage separator there is a level control systems for the oil and another level control systems for the water. In the second stage separator there is a level control system for the oil (it is assumed that all the water has been removed in the first stage separator).

[End of Example 11.4]

11.4.2 Controller tuning

Figure 11.15 shows a block diagram of the tank with level control system.

So, *the control task is to have as sluggish level control as possible but not too sluggish to avoid level alarm triggering*. If we specify that the level is at setpoint at steady state, the level controller must have integral action.

How to tune the level controller for such flow smoothing control? The Good Gain method can not be used in the original way, cf. Sec. 10.2, because that method aims at giving maximum fastness in the control loop, and maximum fastness is not what you want here. In stead, the controller can be tuned with Skogestad's model-based tuning method, cf. Section 10.3, in combination with a trial-and-error on a *simulator* of the control system.⁴ Assuming Skogestad's tuning method, we will *set the control*

⁴The simulator of the whole control system can be implemented in for example block-diagram based simulators as LabVIEW, Simulink, or Scicos (the latter is freeware), see

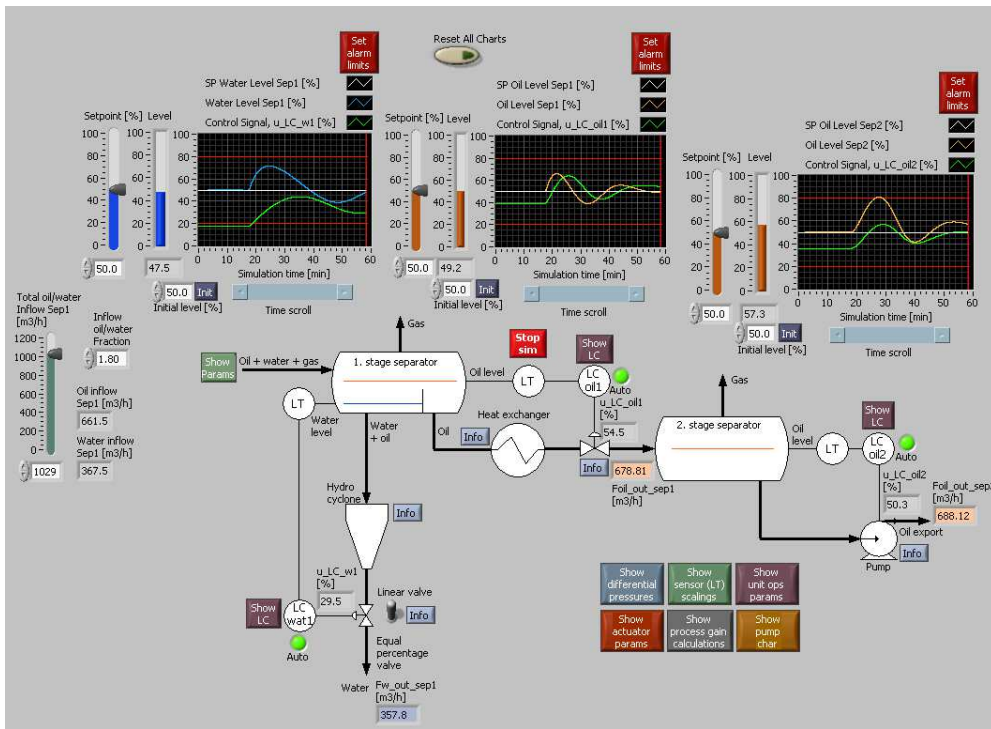


Figure 11.14: Example 11.4: Control structure for a water, oil and gas separator system

system response time, T_C , as low as possible because the slug must not cause too high or too low level. The smallest allowable T_C can be found from simulations. In Skogestad's method we need the transfer function from control variable (pump outflow) to measurement (level). Let us derive the transfer function from the differential equation expressing the mass balance of the liquid of the tank. Mass balance gives

$$\rho A \dot{h}(t) = F_{in}(t) - \overbrace{F_{out}(t)}^{u(t)} \quad (11.4)$$

Taking the Laplace transform of this differential equation gives

$$\rho A s h(s) = F_{in}(s) - u(s) \quad (11.5)$$

Solving for process output variable (level):

$$h(s) = \frac{1}{\rho A s} F_{in}(s) + \underbrace{\left(\frac{-1}{\rho A s} \right)}_{H(s)} u(s) \quad (11.6)$$

<http://techtch.no> for more information. A ready-to-run configurable simulator of a flow smoothing system is also available in TechVid at <http://techtch.no>.

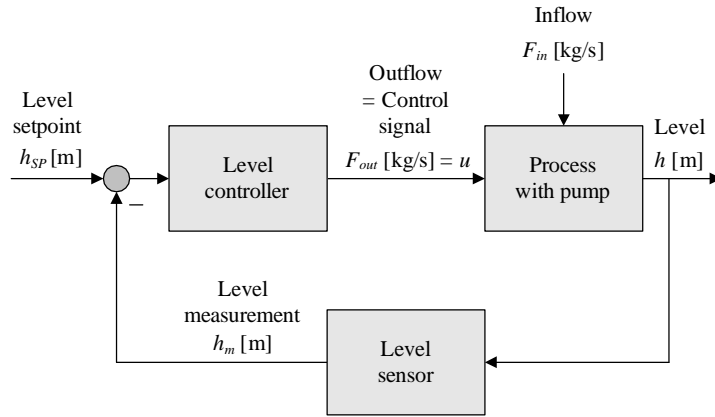


Figure 11.15: Flow smoothing control: Block diagram of the tank with level control system.

Process transfer function from control variable (outflow) to process output variable (level) is

$$H(s) = -\frac{1}{\rho A s} = \frac{K}{s} \text{ (integrator)} \quad (11.7)$$

Hence, the integrator gain of the transfer function is

$$K = -\frac{1}{\rho A} \quad (11.8)$$

Skogestad's tuning of a PI(D) controller for an integrator process is shown in Table 10.1 (with $c = 2$):

$$K_p = \frac{1}{K T_c} = -\frac{\rho A}{T_c} \quad (11.9)$$

$$T_i = 2T_c \quad (11.10)$$

$$T_d = 0 \quad (11.11)$$

where T_C is the specified response-time of the control system.

The integral term of the PI controller will ensure that level is at setpoint at steady state.

Note that the level controller must have *direct action* (not reverse action) because the process gain is negative, cf. Section 7.3.3.

Example 11.5 *Flow smoothing control*

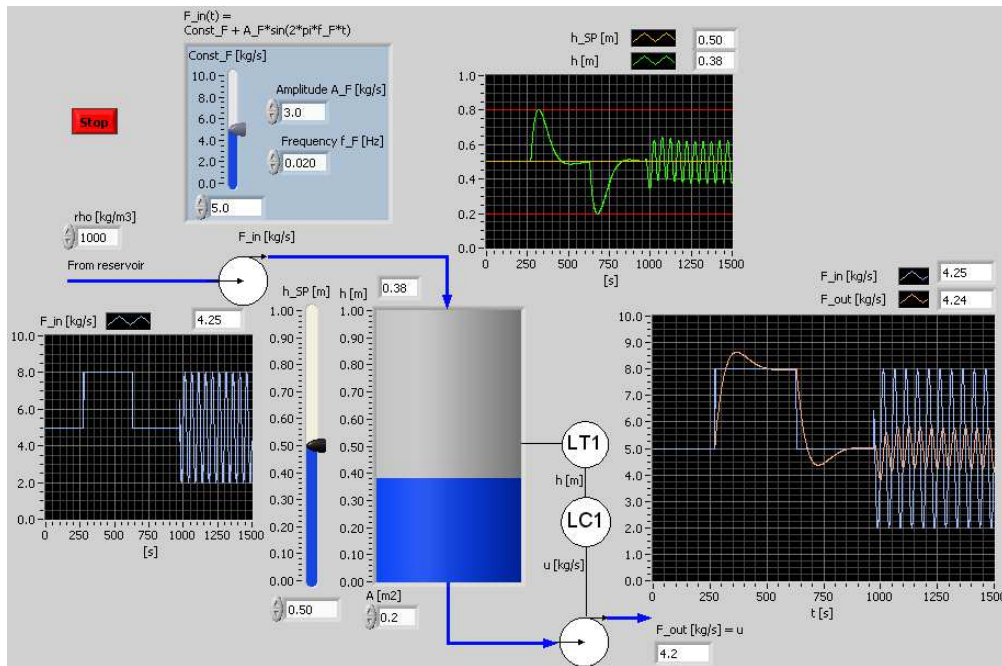


Figure 11.16: Example 11.5: Buffer tank

Figure 11.16 shows the front panel of a simulator of a buffer tank with level control system. The controller is tuned with Skogestad's method as described above. The specification are as follows:

1. Liquid level change less than 0.3 m at slug amplitude 3 kg/s.
2. Maximum flow smoothing, assuming sinusoidal flow with amplitude 3 kg/s and frequency 0.02 Hz.

By adjusting T_C by trial-and-error on the simulator it was found that specification number 1 was satisfied with

$$T_C = 31 \text{ s} \quad (11.12)$$

See the first part of the simulation shown in Figure 11.16. The PI settings corresponding to this value of T_C are

$$K_p = \frac{1}{KT_c} = -\frac{\rho A}{T_c} = -\frac{1000 \text{ kg/m}^3 \cdot 0.2 \text{ m}^2}{31 \text{ s}} = 6.45 \text{ kg}/(\text{m} \cdot \text{s}) \quad (11.13)$$

$$T_i = 2T_c = 2 \cdot 31 \text{ s} = 62 \text{ s} \quad (11.14)$$

Concerning specification number 2, the simulations show that the sinusoidal flow (amplitude 3 kg/s, frequency 0.02 Hz) has been attenuated by a factor approximately equal to 25%. Hence, “flow gain” is 25%. If we try to obtain even better flow smoothing by increasing T_C further, specification number 1 will not be satisfied as the level response to the given slug will be larger than 0.3 m.

[End of Example 11.5]

11.5 Plantwide control

In the process industry products are created after materials being processed in a number of vessels in series, which are typically unit processes as evaporators, blending or heated tanks, buffer tanks, reactors, distillation columns, separators, absorbers, etc. Typical basic control requirements of such a production line are as follows:

- **The product flow must be controlled (to follow a setpoint).**
This is typically done by implementing flow control of one key component or feed.
- **The product quality must be controlled** – more or less directly. Ideally, the product quality is analyzed or measured in real time, and this measurement is used to adjust feeds, supplied heat etc. so that the product quality is controlled to follow its setpoint using feedback. In practice, however, such real time or online measurements are typically not implemented since they rely on time consuming laboratory analysis. In stead, the product quality is controlled indirectly by controlling the flow of the feeds needed to produce the final product, typically using ratio control.
- **The mass balance of each process vessel (tank, reactor, etc.) must be maintained** – otherwise it may run full or empty. The mass balance is maintained with level control systems for liquids, and with pressure control systems for gases. In vessels containing gas there may still not be any pressure control. It depends on the process if it necessary to control the process.

It is crucial to select the correct reverse or direct action for each of the controllers (otherwise the control loop becomes unstable), cf. Section 7.3.3. The general rules are as follows⁵:

⁵It is assumed that increased control signal to the actuator increases flow through the actuator.

- Level/gas controllers that are *upstream* relative to the flow control loop (i.e. are placed before the loop) implementing product flow control of the plant (see above) shall be set into *reverse action mode*.
 - Level/gas controllers that are *downstream* relative to the flow control loop (i.e. are placed after the loop) implementing product flow control of the plant (see above) shall be set into *direct action mode*.
- **The temperature in some of the process lines or vessels must be controlled.**

Figure 11.17 shows a basic, principal example where the above control requirements are implemented.

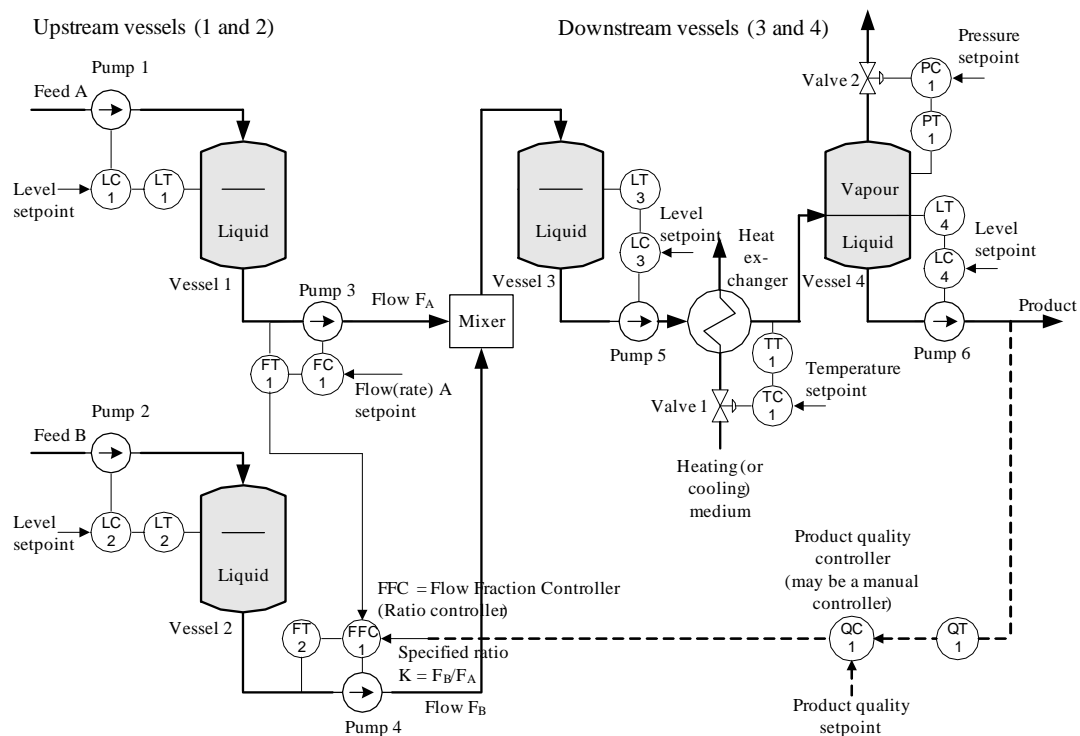


Figure 11.17: Process plant with control loops

Comments to the control system shown in Figure 11.17:

- Two flows, F_A and F_B , are mixed. In general, the mix of two feeds can be fed to a subsequent reactor. It is assumed that F_A contains the key component of the product, and that F_A defines the production rate. Ratio control is used to control F_B so that the ratio between the flows is as specified:

$$K = \frac{F_B}{F_A} \quad (11.15)$$

- The mass balance of upstream vessels 1 and 2 (upstream relative to the point of production flow control in the process line) is controlled by manipulating *inflows* while the mass balance of the downstreams vessels 3 and 4 tank are controlled by manipulating the *outflows*.
- In vessel 4 both the liquid level and the vapour pressure are controlled.
- The mass balances of liquids are maintained using level control. The mass balance of vapour is maintained using pressure control.
- The product quality may be controlled by adjusting the ratio K between the flows into the mixer. Ideally, this quality controller is operating in automatic mode based on online (continuous) quality measurement of the product. However, in many cases (as in the nitric acid production plant described in Example 11.6) the product quality is measured off-line using laboratory analysis which gives quality measures only a few times per day. In these cases, the quality controller is actually in the form of an operator adjusting the ratio K .
- The temperature of the inflow to vessel 4 is controlled.
- It is crucial to select correctly between Reverse action mode and Direct action mode in the controllers, cf. Section 7.3.3). Assume that for each of the pumps and the valves that increased control signal increase pump speeds and increase valve openings. Furthermore, assume that heating is needed by the heat exchanger, the control valve manipulates the amount of heating medium. Then, the controller mode of each of the controllers are as follows:
 - LC1: Reverse
 - LC2: Reverse
 - LC3: Direct
 - LC4: Direct
 - PC1: Reverse
 - TC1: Reverse

– All flow controllers: Reverse

Is this correct?⁶

A specific, industrial example is given in the following.

Example 11.6 *Control structures of a nitric acid plant*

The example is about production of nitric acid, which is used in e.g. the production of fertilizers.⁷ The description of the process is based on [6], however I have simplified the process somewhat because too many details may obscure the basic principles. (One simplification is that the bleacher, which is used to clean the raw, brown-coloured nitric acid which is the output from the absorber, is not included. Another simplification is that the condenser between compressor K2 and absorber A1 is not included.) The control structure is shown in the figure is based on information given by Yara company in Porsgrunn, Norway.

Figure 11.18 shows a Process and Instrumentation Diagram (P & ID) of the (simplified) nitric acid plant.

Description of the nitric acid plant

(Control structures are described below.)

The feeds (input flows) to the production are

- Ammonia
- Air
- Process water

The product (output flow) is

- Nitric acid

⁶No! PC1: Direct.

⁷There are many Nitric Acid production plants around the world. One of them is at the Yara industrial plant area in Porsgrunn, Norway.

Below are descriptions of the components and flows related to these components:

- **Evaporator E1:** Ammonia liquid (NH_3) is evaporated using water (cold!) as heat supply.
- **Compressor K1:** Primary air is compressed.
- **Mixer M1:** Ammonia gas and air are mixed.
- **Reactor R1** (exothermal): Ammonia and oxygen (in the primary air) reacts with a platinum-rhodium catalyst to produce nitrogen monoxide (NO) and water (H_2O) which are passed to condenser C1.
- **Condenser C1:** The water and some of the nitrogen dioxide (gases) are condensed into acid liquid.
- **Condenser C1:** The gaseous output is mixed with secondary air from the bleacher (the bleacher is omitted in this simplified plant), causing oxydation. The product of the oxydation is nitrogen dioxide (NO_2) and water (H_2O).
- **Compressor K2:** Nitrogen dioxide (NO_2) and water (H_2O) is compressed in compressor K2.
- **Condenser C1:** The acid liquid outflow is pumped by pump P1 to a proper tray in the absorber.
- **Absorber A1:** Process water is added at the top. This water and the acid liquid (from condenser C1) flow downwards to meet gaseous nitrogen dioxide (from condenser C2) with a chemical reaction taking place. The product of this reaction is raw nitric acid (HNO_3) liquid collected at the bottom of the absorber, to be passed to bleacher (the bleacher is omitted here), and nitrous gases exiting at the top.
- **Heater H1:** The nitrous gases are heated by the hot product out of reactor R1, before being passed to, and driving, turbine/generator TG2 which produces electrical power that is exported.
- **Steam drum D1:** The reaction taking place in reactor R1 is exothermal. Some of the produced heat is used to heat water, and the resulting high-pressure steam drives turbine/generator TG1 which produces electrical power that is exported.
- **Shaft 1:** The compressors and the turbines/generators have one common shaft. The turbines transfers chemical energy to motional

(kinetic) energy. The generators produce electrical power that is exported. The rotational speed of the common shaft determines the production rate of the plant.

Description of the control structure of the plant

- **Production rate** is controlled with rotational speed control of the common turbine/generator shaft. The mass flow of the air, which is a feed to the process, is proportional to the speed (indicated with the MULT block with a gain k in the upper part of the diagram). The stoichiometric balances are maintained with a ratio of 10 % between the mass flow of ammonia to mass flow of air. Therefore, the ammonia gas flow (feed) is controlled with ratio control based on the rotational speed.

Also the *process water flow* into the absorber is a feed to the process. This flow is controlled with flow control with the flow setpoint adjusted manually according to a laboratory analysis of the nitric acid outflow (from the absorber) accomplished twice a day. The QT (Quality Transmitter) represents this analysis.

- **Mass balances** of the various vessels are maintained as follows:
 - *Evaporator E1*: Ammonia liquid level is controlled using inlet valve V1 as actuator. Since this level control loop is upstream relative to the production flow control loop, the level controller LC1 shall be set in reverse action mode.
 - *Evaporator E1*: Ammonia gas pressure is controlled using water valve V2 as actuator.
 - *Reactor R1*: The contents of the reactor is gaseous only. There is however no control of gas pressure.
 - *Condenser C1*: Acid liquid level is controlled using pump P1 as actuator. Since this level control loop is downstream relative to the production flow control loop, the level controller LC2 shall be set into direct action mode.
 - *Condenser C1*: Gas pressure is not controlled.
 - *Absorber A1*: Raw nitric acid liquid level is controlled using valve V4 as actuator. Since this level control loop is downstream relative to the production flow control loop, the level controller LC3 shall be set into direct action mode.
 - *Absorber A1*: Nitrous gas pressure at the “top” is not controlled.
- **Temperature control**: None.

[End of Example 11.5]

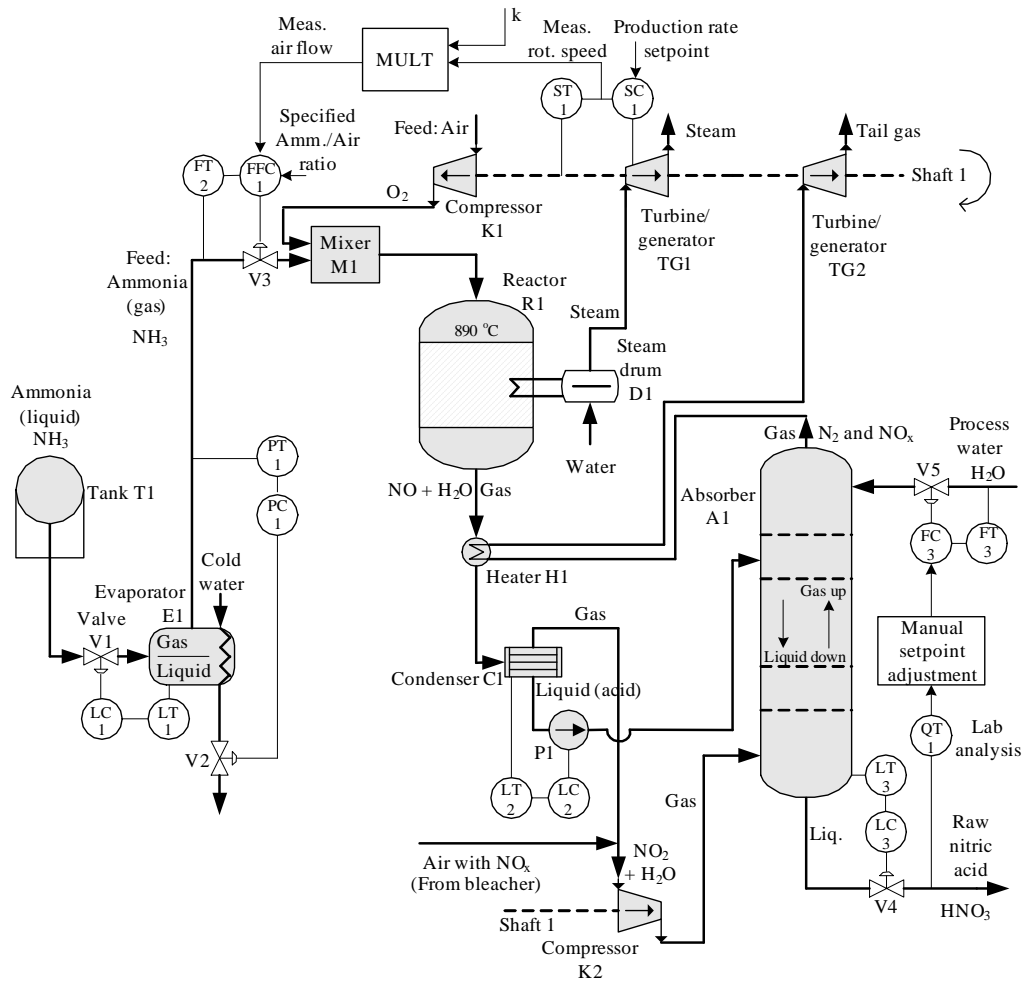


Figure 11.18: Example 11.6: Process and Instrumentation Diagram (P & ID) of simplified nitric acid plant

Chapter 12

Sequential control

Sequential control is used to implement e.g. chemical batch process control and automated mechanical operations.

A sequential control procedure can be represented graphically by

- **Sequential function chart** (SFC), or
- **State diagram.**

SFCs have the following elements:

- **Steps** with a number of associated actions to be executed when the step is active. A step is either active or passive. One particular step is defined as the initial step. It can be symbolized with e.g. a double-lined box, while ordinary steps are represented with single-lined boxes.
- **Actions** are control actions made by the control device (typically a PLC or a PC), e.g. opening a valve, setting a PID controller into automatic mode, starting a motor, lighting a lamp on, etc. The action box indicates the changes of control actions from previous steps (control actions which are not changed, does not have to be listed in the action box).
- **Transitions** from one active step to another taking place when the transition condition is satisfied. Transition conditions are in the form of logical expressions – simple or complicated – having value either

TRUE or FALSE. Here is one example: T_Step1_Step2: Level > 0.9 m.

A transition may take place *unconditionally*, that is, it takes place automatically after the actions of the presently active state have been accomplished. The transition condition of an unconditional transition has permanent value TRUE, and it may be expressed for example as T_Step2_Step3: TRUE.

Figure 12.1 shows how these basic elements (step, action, transition) appear in a Sequential function chart. The := symbol is the assignment operator. The == symbol is the equality check operator.

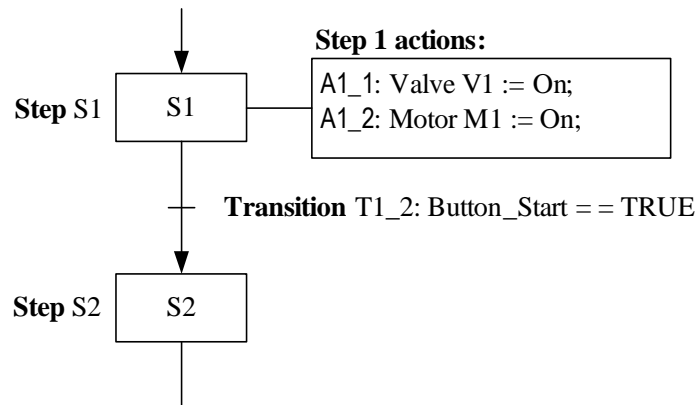


Figure 12.1: Elements of a Sequential Function Chart (SFC): Step, action, and transition.

Sequential function charts may also contain *branches* defining simultaneous or alternative parts of the chart to be executed.

State diagrams can be drawn similar to Sequential function charts. In State diagram the term *state* is used in stead of *step*. The control system represented with a state diagram is denoted a *state machine*.¹

SFC is standardized in the IEC 61131-3 standard about PLC programming², and is available as a programming tool in most PLC

¹The state machine is a very useful concept in programming, independently of the programming tool you are using. You can structure a programming task that contains several alternative paths dependent on certain conditions with a state diagram. After the structure has been made, the implementation with programming code is more or less straightforward.

²PLC = Programmable Logic Controller

systems, e.g. Mitsubishi PLCs, Simatic PLCs, etc. State diagrams are supported in e.g. the Statchart Module of LabVIEW, the Stateflow Toolbox of Matlab/Simulink, and the Graph7 programming tool of Simatic PLCs.

Example 12.1 *Sequential control of a batch process*

Figure 12.2 shows a simple batch process which is to be controlled by sequential control.³ The tank is filled with water up to a certain level. The water is then heated using temperature control (with a PID controller) for a specific time defined in timer, and stirred⁴. Finally the heated water is discharged from the tank. Then the batch can be restarted, or ended.

The *control signals* are

- u_valve_in (boolean, i.e. having value TRUE or FALSE, or ON or OFF)
- u_valve_out (boolean)
- u_motor (boolean)
- u_heat (continuous having any value between 0% and 100%)

The *measurements* are

- Temperature T of the water in the tank
- Level h of the water in the tank

Figure 12.3 shows a Sequential Function Chart defining the control function.

[End of Example 12.1]

³You can find a simulator of this system at <http://techteach.no/simview>.

⁴The motor is assumed to ensure homogenous thermal conditions in the water in the tank.

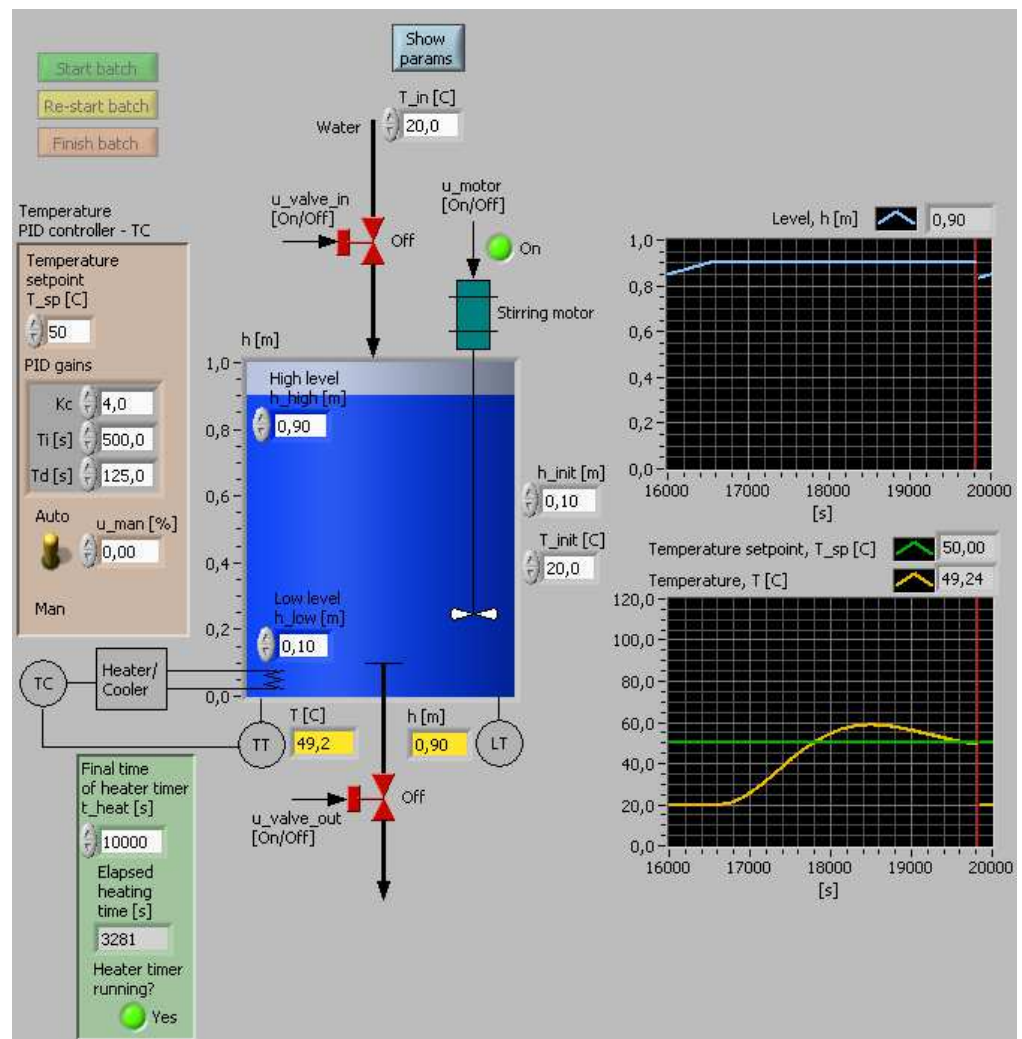


Figure 12.2: A batch process to be controlled by sequential control

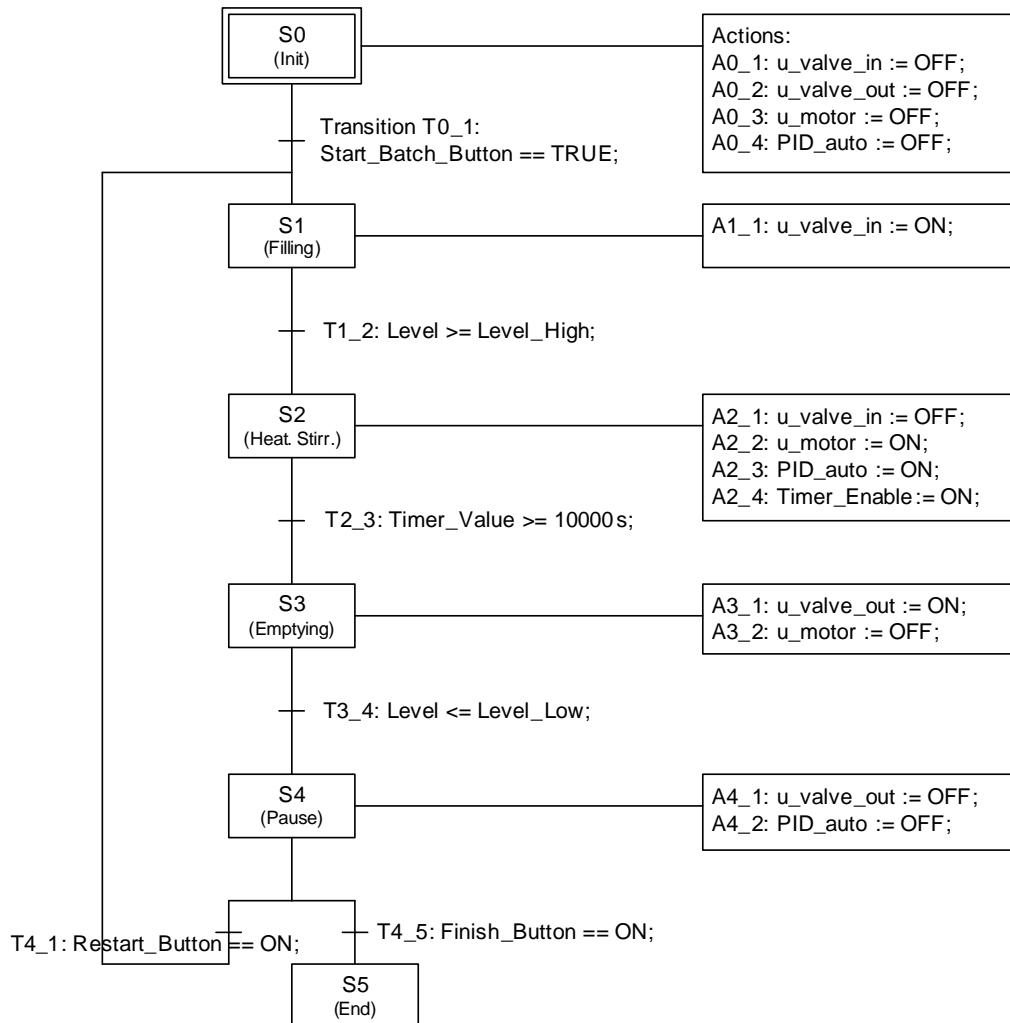


Figure 12.3: Sequential Function Chart (SFC) defining the control function

Appendix A

Codes and symbols used in Process & Instrumentation Diagrams

This appendix gives a brief overview over codes and symbols used in Process & Instrumentation Diagrams – P&IDs. The standards ISA S5.1 (USA) and ISO 3511-1 (International) define the these diagrams. There are also factory internal standards.

A.1 Letter codes

Table A.1 shows the most commonly used letter codes used in Process & Instrumentation Diagrams.

A.2 Instrumentation symbols used in P&IDs

The following figures show common symbols used in Process&Instrumentation Diagrams (P&IDs).

	As first letter	As subsequent letter
A	Alarm	Controller
C		
D	Density. Difference	
F	Flow. Fraction (ratio)	
G	Position	
H	Hand controlled	
I		Indicator
L	Level	
P	Pressure	
Q	Quality	
S	Speed	
T	Temperature	Transmitter (sensor)
V	Viscosity	Valve
Y		Function (e.g. mathematical)
Z		Secure control (e.g. interlock)

Table A.1: Common instrumentation codes used in Process&Instrumentation Diagrams

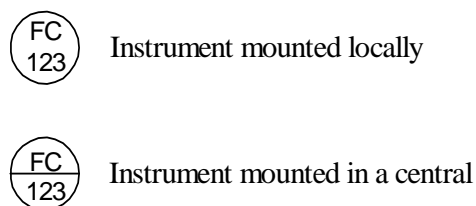


Figure A.1: Main instrument symbols (FC123 is one example of instrument code.) The numbering is based on running numbers, e.g. FC123, TC124 etc., but so that a specific control loop has a fixed number, e.g. FT123, TT124.

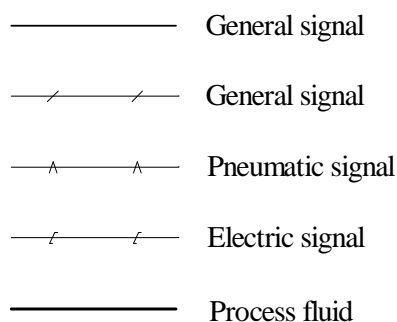


Figure A.2: Symbols of conductors and pipes

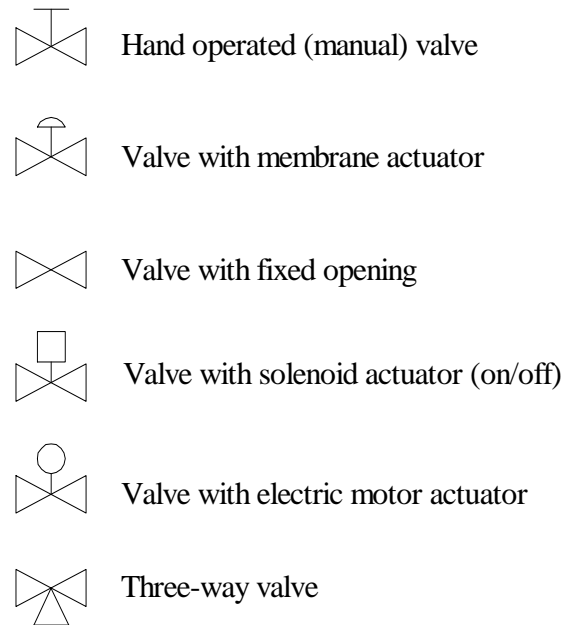


Figure A.3: Valve symbols

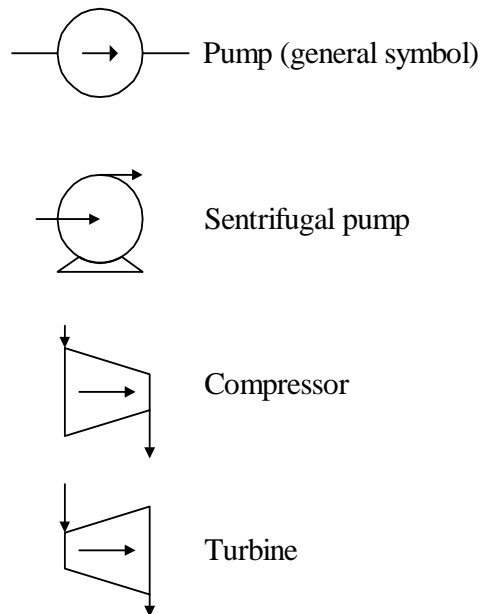


Figure A.4: Symbols of pumps, compressors and turbines

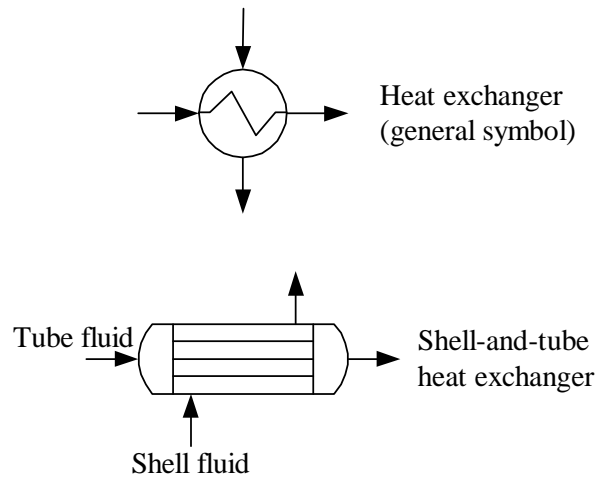


Figure A.5: Symbols of heat exchangers

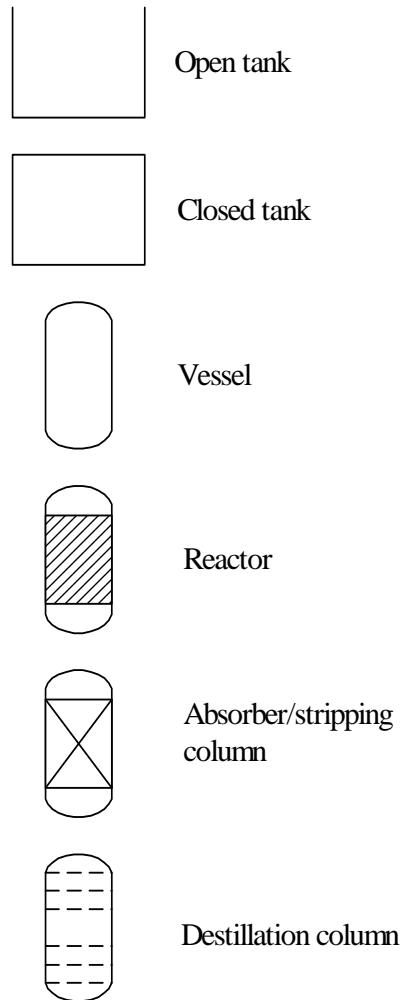


Figure A.6: Symbols of tanks and columns

Bibliography

- [1] O'Dwyer, A., *Handbook of Controller Tuning Rules*, Imperial College Press, London, 2003
- [2] Haugen, F.: *Advanced Dynamics and Control*, TechTeach (<http://techteach.no>), 2010
- [3] Haugen, F., *Comparison of PI tuning methods*, Nordic Process Control Workshop, 2010 (available at <http://techteach.no>.)
- [4] Mayr, O.: *Origins of Feedback Control*, The MIT Press, 1971
- [5] Seborg, D. E., Edgar, Th. F., Mellichamp, D. A.: *Process Dynamics and Control, 2. Ed*, Wiley, 2004
- [6] *Ullmanns' Encyclopedia of Industrial Chemistry*
- [7] Skogestad, S.: *Simple Analytical Rules for Model Reduction and PID Controller Tuning*, J. Process Control, Vol. 13, 2003
- [8] Ziegler, J. G. and Nichols, N. B.: *Optimum Settings for Automatic Controllers*, Trans. ASME, Vol. 64, 1942, s. 759-768
- [9] Åström, K. J. and T. Hägglund, T.: *Automatic Tuning of PID Controllers*, Instrument Society of America, 1988
- [10] Åström, K. J. and Wittenmark, B.: *Adaptive Control*, Addison-Wesley, 1989

Index

- actuator, 6
- adaptive control, 153
- anti wind-up, 98
- auto-tuning, 142
- automatic control, 81

- balance law, 34
- Block diagram, 6
- block diagram, 37
- block diagram manipulation, 65
- block-diagram
 - for differensiallikningsmodeller, 23
- bumpless transfer, 102

- capasitor, 51
- cascade control, 157
- characteristic polynomial, 65
- chip tank, 165
- control error, 1, 8
- control loop, 6
- control variable, 7
- controller, 117

- D-kick, 97
- DCS (Distributed Control Systems), 125
- dead-time, 75
- derivative kick, 97
- direct action, 94
- Distributed Control Systems (DCS), 125
- disturbance, 7, 35
- dynamic system, 22
- dynamics, 69

- effekt, 51
- elektriske systemer, 49
- elementary blocks
 - blocks elementary, 23
- environmental variable, 35
- error-driven control, 2

- feedback control, 2
- feedback loop, 6
- feedforward control, 105
- filter, 82
- filter in the D-term, 96
- first order systems, 71
- force balance, 45

- gain, 71
- gain scheduling, 147
- Good Gain method, 130

- inductor, 51
- input disturbance, 136
- integrator, 69
- integrator anti wind-up, 98
- integrator wind-up, 98
- inventory, 34

- Kirchhoff's current law, 50
- Kirchhoffs spenningslov, 50

- Laplace transform, 55
- linear motion, 45
- lowpass filter, 82
- lowpassfilter in the D-term, 96

- manipulating variable, 7
- manual control, 82
- manual control value, 82
- mass balance, 36

- mass system, 36
- mass-spring-damper, 25
 - static response, 32
- mathematical modeling, 33
- measurement filter, 82
- model, 33
- model errors, 33
- model uncertainty, 33
- modeling, 33
- mole balance, 39
- momentum, 45, 46
- momentum balance, 45, 46
- motion systems, 45

- Newton's second law, 45
- nominal control value, 82
- non-linear functions
 - block diagram, 24
- order
 - state-space model, 29
 - transfer function, 65
- P-kick, 97
- PAC (Programmable Automation Controller), 123
- PI diagrams, 4
- PLC (Programmable Logical Controller), 121
- pole, 65
- positioner, 168
- primary loop, 157, 158
- process, 6
- process and instrumentation
 - diagram, 4
- process controller, 117
- process output variable, 7
- Programmable Automation Controller (PAC), 123
- Programmable Logical Controller (PLC), 121
- proportional band, 89
- proportional kick, 97
- Pulse Width Modulation, 119
- PWM, 119

- ratio control, 168
- RC-circuit, 52
- repeats per minute, 89
- resistors, 50
- response-time, 76
- reverse action, 94
- rotational motion, 46

- SCADA systems, 123
- scaling, 84, 87
- secondary controller, 158
- secondary loop, 158
- sequential control, 185
- Skogestad's method, 135
- split-range control, 169
- stability, 102
- state feedback, 161
- state machine, 186
- state-space model, 28
- state-variable, 25
- static response, 31
- statisk transferfunksjon, 67
- Supervisory Control and Data Aquisition (SCADA), 123

- temperature control, 149
- time constant, 71
- time delay, 75
- time-constant, 71
- torque balance, 46
- transfer functions, 61, 62

- wind-up, 98
- wood-chip tank, 165

- zero, 64
- zero-pole gain form of transfer functions, 64