3. DIRECT DIGITAL CONTROL

3.1. Types of Control Systems

There exist two main ways of control:
- Open-loop system,
- Closed-loop systems meaning systems with feedback control (control system).

An open-loop system consists of a control element and the object being controlled (fig. 3.1a). The control element does not receive information about the output signal $y$, however it does have a specified control goal—keeping the output signal on a level specified by the demanded value $y_0$. Unfortunately, this goal in an open-loop control system is difficult to realize and the output signal is established according to values, which are dependent on the amount of noise (load). The changes in the output signal $y$ caused by the changes in noise $z$, are illustrated on the graph of signal $y$ in figure 1a. Beyond this it is possible to utilize the information from a known noise $z_1$ and performing the necessary correction on the demanded signal—this is so called a compensating system.

![Fig. 3.1. Control system: a) open-loop, b) closed-loop](image-url)

A closed-loop system is achieved as a result of inputting an output signal into the input utilizing a negative loop feedback. During operation noise affects the object, through which one may change insignificantly the
parameters of the object for example damping, springiness, thermal conductivity, all of which result from the aging of elements or the effects of external atmospheric conditions especially the changes of temperature. The second however causes a change in the value of signal in the system, for example the main interference in drives is the load moment, in a cooling chamber the interference’s are inadequate insulation and the time of product input.

From here the main aim of the control system is achieving the goal without regard to the affects of noise or interference. The goals of a control system result from their type. The main type is constant value control meaning to keep the output signal on a constant level independent of noise. In a phase-lag network the output signal changes freely, according to the needs of the technological process, however the output signal precisely lags behind these changes. We deal with these types of systems in aeronautics, guidance of airplanes and during the precise regulation of the phase angle in motors in servomechanisms. The last type are programmed control systems, where the output signal changes according to a well defined dependence resulting from the technology. This type of control system is utilized in numerically controlled lathes, in programmed cooling systems and the like.

In a constant value control system the goal is keeping the output value \( y \), beyond the summation loop, represented in figure 3.1b in the form of a circle, we receive deviation error \( e = y_0 - y \). In the case of a positive deviation error \( e > 0 \), meaning \( y_0 > y \) the system changes the orientation of the control element in a way so as to increase the value of the signal \( y \). However, in the case of a negative deviation error \( e < 0 \), meaning \( y_0 < y \), the system changes the orientation of the control element so as to decrease the value of signal \( y \). The change in orientation of the control element consists of, for example in electrical systems, changing the control angle of a thyristor drive supplying an motor, in a hydraulic system this will be the change of the deflection of the rotor in a pump, which will cause a change in the value of flow of liquid. The system is considered better when these changes (transient states) last as short as possible. The amount of time and the value of the deviation error is the basic measure of quality of a control system, and the main goal of a control system is keeping the deviation error on a level close to zero. Realization of the above criteria enables us in control systems:

- To limit the effect of noise on an object,
• To limit the effect of change of parameters of an object,
• Formation of dynamic characteristics of an object (improving its properties), and depending on the kind of the system: stabilizing of the output signal (fixed value systems), following the input value with output value (follow-up systems) and changing the output signal according to the defined program (programmed systems).

A different form of categorizing systems on the basis of work performed is:

• Continuous control systems,
• Impulse control systems (non continuous) (See chapter 4.9)

Impulse control systems are characterized, in that the output signal from the regulator or performing element affects the object only in certain time intervals, after which it takes on a zero value. These systems are often used in processes where slow changes occur, for example chemical, heating and cooling.

The most important element in a control system is the regulator (fig. 3.1b). This is where transformation of signal is performed, and quality of control results from the selection of its structure and parameters. The whole structure of the regulator consists of the proportional term P and its parameter $k_c$, integration term I and its parameter $T_i$, as well as a derivation term D and its parameter $T_d$. Controllers often take on the name depending on the type of control value, type of sensors or the characteristic constructional elements. For example thermoregulator and bob regulator.

Using only the proportional term (controller P) causes, that at the instant, when the demanded value $y_0$ equals the measured value $y$, the deviation error becomes equal to zero and the controller did nothing. This is why in a control system with a proportional P controller we always encounter a set deviation error $e_s$, whose value is that much smaller the greater the value of gain of controller $k_c$.

Applying only the integral term (controller PI,PID) results in the controller changing it’s output signal $y$, until it equals to the demanded value $y_0$. The rate of these changes is determined by the time of doubling $T_i$. At the instant when the signal of deviation of error $e$ becomes equal to zero and the output voltage from the controller stops changing, the final value is preserved. Using this controller often results in over regulation of the output signal. What this means is that during the short sampling time the output signal is greater than the demanded value.
Using the derivation term (controller PI,PID) accelerates the growth of the output signal during transient states.

3.2. Structures of Control Systems

Direct Digital Control consists of enabling the computer (of the controller) to work in a feedback loop configuration as well as using, for instance, a three-term control algorithm (PID-Proportional+Integral+Derivative), well known in analog technology. We are of course allowed to use different algorithms, however in 90% of the problems PID is sufficient.

![Diagram of System with Direct Digital Control](image)

Fig. 3.2. System with Direct Digital Control

In small systems a few loops are used, however in large ones a few hundred (Fig.3.2). We must mention that in this example the controller is in a direct layer, the set point is calculated by using a static or dynamic optimization method and is directed from the computer’s tool layer.

The basic structures are depicted in fig.3.3. The most frequently used is a structure with a controller in place of a classical regulator, meaning a structure with feedback control (Fig.3.3a).

If the effect of noise is significant a control system isn’t able to satisfy the required quality of control. In this case we are obligated to use feedforward control[2]. Its operation is based on the following:

- direct measurement of basic interference (Fig.3.3 b),
- indirect measurement of interference (Fig. 3.3 c) inferential control, internal model control, model algorithmic control, model predictive control, dead time control
Chapter 3. DIRECT DIGITAL CONTROL

In literature we often see the term invariant control, this means that the output signal doesn't change under the influence of noise. Feedforward control with indirect measurement can be realized by the measurement of the direct value, on the basis of which we can estimate the size of noise or measurement of two values, before and after the place of interference.

If the effect of variability of parameters of the object is significant or if the object is nonlinear, and the system isn’t able to provide the demanded quality of control, which in reality boils down to the fact that the system works correctly but only in the proximity of the specified point, then we use adaptive control. The way that these operate is that during use the object is identified in real time. This means that its parameters are derived on the basis of received information from which new settings are determined. Most frequently we deal with [2]:

- programmed adaptive control, gain scheduling utilizing value measurement, which is a measure of the variability of parameters of a process(Fig. 3.4 a) or the measurement of the environmental parameters,
which is a measure of the variability of the environmental parameters (Fig. 3.4 b),

- self-tuning control, where methods of identification of parameters are utilized (Fig. 3.4 c),
- adaptive control based on a reference model of a process (Fig. 3.4 d).

From the hardware point of view in order to realize adaptive control we need additional cards, which reside in the controller or computer tool layer. These form the sublayer of adaptive control.

Fig. 3.4. Adaptive control systems


3.3. Position Algorithm PID

A control system may be described by a mathematical model in the continuous or discrete form (Fig.3.5). Both are used interchangeably and there exist many methods which change the transfer function \( G(s) \) into the impulse transfer function \( G(z) \) or vice-versa. However after such a transformation of the model, it doesn’t give the same characteristics that is why we must test their stability. With a discrete model every change of sampling rate also necessitates this.

![Diagram of continuous and discrete control systems](image)

Fig. 3.5. Continuous and discrete control systems

In order to derive the transfer function \( G_i \) we utilize the following method [2]:
- three term algorithm of a PID controller,
- knowledge of the model of the object (See heading 3.7).

The general form describing the three term algorithm of a PID controller is as follows:

\[
mt = ke \cdot T_i \int e(t) dt + Te \cdot \frac{de(t)}{dt} \tag{3.1}
\]

where:
- \( e = y_0 - y \)
- \( y \) – instantaneous output value,
- \( y_0 \) – set value,
- \( e \) - deviation error,
- \( k_c \) – controllers gain,
- \( T_i \) – integration constant, doubling time,
- \( T_d \) – derivation constant, lead time.
Often $\frac{de}{dt}$ is substituted with $\frac{dy}{dt}$ in order to avoid derivation of the demanded value. If sampling time equals $T$ seconds then simple approximations:

$$\frac{de}{dt} = \frac{(e_n - e_{n-1})}{T} \quad i \quad \int e \, dt = \sum_{k=0}^{n} e_k T$$

(rectangle rule) \hspace{1cm} (3.2)

for $k = 0, 1, 2, ..., n$ enable us to transform formula (3.1) into:

$$m_k = k_c \left[ e_k + \frac{T}{T_i} s_k + \frac{T_d}{T} (e_k + e_{k-1}) \right]$$

(3.3)

where: $s_k = s_{k-1} + e_k$ (integral summation) \hspace{1cm} (3.4)
Substituting:

\[ K_c = k_c, \quad K_i = k_c \frac{T}{T_i}, \quad K_d = k_c \frac{T_i}{T} \]  \hspace{1cm} (3.5)

we achieve the so called position algorithm:

\[ m_k = K_c e_k + K_i s_k + K_d (e_k - e_{k-1}) \] \hspace{1cm} (3.6)

From which on the basis of equations (3.4) and (3.6) a simple PID controller programmed in PASCAL has the following form [2]:

```
PROGRAM PIDController;
CONST
  kpvalue = 1.0;
  kivalue  = 0.8;
  kdvalue = 0.3;
VAR
  s,kp,ki,kd,ek,ekold,mk      : REAL;
  stop                                    : BOOLEAN;
FUNCTION ADC                 : REAL;EXTERNAL;
PROCEDURE DAC(VAR mn:REAL);EXTERNAL;
BEGIN                                    (program główny)
  stop   := FALSE;
  s        := 0.0;
  kp     := kpvalue;
  ki      := kivalue;
  kd     := kdvalue;
  skold := ADC;
REPEAT         (control loop)
  ek      := ADC;                       (value of error ek from ADC)
  s        := s + ek;                      (integrating sum)
  mk    := kp * ek + ki * s + kd * (ek - ekold);
  DAC(mk);
  ekold:= ek;
UNTIL stop;
END.
```

The mentioned program is not correct and not practical. Not correct because the controlling variables are not synchronized with real time and sampling time \( T \) is dependent on the speed of the computer on which it is run.
Chapter 3. DIRECT DIGITAL CONTROL

It is impractical because it doesn’t take into account the limitations of the performing elements, interference in the measurement path and process. Beyond this the parameters of the controller are built into the program, this is why every change necessitates a recompilation of the program.

3.4. Timing

The basic characteristic of programs working in real time is that they must be run in a loop throughout the whole time. This is realized in PASCAL using the REPEAT...UNTIL loop. This is why the general form of the program is the following[2]:

```
PROGRAM ControlInRealTime;
...
(declaration)
BEGIN
  REPEAT
    Control Task;
  UNTIL stop;  
(declaration)
END.          
(ControlInRealTime)
```

Unfortunately, the time of execution of this loop meaning the time of the cycle is variable, which results in a change of the coefficients $K_i$ and $K_d$ in the controller equation (3.6) containing the sample time. This forces the Control method inside the loop to be synchronized with the needs of the process. Synchronization, which is a fragment of this task, can be realized according to the following method [2]:

• polling,
• internal interrupts,
• ballast coding,
• signals from the real time clock.

The first two methods rely on taking the values of signals, when the controller is ready to control, of course in the sampling period $T$.

The difference between polling and internal interrupts is that in a polling system the controller continuously reads values, while in a system with interrupts the controller can perform different calculations, at the moment the interrupt starts working it causes a halt in calculation and execution of ControlTask [2].

```
PROCEDURE ControlTask;
BEGIN
  REPEAT
    Control Task;
  UNTIL stop;
END. 
(ControlInRealTime)
```


**Chapter 3. DIRECT DIGITAL CONTROL**

```plaintext
PROGRAM PIDController;
CONST
    kpvalue = 1.0;
    kivalue = 0.8;
    kdvalue = 0.3
```

The polling method is restricted to small multicomputer systems, its strong point is simplicity and ease of programming. The similar situation exists in ballast coding. Time necessary to execute program code $P$ in the controller is dependent on the instructions and values of the information, even for the code of the afore mentioned examples, which do not include any branches. The time to execute basic arithmetic operations will change in accordance with the sign and actual value of the variable. This is more evident on the figure 3.6, where for every branch the time of calculation is calculated (or measured), and the ballast code $B$ is added to every path so that the average calculation time may be equal to the sampling time $T$. This method may not be used exclusively with interrupts or when the CPU’s clock is changed, because then it is necessary to define the ballast code values once more. Using the real time clock gives a great deal of ability in solving the problem of time synchronization and is very often applied in controllers. Below is a simple example [2].
SampleInterval = 0.01;
VAR
  s,kp,k_1,ke,en,enold,mn : REAL;
  time, nextSampleTime : REAL;
  stop : BOOLEAN;
FUNCTION ADC : REAL; EXTERNAL;
FUNCTION SetTime : REAL; EXTERNAL;
PROCEDURE DAC (VAR mn : REAL); EXTERNAL;
BEGIN
  (main program)
  stop := FALSE;
  s := 0.0;
  kp := kpvalue;
  ki := k_1value;
  kd := kdvalue;
  enold := ADC;
  time := GetTime;
  nextSampleInterval := time + SampleInterval;
REPEAT
  (Control loop)
  WHILE time < nextSampleTime DO
    time := GetTime;
  END
  (end of while loop)
  en := ADC; (error value of en from ADC)
  s := s + en; (integral summation)
  mn := kp * en + ki * s + kd * (en - enold);
  DAC(mn);
  enold := en;
  nextSampleTime := time + SampleInterval;
UNTIL stop;
END
  (PIDController)
In the example it is assumed that GetTime is a function which inputs the value of current time. The WHILE loop <condition> DO works as a delay loop, in which the program waits for the next sample. It should be noticed that the variable “time” is continuously updated. After every calculation of control the time for the next sample is updated by adding the time of sampling to the variable “time”.
In controllers this is realized by the use of an internal impulse clock, which synchronizes the work of the whole controller program. It must be
underlined that the time of the clock impulse must be at the least two times greater than the cycle time of the controller.

3.5. Practical modifications of the PID controller

The previously mentioned algorithm regards an ideal controller and non interactive. In practice we apply the following modifications of this algorithm, which enable[2]:

a) bumpless transfer,

b) eliminating the effects of saturation, integral wind-up,

c) limiting the effects of noise,

d) better smoothing of control signal due to use of modified integration and derivation.

Ad. a) Equation (3.6) assumes, that in an assumed state, with a zero deviation error, and sometimes at \( K_i s_k = 0 \), the output variable is equal to zero. In many applications, for instance in electric motor, a real value is necessary, that is why the algorithm takes on the following form:

\[
m_k = K_c e_k + K_i s_k + K_d (e_k - e_{k-1}) + C
\]

where: \( C \) – set-point of controller operation

It’s presence, however, causes large interference in the object especially while switching from manual to automatic operation, as well as during switching hardware elements producing large changes in signal (for example changing the state of a hydraulic valve). That is why it is necessary to take into account the ability of smooth switching. This can be realized in one of the following ways:

- calculate the value of \( C \) for a given work point in an steady state and place it into the program, of course this value is only valid for a given load,

- the value of the output signal is followed in the manual mode of operation and at the instant of switching \((p)\) this is value \( C \), but then the integral summation is defined in the following manner:

\[
s_{ik(p)} = m_{c(p)} - K e_{c(p)} - C
\]
Chapter 3. DIRECT DIGITAL CONTROL

where: $m_{c(p)}$ - value of $m$ at the instant upon switching,
$e_{c(p)}$ - value of deviation error at the instant upon switching,
$K$ - rate of change of integral summation coefficient,

- using the widely applied velocity algorithm, which gives the change of output value for every sample. For a continuous form (3.1) the equation of the controller after derivation in the function of time yields the form, in which integration does not appear:

$$\frac{dm(t)}{dt} = K_c \frac{de(t)}{dt} + K_i e(t) + K_d \frac{d^2 e(t)}{dt^2}$$  \hspace{1cm} (3.9)

The difference equation may be obtained by using backward differences in the above equation or defining $m_n - m_{n-1}$ from equation (3.3), which gives:

$$\Delta m_n = m_n - m_{n-1} = k_c \left[ (e_n - e_{n-1}) + \frac{T}{T_i} e_n + \frac{T_d}{T} (e_n - 2e_{n-1} + e_{n-2}) \right]$$  \hspace{1cm} (3.10)

Substituting:

$$K_i = k_c \left( 1 + \frac{T}{T_i} + \frac{T_d}{T} \right)$$
$$K_2 = - k_c \left( 1 + 2 \frac{T_d}{T} \right)$$
$$K_3 = k_c \frac{T_d}{T}$$

Equation (3.9) takes on the following form:

$$\Delta m_n = K_1 e_n + K_2 e_{n-1} + K_3 e_{n-2} \text{ lub } m_n = m_{n-1} + K_1 e_n + K_2 e_{n-1} + K_3 e_{n-2}$$  \hspace{1cm} (3.11)

In practical solutions the $k_c$ parameter is often adjusted, that is why the form of the above equation is often applied in the following manner:

$$m_n = m_{n-1} + k_c (K_1^* e_n + K_2^* e_{n-1} + K_3^* e_{n-2})$$  \hspace{1cm} (3.12)

where: $K_1^*$, $K_2^*$, $K_3^*$ are coefficients $K_1$, $K_2$, $K_3$ without $k_c$.

This algorithm automatically gives smooth transitions, however if there exists a constant large deviation error during change, the response of the
controller is sluggish, especially when the time constant of integration $T_i$ is large. Comparing the position algorithm with the velocity one can notice that the second is easier to program because it doesn’t contain integration. However it has to present in the object.

During the incremental change of the set point the derivation term causes a rapid jump of the output signal. In the interest of reducing this jump in the velocity algorithm (3.10) we perform the following substitution (see fig.3.7)[12]:

$$e_n = y_0 - x_n, \quad e_{n-1} = y_0 - x_{n-1}, \quad e_{n-2} = y_0 - x_{n-2}$$

(3.13)

where:

- $y_0$ – set point,
- $x_n$ – output signal directed into the summation loop

from which:

$$\Delta m_n = k_c \left[ (x_{n-1} - x_n) + \frac{T}{T_i}(y_0 - x_n) + \frac{T}{T}(2x_{n-1} - x_{n-2} - x_n) \right]$$

(3.14)

The set point only appears in the integration term that is why the controller must always contain the integration term, meaning the program should always satisfy: $\frac{T}{T_i} > 0$.

Ad. b) In practical considerations the output variable $m$ is physically limited. Output signals from the controller will not be greater than the supply voltage, the valve after opening with a constant pressure will not yield greater flow etc. At these states called saturation feedback control does not function. The most frequently used methods accounting for this phenomena are:

- constant boundaries for the summation integral,
- halted summation at saturation,
- integral subtraction,
- applying the velocity algorithm.

The simplest method results in constant boundaries for the integral summation:

$$s_{\max} = m_{\max} \quad \text{and} \quad s_{\min} = m_{\min},$$

(Due to limited range of operation of a D/A converter, the same boundaries are applied for output variable $m$).

that is why in the program PIDController the following modification should be made:

REPEAT
Chapter 3. DIRECT DIGITAL CONTROL

\[
\begin{align*}
en & := \text{ADC;} \quad \text{(error value en from ADC)} \\
s & := s + en; \quad \text{(integral summation)} \\
\text{IF } s > \text{smax } \text{THEN } s & := \text{smax} \quad \text{(upper limit)} \\
\text{ELSE IF } s < \text{smin } \text{THEN } s & := \text{smin}; \quad \text{(lower limit)} \\
\text{IF } mn > \text{smax } \text{THEN } mn & := \text{smax} \quad \text{(upper limit of DAC)} \\
\text{ELSE IF } ma < \text{smin } \text{THEN } ma & := \text{smin}; \quad \text{(lower limit of DAC)} \\
mn & := kp * en + ki * s + kd * (en - enold); \\
\text{DAC}(mn); \\
enold & := en; \\
\text{UNTIL stop;} \\
\end{align*}
\]

In the next method the integral summation is sort of frozen, while the output is saturated. That is when the value in the integrator stays constant. The system may be realized, utilizing the signal from the sensor which informs about the state of saturation of the performing element.

Both of the above methods are used for large values of saturation, beyond this in both methods, a large value in the integrator, when the system comes out of saturation does not ensure satisfactory dynamics of the object.

The idea of the method of subtractional integration is that the integral summation decreases proportionally to the difference between the calculated output variable value \( mn \) and the maximum allowable \( m_{max} \). That is why the equation

\[
s_n = s_{n-1} + e_n
\]

takes on the form

\[
s_n = s_{n-1} - K ( m_n - m_{max} ) + e_n \quad \text{with the assumption, that for } m_n > m_{max} \quad (3.15)
\]

The degree of decrease of this sum is dependent on the experimental choice of \( K \). The improper selection can cause oscillations on the boundary of saturation. This method is modified by halting the addition of error \( e_n \) in saturation on the basis of saturation taken from the performing element. The strong point of this is a fast exit from saturation.

Application of the previously mentioned velocity algorithm causes automatic setting of the limits of integration.
Ad. c) In analog systems noise with high frequency in measured signals don’t cause any problems especially due to the fact that almost all the elements of the system work as lowpass filters weakening the noise. In digital systems, noise with high frequencies may cause interference with low frequencies because of aliasing error. Low frequency interference has the same amplitude as the original noise and its frequency is the difference between the frequency of noise and the exponent of the sampling period (fig.3.7). In the interest of reducing this effect the signal should be filtered before sampling. In many industrial applications all that is necessary is to use a simple first-order analog filter with a time constant $T_f$ and transfer function [17]:

$$G_f(s) = \frac{u(s)}{y(s)} = \frac{1}{1 + \frac{T_f}{T} s}$$

(3.16)

where: $T_f = T/2$ and $T$ is sampling period.

Fig. 3.7. Aliasing Error

Fig. 3.8. General Scheme of control system with filter

When $T_f$ is small we apply only an analog filter, however when $T_f$ is greater than a few seconds we apply an analog as well as a digital filter (fig. 3.8).

Numerical approximation of a first-order phase-lag system
where: $u$ - input and $x$ - output, is the following:

$$x_{n+1} = [1 - e^{-\frac{T}{T_f}}] x_n + e^{-\frac{T}{T_f}} u_n$$

where: $T$ - sampling rate of filter and $T < T_f$.

Inputting: $\alpha = e^{-\frac{T}{T_f}}$ and the previous time instant we receive the following form of a digital filter:

$$x_n = [1 - \alpha] x_{n-1} + \alpha u_{n-1}$$

Notice, that if $\alpha = 1$, then there is no smoothing, and if $\alpha = 0$ then the signal is zero, the input doesn’t provide a signal. Beyond this the time constant of the analog filter is usually so small that it doesn’t worsen the performance of the system. One also should not allow an unusually large value $T_f$.

An alternative solution is applying a real time PID control algorithm with the following transfer function:

$$G(s) = \frac{k_c (1 + T_i s)}{T_i s} \cdot \frac{(1 + T_d s)}{(1 + \alpha T_d s)}$$

where in practical applications we assume that $\alpha = 1/6 \div 1/20$.

The second part of this transfer function is the real time derivation term (with inertia) instead of a clear derivation term. In these solutions the signals at the input and output must both be filtered.

**Ad.d)** Applying modified algorithms in calculating integration and derivation.

Position and velocity algorithms utilize the first and second-order difference to calculate the derivation term. However this algorithm inputs roughness in the process. Applying an algorithm, which averages the values of a few samples, allows limiting the effect of noise and errors. Beyond this it has the basic properties of a velocity algorithm meaning smooth transitions. Applying the method of four point central differences we obtain [12, 36]:

$$u = T_f \frac{dx}{dt} + x$$ (3.17)
\[
\frac{de}{dt} = \frac{\Delta e}{T} \left[ e_n + 3e_{n-1} - 3e_{n-2} - e_{n-3} \right] \tag{3.20}
\]

After application in equation (3.3) substitutions \( \frac{T_d}{T} [ e_n - e_{n-1} ] = T_d \frac{\Delta e}{T} \) and the above dependency, we receive:

\[
m_n = k_c \left[ e_n + \frac{T_d}{6T} ( e_n + 3e_{n-1} - 3e_{n-2} - e_{n-3} ) + \frac{T}{T_i} \sum_{k=0}^{n} e_k \right] \tag{3.21}
\]

The positioning algorithm takes on the following form:

\[
m_k = p_1 e_k + p_2 e_{k-1} + p_3 e_{k-2} + p_4 s_k \tag{3.22}
\]

where: \( s_k = s_{k-1} + e_k \) as well as

\[
p_1 = k_c (1 + \frac{T_d}{2T}); \quad p_2 = -k_c \frac{T_d}{2T}; \quad p_3 = -k_c \frac{T_d}{6T}; \quad p_4 = k_c \frac{T}{T_i}
\]

In the interest of improving the precision of integration we can also apply the trapezoid rule instead of the rectangle rule. That is why equation (3.3) takes on the following form:

\[
m_n = k_c \left[ e_n + \frac{T_d}{T} ( e_n - e_{n-1} ) + \frac{T}{T_i} \sum_{k=0}^{n} \frac{e_k + e_{k-1}}{2} \right] \tag{3.23}
\]

for the previous time instant we have:

\[
m_{n-1} = k_c \left[ e_{n-1} + \frac{T_d}{T} ( e_{n-1} - e_{n-2} ) + \frac{T}{T_i} \sum_{k=0}^{n-1} \frac{e_k + e_{k-1}}{2} \right] \tag{3.24}
\]

so the positioning algorithm has the following form:

\[
\Delta m_n = m_n - m_{n-1}
\]

\[
\Delta m_n = k_c \left[ (1 + \frac{T}{2T_i} + \frac{T_d}{T}) e_n + (\frac{T}{2T_i} - \frac{2T_d}{T} - 1) e_{n-1} + \frac{T_d}{T} e_{n-2} \right] \tag{3.25}
\]

In literature [17] we can encounter many more precise methods of integration for example the Adams two-point method and Simson’s three-point method.

Notice should also be taken to the fact that for objects with phase-lag the effectiveness of the PID controller is seriously limited. That is why a series of
methods is used for example an additional compensating term (see chapter 4.31).

The above mentioned algorithms as can be seen are particular examples of a digital algorithm described in the previous chapter (chapter 2.19) of the general form:

\[ m_n = -a_1 m_{n-1} - a_2 m_{n-2} - ... + b_0 e_{n} + b_1 e_{n-1} + ... + b_m e_{n-m} \]  

(3.26)

In which we have summation of the elements of the input and output series.

Figure 3.9 illustrates the basic structures of control systems with above mentioned PID controller.

### 3.6. Selection of controller settings

For a designer of a control system (control, compensation or open) the main goal is improvement of quality of control, in which the performance criterion of quality of control is \( I \). In the most general form it is a function of the structure of a system and parameters of control. The whole problem boils down to finding such a function so that \( I \) can take on minimum or maximum values.

\[ I = I [ u(t); x(t); O; R ] \]  

(3.27)

where: 
- \( u \) – control vector (for on dimensional objects this can be the response of the system to a unit step and its’ derivative),
- \( x \) – state vector (for one dimensional objects this can be the response of the system to a unit step or noise, deviation error and its derivative),
- \( O \) - matrices \( A, B, C, D \) in the state equation and the output (equation 1.5) or object’s parameters,
- \( R \) – for example the controllers parameters \( K_c, K_i, K_d \)

The process of defining function \( u(t) \), whose input in the form of a signal on the input of the system will optimize the performance of the system due to certain performance criteria carries the name optimization. Searching for this optimum strategy occurs with known parameters of the object and controller.

In most methods the following transfer function of operation of the system is adopted:

\[ G (s) = \frac{Y(s)}{X(s)} = K \frac{e^{-Ls}}{1 + sT_p} \]  

(3.28)
Chapter 3. DIRECT DIGITAL CONTROL

Fig. 3.9a. Fundamental structures of control system with PID controller
Chapter 3. DIRECT DIGITAL CONTROL

Fig. 3.9b,c,d. Fundamental structures of control system with PID controller.
Chapter 3. DIRECT DIGITAL CONTROL

For such a transfer function of an object the response to a unit step input \( X(t) = A \, 1(t) \) has the form of figure 3.10, where: \( K \) – is equal to 1,

\[
K \quad \text{real} \quad \text{according to (3.28)}
\]

Fig. 3.10. Response to unit step input

However the operational control transfer function takes on the following form:

\[
m(s) = k_c \left( 1 + \frac{1}{sT_i} + sT_d \right)
\]

(3.29)

with an eventual substitution \( K_c = k_c, \quad K_i = \frac{k_c}{T_i}, \quad K_d = k_c T_d \)

In analog systems, knowing \( T_p \) and \( L \) (parameters of the object), a structure of controller \( P, PI, PID \) is selected and its parameters \( K_c, K_i, K_d \) defined. These optimize the chosen values of the performance criterion. This is so called parametrical optimization.

The following performance criteria are used in applications:

a) Number criteria defined directly from the step-response of the system or from the error deviation of a standard (step, impulse), from the frequency characteristics as well as decomposition of elements. For example the most important performance criterion achieved from the step-response are: basic time constant \( T_p \), time lag \( L \), and for oscillating objects overshoot, meaning the first amplitude accounted against the set point depicted as a percentage.
Chapter 3. DIRECT DIGITAL CONTROL

Fig. 3.11. Characteristics of deviation for non-oscillatory and oscillatory object

For a graph of error deviation of a oscillating and non oscillating object the most important criteria are (fig.3.11):

- maximum dynamic error deviation \( e_m = \max\{e(t)\} \)
- error deviation set \( e_s = e(x) = \lim_{t \to \infty} e(t) \)
- oscillation \( d = 100 \frac{e_1}{e_2} [%] \)
- time of control (response) \( t_r \) for a deviation of \( \pm 5\% e_s \)

b) An integral criterion only gives us a measure of the non-defined states in a system and from their behalf nothing can be concluded about the statical error deviation. The most general form of a performance criterion is the following function:

\[
I_x = \int_0^\infty V(t) dt = \int_0^\infty X(t) \cdot P \cdot X(t) dt
\]

(3.30)

Where: \( V(t) \) – square form of the coordinates of states (second-order polynomial) positively half defined meaning \( V(t) \geq 0 \) when matrix \( P \) meets Sylvester’s condition. We can also define a different kind of performance criteria for example: cost, profit, output and proper functioning. Kalman formulated a performance criterion of the following form [59]:

\[
I_{en} = \frac{1}{2} e^T (t_f) F e(t_f) + \frac{1}{2} \int_{t_0}^{t_f} [e^T Q(t) e + u^T R(t) u] dt
\]

(3.31)

where the first term is at the expense of deviation in a final state, the second is loss from the existence of error deviation and the third is at the expense of control. Simpler forms of the above for one dimensional systems are:
Chapter 3. DIRECT DIGITAL CONTROL

\[ I_1 = \int_0^\infty e(t) \, dt \]
\[ I_{1t} = \int_0^\infty t \, e(t) \, dt \]
\[ I_{1m} = \int_0^\infty |e(t)| \, dt \quad \text{oscillations up to 10\%} \]
\[ I_{1t, m} = \int_0^\infty t|e(t)| \, dt \quad \text{almost aperiodic response} \]
\[ I_{1t, 2m} = \int_0^\infty t^2 |e(t)| \, dt \quad \text{gives aperiodicity and little time to control} \]
\[ I_2 = \int_0^\infty e^2(t) \, dt \quad \text{gives oscillations up to 35\%} \]
\[ I_\lambda = \int_0^\infty \left[ e^2 + \lambda \dot{e}^2 \right] dt \quad \text{gives one over gain and short control time,} \]
\[ \lambda \quad \text{is taken on the interval 1.22 \div 1.56.} \]

Among the many suboptimal methods for the selection of parameters of controller the one with the greatest significance is the Ziegler-Nichols method, which minimizes the integral \( I_{1m} \). It makes use of the fact that the object is controlled by a controller set to work proportionally (P), carefully increasing the coefficient of gain until the \( k_0 \) we come to the boundary of stability (oscillations with the following period occur \( t_0 \)), from this we obtain for controller (3.29) the following values of settings:

- \( P \quad k_e = 0.5 \quad k_0 \)
- \( \text{PI} \quad k_e = 0.45 \quad k_0, \quad T_i = 0.85 \quad t_0 \)
- \( \text{PID} \quad k_e = 0.6 \quad k_0, \quad T_i = 0.5 \quad t_0, \quad T_d = 0.12 \quad t_0 \)

(3.32)
There also exists a modified version of this method, which takes into account the sampling time $T$, in which we define the settings of the controller according to the following formulas:

$$\text{PI} \quad k_c = 0.45 k_0 \left( 1 - 0.6 \frac{T}{t_0} \right), \quad T_i = 1.85 t_0 \frac{k_c}{k_0}$$

$$\text{PID} \quad k_c = 0.6 k_0 \left( 1 - \frac{T}{t_0} \right), \quad T_i = 0.83 t_0 \frac{k_c}{k_0}, \quad T_d = 0.075 t_0 \frac{k_0}{k_c}$$

(3.33)

Taking into account the step-response of the object (rys.3.9) without feedback control, if such a possibility exists, we receive for the structure of the regulator (3.29) the following values of settings:

$$\text{P} \quad k_c = (0.57 \div 0.7) \frac{T_p}{KL}$$

$$\text{PI} \quad k_c = 0.7 \frac{T_p}{KL}, \quad T_i = L + 0.3 T_p$$

$$\text{PID} \quad k_c = 1.2 \frac{T_p}{KL}, \quad T_i = 2L, \quad T_d = 0.4 L$$

(3.34)

This method minimizes time of control $t_r$, and overshoot doesn’t exceed 20%.

In digital systems the number of quantization $q$ and the sampling period $T$ decide about the selection of settings for the above-mentioned controller.

$$I = I \left( q, T, k_c, K_i, K_d, T_p, L \right)$$

(3.35)

During the control of industrial processes the calculations are performed on real numbers or floating point arithmetic with the appropriate precision (word length) and the selection of levels does not pose a problem. The only places where word length is limited in the interest of attaining the shortest time of calculation are control of rockets and airplanes where short time constants are necessary.

Intuition leads us to the following question. Should the sampling time be as small as possible? In fact, taking into account a PID controller, it is noticed that there exists a certain “optimal” sampling period. For objects with properties similar to a lowpass filter the sampling period $T$ should be between the interval $0.07 t_r \leq T \leq 0.17 t_r$, where $t_r$ is the response time, meaning the step
response of the output signal up to 95% of the aimed at value. For object with considerable phase-lag according to Goff [9] the following reasoning is applied. \( T \leq 0.3L \) to be more precise \( 0.12 L \leq T \leq 0.25 L \). What’s more the value of \( \frac{T}{T_i} \) should be from the interval \( 2 \div 6 \), where the value should be small for processes with dominating phase-lag time and small for processes with small phase-lag time.

For a PID controller whose algorithm has the form (3.14), where: \( K_p = k_c \), \( K_i = k_c \frac{T}{T_i} \),
\[ K_d = k_c \frac{T_d}{T} \] and making use of criteria \( I_2 \), Tokahashi [35], with the assumption \( \frac{L}{T} \geq 5 \), stated the following ways of selecting parameters of a controller which account for sampling rate, defined from the step response (fig.3.9) in an open-loop system:

\[
K_p = \frac{1.2 \times T_p}{K(L+T)} - 0.5 K_i,
\]
\[
K_i = \frac{0.6 \times T_p}{K(L+0.5T)^2},
\]
\[
K_d = \frac{0.5T_p}{KT} \quad \text{or} \quad \frac{0.6T_p}{KT} \quad \text{when} \quad \frac{L}{T} \quad \text{is an integer.} \quad (3.36)
\]

After substitution of \( T = 0 \) the above methods are in agreement with the previously mentioned Ziegler-Nichols. It must be mentioned that two well known Kessler criteria stem from the integral criteria: modular optimum and symmetrical optimum, discussed in more detail in [15,38].
3.7. Changing discrete transfer functions into differential equation form

In literature [12, 15, 23, 36] many methods utilizing the model of the object to design a controller are mentioned. In all these methods pertaining mainly to linear time-invariant models the two forms of depiction of controller are used:

- transfer function,
- state equations.

If we have a description in the form of a differential equation, then we are able to program them directly. However when the description is in the form of a transfer function, it must be changed into a differential equation. A few ways of accomplishing this exist:

- Direct,
- cascade,
- parallel.

In work [2] it was shown, that both of these last methods with the same quantization level give lesser numerical errors:

a) The transfer function of the controller is of the following form (polynomial form):

\[ M(z) = \sum_{j=0}^{n} a_j z^{-j} \]

\[ E(z) = 1 + \sum_{j=1}^{n} b_j z^{-j} \]

We can directly change this function into recurrent form, which is easier to program[2]:

\[ m_i = \sum_{j=1}^{n} a_j e_{i-j} - \sum_{j=1}^{n} b_j m_{i-j} \]  

Example: We received the following transfer function of a controller:

\[ M(z) = \frac{5 + 3.2 z^{-1} + 0.5 z^{-2}}{1 + 0.3 z^{-1} - 2.3 z^{-2}} \]

The computer algorithm will have the following form:

\[ m_i = 5 e_i + 3.2 e_{i-1} + 0.5 e_{i-2} - 0.3 m_{i-1} + 2.3 m_{i-2} \]
Chapter 3. DIRECT DIGITAL CONTROL

b) In the cascade method the numerator and denominator of the transfer function should be separated into first or second order polynomials. This form is often referred to as factored or zero-pole-gain. Next both of the polynomials from the cascade method are changed utilizing the direct method. In the above example taking apart the numerator and denominator we receive the following (fig. 3.12):

\[
M(z) = \frac{3(1 + z^{-1})(1 - 0.2z^{-1})}{(1 + 0.5z^{-1})(1 - 0.4z^{-1})}
\]

From which:
\[
W_1 = 3 \\
W_2 = 1 + z^{-1} \\
W_3 = 1 - 0.2z^{-1} \\
W_4 = \frac{1}{1 + 0.5z^{-1}} \\
W_4 = \frac{1}{1 - 0.4z^{-1}}
\]

Rearranging the above transfer function we receive
\[
x_i^1 = 3 e_i \\
x_i^2 = x_i^1 + x_{i-1}^2 \\
x_i^3 = x_i^2 - 0.2 x_{i-1}^3 \\
x_i^4 = x_i^3 - 0.5 x_{i-1}^4 \\
m_i = x_i^5 = x_i^4 + 0.4 x_{i-1}^5
\]

c) In the parallel methods the fractions are broken down into partial fractions or residue form. Next each of the simple fractions is changed using the direct method where the output is the sum of individual block’s outputs. The previously mentioned transfer function after being broken down into simple fractions has the following form (fig. 3.13)
Fig. 3.12. Representation of the parallel transfer function

and the algorithm takes on the following form:

\[ x_i^1 = -3 \, e_i \]
\[ x_i^2 = -e_i - 0.5 \, x_{i-1} \]
\[ x_i^3 = 7 \, e_i + 0.4 \, x_{i-1} \]
\[ m_i = x_i^1 + x_i^2 + x_i^3 \]

Let’s analyze as an example the PID controller:

\[ m_n = K_c \left[ e_n + K_i \sum_{k=0}^{n} e_k + K_d (e_n - e_{n-1}) \right] \]

where: \( K_i = \frac{T}{T_i}, \quad K_d = \frac{T_d}{T} \)

Utilizing the Z transform we attain:

\[ \frac{M(z)}{E(z)} = K_c \left[ 1 + K_i \frac{z^{-1}}{z-1} + K_d \left(1 - z^{-1}\right) \right] \]

Dividing by \( z \) the transfer function of the integral term we receive:

\[ \frac{M(z)}{E(z)} = K_c \left[ 1 + K_i \frac{1}{1 - z^{-1}} + K_d \left(1 - z^{-1}\right) \right] \]

rearranging

\[ \frac{M(z)}{E(z)} = K_c (1 + K_d) + K_c K_i \frac{1}{1 - z^{-1}} - K_c K_d z^{-1} \]

the algorithm has the following form

\[ x_i^1 = K_c (1 + K_d) \, e_i = K_c (1 + \frac{T_d}{T}) \, e_i \]
\[ x_i^2 = K_c K_i \, e_i + x_{i-1}^2 = K_c \frac{T}{T_i} \, e_i + x_{i-1}^2 \]
\[ x_i^3 = -K_c K_d \, e_{i-1} = -K_c \frac{T_d}{T} \, e_{i-1} \]
Chapter 3. DIRECT DIGITAL CONTROL

\[ m_i = x_i^1 + x_i^2 + x_i^3 \]

It is clearly evident that the second equation has the following form:
\[ s_i = K_1 e_i + s_{i-1} \]

Taking this into account it can be written that:
\[ m_i = K_c (1 + K_d) e_i + s_i - K_c K_d e_{i-1} \]

in other words
\[ m_i = K_2 e_i + K_3 e_{i-1} + s_i \]
\[ s_i = K_1 e_i + s_{i-1} \]

where:
\[ K_1 = K_c \frac{T_i}{T} \]
\[ K_2 = K_c (1 + \frac{T_d}{T}) \]
\[ K_3 = -K_c \frac{T_d}{T} \]

Alternatively let’s write the controller equation using only the lead operator:
\[
\frac{M(z)}{E(z)} = K_c \left[ \frac{(1 + K_i + K_d)z^2 - (1 + 2K_d)z + K_d}{z(z-1)} \right]
\]

Dividing by \( z^2 \) we achieve the lag operator transfer function:
\[
\frac{M(z)}{E(z)} = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1}}
\]

where:
\[ a_0 = K_c (1 + K_i + K_d) \]
\[ a_1 = -K_c (1 + 2K_d) \]
\[ a_2 = K_c K_d \]
\[ b_1 = -1 \]

Which after rearrangement gives us the same previously mentioned velocity algorithm:
\[ m_i = a_0 e_i + a_1 e_{i-1} + a_2 e_{i-2} - b_1 m_{i-1} \]