1
Introduction

After reading this chapter the reader will:

1. have an idea on how we design mechatronic systems
2. know what are the phases of the design of such systems
3. have a clear idea on how to deal with each phase of the design of the mechatronic systems

The progress and the miniaturization we have seen in electronics during the last decades have allowed engineers to come up with new products and new engineering disciplines. Early in the eighteens we have seen the introduction of new products that combines mechanical parts with electronics parts. Another factor that gives a booming to mechatronics applications is the continuously decreasing prices of the electronic parts and the challenges to design very small systems. Today, for instance microprocessors with high performances are becoming very cheap which encourages their uses in computer controlled systems.

A microprocessor is an integrated circuit that contains the entire central processing unit of a computer on a single chip. The microprocessor is the main part in our nowadays computers. It does all the necessary computations and treats the data. The microprocessors have the following components:

- control unit
- arithmetic and logic unit
- input/output (I/O) data bus
- address bus
- internal registers
- clock
- etc.

To construct the computers, other peripherals and components are added to the main part which is the microprocessor. Screens, hard disk, floppies, memory, etc. are examples of such peripherals that we can have in our computers. For the computer controlled systems, we need appropriate cards known as data acquisition cards. These devices come with analog to digital (ADC) and digital to analog (DAC) converters and other necessary components real-time control applications. For some mechatronic systems, the use of computers and data acquisition cards are not appropriate and more often we use instead electronic circuit built around microcontrollers that can be considered as small microprocessor with their own peripherals.

A microcontroller is an integrated circuit as it is the case of the microprocessor and consisting of:

- a relatively simple central processing unit (CPU)
- memory
- a crystal oscillator
- timers,
- watchdog,
- serial and analog I/O
- pulse-width modulation (PWM) modules
- etc.

Microcontrollers are designed for small applications, while the microprocessors are used in high performance applications and personal computers. The Intel microprocessors that run in our laptops are examples of these microprocessors and the PICs of Microchip[1] are examples of microcontrollers. These machines are used in almost all the products that we use in our daily life. As examples that use microcontrollers, we quote:

- cars
- airplanes

[1] Microchip is a trademark, see www.microchip.com

- cellular phones
- digital cameras
- etc.

Nowadays most of the systems are computer controlled where the intelligence of these mechatronic systems is implemented in microcontrollers. The discipline that deals with such systems is mechatronics that we define as the synergistic combination of mechanical engineering, electronic engineering, and software engineering. The purpose of this interdisciplinary engineering field is to build and control complex systems by providing hardware and software solutions. The engineers working in this field must master concepts in electronics, control and programming. Examples of such systems can be found in industrial areas ranging from aerospace industry to car industry.

The design of mechatronic systems is a task that requires engineers from different disciplines like mechanical engineering, electrical engineering, control engineering, computer engineering, etc. The knowledge of these engineers are combined to produce the best mechatronic system. Most of these mechatronic systems are composed of:

- a mechanical part including the actuators and sensrors
- an electronic circuit that is built around a microcontroller or a set of microcontrollers
- a real-time implementation that represents the intelligence of the system

As example of mechatronic system, let us consider a laboratory setup for real-time implementation of control algorithms. This setup must have all the functionalities that allow learning real-time control. More specifically,

- the mechanical part must allow the user to check the output of the control algorithm
- an electronic circuit must be simple and easy to reproduce by the user in case
- the implementation must be easy to do and well documented.

In the rest of this chapter we will describe briefly each phase of the design of the whole mechatronic systems.

1.1 Mechanical Part Design

The mechanical part is a principle part in the mechatronic system. In the phase design of this part, we will conceive and manufacture the parts that compose the mechatronic system. We will also choose the actuators and the sensors we will use for this mechatronic system. Either the design of the mechanical part or the choice of the actuators and sensors are done by respecting some design rules that will be presented in a forthcoming chapter of the volume. It is also important to keep in mind

that the recycling of the mechatronic system once it becomes obsolete to respect our environment is an important matter that we must consider during the design phase. The assembly and disassembly of the system either for maintenance or any other purpose should be considered also during the design phase.

For the design of the mechanical part, the steps of the mechanical design such as definition of the problem, research of solution using brainstorming or any equivalent approach, practicability study, prototyping, etc. are used. The choice of the actuators and the sensors are also done by following the guidelines and the norms that are in use. As an example, if the mechatronic system is designed to operate in mines, electrical actuators are avoided since they may cause fires, while for food industries hydraulic actuators are excluded also.

For the setup of the real-time implementation that we are considering as example, the mechanical part in this case is only a small graduated disk (in degree) that will be attach solidly to the shaft of the actuators. This mechanical part is made from aluminium. The actuator is a dc motor that is equipped with a gearbox and an encoder. The role of the gearbox is to reduce the velocity of the mechanical part and also to apply a high torque. The encoder is used to measure the disk position and therefore, use this information for feedback. The whole is mounted on a plexiglass as it is shown in Fig. 1.1. More details on the conception of this mechanical part will be given in a forthcoming chapter of this volume.

1.2 Electronic Circuit Design

In the electronics part, the engineers must design the circuit that will assure the functioning of the mechatronics systems. It covers the integration of the required electronics parts such as resistors, capacitors, integrated circuits and the chosen microcontroller or microcontrollers. The required regulated voltages for the different components are also part of this step. The main part of the electronic circuit is the microcontroller or a set of microcontrollers. In this volume we decided to use one type of microcontroller which is the dsPIC30F4011 manufactured by Microchip. There is no real justification that we can give but only our desire is to adopt one microcontroller for all the examples we will cover in this volume. This choice will also make the real-time implementation easy for the reader since we will use the same structure for all the examples.

The regulated voltages will depend on the components we will use beside the microcontroller that requires following its datasheet a voltage between 2.5 V and 5 V. Since most of the examples use dc actuators and to drive them we need an analog signal that we can get either using a DAC or just PWM and an integrated circuit named L293D (a H-bridge). This integrated circuit needs a regulated voltage of 5 V and it will deliver a signal output that will feed the dc motor between 0V and 24V. We are also using many sensors that need regulated voltages to operate properly. Most of these devices need 5V exception made for the accelerometers and gyroscopes that requires a less regulated voltages (see the two wheels robot). For the

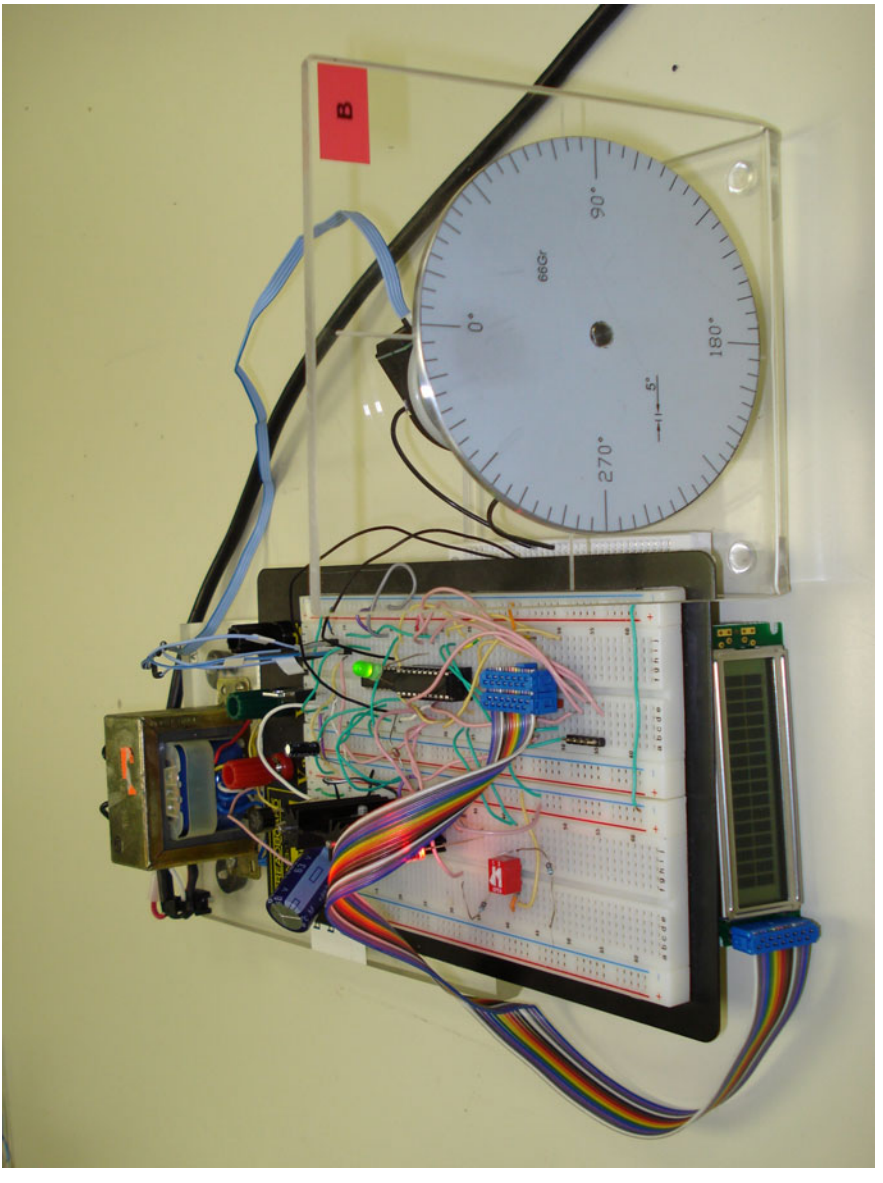


Fig. 1.1 Load driven by a dc motor kit

dc motor kit Figs. (1.1)-(1.2) give an idea of the electronic circuit of the dc motor kit that we will use in this volume.

To control the mechanical part two structures are possible. These structures are illustrated by Figs. 1.3-1.4.

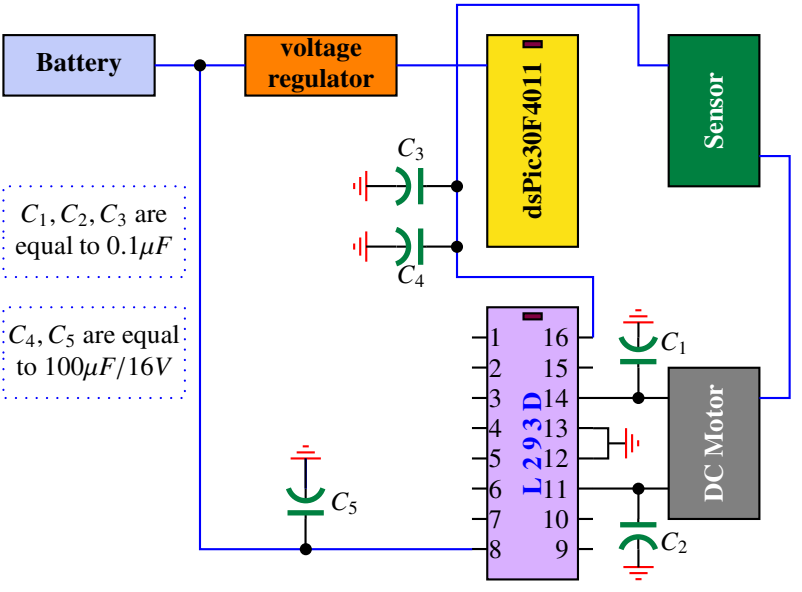


Fig. 1.2 Electronic circuit of the dc motor kit

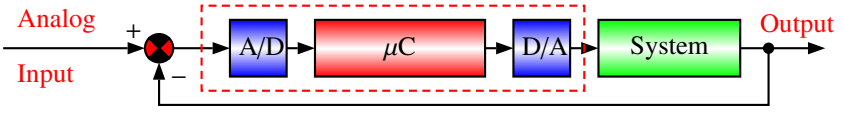


Fig. 1.3 Signal conversion made in the forward path

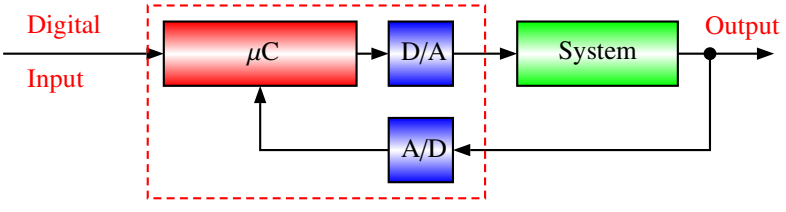


Fig. 1.4 Signal conversion made in the feedback path

If we compare these two structures, we remark that in the first one the references are analog while in the second one, they are digital. The second structure has the advantage that we can eliminate the noises. In the rest of this volume, we will adopt this structure.

The functioning of this structure is simple and it can be explained as follows. The microcontroller runs in indefinite loop and at each interrupt, the microcontroller reads the value of the output using the sensor and the ADC, then using the control

algorithm a control action is computed and sent to the system via the DAC. All these steps are done inside the interrupt routine. To avoid error calculation and error quantization the choice of number of bits either for the microcontroller or the ADC is an important issue. For the micrcontroller, a choice of 16 bits is done and this gives a good precision while for the ADC, a 10 bits will be used for all the examples we are presenting. This will not give a good precision but the results are acceptable.

If we go back to our real-time implementation setup, its electronic circuit is built around the dsPIC30F4011. The PWM module is used to deliver the voltage to the L293D integrated circuit that is in turn delivers the necessary power to drive the actuator. An encoder is used to measure the position of the small disk and also the velocity by simple calculations.

1.3 Real-Time Implementation

In the control part, the engineer must analyzes the system under study and design the appropriate controller to get the desired performances. In the analysis part, we should start by establishing an acceptable model that gives the relationship between the inputs and the outputs. Once the dynamics is mastered a sampling period is chosen and the model is converted to a discrete-time form and an appropriate controller can be chosen among the classical proportional integral and derivative (PID) controller or the state feedback controller or any other controller that can give the desired performances. To respond to the control specifications, a controller structure and its parameters are computed, then a recurrent equation is established for the determination of the control action that we must send at each sampling period to the system.

In the programming part, the engineer enters the algorithms of the chosen algorithm in the memory of the microcontroller. Many languages can be used for this purpose. In the rest of this volume, the C language is used to implement the developed algorithms.

Again if we go back to our real-time implementation setup and consider the case of two simple algorithms the PID controller and the state feedback controller. For these controllers the control action is computed using the measurement, the references, etc. In all the cases, the expression of the control law is simple and should not take a time that exceeds the sampling period (see Fig. 1.5). The implementation is done using the interrupt concept. The following example shows how the position of the load is controlled.

```
//
// A C program for the dsPic4011 for control the position of a
// dc motor driving a small disk
//

//
//  Includes and defines
//
#include <p30f4011.h>
```

```
#include <pwm.h>
#include <stdio.h>
#include <stdlib.h>
#include "xlcd.h"

#define ENABLETRIS  TRISEbits.TRISE2
#define ENABLE      LATEbits.LATE2

#define ENCODER_PRIORITY     7
#define CONTROLLER_PRIORITY  5
#define DISPLAY_PRIORITY     2

#define    Ts    0.005; // 1.0/200;
#define    Fs    200.0;

typedef struct {
     float KP;     // Proportional gain
     float KI;      // Integral gain
     float KD;    // Derivative gain
} PIDstruct;

PIDstruct thePID;

typedef struct {
     long Position;      // Shaft position
     long error[3];       // the errors
     long ref;               // the reference

     double u[2];        // control (actual and past)
}motorData;

motorData themotorData;

//
// dsPic configuration
//
_FOSC(CSW_FSCM_OFF & FRC_PLL16);;
_FWDT(WDT_OFF);
_FBORPOR(PBOR_OFF & MCLR_DIS);
_FGS(CODE_PROT_OFF);
_FICD( ICS_NONE );

//
//  Variables
//

typedef enum _BOOL { FALSE = 0, TRUE } BOOL;

BOOL    A;
```

```
BOOL    B;
BOOL prevA;
BOOL prevB;

unsigned int dutycycle;


//
//  Functions
//

// Initialization function
void Initialize(void);

// Interrupt functions
void __attribute__((interrupt, auto_psv)) _CNInterrupt(void);
void __attribute__((__interrupt__)) _T1Interrupt(void);

//
//  Main function
//
int main(void)
{
    Initialize();

    themotorData.ref = 600;     // (90 deg)

   while(1);
}

//
//  Initialize  function
//
void Initialize(void)
{
    // variables initialization

    thePID.KA = 70.14;
    thePID.KI = -128.62;
    thePID.KD = 58.54;

    themotorData.u[0] = 0.0;
    themotorData.u[1] = 0.0;
    themotorData.error[0] = 0;
    themotorData.error[1] = 0;
    themotorData.error[2] = 0;

    // Activation of the interrupts priority
    INTCON1bits.NSTDIS = 0;
```

```
    // Digital pins
    ADPCFG = 0b11111111;

    // I/O
    TRISEbits.TRISE0 = 0;   // PWM1H
    TRISEbits.TRISE1 = 0;   // PWM1L
    TRISBbits.TRISB2 = 1;   // Encoder Chanal A : RB2 -- CN4
    TRISBbits.TRISB3 = 1;   // Encoder Chanal B : RB3 -- CN5
    ENABLETRIS = 0;

    /* start-up LCD */
    OpenXLCD(FOUR_BIT & LINES_5X7);


    //
    // initialize variables for the encoder
    //
    prevA = PORTBbits.RB2;
    prevB = PORTBbits.RB3;
//
// Initialize CN interrupts *
//

    CNEN1bits.CN0IE=0;              // CN0 interrupt disable
    CNEN1bits.CN1IE=0;              // CN1 interrupt disable
    CNEN1bits.CN2IE=0;              // CN2 interrupt ENABLE
    CNEN1bits.CN3IE=0;              // CN3 interrupt ENABLE
    CNEN1bits.CN4IE=1;              // CN4 interrupt disable
    CNEN1bits.CN5IE=1;              // CN5 interrupt disable
    CNEN1bits.CN6IE=0;              // CN6 interrupt disable
    CNEN1bits.CN7IE=0;              // CN7 interrupt disable
    CNEN2bits.CN17IE=0;            // CN17 interrupt disable
    CNEN2bits.CN18IE=0;            // CN18 interrupt disable

    IFS0bits.CNIF = 0;          // clear CN interrupt flag
    IPC3bits.CNIP = ENCODER_PRIORITY;  // CN interrupt max priority (7)
    IEC0bits.CNIE = 1;          // CN interrupt enable

//
// Configure  PWM
//
    ConfigIntMCPWM(PWM_INT_DIS & PWM_FLTA_DIS_INT);

    SetDCMCPWM(1, 1024, 0);

    OpenMCPWM (0x3FF, 0x0, PWM_EN & PWM_IDLE_CON & PWM_OP_SCALE1
                              & PWM_IPCLK_SCALE1 & PWM_MOD_FREE,
                  PWM_MOD1_COMP & PWM_PDIS3H & PWM_PDIS2H & PWM_PEN1H
```

```
                    & PWM_PDIS3L & PWM_PDIS2L & PWM_PEN1L,
                         PWM_SEVOPS1 & PWM_OSYNC_TCY & PWM_UEN);

//
// Initialize Timer 1 interrupt
//

    T1CONbits.TON=1;            // turn timer 1 on
    T1CONbits.TGATE=0;
    T1CONbits.TSIDL=0;          // stop timer in idle mode (0=non)
    T1CONbits.TCKPS=1;          // prescaler (0=1:1, 1=1:8, 2=1:64)
    T1CONbits.TCS=0;            // clock source (0=FOSC/4)
    PR1 = 18424;                // 200Hz
    IFS0bits.T1IF = 0;          // clear timer 1 interrupt flag
    IPC0bits.T1IP = CONTROLLER_PRIORITY;
    IEC0bits.T1IE=1;            // enable timer 1 interrupt

//
// Initialize Timer 2 interrupt
//

    T2CONbits.TON=1;            // turn timer 2 on
    T2CONbits.TGATE=0;
    T2CONbits.TSIDL=1;          // stop timer in idle mode (0=non)
    T2CONbits.TCKPS=2;          // prescaler (0=1:1, 1=1:8, 2=1:64)
    T2CONbits.TCS=0;            // clock source (0=FOSC/4)
    PR2 = 0xFFFF;           // slower possible
    IFS0bits.T2IF = 0;          // clear timer 2 interrupt flag
    IPC1bits.T2IP = DISPLAY_PRIORITY;
    IEC0bits.T2IE = 1;          // timer 2 interrupt enable
}

//
//     C N  Interrupt routine
//

// Decode of the position
void __attribute__((interrupt, auto_psv)) _CNInterrupt(void)
{
    if(IFS0bits.CNIF)
    {
      CNLED = !CNLED;

      // Get current Encoder signals
      // Must read port before clearing flag!!
           A = PORTBbits.RB2;
           B = PORTBbits.RB3;

      // Compare current signals with previous ones to see which
```

```
      // one has changed

      // Change occurs on A
      if(A != prevA){
        if(A == B){
                        themotorData.Position++;
        }else{
                        themotorData.Position--;
        }
      // Change occurs on B
      }else if(B != prevB){
        if(A == B){
                          themotorData.Position--;
        }else{
                         themotorData.Position++;
        }
      }

      // Save current signals for next time
      prevA = A;
      prevB = B;

      IFS0bits.CNIF=0;    // clear interrupt flag
    }

}    //end of CN_interrupt function

//
// T 1  Interrupt service routine
//

// Sampling period
void __attribute__((__interrupt__)) _T1Interrupt(void)
{
 if (IFS0bits.T1IF)
{
// Error
themotorData.error[0] = themotorData.ref - themotorData.Position;

// Control equation
themotorData.u[0] = themotorData.u[1] + thePID.KA*themotorData.error[0];
themotorData.u[0] += thePID.KI*themotorData.error[1];
themotorData.u[0] += tthePID.KD*themotorData.error[2];

// send control
SetDCMCPWM(1, 1024 + (int)(themotorData.u[0]), 0);

// save the actual data
themotorData.u[1] = themotorData.u[0];
```

```
themotorData.error[2] = themotorData.error[1];
themotorData.error[1] = themotorData.error[0];

IFS0bits.T1IF = 0;    // Clear Timer interrupt flag
  }
}

//
// T 2  I N T E R R U P T service routine
//

// LCD
void __attribute__((interrupt, auto_psv)) _T2Interrupt(void)
{
    if (IFS0bits.T2IF)
    {
        while(BusyXLCD());
        XLCDLine1();
        printf("e: %ld", themotorData.error[0]);
        while(BusyXLCD());
        XLCDLine2();
        printf("u: %8.3f ", themotorData.u[0]);

        IFS0bits.T2IF = 0;
    }
}
```

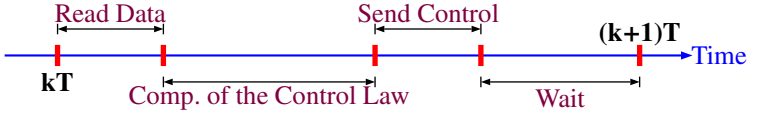


Fig. 1.5 Partition of the sampling period T

Example 1.3.1 *As a second example of mechatronic system, let us consider the design of a traffic light control system. We suppose that we have two streets, a main one with 80 % of the traffic while the other one has 20 % of the traffic. Fig. 1.6 illustrates the traffic light system we are dealing with and for which we should design the mechatronic system. Our goal is to design a mechatronic system that controls the traffic flow for these two streets. More specifically, we must control the lights (red, yellow and green) in each street. Most of the common traffic lights around the world consists of three lights, red, yellow and green. Fig. 1.7 gives an idea of the light used in our traffic system. In each corner of the traffic system we place a light in order that the pedestrian and the driver can see the light and take the appropriate action.*

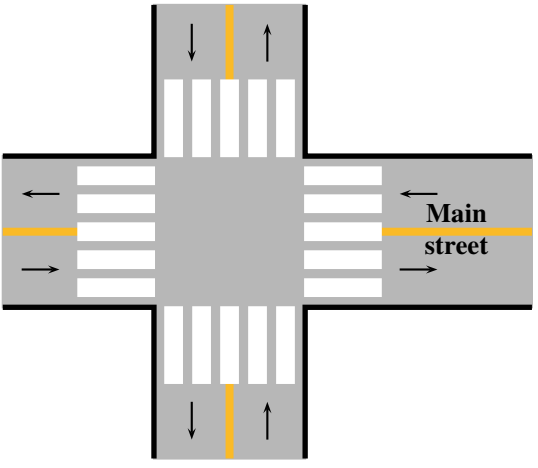


Fig. 1.6 Traffic system

When the light turns to red, the drivers must stop their car, while when it turns to green, the drivers have the right to move their car. The yellow light is used as a cautious step indicating either that the light is about to turn to green or to red and the drivers must take the appropriate actions either move or stop their cars. More often the yellow is used when the light is about to switch from green to red as an intermediate step that takes short time.

Each street is divided into two ways for two directions and each way has two lanes. The cars can either go straight or turn left or right in each way. We have also in each intersection to control the requests of the pedestrians. These requests are random and must be taken into account in a short time with a certain priority.

The mechatronic system for the traffic light is a simple system and it is composed of:

- *lights that are located at each corner of the streets with some push buttons for pedestrians to request permission to cross the street*
- *an electronic circuit built around a dsPIC30F4011*
- *an algorithm in C language for control*

The lights that control the traffic are placed at each corner of the street. The type of these lights is shown in Fig. 1.7. The push bottoms are also placed to help the pedestrians to cross the street when it is needed in safe way.

To simulate our traffic light we represent lights by colored light-emitting diode (LED) using the same colors as in the traffic light control system. For pedestrian we use the blue color.

The algorithm we will use for the control of the flow traffic is very simple and it is executed in a sequential manner except for the requests of pedestrians that are treated as interrupts routines. If we denote by Gmain, Ymain, Rmain, Gsec,

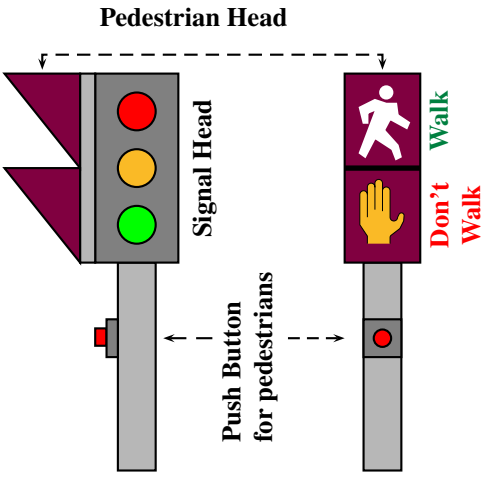


Fig. 1.7 Type of light used in the traffic light system

Ysec, Rsec the light green, yellow and red respectively for the main street and the secondary streets.The algorithm is as follows:

Begin loop

- *put Gmain on, Ymain off, Rmain off, Gsec off, Ysec off and Rsec on, and wait for a time tmain*
- *put Gmain off, Ymain on, Rmain off, Gsec off, Ysec off and Rsec on, and wait for a time tswitch*
- *put Gmain off, Ymain off, Rmain on, Gsec on, Ysec off and Rsec off, and wait for a time tsec*
- *put Gmain off, Ymain off, Rmain on, Gsec off, Ysec on and Rsec off, and wait for a time tswitch*

End loop

When an interrupt occurs, we identify on which corner the pedestrian pushed the button and act in consequence by stopping the traffic of the cars to allow the pedestrian to cross the street in a safe way.

The structure of the program used for the control light system is given by:

```
// Include here the headers
#include <dspic30f4011.h>

// Define variables

unsigned int i;
unsigned int Tmax = 65535;
unsigned int tmain = 8;
unsigned int tsec = 4;
unsigned int  tswitch = 1;
```

```
#define delaytmain() {for i=0;i<tmain*Tmax;i++) Nop(); }
#define delaytsec() {for i=0;i<tsec*Tmax;;i++) Nop(); }
#define delaytswitch() {for i=0;i<tswitch*Tmax;;i++) Nop(); }

#define greenMain RE0    // green light of the main street
#define yellowMain RE1   // yellow light of the main street
#define redMain RE2        // red light of the main street

#define greenSecondary RE3  // green light of the secondary street
#define yellowSecondary RE4  // yellow light of the secondary street
#define redSecondary RE5  // red light of the secondary street

typedef enum _BOOL { FALSE = 0, TRUE } BOOL;
BOOL A;
BOOL prevA;

// Initialization of the streets

// Main street
MainStreet.green = TRUE;
MainStreet.orange = FALSE;
MainStreet.rouge = FALSE;

// Secondary street
SecondaryStreet.green = FALSE;
SecondaryStreet.orange = FALSE;
SecondaryStreet.rouge = TRUE;


//  Assign the dsPic ports to the lights

//
// Functions
//

void Initialize(void);

void __attribute__((interrupt, auto_psv)) _CNInterrupt(void)

//
// main function
//
int main (void)
{
Initialize();
while (1)
{
//            tmain
```

```
// Main Street during the tmain
greenMain = 1;
yellowMain = 0;
redMain = 0;

// Secondary street during the tmain
greenSecondary = 0;
yellowSecondary = 0;
redSecondary = 1;

delaytmin();
//            tswitch
// Main Street during the tswitch
greenMain = 0;
yellowMain = 1;
redMain = 0;

// Secondary street during the tswitch
greenSecondary = 0;
yellowSecondary = 0;
redSecondary = 1;

delaytswitch();
//             tsec
// Main Street during the tsec
greenMain = 0;
yellowMain = 0;
redMain = 1
;
// Secondary street during the tsec
greenSecondary = 1;
yellowSecondary = 0;
redSecondary = 0;

delaytsec();

//            tswitch
// Main Street during the tswitch
greenMain = 0;
yellowMain = 0;
redMain = 1;

// Secondary street during the tswitch
greenSecondary = 0;
yellowSecondary = 1;
redSecondary = 0;
delaytswitch();
}
}
```

```
void Initialize(void)
{

TRISE = 0x00     // configure the port E  as output

//
// initialize variables for the encoder
//
prevA = PORTBbits.RB2;
//
// Initialize CN interrupts *
//

CNEN1bits.CN0IE=0;              // CN0 interrupt disable
CNEN1bits.CN1IE=0;              // CN1 interrupt disable
CNEN1bits.CN2IE=0;              // CN2 interrupt ENABLE
CNEN1bits.CN3IE=0;              // CN3 interrupt ENABLE
CNEN1bits.CN4IE=1;              // CN4 interrupt disable
CNEN1bits.CN5IE=1;              // CN5 interrupt disable
CNEN1bits.CN6IE=0;              // CN6 interrupt disable
CNEN1bits.CN7IE=0;              // CN7 interrupt disable
CNEN2bits.CN17IE=0;            // CN17 interrupt disable
CNEN2bits.CN18IE=0;            // CN18 interrupt disable

IFS0bits.CNIF = 0;          // clear CN interrupt flag

IPC3bits.CNIP = ENCODER_PRIORITY;     // CN interrupt max priority (7)

IEC0bits.CNIE = 1;          // CN interrupt enable
}

//
//     C N  Interrupt routine
//

// Pedestrian ask to cross
void __attribute__((interrupt, auto_psv)) _CNInterrupt(void)
{
if(IFS0bits.CNIF)
{
CNLED = !CNLED;

// Get the switch signal
// Must read port before clearing flag!!
A = PORTBbits.RB2;

// Compare the current signal with the previous signal to see the change
// Change occurs on A
```

```
if(A != prevA){
// put all the red lights on
}

// Save current signal for next time
prevA = A;

IFS0bits.CNIF=0;    // clear interrupt flag
}

}    //end of CN_interrupt function
```

The program starts by initializing all the variables and also configure the inputs and outputs of the dsPIC30F4011. After this, the program enters in indefinite loop in which we execute the sequence that controls the light for the intersections. If a pedestrian asks for the permission to cross the street, we shorten the time for actual activity since we can not stop abruptly the activity to prevent accidents. A given time is allocated for the pedestrian to cross the street. Once this time is finished, the sequence in the loop is resumed.

Remark 1.3.1 *For pedestrians, there is also the possibility to include right to cross the streets in the sequences that we have to execute in the program. Also late in the night, we can eliminate the rights for pedestrians since there is a small probability that a pedestrian will be at the corner and he will cross the street. But with the interrupts solution, it is possible to keep the same algorithm for all the time and we don't have to change it.*

We can improve our algorithm to make it more intelligent by adding appropriate sensors that memorize the queues in each street and act appropriately by adjusting the time of the lights in each street to reduce the waiting time of the drivers in the traffic light.

These two examples give an idea on mechatronic systems and how they can be difficult and complex to design. It is important to notice that the solution for a given mechatronic system is not unique and it varies with the knowledge of the design team. It is also important to keep in mind that the optimization should be used during the phases to obtain a competitive system.

1.4 Organization of the Book

This book can be considered as second course in mechatronics curriculum where the students are supposed to have a prerequisite course in which the structure and the different components on mechatronic systems have been presented. It focuses only on the analysis, design and implementation of continuous-time systems controlled by microcontrollers using advanced algorithms to get the desired performances.

In the modeling part a model to describe the behavior of the systems is developed either using the transfer function or the state space representation.

In the transfer function approach part, the model of the continuous-time systems is converted to a discrete-time system and different techniques for analysis and synthesize of controllers to guarantee some desired performances are developed.

In the state space approach part, the model of the continuous-time systems is converted to a discrete-time state space representation and different techniques for analysis and synthesize of controllers to guarantee some desired performances are developed.

The part on implementation will focus on how we can implement the control algorithms we developed either in the part on transfer function approach or the one based on state space. Both the hardware and software parts will be covered to give an idea on the reader on how to deal with such problems.

In the part of advance control, some algorithms that can be used to control systems with uncertainties and/or external disturbances are presented to give a flavor to the reader on the robust control theory and introduce him to the research in this field.

In the case studies part, a certain number of practical examples are presented to show how the concepts we presented earlier are implemented to obtain a functional mechatronic systems.

Part I

Mechatronic Systems

2
Mechatronic Systems

After reading this chapter the reader will:

1. master the concepts of Mechatronics and Mechatronic systems
2. be able to execute each phase in the design of mechatronic systems
3. be capable to design the mechanical part, the electronic circuit and to compute the control law and implement it in real time
4. be able to write a program in C language and how to insert it in the dsPIC30F4011

2.1 Mechatronics

Let us examine the design of an autonomous car which may be used for navigating in the floor of a building, and to move from different offices in the same floor. There are two main approaches for achieving the design of this autonomous car. The first design approach follows the classical design method. In this approach, the mechanical design is done first. After getting a satisfactory mechanical design, the electronic system is designed. In the final stage, the control system is designed.

The second design approach is to design the AC while observing the effects of each system on the overall design. In this approach, the design of the mechanical, electronic and control systems of the autonomous car are designed while taking the interaction of the design of each system and its effects on the other two systems. So, the mechanical system is finalized only after studying the effects of such design on the electronic design, and on the control system design. In this approach, the interaction between the three systems (mechanical, electronic, and control) and their effects on the final design and performance are taken into consideration at every step of the design of each system. The benefits of the second design approach are very obvious. One of the main benefits is the possibility for making the best design of each system that will make the best overall performance. This is not possible in the classical design approach because once a mechanical system is designed, it will be the final mechanical design. Also, once a mechanical design (such as the materials used, the size of the design) is decided, it will dictate and may limit the available alternatives in the electronic system (the size of the motors used, the location of the motors etc.) design which in its turn will also limit the alternatives for the instrumentation and the control system used for the overall system. The interactions between the design of the three systems (mechanical, electronics, and control) is what mechatronics offers for a better design.

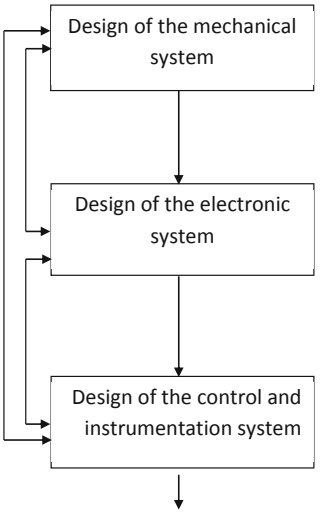


Fig. 2.1 Mechatronic design approach

Figure 2.1 presents the mehcatronic design approach in a simple way. The one-sided arrows indicates the flow of the design process, while the two-sided arrows represent the interactions between the designs of the different systems of the final product. The two-sided arrows represents the need to think about the overall design

at any point in time in the design process and at any system being designed. They mean that at any step, the design could impact the design of the other systems.

If the two-sided arrows are removed, we get the classical design approach where no interactions exist.

In the literature, there exist many definitions that have been proposed for mechatronics. These definitions depend mainly on the vision and the research interests on the field by a person or a group of persons working in some directions of mechatronics. In our vision mechatronics can be defined as an interdisciplinary field of engineering that deals with the design of products whose function relies on the integration of mechanical, electrical, and electronic components connected by a control scheme.

Nowadays, the word mechatronics is worldwide known and many mechatronic systems where designed either for personal or commercial uses. All the gadgets we are using in our daily life are in fact mechatronic systems. As an example of these gadgets, we mention our laptop and our car where many mechatronic systems are used.

A mechatronic system can be seen as a device that is able to perceive the surrounding environment and take the appropriate decisions based on the collected information. To perceive the surrounding environment sensors are needed and that without these devices the mechatronic system can not perform their tasks for which it was designed. Nowadays, cars possess many mechatronic systems to assist the driver in a safe drive among then we quote:

- airbag
- ABS brake
- speed control
- etc.

Also, to take the appropriate action, the mechatronic system needs a smart algorithm that gives actions to some appropriate actuators which can be simple switches, dc motors, stepper motors, ac motors, hydraulic motors or pneumatic motors to position for instance the mechanical part that we would like to control.

The intelligence of the mechatronic system is programmed as an efficient algorithm that coordinates all the task of the used devices. This algorithm runs in general on a powerful microcontroller.

The design of a mechatronic system is a hard task that needs interdisciplinary engineers that can understand the different elements of the system. The main components of each mechatronic system are:

- the mechanical part
- the sensors
- the actuators

- the electronic circuit
- and the program

As examples of mechatronic systems we quote

- the position control of a dc motor
- the balancing robot
- the mobile robot
- and the magnetic levitation

These systems will be used extensively in this volume to illustrate the important concepts we will cover. Before presenting these examples, let us focus on the main parts of the mechatronic systems and give some guidelines on how to design or to select them.

2.2 Mechanical Part

The mechanical part represents the main component in the mechatronic system we are trying to design. It can either be manufactured or built from existing components. In the phase design of this part more care should be paid to the following points:

- the price
- the shape
- the weight
- the size
- etc.

It is also important to pay attention to the environment in which the mechatronic system will operate. This will help us to select the appropriate material from which the mechanical part should be made. The maintenance of the mechatronic system is also a critical point, it is why we should pay attention during the design phase to the accessibility of all critical parts of the system. It is also important at this stage to think about the recycling of all the mechatronic system once it will be useless to respect the environment that we need to protect for our new generations.

The mechanical part can be made from iron, aluminium, plastic, composite or any other material. The choice of one of these materials or a combination of them will depend on many factors such the environment in which the system will operate, the weight, the task for which the mechanical part is designed for, etc.

During the design of this part care should paid also to the look of the pieces and to the fact that other parts of the mechatronic system have to be integrated later such as sensors, actuators, electronic circuit, etc. The assembly or the disassembly of all the

system should be simplified such that everybody can assemble of the disassemble the system when it is needed.

2.3 Sensors

The sensors are the key points in each mechatronic system. There are in some sense the eyes of the system through which all the type of variations are detected and the appropriate actions are taken. A sensor can be seen as a device that converts a physical phenomena like position, velocity, acceleration, temperature, flow, etc. into an electrical signal that can be easily measured or processed. A sensor is composed of a transducer and a signal conditioning unit. Nowadays, for some phenomena there exist many sensors that can be used to sense them which makes the choice very hard. Selecting a sensor for an appropriate application is always a difficult task even for experienced person in the field. The engineer must take into consideration the following points during the choice of any sensor:

- the error/precision/resolution
- the range or span
- the nonlinearity
- the repeatability
- the hysteresis
- the stability/drift
- the bandwidth
- the reliability
- the cost
- the ease of utilization

Nowadays there are a lot of type of sensors that can be categorized based on their applications or their theory of operations. Among the most used sensors in the mechatronic systems we quote:

- the encoders
- the accelerometers
- the gyroscopes
- and the cameras (image sensors)

An encoder can be defined as a device that assigns a code to represent some data. More specifically, it can be seen as a sensor or a transducer for converting rotary motion or angular position to series of electronic pulses that are appropriate for computer use. The pulses are counted and the value of the measured input is

deduced. The existing encoders in the market are either absolute or incremental. They are used in many applications among them we quote:

- the position control
- robots
- CNC machines
- medical equipment such as MRI, CT-Scan and PET-Scan machines
- etc.

The absolute encoder is mainly composed of an optical disk that has a number of tracks that gives a digital word depending of the position of the shaft. As an example, if we consider a disk with 8 tracks, in this case the encoder has 256 distinct positions, which gives an angular resolution of 1.4 degrees. The Gray and the binary codes are commonly used in the absolute encoder.

The incremental encoder is simpler compared to the absolute encoder and it consists of two tracks and two sensors that give two channels A and B. When the shaft of the sensor rotates pulse trains appear on the two channels that are quadrature signals. These signals can be used to determine the angular position and the rotation sense. A third output channel referred to as *Index* that produces a pulse by revolution and it is used to count the number of revolutions.

The accelerometer is a device that can be used to detect the acceleration and tilt. Nowadays accelerometers are used in cars for passenger security. Their role is to detect the impact and deploy the car airbag when it is necessary to save the life of the passengers. These type of sensors found use in digital cameras where their role is to guarantee the stability of the image.

Nowadays, the accelerometer comes in MEMS technology. The MEMS accelerometer usually comes in the smallest surface mount package and can detect acceleration in up to 3 axis. The data from this accelerometer can directly be used by the microcontroller and therefore take the appropriate action when it is required. The accelerometer can be used to measure the acceleration of the object or measure the tilt of the object to which the sensor is attached to.

The gyroscope can be seen as a device consisting of a rotating wheel mounted on a base so that its axis can turn freely in certain or all directions, and capable of maintaining the same absolute direction in space regardless of any movement of the base. This device is used in airplanes, satellites, robots, etc. Nowadays, the gyroscopes come in MEMS technology which facilitates their application in mechatronic systems.

The cameras (image sensors) can be seen as complementary metal oxide semiconductor (CMOS) or charge-coupled device (CCD)-based chips that record the intensities of light as variable charges. The cameras contains millions of pixels arranged in a matrix which catches and records light when a picture of an element is taken. The cameras are used extensively in image processing for quality control, supervision, etc.

2.4 Actuators

Actuators are defined as devices that convert some kind of power, such as hydraulic or electric power, into linear or rotary motion. They represent the arms of the mechatronic systems. In practice different type of actuators are used, among them we quote:

- electric actuators
- hydraulic actuators
- and pneumatic actuators

An electric actuator is a device that convert electric power into a linear or a rotary motion. They are used to position or to give the speed to the mechanical part of the mechatronic system. The common electric actuators are:

- the dc motors
- the ac motors
- the stepper motors
- and the switches

These actuators own the following advantages:

- high speeds
- self contained
- low cost
- simple design
- reliable operation (less maintenance)
- high efficiency
- long life

The dc motors beside being cheap and simple are easy to control in speed, position and torque. While their homolog ac motors are in general expensive in speed control, show some instability in operating at low speeds and own poor positioning control.

Electric actuators are in general precise and flexible. They are ideal to position mechanical part precisely or to develop forces quickly when it is required. Their major disadvantage is that they need cooling systems during their operation. When they are well designed and well protected, their maintenance is reduced to the changes of the sliding contacts or the commutators. Large load may burn the winding of the electric actuators if the protection is not installed properly.

Stepper motors are more appropriate to control mechanical parts that don't require feedback. Mostly these type of actuators are used in open loop control and to position the mechanical part. For this purpose a certain number of pulses are sent

by the microcontroller. These actuators are used in laser printers, faxes, and most of the appliances for computers.

A hydraulic actuator can be defined as a cylinder or fluid motor that converts hydraulic power provided by a pump into a useful mechanical work. The mechanical motion that results may be linear, rotary, or oscillatory. This type of actuator provides the following advantages:

- high dynamic response
- high force capability
- high power per unit weight and volume
- good mechanical stiffness

while the disadvantages are:

- leakage
- need more maintenance (filers)
- need external hydraulics pump

These features lead to wide use in precision control systems and in heavy-duty machine tool, mobile, marine, and aerospace applications.

The pneumatic actuator is defined as a device that uses pressurized air to create mechanical motion (linear or rotary). Similarly to the hydraulic actuator, this one also requires a compressor for air to operate. It is also important to mention that the efficiency of this kind of actuators is low. The pneumatic actuators are in general inexpensive and their operations are not affected by difficult environmental factors such as dust, etc. and they are easy to install and operate. They have less precision compared to the other actuators due the compressibility of the air. Pneumatic actuators are appropriate for use in potentially explosive environments. Contrary to the electric actuators, the pneumatic ones can support large loads and don't require the cooling system.

Selecting an actuator for an appropriate application is always a difficult task even for experienced engineers in the field, meanwhile main guidelines should be kept in mind. In fact the power, the environment of operation are main points to be considered and can help in choosing the type of actuators. For instance, if the mechatronic system is designed to operate in mining where sparks may cause fire, the electric actuators are excluded and the hydraulic actuators are possible solutions. In food industry, the hydraulic actuators are excluded and electric or pneumatic actuators are the possible solutions.

2.5 Electronic Circuit

The electronic circuit is the brain of the mechatronic systems. It regroups passive and active components beside integrated circuits. Its role is to manage and coordinate in a desired way the functioning of all the components that compose the system.

The passive components include resistors and capacitors, while the active ones can be a simple diode or a transistor or any integrated circuit that performs the desired task. The electronic circuit manages and orchestrates a variety of functions that the mechatronic system allows beside providing the desired regulated voltage for the different integrated circuit, the sensors, the actuators and the micrcocontroller.

When designing the electronic circuit we must keep in mind that the size of the circuit and its consumption in power should be minimized. The safety of the circuit and its cooling are also of importance. In case of manipulating high voltage security rules should be followed seriously.

2.6 Real-Time Implementation

Once the hardware part of the mechatronic system is built, the next step is to design the control algorithm that we should implement to guarantee that the system will perform properly the tasks for which it was designed for. The design of such algorithm is done into two steps. The first one consists of establishing the mathematical model that describes properly the relationship between the inputs and the outputs of the system. This model can be determined either analytically with some limited experiments to the values for some parameters, or experimentally using the identification techniques. In the second step, the desired performances are fixed and the controller is designed using the appropriate techniques. The results of this step is the determination of the recurrent equation that will compute the decision at each interrupt. This equation represents the algorithm that we have to implement in the microcontroller.

The microcontroller is used to provide real time response to the different events for which the system is designed for. In general is running in a loop and when an event occurs, the associated interrupt system alerts the processor to suspend processing of its current instruction and to start an interrupt service routine. This interrupt routine executes the main steps of the control algorithm that we are using. Once the task of the interrupt service routine is completed, the processor returns to the place where the execution were suspended.

The implementation is done following the following structure:

```
//
// Put here title and comments
//
#include "p30F4011.h"       // proc specific header

//
// Define gobal variables in RAM
//
float  Reference;   // simple variable
int variable0;         // (16 bits)
char myVariable; // (8 bits)
```

```
#define n1 10   /* sample constant definition */
#define n2 20;
int array1[n1] __attribute__((__space__(xmemory), __aligned__(32)));
               // array with dsPIC30F attributes
int array5[n2];     // simple array
int variable3 __attribute__((__space__(xmemory)));
                // variable with attributes
int array1[n1] __attribute__((__space__(xmemory), __aligned__(32)));
                // array with dsPIC30F attributes
int array5[n2];       // simple array
int variable4 __attribute__((__space__(xmemory)));
                // variable with attributes

 //
 // Define a struct
 //
  typedef struct {
       // PID Gains
       float KP;      // Propotional gain
       float KI;       // Integral gain
       float KD;     // Derivative gain

    //
    // PID Constants
    //
    float Const1_pid;  // KP + T KI + KD/T
    float Const2_pid;  // KP + 2KD/T
    float Const3_pid;  // Kd/T
    float Const4_pid;  // KP + KD/T
    float Const5_pid;  // T KI

    //
    // System variables
    //
    float y_c;     // y_c[k] -> controlled output
    float y_m;    // y_m[k] -> measured output
    float u_k;     // u[k]   -> control at time k
    float e_k;     // e[k]   -> error at time k

    //
    // System past variables
    //
    float u_km1;     // u[k-1] -> output at time k-1
    float e_km1;     // e[k-1] -> error at time k-1
    float e_km2;     // e[k-2] -> error at time k-2
    float y_mkm1;  // y_m[k-1] -> measured output at time k-1
    float y_mkm2;  // y_m[k-2] -> measured output at time k-2

  }PIDStruct;
```

```
  PIDStruct thePID;

//
// Constants in ROM
//
const char Variable_Rom[] = {1,2,3,4};
const int myConstant = 100;

//
// Non memorized constants
//
#define var1 0x1234;
#define var2 "ma chaine";

//
// Functions
//
float my_Function(float a, float b)
    {
       int local_var;

       local_var = a - b;
       return local_var;
    }


//
// Interrupt program here using Timer 1 (overflow of counter Timer 1)
//
void __ISR _T1Interrupt(void)     // interrupt routine code
  {
       // Interrupt Service Routine code goes here
       float Position_error;

       // get the actual position from the encoder
       // ThePID.y_m

       Position_error = my_Function(Reference, ThePID.y_m);
       .......

       IFS0bits.T1IF=0;    // Disable the interrupt
  }


int main ( void )                 // start of main application code
{
// Application code goes here
int i;
```

```
// Initialize the variables Reference and ThePID.y_m
(it can be read from inputs) Reference = 0x8000; // Hexadecimal number
(0b...  Binary number) ThePID = 0x8000;

// Initialize the registers
TRISC=0x9fff;  // RC13 and RC14 (pins 15 and 16) are configured as
outputs IEC0bits.T1IE=1;   // Enable the interrupt on Timer 1

// Infinite loop
while (1)
{
}
return 0
}
```

2.7 Examples of Mechatronic Systems

The aim of this section is to present some mechatronic systems that may be used in the rest of this volume to show the different concepts we will develop. We will try to present all the parts of these mechatronic systems to help the reader to make a clear idea on the design of mechatronic systems and hope that this will help him to design his own system in the future.

We will restrict ourself to mechatronic systems that use common components like electric actuators, encoders, accelerometers, gyroscopes, etc.

2.7.1 Dc Motor Control

As a basic mechatronic system, let us design a setup that can be used either for speed or position control. This system will be the basis of almost all the coming mechatronic systems. The system we will present here consists of a dc motor that drives via a gear a small disk. In order to control it properly either in speed or in position an incremental encoder is used.

The mechanical part of this system is a small disk that is manufactured in our mechatronics laboratory. Graduations are indicated on the disk to help us to position it at any desired position we want. The disk is made from aluminium and attached solidly to the motor shaft using a screw.

The actuator is a small dc motor that we bought from a surplus store. It has already a gear (ratio is 1 : 6) and an incremental encoder (100 pulses/rev). The electronic circuit of this system is too simple and it can be summarized to:

- a transformer
- two voltage regulators (T78012 and T7805)

- resistors (2 resistors of 10 KΩ, 2 resistor of 220 Ω and a variable resistor of 20 KΩ and capacitors (3 of 0.1 μF)
- diodes
- an H bridge
- a liquid crystal display (LCD)
- switch (to put the system on or off)
- a microcontroller

This setup is designed to operate in a fixed place. Therefore, we do not need to use batteries to deliver the necessary power to the different components. The necessary regulated voltages are obtained from the ac current. Firstly, the ac voltage (115 V) is changed to a lower level 36V using a transformer in our case. A Graetz bridge is combined with a low pass filter to rectify the voltage and smooth it for proper use in the components.

To drive the dc motor, a 24 V is needed and therefore an integrated circuit (IC) motor driver named L293D (dual H-bridge) is used. The presence of the letter "D", means that it is built in flyback diodes to minimize inductive voltage spikes. The L293D chip gives the desired power to the dc motor to move the load to the desired position if it is the case. This IC has an output current of 600 mA and a peak output current of 1.2 A per channel. It is important to notice this limitation since if the motor requires more current, the IC L293D will burn each time we exceed 1.2 A and a protection such as a fuse is needed in this case.

For the speed or the position control, we use the Microchip dsPIC30F4011. The intelligence that we will implement in the system is programmed in C language and after compilation, it is downloaded in the memory of the microcontroller.

Fig. 2.2 gives an idea of the whole mechatronic systems. The dc motor we use in this setup is manufactured by Maxon and it has a gear of 1: 6 ratio. An incremental encoder attached to shaft of the motor is also used to measure the position of the disk. With this setup we get 600 pulses per revolution. Our incremental encoder uses two output channels (A and B) like most of the incremental encoders to sense position. Based on the two code tracks on the disk of the encoder (positioned 90 degrees out of phase), the two output channels of the quadrature encoder indicate both position and direction of rotation. Therefore, if A leads B, for example, the disk is rotating in a clockwise direction, meanwhile if B leads A, then the disk is rotating in a counterclockwise direction. Another benefit of the quadrature signal scheme is its ability to electronically multiply the counts during one encoder cycle. Mostly the following is used for this purpose:

- all counts are generated on the rising edges of channel A
- both the rising and falling edges of channel A are used to generate counts
- the rising and falling edges of channel A and the channel B are used to generate counts

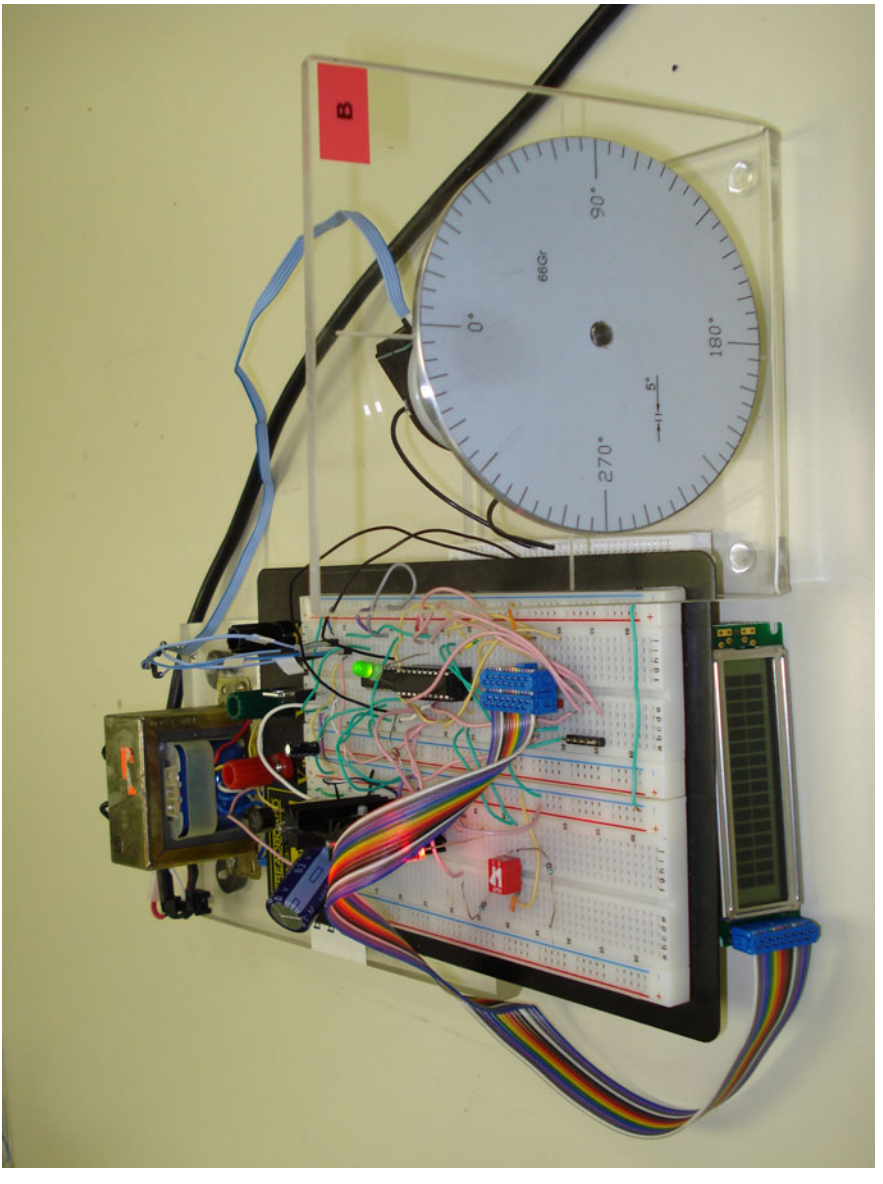


Fig. 2.2 Real-time implementation setup

Using the second or the third options we can increase the resolution and consequently improve the control precision. For instance, if the third option is used the resolution is increased by a factor of four and therefore we get 2400 pulses/rev.

For the speed control if the controller is chosen as proportional controller with a gain K_p, the system will work as follow. Firstly a speed reference is selected let

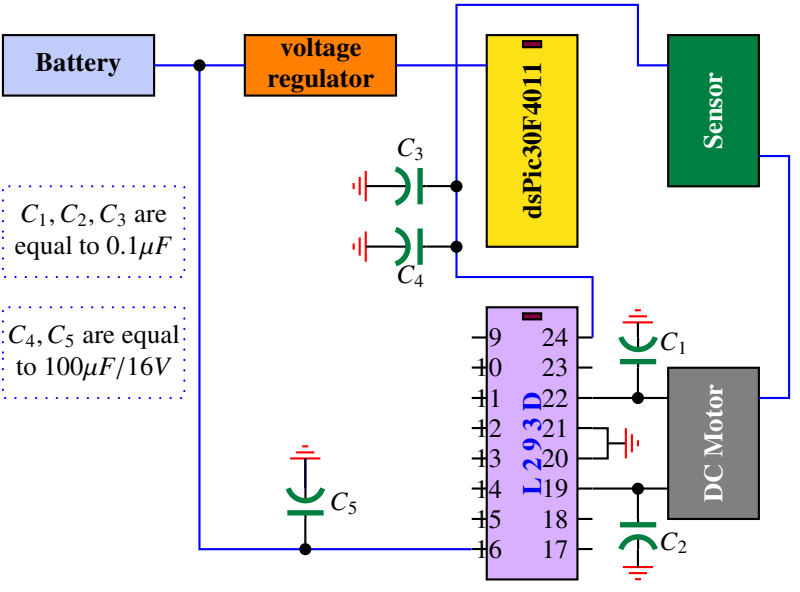


Fig. 2.3 Electronic circuit of the dc motor kit

say 100 rev/s. At each interrupt, the microcontroller will read the speed of the disk, compare it to the reference and compute the error. This error is multiplied by the gain K_p to generate the voltage to be delivered to the dc motor. Since the microcontroller can deliver a voltage between 0 and 5 V, the integrated circuit motor driver L293D will do the necessary to deliver only a voltage between 0 and 24 V with sufficient power to move the motor. The pulse width modulation (PWM) technique is used in this case. This technique is used to generate variable-width pulses to represent the amplitude of an analog input voltage that we should deliver to the dc motor. The PWM technique is characterized by its duty cycle which gives an indication of the fraction of time when the signal is on. The control of the voltage or the speed of the small disk is obtained by adjusting the duty cycle. The PWM works by making a square wave with a variable on-to-off ratio, the average on time may be varied from 0 to 100 percent. Fig. 2.3 gives an idea of the electronic circuit.

2.7.2 Two Wheels Robot

The idea of the two wheels robot has attracted a lot of researchers for the challenges it offers either in the modeling or in control. Different types of robots have been developed in research laboratories around the world. In our mechatronics laboratory, we have designed an experimental one that we use in our research to experiment our control algorithms. This robot has a compact structure and can be assembled or disassembled easily and quickly. It is composed of a platform on which a rod is attached at its middle. The whole is mounted on two wheels that are solidly attached

to the platform and are driven by two independent dc motors of the same type we used in the previous mechatronic system. The major parts of this robot are made from aluminium to reduce the robot weight. The electronic circuit which is a little bit more complicated compared to the previous system. This circuit is mounted on a breadboard and fixed to the platform. A set of batteries to obtain 24 V is used to deliver the different regulated voltages we need in this system. The batteries are put between the electronics and the platform.

The electronic circuit of this system is in some sense similar to the previous mechatronic system except for this system we need more components since we have two dc motors. The electronic circuit is built around the dsPIC30F4011 that orchestrates and manages all the tasks of the different parts of this system. For this electronic circuit we need more voltages since the LCD and the L293D need 5 V

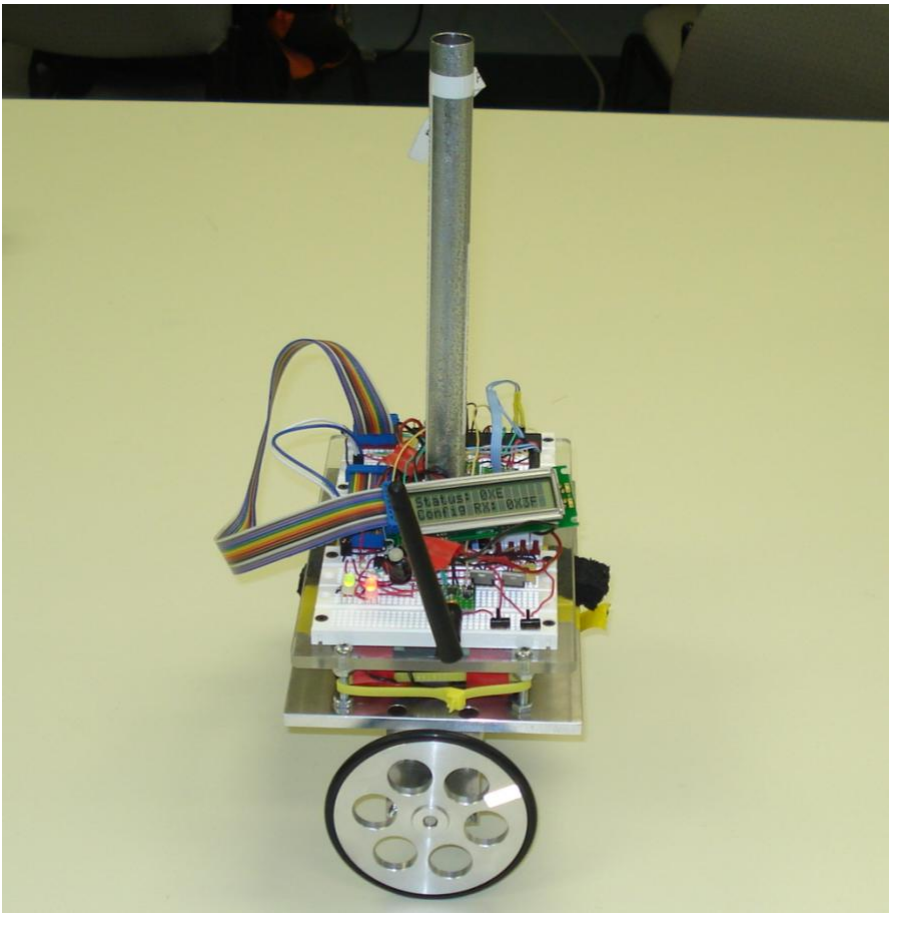


Fig. 2.4 Balancing robot

to operate, while dsPIC30F4011, the wireless, the accelerometer and the gyroscope need 3.5 V.

Beside the two encoders that are used to measure the positions of the wheels and therefore the one of the robot, an accelerometer and a gyroscope are used to measure the tilt of the robot. The goal is to keep the robot in the vertical position while moving along a desired trajectory. All this is done by controlling the dc motors. The PWM technique is also used here to deliver the desired voltages that are generated by the control algorithm we implement in the dsPIC.

The references to the robot can be either entered by program or sent wireless using a telecommunication system. Different control algorithms are experimented on this system. Some of these algorithms will be developed in the rest of this book.

Fig. 2.4 gives an idea of the whole mechatronic system, while the Fig. 2.5 gives an idea on the electronic circuit. The program is similar to the one of the dc motor kit except that more complex and too long to be presented here.

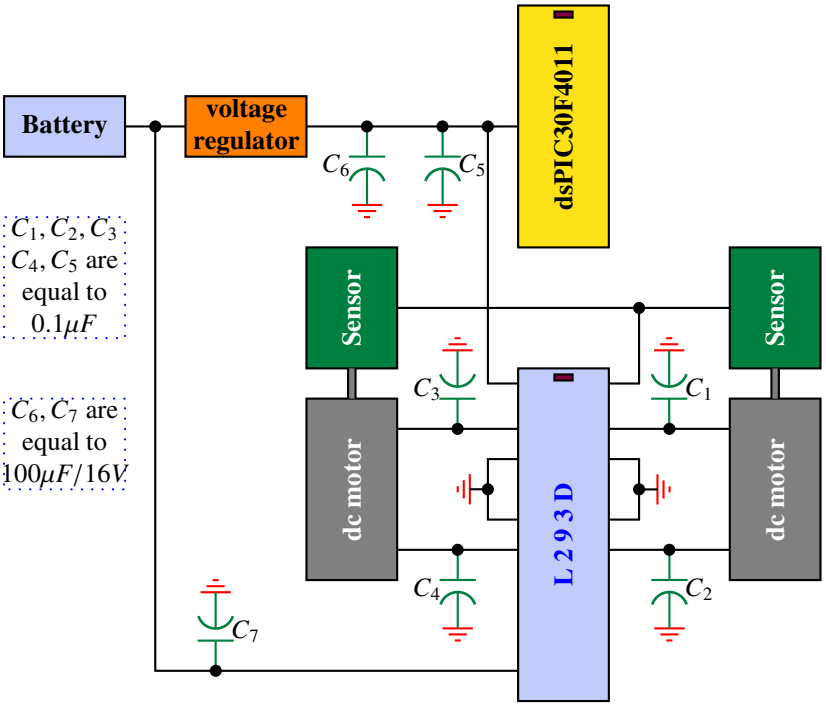


Fig. 2.5 Electronic circuit of the balancing robot

2.7.3 Magnetic Levitation

Magnetic levitation is a technology that has a lot applications which attracted a lot of researchers to this field. As an example where this technology is extensively used is in fast magnetic levitation trains since it permits to reduce the friction and therefore eliminates energy losses. In this section we will develop a system that used this technology and show that the principle works. The mechatronic system developed here is composed of two parts: a fixed one that represents the coil that generates the electromagnetic force and a ferromagnetic object that we would like to place at a certain position by acting on the electromagnetic force generated by the coil. The objective of the system is to control the vertical position of the moving object by adjusting the current in the electromagnet through the input voltage. The object position is measured using a Hall effect sensor. An elecgronic circuit build around a dsPIC30F4011 supplies the coil through an L298, an integrate circuit, with a current that is proportional to the command voltage of the actuator. Fig. 2.6 gives an idea of the whole mechatronic system.

2.8 Conclusions

In this chapter, we have presented the different components of mechatronic systems and we gave some mechatronic systems that we will use here to show the concepts developed in this volume. Some guidelines that can be used during the design phase of mechatronic systems are developed to give an idea to the reader and help him to design his own system.

2.9 Problems

1. In this problem we would like to design a one leg robot that can move using one wheel while remaining in a vertical position. Provide the design of such mechatronic system.
2. Solar energy is an alternate source of power that can be used. In this problem we ask you to design a solar system that maximizes the energy generated by the solar panel.
3. In this problem we ask you to design an insect with four legs that can walk and at the same time avoid obstacles.
4. Make the design of a small car that may use magnetic levitation to move. Give the different parts of such system.
5. In this problem we ask you to design a small airplane that may be used as a drone to give information of a certain region when flying over such region. Enumerate the different parts that may be used in such system.

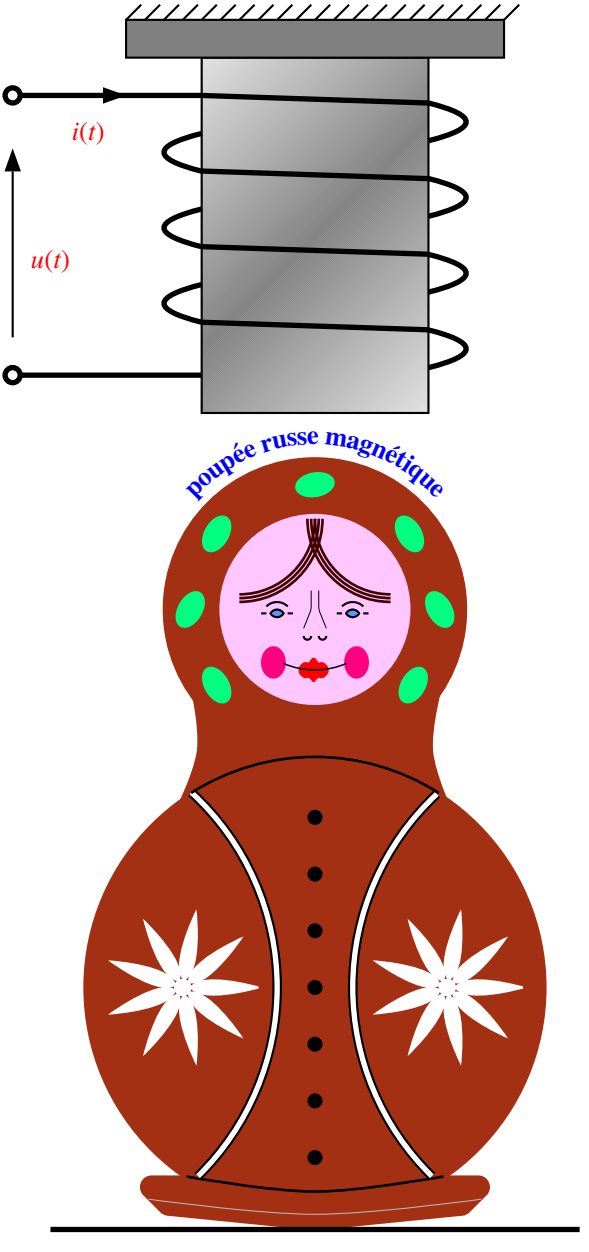


Fig. 2.6 Magnetic levitatios system

6. Design a small boat that we move on a small lake using a joystick. Enumerate the different components of this system.
7. In this problem we ask to design a hoover that can be controlled to seal on water via a emitter and a receiver using a joystick.

Part II

Modeling

In this modeling part we will cover different representations that may be used to describe a dynamical system that we would like to control in order to improve its performances. As it was said earlier, the focus is made on the control of continuous-time systems by microcontrollers that we can represent using one of the following representation:

1. transfer function
2. state space representation

More often, the relationship between the inputs and the outputs is described by differential equations that may be linear or nonlinear. For single input single output linear time invariant system, the transfer function, $G(s)$ is defined as follows:

$$G(s) = \frac{Y(s)}{R(s)} \tag{2.1}$$

where s is a complex variable that belongs to the set of complex number $\mathbb{C}$, $Y(s)$ and $R(s)$ represent respectively the Laplace transform [1] of the output, $y(t)$ and the input, $r(t)$ respectively, i.e.:

$$\begin{aligned} Y(s) &= \mathscr{L}[y(t)] \\ R(s) &= \mathscr{L}[r(t)]. \end{aligned}$$

The relation between the input and the output is then given by:

$$Y(s) = G(s)R(s) \tag{2.2}$$

For the multi-inputs multi-outputs case, we get similarly the following relation:

$$Y(s) = G(s)R(s) \tag{2.3}$$

with

$$\begin{aligned} R(s) &= \begin{bmatrix} R_1(s) \\ \vdots \\ R_m(s) \end{bmatrix} \\ Y(s) &= \begin{bmatrix} Y_1(s) \\ \vdots \\ Y_p(s) \end{bmatrix} \\ G(s) &= \begin{bmatrix} G_{11}(s) & \cdots & G_{1m}(s) \\ \vdots & \ddots & \vdots \\ G_{p1}(s) & \cdots & G_{pm}(s) \end{bmatrix} \end{aligned}$$

where $R_i(s)$, $Y_j(s)$ and G_{ji} represent respectively the ith input, the jth output and the transfer function between them when the other inputs are fixed to zero.

[1] The Laplace transform of a function $f(.)$ that satisfies the appropriate assumptions is defined by $F(s) = \int_0^\infty f(v)dv$

Notice that the jth output is given by the following expression:

$$Y_j(s) = G_{j1}(s)R_1(s) + G_{j2}(s)R_2(s) + \cdots + G_{jm}(s)R_m(s) \tag{2.4}$$

which implies the dependence of the outputs on the different inputs.

Usually, we use also the block diagram of Fig. 2.7 to represent dynamical systems.

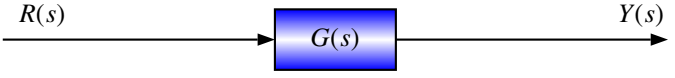


Fig. 2.7 Block diagram of continuous-time system

The state space representation is another way of representing the relationship between the input $u(t) \in \mathbb{R}^m$ and the output $y(t) \in \mathbb{R}^p$ of a given system and we can obtain it by proceeding with some mathematical transformation either of the differential equations or its corresponding transfer function. Its general structure is given by:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t), x(0) = x_0 \\ y(t) = Cx(t) + Du(t) \end{cases} \tag{2.5}$$

where A, B, C and D are constant real matrices with appropriate dimensions; and $x(t) \in \mathbb{R}^n$ and x_0 represent respectively the state of the system and its initial condition.

Usually the following block diagram (see Fig. 2.8) is used to represent dynamical systems in state space description:

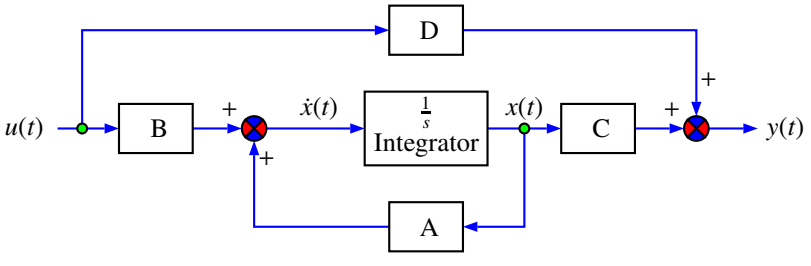


Fig. 2.8 Block diagram of continuous-time linear system

The goal of this part is to show to the reader how we can establish the mathematical model of a given dynamical system. The model can either be obtained through experiment or using the physics law with some specific experiments that may be used to determine the appropriate parameters that enter in the mathematical model obtained by this approach.

3
Mathematical Modeling

After reading this chapter the reader will:

1. be able to establish a mathematical model for any mechatronic system either analytically based on physics law or experimentally using the identification techniques
2. be able to build mathematical models for the mechatronic system using the transfer function concept
3. be able to build the state space representation for any given mechatronic system

It is well known that the mathematical modeling is a hard problem in control engineering. Most of the engineers working in this field agree on that. Any practical system has inputs and outputs. The outputs are variables that we would like to control or keep at certain levels, while, some of the inputs are variables on which we can act to change the outputs of the dynamical system. The rest of the inputs are referred to as external disturbances that are beyond our control.

A mathematical model is a representation that uses mathematical language, more often differential equations or difference equations, to describe the behavior of a dynamical system. Mathematical models are extensively used in engineering

disciplines to describe the relationship between inputs and outputs and the dynamical system parameters.

Mathematical models of dynamical system can be split into two categories depending on how the time variable is to be treated. A continuous-time mathematical model is based on a set of differential equations that are valid for any value of the time variable, whereas a discrete-time mathematical model provides information about the state of the physics system only at a selected set of distinct times.

The development of an appropriate model to describe the behavior of a given dynamical system can be done in different steps. At the first step, the inputs and the outputs variables are chosen. Then, at a second one the appropriate assumptions are made and the mathematical relationships between these variables are established using physics laws. Some experiments are required to determine the system's parameters.

In some circumstances, this approach is too complex and an another alternate is adopted to avoid this complexity. This approach consists of considering the dynamical system as a black box and recourse to the identification techniques. In the rest of this chapter we will cover these techniques and show to the reader how we can handle the mathematical modeling of some dynamical systems. In both cases we will be looking for the simplest accurate model we can get since this will facilitate the analysis and the design phases.

3.1 Mathematical Modeling Based on Physics Laws

To show how this technique can be applied let us consider a certain number of dynamical systems. As a first example let us consider a dc motor with a mechanical load that we like to control either in speed or in position. The dc motor represents the actuator that is mostly used in the position control servomechanism. It is the means by which the electrical energy is converted to mechanical energy. The block diagram of the dc motor driving the load of our example is illustrated by Fig. 3.1. If we let $u(t)$, $i(t)$ and $\omega(t)$ denote respectively the voltage of the armature, the current in the armature and the speed of the shaft at time t, based on the basic electrical and mechanics laws we have the following:

$$\begin{cases} u(t) = Ri(t) + L_m \frac{di}{dt}(t) + K_w w(t) \\ J \frac{dw}{dt}(t) = K_t i(t) - b\omega(t) \end{cases} \tag{3.1}$$

where R, L_m, K_w, K_t represent respectively the electric resistor of the armature, the inductance of the armature, the electromotive force constant, the torque constant (in the international system (IS) these both constants are equal), J and b are defined by:

$$\begin{aligned} J &= J_m + \frac{J_c}{n^2} \\ b &= b_m + \frac{b_c}{n^2} \end{aligned}$$

with J_m and J_c are the moments of inertia of the rotor and the load respectively, and b_m and b_c are the damping ratios of the motor and the load, and n is the gear ratio.

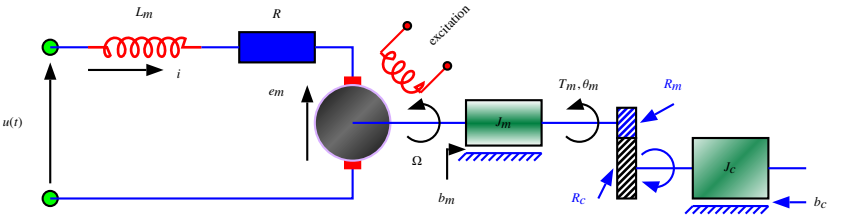


Fig. 3.1 Block diagram of a dc motor

3.1.1 Concept of Transfer Function

If we use the Laplace transform with the initial conditions equal to zero, we get:

$$\begin{cases} U(s) = RI(s) + L_m sI(s) + K_w \Omega(s) \\ Js\Omega(s) = K_t I(s) - b\Omega(s) \end{cases} \tag{3.2}$$

where $U(s)$, $I(s)$, and $\Omega(s)$ are respectively the Laplace transform of $u(t)$, $i(t)$ and $\omega(t)$.

Combining these relations and the definition of the transfer function between the velocity $\Omega(s)$ and the voltage $U(s)$, we get:

$$G(s) = \frac{\Omega(s)}{U(s)} = \frac{K_t}{(Js + b)(L_m s + R) + K_t K_w} \tag{3.3}$$

If the armature inductance L_m can be neglected, the transfer function becomes:

$$G(s) = \frac{K}{\tau s + 1} \tag{3.4}$$

with

$$\begin{aligned} K &= \frac{K_t}{Rb + K_t K_w} \\ \tau &= \frac{JR}{Rb + K_t K_w} \end{aligned}$$

Remark 3.1.1 *When the armature inductance L_m can be neglected the mathematical model can be simplified to a first order system, otherwise we have a second order one. It may happen in some applications that the dynamics of the driven load is too slow compared to the actuator one and in this case, the dynamics of the actuator is reduced to a simple gain.*

Remark 3.1.2 *The parameters of the dc motor are in general available in its data sheet. Once the inertia of the load and the gear ratio are known, all the data of the model are then known. It is also important to mention that the data sheet contains the average data for a sample that has been tested to get these parameters. Therefore, it may happen that the considered actuator may have uncertainties in its model that can be compensated by the choice of the appropriate controller.*

Notice also that the position, $\theta(t)$, of the dc motor is obtained from the velocity, $\omega(t)$, by using:

$$\Theta(s) = \frac{\Omega(s)}{s}$$

where $\Theta(s) = \mathscr{L}[\omega(t)]$.

Using this and the simplified model between the voltage and the velocity, we get the following relationship between the voltage and position:

$$G(s) = \frac{\Theta(s)}{U(s)} = \frac{K}{s(\tau s + 1)}$$

where K and τ are defined previously.

Notice that the previous relations of the mathematical model between the voltage and the velocity can be rewritten as follows:

$$\begin{cases} \frac{di}{dt}(t) = -\frac{R}{L_m} i(t) - \frac{K_w}{L_m} w(t) + \frac{1}{L_m} u(t) \\ \frac{dw}{dt}(t) = \frac{K_t}{J} i(t) - \frac{b}{J} \omega(t) \end{cases} \tag{3.5}$$

3.1.2 State Space Description

Now if we let $x_1(t) = i(t)$, $x_2(t) = \omega(t)$ and $y(t) = x_2(t)$ we get:

$$\begin{cases} \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -\frac{R}{L_m} & -\frac{K_w}{L_m} \\ \frac{K_t}{J} & -\frac{b}{J} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{L_m} \\ 0 \end{bmatrix} u(t) \\ y(t) = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \end{cases} \tag{3.6}$$

that gives the following standard form:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases} \tag{3.7}$$

where

$$A = \begin{bmatrix} -\frac{R}{L_m} & -\frac{K_w}{L_m} \\ \frac{K_t}{J} & -\frac{b}{J} \end{bmatrix},$$

$$B = \begin{bmatrix} \frac{1}{L_m} \\ 0 \end{bmatrix}, C = \begin{bmatrix} 0 & 1 \end{bmatrix}.$$

This mathematical form of the system is known in the literature as the state space representation.

Remark 3.1.3 *In this example we assumed that we have access only to the velocity which implies that that* $C = \begin{bmatrix} 0 & 1 \end{bmatrix}$. *If we have access only access to the current or to the two variables the corresponding output matrices become respectively* $C = \begin{bmatrix} 1 & 0 \end{bmatrix}$, $C = \begin{bmatrix} 1 & 1 \end{bmatrix}$.

For the state space representation that gives the position, notice that the previous relations of the mathematical model between the voltage and the velocity and the relation that links the velocity and position can be rewritten as follows:

$$\begin{cases} \frac{di}{dt}(t) = -\frac{R}{L_m} i(t) - \frac{K_w}{L_m} w(t) + \frac{1}{L_m} u(t) \\ \frac{dw}{dt}(t) = \frac{K_t}{J} i(t) - \frac{b}{J} \omega(t) \\ \frac{d\theta}{dt}(t) = \omega(t) \end{cases} \tag{3.8}$$

Now if we let $x_1(t) = i(t)$, $x_2(t) = \omega(t)$, $x_3(t) = \theta(t)$ and $y(t) = x_3(t)$ we get:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \end{bmatrix} = \begin{bmatrix} -\frac{R}{L_m} & -\frac{K_w}{L_m} & 0 \\ \frac{K_t}{J} & -\frac{b}{J} & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{L_m} \\ 0 \\ 0 \end{bmatrix} u(t) \tag{3.9}$$

$$y(t) = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} \tag{3.10}$$

that gives the standard form (3.7) with:

$$A = \begin{bmatrix} -\frac{R}{L_m} & -\frac{K_w}{L_m} & 0 \\ \frac{K_t}{J} & -\frac{b}{J} & 0 \\ 0 & 1 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} \frac{1}{L_m} \\ 0 \\ 0 \end{bmatrix}, C = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}.$$

To use these models, we need to know the different parameters in each one. This may be in some circumstances difficult to measure and therefore another alternate is required. In the next section, the situation will overcome by using the identification techniques.

Remark 3.1.4 *It is important to mention that the state space description is not unique, which means that for a given system, we can find many state space description. This matter will be explained later in this volume.*

Remark 3.1.5 *Notice that in general, the manufacturer of dc motors provides the data sheet in which we can find all these parameters that correspond to a sample that was chosen for test. These parameters may be not identical to those of the dc motor we are using and this may cause some error in modeling. The feedback control will cope with such errors.*

As a second example, we consider the model of the Segway (see [6]). The dynamics of this system is composed of two models that will be decoupled under some appropriate assumptions. We assume that the Segway remains close to its vertical

position when moving with small speed and the wheels remain in touch with the ground and don't slip.

Under these assumptions the dynamics of our Segway will be partitioned into two parts. The first one gives the behavior of the tilt and linear displacement dynamics while the second one governs the heading angle dynamics. Now if we define the variables of the Table 3.1.

Table 3.1 Variables definition

Variable	definition
$\psi(t)$	tilt angle
$x(t)$	linear position
$\theta_i(t)$	motors' shaft angle
$\theta_o(t)$	gear box shaft angle
$\theta(t)$	wheels' angle
$T_i(t)$	torque delivered to a gear box by one of the dc motors
$T(t)$	torque delivered to a wheel by one of the dc motors
$F(t)$	resultant force between the ground and each of the wheels
$u_x(t)$	motors' voltage input controlling tilt and linear displacement
r_w	wheels' radius
M	mass of the half robot including one wheel
m_b	mass of half the body of the robot
m_w	mass of one of the wheels
J_b	moment of inertia of half the body of the robot
J_w	moment of inertia of one of the wheels
d	distance between motors' shafts and center of gravity of the body
K_t	motors' torque constant
K_e	motors' back emf constant
r_a	motors' armature resistance
r_g	gear boxes' ratio
η	gear boxes' Efficiency
C_f	rotational damping constant

and noticing that the following relations hold always:

$$\begin{aligned} M &= m_b + m_w \\ K_t &= K_e = K \\ x(t) &= r_w\theta(t) \\ \theta_i(t) &= r_g\theta_o(t) \\ \theta_o(t) &= \theta(t) + \psi(t) \\ F(t) &= M\ddot{x}(t) \end{aligned}$$

we have the following relations:

- motor' dynamics

$$\begin{aligned} T_i(t) &= K_t i(t) \\ i(t) &= \frac{u_x(t)}{r_a} - \frac{K_e \dot{\theta}_i(t)}{r_a} \\ T_i(t) &= K_t \left[\frac{u_x(t)}{r_a} - \frac{K_e \dot{\theta}_i(t)}{r_a} \right] \end{aligned} \tag{3.11}$$

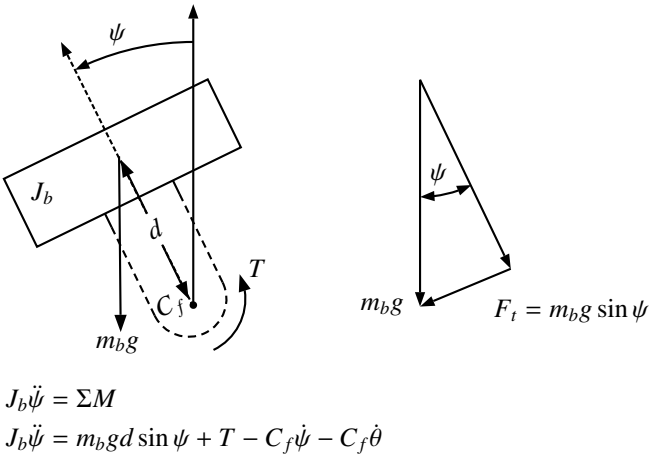


Fig. 3.2 Tilt dynamics free body diagram

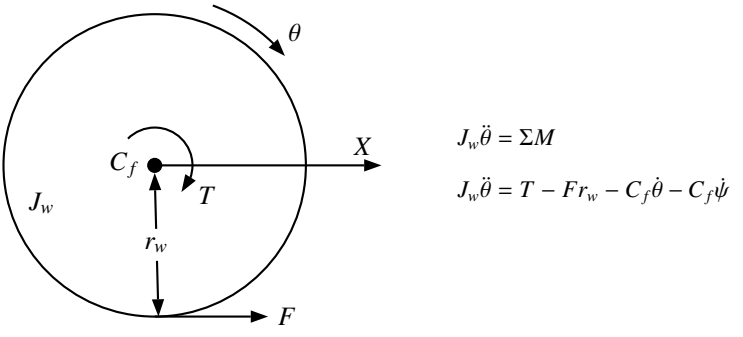


Fig. 3.3 Wheels and linear displacement free body diagram

- torque applied to the wheels

$$\begin{aligned} T(t) &= \eta r_g T_i(t) \\ &= \frac{\eta r_g K u_x(t)}{r_a} - \frac{\eta r_g K^2}{r_a}\dot{\theta}_i(t) \\ T(t) &= \frac{\eta r_g K}{r_a}u_x(t) - \frac{\eta r_g^2 K^2}{r_a}\dot{\theta}(t) - \frac{\eta r_g^2 K^2}{r_a}\dot{\psi}(t) \\ &= \frac{\eta r_g K}{r_a}u_x(t) - \frac{\eta r_g^2 K^2}{r_a r_w}\dot{x}(t) - \frac{\eta r_g^2 K^2}{r_a}\dot{\psi}(t) \end{aligned} \tag{3.12}$$

- robot tilt dynamics, referring to Fig. 3.2 we have:

$$\begin{aligned} J_b\ddot{\psi}(t) &= m_b g d\sin(\psi(t)) + T(t) - C_f\dot{\psi}(t) - C_f\dot{\theta}(t) \\ &= m_b g d\sin(\psi(t)) + \frac{\eta r_g K}{r_a}u_x(t) - \frac{\eta r_g^2 K^2}{r_a r_w}\dot{x}(t) - \frac{\eta r_g^2 K^2}{r_a}\dot{\psi}(t) - C_f\dot{\psi}(t) - C_f\frac{\dot{x}(t)}{r_w} \\ \ddot{\psi}(t) &= \frac{m_b g d\sin(\psi(t))}{J_b} - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a J_b}\right]\dot{\psi}(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w J_b}\right]\dot{x}(t) + \frac{\eta r_g K}{r_a J_b}u_x(t) \end{aligned}$$

If we assume that $\psi(t)$ is small we get $\sin(\psi(t)) \approx \psi(t)$ which implies in turn:

$$\begin{aligned} \ddot{\psi}(t) &= \frac{\eta r_g K}{r_a J_b}u_x(t) + \frac{m_b g d}{J_b}\psi(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a J_b}\right]\dot{\psi}(t) \\ &\quad - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w J_b}\right]\dot{x}(t) \end{aligned} \tag{3.13}$$

- robot wheels and linear displacement dynamics, referring to Fig. 3.3

$$\begin{aligned} J_w\ddot{\theta}(t) &= T(t) - F(t)r_w - C_f\dot{\theta}(t) - C_f\dot{\psi}(t) \\ &= \frac{\eta r_g K}{r_a}u_x(t) - \frac{\eta r_g^2 K^2}{r_a r_w}\dot{x}(t) - \frac{\eta r_g^2 K^2}{r_a}\dot{\psi}(t) - r_w M\ddot{x}(t) - \frac{C_f}{r_w}\dot{x}(t) - C_f\dot{\psi}(t) \end{aligned} \tag{3.14}$$

which in turn gives:

$$\left[\frac{J_w}{r_w} + M r_w\right]\ddot{x}(t) = \frac{\eta r_g K}{r_a}u_x(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a}\right]\dot{\psi}(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w}\right]\dot{x}(t)$$

and finally, we obtain:

$$\begin{aligned} \ddot{x}(t) &= \left[\frac{\eta r_w r_g K}{r_a(J_w + M r_w^2)}\right]u_x(t) - \left[\frac{\eta r_w r_g^2 K^2 + C_f r_w r_a}{r_a(J_w + M r_w^2)}\right]\dot{\psi}(t) \\ &\quad - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a(J_w + M r_w^2)}\right]\dot{x}(t) \end{aligned} \tag{3.15}$$

If we define $\mathbf{x}^\top(t) = \begin{bmatrix} \psi(t) & \dot{\psi}(t) & x(t) & \dot{x}(t) \end{bmatrix}$ and $\mathbf{y}^\top(t) = \begin{bmatrix} \psi(t) & x(t) \end{bmatrix}$, we get the following state space representation:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= A\mathbf{x}(t) + Bu_x(t) \\ \mathbf{y}(t) &= C\mathbf{x}(t) \end{aligned}$$

where

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{m_b g d}{J_b} & -\frac{\eta r_g^2 K^2 + C_f r_a}{r_a J_b} & 0 & -\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w J_b} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{\eta r_w r_g^2 K^2 + C_f r_w r_a}{r_a (J_w + M r_w^2)} & 0 & \frac{\eta r_g^2 K^2 + C_f r_a}{r_a (J_w + M r_w^2)} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ \frac{\eta r_g K}{r_a J_b} \\ 0 \\ \frac{\eta r_w r_g K}{r_a (J_w + M r_w^2)} \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

We will now establish the model representing the heading angle dynamics of the robot taking into consideration that an equal but opposite torque has to be applied by the two motors in order to induce a purely rotational motion on the robot without affecting its tilt and linear position. Therefore, an equal but opposite voltage has to be applied to the two motors and this voltage is taken as the input of this system. Here, we are taking the assumption that the robot is staying in the vertical position and that its moment of inertia around the vertical axis is not changing. If we introduce the additional variables of the Table 3.2 and noticing again that the following hold:

Table 3.2 Variables definition

Variable	definition
$\delta(t)$	heading angle
$x_r(t)$	linear position of the right wheel
$x_l(t)$	linear position of the left wheel
$\theta_r(t)$	right wheel angle
$\theta_l(t)$	left wheel angle
$T_r(t)$	torque delivered by the right dc motor
$T_l(t)$	torque delivered by the left dc motor
$F_r(t)$	driving force of right wheel
$F_l(t)$	driving force of left wheel
$u_r(t)$	right motor voltage input
$u_l(t)$	left motor voltage input
$u_h(t)$	motors' voltage input controlling heading
J_d	moment of inertia of the robot around the vertical axis
S	wheel span

$$
\begin{aligned}
x_r(t) &= r_w\theta_r(t) \\
x_l(t) &= r_w\theta_l(t) \\
\delta(t) &= \left[\frac{x_l(t) - x_r(t)}{S}\right] \\
u_l(t) &= -u_r(t) = u_h(t) \\
u_l(t) &- u_r(t) = 2u_h(t)
\end{aligned}
$$

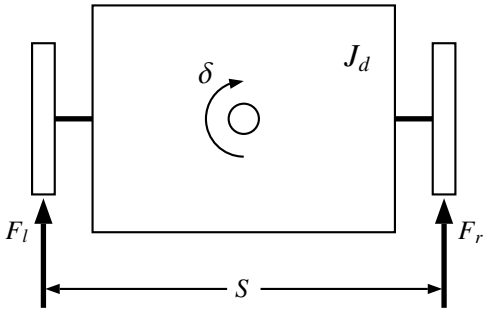
we have the following relations:

- from (3.14), solving for $F(t)$, we have:

$$
\begin{aligned}
F(t) &= \frac{T(t) - J_w\ddot{\theta}(t) - C_f\dot{\theta}(t) - C_f\dot{\psi}(t)}{r_w} \\
&= \frac{\eta r_g K}{r_a r_w}u(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w}\right]\dot{\psi}(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w^2}\right]\dot{x}(t) - \frac{J_w}{r_w}\ddot{\theta}(t)
\end{aligned}
$$

now making reference to left and right, we get:

$$
\begin{aligned}
F_l(t) &= \frac{\eta r_g K}{r_a r_w}u_l(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w}\right]\dot{\psi}(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w^2}\right]\dot{x}_l(t) - \frac{J_w}{r_w}\ddot{\theta}_l(t) \\
F_r(t) &= \frac{\eta r_g K}{r_a r_w}u_r(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w}\right]\dot{\psi}(t) - \left[\frac{\eta r_g^2 K^2 + C_f r_a}{r_a r_w^2}\right]\dot{x}_r(t) - \frac{J_w}{r_w}\ddot{\theta}_r(t)
\end{aligned}
$$



$$
\begin{aligned}
J_d\ddot{\delta} &= \Sigma M \\
J_d\ddot{\delta} &= F_l\frac{S}{2} - F_r\frac{S}{2} \\
J_d\ddot{\delta} &= (F_l - F_r)\frac{S}{2}
\end{aligned}
$$

Fig. 3.4 Heading dynamics free body diagram

- referring to Fig. 3.4 we get:

$$\begin{aligned} J_d\ddot{\delta}(t) &= [F_l(t) - F_r(t)]\frac{S}{2} \\ &= \frac{\eta r_g KS}{2r_a r_w}[u_l(t) - u_r(t)] + \left[\frac{\eta S r_g^2 K^2 + S C_f r_a}{2r_a r_w^2}\right][\dot{x}_r(t) - \dot{x}_l(t)] + \frac{J_w S}{2r_w}\left[\ddot{\theta}_r(t) - \ddot{\theta}_l(t)\right] \\ &= \frac{\eta r_g KS}{r_a r_w}u_h(t) - \left[\frac{\eta S^2 r_g^2 K^2 + S^2 C_f r_a}{2r_a r_w^2}\right]\dot{\delta}(t) - \frac{J_w S^2}{2r_w^2}\ddot{\delta}(t) \end{aligned}$$

which in turn gives:

$$\left[J_d + \frac{J_w S^2}{2r_w^2}\right]\ddot{\delta}(t) = \frac{\eta r_g KS}{r_a r_w}u_h(t) - \left[\frac{\eta S^2 r_g^2 K^2 + S^2 C_f r_a}{2r_a r_w^2}\right]\dot{\delta}(t)$$

and finally, we obtain:

$$\ddot{\delta}(t) = \left[\frac{2\eta r_w r_g KS}{r_a(2J_d r_w^2 + J_w S^2)}\right]u_h(t) - \left[\frac{\eta S^2 r_g^2 K^2 + S^2 C_f r_a}{r_a(2J_d r_w^2 + J_w S^2)}\right]\dot{\delta}(t) \quad (3.16)$$

If we define $\mathbf{x}_h^\top(t) = \left[\delta(t)\ \dot{\delta}(t)\right]$ and $\mathbf{y}_h(t) = \delta(t)$, we get the following state space representation:

$$\begin{aligned} \dot{\mathbf{x}}_h(t) &= A_h\mathbf{x}_h(t) + B_h u_h(t) \\ \mathbf{y}_h(t) &= C_h\mathbf{x_h}(t) \end{aligned}$$

where

$$\begin{aligned} A_h &= \begin{bmatrix} 0 & 1 \\ 0 & -\frac{\eta S^2 r_g^2 K^2 + S^2 C_f r_a}{r_a(2J_d r_w^2 + J_w S^2)} \end{bmatrix} \\ B_h &= \begin{bmatrix} 0 \\ \frac{2\eta r_w r_g KS}{r_a(2J_d r_w^2 + J_w S^2)} \end{bmatrix}, C_h = \begin{bmatrix} 1 & 0 \end{bmatrix} \end{aligned}$$

The last example is the magnetic levitation system. This system is represented by the Fig. 4.10. The data of this system are summarized in the Table 3.3.

Table 3.3 Data of the magnetic levitation system

Variable	value
R	62.7 Ω
L	60 mH
m (object mass)	7.64 g
diameter of the permanent magnet	9 mm

Let $x(t)$ denote the position of the object at time t measured from the bottom of the coil. The dynamics of the moving object is described by the following differential equation:

$$m\ddot{x}(t) = mg - F_c - F_p \tag{3.17}$$

where g is the gravity, F_c and F_p are the magnetic forces generated respectively by the coil and the permanent magnet.

Remark 3.1.6 *It is important to notice that the direction of the magnetic force F_c is linked to the direction of the current in the coil.*

If we denote by $i(t)$ the current at time t that give a force F_c pointing down at time t with the following expression:

$$F_c(t) = k_c \frac{i^2(t)}{x^2(t)}$$

The permanent force F_p is given by the following expression:

$$F_p(t) = k_p \frac{1}{x^2(t)}$$

Using these expressions we get:

$$m\ddot{x}(t) = mg - k_c \frac{i^2(t)}{x^2(t)} - k_p \frac{1}{x^2(t)} \tag{3.18}$$

From the other side, we have the following relation between the current $i(t)$ and the applied voltage $u(t)$:

$$u(t) = Ri(t) + L\frac{di(t)}{dt}$$

If we neglect the effect of the coil, this relation becomes:

$$u(t) = Ri(t)$$

and the dynamics become:

$$m\ddot{x}(t) = mg - k_c \frac{u^2(t)}{x^2(t)} - k_p \frac{1}{x^2(t)} \tag{3.19}$$

For the output equation notice that we have a Hall effect sensor that generates a voltage that is function of the position, $x(t)$ of the object and therefore it is function of the magnetic field B (sum of the two fields (the one due to the coil and the one due to permanent magnet)). If we denote by $y(t)$ this voltage and using the data sheet of this sensor, we get:

$$y(t) = 0.003125B + 2.5$$

where B is measured in Gauss (1 Telta = 1000 Gauss).

This gives the following one:

$$y(t) = 31.25B + 2.5$$

where B is measured in Tesla.

It can be shown that the expression of the total magnetic field is given by:

$$B = C_p \frac{1}{x^3(t)} + C_b i(t) + C_1 + C_2$$

where $C_p = -1.9446\ 10^{-8}$, $C_b = -0.1671$, $C_1 = -0.011027$ and $C_2 = 0.003568$.

In conclusion the output of the sensor is then given by:

$$y(t) = \left[\frac{1}{0.032}\left[C_p \frac{1}{x^3(t)} + C_b i(t) + C_1 + C_2\right]\right] + 2.5$$

It can seen that the model is nonlinear and the theory we will present in this volume will not help. Therefore a linearization around an equilibrium point is required. At the equilibrium point the speed and the acceleration of the object are equal to zero and the current is constant in time and the total force is equal to the gravitational force. Using this and the previous dynamics, we get:

$$\begin{cases} x^2(t) = sign(u(t))\frac{k_c}{mgR^2}u^2(t) + \frac{k_p}{mg} \\ y(t) = \left[\frac{1}{0.032}\left[C_p \frac{1}{x^3(t)} + C_b i(t) + C_1 + C_2\right]\right] + 2.5 \end{cases}$$

Using these conditions and some appropriate experiments, we can determine the values for k_c and k_p and these values are given by:

$$\begin{aligned} k_c &= 5.9218\ 10^{-4} \\ k_p &= 4.0477\ 10^{-6}. \end{aligned}$$

At the equilibrium point, the object occupies a fixed position x_e that corresponds to the voltage u_e ($u_e = Ri_e$). The corresponding voltage delivered by the sensor is y_e. In the neighborhood of this equilibrium point (x_e, u_e, i_e, y_e), the system has a linear behavior. The linearized model is given by ([1]):

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}$$

where

$$\begin{aligned} x(t) &= \begin{bmatrix} x_1(t)(position) \\ x_2(t)(velocity) \end{bmatrix} \\ A &= \begin{bmatrix} 0 & 1 \\ \frac{2\left[sign(u_e)k_c u_e^2 + k_p R^2\right]}{mR^2 x_e^3} & 0 \end{bmatrix} \\ B &= \begin{bmatrix} 0 \\ \frac{-2sign(u_e)k_c u_e}{mR^2 x_e^2} \end{bmatrix} \\ C &= \begin{bmatrix} \frac{-3C_p}{0.032x_e^4} & 0 \end{bmatrix} \\ D &= \frac{C_b}{0.032R} \end{aligned}$$

3.2 Identification

From the previous example, it can be seen from that the establishment of the mathematical model that we can use for analysis and design is not an easy task and even if we can get the model from physics laws, the value of the different parameters of the model may be impossible to get and therefore the analytical model is useless.

System identification is a procedure by which a mathematical description of a dynamical system is extracted from test data. The aim of the identification is to construct an algorithm that will allow to build a mathematical model from observed data. Mainly the system we would like to model is seen as black box with some inputs and outputs that are collected at some finite instants of time.

The establishment of an appropriate model for a given linear time invariant system can be done into two steps. Firstly, a structure of a model that may fit with the collected data is chosen and then the parameter of this model are determined.

The identification problem can be stated as follows: given N samples of the pair $(u(k), y(k))$ where $u(k)$ and $y(k)$ denote respectively the input and output collected from experiments on the real system, we wish to determine the system's parameters of the chosen model such that it matches the real system sufficiently well.

3.2.1 Transfer Function Approach

One of the approaches that we may use to build a model with transfer function description is the least-square system identification. To show how this algorithm works, let us assume the structure of the chosen model is given by:

$$G(z) = \frac{Y(z)}{U(z)} = \frac{b_1 z^{n-1} + b_2 z^{n-2} + \cdots + b_n}{z^n - a_1 z^{n-1} - \cdots - a_n}$$

where $Y(z)$ and $U(z)$ are respectively $\mathscr{Z}$-transform[1] of the output $y(k)$ and $u(k)$, $a_1, \cdots, a_n$ and $b_1, \cdots, b_n$ are the model parameters that we have to determine.

Using $\mathscr{Z}$-transform inverse we get following model:

$$\begin{aligned} y(k) &= a_1 y(k-1) + a_2 y(k-2) + \cdots + a_n y(k-n) \\ &\quad + b_1 u(k-1) + b_2 u(k-2) + \cdots + b_n u(k-n) \end{aligned}$$

The objective of the identification procedure is to determine the model parameters from measurements of the inputs, $u(k), k = 0, 1, \cdots, N$ and the corresponding outputs, $y(k), k = 0, 1, \cdots, N$. For this purpose let:

$$\theta = \begin{bmatrix} a_1 & a_2 & \cdots & a_n & b_1 & b_2 & \cdots & b_n \end{bmatrix} \tag{3.20}$$

Let us now assume that we collected $N + 1$ measurements pairs:

$$(u(0), y(0)), (u(1), y(1)), \cdots, (u(N), y(N))$$

with $N > n$.

[1] The definition of the $\mathscr{Z}$-transform will be given later in this book

By defining $f(k)$ as follows:

$$f^\top(k) = \Big[y(k-1)\ y(k-2) \cdots\ y(k-n)\ u(k-1)\ u(k-2) \cdots\ u(k-n) \Big]$$

then for the sample periods $n, n+1, \cdots, N$ we have:

$$\begin{cases} y(n) = f^\top(n)\theta + e(n) \\ y(n+1) = f^\top(n+1)\theta + e(n+1) \\ \qquad \vdots \\ y(N) = f^\top(N)\theta + e(N) \end{cases} \tag{3.21}$$

where $e(k)$ is the error estimation at period kT.

If we define $\mathbf{y}(N)$, $\mathbf{f}(N)$ and $\mathbf{e}(N)$ as follows:

$$\mathbf{y}(N) = \begin{bmatrix} y(n) \\ y(n+1) \\ \vdots \\ y(N) \end{bmatrix}, \mathbf{f}(N) = \begin{bmatrix} f^\top(n) \\ f^\top(n+1) \\ \vdots \\ f^\top(N) \end{bmatrix}, \mathbf{e}(N) = \begin{bmatrix} e(n) \\ e(n+1) \\ \vdots \\ e(N) \end{bmatrix}$$

then the previous relation becomes:

$$\mathbf{y}(N) = \mathbf{f}(N)\theta + \mathbf{e}(N) \tag{3.22}$$

where $\mathbf{y}(N) \in \mathbb{R}^{N-n+1}$, $\mathbf{e}(N) \in \mathbb{R}^{N-n+1}$, $\mathbf{f}(N) \in \mathbb{R}^{(N-n+1)\times 2n}$ and $\theta \in \mathbb{R}^{2n}$

Using now the least square algorithm with the following cost:

$$J(\theta) = \sum_{k=n}^{N} e^2(k) = \mathbf{e}^\top(N)\mathbf{e}(N) \tag{3.23}$$

This implies the following:

$$\begin{aligned} J(\theta) &= [\mathbf{y}(N) - \mathbf{f}(N)\theta]^\top [\mathbf{y}(N) - \mathbf{f}(N)\theta] \\ &= \mathbf{y}^\top(N)\mathbf{y}(N) - \theta^\top \mathbf{f}^\top(N)\mathbf{y}(N) - \mathbf{y}^\top(N)\mathbf{f}(N)\theta + \theta^\top \mathbf{f}^\top(N)\mathbf{f}(N)\theta \\ &= \mathbf{y}^\top(N)\mathbf{y}(N) - 2\theta^\top \mathbf{f}^\top(N)\mathbf{y}(N) + \theta^\top \mathbf{f}^\top(N)\mathbf{f}(N)\theta \end{aligned} \tag{3.24}$$

To search for the optimal solution $\theta^\star$ that minimizes the cost $J(\theta)$, we can use the optimization conditions (see [3]). By these optimality conditions, we get:

$$\frac{\partial J(\theta)}{\partial \theta} = -2\mathbf{f}^\top(N)\mathbf{y}(N) + 2\mathbf{f}^\top(N)\mathbf{f}(N)\theta^\star = 0$$

that can be rewritten as:

$$\mathbf{f}^\top(N)\mathbf{f}(N)\theta^\star = \mathbf{f}^\top(N)\mathbf{y}(N)$$

from which we obtain the optimal solution as follows:

$$\theta^\star = \Big[\mathbf{f}^\top(N)\mathbf{f}(N)\Big]^{-1} \mathbf{f}^\top(N)\mathbf{y}(N)$$

provided that the matrix, $[\mathbf{f}^\top(N)\mathbf{f}(N)]$, is not singular.

Remark 3.2.1 *The formula we just developed allows us to compute the parameters off-line after collecting the data. But in some applications we may need to compute these parameters on-line and therefore adapt the controller's parameters as it is the case for adaptive control. This can be done using the recursive form of the least square algorithm.*

To establish the recursive algorithm, we will use some forgetting factors. Consequently, the cost is modified to:

$$\begin{aligned} J(\theta) &= \mu(n)e^2(n) + \mu(n+1)e^2(n+1) + \cdots + \mu(N)e^2(N) \\ &= \sum_{k=n}^{N} \mu(k)e^2(k) = \mathbf{e}^\top(N)\mathbf{F}(N)\mathbf{e}(N) \end{aligned} \tag{3.25}$$

where $\mathbf{F}(N)$ is a diagonal matrix, $F(N) = \text{diag}(\mu(1), \cdots, \mu(N))$.

Proceeding similarly as we did previously, we get:

$$\theta^\star = \left[\mathbf{F}^\top(N)\mathbf{F}(N)\mathbf{f}(N)\right]^{-1} \mathbf{f}^\top(N)\mathbf{F}(N)\mathbf{y}(N) \tag{3.26}$$

Remark 3.2.2 *The forgetting factors are used to give more weight for the recent data.*

Let $\mu(k) = \alpha\beta^{N+1-k}$, with $\beta \leq 1$. Based now on the expression of $\theta^\star$, we get:

$$\begin{aligned} \mathbf{f}^\top(N+1)\mathbf{F}(N+1)\mathbf{f}(N+1) &= \sum_{k=n}^{N+1} \alpha\beta^{N+1-k} f(k)f^\top(k) \\ &= \sum_{k=n}^{N} \alpha\beta\beta^{N-k} f(k)f^\top(k) + \alpha f(N+1)f^\top(N+1) \end{aligned}$$

Let us now define $\Psi(k)$ as follows:

$$\Psi(k) = \left[\mathbf{f}^\top(k)\mathbf{F}(k)\mathbf{f}(k)\right]^{-1} \tag{3.27}$$

Using this we get:

$$\Psi^{-1}(N+1) = \beta\Psi^{-1}(N+1) + \alpha f(N+1)f^\top(N+1) \tag{3.28}$$

Using now the following relation:

$$[A + BCD]^{-1} = A^{-1} - A^{-1}\left[C^{-1} + DA^{-1}B\right]^{-1} DA^{-1}$$

the previous relation becomes:

$$\begin{aligned} \Psi(N+1) &= \beta^{-1}\Psi(N) - \beta^{-1}\Psi(N)f(N+1) \\ &\quad \times \left[\alpha^{-1} + \beta^{-1}f^\top(N+1)\Psi(N)f(N+1)\right]^{-1} \beta^{-1}f^\top(N+1)\Psi(N) \end{aligned}$$

For the second term in the expression of $\theta^\star$ we have:

$$\begin{aligned}\mathbf{f}^\top(N)\mathbf{F}(N)\mathbf{y}(N) &= \begin{bmatrix} f(n) & \cdots & f(N+1) \end{bmatrix} \begin{bmatrix} \alpha\beta^{N+1-n} & & \\ & \ddots & \\ & & \alpha \end{bmatrix} \begin{bmatrix} y(n) \\ \vdots \\ y(N) \\ y(N+1) \end{bmatrix} \\ &= \beta\mathbf{f}^\top(N)\mathbf{F}(N)\mathbf{y}(N) + \alpha f(N+1)y(N+1)\end{aligned}$$

Combining the previous relations and after some algebraic manipulations we get:

$$\begin{aligned}\Psi(N) &= [\mathbf{f}^\top(N)\mathbf{F}(N)\mathbf{f}(N)]^{-1} \\ Q(N+1) &= \beta^{-1}\Psi(N)f(N+1)\left[\alpha^{-1} + \beta^{-1}f^\top(N+1)\Psi(N)f(N+1)\right]^{-1}\beta^{-1}f^\top(N+1)\Psi(N) \\ \theta(N+1) &= \theta(N) + Q(N+1)\left[y(N+1) - f^\top(N+1)\theta(N)\right] \\ \Psi(N+1) &= \beta^{-1}\left[\mathbb{I} - Q(N+1)f^\top(N+1)\right]\Psi(N)\end{aligned}$$

which apply for $N \geq n$.

Example 3.2.1 *To show how to use this technique to identify a given system, let us consider the setup of the dc motor kit. It consists of a dc motor driving a mechanical load. We know that the system a single input single output and its transfer function between the speed of the shaft and the voltage is a first order of the following form:*

$$G(s) = \frac{K_m}{\tau_m s + 1}$$

where K_m and τ_m are the two parameters that we have to determine.

For this system we can use two ways to get the model. The first one consists of getting the data $((u(k), y(k))$ using an UART to communicate with a PC and then use the least square method to build the model. The second one consists of using the microcontroller and then take the system as a black box.

For the second method the gain K_m is determined at the steady state regime as the ratio between the output and the input voltage. While for the time constant, τ_m, we take it as the instant at which the output takes 63 *% of the steady state value of the output. This procedure can be programmed in our microcontroller and easily the model is established. We use this approach in our mechatronics laboratory.*

3.2.2 State Space Description Approach

Consider a dynamical system system described by the following state space description:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) \end{cases}$$

where $x(k) \in \mathbb{R}^n$ is the state and $u(k) \in \mathbb{R}^m$ is the input and $y(k) \in \mathbb{R}^l$ be the output.

It is important to notice that the state description is not unique and any transformation: $\tilde{x}(k) = T^{-1}x(k)$, with T a nonsingular matrix, will give another description:

$$\begin{cases} \tilde{x}(k+1) = \tilde{A}\tilde{x}(k) + \tilde{B}u(k) \\ y(k) = \tilde{C}\tilde{x}(k) + \tilde{D}u(k) \end{cases}$$

where $\tilde{A} = T^{-1}AT$, $\tilde{B} = T^{-1}B$, $\tilde{C} = CT$ and $\tilde{D} = D$.

To determine the model of this system we need to determine the matrices A, B, C and D. If the system is single input single output, we can compute the transfer function and proceed with the previous approach to establish the mathematical model. In the state space description, we try to determine the state space description (A, B, C, D) that matches the set of input-output data. In the literature there exist many approaches to identify system in state space description. The reader is invited to consult the literature for this topic. Here we will present a simple algorithm that can be used to determine the state space description.

Before presenting this algorithm we will establish some relations that the algorithm uses in its computation. If we denote by $u(k), u(k+1), \ldots, y(k), y(k+1), \ldots$ and $x(k), x(k+1), \ldots$ the sequences of inputs, outputs and states, it can be shown that the Hankel matrix, Y_h can be given by:

$$Y_h = \Gamma_i X + H_t U_h$$

where:

$$Y_h = \begin{bmatrix} y(k) & y(k+1) & \cdots & y(k+j-1) \\ y(k+1) & y(k+2) & \cdots & y(k+j) \\ \vdots & \vdots & \ddots & \vdots \\ y(k+i-1) & y(k+i) & \cdots & y(k+j+i-2) \end{bmatrix}$$

U_h is a Hankel block with the same block dimensions as Y_h containing the consecutive inputs

$$U_h = \begin{bmatrix} u(k) & u(k+1) & \cdots & u(k+j-1) \\ u(k+1) & u(k+2) & \cdots & u(k+j) \\ \vdots & \vdots & \ddots & \vdots \\ u(k+i-1) & u(k+i) & \cdots & u(k+j+i-2) \end{bmatrix}$$

X contains the consecutive state vectors:

$$X = \begin{bmatrix} x(k) & x(k+1) & \cdots & x(k+j-1) \end{bmatrix}$$

Γ_i as the extended observability matrix:

$$\Gamma_i = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{i-1} \end{bmatrix}$$

and H_t is the block Toeplitz matrix:

$$H_t = \begin{bmatrix} D & 0 & 0 & \cdots & 0 \\ CB & D & 0 & \cdots & y(k+j) \\ CAB & CB & D & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{i-2}B & CA^{i-3}B & CA^{i-4}B & \cdots & D \end{bmatrix}$$

Let H be the concatenation of the matrices H_1 and H_2, i.e.:

$$H = \begin{bmatrix} H_1 \\ H_2 \end{bmatrix}$$

with

$$H_1 = \begin{bmatrix} Y_{h1} \\ U_{h1} \end{bmatrix}$$
$$H_2 = \begin{bmatrix} Y_{h2} \\ U_{h2} \end{bmatrix}$$

where

$$Y_{h1} = \begin{bmatrix} y(k) & y(k+1) & \cdots & y(k+j-1) \\ y(k+1) & y(k+2) & \cdots & y(k+j) \\ \vdots & \vdots & \ddots & \vdots \\ y(k+i-1) & y(k+i) & \cdots & y(k+j+i-2) \end{bmatrix}$$

$$Y_{h2} = \begin{bmatrix} y(k+i) & y(k+i+1) & \cdots & y(k+i+j-1) \\ y(k+i+1) & y(k+i+2) & \cdots & y(k+i+j) \\ \vdots & \vdots & \ddots & \vdots \\ y(k+2i-1) & y(k+2i) & \cdots & y(k+j+2i-2) \end{bmatrix}$$

$$U_{h1} = \begin{bmatrix} u(k) & u(k+1) & \cdots & u(k+j-1) \\ u(k+1) & u(k+2) & \cdots & u(k+j) \\ \vdots & \vdots & \ddots & \vdots \\ u(k+i-1) & u(k+i) & \cdots & u(k+j+i-2) \end{bmatrix}$$

$$U_{h2} = \begin{bmatrix} u(k+i) & u(k+i+1) & \cdots & u(k+i+j-1) \\ u(k+i+1) & u(k+i+2) & \cdots & u(k+i+j) \\ \vdots & \vdots & \ddots & \vdots \\ u(k+2i-1) & u(k+2i) & \cdots & u(k+j+2i-2) \end{bmatrix}$$

that satisfy:

$$\begin{aligned} Y_{h1} &= \Gamma_i X_1 + H_t U_{h1} \\ Y_{h2} &= \Gamma_i X_2 + H_t U_{h2} \end{aligned}$$

with

$$\begin{aligned} X_1 &= \begin{bmatrix} x(k) & x(k+1) & \cdots & x(k+j-1) \end{bmatrix} \\ X_2 &= \begin{bmatrix} x(k+i) & x(k+i+1) & \cdots & x(k+i+j-1) \end{bmatrix} \end{aligned}$$

The following algorithm can be used to compute the matrices (A, B, C, D) in off-line:

1. calculate U and S in the SVD of H:

$$H = USV^\top = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix} \begin{bmatrix} S_{11} & 0 \\ 0 & 0 \end{bmatrix} V^\top$$

2. calculate the SVD of $U_{12}^\top U_{11} S_{11}$

$$U_{12}^\top U_{11} S_{11} = \begin{bmatrix} U_q & U_q^\perp \end{bmatrix} \begin{bmatrix} S_q & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_q^\top \\ V_q^{\perp\top} \end{bmatrix}$$

3. solve the following set of linear equations:

$$\begin{bmatrix} U_q^\top U_{12}^\top U(m+l+1:(i+1)(m+l),:)S \\ U(mi+li+m+1:(m+l)(i+1),:)S \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} U_q^\top U_{12}^\top U(1:mi+li,:)S \\ U(mi+li+1:mi+li+m,:)S \end{bmatrix}$$

This algorithm can be programmed to get the matrices (A, B, C, D) of the system. Another version for on-line computation can be also obtained and for interested reader we refer to the literature for more details.

3.3 Conclusions

In this chapter, we covered the mathematical modeling of dynamical systems. We presented the technique that uses the physics laws to generate the model. We also developed the identification technique that may be used in some circumstances to establish a valid model that describes adequately the dynamical system under consideration. Both of the techniques require experiment data to establish the desired model.

3.4 Problems

1. In this problem, we ask to build the mathematical model for the dc motor kit without neglecting L. We ask to establish:

(a) the transfer function
(b) the state space description

2. Establish the mathematical model of the two wheels robot
3. Establish the mathematical model for the levitation system
4. Consider a dynamical system with a transfer function that you can give. Write a Matlab program that generates a sequence of data $(u(k), y(k))$. Using this data, write a program in Matlab to identify the system and to establish the mathematical model. Compare the two models and conclude.
5. Consider a dynamical system in state space description. Write a Matlab prgram to generate the appropriate data to identify the system using the state space description approach. Using this data write a Matlab program to establish a state space description and compare it with the original one.

Part III

Transfer Function Approaches

Mechatronic systems are in general a combination of hardware and software to assure the desired tasks for which the system was designed for. The analysis and design of such mechatronic systems can be done using different approaches. Among these approaches we quote the ones based on transfer function and the ones using the state space techniques.

This part deals with the analysis and synthesize of mechatronic systems using the transfer function approach. Mainly, we will focus on the analysis of dynamical systems controlled by microcontrollers. We will learn how to determine the performances that the system has. The design of controllers is also tackled. Mainly, we will see how to state the control design problem and how to solve it. The design part focuses on the determination of the controller that gives the desired performances to our dynamical system.

This part is divided into two chapters. The first one treats the analysis tools. Particularly, we will see how we can transform a continuous-time model to a discrete-time version by choosing appropriately the sampling period. Once this period is chosen, a discrete-time version of the model of the system under consideration is obtained that will be used for analysis and design of the dynamical system under study. The first chapter in this part covers the different tools that we may use to get the system's performances. The second chapter presents the design techniques that may be used to design the appropriate controller that will guarantee the desired performances. The design approach is composed of two steps. The first one, determines what will be the structure of the appropriate controller that can guarantee the desired performances. The second one computes the controller's parameters. Some simulations results are needed before implementing the developed algorithm. Matlab and Simulink are used for this purpose.

4
Analysis Based on Transfer Function

After reading this chapter the reader will:

1. master the concept of the transfer function concept
2. be able to perform the analysis of any LTI system and determine the specifications that the system has
3. be able to compute any time response of LTI system for a given input
4. be able to check the stability of any dynamical system
5. plot the root locus of any LTI system and use it for analysis and design purpose
6. plot the Bode diagram of any LTI system and use it for analysis and design purpose

4.1 Introduction

Nowadays the microcontrollers are more powerful and their prices are affordable which makes their use attractive. In mechatronic systems they are used either for On/Off or continuous-time controls. In both cases, the microcontroller is the hearth

of the mechatronic system. In the On/Off case, it is used for security and control purposes. The algorithm in this case is easy and doesn't take time in general to compute the action to be taken. While for the continuous-time case, the microcontroller receives the data at each sampling period and compute the desired action according to a chosen algorithm. The computation in this case may take more time and more care should be taken to prevent surprises. In both cases interrupts are used. The microcontrollers we will use in this book must have a quite high processing speed.

In practice when controlling real processes using microcontrollers two structures can be adopted. In the first one, the error between the output and the reference is done in continuous-time and then sent to the microcontroller via analog-digital converter (A/D) and the control action is computed following the chosen algorithm, while in the second case, the output is converted to a digital value via a A/D. The reference in this case is fed in a digital form. The control action is computed in a similar way as for the first case. These two structures are illustrated respectively by Figs. 4.1 and 4.2.

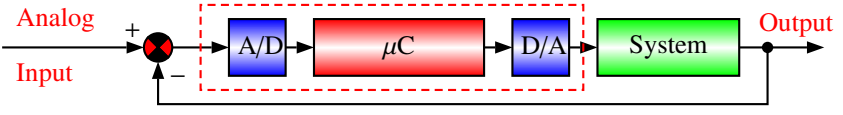


Fig. 4.1 Signal conversion is made in the forward path

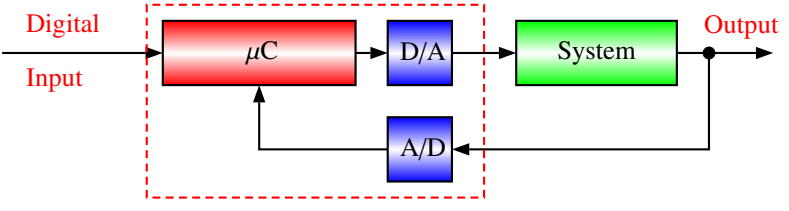


Fig. 4.2 Signal conversion is made in the feedback path

The second structure is more often used in practice and therefore, it is the one that we will use in the rest of the book.

Remark 4.1.1 *In the structure of Fig. 4.2, we have sampled signals that have the following advantages:*

1. *easy modulated*
2. *easy to code*
3. *easy to transmit and to regenerate*

and there are positive points.

In the rest of this chapter we will present the tools that can be used for the analysis of this type of system. Mainly we will show for a given practical system:

1. how to determine the sampling period
2. how to convert the continuous-time model to discrete-time one using the chosen sampling period
3. how to determine the performances of such system such as the stability, the overshoot, the settling time, etc.
4. how to use the root-locus and the Bode-plot techniques for discrete-time case

All these questions are addressed in the rest on this chapter. The rest of the chapter is organized as follows. In Section 2, the sampling process is developed and the relation between the continuous-time and the discrete-time is established. Mainly, the relationship between the poles is established for the two domains (s-domain and z-domain). Section 3 introduces the transfer function concept and the one of poles and zeros. In Section 4, the time response for a given input is developed and the approaches to compute it are presented. Section 5 covers the stability problem and the system error. The techniques of root locus and the Bode plot are developed respectively in Sections 6 and 7. These techniques are used in the analysis and design phases.

4.2 Sampling Process

Real practical processes are more often continuous systems that evolve continuously in time. Their analysis and design should be done carefully. In fact we will always need to convert the dynamics of such systems to a discrete-time corresponding one to analyze and proceed with the design of the controller that will be used to control them using microcontrollers. The choice of the sampling period is a critical problem. In fact a small sampling period will result with huge amount of data that the microcontroller will not be able to handle while a large one will give poor results and may be the system will not be controlled properly. The sampling period must be properly chosen to avoid such problems. It can be chosen smaller than the smallest time constant in the dynamics of the process. The bandwidth is also used for the choice of the sampling period.

The Shannon theorem is used for the choice of the sampling period. This statement of the Shannon theorem is given by the following result:

Theorem 4.2.1 *A signal $y(t)$ which contains no frequency components greater than f_h is uniquely reconstructed by a set of sampled from $y(t)$ spaced by $\frac{1}{2f_h}$. A proper choice for the sampling frequency should satisfy:*

$$f_s \geq 2f_h \tag{4.1}$$

Remark 4.2.1 *In practice the factor two is not enough and generally we choose the number more greater than two. A good choice consists of taking the sampling rate greater than* $30w_b$*, where* w_b *is the bandwidth of the closed-loop system.*

Remark 4.2.2 *It is important to notice that we have the following relations between the period,* T*, the frequency,* f *and the pulsation,* w*:*

$$\begin{aligned} T &= \frac{1}{f} \\ w &= 2\pi f \end{aligned}$$

which implies:

$$w = \frac{2\pi}{T}$$

Example 4.2.1 *To show how the sampling period can be chosen for a continuous-time system, let us consider a dynamical system with the following transfer function:*

$$G(s) = \frac{10}{(s+1)(s+2)(s+5)}$$

and determine the sampling period for this system.

First of all notice that there exist an infinite number of sampling of periods that can be chosen for this system. In this example we define the sampling period using two approaches.

From the transfer function of the system we conclude that the highest frequency in the system is $w_b = 5rad/s$*. This corresponds to faster dynamics in the system and therefore when sampling we should use this information and sample faster than this.*

Notice that we have $w_s T = 2\pi$*. Now if we sample thirty times of the highest frequency in the system, we have* $w_s = 30 \times 5 = 150rad/sec$*. This gives:*

$$T = \frac{2\pi}{150} = 0.021s$$

From the other side, the constant times of the system are respectively $\tau_1 = 1s$*,* $\tau_2 = 0.5s$ *and* $\tau_1 = 0.2s$*. This implies that fast dynamics in the system has a time constant equal to* $0.2s$*. A rule to select the sampling period consists of using the following formula:*

$$T = \frac{0.2}{a} \tag{4.2}$$

where a *is positive real number to be selected between* 7 *and* 14*. A proper choice is* 10*.*

Using this rule we get

$$T = 0.02s$$

Once the sampling period is chosen the next step is to convert the continuous-time dynamics to an equivalent discrete-time one. In fact, if the sampling period,

T, is properly chosen the real output can be obtained from the sampled one for a given input and therefore there is no lost of information.

The conversion from continuous-time system to sampled system passes through two devices:

- sampler
- zero-order-hold (ZOH)

The role of the sampler is to convert the continuous-time signal to an equivalent train of pulses while the ZOH blocks the values received from the sampler to make them available to the microcontroller that reads them through the analog/digital converter. The sampling process is illustrated in Fig. 4.3.

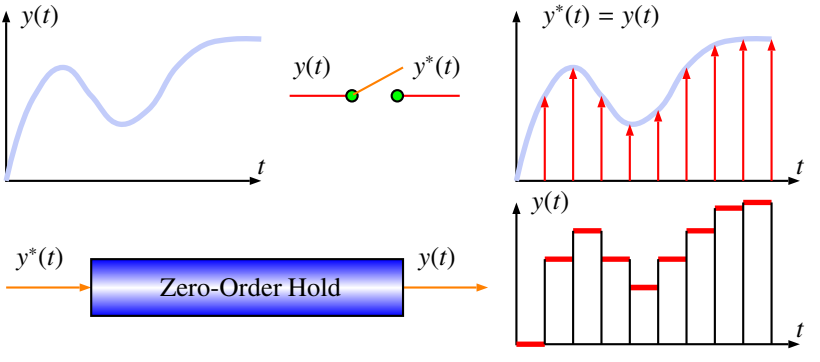


Fig. 4.3 Sampling process

The main objective of the sampling process of a signal $y(t)$ is to keep most of its information in the sampled one. It is also important to notice that the number of bits of the used microcontroller to process this signal has a significant effect on the quantization and therefore on the result. The quantization is the process of approximating the continuous range of values by a set of discrete integer values. It is also referred to as the discretization of the amplitude values of the signal. If a microcontroller with 16 bits is used, we will have $2^{16} = 65536$ possible values per sample.

Let $y(t)$ be an analog signal whose maximum frequency that a sampler should take into account is f_h (bandwidth). Assume $y(t)$ is sampled at frequency f_s. Shannon theorem states that it is possible to reconstruct the signal $y(t)$ from $y^*(t) = y(kT)$ if and only if $f_s \geq 2f_h$. Mathematically, the sampling process of an analog signal can be seen as a mathematical product between the signal $y(t)$ and a train of impulses. This is given by the following expression:

$$y(kT) = \sum_{k=0}^{\infty} y^{\star}(t)\delta(t - kT) \tag{4.3}$$

where $\delta(t)$ is the Dirac impulse and T is the sampling period.

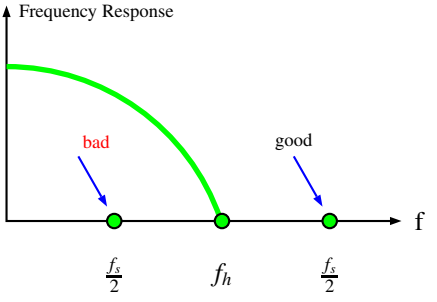


Fig. 4.4 Sampling period choice

For the continuous-time systems, the Laplace transform has been used to transform the set of linear differential equations that describes the dynamics into an algebraic one and the concepts of the transfer function or the transfer matrix function have been defined. Then, the tools for analysis and design that have been developed can be used for this purpose. For sampled systems we will use the same approach since their dynamics are equivalent and the transformation used for the analysis and design called the $\mathscr{Z}$-transform is obtained from the Laplace transform. There exist many similarities between the two transformations.

Let $f^*(t)$ be a sampled signal, such as:

$$\mathscr{L}\left[f^*(t)\right] = F^*(s) = \int_0^{\infty} f^*(t)e^{-st}dt$$

where $f^*(t)$ is equal to zero everywhere except at instants $t = kT$, where $k = 0, 1, 2, 3, \cdots$

As an example of the signal $f(t)$ we give the step signal defined as follows:

$$f(t) = \begin{cases} 1 & \forall t \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Notice that the Laplace transform of $f^*(t)$, yields:

$$\int_0^{\infty} f^*(t)e^{-st}dt = \sum_{k=0}^{\infty} f(kT)e^{-skT} = \sum_{k=0}^{\infty} f(kT)(e^{sT})^{-k} = F^*(s)$$

The $\mathscr{Z}$-transform of $f(t)$ is defined as equal to Laplace transform of $f^*(t)$:

$$\mathscr{Z}\left[f(t)\right] = \mathscr{L}\left[f^*(t)\right]$$

Now, if we define $z = e^{sT}$, then we have:

$$F(z) = \sum_{k=0}^{\infty} f(kT)z^{-k}$$

This expression can be used to compute the $\mathscr{Z}$-transform of any signal.

Example 4.2.2 *Let us now give some examples to show how we use the $\mathscr{Z}$-transform and its properties.*

- *$\mathscr{Z}$ transform of the unit pulse function:*

$$f(k) = \begin{cases} 1 & when\ k = 0 \\ 0 & otherwise \end{cases}$$

Using the definition of $\mathscr{Z}$-transform, we get:

$$F(z) = \sum_{k=0}^{\infty} f(k)z^{-k} = 1.z^{-0} = 1$$

- *$\mathscr{Z}$-transform of the unit step function:*

$$f(k) = \begin{cases} 1 & when\ k \geq 0 \\ 0 & otherwise \end{cases}$$

Proceeding similarly we have:

$$F(z) = \sum_{k=0}^{\infty} f(k)z^{-k} = \sum_{k=0}^{\infty} z^{-k} = \sum_{k=0}^{\infty} (z^{-1})^k$$

This last expression is equivalent to the following series

$$\sum_{k=0}^{\infty} a^k$$

that will converge if $|a| < 1$ and we get:

$$\sum_{k=0}^{\infty} a^k = \frac{1}{1-a}$$

To get this relation notice that if we let:

$$S = \sum_{k=0}^{\infty} a^k$$

Computing $S - aS$ implies the results.
Using this, we get:

$$F(z) = \frac{1}{1-z^{-1}}$$

- *$\mathscr{Z}$-transform of the exponential function:*

$$f(k) = \begin{cases} a^k & \text{when } k = 0 \\ 0 & \text{otherwise} \end{cases}$$

Using again the definition of $\mathscr{Z}$-transform, we get:

$$F(z) = \sum_{k=0}^{\infty} a^k z^{-k} = \sum_{k=0}^{\infty} (az^{-1})^k = \frac{1}{1 - az^{-1}}$$

provided that $|az^{-1}| < 1$.

The Table 4.1 gives $\mathscr{Z}$-transform table of some common signals. The computation of the $\mathscr{Z}$-transform of these functions is left as exercises for the reader.

From basic course on control system, the Laplace transform has interesting properties like linearity, homogeneity, etc. and since the $\mathscr{Z}$-transform is obtained from this transform, the properties of the $\mathscr{Z}$-transform are directly obtained:

- Linearity - The $\mathscr{Z}$-transform is a linear operator:

$$\begin{aligned} \mathscr{Z}\left[f_1(t) \pm f_2(t)\right] &= \mathscr{Z}\left[f_1(t)\right] \pm \mathscr{Z}\left[f_2(t)\right] = F_1(z) \pm F_2(z) \\ \mathscr{Z}\left[af(t)\right] &= aF(z) \end{aligned}$$

where a is real number, $f(t)$, $f_1(t)$ and $f_2(t)$ are given functions that admit Laplace transform, $F(z)$, $F_1(z)$ and $F_2(z)$ are the $\mathscr{Z}$-transform of the functions $f^\star(t)$, $f_1^\star(t)$ and $f_2^\star(t)$ respectively.

- Initial value theorem

$$\lim_{k \to 0} f(kT) = \lim_{z \to \infty} F(z)$$

- Final value theorem

$$\lim_{k \to \infty} f(kT) = \lim_{z \to 1} (1 - z^{-1}) F(z)$$

- Shift property:

$$\mathscr{Z}\left[f(t - kT)\right] = z^{-k} F(z)$$

- Back-shift property:

$$\mathscr{Z}\left[f(t + nT)\right] = z^n \left[F(z) - \sum_{k=0}^{n-1} f(kT) z^{-k} \right]$$

then, for $k = 0, 1, 2, \cdots n$, we have:

$$\begin{aligned} \mathscr{Z}\left[f(t + T)\right] &= \mathscr{Z}\left[f[(k+1)T]\right] = zF(z) - zf(0) \\ \mathscr{Z}\left[f(t + 2T)\right] &= \mathscr{Z}\left[f[(k+2)T]\right] = z^2 F(z) - z^2 f(0) - zf(T) \\ \mathscr{Z}\left[f(t + nT)\right] &= \mathscr{Z}\left[f[(k+n)T]\right] = z^n F(z) - z^n f(0) \\ & \quad - z^{n-1} f(T) - \cdots - zf((n-1)T) \end{aligned}$$

Table 4.1 Z-transform table

$F(s)$	$f(t)$ or $f(k)$	$F(z)$
1	$\delta(t)$	1
e^{-kTs}	$\delta(t-kT)$	z^{-k}
$\frac{1}{s}$	$1(t)$	$\frac{z}{z-1}$
$\frac{1}{s^2}$	t	$\frac{Tz}{(z-1)^2}$
$\frac{2}{s^3}$	t^2	$\frac{T^2z(z+1)}{(z-1)^3}$
$\frac{(k-1)!}{s^k}$	t^{k-1}	$\lim_{a\to 0}(-1)^{k-1}\frac{\partial^{k-1}}{\partial a^{k-1}}\left[\frac{z}{z-e^{-aT}}\right]$
$\frac{(k-1)!}{(s+a)^k}$	$t^k e^{-aT}$	$(-1)^k\frac{\partial^k}{\partial a^k}\left[\frac{z}{z-e^{-aT}}\right]$
$\frac{a^2}{s(s+a)^2}$	$1-e^{-at}(1+at)$	$\frac{z\left[\alpha z+\beta\right]}{(z-1)(z-e^{-aT})^2}$ $\alpha = 1-e^{-at}-aTe^{-at}$ $\beta = e^{-2at}-e^{-at}+aTe^{-at}$
$\frac{1}{s+a}$	e^{-at}	$\frac{z}{z-e^{-aT}}$
$\frac{a}{s(s+a)}$	$1-e^{-at}$	$\frac{z(1-e^{-aT})}{(z-1)(z-e^{-aT})}$
$\frac{1}{(s+a)^2}$	te^{-at}	$\frac{Tze^{-aT}}{(z-e^{-aT})^2}$
$\frac{1}{(s+a)(s+b)}$	$\frac{1}{b-a}\left(e^{-at}-e^{-bt}\right)$	$\frac{1}{b-a}\left[\frac{z}{z-e^{-aT}}-\frac{z}{z-e^{-bT}}\right]$
$\frac{(b-a)s}{(s+a)(s+b)}$	$be^{-bt}-ae^{-at}$	$\frac{z\left[z(b-a)-(be^{-aT}-ae^{-bT})\right]}{(z-e^{-aT})(z-e^{-bT})}$
$\frac{a}{s^2(s+a)}$	$t-\frac{1}{a}\left(1-ae^{-at}\right)$	$\frac{Tz}{(z-1)^2}-\frac{z\left(1-e^{-aT}\right)}{a(z-1)(z-e^{-aT})}$
$\frac{s}{(s+a)^2}$	$(1-at)e^{-at}$	$\frac{z\left[z-e^{-aT}(1+aT)\right]}{(z-e^{-aT})^2}$
$\frac{\omega}{s^2+\omega^2}$	$\sin\omega t$	$\frac{z\sin\omega T}{z^2-2z\cos\omega T+1}$
$\frac{s}{s^2+\omega^2}$	$\cos\omega t$	$\frac{z(z-\cos\omega T)}{z^2-2z\cos\omega T+1}$
$\frac{\omega}{(s+a)^2+\omega^2}$	$e^{-at}\sin\omega t$	$\frac{ze^{-aT}\sin\omega T}{z^2-2ze^{-aT}\cos\omega T+e^{-2aT}}$
$\frac{s+a}{(s+a)^2+\omega^2}$	$e^{-at}\cos\omega t$	$\frac{z^2-ze^{-aT}\cos\omega T}{z^2-2ze^{-aT}\cos\omega T+e^{-2aT}}$
	a^k	$\frac{z}{z-a}$
	$a^k cosk\pi$	$\frac{z}{z+a}$

Example 4.2.3 *Let us compute the $\mathscr{Z}$-transform of the ramp. This function is defined mathematically as follows:*

$$f(t)=\begin{cases} t & when\ t\geq 0 \\ 0 & otherwise \end{cases}$$

At sampling instants the function takes values as follows:

$$f(kT)=kT, k=0,1,2,\cdots$$

where T is the sampling period.

Using now the definition of the $\mathscr{Z}$-transform, we get:

$$\begin{aligned} F(z) &= \sum_{k=0}^{\infty} f(kT)z^{-k} \\ &= \sum_{k=0}^{\infty} kTz^{-k} \\ &= T\sum_{k=0}^{\infty} kz^{-k} \\ &= T\left(0 + z^{-1} + 2z^{-2} + \cdots + kz^{-k} + \cdots\right) \\ &= T\frac{z^{-1}}{(1-z^{-1})^2} \\ &= \frac{Tz}{(z-1)^2} \end{aligned}$$

Example 4.2.4 *Let us compute the $\mathscr{Z}$-transform of the exponential function. This function is defined mathematically as follows:*

$$f(t) = \begin{cases} e^{-\alpha t} & when\ t \geq 0 \\ 0 & otherwise \end{cases}$$

At sampling instants the function takes values as follows:

$$f(kT) = e^{-k\alpha T}, k = 0, 1, 2, \cdots$$

where T is the sampling period.

Using now the definition of the $\mathscr{Z}$-transform, we get:

$$\begin{aligned} F(z) &= \sum_{k=0}^{\infty} f(kT)z^{-k} \\ &= \sum_{k=0}^{\infty} e^{-k\alpha T}z^{-k} \\ &= 1 + e^{-\alpha T}z^{-1} + e^{-2\alpha T}z^{-2} + \cdots + e^{-k\alpha T}z^{-k} + \cdots \\ &= \frac{1}{1 - e^{-\alpha T}z^{-1}} \\ &= \frac{z}{z - e^{-\alpha T}} \end{aligned}$$

Example 4.2.5 *Let us consider the computation of $\mathscr{Z}$-transform of the following function:*

$$f(t) = \begin{cases} cos(wt) & for\ t \geq 0 \\ 0 & otherwise \end{cases}$$

At the sampling instants, we have:

$$f(kT) = \begin{cases} \cos(kwT) & for\ \ k = 0, 1, 2, \cdots, \\ 0 & otherwise \end{cases}$$

Using the definition of the $\mathscr{Z}$-transform, we get:

$$\begin{aligned} F(z) &= \sum_{k=0}^{\infty} f(kT) z^{-k} \\ &= \sum_{k=0}^{\infty} \cos(kwT) z^{-k} \end{aligned}$$

Notice that

$$\cos(kwT) = \frac{e^{jkwT} + e^{-jkwT}}{2}$$

Using this we have:

$$\begin{aligned} F(z) &= \frac{1}{2} \mathscr{Z} \left[e^{jkwT} + e^{-jkwT} \right] \\ &= \frac{1}{2} \left[\frac{1}{1 - e^{jwT} z^{-1}} + \frac{1}{1 - e^{-jwT} z^{-1}} \right] \end{aligned}$$

Using now the fact that $e^{jwT} = \cos(wT) + j \sin(wT)$, we get:

$$F(z) = \frac{z^2 - z \cos(wT)}{z^2 - 2z \cos(wT) + 1}$$

Example 4.2.6 *Let us compute the $\mathscr{Z}$-transform of a complex function. For this purpose, let us consider the following function:*

$$f(t) = \begin{cases} e^{-\alpha t} \cos(wt) + e^{-\alpha t} \sin(wt) & when\ t \geq 0 \\ 0 & otherwise \end{cases}$$

At sampling instants the function takes values as follows:

$$f(kT) = e^{-\alpha kT} \cos(wkT) + e^{-\alpha kT} \sin(wkT), k = 0, 1, 2, \cdots$$

where T is the sampling period.

Using now the definition of the $\mathscr{Z}$-transform, we get:

$$\begin{aligned} F(z) &= \sum_{k=0}^{\infty} f(kT) z^{-k} \\ &= \sum_{k=0}^{\infty} \left[e^{-\alpha kT} \cos(wkT) + e^{-\alpha kT} \sin(wkT) \right] z^{-k} \end{aligned}$$

Using now the facts that:

$$\begin{aligned}\cos(kwT) &= \frac{e^{jwkT} + e^{-jwkT}}{2}\\ \sin(kwT) &= \frac{e^{jwkT} - e^{-jwkT}}{2j}\end{aligned}$$

Using now the linearity property of the $\mathscr{Z}$-transform we get:

$$\begin{aligned}F(z) &= \sum_{k=0}^{\infty} e^{-\alpha kT}\cos(wkT)z^{-k} + \sum_{k=0}^{\infty} e^{-\alpha kT}\sin(wkT)z^{-k}\\ &= \frac{1}{2}\sum_{k=0}^{\infty} e^{-\alpha kT}\left[e^{jwkT} + e^{-jwkT}\right]z^{-k} + \frac{1}{2j}\sum_{k=0}^{\infty} e^{-\alpha kT}\left[e^{jwkT} - e^{-jwkT}\right]z^{-k}\\ &= \frac{z^2 - e^{-\alpha T}z\cos(wT)}{z^2 - 2e^{-\alpha T}z\cos(wT) + e^{-2\alpha T}} + \frac{e^{-\alpha T}z\sin(wT)}{z^2 - 2e^{-\alpha T}z\cos(wT) + e^{-2\alpha T}}\\ &= \frac{z^2 - e^{-\alpha T}z\cos(wT) + e^{-\alpha T}z\sin(wT)}{z^2 - 2e^{-\alpha T}z\cos(wT) + e^{-2\alpha T}}\\ &= \frac{z^2 + e^{-\alpha T}z\left[\sin(wT) - \cos(wT)\right]}{z^2 - 2e^{-\alpha T}z\cos(wT) + e^{-2\alpha T}}\end{aligned}$$

Previously we were able to compute the $\mathscr{Z}$-transform of a signal that may represent the output system that corresponds to a given input. Sometimes we are interested by knowing its expression in time. The inverse $\mathscr{Z}$-transform may be used for this purpose. To perform the inverse $\mathscr{Z}$-transform we can use the following methods:

- expansion into partial fraction
- polynomial division
- residues method

The inverse $\mathscr{Z}$-transform consists of finding the expression of $f(k)$ that corresponds to a given function $F(z)$. A very useful method to find the inverse transform of the function $F(z)$ is the expansion into partial fractions whose inverse transforms can be found in the table. The idea behind this method is firstly write the expression of the function in term of z^{-1}, then perform the expansion into partial fraction as usually done for the continuous-time case. This technique is illustrated by the following example.

Example 4.2.7 *Let us consider the following function $F(z)$*

$$F(z) = \frac{2z^2}{2z^2 - 3z + 1}$$

and determine the expression of $f(k)$.

To answer this question, let us first of all divide the numerator and the denominator simultaneously by $2z^2$. This gives:

$$F(z) = \frac{1}{1 - \frac{3}{2}z^{-1} + \frac{1}{2}z^{-2}}$$

It is not obvious how an inverse transform looks like, but if we factorize the denominator of $F(z)$, then partial expansion gives:

$$F(z) = \frac{1}{(1 - z^{-1})(1 - \frac{1}{2}z^{-1})} = \frac{A}{(1 - z^{-1})} + \frac{B}{(1 - \frac{1}{2}z^{-1})}$$

As for Laplace transform, the residues are :

$$A = \lim_{z \to 1} \frac{(1 - z^{-1})}{(1 - z^{-1})(1 - \frac{1}{2}z^{-1})} = 2$$

$$B = \lim_{z \to \frac{1}{2}} \frac{(1 - \frac{1}{2}z^{-1})}{(1 - z^{-1})(1 - \frac{1}{2}z^{-1})} = -1$$

Finally, we obtain:

$$F(z) = \frac{2}{(1 - z^{-1})} + \frac{-1}{(1 - \frac{1}{2}z^{-1})}$$

and its inverse transform is

$$f(k) = \mathscr{Z}^{-1}[F(z)] = 2 - \left(\frac{1}{2}\right)^k$$

The second method that can be used to compute the inverse $\mathscr{Z}$-transform is the polynomial division method. This technique consists of performing the polynomial division of the numerator by the denominator of the function $F(z)$. To illustrate this method us consider the previous example.

Example 4.2.8 *To show how the polynomial division works, let us continue the same expression for $F(z)$ as for the previous example.*

$$F(z) = \frac{1}{1 - \frac{3}{2}z^{-1} + \frac{1}{2}z^{-2}}$$

Dividing the numerator by the denominator, we obtain :

$$F(z) = 1 + \frac{3}{2}z^{-1} + \frac{7}{4}z^{-2} + \frac{15}{8}z^{-3} + \cdots$$

Since $\mathscr{Z}[\delta(t - kT)] = z^{-k}$, we then obtain:

$$f(kT) = \delta(t) + \frac{3}{2}\delta(t - T) + \frac{7}{4}\delta(t - 2T) + \frac{15}{8}\delta(t - 3T) + \cdots$$

Example 4.2.9 *In this example, we consider the following function:*

$$F(z) = \frac{0.3z}{z^2 - 1.7z + 0.7} = \frac{0.3z^{-1}}{1 - 1.7z^{-1} + 0.7z^{-2}}$$

Polynomial division gives :

$$F(z) = 0.3z^{-1} + 0.51z^{-2} + 0.657z^{-3} + \cdots$$

According to the table of $\mathscr{Z}$-transform, we have :

$$f(kT) = 0.3\delta(t-T) + 0.5\delta(t-2T) + 0.657\delta(t-3T) + \cdots$$

As a third method to compute the inverse $\mathscr{Z}$-transform we can use the the method of residues. It consists of using the following expression:

$$f(kT) = \text{sum of residues of } \left[z^{k-1}F(z)\right] = \sum_{n=1}^{n_s} r_n$$

n_s is the number of singularities of $F(z)$

r_n is the residue of $(z-z_n)F(z)z^{k-1}$ corresponding to the singularity z_n
$r_n = \lim_{z\to z_n}(z-z_n)F(z)z^{k-1}$

Example 4.2.10 *Let us consider the following expression:*

$$F(z) = \frac{0.3z}{(z-1)(z-0.7)}$$

and compute the corresponding $f(kT)$.

The inverse transform $f(kT)$ of the function $F(z)$ is:

$$f(kT) = \textit{sum of residues of } \left[z^{k-1}\frac{0.3z}{(z-1)(z-0.7)}\right] \textit{ at z=1 and z=0.7}$$

$$f(kT) = \frac{0.3z}{(z-1)(z-0.7)}z^{k-1}(z-1)|_{z=1} + \frac{0.3z}{(z-1)(z-0.7)}z^{k-1}(z-0.7)|_{z=0.7}$$

$$f(kT) = \frac{0.3z^k}{z-0.7}|_{z=1} + \frac{0.3z^k}{z-1}|_{z=0.7} = 1-(0.7)^k$$

Example 4.2.11 *As another example of how to compute the residue for an expression that contains multiple poles, let us consider the following expression:*

$$F(z) = \frac{0.5z^2}{(z-1)^2(z-0.5)}$$

and compute the corresponding $f(kT)$.

The inverse transform $f(kT)$ of the function $F(z)$ is:

$$\begin{aligned}
f(kT) &= \textit{sum of residues of } \left[z^{k-1}\frac{0.5z^2}{(z-1)^2(z-0.5)}\right] \textit{ at z=1 and z=0.5}\\
&= \frac{1}{(2-1)!}\lim_{z\to 1}\frac{d}{dz}\left[\frac{0.5z^{k+1}}{(z-1)^2(z-0.5)}(z-1)^2\right]\\
&+\left[\frac{0.5z^{k+1}}{(z-1)^2(z-0.5)}(z-0.5)\right]|_{z=0.5}\\
f(kT) &= \left[\frac{0.5(k+1)z^k(z-0.5)-0.5z^{k+1}}{(z-0.5)^2}z^{k-1}\right]|_{z=1}+\\
&\left[\frac{0.5z^{k+1}}{(z-1)^2(z-0.5)}(z-0.5)\right]|_{z=0.5}\\
f(kT) &= \frac{0.5z^{k+1}}{z-0.5}|_{z=1}+\frac{0.5z^{k+1}}{(z-1)^2}|_{z=0.5} = 1-(0.5)^k
\end{aligned}$$

In order to understand well the $\mathscr{Z}$-transform, let us see how the complex s-plane is transformed. Based on the definition of the $\mathscr{Z}$-transform, the main relationship between the s-plane and the z-plane is given by $z = e^{sT}$. This expression gives the mapping, called M of the s-plane into the z-plane. Therefore for any s in the s-plane we get the following point in the z-plane:

$$M(s) = e^{sT} = \text{corresponding value in } z$$

Let M^{-1} be the inverse transform, such as $s = \frac{1}{T}\ln z$. Usually, $s = \sigma \pm j\omega$ and $T = \frac{2\pi}{\omega_s}$ is the sampling period. Using these relations we get:

$$\begin{aligned}
M(s) &= M(\sigma \pm j\omega) = e^{(\sigma\pm j\omega)T} = e^{(\sigma\pm j\omega)\frac{2\pi}{\omega_s}} = e^{\frac{2\pi\sigma}{\omega_s}}.e^{\pm j\frac{2\pi\omega}{\omega_s}}\\
M(s) &= \|M\|e^{\pm j\theta}\\
M(s) &= [\text{magnitude}]\, e^{\pm[\text{angle}]}
\end{aligned}$$

Example 4.2.12 *Let us assume that $\sigma = 0$, i.e. all the roots are on the imaginary axis in the s-plane, and let us change ω from 0 to $\frac{\omega_s}{2}$. The corresponding roots in the z-plane are given by:*

$$z = e^{j\frac{2\pi\omega}{\omega_s}}$$

Table 4.2 Poles in the z-plane using $z = e^{j\frac{2\pi\omega}{\omega_s}}$

ω	corresponding poles
0	$z_1 = 1$
$\frac{\omega_s}{8}$	$z_2 = e^{j\frac{\pi}{4}} = \cos\frac{\pi}{4} + j\sin\frac{\pi}{4} = 0.707 + j0.707$
$\frac{\omega_s}{4}$	$z_3 = e^{j\frac{\pi}{2}} = \cos\frac{\pi}{2} + j\sin\frac{\pi}{2} = j1$
$\frac{\omega_s}{2}$	$z_4 = e^{j\pi} = \cos\pi + j\sin\pi = -1$

When σ is fixed, $\sigma = 0$, and making varying ω from 0 to $\frac{\omega_s}{2}$, we notice that the corresponding variable follows a half-circle of radius 1. This is shown in Fig. 4.5.

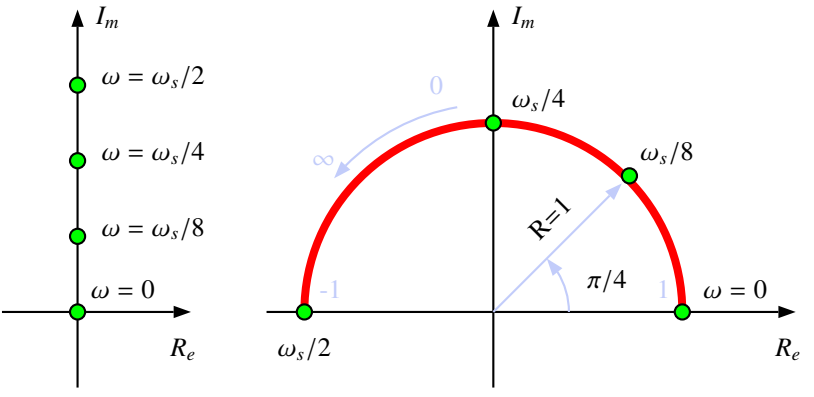


Fig. 4.5 Transformation of the s-plane into z-plane

When σ is fixed but not equal to zero, i.e.: $\sigma \neq 0$, then the radius of the circle is $R = e^{\sigma T}$. All the roots belong to the straight line $\omega = 0$ corresponding to an aperiodic response or oscillatory response. For all the other roots on the circle, the response is oscillatory.

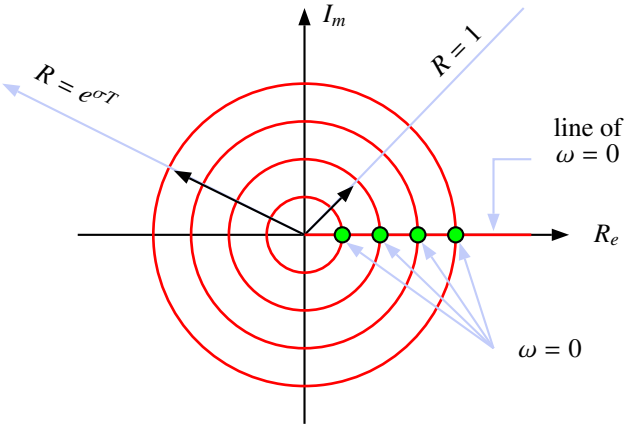


Fig. 4.6 Transformation of the s-plane when the real part is constant

For the resolution purposes, we will be interested to get the solution of a given difference equation for a fixed input. To obtain such solution we proceed as follows:

- we find the $\mathscr{Z}$-transform
- we take the inverse $\mathscr{Z}$-transform to find $y(kT)$

To illustrate this we consider the following example.

Example 4.2.13 *In this example, we consider the Fibonacci equation:*

$$y(k+2) = y(k+1) + y(k) \text{ with } y(0) = y(1) = 1$$

The $\mathscr{Z}$-transform of the Fibonacci equation is:

$$z^2Y(z) - z^2y(0) - zy(1) = zY(z) - zy(0) + Y(z)$$

that gives in turn:

$$Y(z) = \frac{z^2 - z}{z^2 - z - 1}y(0) + \frac{z}{z^2 - z - 1}y(1)$$

The roots of the characteristic equation are solution of the following equation:

$$z^2 - z - 1 = 0$$

which gives:

$$z_1 = 0.5 + \frac{\sqrt{5}}{2} \text{ and } z_2 = 0.5 - \frac{\sqrt{5}}{2}$$

Using for example the method of residues, we find:

$$y(k) = \frac{(1 + \sqrt{5})^{k+1} - (1 - \sqrt{5})^{k+1}}{2^{k+1}\sqrt{5}}$$

Each time, we introduce a sampling in analog operations, the transfer function should be transformed in the $\mathscr{Z}$-domain by:

$$z = e^{sT}$$

which corresponds to

$$s = \frac{1}{T}\ln z$$

The transformation $s = \frac{1}{T}\ln z$ is exact, but it is also difficult to implement in practice. That's the reason why we use two approximation methods:

- Numerical Integration
- Poles/zeros transforms

For numerical integration method care should taken when using it since we may get an unstable system after transformation. To illustrate the numerical integration approach, let us consider the following transfer function that represents a first order system:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{a}{s+a}, a > 0$$

which gives in time domain:

$$\frac{dy(t)}{dt} + ay(t) = au(t)$$

that gives in turn:

$$\int \frac{dy(t)}{dt} = \int (-ay(t) + au(t))$$

Integrating between 2 consecutive samples, i.e. from $(k-1)T$ to kT, we obtain:

$$y(kT) - y((k-1)T) = \int_{(k-1)T}^{kT} f(t)dt$$

where $f(t) = -ay(t) + au(t)$.

In this last equation, the major problem is how to integrate the right-hand term?

- First numerical integration method: The approximation of the integral is taken equal to the one of the area shown in the Fig. 4.7.

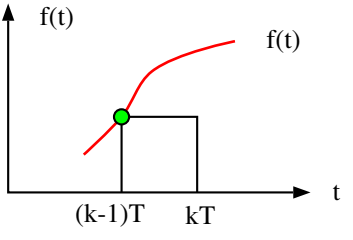


Fig. 4.7 Forward integration

Based on Fig. 4.7, we get:

$$y(kT) = y((k-1)T) + T\left[-ay((k-1)T) + au((k-1)T)\right]$$

that gives in turn:

$$y(kT) = y((k-1)T)\left[1 - aT\right] + aTu((k-1)T)$$

Using now the $\mathscr{Z}$-transform, we obtain:

$$Y(z) = z^{-1}Y(z)\left[1 - aT\right] + aTz^{-1}U(z)$$

Finally we get:

$$\frac{Y(z)}{U(z)} = \frac{aTz^{-1}}{1 - z^{-1}(1 - aT)} = \frac{a}{\frac{z-1}{T} + a}$$

Now if we compare the two transfer functions (in the s-domain and in the z-domain), we conclude that the expression in z-domain is obtained from the one in the s-domain by using the following transformation:

$$s = \frac{z-1}{T}$$

- Second numerical integration method: The approximation of the integral is taken equal to the one of the area shown in the Fig. 4.8.

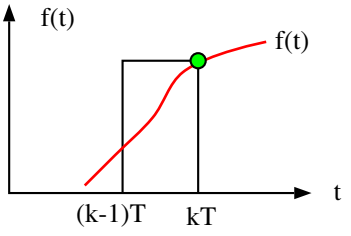


Fig. 4.8 Backward integration

Following the same steps as before and using now Fig. 4.8, we obtain:

$$y(kT) = y((k-1)T) + T\left[-ay(kT) + au(kT)\right]$$

that gives in turn in the z-domain:

$$Y(z) = z^{-1}Y(z) - aTY(z) + aTU(z)$$

From which we have:

$$\frac{Y(z)}{U(z)} = \frac{aT}{1 + aT - z^{-1}} = \frac{a}{\frac{z-1}{zT} + a}$$

Comparing again the two transfer functions as we did previously, we obtain the following transformation:

$$s = \frac{z-1}{zT}$$

- Third numerical integration method: In the two previous schemas, we have either underestimate or overestimate the area of the curve. Another alternate consists of computing the average of these two approaches. Referring now to the Fig. 4.9, we obtain the following for the approximation of the integral is that of the area shown in the figure.

$$y(kT) = y((k-1)T) + \frac{T}{2}\left[f(kT) + f((k-1)T)\right]$$

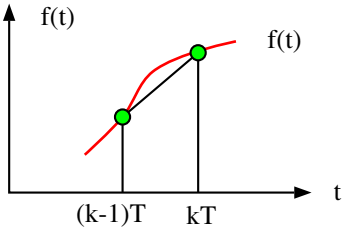


Fig. 4.9 Trapezoidal integration

From this expression we get:

$$Y(z) = z^{-1}Y(z) + \frac{T}{2}F(z) + \frac{T}{2}z^{-1}F(z)$$

Using now the expression of $F(z)$, we obtain:

$$Y(z) = z^{-1}Y(z) + \frac{T}{2}\left[-aY(z) + aU(z)\right] + \frac{T}{2}z^{-1}\left[-aY(z) + aU(z)\right]$$

that gives finally

$$\frac{Y(z)}{U(z)} = \frac{a}{\frac{2}{T}\left(\frac{z-1}{z+1}\right) + a}$$

Proceeding as before we get the following transformation:

$$s = \frac{2}{T}\left(\frac{z-1}{z+1}\right)$$

Example 4.2.14 *Consider the following transfer function:*

$$\frac{Y(s)}{U(s)} = \frac{1}{s^2 + 0.4s + 0.4} = \frac{1}{(s + 0.2 + j0.6)(s + 0.2 - j0.6)}$$

Our objective is to see the effect of the transformation we will use of the poles of the system. First of all, let us determine the sampling period. Since we have a second order, we have:

$$w_n = \sqrt{0.4} = 0.6325 rad/s$$

which gives $w_b = w_n$ and a proper choice for the sampling period is given by:

$$T = \frac{2\pi}{30 w_b} = 0.33 s$$

For this purpose let us compute the poles using the previous transformation for this system:

- *Using $s = \frac{z-1}{T}$, the corresponding transfer function is:*

$$G(z) = \frac{T^2}{z^2 + (-2 + 0.4T)z + 1 - 0.4T + 0.4T^2}$$

The poles of the system in the z-plane are: $z_{1,2} = 0.9338 \pm 0.1987\, j$

- *Using $s = \frac{z-1}{Tz}$, the corresponding transfer function is:*

$$G(z) = \frac{T^2}{(1 + 0.4T + 0.4T^2)z^2 + (-2 - 0.4T)z + 1}$$

The poles of the system in the z-plane are: $z_{1,2} = 0.9064 \pm 0.1689\, j$

- *Using $s = \frac{2}{T}\frac{z-1}{z+1}$, the corresponding transfer function is:*

$$G(z) = \frac{0.25T^2(z+1)^2}{(1 + 0.2T + 0.1T^2)z^2 + (-2 + 0.2T^2)z + 1 - 0.2T + 0.1T^2}$$

The poles of the system in the z-plane are: $z_{1,2} = 0.9182 \pm 0.1845\,j$

- *Using the transformation* $s = \frac{1}{T}\ln z$ *(*$z = e^{Ts}$*), the poles are* $0.9175 \pm 0.1847\,j$.

As it can be seen from this example that the trapezoidal approximation is the more close to the exact transformation since it gives almost the same poles. The other approximations give different results. Therefore the stability and precision should be tested before choosing a particular method.

As another approach that can be used to approximate the transfer function in $\mathscr{Z}$- domain is what it is always referred in the literature to as the poles/zeros transformation. It consists of doing the following steps:

- make all the poles of $G(s)$ correspond to $z = e^{-sT}$. That is, if $s = -a$, is a pole in the s-domain, then $G(z)$ will have a pole in the z-domain at $z = e^{-aT}$
- do the same thing for the zeros of $G(s)$
- place all the poles of $G(s)$ corresponding to $s = \infty$ at $z = -1$. This means adding $(z+1), (z+1)^2, \cdots$ to the numerator of $G(z)$ such that the degree of the numerator will be equal to the one of the denominator.
- make the gain of $G(s)$ correspond to the one of $G(z)$. This means that we must do the following for that:

$$[G(s)]_{s=0} = [G(z)]_{z=1}$$

Example 4.2.15 *To show how this procedure works, let us consider the following transfer function:*

$$G(s) = \frac{10}{(s+1)(s+2)}.$$

The poles of this transfer function are $s_1 = -1$ *and* $s_2 = -2$*. Their corresponding poles are respectively* $z_1 = e^{-T}$ *and* $z_2 = e^{-2T}$*. If we fix the sampling period to* $T = 0.02s$*, then these poles becomes* $z_1 = 0.9802$ *and* $z_2 = 0.9608$.

Since the denominator is of degree 2*, then the numerator also should be of degree* 2*. To do that, we add to the numerator the term* $(z+1)^2$.

The gain is then calculated by:

$$\left[\frac{10}{(s+1)(s+2)}\right]_{s=0} = \left[K\frac{(z+1)^2}{(z-0.9802)(z-0.9608)}\right]_{z=1}$$

$$1 = K\frac{4}{(0.285)(0.487)}$$

which gives:

$$K = 0.0019$$

Finally the transfer function in the $\mathscr{Z}$*-domain is given by:*

$$G(z) = \frac{0.0019(z+1)^2}{(z-0.9802)(z-0.9608)}$$

As another approach, it is possible to derive $G(z)$ from $G(s) = \frac{N(s)}{D(s)}$ when $D(s)$ has distinct roots. This can be computed using the following formula:

$$G(z) = \sum_{n=1}^{p} \frac{N(x_n)}{D'(x_n)} \frac{z}{z - e^{x_n T}}$$

with $D'(x_n) = \frac{\partial D}{\partial s}|_{s=x_n}$ for $n = 1, 2, 3, \cdots, p$

Example 4.2.16 *To show the idea how to get a $G(z)$ from a $G(s)$ with a denominator that has distinct roots, let us consider the following transfer function:*

$$G(s) = \frac{1}{(s+a)(s+b)} = \frac{1}{s^2 + (a+b)s + ab}$$

The denominator and the numerator of this transfer function are given by:

$$\begin{aligned} D(s) &= (s+a)(s+b) \\ N(s) &= 1 \end{aligned}$$

The denominator derivative with respect to s is given by:

$$D'(s) = 2s + (a+b)$$

The values of the derivatives at the two roots are:

$$\begin{aligned} D'(x_1 = -a) &= b - a \\ D'(x_2 = -b) &= -(b-a) \end{aligned}$$

Using this and the previous formula, we get:

$$G(z) = \frac{1}{b-a}\frac{z}{z - e^{-aT}} - \frac{1}{b-a}\frac{z}{z - e^{-bT}} = \left(\frac{1}{b-a}\right)\left[\frac{z}{z - e^{-aT}} - \frac{z}{z - e^{-bT}}\right]$$

4.3 Transfer Function Concept

The concept of transfer function for sampled systems can be defined similarly as it has been done for continuous-time one. To clarify this, let us refer to the Fig. 4.10 where the upstream sampler is a real one while the downstream one is a fictitious that we assume to be ideals and synchronized at the same sampling period. The second sampler is introduced for the purpose to define $Y(z)$ and therefore define properly the pulse transfer function. Based on the Fig. 4.10, we get:

$$Y(s) = G(s)U^{\star}(s)$$

Since the output is sampled by the fictitious sampler, we can then have:

$$\begin{aligned} Y^{\star}(s) &= \left[G(s)U^{\star}(s)\right]^{\star} \\ &= G^{\star}(s)U^{\star}(s) \end{aligned}$$

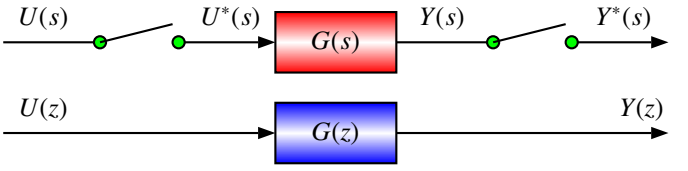


Fig. 4.10 Pulse transfer function definition

and if we apply the $\mathscr{Z}$-transform, we obtain:

$$Y(z) = G(z)U(z)$$

where $Y(z) = \mathscr{Z}[Y^\star(s)]$ and $U(z) = \mathscr{Z}[U^\star(s)]$.

This relation can be proved in an elegant way starting from the time domain. In fact, we have:

$$y(t) = \mathscr{L}^{-1}\left[G(s)U^\star(s)\right]$$

Using now the convolution theorem, we get:

$$y(t) = \int_0^t g(t-\sigma)u^\star(\sigma)d\sigma$$

From the other side we know that $u^\star(\sigma)$ can be written as follows:

$$u^\star(\sigma) = \sum_{k=0}^{\infty} u(kT)\delta(t-kT)$$

Using this, the expression of $y(t)$ becomes:

$$\begin{aligned} y(t) &= \int_0^t g(t-\sigma)\sum_{k=0}^{\infty} u(kT)\delta(t-kT)d\sigma \\ &= \sum_{k=0}^{\infty}\int_0^t g(t-\sigma)u(kT)\delta(t-kT)d\sigma \\ &= \sum_{k=0}^{\infty} g(t-kT)u(kT) \end{aligned}$$

Using now the definition of the $\mathscr{Z}$-transform of the sampled signal $y^\star(t)$ we have:

$$\begin{aligned} Y(z) &= \sum_{k=0}^{\infty} y(kT)z^{-k} \\ &= \sum_{k=0}^{\infty}\left[\sum_{l=0}^{\infty} g(kT-lT)u(lT)\right]z^{-k} \end{aligned}$$

Performing the change of variable, $m = k - l$, we get:

$$Y(z) = \sum_{m=0}^{\infty} \sum_{l=0}^{\infty} [g(mT)u(lT)]\, z^{-m-l}$$

that can be rewritten as follows:

$$\begin{aligned} Y(z) &= \left[\sum_{m=0}^{\infty} g(mT)z^{-m}\right]\left[\sum_{l=0}^{\infty} u(lT)z^{-l}\right] \\ &= G(z)U(z) \end{aligned}$$

Finally, the transfer function is given by:

$$G(z) = \frac{Y(z)}{U(z)}$$

which is the ratio between the $\mathscr{Z}$-transform of the output and $\mathscr{Z}$-transform of the input .

When manipulating the block diagrams of sampled systems, care should be taken. The following relations will help for this purpose.

- If $Y(s) = G(s)U(s)$, then

$$Y(z) = \mathscr{Z}\left[Y^*(s)\right] = \mathscr{Z}\left[[G(s)U(s)]^*\right] \neq \mathscr{Z}\left[G^*(s)U^*(s)\right] = G(z)U(z).$$

- If $Y(s) = G(s)U^*(s)$, then

$$Y(z) = \mathscr{Z}\left[Y^*(s)\right] = \mathscr{Z}\left[[G(s)U^*(s)]^*\right] = \mathscr{Z}\left[G^*(s)U^*(s)\right] = G(z)U(z).$$

Example 4.3.1 *In this example we consider the system of the Fig. 4.11 that represents two systems in serial with an ideal sampler between. The expression of the two transfer functions are:*

$$\begin{aligned} G_1(s) &= \frac{1}{s+a} \\ G_2(s) &= \frac{a}{s(s+a)} \end{aligned}$$

Our goal is to compute the equivalent transfer function for this system.

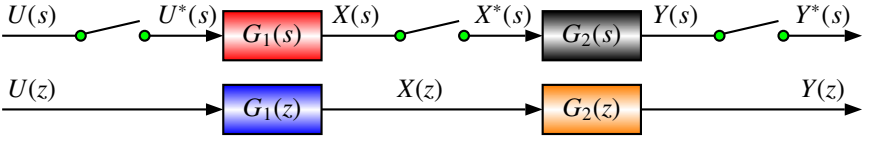


Fig. 4.11 Cascade transfer functions with sampler between

Based on this figure, we get:

$$\begin{aligned} Y^*(s) &= G_2^*(s)X^*(s) \\ X^*(s) &= G_1^*(s)U^*(s) \end{aligned}$$

which gives:

$$Y^*(s) = G_2^*(s)G_1^*(s)U^*(s)$$

that implies in turn:

$$\frac{Y(z)}{U(z)} = G_1(z)G_2(z)$$

Using the table of $\mathscr{Z}$-transform, we have:

$$\begin{aligned} \frac{Y(z)}{U(z)} &= G_1(z)G_2(z) = \frac{z}{(z-e^{-aT})}\frac{z(1-e^{-aT})}{(z-1)(z-e^{-aT})} \\ &= \frac{z^2(1-e^{-aT})}{(z-1)(z-e^{-aT})^2} \end{aligned}$$

Example 4.3.2 *In this example we consider the situation where the sample is removed between the two transfer function in serial. This situation is illustrated by the Fig. 4.12. The transfer function $G_1(s)$ and $G_2(s)$ are given by the following expression:*

$$\begin{aligned} G_1(s) &= \frac{a}{s+a} \\ G_2(s) &= \frac{a}{s(s+a)} \end{aligned}$$

where a is a positive scalar.

Our goal is to compute the equivalent transfer function and compare it with the one obtained in the previous example.

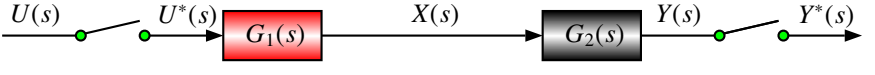


Fig. 4.12 Cascade transfer functions without sampler between

In this case we have:

$$\frac{Y^*(s)}{U^*(s)} = [G_1(s)G_2(s)]^*$$

that gives in turn

$$\frac{Y^*(s)}{U^*(s)} = \mathscr{Z}[G_1(s)G_2(s)] = G_1G_2(z)$$

It is important to notice that the equivalent transfer function we obtain for this case is different from the one we obtained for the system of the previous example.

Using the expression of $G_1(s)$ and $G_2(s)$, we get:

$$G_1(s)G_2(s) = \frac{a^2}{s(s+a)^2}$$

Based on the table of $\mathscr{Z}$-transform, we have:

$$\mathscr{Z}[G_1(s)G_2(s)] = G_1G_2(z) = \frac{z}{z-1} - \frac{z}{z-e^{-aT}} \frac{zaTe^{-aT}}{(z-e^{-aT})^2}$$

Example 4.3.3 *In this example we consider the case where we have transfer functions in feedback and we search to compute the equivalent one as we did in the previous examples. The system is illustrated by the Fig. 4.13. The transfer functions are given by the following expression:*

$$\begin{aligned} G(s) &= \frac{a}{s(s+a)} \\ H(s) &= \frac{1}{s} \end{aligned}$$

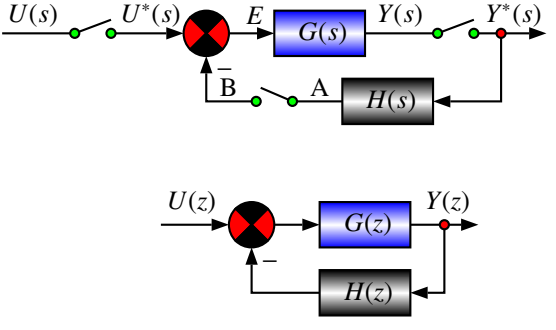


Fig. 4.13 Transfer functions in feedback

Based on this figure we have:

$$\begin{aligned} A &= H(s)Y^*(s) \\ B &= [A]^* = [H(s)Y^*(s)]^* = H^*(s)Y^*(s) \\ E &= U^*(s) - B = U^*(s) - H^*(s)Y^*(s) \\ Y(s) &= G(s)\left[U^*(s) - H^*(s)Y^*(s)\right] \end{aligned}$$

that gives in turn:

$$Y^*(s) = [Y(s)]^* = \left[G(s)\left[U^*(s) - H^*(s)Y^*(s)\right]\right]^* = G^*(s)\left[U^*(s) - H^*(s)Y^*(s)\right]$$

From this we get:

$$\frac{Y^*(s)}{U^*(s)} = \frac{G^*(s)}{1 + G^*(s)H^*(s)}$$

That gives the following pulse transfer function:

$$\frac{Y(z)}{U(z)} = \frac{G(z)}{1 + G(z)H(z)}$$

From the table of $\mathscr{Z}$-transform we get:

$$\begin{aligned} G(z) &= \frac{z(1 - e^{-at})}{(z-1)(z - e^{-aT})} \\ H(z) &= \frac{z}{z-1} \end{aligned}$$

Using this we obtain:

$$\begin{aligned} \frac{Y(z)}{U(z)} &= \frac{G(z)}{1 + G(z)H(z)} = \frac{\frac{z(1-e^{-at})}{(z-1)(z-e^{-aT})}}{1 + \frac{z(1-e^{-aT})}{(z-1)(z-e^{-aT})}\frac{z}{(z-1)}} \\ &= \frac{(1 - e^{-aT})z(z-1)(z - e^{-aT})}{(z-1)^2(z - e^{-aT}) + z^2(1 - e^{-aT})} \end{aligned}$$

Example 4.3.4 *As a second example of the previous case let us consider the system of the Fig. 4.14. The question is how to compute the pulse transfer function $F(z) = \frac{Y(z)}{G(z)}$ of this system.*

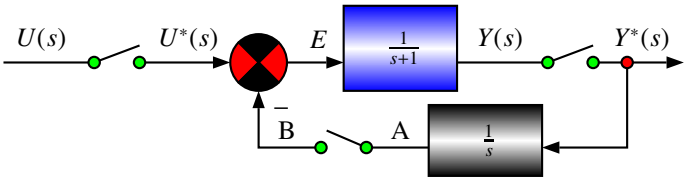


Fig. 4.14 Transfer functions in feedback

Since (see the table for $\mathscr{Z}$-transform)

$$G(z) = \mathscr{Z}\left[\frac{1}{s+1}\right] = \frac{z}{z - e^{-T}} \text{ and } H(z) = \mathscr{Z}\left[\frac{1}{s}\right] = \frac{z}{z-1}$$

we get the following expression for the closed-loop pulse transfer function:

$$\frac{Y(z)}{U(z)} = \frac{G(z)}{1 + G(z)H(z)} = \frac{z(z-1)}{(z - e^{-T})(z-1) + z^2}$$

Example 4.3.5 *In this example the system represented by the Fig. 4.15 where a zero order hold (ZOH) is used.*

1. *Find the open loop and closed loop pulse transfer functions $\frac{Y(z)}{U(z)}$*
2. *Find the unit-step response if $K = 1$ for $T = 0.1$*

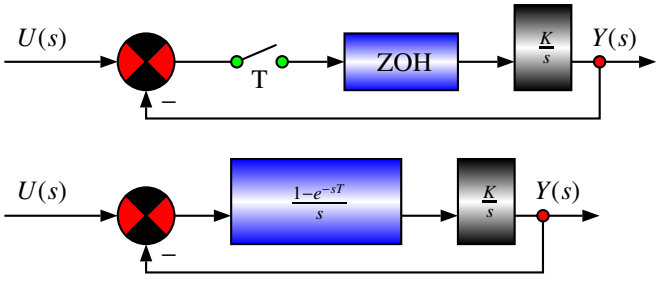


Fig. 4.15 Transfer functions in feedback

The solution of this example can be obtained easily. In fact we have:

- *Open loop:*

$$\frac{Y(s)}{U(s)} = \frac{K}{s^2}(1 - e^{-sT})$$

From which we have:

$$\frac{Y(z)}{U(z)} = \frac{KTz}{(z-1)^2}\frac{z-1}{z} = \frac{KT}{z-1}$$

Finally we obtain:

$$Y(z) = \frac{KT}{z-1}U(z)$$

- *Closed loop:*

$$Y(z) = \frac{KT/(z-1)}{1 + \frac{KT}{z-1}}U(z) = \frac{KT}{z-(1-KT)}\frac{z}{z-1}$$

Using the method of residues for $z_1 = 1$ and $z_2 = 1 - KT$, and the fact that $K = 1$, we find:

$$y(kT) = 1 - (1-T)^k \quad pour \quad k = 0, 1, 2, 3, \cdots$$

If we use $T = 0.1s$, we get:

$$y(k) = 1 - 0.9^k$$

Example 4.3.6 *Let us consider the system of the Fig. 4.16 and compute the transfer function.*

Using this figure, we have:

$$\begin{aligned} E(s) &= R(s) - H(s)Y(s) \\ Y(s) &= G(s)E^\star(s) \end{aligned}$$

which gives in turn:

$$\begin{aligned} E^\star(s) &= R^\star(s) - [H(s)Y(s)]^\star \\ Y^\star(s) &= G^\star(s)E^\star(s) \end{aligned}$$

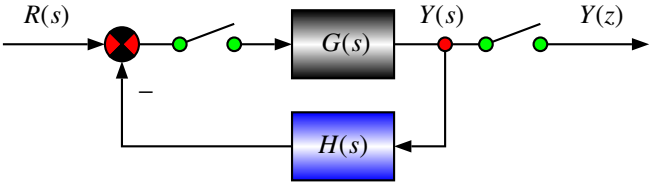


Fig. 4.16 Transfer functions in feedback

Using the $\mathscr{Z}$-transform, we obtain:

$$Y(z) = \frac{G(z)R(z)}{1 + GH(z)}$$

Example 4.3.7 *Let us consider the system of the Fig. 4.17 and compute the transfer function.*

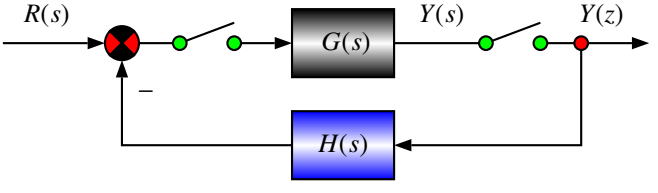


Fig. 4.17 Transfer functions in feedback

Using this figure, we have:

$$\begin{aligned} E(s) &= R(s) - H(s)Y^{\star}(s) \\ Y(s) &= G(s)E^{\star}(s) \end{aligned}$$

which gives in turn:

$$\begin{aligned} E^{\star}(s) &= R^{\star}(s) - H^{\star}(s)Y^{\star}(s) \\ Y^{\star}(s) &= G^{\star}(s)E^{\star}(s) \end{aligned}$$

Using now the $\mathscr{Z}$-transform, we obtain:

$$Y(z) = \frac{G(z)R(z)}{1 + G(z)H(z)}$$

Example 4.3.8 *Let us consider the dynamical system of the block diagram illustrated by Fig. 4.18*

$$Y(z) = \frac{RG(z)}{1 + HG(z)}$$

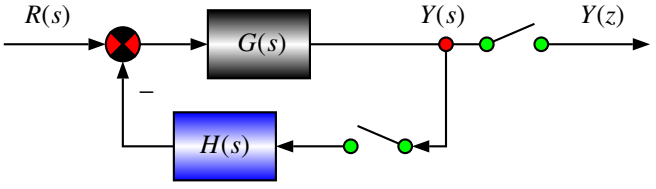


Fig. 4.18 Transfer functions in feedback

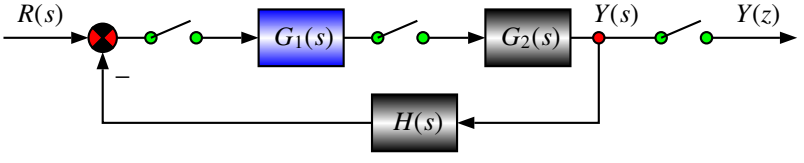


Fig. 4.19 Transfer functions in feedback

Example 4.3.9 *Let us consider the system of the block diagram of the figure 4.19 and compute the transfer function.*

$$Y(z) = \frac{G_2(z)RG_1(z)}{1 + G_1G_2H(z)}$$

Example 4.3.10 *Let us consider the system of the block diagram of the figure 4.20 and compute the transfer function.*

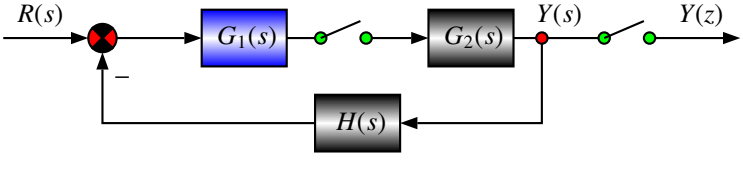


Fig. 4.20 Transfer functions in feedback

Using this figure, we have:

$$E(s) = R(s) - H(s)Y(s)$$
$$Y(s) = [G(s)E(s)]^{\star}$$

which gives in turn:

$$Y^\star(s) = \left[\left[(R(s) - H(s)Y(s))\,G_1(s)\right]^\star\right]^\star G_2(s)$$

Using now the $\mathscr{Z}$-transform, we obtain:

$$Y(z) = \frac{G_1(z)G_2(z)R(z)}{1 + G_1(z)G_2(z)H(z)}$$

Based on these examples, we are always able to compute the transfer function of the system and its expression is given by:

$$G(z) = \frac{Y(z)}{U(z)}$$

where $Y(z)$ and $U(z)$ are respectively the $\mathscr{Z}$-transform of the output $Y(s)$ and the input $U(s)$.

This transfer function is always in the following form:

$$\begin{aligned} G(z) &= \frac{N(z)}{D(z)} \\ &= \frac{b_n z^n + b_{n-1}z^{n-1} + \cdots + b_1 z + b_0}{z^n + a_{n-1}z^{n-1} + \cdots + a_1 z + a_0} \end{aligned}$$

where a_i and b_i are real scalars and n is an integer representing the degree of the system.

The roots of the polynomials $N(z)$ and $D(z)$, i.e.: the solutions of the following equations:

$$\begin{aligned} N(z) &= 0 \\ D(z) &= 0 \end{aligned}$$

are called respectively zeros and poles of the system.

The poles play an important role in the system response. Their location is very important and it related to the system performances like the stability, the transient regime, etc. as it will be shown later on.

Example 4.3.11 *Let us consider a dynamical system with the following transfer function:*

$$\begin{aligned} G(z) &= \frac{N(z)}{D(z)} \\ &= \frac{z^2 - z + 0.02}{z^3 - 2.4z^2 + z - 0.4} \end{aligned}$$

Compute the poles and zeros of the system and plot them in the z-domain.
From the expression of the transfer function we have:

$$\begin{aligned} N(z) &= z^2 - z + 0.02 \\ D(z) &= z^3 - 2.4z^2 + z - 0.4 = (z-2)\left(z^2 - 0.4z + 0.2\right) \end{aligned}$$

The roots of this polynomials are $0.1 \pm 0.1\,j$ *for the zeros and* 2 *and* $0.2 \pm 0.4\,j$ *for the poles. The zeros are all inside the unit circle. The complex poles are also inside the unit circle while the real one is outside this circle.*

We have introduced the concept of transfer function and we have learnt how to manipulate the block diagrams. It is now time to compute the time response of the system for given signal inputs. This is the subject of the next section.

4.4 Time Response and Its Computation

More often, the control system has to guarantee certain performances such as:

- the settling time at a given percentage
- the overshoot
- the damping ratio
- etc.

For time definitions we ask the reader to look to the Fig. 4.21. To have an idea on the concept of the settling time, the overshoot, etc., let us consider a linear time invariant system with an input $r(t)$ and an output $y(t)$. If we apply a step function at the input, the output of this system will be as shown in Fig. 4.21. From this figure, it can be seen that the settling time is defined as the time for the system response, $y(t)$ to reach the error band (that is defined with a certain percentage, 2 %, 5 %, etc.) and stay for the rest of the time. The lower the percentage is, the longer the settling time will be.

The overshoot is another characteristic of the time response of a given system. If we refer to the previous figure, the overshoot is defined as the maximum exceed of the steady state value of the system output. More often, we use the percentage overshoot, which is defined as the maximum value of the output minus the step value divided by the step value.

The error is also another characteristic of the output behavior. It is defined as the difference between the steady value taken by the output and the desired value. For a closed-loop system with a unity feedback, the error, $E(z)$, is defined mathematically as:

$$E(z) = R(z) - Y(z)$$

where $R(z)$ is the reference input and $Y(z)$ is the output.

Previously we developed tools that can be used to compute the expression in time of a given signal. Here we will use this to compute the time response of a given system to a chosen input that may be one or a combination of the following signals:

- Dirac impulse
- step
- ramp

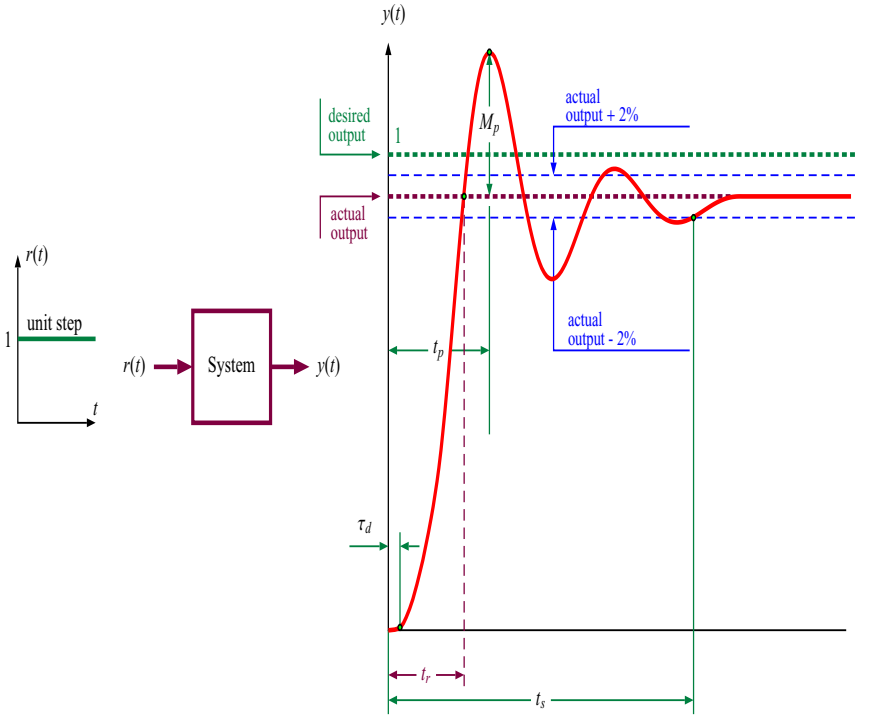


Fig. 4.21 Behavior of the time response for a step input

To compute the time response let us consider a system which has a pulse transfer function $G(z)$ with a given input signal, $U(z)$, and consider the computation of the expression of $y(kT)$. The system is represented in Fig. 4.22. This figure may represent either an open loop pulse transfer function or its equivalent closed-loop pulse transfer function that we get after simplifying the system block diagram.

From this figure, we get:

$$Y(z) = G(z)U(z)$$

The computation of time response, $y(kT)$, is brought to the computation of the inverse $\mathscr{Z}$-transform that be can be determined using one of the following methods:

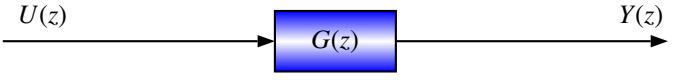


Fig. 4.22 Block diagram (BD)

- expansion into partial fraction
- polynomial division
- residues method

To illustrate how the time response, let us consider the following examples.

Example 4.4.1 *In this example we consider the speed control of a dc motor driving via a gear a given mechanical load. We assume that the system is controlled using a microcontroller. The transfer function of the system is given by:*

$$G(s) = \frac{K}{\tau s + 1}$$

with $K = 2$ *and* $\tau = 2$.

The system is considered in open-loop. In this case since we have the presence of a ZOH, we obtain:

$$G(z) = \left(1 - z^{-1}\right) \mathscr{Z}\left[\frac{2}{s\,(2s+1)}\right]$$

Using the $\mathscr{Z}$*-transform table, we get:*

$$\begin{aligned} G(z) &= \left(1 - z^{-1}\right)\left[\frac{z\left(1 - e^{-\frac{T}{2}}\right)}{(z-1)\left(z - e^{-\frac{T}{2}}\right)}\right] \\ &= \frac{\left(1 - e^{-\frac{T}{2}}\right)}{\left(z - e^{-\frac{T}{2}}\right)} \end{aligned}$$

where T *is the sampling period.*

For our system, since the time constant is equal to $2sec$*, a proper choice for the sampling period is* $T = 0.2sec$*. Using this, we get:*

$$G(z) = \frac{0.0952}{z - 0.9048}$$

If now we consider that the signal input is unit step, we get

$$Y(z) = \frac{0.0952z}{(z-1)\,(z - 0.9048)}$$

To compute the time response either we can use the table or proceed with the expansion into partial fraction.

Using the $\mathscr{Z}$-transform table, we have:

$$y(kT) = 1 - e^{-0.1k}$$

With the expansion into partial fraction we obtain:

$$\begin{aligned}\frac{Y(z)}{z} &= \frac{0.0952}{(z-1)(z-0.9048)} \\ &= \frac{K_1}{z-1} + \frac{K_2}{z-0.9048} \\ &= \frac{1}{z-1} + \frac{-1}{z-0.9048}\end{aligned}$$

From this we get:

$$Y(z) = \frac{z}{z-1} + \frac{-z}{z-0.9048}$$

Using now the $\mathscr{Z}$-transform table, we get:

$$y(kT) = 1 - e^{-0.1k}$$

since $e^{-0.1} = 0.9048$.

Example 4.4.2 *In this example we consider the position control of a dc motor driving via a gear a given mechanical load. We assume that the system is controlled using a microcontroller. The transfer function of the system is given by:*

$$G(s) = \frac{K}{s(\tau s + 1)}$$

with $K = 2$ and $\tau = 2$.

The system is considered in open-loop. In this case since we have the presence of a ZOH, we obtain:

$$G(z) = \left(1 - z^{-1}\right) \mathscr{Z}\left[\frac{2}{s^2(2s+1)}\right]$$

Using the $\mathscr{Z}$-transform table with $T = 0.2 second$, we get:

$$\begin{aligned}G(z) &= \left(1 - z^{-1}\right)\left[\frac{Tz}{(z-1)^2} - \frac{z\left(1 - e^{-\frac{T}{2}}\right)}{0.5(z-1)\left(z - e^{-\frac{T}{2}}\right)}\right] \\ &= \frac{(0.4048z - 0.5476)}{0.5(z-1)(z-0.9048)}\end{aligned}$$

If now we consider that the signal input is unit step, we get

$$Y(z) = 0.8096\frac{z(z-1.3528)}{(z-1)^2(z-0.9048)}$$

To compute the time response either we can use the $\mathscr{Z}$-transform table or proceed with the method of expansion into partial fraction or with the method of residues.

Using the $\mathscr{Z}$-transform table, we get:

$$y(kT) = kT - \frac{1}{a}\left[1 - e^{akT}\right]$$

with $a = 0.5$ and $T = 0.2$

With the method expansion into partial fraction we have:

$$Y(z) = \frac{K_1}{(z-1)^2} + \frac{K_2}{(z-1)} + \frac{K_3}{(z-0.9048)}$$

With the method of residues, we obtain:

$$y(kT) \ = \sum \textit{residues of } 0.8096 \frac{z\,(z-1.3528)\,z^{k-1}}{(z-1)^2\,(z-0.9048)}$$

at the poles $z = 1$ and $z = 0.9048$.

These residues are computed as follows:

- *residue at pole $z = 1$*

$$\begin{aligned}
&\frac{1}{(2-1)!}\frac{d}{dz}\left[(z-1)^2 0.8096 \frac{z\,(z-1.3528)\,z^{k-1}}{(z-1)^2\,(z-0.9048)}\right]_{|_{z=1}} \\
&= \frac{d}{dz}\left[0.8096 \frac{(z-1.3528)\,z^k}{(z-0.9048)}\right]_{|_{z=1}} \\
&= 119.0464 - 2.9568k
\end{aligned}$$

- *residue at pole $z = 0.9048$*

$$\begin{aligned}
&\left[0.8096 \frac{(z-1.3528)\,z^k}{(z-1)^2}\right]_{|_{z_r=0.9048}} \\
&= -40.0198\,(0.9048)^k
\end{aligned}$$

Using now the table we get:

$$y(kT) = 1 - e^{-0.1k}$$

since $e^{-0.1} = 0.9048$.

From the time response we computed in the previous section, it can be seen that for a given system the output can take either finite or infinite value for a given signal signal input. The question is why this happen. The answer of this question is given by the stability analysis and this will be covered in the next section.

4.5 Stability and Steady-State Error

For systems in the continuous-time domain, the stability implies that all the poles must have negative real parts. With the transform $z = e^{Ts}$, with T is the sampling period, we saw that the left half plane of the s-domain corresponds to the inside unit

circle and therefore, in the z-domain, the system will be stable if all the poles are inside this unit circle.

To analyze the stability of discrete-time systems, let us consider the system of the Fig. 4.23. The closed loop transfer function of this system is given by:

$$F(z) = \frac{Y(z)}{R(z)} = \frac{C(z)G(z)}{1 + C(z)G(z)}$$

where $R(z)$ and $Y(z)$ are respectively the input and the output.

The poles of the system are the solution of the following characteristic equation:

$$1 + C(z)G(z) = 0$$

The study of stability requires the computation of these roots. For small order system we can always solve the characteristic equation by hand and then obtain the poles and the conclusion on stability will be done based on the fact where the poles are located. For high order this approach is not recommended and an alternate is needed. Some criterions have been developed to study the stability. Among these criterions we quote the one of Jury and the one of Raible.

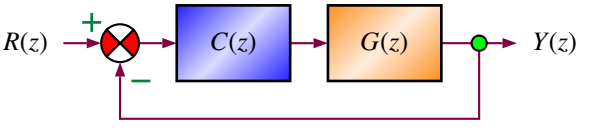


Fig. 4.23 Block diagram of the closed-loop

Let $z = e^{sT}$ with $s = \sigma \pm j\omega$. Therefore,

if $\sigma < 0$ then $|z| < 1$ and the system is stable
if $\sigma > 0$ then $|z| > 1$ and the system is unstable
if $\sigma = 0$ then $|z| = 1$ and the system is at the limit of stability

Example 4.5.1 *Let us consider a dynamical system with the following characteristic equation:*

$$1 - \frac{3}{4}z^{-1} + \frac{1}{8}z^{-2} = 0$$

The roots of the characteristic equation are: $z = \frac{1}{2}$ *and* $z = \frac{1}{4}$. *These roots are located inside the unit circle and therefore the system is stable.*

Example 4.5.2 *Let us consider a dynamical system with the following characteristic equation:*

$$1 - 2z^{-1} + \frac{5}{4}z^{-2} = 0$$

or equivalently:

$$z^2 - 2z + \frac{5}{4} = 0$$

The roots of the system are $z_{1,2} = 1 \pm j\frac{1}{2}$ *and are both outside the unit circle which implies that the system is unstable.*

A direct approach to study the stability of discrete-time system is to convert it to an equivalent continuous-time one, and then use the Routh-Hurwitz's Criterion. The idea is to find an adequate application that maps the inside of the unit circle onto the left-hand half plane. Then, we can apply the Routh-Hurwitz criterion. The transformation we're looking for is:

$$z = \frac{1+w}{1-w} \text{ with } w \neq 1$$

Replacing z by this expression in the characteristic equation will give a new one in w and we can apply the Routh-Hurwitz's Criterion.

Example 4.5.3 *To show how we use the Routh-Hurwitz's Criterion, let us consider the dynamical system with the following characteristic equation:*

$$z^3 - 2.4z^2 + z - 0.4 = 0$$

It can be shown that the poles are 2 *and* $0.2 \pm 0.4j$*. Therefore the system is unstable.*

Let us now replace z *by* $\frac{1+w}{1-w}$ *in the characteristic equation. This gives:*

$$\left[\frac{1+w}{1-w}\right]^3 - 2.4\left[\frac{1+w}{1-w}\right]^2 + \left[\frac{1+w}{1-w}\right] - 0.4 = 0$$

which can be put in the following form:

$$4.8w^3 + 3.2w^2 + 0.8w - 0.8 = 0$$

The Routh-Hurwitz's Criterion consists then of filling the following table:

w^3	4.8	0.8	0
w^2	3.2	−0.8	0
w^1	2	0	
w^0	−0.8		

Based on the first column, we can see that there one change in the sign and therefore the system is unstable. This confirm the results we has already remarked earlier.

It is also important to notice that the roots of the characteristic equation in w are given by:

$$\begin{aligned} w_1 &= 0.3333 \\ w_{2,3} &= -0.5000 \pm 0.5000j \end{aligned}$$

These roots can also be obtained from the ones in z-domain using $w = \frac{z-1}{z+1}$.

Example 4.5.4 *Consider the characteristic equation:*

$$z^2 + z(6.32K - 1.368) + 0.368 = 0$$

Applying the bilinear transform yields:

$$\left(\frac{1+w}{1-w}\right)^2 + \left(\frac{1+w}{1-w}\right)(6.32K - 1.368) + 0.368 \ = 0$$

that gives in turn:

$$w^2[2.736 - 6.32K] + 1.264w + (6.32K - 1) \ = 0$$

Applying Routh-Hurwitz gives:

w^2	$2.736 - 6.32K$	$6.32K - 1$
w^1	1.264	0
w^0	$6.32K - 1$	

To guarantee the stability we should determine the range of the parameter K such that we don't have sign change in the first column. For the row w^0*, we should have* $6.32K - 1 > 0$*, i.e.* $K > \frac{1}{6.32} = 0.158$*. For the row* w^2*, we should also have* $2.736 - 6.32K > 0$*, i.e.* $K < \frac{2.736}{6.32} = 0.4329$*. If we look to these two conditions, we conclude that the system is stable for* $0.158 < K < 0.4349$*.*

To check this, let us consider $K = 0.2$*, which is inside the interval. Using this value, we obtain the following characteristic equation:*

$$z^2 - 0.104z + 0.368 = 0$$

that has as roots $z_1 = 0.052 + j0.6044$ *and* $z_2 = 0.052 - j0.6044$*. The roots are located inside the unit circle and therefore, the system is then stable. For* $K = 1$*, we obtain:*

$$z^2 + 4.952z + 0.368 = 0$$

The roots are $z_1 = -0.076$ *and* $z_2 = -4.876$*. The system is then unstable because* $|z_2| > 1$*.*

For discrete-time Jury has developed a criterion that gives an idea on stability of any system without solving the characteristic equation. To show how this approach works, let us consider the following characteristic polynomial with real coefficients:

$$P(z) \ = a_n z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0 = 0$$

where $a_n > 0$ and a_i is a real scalar.

Jury's stability criterion consists of building the following array of coefficients:

row 1	a_0	a_1	a_2	$\cdots$	a_{n-k}	$\cdots$	a_{n-1}	a_n
row 2	a_n	a_{n-1}	a_{n-2}	$\cdots$	a_k	$\cdots$	a_1	a_0
row 3	b_0	b_1	b_2	$\cdots$	b_{n-k}	$\cdots$	b_{n-1}	
row 4	b_{n-1}	b_{n-2}	b_{n-3}	$\cdots$	b_k	$\cdots$	b_0	
row 5	c_0	c_1	c_2	$\cdots$		c_{n-2}		
row 6	c_{n-2}	c_{n-3}	c_{n-4}	$\cdots$		c_0		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\cdots$				
row 2n-5	p_0	p_1	p_2	p_3				
row 2n-4	p_3	p_2	p_1	p_0				
row 2n-3	q_0	q_1	q_2					

The Jury's array coefficients are computed as follows:

$$b_k = \begin{vmatrix} a_0 & a_{n-k} \\ a_n & a_k \end{vmatrix}, \quad c_k = \begin{vmatrix} b_0 & b_{n-1-k} \\ b_{n-1} & b_k \end{vmatrix},$$

$$d_k = \begin{vmatrix} c_0 & c_{n-2-k} \\ c_{n-2} & c_k \end{vmatrix}, \qquad \cdots$$

$$q_0 = \begin{vmatrix} p_0 & p_3 \\ p_3 & p_0 \end{vmatrix}, \qquad q_2 = \begin{vmatrix} p_0 & p_1 \\ p_3 & p_2 \end{vmatrix}$$

The necessary and sufficient conditions that the system described by P(z) is stable are:

$$P(1) > 0$$

$$P(-1) \quad \begin{cases} > 0 & \text{if n is even} \\ < 0 & \text{if n is odd} \end{cases}$$

$$\text{with (n - 1) constraints} \quad \begin{bmatrix} |a_0| < a_n & |b_0| > |b_{n-1}| \\ |c_0| > |c_{n-2}| & |d_0| > |d_{n-3}| \\ \cdots & \cdots \\ |q_0| > |q_2| \end{bmatrix}$$

Example 4.5.5 *Examine the stability of the system described by the following polynomial:*

$$P(z) = z^3 + 3.3z^2 + 3z + 0.8 = 0$$

We form the Jury's array of coefficients:

*row*1.	0.8	3	3.3	1
*row*2.	1	3.3	3	0.8
*row*3.	−0.36	−0.9	−0.36	0

$$b_0 = \begin{vmatrix} a_0 & a_3 \\ a_3 & a_0 \end{vmatrix}, b_1 = \begin{vmatrix} a_0 & a_2 \\ a_3 & a_1 \end{vmatrix}, b_2 = \begin{vmatrix} a_0 & a_1 \\ a_3 & a_2 \end{vmatrix}, b_3 = \begin{vmatrix} a_0 & a_0 \\ a_3 & a_3 \end{vmatrix}$$

Since $n = 3$, then the following conditions should apply:

- $P(1)$ *must be positive:* $1 + 3.3 + 3 + 0.8 = 8.1 > 0$ *which is true*
- $P(-1)$ *must be negative because* $n = 3 =$ *is odd:* $-1 + 3.3 - 3 + 0.8 = 0.1 > 0$ *and this is false*
- $|a_0| < a_n$, *i.e:* $|0.8| < 1$ *which is true*
- $|b_0| < |b_{n-1}|$, *i.e:* $|-0.36| = |-0.36|$ *which is false*

One false condition is enough to conclude that the system is unstable.

Example 4.5.6 *Let us consider a dynamical system with the following characteristic equation:*

$$1 + K\frac{z}{(z-1)(z-0.4)} = 0$$

where K is a parameter to determine such that the system is stable.

This characteristic equation can be rewritten as follows:

$$z^2 + (K - 1.4)\,z + 0.4 = 0$$

Applying Jury criterion gives:

- $P(1) > 0$, *which gives* $K > 0$
- $P(-1) > 0$ *which gives* $K < 2.8$
- $|a_0| < a_n$, *i.e:* $0.4 < 1$ *which is true*

Therefore, our system will be stable if $K \in]0, 2.8[$. *For instance, if we fix K to* 2, *which gives the following characteristic equation:*

$$z^2 + 0.6z + 0.4 = 0$$

the roots are $z_{1,2} = -0.3000 \pm 0.5568\,j$ *which are inside the unit circle since* $|z_{1,2}| < 1$.

Another criterion to study the stability bas been developed by Raible. This stability Criterion consists also as for the Jury criterion to fill an array and then conclude on stability. To show how this criterion works, let us consider the following characteristic equation:

$$P(z) \;= a_0 z^n + a_1 z^{n-1} + \cdots + a_{n-1} z + a_n$$

where a_i is a real scalar.

row 1	a_0	a_1	$\cdots$	a_{n-1}	a_n	multiplier
row 2	a_n	a_{n-1}	$\cdots$	a_1	a_0	$\alpha_n = \frac{a_n}{a_0}$
row 3	$a_0^{(n-1)}$	$a_1^{(n-1)}$	$\cdots$	$a_{n-1}^{(n-1)}$	0	multiplier
row 4	$a_{n-1}^{(n-1)}$	$a_{n-2}^{(n-1)}$	$\cdots$	$a_0^{(n-1)}$	0	$\alpha_{n-1} = \frac{a_{n-1}^{(n-1)}}{a_0^{(n-1)}}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$			
row 2n-1	$a_0^{(1)}$	$a_1^{(1)}$				multiplier
row 2n	$a_1^{(1)}$	$a_0^{(1)}$				$\alpha_1 = \frac{a_1^{(1)}}{a_0^{(1)}}$
row 2n+1	$a_0^{(0)}$					

- The 1^{st} row is formed by the polynomial coefficients
- The 2^{nd} row is formed by the same coefficients but in the opposite order
- The 3^{rd} row is obtained by multiplying the 2^{nd} row by $\alpha_n = \frac{a_n}{a_0}$, then by subtracting the result of the 1^{st} row
- The 4^{th} row is formed by coefficients of the 3^{rd} row placed in the opposite order.

These procedures are repeated until the array gets $2n + 1$ rows. The last row contains only one number.

Raible's Stability Criterion

When $a_0 > 0$, the roots of the polynomial are all inside the unit circle if and only if $a_0^{(i)} > 0, i = 0, 1, \cdots, n-1$

The coefficients $a_0^{(i)} > 0, i = 0, 1, \cdots, n-1$ appear in the Raible's array .

Remark 4.5.1 *The assumption $a_0 > 0$ is not restrictive. In fact, when $a_0 < 0$, it is enough to change the signs of all coefficients of the polynomial $P(z)$ to obtain $-P(z)$, which in turn is used for Raible's criterion.*

This procedure is correct since the roots of $P(z)$ and of $-P(z)$ are identical.

Example 4.5.7 *To show how the Raible's criterion works, let us consider the following characteristic equation:*

$$P(z) = -z^3 - 0.7z^2 - 0.5z + 0.3$$

The coefficient a_0 must be positive, then we form the coefficient array of the polynomial $-P(z) = z^3 + 0.7z^2 + 0.5z - 0.3$

1	0.7	0.5	−0.3	
−0.3	0.5	0.7	1	$\alpha_3 = \frac{0.3}{-1} = -0.3$
0.91	0.85	0.71		
0.71	0.85	0.91		$\alpha_2 = \frac{0.71}{0.91} = 0.78$
0.36	0.19			
0.19	0.36			$\alpha_1 = \frac{0.19}{0.36} = 0.53$
0.26				

The system is stable because $a_0^{(i)} > 0$, $i = 0, 1, \cdots, n-1$

We have presented some techniques to study the stability of discrete-time systems. It is also important to notice that we can also apply the criterions in the frequency domain.

4.6 Root Locus Technique

The root locus technique is a powerful approach that is usually used for continuous-time or discrete-time systems either for analysis or design. The technique gives an idea on how the poles of the closed-loop dynamics behave when a gain or more (a parameter or more) are changed. The direct conclusion is that we know immediately how the stability and the other performances of the system are affected by the parameters changes.

Nowadays there exist many tools to plot the root loci of any dynamical system some of them are available free for use. In the rest of this section, we will use Matlab for our plotting but we will develop rules of how obtain a sketch of the root locus in case we don't have a computer at hand.

As for the continuous case, the root locus for the discrete system is described by the characteristic equation that we write in the following form:

$$1 + KG(z) \ = 0$$

where K is the parameter that varies and

$$G(z) = \frac{(z - n_1)(z - n_2)\cdots(z - n_m)}{(z - z_1)(z - z_1)\cdots(z - z_n)}$$

with $z_1, z_2, \cdots, z_n$ are the poles and $n_1, n_2, \cdots, n_m$ are the zeros of the open loop transfer function.

When the parameter K varies from 0 to infinity (∞). The same rules as we use for the plotting of the root locus of the continuous-time systems in the s-plane apply to the plotting of the one of discrete-time systems in the z-plane, except that the interpretation of the results is different mainly in regard of stability.

From the characteristic equation, we get the following conditions:

$$\frac{1}{K} = \frac{\Pi_{i=1}^{m}|z - n_i|}{\Pi_{i=1}^{n}|z - z_i|} \tag{4.4}$$

$$\sum_{i=1}^{m} \arg(z - n_i) - \sum_{i=1}^{n} \arg(z - z_i) = (2k + 1)\pi, k = 0, 1, 2, \cdots, \tag{4.5}$$

The first condition is referred to as the magnitude condition while the second is referred to as angle condition. Any point in the z-plane that satisfies these two conditions belongs to the root locus of the system. To this point corresponds a gain K_{z_0}. If this point is z_0, then we have:

$$\frac{1}{K_{z_0}} = \frac{\Pi_{i=1}^{m}|z_0 - n_i|}{\Pi_{i=1}^{n}|z_0 - z_i|}$$

$$\sum_{i=1}^{m} \arg(z_0 - n_i) - \sum_{i=1}^{n} \arg(z_0 - z_i) = \theta_0$$

where θ_0 is the corresponding angle of this point.

A point of the z-plane will belong to the root locus, if it satisfies these two conditions. In general plotting the exact root locus for a given system is a hard task unless we have the appropriate tools for that. More often a sketch of this root locus can be easily obtained using some simple rules. Some of these rules are:

1. the number of branches is equal to the order of the system, i.e.: n;

2. the root locus is symmetric with respect to the real axis. This is due to the fact that the roots of the characteristic equation are either real or complex. And if there is a complex root, we have automatically its conjugate.

3. The loci originate from the poles of the open loop transfer function and terminate on the zeros of the this transfer function. To explain why the loci originate from the poles, we can make K equal to zero, while why the loci terminate on the zeros can be explained by letting K goes to ∞ in Eq. (4.4).

4. the number of asymptotes is equal to the difference between the number of poles, n, and the number of zeros, m, of the open loop transfer function. These asymptotes are characterized by:

$$\begin{aligned} \delta &= \frac{\sum \text{poles} - \sum \text{zeros}}{n - m} \\ \beta_k &= (2k + 1)\frac{\pi}{n - m}, k = 0, 1, 2, \cdots, \end{aligned}$$

The parameter, δ, gives the intersections of the asymptotes with the real axis, while β_k gives the angle that make each asymptote with the real axis.

5. for the breakpoints of the root locus, firstly we determine the expression of the varying parameters K, i.e.:

$$K = \frac{\Pi_{i=1}^{n}|z - z_i|}{\Pi_{i=1}^{m}|z - n_i|}$$

The breakpoints are solution of the following equation:

$$\frac{dK}{dz} = 0$$

It is important to select from the roots of this equation those are feasible solution for the breakpoints.

6. the intersection of the imaginary axis in the z-plane can be determined by replacing z by $j\nu$ in the characteristic equation and write it as follows:

$$\Re(K, \nu) + j\Im(K, \nu) = 0$$

that gives in turn two equations:

$$\begin{aligned} \Re(K, \nu) &= 0 \\ \Im(K, \nu) &= 0 \end{aligned}$$

The solution gives the frequency at which the intersection occurs and the corresponding gain.

7. the angle of departure from a complex pole or the angle of arrival to a complex zero is computed using the angle condition. If the point at which we want to calculate the angle is z_0, the condition angle becomes:

$$\sum_{i=1}^{m} \arg(z_0 - n_i) - \sum_{i=1}^{n} \arg(z_0 - z_i) - \theta_0 = 180$$

where θ_0 is the corresponding angle of this point.

Example 4.6.1 *To show how the technique of root locus works, let us consider the system of the Fig. 4.24 where the plant is the double integrator and the controller is a proportional action with a gain K, that we will assume to change between zero and infinity for some physical reasons like heating, aging, etc.*

Using the $\mathcal{Z}$-transform table and the expression of the closed-loop transfer function we get the following characteristic equation of this system:

$$1 + K\frac{(z+1)}{(z-1)^2} = 0, \textit{with } K = \frac{k}{2}$$

- *Number of branches:* $n = 2$
- *Finite number of branches:* $m = 1$
- *Infinite number of branches:* $n - m = 2 - 1 = 1$
- *Angle of asymptotes:* $\beta = \frac{\pi(2k+1)}{n-m} = \frac{\pi(2k+1)}{2-1} = \pi, k = 0$

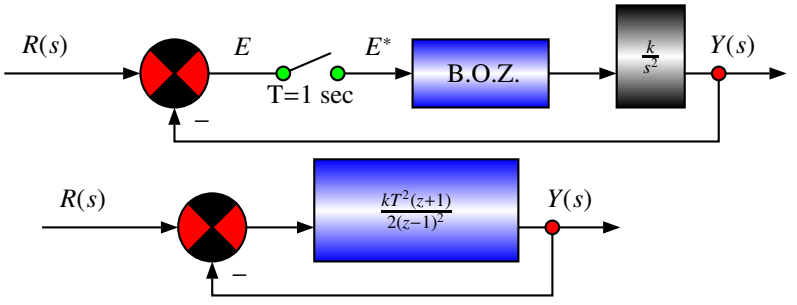


Fig. 4.24 BD of the system with characteristic eqn: $1 + K\frac{(z+1)}{(z-1)^2} = 0$

- *Intersection of the asymptote with the real axis:* $\delta = \frac{(1)+(1)-(-1)}{2-1} = 3$
- *Intersection of the locus with the real axis:* $\frac{dK}{dz} = 2z^2 + 4z - 6 = 0$, *which gives* $z_1 = -1$ *et* $z_2 = -3$.

The root locus is illustrated in Fig. 4.25. All the roots are outside the unit circle in blue. The system is unstable. THis means that a proportional controller is not able to stabilize a double integrator.

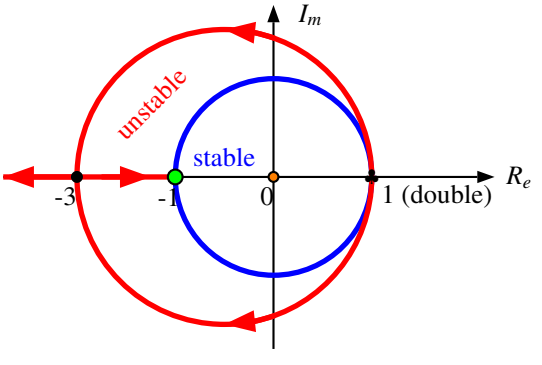


Fig. 4.25 RL of the system with characteristic eqn: $1 + K\frac{(z+1)}{(z-1)^2} = 0$

Example 4.6.2 *As a second example for the root locus technique let us consider the system of the Fig. 4.26.*

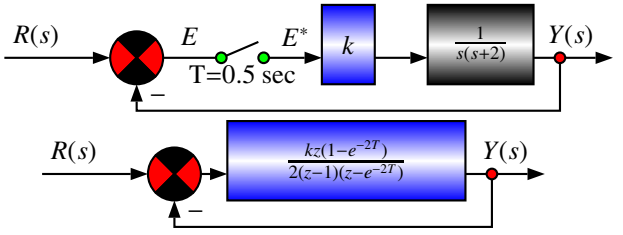


Fig. 4.26 BD of the system with characteristic eqn: $1 + K\frac{z}{(z-1)(z-0.368)} = 0$

The characteristic equation of this system is given by:

$$1 + k\frac{z(1-e^{-2T})}{2(z-1)(z-e^{-2T})} = 1 + K\frac{z}{(z-1)(z-0.368)} = 0$$
$$\text{with } K = 0.316k$$

- *Number of branches:* $n = 2$.
- *Finite Number of branches:* $m = 1$.
- *Infinite Number of branches* $n - m = 2 - 1 = 1$.
- *Angle of asymptotes:* $\beta = \frac{\pi(2k+1)}{n-m} = \frac{\pi(2k+1)}{2-1} = \pi$.
- *Intersection of the locus with the real axis:* $\frac{dK}{dz} = -z^2 + 0.368 = 0$. *The resolution of this equations gives:* $z_1 = -0.606$ *et* $z_2 = +0.606$.

If we replace z *by* -1 *in the characteristic equation, we find:*

$$1 + K\frac{z}{(z-1)(z-0.368)} = 1 + K\frac{(-1)}{(-1-1)(-1-0.368)} = 0$$

which implies in turn:

$$K = 2.738$$

$$K = 0.316k$$

which gives:

$$k = \frac{K}{0.316} = \frac{2.738}{0.316} = 8.65$$

The root locus is drawn in Fig. 4.27. All the the roots are inside the unit circle in blue. Therefore, the system is stable for all gains $k < 8.65$.

4.7 Bode Plot Technique

The frequency response plays an important role in the analysis and design of continuous-time and discrete-time systems. As for the time response, the frequency

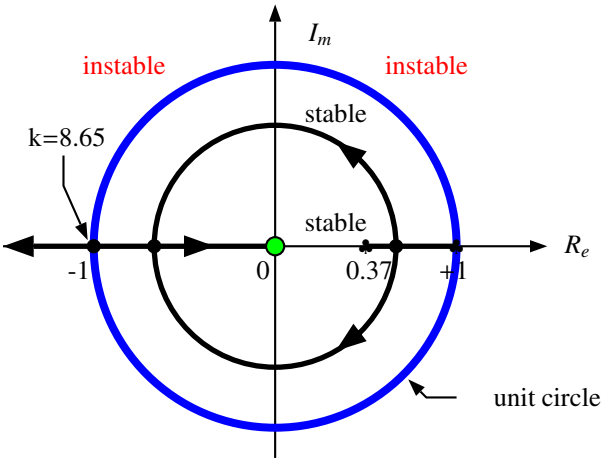


Fig. 4.27 RL of the system with characteristic eqn: $1 + K\frac{z}{(z-1)(z-0.368)} = 0$

response consists of exciting the system by a sinusoidal input. In the continuous-time system, it was proven that for a sinusoidal input, the output of the a stable linear system is sinusoidal with same frequency of the input, and the magnitude and the phase of the output are function of this frequency. For discrete-time system, the output is also sinusoidal with the same frequency as the input signal and the phase and the magnitude are still function of this frequency. To show this, let us consider a stable linear system with the following transfer function:

$$\begin{aligned} G(z) &= \frac{Y(z)}{R(z)} = \frac{b_m z^m + b_{m-1} z^{m-1} + \cdots + b_1 z + b_0}{z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0} \\ &= K \frac{\Pi_{i=1}^{m}(z - n_i)}{\Pi_{i=1}^{n}(z - z_i)} \end{aligned}$$

Let the input $r(t)$ has the following expression:

$$r(t) = \sin(wt)$$

where w is the frequency of the input. The magnitude is taken here equal to one.

The $\mathscr{Z}$-transform of this signal is given by (see $\mathscr{Z}$-transform table):

$$R(z) = \frac{z \sin(wt)}{z^2 - 2z cost(wt) + 1} = \frac{z sin(wT)}{(z - e^{-jwT})(z - e^{jwT})}$$

Now if we consider that the system is excited by $R(z)$ the corresponding output, $Y(z)$ is given by:

$$\begin{aligned} Y(z) &= G(z)R(z) \\ &= K \frac{\Pi_{i=1}^{m}(z - n_i)}{\Pi_{i=1}^{n}(z - z_i)} \frac{z sin(wT)}{(z - e^{-jwT})(z - e^{jwT})} \end{aligned}$$

To get the expression of the output, let us proceed with a partial fraction of $Y(z)$. This gives:

$$Y(z) = \frac{cz}{z - e^{-jwT}} + \frac{\bar{c}z}{z - e^{jwT}} + \text{terms due to } G(z)$$

Let s now multiply both sides this equality by $\frac{(z-e^{-jwT})}{z}$ to get the following:

$$G(z)\frac{sinwT}{(z - e^{jwT})} = c + \frac{\bar{c}\left(z - e^{-jwT}\right)}{z - e^{jwT}} + \left[\frac{\left(z - e^{-jwT}\right)}{z}\right] \text{terms due to } G(z)$$

where

$$\begin{aligned} c &= \left[G(z)\frac{sin(wT)}{(z - e^{jwT})}\right]_{|_{z=e^{-jwT}}} \\ \bar{c} &= \text{conjugate of } c \end{aligned}$$

Notice that $e^{-jwT} = cos - wT + jsin - wT = coswT - jsinwT$, which implies that

$$\left(z - e^{jwT}\right)_{|_{z=e^{-jwT}}} = -2jsinwT$$

Using this we get:

$$\begin{aligned} c &= \frac{G(e^{-jwT})}{-2j} \\ \bar{c} &= \frac{G(e^{jwT})}{2j} \end{aligned}$$

Using now the fact that for any complex number we have:

$$G(e^{jwT}) = M(w)e^{j\theta(w)}$$

where M and θ represent respectively the magnitude and the phase at the frequency w.

The steady state, the terms due to $G(z)$ vanish and we have:

$$\begin{aligned} Y(z) &= \frac{G(e^{-jwT})}{-2j}\frac{z}{z - e^{-jwT}} + \frac{G(e^{jwT})}{2j}\frac{z}{z - e^{jwT}} \\ &= \frac{M(w)}{2j}\left[-\frac{e^{-\theta(w)}z}{z - e^{-jwT}} + \frac{e^{\theta(w)}z}{z - e^{jwT}}\right] \end{aligned}$$

The $\mathscr{Z}$-transform inverse of $Y(z)$ at the steady state is given by:

$$\begin{aligned} y(kT) &= \frac{M(w)}{2j}\left[e^{j\theta(w)}e^{jwT} - e^{-j\theta(w)}e^{-jwT}\right] \\ &= \frac{M(w)}{2j}\left[e^{j(\theta(w)+wT)} - e^{-j(\theta(w)+wT)}\right] \\ &= M(w)sin\left(wT + \theta(w)\right) \end{aligned}$$

Remark 4.7.1 *It is important to mention that the magnitude and the phase of the output for a sinusoid input are both functions of its frequency. Therefore, their values will change when the frequency changes.*

A certain parallel can be made with frequency response of continuous time. In fact, for these system, the frequency response can be obtained from the transfer function, $G(s)$ that describes the system by doing the following:

- *the magnitude $M(w)$ is given by:*

$$M(w) = |G(jw)|$$

- *the phase $\theta(w)$ is given by:*

$$\theta(w) = \arg\left(G(jw)\right)$$

This means that the magnitude and the phase of the output at frequency w are obtained from the transfer function of the system by replacing firstly s by jw and then compute the magnitude and the phase using the previous formulas.

For the discrete time, the same reasoning applies except that we have to replace z by e^{jwT} and use the following formulas:

- *the magnitude $M(w)$ is given by:*

$$M(w) = \left|G(e^{jwT})\right|$$

- *the phase $\theta(w)$ is given by:*

$$\theta(w) = \arg\left(G(e^{jwT})\right)$$

Some precautions have to be taken for the frequency response of discrete time system. In fact, the $\mathscr{Z}$-transform is obtained by replacing z by e^{sT}. Therefore, the primary and the complementary strips of the left hand side of the s-domain are mapped to the interior of the unit circle in the z-domain. If we replace in turn z by e^{jwT} to get the frequency response of the discrete time system, the result we will get has no sense since it deals with the entire z-plane. To avoid this the following transformation is usually used:

$$z = \frac{1 + \frac{T}{2}\omega}{1 - \frac{T}{2}\omega}$$

which implies:

$$\omega = \frac{2}{T}\frac{z-1}{z+1}$$

Using the $\mathscr{Z}$-transform and and the w-transform respectively, the primary trip of the left half of the s-plane is then transformed into the unit circle which in turn transformed to the entire left half of the w-plane. More specifically, the range of frequencies in the s-plane $-\frac{\omega_s}{2} \leq w \leq \frac{\omega_2}{2}$ is firstly transformed into the unit circle in the z-plane, which in turn transformed into the entire left half of the w-plane.

Finally, it is important to notice the relationship between the frequencies ω and ν. In fact, ω is defined by:

$$\begin{aligned}\omega_{|_{\omega=i\nu}} &= j\nu = \left[\frac{2}{T}\frac{z-1}{z+1}\right]_{|_{z=e^{jwT}}} \\ &= \frac{2}{T}\frac{e^{jwT}-1}{e^{jwT}+1}\end{aligned}$$

Multiplying the numerator and the denominator by e^{-jwT}, we get:

$$\begin{aligned}w_{|_{w=i\nu}} &= j\nu \\ &= j\frac{2}{T}\tan\left(\frac{wT}{2}\right)\end{aligned}$$

which gives the following relationship between w and ν:

$$w = \frac{2}{T}\tan\left(\frac{wT}{2}\right)$$

At low frequencies, we have equality between these frequencies. In fact, when w is low, we have $\tan\left(\frac{wT}{2}\right) = \frac{wT}{2}$, which gives $w = \nu$.

Based on this remark, the frequency response of the discrete time consists then of replacing w by $j\nu$, with ν is a fictitious frequency, in the new expression of the transfer function obtained after replacing z by $z = \frac{1+\frac{T}{2}w}{1-\frac{T}{2}w}$. To have an idea on how the frequency response can be plotted, let us consider the following example.

Example 4.7.1 *As a first example of the frequency response, let us consider the system of the Fig. 4.28. It represents the speed control of a load driven by a dc motor. The controller is a proportional. The transfer function of the system and the controller is given by:*

$$\bar{G}(s) = \frac{K_p k}{\tau s + 1} = \frac{K}{\tau s + 1}$$

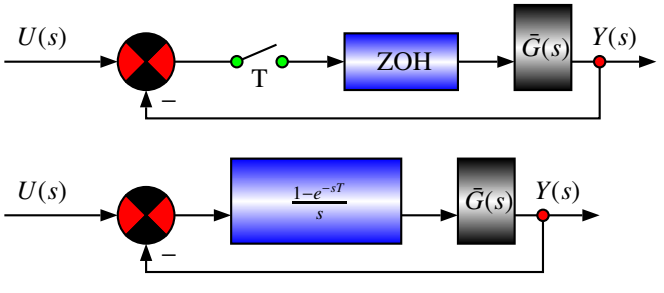


Fig. 4.28 Speed control of mechanical part driven by a dc motor

Firstly, let us compute the open loop transfer function of the system in Fig. 4.26. Since we have a ZOH we get:

$$G(s) = \left(1 - e^{-sT}\right) \frac{K}{s\left(\tau s + 1\right)}$$

where $K = K_p k = 2$, $\tau = 1s$ *and* T *is the sampling period used for our system and it is equal to* $0.1s$.

Using the $\mathscr{Z}$*-transform table we get:*

$$\begin{aligned} G(z) &= K \frac{(z-1)}{z} \frac{z\left(1 - e^{-T}\right)}{(z-1)\left(z - e^{T}\right)} \\ &= K \frac{\left(1 - e^{-T}\right)}{\left(z - e^{T}\right)} \\ &= \frac{0.1903}{z - 0.9048} \end{aligned}$$

Replacing now z *by* $\frac{1+\frac{T}{2}w}{1-\frac{T}{2}w} = \frac{1+0.05w}{1-0.05w}$, *we get:*

$$\begin{aligned} G(z) &= \frac{0.1903}{\frac{1+0.05w}{1-0.05w} - 0.9048} \\ &= \frac{0.1903\left(1 - 0.05w\right)}{0.0952 + 0.0952w} \\ &= \frac{1.9989\left(1 - 0.05w\right)}{1 + w} \end{aligned}$$

Using Matlab, we can get the bode diagram of this transfer function as illustrated by Fig. 4.29.

4.8 Conclusions

This chapter covers the analysis tools based on the transfer function concept. Mainly, we developed the techniques of how to compute the time response and determine the system performances. We also presented the root locus and bode plot techniques.

4.9 Problems

1. Compute the $\mathscr{Z}$-transform of the following signals:
   (a) the unit step
   (b) the unit ramp
   (c) the unit exponential
   (d) $r(t) = t + sinwt$
   (e) $1 - coswt$

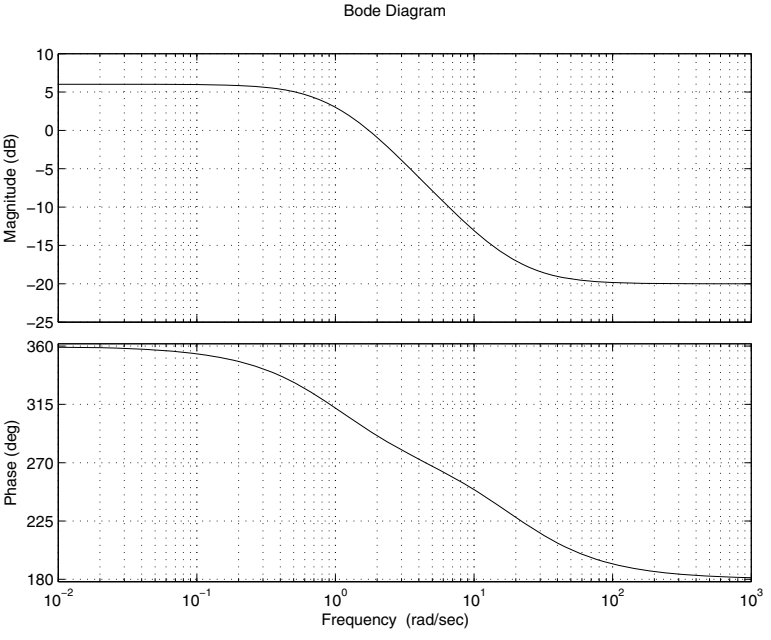


Fig. 4.29 Bode diagram of $\frac{1.9989(1-0.05w)}{1+w}$

2. Compute the expression of the signal in time of the following ones in z:

   (a) $Y(z) = \frac{Tze^{aT}}{(z-e^{-aT})^2}$, $a > 0$

   (b) $Y(z) = \frac{z(1-e^{aT})}{(z-1)(z-e^{-aT})}$, $a > 0$

   (c) $Y(z) = \frac{1}{b-a}\left[\frac{z}{z-e^{aT}} - \frac{z}{z-e^{bT}}\right]$, $a > 0$, $b > 0$ and $a \neq b$

3. For the dynamical systems with the input $u(t)$ and the output $y(t)$ with the following dynamics:

   - $\frac{d^2y(t)}{dt^2} + \frac{dy(t)}{dt} = u(t)$
   - $\frac{d^2y(t)}{dt^2} + 4\frac{dy(t)}{dt} + 4y(t) = 4u(t)$
   - $\frac{d^2y(t)}{dt^2} + 6\frac{dy(t)}{dt} + 8y(t) = 8u(t)$
   - $\frac{d^3y(t)}{dt^3} + 3\frac{d^2y(t)}{dt^2} + 2\frac{dy(t)}{dt} = u(t)$

   (a) determine the sampling period T

   (b) using the approximation methods determine the relationship between the input $U(z)$ and the output $Y(z)$

   (c) determine the pulse transfer function for each dynamics

   (d) using Matlab compute the step response of each dynamics

(e) using now the zero-order-hold, determine the corresponding transfer function and compute the step response. Compare this response to the one of the previous question

4. In this problem we consider the system of the Fig. 4.30 where the transfer function of the system is given by:

$$G(s) = \frac{10}{(s+1)(s+10)}$$

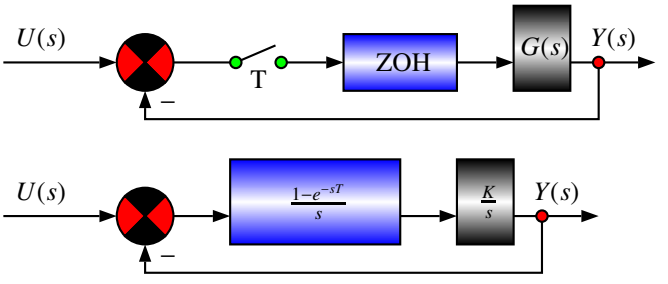


Fig. 4.30 Transfer functions in feedback

(a) determine the sampling period that we can use for this system
(b) using this sampling period determine the open loop transfer function and the closed-loop one
(c) determine the step response of the system
(d) plot the behavior of the output with respect to time

5. Study the stability of the dynamical systems with the following characteristic equation:
   (a) $z^3 + 0.8z^2 + 0.17z + 0.01$
   (b) $z^4 + 1.4z^3 + 0.65z^2 + 0.112z + 0.006$
   (c) $z^5 + 2.39z^4 + 2.036z^3 + 0.7555z^2 + 0.1169z + 0.0059$
   (d) $z^5 + 11.4z^4 + 14.65z^3 + 6.6120z^2 + 1.126z + 0.06$

6. In this problem we consider the dynamical system show in the block diagram illustrated by the Fig. 4.31. The transfer functions are given by:

$$\begin{aligned} G(z) &= \frac{z\left(1 - e^{aT}\right)}{(z-1)\left(z - e^{-aT}\right)} \\ C(z) &= K \end{aligned}$$

with $a = 0.1$ and $T = 0.01$

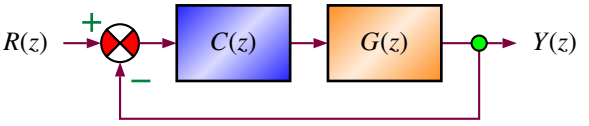


Fig. 4.31 Block diagram of the closed-loop

(a) study the stability in function of the gain K
(b) plot the root locus of the system and conclude on the stability

7. Consider the system of the Fig. 4.30 with the following expression for $G(s)$:

$$G(s) = \frac{K}{s(\tau s + 1)}$$

with K is the gain and $\tau = 1s$ is the time constant of the system.

- determine the sampling period
- compute the transfer function $G(z)$
- plot the root locus the system when the gain K varies between 0 and ∞

8. Consider the system of the Fig. 4.30 with the following expression for $G(s)$:

$$G(s) = \frac{K}{s(\tau s + 1)}$$

with $K = 10$ is the gain and $\tau = 0.1s$ is the time constant of the system.

- determine the sampling period
- compute the transfer function $G(z)$
- plot the Bode diagram of the system

5
Design Based on Transfer Function

After reading this chapter the reader will:

1. master the concept of the design of classical controllers based on the transfer function of the system
2. be able to choose the structure of the classical controller that responds to the desired performances and determine its parameters
3. be familiar with the design of the proportional, proportional and integral, proportional integral and derivative controllers and their approximations
4. be able to determine the recurrent equation for the control law that we must implement in the microcontroller

5.1 Introduction

Tackling a control design problem is always a challenge even for more experienced control engineers. The system for which the controller must be designed, may be an existing one with some poor performances and that we would like to improve, or a new system that we are building. In both cases, the design procedure starts, after

getting the mathematical model for the system, by defining the desired performances that will allow us to determine the structure of the controller and its parameters.

More often the control systems are designed to guarantee certain performances to the closed-loop dynamics of the system under consideration. Such performances can be summarized to the stability and the behaviors of the transient and the steady state regimes. By respecting the limitations of the given system, it is always the case that we search to improve the transient regime by searching for a compromise between the overshoot that the system may have and its rapidity. For the steady state, we search to guarantee that the error is less than a certain chosen tolerance. The controllers we will consider in this chapter to respond to the design requirements are classical ones like the proportional, integral and derivative actions and their approximations.

The rest of the chapter is organized as follows. In Section 2, the control design problem is formulated. Section 3 presents the empirical methods to design classical controllers. In Section 4, the design of classical controllers using the root locus method is developed. Section 5 presents the Bode method. Section 6 presents a case study which consists of designing different controllers for the dc motor kit.

5.2 Formulation of the Control Design Problem

In this chapter we will consider an existing system with poor performances that we would like to improve. Our desire is to act simultaneously on the transient and steady state regimes by introducing a controller in the closed-loop to force the overall system to behave as it is desired.

The performances may be given either in time or frequency domains. In both domains, the stability is the first requirement in the design procedure. Beside the stability, we would like the transient and the steady state regimes to behave in a desirable ways.

In the time domain for the transient regime, we should control the overshoot, the rising time and the settling time for a chosen percentage that will depend on the precision we would like to guarantee to our system. For the steady state regime, we would like to assure that the system's error is less than a certain specified value.

In the frequency domain, the situation is similar except that the performances are given in function of the stability of the closed-loop dynamics, the gain phase and the margin phase, the bandwith, etc. In general, it is hard to establish a link between the performances in the time domain and the ones in the frequency domain.

More specifically, the system under study is described by a transfer function that can be obtained using the identification approach for instance. Let us denote by $G(s)$ this transfer function. This model must be determined in the first stage of the control design. Then, from the performances and the expertise of the control engineer design we can choose the structure of the controller that may respond properly to the design goal. Then using the appropriate approach we can determine the controller gains.

Therefore, the control design problem consists of determining:

- the structure of the controller
- and its parameters

using the desired performances and some heuristics approaches to force the closed-loop dynamics with the chosen controller to behave as it is desirable. This approach may require some refinement in practice due to different phenomena like neglected dynamics.

The controllers we will consider in this chapter are the classical controllers referred in the literature to as the combination of the proportional (P), integral (I) and derivative (D) actions and their approximations referred also to as phase lag, phase lead and phase lead-lag. The transfer function of the controller will be denoted by $C(z)$. Once the controller is determined, the corresponding difference equation is obtained and implemented in real time using an appropriate microcontroller. For more detail on this topic, we refer the reader to the implementation part where this is detailed.

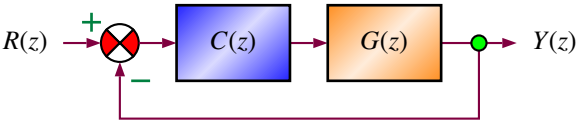


Fig. 5.1 Block diagram of the closed-loop

More often, the system's performances are given in continuous-time since it is more natural to do so. The design procedure can be done either in the continuous-time or the discrete-time. Generally speaking, the design approach uses the following steps:

- the performances are converted to poles
- the structure of the desired controller is chosen
- the controller parameters are determined using the desired poles
- some tunings of the controller's parameters are done to compensate for the discrepancy between the desired and the real behaviors that may result from system's zeros that are not considered in the design procedure.

It is important to notice that determination of the controller parameters can be done either in the continuous-time or the discrete-time. In the continuous-time case, the controller parameters are determined and after that the controller transfer function is converted to discrete-time domain to get the difference equation that we should implement in real time. For the discrete-time, the difference equation is directly obtained and implemented.

The design approach can be one of the following methods:

- Design based on empirical methods
- Design based on root locus method
- Design based on Bode plot method

In the rest of this chapter we will cover these methods and present some examples to show how these techniques apply for real systems. Simulations results will be used to show their validity. The design of the controller is done in continuous-time and then the corresponding discrete-time version of the controller is obtained. The methods developed in Boukas (see [1]) are used in this chapter.

5.3 Design Based on Empirical Methods

The empirical methods are based on the work of Ziegler-Nichols. These methods have the advantage over the other methods since they permit the design of the desired controller even in the absence of the mathematical model of the system. The Ziegler-Nichols methods are mainly based on the response of the dynamical system. Ziegler-Nichols proposed methods that use the time response and others using the frequency response. In the rest of this section we will cover these methods.

Let us first of all concentrate on the time response methods. In these methods, we can handle systems that are stable and unstable in open loop. The first method considers the case of stable system with no poles at the origin neither dominant complex pair of poles. In this case, the step response is given by the one in Fig. 5.1, from which the parameters T, τ and k are determined directly and the Tab. 5.1 is used to fix the controller parameters directly. The corresponding expression of $G(s)$ is given by the following:

$$G(s) = k\frac{e^{-\tau s}}{Ts+1}$$

where k is the gain of the system, τ is the rise time and T is the delay time.

The general expression for the controllers used by the Tab. 5.1 is given by:

$$C(s) = K_P\left[1+\frac{1}{T_I s}+T_D s\right]$$

where K_P, T_I and T_D are the controller parameters to be fixed using Tab. 5.1.

Remark 5.3.1 *It is important to notice that the Ziegler-Nichols method is applicable only when the following holds:*

$$0.15 \leq \frac{\tau}{T} \leq 0.6$$

The following procedure can be used to fix the controller parameter:

1. obtain the step response of the open loop system
2. determine the values of the parameters τ and T from this time response
3. compute the controller parameters using Tab. 5.1
4. compute the closed-loop transfer function and check if the performances are obtained
5. adjust the parameters of the controller if necessary to obtain the desired performances

Remark 5.3.2 *Mostly the time response we will obtain using the controllers fixed by Tab. 5.1 has an overshoot between 10 % and 60 % and an adjustment of the controller parameters is always necessary.*

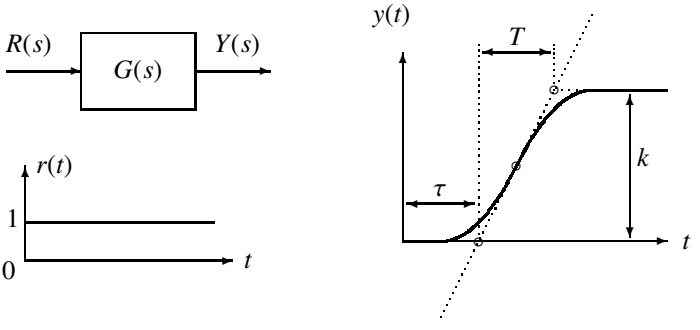


Fig. 5.2 Ziegler-Nichols methods: stable case

Table 5.1 Ziegler-Nichols methods: controller parameters

Controllers	Parameters
P	$K_P = \frac{T}{\tau}$
PI	$K_P = \frac{0.9T}{\tau}$ $T_I = 3.3\tau$
PID	$K_P = \frac{1.2T}{\tau}$ $T_I = 2\tau$ $T_D = 0.5\tau$

Remark 5.3.3 *The values of the gains, K_P is computed using $k = 1$. If it is not the case, the controller gain, K_P has be to corrected by dividing the value of Tab. 5.1 by k. As an example, the gain in case of PID is $K_P = \frac{1.2T}{k\tau}$ instead of $K_P = \frac{1.2T}{\tau}$.*

Example 5.3.1 *To show how this method works, let us consider a dynamical stable system with step response as illustrated in Fig. 5.3. From this figure we get the following parameters:*

$$\begin{aligned} k &= 2 \\ \tau &= 0.2 \\ T &= 1 \end{aligned}$$

From these data, we conclude that the condition of the Ziegler-Nichols is satisfied and therefore, we can use the Tab. 5.1 to fix the desired controller.

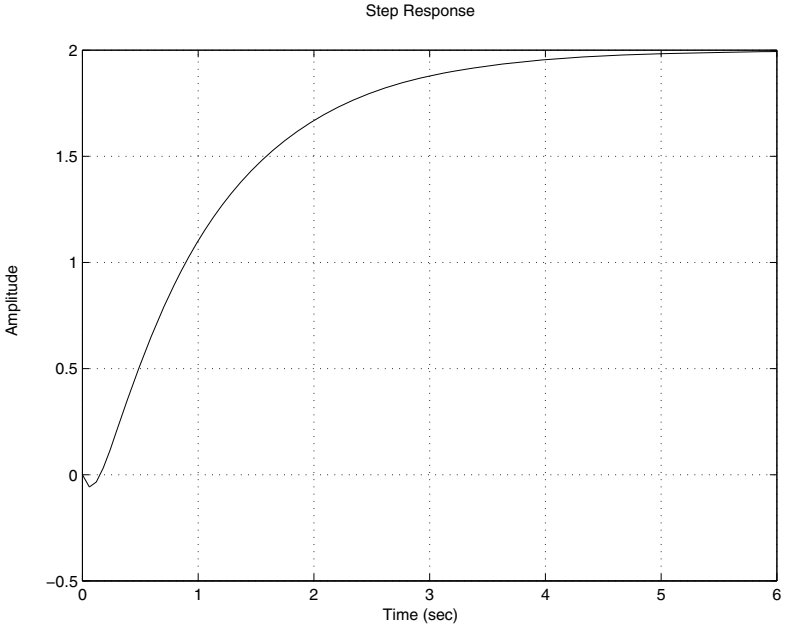


Fig. 5.3 Step response of a stable dynamical system

If we opt for a PID, the parameters of this controller are given by:

$$\begin{aligned} K_P &= \frac{1.2T}{\tau} = 1.2 \\ T_I &= 2\tau = 0.4 \\ T_D &= 0.5\tau = 0.1 \end{aligned}$$

The closed-loop dynamics with this controller is given by:

$$F(s) = \frac{2K_P(T_I T_D s^2 + T_I s + 1)e^{-\tau s}}{s(T_I T s + T_I) + 2K_P(T_I T_D s^2 + T_I s + 1)e^{-\tau s}}$$

Using the Padé approximation, i.e.:

$$e^{-\tau s} = \frac{1 - \frac{\tau}{2}s}{1 + \frac{\tau}{2}s}$$

we get:

$$F(s) = \frac{2K_P(T_I T_D s^2 + T_I s + 1)(1 - \frac{\tau}{2}s)}{a_3 s^3 + a_2 s^2 + a_1 s + a_0}$$

with $a_3 = \frac{\tau}{2}T_I[T - 2K_P T_D]$, $a_2 = T_I\left[T + \frac{\tau}{2} + 2K_P(T_D - \frac{\tau}{2})\right]$, $a_1 = \left[T_I + 2K_P(T_I - \frac{\tau}{2})\right]$ *and* $a_0 = 2K_P$.

The step response of the closed-loop dynamics with this controller is illustrated by Fig. 5.4. From this figure we can see that the overshoot is approximatively 20 % and the other performances are acceptable.

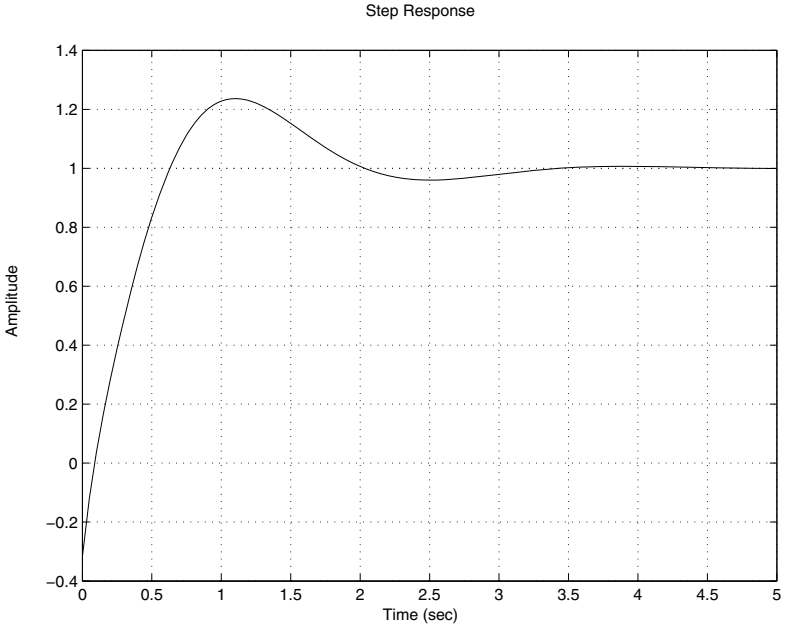


Fig. 5.4 Step response of the closed-loop dynamics with a PID controller

.

Let us now consider the case of unstable systems in open loop. For this class of systems, the approach consists of mounting the system with a PID controller with $T_I = \infty$ and $T_D = 0$ and by varying the gain K_P to bring the closed loop dynamics to the limit of stability (periodic oscillations). Let denote by $\bar{K}_P$ and T_c the corresponding gain and the corresponding period. Fig. 5.5 gives an idea of such set-up. Once these two parameters are determined the ones for the controllers can be obtained using Tab. 5.2.

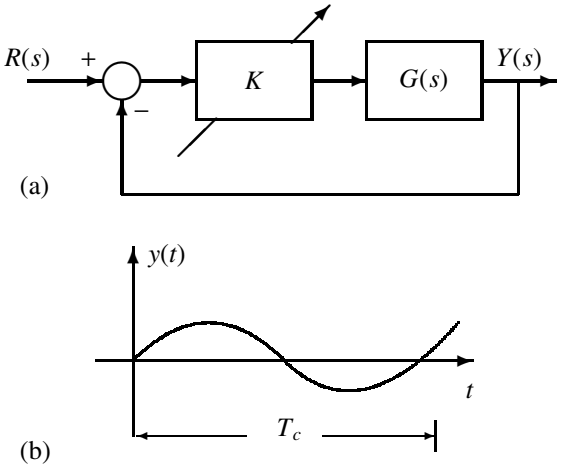


Fig. 5.5 Ziegler-Nichols: unstable case (a) and determination of T_c (b)

Table 5.2 Ziegler-Nichols method: case of unstable systems

Controllers	Parameters
P	$K_P = 0.5\bar{K}_P$
PI	$K_P = 0.45\bar{K}_P$ $T_I = 0.83T_c$
PID	$K_P = 0.6\bar{K}_P$ $T_I = 0.5T_c$ $T_D = 0.125T_c$

The following procedure can be used to fix the controller parameter:

1. mount the system in closed loop with $T_I = \infty$ and $T_D = 0$ and vary the proportional gain of the controller, K_P till the time response gives oscillations as in Fig. 5.5
2. determine the values of the parameters $\bar{K}_P$ and T_c from this time response
3. compute the controller parameters using Tab. 5.2
4. compute the closed-loop transfer function and check if the performances are obtained
5. adjust the parameters of the controller if necessary to obtain the desired performances

Example 5.3.2 *To show how the Ziegler-Nichols method in case of unstable system works, let us consider the following dynamical system:*

$$G(s) = \frac{1}{s(0.1s+1)(0.2s+1)}$$

It is important to notice that this transfer function has a pole at the origin and therefore, the first method will not work.

Now if we mount this system with a proportional controller, we get the following characteristic equation:

$$1 + K_P \frac{1}{s(0.1s+1)(0.2s+1)} = 0$$

The corresponding Routh Hurwitz table is given by:

s^3	1	50
s^2	15	$50K_P$
s^1	$\frac{15\times 50 - 50K_P}{15}$	0
s^0	K_P	0

The critical gain, $\bar{K}_P$ is given by $\bar{K}_P = 15$. The corresponding complex poles are solution of the following equation:

$$15s^2 + 50\bar{K}_P = 0$$

which gives:

$$s = \pm j\sqrt{50} = \pm 7.0711\, j$$

The period T_c is equal to $\sqrt{50}$.

If we choose a PID controller its parameters are given by:

$$\begin{aligned} K_P &= 0.6\bar{K}_P = 9 \\ T_I &= 0.5T_c = 0.4443 \\ T_D &= 0.125T_c = 0.1111 \end{aligned}$$

The closed-loop dynamics with this controller is given by:

$$F(s) = \frac{K_P\left(T_I T_D s^2 + T_I s + 1\right)}{0.02T_I s^4 + 0.3T_I s^3 + (T_I + K_P T_I T_D)\, s^2 + K_P T_I s + K_P}$$

The step response of the closed-loop dynamics is illustrated by Fig. 5.6.

To close this section let us see how we can design PID controllers (P, PI, PID) using the Ziegler-Nichols frequency methods (these methods are mainly based on the idea to assure for the closed-loop dynamics a margin phase between 45^o and 50^o and gain margin greater than 8 *db*). For this purpose, let us assume that the dynamics of the system in open loop is described by:

$$G(s) = k\frac{1}{\Pi_{i=1}^{n}(\tau_i s + 1)}$$

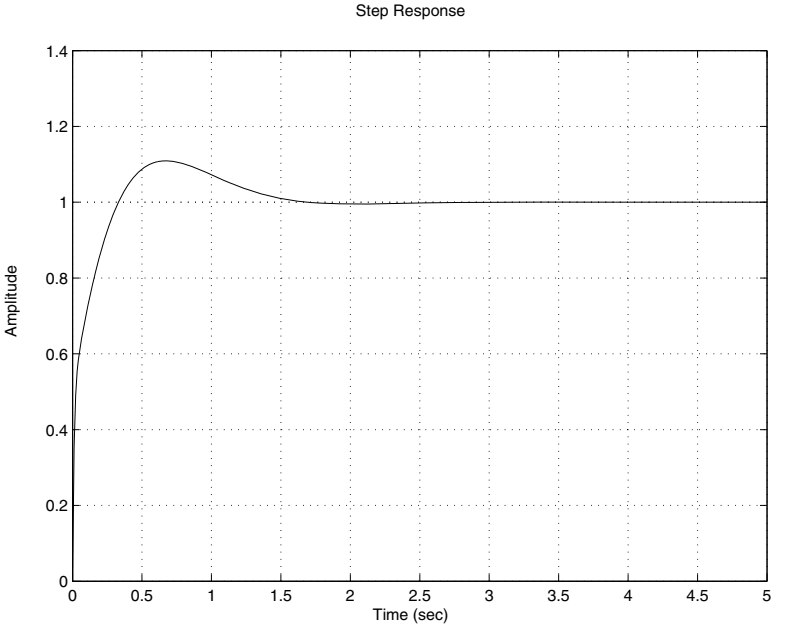


Fig. 5.6 Step response of the closed-loop dynamics with a PID controller

where k is the gain of the system and $\tau_i, i = 1, \cdots, n$ are the different constant time of the system.

By defining $\bar{K}_P$ as the gain in open loop that assures the gain margin and the phase margin, and τ^1 and τ^2 as follows:

$$\begin{aligned} \tau^1 &= \max\{\tau_1, \cdots, \tau_n\} \\ \tau^2 &= \max\{\{\tau_1, \cdots, \tau_n\} - \{\tau^1\}\} \end{aligned}$$

the controller parameters are fixed by Tab. 5.3 . The expression of the PID controller is given by:

$$C(s) = K_P \frac{(\tau^1 s + 1)(\tau^2 s + 1)}{(\tau^1 + \tau^2)s}$$

It is important to notice that the open transfer function is given by:

$$T(s) = C(s)G(s) \tag{5.1}$$

$$= \begin{cases} \frac{kK_P}{\Pi_{i=1}^n(\tau_i s+1)} = \frac{K}{\Pi_{i=1}^n(\tau_i s+1)} & \text{for P controller, with } K = kK_P \\ \frac{kK(T_I s+1)}{T_I s\Pi_{i=1}^n(\tau_i s+1)} = \frac{K(T_I s+1)}{s\Pi_{i=1}^n(\tau_i s+1)} & \text{for PI controller, with } K = \frac{kK_P}{T_I} \\ \frac{kK_P(T_I T_D s^2+T_I s+1)}{T_I s\Pi_{i=1}^n(\tau_i s+1)} = \frac{K(T_I T_D s^2+T_I s+1)}{s\Pi_{i=1}^n(\tau_i s+1)} & \text{for PID controller, with } K = \frac{kK_P}{T_I} \end{cases}$$

The following procedure can be used to design the appropriate controller using the following steps:

Table 5.3 Ziegler Nichols method in frequency domain

Controllers	Parameters
P	$K_P = \frac{\bar{K}_P}{k}$
PI	$K_P = \frac{\bar{K}_P}{k}$ $T_I = \tau^1$
PID	$K_P = \frac{\bar{K}_P}{k}$ $T_I = \tau^1 + \tau^2$ $T_D = \frac{\tau^1\tau^2}{T_I}$

1. determine the open loop transfer function with the compensator as in 5.1
2. plot the bode diagram for $K = 1$ and determine the gain $\bar{K}_P$ that gives the desired phase margin and a gain margin greater than 8 *db*
3. determine the gain, K_P of the controller using:

$$K_P = \begin{cases} \frac{\bar{K}_P}{k} & \text{fpr P controller} \\ \frac{\bar{K}_P T_I}{k} & \text{fpr PI controller, with } T_I = \tau^1 \\ \frac{\bar{K}_P T_I}{k} & \text{fpr PID controller, with } T_I = \tau^1 + \tau^2, T_D = \frac{\tau^1\tau^2}{T_I} \end{cases}$$

4. check if the performances of the system are satisfied. In case of negative answer, adjust the controller parameters to get such performances.

Example 5.3.3 *To show how this method work let us consider the following dynamical system:*

$$G(s) = \frac{4}{(0.1s+1)(0.2s+1)(0.5s+1)}$$

Our goal is to design a PID controller that provides the following performances:

1. *stable system*
2. *margin phase between* 45^o *and* 50^o
3. *margin gain greater than* 8 *db*

First of all following the step of the previous, we have:

$$\begin{aligned} \tau^1 &= 0.5 \\ \tau^2 &= 0.2 \end{aligned}$$

which gives:

$$\begin{aligned} T_I &= \tau^1 + \tau^2 = 0.5 + 0.2 = 0.7 \\ T_D &= \frac{\tau^1\tau^2}{T_I} = \frac{0.2 \times 0.5}{0.7} = 0.1429 \end{aligned}$$

The gain, $\bar{K}_P$ that gives the desired phase margin and gain margin greater than 8 *db is given by:*

$$\bar{K}_P = 3.8019$$

which gives the following gain for the PID controller:

$$K_P = \frac{\bar{K}_P T_I}{k} = \frac{3.8019 \times 0.7}{4} = 0.6653$$

The transfer function of the closed-loop dynamics with this controller is given by:

$$F(s) = \frac{kK_P T_D s^2 + kK_P s + \frac{kK_P}{T_I}}{0.01s^4 + 0.17s^3 + (0.8 + KK_P T_D)\, s^2 + (1 + kK_P)\, s + \frac{kK_P}{T_I}}$$

The step response of the closed-loop dynamics is illustrated by Fig. 5.7

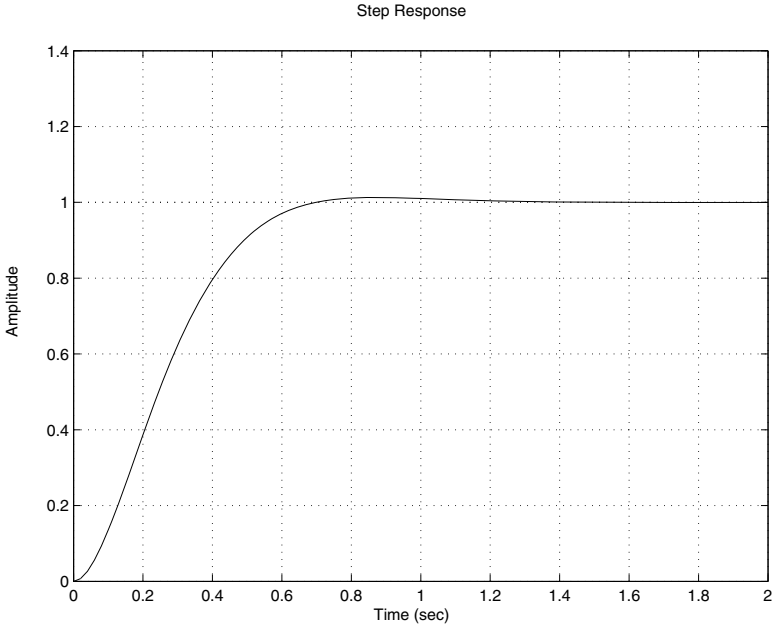


Fig. 5.7 Step response of the closed-loop dynamics with a PID controller

Remark 5.3.4 *For the expression of the controller in the discrete-time and its analytical expression of the recurrent equation for real-time implementation, we will cover this in the next section.*

5.4 Design Based on Root Locus

The root locus technique is a powerful tool for analysis and design of control systems. In this section, we will use it to design a controller that will guarantee the desired performances. The model of the system is supposed to be given in term of a transfer function.

The root locus technique can be used to design the classical controllers. The technique behind this method consists of choosing the controller gains that make the loci passes through given poles that come from the performances. In the rest of this section we will assume that the transfer function $G(s)$ is given by the following expression:

$$G(s) = k\frac{\Pi_{i=1}^{n}(s+z_i)}{\Pi_{i=1}^{n}(s+p_i)}$$

where k, $-z_i$ and $-p_i$ are respectively the gain, the zero and the pole of the system.

Let us firstly concentrate on the design of the proportional controller. Let its transfer function be given by:

$$C(s) = K_P$$

where K_P is the gain of the controller to be determined.

As it is well known from basic control course, the proportional controller acts simultaneously on the transient and the steady state regimes but its capacity is limited. It can reduce the error but never makes it equal to zero.

To compute the gain of the controller we will use the following procedure (see Boukas [1]):

1. compute the characteristic equation of the closed-loop dynamics, i.e.: $1+K_pG(s)$ and let $K = kK_p$
2. draw the root locus for K varying from 0 to infinity
3. determine the intersection between the loci and the line corresponding to the desired damping ratio ξ, ($\cos\theta = \xi$) and get the dominant pair of poles. Let s_d be the one with the positive imaginary part.
4. compute the gain K that gives the pole s_d, then obtain the desired gain for the proportional controller by:

$$K_P = \frac{\Pi_{i=1}^{n}|(s_d+p_i)|}{K\Pi_{i=1}^{m}|(s_d+z_i)|}$$

The lines that we should include in the control loop during the implementation part are:

```
compute the system's error, e
compute the control law using   u = Kp*e
 send the control and wait for the next interrupt
```

Example 5.4.1 *To illustrate this design approach, let us consider a physical system that consists of a dc motor that drives a mechanical load that we would like to control in position.*

The transfer function of this system is given by the following expression:

$$G(s) = \frac{k}{s(\tau s + 1)}$$

with $k = 5$, *and* $\tau = 1s$.

From basic control theory, we can see that the system is unstable. Our desire is to make it stable in the closed-loop with an overshoot less or equal to 5 % and a steady state error equal to zero.

From basic control theory, a proportional controller is enough to reach our goal. To obtain the controller gain, let us follow the steps of the precedent procedure. The characteristic equation is:

$$1 + K_p \frac{5}{s(s+1)} = 0$$

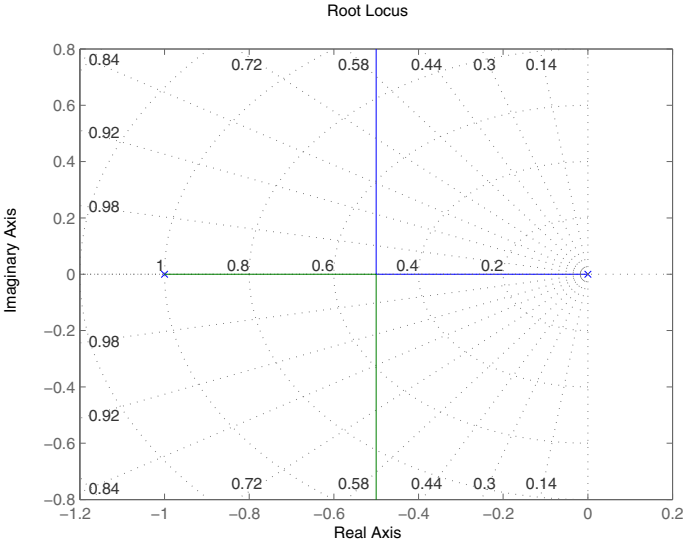


Fig. 5.8 Root locus of $\frac{1}{s(s+1)}$

The root locus of the closed-loop dynamics is given by Fig. 5.8 and from which we get:

$$s_d = -0.5 + j0.5$$

and the corresponding gain is $K = 0.5$. *This gives the following gain for the controller:*

$$K_P = \frac{K}{5} = 0.1$$

The behavior of the closed-loop dynamics is illustrated in Fig. 5.9. The simulation results show the efficiency of the designed controller. The closed-loop dynamics is stable and the overshoot is less than 5 % as it is expected.

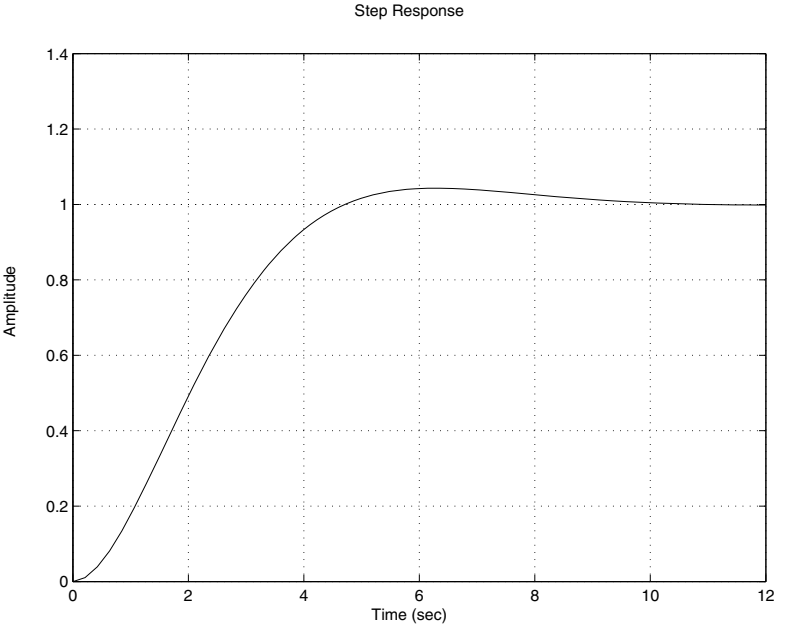


Fig. 5.9 Step response of $\frac{0.5}{s(s+1)+0.5}$

Remark 5.4.1 *It is important to notice that the best setting time at* 2 % *that we can get with this type of controller*

$$t_s = \frac{4}{\zeta w_n} = \frac{4}{0.5} = 8\ sec$$

where ζ *and* w_n *are respectively the damping ratio and the natural frequency of the closed-loop dynamics. This can be checked from the Fig. 5.9*

For the design of a proportional and integral controller the same technique can be used. If we let the transfer function of this controller be as follows:

$$C(s) = K_P + \frac{K_I}{s} = K_P \frac{s+z}{s}, z = \frac{K_I}{K_P}$$

where the gains K_P and K_I have to be determined.

This controller may be used to act simultaneously on the transient and the steady state regimes and therefore overcomes what the proportional controller alone can not perform. Most often the proportional and integral controller is used to make the error equal to zero for a step input and fix the overshoot and the settling time. The following procedure can be used (see Boukas [1]):

1. with the damping ratio and the settling time values, we can determine the dominant pole with the positive imaginary part, s_d
2. using this pole and the angle condition, we can determine the angle of the controller's zero, i.e.:

$$\alpha = \pi - \sum_{i=1}^{m} \angle(s_d + z_i) + \sum_{i=1}^{n+1} \angle(s_d + p_i)$$

The value of the zero is then given by:

$$z = \sigma + \frac{\Im(s_d)}{\tan(\alpha)}$$

with $\sigma = \frac{3}{\zeta\omega_n}$ if the settling is fixed at 5 %

3. plot the loci of

$$K\frac{s+z}{s}\frac{\Pi_{i=1}^{m}(s+z_i)}{\Pi_{i=1}^{n}(s+p_i)}$$

and determine the gain K that gives the pole s_d using

$$K = \frac{\Pi_{i=1}^{n+1}|(s_d + p_i)|}{\Pi_{i=1}^{m+1}|(s_d + z_i)|}$$

4. the controller gains are given by:

$$\begin{aligned} K_P &= \frac{K}{k} \\ K_I &= zK_P \end{aligned}$$

To obtain the corresponding discrete-time transfer function we can use one of the approaches presented earlier. The third approach (trapezoidal method) is used here and it consists of replacing s by $\frac{2}{T}\frac{z-1}{z+1}$, where T is the chosen sampling period. Using this we get:

$$C(z) = \frac{\left(\frac{K_I T}{2} + K_P\right)z + \frac{K_I T}{2} - K_P}{z - 1}$$

This gives the relationship that links the control and the error at sample k:

$$u_k = u_{k-1} + ae_k + be_{k-1} \tag{5.2}$$

where $a = \frac{K_I T}{2} + K_P$ and $b = \frac{K_I T}{2} - K_P$

The lines that we should include during the implementation in the control loop are:

```
compute the system's error, e
compute the control law using the controller expression
save the present error and the present control
send the control and wait for the next interrupt
```

Example 5.4.2 *To illustrate this design approach, let us consider a physical system that consists of a dc motor that drives a mechanical load that we would like to control in speed.*

The transfer function of this system is given by the following expression:

$$G(s) = \frac{k}{\tau s + 1}$$

with $k = 5$, *and* $\tau = 1s$.

From basic control theory, we can see that the settling time of the open-loop system with 5 % is $t_s = 3\tau = 3s$. *The system doesn't have an overshoot and the response is a little bit slow.*

Our desire is to make the system faster with an overshoot less or equal to 5 %, a settling time t_s *at 5% less or equal to* $1s$, *and a steady state error for a step input equal to zero.*

To solve this design problem, let us proceed in continuous-time domain. For this purpose, let us first of all mention that the type of the system is equal to zero and therefore to guarantee that the error is equal to zero at the steady state for a step input, we need at least a proportional and integral controller.

Following the procedure of the proportional integral controller we have:

1. *the dominant pole with the positive imaginary value is given by:*

$$\begin{aligned} s_d &= -\zeta\omega_n + j\omega_n\sqrt{1-\zeta^2} \\ &= -3 + 3j \end{aligned}$$

This comes from the fact that we have:

$$\begin{aligned} \zeta &= -\frac{\log\left(\frac{d}{100}\right)}{\sqrt{\pi^2 + \left(\log\left(\frac{d}{100}\right)\right)^2}} = 0.6901 \\ \omega_n &= \frac{3}{\zeta t_s} = 4.3472 \end{aligned}$$

2. *using this pole, we get:*

$$\begin{aligned} \alpha &= \pi - 0 + \angle(-3 + 3j) + \angle(-2 + 3j) \\ &= 180 + 135 + 123.6901 \\ &= 78.6901 \end{aligned}$$

which gives the following value for the zero

$$\begin{aligned} z &= -3 - \frac{3}{\tan(78.6901)} \\ &= -3.6 \end{aligned}$$

3. *the loci of the controlled system is given by Fig. 5.10, from which we conclude that* $K = 4.73$.

4. *the controller gains are:*

$$\begin{aligned} K_P &= 0.9460 \\ K_I &= 3.4056 \end{aligned}$$

The root locus of the system is illustrated in Fig. 5.10.

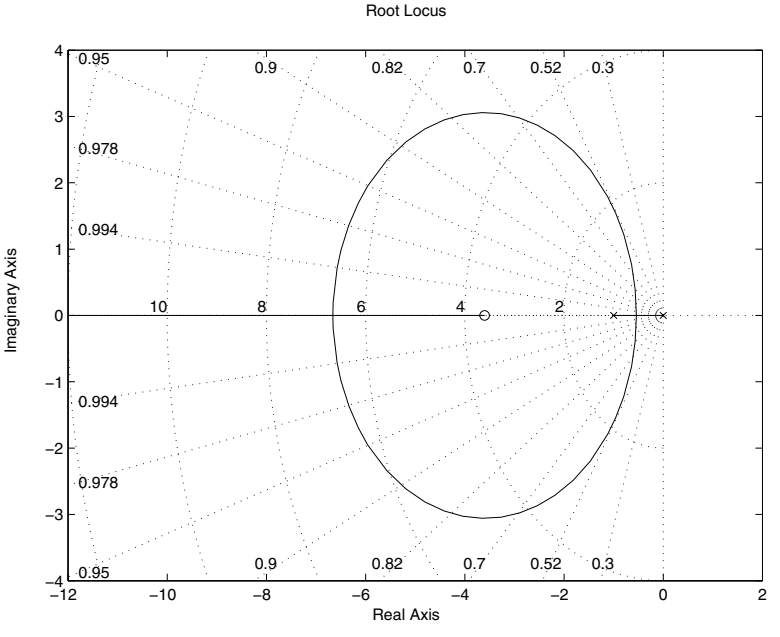


Fig. 5.10 Root locus of $\frac{s+z}{s(s+1)}$, $z = -3.6$

The behavior of the closed-loop dynamics is illustrated in Fig. 5.11. The simulation results show that the overshoot is over what we desire while the settling time is acceptable. To reduce the overshoot, we can redo the design by pushing a little bit the zero to the left and get new set of gains for the controller.

In some circumstances, we may have a system that has acceptable steady regime but the transient one needs some improvements. In this case the proportional and

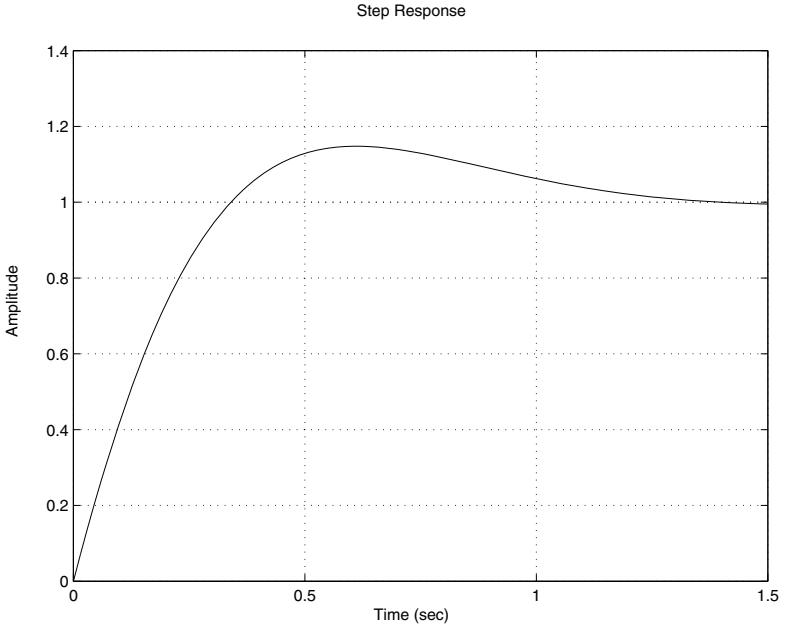


Fig. 5.11 Step response of $\frac{5K_P s+5K_I}{s^2+(1+5K_P)s+5K_I}$

derivative controller can be used. The transfer function of this controller is given by:

$$C(s) = K_P + K_D s$$

where K_P and K_D are the controller gains that we should compute in order to satisfy the desired performances.

To design the proportional and derivative controller the following procedure can be used (see [1]):

1. with the damping ratio and the settling time values, we can determine the dominant pole with the positive imaginary part, s_d
2. using this pole and the angle condition, we can determine the angle of the controller zero as it was done for the proportional and integral controller previously, i.e.:

$$\alpha = \pi - \sum_{i=1}^{m} \angle(s_d + z_i) + \sum_{i=1}^{n} \angle(s_d + p_i)$$

The value of the zero is then given by:

$$z = \sigma + \frac{\Im(s_d)}{\tan(\alpha)}$$

with $\sigma = \frac{3}{\zeta\omega_n}$ if the settling fixed at 5 %

3. plot the loci of

$$K\frac{(s+z)\Pi_{i=1}^{m}(s+z_i)}{\Pi_{i=1}^{n}(s+p_i)}$$

and determine the gain K that gives the pole s_d using

$$K = \frac{\Pi_{i=1}^{n}|(s_d+p_i)|}{\Pi_{i=1}^{m+1}|(s_d+z_i)|}$$

4. the controller gains are given by:

$$\begin{aligned} K_D &= \frac{K}{k} \\ K_P &= zK_D \end{aligned}$$

The corresponding discrete-time version of this controller, using the same transformation as for the proportional and integral controller, is given by:

$$C(z) = \frac{(K_P + \frac{2K_D}{T})z + (K_P - \frac{2K_D}{T})}{z+1}$$

This gives the relationship that links the control and the error at sample k:

$$u_k = -u_{k-1} + ae_k + be_{k-1} \tag{5.3}$$

where $a = K_P + \frac{2K_D}{T}$ and $b = K_P - \frac{2K_D}{T}$.

Remark 5.4.2 *For this controller the backward approach is more appropriate for the derivative action and the trapezoidal for the integral action. In tis case we get:*

$$u(k) = ae(k) + be(k-1)$$

with $a = K_P + \frac{K_D}{T_s}$ *and* $b = -\frac{K_D}{T_s}$.

The lines that we should include in the control loop are:

```
compute the system's error, e
compute the control law using the controller expression
save the present error and the present control
send the control and wait for the next interrupt
```

Example 5.4.3 *To illustrate this design approach, let us consider a physical system that consists of a dc motor that drives a mechanical load that we would like to control in position.*

The transfer function of this system is given by the following expression:

$$G(s) = \frac{k}{s(\tau s+1)}$$

with $k = 5$, *and* $\tau = 1s$.

The settling time of the system with 5 % is $t_s = 3\tau = 3s$. *The system is unstable. Our desire is to make the system stable and faster with an overshoot less or equal to 5 % and a settling time* t_s *at 5% less or equal to* $0.5s$.

To solve this design problem, let us proceed in continuous-time domain. For this purpose, let us first of all mention that the type of the system is equal to one and therefore the error is equal to zero at the steady state for a step input. For the transient to be improve we need at least a proportional controller but here we use a proportional and derivative controller to get better settling time.

Following the previous procedure we have:

1. *the dominant pole with the positive imaginary value is given by:*

$$\begin{aligned} s_d &= -\zeta\omega_n + j\omega_n\sqrt{1-\zeta^2} \\ &= -6 + 6j \end{aligned}$$

2. *using this pole, we get:*

$$\begin{aligned} \alpha &= \pi + \angle(-6+6j) + \angle(-5+6j) \\ &= 180 + 135 + 129.8056 \\ &= 84.8056 \end{aligned}$$

which give the following value for the zero

$$\begin{aligned} z &= -6 - \frac{3}{\tan(84.8056)} \\ &= -6.7273 \end{aligned}$$

3. *the loci of the controlled system is given by Fig. 5.12, from which we conclude that* $K = 10.8$.

4. *the controller gains are:*

$$\begin{aligned} K_D &= 2.1600 \\ K_P &= 14.5310 \end{aligned}$$

The simulation results illustrated in Fig. 5.13 show that we have an overshoot greater than the one we need but the settling time is acceptable. To reduce the overshoot we can move the zero a little bit to left and get new set of gains for the controller.

For some systems, we need to improve simultaneously the transient and the steady regimes. In this case, the most appropriate choice is the proportional, integral and derivative controller. Its transfer function is given by:

$$\begin{aligned} C(s) &= K_P + \frac{K_I}{s} + K_D s \\ &= K_D \frac{(s+a_1)(s+a_2)}{s} \end{aligned}$$

with $K_P = K_D\,(a_1 + a_2)$ and $K_I = K_D a_1 a_2$

To design the proportional, integral and derivative controller, the following procedure can be used (see [1]):

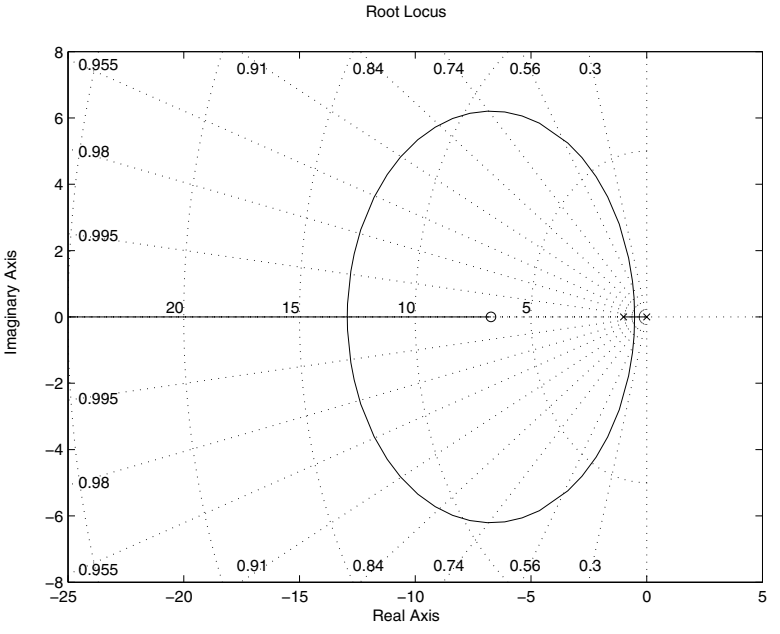


Fig. 5.12 Root locus of $\frac{s+z}{s(s+1)}$, $z = 6.7273$

1. to determine the parameter a_1 (that is related to the first zero), we choose the slow pole and proceed with a zero-pole cancelation. If we denote by $-p_s$ the slow pole, the parameter a_1 is then given by:

$$a_1 = p_s$$

2. with the damping ratio value and the settling time, we can determine the dominant pole with the positive imaginary part, s_d

3. using this pole and the angle condition, we can determine the angle of the controller zero that corresponds to $s + a_2$, i.e.:

$$\alpha = \pi - \sum_{i=1}^{m} \angle(s_d + z_i) + \sum_{i=1}^{n} \angle(s_d + p_i)$$

Remark 5.4.3 *It is important to notice that a pole has been cancelled by the zero at position* $-a_1$ *and a new pole at* 0 *has been added to the equation.*

The value of the second zero is then given by:

$$a_2 = \sigma + \frac{\Im(s_d)}{\tan(\alpha)}$$

with $\sigma = \frac{3}{\zeta\omega_n}$ if the settling fixed at 5 %

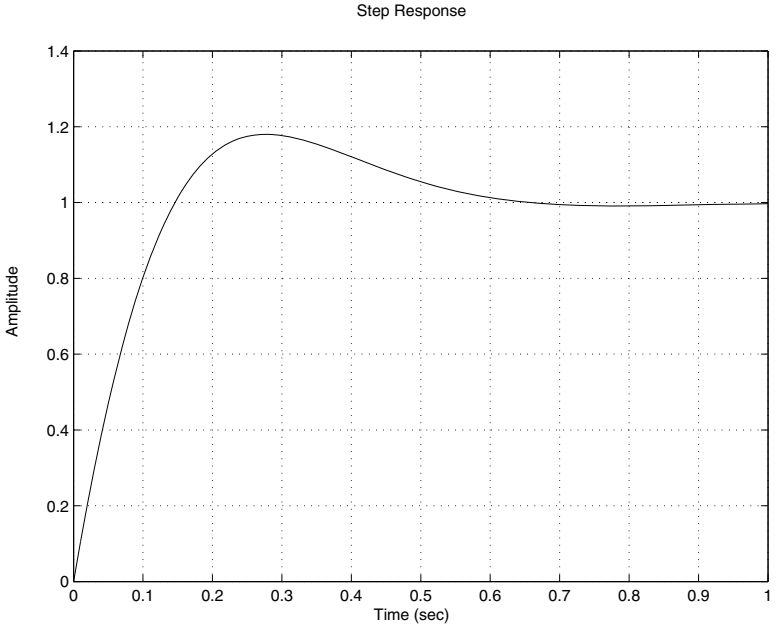


Fig. 5.13 Step response of $\frac{s+z}{s(s+1)}$, $z = 6.7273$

4. plot the loci of

$$K\frac{(s+a_2)\Pi_{i=1}^{m}(s+z_i)}{s\Pi_{i=1}^{n-1}(s+p_i)}$$

and determine the gain K that gives the pole s_d using

$$K = |s_d|\frac{\Pi_{i=1}^{n-1}|s_d+p_i|}{|s_d+a_2|\Pi_{i=1}^{m}|s_d+z_i|}$$

5. the controller gains are given by:

$$\begin{aligned} K_D &= \frac{K}{k} \\ K_P &= K_D\,(a_1+a_2) \\ K_I &= K_D a_1 a_2 \end{aligned}$$

The corresponding discrete-time version of this controller, using the same transformation as for the proportional and integral controller, is given by:

$$C(z) = \frac{K_P z^2 + \left(\frac{K_I T}{2} + \frac{2K_D}{T}\right)z + \left(\left(\frac{K_I T}{2} + \frac{2K_D}{T}\right) - K_P\right)}{(z-1)(z+1)}$$

This gives the relationship that links the control and the error at sample k:

$$u_k = u_{k-2} + ae_k + be_{k-1} + ce_{k-2} \tag{5.4}$$

where $a = K_P + \frac{K_I T}{2} + \frac{2K_D}{T}$, $b = K_I T - \frac{4K_D}{T}$ and $c = \frac{K_I T}{2} + \frac{2K_D}{T} - K_P$.

Remark 5.4.4 *As we said regarding the schema for the discretization of the controller we can use here also the trapezoidal schema for the integral action and the backward schema for the derivative one. In this case we get:*

$$u(k) = u(k-1) + ae(k) + be(k-1) + ce(k-2)$$

with $a = K_P + \frac{K_D}{T_s} + \frac{K_I T_s}{2}$, $b = -K_P - 2\frac{K_D}{T_s} + \frac{K_I T_s}{2}$, $c = \frac{K_D}{T_s}$;

The lines that we should include in the control loop are:

```
compute the system's error, e
compute the control law using the controller expression
save the present error and the present control
send the control and wait for the next interrupt
```

Example 5.4.4 *To show how this procedure can be used to design a proportional, integral and derivative controller, let us consider the following system with:*

$$G(s) = \frac{3}{(s+1)(s+3)}$$

For this system we would like to guarantee that the steady state error for a step input is equal to zero with an overshoot less or equal to 5 % and a settling time about 1*s. Following the steps of the previous procedure we get:*

1. *the slow pole in this case is equal to* -1 *and therefore, the parameter is then* $a_1 = 1$.
2. *the dominant pole with the positive imaginary value is given by:*

$$\begin{aligned} s_d &= -\zeta\omega_n + j\omega_n\sqrt{1-\zeta^2} \\ &= -3 + 3j \end{aligned}$$

3. *using this pole, we get since the pole* -1 *has been cancelled with the zero at* $-a_1$*:*

$$\begin{aligned} \alpha &= \pi + \angle(-3+3j) + \angle(3j) \\ &= 180 + 135 + 90 \\ &= 45 \end{aligned}$$

which give the following value for the zero

$$\begin{aligned} a_2 &= -3 + \frac{3}{\tan(45)} \\ &= -6 \end{aligned}$$

4. *the loci of the controlled system is given by Fig. 5.14, from which we conclude that* $K = 2.99$ *is the appropriate one that give the closest dominant poles and the damping ratio (*$s_d = -2.99 \pm 2.99j$, $\zeta = 0.707$, $d = 4.51$ % $w_n = 4.23$*).*

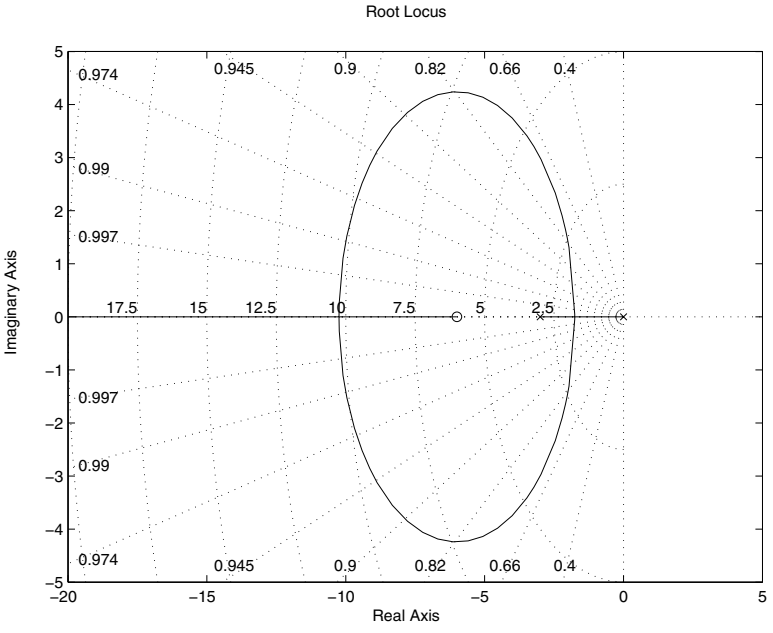


Fig. 5.14 Root locus of $\frac{s+a_2}{s(s+3)}$, $a_2 = 6$

5. *the controller gains are:*

$$\begin{aligned} K_D &= 0.9967 \\ K_P &= 6.9769 \\ K_I &= 5.9802 \end{aligned}$$

The closed-loop transfer function is given by:

$$F(s) = \frac{kK_D s^2 + kK_P s + kK_I}{s^3 + (4 + kK_D)\, s^2 + (3 + kK_P)\, s + kK_I}$$

The behavior of the closed-loop dynamics is illustrated in Fig. 5.15
The simulation results show the efficiency of the designed controller.

The phase lead controller can be used to approximate the proportional and derivative one. The transfer function of this controller is given by:

$$C(s) = K_P \frac{aTs + 1}{Ts + 1}$$

where K_P, a and T are parameters to be computed with $a > 1$.

This controller offers the advantage to improve the transient regime. This can be obtained if the placement of the pair pole/zero is well positioned since we can pull the asymptotic branches to get a smaller settling time.

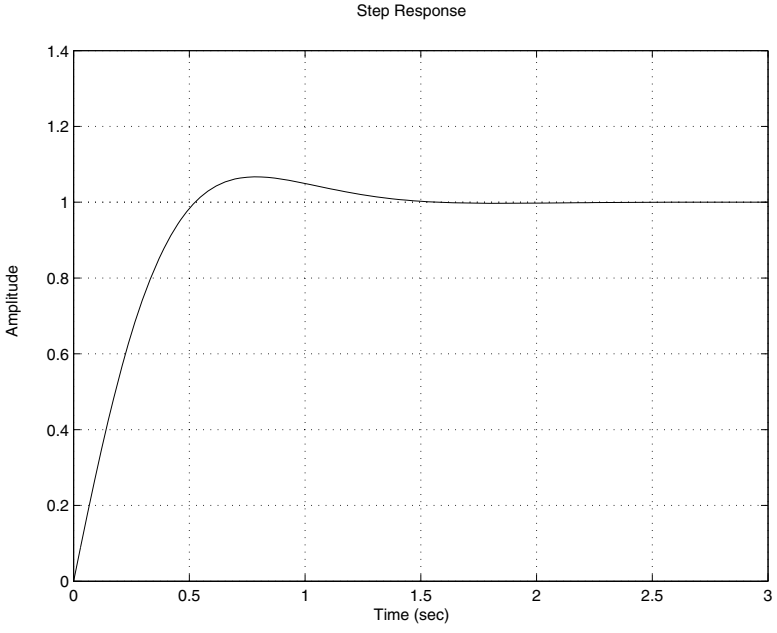


Fig. 5.15 Step response of $\frac{s+a_2}{s(s+3)}$, $a_2 = 6$

The following procedure can be used to design such controller (see [1]):

1. with the damping ratio and the settling time values, we can determine the dominant pole with the positive imaginary part, s_d
2. by varying the system gain, try to get the desired dominant poles, if it is not possible, determine the contribution in angle of the pair pole/zero that the controller has to add
3. place the pole and the zero of the phase lead controller in order to compensate for this desired angle. Among the possibilities that can be used in this case, we can place the zero at a value equal to the real part of the dominant poles and then using the angle condition we can determine the pole position.
4. determine the value for the controller gain in order to satisfy the error
5. check if the desired specifications are obtained. In case of negative answer replace the pair pole/zero of the controller and repeat the design procedure

Example 5.4.5 *To show how the procedure of the phase lead controller can be applied, let us consider the position control for a dc motor driving a mechanical load as it was considered before. Let the dynamics be given by:*

$$G(s) = \frac{2}{s\,(s+2)}.$$

Our goal in this example is to guarantee that the closed-loop system is stable, with a settling time at 5% equal to 0.5 s, an overshoot less or equal to 5% and having a zero error for a step input.

First of all notice that the time constant of the system is equal to 0.5 *s which may give the best settling time at 5 % with a proportional controller equal to* 6 *s. Our requirement in regard to the settling time is far from this value and therefore a proportional controller is not sufficient for our case.*

To respond to these specifications a phase lag controller can be used and its design can be done using the previous procedure.

1. *based on the settling time and the overshoot requirements we get the following dominant pole with positive imaginary value:*

$$s_d = -6 + 6j$$

This desired poles can not be obtained by varying the gain of a proportional controller and therefore a design of a phase lead controller is needed. From this value for the dominant pole, we have:

$$\begin{aligned} \angle G(s_d) &= \angle(2) - \angle(-6+6j) - \angle(-5+6j) \\ &= 0 - 135 - 123.6901 = -258.6901 \end{aligned}$$

The controller can be designed to bring an angle $258.6901 - 180 = 78.6901$. *This is obtained if* $\angle(aTs+1) - \angle Ts + 1 = 78.6901$

2. *following the method we used in the procedure, we get* $aT = \frac{1}{6}$ *and therefore* $\angle(Ts+1) = 90 - 78.6901 = 11.3099$. *This gives the location of the controller pole. Using now the following trigonometric relation we get:*

$$\tan(11.3099) = \frac{\Im(s_d)}{\frac{1}{T} - |\Re(s_d)|}$$

which gives $T = 0.0278$. *This in turn implies that* $a = \frac{1}{6T} = 5.9952$.

3. *The open-loop transfer function of the compensated system is then given by:*

$$G_c(s) = K\frac{(s+6)}{s(s+2)(s+35.9712)}$$

which gives the following gain, K that corresponds to the desired pole s_d*:*

$$K = \frac{|s_d|s_d+2||s_d+35.9712||}{|s_d+6|} = 311.7120$$

The corresponding controller gain is $K_P = \frac{K}{ak} = 25.9968$. *The root locus of the compensated system is illustrated by Fig. 5.16*
The closed-loop transfer function with this controller is given by:

$$F(s) = \frac{2aK_P\left(s+\frac{1}{aT}\right)}{s^3 + \left(2+\frac{1}{T}\right)s^2 + \left(\frac{2}{T}+2aK_P\right)s + \frac{2K_P}{T}}$$

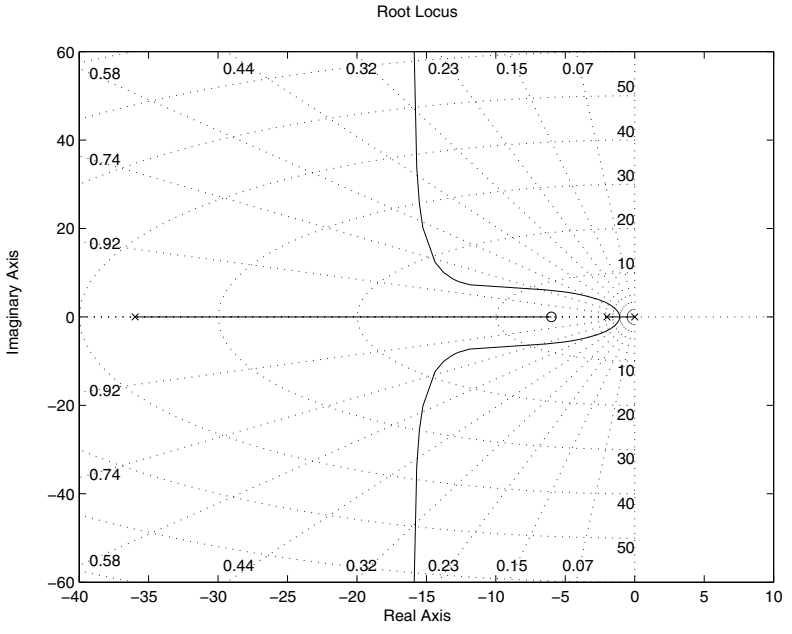


Fig. 5.16 Root locus of $\frac{s+\frac{1}{aT}}{s(s+2)(s+\frac{1}{T})}$

The behavior of the closed-loop dynamics is illustrated in Fig. 5.17.
The simulation results show the efficiency of the designed controller. It is clear that the performances are a little bit far from the desired ones. This is due to the place of the zero of the controller that we can from see from Fig. 5.16. We can play with this position by pushing it to the left and we will get what we want.

Remark 5.4.5 *It is important to notice that phase lead controller or the phase lag or the phase lead-lag controllers are not able to to make the error equal to zero since they can't improve the type the system. But they can improve it if it is constant.*

The phase lag controller can be used to approximate the proportional and integral one. Its task is to improve the steady state regime if it is well designed. The pair pole/zero of the controller is put close to the origin.The transfer function of this controller is given by:

$$C(s) = K_p \frac{aTs + 1}{Ts + 1}$$

where K_P, a and T are parameters to be computed with $a < 1$.

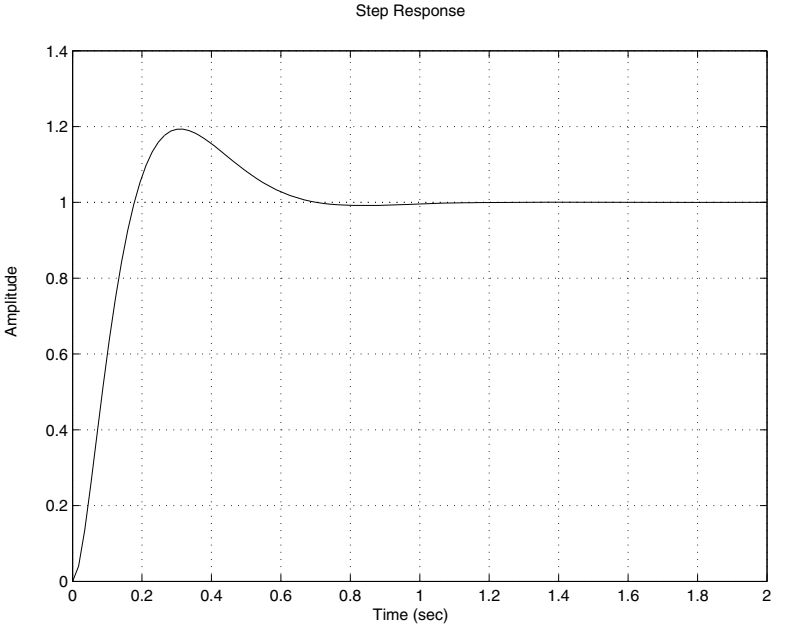


Fig. 5.17 Step response of $F(s) = \frac{2aK_P\left(s+\frac{1}{aT}\right)}{s^3+\left(2+\frac{1}{T}\right)s^2+\left(\frac{2}{T}+2aK_P\right)s+\frac{2K_P}{T}}$

To have an idea of the design approach let us assume that the system to be controller is described by:

$$G(s) = k\frac{\Pi_{i=1}^{m}(s+z_i)}{\Pi_{i=1}^{n}(s+p_i)}$$

where k is the gain, $-z_i, i = 1, \cdots, m$ and $-p_i, i = 1, \cdots, n$ are respectively the zeros and the poles of the system.

In fact, if we write the transfer function of the controller as:

$$C(s) = K_P\frac{s+z}{s+p}$$

with $K_P = aK_p$, $z = \frac{1}{aT}$ and $p = \frac{1}{T}$.

With the gain of the controller only, the constant error is given by:

$$K_1 = kK_P\frac{\Pi_{i=1}^{m} z_i}{\Pi_{i=1}^{n} p_i}$$

In order to improve the steady state error, we would get a constant error, K_2, greater than K_1. By introducing the zero and the pole of the controller this constant error is given by:

$$K_2 = kK_P\frac{z}{p}\frac{\Pi_{i=1}^{m} z_i}{\Pi_{i=1}^{n} p_i}$$

Our desire is that the new pair of pole/zero of the controller doesn't change the transient regime which is acceptable for the designer and the main goal is to change the steady state regime only by reducing the error.

Using the expressions of K_1 and K_2, we get:

$$\frac{K_1}{\frac{\Pi_{i=1}^{m} z_i}{\Pi_{i=1}^{n} p_i}} = kK_P = \frac{K_2}{\frac{z}{p}\frac{\Pi_{i=1}^{m} z_i}{\Pi_{i=1}^{n} p_i}}$$

This implies that:

$$a = \frac{p}{z} = \frac{K_1}{K_2} < 1$$

Therefore, if we choose T in a way that the pole and the zero are close each other (to be cancelled in the open transfer function of the system), the open loop transfer function of the controlled system becomes:

$$C(s)G(s) = kK_P \frac{\Pi_{i=1}^{m}(s + z_i)}{\Pi_{i=1}^{n}(s + p_i)}$$

The idea we will use here is mainly based on the improvement of the steady state error. The following procedure can be used to design such controller (see [1]):

1. with the damping ratio and the settling time values, we can determine the pole dominant with the positive imaginary part, s_d and determine the gain that gives such poles. Compute the corresponding constant error.

2. determine the constant error, K_1, with a proportional controller. Determine the constant error, K_2 when the pole and the zero of the controller are considered. The parameter a of the controller is given by:

   $$a = \frac{K_1}{K_2}$$

   This parameter, a is also given by:

   $$a = \frac{p}{z}$$

3. the value for T is chosen in a way to make the pole and the zero of the controller are close each other and at the same time close to the origin to improve the steady error. This choice will imply that the angle contribution of the controller is very small.

4. determine the gain, $\bar{K}_P$, using the following relation:

   $$\bar{K}_P = \left[\frac{|s_d + p|}{|s_d + z|}\right]\left[\frac{\Pi_{i=1}^{n}|s_d + p_i|}{\Pi_{i=1}^{n}|s_d + z_i|}\right]$$

   then detrmine the controller gain, K_P by:

   $$K_P = \frac{\bar{K}_P}{ak}$$

5. check if the specifications are similar to the desired ones. In the case of negative answer adjust the placement of the pole and the zero of the controller and repeat the procedure

Example 5.4.6 *To show how the procedure of the phase lag controller design can be applied, let us consider the position control for a dc motor driving a mechanical load as it was considered before. Let the dynamics be given by:*

$$G(s) = \frac{2}{s\,(s+2)}.$$

Our goal in this example is to guarantee that the closed-loop system is stable, with a settling time at 5% equal to 3 s, an overshoot less or equal to 5% and having an error for a ramp input less or equal to 0.01.

To respond to these specifications a phase lag controller can be used and its design can be done using the previous procedure.

1. *based on the settling time and the overshoot requirements we get the following dominant pole with positive imaginary value:*

$$s_d = -1 + 1\,j$$

The root locus the system with a proportional controller is given by Fig. 5.18

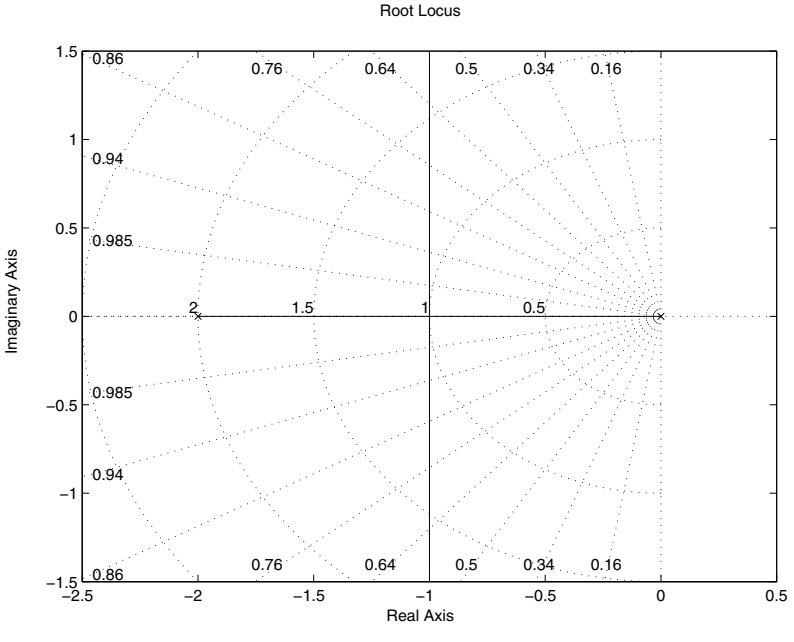


Fig. 5.18 Root locus of $\frac{1}{s(s+2)}$

The gain that gives this pair of poles is given by:

$$K_1 = \frac{|s_d||s_d + 2|}{1} = 2.0$$

This correspond to an error equal:

$$e(\infty) = \frac{1}{2} = 0.5$$

which far from the desired one.

2. *To get our error we need a constant K_2 equal to 100. This implies that the factor a of the controller is given by:*

$$a = \frac{K_1}{K_2} = \frac{2}{100} = 0.02 = \frac{p}{z}$$

3. *since the pole and the zero of the controller have to be placed closed to each other and close to the origin. If we place the zero at -0.3 which a pole for the controller at -0.006 using the fact that $a = \frac{p}{z}$. The value of T can be computed using either the expression of the zero of the one of the pole. This gives $T = 166.6667$.*

4. *The open-loop transfer function of the compensated system is then given by:*

$$G_c(s) = K\frac{(s + 0.3)}{s(s + 2)(s + 0.006)}$$

which gives the following gain, K that corresponds to the desired pole s_d:

$$K = \frac{|s_d|s_d + 2||s_d + 0.01||}{|s_d + 0.5|} = 2.3102$$

The corresponding controller gain is $K_P = \frac{K}{ak} = 57.7549$. The root locus the system with a proportional controller is given by Fig. 5.19
The closed-loop transfer function with this controller is given by:

$$F(s) = \frac{2aK_P\left(s + \frac{1}{aT}\right)}{s^3 + \left(2 + \frac{1}{T}\right)s^2 + \left(\frac{2}{T} + 2aK_P\right)s + \frac{2K_P}{T}}$$

The behavior of the closed-loop dynamics is illustrated in Fig. 5.20
The root locus of the compensated system unfortunately doesn't pass through the desired poles. The closed ones are $s_d = -0.8082 \pm 1.14j$ that corresponds to a gain $K = 2.56$, that gives a gain $K_P = 64$. With this gain we get an overshoot approximately equal to 11 %.
The behavior of the closed-loop dynamics with this new setting is illustrated in Fig. 5.21

Remark 5.4.6 *It is important to notice that the overshoot is a little bit far from the desired one and it is the same for the settling time. This discrepancy is due to the presence of the zero that he introduce high overshot once it is close to the origin.*

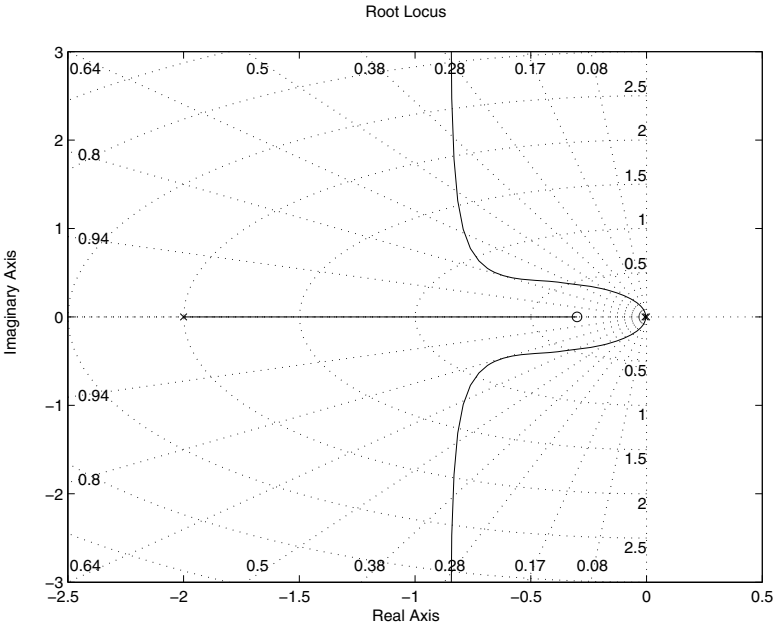


Fig. 5.19 Root locus of $\frac{s+0.3}{s(s+2)(s+0.06)}$

And also due to the fact the cancellation pole zero of the controller is not correct since the pole is a little bit far from the zero.

The phase lead-lag controller is designed to approximate the PID controller. It has the advantage as the PID has to act on both the transient and the steady regimes. Previously we have seen how to design the phase lead controller and the phase lag controller. The first one is used to act on the transient while the second acts of the steady state regime.

The transfer function of this controller is given by:

$$C(s) = K_P \frac{s + \frac{1}{a_1 T_1}}{s + \frac{1}{T_1}} \frac{s + \frac{1}{a_2 T_2}}{s + \frac{1}{T_2}}$$

where K_P is the controller gain, a_1 with $a_1 > 1$ and T_1 are the parameter of the lead part, while a_2 with $a_2 < 1$ and T_2 are the parameter of the phase lag part.

To design such controller, we use the approaches used to design separately the phase lead and the phase lag controller. First, without the phase lag controller, we design the phase lead controller to improve the transient regime. After, we add the phase lag controller to improve the steady state regime while keeping the transient regime as it was improved by the phase lead controller.

The following procedure procedure can be used to design the phase lead-lag controller:

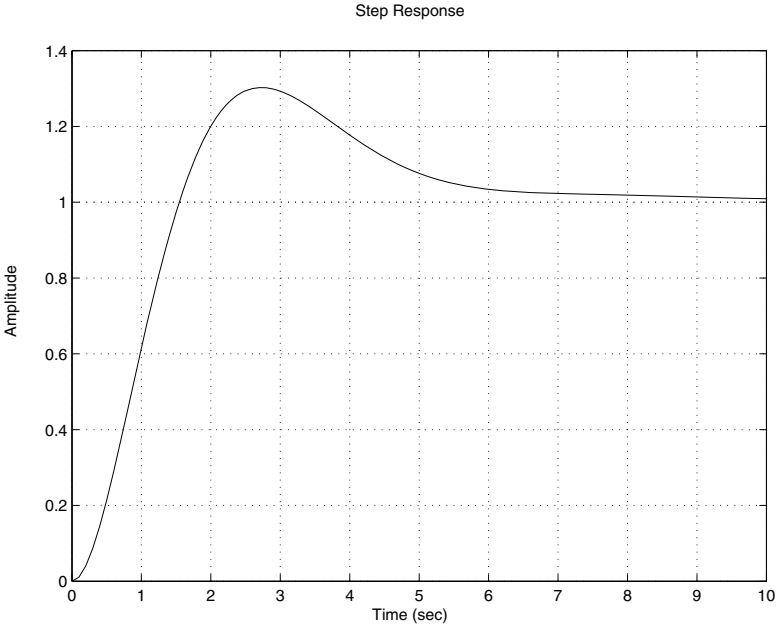


Fig. 5.20 Step response of $F(s) = \frac{2aK_P\left(s+\frac{1}{aT}\right)}{s^3+\left(2+\frac{1}{T}\right)s^2+\left(\frac{2}{T}+2aK_P\right)s+\frac{2K_P}{T}}$

1. without the phase lead-lag controller, see if with a proportional controller, we can guarantee the desired performances. Analyze the system with a proportional controller and determine how much the transient regime has to be improved
2. design the phase lead controller (gain, pole and zero)
3. analyze the compensated system with a phase lead controller and determine how much the steady state regime has to be improved
4. design the phase lag controller (gain, pole and zero)
5. check if the specifications are similar to the desired ones. In the case of negative answer adjust the placement of the pole and the zero of the controller and repeat the procedure

Example 5.4.7 *To see how the procedure for the design of phase lead-lag controller applies, let us consider the following dynamical system:*

$$G(s) = \frac{2}{s(s+2)}$$

For this system with a proportional controller, the best settling time at 5 % we can obtain is equal to 3 s. We can also get an overshot less or equal to 5 %. The steady state error for a step input is equal to zero, while the one for a ramp is constant and

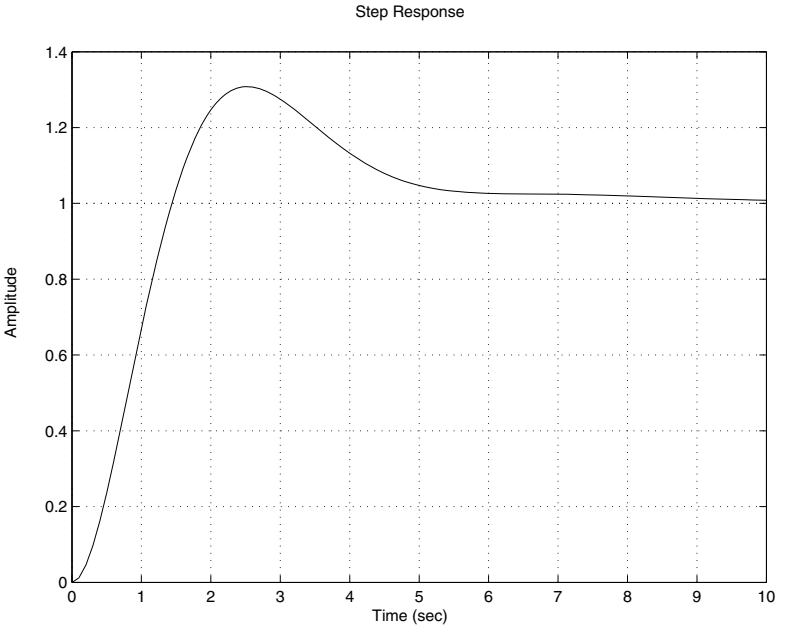


Fig. 5.21 Step response of $F(s) = \frac{2aK_P\left(s+\frac{1}{aT}\right)}{s^3+\left(2+\frac{1}{T}\right)s^2+\left(\frac{2}{T}+2aK_P\right)s+\frac{2K_P}{T}}$

can be fixed by acting on the gain controller. A "trade-off" between the overshoot and the steady state error has to be done. It is clear that the proportional controller will not give the good trade-off. The phase lead-lag controller will give the better one.

For this purpose let us assume that we desire the following specifications:

1. *stable system in closed loop*
2. *an overshoot less or equal to 5 %*
3. *a settling time at 5 % about 2 s*
4. *a steady state error for a ramp input less or equal to 0.01*

To design the phase lead-lag controller that provides the desired performances, let us follow the previous procedure:

1. *From Fig. 5.18, it is clear that the settling time requirement can be obtained using a proportional controller. From the specifications, we get the dominant pair of poles that gives what we are looking for:*

$$s_d = -1.5 + 1.5\,j$$

This desired pair of poles can not be obtained by just varying the gain of the proportional controller. A phase lead controller is needed for this purpose. The phase of the transfer function at s_d is given by:

$$\begin{aligned}\angle G(s_d) &= \angle(2) - \angle(0.5000 + 1.5000\,j) - \angle(-1.5000 + 1.5000\,j)\\ &= 0 - 71.5651 - 135 = 206.5651\end{aligned}$$

The controller phase lead controller can be used to bring the angle contribution of $206.5651 - 180 = 26.5651$. *This can be obtained if we impose that* $\angle(s_d + \frac{1}{a_1 T_1}) - \angle(s_d + \frac{1}{T_1}) = 26.5651$.
Following the procedure of the phase lead controller design, if we impose that the zero of the controller is place at the real part of the dominant poles, we get:

$$a_1 T_1 = \frac{1}{1.5}$$

and therefore, we have:

$$\angle(s_d + \frac{1}{T}) = 90 - 26.5651 = 63.4349$$

To get the position of the pole, i.e:, we use the following trigonometric relation:

$$\frac{1}{T_1} = |\Re(s_d)| + \frac{\Im(s_d)}{\tan(11.3099)} = 2.2500$$

which implies $T_1 = 0.4444$. *And from the relation* $a_1 T_1 = \frac{1}{1.5}$, *we get* $a_1 = 1.5002$.

2. *for the design of the phase lag controller, notice that the compensated system with a phase lead controller has the following open transfer function:*

$$G_1(s) = a_1 k K_p \frac{s + \frac{1}{a_1 T_1}}{s(s+2)\left(s + \frac{1}{T_1}\right)}$$

with $k = 2$.
The root locus of this transfer function is illustrated by Fig. 5.22
The gain K_1 *that gives the poles that are close to the desired poles is given by:*

$$K_1 = 3.87$$

The corresponding poles are $s_d = -1.5 \pm 1.54\,j$ *with an overshoot approximatively equal to 5 %.*
The error constant with this controller is given by:

$$K_2 = \lim_{s\to 0} s a_1 k K_p \frac{s + \frac{1}{a_1 T_1}}{s(s+2)\left(s + \frac{1}{T_1}\right)} = a_1 k K_p \frac{1}{a_1 T_1}\frac{T_1}{2} = K_P$$

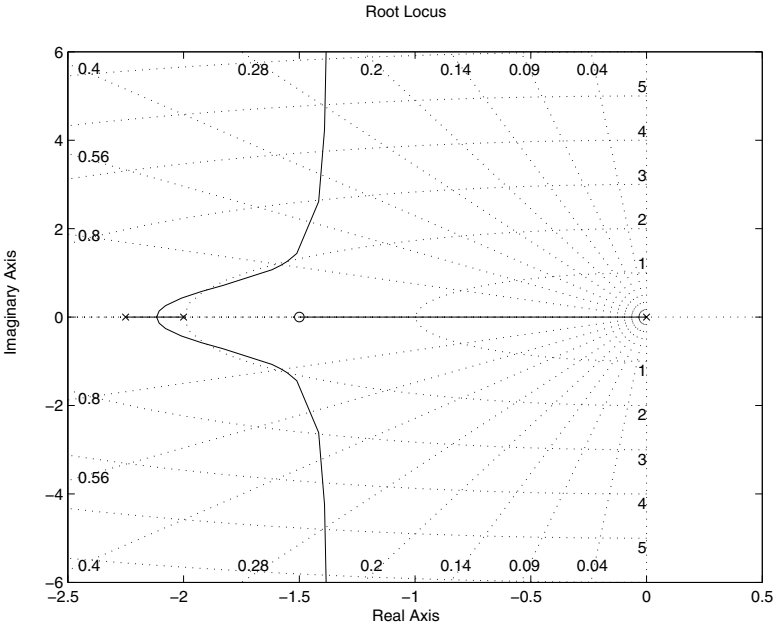


Fig. 5.22 Root locus of $\frac{s+\frac{1}{a_1 T_1}}{s(s+2)\left(s+\frac{1}{T_1}\right)}$

To get the desired error we need to fix K_p to 100. This gives the following parameter, a_2 for the phase lag controller:

$$a_2 = \frac{K_1}{K_2} = \frac{3.87}{100} = 0.0387$$

Since the procedure for the design of the phase lag controller requires that we have to place the pole and the zero of the controller close each other and close to the origin. A proper choice consists of placing the zero at -0.1. This implies using the relation

$$a_2 = \frac{p}{z}$$

that the pole is placed at $p = -0.0039$ and since the pole is equal to:

$$p = -\frac{1}{T_2}$$

we get $T_2 = 256.4103$.
The open loop transfer function of the system with the phase lead-lag controller is give by:

$$G_2(s) = a_1 k K_p \frac{s + \frac{1}{a_1 T_1}}{s(s+2)\left(s + \frac{1}{T_1}\right)}$$

with $k = 2$.
The root locus of this transfer function is illustrated by Fig. 5.23

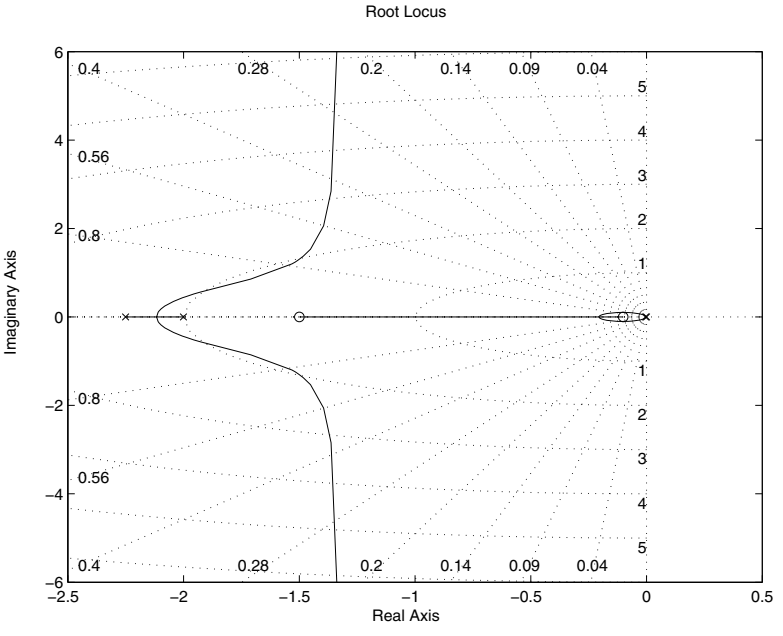


Fig. 5.23 Root locus of $\frac{\left(s+\frac{1}{a_1T_1}\right)\left(s+\frac{1}{a_2T_2}\right)}{s(s+2)\left(s+\frac{1}{T_1}\right)\left(s+\frac{1}{T_2}\right)}$

From this figure we get the closest poles of the desired ones are:

$$s_d = -1.5 \pm 1.31j$$

that gives an damping ratio about 0.753 and an overshoot equal to 2.73 %.
The gain that gives this pair of poles is equal to:

$$\bar{K}_P = 3.35$$

From this data, we get the following gain for the controller:

$$K_P = \frac{\bar{K}_P}{a_1 a_2 k} = \frac{3.35}{2 \times 1.5002 \times 0.0387} = 28.8506$$

The expression of the designed controller is given by:

$$C(s) = K_P \frac{s+\frac{1}{a_1T_1}}{s+\frac{1}{T_1}} \frac{s+\frac{1}{a_2T_2}}{s+\frac{1}{T_2}}$$

with $K_P = 28.8506$, $a_1 = 1.5002$, $T_1 = 0.4444$, $a_2 = 0.0387$ *and* $T_2 = 256.4103$.

The closed-loop transfer function with this controller is given by:

$$F(s) = \frac{kK_P\left(a_1T_1a_2T_2s^2 + (a_1T_1 + a_2T_2)s + 1\right)}{b_4s^4 + b_3s^3 + b_2s^2 + b_1s + b_0}$$

with $b_4 = T_1T_2$, $b_3 = (T_1 + T_2 + 2T_1T_2)$, $b_2 = (1 + 2(T_1 + T_2) + kK_Pa_1T_1a_2T_2)$, $b_1 = (2 + kK_P(a_1T_1 + a_2T_2))$ *and* $b_0 = kK_P$.
The behavior of the closed-loop dynamics is illustrated in Fig. 5.24

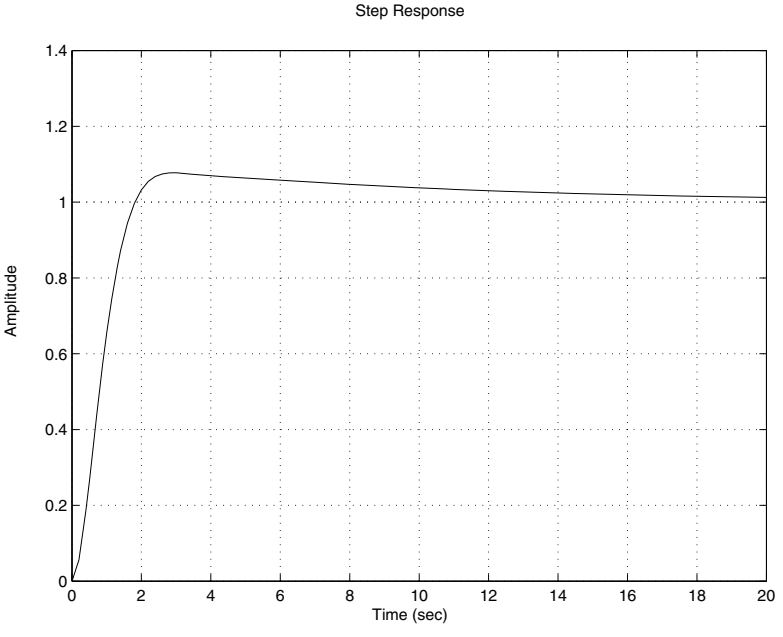


Fig. 5.24 Step response of $F(s)$

It seems that the settling time is a little bit far from the desired one. To overcome this, we play with the positions of the poles and the zeros of the controller and repeat the procedure.

5.5 Design Based on Bode Plot

The design methods we will develop in this section have the advantage over those presented in the previous section by the fact they don't need the knowledge of the mathematical model of the system to be controlled as it required by the techniques based on root locus method. The objective of this section is to cover the methods that we can use for designing the controllers treated in the previous section by using the frequency domain.

The design procedures for the different controllers we will cover here are mainly based on the fact to assure that the closed-loop dynamics of the system will have a phase margin, $\Delta\phi$ satisfying:

$$45^o \leq \Delta\phi \leq 50^o$$

while the gain margin, ΔG satisfies $\Delta G \geq 8\ db$.

In the rest of this section we assume that the system is described by the following transfer function:

$$G(s) = k\frac{a_m s^m + \cdots + 1}{s^l(a_n s^n + \cdots + 1)}$$

where l is the type of the system, $l + n$ is the degree of the system and $m < n + l$ is the degree of the numerator of the system that we suppose to be causal.

Our goal in this section consists of designing a controller that respond to some given performances. The controllers we consider in this section are those treated in the previous sections. It is important to notice that the idea used in the methods we will cover is based on the deformation of the magnitude and phase curves locally to satisfy the desired performances.

Remark 5.5.1 *It is important to notice that this method doesn't apply for unstable system.*

Let firstly consider the design of the proportional controller ($C(s) = K_P$). This controller has limited actions and can only move vertically the magnitude curve without affecting the phase curve. The open loop transfer function of the compensated system is given by:

$$\begin{aligned} G_c(s) &= kK_P\frac{a_m s^m + \cdots + 1}{s^l(a_n s^n + \cdots + 1)} \\ &= K\frac{a_m s^m + \cdots + 1}{s^l(a_n s^n + \cdots + 1)} \end{aligned}$$

The following procedure can be used for the design of the proportional controller that responds to the desired performances:

1. obtain the Bode plot for the compensated system, $G_c(s)$, with $K = 1$
2. determine the frequency, w_c, for which the phase margin is equal to 45^o
3. determine the magnitude at this frequency and compute the gain, $\bar{K}_P$ that will move the magnitude curve vertically to get the desired phase margin. A gain greater than one will move the magnitude curve up while a gain less than one will move it down. The controller gain is given by:

$$K_P = \frac{\bar{K}_P}{k}$$

4. draw the Bode diagram of the compensated system, with the computed gain and check that the gain margin is greater than 8 db

Example 5.5.1 *To show how the design procedure for the proportional controller works, let us consider the following dynamical system:*

$$G(s) = \frac{2}{(0.1s+1)(0.2s+1)(0.5s+1)}$$

The performances we would like to have for this system are:

1. *stable system in closed-loop*
2. *phase margin about* 45^o
3. *gain margin greater than* 10 *db*

To design our proportional controller, let us follow the previous procedure.

1. *the open loop transfer function of the compensated system,* $T(s)$ *is given by:*

$$T(s) = 2K_P \frac{1}{(0.01s+1)(0.2s+1)(0.5s+1)}$$

2. *The Bode plot of this transfer function with* $K = 1$ *is illustrated in Fig. 5.25*

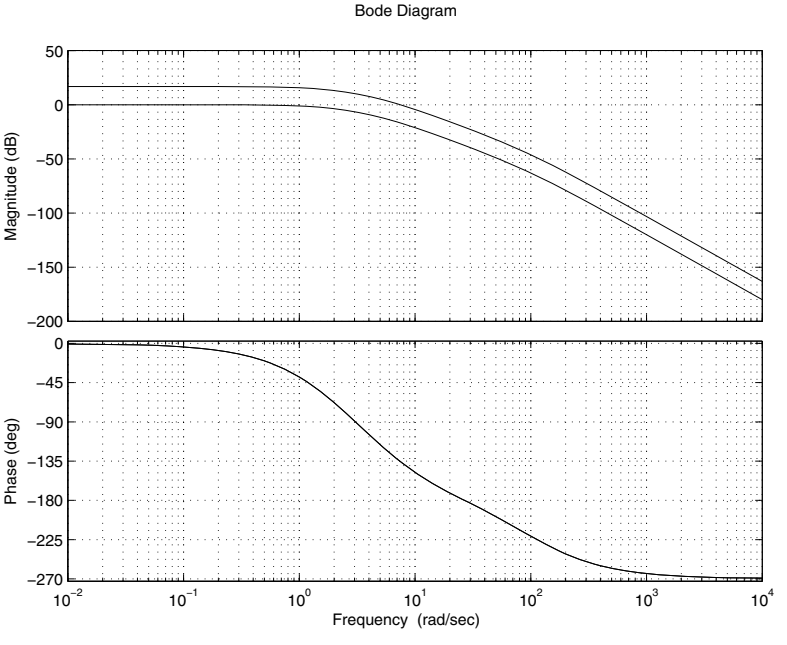


Fig. 5.25 Bode plot of $T(s)$, with $K = 1$, and $K = kK_P$

From this figure we conclude that at the frequency $w_1 = 7.44\ rd/s$ *we have* $\Delta\phi = 45^o$. *The corresponding magnitude is equal to* $|T(jw_1)| = -16.8$.

3. *The corresponding gain that allows us to move the magnitude curve by* 16.8 *db to get the desired phase margin is given by:*

$$\bar{K}_P = 16.8\ db$$

which implies $\bar{K}_P = 10^{\frac{16.8}{20}} = 6.9183$
Finally, we get the controller gain as follows:

$$K_P = \frac{\bar{K}_P}{k} = \frac{6.9183}{2} = 3.4592$$

The open loop transfer function of the compensated system is given by:

$$T(s) = 2 \times 3.4592 \frac{1}{(0.01s+1)(0.2s+1)(0.5s+1)}$$

The Bode plot of this transfer function is reported in Fig. 5.25. If we compute the phase and the gain margins we get:

$$\Delta G = 10.8363$$
$$\Delta\phi = 44.9849$$

The corresponding frequencies are:

$$w_g = 26.65\ rd/s, for\ the\ gain\ margin$$
$$w_p = 7.44\ rd/s, for\ the\ phase\ margin$$

4. *The transfer function of the closed loop with this controller is given by:*

$$F(s) = \frac{kK_P}{0.001s^3 + 0.107s^2 + 0.71s + 1 + kK_P}$$

with $k = 2$.
The behavior of the closed-loop dynamics is illustrated in Fig. 5.26.
This response has a steady state error equal to 0.13. *The proportional controller is unable to make equal to zero but it can be reduced by increasing the gain. This may degrade the transient regime.*

Remark 5.5.2 *It is important to notice that the system considered in the previous example is of type zero and therefore, the error for a step input with a proportional controller is constant and it is given by:*

$$e(\infty) = \frac{1}{1 + kK_p}.$$

From this expression, it is impossible to make the error equal to zero by increasing the gain of the controller. Incrementing the type of the system is a solution that can be given by the PI controller.

Let us now focus on the design of the PI controller using the Bode method. As we have seen previously increase the type of the system by one and therefore, it may bring the steady state error to zero. Its disadvantage is that settling time may increase.

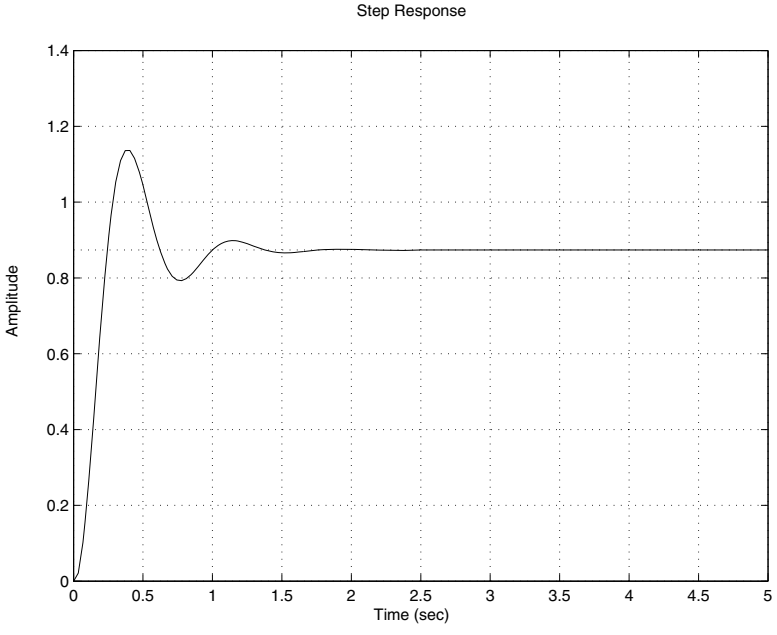


Fig. 5.26 Step response of $F(s)$

To design the PI controller, let us assume that its transfer function is described by:

$$\begin{aligned} C(s) &= K_P + \frac{K_I}{s} \\ &= \frac{1 + \tau_n s}{\tau_i s} \end{aligned}$$

with $K_P = \frac{\tau_n}{\tau_i}$ and $K_I = \frac{1}{\tau_i}$.

Using this, the open loop transfer function of the compensated system is given by:

$$T(s) = C(s)G(s) = K\,(1 + \tau_n s)\,\frac{b_m s^m + \cdots + b_1 s + 1}{s^{l+1}\,(a_n s^n + \cdots + a_1 s + 1)}$$

with $K = \frac{k}{\tau_i}$

The following procedure can be used for the design of this controller:

1. determine the slowest pole that is not equal to those at the origin (pole that corresponds the highest time constant) and proceed with a zero/pole cancelation. This will allow us to determine the parameter τ_n by:

$$\tau_n = \max\{\tau_1, \cdots, \tau_\nu\}$$

where τ_j, $j = 1, \cdots, \nu$ are the time constant of the system to be controlled.

2. determine the gain $\bar{K}_P$ that gives the desired phase margin using the Bode plot and obtain:

$$\tau_i = \frac{k}{\bar{K}_P}$$

3. determine the gains K_P and K_I of the controller using:

$$K_P = \frac{1}{\tau_i}$$
$$K_I = \frac{\tau_n}{\tau_i}$$

4. determine the open loop transfer function of the compensated system and check if the desired performances are obtained or not. In case of negative response adjust τ_n and repeat the procedure design.

Example 5.5.2 *To show how this procedure works let us consider the following dynamical system:*

$$G(s) = \frac{1}{(s+1)(s+5)(s+10)}$$

and design a PI controller that gives a steady error equal to zero and a phase margin about 45^o *and a gain margin greater than* 8 *db.*

To answer these performances, let us follow the previous procedure:

1. *the open transfer function of the system to be controller has* 1, 0.2 *and* 0.1 *as time constants. The maximum one is equal to* 1 *and therefore by canceling the corresponding pole by the controller's zero, we get:*

$$\tau_n = 1\ s$$

2. *the open loop transfer function with the pole/zero cancellation is given by:*

$$T(s) = \frac{0.02K}{s(0.2s+1)(0.1s+1)}$$

The Bode plot of this transfer function is shown at Fig. 5.27
At $w = 2.8\ rd/s$, *the phase margin is equal to* 45^o *and at this frequency the magnitude is equal to* -10.5 *db. To get such phase margin we need to translate up the magnitude curve by* 17.5 *db which implies the use of a gain:*

$$\bar{K}_P = 10^{\frac{10.5}{20}} = 3.3497$$

which implies in turn:

$$\tau_i = \frac{0.02}{\bar{K}_P} = \frac{0.02}{3.3497} = 0.0060$$

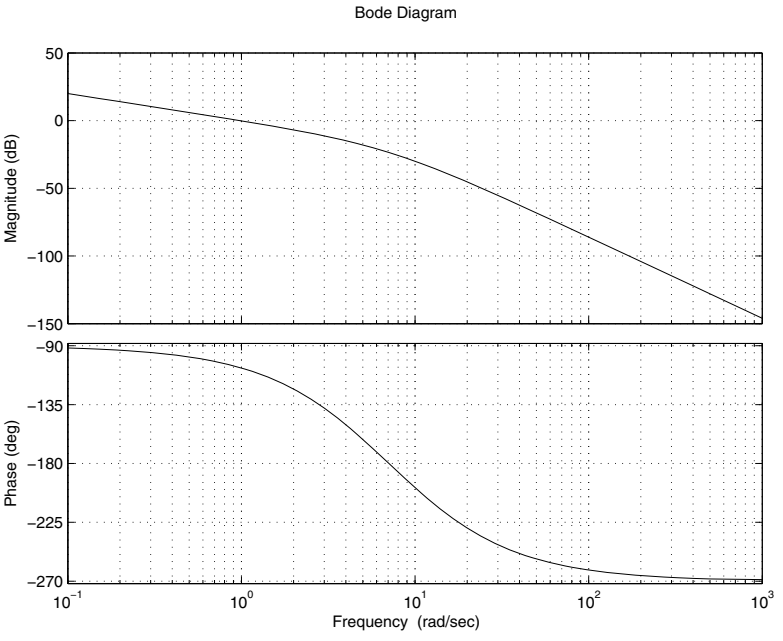


Fig. 5.27 Bode plot of $T(s)$, with $K = 1$

3. *the controller gains are given by:*

$$K_P = \frac{1}{\tau_i} = \frac{1}{0.0027} = 166.6667$$
$$K_I = \frac{\tau_n}{\tau_i} = \frac{1}{0.0027} = 166.6667$$

4. *with this controller we can check that the phase margin is equal to* 45.1^o *but the gain margin is equal to* 4.5 *db. The closed loop transfer function with this controller is given by:*

$$F(s) = \frac{K_P}{s^3 + 15s^250s + K_P}$$

If we accept the gain margin as it is now, the design is complete otherwise we have to modify the value for τ_n *and repeat the design*
The behavior of the closed-loop dynamics with the computed controller is illustrated in Fig. 5.28
The settling time at 5 % is equal to 1.47 *s which is acceptable and the error for a step input is equal to zero.*

Let us now focus on the design of the PD controller using the Bode method. This controller improves the transient regime. The transfer function of this controller is given by:

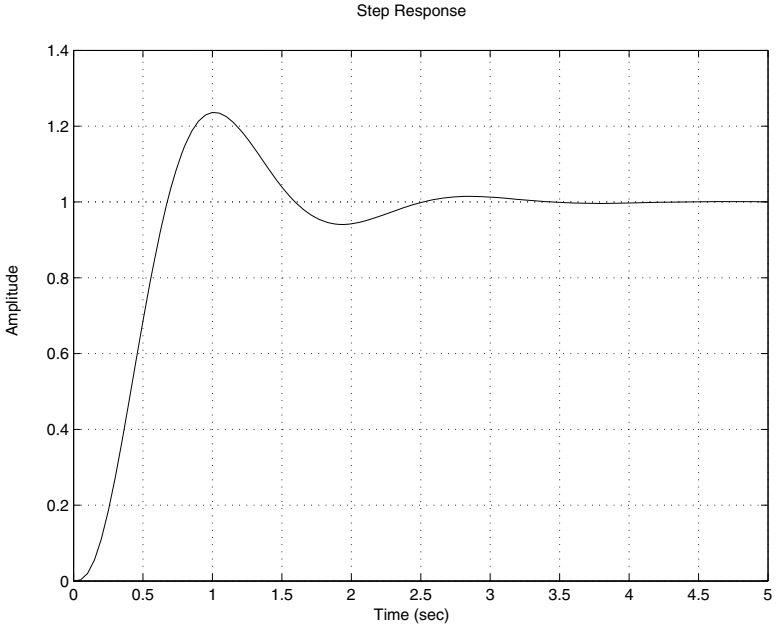


Fig. 5.28 Step response of $F(s)$

$$C(s) = K_P + K_D s = K_P (1 + \tau_D s)$$

with $\tau_D = \frac{K_D}{K_P}$.

The open loop transfer function of the compensated system is given by:

$$T(s) = K \frac{(1 + \tau_D s)(b_m s^m + \cdots, b_1 s + 1)}{s^l (a_n s^n + \cdots, a_1 s + 1)}$$

where $K = kK_P$

The design of the PD controller is brought to the determination of the two gains K_P and K_D. The following procedure can be used for the design of this controller:

1. from the error specifications, determine the gain, $\bar{K}_P$ that gives the desired error
2. draw the Bode diagram of the system:

$$\bar{K}_P \frac{b_m s^m + \cdots, b_1 s + 1}{s^l (a_n s^n + \cdots, a_1 s + 1)}$$

   and determine the frequency, w_m at the which the magnitude is equal to $-20\ db$
3. since the cut frequency of the PD controller is equal to $\frac{1}{\tau_D}$, at the frequency $\frac{10}{\tau_D}$, the contribution of the PD controller to the magnitude and the phase are respectively $20\ db$ and 90^o. If we select τ_D such that:

$$\tau_D = \frac{10}{w_m}$$

the phase margin of the compensated system is given:

$$\Delta\phi_c = \Delta\phi + 90$$

where $\Delta\phi$ is the phase margin of the system without the controller at the frequency w_m
If

$$\Delta\phi_c \begin{cases} < 40^o & \text{choose another controller} \\ > 50^o & \text{reduce the parameter, } \tau_D \text{ till } \Delta\phi_c = 45^o \end{cases}$$

4. compute the controller's gains using:

$$K_P = \frac{\bar{K}_P}{k}$$
$$K_D = \bar{K}_P \tau_D$$

5. check if the desired specifications are obtained or not

Example 5.5.3 *To show how the procedure of the design of the PD controller works, let us consider the following dynamical system:*

$$G(s) = \frac{4}{s(0.1s+1)(4s+1)}$$

As specifications we consider the following:

1. *stable system*
2. *phase margin equal to* 45^o
3. *steady state error equal to 0.1*

To satisfy these specifications a PD controller has to be designed. For this purpose let is follow the previous procedure:

1. *from the error specification, we need to fix* $\bar{K}_P$ *to 10.*
2. *the Bode diagram of:*

$$\frac{\bar{K}_P}{s(0.1s+1)(4s+1)}$$

is illustrated by Fig. 5.29 which shows that magnitude is equal to -20 *db when the frequency* $w_m = 4.73\ rd/s$. *The parameter* τ_D *is then given by:*

$$\tau_D = \frac{10}{w_m} = \frac{10}{4.73} = 2.1142$$

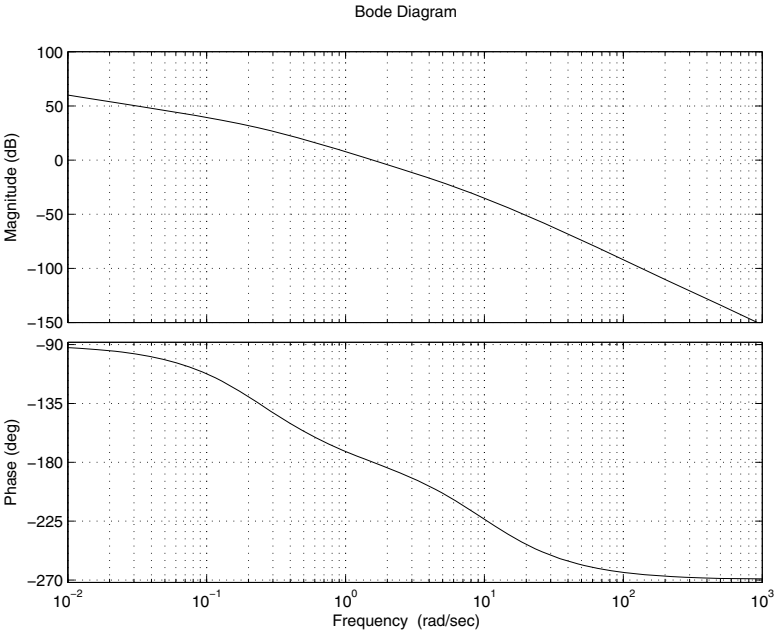


Fig. 5.29 Bode plot of $T(s)$, with $K = 10$

3. *the phase of the system with the controller at* $w_m = 4.73\ rd/s$ *is equal to* -202^o. *The phase margin of the compensated system is given by:*

$$\Delta\phi_c = 180 - 202 + 90 = 68^o$$

The phase margin is greater than 45^o *and we should decrease the parameter* τ_D. *Therefore if we select* $\tau_D = \frac{10}{9.1} = 1.0989$, *the phase margin in this case is equal to* 49^o

4. *the controller gains are give by:*

$$\begin{aligned} K_P &= \frac{\bar{K}_P}{k} = \frac{10}{4} = 2.5 \\ K_D &= \bar{K}_P \tau_D = 2.4 \times 1.0989 = 2.7473 \end{aligned}$$

5. *the open loop transfer function of the compensated system is given by:*

$$T(s) = \frac{4\,(K_P + K_D s)}{s(0.1s+1)(4s+1)}$$

This controller gives a phase margin about 61.5^o. *The closed-loop transfer function is given by:*

$$F(s) = \frac{4(K_s + K_p)}{0.1s^3 + 4.1s^2 + (1 + 4K_D)s + 4K_P}$$

The step response of the compensated system is represented in Fig. 5.30

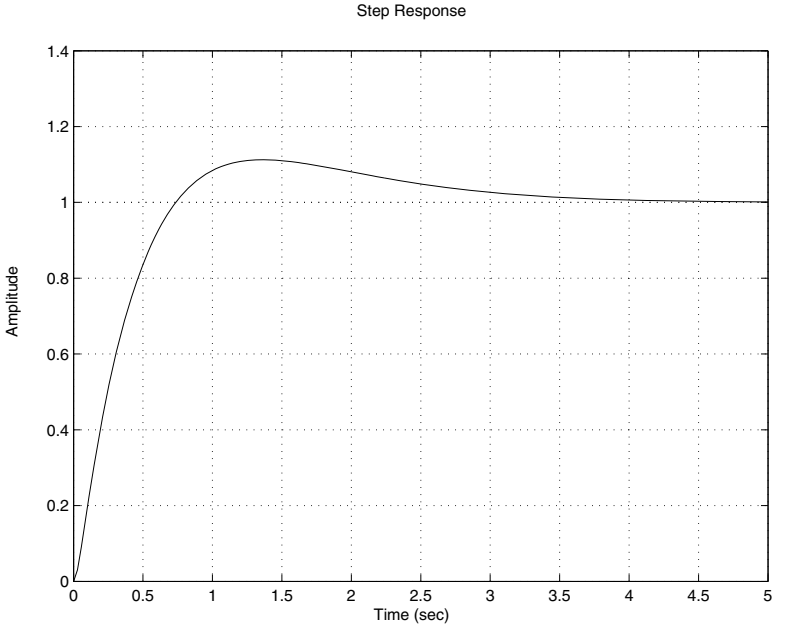


Fig. 5.30 Step response of $F(s)$

Let us now focus on the design of the PID controller using the Bode method. This controller acts on the transient and steady state regimes. The transfer function of this controller is given by:

$$C(s) = K_P \frac{K_I}{s} + K_D s = \frac{(1+\tau_n s)(1+\tau_v s)}{\tau_i s}$$

where $K_P = \frac{\tau_n + \tau_v}{\tau_i}$, $K_I = \frac{1}{\tau_i}$ and $K_D = \frac{\tau_n \tau_v}{\tau_i}$.

The open loop transfer function of the compensated system is given by:

$$T(s) = K \frac{(1+\tau_n s)(1+\tau_i s)\,(b_m s^m + \cdots + b_1 s + 1)}{s^{l+1}\,(a_n s^n + \cdots + a_1 s + 1)}$$

with $K = \frac{k}{\tau_i}$.

To design such controller we use the ideas used to design separately the PI and the PD controllers. The procedure to design such controller is based on the fact that a pole is introduced at the origin, the gain, $\bar{K}_P$ that gives the steady error and the use of the maximum phase, 90^o (introduced by the PD controller) that corresponds to the frequency when the magnitude is to $-20\ db$ ($w_m \tau_v = 10$). The following procedure can be used for the design of this controller:

1. determine the slowest pole of the system to controller except those at the origin and proceed with a pole/zero cancellation. This will help to fix, τ_n, i.e.:

$$\tau_n = \max\{\tau_1, \cdots, \tau_v\}$$

2. determine the gain $\bar{K}_P$ that gives the desired error

3. plot the Bode diagram of:

$$T(s) = \bar{K}_P \frac{(1+\tau_n s)(b_m s^m + \cdots + b_1 s + 1)}{s^{l+1}(a_n s^n + \cdots + a_1 s + 1)}$$

and determine the frequency w_m at which the magnitude is equal to -20 db. Using this frequency we determine τ_v by:

$$\tau_v = \frac{10}{w_m}$$

the phase margin of the compensated system is given:

$$\Delta\phi_c = \Delta\phi + 90$$

where $\Delta\phi$ is the phase margin of the system without the controller at the frequency w_m
If

$$\Delta\phi_c \begin{cases} < 40^o & \text{choose another controller} \\ > 50^o & \text{reduce the parameter, } \tau_D \text{ till } \Delta\phi_c = 45^o \end{cases}$$

4. compute the controller's gains using:

$$K_P = \frac{\tau_n + \tau_v}{\tau_i}$$
$$K_I = \frac{1}{\tau_i}$$
$$K_D = \frac{\tau_n \tau_v}{\tau_i}$$

5. check if the desired specifications are obtained or not

Example 5.5.4 *To show how the design of the PID controller works, let us consider the following dynamical system:*

$$G(s) = \frac{2}{(0.1s+1)(0.2s+1)(0.5s+1)}$$

A steady state error to a unit ramp equal 0.1 is needed.

This system is of type zero and has three time constant, 0.5, 0.2 *and* 0.1. *The maximum time constant is* 0.5.

Following the procedure design, we get:

1. *using the maximum time constant of the system we have:*

$$\tau_n = 0.5$$

2. *using the error specification, we get:*

$$\bar{K}_P = \frac{1}{0.1} = 10$$

3. *draw the Bode diagram of:*

$$T(s) = \frac{\bar{K}_P}{s(0.1s+1)(0.2s+1)}$$

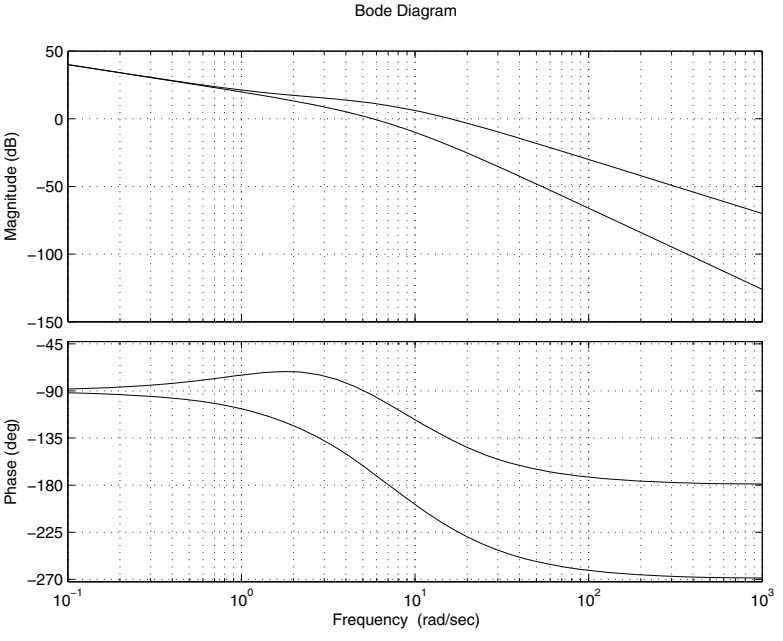


Fig. 5.31 Bode plot of $T(s)$

This diagram is illustrated by Fig. 5.31. The frequency at which the magnitude is equal to -20 *db is equal to* $w_m = 15.9$*. The phase at this frequency is equal to* -220^o*. The phase margin at this frequency is given by:*

$$\Delta\phi = 180 + \phi(w_m) + 90 = 180 - 220 + 90 = 50$$

The second parameter, τ_v *of the controller is determined by:*

$$\tau_v = \frac{10}{w_m} = 0.6289$$

4. *compute the controller's gains using:*

$$\begin{aligned} \tau_i &= \frac{2}{10} = 0.2 \\ K_P &= \frac{\tau_n + \tau_v}{\tau_i} = 5.6447 \\ K_I &= \frac{1}{\tau_i} = 5 \\ K_D &= \frac{\tau_n \tau_v}{\tau_i} = 1.5723 \end{aligned}$$

5. *The closed-loop transfer function with this controller is given:*

$$F(s) = \frac{\frac{2}{\tau_i}(\tau_v s + 1)}{0.02s^2 + 0.2s^2 + (1 + \frac{2\tau_v}{\tau_i})s + \frac{2}{\tau_i}}$$

The step response of the compensated system is represented in Fig. 5.32.

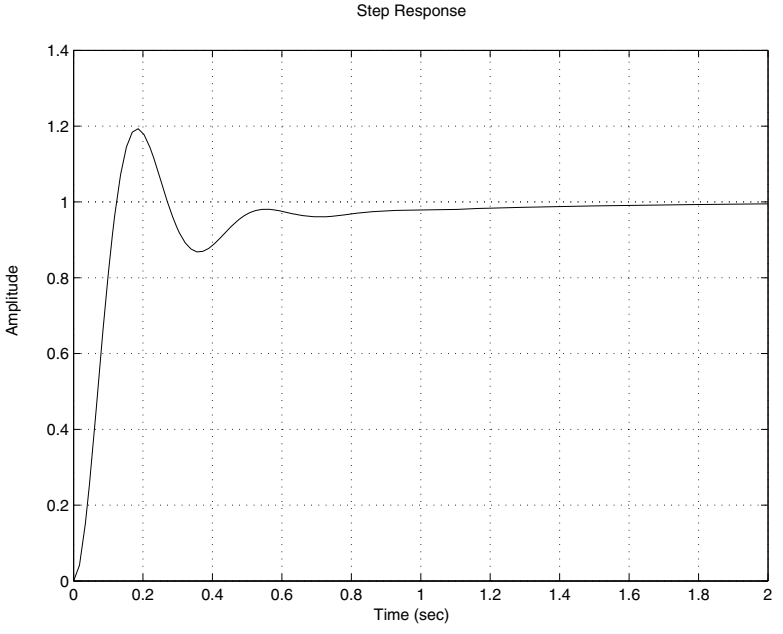


Fig. 5.32 Step response of $F(s)$

Let us now focus on the design of the phase-lead controller using the Bode method. The transfer function of this controller is given by:

$$C(s) = K_P \frac{aTs + 1}{Ts + 1}, a > 1$$

It can be shown that this controller can be deliver a maximum of phase for each value for a. The value of this maximum and the frequency at which this happens are given by:

$$\begin{aligned} w_m &= \frac{1}{T\sqrt{a}} \\ \sin(\phi_m) &= \frac{a-1}{a+1} \end{aligned}$$

The second relation gives also:

$$a = \frac{1+\sin(\phi_m)}{1-\sin(\phi_m)}$$

These relations are of great importance in the design procedure of the phase-lead controller.

The following procedure can be used for the design of this controller:

1. using the error specification, determine the gain $\bar{K}_P$ and compute the controller gain by:

$$\bar{K}_P = \frac{\bar{K}_P}{k}$$

2. plot the Bode diagram of:

$$\bar{K}_P \frac{b_m s^m + \cdots + b_1 s + 1}{s^l (a_n s^n + \cdots + a_1 s + 1)}$$

and determine the phase and gain margins of the non-compensated system. Then compute the phase margin missing. This value increased by a factor (5^o) for safety is considered as ϕ_m, then compute the parameter a by:

$$a = \frac{1+\sin(\phi_m)}{1-\sin(\phi_m)}$$

3. determine the frequency, w_m for which the magnitude of the non-compensated system is equal to $-20\log\sqrt{a}$ and consider it as the crossover of the compensated system. The parameter T of the controller is determined using:

$$T = \frac{1}{w_m\sqrt{a}}$$

4. check if the desired specifications are obtained or not

Example 5.5.5 *Let us consider the following dynamical system:*

$$G(s) = \frac{5(0.125s+1)}{s(2s+1)(0.1s+1)}$$

Our objective in this example is to design a phase-lead controller satisfies the following specifications:

1. *stable system*
2. *steady state error for a ramp input equal to 0.1*

3. *phase margin greater than* 40^o
4. *gain margin greater than* 6 *db*

The design of the phase-lead controller is brought to the determination of the parameters a and T. To accomplish this, we follow the previous procedure.

1. *since the system is of type one, therefore the error for a ramp input is given by:*

$$e(\infty) = \frac{1}{\bar{K}_P}$$

which gives in turn:

$$\bar{K}_P = 10$$

which gives:

$$K_P = \frac{\bar{K}_P}{k} = 2$$

2. *with this gain, the open loop transfer function of the system becomes:*

$$T(s) = \frac{10}{s(2s+1)(0.1s+1)}$$

The Bode diagram of this system is given by Fig. 5.33
From this diagram we conclude that the system with a proportional controller has a phase margin equal to 15.67^o *and a gain margin equal to* ∞ *db. To get our desired phase margin we need to add* 24.33^o*. If we take a* 5^o *safety, the controller should add a phase,* ϕ_m *equal to* 29.33^o*. This gives:*

$$a = \frac{1+\sin(29.33)}{1-\sin(29.33)} = 2.9201$$

3. *with this value of a we have:*

$$-20log\sqrt{a} = -4.6540$$

From 5.33, we remark that the magnitude curve takes -4.6540 *at the frequency* $w_m = 2.93\ rd/s$*. This gives:*

$$T = \frac{1}{w_m\sqrt{a}} = 0.1997$$

The controller is then given by:

$$C(s) = K_P\frac{aTs+1}{Ts+1} = 2\frac{0.5832s+1}{0.1997s+1}$$

The open loop transfer function of the compensated system is given by:

$$T(s) = 10\frac{0.5832s+1}{s(2s+1)(0.1s+1)(0.1997s+1)}$$

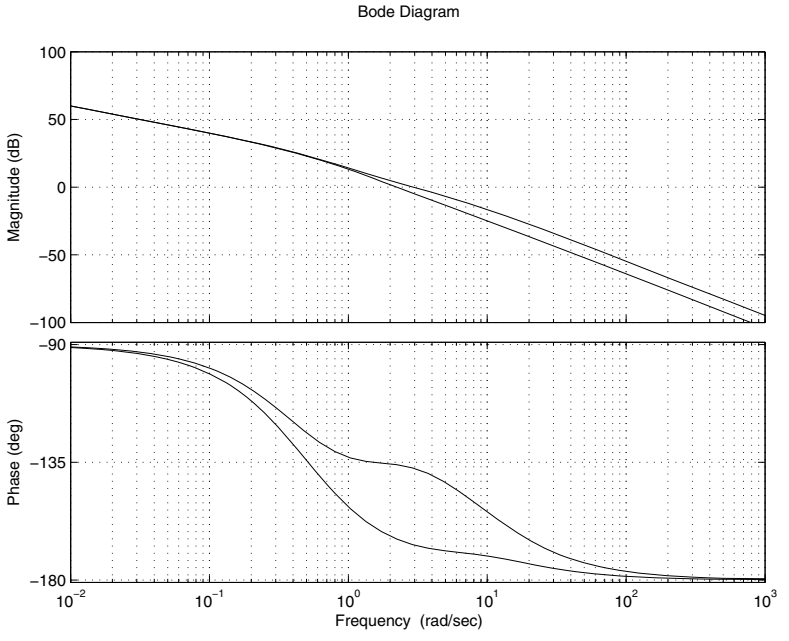


Fig. 5.33 Bode plot of $T(s)$

4. *with controller we get* 42.8^o *and* ∞ *db as phase margin and gain margin respectively.*
   The closed-loop transfer function is given by:

$$F(s) = \frac{kK_P\left(0.125aTs^2 + (0.125 + aT)s + 1\right)}{b_4s^4 + b_3s^3 + b_2s^2 + b_1s + b_0}$$

   with $k = 5$, $b_4 = 0.2T$, $b_3 = 0.2 + 2.1T$, $b_2 = 2.1 + T + 0.125aTkK_P$, $b_1 = 1 + akK_P(0.125 + aT$ *and* $b_0 = kK_P$.
   The behavior of the closed-loop dynamics with the computed controller is illustrated in Fig. 5.34
   The settling time at 5 % is equal to 1.68 *s which is acceptable and the error for a step input is equal to zero while the overshot is about 30 %.*

Let us now focus on the design of the phase-lag controller using the Bode method. The transfer function of this controller is given by:

$$C(s) = K_P\frac{aTs + 1}{Ts + 1}, a < 1$$

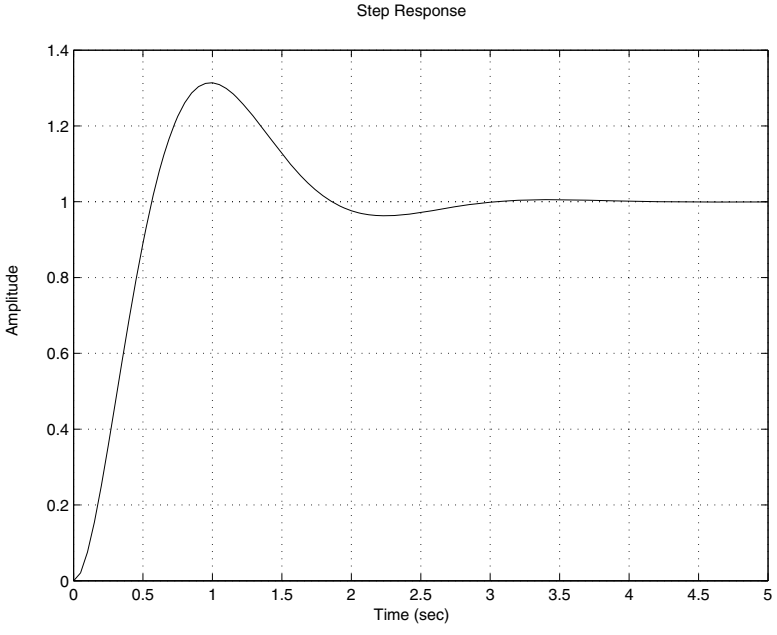


Fig. 5.34 Step response of $F(s)$

The following procedure can be used for the design of this controller:

1. using the error specification, determine the gain $\bar{K}_P$ and compute the controller gain by:

$$\bar{K}_P = \frac{\bar{K}_P}{k}$$

2. plot the Bode diagram of:

$$\bar{K}_P \frac{b_m s^m + \cdots + b_1 s + 1}{s^l (a_n s^n + \cdots + a_1 s + 1)}$$

and determine the frequency, w_m of the non-compensated system at which we have the desired phase margin. Then compute of how much decibels, m to bring the magnitude to 0 db at w_m.The parameter a of the controller is given by:

$$a = 10^{\frac{m}{20}}$$

3. To get an appreciable change the phase curve, we need to choose, the parameter, T as follows:

$$T = \frac{10}{a w_m}$$

4. check if the desired specifications are obtained or not

Example 5.5.6 *Let us consider the following dynamical system:*

$$G(s) = \frac{2}{s(0.1s+1)(0.05s+1)}$$

Our objective in this example is to design a phase-lag controller satisfies the following specifications:

1. *stable system*
2. *steady state error for a ramp input equal to 0.1*
3. *phase margin greater than* 40^o
4. *gain margin greater than* 4 *db*

The design of the phase-lag controller is brought to the determination of the parameters a and T. To accomplish this, we follow the previous procedure.

1. *the system to be controlled is of type one. The steady error to a unit ramp as input is given by:*

$$e(\infty) = \frac{1}{\bar{K}_P}$$

which implies:

$$\bar{K}_P = 10$$

From this we conclude that the gain of the controller is $K_P = 5$.

2. *with this gain, the open loop transfer function of the system becomes:*

$$T(s) = \frac{10}{s(0.1s+1)(0.05s+1)}$$

The Bode diagram of this system is given by Fig. 5.35
From this figure, we conclude, that at $w_m = 5.59\ rd/s$, *the phase margin is equal to* 45^o. *At this frequency the magnitude is equal to* 3.52 *db. Using this, the parameter, a is given bY:*

$$a = 10^{\frac{-3.52}{20}} = 0.6668$$

Remark 5.5.3 *The fact that we consider* -3.52 *db means that we want the controller to introduce this amplitude at this frequency.*

3. *the choice of T is done by placing the frequency* $\frac{1}{aT}$ *at a decade from* $w_m = 5.59\ rd/s$, *i.e.:*

$$w_m = \frac{10}{aT}$$

which implies $T = 2.6828$.
The transfer function of our phase-lag controller is given by:

$$C(s) = K_P \frac{aTs+1}{Ts+1}$$

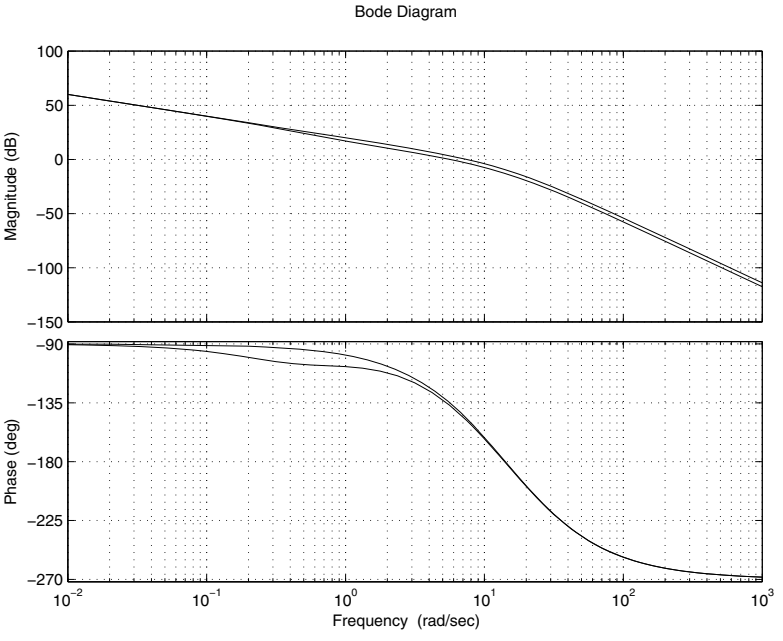


Fig. 5.35 Bode plot of $T(s)$

with $K_P = 5$.
With this controller we get:

$$\begin{aligned} \Delta\phi &= 43.13^o \\ \Delta G &= 4.37\ db \end{aligned}$$

The closed-loop transfer function is given by:

$$F(s) = \frac{kK_P\,(aT\,s+1)}{0.005Ts^4 + (0.005+0.15T)s^3 + (0.15+T)s^2 + (1+kK_PaT)s + kK_P}$$

with $k = 2$.
The behavior of the closed-loop dynamics with the computed controller is illustrated in Fig. 5.36
The settling time at 5 % is equal to 0.78 *s which is acceptable and the error for a step input is equal to zero while the overshot is about 27 %.*

Let us now focus on the design of the phase lead-lag controller using the Bode method. The transfer function of the controller is given by:

$$C(s) = K_P \frac{a_1T_1s+1}{T_1s+1}\frac{a_2T_2s+1}{T_2s+1}, a_1 > 1, a_2 < 1$$

The following procedure can be used for the design of this controller:

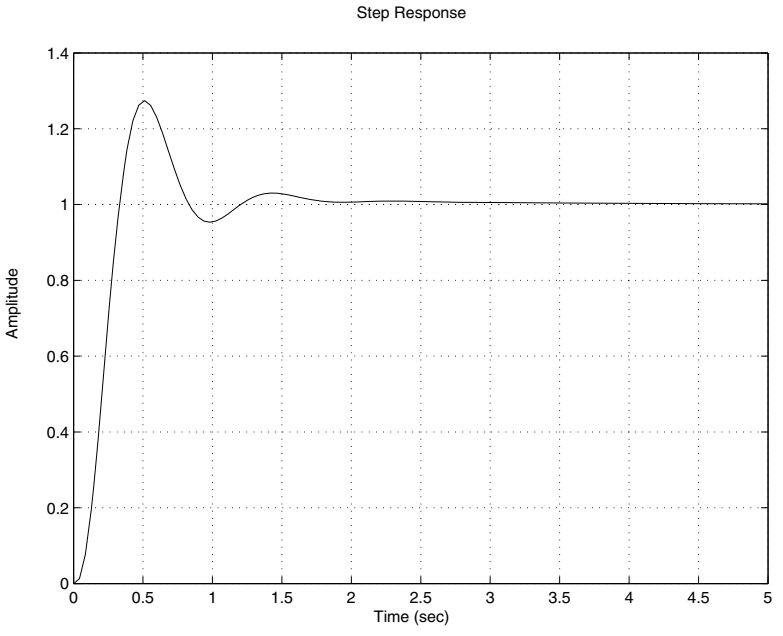


Fig. 5.36 Step response of $F(s)$

1. using the error specification, determine the gain $\bar{K}_P$ and compute the controller gain by:

$$\bar{K}_P = \frac{\bar{K}_P}{k}$$

2. draw the Bode diagram of:

$$\bar{K}_P \frac{b_m s^m + \cdots + b_1 s + 1}{s^l (a_n s^n + \cdots + a_1 s + 1)}$$

   and determine the phase margin of the non-compensated system

3. determine the phase-lead controller's parameters, a_1 and T_1
4. determine the phase-lag controller's parameters, a_2 and T_2
5. check if the desired specifications are obtained or not

Example 5.5.7 *To show how to design a phase lead-lag controller let us consider the following dynamical system:*

$$G(s) = \frac{4(0.125s + 1)}{s(0.1s + 1)(0.2s + 1)}$$

As specifications we search to get the following ones:

1. stable system

2. *steady state error to a unit ramp equal to 0.05*
3. *a phase margin greater than* 40^o
4. *a gain margin greater than* 8 *db*

To design the phase lead-lag controller let us follow the steps of the previous procedure.

1. *to get the desired error a gain* $\bar{K}_P$ *equal to 20, which corresponds to* $K_P = 5$.
2. *The transfer function of the open loop of the non compensated system with this gain is given by:*

$$T(s) = \frac{20(0.125s + 1)}{s(0.1s + 1)(0.2s + 1)}$$

The Bode diagram of this system is given by Fig. 5.37

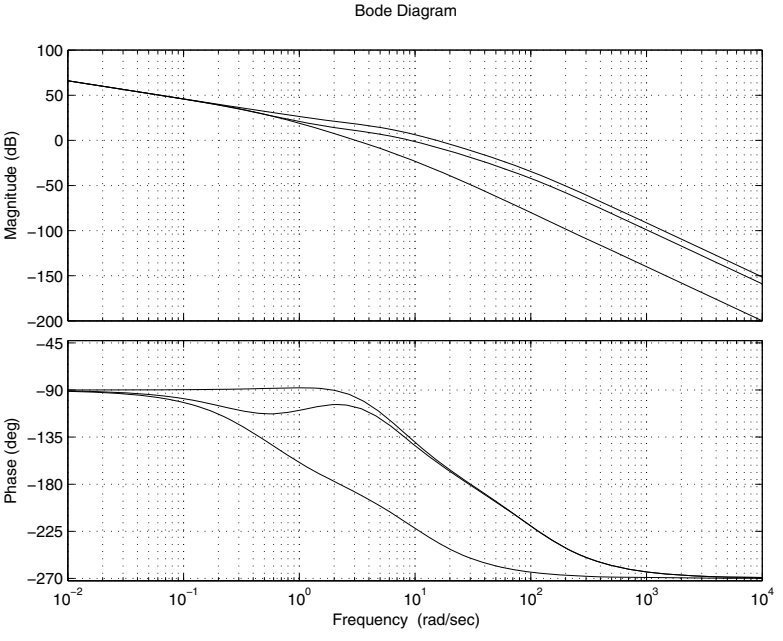


Fig. 5.37 Bode plot of $T(s)$

With this proportional controller the system has:

$$\Delta\phi = 32.7^o$$
$$\Delta G = \infty\ db$$

3. *to design the phase-lead controller can be done following the previous procedure for this purpose. Notice that to get the desired phase margin, the*

phase-lead controller must bring a phase of $45^o - 32.7^o = 12.3^o$. *Using this, we have:*

$$a_1 = \frac{1+\sin(12.3)}{1-\sin(12.3)} = 1.5414$$

Using the value of a_1, *we get:*

$$-20\log\sqrt{a_1} = -1.8791$$

Now if we refer to the Fig. 5.37, the magnitude will have -1.8791 *at the frequency* $w_m = 11.4\ rd/s$. *This implies:*

$$T_1 = \frac{1}{w_m\sqrt{a_1}} = 0.0707$$

The transfer function of the phase lead controller is given:

$$C(s) = \frac{0.4231s+1}{0.0707s+1}$$

The open loop transfer function of the system with controller is given by:

$$T(s) = 20\frac{a_1T_1s+1}{s(0.2s+1)(0.01s+1)(T_1s+1)}$$

4. *the system compensated with the phase lead controller has:*

$$\Delta\phi = 10.9624^o$$
$$\Delta G = \infty\ db$$

To get a phase margin equal to 45^o *and if we report to the Fig. 5.37, we have this at the frequency* $w_m = 10\ rd/s$. *Also at this frequency, the magnitude is equal to 1.76 db. using this we get the parameter* a_2 *for the phase lag controller:*

$$a_2 = 10^{\frac{-1.76}{20}} = 0.8166$$

The choice of T_2 *is given by:*

$$T_2 = \frac{10}{w_m a_2} = \frac{10}{9.07\times 0.4154} = 2.6542$$

The transfer function of the phase lead controller is given:

$$C(s) = \frac{1.1026s+1}{2.6542s+1}$$

5. *The open loop transfer function of the compensated system is given by:*

$$T(s) = 20\frac{(a_1T_1s+1)(a_2T_2s+1)(0.125s+1)}{s(0.2s+1)(0.1s+1)(T_1s+1)(T_2s+1)}$$

The Bode diagram of this transfer function is reported in Fig. 5.37 and from which we get:

$$\Delta\phi = 44.1^o$$
$$\Delta G = \infty\ db$$

The closed-loop transfer function of the compensated system

$$F(s) = kK_P \frac{\alpha_3 s^3 + \alpha_2 s^2 + \alpha_1 s + \alpha_0}{b_5 s^5 + b_4 s^4 + b_3 s^3 + b_2 s^2 + b_1 s + b_0}$$

with $\alpha_3 = 0.125 a_1 a_2 T_1 T_2$, $\alpha_2 = 0.125(a_1 T_1 + a_2 T_2) + a_1 a_2 T_1 T_2$, $\alpha_1 = 0.125 + a_1 T_1 + a_2 T_2$ *and* $\alpha_0 = 1$; $b_5 = 0.02 T_1 T_2$, $b_4 = 0.3 T_1 T_2 + 0.02(T_1 + T_2)$, $b_3 = 0.02 + T_1 T_2 + 0.3(T_1 + T_2) + 0.125 k K_P a_1 a_2 T_1 T_2$, $b_2 = 0.3 + T_1 + T_2 + kK_P(0.125(a_1 T_1 + a_2 T_2) + a_1 a_2 T_1 T_2)$, $b_1 = 1 + kK_P(0.125 + a_1 T_1 + a_2 T_2)$ *and* $b_0 = kK_P$

5.6 Case Study

The goal of this section is to make the design of different controllers for our dc motor kit using the developed methods and show the reader how things apply in practice. It was shown that the model of this system is given by:

$$G(s) = \frac{K_m}{s(\tau_m s + 1)}$$

where $K_m = 48.5$ is the gain and $\tau_m = 0.060\ s$ is the time constant.

Our objective is to design the proportional controller, the proportional and integral controller, the proportional and derivative controller, the proportional, integral and derivative controller, the phase lead controller, the phase lag controller and the phase lead-lag controller using the three methods and implement them in real-time on our dc motor kit.

Regarding the specifications, we will not fix them but during the design of each controller we will try to get the best specifications that may offer each controller.

5.6.1 Proportional Controller

Let us first of all consider the design of the proportional controller. This controller is assumed to have the following transfer function:

$$C(s) = K_P$$

where K_P is the gain to be determined.

For the empirical methods, it is clear that the time domain methods will not apply since the transfer function of the system has a pole at the origin and will never provide a step response with periodic oscillations.

To compute the gain of the controller, we notice that we have to move up the magnitude by 27.27 db, from Fig. 5.38, which gives a gain equal to:

$$\bar{K}_P = 10^{\frac{27.27}{20}} = 22.9087$$

The gain of the controller is given by:

$$K_P = \frac{\bar{K}_P}{K_m} = \frac{22.9087}{48.5} = 0.4723$$

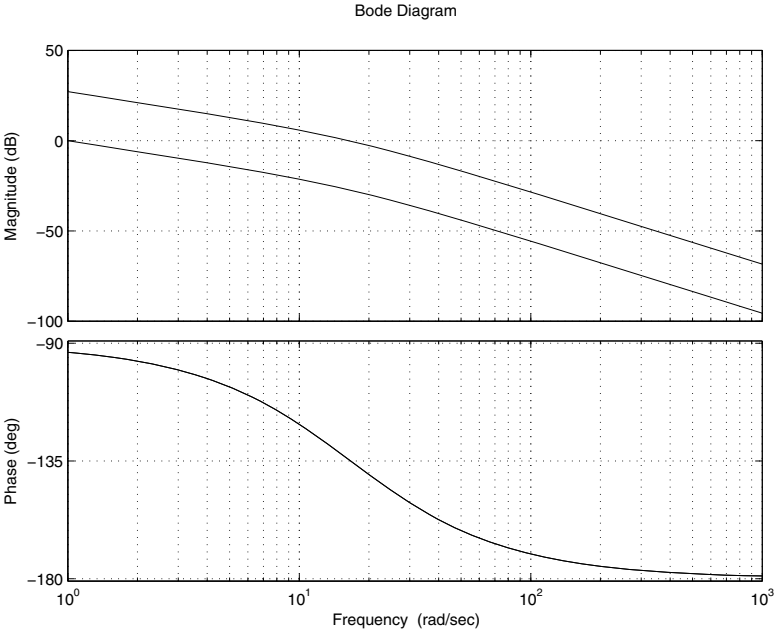


Fig. 5.38 Bode plot of $T(s)\frac{K}{s(\tau_m s+1)}$, with $K = 1$, and $K = K_m K_P$

We can check that with this gain, the closed loop system has a phase margin close to 45^o and gain margin equal to infinity. This responds to the general specifications.

For the root locus method, we know that the proportional controller is unable to change the shape of the root locus and the only thing that we can do is to select an appropriate gain for the controller to get best performances. The root locus of the system is given by Fig. 5.39. From this figure with a gain $K = 8.35$ we get a damping ratio equal to 0.707 and the complex poles are $s_{1,2} = -8.33 \pm 8.35j$. This gives a settling time at 5 % equal to 0.3601 s. The controller's gain in this case is:

$$K_P = \frac{K}{K_m} = \frac{8.35}{48.5} = 0.1722$$

The design of the proportional controller with the Bode method will give the same result as we did for the empirical method. It is important to notice that the methods (empirical method and Bode method and root locus method) give different gains. The step response with the two controllers is plotted in Fig. 5.40. The two methods (empirical and bode) give high value for the controller's gain, which corresponds to a smaller damping ratio and therefore an important overshoot.

As a comparative study of these methods we have the results of the Tab. 5.4. The error for a step input in all the case is equal to zero.

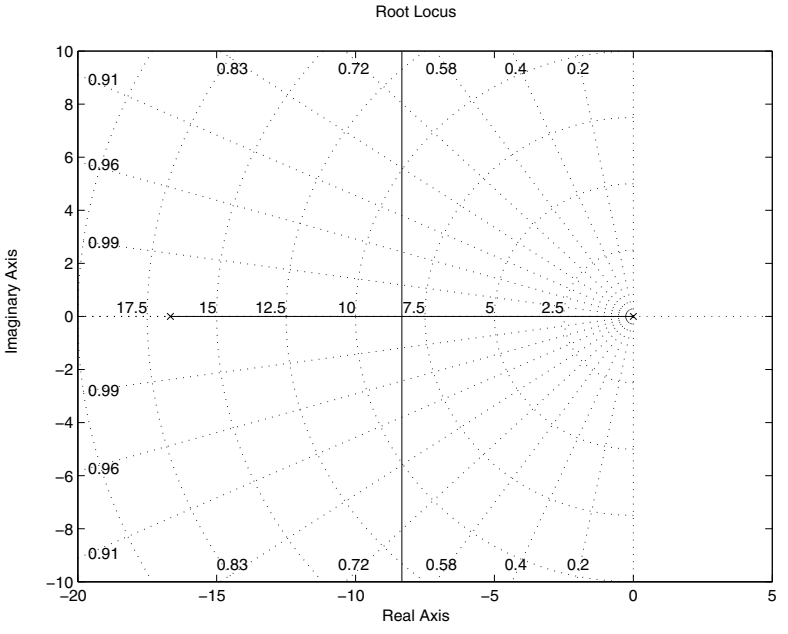


Fig. 5.39 Root locus of $T(s) = \frac{1}{s(\tau_m s+1)}$

Table 5.4 Comparative study of the design of P controller

Method	K_P	t_s	Overshoot	$\Delta\phi$	ΔG
Empirical	0.4723	0.3 s	23 %	45.6^o	∞
Root locus	0.1722	0.3 s	4 %	65.5^o	∞
Bode	0.4723	0.3 s	23 %	45.6^o	∞

5.6.2 Proportional and Integral Controller

Let us now focus on the design of the PI controller using the previous methods that gives the best performances for our dc motor kit. As for the proportional controller, the PI controller can not be designed using the time domain empirical method. While the frequency method can be used. It is important in this case that we can not use our procedure since we can not cancel the pole at the origin but placing the zero at -2 will give good performances. Using this, we get:

$$\begin{aligned} K_P &= 0.0497, \text{ same computations as before} \\ K_I &= 0.0994 \end{aligned}$$

The bode diagram of the open-loop transfer function of the compensated system is illustrated at the Fig. 5.41.

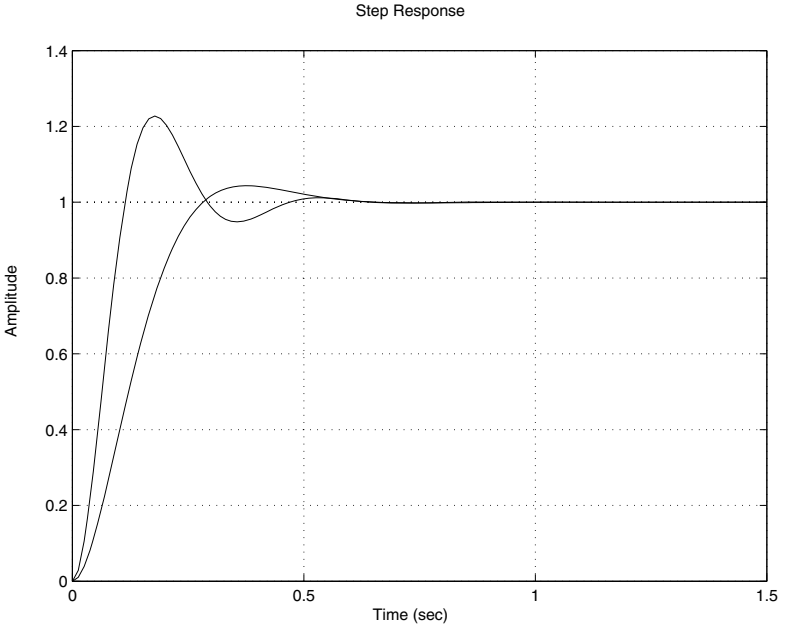


Fig. 5.40 Step response of $F(s) = \frac{K_m K_P}{\tau_m s^2 + s + K_m K_P}$

With this controller we get a phase margin equal to 45^o but the gain margin is close to zero.

Remark 5.6.1 *It is important to notice the approach used here to design the PI controller is a heuristic that I propose to overcome the problem with the previous procedure.*

If we place the zero at -3, the gain that gives the dominant poles $s_{1,2} = -5.23 \pm 5.95j$ is $\bar{K}_P = 22.9$, which gives $K_P = \frac{22.9}{48.5} = 0.4722$. From this we conclude that $K_I = K_P z = 1.4165$

The Bode method will give the same results as for the Zigeler-Nichols method and we don't repeat the computation again.

The closed-loop transfer function with the PI controller is given by:

$$F(s) = \frac{K_m K_P s + K_m K_I}{\tau_m s^3 + s^2 + K_m K_P s + K_m K_I}$$

The behavior of the system with this controller for a step input is illustrated at the Fig. 5.43. As it can be seen that the two methods give two controllers that are almost identical and the step response are also almost identical. The settling time for the frequency methods is higher than that the one obtained by the root locus method.

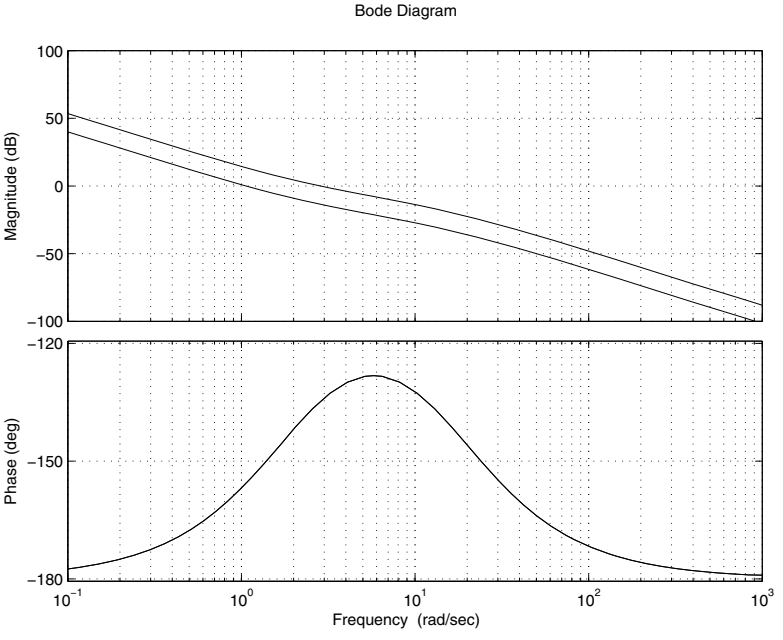


Fig. 5.41 Bode plot of $T(s)\frac{K(0.5s+1)}{s^2(\tau_m s+1)}$, with $K = 1$, and $K = K_m K_P$

5.6.3 Proportional and Derivative Controller

The PD controller can not be designed by any of the proposed Ziegler-Nichols methods. The only methods we can used for this controller are the root locus method and the Bode method. Let us firstly design this controller by the first method. For this controller, we can proceed by pole/zero cancellation or place the zero at the right of the pole of the system. The first case is easy and gives a first order while the second one gives an interesting case. It is important to notice that the damping ratio in this case will be close 1. This doesn't imply that there is no overshoot due the presence of the zero. We will design two cases.

Let the zero be at the position -30, placed at the left of the system pole. The first case case gives the dominant poles $s_{1,2} = -16 \pm 14.2j$ which corresponds to the gain $\bar{K}_P = 0.915$. This gives the gain $K_D = \frac{0.915}{48.5} = 0.0189$. The second gain is $K_P = K_D z = 0.0189 \times 30 = 0.5660$.

The second case case gives the dominant poles $s_{1,2} = -43.2 \pm 15j$ which corresponds to the gain $\bar{K}_P = 4.19$. This gives the gain $K_D = \frac{4.19}{48.5} = 0.0864$. The second gain is $K_P = K_D z = 0.0864 \times 30 = 2.5918$. As it will seen from the Fig. 5.45 this case will gives good performances at least in simulation.

For the design of the PD controller, let us assume that we want to assure a steady state error for a unit ramp input equal to 0.008. This corresponds to a gain $\bar{K}_P = 125$. The Bode diagram of $T(s) = \frac{\bar{K}_P}{s(\tau_m s+1)}$ is represented at the Fig. 5.44. The magnitude

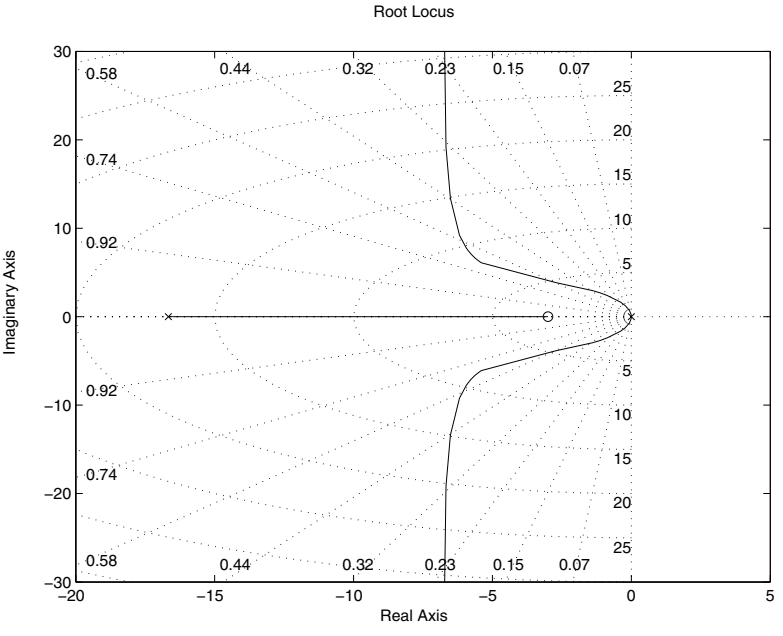


Fig. 5.42 Root locus of $T(s) = \frac{0.25s+1}{s^2(\tau_m s+1)}$

is equal to -20 db at the frequency $w_m = 144\ rd/s$. The phase at this frequency is equal to -173^o which corresponds to a phase margin equal to 7^o and it is far from the desired phase margin.

The parameter τ_D is determined by:

$$\tau_D = \frac{10}{200} = 0.05$$

The parameters of the PD controller are given by:

$$\begin{aligned} K_P &= \frac{125}{K_m} = 2.5773 \\ K_D &= K_P \tau_D = 2.5773 \times 0.05 = 0.1289 \end{aligned}$$

The phase margin of the compensated system is almost equal to 90^o.

The closed-loop transfer function with the PD controller is given by:

$$F(s) = \frac{K_m K_D s + K_m K_P}{\tau_m s^2 + (1 + K_m K_D)s + K_m K_P}$$

The behavior of the system with this controller for a step input is illustrated at the Fig. 5.45. As it can be seen that the two methods give two controllers that are different and the step response are also different. The settling time for the frequency methods is higher than that the one obtained by the root locus method.

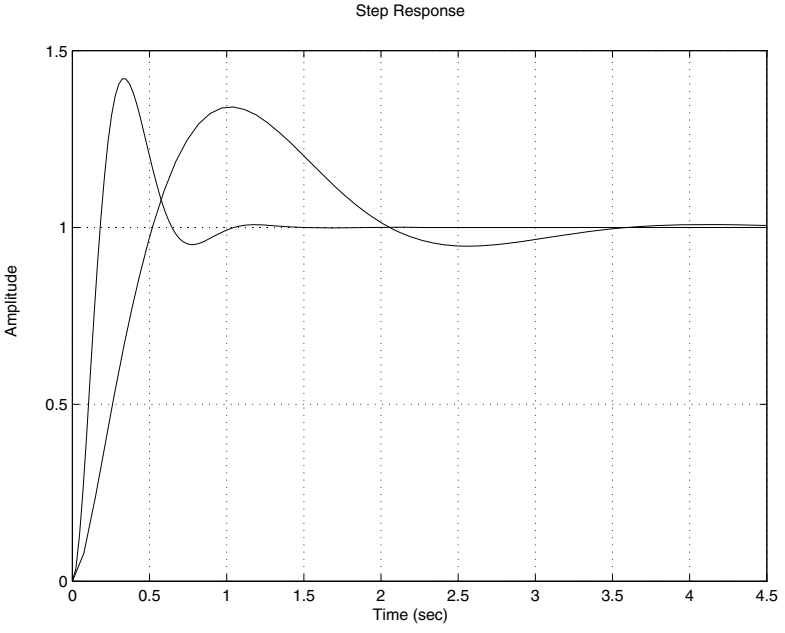


Fig. 5.43 Step of $F(s)$ with two controllers for two design methods

5.6.4 Proportional Integral and Derivative Controller

None of the heuristic methods proposed by Ziegler-Nichols can be used to design the PID controller. In the rest of this subsection we focus on the design of this controller using the root locus and Bode methods. The procedures we proposed previous can not be used here and we have to use another heuristic methods for this system.

For the root locus method, since the system has only one pole non equal to zero. The case that consists of placing the zeros between the two poles of the system is interesting since it can give short settling time.

If we place the two zeros of the controllers respectively at -13 and -15, the root locus of the system in this case is represented by Fig. 5.46.

From the root locus we see that for the gain $\bar{K}_P = 1.43$, the dominant poles are:

$$s_{1,2} = -11.4 \pm 11.5j$$

If we refer to the procedure used for the design of the PID controller and the expression of the controller, we have:

$$a_1 = 13$$
$$a_2 = 15$$

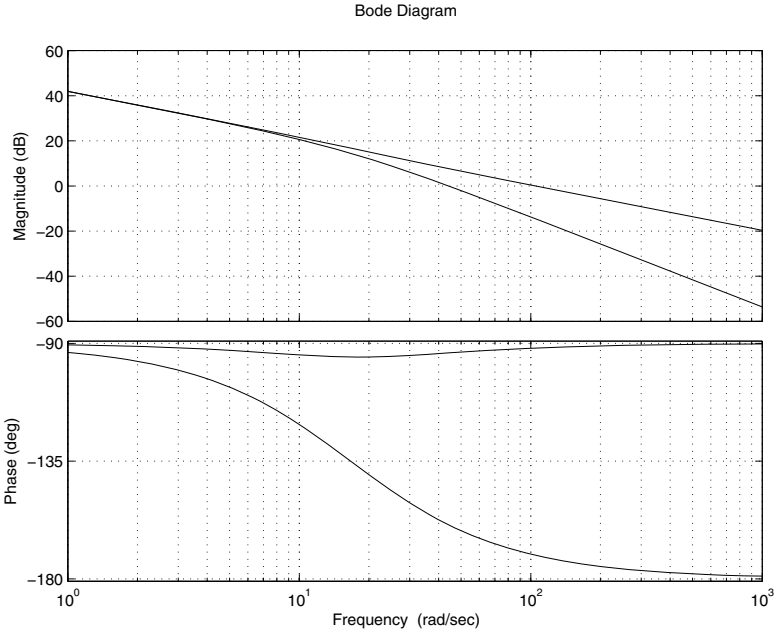


Fig. 5.44 Bode plot of $T(s)$ (compensated and non compensated system

From this we have:

$$\begin{aligned} K_D &= \frac{1.43}{K_m} = 0.0295 \\ K_P &= K_D(a_1 + a_2) = 0.0295(13 + 15) = 0.8260 \\ K_I &= K_D a_1 a_2 = 5.7525 \end{aligned}$$

For the design of the PID using the Bode method, we will use the same idea of placing the zeros of the controller as we did for the root locus method. Also we would like to have a steady state error to a unit ramp equal to 0.01. To get such error a gain equal to $\bar{K}_P = 100$ is necessary for this purpose.

Now, if we place the two zeros of the controller respectively at -12 and -15, i.e.:

$$\begin{aligned} \tau_n &= \frac{1}{15} = 0.0667 \\ \tau_v &= \frac{1}{12} = 0.0833 \end{aligned}$$

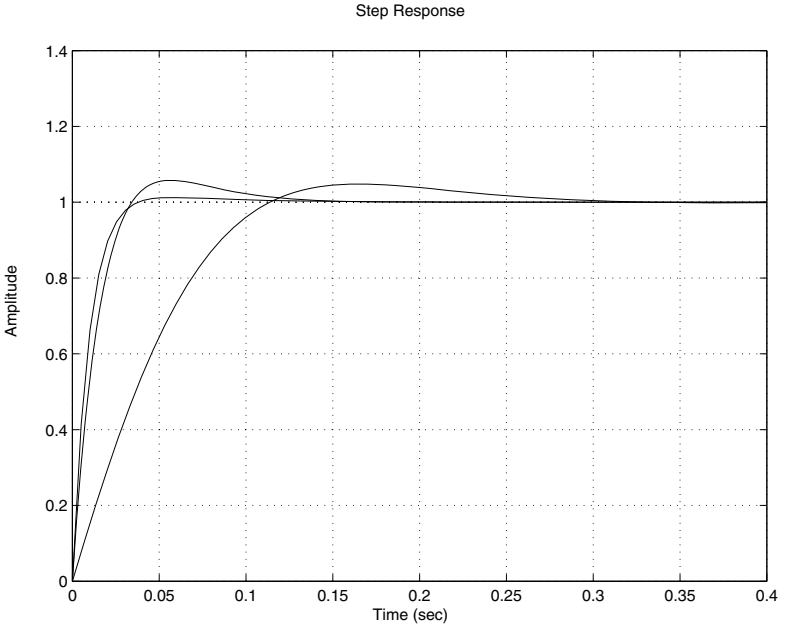


Fig. 5.45 Step of $F(s)$ with two controllers for two design methods

Using these data, we get:

$$\begin{aligned}
\tau_i &= \frac{K_m}{\bar{K}_P} = \frac{48.5}{100} = 0.4850 \\
K_P &= \frac{\tau_n + \tau_v}{\tau_i} = \frac{0.0667 + 0.0833}{0.4850} = 0.3093 \\
K_I &= \frac{1}{\tau_i} = \frac{1}{0.4850} = 2.0619 \\
K_D &= \frac{\tau_n \tau_v}{\tau_i} = \frac{0.0667 \times 0.0833}{0.4850} = 0.0115
\end{aligned}$$

The Bode diagram of the compensated system is represented at the Fig. 5.47. From this figure we conclude that the phase margin is equal 48^o.

The closed loop transfer function of the compensated system is given by:

$$F(s) = \frac{K(K_D s^2 + K_P s + K_I)}{\tau_m s^3 + (1 + K_m K_D)s^2 + K_m K_P s + K_m K_I}$$

The step responses with the two controllers is illustrated by Fig. 5.48

5.6.5 *Phase Lead Controller*

Firstly, it is important to mention that this controller can not be designed with the empirical methods. The two other methods are still valid for the design of this con-

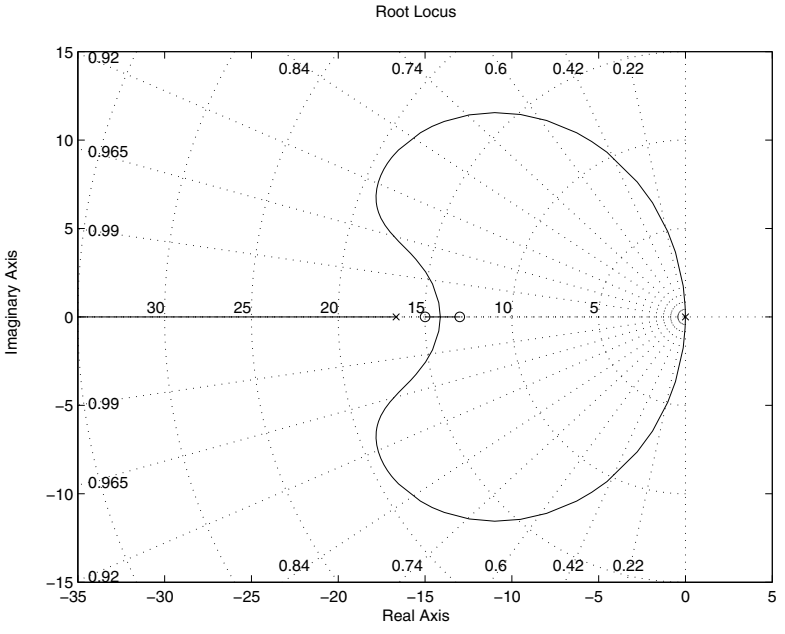


Fig. 5.46 Root locus of $T(s) = \frac{(\frac{1}{13}s+1)(\frac{1}{15}s+1)}{s^2(\tau_m s+1)}$,

troller. Let us firstly focus on the design the phase lead-controller using the root locus method. It is important to notice that the best settling time at 5 % with a proportional controller is about 0.36 s. With the phase lead controller we want to improve this time. Let the desired pole dominant with positive imaginary part be $s_d = -11.3 + 11.3j$ which corresponds to a settling time equal to 0.2655 s and an overshoot equal to 5 %. The phase of the system without the controller is given by:

$$\arg\left(\frac{48.5/0.06}{s_d(s_d + 16.6667)}\right) = 0 - 90 - 64.9830 = -153.9931$$

The phase lead controller must increase the phase with $180 - 153.9931 = 26.0069$ This implies:

$$\beta - \alpha = 26.0069$$

If we place the zero at -15, this implies that $\beta = 72.17^o$ and the pole at -20 gives an angle of 52.89^o. This gives a contribution of 19.27^o by the controller and which close to the desired one.

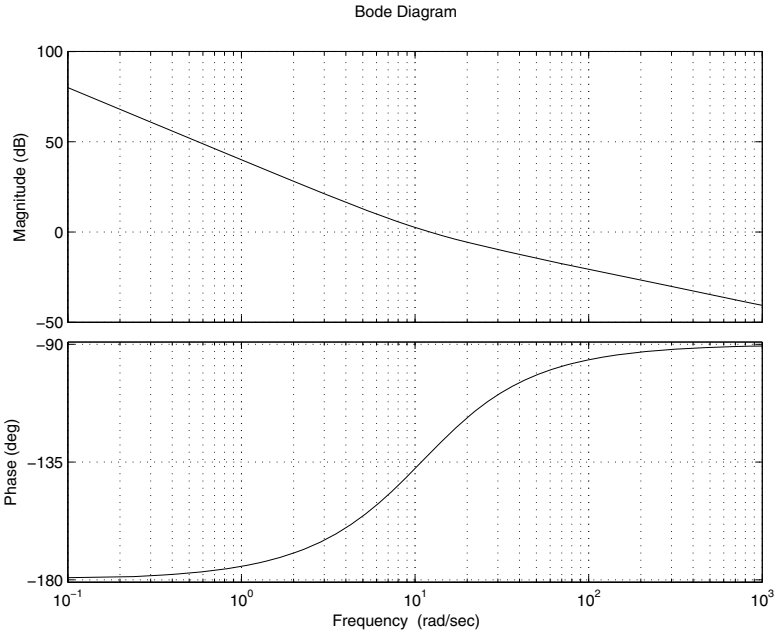


Fig. 5.47 Bode plot of $T(s) = \frac{100(\frac{1}{12}s+1)(\frac{1}{15}s+1)}{s^2(\tau_m s+1)}$

From this, we have:

$$\frac{1}{T} = 20$$
$$\frac{1}{aT} = 15$$

this gives $T = 0.05$ and $a = 1.3333$.

The root locus of the system with the phase lead controller is presented in the Fig. 5.49

The gain that gives the dominant poles is $\bar{K}_P = 10.8$, which gives a gain $K_P = 0.2227$ for the phase lead controller.

The closed-loop transfer function with the controller is given by:

$$F(s) = \frac{K_m K_P(aTs+1)}{T\tau_m s^3 + (\tau_m + T)s^2 + (1 + aTK_m K_P)s + K_m K_P}$$

The behavior of the step response of the system with this controller is illustrated in Fig. 5.51.

Using the Bode method, we design a controller that provides the following specifications:

1. stable system
2. steady state error to a unit ramp is less than 0.01

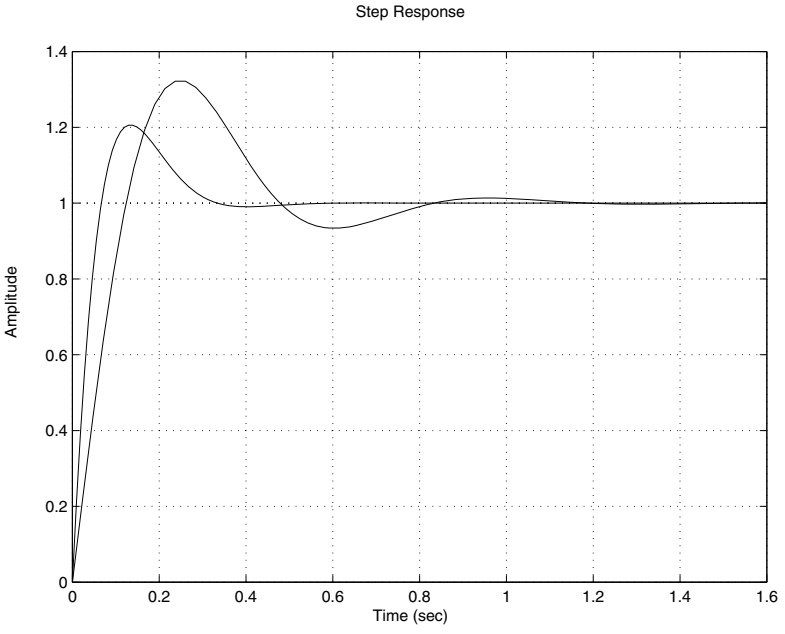


Fig. 5.48 Step response of $F(s)$ with the two controllers

3. phase margin greater than 40^o
4. gain margin greater than 8 db

Using the error specification, a gain $\bar{K}_P$ equal to 100 is needed. This gives a gain $K_P = 2.0619$ for the phase lead controller. The Bode diagram of the open loop transfer of the system with this gain is illustrated in Fig. 5.50. From this figure we have:

$$\begin{aligned} \Delta\phi &= 23.1^o \\ \Delta G &= \infty \end{aligned}$$

For the design of phase lead controller notice that this controller should bring $45^o - 23.1^o = 22.9^o$, which gives:

$$a = \frac{1 + \sin(22.9)}{1 - \sin(22.9)} = 2.2740.$$

Using this values the magnitude will take the value $-20log(\sqrt{a}) = -3.5679$ at the frequency $w_m = 48.9\ rd/s$. This implies:

$$T = \frac{1}{w_m \sqrt{a}} = 0.0136$$

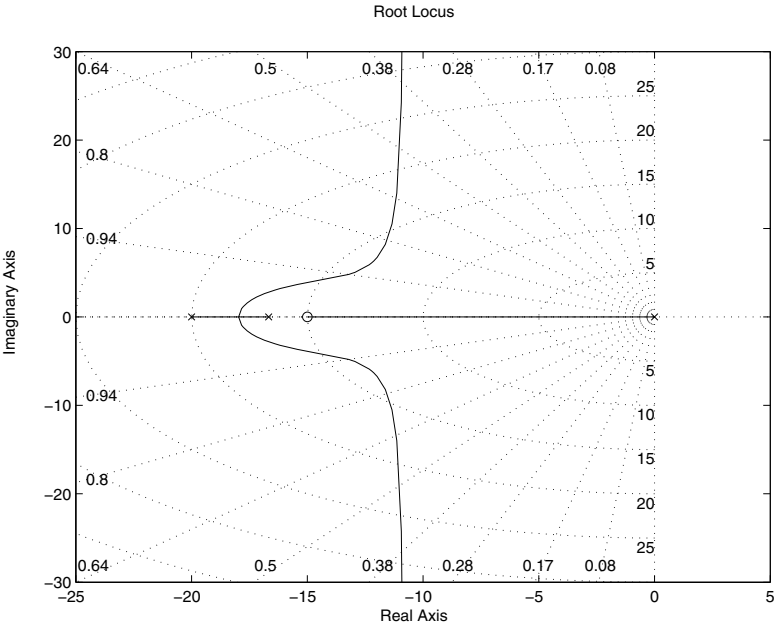


Fig. 5.49 Root locus of $T(s) = \frac{aTs+1}{s(\tau_m s+1)(Ts+1)}$

The phase lead controller is then given by the following transfer function:

$$C_1(s) = \frac{aTs+1}{Ts+1}$$

With this controller, the compensated system has:

$$\Delta\phi = 41.8^o$$
$$\Delta G = \infty$$

The behavior of the system with this controller for a step input is illustrated at the Fig. 5.51. As it can be seen that the two methods give two controllers that are almost identical and the step response are also almost identical.

5.6.6 Phase Lag Controller

As it was the case for the phase lead controller, the empirical methods can not help in the design of the phase lag controller. Here we will design this controller using the two other methods. For the root locus technique, we will assume that we want the following specifications:

1. stable system
2. a steady state error to a unit ramp input equal to 0.01

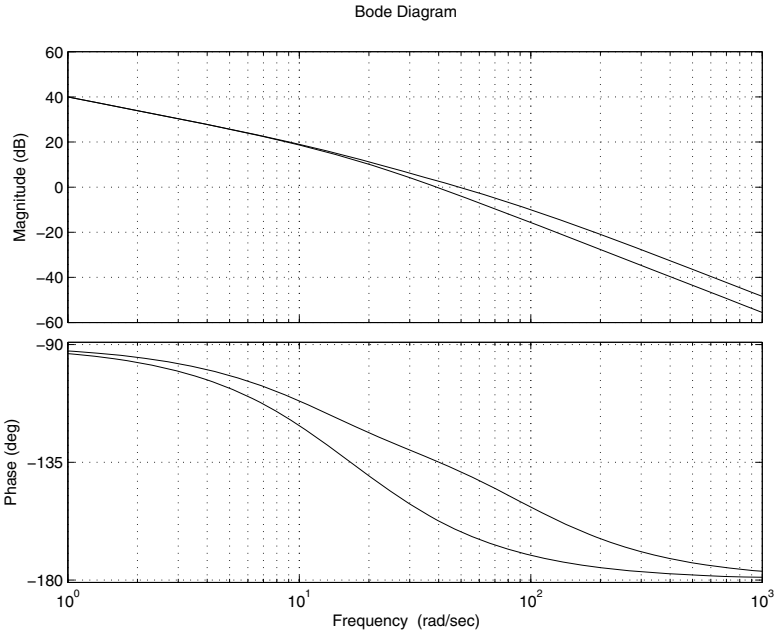


Fig. 5.50 Bode plot of $T(s)\frac{100}{s(\tau_m s+1)}$

3. an overshoot about 5 %

4. a settling time at 5 % equal to 0.36 s

Using the settling and the overshoot specifications, we conclude that the dominant poles are $s_{1,2} = -8.33 \pm 8.35j$ and from the root locus of the system, we get that the gain K_1 that gives these poles is $K_1 = 8.35$

Using now the steady state specifications, we conclude that K_2 is equal to 100.

From the values of these two gains, we get the parameter, a of the controller:

$$a = \frac{K_1}{K_2} = \frac{8.35}{100} = 0.0835$$

It is also important to notice that $a = \frac{p}{z}$, where p and z are respectively the pole and the zero of the controller. Now, if we place the zero at -1.5, we get:

$$p = az = 0.1253$$

and since $p = \frac{1}{T}$, we get: $T = 7.9808$.

For the controller gain, it is given by:

$$K_P = \frac{100}{48.5} = 2.0619$$

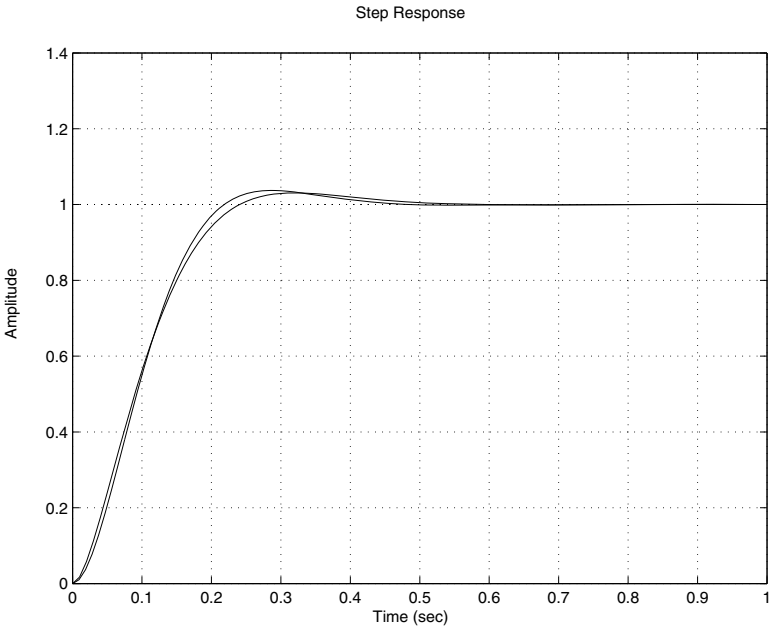


Fig. 5.51 Step of $F(s)$ with two controllers for two design methods

Finally, the transfer function of the controller is given by:

$$C(s) = K_P \frac{aTs + 1}{Ts + 1}, a < 1$$

Using the Bode method, we design a controller that provides the following specifications:

1. stable system
2. steady state error to a unit ramp is less than 0.01
3. phase margin greater than 40^o
4. gain margin greater than 8 db

Using the error specification, a gain $\bar{K}_P$ equal to 100 is needed. This gives a gain $K_P = 2.0619$ for the phase lead-lag controller. The Bode diagram of the open loop transfer of the system with this gain is illustrated in Fig. 5.50. From this figure we have:

$$\Delta\phi = 23.1^o$$
$$\Delta G = \infty$$

The open loop transfer of the system with this controller is illustrated at the Fig. 5.52. The system will have a phase margin equal to 45^o at the frequency $w_m = 16.9\ rd/s$.

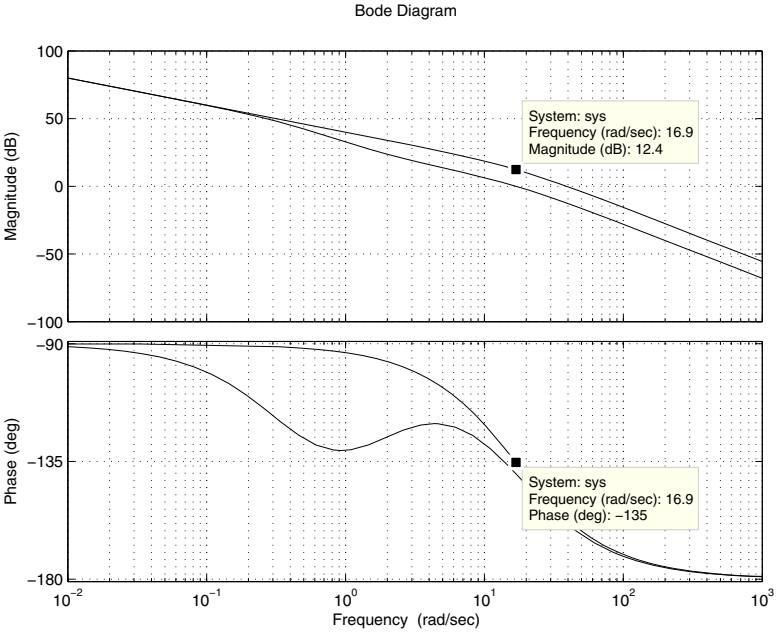


Fig. 5.52 Bode plot of $T(s)\frac{100}{(\tau_m s+1)}$

For the design of the phase lag controller, notice that at $w_m = 16.9\ rd/s$, the magnitude is equal to 12.4 db. Therefore,

$$a = 10^{\frac{-12.4}{20}} = 0.2399$$

The parameter T is given by:

$$T = \frac{10}{aw_m} = 2.4667$$

The controller phase-lag is given by the following transfer function:

$$C_2(s) = \frac{aTs+1}{Ts+1}$$

Combining now the two controllers, the open loop transfer function is given by:

$$T(s) = \frac{59.1716s + 100}{s(0.1480s^2 + 2.5267s + 1)}$$

The Bode diagram of this transfer function is represented at the 5.52. The specifications are:

$$\Delta\phi = 40.3^o$$
$$\Delta G = \infty$$

which are acceptable.

The behavior of the system with this controller for a step input is illustrated at the Fig. 5.53. As it can be seen that the two methods give two controllers that are almost identical and the step response are also almost identical.

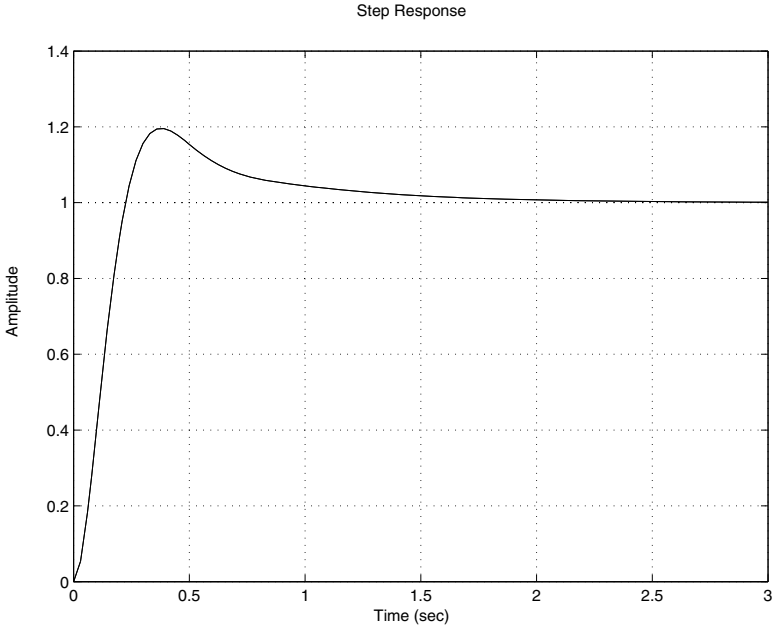


Fig. 5.53 Step of $F(s)$ with two controllers for two design methods

5.6.7 Phase Lead-Lag Controller

For this controller, we can use only the root-locus and the Bode methods to design it. Let us first of start the design of the controller using the root-locus method. It is important to notice that the best settling time at 5 % with a proportional controller is about 0.36 s. With the phase lead controller we want to improve this time. Let the desired pole dominant with positive imaginary part be $s_d = -11.5 + 11.6j$ which corresponds to a settling time equal to 0.27 s and an overshoot equal to 5 %. The phase of the system without the controller is given by:

$$\arg\left(\frac{48.5/0.06}{s_d(s_d + 16.6667)}\right) = 0 - 90 - 65.9917 = -155.9917$$

The phase lead controller must increase the phase with $180 - 155.9917 = 24.0083$

This implies:

$$\beta - \alpha = 24.0083$$

If we place the zero at -20, this implies that $\beta = 53.7676^o$ and the pole at -30 gives an angle of 52.89^o. This gives a contribution of 21.6788^o by the controller and which close to the desired one.

From this, we have:

$$\frac{1}{T_1} = 30$$
$$\frac{1}{a_1 T_1} = 20$$

this gives $T_1 = 0.0333$ and $a_1 = 1.5$.

For the phase lag controller design using the root locus technique, we will assume that we want the following specifications:

1. stable system
2. a steady state error to a unit ramp input equal to 0.01
3. an overshoot about 5 %
4. a settling time at 5 % equal to 0.27 s

Using the settling and the overshoot specifications, we conclude that the dominant poles are $s_{1,2} = -11.5 \pm 11.5j$ and from the root locus of the system, we get that the gain K_1 that gives these poles is $K_1 = 12.5$

Using now the steady state specifications, we conclude that K_2 is equal to 100.

From the values of these two gains, we get the parameter, a_2 of the controller:

$$a_2 = \frac{K_1}{K_2} = \frac{12.5}{100} = 0.125$$

It is also important to notice that $a_2 = \frac{p}{z}$, where p and z are respectively the pole and the zero of the controller. Now, if we place the zero at -0.1, we get:

$$p = a_2 z = 0.0125$$

and since $p = \frac{1}{T_2}$, we get: $T_2 = 80$.

For the controller gain, it is given by:

$$K_P = \frac{100}{48.5} = 2.0619$$

Finally, the transfer function of the controller is given by:

$$C(s) = K_P \frac{(a_1 T_1 s + 1)(a_2 T_2 s + 1)}{(T_1 s + 1)(T_2 s + 1)}, a_1 > 1, a_2 < 1$$

Using the Bode method, we design a controller that provides the following specifications:

1. stable system

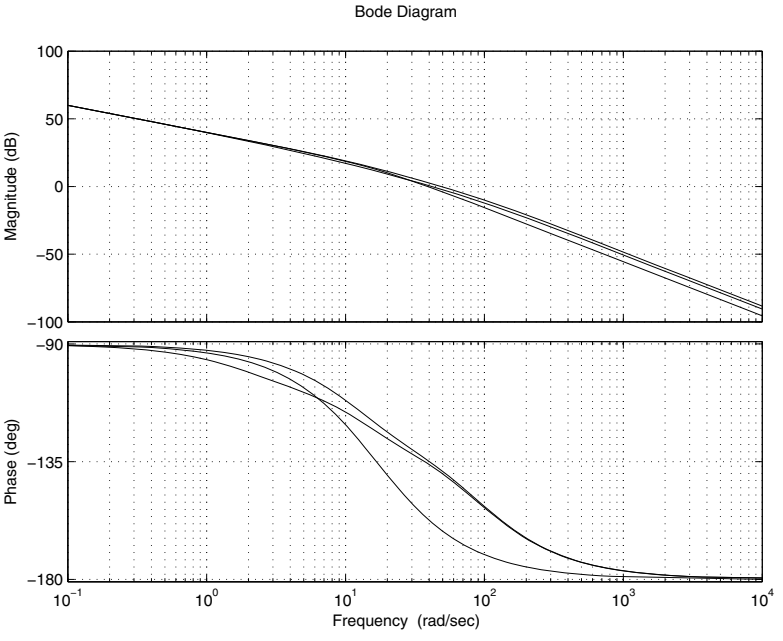


Fig. 5.54 Root locus of $T(s)\frac{K(0.5s+1)}{s^2(\tau_m s+1)}$, with $K = 1$, and $K = K_m K_P$

2. steady state error to a unit ramp is less than 0.01
3. phase margin greater than 40^o
4. gain margin greater than 8 db

Using the error specification, a gain $\bar{K}_P$ equal to 100 is needed. This gives a gain $K_P = 2.0619$ for the phase lead-lag controller. The Bode diagram of the open loop transfer of the system with this gain is illustrated in Fig. 5.55. From this figure we have:

$$\begin{aligned} \Delta\phi &= 23.1^o \\ \Delta G &= \infty \end{aligned}$$

For the design of phase lead controller notice that this controller should bring $45^o - 23.1^o = 22.9^o$, which gives:

$$a_1 = \frac{1+\sin(22.9)}{1-\sin(22.9)} = 2.2740.$$

Using this values the magnitude will take the value $-20log(\sqrt{a_1}) = -3.5679$ at the frequency $w_m = 48.9\ rd/s$. This implies:

$$T_1 = \frac{1}{w_m\sqrt{a_1}} = 0.0136$$

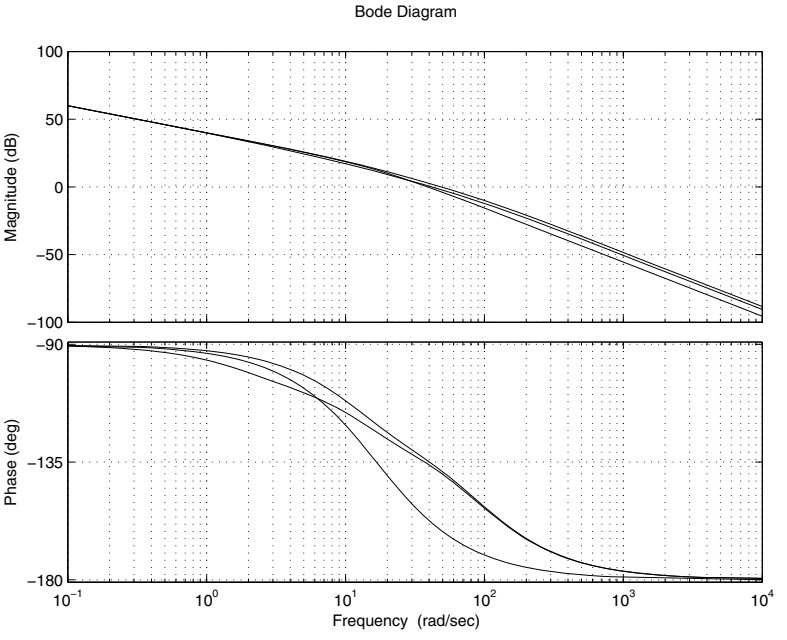


Fig. 5.55 Bode plot of $T(s)\frac{K(0.5s+1)}{s^2(\tau_m s+1)}$, with $K = 1$, and $K = K_m K_P$

The phase-lead controller is then given by the following transfer function:

$$C_1(s) = \frac{a_1 T_1 s + 1}{T_1 s + 1}$$

With this controller, the compensated system has:

$$\Delta\phi = 41.8^o$$
$$\Delta G = \infty$$

The open loop transfer of the system with this controller is illustrated at the Fig. 5.55. The system will have a phase margin equal to 45^o at the frequency $w_m = 41.3\ rd/s$.

For the design of the phase lag controller, notice that at $w_m = 41.3\ rd/s$, the magnitude is equal to 2.13 db. Therefore,

$$a_2 = 10^{\frac{-2.13}{20}} = 0.7825$$

The parameter T_2 is given by:

$$T_2 = \frac{10}{a_2 w_m} = 0.3094$$

The controller phase-lag is given by the following transfer function:

$$C_2(s) = \frac{a_2 T_2 s + 1}{T_2 s + 1}$$

Combining now the two controllers, the open loop transfer function is given by:

$$T(s) = \frac{0.7467s^2 + 27.2969s + 100}{s(0.0003s^3 + 0.02830s^2 + 0.3830s)}$$

The Bode diagram of this transfer function is represented at the 5.55. The specifications are:

$$\Delta\phi = 43.3^o$$
$$\Delta G = \infty$$

which are acceptable.

The closed-loop transfer function of the system with this controller is given by:

$$F(s) = \frac{K_m K_P \left(a_1 a_2 T_1 T_2 s^2 (a_1 T_1 + a_2 T_2)s + 1\right)}{b_4 s^4 + b_3 s^3 + b_2 s^2 + b_1 s + b_0}$$

with $b_4 = \tau_m T_1 T_2$, $b_3 = \tau_m(T_1 + T_2) + T_1 T_2$, $b_2 = \tau_m + T_1 + T_2 + K_m K_P a_1 a_2 T_1 T_2$, $b_1 = 1 + K_m K_P (a_1 T_1 + a_2 T_2)$ and $b_0 = K_m K_P$.

The behavior of the system with this controller for a step input is illustrated at the Fig. 5.56. As it can be seen that the two methods give two controllers that are almost identical and the step response are also almost identical.

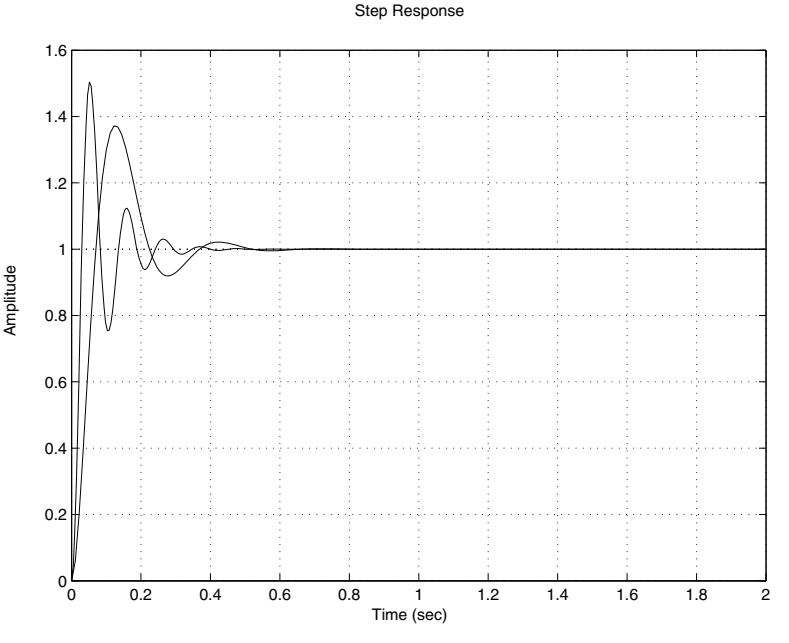


Fig. 5.56 Step of $F(s)$ with two controllers for two design methods

Remark 5.6.2 *From this section, and the previous two ones, we can conclude for a given system, the phase lead-lag controller can not be obtained by the multiplication of the of the phase lead controller and the phase lag controller transfer function designed separately.*

The implementation of the different algorithms we developed in this case study can be done as it will be done in the implementation part. Tab. 5.5 gives the different difference equations to program in each controller. To get these equations we have used the trapezoidal schema and by denoting the sampling period by T_s.

Table 5.5 Difference equations for the different controllers: dc motor kit

Controller	Algorithm
P	$u(k) = K_P e(k)$
PI	$u(k) = u(k-1) + ae(k) + be(k-1)$ $a = K_P + \frac{K_I T_s}{2}, b = -K_P + \frac{K_I T_s}{2}$
PD	$u(k) = -u(k-1) + ae(k) + be(k-1)$ $a = K_P + \frac{2K_D}{T_s}, b = K_P - \frac{2K_D}{T_s}$
PID	$u(k) = u(k-2) + ae(k) + be(k-1) + ce(k-2)$ $a = K_P + \frac{K_I T_s}{2} + 2\frac{K_D}{T_s}, b = K_I T_s - \frac{4K_D}{T_s}, c = \frac{K_I T_s}{2} + \frac{2K_D}{T_s} - K_P$
Lead	$u(k) = -a_0 u(k-1) + be(k) + ce(k-1)$ $a_0 = \frac{T_s - 2T}{T_s + 2T}, b = K_P \frac{T_s + 2aT}{T_s + 2T}, c = K_P \frac{T_s - 2aT}{T_s + 2T}$
Lag	$u(k) = -a_0 u(k-1) + be(k) + ce(k-1)$ $a_0 = \frac{T_s - 2T}{T_s + 2T}, b = K_P \frac{T_s + 2aT}{T_s + 2T}, c = K_P \frac{T_s - 2aT}{T_s + 2T}$
Lead-Lag	$u(k) = -a_0 u(k-1) - bu(k-2) + ce(k) + de(k-1) + fe(k-2)$ $a_0 = \frac{(T_s - 2T_1)(T_s + 2T_2) + (T_s + 2T_1)(T_s - 2T_2)}{(T_s + 2T_1)(T_s + 2T_2)}, b = \frac{(T_s - 2T_1)(T_s - 2T_2)}{(T_s + 2T_1)(T_s + 2T_2)},$ $c = K_P \frac{(T_s + 2a_1 T_1)(T_s + 2a_2 T_2)}{(T_s + 2T_1)(T_s + 2T_2)}, d = K_P \frac{(T_s - 2a_1 T_1)(T_s + 2a_2 T_2) + (T_s + 2a_1 T_1)(T_s - 2a_2 T_2)}{(T_s + 2T_1)(T_s + 2T_2)},$ $f = K_P \frac{(T_s - 2a_1 T_1)(T_s - 2a_2 T_2)}{(T_s + 2T_1)(T_s + 2T_2)}$

5.7 Conclusion

Practical systems when designed need in general the design of a controller that improves the performances of such systems. The performances give an idea on the transient and transient regimes. Mostly, the overshoot, the settling time, the steady state error are considered as for the design of controllers. This chapter covers the design of the classical controllers like proportional, integral and derivative actions. Procedures using the empirical methods, root-locus technique and Bode plot technique are proposed and illustrated by numerical examples.

5.8 Problems

1. In this problem we consider the control of a small satellite. The mathematical model of this dynamical system is given by:

$$G(s) = \frac{\Theta(s)}{U(s)} = \frac{k}{s^2}$$

where $\Theta(s)$ is the angle to be controlled, $U(s)$ is the force to apply to the satellite and $k = 2$ is the gain of the satellite that depends on many parameters of the system.
Using the three techniques developed in this chapter to design the controller that gives the best performances and stabilizes the system.

2. Consider the following dynamical system:

$$G(s) = \frac{4}{s(0.1s + 1)(s - 1)}$$

Determine the appropriate technique developed in this chapter to design the controller that gives the best performances and stabilizes the system.

3. Consider the following dynamical system:

$$G(s) = \frac{4}{s(0.2s + 1)^2}$$

Determine the appropriate technique developed in this chapter to design the controller that gives the best performances and stabilizes the system.

4. Consider the following dynamical system:

$$G(s) = \frac{5}{s(0.1s + 1)(0.2s + 1)}$$

Design a controller that assures the following performances:

(a) stable system
(b) steady state error to a unit ramp equal to 0.1
(c) settling time at 5 % less than 1 s
(d) overshoot less than 5 %

5. A dynamical system is described by the following dynamics:

$$G(s) = \frac{10}{(s + 1)(s + 5)(s + 10)}$$

Using the Ziegler-Nichols methods design the different controllers that we can design for this system and compare their performances
Using now the root locus and Bode methods design the controllers that gives good performances for this system. Make a comparative study of these controllers.

6. Consider a dynamical system with the following dynamics:

$$G(s) = \frac{5(s+2)}{s(s+1)(s+5)(s+10)}$$

Determine the appropriate technique developed in this chapter to design the controller that gives the best performances and stabilizes the system.

7. A dynamical system is described by the following transfer function:

$$G(s) = \frac{0.4(0.2s+1))}{(0.1s+1)(0.4s+1)(0.5s+1)(0.8s+1)}$$

Determine the appropriate technique developed in this chapter to design the controller that gives the best performances and stabilizes the system.