Discrete control algorithms for systems with big dead time for control101 MATLAB toolbox

Ruth Bars, Gyula Max
Department of Automation and Applied Informatics
Budapest University of Technology and Economics
Budapest, Hungary
bars@aut.bme.hu,
maxgyula@freemail.hu

J. Anthony Rossiter
School of Electrical and Electronic Engineering
University of Sheffield
Sheffield, UK
j.a.rossiter@sheffield.ac.uk

László Keviczky
Systems and Control Laboratory
Institute for Computer Science and Control
Budapest, Hungary
keviczky@sztaki.hu

***Abstract* – The first course of control is under a critical review. Both the teaching material covered and the teaching methods require new considerations. Introducing interactivity in the education process makes the learning more successful and enjoyable. MATLAB provides an effective environment for learning and applying different disciplines. Control101 is a new MATLAB toolbox under development which provides tools for interactive learning of control disciplines. This paper presents the framework for teaching discrete control algorithms applied for processes containing large dead times.**

Keywords—control education, control101 MATLAB toolbox, independent learning, discrete control algorithms, dead time.

I. Introduction

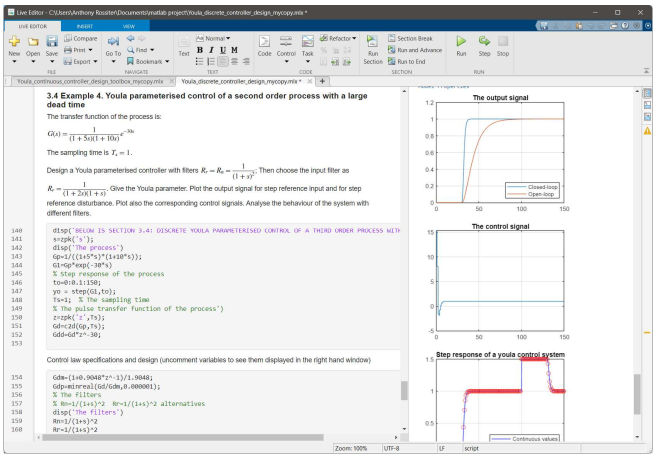
In recent years the control community has been evaluating and investigating the content and teaching methods of a first control course [1], [2], [3], [4]. Some important aspects are, for example, to focus on motivation and on discussing control concepts using interesting case studies, visualization and laboratories. These should emphasize the role of feedback and analyse its behaviour. Control algorithms should be designed both in the continuous and discrete environments, where discrete control algorithms are applied in computer control. The proposal is to use software to support computation and illustration and provide interactivity in the learning process. Interactivity makes the learning process more effective and more enjoyable. The community also would like to share learning and teaching resources to provide high quality learning experiences to all students [5], [6], [7]. Furthermore, it was noted that MATLAB is widely used to solve mathematical and technical problems, especially within control education. A new toolbox, Control101 is under development to support a first control course [8]-[11]. The authors are not aware of similar resources with the same objectives, although of course multiple resources exist with a much narrower remit.

This paper describes some new files developed for the Control101 toolbox providing interactive teaching materials to analyse discrete control algorithms used in sampled control systems. Besides the most frequently used PID algorithms (which have different forms), dead beat control, Smith predictor control and Youla parameterisation are presented [12], [13]. The behaviour of the algorithms is compared in the case of large dead times.

The paper is organised as follows. Section II gives some background on the use of MATLAB. Section III. introduces the Control101 toolbox. Section IV discusses the basic architecture of control of sampled systems used in computer control. Section V describes several discrete control algorithms which can be designed and evaluated by the elaborated mlx files in the toolbox. Section VI compares the behaviour of the different algorithms through an example of control of a third order process with big dead time and demonstrates how the different algorithms tolerate mismatch in dead time. This is followed by the summary and the references.

II. The Use of MATLAB

MATLAB is widely used in the control community, so many students may already be familiar with its use. Of particular relevance here, MATLAB provides a *livescript* environment (mlx files) which combines both notes and code within an integrated setting. A detailed explanation of a topic can be given, followed by the code required to do the associated computations and illustrations. When using the toolbox, a personalized copy of the file is created automatically so that students can analyse the behaviour using their own modifications. Fig. 1. illustrates the livescript environment. The explanation and the code appears in the left hand column on the screen, while the results are shown in the right hand column. Livescripts can be organised into sections, so that users only need to run the code snippets within a single section at a time.


Live Editor - C:\Users\Anthony Rossiter\Documents\matlab project\Youla_discrete_controller_design_mycopy.mlx *
LIVE EDITOR
INSERT
VIEW
New
Open
Save
Compare
Print
Export
Go To
Find
Bookmark
Text
Normal
Code
Control
Task
Refactor
Run Section
Section Break
Run and Advance
Run to End
Run
Step
Stop
Youla_continuous_controller_design_toolbox_mycopy.mlx
Youla_discrete_controller_design_mycopy.mlx *
The output signal
Closed-loop
Open-loop
The control signal
Step response of a youla control system
Continuous values
Zoom: 100%
UTF-8

3.4 Example 4. Youla parameterised control of a second order process with a large dead time

The transfer function of the process is:

$$G(s) = \frac{1}{(1+5s)(1+10s)}e^{-30s}$$

The sampling time is $T_s = 1$.

Design a Youla parameterised controller with filters $R_r = R_n = \frac{1}{(1+s)^2}$; Then choose the input filter as $R_r = \frac{1}{(1+2s)(1+s)}$. Give the Youla parameter. Plot the output signal for step reference input and for step reference disturbance. Plot also the corresponding control signals. Analyse the behaviour of the system with different filters.

```
disp('BELOW IS SECTION 3.4: DISCRETE YOULA PARAMETERISED CONTROL OF A THIRD ORDER PROCESS WITH
s=zpk('s');
disp('The process')
Gp=1/((1+5*s)*(1+10*s));
G1=Gp*exp(-30*s)
% Step response of the process
to=0:0.1:150;
yo = step(G1,to);
Ts=1;  % The sampling time
% The pulse transfer function of the process')
z=zpk('z',Ts);
Gd=c2d(Gp,Ts);
Gdd=Gd*z^-30;
```

Control law specifications and design (uncomment variables to see them displayed in the right hand window)

```
Gdm=(1+0.9048*z^-1)/1.9048;
Gdp=minreal(Gd/Gdm,0.000001);
% The filters
% Rn=1/(1+s)^2  Rr=1/(1+s)^2 alternatives
disp('The filters')
Rn=1/(1+s)^2
Rr=1/(1+s)^2
```

Fig. 1. Illustration of the livescript environment

While m-files and mlx-files code are effective for handling simple analysis and simulation of linear systems, there are some aspects which can be better represented using block diagrams, thus exploiting the SIMULINK environment. Critically however, SIMULINK block diagrams can also be activated and run from within the livescript file, thus masking unnecessary detail and complexity from the user.

A further observation is that while livescripts fill a function, they rely on the user understanding the associated code and linking this to the engineering scenario. A more intuitive environment would be more visual. Consequently, virtual laboratories (or mlapp files) can also be developed to provide an interactive environment which allows users to analyse and try different disciplines and scenarios. The interface allows the user to select buttons, sliders, drop down menus and so forth and the results are presented in figures and text as appropriate. Several examples are in the toolbox [9,10].

In summary, a holistic learning environment may contain both virtual laboratories and livescripts. The virtual laboratories introduce the concepts and allow the users to learn by trial and error, while the livescripts provide deeper understanding and practice in analysis and design.

III. The Control101 Matlab Toolbox

A MATLAB toolbox [10] was proposed recently for a first control course, where the priority is helping students focus on core concepts and visualization, before getting into detailed coding. A first draft of the toolbox was released at the IFAC world congress in 2023. Since then new releases have added more basic and also advanced topics. The toolbox can be downloaded freely through Add-Ons option of MATLAB. The code is open source available at

(https://github.com/jarossiter/control101, or https://sites.google.com/sheffield.ac.uk/controleducation/matlabresources).

The initial resources focused on a first control course and discuss the basic ideas of control. Mainly linear control systems are considered. The topics cover modelling of systems, describing the system behaviour in the time, operator and frequency domain, defining the quality specifications, analysing the properties of the control structure realised by negative feedback, stability considerations, controller design in continuous and discrete (sampled) control systems, PID controller design in the Laplace/z-operator and in the frequency domain. The Control101 toolbox provides livescript files and virtual laboratories to study these topics.

The purpose of this paper is to highlight some of the more recent resources which have been added. These cover topics which would more likely be covered in later or advanced control courses and thus allow continuity for students who are ready to move on from a first course and thus find relevant resources in the same place. The focus of this paper is specifically on sampled data systems.

IV. Control of Sampled Systems

In practice it is common for the control system to be realized by a process control computer equipped with real time facilities. The block diagram of a sampled control system is given in Fig. 2. The plant to be controlled is continuous. Its model is characterized by its transfer function $G(s)$. The aim of the control system is to track a reference signal (here $r[k]$) and to attenuate the effect of any disturbances while meeting quality specifications (stability, static accuracy, prescribed dynamic response, robustness, etc.). The control is realized by negative feedback. The output $y(t)$ of the plant is sampled with an appropriate sampling time. An A/D converter quantizes and digitalizes the signal and forwards it to the computer. The computer creates the reference signal at the sampling points, calculates the difference between the reference signal and the measured output signal, and a digital control algorithm, characterized by the $M(z)$ pulse transfer function, calculates the control signal in the sampling points and forwards it to the input of the plant via a D/A converter, which provides a physical analogue signal. As the plant is continuous, a holding unit is required which produces the input between the sampling points. This is generally zero order hold (ZOH) unit, which keeps the actual control signal until the next sampling point.

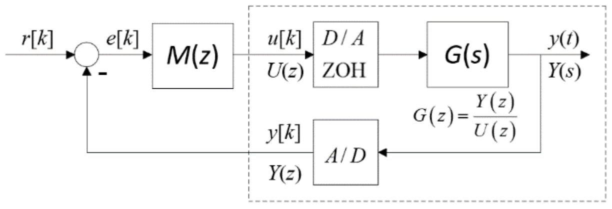


Fig. 2. Block diagram of a sampled control system

The pulse transfer function $M(z)$ of the controller is to be designed to meet the quality specifications (stability, static accuracy, dynamic behaviour, robustness). Assume the plant $G(s)$ includes a large dead time (which frequently is transport delay). Several discrete control algorithms are given and their behaviour is compared using the related livescript files.

V. Discrete Control Algorithms

This section introduces 4 different control design approaches for discrete systems and the toolbox files that have been developed to support student learning and understanding of those approaches.

A. *PID control with pole cancellation*

The most frequently used control algorithms are some discretised forms of PID control, using proportional, integral and derivative effects applied on the error signal. The pulse transfer function of the controller is determined in pole-zero form. One form of the controller is given as a product of PI and PD effects. Its pulse transfer function is described as

$$M(z) = A\,\frac{z-e^{-T_S/T_1}}{z-1}\,\frac{z-e^{-T_S/T_2}}{z} \qquad (1)$$

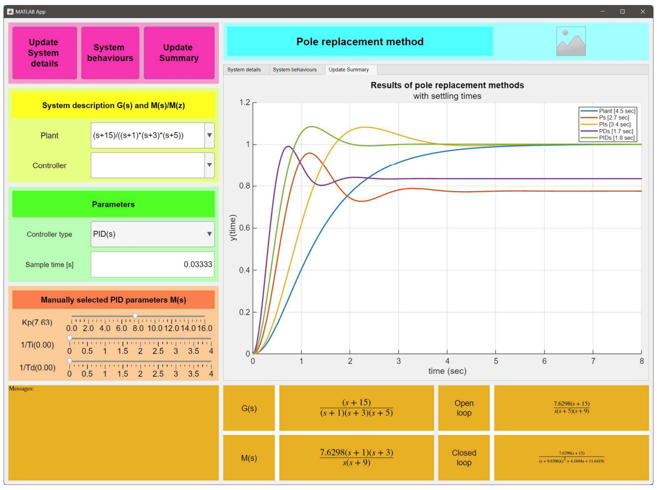
and the control algorithm is executed by a difference equation. For example, in the case of only a PI controller the control signal is calculated by the following difference equation, which recursively calculates the current control signal from the actual and the previous error signal and the previous value of the control signal.

$$u[k] = Ae[k] - A\exp\left(-{T_S}/{T_1}\right)e[k-1] + u[k-1] \quad (2)$$

One possibility to design the parameters of the controller is using a pole cancellation technique [12] based on frequency domain considerations.

Generally the biggest time constant of the plant model is cancelled and an integrating effect is introduced instead, and the second biggest time constant is also cancelled, being substituted by a differentiating effect. The gain can be calculated to ensure a phase margin of about 60° (obviously some fine tuning is possible if needed). It has to be emphasized that sampling and using a zero order hold introduces an extra delay in the system whose value is about half of the sampling time.

A virtual laboratory mlapp file has been prepared to do experiments with control of different plants and different PID controllers. Fig. 3. illustrates a possible use of this file. The related livescript file explains the controller design and provides the code which can be fitted to the design task.



System description G(s) and M(s)/M(z)

Plant (s+15)/((s+1)*(s+3)*(s+5))

Controller

Parameters

Controller type PID(s)

Sample time [s] 0.03333

Manually selected PID parameters M(s)

Kp(7.63)

1/Ti(0.00)

1/Td(0.00)

G(s) $\frac{(s+15)}{(s+1)(s+3)(s+5)}$

Open loop $\frac{7.6298(s+15)}{s(s+5)(s+9)}$

M(s) $\frac{7.6298(s+1)(s+3)}{s(s+9)}$

Closed loop $\frac{7.6298(s+15)}{(s+9.8306)(s^2+4.1694s+11.6419)}$

Fig. 3. Interface of the PID controller designer virtual laboratory

For processes with a large dead time the settling time will be high and the control process will be slow.

The elaborated livescript file:

- shows how to calculate the pulse transfer functions of different elements.
- visualises the step responses of the PI, PD and PID controllers.
- gives examples for controller design for a process with different controllers.
- shows also the design for systems with dead time.

The students can change the process and design their own controllers. The code snippet below (Fig. 4.) illustrates how the livescript supports student design, while emphasising the core design decisions required. Figure 4 shows the step responses in the sampling points with different controllers.

```
%Frequency function with the PIPD controller
[mag,phase]=bode(Lpipd,w);
Table=[mag(:),phase(:),w']
%evaluate the table:The phase angle is -120° at w=0.2674. The gain here
%is 0.0756.The cut-off frequency is located at this w value if is
Apid=1/0.0756
Mpipd=Apid*Mpipd; Lpipd=Apid*Lpipd;
%check the phase margin
margin(Lpipd)
%Plot the output signal
step(Lpipd/(1+Lpipd)),grid
```

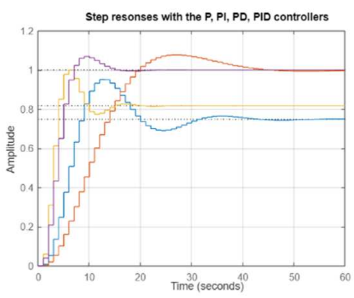


Fig. 4. Code snippet of PIPD design and the obtained outputs

B. *Dead beat control*

For discrete systems it can be prescribed that the output should reach the step reference value within finite sampling steps. So a faster settling process is expected. The pulse transfer function of the plant is given as:

$$G(z) = \frac{B(z)}{A(z)} z^{-d} \tag{3}$$

where $B(z)$ and $A(z)$ are the numerator and denominator polynomials in the z variable and d is the discrete dead time,

$$d = round\left(\frac{T_d}{T_s} + 1\right) \tag{4}$$

where T_d is the continuous dead time and T_s is the sampling time. It is desired that the settling would be realised within d steps.

The pulse transfer function of the closed loop is then prescribed as:

$$T(z) = z^{-d} \tag{5}$$

and as for the closed loop the overall transfer function is:

$$\frac{M(z)G(z)}{1+M(z)G(z)} = T(z) \tag{6}$$

the pulse transfer function of the controller is:

$$M(z) = \frac{T(z)}{G(z)(1-T(z))} = \frac{A(z)}{B(z)(1-z^{-d})} \tag{7}$$

The algorithm assumes a step reference signal here but can be extended to other reference signals as well.

If $B(z)$ contains roots outside the unit circle (or in undesirable regions of the circle), then these should not appear in the denominator of the controller, as they would cause intersampling oscillations. In this case let us separate the numerator to the product of the cancellable B_+ and the non-cancellable B_- factors, where the gain of B_- should be 1.

$$\frac{M(z)G(z)}{1+M(z)G(z)} = T(z) = B_-(z)\, z^{-d} \tag{8}$$

Hence the pulse transfer function of the controller is:

$$M(z) = \frac{A(z)}{B_+(z)(1-B_-(z)\, z^{-d})} \tag{9}$$

The elaborated livescript file supports the calculations of the dead beat controller design. After the explanation of the algorithm it shows some examples and calls also the corresponding SIMULINK file to demonstrate the behaviour. The students can try other examples as well in their own copy.

Fig. 5. shows a code snippet and the results of the design, the control signal and the output signal obtained by calling the SIMULINK model.

```
disp('Section 4.1: Design of a dead beat controller avoiding …
%Continuous and discrete models
s=zpk('s');
G=1/((1+4*s)*(1+10*s));
Ts=2;          %sampling time
z=zpk('z',Ts);
disp('The pulse transfer function')
Gz=c2d(G,Ts);
Gz1=Gz*z^-6;
% define the controller
% Extract the zeros, the poles and gain of the pulse transfer function
[zer,p,k]=zpkdata(Gz,'v')
A=(z-p(1))*(z-p(2))/(z*z); % expressed in terms of the shift operator
Bminus=(z-zer)/(z*(1-zer))   % Scaled to give dcgain of unity
Bplus=k*(1-zer);
disp('The controller')
Mz=A/(Bplus*(1-Bminus*z^-7))
disp('Closed-loop transfer functions')
Gcuz=feedback(Mz,Gz1);      % Closed-loop transfer function …
Gcyz=feedback(Gz1*Mz,1);    % Closed-loop transfer function …
```

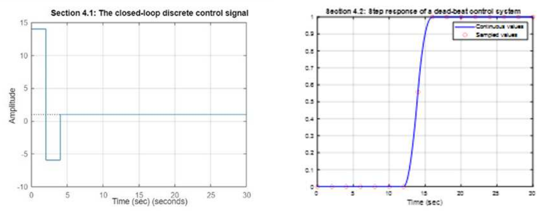


Fig. 5. Code snippet of dead beat design; the control signal and the output signal

C. *Smith predictor*

O.J. Smith [13] suggested a control algorithm which behaves as if the dead time would have been located outside of the feedback loop, as shown in Fig. 6, supposing a continuous control system. The upper figure shows the original control system, while the lower figure puts the dead time outside of the feedback loop. So the controller M_+ is designed for the dead time free process thus ensuring a faster settling process. The design algorithm can be a PID controller.

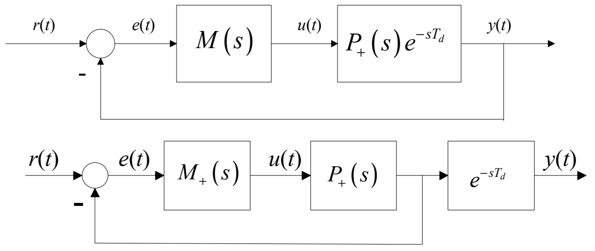


Fig. 6. The idea of Smith predictor

For the equivalence of the two control circuits the following relationship can be given:

$$\frac{M\,P_+\,e^{-sT_d}}{1+M\,P_+\,e^{-sT_d}} = \frac{M_+\,P_+}{1+M_+\,P_+}e^{-sT_d} \qquad (10)$$

Hence the controller M is expressed as:

$$M = \frac{M_+}{1+M_+\,P_+(1-e^{-sT_d})} \qquad (11)$$

For continuous control it is difficult to realise the controller as the dead time does appear in the controller algorithm. This is not a problem in a discrete control system, where the pulse transfer function of the controller is given as:

$$M(z) = \frac{M_+(z)}{1+M_+(z)G_+(z)\,(1-z^{-d})} \qquad (12)$$

where $G_+(z)$ is the pulse transfer function of the delay free process and d is the discrete dead time, i.e. the ratio of the dead time and the sampling time.

```
disp('EXAMPLE 3.1: Design of Smith predictor controller')
s=zpk('s'); G=1/(1+5*s)        % Continuous Model without the delay
Ts=2.5;                        % sampling time
z=zpk('z',Ts);
% The pulse transfer function without the delay
Gz=c2d(G,Ts);   %or alternatively Gz=c2d(G,Ts,'zoh'), discrete model
[zer,pol,kd]=zpkdata(Gz,'v');
disp('The pulse transfer function with the delay')
Gz1=Gz*z^-4                    % Discrete model with the delay
disp('PI controller for the delay free process with pole cancellation')
Mplus=(z-pol)/(z-1);           % PI structure
Lz=minreal(Mplus*Gz,0.001); w=logspace(-1,0,200); [mag,phase]=bode(Lz,w);
ki=margin(mag,phase-60,w);% choose gain of the controller for 60 deg PM
Mplus1=ki*Mplus
disp('The Smith controller')
M=Mplus1/(1+Mplus1*Gz*(1-z^-4)); M=minreal(M,0.001)
L=minreal(M*Gz1,0.001);
T=minreal(L/(1+L),0.001); U=minreal(M/(1+L),0.001);
```

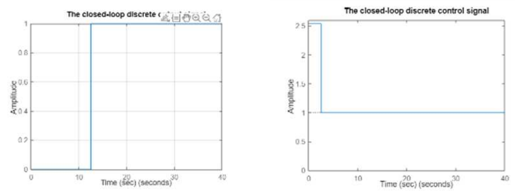


Fig. 7. Code snippet of Smith predictor design; the output signal and the control signal

The elaborated livescript file supports the calculations of Smith predictor design. After the explanation of the algorithm it shows some examples and calls also the corresponding SIMULINK file to demonstrate the behaviour. The students can try other examples as well in their own copy. Fig. 7. shows a code snippet and the results of the design, the output signal and the control signal.

D. *Discrete Youla controller*

The relationship between the z transforms of the *u* control signal and the *r* reference signal supposing no disturbances is given by the following relationship with the pulse transfer functions:

$$\frac{U(z)}{R(z)} = \frac{M(z)}{1+G(z)M(z)} \qquad (13)$$

The pulse transfer function Q is called Youla parameter. The block diagram of the control system showing also the input and output disturbances is given in Fig. 8.

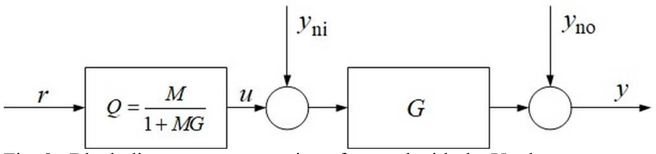


Fig. 8. Block diagram representation of control with the Youla parameter

The Q controller can be designed in open loop. The best controller choice would be $Q = G^{-1}$. Then the system will track the reference signal accurately. It should be mentioned that generally this pulse transfer function is not realisable. The open loop structure can not reject the effect of the inner and output disturbances. Therefore it is enhanced with internal model control (IMC) structure as shown in Fig. 9. This structure can be redrawn to get a usual feedback structure.

To get realisable control system the pulse transfer function is separated to the product of the invertible G_+ and the noninvertible G_- factors, where the static gain of the noninvertible factor should be 1.

$$G = G_+G_-z^{-d} \qquad (14)$$

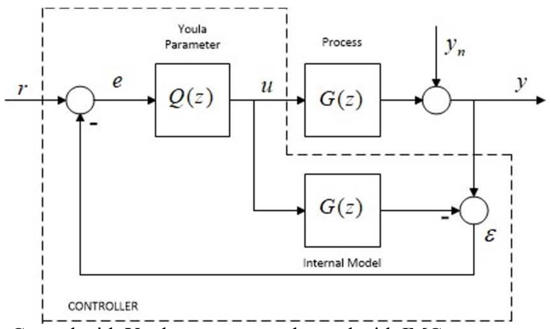


Fig. 9. Control with Youla parameter enhanced with IMC

In the open loop only the cancellable part of the model of the plant is cancelled. The control structure can be enhanced with the reference signal filter R_r and the disturbance signal filter R_n according to Fig. 10. The static gain of the filters should be 1. With the filters the dynamic of reference signal tracking and of disturbance rejection will be different. The filters influence the maximum value of the control signal. With the filters the robustness properties of the control system can be improved. The block diagram can be redrawn as shown in Fig. 11. Now the Youla parameter is:

$$Q = R_nG_+^{-1} \qquad (15)$$

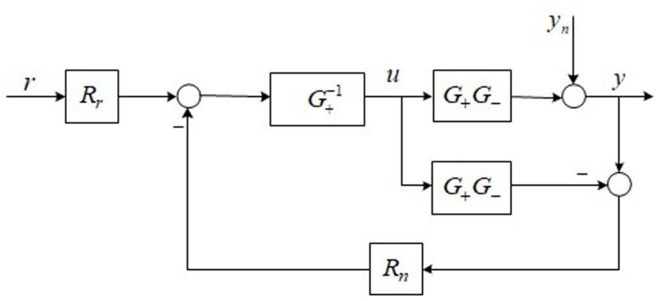


Fig. 10. Control with Youla parameter using reference and disturbance filters

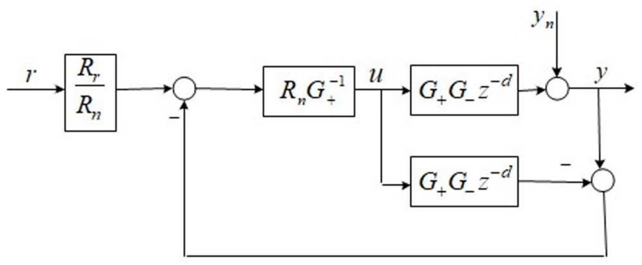


Fig. 11. Rearranged control structure with filters

Summarising the design: The pulse transfer function of the process is separated to invertible and noninvertible parts. The filters are chosen. The Q parameter is calculated ensuring good reference signal tracking in open loop. The control system is realised in IMC structure or the controller is transformed to the usual feedback form.

```
s=zpk('s'); disp('The process')
Gp=1/((1+5*s)*(1+10*s));
G1=Gp*exp(-30*s)
% Step response of the process
to=0:0.1:150;yo = step(G1,to);
Ts=1;  % The sampling time
% The pulse transfer function of the process')
z=zpk('z',Ts);Gd=c2d(Gp,Ts);Gdd=Gd*z^-30;
Gdm=(1+0.9048*z^-1)/1.9048;
Gdp=minreal(Gd/Gdm,0.000001);
disp('The filters')
Rn=1/(1+s)^2;Rr=1/(1+s)^2;
Rrd=c2d(Rr,Ts);Rnd=c2d(Rn,Ts);
%Youla design
Q=Rnd/Gdp;Q1=minreal(Q,0.001);
%Closed-loop transfer functions and simulation
T=(Rrd/Rnd)*Q1*Gdd;U=(Rrd/Rnd)*Q1;
t=0:Ts:150;y=step(T,t);
plot(t,y,to,yo),grid
title('The output signal');u=step(U,t);
stairs(t,u),grid
title('The control signal')
```

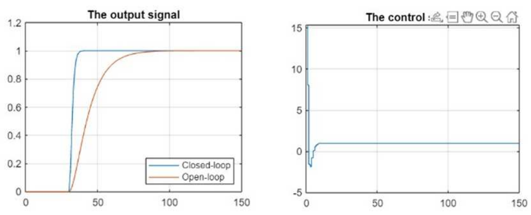


Fig. 12. Code snippet of Youla parameterisation design; the output signal and the control signal

The elaborated livescript file supports the calculations of the Youla controller design. After the explanation of the algorithm it shows some examples and calls also the corresponding SIMULINK file to demonstrate the behaviour. The students can try other examples as well in their own copy. Fig. 12 shows a code snippet and the results of the design, the output signal and the control signal.

VI. Discrete Control Algorithms in Case of Big Dead Time

In this section the proposed toolbox files are used to undertake the control designs described in the previous section and compare the results. Specifically, the behaviours of the control algorithms are compared in case of a third order lag element with big dead time. The effect of mismatch in the dead time is also shown. After running the four algorithms the outputs of the control system are stored and compared visualising them in one figure. In this example the transfer function of the plant is:

$$G(s) = \frac{2\,e^{-20s}}{(1+4s)(1+6s)(1+10s)}$$

The sampling time is $T_s = 2$. Design a controller with the above discussed algorithms. Compare the step responses and the control signals.

The pulse transfer function of the plant is:

$$G_d(z) = \frac{0.0086104(z+0.2061)(z+2.894)}{(z-0.8085)(z-0.7165)(z-0-6065)}\,z^{-10}$$

The *PID controller* designed for 60° phase margin is

$$M_{PID}(z) = \frac{0.394(z-0.8187)(z-0.7165)}{z(z-1)}$$

The pulse transfer function of the *dead beat controller* is:

$$M_{DB}(z) = \frac{(1-0.8187z^{-1})(1-0.7165z^{-1})(1-0.6065z^{-1})}{0.0404(1-0.21294(1+0.2061z^{-1})(1+2.894z^{-1})z^{-11})}$$

In *Smith predictor* design the PID controller designed for the delay free process is:

$$M_+(z) = \frac{1.894(z-0.8087)(z-0.7165)}{z(z-1)}$$

and the Smith predictor is calculated by equation (12).

In *Youla design* let the filters be 3 serially connected discrete first order lag elements. The transfer function of the reference filter is 1/(1+2s) and the discrete filter is $R_r(z) = \frac{0.2526}{(z-0.3679)^3}$; the transfer of the disturbance filter is 1/(1+3s) and the discrete filter is $R_n(z) = \frac{0.1152}{(z-0.5134)^3}$.

The pulse transfer function is separated to

$$G_+(z) = \frac{0.040436}{(z-0.8187)(z-0.7165)(z-0.6065)}$$

and $G_-(z) = 0.21294(z+0.2061)(z+2.894)$

The Youla parameter is obtained as

$$Q = \frac{R_n(z)}{G_+} = \frac{2.8491(z-0.8187)(z-0.7165)(z-0.6065)}{(z-0.5134)^3}$$

Fig. 13. shows the output signals of the different algorithms. It is seen that the dead beat, the Smith and the Youla algorithms result much faster behaviour than the PID control. The maximum value of the control signal in the dead beat, Smith and Youla control is much higher than the maximum value of the control signal in PID control. The higher control values result the faster settling process. Fig. 14. demonstrates the effect of mismatch in the dead time in case of PID control, when the controller is designed for dead time of 20 sec, while in reality its value is 16 or 24 sec. It is seen that the PID control algorithm is quite robust tolerating this mismatch in the dead time. Fig. 15. demonstrates the effect of mismatch in case of dead beat control. Here instead of 20 sec the dead time is 18 or 22 sec. Dead beat control is not robust, it is very sensitive to the accurate knowledge of the dead time. Fig. 16. shows that Smith predictor control tolerates better the mismatch. Fig.17. shows the effect of mismatch in case of the Youla controller. With appropriately chosen filters the behaviour can be improved.

The fundamental observation in this section is that proposed livescript files have supported a number of systematic designs in an easy to use fashion and facilitated the generation of appropriate evaluations and plots.

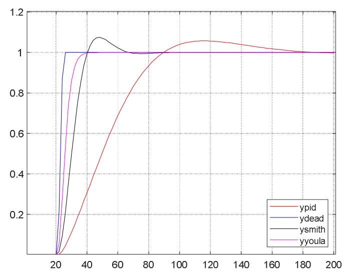


Fig. 13. Output signals of the different algorithms

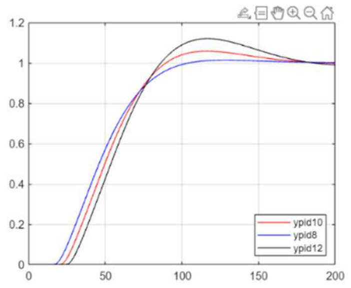


Fig. 14. Effect of mismatch in the dead time in case of PID control

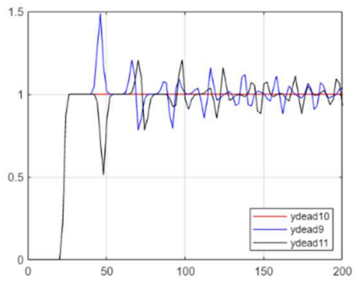


Fig. 15. Effect of mismatch in the dead time in case of dead beat control

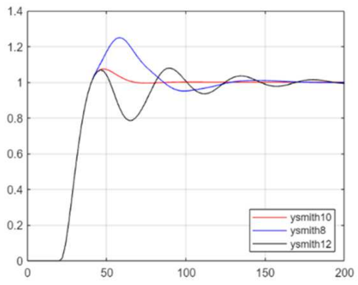


Fig. 16. Effect of mismatch in the dead time in case of Smith predictor

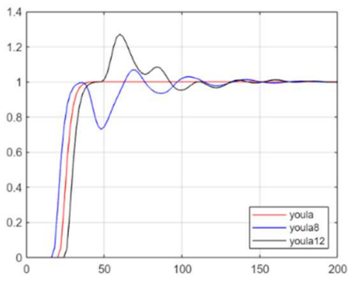


Fig. 17. Effect of mismatch in the dead time in case of Youla controller

VII. CONCLUSION

The first course of control is under critical review regarding both the teaching material and methods. MATLAB is widely used in control courses. A new control toolbox, Control101 is under development; this uses livescripts with appropriate code and virtual laboratories to explain the theory while running examples and visualisation. Students can use these files interactively to investigate the effect of simple modifications and to support systematic design.

This paper has summarised a number of recent contributions to the toolbox with a focus on discrete control algorithms. Livescript files have been developed for interactive learning of discrete PID control, dead beat control, Smith predictor control and Youla parameterization are presented here. The paper has highlighted concisely the associated algorithms and the associated files the students can use to analyse and compare the behaviour of the algorithms even in case of big dead time considering also plant/model mismatch. These tools can make the learning process both interactive and enjoyable. It is expected that the students will benefit from using these interactive tools in learning basic control theory.

REFERENCES

[1] Rossiter, J.A., Serbezov, A., Visioli, A., Zakova, K. and Huba, M., A survey of international views on a first course in systems and control for engineering undergraduates, IFAC Journal of Systems and Control, Vol. 13, Article 100092, 15 pages, 2020. https://doi.org/10.1016/j.ifacsc.2020.100092

[2] Rossiter, J.A., B. Pasik-Duncan, S. Dormido, L. Vlacic, B. Jones, and R. Murray, 2018, Good Practice in Control Education, European Journal of Engineering Education, http://dx.doi.org/10.1080/03043797.2018.1428530 .

[3] Rossiter,J.A., Christos G. Cassandras, João Hespanha, Sebastian Dormido , Luis de la Torre, Gireeja Ranade , Antonio Visioli, John Hedengren, Richard M. Murray, Panos Antsaklis, Francoise Lamnabhi-Lagarrigue, Thomas Parisini, 2023, Control education for societal-scale challenges: A community roadmap, Annual Reviews in Control, 55 (2023), pp. 1-17.

[4] Murray, R.M., Waydo, S., Cremean, L. and Mabuchi, H., 2004, A new approach to teaching feedback, IEEE Control Systems Magazine, 24, 38-42

[5] Guzman, J.L., Costa-Castello, R., Berenguel, M. and Dormido, S., 2023, Automatic control with interactive tools, Springer. https://doi.org/10.1007/978-3-031- 09920-5

[6] Douglas, B., 2022, Resourcium, https://resourcium.org/

[7] Serbezov, A., Zakova, K., Visioli, A., Rossiter, J.A., Douglas, B. and Hedengren, J., 2022, Open access resources to support the first course in feedback, dynamics and control, IFAC Symposium on Advances in Control Education.

[8] Rossiter, J.A., 2021, Modelling, dynamics and control website, http://controleducation.group.shef.ac.uk/mainindex.html

[9] Rossiter, J.A., 2022, MATLAB apps to support the learning and understanding of simple system dynamics, IFAC Symposium on Advances in Control Education (ACE,2022)

[10] Rossiter, J.A., 2024, A novel MATLAB toolbox for Control101 courses, European Journal of Control, 2024. https://doi.org/10.1016/j.ejcon.2024.101041

[11] Rossiter, J.A., Visioli, A., Dormido, S. and Bars, R., 2024, A MATLAB virtual laboratory to support learning of auto-tuning PID approaches, 4th IFAC Conference on Advances in Proportional-Integral-Derivative Control.

[12] Keviczky, L., R. Bars, J. Hetthéssy and Cs. Bányász, 2019, Control Engineering and Control Engineering: MATLAB Exercises, Springer.

[13] Smith, O.J. 1957, Closer Control of Loops with Dead Time, Chemical Engineering Progress, 53. pp. 217-219.