Korisničko sučelje za simulaciju upravljanja sustavom toplinskog izmjenjivača

Poparić, Hrvoje

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

KORISNIČKO SUČELJE ZA SIMULACIJU UPRAVLJANJA SUSTAVOM TOPLINSKOG IZMJENJIVAČA

Završni rad

Hrvoje Poparić

Osijek, 2017.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 06.09.2017.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada

Ime i prezime studenta:	Hrvoje Poparić
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3688, 23.07.2014.
OIB studenta:	82083424725
Mentor:	Doc.dr.sc. Emmanuel Karlo Nyarko
Sumentor:	
Sumentor iz tvrtke:	
Naslov završnog rada:	Korisničko sučelje za simulaciju upravljanja sustavom toplinskog izmjenjivača
Znanstvena grana rada:	**Procesno računarstvo (zn. polje računarstvo)**
Predložena ocjena završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	06.09.2017.
Datum potvrde ocjene Odbora:	11.09.2017.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 28.09.2017.

Ime i prezime studenta:	Hrvoje Poparić
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3688, 23.07.2014.
Ephorus podudaranje [%]:	5%

Ovom izjavom izjavljujem da je rad pod nazivom**: Korisničko sučelje za simulaciju upravljanja sustavom toplinskog izmjenjivača**

izrađen pod vodstvom mentora Doc.dr.sc. Emmanuel Karlo Nyarko

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

Sadržaj

1. UVOD

Izmjenjivač topline je postrojenje namijenjeno prijelazu topline s jednog medija (fluida) na drugi. Izvedba može biti različita, pri čemu se u pravilu mediji ne miješaju nego su odvojeni pregradom koja sprječava njihov izravni kontakt. Najčešći je tzv. cijevni izmjenjivač topline. Cilj ovog završnog rada je napraviti korisničko sučelje pomoću kojeg će korisnik moći unijeti parametre procesa na temelju kojih će se projektirati sustav upravljanja i simulirati njegov rad. U drugom poglavlju se daje opis procesa na temelju kojeg se određuje nelinearni matematički model te se određuje prijenosna funkcija procesa koja se koristi u postupku sinteze regulatora. U trećem poglavlju se izrađuje blokovska shema modela sustava upravljanja u *Simulinku* koji se koristi prilikom simulacije te se izrađuje korisničko sučelje. U četvrtom poglavlju se nakon unosa parametara provodi simulacija sustava upravljanja.

1.1. Zadatak završnog rada

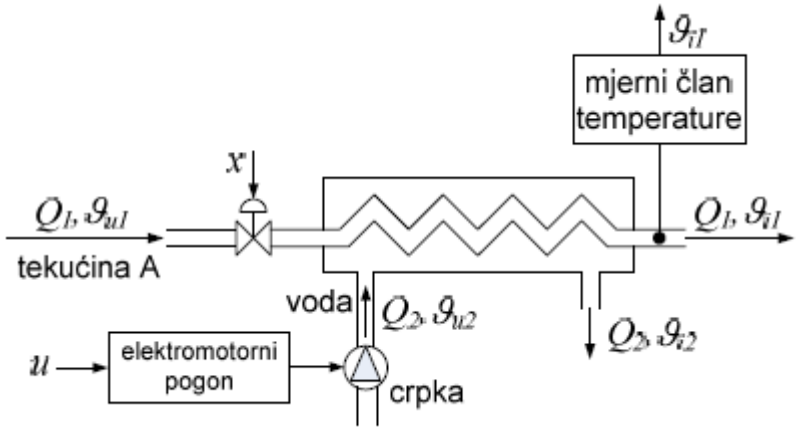
Treba napraviti korisničko sučelje za sustav upravljanja modelom toplinskog izmjenjivača u *MATLAB*-u. Za nadzor treba implementirati prikladnu animaciju koja prikazuje temperature tekućina u cijevima izmjenjivača.

2. ANALIZA PROCESA I SINTEZA SUSTAVA UPRAVLJANJA

U ovom poglavlju se daje opis procesa na temelju kojeg se određuje nelinearni matematički model te se određuje prijenosna funkcija procesa koja se koristi u postupku sinteze regulatora.

2.1. Opis procesa

Na slici 2.1. prikazan je toplinski izmjenjivač čiji je zadatak regulacija temperature ϑ_{i1} tekućine A pomoću zagrijane vode koja struji kroz toplinski izmjenjivač i preko stijenke cijevi predaje toplinu tekućini.



Sl. 2.1. *Shema postrojenja toplinskog izmjenjivača* [1]

Prijenos topline s ogrjevnog medija na grijani opisan je jednadžbama:

$$\rho_1 c_1 V_1 \frac{d\vartheta_{i1}}{dt} = \rho_1 c_1 Q_1 \vartheta_{u1} - \rho_1 c_1 Q_1 \vartheta_{i1} + \alpha A(\vartheta_{i2} - \vartheta_{i1}) \qquad (2-1)$$

gdje je:

- ϑ_{u1} – temperatura tekućine A na ulazu u toplinski izmjenjivač,
- ϑ_{i1} – temperatura tekućine A na izlazu iz toplinskog izmjenjivača,
- ϑ_{i2} – temperatura vode na izlazu iz toplinskog izmjenjivača,
- ρ_1 – gustoća tekućine A,
- c_1 – specifični toplinski kapacitet tekućine A,

- V_1 – unutarnji volumen cijevi kojom tekućina A protječe kroz toplinski izmjenjivač,

- Q_1 – protok tekućine A

- α – efektivni koeficijent prijenosa topline,

- A – efektivna površina izmjene topline u izmjenjivaču

$$\rho_2 c_2 V_2 \frac{d\vartheta_{i2}}{dt} = \rho_2 c_2 Q_2 \vartheta_{u2} - \rho_2 c_2 Q_2 \vartheta_{i2} + \alpha A(\vartheta_{i2} - \vartheta_{i1}) \qquad (2-2)$$

gdje je:

- ϑ_{u2} – temperatura vode na ulazu u toplinski izmjenjivač,

- ρ_2 – gustoća vode,

- c_2 – specifični toplinski kapacitet vode,

- V_2 – volumen vode u toplinskom izmjenjivaču,

- Q_2 – protok vode

Kako bi se izmjena topline kontinuirano odvijala potrebno je osigurati protok tekućine. Protok vode Q_2 osigurava crpka pogonjena asinkronim elektromotorom. Brzinom vrtnje elektromotora upravlja se pomoću frekvencijskog pretvornika. Ovisnost protoka Q_2 o upravljačkom signalu *u* opisana je jednadžbom:

$$T_c \frac{dQ_2}{dt} = -Q_2 + K_c u \qquad (2-3)$$

gdje je:

- K_c – pojačanje elektromotornog pogona i crpke,

- T_c – vremenska konstanta elektromotornog pogona i crpke,

- u – upravljački signal

Protok Q_1 tekućine A mijenja se u ovisnosti o otvorenosti regulacijskog ventila *x* ugrađenog na dovodnoj cijevi koji predstavlja poremećajnu veličinu za proces zagrijavanja tekućine. Ovisnost protoka Q_1 o otvorenosti ventila *x* izraženoj u postocima opisana je jednadžbom:

$$Q_1 = K_v x \qquad (2-4)$$

gdje je:

- K_v – pojačanje ventila,

- x – otvorenost ventila

2.2. Nelinearni matematički model procesa

Uvrštavanjem jednadžbe (2-4) u jednadžbu (2-1) i sređivanjem dobije se:

$$\frac{d\vartheta_{i1}}{dt} = \frac{1}{V_1}\left(\vartheta_{u1}K_v x - K_v x\vartheta_{i1} + \frac{\alpha A}{\rho_1 c_1}(\vartheta_{i2} - \vartheta_{i1})\right) \qquad (2-5)$$

Sređivanjem jednadžbe (2-2) dobije se:

$$\frac{d\vartheta_{i2}}{dt} = \frac{1}{V_2}\left(\vartheta_{u2}Q_2 - Q_2\vartheta_{i2} + \frac{\alpha A}{\rho_2 c_2}(\vartheta_{i2} - \vartheta_{i1})\right) \qquad (2-6)$$

Sređivanjem jednadžbe (2-3) dobije se:

$$\frac{dQ_2}{dt} = \frac{1}{T_c}(-Q_2 + K_c u) \qquad (2-7)$$

Jednadžbe (2-5), (2-6) i (2-7) predstavljaju nelinearni matematički model procesa koji se može prikazati pomoću blokovske sheme u *Simulinku*. Primjer parametara procesa i radne točke nalazi se na slici 2.2.

Parametri procesa	
ϑ_{u1} [ºC]	15
ϑ_{u2} [ºC]	73
ρ_1 [kg/m^3]	800
ρ_2 [kg/m^3]	1000
c_1 [J/kgK]	270
c_2 [J/kgK]	4200
V_1 [m^3]	0,004
V_2 [m^3]	0,007
α [W/m^{2}K]	71
A [m^2]	2,6
K_v [m^3/s]	6,0·10^{-6}
K_c [m^3/Vs]	3,5·10^{-5}
T_c [s]	0,54
Parametri radne točke	
ϑ_{i10} [ºC]	36
x_0 [%]	38
Promjena napona na pogonskom motoru Δu [V]	0,2401

Sl. 2.2. *Primjer parametara procesa i radne točke* [1]

Budući da nisu poznati svi parametri radne točke (ϑ_{i20}, Q_{20}, u_0), potrebno je doći do izraza pomoću kojih se mogu izračunati. Parametri radne točke se određuju tako da se diferencijalne jednadžbe koje opisuju model procesa ((2-5), (2-6), (2-7)) izjednače s 0 i umjesto svih vremenski promjenjivih veličina uvrste njihove vrijednosti u radnoj točki:

$$\frac{d\vartheta_{i1}}{dt} = 0 = \frac{1}{V_1}\left(\vartheta_{u1}K_v x_0 - K_v x_0 \vartheta_{i10} + \frac{\alpha A}{\rho_1 c_1}(\vartheta_{i20} - \vartheta_{i10})\right) \qquad (2-8)$$

$$\frac{d\vartheta_{i2}}{dt} = 0 = \frac{1}{V_2}\left(\vartheta_{u2}Q_{20} - Q_{20}\vartheta_{i20} + \frac{\alpha A}{\rho_2 c_2}(\vartheta_{i20} - \vartheta_{i10})\right) \qquad (2-9)$$

$$\frac{dQ_2}{dt} = 0 = \frac{1}{T_c}(-Q_{20} + K_c u_0) \qquad (2-10)$$

Sređivanjem jednadžbe (2-8) dobije se:

$$\vartheta_{i20} = \vartheta_{i10} + \frac{\rho_1 c_1 K_v x_0}{\alpha A}(\vartheta_{i10} - \vartheta_{u1}) \qquad (2-11)$$

Sređivanjem jednadžbe (2-9) dobije se:

$$Q_{20} = \frac{\alpha A(\vartheta_{i20} - \vartheta_{i10})}{\rho_2 c_2(\vartheta_{u2} - \vartheta_{i20})} \qquad (2-12)$$

Sređivanjem jednadžbe (2-10) dobije se:

$$u_0 = \frac{Q_{20}}{K_c} \qquad (2-13)$$

Pomoću jednadžbi (2-11), (2-12) i (2-13) mogu se izračunati preostali parametri radne točke.

2.3. Prijenosna funkcija procesa

Prijenosna funkcija procesa $G(s)$ predstavlja omjer izlazne i ulazne veličine u Laplaceovom području ($\vartheta_{i1}(s)/U(s)$). Kako bi se do nje došlo potrebno je najprije linearizirati sve nelinearne diferencijalne jednadžbe koje se nalaze u modelu. Linearizacija se provodi kroz razvoj nelinearnih diferencijalnih jednadžbi u Taylorove redove (uzimaju se samo članovi prvoga reda) u okolini radne točke koji sadrže promjene veličina oko radne točke.

Linearizacijom jednadžbe (2-5) dobije se:

$$\frac{d\Delta\vartheta_{i1}}{dt} = \frac{1}{V_1}\left(K_v(\vartheta_{u1} - \vartheta_{i10})\Delta x - \left(K_v x_0 + \frac{\alpha A}{\rho_1 c_1}\right)\Delta\vartheta_{i1} + \frac{\alpha A}{\rho_1 c_1}\Delta\vartheta_{i2}\right) \quad (2-14)$$

Linearizacijom jednadžbe (2-6) dobije se:

$$\frac{d\Delta\vartheta_{i2}}{dt} = \frac{1}{V_2}\left((\vartheta_{u2} - \vartheta_{i20})\Delta Q_2 + \frac{\alpha A}{\rho_2 c_2}\Delta\vartheta_{i1} - \left(Q_{20} + \frac{\alpha A}{\rho_2 c_2}\right)\Delta\vartheta_{i2}\right) \quad (2-15)$$

Linearizacijom jednadžbe (2-7) dobije se:

$$\frac{d\Delta Q_2}{dt} = \frac{1}{T_c}(-\Delta Q_2 + K_c\Delta u) \quad (2-16)$$

Nakon linearizacije potrebno je odrediti Laplaceove transformacije lineariziranih jednadžbi.

Laplaceova transformacija jednadžbe (2-14):

$$V_1 s\vartheta_{i1}(s) = K_v(\vartheta_{u1} - \vartheta_{i10})X(s) - \left(K_v x_0 + \frac{\alpha A}{\rho_1 c_1}\right)\vartheta_{i1}(s) + \frac{\alpha A}{\rho_1 c_1}\vartheta_{i2}(s) \quad (2-17)$$

Laplaceova transformacija jednadžbe (2-15):

$$V_2 s\vartheta_{i2}(s) = (\vartheta_{u2} - \vartheta_{i20})Q_2(s) + \frac{\alpha A}{\rho_2 c_2}\vartheta_{i1}(s) - \left(Q_{20} + \frac{\alpha A}{\rho_2 c_2}\right)\vartheta_{i2}(s) \quad (2-18)$$

Laplaceova transformacija jednadžbe (2-16):

$$T_c s Q_2(s) = -Q_2(s) + K_c U(s) \quad (2-19)$$

Sređivanjem jednadžbe (2-17) dobije se:

$$\vartheta_{i1}(s)\left(K_v x_0 + \frac{\alpha A}{\rho_1 c_1} + V_1 s\right) = \frac{\alpha A}{\rho_1 c_1}\vartheta_{i2}(s) + K_v(\vartheta_{u1} - \vartheta_{i10})X(s) \quad (2-20)$$

Sređivanjem jednadžbe (2-18) dobije se:

$$\vartheta_{i2}(s)\left(Q_{20} + \frac{\alpha A}{\rho_2 c_2} + V_2 s\right) = \frac{\alpha A}{\rho_2 c_2}\vartheta_{i1}(s) + (\vartheta_{u2} - \vartheta_{i20})Q_2(s) \quad (2-21)$$

Sređivanjem jednadžbe (2-19) dobije se:

$$Q_2(s) = \frac{K_c}{1 + T_c s}U(s) \quad (2-22)$$

Uvrštavanjem jednadžbe (2-22) u jednadžbu (2-21) dobije se:

$$\vartheta_{i2}(s)\left(Q_{20}+\frac{\alpha A}{\rho_2 c_2}+V_2 s\right)=\frac{\alpha A}{\rho_2 c_2}\vartheta_{i1}(s)+\frac{K_c(\vartheta_{u2}-\vartheta_{i20})}{1+T_c s}U(s) \qquad (2-23)$$

Rješavanjem sustava jednadžbi (2-20) i (2-23) dobije se:

$$\vartheta_{i1}(s)=\frac{\frac{\alpha A K_c(\vartheta_{u2}-\vartheta_{i20})}{\rho_1 c_1(1+T_c s)}}{\left(K_v x_0+\frac{\alpha A}{\rho_1 c_1}+V_1 s\right)\left(Q_{20}+\frac{\alpha A}{\rho_2 c_2}+V_2 s\right)-\frac{\alpha^2 A^2}{\rho_1 c_1 \rho_2 c_2}}U(s)$$

$$+\frac{K_v(\vartheta_{u1}-\vartheta_{i10})\left(Q_{20}+\frac{\alpha A}{\rho_2 c_2}+V_2 s\right)}{\left(K_v x_0+\frac{\alpha A}{\rho_1 c_1}+V_1 s\right)\left(Q_{20}+\frac{\alpha A}{\rho_2 c_2}+V_2 s\right)-\frac{\alpha^2 A^2}{\rho_1 c_1 \rho_2 c_2}}X(s) \qquad (2-24)$$

Omjer $\vartheta_{i1}(s)/U(s)$, odnosno prijenosna funkcija procesa se dobije tako da se zanemari poremećaj $X(s)$. Uvrštavanjem $X(s)=0$ u jednadžbu (2-24) i sređivanjem dobije se prijenosna funkcija procesa:

$$G(s)=\frac{\alpha A K_c(\vartheta_{u2}-\vartheta_{i20})}{\rho_1 c_1(1+T_c s)\left(\left(K_v x_0+\frac{\alpha A}{\rho_1 c_1}+V_1 s\right)\left(Q_{20}+\frac{\alpha A}{\rho_2 c_2}+V_2 s\right)-\frac{\alpha^2 A^2}{\rho_1 c_1 \rho_2 c_2}\right)} \qquad (2-25)$$

2.4. Sinteza PID regulatora pomoću krivulje mjesta korijena

Krivulja mjesta korijena je grafički postupak pomoću kojeg se na temelju prijenosne funkcije otvorenog regulacijskog kruga mogu odrediti položaji polova zatvorenog regulacijskog kruga za različita pojačanja regulacijskog kruga. U okviru ovog završnog rada krivulja mjesta korijena će se provesti pozivanjem odgovarajuće *MATLAB* funkcije.

Prijenosna funkcija realnog PID regulatora:

$$G_R(s)=K_P+\frac{K_I}{s}+K_D\frac{T_v s}{1+T_v s}=K\left(1+\frac{1}{T_I s}+T_D\frac{s}{1+T_v s}\right)$$

$$=K\frac{T_I(T_D+T_v)s^2+(T_I+T_v)s+1}{T_I s(1+T_v s)}=K_R\frac{(s-s_{n1})(s-s_{n2})}{s(1+T_v s)} \qquad (2-26)$$

gdje je:

- K_P – pojačanje proporcionalnog djelovanja,

- K_I – pojačanje integracijskog djelovanja,

- K_D – pojačanje derivacijskog djelovanja,

- T_I – integracijska vremenska konstanta,

- T_D – derivacijska vremenska konstanta,

- T_v – parazitna vremenska konstanta,

- K_R – pojačanje regulatora,

- s_{n1} – prva nula regulatora,

- s_{n2} – druga nula regulatora

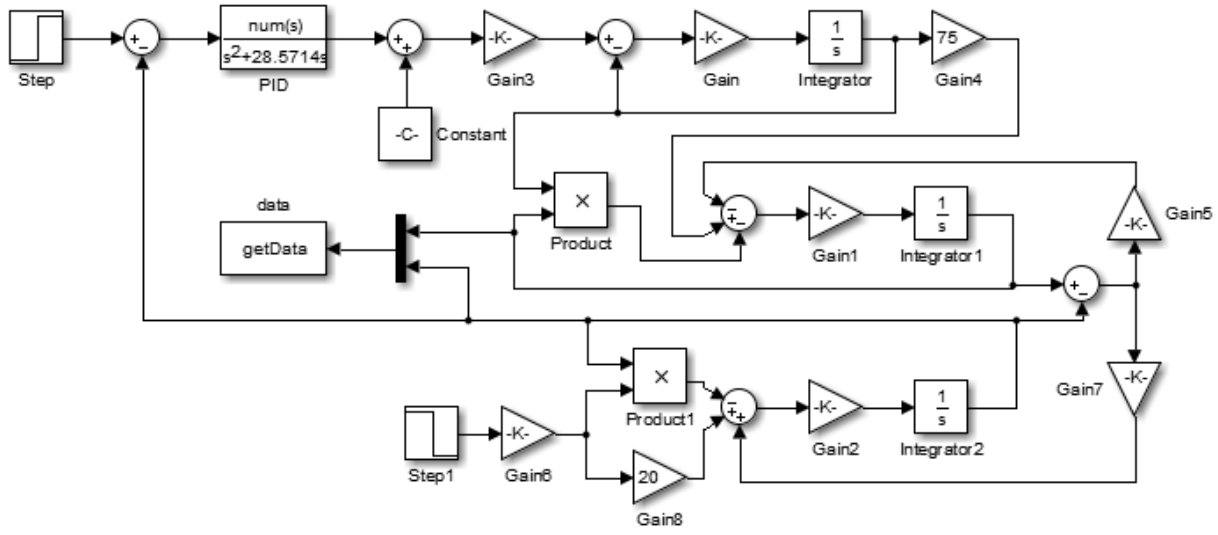
Iz jednadžbe (2-26) je vidljivo da realni PID regulator ima 2 nule i 2 pola od kojih se jedan pol nalazi u ishodištu s-ravnine. U okviru ovog završnog rada PID regulator će se projektirati tako da će njegova jedna nula biti postavljena na mjestu najdominantnijeg pola procesa ulijevo od imaginarne osi s-ravnine, druga nula na mjestu pola koji je najdalje od imaginarne osi, a pol na 20 puta veću udaljenost od udaljenosti druge nule regulatora od imaginarne osi. Na temelju nacrtane krivulje mjesta korijena će se odrediti pojačanje regulatora tako da relativni koeficijent prigušenja dominantnih polova zatvorenog regulacijskog kruga bude između 0,6 i 0,8.

3. KORISNIČKO SUČELJE

Unutar ovog poglavlja se izrađuje blokovska shema modela sustava upravljanja u *Simulinku* koji se koristi prilikom simulacije te se izrađuje korisničko sučelje.

3.1. Simulink model

Simulacija upravljanja sustavom toplinskog izmjenjivača će se provesti na nelinearnom modelu prikazanom pomoću blokovske sheme u *Simulinku* (slika 3.1.) koji se izrađuje na temelju jednadžbi koje opisuju proces ((2-5), (2-6), (2-7)).



Sl. 3.1. *Shema nelinearnog modela sustava upravljanja toplinskim izmjenjivačem*

Parametri pojedinih blokova modela će se postaviti na vrijednosti određene parametrima procesa i radne točke koje će korisnik unositi preko korisničkog sučelja.

3.2. Dizajn

Korisničko sučelje moguće je napraviti pomoću *MATLAB*-ovog razvojnog okruženja *GUIDE*. Sadrži:

- 16 *Static Text*-ova koji će sadržavati nazive parametara procesa i radne točke koje korisnik unosi,

- 16 *Edit Text*-ova u koje će korisnik unositi vrijednosti pojedinih parametara procesa i radne točke,

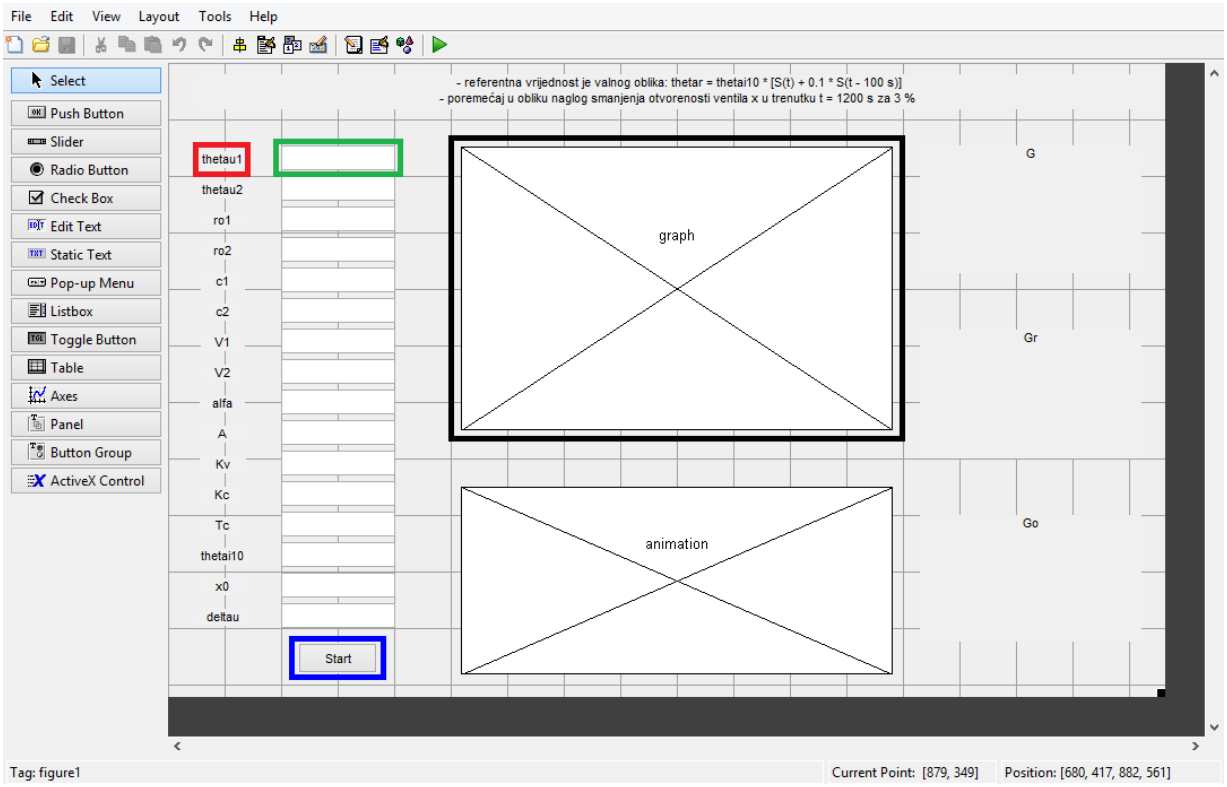
- 1 *Static Text* koji će sadržavati uvjete u kojima se provodi simulacija,

- 1 *Push Button* kojim će se pokrenuti simulacija,

- 3 *Static Text*-a koji će sadržavati prijenosne funkcije procesa, regulatora i otvorenog regulacijskog kruga,

- 1 *Axes* u kojem će se prikazati odziv izlazne veličine ϑ_{i1},

- 1 *Axes* u kojem će se prikazati temperature tekućina u toplinskom izmjenjivaču prikazanog pomoću obojanih pravokutnika



Sl. 3.2. *Dizajn korisničkog sučelja u GUIDE-u*

Na slici 3.2. je crvenom bojom označen *Static Text*, zelenom bojom *Edit Text*, plavom bojom *Push Button* i crnom bojom *Axes*.

GUIDE automatski kreira *MATLAB* kod za konstruiranje korisničkog sučelja koji se može modificirati kako bi se isprogramiralo ponašanje korisničkog sučelja. Neke od funkcija koje se automatski kreiraju: *ui_OpeningFcn, thetau1_Callback, thetau2_Callback, ro1_Callback, ro2_Callback...*

3.3. MATLAB kod

Potrebno je modificirati funkciju *ui_OpeningFcn* koja se izvršava prije nego što korisničko sučelje postane vidljivo.

Na početku se definiraju 4 varijable kao globalne što znači da su dostupne svim funkcijama u kojima postoji njihova globalna definicija. U varijable *r1*, *r2* i *r3* će biti spremljeni pravokutnici koji će predstavljati cijevi toplinskog izmjenjivača, a u varijablu *c* će biti spremljena matrica čiji redci predstavljaju boje definirane intenzitetom crvene, zelene i plave (RGB zapis). Funkcija *open_system* otvara *Simulink* model kako bi se moglo pristupati parametrima njegovih blokova. (listing 3.1.)

```
% --- Executes just before ui is made visible.
function ui_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ui (see VARARGIN)

global r1 r2 r3 c;

open_system('model');
```

Listing 3.1.

Handles je struktura pomoću koje se može pristupiti svim elementima korisničkog sučelja. Također se u nju mogu spremati nove varijable. U ovoj funkciji se u *handles* spremaju zastavice (*flags*) za svaki od 16 parametara procesa i radne točke koje korisnik unosi i postavljaju na 0. Služit će za provjeru ispravnosti unosa (0 - neispravan, 1 - ispravan). (listing 3.2.)

```
handles.thetau1Flag = 0;
handles.thetau2Flag = 0;
handles.ro1Flag = 0;
handles.ro2Flag = 0;
handles.c1Flag = 0;
handles.c2Flag = 0;
handles.V1Flag = 0;
handles.V2Flag = 0;
handles.alfaFlag = 0;
handles.AFlag = 0;
handles.KvFlag = 0;
handles.KcFlag = 0;
handles.TcFlag = 0;
handles.thetai10Flag = 0;
handles.x0Flag = 0;
handles.deltauFlag = 0;
```

Listing 3.2.

Potom se definira naslov, naziv x-osi i naziv y-osi elementa *graph* korisničkog sučelja u kojem će se prikazati odziv izlazne veličine ϑ_{i1}. (listing 3.3.)

```
handles.graph.Title.String = 'Odziv izlazne veličine';
handles.graph.XLabel.String = 't [s]';
handles.graph.YLabel.String = 'thetai1 [°C]';
```

Listing 3.3.

Nakon tog se stvaraju pravokutnici s definiranim pozicijama unutar elementa *animation* korisničkog sučelja, matrica boja pomoću funkcije *colormap* i skala boja za temperature između 0 i 100 °C pomoću funkcije *colorbar*. (listing 3.4.)

```
axis off;
r1 = rectangle('Position', [0 0 1 0.4]);
r2 = rectangle('Position', [0 0.4 1 0.2]);
r3 = rectangle('Position', [0 0.6 1 0.4]);
c = colormap(jet);
colorbar('TickLabels', {'0', '20', '40', '60', '80', '100'});
```

Listing 3.4.

Na kraju se pomoću funkcije *guidata* spremaju sve promjene nad *handles* strukturom. (listing 3.5.)

```
% Choose default command line output for ui
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes ui wait for user response (see UIRESUME)
% uiwait(handles.figure1);
```

Listing 3.5.

GUIDE automatski kreira *callback* funkcije koje se izvršavaju kada dođe do nekog događaja nad elementom korisničkog sučelja (npr. promjena sadržaja *Edit Text*-a, pritisak na *Push Button* itd.).

Na listingu 3.6. se nalazi *callback* funkcija za jedan od 16 *Edit Text*-ova u koje korisnik unosi vrijednosti pojedinih parametara procesa i radne točke. U njoj je potrebno provjeriti ispravnost korisnikovog unosa. Ako je unesen pozitivan broj, onda je unos ispravan i zastavica za taj parametar se postavlja na 1, u suprotnom se postavlja na 0. Na kraju se pomoću funkcije *guidata* sprema promjena nad *handles* strukturom. Analogno je potrebno provjeriti vrijednosti preostalih parametara.

```
function thetau1_Callback(hObject, eventdata, handles)
% hObject    handle to thetau1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of thetau1 as text
%        str2double(get(hObject,'String')) returns contents of thetau1 as a
%        double

if str2double(handles.thetau1.String) > 0
    handles.thetau1Flag = 1;
else
    handles.thetau1Flag = 0;
end
guidata(hObject, handles);
```

Listing 3.6.

Pritiskom na *Push Button* (*start*) potrebno je izračunati i postaviti parametre pojedinih blokova *Simulink* modela na temelju korisnikovih unosa.

Na početku funkcije *start_Callback* se provjerava jesu li sve zastavice jednake 1, odnosno je li korisnik ispravno unio sve vrijednosti parametara procesa i radne točke. Ako jesu, onda će se izvršiti ostatak funkcije, u suprotnom se neće. (listing 3.7.)

```
% --- Executes on button press in start.
function start_Callback(hObject, eventdata, handles)
% hObject    handle to start (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

if handles.thetau1Flag == 1 && handles.thetau2Flag == 1 && handles.ro1Flag
== 1 && handles.ro2Flag == 1 && handles.c1Flag == 1 && handles.c2Flag == 1
&& handles.V1Flag == 1 && handles.V2Flag == 1 && handles.alfaFlag == 1 &&
handles.AFlag == 1 && handles.KvFlag == 1 && handles.KcFlag == 1 &&
handles.TcFlag == 1 && handles.thetai10Flag == 1 && handles.x0Flag == 1 &&
handles.deltauFlag == 1
```

Listing 3.7.

Nakon toga se definiraju dvije varijable kao globalne. U varijablu *st* se sprema element korisničkog sučelja *start* kako bi mu se kasnije moglo pristupiti izvan ove funkcije. Nakon tog se briše krivulja s elementa korisničkog sučelja *graph* (ukoliko se već izvršila neka simulacija) te se u varijablu *l* sprema objekt *line* pomoću kojeg će se prikazati odziv izlazne veličine ϑ_{i1}. (listing 3.8.)

```
global st l;
st = handles.start;
axes(handles.graph);
delete(handles.graph.Children);
l = line(NaN, NaN);
```

Listing 3.8.

Potom se onemogućava pritisak na *Push Button* dok se ne završi simulacija. Također se svi korisnikovi unosi spremaju u lokalne varijable kako bi im se kasnije jednostavnije pristupilo. (listing 3.9.)

```
st.Enable = 'off';
st.String = 'Simulating...';

thetau1 = str2double(handles.thetau1.String);
thetau2 = str2double(handles.thetau2.String);
ro1 = str2double(handles.ro1.String);
ro2 = str2double(handles.ro2.String);
c1 = str2double(handles.c1.String);
c2 = str2double(handles.c2.String);
V1 = str2double(handles.V1.String);
V2 = str2double(handles.V2.String);
alfa = str2double(handles.alfa.String);
A = str2double(handles.A.String);
Kv = str2double(handles.Kv.String);
Kc = str2double(handles.Kc.String);
Tc = str2double(handles.Tc.String);
thetai10 = str2double(handles.thetai10.String);
x0 = str2double(handles.x0.String);
deltau = str2double(handles.deltau.String);
```

Listing 3.9.

Donja granica y-osi elementa korisničkog sučelja *graph* se postavlja na vrijednost izlazne veličine u radnoj točki (ϑ_{i10}), a gornja granica na 15 % te vrijednosti budući da je jedan od uvjeta u kojima se provodi simulacija povećanje izlazne veličine za 10 %. (listing 3.10.)

```
handles.graph.YLim = [thetai10 1.15*thetai10];
```

Listing 3.10.

Nakon tog se računaju nepoznati parametri radne točke, prijenosna funkcija procesa $G(s)$ i prijenosna funkcija otvorenog regulacijskog kruga $G_O(s)$. Postupak krivulje mjesta korijena se provodi pomoću funkcije *rlocus* koja vraća položaje polova zatvorenog regulacijskog kruga i pripadna pojačanja regulatora. Pomoću funkcije *find* moguće je pronaći pol za koji će relativni koeficijent prigušenja dominantnih polova zatvorenog regulacijskog kruga biti između 0,6 i 0,8 te preko njega pripadno pojačanje regulatora. (listing 3.11.)

```
thetai20 = thetai10+(ro1*c1*Kv*x0*(thetai10-thetau1))/(alfa*A);
Q20 = (alfa*A*(thetai20-thetai10))/(ro2*c2*(thetau2-thetai20));
u0 = Q20/Kc;

s = tf('s');
G = (alfa*A*Kc*(thetau2-
thetai20))/(ro1*c1*(1+Tc*s)*((Kv*x0+(alfa*A)/(ro1*c1)+V1*s)*(Q20+(alfa*A)/(
ro2*c2)+V2*s)-(alfa^2*A^2)/(ro1*c1*ro2*c2)));
[z, p, k] = zpkdata(G, 'v');
Go = k/(s*(s-20*p(1))*(s-p(2)));
[poles, gains] = rlocus(Go);
i = find(abs(real(poles))./sqrt((real(poles)).^2+(imag(poles)).^2) > 0.6 &
abs(real(poles))./sqrt((real(poles)).^2+(imag(poles)).^2) < 0.8, 1);
Kr = gains(ceil(i/3));
```

Listing 3.11.

Potom se u lokalne varijable spremaju *zero-pole-gain* zapisi prijenosne funkcije procesa $G(s)$, prijenosne funkcije regulatora $G_R(s)$ i prijenosne funkcije otvorenog regulacijskog kruga $G_O(s)$ te se prikazuju na korisničkom sučelju pomoću funkcije *evalc*. (listing 3.12.)

```
G = zpk(G);
Gr = zpk((Kr*(s-p(1))*(s-p(3)))/(s*(s-20*p(1))));
Go = zpk(Kr*Go);
handles.G.String = evalc('G');
handles.Gr.String = evalc('Gr');
handles.Go.String = evalc('Go');
```

Listing 3.12.

Na kraju se pomoću funkcije *set_param* postave parametri blokova *Simulink* modela na prethodno izračunate vrijednosti i pokrene se simulacija. (listing 3.13.)

```
set_param('model/PID', 'Denominator', strcat('[',num2str([1 -20*p(1)
0]),']'));
set_param('model/PID', 'Numerator', strcat('[',num2str(Kr*[1 -(p(1)+p(3))
p(1)*p(3)]),']'));
set_param('model/Step', 'Before', num2str(thetai10));
set_param('model/Step', 'After', num2str(1.1*thetai10));
set_param('model/Constant', 'Value', num2str(u0));
set_param('model/Gain3', 'Gain', num2str(Kc));
set_param('model/Gain', 'Gain', num2str(1/Tc));
set_param('model/Integrator', 'InitialCondition', num2str(Q20));
set_param('model/Gain4', 'Gain', num2str(thetau2));
set_param('model/Gain1', 'Gain', num2str(1/V2));
set_param('model/Integrator1', 'InitialCondition', num2str(thetai20));
set_param('model/Gain5', 'Gain', num2str((alfa*A)/(ro2*c2)));
set_param('model/Gain7', 'Gain', num2str((alfa*A)/(ro1*c1)));
set_param('model/Step1', 'Before', num2str(x0));
set_param('model/Step1', 'After', num2str(x0-3));
set_param('model/Gain6', 'Gain', num2str(Kv));
set_param('model/Gain8', 'Gain', num2str(thetau1));
set_param('model/Gain2', 'Gain', num2str(1/V1));
set_param('model/Integrator2', 'InitialCondition', num2str(thetai10));
set_param('model', 'SimulationCommand', 'start');

end
```

Listing 3.13.

Za prikaz odziva izlazne veličine ϑ_{i1} i temperatura tekućina u toplinskom izmjenjivaču potrebno je znati vrijednosti veličina ϑ_{i1} i ϑ_{i2} u svakom trenutku simulacije. U tu svrhu moguće je koristiti *S-Function* blok čiji su ulazi veličine ϑ_{i1} i ϑ_{i2} (slika 3.1., blok *data*) i koji povezuje *Simulink* model sa S-funkcijom (*getData*) u kojoj je onda moguće koristiti veličine iz *Simulink* modela.

Argumenti S-funkcije:

- *t* – trenutno vrijeme,

- *x* – vektor stanja (neće se koristiti),

- *u* – ulazni vektor (sadrži trenutne vrijednosti veličina ϑ_{i1} i ϑ_{i2}),

- *flag* – cjelobrojna vrijednost koja određuje što će se izvršiti u S-funkciji

Ako je *flag* jednak 0, onda je potrebno izvršiti inicijalizaciju karakteristika bloka. Sve vrijednosti varijabli su unaprijed zadane i ne treba ih mijenjati, osim varijable *NumInputs* koju je potrebno postaviti na 2 budući da ulazni vektor sadrži 2 veličine (ϑ_{i1} i ϑ_{i2}). (listing 3.14.)

```
function [sys,x0,str,ts] = getData(t,x,u,flag)
global r1 r2 r3 c st l;
switch flag,
    case 0,
        sizes = simsizes;
        sizes.NumContStates = 0;
        sizes.NumDiscStates = 0;
        sizes.NumOutputs = 0;
        sizes.NumInputs = 2;
        sizes.DirFeedthrough = 0;
        sizes.NumSampleTimes = 1;
        sys = simsizes(sizes);
        x0 = [];
        str = [];
        ts = [0 0];
```

Listing 3.14.

Ako je *flag* jednak 2, onda je potrebno prikazati trenutne vrijednosti veličina ϑ_{i1} i ϑ_{i2}. Prvo se pomoću funkcije *get* dohvate prethodne x i y vrijednosti objekta *l* te im se dodaju trenutno vrijeme i trenutna vrijednost izlazne veličine ϑ_{i1} čiji se odziv prikazuje. Ako je trenutno vrijeme između 100 i 200 sekundi ili između 1200 i 1300 sekundi, onda je potrebno usporiti simulaciju budući da tada dolazi do znatnijih promjena veličina ϑ_{i1} i ϑ_{i2}. Nakon tog je potrebno pomoću funkcije *set* postaviti x i y vrijednosti objekta *l* na prethodno dodane vrijednosti. (listing 3.15.)

```
case 2,
        time = [get(l, 'XData') t];
        thetai1 = [get(l, 'YData') u(2)];
        if (t >= 100 && t <= 200) || (t >= 1200 && t <= 1300)
            pause(0.01);
        end
        set(l, 'XData', time, 'YData', thetai1);
```

Listing 3.15.

Bojanje pravokutnika se provodi tako da se boje matrice *c* raspodijele na temperature između 0 i 100 °C (matrica boja *c* sadrži 64 boje) te se na temelju trenutnih vrijednosti veličina ϑ_{i1} i ϑ_{i2} odrede boje pravokutnika. (listing 3.16.)

```
r1.FaceColor = [c(ceil((u(1)*64)/100),1) c(ceil((u(1)*64)/100),2)
c(ceil((u(1)*64)/100),3)];
r2.FaceColor = [c(ceil((u(2)*64)/100),1) c(ceil((u(2)*64)/100),2)
c(ceil((u(2)*64)/100),3)];
r3.FaceColor = [c(ceil((u(1)*64)/100),1) c(ceil((u(1)*64)/100),2)
c(ceil((u(1)*64)/100),3)];
```

Listing 3.16.

Ako je *flag* jednak 9, onda je simulacija završila te je potrebno omogućiti pritisak na *Push Button* kako bi se mogla pokrenuti nova simulacija. (listing 3.17.)

```
case 9,
        st.String = 'Start';
        st.Enable = 'on';
end
```

Listing 3.17.

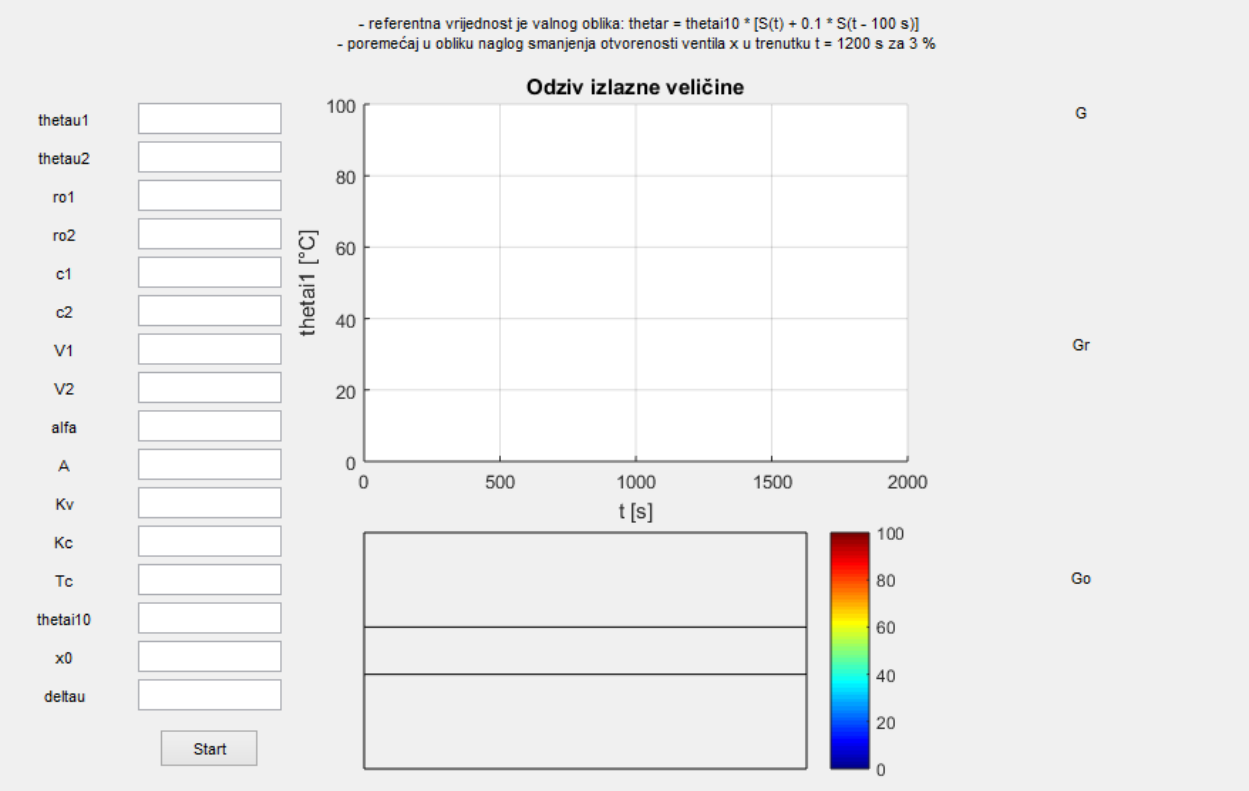
4. SIMULACIJA

U ovom poglavlju se nakon unosa parametara provodi simulacija sustava upravljanja.

Uvjeti u kojima se provodi simulacija:

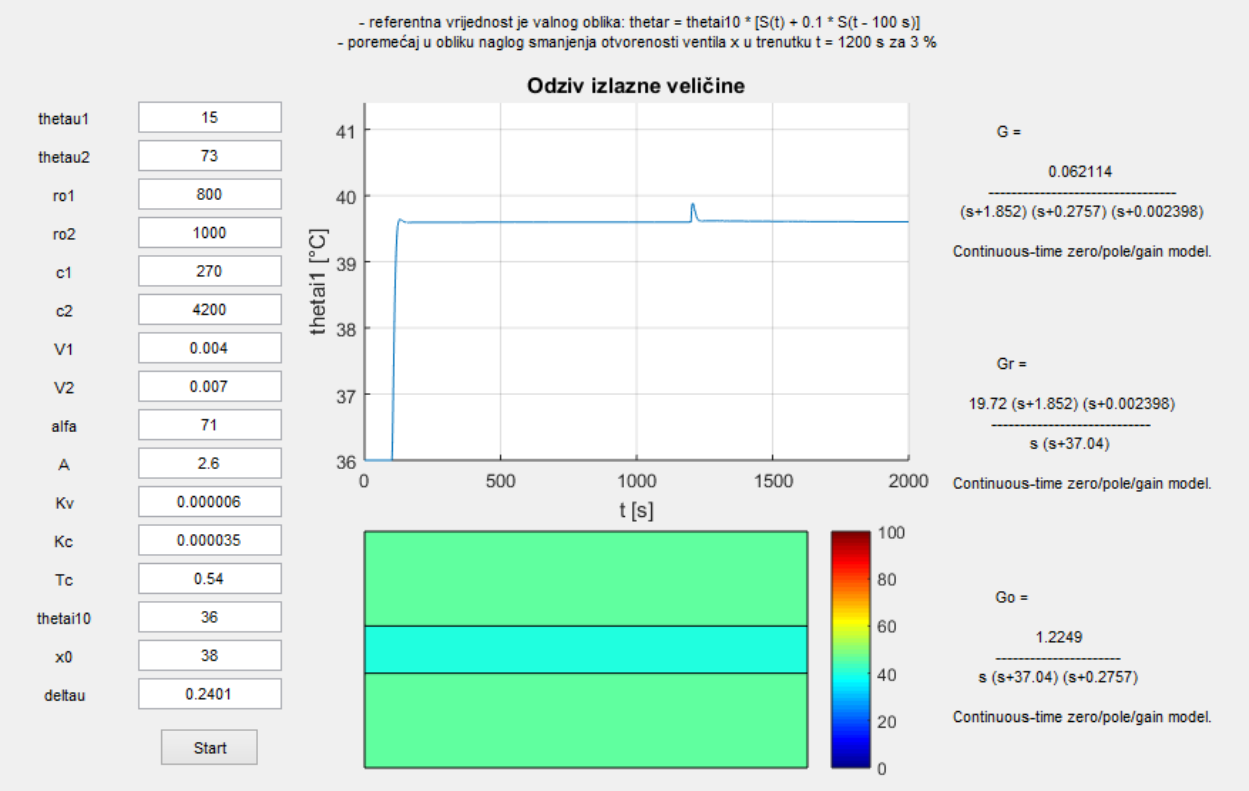
- referentna vrijednost je valnog oblika: $\vartheta_r = \vartheta_{i10}[S(t) + 0{,}1S(t - 100\ \text{s})]$,

- poremećaj u obliku naglog smanjenja otvorenosti ventila x u trenutku t = 1200 s za 3 %



Sl. 4.1. *Korisničko sučelje*

Unosom parametara procesa i radne točke sa slike 2.2. te pokretanjem simulacije dobije se:



Sl. 4.2. *Rezultati simulacije*

Na slici 4.2. je vidljivo da se s projektiranim PID regulatorom postiže kratko vrijeme porasta izlazne veličine ϑ_{i1} te brza kompenzacija poremećaja i točnost u stacionarnom stanju. Također se može uočiti da boje pravokutnika odgovaraju konačnim temperaturama tekućina u toplinskom izmjenjivaču.

5. ZAKLJUČAK

Rezultati simulacije pokazuju da je korisničko sučelje funkcionalno, što znači da se na temelju korisnikovog unosa parametara procesa može odrediti model procesa, projektirati sustav upravljanja te pomoću simulacije pokazati valjanost sustava upravljanja toplinskim izmjenjivačem. Na temelju dobivenih rezultata moguće je projektirati sustav upravljanja u područjima u kojima se koristi toplinski izmjenjivač (npr. kućni grijači, hladnjaci, automobilski rashladnici, industrijski izmjenjivači itd.).

Literatura

[1] E. K. Nyarko, R. Grbić, D. Slišković, R. Cupec, Osnove automatskog upravljanja, Elektrotehnički fakultet Osijek, Osijek, 2015.

[2] https://www.mathworks.com/products/simulink.html

[3] https://www.mathworks.com/discovery/matlab-gui.html

[4] https://www.mathworks.com/help/simulink/sfg/maintaining-level-1-matlab-s-functions.html

Sažetak

U završnom radu je napravljeno korisničko sučelje za sustav upravljanja modelom toplinskog izmjenjivača u *MATLAB*-u. Započinje opisom procesa pomoću diferencijalnih jednadžbi. Zatim slijedi projektiranje PID regulatora pomoću krivulje mjesta korijena. Nakon toga slijedi izrada *Simulink* modela i programiranje korisničkog sučelja. Rad završava simulacijom sustava upravljanja i prikazom rezultata koji pokazuju da je korisničko sučelje funkcionalno.

Ključne riječi: korisničko sučelje, *MATLAB*, model, proces, simulacija, *Simulink*, sustav upravljanja

Graphical user interface for the simulation of a heat exchanger control system

A user interface for the simulation and control of a *MATLAB* model of a heat exchanger is developed and described in this thesis. Initially, the mathematical model of the process is described with the aid of nonlinear differential equations. A PID controller is then designed using the root locus method. A *Simulink* model of the process is created and a user interface designed. Finally, the functionality of the user interface is shown by running a simulation of the control system.

Keywords: control system, *MATLAB*, model, process, simulation, *Simulink*, user interface

Životopis

Hrvoje Poparić rođen je 27.11.1995. godine u Osijeku. Pohađao je III. gimnaziju u Osijeku. Tijekom srednjoškolskog obrazovanja je sudjelovao na županijskim natjecanjima iz fizike. Trenutno je student 3. godine računarstva na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija Osijek.