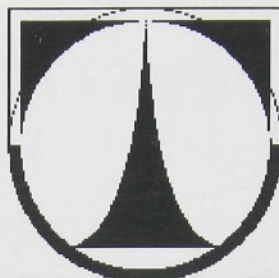


Technická univerzita Liberec

Fakulta strojní

Katedra technické kybernetiky



DIPLOMOVÁ PRÁCE

Jan Podzimek

DIPLOMOVÁ PRÁCE

**Jednoduché algoritmy řízení soustav s regulačním
ventilem s elektrickým pohonem**

Vedoucí diplomové práce: Doc. Ing. Oswald Modrák, CSc.

Vypísal: Jan Podzimek

Liberec 1996

1996

Technická univerzita Liberec

Fakulta strojní

Katedra technické kybernetiky

DIPLOMOVÁ PRÁCE

**Jednoduché algoritmy řízení soustav s regulačním
ventilem s elektrickým pohonem**

Vedoucí diplomové práce: Doc. Ing. Osvald Modrlák, Csc.

Vypracoval: Jan Podzimek

Liberec 1996

ZADÁNÍ DIPLOMOVÉ PRÁCE

pro Jana PODZIMKA

obor 23-40-8 Automatizované systémy řízení ve strojírenství

Vedoucí katedry Vám ve smyslu zákona č. 172/1990 Sb. o vysokých školách určuje tuto diplomovou práci:

Název tématu:

JEDNODUCHÉ ALGORITMY ŘÍZENÍ SOUSTAV S REGULAČNÍM VENTILEM
S ELEKTRICKÝM POHONEM

Zásady pro vypracování:

Navrhněte:

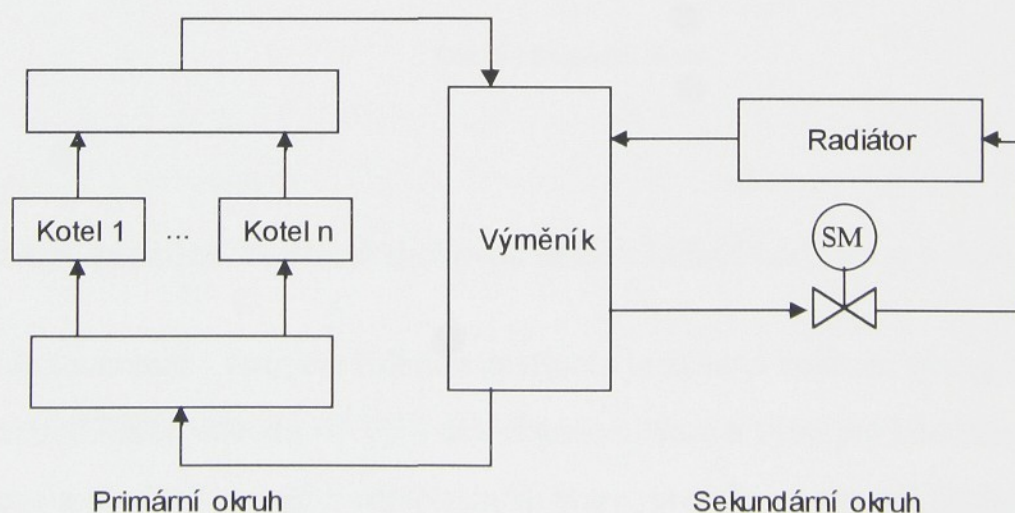
1. Algoritmické řízení s třípolohovým a dvoupolohovým relé s hysterezí.
2. Algoritmus pracující na principu přepínacích přímek.
3. Algoritmus řízení pracující na jiném principu než je uvedeno pod bodem 1. a 2.

Obsah

1. Úvod.....	5
2. Regulační obvod	7
3. Algoritmické řízení s dvupolohovým a třípolohovým relé s hysterezí	9
3.1. Relé s pevnou zpětnou vazbou	11
3.2. Relé se zpožďující zpětnou vazbou	11
3.3. Relé s pružnou zpětnou vazbou.....	12
3.4. Seřízení zpětných vazeb relé.....	14
3.5. Příklady regulace sekundárního okruhu vytápění	18
3.5.1. Seřízení regulátoru	19
3.5.2. Regulace s použitím relé s pevnou zpětnou vazbou	21
3.5.3. Regulace s použitím relé se zpožďující zpětnou vazbou.....	24
3.5.4. Regulace s použitím relé s pružnou zpětnou vazbou	27
4. Algoritmus pracující na principu přepínacích přímek.....	29
4.1. Dynamické chování nelineárních systémů ve stavovém prostoru	29
4.2. Zpětná vazba tachometrická	32
4.3. Zavedení derivace do regulační odchylky	32
4.4. Příklady regulace sekundárního obvodu vytápění	36
5. Algoritmus generující doby zapnutí či vypnutí	39
5.1. Příklady regulace sekundárního okruhu vytápění	41
6. Program RTSO	43
6.1. Popis programu.....	43
6.2. Popis základních tříd programu	44
6.3. Popis tříd realizujících bloky regulačního obvodu.....	48
6.4. Popis implementace algoritmů řízení	52
6.5. Ovládání programu	59
7. Závěr	62
8. Literatura.....	64

1. Úvod

Topný obvod se skládá z primárního a sekundárního okruhu. Tato diplomová práce, zadaná firmou AUTRON z Jablonce nad Nisou, se zabývá metodami řízení regulačního ventilu v sekundární části topného okruhu. Primární okruh je tvořen několika kotly, které ohřívají vodu proudící do výměníku. Teplota vody v primárním okruhu je pro většinu aplikací udržována na konstantní hodnotě v povoleném teplotním pásmu. Podle kolísání této teploty řídící automat buď startuje či odstavuje jednotlivé kotle, nebo snižuje jejich výkon. Ve výměníku je primárním okruhem ohřívána voda jednotlivých sekundárních okruhů vytápění budovy. Každý sekundární obvod má svůj regulační ventil se servopohonem. Servopohon ventilu je řízen mikropočítačovým řídícím automatem, který reguluje otevření ventilu na základě výstupní teploty sekundárního okruhu a žádané teploty. V řídícím automatu je naprogramován v paměti EPROM řídící algoritmus, který je prováděn mikroprocesorem INTEL 80196.



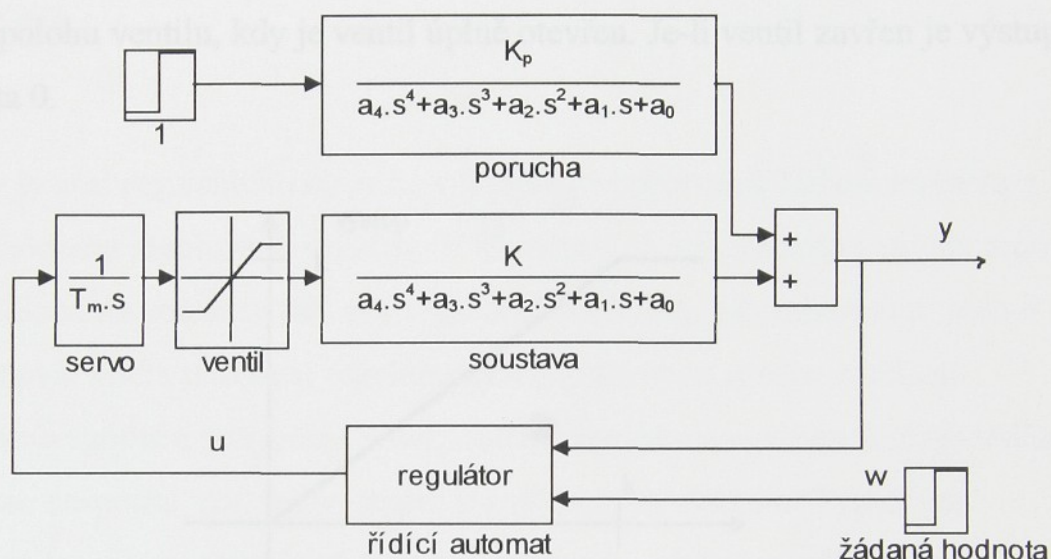
Obr. 1-1 Topný obvod s jedním z m sekundárních okruhů

Podle zadání je úkolem této diplomové práce navrhnout algoritmy řízení řídícího automatu sekundárního obvodu vytápění pracujících na principu:

- A pro uvedené algoritmy naprogramovat simulační prostředek v jazyce C++, který bude pro zvolenou soustavu simulovat řízení sekundárního okruhu vytápění.

2. Regulační obvod

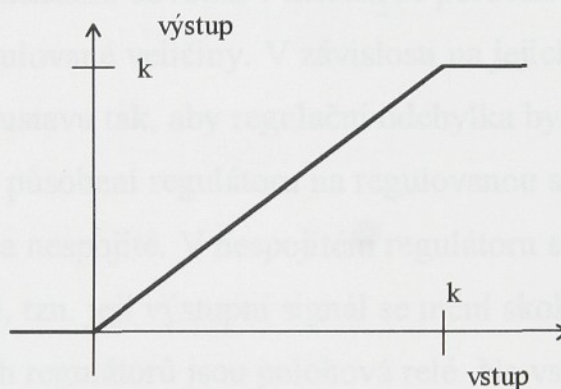
Schéma regulačního obvodu sekundárního okruhu vytápění se skládá z následujících bloků:



Obr. 2-1 Regulační obvod

- **žádaná hodnota** - realizuje skokovou změnu žádané hodnoty w v čase $t = 0$.
- **řídící automat** - vstupem řídicího automatu je žádaná hodnota výstupní teploty sekundárního obvodu $w[^\circ\text{C}]$ v celočíselném tvaru a výstupní hodnota regulované soustavy $y[^\circ\text{C}]$ rovněž v celočíselném tvaru, protože na vstup řídicího automatu lze přivést pouze celočíselnou hodnotu. V řídicím automatu je naprogramován algoritmus řízení. Výstupem je hodnota u , která nabývá hodnot 1, 0, -1, a určuje zda servopohon bude regulační ventil otevírat, zavírat či bude v klidu.

- servopohon - servopohon je realizován přenosem $F_s(s) = \frac{1}{T_s \cdot s}$, jde v podstatě o integrátor s časovou konstantou T_s , který integruje vstupní hodnotu u . Výstupem je poloha ventilu. Servopohon reaguje na změnu vstupního signálu až po uplynutí určitého času od poslední změny vstupního signálu.
- ventil - ventil je realizován blokem nasycení jehož parametrem je hodnota k , představující nejvyšší možnou výstupní hodnotu. V praxi hodnota k představuje krajní polohu ventilu, kdy je ventil úplně otevřen. Je-li ventil zavřen je výstupní hodnota 0.



Obr. 2-2 Přenos bloku ventil

- soustava - v praxi se jedná o složité soustavy vyšších řádů zahrnující potrubí a radiátor v sekundárním obvodu. Pro potřeby této diplomové práce byla po dohodě se zadavatelem zvolena soustava 4. řádu s přenosem

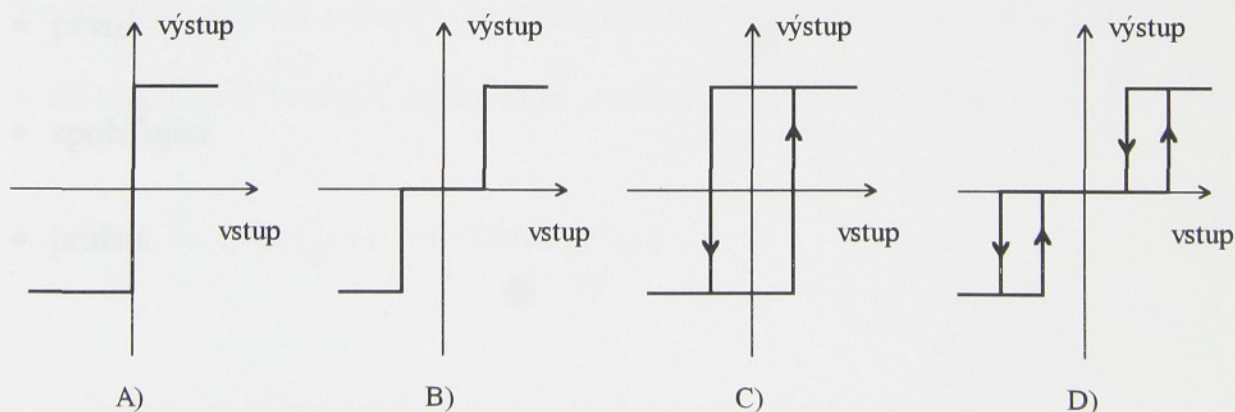
$$S(s) = \frac{K}{a_4 \cdot s^4 + a_3 \cdot s^3 + a_2 \cdot s^2 + a_1 \cdot s + a_0}$$
. Výstupní veličinou soustavy je teplota sekundárního okruhu vytápění $y[^\circ\text{C}]$.

- porucha - v praxi může porucha do obvodu vstupovat prakticky kdekoliv. V tomto případě má porucha přenos $S_p(s) = \frac{K_p}{a_4 \cdot s^4 + a_3 \cdot s^3 + a_2 \cdot s^2 + a_1 \cdot s + a_0}$ a přičítá se k výstupní veličině regulačního obvodu y .

Regulační procesy uvedené v kapitolách 3. a 4. byly simulovány v prostředí MATLAB 4.2. s využitím toolboxu Simulink. Zdrojové texty simulačních obvodů jsou na přiložené disketě v adresáři MATLAB. V kapitole 5. byl pro simulaci použit program RTSO s upraveným výstupem dat, protože realizace simulačního obvodu Matlabu by byla náročná, jelikož Matlab neobsahuje potřebné bloky.

3. Algoritmické řízení s dvupolohovým a třípolohovým relé s hysterezí

Regulátor je část regulačního obvodu, v kterém se porovnává žádaná hodnota a skutečná hodnota regulované veličiny. V závislosti na jejich rozdílu působí regulátor akční veličinou na soustavu tak, aby regulační odchylka byla minimální, pokud možno nulová. Podle působení regulátoru na regulovanou soustavu dělíme regulátory na spojité a nespojité. V nespojitém regulátoru se alespoň jedna veličina mění v čase nespojitě, tzn. její výstupní signál se mění skokem. Typickým příkladem nespojitých regulátorů jsou polohová relé. Na vstup relé je přiváděn spojitý signál, výstup relé se mění skokem, je nespojitý.



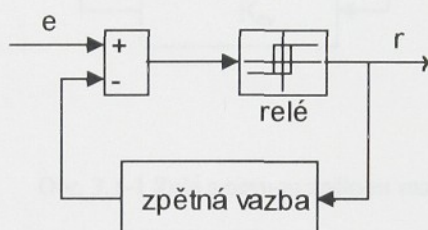
Obr. 3-1 A) ideální dvupolohové relé, B) relé s pásmem necitlivosti (třípolohové relé), C) dvupolohové relé s hysterezí, D) třípolohové relé s hysterezí

Nespojité regulátory lze rozdělit do dvou skupin:

- nespojité regulátory bez zpětné vazby

- nespojité regulátory se zpětnou vazbou, neboli impulsové

K nespojitým regulátorům bez zpětné vazby patří dvupolohové a třípolohové relé. Použití samotného relé k řízení soustav s velkými časovými konstantami nebo s velkým časovým zpožděním není vhodné, protože dochází k velkým překmitům regulované veličiny a regulace není kvalitní. Zapojením vhodných soustav do zpětné vazby relé vznikne impulsový regulátor, s nímž lze dosáhnout lepších výsledků regulace. Relé se zpětnou vazbou se svojí funkcí více přibližují spojitým regulátorům.



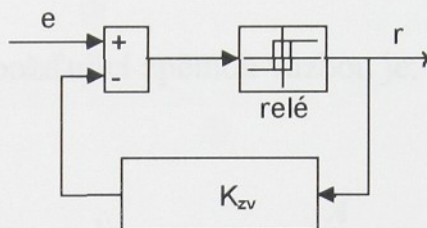
Obr. 3-2 Relé se zpětnou vazbou

Podle dynamických vlastností dělíme zpětné vazby na:

- pevné
- zpožďující
- pružné

3.1. Relé s pevnou zpětnou vazbou

Zapojením pevné zpětné vazby relé se zesílením K_{zv} získáme impulsový regulátor, s nímž je regulační proces podobný jako s proporcionálním regulátorem P. Přenos proporcionálního regulátoru je $R_p = r_0$.



Obr. 3.1-1 Relé s pevnou zpětnou vazbou

Přenos relé s pevnou zpětnou vazbou je následující:

$$F_{re} = \lim_{A \rightarrow \infty} \frac{-A}{1 + K_{zv} \cdot (-A)} = \lim_{A \rightarrow \infty} \frac{1}{\frac{-1}{A} + K_{zv}} = \frac{1}{K_{zv}}$$

Z tohoto přenosu a z přenosu proporcionálního regulátoru plyne vztah mezi zesílením zpětné vazby K a zesílením proporcionálního regulátoru $K_{zv} = \frac{1}{r_0}$.

3.2. Relé se zpožďující zpětnou vazbou

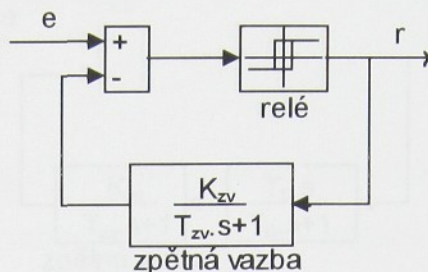
Zapojíme-li do zpětné vazby relé soustavu prvního řádu s přenosem

$$F(s) = \frac{K_{zv}}{T_{zv} \cdot s + 1},$$

bude mít výsledný regulační proces charakteristiku regulace s PD

regulátorem. PD regulátor má přenos $R_{PD}(s) = r_0 + r_2 \cdot s = r_0 \cdot \left[1 + \frac{r_2}{r_0} \cdot s \right]$, kde r_0 je

proporcionální složka regulátoru a r_2 derivační složka regulátoru.



Obr. 3.2-1 Relé se zpožďující zpětnou vazbou

Výsledný přenos relé se zpožďující zpětnou vazbou je:

$$F_{re} = \lim_{A \rightarrow \infty} \frac{-A}{1 + \frac{K_{zv}}{T_{zv} \cdot s + 1} \cdot (-A)} = \lim_{A \rightarrow \infty} \frac{A}{(T_{zv} \cdot s + 1) - A \cdot K_{zv}} \cdot (T_{zv} \cdot s + 1) =$$

$$= \lim_{A \rightarrow \infty} \frac{-(T_{zv} \cdot s + 1)}{\frac{T_{zv} \cdot s + 1}{A} - K_{zv}} = \frac{1}{K_{zv}} \cdot (T_{zv} \cdot s + 1)$$

Z uvedených vztahů plyne:

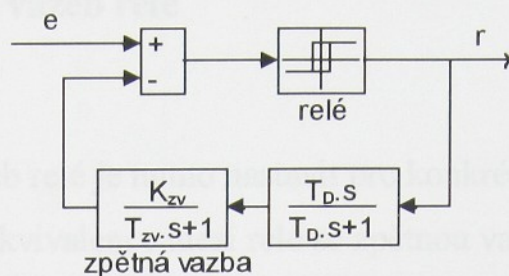
$$K_{zv} = \frac{1}{r_0} \quad , \quad T_{zv} = \frac{r_2}{r_0} \quad .$$

3.3. Relé s pružnou zpětnou vazbou

Má-li zpětná vazba relé přenos $F(s) = \frac{K_{zv}}{T_{zv} \cdot s + 1} \cdot \frac{T_D \cdot s}{T_D \cdot s + 1}$, má výsledný přenos

obvodu charakteristiku PID regulátoru. PID regulátor je charakterizován přenosem

$$R_{PID}(s) = r_0 + \frac{r_1}{s} + r_2 \cdot s = r_0 \cdot \left[1 + \frac{r_1}{r_0 \cdot s} + \frac{r_2 \cdot s}{r_0} \right].$$



Obr. 3.3-1 Relé s pružnou zpětnou vazbou

Výsledný přenos relé s pružnou zpětnou vazbou je:

$$\begin{aligned}
 F_{re}(s) &= \lim_{A \rightarrow \infty} \frac{-A}{1 + \frac{K_{zv}}{T_{zv} \cdot s + 1} \cdot \frac{T_D \cdot s}{T_D \cdot s + 1}} = \lim_{A \rightarrow \infty} \frac{-A}{\frac{(T_{zv} \cdot s + 1) \cdot (T_D \cdot s + 1) - K_{zv} \cdot T_D \cdot A \cdot s}{(T_{zv} \cdot s + 1) \cdot (T_D \cdot s + 1)}} = \\
 &= \lim_{A \rightarrow \infty} \frac{-(T_{zv} \cdot s + 1) \cdot (T_D \cdot s + 1)}{\frac{(T_{zv} \cdot s + 1) \cdot (T_D \cdot s + 1) - K_{zv} \cdot T_D \cdot s}{A}} = \frac{(T_{zv} \cdot s + 1) \cdot (T_D \cdot s + 1)}{K_{zv} \cdot T_D \cdot s} = \\
 &= \frac{T_{zv} \cdot T_D \cdot s^2 + (T_{zv} + T_D) \cdot s + 1}{K_{zv} \cdot T_D \cdot s} = \frac{T_{zv} + T_D}{K_{zv} \cdot T_D} + \frac{1}{K_{zv} \cdot T_D \cdot s} + \frac{T_{zv} \cdot T_D}{K_{zv} \cdot T_D} \cdot s = \\
 &= \frac{T_{zv} + T_D}{K_{zv} \cdot T_D} \cdot \left[1 + \frac{1}{s \cdot (T_{zv} + T_D)} + \frac{T_{zv} \cdot T_D \cdot s}{(T_{zv} + T_D)} \right]
 \end{aligned}$$

Z uvedených vztahů plyne:

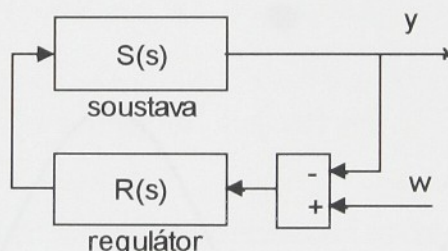
$$K_{zv} = \frac{r_0 \pm \sqrt{r_0^2 - 4 \cdot r_1 \cdot r_2}}{2 \cdot r_1 \cdot r_2}$$

$$T_{zv} = r_2 \cdot K_{zv}$$

$$T_D = \frac{1}{r_1 \cdot K_{zv}}$$

3.4. Seřízení zpětných vazeb relé

Parametry zpětných vazeb relé je nutno nastavit pro konkrétní regulovanou soustavu. Využijeme-li ekvivalence mezi relé se zpětnou vazbou a spojitými regulátory, můžeme seřadit spojitý regulátor P, PD nebo PID a z jeho parametrů spočítat konstanty zpětných vazeb relé. Pro svojí jednoduchost je použita metoda kritického zesílení regulátoru. Princip této metody spočívá v přivedení regulačního obvodu do tzv. kritického stavu, z jeho chování lze pak odvodit optimální parametry regulátoru. V kritickém stavu kmitá obvod na mezi stability s konstantní amplitudou a regulátor má integrační a derivační složku rovnu 0. Hodnota proporcionální složky při které došlo ke kritickému stavu se nazývá kritické zesílení regulátoru r_{ok} a periodě vzniklých kmitů říkáme perioda kritických kmitů T_k .



Obr. 3.4-1 Regulační obvod se spojitým regulátorem

Přenos uzavřeného regulačního obvodu je $F_{yw}(s) = \frac{1}{1 + R(s) \cdot S(s)}$, charakteristická rovnice regulačního obvodu pro soustavu 4. řádu

$$S(s) = \frac{K}{a_4 \cdot s^4 + a_3 \cdot s^3 + a_2 \cdot s^2 + a_1 \cdot s + a_0} \text{ je}$$

$$a_4 \cdot s^4 + a_3 \cdot s^3 + a_2 \cdot s^2 + a_1 \cdot s + a_0 + K \cdot r_0 = 0.$$

Hodnotu kritického zesílení regulátoru vypočteme z charakteristické rovnice obvodu použitím Hurwitzova kritéria.

Z charakteristického polynomu sestojíme Hurwitzovu matici H.

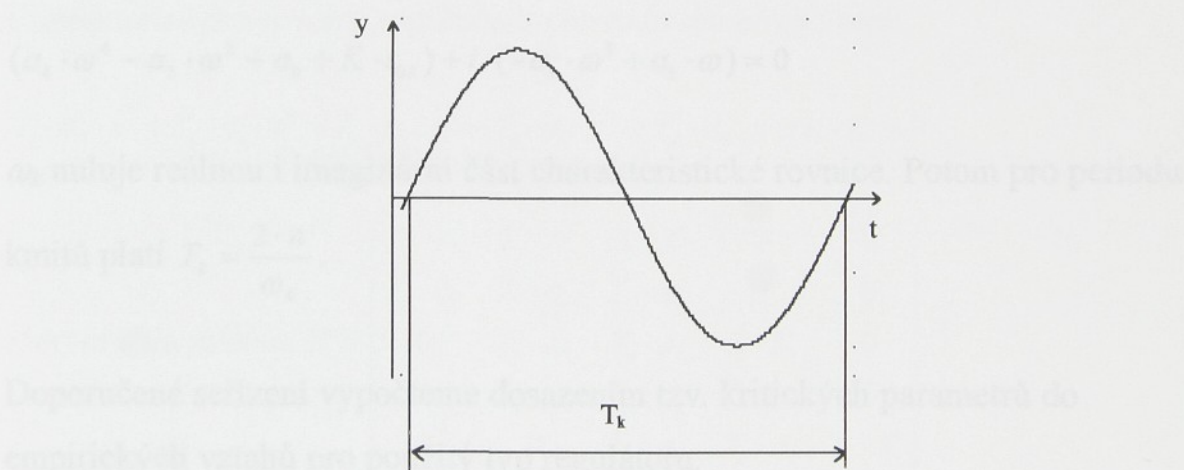
$$H = \begin{bmatrix} a_3 & a_1 & 0 & 0 \\ a_4 & a_2 & a_0 + K \cdot r_0 & 0 \\ 0 & a_3 & a_1 & 0 \\ 0 & a_4 & a_2 & a_0 + K \cdot r_0 \end{bmatrix}$$

Determinant H_3 Hurwitzovy matice se musí rovnat nule, aby byl obvod na mezi stability.

$$H_3 = a_3 \cdot a_2 \cdot a_1 - a_3^2 \cdot (a_0 + r_0 \cdot K) - a_4 \cdot a_1^2 = 0$$

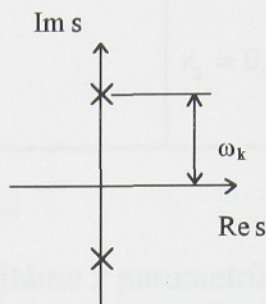
z této rovnice dostaneme kritické zesílení regulátoru r_{0k} .

$$r_{0k} = \frac{a_3 \cdot a_2 \cdot a_1 - a_3^2 \cdot a_0 - a_4 \cdot a_1^2}{a_3^2 \cdot K}$$



Obr. 3.4-2 Obvod na mezi stability kmitá s periodou kritických kmitů T_k

Pro toto kritické zesílení je obvod na mezi stability a kmitá s periodou T_k . Nutně tedy existuje dvojice ryze imaginárních kořenů $s_{1,2} = \pm i \cdot \omega$, ostatní kořeny leží v levé polorovině komplexní roviny.



Obr. 3.4-3. Dvojice ryze imaginárních kořenů

Periodu kritických kmitů určíme dosazením jednoho z těchto kořenů do charakteristické rovnice.

$$a_4 \cdot (i \cdot \omega)^4 + a_3 \cdot (i \cdot \omega)^3 + a_2 \cdot (i \cdot \omega)^2 + a_1 \cdot (i \cdot \omega) + a_0 + K \cdot r_0 = 0$$

$$(a_4 \cdot \omega^4 - a_2 \cdot \omega^2 + a_0 + K \cdot r_{0k}) + i \cdot (-a_3 \cdot \omega^3 + a_1 \cdot \omega) = 0$$

ω_k nuluje reálnou i imaginární část charakteristické rovnice. Potom pro periodu

$$\text{kmitů platí } T_k = \frac{2 \cdot \pi}{\omega_k}.$$

Doporučené seřízení vypočteme dosazením tzv. kritických parametrů do empirických vztahů pro použitý typ regulátoru.

Typ regulátoru	Parametry regulátoru
P	$r_0 = 0,5 \cdot r_{0k}$
PD	$r_0 = 0,4 \cdot r_{0k}$ $r_2 = 0,02 \cdot r_{0k} \cdot T_k$

PID	$r_0 = 0,6 \cdot r_{0k}$ $r_1 = \frac{1,2 \cdot r_{0k}}{T_k}$ $r_2 = 0,075 \cdot r_{0k} \cdot T_k$
-----	--

Podle zvoleného regulátoru spočítáme z parametrů r_0 , r_1 a r_2 konstanty odpovídajících soustav zapojených ve zpětné vazbě relé.

Zahrneme-li do přenosu soustavy i přenos servopohonu s ventilem dostaneme odlišné hodnoty seřízení regulátoru. Přenos soustavy se servopohonem s ventilem je

$$5. \text{ řádu s astatismem } 1. \text{ řádu } S(s) \cdot F_s(s) = \frac{K}{T_s \cdot s \cdot (a_4 \cdot s^4 + a_3 \cdot s^3 + a_2 \cdot s^2 + a_1 \cdot s + a_0)}.$$

Charakteristická rovnice regulačního obvodu má potom tvar:

$$T_s \cdot a_4 \cdot s^5 + T_s \cdot a_3 \cdot s^4 + T_s \cdot a_2 \cdot s^3 + T_s \cdot a_1 \cdot s^2 + T_s \cdot a_0 \cdot s + K \cdot r_0 = 0$$

$$\text{Hurwitzova matice } H = \begin{bmatrix} T_s \cdot a_3 & T_s \cdot a_1 & r_0 \cdot K & 0 & 0 \\ T_s \cdot a_4 & T_s \cdot a_2 & T_s \cdot a_0 & 0 & 0 \\ 0 & T_s \cdot a_3 & T_s \cdot a_1 & r_0 \cdot K & 0 \\ 0 & T_s \cdot a_4 & T_s \cdot a_2 & T_s \cdot a_0 & 0 \\ 0 & 0 & T_s \cdot a_3 & T_s \cdot a_1 & r_0 \cdot K \end{bmatrix}$$

Determinant Hurwitzovy matice H_4 :

$$H_4 = r_0^2 \cdot (-a_4^2 \cdot K^2) + r_0 \cdot (-T_s^3 \cdot a_3 \cdot a_2^2 \cdot K + 2 \cdot T_s^3 \cdot a_4 \cdot a_3 \cdot a_0 \cdot K + T_s^3 \cdot a_4 \cdot a_2 \cdot a_1 \cdot K) + T_s^4 \cdot a_3 \cdot a_2 \cdot a_1 \cdot a_0 - T_s^4 \cdot a_3^2 \cdot a_0^2 - T_s \cdot a_4 \cdot a_1^2 \cdot a_0$$

Kritické zesílení obvodu r_{0k} :

$$r_{0k,2} = \frac{-T_s \cdot \left[a_3 \cdot a_2^2 \cdot K - 2 \cdot a_4 \cdot a_3 \cdot a_0 \cdot K - a_4 \cdot a_2 \cdot a_1 \cdot K \pm K \cdot \sqrt{a_2^2 - 4 \cdot a_4 \cdot a_0} \cdot (a_3 \cdot a_2 - a_4 \cdot a_1) \right]}{2 \cdot a_4^2 \cdot K^2}$$

Charakteristická rovnice po dosazení jednoho z dvojice ryze imaginárních kořenů:

$$a_4 \cdot T_s \cdot (i \cdot \omega)^5 + a_3 \cdot T_s \cdot (i \cdot \omega)^4 + a_2 \cdot T_s \cdot (i \cdot \omega)^3 + a_1 \cdot T_s \cdot (i \cdot \omega)^2 + a_0 \cdot T_s \cdot (i \cdot \omega) + K \cdot r_0 = 0$$

$$(T_s \cdot a_3 \cdot \omega^4 - T_s \cdot a_2 \cdot \omega^2 + K \cdot r_{0k}) + i \cdot T_s \cdot (a_4 \cdot \omega^5 - a_2 \cdot \omega^3 + a_0 \cdot \omega) = 0$$

ω_k nuluje reálnou i imaginární část charakteristické rovnice. Periodu kritických

kmitů regulačního obvodu opět spočítáme ze vztahu $T_k = \frac{2 \cdot \pi}{\omega_k}$. Výpočet

doporučeného seřízení regulátoru se neliší od prvního případu.

3.5. Příklady regulace sekundárního okruhu vytápění

Ukažme si chování regulačního obvodu se všemi typy zpětných vazeb relé seřízených metodou kritického zesílení pro soustavu 4. řádu s obrazovým přenosem

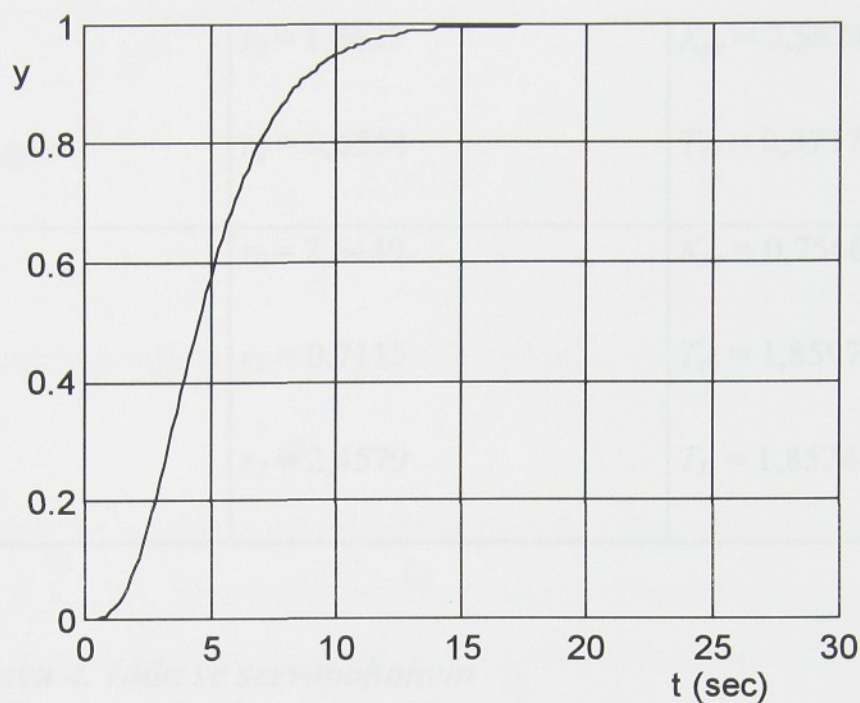
$$S(s) = \frac{1}{2 \cdot s^4 + 7 \cdot s^3 + 9 \cdot s^2 + 5 \cdot s + 1}. \text{ Seřízení regulátoru provedeme pro samotnou}$$

soustavu a pro soustavu se servopohonem a ventilem. Servopohon jehož přenos je

$$F_s(s) = \frac{1}{10 \cdot s}, \text{ reaguje na změnu signálu po 1 sekundě. Konstanta ventilu (poloha}$$

v které je ventil úplně otevřen) $k = 40$.

V každém grafu je znázorněn průběh regulované teploty y v čase a akční zásahy regulátoru (vstupní hodnota servopohonu).



Obr. 3.5-1 Přechodová charakteristika soustavy

3.5.1. Seřízení regulátoru

a) pro samotnou soustavu 4. řádu

Charakteristická rovnice přenosu soustavy je $2 \cdot s^4 + 7 \cdot s^3 + 9 \cdot s^2 + 5 \cdot s + 1 + r_{0k} = 0$.

Z ní vypočítáme kritické zesílení obvodu $r_{0k} = \frac{216}{49} = 4,4082$, a periodu kritických

kmitů $T_k = 7,4343$. Doporučené nastavení regulátorů i odpovídající parametry zpětných vazeb relé jsou v následující tabulce.

Typ regulátoru	Parametry regulátoru	Parametry zpětné vazby relé
P	$r_0 = 2,2041$	$K_{zv} = 0,4537$

PD	$r_0 = 1,7633$ $r_2 = 0,6554$	$K_{zv} = 0,5671$ $T_{zv} = 0,3717$
PID	$r_0 = 2,6449$ $r_1 = 0,7115$ $r_2 = 2,4579$	$K_{zv} = 0,7566$ $T_{zv} = 1,8597$ $T_D = 1,8574$

b) pro soustavu 4. řádu se servopohonem

Je-li přenos servopohonu $F_s = \frac{1}{T_s \cdot s}$ a $T_s = 10$, je charakteristická rovnice přenosu

soustavy $20 \cdot s^5 + 70 \cdot s^4 + 90 \cdot s^3 + 50 \cdot s^2 + 10 \cdot s + r_{0k} = 0$.

Kritické zesílení obvodu $r_{0k} = 4.790$, a perioda kritických kmitů $T_k = 18,6113$.

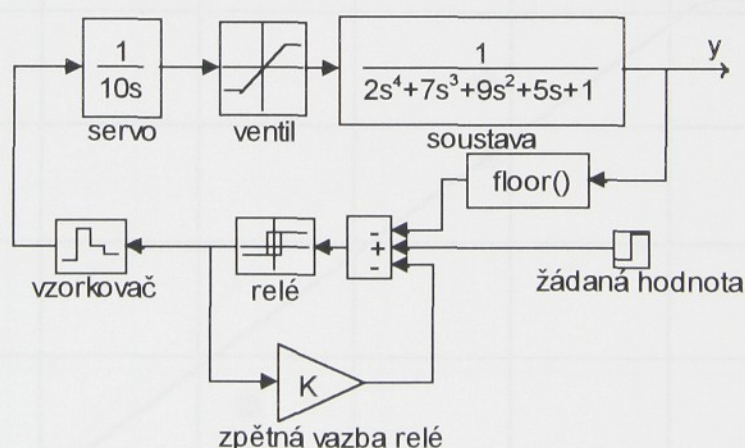
Doporučené nastavení regulátorů i odpovídající parametry zpětných vazeb relé jsou v následující tabulce.

Typ regulátoru	Parametry regulátoru	Parametry zpětné vazby relé
P	$r_0 = 2,3951$	$K_{zv} = 0,4175$
PD	$r_0 = 1,9161$ $r_2 = 1,7828$	$K_{zv} = 0,5219$ $T_{zv} = 0,9305$
PID	$r_0 = 2,8741$	$K_{zv} = 0,6969$

	$r_1 = 0,3089$	$T_{zv} = 4,6592$
	$r_2 = 6,6857$	$T_D = 4,6454$

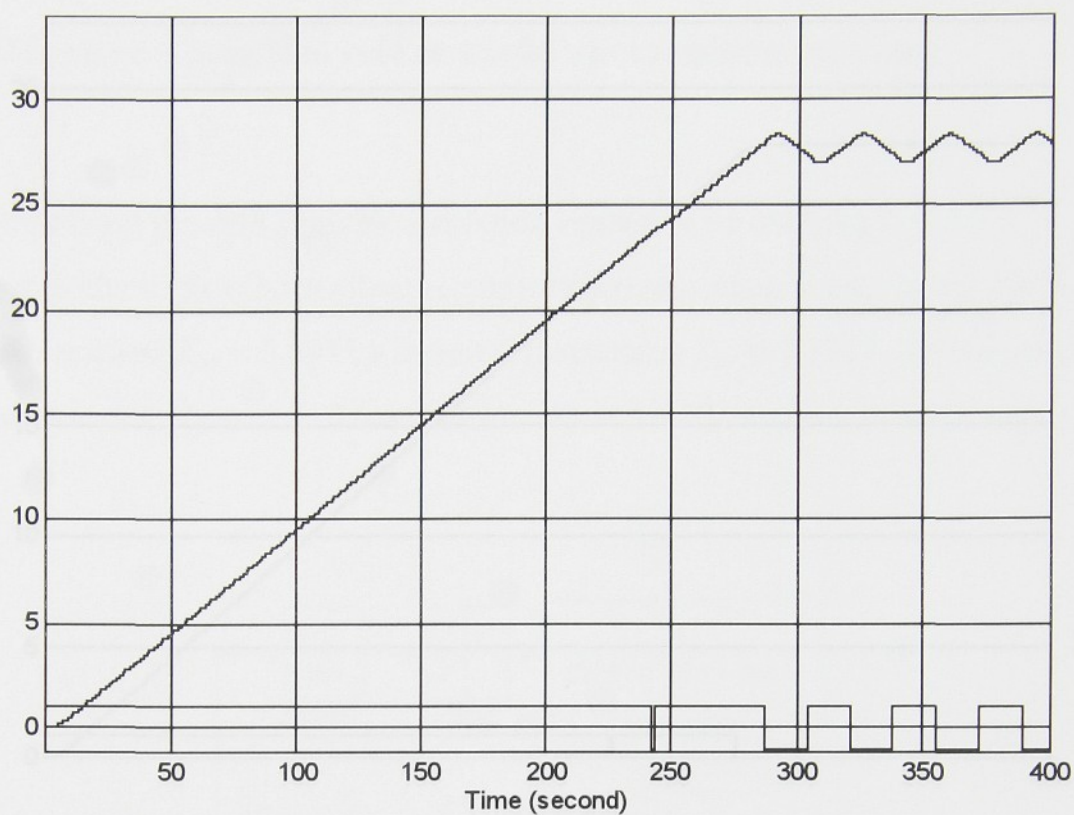
3.5.2. Regulace s použitím relé s pevnou zpětnou vazbou

Regulační obvod, ve kterém je jako regulátor použito relé s pevnou zpětnou vazbou, je na obrázku č. 3.5.2-1. Blok s názvem *floor* zajišťuje převod výstupní hodnoty y na celočíselný tvar před vstupem do regulátoru. Vzorkovač je do obvodu zařazen, aby zamezil reakci servopohonu na vstupní signál v okamžiku, kdy ještě neuplynula potřebná doba od posledního impulsu a ve které by reálný servopohon nebyl schopen reagovat. Matlab tuto funkci neobsahuje, proto byl použit vzorkovač, který přesně tuto funkci neplní, ale při naší zvolené periodě vzorkování 1 sekunda je odchylka zanedbatelná.

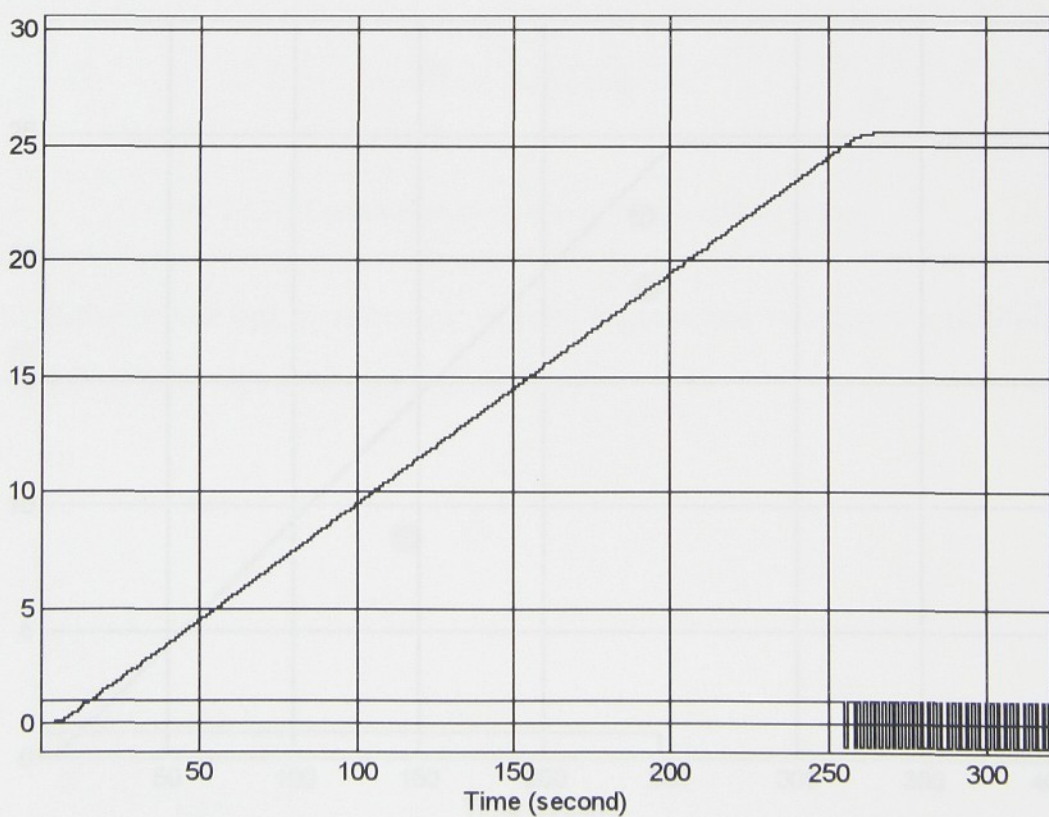


Obr. 3.5.2-1 Regulační obvod s relé s pevnou zpětnou vazbou relé

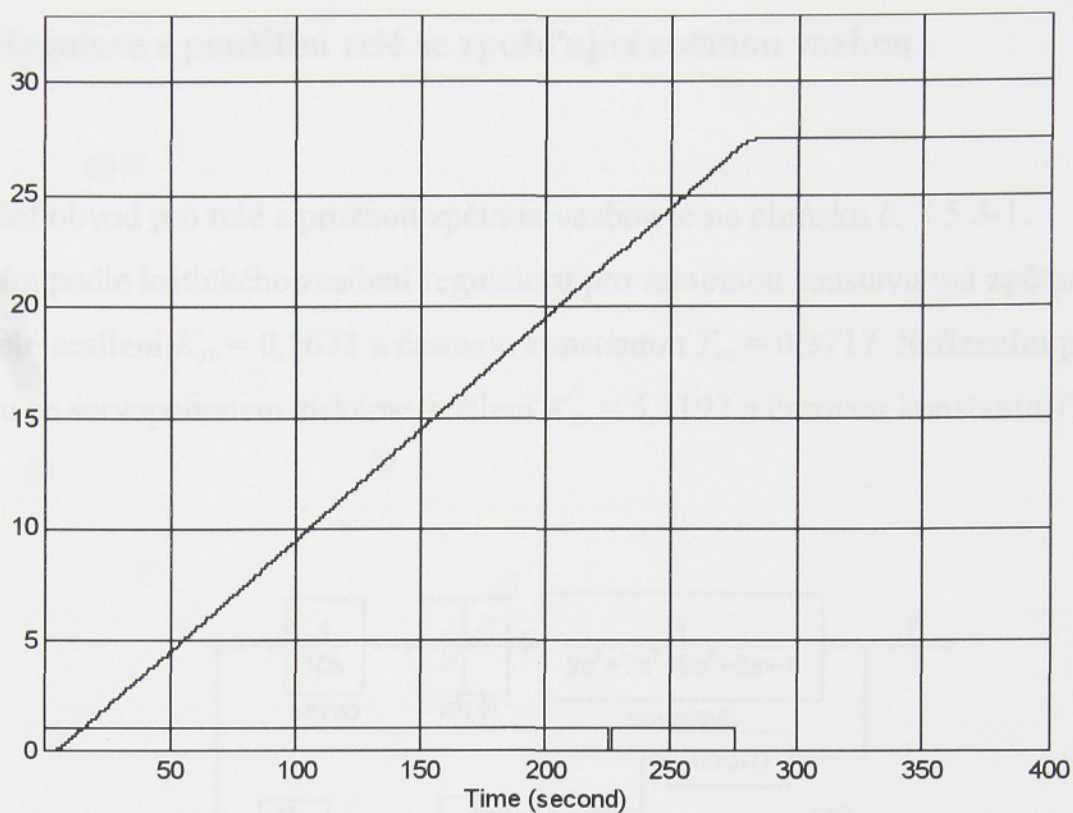
Na následujících obrázcích jsou průběhy regulačních pochodů s dvupolohovým relé bez hystereze a třípolohovým relé s posunutím $D = 1$ bez hystereze s pevnou zpětnou vazbou seřízenou na samotnou soustavu a soustavu se servopohonem.



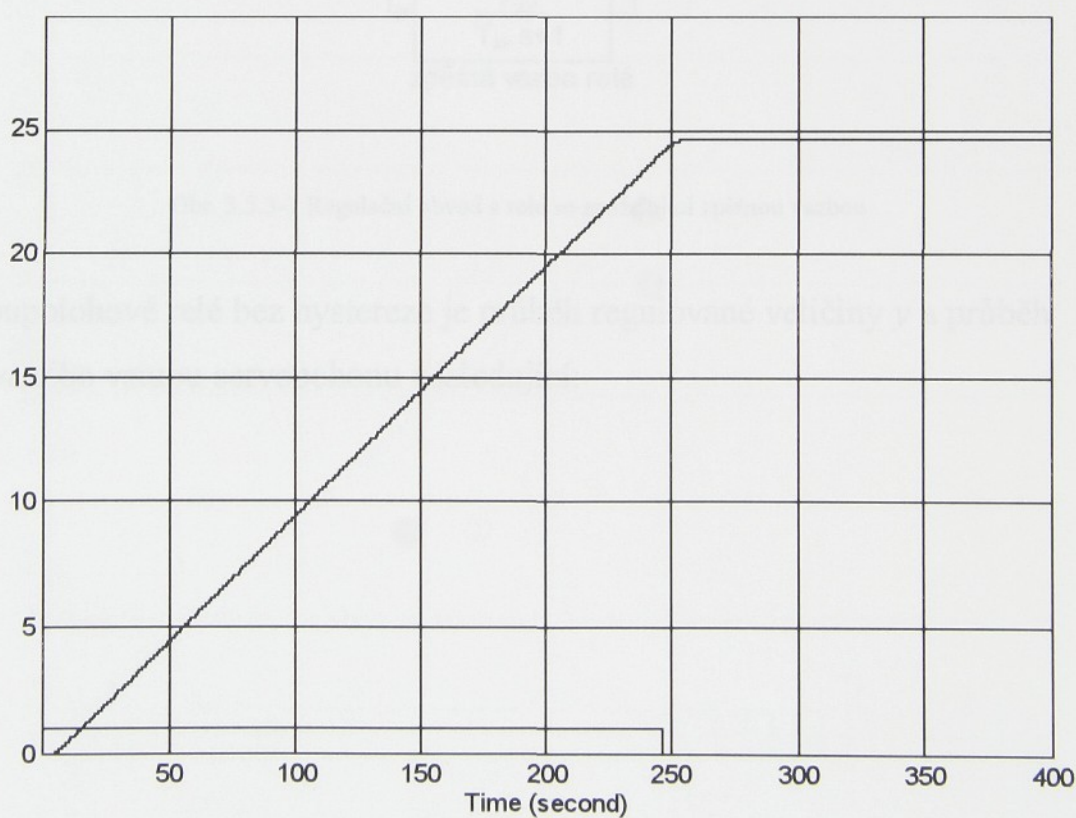
Obr. 3.5.2-2 Regulační pochod s dvupolohovým relé s pevnou zpětnou vazbou seřízenou na soustavu



Obr. 3.5.2-3 Regulační pochod s dvupolohovým relé s pevnou zpětnou vazbou seřízenou na soustavu se servopohonem



Obr. 3.5.2-4 Regulační pochod s třípolohovým relé s pevnou zpětnou vazbou seřizovanou na soustavu

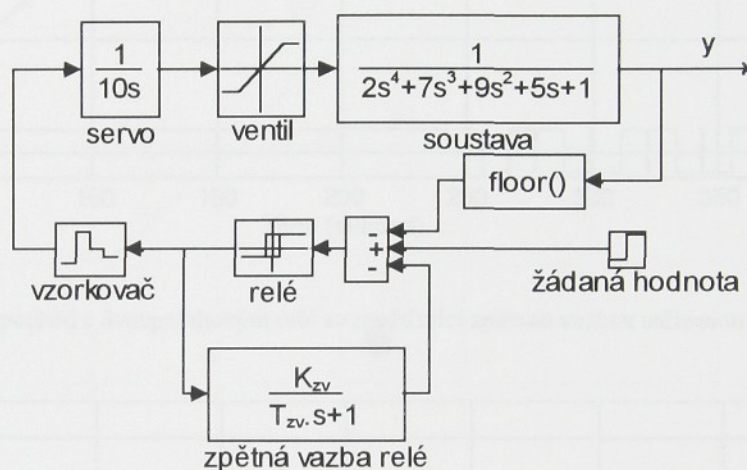


Obr. 3.5.2-5 Regulační pochod s třípolohovým relé s pevnou zpětnou vazbou seřizovanou na soustavu se servem

3.5.3. Regulace s použitím relé se zpožďující zpětnou vazbou

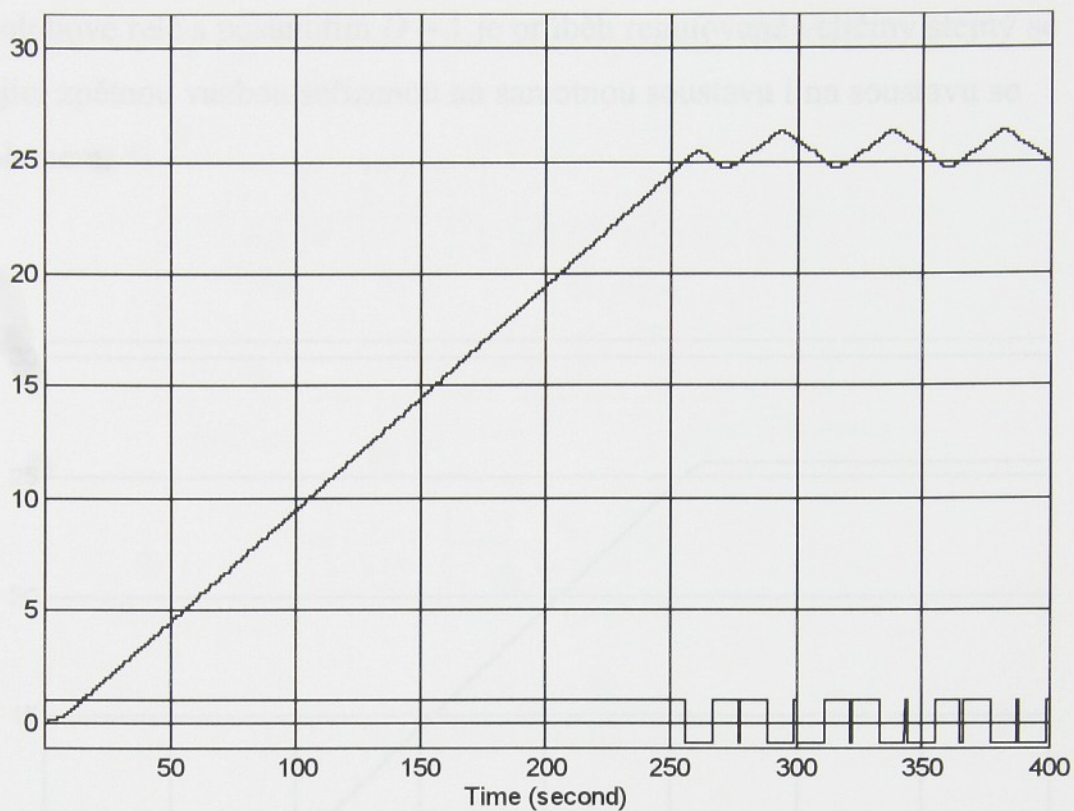
Regulační obvod pro relé s pružnou zpětnou vazbou je na obrázku č. 3.5.3-1.

Seřazením podle kritického zesílení regulátoru pro samotnou soustavu má zpětná vazba relé zesílení $K_{zv} = 0,5671$ a časovou konstantou $T_{zv} = 0,3717$. Seřazením pro soustavu se servopohonem získáme zesílení $K_{zv} = 5,2192$ a časovou konstantu $T_{zv} = 0,9305$.

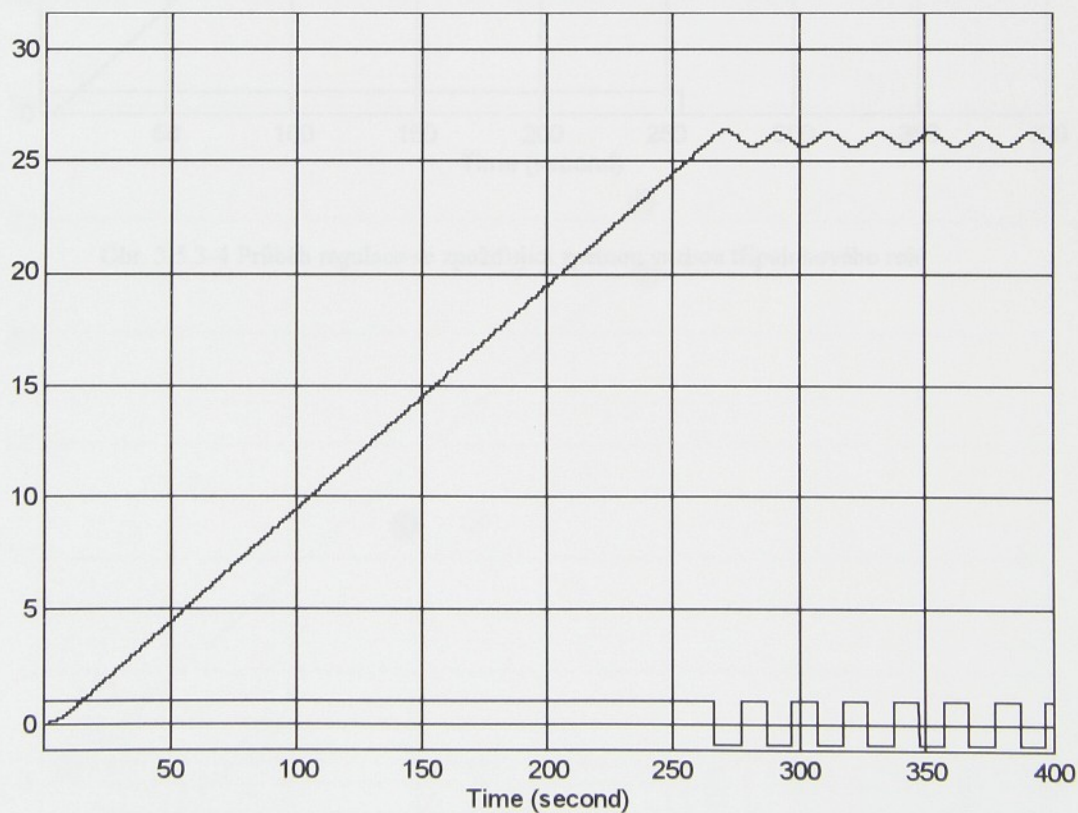


Obr. 3.5.3-1 Regulační obvod s relé se zpožďující zpětnou vazbou

Pro dvoupolohové relé bez hystereze je průběh regulované veličiny y a průběh vzorkovaného vstupu servopohonu následující:

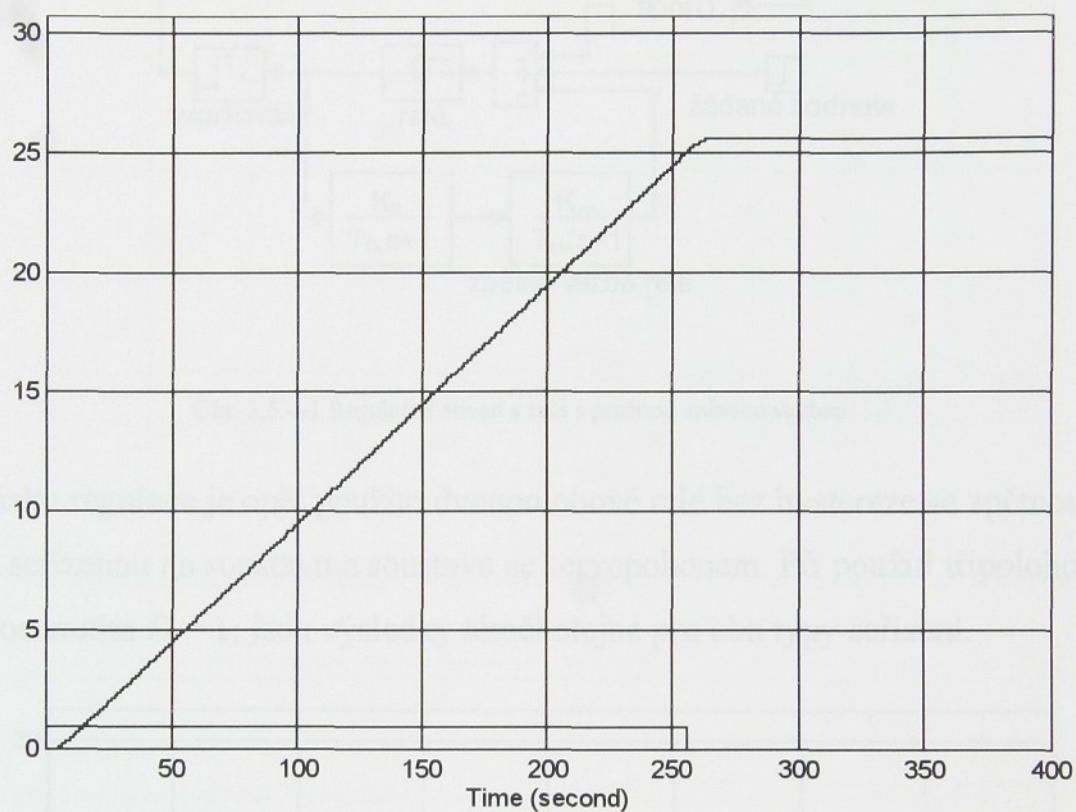


Obr. 3.5.3-2 Regulační pochod s dvoupolohovým relé se zpožďující zpětnou vazbou seřazenou na soustavu



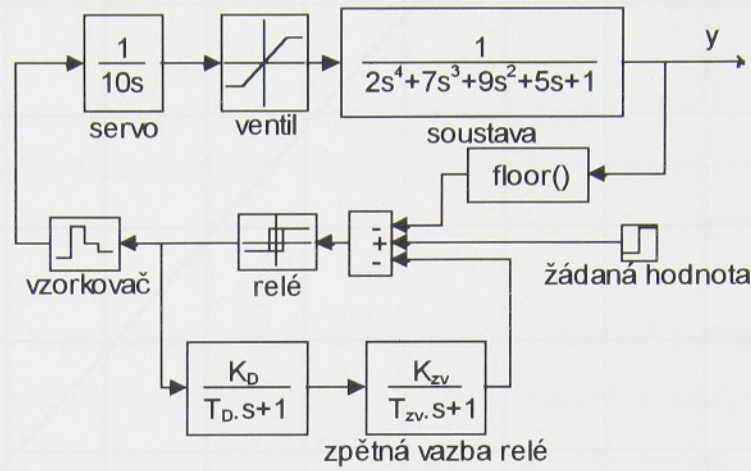
Obr. 3.5.3-3 Průběh regulace při použití zpožďující zpětné vazby seřazené pro soustavu se servopohonem

Pro třípolohové relé s posunutím $D = 1$ je průběh regulované veličiny stejný se zpožďující zpětnou vazbou seřízenou na samotnou soustavu i na soustavu se servopohonem.



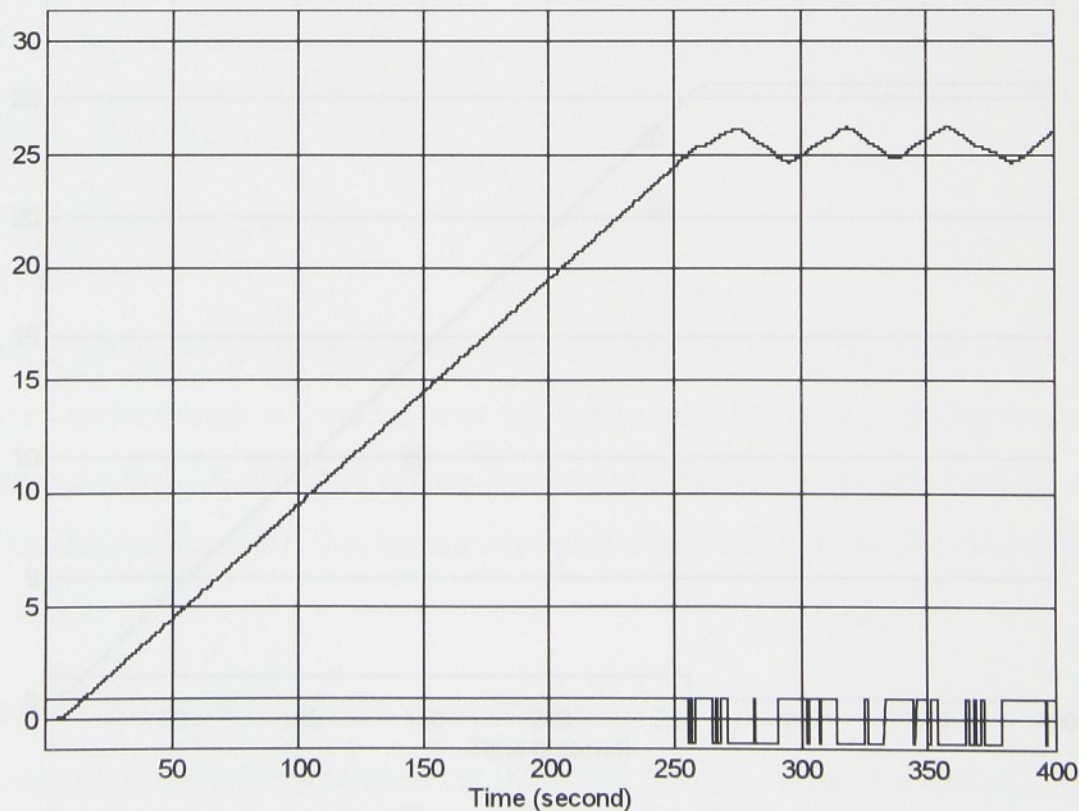
Obr. 3.5.3-4 Průběh regulace se zpožďující zpětnou vazbou třípolohového relé

3.5.4. Regulace s použitím relé s pružnou zpětnou vazbou

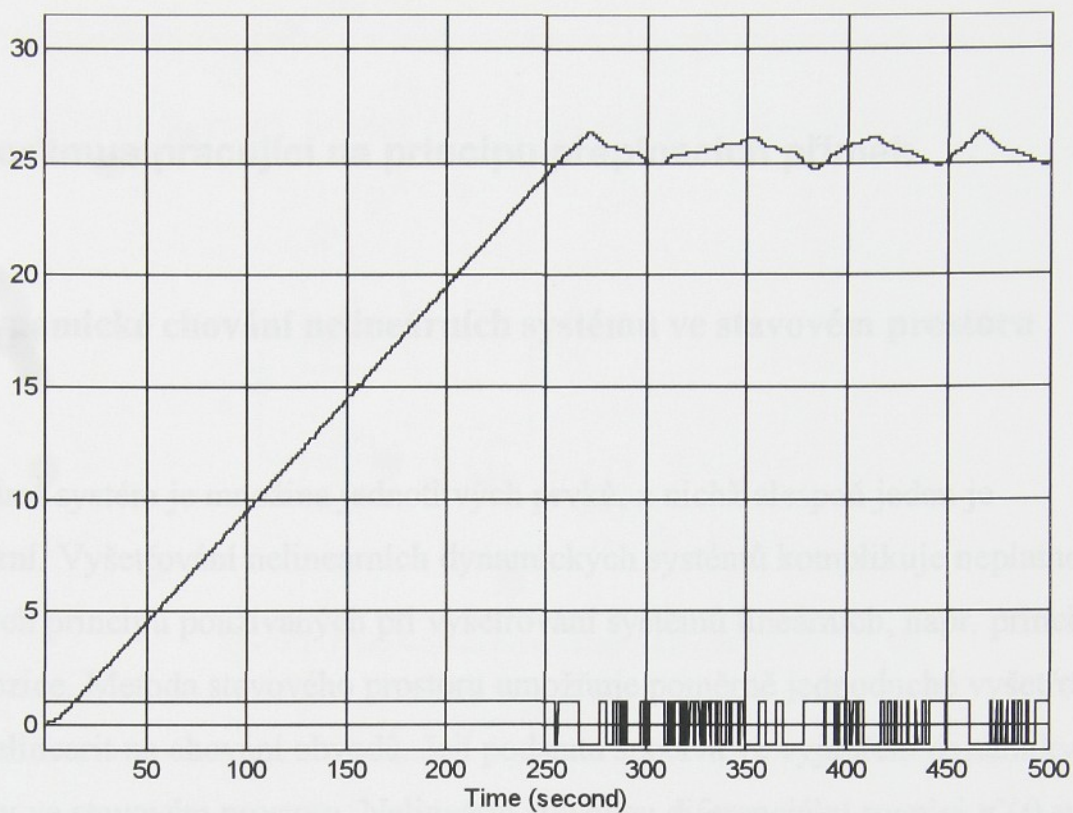


Obr. 3.5.4-1 Regulační obvod s relé s pružnou zpětnou vazbou

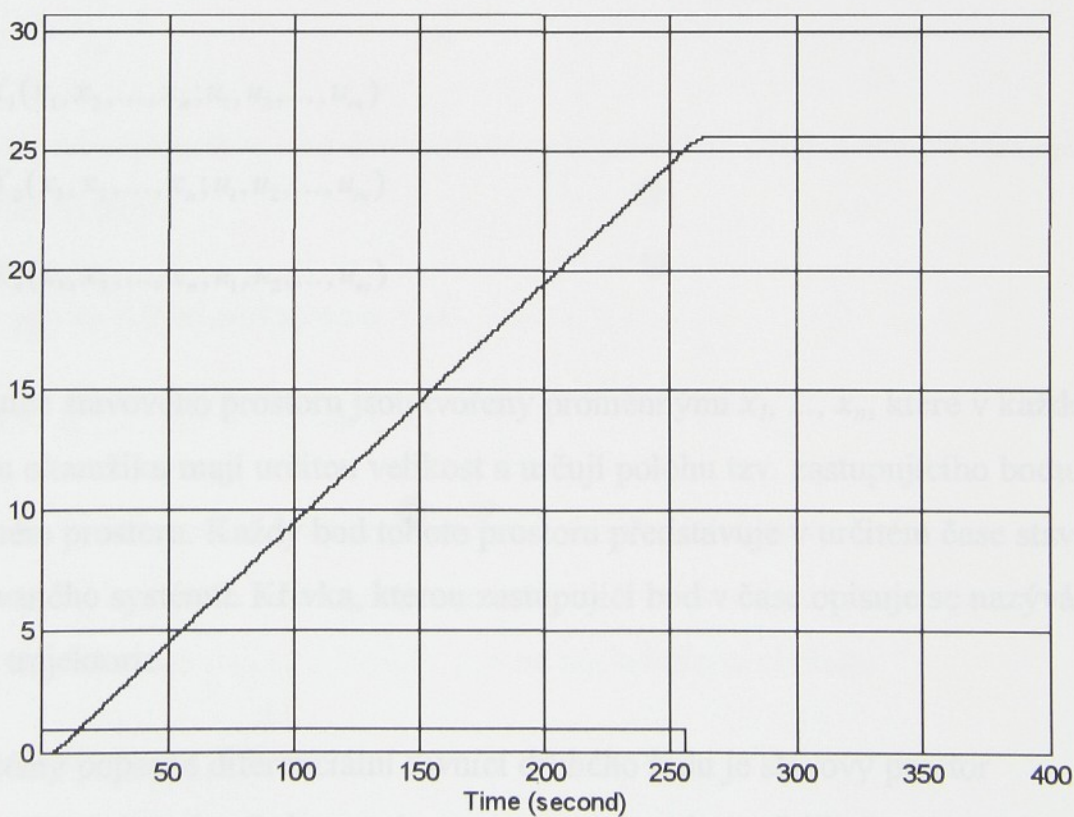
Pro ukázkou regulace je opět použito dvupolohové relé bez hystereze se zpětnou vazbou seřízenou na soustavu a soustavu se servopohonem. Při použití třípolohového relé s posunutím $D = 1$, jsou výsledky téměř stejné pro oba typy seřízení.



Obr. 3.5.4-2 Regulační pochod s dvupolohovým relé s pružnou zpětnou vazbou seřízenou na soustavu



Obr. 3.5.4-3 Regulační pochod s dvoupolohovým relé s pružnou zpětnou vazbou seřízenou na soustavu se servem



Obr. 3.5.4-3 Regulační pochod s třípolohovým relé s pružnou zpětnou vazbou seřízenou na soustavu se servem

4. Algoritmus pracující na principu přepínacích přímek

4.1. Dynamické chování nelineárních systémů ve stavovém prostoru

Nelineární systém je množina jednotlivých prvků, z nichž alespoň jeden je nelineární. Vyšetřování nelineárních dynamických systémů komplikuje neplatnost některých principů používaných při vyšetřování systémů lineárních, např. principu superpozice. Metoda stavového prostoru umožňuje poměrně jednoduché vyšetřování vlivu nelinearit na chování obvodů. Její podstata spočívá ve vyjádření dynamiky soustavy ve stavovém prostoru. Nelineární stavovou diferenciální rovnici $\dot{x}(t) = X[x(t), u(t)]$ lze převést na soustavu n nelineárních diferenciálních rovnic prvního řádu

$$\frac{dx_1}{dt} = X_1(x_1, x_2, \dots, x_n; u_1, u_2, \dots, u_m)$$

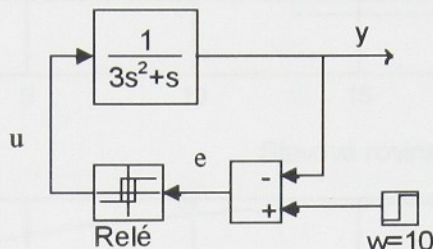
$$\frac{dx_2}{dt} = X_2(x_1, x_2, \dots, x_n; u_1, u_2, \dots, u_m)$$

$$\frac{dx_n}{dt} = X_n(x_1, x_2, \dots, x_n; u_1, u_2, \dots, u_m)$$

Souřadnice stavového prostoru jsou tvořeny proměnnými $x_1, \dots, x_n$, které v každém časovém okamžiku mají určitou velikost a určují polohu tzv. zastupujícího bodu v n -rozměrném prostoru. Každý bod tohoto prostoru představuje v určitém čase stav vyšetřovaného systému. Křivka, kterou zastupující bod v čase opisuje se nazývá stavová trajektorie.

Pro systémy popsané diferenciální rovnicí druhého řádu je stavový prostor dvourozměrný, je tedy představován tzv. stavovou rovinou. Jelikož v rovině lze lépe

zachytit vliv relé na regulační pochod, ukážeme chování regulovaného systému ve stavové rovině na soustavě druhého řádu s obrazovým přenosem $F(s) = \frac{1}{s \cdot (3 \cdot s + 1)}$.



Obr. 4.1.-1 Regulační obvod se soustavou druhého řádu

Diferenciální rovnici druhého řádu $3y'' + y' = u$, popisující soustavu, převedeme na soustavu dvou diferenciálních rovnic prvního řádu

$$\dot{x}_1 = x_2$$

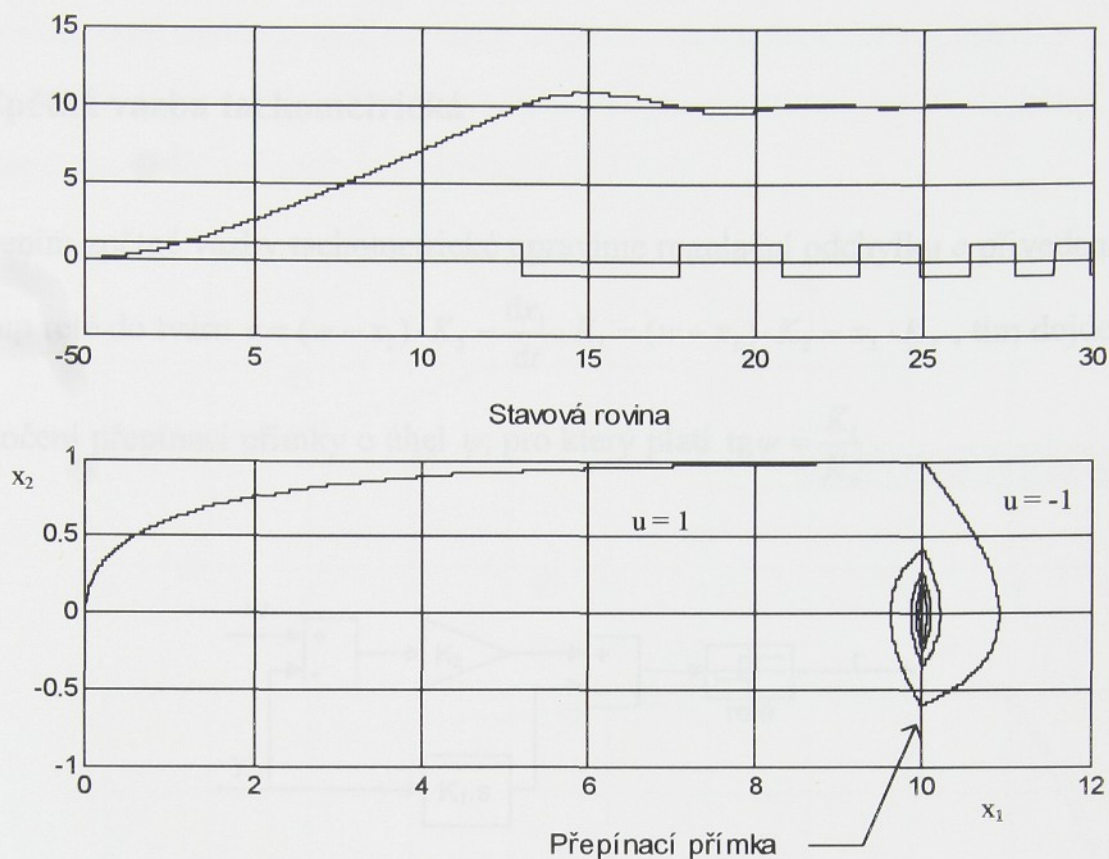
$$\dot{x}_2 = -\frac{1}{3} \cdot x_2 + \frac{1}{3} \cdot u$$

pro žádanou hodnotu $w = 10$ a dvupolohový regulátor bez hystereze dostaneme následující vztahy:

$$e < 0 \Rightarrow w - x_1 < 0 \Rightarrow w < x_1 \Rightarrow u = -1$$

$$e > 0 \Rightarrow w - x_1 > 0 \Rightarrow w > x_1 \Rightarrow u = 1$$

Přepínací přímka obvodu je dána rovnicí $x_1 = w$ a rozděljuje stavovou rovinu na dvě poloroviny. V levé polorovině je výstup relé $u = 1$ a v pravé polorovině je $u = -1$. Průběh regulace a závislost $x_2 = x_2(x_1)$ je na následujícím obrázku.



Obr. 4.1.-2 Průběh regulačního procesu s dvoupolohovým relé

Ze stavové roviny je patrné, že pootočením přepínací přímky doleva by došlo k menšímu překmitu regulované soustavy a ustálení regulované veličiny by proběhlo v kratším čase.

Pootočení přepínací přímky dosáhneme těmito způsoby:

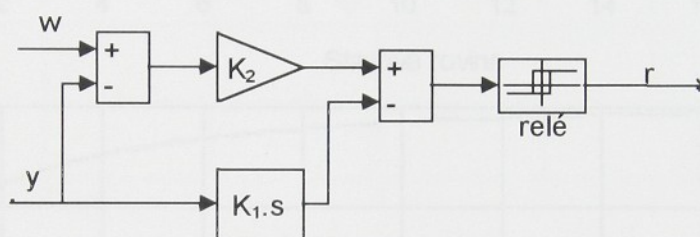
- zavedením zpětné vazby tachometrické
- zavedením derivace do odchytky

4.2. Zpětná vazba tachometrická

Zavedením zpětné vazby tachometrické upravíme regulační odchylku e přivedenou

na vstup relé do tvaru $e = (w - x_1) \cdot K_2 - \frac{dx_1}{dt} \cdot K_1 = (w - x_1) \cdot K_2 - x_2 \cdot K_2$, tím dojde

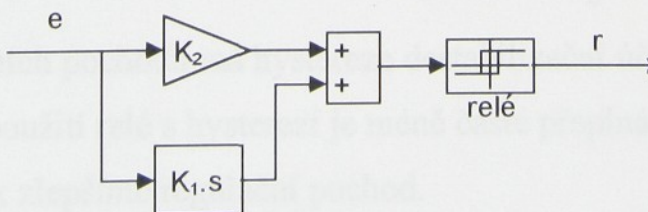
k pootočení přepínací přímky o úhel ψ , pro který platí $\operatorname{tg} \psi = \frac{K_1}{K_2}$.



Obr. 4.2.-1 Zpětná vazba tachometrická

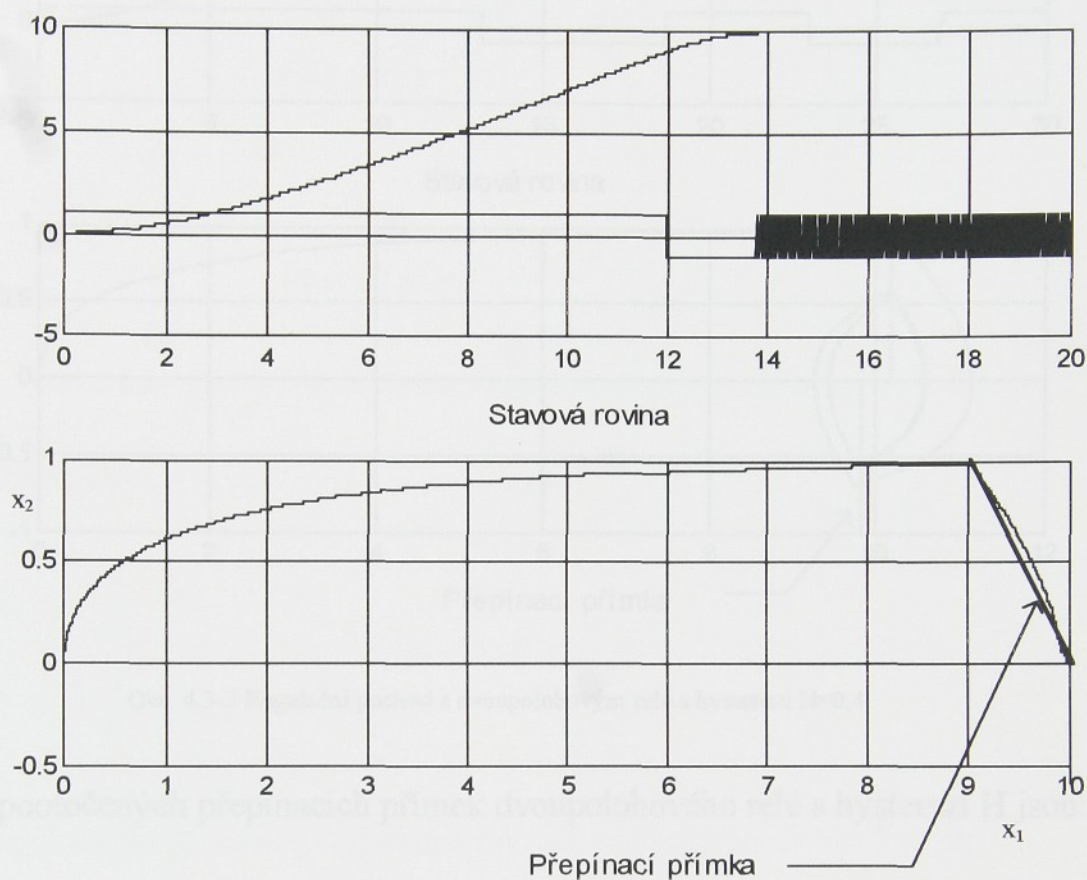
4.3. Zavedení derivace do regulační odchylky

Nevýhodou zpětné vazby tachometrické je, že na vstupu a výstupu členu $K_1.s$ se v praktických aplikacích objevují signály značně různých energetických úrovní. Tuto nevýhodu odstraňuje zavedení derivace do regulační odchylky. Z funkčního hlediska jsou ovšem oba způsoby ekvivalentní.



Obr. 4.3.-1 Zavedení derivace do regulační odchylky

Po zavedení tachometrické zpětné vazby nebo derivace do regulační odchylky s parametry $K_1 = K_2 = 1$ dosáhneme pootočení přepínací přímky o 45° . Průběh regulace a stavová trajektorie je na obrázku č. 4.3.-2.



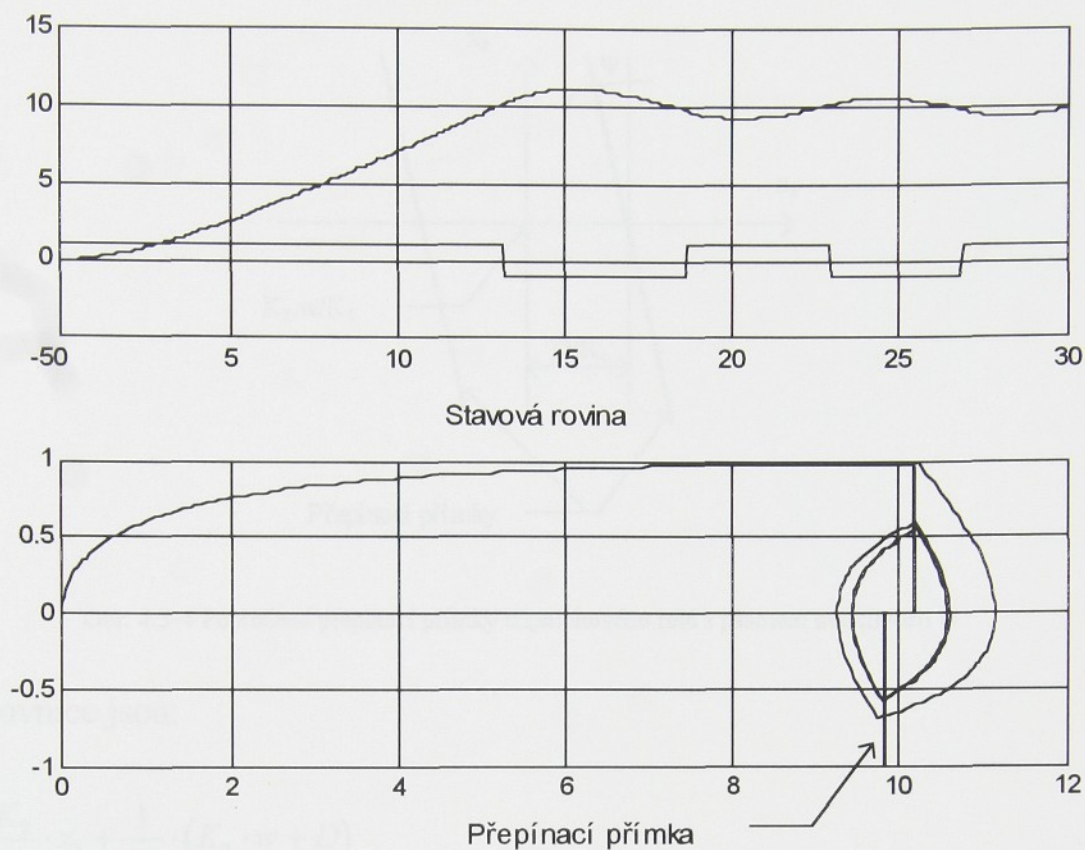
Obr 4.3.-2 Průběh regulačního procesu s dvupolohovým relé bez hystereze a pootočenou přepínací přímkou

Použijeme-li relé s hysterezí H bez zpětné vazby tachometrické a bez zavedení derivace do regulační odchylky, dostaneme ve stavové rovině dvě přepínací přímky, viz obr. 3.4-3. Nad osou x_1 se stavová trajektorie přepíná podle přímky s rovnicí

$$x_1 = w + \frac{H}{2}, \text{ pod osou } x_1 \text{ podle přímky s rovnicí } x_1 = w - \frac{H}{2}.$$

Jak je patrné

z průběhů regulačních pochodů má hystereze destabilizační účinky na regulační proces. Výhodou použití relé s hysterezí je méně časté přepínání relé. Pootočením přepínacích přímek zlepšíme regulační pochod.



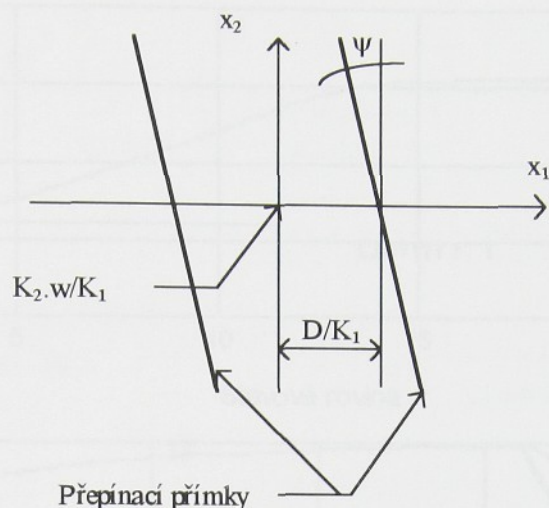
Obr. 4.3-3 Regulační pochod s dvoupolohovým relé s hysterezí $H=0,4$

Rovnice pootočených přepínacích přímek dvoupolohového relé s hysterezí H jsou:

$$x_2 = -\frac{K_2}{K_1} \cdot x_1 + \frac{1}{K_1} \cdot \left(K_2 \cdot w - \frac{H}{2} \right) \quad \text{pro } x_2 < 0$$

$$x_2 = -\frac{K_2}{K_1} \cdot x_1 + \frac{1}{K_1} \cdot \left(K_2 \cdot w + \frac{H}{2} \right) \quad \text{pro } x_2 > 0$$

Pro třípolohové relé s posunutím D jsou pootočené přepínací přímky nakresleny na následujícím obrázku 4.3-4.



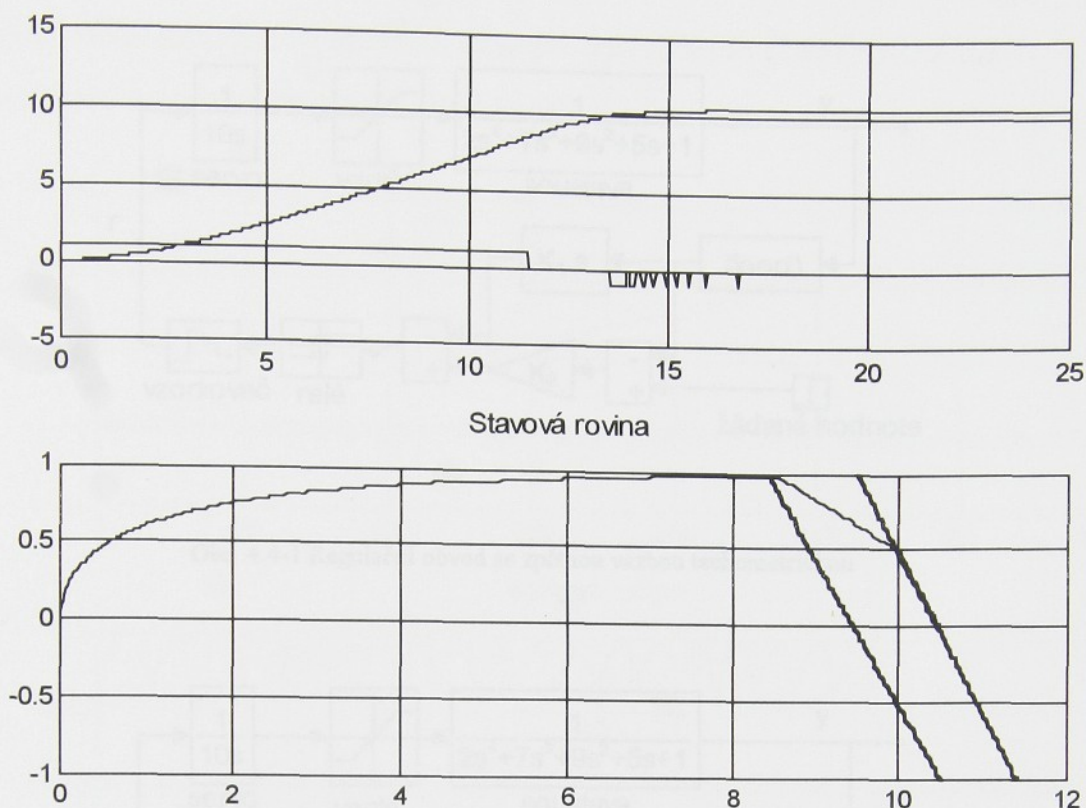
Obr. 4.3-4 Pootočené přepínací přímky třípolohového relé s pásmem necitlivosti D

Jejich rovnice jsou:

$$x_2 = -\frac{K_2}{K_1} \cdot x_1 + \frac{1}{K_1} \cdot (K_2 \cdot w + D)$$

$$x_2 = -\frac{K_2}{K_1} \cdot x_1 + \frac{1}{K_1} \cdot (K_2 \cdot w - D)$$

Průběh regulačního procesu s třípolohovým relé s posunutím $D = 0,5$ je na následujícím obrázku 4.3-5. Parametry tachometrické zpětné vazby jsou $K_I = K_I = 1$.



Obr. 4.3-5 Regulační pochod s třípolohovým relé s pootočenými přepínacími přímkami

4.4. Příklady regulace sekundárního obvodu vytápění

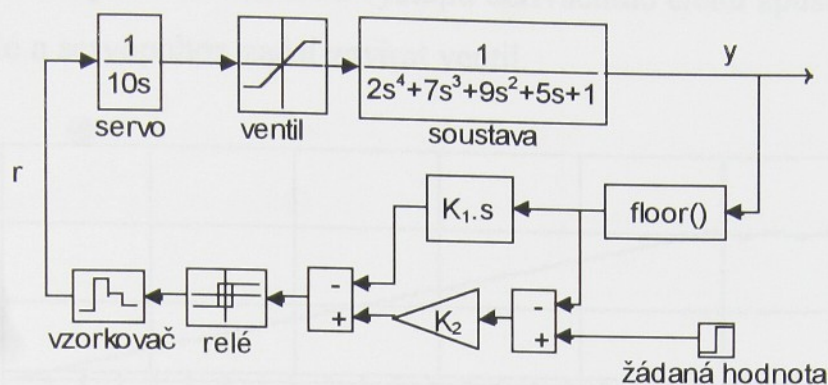
Pro soustavu n -tého řádu se zastupující bod pohybuje po stavové trajektorii v n rozměrném prostoru. Stavová trajektorie se přepíná podle přepínacích hyperploch. Regulaci sekundárního okruhu vytápění metodou přepínacích přímek ukážeme na již

použité regulované soustavě čtvrtého řádu $S(s) = \frac{1}{2 \cdot s^4 + 7 \cdot s^3 + 9 \cdot s^2 + 5 \cdot s + 1}$ a

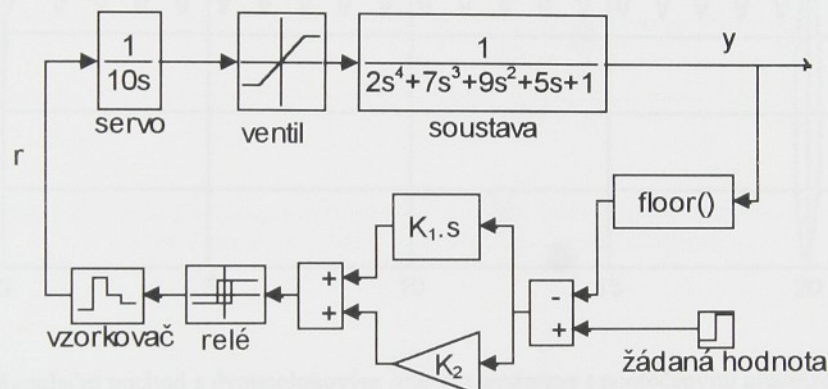
servopohonu s ventilem s přenosem $F_s(s) = \frac{1}{10 \cdot s}$, se vzorkováním 1 sekunda a s

konstantou, která představuje úplné otevření ventilu, $k = 40$. Žádaná hodnota $w = 20$. Jelikož v prostoru 4. řádu je obtížné vykreslit stavovou trajektorii, omezíme se i u této soustavy pouze na stavovou rovinu tvořenou osami x_1 a x_2 .

Regulační blokové schéma s tachometrickou zpětnou vazbou a se zavedenou derivací do regulační odchylky je na následujících obrázcích 4.4-1 a 4.4-2



Obr. 4.4-1 Regulační obvod se zpětnou vazbou tachometrickou

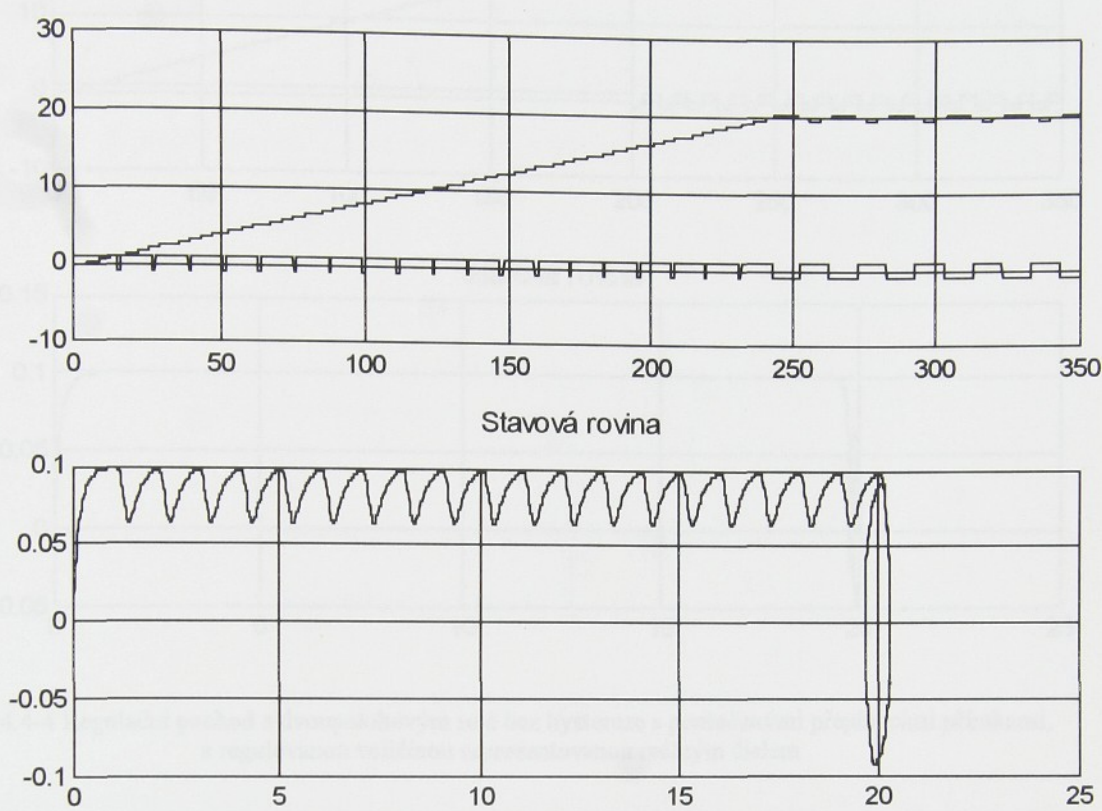


Obr. 4.4-2 Regulační obvod se zavedenou derivací do odchylky

Použití derivátoru v regulační odchylce je v tomto případě problematické, protože do řídicího automatu vstupuje hodnota regulované teploty pouze v celočíselném tvaru. Regulační odchylka e se tedy mění v čase nespojitě. Výstup z derivátoru je nulový. Pouze v okamžiku, kdy se teplota mění, je derivace teoreticky rovna $+\infty$ nebo $-\infty$, podle toho, zda teplota roste či klesá. Prakticky velikost derivace závisí na kroku výpočtu algoritmu.

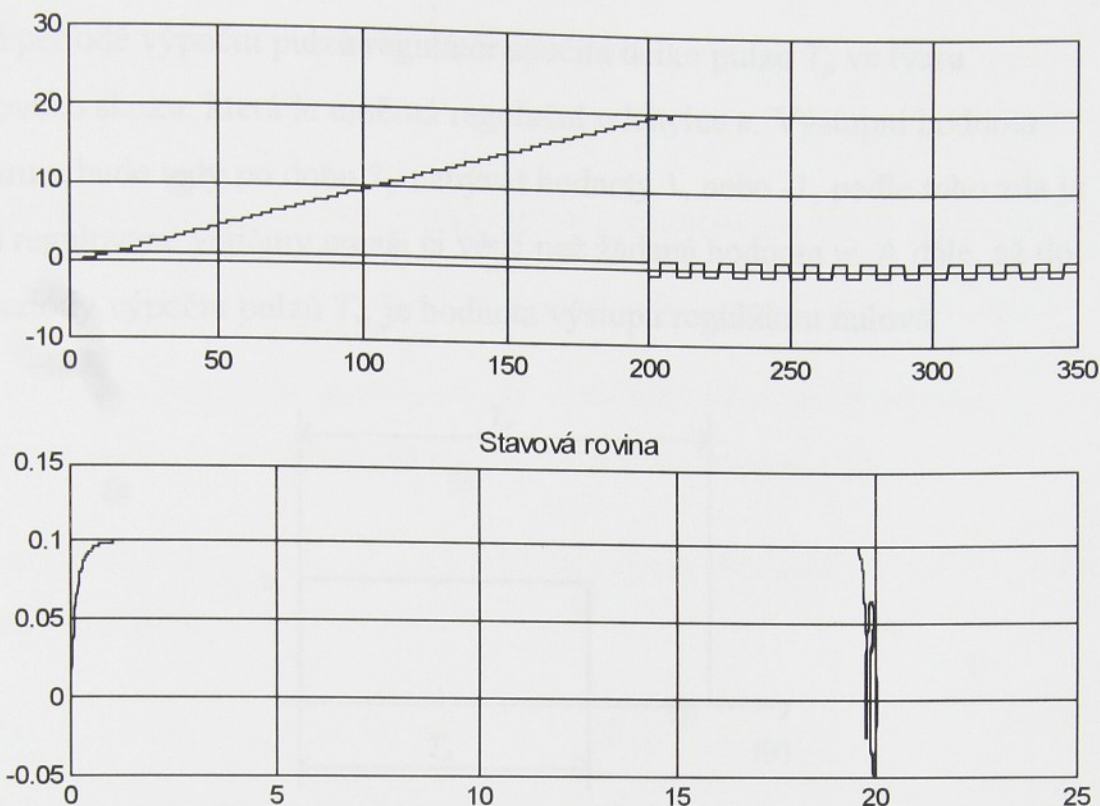
Z průběhu regulačního procesu na obrázku 4.4-3, pro konstanty $K_1 = 5$ a $K_2 = 1$ je patrné, že v každém časovém okamžiku před dosažením žádané hodnoty, ve kterém stoupla celočíselná hodnota teploty y o 1°C , a tedy klesla regulační odchylka e o 1,

nekonečně velká záporná hodnota na výstupu derivačního členu způsobila překlopení relé a servopohon začal zavírat ventil.



Obr. 4.4-3 Regulační pochod s dvupolohovým relé bez hystereze s pootočenými přepínacími přímkami

Pro hodnotu regulované teploty vyjádřenou reálným číslem je průběh regulace znázorněn na následujícím obrázku 4.4-4.



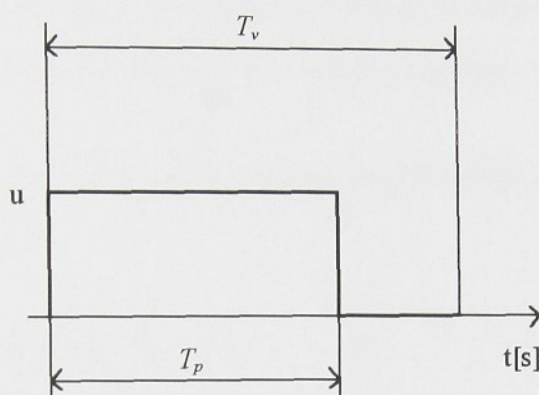
Obr. 4.4-4 Regulační pochod s dvupolohovým relé bez hystereze s pootočenými přepínacími přímkami, s regulovanou veličinou reprezentovanou reálným číslem

5. Algoritmus generující doby zapnutí či vypnutí

Tento algoritmus vychází z principu regulace primární části topného okruhu. Parametry regulátoru jsou maximální povolená teplota T_{max} , minimální povolená teplota T_{min} a perioda výpočtu pulzů T_v . Pro minimální a maximální povolenou teplotu platí $T_{min} < w < T_{max}$. Tento algoritmus udržuje výstupní teplotu v tomto povoleném pásmu.

Je-li regulovaná teplota y menší než minimální povolená teplota T_{min} , je výstup regulátoru $u = 1$, servopohon otevírá regulační ventil. Je-li regulovaná teplota y větší než maximální povolená teplota T_{max} , je výstup regulátoru $u = -1$, servopohon zavírá regulační ventil. Nachází-li se regulovaná teplota v pásmu mezi maximální a minimální povolenou teplotou, algoritmus pracuje s periodou T_v .

V každé periodě výpočtu pulzů regulátor spočítá délku pulzu T_p ve tvaru jednotkového skoku, která je úměrná regulační odchylce e . Výstupní hodnota regulátoru u bude tedy po dobu T_p nabývat hodnoty 1, nebo -1, podle toho zda je hodnota regulované veličiny menší či větší než žádaná hodnota w . A dále, až do konce periody výpočtu pulzů T_v , je hodnota výstupu regulátoru nulová.



Obr. 5-1 Tvar pulzu generovaný regulátorem

Délka pulzu je přímo úměrná regulační odchylce, přičemž $T_{pmax} = T_v$, a T_{pmin} se rovná minimální době odezvy servopohonu. Je-li hodnota regulované teploty větší než žádaná hodnota w a tedy regulační odchylka je menší než nula, je délka pulzu spočítaná z následujícího vztahu:

$$T_p = \frac{(y - w) \cdot T_v}{T_{max} - w} \quad \text{pro } e < 0.$$

Pro hodnotu regulované veličiny menší než žádaná hodnota w , platí pro výpočet délky pulzu následující vztah:

$$T_p = \frac{(w - y) \cdot T_v}{w - T_{min}} \quad \text{pro } e > 0.$$

5.1. Příklady regulace sekundárního okruhu vytápění

Na ukázkou regulačních pochodů použijeme soustavu čtvrtého řádu s přenosem

$$S(s) = \frac{1}{2 \cdot s^4 + 7 \cdot s^3 + 9 \cdot s^2 + 5 \cdot s + 1} \text{ a servopohon s přenosem } F_s(s) = \frac{1}{10 \cdot s}, \text{ který}$$

reaguje na změnu vstupního signálu po uplynutí jedné sekundy. Konstanta představující úplné otevření ventilu $k = 40$, žádaná hodnota $w = 25$.

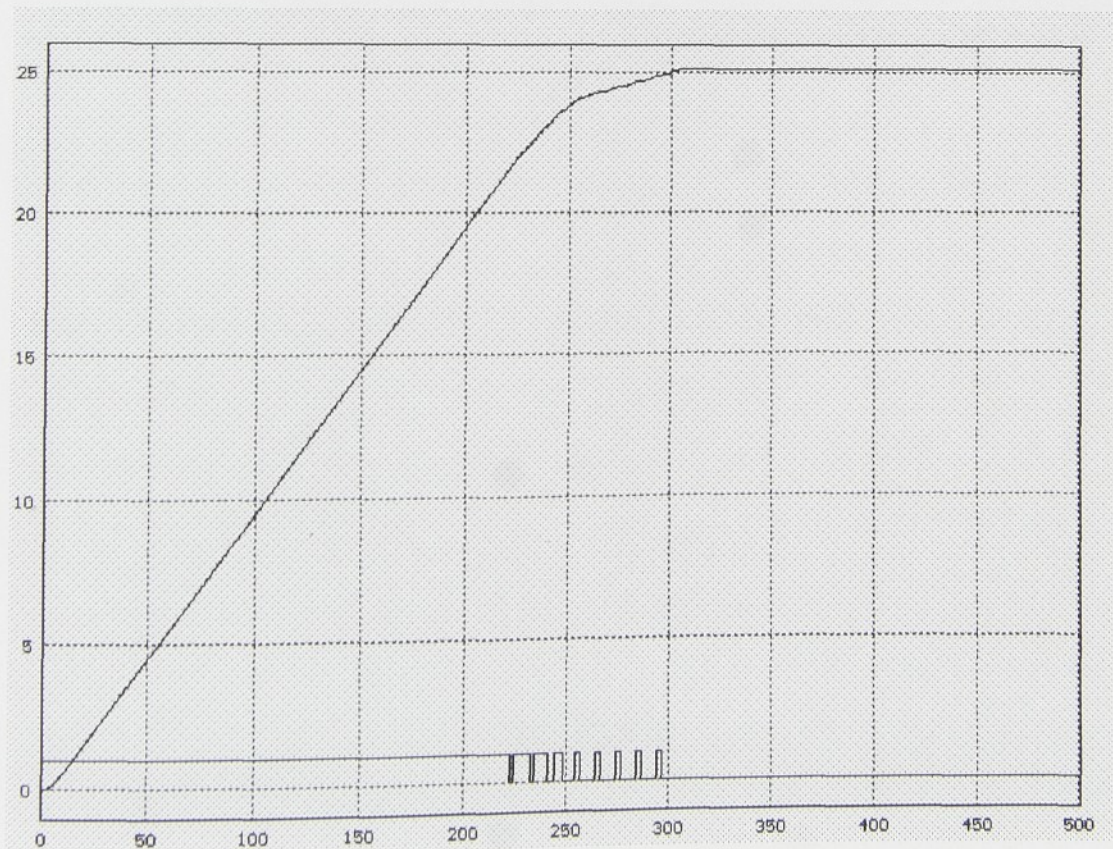
Na následujících obrázcích 5.1-1 až 5.1-4 jsou regulační pochody s následujícími parametry regulátoru:

Obr. 5.1-1 $T_{min} = 20$, $T_{max} = 30$, $T_v = 10$.

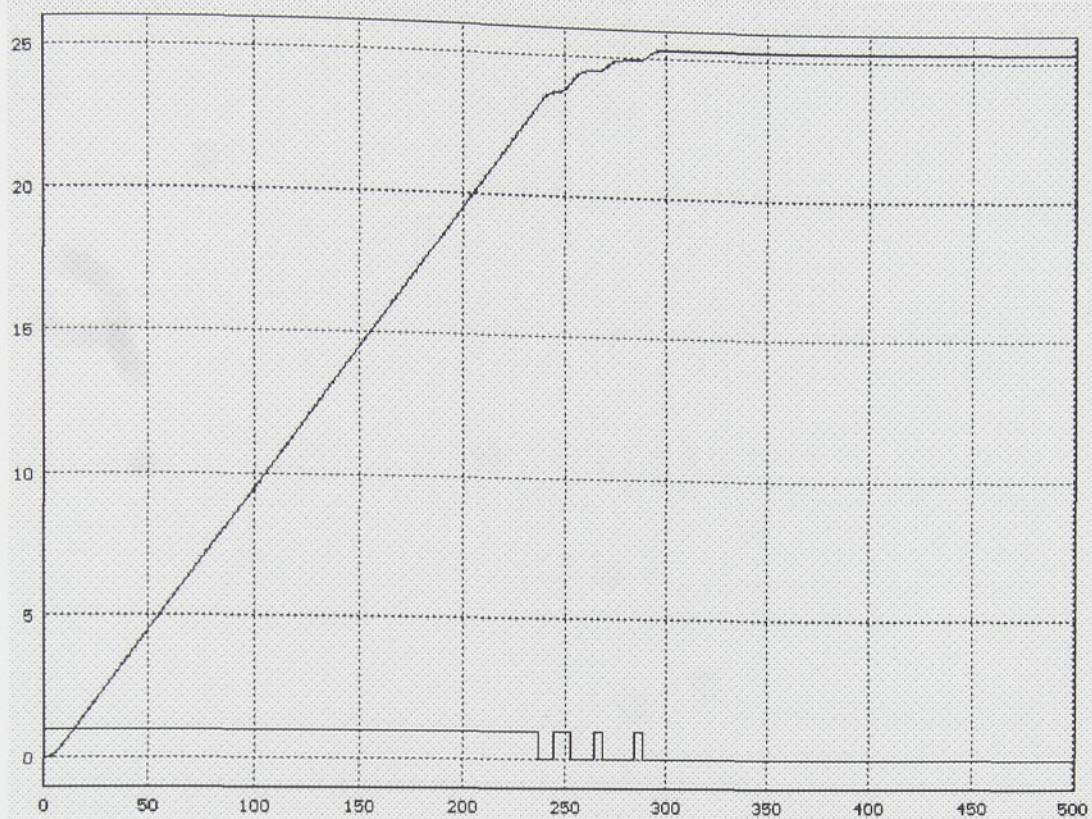
Obr. 5.1-2 $T_{min} = 20$, $T_{max} = 30$, $T_v = 20$.

Obr. 5.1-3 $T_{min} = 20$, $T_{max} = 30$, $T_v = 5$.

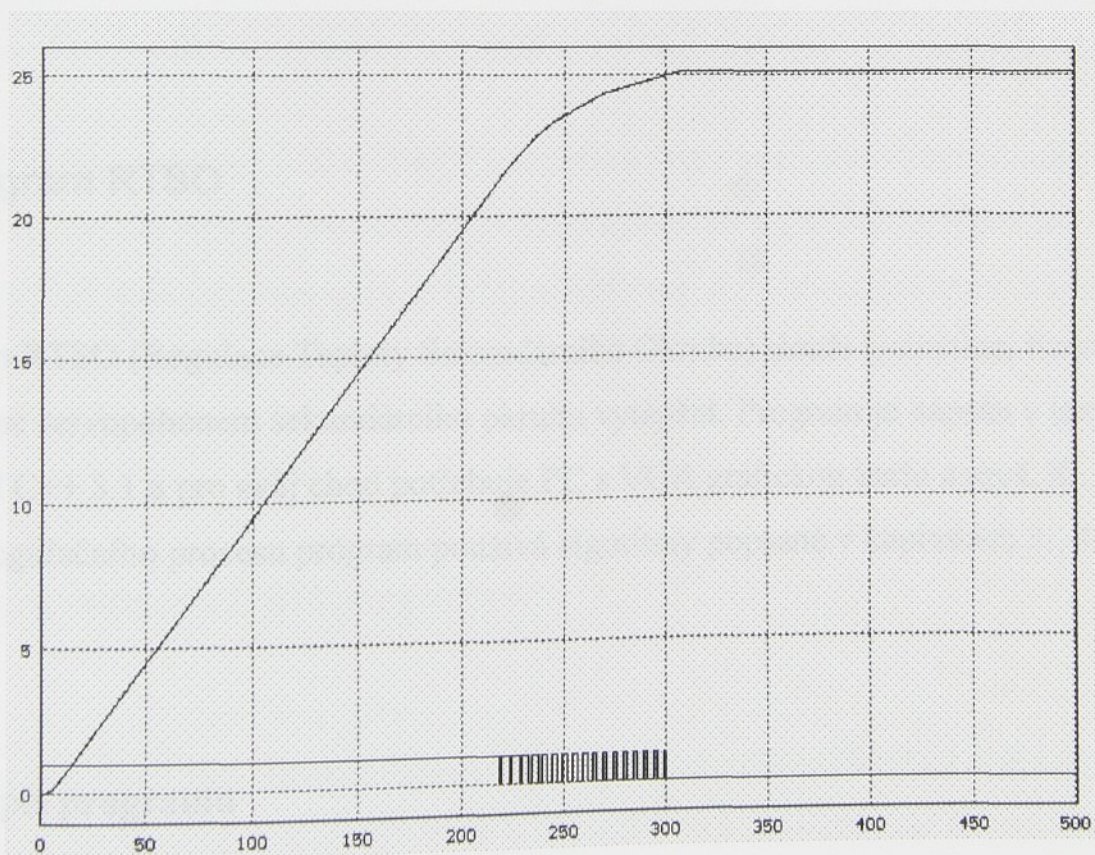
Obr. 5.1-4 $T_{min} = 23$, $T_{max} = 27$, $T_v = 10$.



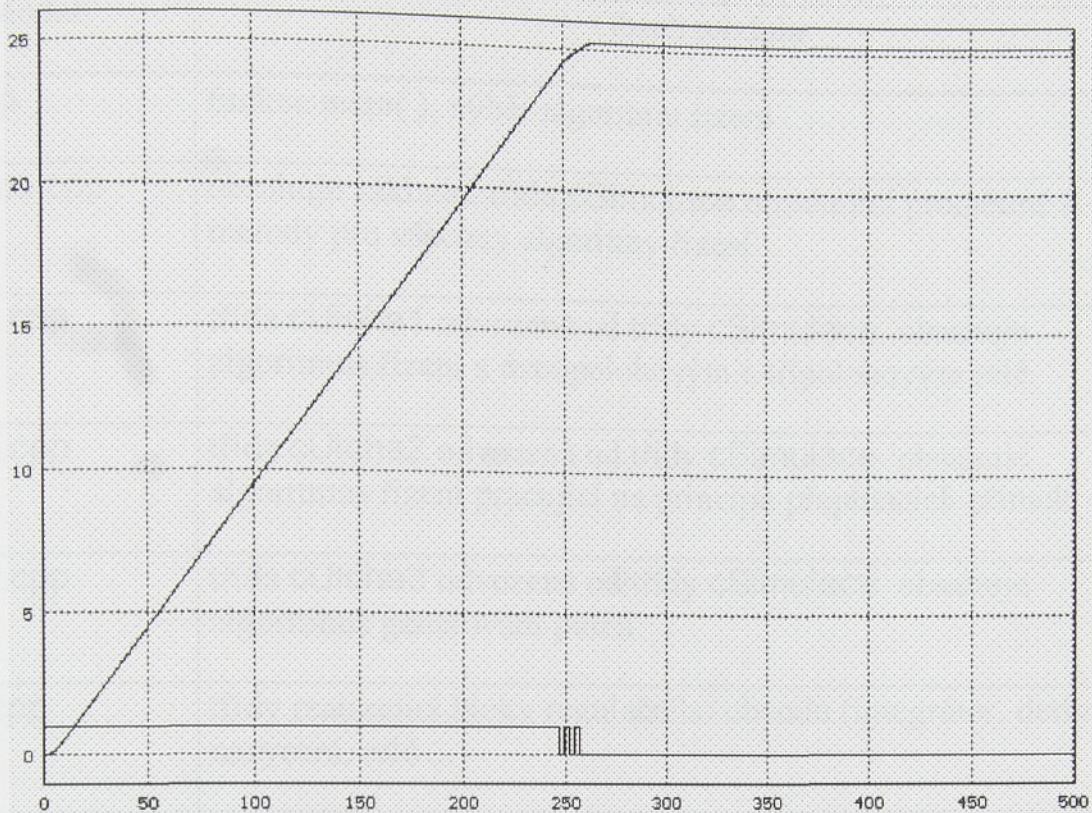
Obr. 5.1-1 Regulační pochod s parametry regulátoru $T_{min} = 20$, $T_{max} = 30$, $T_v = 10$.



Obr. 5.1-2 Regulační pochod s parametry regulátoru $T_{min} = 20$, $T_{max} = 30$, $T_{vz} = 20$.



Obr. 5.1-3 Regulační pochod s parametry regulátoru $T_{min} = 20$, $T_{max} = 30$, $T_{vz} = 5$.



Obr. 5.1-1 Regulační pochod s parametry regulátoru $T_{min} = 23$, $T_{max} = 27$, $T_{vz} = 10$.

6. Program RTSO

Program RTSO (Regulace Teploty Sekundárního Okruhu) slouží k simulaci řízení ventilu se servopohonem sekundárního okruhu vytápění. Program je napsán v jazyce Borland C++ 3.1 a pro svůj chod potřebuje PC s VGA grafickou kartou a myš. K řízení regulačního procesu program používá algoritmy popsané v kapitolách 3., 4. a 5.

6.1. Popis programu

Program je složen z následujících programových modulů.


```

double s_K, s_a4, s_a3, s_a2, s_a1, s_a0, p_K, v_a, se_T, se_vz, w_W;
double krok_vypoctu, krok_ulozeni, cas_simulace; /* parametry simulace */
double *outy, *outu; /* pointery na pole pro hodnoty výstupní teploty a akčních zásahů regulátoru */
long int N, NN; /* parametry smyček výpočtu a uložení hodnot */
cDira *Stav_radka;

public:
cSimulace(void) { /*konstruktor*/
    Stav_radka = new cDira(10,449,539,469);
    if (!(outu = new double [POLE])) /* alokace paměti */
    { Stav_radka->put("Nedostatek pameti"); }
    if (!(outy = new double [POLE]))
    { Stav_radka->put("Nedostatek pameti"); }
}

~cSimulace(void) { delete Stav_radka; delete [ ] outu; delete [ ] outy; } /* destruktor */

int Nastav_Servo (void);
int Nastav_Ventil (void);
int Nastav_Poruchu (void);
int Nastav_W (void);
int sipka (int xx, int yy, SMER smer);
int Prechodovka (void); /* metoda pro výpočet přechodové charakteristiky soustavy */
void Kresli (int xx1, int yy1, int xx2, int yy2); /* metoda pro vykreslení regulačního pochodu */
virtual void Stav (void); /* virtuální metoda pro výpis parametrů regulačního obvodu */
};

```

Kromě zděděných proměnných a metod, třídy obsahují proměnné a metody potřebné k realizaci svých algoritmů.

Třída **cUloha1**, obsahuje uživatelské rozhraní sloužící k simulaci regulačních procesů s použitím dvoupolohového a třípolohového relé se zpětnou vazbou.

```

class cUloha1 : public cSimulace
{
    eTyp_regulatoru typ_reg; /* výčtový typ udávající typ regulátoru */
    eTyp_rele typ_rele; /* výčtový typ udávající typ relé */
    double PD_K, PD_T, P_K, PID_K, PID_T, PID_Td, r3_D, r3_H, r2_H; /* parametry zpětných vazeb relé a relé */

public:

```


Modul	Obsah modulu
rtso.cpp	funkce main(), výběr algoritmu řízení
simulace.cpp	obsahuje základní třídu cSimulace obsahující proměnné a metody pro všechny algoritmy řízení
uloha1.cpp	třída cUloha1 odvozená od třídy cSimulace obsahuje algoritmus řízení s dvoupolohovým a třípolohovým relé
uloha2.cpp	třída cUloha2 odvozená od třídy cSimulace obsahuje algoritmus řízení pracující na principu přepínacích přímek
uloha3.cpp	třída cUloha3 odvozená od třídy cSimulace obsahuje algoritmus generování pulzů
bloky.cpp	třídy realizující bloky regulačního obvodu (integrátor, derivátor, nasycení, relé ...)
button.cpp, dira.cpp, dialog.cpp, textin.cpp	třídy pro grafické objekty (tlačítka, dialogové boxy, pole pro zadávání hodnot v grafickém režimu)

Modul rso.cpp obsahuje funkci main(), která zajišťuje pouze zobrazení úvodní obrazovky a výběr řídicího algoritmu. Po výběru řídicího algoritmu regulace je inicializována třída reprezentující vybraný algoritmus a zavolána její metoda Start(), která dále volá metody své třídy určené k zadávání parametrů a provádění simulačních výpočtů.

6.2. Popis základních tříd programu

Třídy představující jednotlivé algoritmy dědí vlastnosti rodičovské třídy cSimulace:

```
class cSimulace
{
protected:
/* parametry soustavy, poruchy, serva, ventilu, žádaná hodnota */
```



```

double s_K, s_a4, s_a3, s_a2, s_a1, s_a0, p_K, v_a, se_T, se_vz, w_W;
double krok_vypoctu, krok_ulozeni, cas_simulace; /* parametry simulace */
double *outy, *outu; /* pointery na pole pro hodnoty výstupní teploty a akčních zásahů regulátoru */
long int N, NN; /* parametry smyček výpočtu a uložení hodnot */
cDira *Stav_radka;

public:
cSimulace(void) { /*konstruktor*/
    Stav_radka = new cDira(10,449,539,469);
    if (!(outu = new double [POLE])) /* alokace paměti */
    { Stav_radka->put("Nedostatek pameti"); }
    if (!(outy = new double [POLE]))
    { Stav_radka->put("Nedostatek pameti"); }
}

~cSimulace(void) { delete Stav_radka; delete [ ] outu; delete [ ] outy; } /* destruktork */

int Nastav_Servo (void);
int Nastav_Ventil (void);
int Nastav_Poruchu (void);
int Nastav_W (void);
int sipka (int xx, int yy, SMER smer);
int Prechodovka (void); /* metoda pro výpočet přechodové charakteristiky soustavy */
void Kresli (int xx1, int yy1, int xx2, int yy2); /* metoda pro vykreslení regulačního pochodu */
virtual void Stav (void); /* virtuální metoda pro výpis parametrů regulačního obvodu */
};

```

Kromě zděděných proměnných a metod, třídy obsahují proměnné a metody potřebné k realizaci svých algoritmů.

Třída cUloha1, obsahuje uživatelské rozhraní sloužící k simulaci regulačních procesů s použitím dvoupolohového a třípolohového relé se zpětnou vazbou.

```

class cUloha1 : public cSimulace
{
    eTyp_regulatoru typ_reg; /* výčtový typ udávající typ regulátoru */
    eTyp_rele typ_rele; /* výčtový typ udávající typ relé */
    double PD_K, PD_T, P_K, PID_K, PID_T, PID_Td, r3_D, r3_H, r2_H; /* parametry zpětných vazeb relé a relé */

public:

```



```

int Vypocet (void);
int Simulace (void);
void Start (void);
virtual void Stav (void);
int Nastav_Soustavu (void);
int Nastav_Regulator (void);
eTyp_regulatoru Nastav_PD(void);
eTyp_regulatoru Nastav_P(void);
eTyp_regulatoru Nastav_PID(void);
int Nastav_rele (void);
eTyp_rele Nastav_2rele (void);
eTyp_rele Nastav_3rele (void);
int srMKZRsv (void);  /* seřazení regulátoru metodou kritického zesílení pro soustavu se servem
*/
int srMKZR (void);    /* seřazení regulátoru metodou kritického zesílení pro samotnou soustavu */
};

```

Třída cUloha2, obsahuje proměnné a metody realizující algoritmus pracující na principu přepínacích přímek.

```

class cUloha2 : public cSimulace
{
    double r_D, r_H, r3_D, r3_H, r2_H, rg_K1_de, rg_K2_de, rg_K1_tc, rg_K2_tc; /* parametry relé,
    tachometrické zpětné vazby a derivace v regulační odchylce */
    eTyp_rele typ_rele;
    eTyp_reg typ_reg;
    double *outy_der; /* pointer na pole uložení derivace výstupní veličiny, využíváno pro zobrazení
    ve stavové rovině */
public:
    cUloha2 (void) { /* konstruktor */
        if (!(outy_der = new double [POLE])) /* alokace paměti */
            { Stav_radka->put("Nedostatek paměti"); }
    }
    ~cUloha2 (void) { delete [ ] outy_der; } /* destruktork */
    int Vypocet (void);
    int Simulace (void);
    void Start (void);

```

```

virtual void Stav (void);
int Nastav_Soustavu (void);
int Nastav_Regulator (void);
int Nastav_rele (void);
eTyp_rele Nastav_2rele (void);
eTyp_rele Nastav_3rele (void);
eTyp_reg cUloha2::Nastav_DE (void);
eTyp_reg cUloha2::Nastav_TZV (void);
void Stavova_rovina (int x1, int y1, int x2, int y2);
};

```

Ve třídě cUloha3 je implementováno rozhraní a metody pro regulaci řídicím algoritmem generujícím pulzy.

```

class cUloha3 : public cSimulace
{
    double Tmax, Tmin, Tvz; /* parametry regulátoru */
    public:
        int Vypocet (void);
        int Simulace (void);
        void Start (void);
        virtual void Stav (void);
        int Nastav_Soustavu (void);
        int Nastav_Regulator (void);
};

```


6.3. Popis tříd realizujících bloky regulačního obvodu

Každá třída používá pro realizaci výpočtu průběhu regulačního procesu metodu `Vypocet( )`, která využívá třídy `cSet`, `c2Rele`, `c3Rele`, `cDerivator`, `cIntegrator`, `cNasyceni`, `cServoVz` z modulu `bloky.cpp`. Každá z těchto tříd má konstruktor pro nastavení proměnných třídy a metodu `step`, která je volána v každém kroku výpočtu.

```
class cSet
```

```
{ /* realizuje blok s přenosem soustavy prvního řádu */
```

```
double K /* zesílení */, T /* časová konstanta */, y, y1, y1_last, dt /* krok výpočtu */;
```

```
public:
```

```
cSet(double zes, double caskonst, double _dt, double pocpod=0);
```

```
double step (double in);
```

```
};
```

```
cSet::cSet (double zes, double caskonst, double _dt, double pocpod)
```

```
{
```

```
    K = zes; T = caskonst; dt = _dt; y = pocpod; y1_last=0;
```

```
}
```

```
double cSet::step (double in)
```

```
{
```

```
    y1 = in*(K/T) + (-1/T)*y;
```

```
    y = y + (dt * ((y1_last + y1)/(double2)));
```

```
    y1_last = y1;
```

```
    return y;
```

```
}
```

```
class c2Rele
```

```
{ /* realizuje blok s charakteristikou dvoupolohového relé s hysterezí H */
```

```
double out_last, H /*hystereze*/;
```

```
public:
```

```
c2Rele (double hyst);
```

```
double step (double in);
```

```
};
```

```
void c2Rele::c2Rele(double hyst=0)
```

```
{
```

```

H = hyst;
}
double c2Rele::step (double in)
{
    if (in > .5 * H) { out_last=1. ; return 1.; }
    if (in < -.5 * H) { out_last=-1.; return -1.; }
    if (out_last == 1) return 1.;
    else return -1.;
}

class c3Rele
{ /* realizuje blok s charakteristikou třípolohového relé */
    double out_last, H /* hystereze */, D /* posunutí */;
public:
    c3Rele(double pos=1, double hyst=0);
    double step (double in);
};

void c3Rele::c3Rele (double pos, double hyst)
{
    H = hyst; D = pos;
}

double c3Rele::step (double in)
{
    if (in > D) { out_last=1. ; return 1.; }
    if (in < -D) { out_last=-1.; return -1.; }
    if ((in > -D+H) && (in < D-H) ) { out_last=0; return 0; }
    if ((in < D) && (in > D-H) && (out_last==1)) return 1.;
    if ((in < D) && (in > D-H) && (out_last==0)) return 0.;
    if ((in > -D) && (in < -D+H) && (out_last==-1)) return -1.;
    if ((in > -D) && (in < -D+H) && (out_last==0)) return 0.;
    return 0;
}

```

```

class cDerivator
{ /* derivace vstupní veličiny je dána směrnici přímky spojující vstupní hodnoty v čase t-dt a t */
    double in_last, out , dt /* krok výpočtu */;
public:
    double step (double in);
    cDerivator (double _dt);
}

```



```

};
cDerivator::cDerivator (double _dt)
{
    dt = _dt;
}
double cDerivator::step (double in)
{
    out = (in - in_last)/dt;
    in_last = in;
    return out;
}

```

```

class cIntegrator
{ /* integrátor používá lichoběžníkovou metodu integrace */
    double in_last, y /* hodnota integrálu */, dt /* krok výpočtu */;
public:
    cIntegrator (double _dt, double pocpod=0);
    double step (double in);
};
cIntegrator::cIntegrator (double _dt, double pocpod)
{
    dt = _dt; y=pocpod; in_last=pocpod;
}
double cIntegrator::step (double in)
{
    y = y + dt * (in_last + in)/2;
    in_last=in;
    return y;
}

```

```

class cNasyceni
{ /* blok realizující nasycení, výstupní hodnota nabývá hodnot z intervalu <0,a> */
    double a /* maximální hodnota výstupu */;
public:
    cNasyceni (double aa);
    double step (double in);
};
cNasyceni::cNasyceni (double aa)
{

```

```

a=aa;
}
double cNasyceni::step (double in)
{
    if (in>a) return a;
    if (in<0) return (0);
    return in;
}

class cServoVz
{
    /* udržuje výstupní hodnotu minimálně po dobu per_vzor na konstantní hodnotě, tento blok je
    používán na vstupní hodnotu servopohonu, který reaguje na změnu signálu až po uplynutí času
    per_vzor od poslední změny signálu */
    double per_vzor, last_out, cas_zmeny;
public:
    cServoVz (double _per_vzor);
    double step (double time, double in);
};
cServoVz::cServoVz (double _per_vzor)
{
    per_vzor=_per_vzor; last_out=0; cas_zmeny=0;
}
double cServoVz::step (double time, double in)
{
    if ( last_out == in ) return in; /* vstupní hodnota se nezměnila */
    else
    {
        if (time > cas_zmeny) /* výstupní hodnota už byla potřebný čas na konstantní hodnotě */
        {
            cas_zmeny = time+per_vzor;
            last_out = in;
            return in;
        }
        else /* změna výstupní hodnoty ještě není možná */
        {
            return last_out;
        }
    }
}

```


6.4. Popis implementace algoritmů řízení

Všechny výpočtové algoritmy pracují s krokem výpočtu 0,01 sekundy. Celkový čas simulace musí být dělitelný 10, což si sám program ošetří. Je to z toho důvodu, aby krok uložení výstupních hodnot výpočtu do pole 500 hodnot typu **double** byl vždy násobkem kroku výpočtu. Spočítané hodnoty výstupní regulované teploty a akčních zásahů regulátoru uložené v polích jsou potom vykresleny ve tvaru grafu metodou `cSimulace::Kresli(int, int, int, int)`. Výpočet je realizován metodou `Vypocet( )`, která má u všech tří úloh podobnou strukturu. Nejdříve se spočítají parametry simulace, inicializují se třídy odpovídající blokům regulačního obvodu a poté je ve výpočtové smyčce proveden výpočet. Doba výpočtu není vzhledem k parametrům dnešních počítačů velká i při velkých simulačních časech. V následujícím textu jsou popsány výpočtové metody pro všechny tři algoritmy řízení.

a) Algoritmické řízení s třípolohovým a dvoupolohovým relé

```
int cUloha1::Vypocet (void)
{ /* nastavení parametrů výpočtu */
  N = (long int)(cas_simulace/krok_vypoctu+1.1);
  krok_ulozeni = cas_simulace/(POLE-1);
  NN = POLE;
  double _KU = krok_ulozeni/krok_vypoctu;
  int KU = (int)(_KU+.1);
  Stav_radka->put("",BLACK);
  /* inicializace tříd reprezentující bloky obvodu */
  cIntegrator iY1 ( krok_vypoctu, 0);
  cIntegrator iY2 ( krok_vypoctu, 0);
  cIntegrator iY3 ( krok_vypoctu, 0);
  cIntegrator iY4 ( krok_vypoctu, 0);
  cIntegrator iServo (krok_vypoctu, 0);
  cNasyceni nVent (v_a);
  cSet sPDzv (PD_K, PD_T, krok_vypoctu,0);
```

```

c2Rele r2Rele (r2_H);
c3Rele r3Rele (r3_D,r3_H);
cIntegrator iPIDzv (krok_vypoctu, 0);
cSet sPIDzv (PID_K, PID_T, krok_vypoctu, 0);
cServoVz vVzor (se_vz);
double __y0=s_y0,__y1=0,__y2=0,__y3=0,__y4=0;
double PIDhelp=0, zv=0;
double e=0, r=0, s=0, v=0, cas=0, floor__y0;
double Mx=(370-30)/cas_simulace;
long int i;
outy[0]=0;
__y0=0;
outu[0]=1;
int _pom_=0;
double Ka4 = (s_K+p_K)/s_a4, a04=s_a0/s_a4, a14=s_a1/s_a4, a24=s_a2/s_a4,
a34=s_a3/s_a4;
mouseoff();
for ( i=1 ; i<N ; i++)    /* vlastní smyčka výpočtu */
{
    floor__y0 = floor (__y0); /* převod regulované veličiny na celočíselný tvar */
    e = w_W - floor__y0; /* regulační odchylka */
    switch (typ_reg) /* použije zvolený typ regulátoru a relé */
    {
        case PD:
            switch (typ_rele)
            {
                case _2P: r=r2Rele.step(e-sPDzv.step(r)); break;
                case _3P: r=r3Rele.step(e-sPDzv.step(r)); break;
            }
            break;
        case P:
            switch (typ_rele)
            {
                case _2P: r=r2Rele.step(e-P_K*r); break;
                case _3P: r=r3Rele.step(e-P_K*r); break;
            }
            break;
        case PID:
            zv=sPIDzv.step(r-PIDhelp*(1/PID_Td));

```



```

    PIDhelp=iPIDzv.step(r-PIDhelp*(1/PID_Td));
    switch (typ_rele)
    {
        case _2P: r=r2Rele.step(e-zv); break;
        case _3P: r=r3Rele.step(e-zv); break;
    }
}

r = vVzor.step(cas,r); /* kontroluje změnu akčního zásahu před vstupem do serva */
s = iServo.step(r*(1/se_T)); /* servopohon */
v = nVent.step(s); /* ventil */

/* výpočet jednotlivých derivací soustavy metodou snižování řádu derivace */
__y4 = v * Ka4 - __y0*a04 - __y1*a14 - __y2*a24 - __y3*a34; /* čtvrtá derivace y */
__y3 = iY4.step(__y4); /* třetí derivace y */
__y2 = iY3.step(__y3); /* druhá derivace y */
__y1 = iY2.step(__y2); /* první derivace y */
__y0 = iY1.step(__y1); /* hodnota výstupní veličiny y */

if (!(i%KU)) /* testuje zda v tomto kroku uložit hodnoty do pole */
{
    _pom_++;
    outy[_pom_] = __y0; /* uložení hodnoty výstupu teploty */
    outu[_pom_] = r; /* uložení vstupní hodnoty servopohonu */
}

line((int)(30+Mx*cas),300,(int)(30+Mx*cas),320); /* grafická znázornění průběhu výpočtu na
obrazovce */
cas=cas+krok_vypoctu;
}

mouseon();
return 1;
}

```

b) Algoritmus pracující na principu přepínacích přímek

```

int cUloha2::Vypocet (void)
{ /* nastavení parametrů výpočtu */
    N = (long int)(cas_simulace/krok_vypoctu+1.1);
    krok_ulozeni = cas_simulace/(POLE-1);

```

```

NN = POLE;

double _KU = krok_ulozeni/krok_vypoctu;
int KU = (int)(_KU+.1);

Stav_radka->put("",BLACK);

/* inicializace tříd reprezentujících bloky obvodu */
cIntegrator iY1 (krok_vypoctu, 0);
cIntegrator iY2 (krok_vypoctu, 0);
cIntegrator iY3 (krok_vypoctu, 0);
cIntegrator iY4 (krok_vypoctu, 0);
cIntegrator iServo (krok_vypoctu, 0);
cNasyceni nVent (v_a);
c2Rele r2Rele (r2_H);
c3Rele r3Rele (r3_D,r3_H);
cServoVz vVzor (se_vz);
cDerivator dDer_e (krok_vypoctu);

double __y0=0,__y1=0,__y2=0,__y3=0,__y4=0;
double e=0, r=0, s=0, v=0, cas=0, floor__y0;
double Mx=(370-30)/cas_simulace;

long int i;
int _pom_=0;
outy[0]=0;
outy_der[0]=0;
outu[0]=1;

double Ka4 = (s_K+p_K)/s_a4, a04=s_a0/s_a4, a14=s_a1/s_a4, a24=s_a2/s_a4,
a34=s_a3/s_a4;

mouseoff();

for (i=1 ; i<N ; i++) /* vlastní smyčka výpočtu */
{
    floor__y0 = floor (__y0);
    switch (typ_reg) /* provede výpočet zvolenou metodou */
    {
        case DERIV_E:
            e = w_W - floor__y0;
            e = rg_K1_de * e + rg_K2_de * dDer_e.step(e);
            break;

        case TACHOM_ZV:
            e = (w_W - long_int__y0)*rg_K2_tc - rg_K1_tc*dDer_e.step(floor__y0);
            break;
    }
}

```



```

switch (typ_rele) /* použije vybrané relé */
{
    case _2P: r=r2Rele.step(e); break;
    case _3P: r=r3Rele.step(e); break;
}

r = vVzor.step(cas,r); /* kontroluje změnu akčního zásahu před vstupem do serva */
s = iServo.step(r*(1/se_T)); /* servopohon */
v = nVent.step(s); /* ventil */

/* výpočet jednotlivých derivací soustavy metodou snižování řádu derivace */
__y4 = v * Ka4 - __y0*a04 - __y1*a14 - __y2*a24 - __y3*a34; /* čtvrtá derivace y */
__y3 = iY4.step(__y4); /* třetí derivace y */
__y2 = iY3.step(__y3); /* druhá derivace y */
__y1 = iY2.step(__y2); /* první derivace y */
__y0 = iY1.step(__y1); /* hodnota výstupní veličiny y */

if (!(i%KU)) /* testuje zda v tomto kroku uložit hodnoty do pole */
{
    _pom_++;
    outy[_pom_] = __y0; /* uložení hodnoty výstupní teploty */
    outu[_pom_] = r; /* uložení vstupní hodnoty servopohonu */
    outy_der[_pom_] = __y1; /* uložení první derivace výstupní teploty, využívá metoda pro
vykreslení stavové roviny */
}

line((int)(30+Mx*cas),300,(int)(30+Mx*cas),320);
cas=cas+krok_vypoctu;
}

mouseon();
return 1;
}

```

c) Algoritmus generující doby zapnutí či vypnutí

```

int cUloha3::Vypocet (void)
{
    N = (long int)(cas_simulace/krok_vypoctu+1.1);
    krok_ulozeni = cas_simulace/(POLE-1);
    NN = POLE;
    double _KU = krok_ulozeni/krok_vypoctu;
}

```

```

int KU=(int)(_KU+.1);
Stav_radka->put("",BLACK);
cIntegrator iY1( krok_vypoctu, 0);
cIntegrator iY2( krok_vypoctu, 0);
cIntegrator iY3( krok_vypoctu, 0);
cIntegrator iY4( krok_vypoctu, 0);
cIntegrator iServo(krok_vypoctu, 0);
cNasyceni nVent( v_a);
cServoVz vVzor(se_vz);

long int i;

double __y0=s_y0,__y1=0,__y2=0,__y3=0,__y4=0;
double r=0, s=0, v=0;
double cas=0;
double Mx=(370-30)/cas_simulace;
outy[0]=0;
outu[0]=1;
int _pom_=0;
double Ka4 = (s_K+p_K)/s_a4, a04=s_a0/s_a4, a14=s_a1/s_a4, a24=s_a2/s_a4,
a34=s_a3/s_a4;
double konec_zasahu=0, konec_impulzu;
double floor__y0;
mouseoff();
for ( i=1 ; i<N ; i++) //vypocet
{
    floor__y0 = floor( __y0); /* převod regulované teploty na celočíselný tvar */
    if ( (floor__y0 > Tmax) || (floor__y0 < Tmin) ) /* testuje kritický stav, kdy je teplota buď menší
než minimální povolená a nebo větší než maximální povolená teplota */
    {
        if ( floor__y0 > Tmax ) r = -1;
        else r = 1;
        konec_zasahu = cas;
    }
    else
    {
        if ( cas >= konec_zasahu ) /* jestliže již uplynula perioda vzorkování výpočtu spočítá se
délka dalšího impulzu */
        {
            if ( (w_W - floor__y0) > 0 ) /* teplota je menší než žádaná hodnota */
            {
                r = 1;
            }
        }
    }
}

```



```

    konec_impulzu = cas + (w_W-floor__y0)*Tvz/(w_W-Tmin);
}
if ( (w_W - floor__y0) < 0 ) /* teplota je větší než žádaná hodnota */
{
    r = -1;
    konec_impulzu = cas + (floor__y0-w_W)*Tvz/(Tmax-w_W);
}
if ( (w_W - floor__y0) == 0 ) r = 0; /* teplota se rovná žádané hodnotě */
if ((konec_impulzu-cas) < se_vz) konec_impulzu = cas + se_vz; /* jestliže je spočítaný
impulz kratší než na jaký je servopohon schopen reagovat, nastaví se délka impulzu na minimální
hodnotu, kterou je schopen servopohon přijmout */
konec_zasahu = cas + Tvz;
}

if (cas >= konec_impulzu) r = 0; /* po uplynutí času impulzu se výstup až do konce periody
algoritmu nastaví na nulu */
}

s = iServo.step(r*(1/se_T)); /* servopohon */
v = nVent.step(s); /* ventil */

/* výpočet jednotlivých derivací soustavy metodou snižování řádu derivace */
__y4 = v * Ka4 - __y0*a04 - __y1*a14 - __y2*a24 - __y3*a34; /* čtvrtá derivace y */
__y3 = iY4.step(__y4); /* třetí derivace y */
__y2 = iY3.step(__y3); /* druhá derivace y */
__y1 = iY2.step(__y2); /* první derivace y */
__y0 = iY1.step(__y1); /* hodnota výstupní veličiny y */
if (!(i%KU)) /* testuje zda v tomto kroku uložit hodnoty do pole */
{
    _pom_++;
    outy[_pom_] = __y0; /* uložení hodnoty výstupní teploty */
    outu[_pom_] = r; /* uložení vstupní hodnoty servopohonu */
}
cas=cas+krok_vypoctu;
line((int)(30+Mx*cas),300,(int)(30+Mx*cas),320);
}

mouseon();
return 1;
}

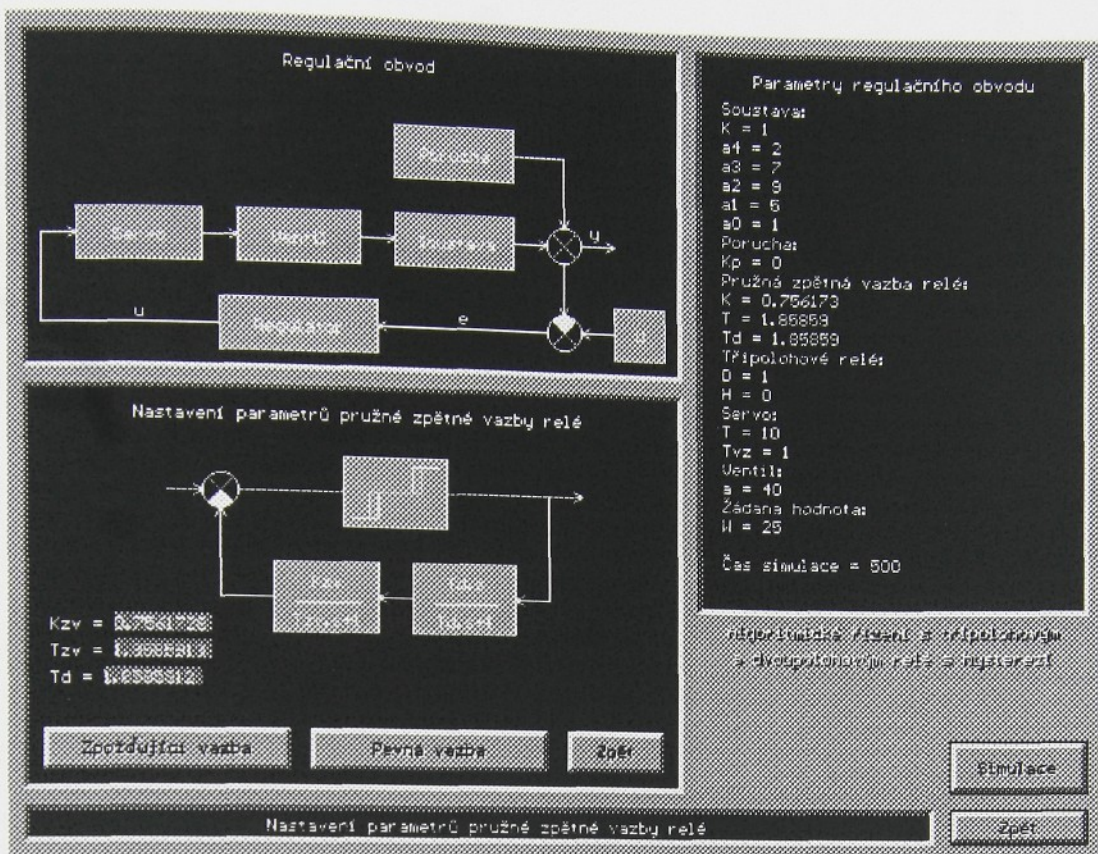
```

6.5. Ovládání programu

Instalaci programu provedeme zkopírováním adresáře SRTO z příložené diskety na pevný disk. Po spuštění programu srto.exe se objeví úvodní obrazovka s informacemi o programu a s výběrovými tlačítky tří algoritmů řízení servopohonu. Výběr algoritmu provedeme myší, stisknutím tlačítka vybraného algoritmu. Po vybrání algoritmu se na obrazovce monitoru zobrazí okno s regulačním obvodem, okno zobrazující parametry regulačního obvodu, stavová řádka a tlačítka **Simulace** a **Zpět**.

Stisknutím tlačítka **Zpět** se vrátíme na úvodní obrazovku, na které je možno program ukončit, nebo zvolit jiný algoritmus řízení servopohonu.

V okně s regulačním obvodem lze kliknutím myši na jednotlivé bloky obvodu otevřít dialogové okno pro nastavení parametrů vybraného bloku. Zadávání nových hodnot provedeme vybráním příslušného zadávacího pole myší. Po výběru pole toto změni barvu ze zelené na modrou a pomocí klávesnice lze zadat novou hodnotu. Nejprve je nutno klávesou **Backspace** starou hodnotu smazat a poté zadat novou číselnou hodnotu. Všechna zadávací pole reagují pouze na stisk číselných kláves, klávesy **Backspace** a desetinné tečky. Výjimku tvoří zadávací pole žádané hodnoty w a času simulace. Tyto hodnoty jsou celočíselné, a proto není povolen znak desetinné tečky. Po zadání nové hodnoty se aktualizují hodnoty v okně, které zobrazuje parametry regulačního obvodu. Dialogová okna se zavírají stiskem tlačítka **Zpět** v pravém dolním rohu okna.



Obr. 6.5-1 Program TRSO s otevřeným dialogovým oknem pro výběr regulátoru

Nastavení parametrů regulátoru je realizováno v dialogovém okně, ve kterém se objeví blokové schéma regulátoru. Na obrázku 6.5-1 je dialogové okno výběru regulátoru s třípolohovým relé s pružnou zpětnou vazbou. Parametry regulátoru se nastavují v zadávacích polích v dialogovém okně. Typ regulátoru je možné změnit příslušným tlačítkem u dolního okraje dialogového okna. Parametry relé lze měnit stisknutím myši na bloku znázorňujícím relé.

Pro algoritmus řízení s dvupolohovým a třípolohovým relé (úloha č. 1.) jsou v dialogovém okně pro zadávání parametrů soustavy dvě tlačítka **MKZR** a **MKZRsv**. První slouží k nastavení hodnot zpětných vazeb relé podle metody kritického zesílení regulátoru pro samotnou soustavu čtvrtého řádu. Tlačítko **MKZRsv** slouží k nastavení zpětných vazeb relé metodou kritického zesílení regulátoru ovšem uvažující soustavu se servopohonem, tj. soustava pátého řádu s astatismem prvního řádu.

Po zadání parametrů regulačního obvodu stiskem tlačítka **Simulace**, které je v pravém dolním rohu obrazovky, se otevře dialogové okno, ve kterém je zadávací pole pro zadání času simulace, tlačítko **Graf** pro vykreslení průběhu regulačního procesu a tlačítko **Přechodovka** pro vykreslení přechodové charakteristiky soustavy.

7. Závěr

V předložené diplomové práci jsou zpracovány a předloženy tři metody řízení servopohonu s ventilem v sekundárním okruhu vytápění pomocí řídicího automatu, ve kterém je naprogramován algoritmus řízení.

První metoda je založena na algoritickém řízení s dvoupolohovým a třípolohovým relé se zpětnou vazbou. Z uvedených odsimulovaných příkladů regulace s konkrétní soustavou je zřejmé, že vhodnější je použití algoritmu s třípolohovým relé, které oproti dvoupolohovému umožňuje při nulové regulační odchylce ponechat servopohon v klidu. Při použití dvoupolohového relé po ustálení regulované veličiny servopohon neustále otvírá a zavírá regulační ventil a hodnota regulované teploty kmitá kolem ustáleného stavu. Z použitých zpětných vazeb relé není vhodná pevná zpětná vazba, protože při jejím použití vzniká trvalá regulační odchylka. Použití zpožďující zpětné vazby nebo pružné zpětné vazby s třípolohovým relé je pro danou problematiku postačující. Z průběhů regulačních pochodů je zřejmé, že i při použití těchto metod dojde k ustálení regulované hodnoty s trvalou regulační odchylkou. Regulační odchylka je ovšem v těchto případech menší než 1°C a tudíž o ní řídicí automat „neví“, protože vstupní hodnoty řídicího automatu jsou, jak už bylo uvedeno v úvodu, v celočíselném tvaru. Pro praktické využití se zdá být tento přístup ne zcela využitelný, protože v praxi nejsou známy přenosy regulovaných soustav a nastavení parametrů zpětných vazeb relé by bylo nutné udělat pro konkrétní regulovaný systém ručně.

Metoda přepínacích přímek používá ve svém výpočtu derivaci regulované veličiny. Regulovaná veličina ovšem vstupuje do řídicího automatu pouze v celočíselném tvaru, takže výpočet derivace numerickými metodami je prakticky nemožný. Z tohoto důvodu je uvedená metoda pro tento typ řídicího automatu nepoužitelná.

Výhoda algoritmu generujícího doby zapnutí a vypnutí je v tom, že uživatel nastaví velikost žádané hodnoty a teplotního pásma omezeného maximální a minimální

povolenou teplotou, ve kterém se má regulovaná teplota pohybovat. A průběh regulace může ovlivňovat pouze hodnotou udávající velikost periody výpočtu pulzů. Čím je perioda kratší, tím častější jsou zásahy regulátoru. Velikost periody výpočtu pulzů by měla být závislá na rychlosti odezvy regulované soustavy. Zde se nabízí možnost vylepšit vlastnosti regulace adaptací periody výpočtu pulzů během regulačního procesu.

Všechny uvedené metody řízení sekundárního okruhu vytápění lze simulačně odzkoušet programem RTSO, který je součástí této práce. Při vývoji tohoto simulačního prostředku byl kladen důraz na jednoduchost simulačních výpočtů a jednoduchost zadávání parametrů regulačního obvodu. Jelikož je ovšem program vytvořen pod operačním systémem DOS, postrádá komfortnost operačního systému Windows, který v dnešní době ovládá trh. Do budoucna by bylo vhodné převést program do prostředí Windows a využít veškerých výhod tohoto systému, jako je výstup výsledných průběhů na tiskárnu, výměna dat s jinými programy apod. Velkým omezením programu je také použití soustav pouze čtvrtého řádu. Toto omezení ovšem zjednodušilo algoritmus simulačního výpočtu a výsledné průběhy regulačních dějů tohoto programu jsou téměř identické s výsledky profesionálního simulačního programu Matlab 4.2.

8. Literatura

- [1] BALDA, M. - HANUŠ, B.: Základy technické kybernetiky. SNTL Praha, 1986
- [2] JANEČEK, J. - MODRLÁK, O.: Základy technické kybernetiky - cvičení.
Skripta VŠST, Liberec 1990
- [3] PETROVIČ, J. - IVANIČ, J.: Automatizácia v priemysle. ALFA Bratislava
- [4] NENADÁL, K. - VÁCLAVÍKOVÁ, D.: Turbo C Popis jazyka. Grada a.s.,
1991
- [5] NENADÁL, K. - VÁCLAVÍKOVÁ, D.: Borland C++ Objektové programování
a popis jazyka. Grada a.s., 1992
- [6] HEROUT, P. - RUDOLF, V. - ŠMRHA, P.: ABC programátora v jazyce C.
KOPP, 1992