UNIVERZITA PARDUBICE

Fakulta elektrotechniky a informatiky

REALIZACE SOFT SENZORU V PLC POMOCÍ UMĚLÉ NEURONOVÉ SÍTĚ

Bc. Tomáš Hort

Diplomová práce

2022

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2021/2022

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Tomáš Hort**
Osobní číslo:	**I20190**
Studijní program:	**N0714A150005 Automatické řízení**
Téma práce:	**Realizace soft senzoru v PLC pomocí umělé neuronové sítě**
Zadávající katedra:	**Katedra řízení procesů**

Zásady pro vypracování

Postup: Cílem práce je návrh a implementace soft senzoru založeného na dopředné vícevrstvé umělé neuronové síti v prostředí programovatelného logického automatu. Student v rámci práce vytvoří soft senzor, tedy estimátor jedné stavové veličiny ve vybraném reálném dynamickém procesu. Tento soft senzor následně bude použit při výpočtu akčního signálu pro řízení procesu. Soft senzor i řídicí systém budou realizovány pomocí jediného programovatelného logického automatu.
Teoretická část: Stručná rešerše problematiky estimace stavových veličin a soft senzorů. Stručná rešerše problematiky umělých neuronových sítí pro aproximaci funkcí a modelování dynamických systémů. Popis rodiny programovatelných logických automatů použitých při řešení práce. Popis řídicího algoritmu vybraného pro realizaci praktické části práce.
Praktická část: Výběr, případně návrh a realizace dynamického procesu k řízení. Návrh a implementace soft senzoru založeného na vybraném paradigmatu umělých neuronových sítí. Návrh a realizace řízení procesu s využitím soft senzoru. Komplexní testování a vyhodnocení kvality regulačního pochodu.

Rozsah pracovní zprávy: **cca 60 stran**
Rozsah grafických prací:
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

HAYKIN, Simon S., c2009. *Neural networks and learning machines*. 3rd ed. New York: Prentice Hall. ISBN 9780131471399.
BALÁTĚ, J. *Automatické řízení*. Praha: BEN, 2003. 654 s. ISBN 80-7300-020-2.
KWASNIEWSKI, J. *Programmable Logic Controllers*. Cracow: ROMA-POL, 2002. ISBN 83-86320-45-1.

Vedoucí diplomové práce: **doc. Ing. Petr Doležel, Ph.D.**
Katedra řízení procesů

Datum zadání diplomové práce: **8. listopadu 2021**
Termín odevzdání diplomové práce: **20. května 2022**

L.S.

Ing. Zdeněk Němec, Ph.D. v.r.
děkan

Ing. Daniel Honc, Ph.D. v.r.
vedoucí katedry

V Pardubicích dne 16. listopadu 2021

Prohlášení

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 17. 8. 2022

Bc. Tomáš Hort

Poděkování

Děkuji vedoucímu diplomové práce doc. Ing. Petru Doleželovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Pardubicích dne 17. 8. 2022

Bc. Tomáš Hort

ANOTACE

Práce je věnována problematice implementace soft senzoru v PLC založeného na paradigmatu umělé neuronové sítě. Byla provedena analýza estimace stavových veličin a umělých neuronových sítí a byl navržen algoritmus řízení s využitím PID regulátoru. Je vypracován simulační program regulované soustavy ve výpočetním prostředí Simulink a program pro vytvoření soft senzoru včetně implementace v programovacím prostředí TIA Portal. Provozuschopnost navrženého algoritmu je demonstrována na řízení soustavy nádrží.

KLÍČOVÁ SLOVA

soft senzor, umělá neuronová síť, programovatelný logický automat, PID regulátor

TITLE

NEURAL NETWORK-BASED SOFT SENSOR USING PLC

ANNOTATION

The work is oriented to the issue of implementation of a soft sensor in PLC based on the artificial neural network paradigm. The analysis of state variables estimation and artificial neural networks was realized and the control algorithm using PID controller was designed. A simulation program in Simulink computing environment is developer, together with a program for creating a soft sensor including implementation in the TIA Portal programming environment. The operability of the proposed algorithm is demonstrated on the example of two tanks system control.

KEYWORDS

Soft sensor, Artificial neural network, Programmable logic controller, PID controller

OBSAH

SEZNAM ZKRATEK A ZNAČEK

AC	alternating current
API	application programming interface
ARX	autoregressive with extra input
CPU	central processing unit
DB	datový blok
DC	direct current
DVUNS	dopředná vícevrstvá umělá neuronová síť
EEG	elektroencefalografie
FBD	funkční blokový diagram
FB	funkční blok
FC	funkce
GPRS	general packet radio service
GSM	groupe spécial mobile
HMI	human machine interface
HP	Hewlet-Packard
LAD	ladder diagram
LAN	local area network
LTI	linear time-invariant
OB	organizační blok
PID	proporcionálně integračně derivační (regulátor)
PLC	programovatelný logický automat
RLY	relay
SCL	structured control language
UI	user interface
UNS	umělá neuronová síť
USB	universal serial bus

SEZNAM SYMBOLŮ PROMĚNNÝCH VELIČIN A FUNKCÍ

t	čas, s
K_p	zesílení proporcionální složky
K_i	zesílení integrační složky
K_d	zesílení derivační složky
Q_{in1}	přítok do horní nádrže, l/min
Q_{out1}	odtok z horní nádrže, l/min
Q_{in2}	přítok do horní nádrže, l/min
Q_{out2}	odtok z dolní nádrže, l/min
l_1	výška hladiny v horní nádrži, m
l_2	výška hladiny v dolní nádrži, m
l	výška obou nádrží, m
d_1	průměr obou nádrží, m
S	obsah podstavy obou nádrží, m2
d_0	průměr výtokového otvoru obou nádrží, mm
U	napětí, V
u	vstupní veličina
x	stavová veličina
$\hat{x}$	predikovaná stavová veličina
y	výstupní veličina
$\hat{y}$	predikovaná výstupní veličina
$\boldsymbol{A}$	matice systému
$\boldsymbol{B}$	matice vstupu
$\boldsymbol{C}$	váhová matice stavu
$\boldsymbol{D}$	váhová matice vstupu
F	přenosová funkce
b	hodnota zkreslení
w_i	hodnota váhy
w	řídicí veličina
y	regulovaná veličina
e	regulační odchylka
x	akční veličina
v_1, v_2	poruchové veličiny

SEZNAM ILUSTRACÍ

SEZNAM TABULEK

ÚVOD

Proces zpracování informací je jedním ze zásadních faktorů pro vývoj celé lidské společnosti. Vzhledem k současným tlakům na zvýšení efektivity napříč všemi oblastmi lidské činnosti nadále získává uchování a využití dat na důležitosti. Pro zajištění rychlé, přesné a bezpečné práce s daty jsou navrhovány komplexní měřicí systémy, které umožňují snímání, získávání dat, zpracování signálů a generování výstupů obsahujících informace reálného světa. Široké nasazení těchto systémů v nejrůznějších oblastech otevřelo cestu k prudkému vývoji v řadě základních odvětví: zemědělství, průmyslu nebo dopravě, až k novějším oborům: informační, výpočetní a komunikační technologii, biotechnologii nebo umělé inteligence (Lei, 2017).

I přes dosavadní mnohdy až překotný vývoj stále existují systémy, u nichž je perfektně známé chování projevující se navenek, ale přesný matematický popis procesů probíhajících uvnitř zůstává neznámý.

Pro praktickou ukázku řešení je vybrána soustava nádrží. Soustavu lze v tomto konkrétním případě snadno popsat pouze pomocí vstupně-výstupního chování. Pro regulování výšky hladiny v nádrži se předpokládá znalost její hodnoty, kterou ovšem nelze měřit. Řešením tohoto problému je implementace soft senzoru do PLC založeného na paradigmatu umělé neuronové sítě, který umožní hodnotu neměřené veličiny odhadovat. Na základě získané informace o stavu výšky hladiny bude možné navrhnout její řízení na požadovanou hodnotu.

1 TEORETICKÁ ČÁST

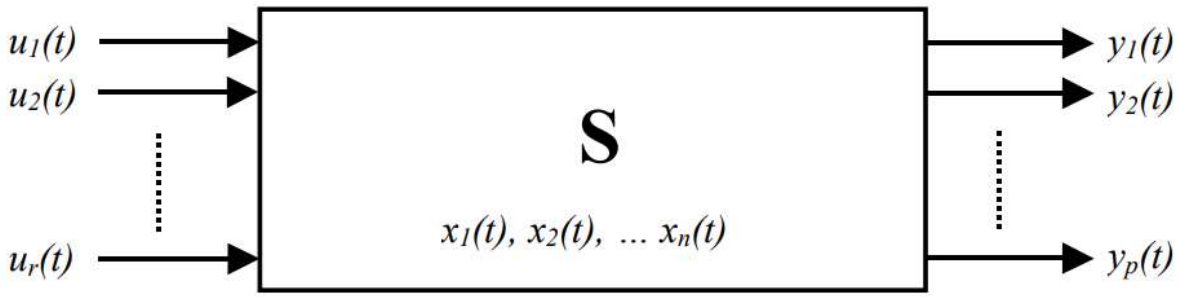
V této kapitole je nejprve nastíněna problematika estimace stavových veličin a možnosti využití soft senzorů. Následuje vhled do moderního oboru umělých neuronových sítí se zaměřením na modelováním dynamických systémů. Další kapitola je zaměřena na rodinu programovatelných logických automatů Siemens Simatic. Nakonec je popsán řídicí algoritmus využitého pro realizaci praktické části práce.

1.1 ESTIMACE STAVOVÝCH VELIČIN

Estimace neboli odhad stavu je proces určování vnitřního stavu systému, který je definován pomocí jednotlivých stavových veličin. Stavová veličina je označení pro časovou funkci, která je schopná určit vnitřní stav, ale také výstupy systému. Stavové veličiny většinou označují měřitelné veličiny obsažené v systému, ale obecně se nejedná o nutnost. K estimaci lze využít spojení matematického modelu systému s měřením vstupních a výstupních dat. Algoritmy odhadu stavu jsou základem mnoha úloh analýzy, monitorování a správy systémů. Znalost vnitřního stavu systému je nezbytná pro řešení řady problémů řízení. Řešení by mohlo spočívat v návrhu umístění velkého množství senzorů všude a na všechno. Ve většině aplikací však není realizovatelné, natož praktické, pokrýt celý systém senzory. Takové řešení by bylo neúměrně drahé, obtížné na správu, případně by mohlo narušovat původní design (Moura, 2018).

1.2 STAVOVÝ POPIS DYNAMICKÝCH SYSTÉMŮ

Stavový popis dynamických systémů je vyjádřen pomocí matematického aparátu, který zahrnuje do popisu dynamických systémů stav systému a současně jeho vnitřní uspořádání. Většinou se používá pro složitější systémy s více vstupy a výstupy.



Obr. 1.1 - Obecný model systému pro stavový popis (Švarc, 2003)

Jednotlivé složky vstupních $\boldsymbol{u}$(t), stavových $\boldsymbol{x}$(t) a výstupních $\boldsymbol{y}$(t) veličin se uskupují do sloupcových vektorů. Popis vztahů těchto veličin lze vyjádřit soustavou diferenciálních rovnic prvního řádu a systémem algebraických rovnic ve složkovém nebo maticovém tvaru, kde f a g jsou funkce. Rovnice lineárně časově invariantního (dále LTI) systému lze vyjádřit

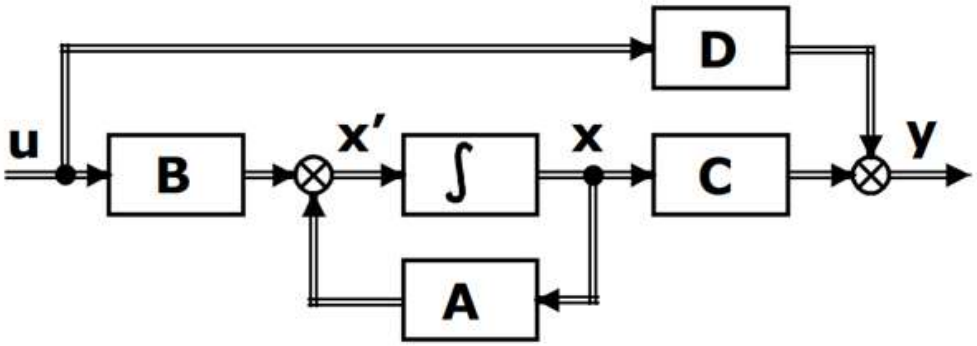
$$\boldsymbol{x}'(t) = \boldsymbol{f}[\boldsymbol{x}(t), \boldsymbol{u}(t)], \tag{1.1}$$

$$\boldsymbol{y}(t) = \boldsymbol{g}[\boldsymbol{x}(t), \boldsymbol{u}(t)], \tag{1.2}$$

$$\boldsymbol{x}'(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t), \tag{1.3}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t). \tag{1.4}$$

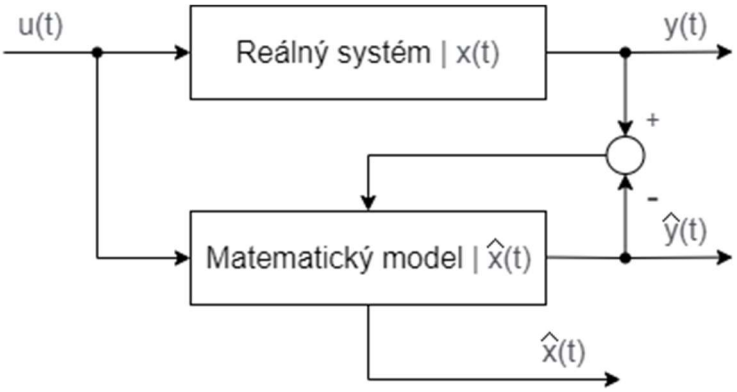
Stavová rovnice (1.3) se nazývá rovnice dynamiky a rovnice (1.4) je rovnice výstupu. Souhrnně se tyto rovnice nazývají stavové rovnice spojitého lineárního stacionárního systému. Prvky matic $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$ jsou v čase konstantní a plně určují vnitřní popis systému (Švarc, 2003).



Obr. 1.2 - Blokové schéma spojitého dynamického systému (Švarc, 2003)

1.3 STAVOVÝ POZOROVATEL

Stavového pozorovatele popisuje obr. 1.3. Horní blok představuje obecný reálný systém. Do něj vstupuje vektor vstupních veličin $\boldsymbol{u}$(t). Na určitý počet míst lze umístit snímače, které umožňují měřit příslušné proměnné veličiny. Souhrnně jsou označeny vektorem $\boldsymbol{y}$(t). Vnitřní stav $\boldsymbol{x}$(t) je neznámý. Dolní blok reprezentuje matematický model. Tento model se jinak nazývá jako stavový pozorovatel. Modelu jsou předávány stejné vstupní proměnné jako reálné soustavě a díky tomu model poskytuje odhad vnitřního stavu označený $\hat{\boldsymbol{x}}$(t). Jelikož je velmi obtížné matematický model sestavit tak, aby bezchybně popisoval dynamiku reálného systému, predikovaný výstup $\hat{\boldsymbol{y}}$(t) je porovnáván se skutečným $\boldsymbol{y}$(t). Rozdíl mezi těmito vektory je následně zahrnut do modelu tak, aby byl odstraněn (Moura, 2018).

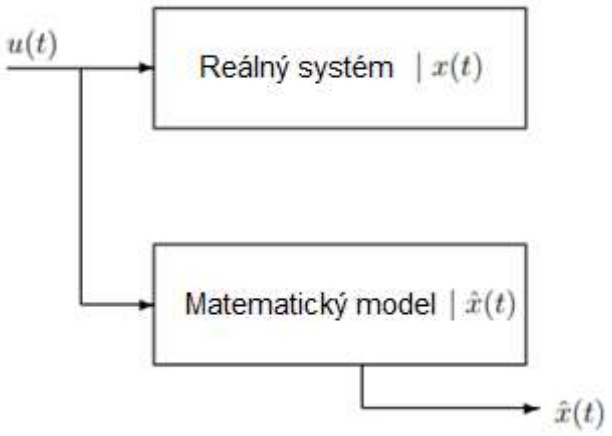


Obr. 1.3 - Obecný pozorovatel stavu (Moura, 2018)

Stavové veličinou nemusí být vždy měřitelné. Může se jednat o lineární kombinace různých veličin, ale i o více abstraktní stavové veličiny. Pro odhad měřitelných veličin se využívají estimátory, pro které se také používají označení soft senzory.

1.3.1 Pozorovatel s otevřenou smyčkou

Definování pozorovatele s otevřenou smyčkou je důležité pro stanovení motivace zařazení zpětné vazby. Na obr. 1.4 je vidět, že rozdíl výstupů reálného systému a výstupu matematického modelu není přiváděn jako korekce do modelu.



Obr. 1.4 - Blokový diagram pozorovatele s otevřenou smyčkou (Moura, 2018)

Matematický popis LTI systému a modelu je

$$\boldsymbol{x}'(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t), \qquad \boldsymbol{x}(0) = \boldsymbol{x_0}, \tag{1.5}$$

$$\hat{\boldsymbol{x}}'(t) = \boldsymbol{A}\hat{\boldsymbol{x}}(t) + \boldsymbol{B}\boldsymbol{u}(t), \qquad \hat{\boldsymbol{x}}(0) = \hat{\boldsymbol{x}}_{\boldsymbol{0}}. \tag{1.6}$$

Chybový stav je definován jako $\tilde{\boldsymbol{x}}(t) = \boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)$. V průběhu sběru dat je obecným cílem chybu zmenšovat a směřovat k nule. Dynamiku chyby lze odvodit

$$\begin{aligned} \tilde{\boldsymbol{x}}'(t) &= \boldsymbol{x}'(t) - \hat{\boldsymbol{x}}'(t), \\ &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) - \boldsymbol{A}\hat{\boldsymbol{x}}(t) - \boldsymbol{B}\boldsymbol{u}(t), \\ &= \boldsymbol{A}\big(\boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)\big), \\ &= \boldsymbol{A}\tilde{\boldsymbol{x}}(t), \qquad \tilde{\boldsymbol{x}}(0) = \boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0 = \tilde{\boldsymbol{x}}_0. \end{aligned} \tag{1.7}$$

Systém může být marginálně nebo asymptoticky stabilní. Jestliže reálná část vlastních čísel $\boldsymbol{A}$ je menší nebo rovna nule, tj. $Re[\lambda_i] \leq 0$ pro všechna i, kde λ_i jsou vlastní čísla $\boldsymbol{A}$, pak je systém marginálně stabilní. Jestliže reálná část vlastních čísel $\boldsymbol{A}$ je striktně menší než nula, tj. $\boldsymbol{Re}[\lambda_i] < 0$, pro všechna i, kde λ_i jsou vlastní čísla **A**, pak je systém asymptoticky stabilní. Z toho plyne, že v případě asymptoticky stabilního reálného systému a dokonalého matematického systému se v čase odhad stavu přibližuje reálnému stavu.

Dokonale se držet teoretických předpokladů je v praxi často nemožné. Z toho plynou tři problematické body:

- Reálné systémy nejsou vždy asymptoticky stabilní,
- matematické modely dokonale nepopisují reálnou soustavu,
- i přes idealizované podmínky je rychlost konvergence odhadů stejná s rychlostí reálného systému.

Výše uvedené skutečnosti znemožňují přesný odhad stavu reálného systému pozorovatelem s otevřenou smyčkou. Zavedení zpětné vazby (výstupní chyby) do modelu umožňuje problémy zmírnit (Moura, 2018).

1.3.2 Pozorovatelnost

Pozorovatelnost je vlastnost, která závisí na dynamice systému a na měřených veličinách. Nezávisí na množství vstupních a výstupních dat. Pokud systém splňuje tuto vlastnost, je možné návrh algoritmu pozorovatele stavu realizovat.

Definice: LTI systém je pozorovatelný, jestliže pozorováním vstupu $\boldsymbol{u}$(t) a výstupu $\boldsymbol{y}$(t) na konečném časovém horizontu t lze určit počáteční stav systému $\boldsymbol{x}(t_0)$ (Moura, 2018).

1.3.3 Luenbergerův pozorovatel

Po předchozích zjištěních nutností měření zpětné vazby a kontroly podmínky pozorovatelnosti je možné definovat jednoduchého Luenbergerova pozorovatele stavu.

Opět je uvažován LTI systém s nulovými počátečními podmínkami. Podobně jako pozorovatel s otevřenou smyčkou bude Luenbergerův pozorovatel využívat kopii reálného systému, ovšem s jedním rozdílem

$$\hat{\boldsymbol{x}}'(t) = \boldsymbol{A}\hat{\boldsymbol{x}}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{L}[\boldsymbol{y}(t) - \hat{\boldsymbol{y}}(t)], \qquad \hat{\boldsymbol{x}}(0) = \hat{\boldsymbol{x}}_{\boldsymbol{0}}, \tag{1.8}$$

$$\hat{\boldsymbol{y}}(t) = \boldsymbol{C}\hat{\boldsymbol{x}}(t) + \boldsymbol{D}\boldsymbol{u}(t). \tag{1.9}$$

Výraz $\boldsymbol{L}[\boldsymbol{y}(t) - \hat{\boldsymbol{y}}(t)]$ vyjadřuje chybu mezi měřením a předpovědí modelu. Velikost hodnoty tohoto výrazu je klíčovou vlastností návrhu Luenbergerova pozorovatele. Zesílení $\boldsymbol{L}$ je uživatelsky definovatelné a může nabývat hodnot $\mathbb{R}^{n\times q}$ ($\mathbb{R}$ - množina reálných čísel, n – počet dimenzí LTI systému, q – počet výstupů LTI systému). Pro lepší pochopení je uvažována dynamika chyby $\tilde{\boldsymbol{x}}(t) = \boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)$, která se vyvíjí následovně

$$\begin{aligned}
\tilde{\boldsymbol{x}}'(t) &= \boldsymbol{x}'(t) - \hat{\boldsymbol{x}}'(t), \qquad (1.10)\\
&= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) - \boldsymbol{A}\hat{\boldsymbol{x}}(t) - \boldsymbol{B}\boldsymbol{u}(t) - \boldsymbol{L}[\boldsymbol{y}(t) - \hat{\boldsymbol{y}}(t)]\,,\\
&= \boldsymbol{A}\big(\boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)\big) - \boldsymbol{L}[\boldsymbol{C}\boldsymbol{x}(\boldsymbol{t}) + \boldsymbol{D}\boldsymbol{u}(\boldsymbol{t}) - \boldsymbol{C}\hat{\boldsymbol{x}}(t) - \boldsymbol{D}u(t)],\\
&= \boldsymbol{A}\big(\boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)\big) - \boldsymbol{L}\boldsymbol{C}(\boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)),\\
&= (\boldsymbol{A} - \boldsymbol{L}\boldsymbol{C})\,\hat{\boldsymbol{x}}(t), \qquad \tilde{\boldsymbol{x}}(0) = \boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0 = \tilde{\boldsymbol{x}}_0.
\end{aligned}$$

Odhad chyby je asymptoticky stabilní, jestliže je $\boldsymbol{L}$ voleno tak, aby ($\boldsymbol{A}$-$\boldsymbol{LC}$) mělo zápornou reálnou část. Pro popis téhož lze využít koncept pozorovatelnosti a dříve zmíněné shrnout následující větou. Je-li dvojice matic $\boldsymbol{A}$ a $\boldsymbol{C}$ LTI systému pozorovatelná, tak vlastní hodnoty výrazu ($\boldsymbol{A}$-$\boldsymbol{LC}$) mohou být libovolně umístěny v komplexní rovině (Moura, 2018).

1.3.4 Kalmánův filtr

Časově proměnný lineární kvadratický estimátor, Kalmánův filtr, je jedním z nejvyužívanějších estimátorů v oblasti systémů a řídicí techniky. Dokáže rekurzivně operovat s nepřesným modelem a se zašuměnými toky dat tak, že vypočítá statisticky optimální odhad výchozího stavu systému. Při odhadu stavových veličin optimálně vyvažuje relevanci modelu a reálných dat.

Model LTI systému je uvažován ve tvaru

$$\boldsymbol{x}'(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{w}(t), \qquad \boldsymbol{x}(0) = \boldsymbol{x}_{\boldsymbol{0}}, \quad \boldsymbol{x}, \boldsymbol{w} \in \mathbb{R}^n, u \in \mathbb{R}^p, \tag{1.11}$$

$$\boldsymbol{y}_{\boldsymbol{m}}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t) + \boldsymbol{n}(t), \qquad \boldsymbol{y}_m, \boldsymbol{n} \in \mathbb{R}^q, \tag{1.12}$$

kde $\boldsymbol{x}$(t) je vstupní veličina

$\boldsymbol{y}_{\mathbf{m}}$(t) je výstupní veličina

$\boldsymbol{w}$ a $\boldsymbol{n}$ jsou šumové členy představující nepřesnost modelu a šum snímačů.

Následující algoritmus generuje odhad stavového vektoru $\hat{\boldsymbol{x}}$, který minimalizuje střední kvadratickou chybu s použitím časově proměnného zesílení $\boldsymbol{L}$(t), které lze vypočítat pomocí Riccatiho diferenciální rovnice

$$\hat{\boldsymbol{x}}'(t) = \boldsymbol{A}\hat{\boldsymbol{x}}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{L}(\boldsymbol{t})(\boldsymbol{y}_{\boldsymbol{m}} - \hat{\boldsymbol{y}}), \qquad \hat{\boldsymbol{x}}(0) = \bar{\boldsymbol{x}}, \tag{1.13}$$

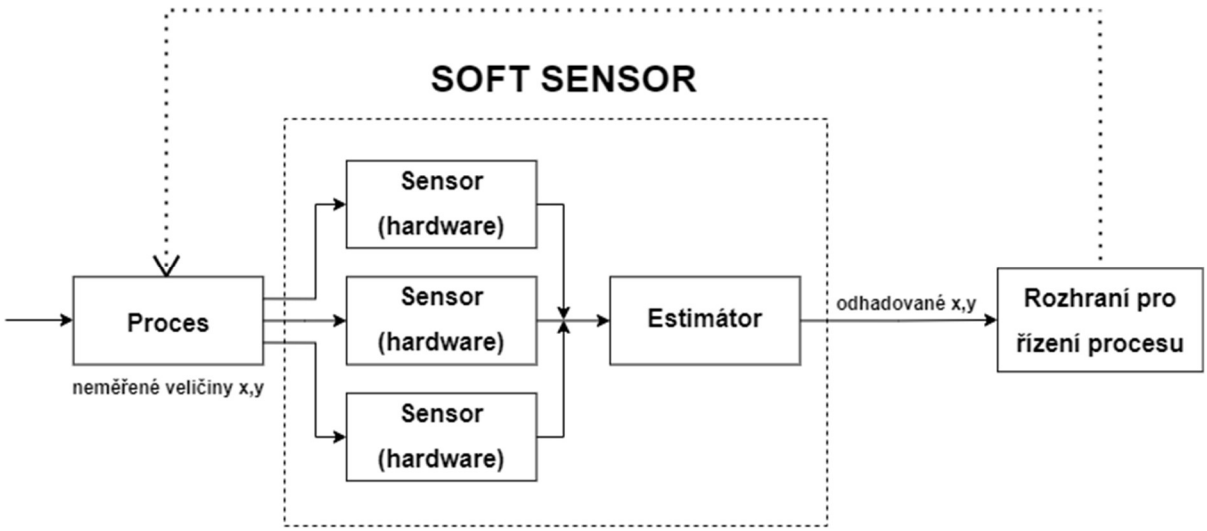
$$\hat{\boldsymbol{y}}(t) = \boldsymbol{C}\hat{\boldsymbol{x}}(t) + \boldsymbol{D}\boldsymbol{u}(t). \tag{1.14}$$

Hlavní nevýhodou Kalmánova filtru je předpoklad lineárního dynamického systému. Ovšem většina systémů v reálném světě jsou nelineární. Rozšířený Kalmánův filtr pracuje s linearizovanými rovnicemi modelu kolem aktuálního stavového odhadu (Moura, 2018).

1.4 SOFT SENZORY

Soft sensor je obecné označení pro model, který zpracovává měřené signály a může predikovat hodnoty signálů neměřených. Toto označení vzniklo kombinací dvou slov. Prvním slovem je „software“, protože modely pro vyhodnocování signálů jsou zpravidla implementovány v programech. Druhé slovo je „sensor“, jelikož modely zprostředkovávají podobné informace jako klasické hardwarové sensory (Kadlec, 2008).

Řada průmyslových zařízení je osazena obrovským počtem sensorů pro monitorování a řízení procesů. Přibližně před dvaceti lety s rozvojem vyspělejších technologií začali inženýři data ze sensorů cíleně shromažďovat. Data jim pomáhala s vylepšováním daných aplikací a umožňovala jim více se zaměřit na tvorbu predikčních modelů. Moderní soft sensory se prosadily jako cenná alternativa k tradičním prostředkům pro získávání kritických proměnných a pro monitorování a řízení procesu (Kadlec, 2008).



Obr. 1.5 - Obecný popis principu soft sensoru (Paulsson, 2014)

1.4.1 Typy

Obecně rozlišujeme dva typy soft sensorů, které jsou naprosto odlišné v oblasti znalosti procesu. Pokud je znám jeho přesný popis včetně všech podrobností, jedná se o modelem řízené soft senzory – tzv. white-box. Modely jsou založeny na rovnicích popisujících fyzikální, případně chemické principy. Typickým příkladem je použití zákonů zachování fyzikálních veličin. Přesné rovnice ovšem nejsou známé vždy. Často rovnice popisují pouze teoretické pozadí procesu a nedokážou přesně a zároveň srozumitelně vystihnout skutečné podmínky.

Neexistují-li informace o procesu, tak se používají datově řízené soft sensory – tzv. black-box. Takový senzor je založen na empirických pozorováních procesu. Používají se především pro vylepšení sady nástrojů, jako jsou diagnostické a predikční metody.

V praxi se lze setkat se systémy, o kterých jsou informace nekompletní, a proto existuje řada hybridních kombinací obou typů soft sensorů – tzv. grey-box (Kadlec, 2008).

1.4.2 Vývoj

Vývoj soft sensoru lze obecně rozdělit do několika fází. Počátečním krokem je první kontrola dat. Cíle tohoto kroku jsou: získání přehledu o datech, vhodné seskupení a kontrola zjevných chyb dat. Následně lze zvážit požadavky na složitost modelu a odhadnout jeho budoucí strukturu. Ve druhém kroku je potřeba vybrat vhodná data, která budou dále použita pro trénování a hodnocení modelu. Třetí krok se zaměřuje na transformaci dat tak, aby bylo dosaženo efektivního uspořádání pro zpracování modelem. Především je nutné řešit problémy, které jsou podrobněji popsané v kapitole 1.4.4. Krok číslo čtyři se zabývá trénováním,

kontrolou a výběrem modelu (Kadlec, 2008). Prozatím neexistuje jednotný teoretický přístup k jeho praktickému řešení.

K vývoji soft senzorů je většinou nutné přistupovat individuálně. Jednou z mála obecně použitelných technik je použití postupného zvyšování složitosti modelu až do momentu, kdy je dosaženo podstatného zlepšení výkonu modelu (Kadlec, 2008).

1.4.3 Využití

Soft sensory se v současnosti využívají při řešení rozmanitých úkolů. Nejrozšířenější použití je v oblasti predikce procesních proměnných, které lze určit on-line pro nízké hodnoty vzorkovací frekvence nebo pouze pomocí off-line analýzy měřených dat (Kadlec, 2008).

Další oblastí aplikace soft sensorů je monitorování a detekce poruch v systému. Zjištěný aktuální stav procesu je dále vyhodnocován a v případě, že se odchyluje od normálního stavu, je obvykle možné určit zdroj této chyby. Vizualizace správy celého zařízení včetně hlášení detekce chyb je obvykle zajištěna ve velínu. Soft sensory umožňují operátorům ve velínu včasně a efektivně reagovat na nestandardní chování systému (Kadlec, 2008).

Soft sensory lze využít i jako záložní čidla v případě poruchy hardwarového sensoru. Pokud soft senzor funguje stejně dobře jako fyzický sensor, lze ho využít jako obvyklé měřicí zařízení v běžných pracovních podmínkách. Výhoda spočívá především v absenci mechanických poruch v důsledku opotřebení. Softwarový kód se snadněji udržuje, upravuje a vylepšuje (Kadlec, 2008).

1.4.4 Problémy a budoucí vývoj

Dva hlavní problémy soft senzorů se nachází v oblastech vývoje a údržby. Při návrhu nového sensoru je nezbytné věnovat velké množství času a práce ručnímu předzpracování dat a výběru funkčního modelu včetně jeho ověření. Při řešení je nutné zohlednit několik problémů, které se týkají zpracovávaných signálů z hardwarových snímačů:

- Chybějící data,
- nesprávná (odchýlená) data,
- nesprávná (unášená) data,
- kolinearita dat,
- vzorkovací frekvence snímání dat,
- dopravní zpoždění dat.

Při řešení těchto problémů vývojář většinou postupuje iterativně. Odstraňuje jednotlivé problémy a kontroluje vliv změn na ostatní části modelu a dále ho ladí. Častou strategií bývá

snaha o zahrnutí maximálního množství procesních dat do modelu. Nevýhodou je nutnost individuálního řešení každého soft sensoru ve vztahu k odlišným procesům. Druhou možností je implementovat do modelu variantní metody řešení včetně mechanismu pro výběr aktuálně nejvhodnější (Kadlec, 2008).

1.5 UMĚLÉ NEURONOVÉ SÍTĚ

Lidský mozek prošel velmi dlouhou evolucí, během které nabyl mnoha pozoruhodných schopností, jako jsou paralelní zpracování informací, učení, zobecňování, přizpůsobivost, zpracovávání informací v širším kontextu, odolnost vůči chybám. Softwarovým napodobením alespoň některých těchto vlastností se zabývá obor umělých neuronových sítí. Jejich cílem je navržení výkonných optimalizovaných výpočetních modelů. Základním stavebním prvkem UNS je umělý neuron, který je kopií biologického neuronu. Tyto procesorové jednotky se v modelech využívají ve větších počtech a dohromady vytvářejí síť (Jain, 1996; Zakaria, 2014).

1.5.1 Historické kořeny

První velmi jednoduchý model umělého neuronu navrhli neurofyziolog Warren McCulloch a logik Walter Pitts již v roce 1943. Zpočátku jejich model pracoval s binárními vstupy a výstupy. Úspěšně implementovali logické funkce AND, OR a další. Následně se jim podařilo model modifikovat tak, aby počítal s libovolnou aritmetikou nebo funkcí (Šíma, 1996).

Dalším průkopníkem byl Donald Hebb. Ve své práci popsal a vysvětlil myšlenku podmíněných reflexů živočichů do učicího pravidla pro synapse. Tento objev veřejně publikoval v roce 1949 v knize The Organization of Behavior (Organizace chování) (Šíma, 1996).

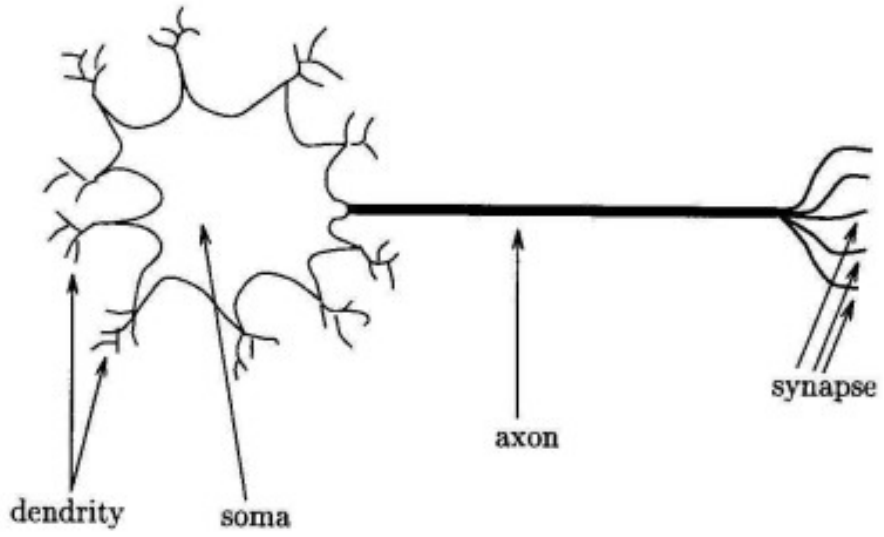
Po kratším období bez průlomových objevů Frank Rossenblatt pokročil ve své práci a v roce 1957 se mu podařilo zobecnit model perceptronu. Zároveň vymyslel učicí algoritmus pro určení váhového vektoru parametrů (Šíma, 1996).

V období vymezeném koncem padesátých a koncem šedesátých let byl výzkum přesměrován k vývoji neuropočítačů. Jednalo se o specializované systémy, které byly využívány nejprve ve vojenské, poté i v civilní sféře. Tento typ zařízení byl ve své době označován jako „univerzální počítač šesté generace“ (Šíma, 1996).

1.5.2 Biologický neuron

Speciální procesní jednotka se skládá ze tří hlavních částí: těla buňky, dendritů a axonu. Tělo buňky neboli soma má jádro a kolem něj plazmu. Jádro obsahuje informace o dědičných

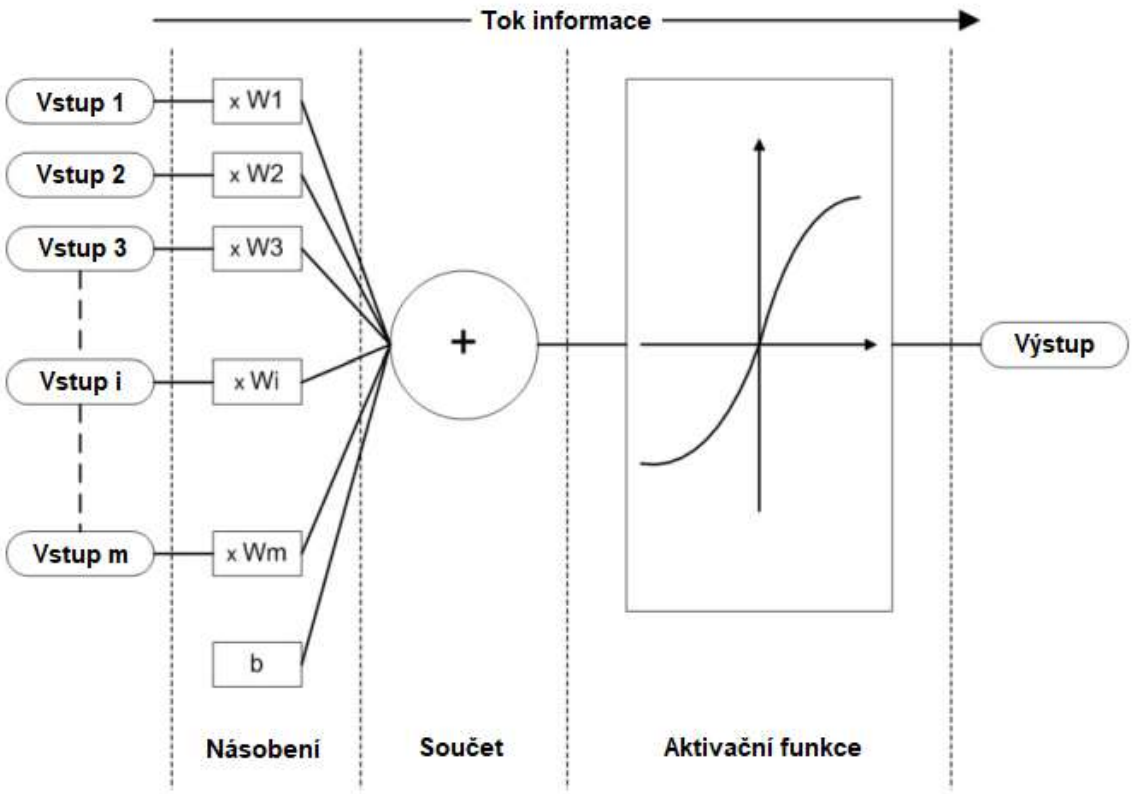
vlastnostech. V plazmě jsou stavební látky nezbytné pro vlastní existenci neuronu. Pro komunikaci je neuron vybaven přijímači a vysílači. Komunikační přijímače, krátké výběžky dostředivého typu, jsou označovány slovem dendrity. Pro vysílání informací má neuron jeden axon, který se na svém konci rozvětvuje. Jejich propojení se nazývá synapse. Jakmile impuls vyslaný z těla buňky doputuje přes axon do přestupního terminálu synapse, uvolní se chemické látky zvané neurotransmitery. V závislosti na typu synapse buď posilují nebo naopak potlačují vlastní tendenci vysílat impulzy. To lze definovat jako schopnost učení a pamatování. Každý neuron v lidském mozku je propojen s dalšími tisící až desetitisíci neurony a v celkovém součtu mozek řádově obsahuje 10^{14} až 10^{15} spojení (Šíma, 1996).



Obr. 1.6 - Biologický neuron (Novák, 1993)

1.5.3 Umělý neuron

Základními stavebními prvky umělé neuronové sítě jsou jednotlivé neurony. Jejich konstrukce a funkce jsou odvozeny ze zkoumání a pozorování biologických neuronů. Tyto jednotky představují jednoduchý matematický model, který vykonává tři po sobě jdoucí operace: násobení, sčítání a aktivace (Suzuki, 2011).



Obr. 1.7 - Umělý neuron (Suzuki, 2011)

Prostřednictvím individuálně vážených vstupů přicházejí do těla umělého neuronu jednotlivé informace. Váhy spojení určují propustnost informací mezi jednotlivými neurony. Váhy se v průběhu učení mění. Jejich optimální nastavení zaručuje shodu mezi výstupem neuronové sítě a výstupem reálného procesu. Tělo umělého neuronu vážené vstupy sečte, včetně prahu neuronu a tento součet dále zpracuje pomocí přenosové, aktivační, funkce. Práh neuronu označuje bariéru mezi vstupem neuronu a okolním světem. V poslední fázi umělý neuron předá zpracovávanou informaci prostřednictvím výstupu. Matematicky lze předchozí popsat

$$y(k) = F\left(\sum_{i=0}^{m} w_i(k)\, x_i(k) + b\right), \tag{1.15}$$

kde $x_i(k)$ je vstupní proměnná v diskrétním čase k, i náleží do intervalu od 0 do m,

$w_i(k)$ je hodnota váhy v diskrétním čase k, i náleží do intervalu od 0 do m,

b je práh neuronu,

F je přenosová funkce,

$y_i(k)$ je výstupní proměnná v diskrétním čase k.

Hlavní neznámou modelu umělého neuronu je vhodná volba přenosová funkce, jíž jsou definovány vlastnosti neuronu. Ve své podstatě to může být jakákoli matematická funkce. Její výběr je závislý na definici problému, který má umělý neuron řešit (Doležel, 2016).

Nejjednodušší možností je použití skokové funkce. Jedná se o bipolární funkci, která má pouze dvě možné výstupní hodnoty (např. jedničku a nulu). Pokud vstupní hodnota splňuje určitou prahovou hodnotu, výstupní hodnota je jedna, v opačném případě nula. Umělý neuron, ve kterém je použit tento typ přenosové funkce, se nazývá perceptron. Používá se především pro klasifikační úlohy (Doležel, 2016).

Mezi další často využívané přenosové funkce se řadí lineární (1.16), nelineární sigmoidní (1.17), hyperbolicko-tangenciální (1.18) nebo Gaussova (1.19). Jejich matematický popis je

$$y = y_a, \tag{1.16}$$

$$y = \frac{1}{1 + e^{-y_a}}, \tag{1.17}$$

$$y = tanh\,(y_a), \tag{1.18}$$

$$y = e^{-y_a^2}, \tag{1.19}$$

kde y je výstupní proměnná,

y_a je aktivační funkce.

1.5.4 Učení

Základní funkcí lidského mozku je schopnost učení. Tento proces je nezbytné přenést do paradigmatu umělých neuronových sítí. Obvykle se při učení upravují hodnoty vah a prahů, které určují sílu spojení mezi neurony. Nejstarší pravidlo, které bylo základem pro další formování strategií učení neuronu, definoval Donald Hebb při studiu podmíněných reflexů. Jeho hlavní myšlenka se zakládá na možnostech posilování a oslabování vazeb mezi jednotlivými neurony. Pro neuron s binárním vstupem x, vahami w, výstupem y a předpokládaným výstupem y_d je Hebbovo pravidlo definováno v (Blaha, 2015):

1. Pokud je neuron excitován korektně (y = 1; y_d = 1), pak se v příštím diskrétním časovém kroku $n+1$ posilují o Δ spoje w_i, které tuto excitaci vyvolaly:
2. Pokud je neuron excitován nekorektně (y = 1; y_d = 0), pak se oslabují o delta spoje, které tuto excitaci vyvolaly:
3. Pokud není neuron excitován (y = 0), nic se neděje (váhy se nemění)

Učení s učitelem

Jedná se o techniku strojového učení, která upravuje parametry neuronové sítě pomocí trénovacích dat. Tato množina se skládá z dvojic vstupních a požadovaných výstupních hodnot. Obvykle jsou data seskupována do vektorové podoby. Pro správné vykonání celého procesu učení s učitelem je nutné splnit řadu dílčích úkolů.

Nejprve je potřeba určit typ trénovací množiny. Poté se musí data v dostatečném množství nashromáždit. Následuje sestavení sesbíraných dat do srozumitelné formy pro neuronovou síť. Po dokončení předchozích kroků je možné přistoupit k samotnému učení a po dokončení pomocí testovací množiny zkontrolovat kvalitu modelu umělé neuronové sítě. Množina dat pro otestování se skládá z dat, která nebyla využita pro proces učení (Zakaria, 2014).

Učení bez učitele

Technika strojového učení bez učitele nastavuje parametry neuronové sítě na základě trénovacích dat a minimalizace účelové funkce. Její definování záleží na formulaci řešeného problému. Obecně je tento typ učení vhodný pro aplikace statistického modelování, komprese, filtrování a shlukování. Umělé neuronové síti jsou předkládána pouze neoznačená data, ve kterých jsou nalézány všechny možné druhy neznámých vzorců a jsou utřiďovány na základě podobností. Oproti učení s učitelem je velkou výhodou snazší získávání trénovacích dat, které není nutné třídit a ručně upravovat. Na druhou stranu je učení bez učitele výpočetně složitější a méně přesné (Zakaria, 2014).

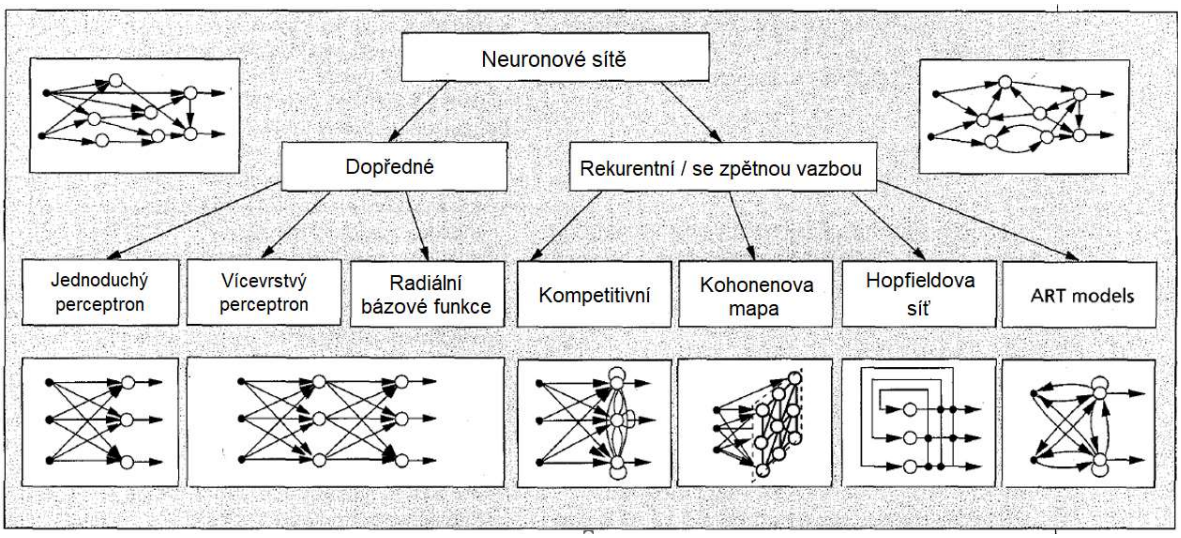
Učení s posilováním

Tento způsob učení umělé inteligence je přirovnáván k analogii výchovy psa v domácnosti. Spočívá v maximalizaci dlouhodobého zisku pomocí motivace pozitivní odměnou. V případě splnění vyžadovaného chování je pes pozitivně motivován pamlskem. V opačném případě je pes upozorněn na nesprávné chování zvýšeným hlasem nebo přísným pohledem.

Pro umělou neuronovou síť je definována tzv. funkce návratnosti. Pro různé výstupy sítě jsou počítány návratnosti a v závislosti na jejich hodnotách jsou udělovány pozitivní nebo negativní odměny. Tato ocenění mohou přicházet i se zpožděním, a proto se učení s posilováním využívá pro řešení aplikací, u nichž je akceptována krátkodobá ztráta a preferován velký dlouhodobý zisk (Zakaria, 2014).

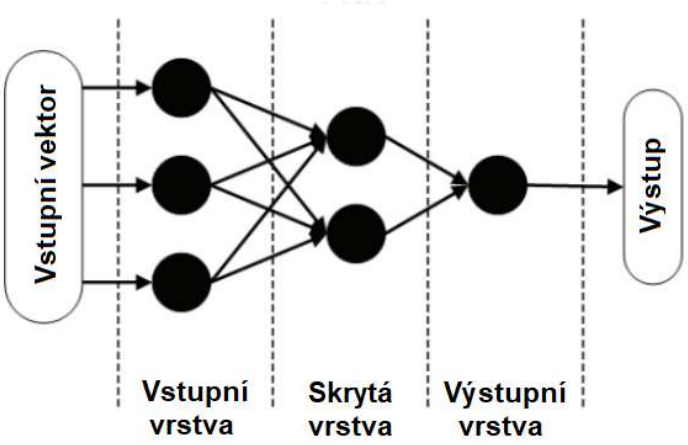
1.5.5 Topologie

Umělá neuronová síť vzniká spojením dvou a více neuronů. Neuronové sítě jsou na rozdíl od samostatných neuronů schopny řešit složité problémy. Slova topologie a architektura označují způsob propojení jednotlivých neuronů.

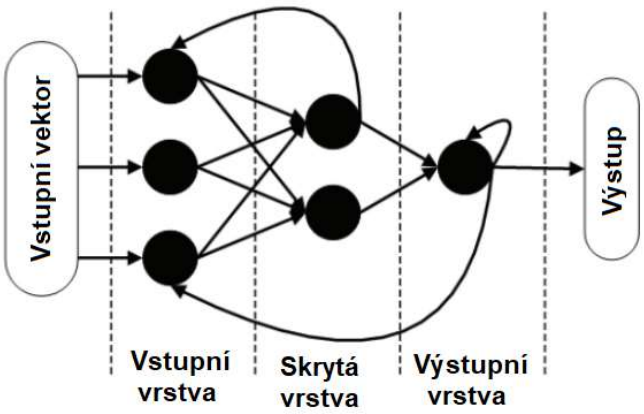


Obr. 1.8 – Dělení umělých neuronových sítí (Suzuki, 2011)

Existují dvě základní kategorie architektury neuronových sítí, které se odlišují směrem toku signálu. Na jejich základech jsou vystavěny všechny ostatní architektury. V dopředné topologii informace proudí postupně jedním směrem ze vstupu na výstup. Rekurentní architektura umožňuje části informací proudit i směrem zpět. Na obr. 1.9 a 1.10 jsou obě topologie znázorněny. Jsou tvořeny vrstevnatou strukturou. První vrstva se nazývá vstupní. Zprostředkovává vstup signálu z okolního prostředí. Následují skryté vrstvy, jejichž počet je definován konkrétní konfigurací. Výstupní vrstva se stará o poslední úpravy signálů a určuje finální podobu vypočteným výsledkům. (Suzuki, 2011).



Obr. 1.9 - Dopředná neuronová síť (Suzuki, 2011)



Obr. 1.10 - Rekurentní neuronová síť (Suzuki, 2011)

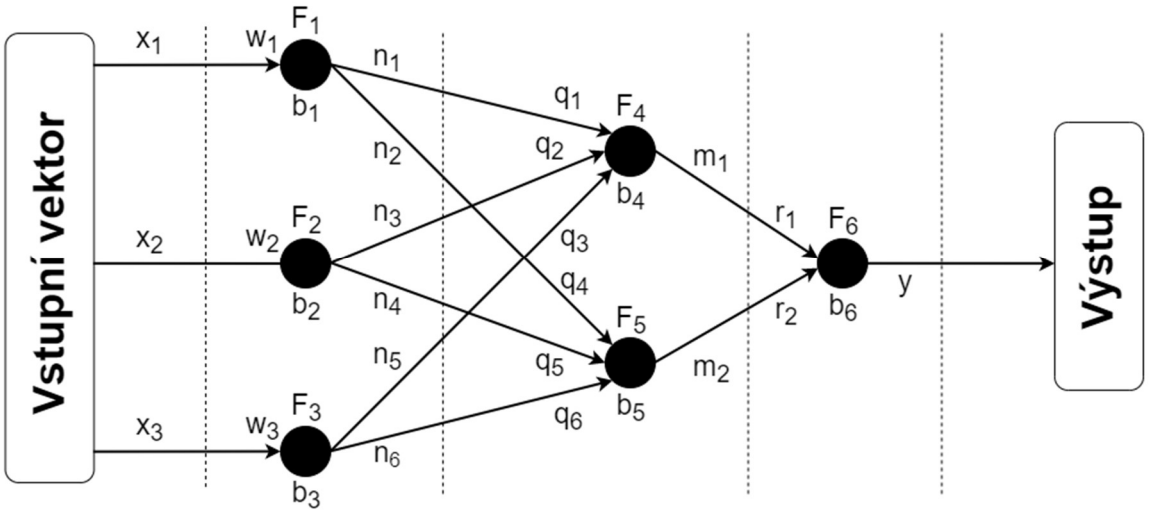
Dopředná vícevrstvá umělá neuronová síť

Nejvíce používanou topologií neuronové sítě je dopředná vícevrstvá. Má schopnost pracovat jako univerzální aproximátor. Dokáže využívat informace, které jsou v datech lidskému rozlišovacímu aparátu neviditelné. Dopřednou sítí informace musí směřovat pouze jedním směrem, a to od vstupu k výstupu bez zpětných smyček. Počet vrstev ve struktuře, typy přenosových funkcí použitých v jednotlivých neuronech, ani počet spojení mezi neurony není nijak omezen (Suzuki, 2011).

Obecný algoritmus pro odezvu DVUNS lze slovně shrnout do následujících vět. Vektor vstupů je dle jeho složek zaveden do vstupních terminálů. Následuje výpočet vstupních potenciálů. Poté jsou vypočítávány výstupní potenciály neuronů. Výpočty potenciálů se opakují pro neurony ve všech skrytých vrstvách až do výstupní vrstvy (Doležel, 2016).

Nejjednodušší dopřednou neuronovou sítí je jednoduchý perceptron, jehož schopností je řešit lineárně separovatelné problémy. Obecně i jednoduché neuronové sítě mohou mít dlouhá a složitá řešení.

Pro ilustraci analytického popisu je uvedena specifická neuronová síť na obr. 1.11 s třemi neurony ve vstupní vrstvě, dvěma neurony ve skryté vrstvě a jedním neuronem ve výstupní vrstvě. V praxi se pro práci s neuronovými sítěmi využívá specializovaných softwarů a pokročilé výpočetní techniky (Suzuki, 2011).



Obr. 1.11 - Příklad dopředné vícevrstvé umělé neuronové sítě

Kde *x, n, m, y* jsou signály,

w, q, r jsou váhy,

F jsou přenosové funkce,

b jsou prahy neuronu.

Odezva této konkrétní neuronové sítě se vypočítá:

$$n_1 = F_1(w_1 x_1 + b_1) \tag{1.20}$$

$$n_2 = F_1(w_1 x_1 + b_1) \tag{1.21}$$

$$n_3 = F_2(w_2 x_2 + b_2) \tag{1.22}$$

$$n_4 = F_2(w_2 x_2 + b_2) \tag{1.23}$$

$$n_5 = F_3(w_3 x_3 + b_3) \tag{1.24}$$

$$n_6 = F_3(w_3 x_3 + b_3) \tag{1.25}$$

$$m_1 = F_4(q_1 n_1 + q_2 n_3 + q_3 n_5 + b_4) \tag{1.26}$$

$$m_2 = F_5(q_4 n_2 + q_5 n_4 + q_6 n_6 + b_5) \tag{1.27}$$

$$y = F_6(r_1 m_1 + r_2 m_2 + b_6) \tag{1.28}$$

1.5.6 Praktické využití

Kvalitu výstupů UNS ovlivňuje výběr správné topologie, pro který je důležitá široká znalost teorie UNS a algoritmů učení. Při vytváření modelů je nutností hledání kompromisů

mezi příliš jednoduchou strukturou a složitou robustní strukturou, která ale často vede na velkou časovou náročnost při výpočtech.

Možnosti využití umělých neuronových sítí jsou velice široké a je velmi pravděpodobné, že se s umělými neuronovými sítěmi budeme v budoucnu stále více setkávat i v aplikacích, ve kterých dosud uplatněny nejsou.

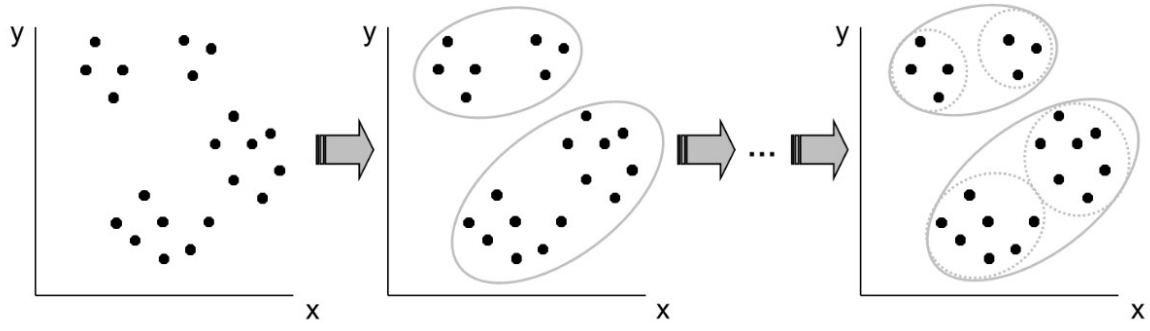
Klasifikace

Otázka klasifikace neboli třídění se zabývá uspořádáváním vstupních vzorů podle kvalitativních znaků do předdefinovaných tříd. Vzory mohou být neuronové síti předkládány v libovolné formě, například i zvukové nebo vizuální. Mezi nejčastější použití klasifikačních problémů patří rozpoznávání řeči a znaků a kontrolní systémy desek plošných spojů. V oblasti lékařství se UNS využívají pro klasifikaci krevních buněk nebo vyšetření EEG, jehož jednoduché schéma je na obr. 1.12 (Jain, 1996).

Obr 1.12 - Příklad klasifikační úlohy (Jain, 1996)

Shlukování

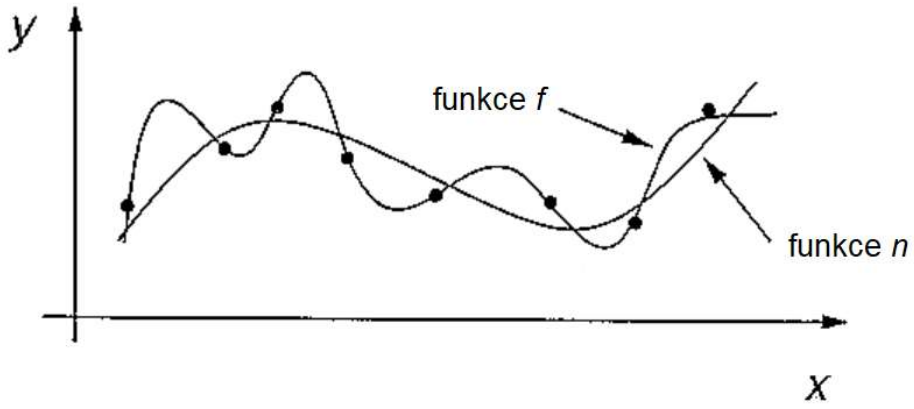
Shlukování neboli kategorizace se oproti klasifikaci odlišuje způsobem učení. Při shlukování nejsou umělé neuronové síti ve fázi učení předkládány předdefinované třídy. V tomto typu úlohy dochází k hledání podobností mezi vzory a na jejich základě jsou vzory tříděny do kategorií. Shlukování se využívá především při práci s daty v případech identifikace vzorců a vztahů, celkovém průzkumu a analýze nebo zmenšování objemu se současným zachování všech informací (Jain, 1996).



Obr. 1.13 - Příklad shlukování (Blaha, 2015)

Aproximace funkcí

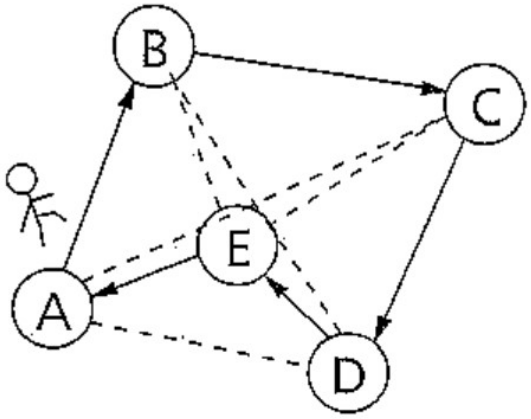
Aproximace spočívá v nahrazení zašuměné funkce *f* jinou funkcí *n*, kterou lze jednodušeji matematicky zpracovávat nebo modelovat. Řešením této problematiky se zabývají numerické metody matematické analýzy. Jedná se o často využívanou metodu především při zpracování a vyhodnocování měření a také při výpočtech určitých integrálů.



Obr. 1.14 - Příklad aproximace (Jain, 1996)

Optimalizace

Cílem optimalizačních algoritmů je nalezení efektivního řešení. V matematice je toto řešení definováno pomocí účelové funkce, která je maximalizována nebo minimalizována. Obecně se s optimalizací setkáváme v nejrůznějších oblastech lidské činnosti od matematiky, statistiky, až po lékařství nebo ekonomiku. Typickým příkladem optimalizace je problém obchodního cestujícího. Jeho snahou je navštívit většinu měst s co nejmenším úsilím, tj. urazit minimální možnou vzdálenost (Jain, 1996).



Obr. 1.15 - Příklad optimalizace (Jain, 1996)

Řízení a predikce

Umělou neuronovou síť lze vytvořit tak, aby na základě minulých dat bylo možné predikovat jejich budoucí vývoj. Tato funkčnost je v dnešní době využívána ve vědě, technologickém inženýrství, ale i ve finančním sektoru. Predikci je možné lépe provádět připojením historického vývoje vstupů. Takto modifikované neuronové sítě lze využívat jako soft senzory. Pro využití v oblasti řízení je možné navrhnout adaptivní model, jehož cílem je generování řídicího signálu (Jain, 1996).

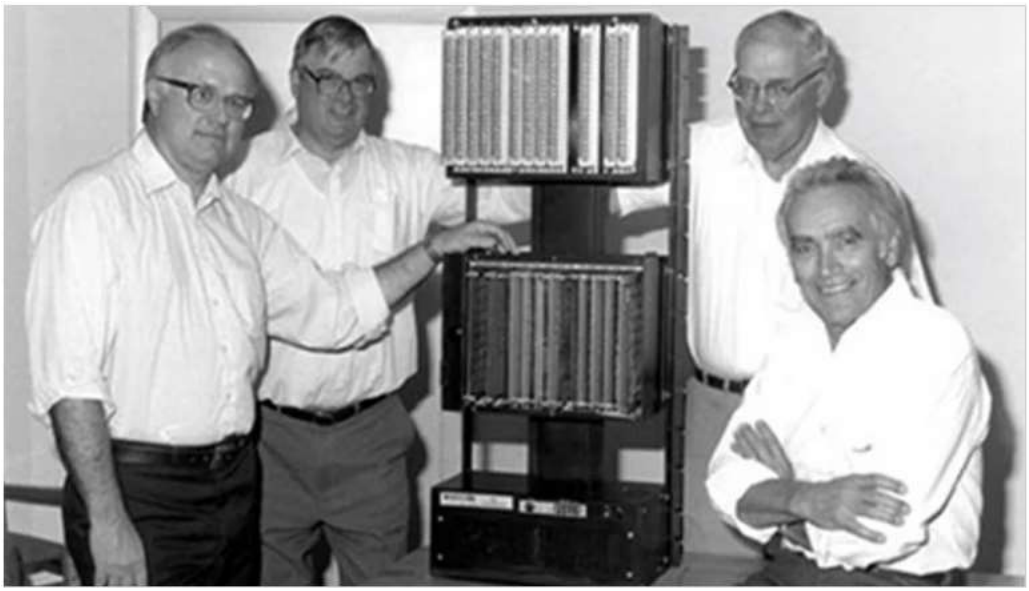
1.6 SIEMENS SIMATIC

V rozsáhlém portfoliu produktů koncernu Siemens AG mají neodmyslitelné zastoupení programovatelné logické automaty (zkráceně PLC). Jedná se o malé průmyslové počítače, které se skládají z centrální procesorové jednotky a vstupních a výstupních modulů. Jejich konstrukce je přizpůsobena tak, aby zařízení spolehlivě fungovala v prostředí s extrémní teplotou, vlhkostí, prašností, vibracemi, případně úniku chemikálií (Beresford, 2011).

1.6.1 Evoluce PLC

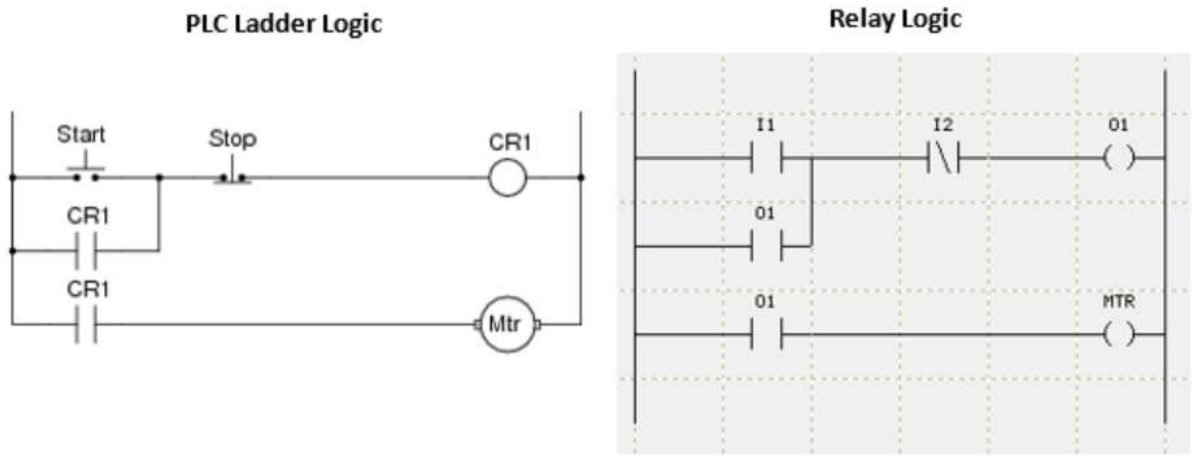
Vynález programovatelných logických automatů započal velký rozvoj v průmyslové automatizaci. Do té doby musel být každý prvek jedinečně zapojen a dále samostatně obsluhován. S příchodem PLC mohl být složitý systém reléových obvodů uložený v obrovských skříních nahrazen jediným řídicím prvkem.

První programovatelný logický automat byl představen v roce 1968. Jako první tuto technologii začala zavádět v automobilovém průmyslu společnost General Motors. Výraznou osobou, která se v této firmě starala o rozvoj nových technologií byl Richard E. Morley. Spolu s týmem vytvořil jednotku PLC Modicon model 084, která byla modulární, robustní a nepoužívala přerušení (Evolution, 2021).



Obr. 1.16 - PLC Modicon 084 a jeho vývojáři (Peterson, 2022)

Pro usnadnění přechodu z reléových systémů k PLC byl vymyšlen softwarový programovací jazyk Ladder Logic neboli žebříčkový diagram, který virtuálně kopíruje konstrukci elektrické sítě reléového logického obvodu. Byl navržen tak, aby s ním mohli pracovat nejen techničtí inženýři, ale i elektrikáři v údržbě.



Obr. 1.17 - Obecné porovnání reléové a ladder logiky (Nanjundaiah, 2019)

V následujících letech byly vyvíjeny a modifikovány programovatelné logické automaty tak, aby vyhovovaly různým aplikacím a prostředím. Konkurenční prostředí vývoje těchto zařízení vedlo k vytvoření systémů pro vzájemnou výměnu dat mezi PLC. S rozšiřováním této pokročilé technologie došlo k vytvoření standardizovaného programovacího pravidel podle normy IEC 61131-3.

V počátcích devadesátých let dvacátého století došlo k rozvoji monitorovacích terminálů pro jednotlivá zařízení nebo celé procesy. Tato zařízení se souhrnně nazývají podle rozhraní člověk-stroj, HMI (human machine interface). Snadné zobrazení chybových stavů usnadnilo technikům práci a ušetřilo čas při jejich odstraňování (Evolution, 2021).

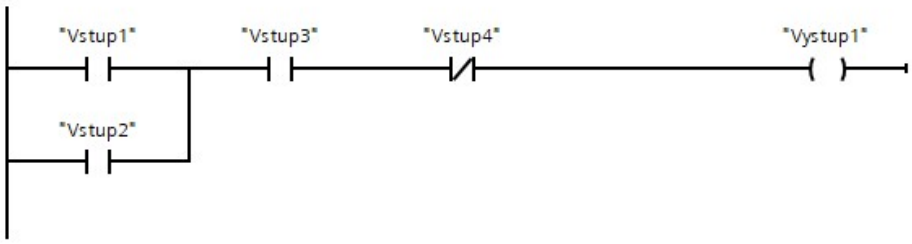
Současná podoba PLC je velmi rozdílná oproti původním zařízením. Jejich velikost je mnohem menší, a naopak disponují násobně větší výpočetní rychlostí a pamětí.

1.6.2 Programovací jazyky

Programovací software pro PLC Simatic, TIA Portal, nabízí různé způsoby programování. Každý jednotlivý programový blok musí být psán právě jedním jazykem, ale současně se často při sestavování jednoho programu využívá výhod různých jazyků tak, že se jednotlivé bloky kombinují.

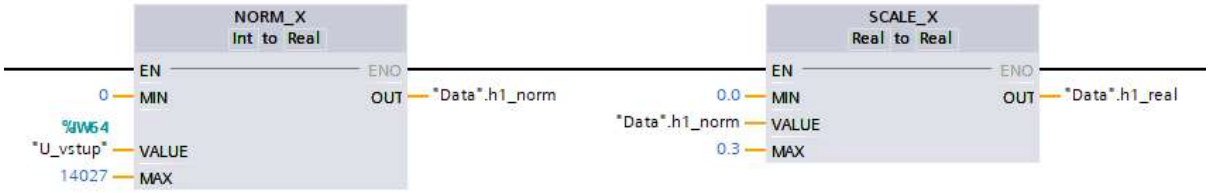
Nejzákladnějším způsobem programování je tzv. instrukční list. Velmi se podobá zápisu kódu v Asembleru. Každá instrukce je zapsána na novém řádku.

Pro hladký přechod z řízení pomocí elektrických zapojení byl vyvinut způsob programování PLC, který se nazývá ladder diagram (LAD), tzv. žebříková logika. Tato metoda intuitivně napodobuje dřívější reléová zapojení. Jedná se o jednoduchý způsob, jak vytvářet logické výrazy v grafické podobě. Je vhodný pro automatizaci opakujících se sekvencí.



Obr. 1.18 - Příklad ladder logiky

Funkční blokový diagram (FBD) se opět řadí mezi grafické programovací jazyky. Jak lze již z názvu odvodit, jeho hlavními prvky jsou funkční bloky. Na základě signálů vstupujících do vstupů funkčního bloku je vykonána vnitřní instrukce a následně jsou nastaveny výstupy.



Obr. 1.19 - Příklad funkčního blokového diagramu

Pro naprogramování složitější řídicí logiky se využívá strukturovaný text (SCL). Je velmi podobný programovacím jazykům jako jsou Pascal nebo C++. Kromě prvků typických pro vyšší jazyky (smyčky, větvení atd.) si zachovává klasické funkcionality typické pro PLC jako jsou vstupy, výstupy, čítače, časovače apod.

```
#vstup[0] := #in1;
#vstup[1] := #in2;
#vstup[2] := #in3;

FOR #pocet_vah := 0 TO 10 - 1 DO
    #skryte_neurony[#pocet_vah] := 0;
END_FOR;
```

Obr. 1.20 - Příklad kódu psaném ve strukturovaném textu

Pro řešení sekvenčních úloh lze využít programovacího jazyka GRAPH. Programátor může využít přehledného grafického zobrazení jednotlivých kroků (instrukcí), které jsou propojeny přechody.

1.6.3 Struktura program

Uživatelský program je strukturovaný do tzv. bloků s vlastními vstupy a výstupy. Organizační blok (OB) zprostředkovává komunikaci mezi kódem uživatelského programu a operačním systémem stanice. Kód programu je dále rozdělen do funkční bloků (FB), funkcí (FC) a datových bloků (DB). Funkční bloky se od funkcí odlišují vlastní trvalou paměťovou oblastí pro správu definovaných proměnných, datovým instančním blokem pevně přiřazeným k funkčnímu bloku. Datové bloky umožňující přístup k uloženým uživatelským datům jakéhokoli funkčního bloku se nazývají globální datové bloky (Berger, 2013).

1.6.4 Vykonávání programu

Znakem všech PLC je cyklické vykonávání uživatelského programu. Zařízení Simatic nejsou výjimkou. Dokončení jednoho běhu programu je dáno vykonáním poslední instrukce.

Výstupní hodnoty jsou přepsány novými příslušnými hodnotami z paměti. Poté operační systém provede vlastní správu, při které aktualizuje vlastní registry a časovače. Prostřednictvím vstupních periférií jsou následně přivedeny vstupní hodnoty. V tomto okamžiku dochází k vykonání celého uživatelského programu od první do poslední instrukce. Tento cyklický proces lze narušit pomocí hardwarového nebo cyklického přerušení (Berger, 2013).

1.6.5 Využití PLC

PLC je robustní spolehlivá výpočetní technika, která se využívá pro řízení kriticky důležitých procesů v automatických provozech. Běžně se vyskytuje ve výrobních halách, elektrárnách, v závodech pro těžbu ropy a plynu a v řídicích místnostech pro vlakovou dopravu. V současné době se stále více PLC využívají i pro domácí automatizaci, kde fungují jako komplexní správci domu nebo bytu. Řídí ekologicko-ekonomickou správu energie, zabezpečení, světelné scény, žaluzie, provoz bazénu a další (Beresford, 2011).

1.7 SIMATIC S7-1200

Simatic je označení pro rodinu programovatelných automatů firmy Siemens. Poslední generace, určená pro průmyslovou generaci, má označení S7. Tato řada je složena z jednotlivých výkonnostních tříd. Modulární systém S7-1200 je určen pro malé a střední automatizační úlohy (Berger, 2013).

Obr. 1.21 - PLC S7-1200, CPU 1212C AC/DC/RLY (6ES72121BE400XB0, 2022)

1.7.1 Moduly

Celá sestava programovatelného logického automatu se skládá z jednotlivých modulů. Minimální funkční sestava je tvořená pouze CPU jednotkou s malým počtem vstupů a výstupů.

Prostřednictvím sběrnicového systému k ní lze připojit rozšiřující signálové vstupní a výstupní moduly. K rozšíření sběrnicového systému se využívají komunikační moduly. Existuje řada dalších připojitelných modulů. Pro programování celého modulárního systému, stanice, se používá průmyslový ethernet (Berger, 2013).

CPU

CPU neboli centrální procesní jednotky jsou rozděleny do čtyř variant: 1211, 1212, 1214 a 1215. Hlavní rozdíly jsou dány velikostí uživatelské paměti, množstvím a typem integrovaných vstupů a výstupů a možností připojení modulů. Každá procesní jednotka je dostupná ve třech verzích, a to: DC/DC/DC, AC/DC/RLY a DC/DC/RLY. První údaj označuje napájení, druhý provozní napětí a třetí typ digitálních výstupů. Konkrétní hodnoty zobrazuje tab. 1.1 (Berger, 2013).

Tab. 1.1: Hodnoty jednotlivých údajů verzí centrálních procesních jednotek

Položka označení	DC	AC	RLY
První	24 V	120/230 V	-
Druhá	24 V	-	-
Třetí	24 V	-	30 VDC / 250 AC

Signálové

Pro správné řízení velkého procesu obvykle nedostačuje množství vstupů a výstupů na základní centrále. Pro řešení tohoto problému existují signálové moduly. Jsou to převodníky signálů. K centrále jich lze připojit až osm. Pro binární komunikaci se snímači a akčními členy se využívají digitální vstupní a výstupní moduly. Vstupy provádějí převod mezi externím signálem (24 VDC, případně 125/230 VAC) na interní signál, výstupy opačně. Pro spojitý přenos signálů jsou používány analogové vstupní a výstupní moduly. Analogové vstupní moduly převádějí spojité veličiny, které mohou být v rozsazích: ±10 V, ±5 V, ±2,5 V, ±1,25 V a 0 až 20 mA nebo 4 až 20 mA na interní číselnou hodnotu -27 648 do +27 648 (Berger, 2013).

Komunikační

Vzájemné sdílení informací s dalšími účastníky řízeného procesu zprostředkovávají komunikační modely. Modul CM 1241 umožňuje rychlý sériový přenos dat prostřednictvím spojení bod-bod s rozhraním RS232 nebo RS422/485. Pro připojení ke sběrnici PROFIBUS se

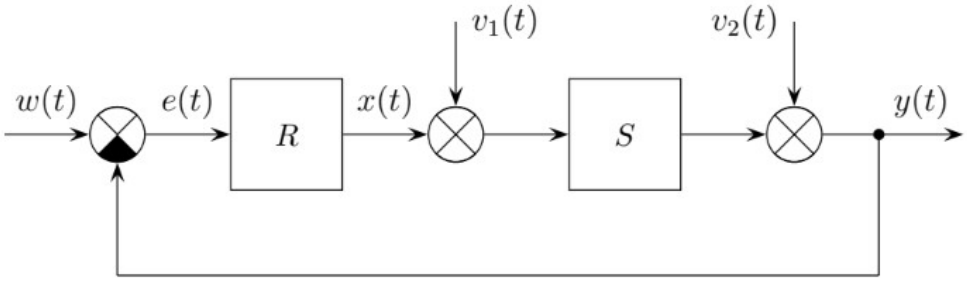
používá modul CM 1243-5. Další typy komunikačních modulů umožňují připojení sběrnice AS-Interface, případně také lze propojit stanici s mobilní sítí GSM/GPRS (Berger, 2013).

1.8 ŘÍZENÍ DYNAMICKÝCH SYSTÉMŮ

Cílem řízení je, aby výstupní veličina dosáhla přesného a rychlého sledování žádané veličiny. Na řízenou soustavu mohou současně působit poruchy, se kterými je nutné počítat. Z toho plynou dvě hlavní funkce regulátorů starajících se o řízení. První je věrné sledování žádané veličiny a druhá je kompenzace poruch.

1.8.1 Zpětnovazební řízení

Hlavní výhodou zpětnovazebního řízení oproti přímému je znalost skutečné hodnoty na výstupu. Pro tento způsob řízení se využívá termínu regulace. Regulátor zapojený do obvodu má možnost zajistit obě hlavní funkce.



Obr. 1.22 - Blokové schéma zpětnovazební regulace (Blaha, 2020)

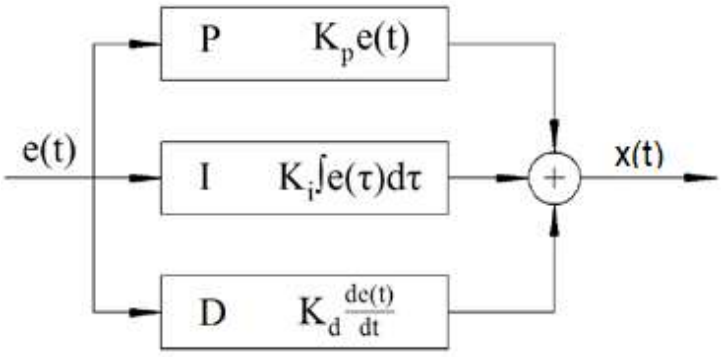
Pro veličiny na obr. 1.22 platí, že

- $w(t)$ je řídicí veličina,
- $y(t)$ je regulovaná veličina,
- $e(t)$ je regulační odchylka,
- $x(t)$ je akční veličina,
- $v_1(t)$ a $v_2(t)$ jsou poruchové veličiny,
- R označuje regulátor,
- S označuje soustavu.

1.8.2 PID regulátor

Globálně nejpoužívanějším regulátorem je typ PID. Využívá se pro řízení dynamických systémů v 85 % aplikací. Pro regulaci využívá složek proporcionální, integrační a derivační,

proto se jeho označení zkracuje na PID. Na obr. 1.23 je strukturální schéma PID regulátoru zobrazeno (Paz, 2001).



Obr. 1.23 - Strukturální schéma PID regulátoru (Bakhanovich, 2017)

Pro akční veličinu, řídící signál, x(t) lze použít i označení u(t). Chování PID regulátoru popisují rovnice

$$u(t) = K_p e(t), \tag{1.29}$$

$$u(t) = K_i \int_0^t e(\tau)d\tau, \tag{1.30}$$

$$u(t) = K_d \frac{de(t)}{dt}, \tag{1.31}$$

kde K_p je zesílení proporcionální složky,

K_i je zesílení integrační složky,

K_d je zesílení derivační složky.

Proporcionální složka

Pokud jsou integrální a derivační složky nulové, tak je řízení v každém časovém okamžiku proporcionálně úměrné regulační odchylce. Čím větší je odchylka, tím větší řídicí signál regulátor vysílá. Pokud je dosaženo cíle, regulátor nic nedělá. Může nastat situace, kdy je regulovaná veličina trvale odchýlena od žádané. S tímto případem si proporcionální složka nedokáže poradit (Paz, 2001).

Integrační složka

Zapojením integračního členu tedy lze eliminovat ustálenou regulační odchylku. Integrační složka reaguje na dobu trvání regulační odchylky. Čím déle odchylka trvá, tím se

řídicí signál zvětšuje. Ze své integrální podstaty člen nepracuje pouze s aktuální hodnotou, ale lze si ho představit jako akumulaci minulých hodnot signálu. Obecně se integrační člen využívá společně s proporcionálním členem. Protože má integrační člen tendenci zpomalovat reakce systému, tak se k němu často přidává i derivační člen (Paz, 2001).

Derivační složka

Princip derivační složky v řízení je založen na rychlosti změny regulační odchylky. Čím rychlejší je reakce regulační odchylky, tím se řídicí signál zvětšuje. V případě, kdy se regulovaná veličina přibližuje k žádané, je derivační složka schopna potlačit kmitání kolem žádané hodnoty. Chování derivačního členu lze tedy popsat tak, že zrychluje regulační děj. Vzhledem k praktické nerealizovatelnosti samotné derivace, se tento člen používá společně s proporcionální nebo i integrační složkou. Při praktickém použití je nutné si dávat pozor na šum, protože hrozí jeho nežádoucí zesílení (Paz, 2001).

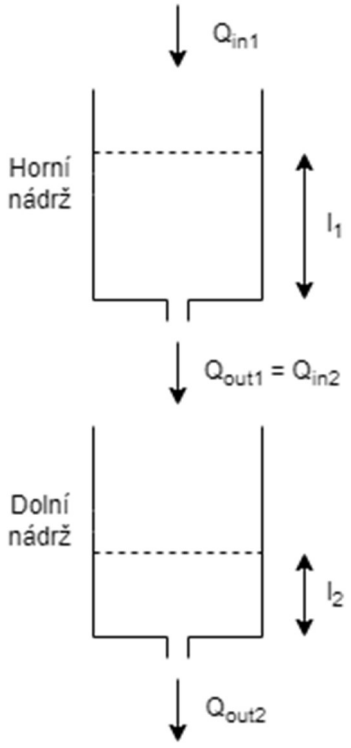
2 PRAKTICKÁ ČÁST

Tato kapitola začíná popisem vybraného řízeného dynamického procesu, soustavy nádrží, včetně jejího simulovaného provedení v prostředí Simulink. To bude použito pro nahrazení reálného systému, protože nebyl k dispozici. Následuje proces návrhu soft senzoru, který je založen na paradigmatu dopředné umělé neuronové sítě. Poté práce pokračuje realizací řízení procesu s využitím PID regulátoru. V poslední části je provedeno komplexní testování a vyhodnocení kvality regulačního pochodu.

2.1 REGULOVANÝ PROCES

Pro řešení řízení byla vybrána soustava dvou nádrží tvaru válce. V grafické podobě si lze tuto soustavu prohlédnout na obr. 2.1. Do horní nádrže vtéká voda o průtoku Q_{in1}. Kruhovým otvorem ve dně voda vytéká přímo do dolní nádrže. Výtok Q_{out1} z horní nádrže se tak logicky rovná přítoku Q_{in2} do dolní nádrže. Výška hladiny v dolní nádrži je značena l_2.
Obě nádrže jsou bez víka, otevřené. Voda proudí z horní nádrže do dolní pouze prostřednictvím otvoru ve dně. Dojde-li k přetečení jakékoli z nádrží, je přebytečná voda odváděna odtokem.

Přítok Q_{in1} a výška hladiny v horní nádrži l_1 jsou známé veličiny. Cílem práce je navrhnout řešení zpětnovazebního řízení výšky hladiny v dolní nádrži pomocí změny přítoku Q_{in1}.



Obr. 2.1 - Soustava dvou nádrží

2.1.1 Model v Simulinku

Pro již popsanou soustavu dvou nádrží byl použit simulační a modelovací software, Simulink. Ve formě blokových schémat byl vytvořen modifikovatelný model, který umožnil vygenerovat dostatečně velkou trénovací a validační množinu dat pro vytvoření soft senzoru založeném na paradigmatu dopředné umělé neuronové sítě.

Nejprve bylo nutné namodelovat chování jedné nádrže. Diferenciální rovnice, které popisují tento děj, jsou:

$$Q_{in1} = Q_{out1} + S \cdot \frac{dl_1}{dt} \tag{2.1}$$

$$Q_{out1} = f_1 \cdot \sqrt{2gl_1} \tag{2.2}$$

kde Q_{in1} – přítok, m^3/s,

Q_{out1} – odtok, m^3/s,

S – obsah podstavy nádrže, m^2,

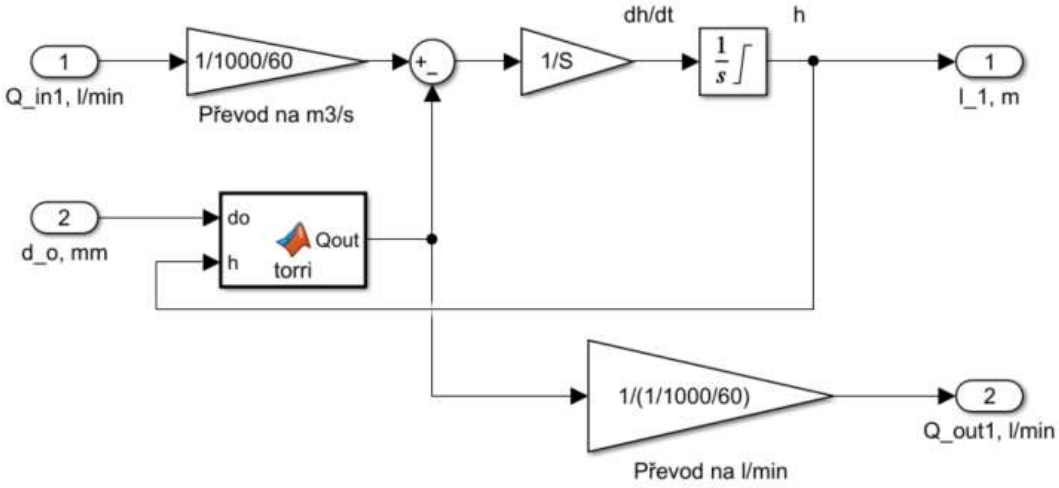
l_1 – výška hladiny, m,

f_1 – obsah podstavy výtokového otvoru, m^2,

g – tíhové zrychlení, m/s.

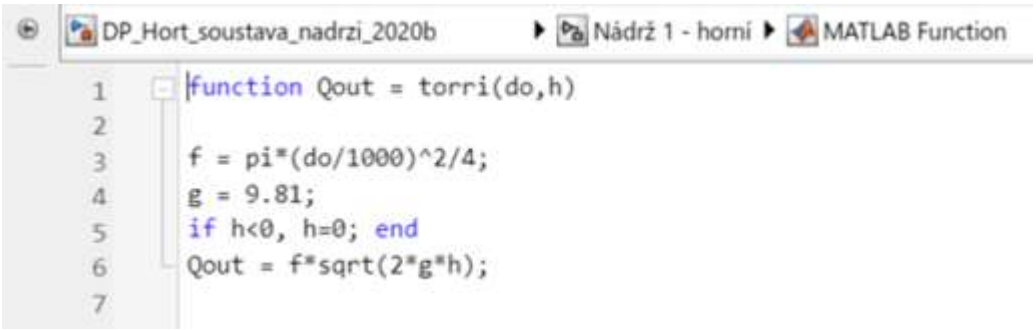
Na základě těchto rovnic bylo vytvořeno simulinkové schéma. Počáteční výška hladiny v nádrži je 0 m. Přítok Q_{in1} je zadáván v l/min a je převáděn na m^3/s. Pro simulaci byly definovány tento tvar a rozměry nádrže:

- Tvar válce,
- výška nádrže … $l = 0{,}3$ m,
- průměr nádrže … $d_1 = 0{,}15$ m,
- obsah podstavy válce … $S = \frac{\pi d_1^2}{4}$,
- průměr výtokového otvoru … $d_o = 4$ mm.



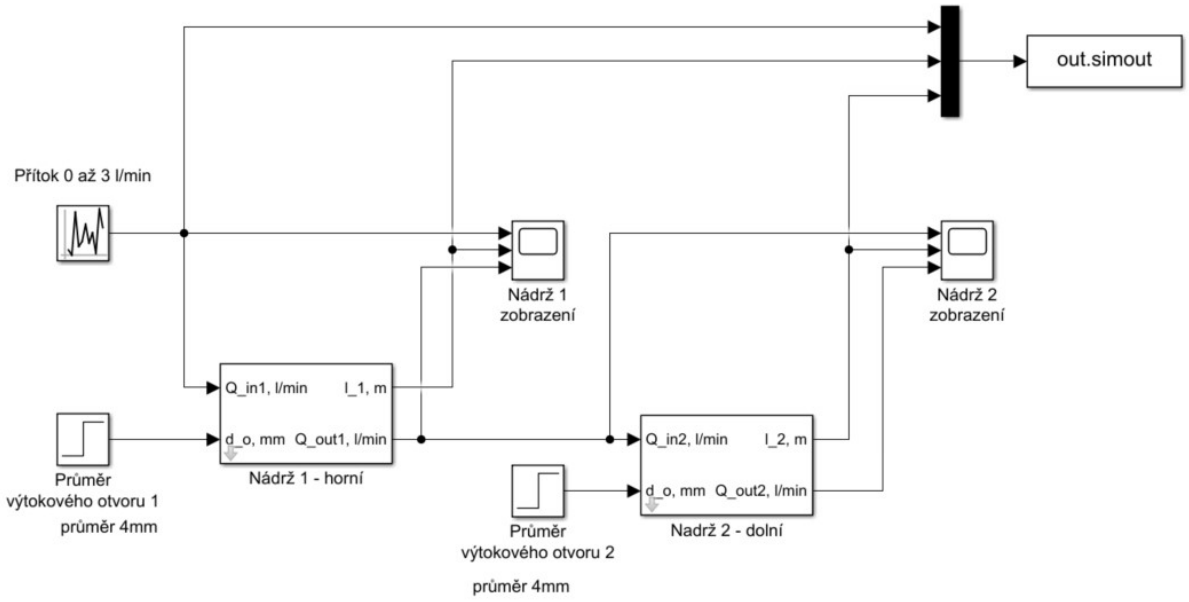
Obr. 2.2 - Model nádrže

Funkční blok pojmenovaný „torri“ zajišťuje výpočet podle (2.2).



Obr. 2.3 – Matlab - funkční blok torri

Po namodelování jedné nádrže bylo možné pokračovat s řešením celé soustavy. Přítok do soustavy je omezen na 0 až 3 l/min.



Obr. 2.4 - Model soustavy nádrží

Uložení dat pro tvorbu UNS

Pomocí bloku pojmenovaném out.simout jsou ukládána data do workspace Matlabu, kde jsou dále zpracovávána do vhodné formy. Pro trénování umělé neuronové sítě bylo potřeba uložit hodnoty přítoků do soustavy a výšky hladin v horní nádrži. Tyto veličiny jsou v řešené úloze měřené. Do množiny trénovacích dat byly přidány také simulované hodnoty výšek hladin v dolní nádrži, aby bylo možné umělou neuronovou sítí tuto veličinu estimovat.

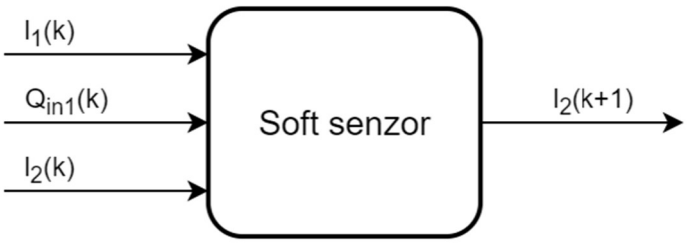
2.2 NÁVRH SOFT SENZORU

Pro řešení úloh, u kterých jsou známé nejen vstupní, ale i výstupní veličiny, je výhodné využití dopředných neuronových sítí. Vytvořený simulační model umožňuje všechna potřebná data vygenerovat v dostatečném množství. Pro danou úlohu bylo proto rozhodnuto o realizaci soft senzoru prostřednictvím dopředné umělé neuronové sítě.

Pro řízení výšky hladiny l_2 v dolní nádrži je nutné znát její aktuální hodnotu. Jelikož není možné hodnoty l_2 měřit, je navržen soft senzor, který činnost hardwarového snímače nahrazuje.

Vstupní data soft senzoru jsou tvořena hodnotami dvou měřených veličin a hodnotami vlastní výstupní proměnné, která je vypočítána v přechozím cyklu. První známou veličinou je proměnná popisující vstup do soustavy, a to přítok do horní nádrže Q_{in1}. Druhou známou veličinou je proměnná reprezentující stav v horní nádrži, výška hladiny l_1. Poslední veličinou

je již zmiňovaná estimovaná výška hladiny v dolní nádrži l_2, která je průběžně získávána právě pomocí tohoto soft senzoru.



Obr. 2.5 – Blokové schéma soft senzoru se zobrazením vstupů a výstupů

2.2.1 Software pro vytvoření UNS

Pro vytvoření umělé neuronové sítě byl ve vývojovém prostředí PyCharm použit vysokoúrovňový programovací jazyk Python. Zde bylo možné zapojit Keras API, které je postavené na softwareové knihovně s kódem pro strojové učení a UI TensorFlow. Tato nadstavba byla vytvořena tak, aby uživateli umožnila jednoduché, rychlé a výkonné experimentování s neuronovými sítěmi.

2.2.2 Tvorba DVUNS

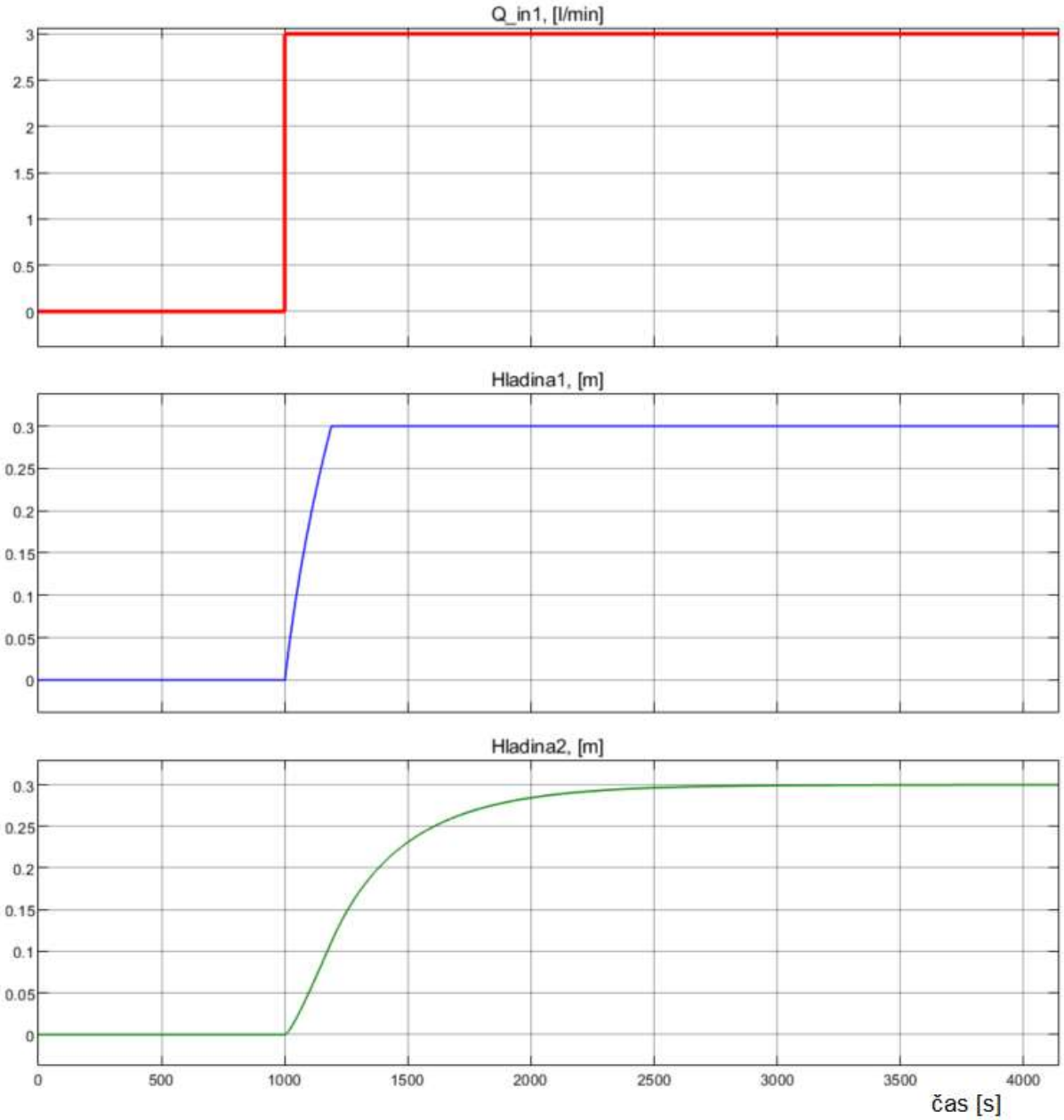
Platforma Keras API pro hluboké učení nabízí připravenou strukturu modelu Sequential. Jedná se o jednoduchý zásobník vrstev. Jednotlivé vrstvy neuronové sítě lze přidávat pomocí příkazu *add*. Dále jsou použity vrstvy typu Dense. Těm jsou nastavovány parametry, které definují, jak bude neuronová síť sestavována. Mezi důležité parametry se řadí počet neuronů v dané vrstvě, počet vstupů do každého neuronu ve vrstvě a typ aktivační funkce. Po vložení uvažovaného počtu vrstev, včetně jejich parametrů, lze přistoupit k nastavení konfigurace procesu učení sestavovaného modelu. To je provedeno pomocí příkazu *compile* minimálně včetně definice algoritmu optimalizační metody a určením ztrátové funkce. Poté už zbývá pouze neuronovou síť natrénovat prostřednictvím příkazu *fit*. Neuronové síti je samozřejmě nezbytné pro proces učení předložit trénovací data. Dále je také definován počet epoch učení, velikost dávky a oddělení části dat pro ověřování. Nakonec je model nutné pojmenovat a uložit pomocí příkazu *save*. Příklad použití nástrojů pro vytvoření neuronové sítě je na obr. 2.6.

```python
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation
import scipy.io

vstupy = scipy.io.loadmat('vstupy.mat')
cile = scipy.io.loadmat('cile.mat')
vstupy_numpy = vstupy['vstupy']
cile_numpy = cile['cile']

model = Sequential()
model.add(Dense(10, input_dim=6, activation='tanh'))
model.add(Dense(1, activation='linear'))

model.compile(optimizer='Adam', loss='mse')
model.summary()

epoch = 1000
batch = 32
val = 0.2
hist = model.fit(vstupy_numpy, cile_numpy, epochs=epoch, batch_size=batch, validation_split=val)
model.save('model.h5')
```

Obr. 2.6 - Příklad kódu pro natrénování UNS

2.2.3 Výběr dat pro trénování

Pro vytvoření funkční umělé neuronové sítě je velmi důležité vhodně určit data, která budou použita pro trénování. Nejprve je nutné určit četnost změn přítoku do první nádrže, což ovlivní celý průběh procesu. Velikost datasetu lze ovlivnit délkou děje a také periodou vzorkování. Jelikož byl pro vytvoření trénovací množiny použit Simulink, bylo možné výše zmíněné parametry trénování měnit a ověřovat velice rychle.



Obr. 2.7 - Přechodová charakteristika soustavy nádrží

Na obr. 2.7 je vidět přechodová charakteristika soustavy nádrží. Přítoku je nastavena skoková změna z minimální hodnoty 0 l/min do maximální hodnoty 3 l/min v čase 1000 s. Hladina v horní nádrži dosáhne maximální hladiny v čase 1189 s, tedy po 189 s od skokové změny přítoku. Hladina v dolní nádrži je na svém maximu v času 9000 s, nicméně hladiny 0,299m (odpovídá 99,66% maximální) dosáhne již po 3076s.

Na základě informací zjištěných pomocí přechodové charakteristiky byly nastaveny výchozí simulační parametry pro tvorbu trénovací množiny. Pomocí různých kombinací parametrů bylo vytvořeno více než dvacet různých datasetů. K úkonu nastavení ideálních

simulačních parametrů bylo nezbytné se během testování již vytvořených neuronových sítí vracet a dělat jejich dodatečné úpravy. Po následném testování kombinací parametrů bylo vyhodnoceno jako nejzajímavější, z hlediska poměru přesnosti a výpočetní náročnosti, nastavení těchto parametrů:

- Délka simulace: 600 000 s,
- čas konstantního přítoku: 800 s,
- perioda vzorkování dat: 10 s.

Délka simulace a perioda vzorkování ovlivňují množství vstupních dat pro trénování umělé neuronové sítě. Perioda vzorkování byla navolena na 10 s. Vzhledem k rychlosti celé soustavy je tato hodnota dostatečná. Z toho plyne, že pro trénování bylo použito 60 000 hodnot pro každý vstup.

Jeden vektor vstupních hodnot je dán řádem modelu ve tvaru ARX rovnice. Řád označuje počet po sobě jdoucích předchozích hodnot každého ze vstupů. Například pro druhý řád je rovnice složena ze dvou po sobě jdoucích hodnot přítoků, hladin v horní nádrži a hladin v dolní nádrži. Matice vstupů do UNS má tedy rozměry 6 krát 60 000.

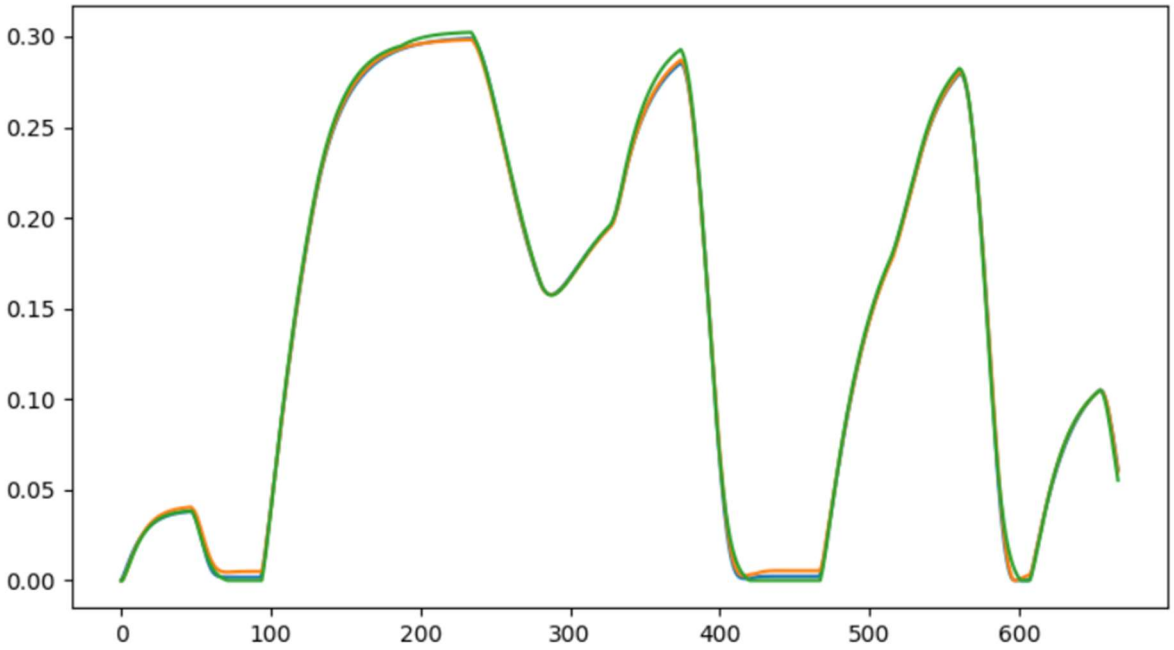
2.2.4 Testování konfigurací

Po nastavení neměnných parametrů simulace bylo možné přistoupit k trénování UNS pro první, druhý a třetí řád ARX modelu. Pro všechny řády byla zkoušena modifikace počtu neuronů ve skryté vrstvě. Počet neuronů byl postupně zvyšován až do počtu 14. Pro porovnání byly vybrány modifikace s 6, 10 a 14 neurony. Po natrénování všech UNS jim byla předložena sada testovacích dat. Jejich přesnost si lze prohlédnout na obr. 2.8. Pojmenování 14_3s označuje konfiguraci s nastavením 14 neuronů ve skryté vrstvě a použití ARX modelu třetího řádu. Tento označovací klíč platí analogicky pro všechny ostatní konfigurace.

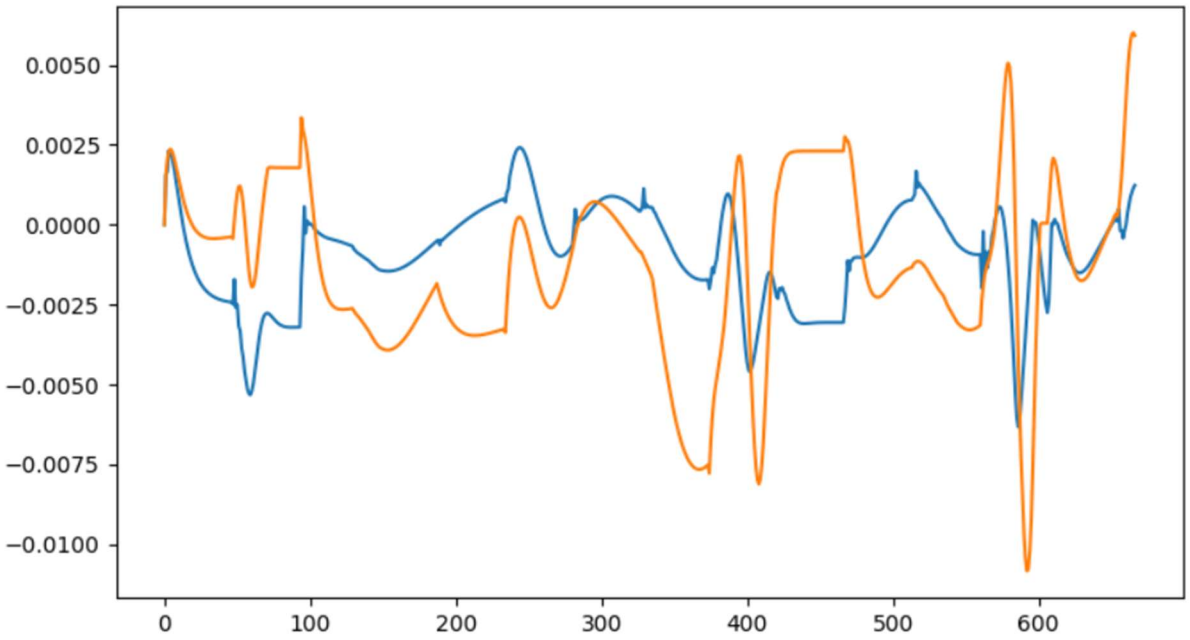
```
Chyba pro konfiguraci 14_3s:  1.9480691684635696
Chyba pro konfiguraci 10_3s:  0.8946191731228256
Chyba pro konfiguraci 6_3s:  5.403059593687883
Chyba pro konfiguraci 14_2s:  2.064665782147661
Chyba pro konfiguraci 10_2s:  2.6388469042972647
Chyba pro konfiguraci 6_2s:  2.6007157797038305
Chyba pro konfiguraci 14_1s:  2.918894122489193
Chyba pro konfiguraci 10_1s:  1.5590323883625645
Chyba pro konfiguraci 6_1s:  1.9812552227777194
```

Obr 2.8 - Porovnání přesnosti jednotlivých UNS

Dvě nejlepší konfigurace jsou 10_3s (modrá barva) a 10_1s (oranžová barva). V obou případech je skrytá vrstva tvořena deseti neurony. Po předložení testovací množiny vstupů jsou výstupy obou konfigurací srovnány s očekávaným výstupem (zelená barva) na obr. 2.9 Velikost chyb obou konfigurací je znázorněna na obr. 2.10.



Obr. 2.9 - Očekávaná reakce UNS a reakce UNS 10_3s a UNS 10_1s



Obr. 2.10 - Srovnání nepřesností natrénovaných UNS 10_3s a 10_1s

I přestože se ukázala jako méně chybová konfigurace obsahující 9 vstupních hodnot (modrá barva), byla v práci dále využita konfigurace se 3 vstupními hodnotami (oranžová barva). Velikost chyby je vzhledem k jednodušší konfiguraci přijatelná.

2.3 IMPLEMENTACE SOFT SENZORU DO PLC

Pro úspěšné řízení dynamické soustavy bylo stěžejní správně implementovat vytvořenou umělou neuronovou síť do PLC Simatic S7-1200 a využít ji jako soft senzor.

2.3.1 Převod modelu UNS do jazyka SCL

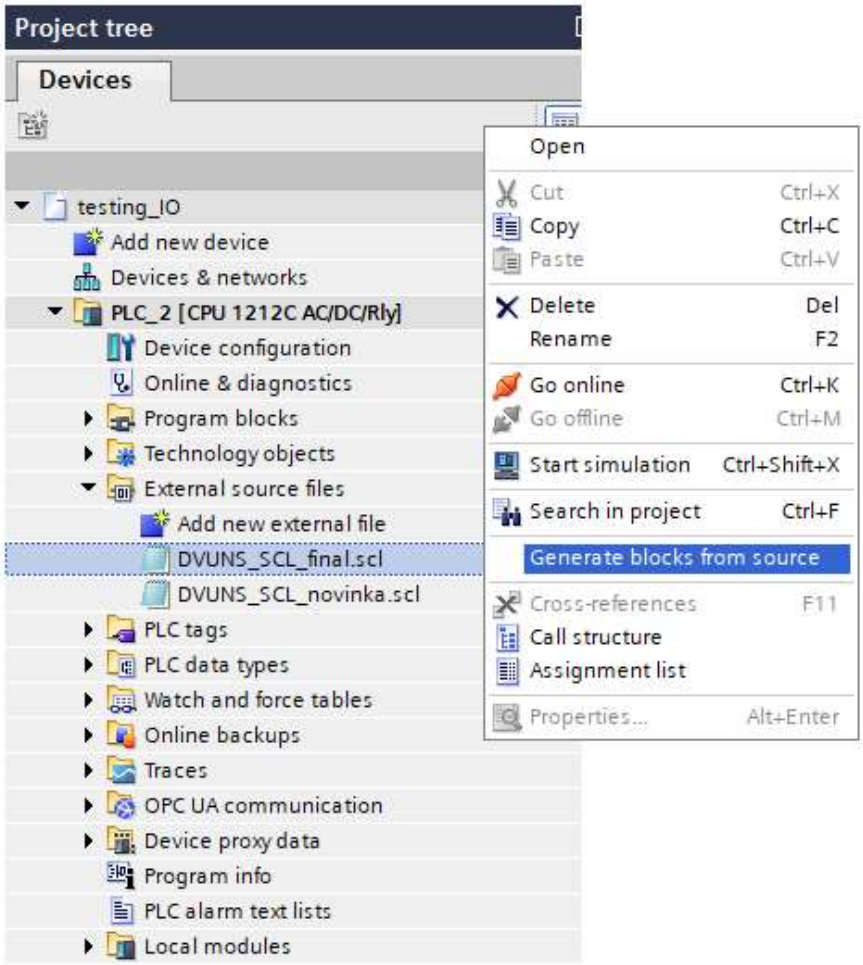
Všechny důležité informace o vybrané umělé neuronové síti byly uloženy ve formátu souboru s příponou .h5. Tyto informace popisující UNS je nutné převést do jazyka SCL, který je vlastní pro PLC. Pro tento účel byl vytvořen kód v jazyce Python, který pro danou neuronovou síť poloautomaticky sestavuje kód do souboru s příponou .scl. Celý kód je připojen v příloze A.

Parametry natrénované umělé neuronové sítě, které jsou potřeba převést:

- Počet vstupů,
- počet výstupů,
- počet vrstev,
- počet neuronů ve skrytých vrstvách,
- váhy,
- prahy neuronů.

2.3.2 Projekt v TIA Portal

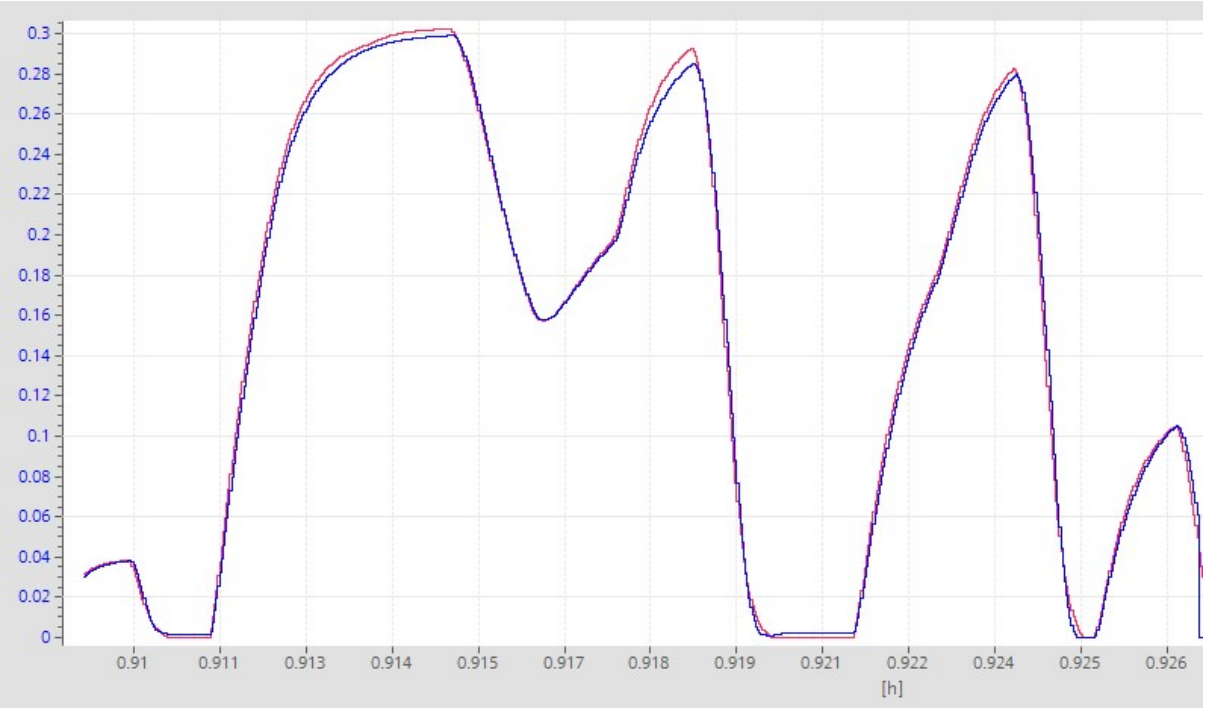
Pro vytvoření SCL kódu bylo využito programovací prostředí pro PLC Simatic, TIA Portal v17. Nejprve byl založen nový prázdný projekt. Následně bylo připojeno a nakonfigurováno fyzické PLC zařízení. Vytvořený SCL kód byl vložen přes příkaz „Add new external file“ do složky „External source files“. Po přidání souboru byl příkazem „Generate blocks from source“ vytvořen funkční blok včetně lokálních proměnných. Tento postup je naznačen na obr. 2.11.



Obr. 2.11 - Vložení externího souboru SCL a vytvoření funkčního bloku

2.3.3 Kontrola správnosti

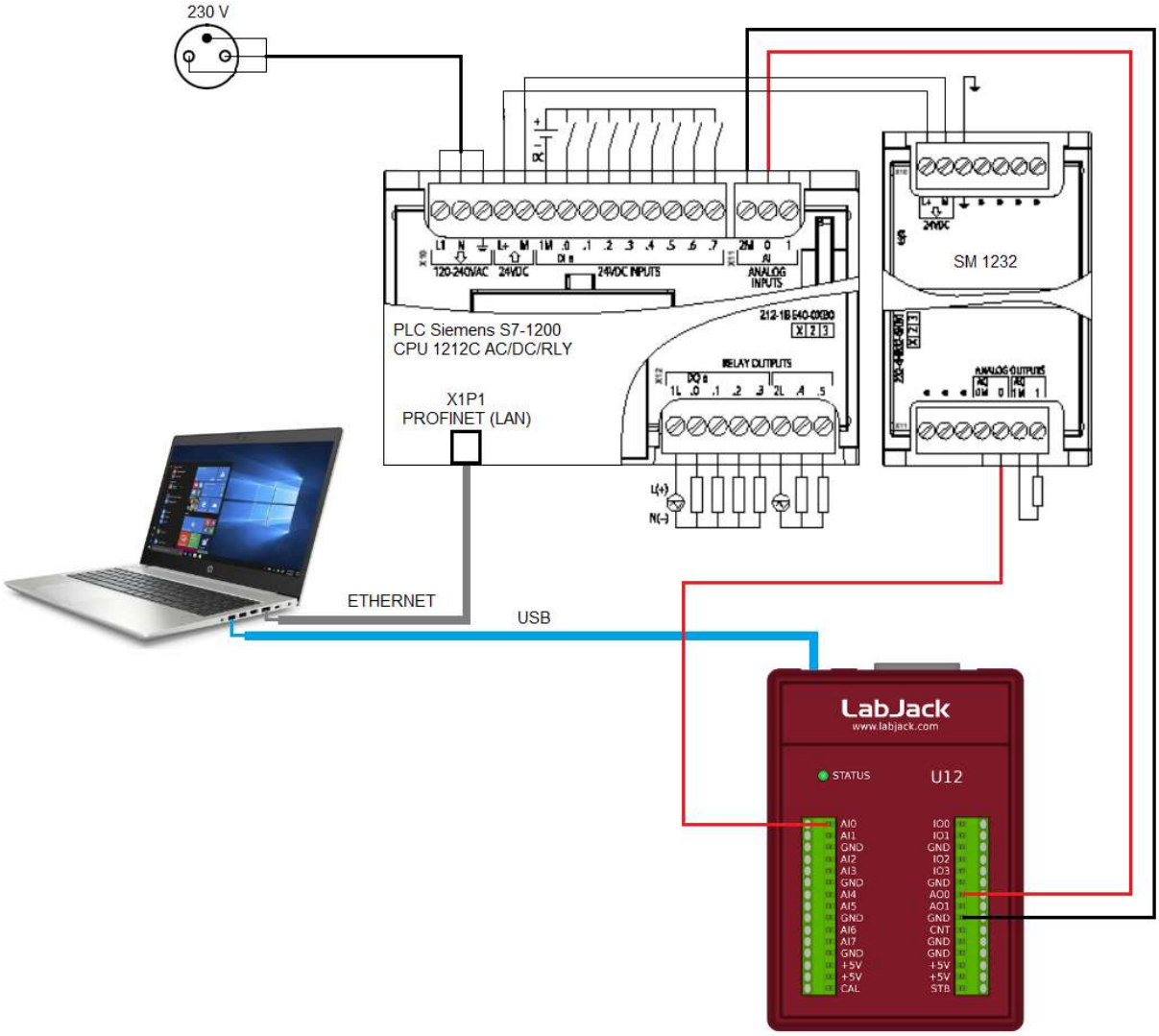
Implementace soft senzoru do PLC byla poměrně obtížná. Vyžadovala přesnou znalost a pochopení matematického procesu, který probíhá v neuronové síti při výpočtu výsledků po předložení vstupů. Pro kontrolu bezchybného přenesení soft senzoru ve formě UNS do PLC byla provedena kontrolní simulace. V PLC byl neuronové síti předložen testovací vzorek dat a očekáváné výstupy (červená barva) byly následně porovnány s výstupy neuronové sítě (modrá barva). Výsledek testu je možné si prohlédnout na obr. 2.12. Kontrolní test dopadl shodně jako při porovnávání v PyCharmu. Implementace soft senzoru do PLC proběhla úspěšně.



Obr. 2.12 - Očekávaný a reálný výstup soft senzoru

2.4 PROSTŘEDKY PRO ŘÍZENÍ PROCESU

Aparát pro realizaci řízení hladiny v dolní nádrži pomocí změny přítoku do horní nádrže vznikl propojením Simulinku na notebooku HP ProBook 450 G7, měřicí karty Labjack U12 a PLC S7-1200. Zapojení všech komponent ilustruje obr. 2.13.



Obr. 2.13 - Zapojení prostředků pro řízení procesu

2.4.1 Labjack U12

Zprostředkovatelem pro přenos informací mezi PLC a softwarově tvořenou soustavou nádrží je již zmíněná měřicí karta Labjack U12 (Obr 2.14). Jedná se o USB zařízení tvořené analogovými a digitálními I/O pro sběr a záznam dat. Získané údaje jsou dále využívány měřicími a řídicími aplikacemi. Měřicí karta Labjack U12 disponuje osmi analogovými vstupy s rozsahem ±10 V a dvěma analogovými výstupy s rozsahem 0 ÷ 5 V.

Obr. 2.14 - Měřicí karta Labjack U12

2.4.2 Simulink

Soustava nádrží je vytvořena softwarově pomocí simulace v Simulinku. Pro tento účel byl využit již hotový model, který vygeneroval trénovací data pro soft senzor. Úpravy modelu se týkaly doplnění obsluhy měřicí karty. Byl přidán blok obsahující pokyny pro čtení analogového vstupu a blok obsahující pokyny pro zápis na analogový výstup měřicí karty.

2.4.3 PLC S7-1200

Stanice je složená z hlavní jednotky Simatic S7-1200 1212C AC/DC/Relay a přídavného modulu SM 1232. Pro řešení dané úlohy byl použit jeden analogový vstup a jeden analogový výstup. Vstup se nachází přímo na hlavní jednotce, ovšem výstup nikoli. Z tohoto důvodu byl připojen rozšiřovací modul obsahující analogové výstupy. Programování probíhalo prostřednictvím programu TIA Portal. Podoba vytvořeného funkčního bloku je na obr. 2.15. Do bloku vstupují aktuální hodnoty přítoku do první nádrže, měřená výška hladiny v horní nádrži a estimovaná výška hladiny v dolní nádrži, která byla získána soft senzorem v minulém výpočetním cyklu PLC zařízení.

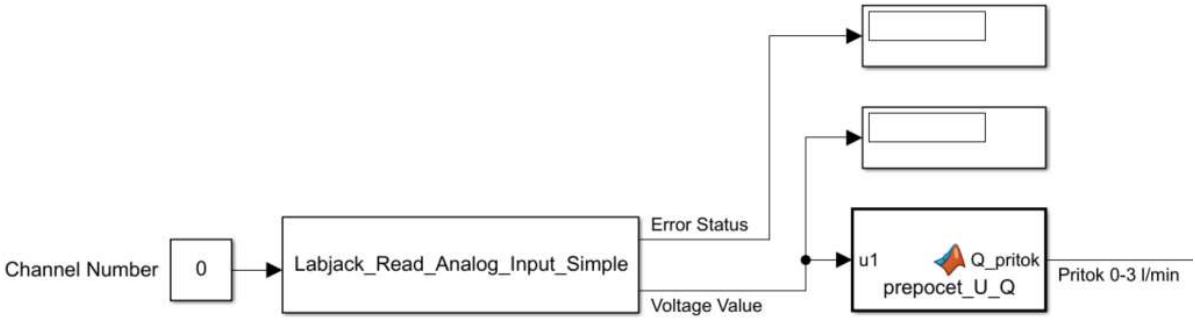
Obr. 2.15 - Funkční blok soft senzoru

2.5 ŘEŠENÍ ŘÍZENÍ

Před samotným vytvářením komplexního řešení řízení byla z důvodu eliminace poškození PLC a měřicí karty provedena kontrola maximálních hladin vstupních a výstupních napětí pomocí digitálního multimetru.

Regulace hladiny v dolní nádrži probíhala na základě zadané žádané hodnoty. Pomocí regulátoru byl určován přítok do soustavy. Hodnota přítoku společně s aktuální měřenou hodnotou hladiny v horní nádrži a estimovanou hladinou v dolní nádrži vstupuje do soft senzoru, který počítá hladinu v dolní nádrži v dalším časovém intervalu. Měřená hodnota v horní nádrži přichází ze simulace v Simulinku prostřednictvím měřicí karty. Informace o hodnotě regulovaného přítoku je odesílána z PLC do simulace.

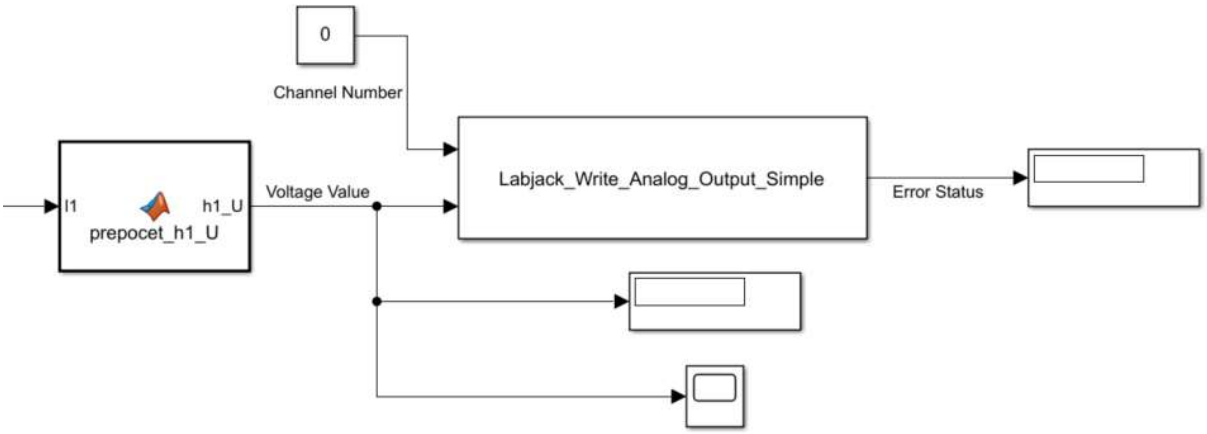
Simulinkový model soustavy nádrží byl doplněn o obsluhu měřicí karty pomocí Matlab toolboxu Labjack_U12_SimulinkImplementation. Na obr. 2.16 je zobrazen blok pro převod signálu do Simulinku, který přichází z PLC na vstup měřicí karty. Vstupní napětí je přepočítáno na hodnotu přítoku, se kterou se v simulaci počítá.



Obr. 2.16 - Čtení analogového vstupu měřicí karty Labjack U12

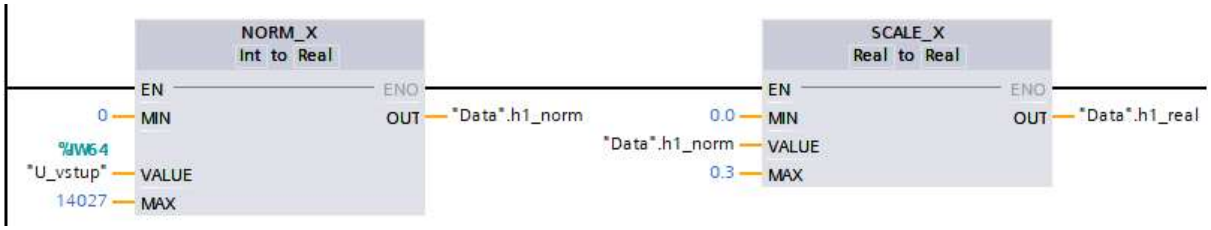
Při zápisu hodnot na analogový výstup měřicí karty je opět nutné provést přepočet. V tomto případě je směr převodu opačný. Hodnota proměnné veličiny výšky hladiny v horní

nádrži, se kterou se pracuje v simulaci, je převedena na hodnotu napětí. Tato napěťová hodnota je zapsána na výstup měřicí karty a dále pokračuje až na vstup PLC.



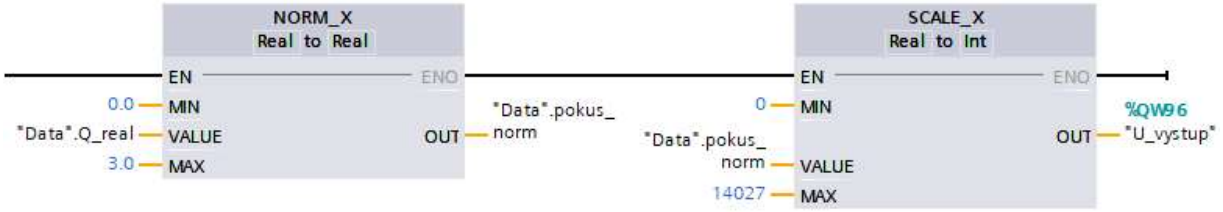
Obr. 2.17 - Zápis na analogový výstup měřicí karty Labjack U12

Jeden vstup a jeden výstup PLC byl přes měřicí kartu Labjack U12 připojen k soustavě. Do PLC vstupovala informace s aktuální hodnotou v horní nádrži. Výstupem z PLC byla hodnota přítoku, která je potřebná pro dosažení žádané hladiny v dolní nádrži. Tato hodnota přítoku je vypočítávána pomocí regulátoru v PLC. Podobně jako v Simulinku bylo i v TIA Portalu potřeba správně přepočítat signál přicházející na vstup a signál vysílaný na výstup. Při vysílání i přijímání signálů byly informace přenášeny v rozsahu 0 ÷ 5 V. Tento napěťový rozsah hodnot odpovídá rozsahu 0 ÷ 14027 v TIA Portalu.



Obr. 2.18 - Zápis na analogový výstup měřicí karty Labjack U12

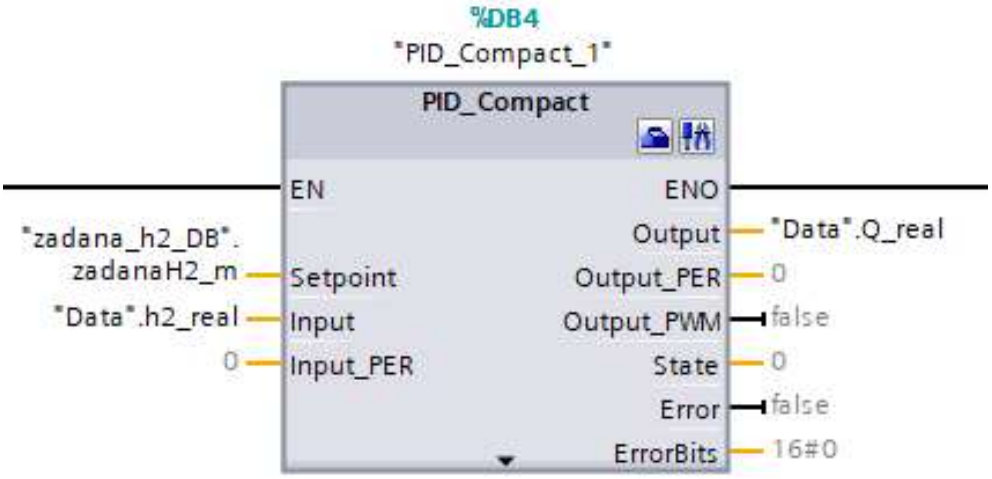
2.5.1 Návrh regulace



Obr. 2.19 - Zápis na analogový výstup měřicí karty Labjack U12

Pro PID regulaci byl využit již předdefinovaný blok „PID compact“, který je obsažen v jedné z knihoven TIA Portalu. Z palety vstupů a výstupů, které blok implicitně obsahuje, byly použity následující:

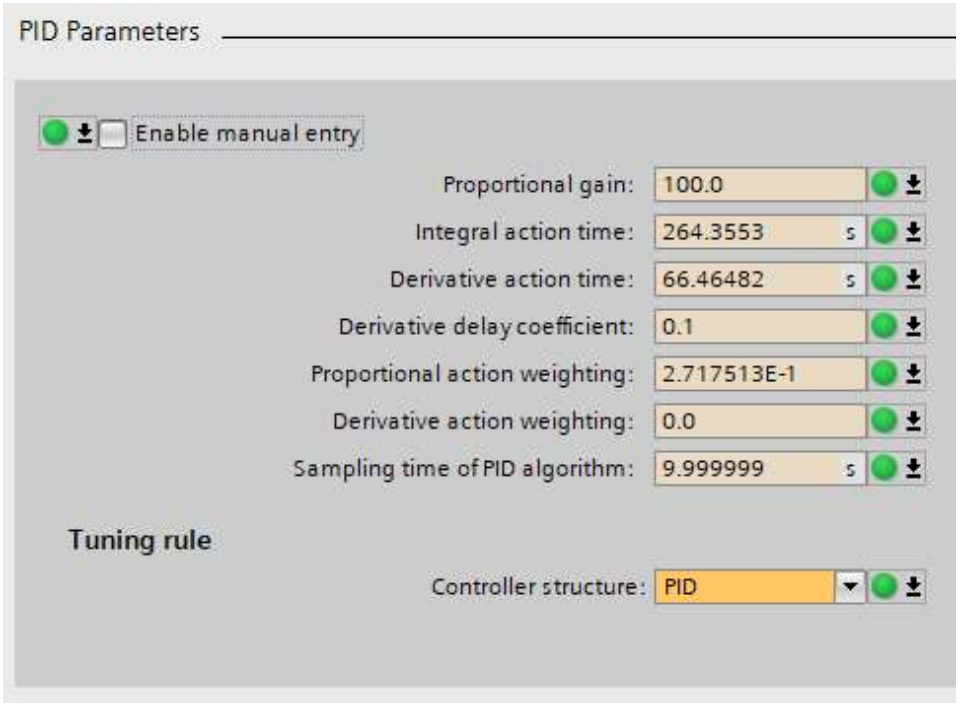
- Vstup „Setpoint“ - hodnota žádané výšky hladiny v dolní nádrži,
- vstup „Input“ - aktuální hodnota výšky hladiny v dolní nádrži,
- výstup „Output“ - hodnota regulovaného přítoku.



Obr. 2.20 - Blok PID compact v TIA Portalu

Regulátoru byly ručně nastaveny základní vlastnosti a limity. Pomocí stisknutí levého tlačítka myši na blok „PID compact“ je zobrazeno okno „Configuration window“. To se skládá ze tří hlavních záložek. V „Basic settings“ byl nastaven hladinový typ regulátoru. V „Process value settings“ byly nastaveny limity procesní veličiny, výšky hladiny, na 0 a 0,3 (minimální a maximální výška hladiny v nádrži). A nakonec v záložce „Advanced settings“ byly definovány limity výstupní veličiny, přítoku, na 0 a 3 (minimální a maximální hodnota přítoku).

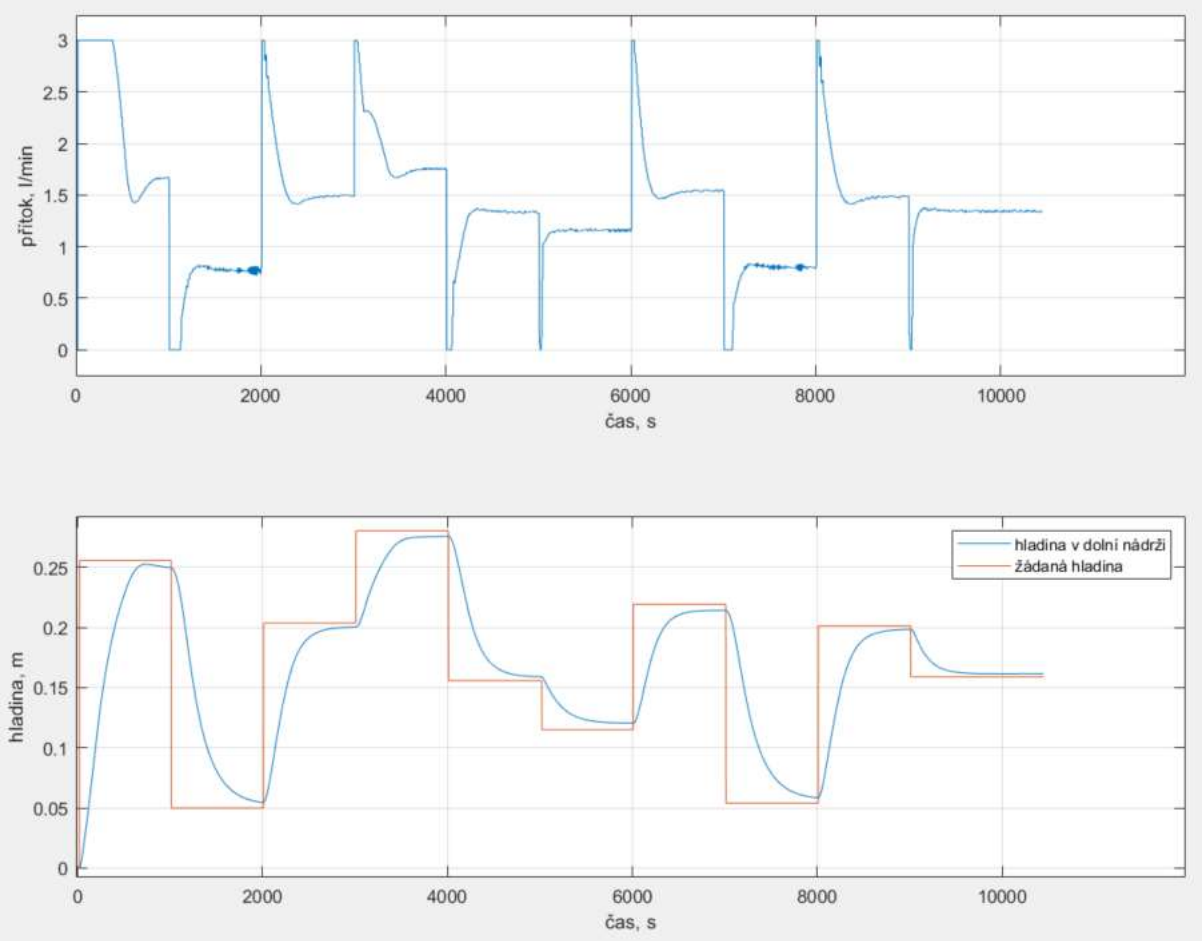
Pro nalezení regulačních parametrů byl použit integrovaný proces „Auto tunning“, tj. automatický proces hledání vhodných parametrů regulátoru pro řízení dané soustavy. Nastavováním různých hladin vstupního přítoku proces zaznamenává reakce soustavy a na základě těchto dat jsou vygenerovány parametry regulátoru. Hodnoty vypočítané pro danou úlohu soustavy nádrží jsou zobrazeny na obr. 2.21.



Obr. 2.21 - Regulační parametry PID regulátoru

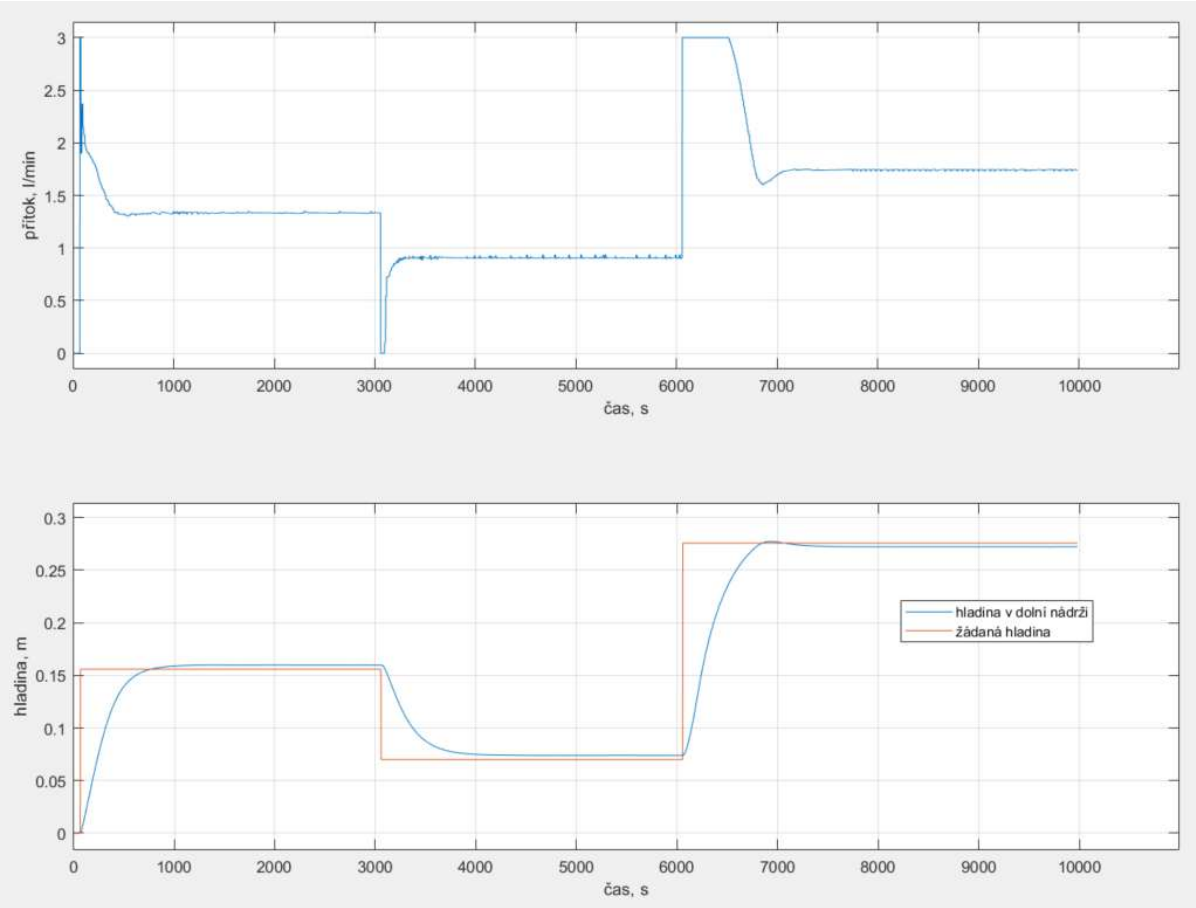
3 TESTOVÁNÍ A VYHODNOCENÍ KVALITY REGULACE

Získané regulační parametry byly nastaveny PID regulátoru, který zajišťuje řízení hladiny v dolní nádrži. Komplexní testování regulace bylo provedeno pro sadu žádaných hladin v dolní nádrži. Na obr. 3.1 je zobrazena regulace pro deset různých žádaných hodnot. Změna hodnoty nastává každých 1000 s.



Obr. 3.1 – Regulace hladiny v dolní nádrži 1

Na obr. 3.2 je vidět obdobná regulace pro tři různé žádané hodnoty se změnou po 3000 s. Tato časová hodnota byla volena z důvodu dosažení maximální hladiny v nádrži při skokové změně přítoku z minima do maxima právě v tomto čase. Tento experiment je popsán v kapitole 2.2.3.



Obr. 3.2 – Regulace hladiny v dolní nádrži 2

Na obr. 3.2 stojí za povšimnutí málo plynulý přítok se skokovými změnami v počáteční fázi regulace na první žádanou hodnotu. Při druhé změně žádané hodnoty je v počáteční fázi hodnota přítoku plynulejší. Udržování hladiny v žádané výšce je zajišťováno pomocí mírně kmitající hodnoty přítoku. Stejného jevu si lze všimnout i pro dvě regulace, při kterých je hodnota ustáleného přítoku nižší než 1 l/min na obr. 3.1. Tento jev je pravděpodobně způsoben nedokonalým nastavením parametrů PID regulátoru a malou frekvencí změn reakcí regulátoru volenou na 10 s.

I přes zmíněné nedostatky dosahuje navržená regulace přijatelné vlastnosti. Ustálené odchylky regulovaných veličin od žádaných hodnot jsou způsobené nepřesným výpočtem výšky hladiny v dolní nádrži pomocí umělé neuronové sítě. Průměrná odchylka je 37 mm, což odpovídá 1,23 % maximálního rozsahu. Pro odstranění tohoto nedostatku by bylo nutné zvolit složitější strukturu UNS.

ZÁVĚR

Výsledkem diplomové práce je realizace soft senzoru v PLC pomocí umělé neuronové sítě. Pro demonstraci řízení byla vybrána soustava dvou nádrží, která byla vytvořena pomocí real-time simulace v Simulinku. Pro bezproblémový převod komunikace mezi soustavou a PLC byla využita akviziční karta LabJack U12.

Vytvoření funkčního soft senzoru se skládalo z několika na sebe navazujících kroků. Nejprve bylo nutné sesbírat data ze soustavy pro trénování a validování. Protože bylo využito simulace, byla potřebná data rychle a jednoduše získána. Menší komplikace nastaly až při výběru trénovacích dat. Bylo nutné nalézt kompromis mezi dostatečnou přesností umělé neuronové sítě a časovou náročností při její tvorbě. Rychlost regulované soustavy je hlavním kritériem při volbě četnosti regulačních zásahů. Při správné volbě periody cyklického vykonávání PLC programu je dosaženo dostatečné přesnosti UNS. Tato frekvence se musí rovnat vzorkovací frekvenci při sběru trénovacích dat. Časová náročnost tvorby UNS je dána množstvím a typem trénovacích dat. Jako optimální řešení popsaného problému byla vyhodnocena perioda regulačního zásahu 10 s, pro kterou bylo použito 3 krát 60 000 vzorků dat pro trénování. Vytvoření umělé neuronové sítě pro toto množství dat trvalo na použitém výpočetním zařízení více než 4 hodiny.

Po vytvoření souboru s modelem UNS v prostředí Python bylo nezbytné převést parametry sítě do jazyka SCL. Tento proces byl časově poměrně náročný, především z důvodu snahy o vytvoření plně automatického převodníku. Nakonec bylo využito poloautomatického převodníku s nutností nastavení několika parametrů a malou ruční korekcí výsledného souboru.

V poslední fázi práce byla navržena PID regulace výšky hladiny dolní nádrže. Řízení funguje správně vzhledem k očekávané zanedbatelné trvalé odchylce, která byla detekována již při návrhu soft senzoru.

Předmětem případné navazující práce by mohlo být použití realizovaného soft senzoru v PLC pro řízení reálných soustav nádrží. Kromě toho lze obecně využít postup mého vypracování jako přibližný návod pro tvorbu nových soft senzorů i v dalších oblastech.

Využívání umělých neuronových sítí má v současnosti vzestupnou tendenci. S výhodami jsou aplikovány tam, kde nejsou dosud využívané řídicí algoritmy optimální. Mimo jiné se může jednat o regulační aplikace v oblastech plánování, průmyslu a energetiky, kde jsou hlavními řídicími prvky programovatelné logické počítače.

POUŽITÁ LITERATURA

6ES72121BE400XB0. Siemens. [online]. ©2022 [cit. 2022-06-20]. Dostupné z: https://mall.industry.siemens.com/mall/en/us/Catalog/Product/6ES72121BE400XB0

BAKHANOVICH, A. G.; KUSYAK, V. A. ; GURIN, A.N.; LE, V. N. Electronic control for fuel supply of diese engine on the basis of programmable PID – regulator. [online]. 2017 [cit. 2022-06-20]. Science & Technique. Dostupné z: https://www.researchgate.net/publication/313223010_ELECTRONIC_CONTROL_FOR_FUEL_SUPPLY_OF_DIESEL_ENGINE_ON_THE_BASIS_OF_PROGRAMMABLE_PID-REGULATOR

BERESFORD, D. Exploiting Siemens Simatic S7 PLCs. [online]. 2011 [cit. 2022-06-20]. Black Hat USA+2011. Dostupné z: https://paper.bobylive.com/Meeting_Papers/BlackHat/USA-2011/BH_US11_Beresford_S7_PLCs_WP.pdf

BERGER, H. Automating with SIMATIC S7-1200. Configuring, Programming and Testing with STEP 7 Basic, Visualization with WinCC Basic. 2013. Second Edition. SIEMENS. ISBN 978-3-89578-385-2

BLAHA, M. Hierarchické divizní shlukování. [online]. 2015 [cit. 2022-06-20]. Matematická biologie. Dostupné z: https://portal.matematickabiologie.cz/index.php?pg=analyza-a-hodnoceni-biologickych-dat--umela-inteligence

BLAHA, P.; VAVŘÍN, P. Řízení a regulace I. [online]. 2020 [cit. 2022-06-20]. Základy regulace lineárních systémů – spojité a diskrétní (v. 1.3.9). VUT v Brně, Fakulta elektrotechniky a komunikačních technologií. Dostupné z: https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxyaXplbmlhcmVndWxhY2UxfGd4OjYxNTNlYmM0NTJjYmViZTE

DOLEŽEL, P. Úvod do umělých neuronových sítí pro studenty technických vysokých škol. 2016. Pardubice: Univerzita Pardubice. ISBN 978-80-7560-022-6.

JAIN, A. K.; MAO, J.; MOHIUDDIN, K.M. Artificial Neural Networks: A Tutorial. [online]. 1996 [cit. 2022-06-20]. Michigan State University. IBM Almaden Research Center. Dostupné z: https://csc.lsu.edu/~jianhua/nn.pdf

KADLEC, P.; GABRYS, B.; STRANDT, S. Computers and Chemical Engineering. Data-driven Soft Sensors in the process industry. [online]. 2008 [cit. 2022-06-20]. Elsevier Ltd.

ISSN 0098-1354. Dostupné z: https://www.academia.edu/2213478/Data_driven_soft_sensors_in_the_process_industry

LEI, B.; XU, G.; FENG, M.; ZOU, Y.; VAN DER HEIJDEN, F.; DE RIDDER, D.; M. J. TAX, D. 2017. Classification, Parameter Estimation and State Estimation. An Engineering Approach Using MATLAB. Second Edition. John Wiley & Sons, Ltd. ISBN 9781119152439

MOURA, S. CE 295 — Energy Systems and Control. [online]. 2018 [cit. 2022-06-20]. University of California, Berkeley. Dostupné z: https://www.coursehero.com/file/53948028/CH02-StateEstimationpdf/

NANJUNDAIAH, V. PLC Ladder Logic Basics. [online]. 2019 [cit. 2022-06-20]. Dostupné z: https://www.ezautomation.net/industry-articles/plc-ladder-logic-basics.htm

NOVÁK, M.; FABER, J.; KUFUDAKI, O. Neuronové sítě a informační systémy živých organismů. 1993. 1.vyd Grada Praha. ISBN 80-85424-95-9

PAULSSON, D.; GUSTAVSSON, R.; MANDENIUS, C.-F. A Soft Sensor for Bioprocess Control Based on Sequential Filtering of Metabolic Heat Signals. [online]. 2014 [cit. 2022-06-20]. Sensors 2014. ISSN 1424-8220. Dostupné z: https://www.researchgate.net/publication/266264062_A_Soft_Sensor_for_Bioprocess_Control_Based_on_Sequential_Filtering_of_Metabolic_Heat_Signals

PAZ, R. A. The Design of the PID Controller. [online]. 2001 [cit. 2022-06-20]. Klipshv School of Electrical and Computer Engineering. Dostupné z: https://www.researchgate.net/profile/Robert-Paz-2/publication/237528809_The_Design_of_the_PID_Controller/links/004635360fa1ebdf63000000/The-Design-of-the-PID-Controller.pdf

PETERSON, D. The Origin Story of the PLC. [online]. 2022 [cit. 2022-06-20]. Dostupné z: https://control.com/technical-articles/the-origin-story-of-the-plc/

SUZUKI, K. Artificial Neural Networks – Methodological Advances and Biomedical Applications. 2011. ISBN 978-953-307-243-2

ŠÍMA, J.; NERUDA, R. Teoretické otázky neuronových sítí. 1996. Praha: Matfyzpress, vyd. 1. , 390 s. ISBN 80-85863-18-9

ŠVARC, I. Teorie automatického řízení. [online]. 2003 [cit. 2022-06-20]. VUT FSI. Dostupné z:

http://www.fsiforum.cz/upload/soubory/knihy/Automatizace/Teorie.automatickeho.rizeni_Svarc_2003.pdf

The Evolution of PLCS. [online]. 2021 [cit. 2022-06-20]. Dostupné z: https://www.plctechnician.com/news-blog/evolution-plcs

ZAKARIA, M.; AL-SHEBANY, M.; SARHAN, S. Artificial Neural Network: A Brief Overview. [online]. 2014 [cit. 2022-06-22]. Shahenda Sarhan Sirte University SIRTE, LIBYA. Dostupné z: https://issuu.com/www.ijera.com/docs/b42010712/1

PŘÍLOHY

A - CD

Příloha A

Příloha k diplomové práci

REALIZACE SOFT SENZORU V PLC POMOCÍ UMĚLÉ NEURONOVÉ SÍTĚ

Bc. Tomáš Hort

CD

Obsah

1 Text diplomové práce ve formátu PDF

2 Zdrojové kódy