VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

VIRTUÁLNÍ MODEL TECHNOLOGICKÉHO PROCESU ŘÍZENÝ PLC
A VIRTUAL MODEL OF A TECHNOLOGY PROCESS WITH PLC CONTROL

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR
Bc. Adam Hendl

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Miroslav Jirgl, Ph.D.

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Kybernetika, automatizace a měření**

Ústav automatizace a měřicí techniky

Student: Bc. Adam Hendl ***ID:*** 186079

Ročník: 2 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Virtuální model technologického procesu řízený PLC

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je implementace funkce pro komunikaci MATLAB s vybraným typem PLC, návrh a implementace virtuálního 3D modelu jednoduchého technologického procesu (zařízení) v prostředí MATLAB/Simulink a následně jeho řízení pomocí PLC.

1. Prostudujte komunikační možnosti prostředí MATLAB s reálným řídicím systémem (PLC).
2. V prostředí MATLAB vytvořte obslužnou funkci pro komunikaci s PLC a otestujte.
3. Prostudujte problematiku tvorby virtuálních modelů v prostředí MATLAB/Simulink.
4. Navrhněte a realizujte jednoduchý 3D model vybraného technologického procesu a popište jeho zjednodušený matematický model.
5. Propojte 3D model s matematickým modelem realizovaným v prostředí Simulink a otestujte.
6. Navrhněte řízení a ověřte jeho vlastnosti pomocí simulace.
7. Propojte model v prostředí MATLAB/Simulink s PLC a implementujte řízení do PLC.
8. Ověřte funkčnost a porovnejte dosažené výsledky se simulací.

DOPORUČENÁ LITERATURA:

DANĚK, Jan. Vizualizace dynamických systémů v prostředí virtuální reality. Humusoft, 2012.

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 1.6.2020

Vedoucí práce: Ing. Miroslav Jirgl, Ph.D.

doc. Ing. Václav Jirsík, CSc.
předseda oborové rady

Abstrakt

Diplomová práce prozkoumává komunikační metody, kterými se dají propojit prostředí MATLAB s reálným PLC. Poté popisuje postup nastavení parametrů komunikačních bloků a funkcí v PLC a PC, tak aby byla možnost vzdáleného řízení části výrobního procesu za použití UDP protokolu. Část výrobní procesu je v PC namodelován zjednodušeným matematickým a 3D model. Součástí práce je návrh modelů teploty a výšky hladiny. Část práce je také věnována návrhu regulátorů různými metodami a následnému porovnání dosažených výsledků. Na závěr této práce jsou porovnány výsledky získanými simulací a vzdáleným řízením.

Klíčová slova

UDP, PID, MATLAB, SIMULINK, VRML

Abstract

The diploma thesis examines the communication methods that can be used to connect the MATLAB environment with a real PLC. It then describes the procedure for setting the parameters of communication blocks and functions in PLC and PC, so that it is possible to remotely control part of the production process using the UDP protocol. Part of the production process is modeled in a PC with a simplified mathematical and 3D model. Part of the work is the design of temperature and level models. Part of the work is also devoted to the design of controllers by various methods and the subsequent comparison of the achieved results. At the end of this work, the results obtained by simulation and remote control are compared.

Keywords

UDP, PID, MATLAB, SIMULINK, VRML

Bibliografická citace:

HENDL, Adam. *Virtuální model technologického procesu řízený PLC* [online]. Brno, 2020 [cit. 2020-06-01]. Dostupné z: https://www.vutbr.cz/studenti/zav-prace/detail/126911. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Miroslav Jirgl.

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma *Virtuální model technologického procesu řízený PLC* jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: **1. června 2020**

.....
podpis autora

Poděkování

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

V Brně dne: **1. června 2020**

……
podpis autora

Obsah

Seznam symbolů a zkratek

Zkratky:

VRML	…	Virtual Reality Modeling Language
VR	…	Vitrual Reality
TCP	…	Transmission Control Protocol
IP	…	Internet Protocol
UDP	…	User Datagram Protocol
ODK	…	Open Data Kit
CPU	…	Central Processing Unit
PLC	…	Programmable Logic Controller
MAUP	…	Předmět Automatizace procesů v magisterském programu

Symboly:

S	…	plocha	$[m^2]$
V	…	objem	$[m^3]$
$q_1(t)$	…	množství přitékající kapaliny	$[m^3/sec]$
$q_2(t)$	…	množství odtékající kapaliny	$[m^3/sec]$
$y_1(t)$	…	výška hladiny	$[m]$
R	…	koeficient úměrnosti	$[sec/m^3]$
P	…	výkon	[W]
ϑ_i	…	teplota topného tělesa	[°C]
ϑ_V	…	teplota vody	[°C]
ϑ_0	…	teplota okolí	[°C]
$d\vartheta_V(t)$	…	změna teploty vody	[°C]
$d\vartheta_i(t)$	…	změna teploty tělesa	[°C]
$p(t)$	…	množství tepla za čas dt	[W]
m_V	…	hmotnost vody	[kg]
c_V	…	specifické teplo vody	$[W_s/(kg°C)]$
K_V	…	koeficient přestupu tepla vody	[W/°C]
m_i	…	hmotnost topného tělesa	[kg]
c_i	…	specifické teplo tělesa	$[W_s/(kg°C)]$
K_i	…	koeficient přestupu tepla topného tělesa	[W/°C]

Seznam obrázků

Seznam tabulek

1. ÚVOD

Tato diplomová práce se bude zabývat vytvořením 3D modelu využitelného pro simulaci jednoduchého výrobního procesu, a to konkrétně ohřevu a následné destilaci lihu ve dvouplášťovém rektifikačním tanku, a návrhem regulátoru pro otopnou soustavu takového tanku.

Rektifikační tank je určený pro zkvalitnění pálenky po první destilaci ovocného kvasu a odstranění nežádoucích látek, jako je například metanol apod. Skládá se ze dvou různě velkých, symetrických válců, vyráběných z nerezové oceli, které jsou zasunuté do sebe. V prostoru mezi nimi je voda, která slouží k ohřevu vnitřního válce a jeho obsahu. Ve chvíli dosažení bodu varu uvnitř válce začne docházet k odpařování a tím k druhotné destilaci, rektifikaci. V závislosti na aktuální teplotě se uvolňují různé látky, které se na základě jejich vlastností buď zachytávají, nebo se likvidují. Proto je vhodné řídit teplotu v tanku pomocí regulátoru. Ten zajistí pozvolné ohřívání a v důsledku toho lepší oddělení jednotlivých složek.

V první části práce nejprve zmapuji možnosti komunikace mezi PLC S7-1500 od firmy Siemens a počítačem, na kterém je spuštěn program MATLAB se simulačním prostředím Simulink.

Po prostudování možností a výběru jedné z nich ve druhé části navrhnu obslužné funkce pro komunikaci v PC a PLC, které mají za úkol výměnu dat mezi komunikačními partnery, a tím připravit možnost řízení dané virtuální technologie vzdáleně pomocí PLC.

Třetí část se bude věnovat seznámení se s 3D modelováním v prostředí Simulink pomocí jazyka VRML a následně vytvoření a namodelování vybraného jednoduchého technologického procesu. V této části bude také vytvořen matematický popis pro tento technologický proces.

Matematický popis tanku je pak spojen s 3D modelem pomocí knihovny 3D Animation do jednoho celku. Takový model srozumitelněji reprezentuje dění v nádrži. Pro tuto soustavou jsou následně navrhnuty dva regulátory, jejichž úkolem je regulovat teplotu v tanku. Tyto regulátory jsou odzkoušeny simulací.

Další část práce se věnuje propojení matematického a 3D modelu s PLC a implementací dříve navržených regulátorů do PLC, a navíc jednoho automatického návrhu pomocí funkce pro nastavení PID regulátorů v programovacím prostředí TIA Portal.

V poslední části porovnám zaznamenané průběhy výstupních hodnot a akčních zásahů při řízení různými regulátory, a vyhodnotím, který je pro danou aplikaci nejvhodnější.

Cílem této diplomové práce je tedy po zprovoznění komunikace, vytvoření 3D a matematického modelu soustavy a návrhu regulátorů naimplementovat řízení do PLC a řídit danou soustavu vzdáleně pomocí obslužných komunikačních funkcí. Na závěr pak ještě porovnat dosažené výsledky se simulací.

Takováto virtualizace výrobního procesu s navrženým regulátorem může v praxi sloužit pro výrobce zařízení pro palírny k otestování výrobků před jejich uvedením do výroby. Umožní tak odstranit případné chyby v návrhu již na začátku vývoje a omezit tak zbytečně vynaložené náklady. Pokud by se výrobek testoval až při nasazení do provozu, mohlo by dojít k materiálním škodám nebo újmě na zdraví, což je nepřípustné.

Navržený model a regulátor mohou být také využity při výuce nebo simulacích v laboratořích.

2. MOŽNOSTI KOMUNIKACE PLC S PROSTŘEDÍM MATLAB/SIMULINK

PC s prostředím MATLAB i PLC S7-1500 od firmy Siemens podporují několik komunikačních protokolů. Některé jsou podporovány nativně (např. TCP/IP, UDP), pro podporu jiných potřebují speciální hardware nebo software (např. RS485, GPIB). Každý komunikační protokol má svoje specifika a hodí se k něčemu jinému.

2.1 Společná komunikační rozhraní

V následující tabulce jsou uvedené možné způsoby komunikace. Pokud zařízení uvedený způsob podporuje, je v jeho sloupci na příslušném řádku uvedena tečka.

Tabulka 2-1 Přehled komunikačních možností

Zařízení / Kom. protokol	S7	MATLAB
TCP	•	•
UDP	•	•
Modbus TCP	•	•
SIMATIC Targer 1500S for Simulink	•	•
IP-S7-LINK	•	•
GPIB		•
VISA		•
I^2C		•
PROFIBUS PA	•	

2.2 TCP

TCP je jeden z nejběžněji používaných protokolů a jeho transportní vrstva zajišťuje kontrolu celistvosti dat a doručování zpráv je díky navazování a ukončení spojení před odesláním dat spolehlivější.

2.2.1 S7 TCP

Siemens řadí TCP mezi tzv. „connection-oriented protocols“ (protokoly orientované na připojení). Tyto protokoly navazují logické spojení s komunikačním partnerem před zahájením přenosu dat. Po dokončení přenosu dat pak připojení v případě potřeby ukončí. Pro přenos dat se používají protokoly zaměřené na připojení, pokud má spolehlivé a zaručené doručení zvláštní význam. Obecně může na jedné fyzické lince existovat mnoho logických spojení. [1]

V PLC S7 jsou podporovány tři protokoly orientované na připojení: „Ad-hoc“, „přenos orientovaný na zprávu“ a „ISO on TCP“.

„Ad-hoc“

Během přenosu nejsou přenášeny informace o tom, kolik bytů je posíláno nebo kde je začátek a konec zprávy. To není problém během odesílání dat, protože odesílatel ví, kolik bytů dat odesílá. Problém vzniká na straně příjemce, protože ten nemá ponětí o tom, kde jedna zpráva končí a druhá začíná. [1]

Pro režim „ad-hoc“ (kdykoliv je potřeba) musí být parametr *LEN* funkčních bloků *TRCV* a *TSEND* roven nule. Přijatá data jsou okamžitě zpřístupněna při zavolání bloku. Počet přijatých bytů je k dispozici v parametru *RCVD_LEN*. [1]

„Přenos orientovaný na zprávu“

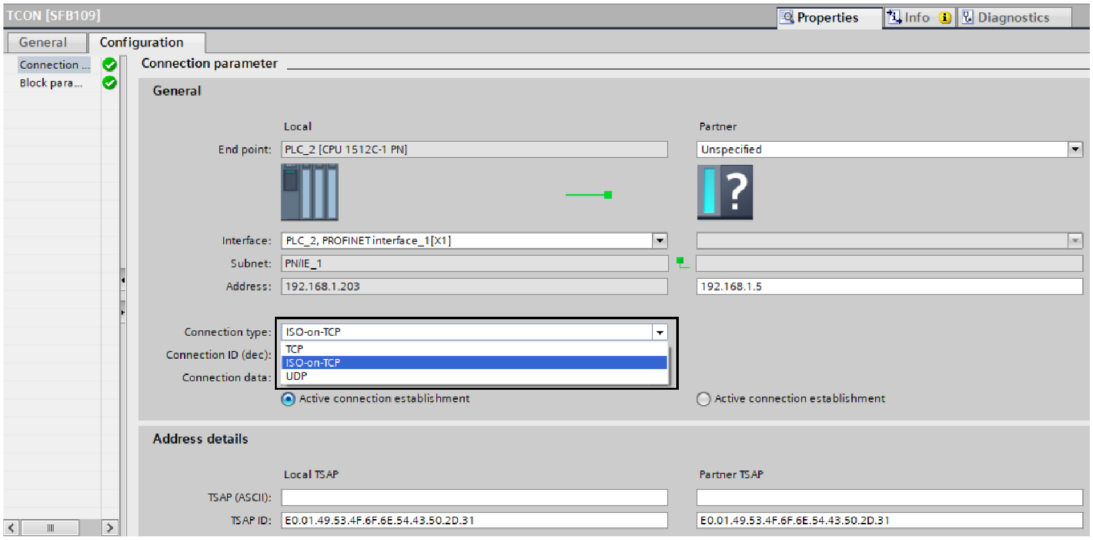
Aby se předešlo problémům s nalezením počátku a konce zprávy, Siemens doporučuje nastavit shodný počet odeslaných, resp. přijatých bytů u bloků TSEND a TRCV parametrem LEN. Pokud je nastaven vyšší počet bytů zprávy u TRCV, než je počet odeslaných bytů, tak blok TRCV přijatá data přesune do specifikované oblasti paměti (parametr DATA) až když se splní podmínka, že se počet přijatých bytů rovná hodnotě LEN. To znamená, že se spojí více zpráv dohromady a opět nebude přehled o začátku a konci zprávy, pokud není známa délka první zprávy. Pokud je nastaven nižší počet u TRCV než u TSEND, tak se zkopíruje pouze počet bytů uveden v *LEN*. To znamená, že jednu zprávu je nutno vyčíst dalším zavoláním funkčního bloku. [1]

„ISO on TCP“

Při tomto způsobu přenosu je přenášen údaj o počtu bytů i o konci zprávy. Výběr tohoto protokolu je uskutečněn pomocí vybrání možnosti ISO-on-TCP v nastavení parametrů funkčního bloku TCON, viz Obrázek 2-1. U odesílatele i příjemce se nastavuje délka zprávy stejně jako v předchozí kapitole 0. [1]

Pokud je přijatá zpráva kratší než nastavená délka, tak se data uloží do zadané oblasti dat a potvrdí se příjem. Ale pokud přijatá data přesáhnou nastavenou délku zprávy, tak se data neuloží, ale místo toho se nastaví *ERROR* na log. „1“ a *STATUS* na hodnotu 8088 hexadecimálně. [1]

Popis tohoto rozšíření protokolu TCP lze nalézt v dokumentu RFC 1006 (Request for Comments). [1]



Obrázek 2-1 Nastavení komunikačního protokolu

2.2.2 MATLAB TCP

V programu MATLAB jsou dvě možnosti, jak vytvořit TCP spojení. Jedna možnost je využít funkci *tcpclient*, která je v MATLABu integrována a je zdarma. Druhou možností je využít Instrument Control Toolbox, který mimo jiné obsahuje funkci *tcpip*. Tento toolbox je placený, což může být limitující faktor.

tcpclient

Tento příkaz vytvoří a nastaví TCP/IP klienta, a tím umožní připojení se ke vzdálenému partnerovi. Dále vykonává příjem a odesílání dat. Pro komunikaci přes TCP/IP se nejdříve musí zavolat příkaz: `<objname> = tcpclient(Address, Port)`,

kde *objname* je proměnná typu *tcpip* a obsahuje všechny nastavené parametry komunikace,

Address je IP adresa komunikačního partnera (serveru), může být zadána ve formátu IPv4, IPv6 i URL adresy,

Port je port, na který se klient připojuje, číslo portu je v rozmezí 1 až 65535,

Timeout je volitelný argument nastavuje, který maximální čas, po který se bude funkce snažit přijmout, nebo odeslat data,

ConnectTimeout je volitelný argument nastavuje, který maximální čas, po který se bude funkce snažit připojit se k partnerovi [2]

tcpip

Příkaz *tcpip* vytvoří *tcpip* komunikační objekt obdobně jako funkce *tcpclient*, ale pomocí této funkce lze vytvořit i TCP/IP server, což *tcpclient* neumožňuje. Příkaz je obdobný jako u předcházející funkce: `<objname> = tcpip(RemoteHost, RemotePort, 'NetworkRole', 'server'` (příp. `'client'`),

kde *objname* je proměnná typu *tcpip* a obsahuje všechny nastavené parametry komunikace,

RemoteHost je IP adresa komunikačního partnera, může být ve formátu IPv4, IPv6,

RemotePort je komunikační port vzdáleného partnera, na který se připojuje, číslo portu je v rozmezí 1 až 65535,

NetworkRole nastaví komunikační roli na server, nebo na klienta podle následujícího argumentu, jak je uvedeno v příkladu funkce,

ConnectTimeout je volitelný argument, který nastavuje maximální čas, po který se bude funkce snažit připojit se k partnerovi [3]

2.3 UDP

UDP je také často využívaný protokol. Jeho transportní vrstva nezajišťuje kontrolu celistvosti dat ani v jakém dojdou pořadí. Tento protokol může být rychlejší než TCP.

2.3.1 S7 UDP

UDP se podle Siemens řadí mezi tzv. „connectionless protocols“ (nespojované protokoly), jelikož nedochází k navázání a ukončení spojení se vzdáleným partnerem. Nedochází ani k potvrzení přijetí zprávy, a jsou tedy bez spolehlivého a zaručeného doručení vzdálenému partnerovi. [1]

Na rozdíl od TCP protokolu se na začátku komunikace nenavazuje spojení, ale místo toho se ve funkčním bloku *TUSEND* specifikuje IP adresa a číslo portu protější strany. Podobně je na konci příchozí zprávy informace o odesílateli v podobě jeho IP adresy a čísla portu. Tyto informace jsou k nalezení ve funkčním bloku *TURCV*. [1]

Aby bylo možné využít bloky *TUSEND* a *TURCV*, musí se nejprve zavolat blok *TCON* na obou stranách komunikace. Tento krok je nutný k nastavení lokálních komunikačních přístupových bodů. Výběr UDP protokolu probíhá stejně jako u variant TCP, viz Obrázek 2-1. [1]

Při každém novém zavolání bloku *TUSEND* se znovu zadají komunikační parametry partnera (IP adresa a číslo portu). Během přenosu se přenáší informace o délce a konci zprávy. [1]

Pokud je zadaná délka (*LEN*) zprávy v bloku *TURCV* větší, než je skutečná délka zprávy, tak se data zkopírují do specifikované oblasti a výstup *NDR* se nastaví na log. „1“ a v *RCVD_LEN* bude skutečná délka přijatých dat. Jestliže bude délka přijaté zprávy větší

než parametr *LEN*, tak se data nezkopírují a místo toho se výstup *ERROR* nastaví na log. „1“ a *STATUS* na 8088 hexadecimálně. [1]

2.3.2 MATLAB UDP

Protokol UDP je v MATLABu podporován prostřednictvím Instrument Control Toolbox. Princip vytvoření *udp* objektu je podobný jako vytváření *tcpip* objektu. Objekt udp musí být před použitím svázán s lokálním socketem zavoláním funkce *fopen*. Příklad funkce: `u = udp(RemoteHost,RemotePort,'LocalPort',LocalPort),`

kde *RemoteHost* je asociace na vzdáleného partnera, může to být IP adresa, nebo jméno partnera.

RemotePort je specifikace vzdáleného portu, výchozí hodnota je 9090, rozsah hodnot je 1 až 65535.

LocalPort nastavuje místní port klienta, jedná se o argument typu pár „jméno-hodnota“.

Velikost vstupního bufferu je uložena v parametru *InputBufferSize*. Obdobně velikost výstupního bufferu je uložena v parametru *OutputBufferSize*. Výchozí velikosti jsou 512 B. Maximální velikost příchozího packetu je 8192 B a odchozího 4096 B. [4]

2.4 IP-S7-LINK

Tento toolbox pro MATLAB byl vyvinut třetí stranou – německou firmou Traeger. Tento toolbox poskytuje propojení MATLABu a PLC od firmy Siemens pomocí komunikace založené na TCP/IP. Umožnuje zápis a čtení dat do, resp. z PLC a podporuje všechny datové typy (např. databloky, čítače, časovače, vstupy a výstupy). Není potřeba žádný další software na straně PLC. Nevýhodou tohoto řešení je, že toolbox je placený. Na stránkách výrobce je ke stažení evaluační verze, která je zdarma. [5]

```
device = SiemensDevice(IPDeviceEndPoint('192.168.0.80'),
SiemensDeviceType.S7300_400);
connection = device.CreateConnection();
connection.Open();
connection.WriteString('DB111.DBB 100', 'Hello World!');
message = connection.ReadString('DB111.DBB 100', 16);
```

Obrázek 2-2 Příklad komunikace pomocí IP-S7-LINK [5]

2.5 SIMATIC Target 1500S

Target 1500S je softwarový modul pro jednoduchý a přímý přenos modelů regulátorů a strojů z prostředí Simulink do softwarového kontroléru SIMATIC S7-1500, SIMATIC ET 200SP Open Controller, nebo do CPU 1518 ODK [6]

Jsou podporovány téměř všechny funkce Simulinku s kontrolérem S7-1500, který podporuje ODK. Je možno navrhnout regulátor v Simulinku, namodelovat kompletní

systém, provést simulaci a potom použít Target 1500S k automatickému vygenerování kódu v jazyce C, ODK objekt a S7 programové moduly. Vygenerovaný kód je možné použít na standardních i bezpečnostních PLC. [6]

Modul umožňuje testování modelů v PLC (tvz. Hardware-in-the-loop). Externí mód Simulinku může být využit pro sledování modelu na PLC a případné změny parametrů se projeví přímo v Simulinku. Také vnitřní parametry modelu mohou být změněny přímo z programu PLC. Tento postup ušetří čas, protože není nutné manuální psaní kódu, a tím je proces návrhu také spolehlivější. Přenos modelů již nevyžaduje znalosti psaní kódu v jazyce C, C++ nebo ODK. Regulátor lze jednoduše odladit v Simulinku, čímž se snižuje počet fyzických prototypů, a to vede k rychlému prototypování. [6]

Nevýhodou tohoto řešení je, že ho lze použít pouze na hardwaru, který má vyšší cenu a není k vyzkoušení v laboratořích.

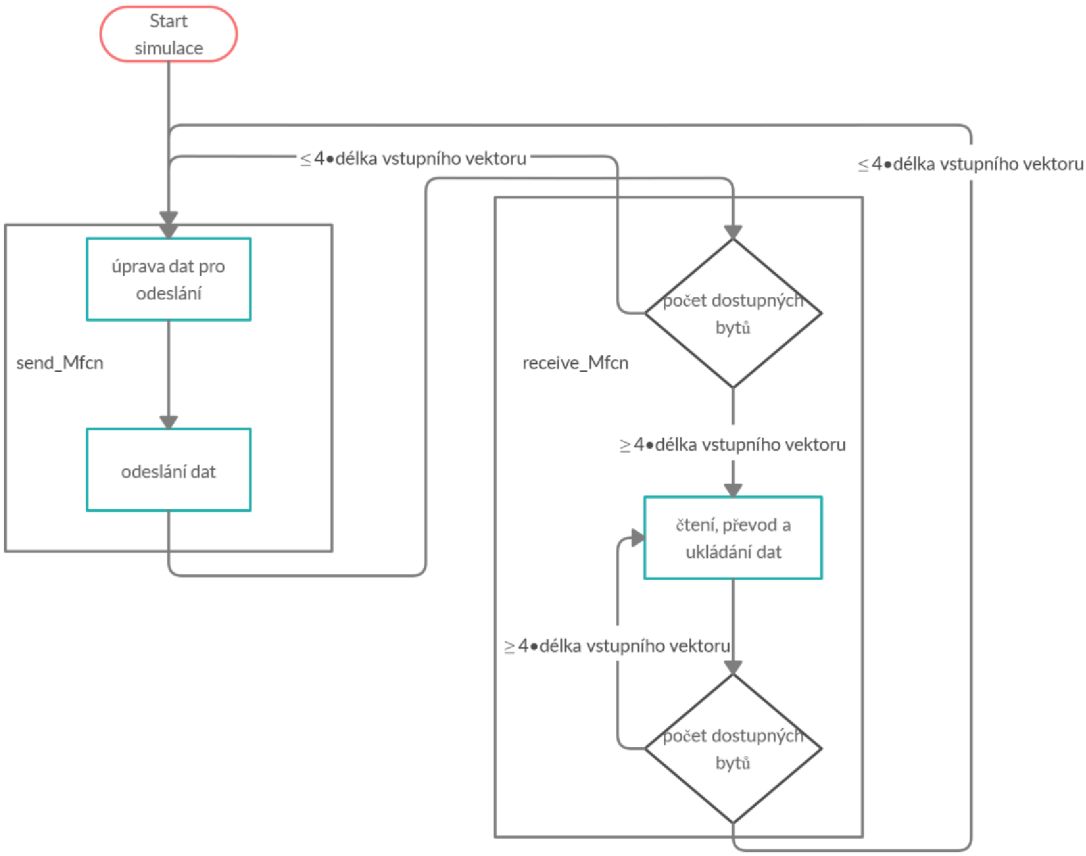
2.6 Modbus TCP

Pro komunikaci mezi PC a PLC by mohl být využit i otevřený komunikační protokol Modbus TCP od společnosti Modicon (nyní Schneider Electric), jelikož ho obě strany podporují. V průmyslu se často využívá, takže je implementován v PLC S7-1500. A na straně MATLABu je integrován v Instrument Control Toolboxu. Jedná se o modifikaci TCP/IP protokolu, který má vlastní aplikační vrstvu.

3. FUNKCE POTŘEBNÉ PRO OBSLUHU KOMUNIKACE

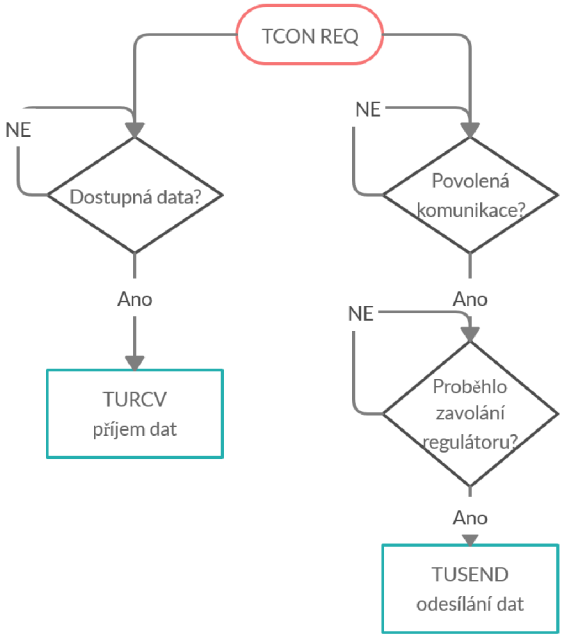
3.1 Popis komunikace

Princip algoritmu komunikace na straně PC objasňuje vývojový diagram na Obrázek 3-1. V simulaci jsou volány dvě funkce, které obsluhují komunikaci: *send_Mfcn* a *receive_Mfcn.* Ty provádějí odesílání, resp. příjem dat a jejich interpretaci pro daný systém.



Obrázek 3-1 Vývojový diagram obsluhy komunikace v prostředí MATLAB/Simulink

Fungování obsluhy komunikace v PLC je naznačeno ve vývojovém digramu na Obrázek 3-2. Pro odesílání a přijímaní dat jsou zde naimplementovány dvě funkce, které jsem použil: *TUSEND* a *TURCV*



Obrázek 3-2 Vývojový diagram principu obsluhy komunikace v PLC

3.2 Funkce pro komunikaci v Matlab

Na straně PC, kde je spuštěna instance MATLAB, je vybraný UDP komunikační protokol realizován pomocí funkcí, které jsou popsány dále.

3.2.1 Instrument Control Toolbox

Tento toolbox umožňuje propojit MATLAB s různými přístroji, jako jsou funkční generátory, osciloskopy, zdroje atd., a to přes různé komunikační protokoly. V mém případě jsem využil možnosti tohoto toolboxu vytvořit UDP komunikačního klienta na počítači. [7]

Funkce udp

Pro vytvoření objektu typu *udp* volám funkci *udp* a přiřazuju ji do proměnné, kterou dále využívám. Tuto funkci využívám se třemi vstupními parametry: IP adresa PLC (partnera), číslo portu partnera a místní port. Příklad funkce *udp* a její syntaxe: *u = udp('192.168.1.203',4019,'LocalPort',4020).*

Po zavolání této funkce se do proměnné *u* uloží informace, jako je například pořadí bitů v bytu (little endian, nebo big endian), kolik je bytů k dispozici ve vstupním a výstupním bufferu, IP adresa partnera, komunikační port, status (jestli je komunikace aktivní, nebo ne), počet přijatých a odeslaných bytů atd. Vypsáním proměnné *u* do *Command Window* se zobrazí stručný výpis těchto informací. [4]

Funkce fopen

Tato funkce otevře komunikační port, který ji se předám ve vstupní proměnné *u* z funkce *udp*. Po zavolání funkce se změní status v proměnné *u* z „closed“ na „opened“. To značí, že je vše připraveno pro přenos dat ze strany MATLABu. Příklad funkce a její syntaxe: *fopen(u)*.

Funkce fread

Funkci *fread* používám k vyčtení dat ze vstupního bufferu v proměnné *u*. Vstupním parametrem funkce je identifikátor vstupního souboru a počet vstupních hodnot, které chci vyčíst. Jelikož programovatelný automat S7 a jeho funkce *TSEND* posílá proměnné typu „real“ rozložené do 4 bytů, tak musí být počet vstupních hodnot vynásoben čtyřmi. Pokud bych posílal například proměnnou typu „bool“, tak tento krok není potřeba. Abych složil zpátky hodnotu, kterou vyjadřuje desetinné číslo místo čtyř bytů, musím použít následující příkazy:

- *flipud* – horizontálně převrátí vertikální vektor, vstup – vektor s hodnotami
- *uint8* – přetypuje vstupní hodnoty na bezznaménkový 8bitový integer, vstup – vektor s hodnotami [8]
- *typecast* – převede vstupní hodnoty ze čtyř 8bitových integerů *uint8* na jedno desetinné číslo s jednoduchou přesností *single*, vstupem je vektor hodnot a datový typ, do kterého mám být vektor převeden [9]
- *flipud* – výsledný vektor čísel ještě jednou převrátí, z důvodu srozumitelnější interpretace přijatých hodnot, vstup – vektor s hodnotami

Příklad funkce a její syntaxe:

```
Data_from_PLC = flipud(typecast(uint8(flipud(fread(t,NumOfBytes))),'single'));
```

Funkce fwrite

Pro odeslání dat do PLC používám funkci *fwrite*. Vstupem této funkce je proměnná *u* a data, která se mají poslat. Na vektor vstupních proměnných jsem použil následující funkce, abych vytvořil takový formát dat, který je vyžadován ze strany PLC:

- *fliplr* – vertikálně převrátí vektor hodnot, hodnoty jsou zapsány ve vektoru tak, aby byly snáze pochopitelné
- *single* – hodnoty převede hodnoty na 32bitová desetinná čísla s plovoucí čárkou s jednoduchou přesností [10]
- *typecast* – převede vstupní hodnoty z jednoho 32bitového desetinného čísla *single* na čtyři 8bitové integery *unit8*, argumentem funkce je vektor hodnot a datový typ, do kterého mám být vektor převeden [9]
- *fliplr* – opětovné převrácení vektoru, aby odpovídalo pořadí bytů reprezentaci v PLC

- *rot90* – otočí vektor okolo své osy o 90 °, aby odpovídal rozměr přenášeného vektoru hodnot v PC i PLC

Příklad funkce a její syntaxe: Data_to_PLC = rot90([zadana_teplota skutecna_teplota zadana_hladina skutecna_hladina]);

fwrite(t,fliplr(typecast(single(Data_to_PLC),'uint8')));

Evalin a Assignin

Aby se mohla funkce *fwrite* nebo *fread* provést, musí být proměnná *u* uložena ve stejném paměťovém prostoru, v jakém je volaná funkce *fwrite,* resp. *fread.*

Pokud je potřeba přiřadit například novou hodnotu do proměnné, která je uložena v paměťové prostoru *base*, tak se využije knihovní funkce *assignin*. Tato funkce má tři parametry: paměťový prostor, proměnnou v daném paměťovém prostoru a hodnotu, popřípadě proměnnou z prostoru, odkud je funkce volána.

Pro zavolání funkce, v jiném paměťovém prostoru, například protože využívá proměnné z toho daného prostoru slouží knihovní funkce *evalin*. Funkce má dva vstupní parametry: paměťový prostor a danou funkci s příslušnými parametry.

Aby bylo možné tyto funkce používat, je potřeba na začátku každé funkce, která je využívá, napsat `coder.extrinsic('evalin')`, případně `('assignin')`. To zajišťuje, že překladač části kódu s těmito funkcemi nepřekládá dopředu, ale až za běhu programu. Díky tomu lze použít proměnné, které v daném prostoru neexistují. Překladač by jinak při překladu hlásil chybu.

3.3 Bloky pro UDP komunikaci Siemens

Komunikačním protokolem, který jsem pro výměnu dat mezi PLC a PC vybral, je UDP protokol. Ten je na straně PLC Siemens S7-1500 obsluhován pomocí bloků *TCON*, *TDISCON*, *TUSEDN* a *TURCV*.

Společné parametry komunikačních bloků:

Vstupní parametry:

- *REQ* – požadavek pro provedení určité funkce (spojení, zrušení spojení, poslání a příjem dat)
- *ID* – reference na přiřazené spojení pomocí *TCON* (1–4095) [1]

Výstupní parametry:

- *DONE* – tento příznak značí, zda byl požadavek uskutečněn (log. „1“), nebo ještě nezačal nebo nebyl proveden (log. „0“)
- *BUSY* – pokud je tento výstup v log. „1“, tak se ještě vykonává předchozí požadavek a nemůže se vykonat další, pokud je v log. „0“, tak ho lze vykonat

- *ERROR* – pokud se objeví chyba, tak se výstup nastaví na log. „1“, jinak v log. „0“
- *STATUS* – vyjadřuje stav, ve kterém se funkční blok nachází [1]

Všechny vstupní a výstupní parametry komunikačních bloků jsou uloženy v datovém bloku *Comm_DB [DB6]*.

Rozdílné parametry jednotlivých komunikačních bloků a jejich popis

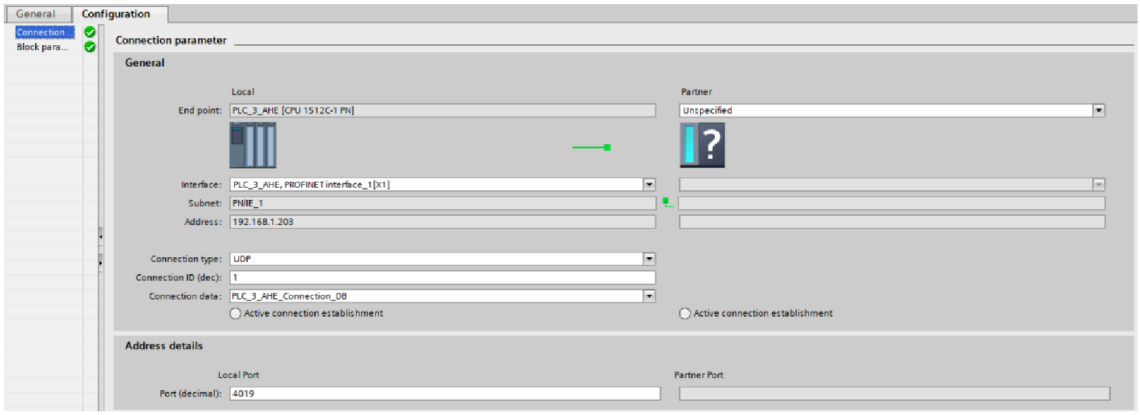
TCON

Tato instrukce slouží k nastavení a navázání komunikace. Jakmile je spojení nastaveno a navázáno, tak ho CPU monitoruje a udržuje. Instrukce je vykonávaná asynchronně. [1]

Vstupní/výstupní parametry:

- CONNECT – ukazatel na popis spojení ve formě struktury (obsahuje například IP adresu vzdáleného zařízení, vzdálený port a typ komunikace (TCP, ISO-on-TCP, nebo UDP) [1]

Nastavení parametrů této instrukce, jako je IP adresa PLC a partnera (PC), typu připojení, portu atd., se nachází v *Properties* → *Configuration* → *Connection* viz Obrázek 3-3.



Obrázek 3-3 Nastavení TCON v TIA Portal

TDISCON

Tento funkční blok ukončuje komunikaci mezi CPU automatu a komunikačním partnerem. Instrukce je vykonávaná asynchronně. Ukončení se vyvolá přivedením vzestupné hrany na vstup *REQ*. Po úspěšném vykonání instrukce již není identifikátor *ID* platný a nelze ho použít pro posílání a příjem. [1]

TUSEND

TUSEND je instrukce určená k posílání dat. Data se při posílání nesmí měnit, protože se instrukce provádí asynchronně. Požadavek na poslání dat představuje vzestupná hrana na vstupu REQ. Pokud se poslání provede úspěšné na výstupu DONE se nastaví log. „1“, to ale neznamená, že se komunikační partner správně přijal. [1]

Vstupní parametry:

- LEN – maximální počet bytů, který se má poslat [1]

Vstupní/výstupní parametry:

- DATA – ukazatel na data, která se mají odeslat, může se jednat od obraz vstupů, výstupů, paměťové bity, datové bloky (pokud se jedná o struktury, tak musí být identické na vysílači i přijímači) [1]
- ADDR – obsahuje IP adresu a vzdálený port komunikačního partnera. Pro zapsaní těchto informací je potřeba vytvořit novou proměnnou v datovém bloku (např. *Comm_DB*) typu TADDR_Param. Tento datový typ obsahuje 5 proměnných typu USInt, do který se ukládá IP adresa a port (zapisují se decimálně) a 1 proměnnou typu Word, která je vyhrazena pro budoucí použití [1]

TURCV

Instrukce TURCV slouží k příjmu dat, která přišla po již vytvořené komunikaci instrukcí TCON. Tato funkce se vykonává asynchronně. Příjem dat se provede, když je na vstupu EN_R log. „1“. Dokud je povolen příjem pomocí EN_R, tak nelze měnit vstupní/výstupní parametr DATA, aby byla zaručena konzistentnost dat. [1]

Vstupní parametry:

- EN_R – povolení příjmu [1]

Vstupní/výstupní parametry:

- LEN – skrytý parametr, který obsahuje délku dat v bytech, kterou chceme přijmout [1]

Výstupní parametry:

- DATA – ukazatel na data, která se mají odeslat, může se jednat od obraz vstupů, výstupů, paměťové bity, datové bloky (pokud se jedná o struktury, tak musí být identické na vysílači i přijímači) [1]
- RCVD_LEN – skutečná délka přijatých dat v bytech [1]

- ADDR – obsahuje IP adresu a port komunikačního partnera, tyto hodnoty se aktualizují při každém přijetí hodnot, ukládají se do proměnné, která se vytváří stejným způsobem, jaký byl popsán v případě *TUSEND*.

3.4 Formát odesílaných a přijímaných dat

Odesílaná data z MATLAB

Jako nejuniverzálnější formát odesílání více hodnot jsem zvolil formu vektoru, do kterého vložím požadované hodnoty, které chci odeslat do PLC. V mém případě jsem složil vektor z hodnot, které reprezentují žádanou hodnotu teploty a aktuální teplotu na výstupu soustavy. (ty se využívají pro vstup funkčního bloku PID regulátoru na straně PLC), a žádanou hodnotu výšky hladiny a aktuální výšku hladiny. Všechny hodnoty jsou v MATLABu reprezentovány jako double a při přenosu jsou přetypovány viz kapitola 0.

Odesílaná data z PLC

Pro odesílání dat z PLC jsem zvolil pole hodnot *real*. Z PLC posílám hodnoty akčního zásahu PID regulátoru, který reguluje teplotu v tanku, a povely pro ovládání ventilů pro napouštění a vypouštění tanku. Výhoda využití pole oproti cyklickému odesílání jednotlivých hodnot je v tom, že celé pole může být dáno na vstup instrukce *TSEND* najednou a je posláno na jednu vzestupnou hranu na vstupu *REQ*. Navíc lze počet odesílaných hodnot jednoduše změnit změnou velikosti pole.

4. TVORBA VIRTUÁLNÍCH MODELŮ V PROSTŘEDÍ MATLAB/SIMULINK

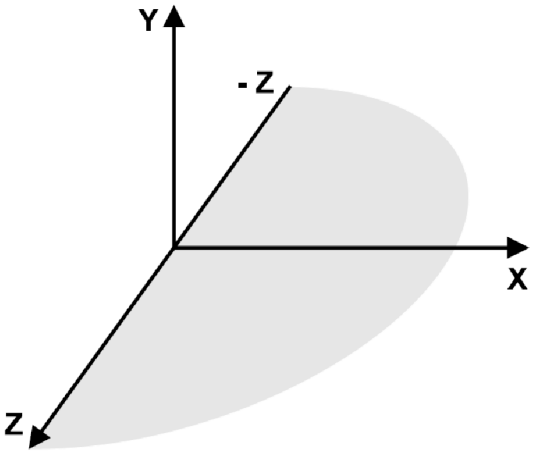
VRML – "Virtual Reality Modeling Language" (Jazyk pro modelování virtuální reality) je jazyk, který jsem využil pro vytvoření 3D modelu rektifikačního tanku. Jako VRML97 (obecně VRML) se označuje mezinárodní norma ISO/IEC 14772-1:1997, která byla přijata v roce 1997. Jedná se o otevřenou a flexibilní platformu pro tvorbu dynamických a interaktivních virtuálních scén. Existuje poměrně velké množství prohlížečů a editorů VRML, které jsou integrovány do webových prohlížečů. Zároveň existují editory, které jsou buď na VRML založeny, nebo umožnují export virtuálních modelů ve formátu vhodném pro následný import do VRML editorů. (např.AutoCAD, SketchUp). [11]

Editorem, ve kterém jsem při vytváření 3D modelu pracoval, byl 3D World Editor, který je implementovaný v programu MATLAB.

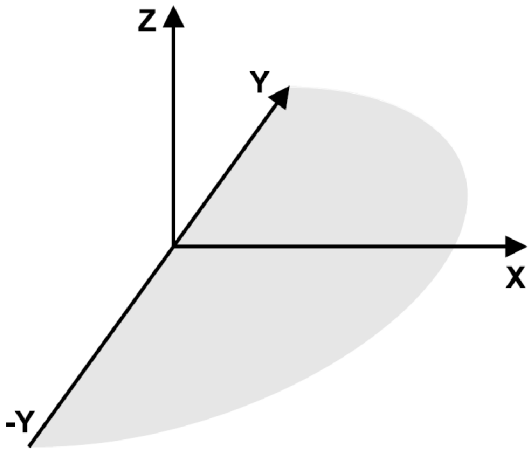
4.1 Souřadný systém

Jako souřadný systém byl pro VRML zvolen pravotočivý Kartézský souřadný systém. Jako mnemotechnická pomůcka ke zorientování se v souřadném systému může posloužit, když palec, ukazováček a prostředníček pravé ruky od sebe roztáhneme tak, aby mezi sebou svíraly pravé úhly. Potom palec představuje kladnou část osy X, ukazováček kladnou část osy Y a prostředníček kladnou část osy Z. Jinými slovy osa +X směřuje doprava, osa +Y nahoru a osa +Z směřuje k pozorovateli, viz Obrázek 4-2. [11]

Tento souřadný systém se od souřadného systému MATLABu liší, u MATLABu osa +Y směřuje od pozorovatele a osa +Z míří nahoru, i přesto, že je to také pravotočivý systém, viz Obrázek 4-2. Kvůli těmto rozdílům je nutné dávat pozor na význam souřadnic v jednotlivých systémech. Transformace spočívá v prohození 2. a 3. souřadnice a změně znaménka u jedné z nich. [11]

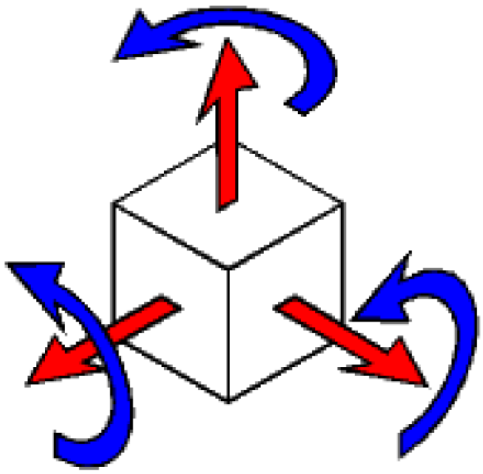


Obrázek 4-2 Souřadný systém VRML [11]



Obrázek 4-2 Souřadný systém MATLAB [11]

Rotace jsou ve VRML definovány notací „osa-úhel“ a tzv. pravidlem pravé ruky, které slouží opět jako mnemotechnická pomůcka – palec pravé ruky držící osu ukazuje směr osy a sevřené prsty ukazují směr otáčení. Kladný směr otáčení je proti směru hodinových ručiček, viz Obrázek 4-3. [11]



Obrázek 4-3 Rotace VRML [11]

Ve VRML jsou všechny rozměry a vzdálenosti definovány v metrech, úhly v radiánech a čas v sekundách. Barvy jsou zapsány v RGB formátu a rozsah hodnot je 0–1. [11]

Pro pozice a rotace potomků určitého objektu platí souřadný systém nadřízeného objektu. [11]

4.2 Popis základních Nodes

Nodes, neboli uzly, mohou popisovat skutečné 3D objekty nebo seskupovat podřízené uzly do jednoho nadřazeného atd. Celá scéna 3D modelu je popsána jednotlivými uzly, které jsou umístěné ve stromové struktuře.

Příklad uzlů:

- *Transform* – jedná se o seskupovací uzel, definuje pozici, měřítko, rotaci a další vlastnosti skupiny podřízených uzlů [11]
- *Shape* – přidá objekt požadovaného geometrického tvaru a vizuálních vlastností, má tyto podřízené uzly: [12]
    - *Geometry* – geometrický tvar (*Box* – krabice, *Cone* – kužel, *Cylinder* – válec, *Sphere* – koule), kde každý tvar má vlastní parametry, jako jsou např. výška, šířka, poloměr, zobrazení stěn atd.
    - *Material* – nastaví jeho vizuální vlastnosti podle hodnot uzlu (např. *diffuseColor* – barva odraženého světla, *emissiveColor* – barva vyzářeného světla, *shininess* – lesklost povrchu, *transparency* – průhlednost)
- *Background* – definuje pozadí světa
- *PointLight* – přidá bodové světlo, které slouží k osvětlení světa
- *ViewPoint* – definuje bod v prostoru, ve kterém pozorovatel světa stojí, směr pohledu, natočení a úhel viditelnosti [12]
- *Avatar* – virtuální pozorovatel světa
- *WordInfo* – obsahuje informace o virtuálním světe jako jeho název, autora, datum vytvoření atd. [12]

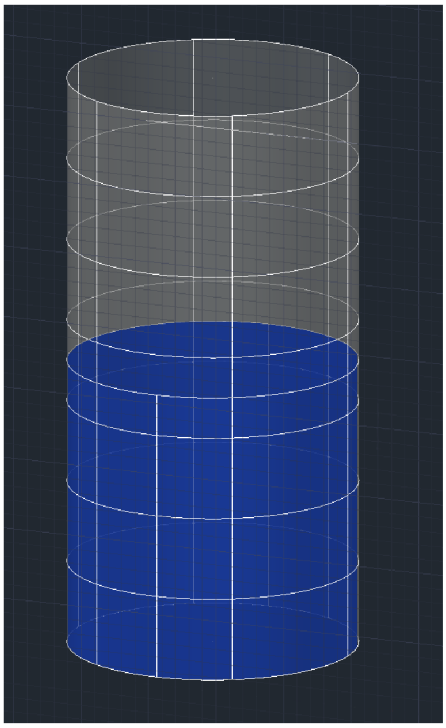
4.3 CAD programy

Pomocí CAD programů lze daleko rychleji tvořit 3D modely, než pomocí uzlů ve VRML modelovacím prostředí. Existuje celá řada programů, které toto umožnují, např. AutoCAD, SolidWorks, Solid Edge a další. [11]

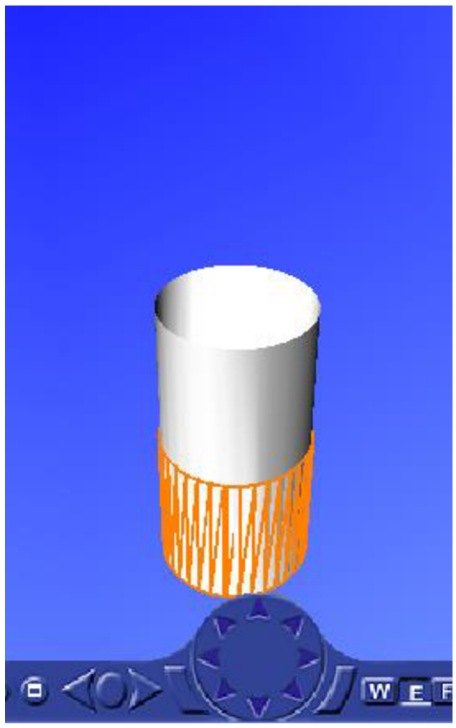
V těchto programech lze vytvořit daleko složitější modely ve vektorové grafice s menším úsilím. Tyto modely lze exportovat do jazyka VRML, ale zpravidla budou obsahovat velké množství vykreslovacích bodů (detailů), které budou mít při vykreslování velké hardwarové požadavky a je možné, že nepůjdou při simulaci vykreslit v reálném čase. [12]

4.3.1 Export z AutoCAD

Export z programu AutoCAD pro vložení do 3D World Editoru v prostředí Simulinku probíhá následujícím způsobem. Nejdříve je potřeba kliknout na logo AutoCADu v levém horním rohu → Export → Other format → formát ACIS *.sat. Vyexportovaný soubor se poté načte z místa uložení do programu CAD Exchanger. Tento program převede soubor z formátu *.sat na *.wrl. Zvolení *.sat formátu při exportu z AutoCADu je důležité, protože program CAD Exchanger neumí pracovat se soubory typu *.dwg, což je výchozí



Obrázek 4-5 Nádrž AutoCAD



Obrázek 4-4 Nádrž 3D World Editor

formát exportu AutoCADu. Nevýhodou tohoto řešení je, že software na převod CAD Exchanger není zdarmaz formátu *.sat na *.wrl. Změna formátu při exportu z AutoCADu je důležitá, protože program CAD Exchanger neumí pracovat se soubory typu *.dwg. Nevýhodou tototo řešení je, že software na převod není zdarma.

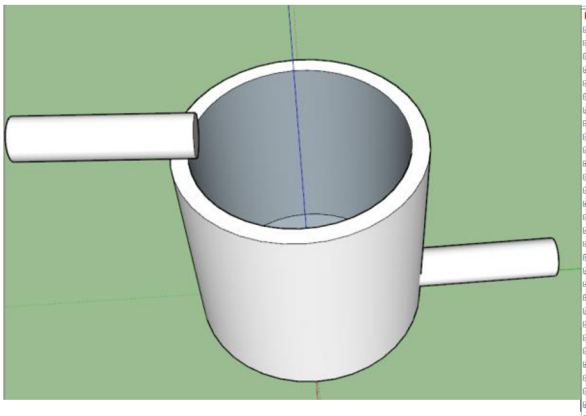
Existují dvě možnosti, jak vložit exportovaný model ve formátu *.wrl do jiného VRML souboru:

- *zkopírování uzlu* Transform*, kde je uložena definice daného objektu, do jiného *.wrl souboru;*
- *využití externího prototypu, který se odkazuje na vyexportovaný objekt.* [12]

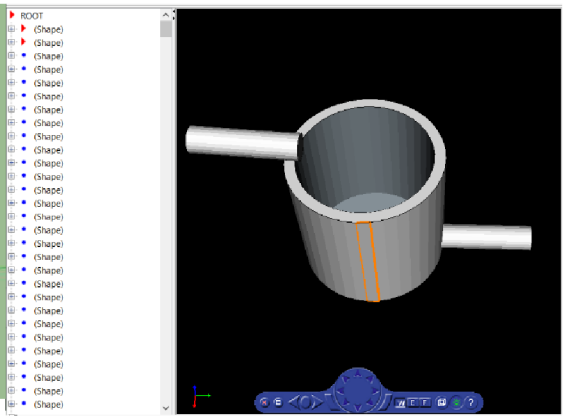
4.3.2 Export ze SketchUp

Export v prostředí SketchUp Pro je o dost jednodušší. V liště nástrojů se zvolí *File→Export→3D Model…*. V novém okně průzkumníku je možné libovolně

pojmenovat soubor a v rolovacím menu je třeba vybrat typ souboru *VRML file* s příponou *.wrl. Velkou výhodou tohoto programu je relativní jednoduchost ovládání, kreslení a pohybu po scéně. Nevýhodou je, že při exportu vznikne velké množství samostatných objektů, protože se oblé tvary skládají z rovin, viz Obrázek 4-7. Pokud je scéna statická, tak to tolik nevadí. Pro zdynamizované objekty bych volil metodu dokreslení těchto objektů přímo ve 3D World Editoru.



Obrázek 4-6 Nádrž v prostředí SketchUp



Obrázek 4-7 Import do 3D World Editoru ze SketchUp

4.3.3 Import do 3D World Editor

Existují dvě možnosti, jak vložit exportovaný model do jiného VRML souboru:

- zkopírování uzlu Transform, kde je uložena definice daného objektu, do jiného *.wrl souboru;
- využití externího prototypu, který se odkazuje na vyexportovaný objekt. [12]

4.4 3D World Editor

3D World Editor slouží k vytvoření a následné editaci virtuálních objektů, k čemuž nabízí velké množství nástrojů a možností. Na rozdíl od výše zmíněných CAD programů a jejich nevýhod v podobě velkého množství samostatných objektů, tento editor umožňuje vytvořit 3D model soustavy reprezentovaný pouze několika objekty, se kterými se následně jednodušeji pracuje. Právě z tohoto důvodu jsem ho použil.

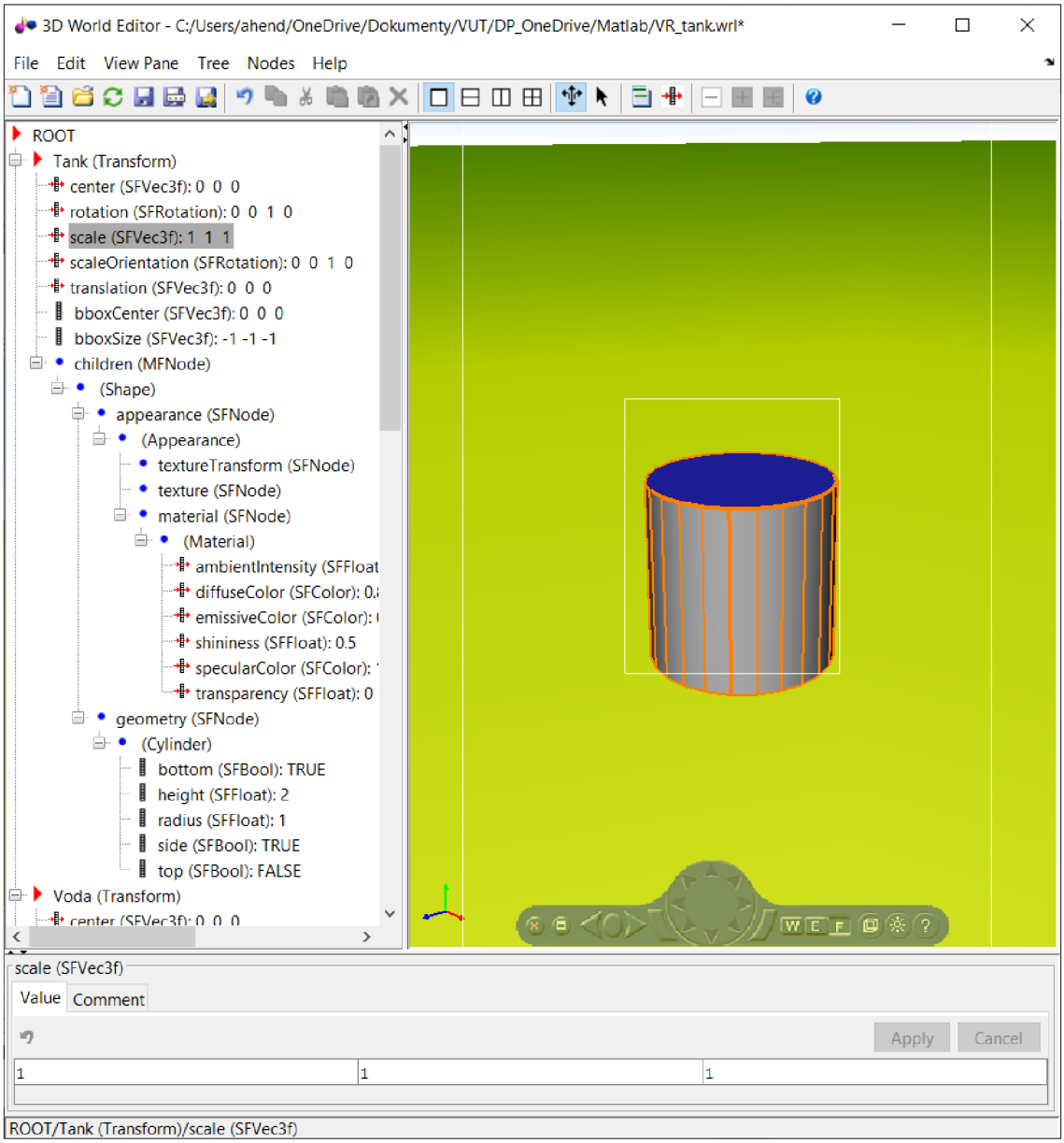
4.4.1 Popis prostředí

Na Obrázek 4-8 je vidět prostředí 3D World Editoru, ve kterém lze vytvářet a upravovat virtuální svět. Editor má čtyři hlavní části. V záhlaví okna se nacházejí záložky a příkazy, kterými lze např. otevřít, vytvořit, uložit model, dále lze také pomocí nabídky *Nodes* přidat další uzly nebo ty stávající připojit k nadřazenému uzlu.

Vlevo uprostřed se nachází stromová struktura virtuálního světa, ve které jsou vypsané všechny použité uzly a jejich parametry.

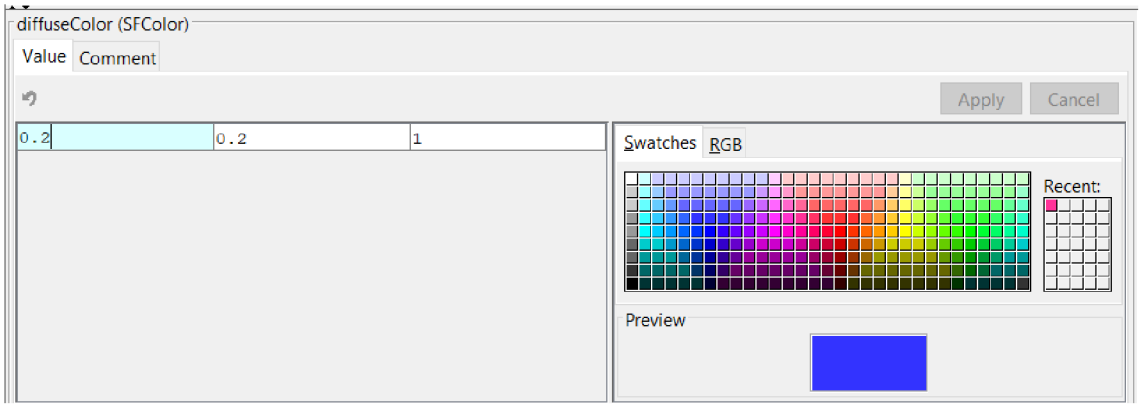
V pravé prostřední části je potom zobrazen vytvořený svět, ve kterém se dá pohybovat pomocí myši a modrého ovládacího panelu, který je umístěn na spodu okna. Na panelu jsou tlačítka pro zavření panelu, pro maximalizaci okna (tato funkce není v 3D World Editoru podporována, je možné ji použít pouze při náhledu modelu), růžici pro pohyb a otáčení se ve světe, výběr módů pohybu, zapnutí/vypnutí drátového drátěného? modelu, vypnutí/zapnutí osvětlení směrem od pozorovatele (čelovky) a tlačítko pro nápovědu.

Ve spodní části se pak nalézají hodnoty uzlu, který je právě vybrán ve stromové struktuře, např. v Obrázek 4-8 jsou vidět hodnoty měřítka velikosti tanku. Pokud bude měřítko 1, tak bude velikost objektu stejná jako je nastavena v uzlu *geometry*.

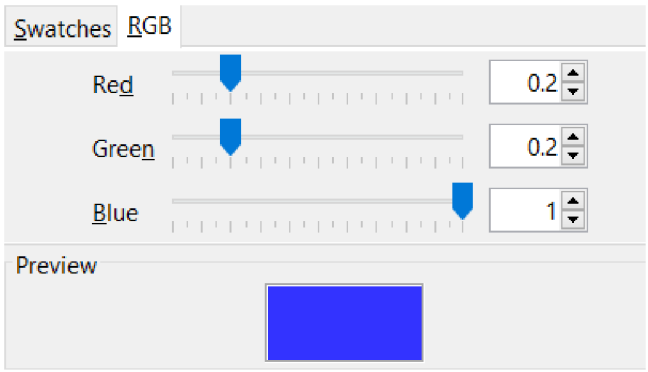


Obrázek 4-8 Prostředí 3D World Editor

Pokud je vybrán uzel určený pro editaci barvy, tak lze zadat hodnotu RGB barvy přímo číslem vyjadřujícím jednu ze složek nebo ji vybrat z barevné palety (viz Obrázek 4-10), lze ji také nastavit pomocí RGB posuvníků (viz Obrázek 4-9). V pravé spodní části je možné vidět náhled navolené barvy.



Obrázek 4-10 Výběr barvy z palety v editoru



Obrázek 4-9 Výběr barvy RGB posuvníky v editoru

4.4.2 Orbisnap

3D World Editor nabízí tři druhy pohybu ve světě pomocí nástroje Orbisnap. Scéna je rozdělena na tři (resp. čtyři) části. Uprostřed je oblast, která je nazývána „navigační zónou“. Po okrajích jsou oblasti, které jsou mimo navigační zónu. Při zapnutém režimu *Fly* se zobrazí ještě jedna zóna, která je „centrální“ – tzn. že je uprostřed. Zóny jsou vidět na Obrázek 4-8 a jsou znázorněny bílou čárou. [13]

Pokud se myš nachází mimo navigační zónu, tak jsou možné tyto pohyby

- *Walk* – při stisknutém tlačítku a pohybu myši nahoru, dolu nebo do stran umožnuje pohyb do těchto směru v jedné rovině [13]
- *Examine* – při stisknutém tlačítku a pohybu myši nahoru a dolu umožnuje pohyb vpřed a vzad. Při pohybu doprava a doleva umožňuje pohyb do stran [13]
- *Fly* – stisknuté tlačítko a pohybu myši nahoru a dolu umožnuje naklonění pohledu doprava nebo doleva [13]

Jestliže je myš umístěna v navigační zóně, tak lze vykonat tyto pohyby

- *Walk* – při stisknutém tlačítku a pohybu myši nahoru nebo dolu umožnuje pohyb dopředu nebo dozadu. Při pohybu doprava a doleva umožňuje pohyb do stran [13]
- *Examine* – stisknuté tlačítko a pohybu myši umožnuje rotaci okolo počátku scény [13]
- *Fly* – zmáčknuté tlačítko a pohybu myši nahoru a dolu umožnuje natočení pohledu nahoru a dolu a pohyb doprava nebo doleva umožnuje pohyb do stran [13]

Pokud se myš nachází ve středu záběru a je zvolen režim *Fly*, tak při pohybu myší nahoru a dolu umožnuje pohyb dopředu a dozadu a při pohybu myši do stran posouvá pohled doprava a doleva. [13]

Některé úkony se dají provádět pomocí klávesnice rychleji. Příklady klávesových zkratek:

- Backspace – vrátit se o krok zpět
- F9 – narovná pohled do vodorovné roviny v aktuálních souřadnicích
- Esc – vrátí pohled do výchozího bodu
- F5 – přepíná mezi normálním a drátovým modelem
- F7 – vypíná a zapíná navigační zóny
- Shift+W – nastaví mód *Walk*
- Shift+E – nastaví mód *Examine*
- Shift+F – nastaví mód *Fly* [13]

4.5 3D ANIMATION SIMULINK

Pro začlenění a propojení virtuálního světa do prostředí Simulink se využívají následující bloky.

4.5.1 VR Sink

Tento blok slouží k předávání signálu ze Simulinku do virtuálního modelu a jeho ovládání. *VR Sink* přepisuje hodnoty z svýchportů (vstupů) do jednotlivých parametrů uzlů. Pro vybrání parametrů je třeba vložit blok do pracovního prostoru Simulink, dvakrát na něj kliknout, vedle adresního řádku v sekci *Source file* vybrat požadovaný 3D model a načíst ho (pokud již není načten; pro vytvoření nového virtuálního světa slouží tlačítko *New*) a v pravé části *Virtual World Tree* vybrat požadovaný parametr. Pokud je parametr skutečně vybrán, tak se vedle názvu „zaškrtne políčko“. Lze vybrat všechny uzly, u kterých není černý křížek. Pokud u sebe mají křížek, tak jejich „rodičovské“ uzly nemají jména nebo nejsou datovou třídou. Pro každý vybraný parametr se zobrazí vstupní porty. V *Block properties* se nastavuje *Sample time*, tj. čas vzorkování. [12]

4.5.2 VR Source

VR Source se využívá pro předávání dat z prostředí virtuálního světa do Simulinku. Může se například jednat o zásahy uživatele. Pokud jsou použity proximitní senzory a pozorovatel se dostane do oblasti, která tyto senzory zaktivuje, tak lze tato data vyčíst tímto blokem. Tím lze dosáhnout vyšší interaktivity. Uzly, které mají u sebe červenou šipku, lze rozbalit a vybrat některé parametry a využít je v Simulinku. Výběr se provede kliknutím na příslušný parametr v oblasti *Virtual Tree* a u názvu se objeví „zaškrtlé políčko". Pro každý vybraný parametr se zobrazí výstupní porty. V *Block properties* se nastavuje *Sample time*, tj. čas vzorkování. [12]

4.5.3 VR Placeholder

Na výstupu tohoto bloku je hodnota, která je interpretována jako „nespecifikovaná". Tato hodnota se přivádí na vstup bloku *VR Sink*. Slouží ke změně pouze některých složek vektoru. Hodnoty, na které je přivedena hodnota z *VR Placehoderu* zůstanou nezměněny. To se využívá například pokud je požadavek na posuv pouze v jednom směru. Potom bude na vstup *translate* bloku *VR Sink* přiveden vektor, který bude obsahovat dvě „nespecifikované" hodnoty a jednu hodnotu, o kterou se má objekt posunout. [12]

4.5.4 VR Signal Expander

Tento blok složí vektor o požadované rozměru pomocí vstupních hodnot a „nespecifikovaných hodnot" jako v případě *VR Placeholder.*

Názorný příklad: Pokud by byl potřeba vektor o rozměrech 6x1 a proměnné hodnoty mají být na prvním, třetím a pátém místě, tak se do parametru *Output width* zadá *6* a do *Output signal indices [1,3,5].* První parametr určuje délku výstupního vektoru a druhý pozice, na které se mají přivést vstupní signály. Zbylé hodnoty budou „nespecifikované". [12]

4.5.5 Real Time Synchronization

Real time synchronization, neboli Synchronizace s reálným časem slouží ke „srovnání" hodin 3D modelu a simulace, aby bylo možné simulovat děj i s připojeným vzdáleným zařízením. [12]

4.5.6 VR Text Output

VR Text Output slouží pro vložení textové nebo číselné proměnné do 3D světa. V nastavení tohoto bloku se specifikuje, do kterého virtuálního světa se má proměnná vložit, se kterým textovým uzlem se má propojit a zadá se textový řetězec s proměnnou, která se má zobrazit. Formát řetězce je stejný jako u *sprintf*. Například pro proměnnou typu double s rozlišením na jedno desetinné místo je zápis takovýto: *%0.1f*. Při spuštění simulace a otevření virtuálního světa se zobrazí aktuální hodnota, která je přivedena na vstup tohoto bloku.

5. REALIZACE MODELU ČÁSTI VÝROBNÍHO PROCESU

Výrobní proces, který jsem si vybral pro vytvoření modelu, je ohřev lihu v rektifikačním kotli. Tento kotel se v případě dvou-kotlového systému výroby lihovin nachází za destilačním (surovinovým) kotlem (v případě dvoukotlového systému) a slouží k druhé destilaci lutru, který vznikne první destilací v kotli destilačním. Lutr je líh, který vznikne jedním vypálením kvasu a obsahuje 20–30 % obj. ethanolu. Z celkového množství kvasu se získá ¼–⅓ lutru. Získaný lutr obsahuje nežádoucí vedlejší látky (složitější alkoholy, estery, aromatické látky apod.), proto není vhodný ke konzumaci a musí se zušlechtit druhou destilací – rektifikací. [14]

5.1 Rektifikace

Cílem rektifikace je zesílení lutru na požadovanou hodnotu lihovitosti a vyčištění od nežádoucích látek. Doba rektifikace je asi 2,5 až 3 hodiny, v závislosti na objemu lutru. Na rozdíl od prvního pálení zde dochází k oddělování jednotlivých frakcí – jedná se o frakční destilaci. Obecně existují tři frakce: úkap, prokap a dokap. Aby došlo ke správnému oddělení jednotlivých látek, je zapotřebí teplotu zvyšovat pozvolna. Při rychlém ohřívání by totiž mohlo dojít ke znehodnocení destilátu. [14]

5.1.1 Úkap

První frakce, která obsahuje látky s nejnižším bodem varu (aldehydy, estery a část podílu metanolu), se nazývá úkap. Úkapu bývá okolo 2 % objemu rektifikovaného destilátu a obsahuje až 80 % obj. alkoholu. Konec úkapu a začátek prokapu se určuje senzoricky (úkap má ostrou, pichlavou vůni a palčivou chuť) a podle zkušeností destilatéra. Při rychlém zvyšování teploty roste podíl úkapu a tím se snižuje objem žádoucích aromatických látek, které budou scházet v prokapu. Jednoduché pravidlo je, že se ze 100 litrů lutru vylije prvních 500 ml destilátu. [14]

5.1.2 Prokap

Druhá frakce, tedy prokap, se taktéž se nazývá jádro. Jedná se o destilát, který je určen ke konzumaci a zachytává se do samostatné, čisté nádoby. Tato nádoba nesmí ovlivnit chuť destilátu, proto se používají skleněné nebo ocelové nádoby. [14]

Množství prokapu závisí na kvalitě kvasu, čím vyšší kvalita kvasu, tím vyšší množství prokapu a opačně. Na začátku prokapu je koncentrace etanolu v rozmezí okolo 70–75 % obj. V prokapu se nachází žádoucí senzorické látky, které dávají destilátu výslednou chuť a vůni, proto je důležitá pomalá destilace. [14]

Konec této fáze se určuje senzorickou zkouškou, kterou provádí destilatér. Pomocným vodítkem je koncentrace právě zkapalněných lihových par. Při poklesu koncentrace ke 45 % obj. se fáze prokapu ukončuje, protože se začínají objevovat vyšší alkoholy, které by znehodnotily jakost destilátu. [14]

Celková koncentrace prokapu po ukončení této fáze je okolo 60–70 % obj. etanolu. Jádro činí cca 30 % objemu destilovaného lutru. [14]

5.1.3 Dokap

Začátek třetí frakce, tedy dokapu, je při lihovitosti okolo 45 % obj. etanolu. Tato frakce obsahuje velké množství přiboudliny (směs vyšších alkoholů, vody a mastných kyselin), menší množství esterů a silic. Dokap se jímá do koncentrace 4–5 % obj. etanolu právě jímaného destilátu. Opět se konec frakce hlídá senzoricky, při prvním náznaku kyselosti již není vhodná k použití. Dokap se nesmí dostat do jádra, protože by ho znehodnotil. [14] [15] [16]

Stejně jako prokap, tak i dokap tvoří 25–30 % objemu z destilovaného lutru. Obsah etanolu v dokapu je cca 20 % obj. Dokap spolu s úkapem jsou odpadní frakce. [14] [15] [16]

Matematický popis tanku

V následujících podkapitolách jsou popsány postupy, pomocí nichž jsem došel k matematickým popisům výšky hladiny a teploty lutru v kotli.

5.2 Model výšky hladiny

Pro vytvářený model uvažuji nádrž, která se skládá ze dvou soustředných válců. V prostoru mezi plášti válců je voda, která slouží k ohřevu lutru v prostoru vnitřního válce. Tento způsob ohřevu se používá, protože omezuje připalování. Vnitřní válec má rozměry 0,51 x 0,25 m (výška x poloměr). Z obecně známého vzorce pro výpočet objemu vychází, že tato nádrž má objem $V_1 \doteq 100{,}14\,l$. Plocha nádrže je $S_1 \doteq 0.20\,m^2$. Rozměry vnějšího válce jsou 0,53 x 0,29 m (výška x poloměr). Objem meziprostoru je $V_2 \doteq V_{2Celkove} - V_1 = 140{,}03 - 100{,}14 \doteq 39{,}89\,l$. Oba údaje jsou zaokrouhleny na dvě desetinná místa.

Jedná se o nelineární systém s jedním akumulátorem, a tudíž jde o systém prvního řádu. Nelinearitou tohoto systému je největší možný objem kapaliny v nádrži, z toho vyplývá, že se jedná o nelinearitu typu omezení. Nicméně toto omezení zanedbám, protože se budu s výškou hladiny pohybovat pouze v lineární části. Dále také předpokládám, že se jedná o pravidelnou válcovou nádobu r rovným dnem.

Budu předpokládat, že tato nádrž bude napouštěna jedním přítokem a vypouštěna jedním odtokem. O napouštění se stará samonasávací čerpadlo SVA s mechanickou ucpávkou pro čerpání hořlavin, které má maximální průtok 2,3 l/s (takovéto čerpadlo

vyrábí např. firma SIGMA PUMPY HRANICE. Technické parametry jejich výrobku jsem použil jako referenční [17]). Spouštění čerpadla je řešeno pomocí ovládaného relé. Odtok je řešen samospádem a je ovládán ventilem typu otevřeno/zavřeno. Princip napouštění a vypouštění tohoto tanku reprezentuje Obrázek 5-1,

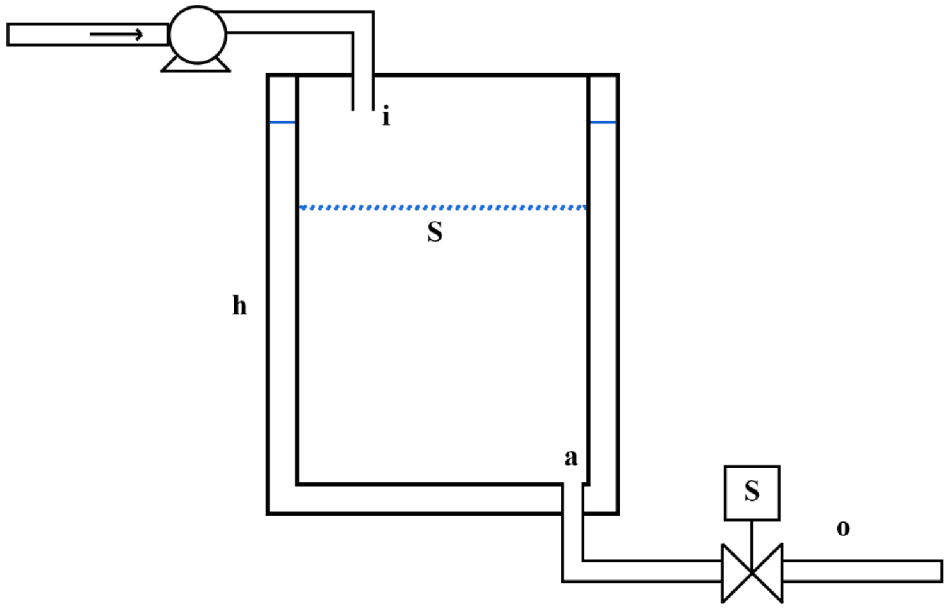
kde i je objemový průtok přítoku $[m^3/s]$,

S je plocha hladiny $[m^2]$

h je výška hladiny $[m]$

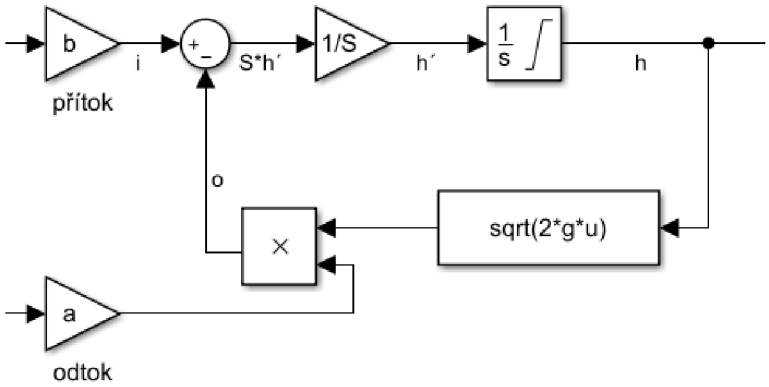
a je průřez vypouštěcího potrubí $[m^2]$

o je objemový průtok výpustě $[m^3/s]$



Obrázek 5-1 Principiální nákres modelu výšky hladiny

Vstupem je objem lutru přitékající do systému i od kterého je odečten objem odtékajícího lutru o a výstupem je výška hladiny h. Množství lutru, které může odtéct pryč, pokud je otevřený ventil. je dáno výškou hladiny.



Obrázek 5-2 Blokové schéma modelu výšky hladiny

Obrázek 5-2 představuje blokové schéma modelu výšky hladiny, kde b představuje objemový průtok čerpadla $[m^3/s]$, které je připojeno na vstup modelu.
Bilanční rovnice tohoto systému je [18]:

$$\frac{dV}{dt} = S\frac{dh}{dt} = i(t)dt - o(t)dt \quad (5.1)$$

Kde o lze vyjádřit jako [12]:

$$o(t) = a\sqrt{2 \cdot g \cdot h(t)} \quad (5.2)$$

g je tíhové zrychlení $[m/s^2]$
Po úpravě je diferenciální rovnice systému [18]:

$$S\frac{dh}{dt} + a\sqrt{2 \cdot g \cdot h(t)} = i(t)dt \quad (5.3)$$

S počáteční podmínkou $h(0) = 0$ (5.4)

5.2.1 Výsledný model výšky hladiny

Dosazením do rovnice (6.3) dostanu diferenciální rovnici pro výšku hladiny, kterou jsem naimplementoval do prostředí Simulink, jak je vidět v Obrázek 5-2.

$$0{,}2\frac{dh}{dt} + a\sqrt{2 \cdot 9{,}81 \cdot h(t)} = b \cdot i(t)dt, \quad (5.5)$$

kde konstanty a a b závisí na tom, v jakém stavu je čerpadlo, resp. ventil.

5.3 Teplotní model

Pro ohřev kapaliny uvažuji elektrické topné těleso o výkonu P=15 kW. U tohoto tělesa pro další výpočty zanedbám jeho vlastní hmotnost a specifické teplo, a tím jeho dynamiku. Toto zanedbání provedu za předpokladu, že časová konstanta vody je mnohonásobně větší než časová konstanta topné spirály. Dále budu uvažovat dvouplášťovou nádrž, kde bude topné těleso umístěno mezi stěnami a meziprostor bude vyplněn vodou, která bude tento prostor celý vyhřívat. Tím se zlepší přenos tepla mezi topným tělesem a ohřívanou kapalinou uvnitř tanku. Budu uvažovat, že vnitřní stěna tanku bude mít mnohonásobně kratší časovou konstantu, než je dynamika ohřívací a ohřívané kapaliny, proto tuto dynamiku zanedbám. Rovněž zanedbám fakt, že se rektifikace provádí za účelem odpaření částí lutru a tím dochází ke změně objemu, resp. hmotnosti lutru v nádrži, ze které se počítá přenos soustavy.

Základní zjednodušený model tepelné soustavy za předpokladu, že t_1 je rozdíl teploty vody a teploty okolí a t_2 je rozdíl teploty lihu a vody, je [19]:

$$m_v c_v t'_v = P - K_{10}t_1 - K_{12}(t_1 - t_2) \quad (5.6)$$

$$m_l c_l t'_l = K_{12}(t_1 - t_2) \quad (5.7)$$

kde m_v je hmotnost vody $[kg]$

- c_v je specifické teplo vody $[Ws/(kg°C)]$
- t'_v je změna teploty vody $[°C]$
- m_c je hmotnost lihu $[kg]$

- c_l je specifické teplo lihu $[Ws/(kg°C)]$
- t'_l je změna teploty lihu $[°C]$
- P je elektrický příkon ohřívače $[W]$
- K_{10} je koeficient přestupu tepla mezi vodou a okolím $[W/°C]$
- K_{12} je koeficient přestupu tepla mezi vodou a lihem $[W/°C]$
- t_1 je rozdíl teploty vody a okolí $[°C]$
- t_2 je rozdíl teploty vody a lihu $[°C]$

Maticový zápis těchto rovnic je [19]:

$$\begin{bmatrix} t'_v \\ t'_l \end{bmatrix} = \begin{bmatrix} \frac{-K_{10}-K_{12}}{m_v c_v} & \frac{K_{12}}{m_v c_v} \\ \frac{K_{12}}{m_l c_l} & \frac{-K_{12}}{m_l c_l} \end{bmatrix} \cdot \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} + \begin{bmatrix} \frac{1}{m_v c_v} \\ 0 \end{bmatrix} \cdot P \tag{5.8}$$

$$y = \begin{bmatrix} 0 & 1 \end{bmatrix} \cdot t_2 \tag{5.9}$$

Úprava maticového zápisu substitucí [19]:

$$\begin{bmatrix} t'_v \\ t'_l \end{bmatrix} = A \cdot \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} + B \cdot P \tag{5.10}$$

$$y = C \cdot \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} \tag{5.11}$$

Výpočet přenosu soustavy [19]:

$$F_s(p) = C \cdot (pI - A)^{-1} \cdot B \tag{5.12}$$

Obecný výsledný přenos soustavy

$$F_s(p) = \frac{m_0}{n_2 p^2 + n_1 p + n_0} \tag{5.13}$$

5.3.1 Výsledný model teploty lutru

Hmotnosti vychází z fyzických rozměrů rektifikačního tanku, které byly zmíněny v podkapitole 5.2. Specifické teplo *c* je materiálová konstanta obou kapalin. Koeficienty přestupu tepla se většinou určují experimentálně, proto jsem je zvolit tak, aby se odezva teplené soustavy co nejvíce blížila realitě.

$m_v \doteq 39{,}77\ kg$
$c_v = 4\,180\ Ws/(kg°C)$
$m_l \doteq 79{,}01\ kg$
$c_l = 2\,430\ Ws/(kg°C)$
$P = 15\,000\ W$

$K_{10} = 0{,}4\ W/°C$

$K_{12} = 35\ W/°C$

Dosazením těchto hodnot rovnic (5.6) a (5.7) vyjdou následující matice:

$$A = \begin{bmatrix} -2{,}13 \cdot 10^{-4} & 2{,}11 \cdot 10^{-4} \\ 1{,}82 \cdot 10^{-4} & -1{,}82 \cdot 10^{-4} \end{bmatrix} \quad B = \begin{bmatrix} 6{,}01 \cdot 10^{-6} \\ 0 \end{bmatrix} \tag{5.14}$$

$$C = \begin{bmatrix} 0 & 1 \end{bmatrix} \tag{5.15}$$

Po dosazení těchto matic do rovnice (5.12) lze vyčíslit a spočítat přenos tepelné soustavy.

$$F_s(p) = \begin{bmatrix} 0 & 1 \end{bmatrix} \cdot \left(\begin{bmatrix} p & 0 \\ 0 & p \end{bmatrix} - \begin{bmatrix} -2{,}13 \cdot 10^{-4} & 2{,}11 \cdot 10^{-4} \\ 1{,}82 \cdot 10^{-4} & -1{,}82 \cdot 10^{-4} \end{bmatrix} \right)^{-1} \cdot \begin{bmatrix} 6{,}01 \cdot 10^{-6} \\ 0 \end{bmatrix} \tag{5.16}$$

$$(pI - A)^{-1} = \frac{1}{\det(pI - A)} \cdot adj(pI - A) =$$

$$\begin{bmatrix} \frac{p + 1{,}82 \cdot 10^{-4}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} & \frac{2{,}11 \cdot 10^{-4}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} \\ \frac{1{,}82 \cdot 10^{-4}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} & \frac{p + 2{,}13 \cdot 10^{-4}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} \end{bmatrix} \tag{5.17}$$

$$(pI - A)^{-1} \cdot B = \begin{bmatrix} \frac{p + 1{,}82 \cdot 10^{-4}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} & \frac{2{,}11 \cdot 10^{-4}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} \\ \frac{1{,}82 \cdot 10^{-4}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} & \frac{p + 2{,}13 \cdot 10^{-4}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} \end{bmatrix} \cdot \begin{bmatrix} 6{,}01 \cdot 10^{-6} \\ 0 \end{bmatrix} =$$

$$\begin{bmatrix} \frac{6{,}02 \cdot 10^{-4}\, p + 1.10 \cdot 10^{-9}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} \\ \frac{1.10 \cdot 10^{-9}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} \end{bmatrix} \tag{5.18}$$

$$C \cdot (pI - A)^{-1} \cdot B = \begin{bmatrix} 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \frac{6{,}02 \cdot 10^{-4}\, p + 1.10 \cdot 10^{-9}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} \\ \frac{1.10 \cdot 10^{-9}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} \end{bmatrix} = \frac{1.10 \cdot 10^{-9}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}} \tag{5.19}$$

Výsledný přenos tepelné soustavy:

$$\boldsymbol{F_s(p) = \frac{1.10 \cdot 10^{-9}}{p^2 + 3{,}95 \cdot 10^{-4} p + 4{,}39 \cdot 10^{-10}}} \tag{5.20}$$

Časové konstanty tepelné soustavy tohoto systému jsou přibližně 2537 a 898 548 sekund. Pokud přivedu jednotkový skok o amplitudě elektrického příkonu topidla, tj. 15 000 W, tak se lutr ohřeje na 87 °C za necelých 69 minut. Taková teplota je potřeba k destilaci zhruba 45% lihu [20]. Pro možnost demonstrace funkčnosti a testování jsem upravil hodnoty koeficientů přestupu tepla a množství objemu vody a lutru, tak aby soustava měla co nejkratší časové konstanty a zároveň byla nejkratší časová konstanta větší než 0,2 sekundy, aby byl splněn vzorkovací teorém (periodu vzorkování jsem zvolil 0,1 sekund). Koeficienty přestupu tepla jsem zvýšil, objemy naopak snížil (čímž se snížila

i hmotnost). Takovéto teoretické hodnoty umožnují zachovat princip chování soustavy, ale zkrátí dobu potřebnou pro simulaci v reálném čase při řízení pomocí PID regulátoru v PLC.

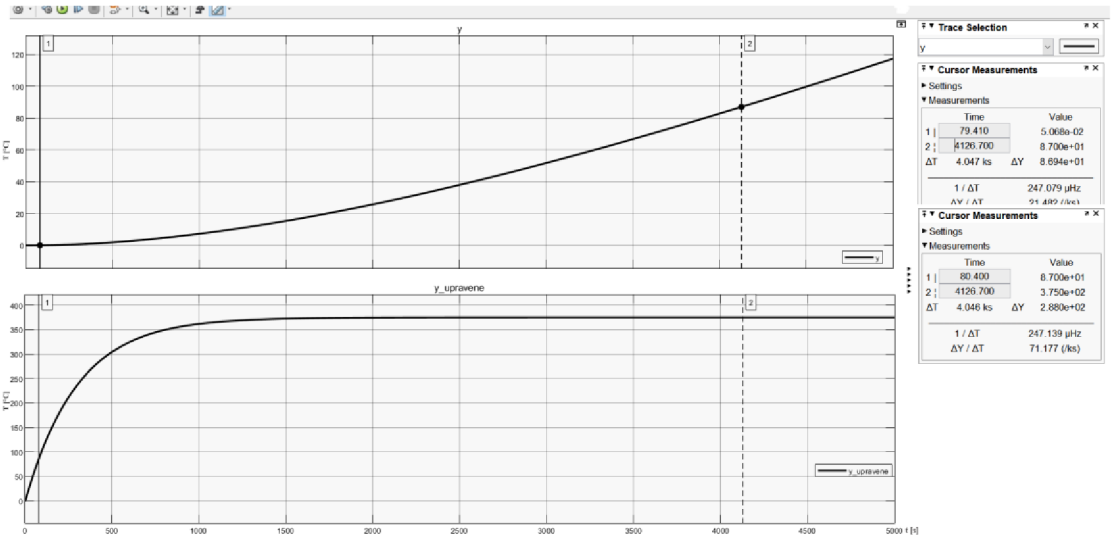
Upravené hodnoty pro soustavu s kratšími časovými konstantami:

$$m_v = 1{,}33\ kg$$
$$c_v = 4\,180\ Ws/(kg°C)$$
$$m_l \doteq 2{,}63\ kg$$
$$c_l = 2\,430\ Ws/(kg°C)$$
$$P = 15\,000\ W$$
$$K_{10} = 40\ W/°C$$
$$K_{12} = 3\,500\ W/°C$$

Přenos soustavy s kratšími časovými konstantami:

$$\boldsymbol{F_s}(p) = \frac{9{,}87 \cdot 10^{-5}}{p^2 + 1.19p + 3{,}95 \cdot 10^{-3}} \tag{5.21}$$

Po úpravě hodnot a přepočítání rovnic (5.6) a (5.7) vycházejí časové konstanty systému na 0,85 a 299,52 sekund. Při porovnání odezvy na stejný vstupní signál jako v předchozím případě se doba, za kterou se lutr ohřeje, zkrátila na necelých 80 sekund, viz Obrázek 5-3 Přechodové charakteristiky systémů. Takovýto čas je už vhodný pro



Obrázek 5-3 Přechodové charakteristiky systémů

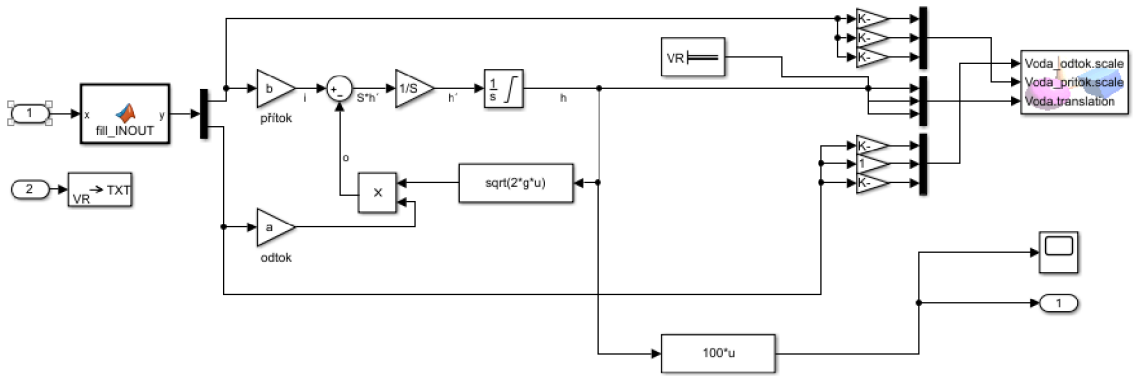
provádění simulace experimentování v laboratoři, jelikož není nutné čekat v řádech hodin, ale pouze parametrů PID regulátoru teploty v PLC.

6. PROPOJENÍ 3D MODELU S MATEMATICKÝM MODELEM

Zjednodušený 3D model rektifikačního tanku jsem vytvořil pomocí WRML modelovacího jazyka a 3D World Editoru, jak bylo popsáno v kapitole 4.

6.1.1 Propojení modelu výšky hladiny se 3D modelem

Propojení modelu výšky hladiny se 3D modelem v prostředí Simulink jsem udělal v subsystému pro lepší přehlednost, do kterého vstupují hodnoty povelů pro zapnutí nebo vypnutí čerpadla, otevření nebo zavření ventilu a aktuální teplota lutru. Po vypočítání aktuální výšky hladiny (podle postupu v podkapitole 5.2) se tato hodnota předá na vstup *Voda.translation* bloku *VR* SINK. Zbylé dva vstupy tohoto bloku složí pro vizualizaci napouštění a vypouštění tanku. Výstupem tohoto subsystému je aktuální výška hladiny v procentech.



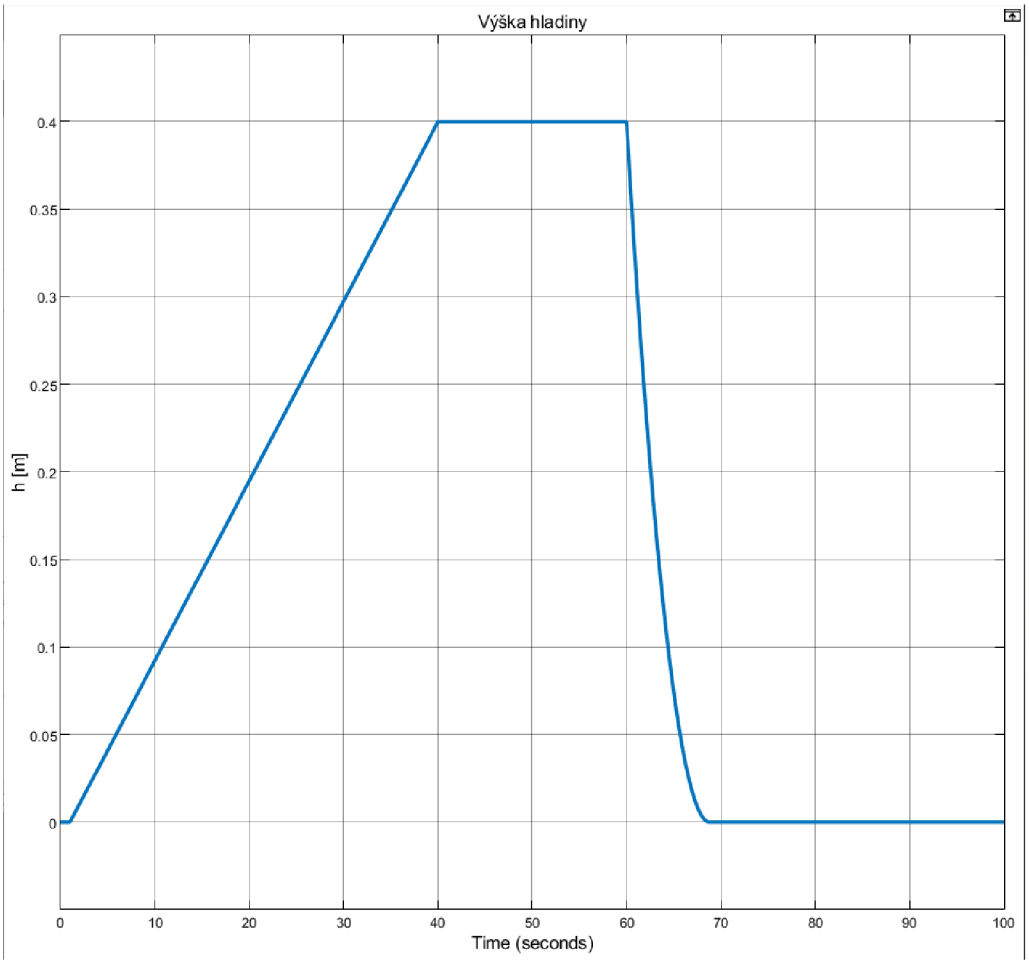
Obrázek 6-1 Propojení modelu výšky hladiny s 3D modelem

6.1.2 Propojení teplotního modelu výšky hladiny se 3D modelem

Předávání a zobrazení aktuální teploty do prostředí 3D modelu probíhá pomocí bloku *VR Text Output,* podle postupu popsaného v podkapitole 4.5.6, který je umístěný v subsystému pro výpočet výšky hladiny, viz Obrázek 6-1, a pomocí textového uzlu ve 3D World Editoru. Do tohoto bloku přivádím aktuální teplotu, která je na výstupu tepelné soustavy.

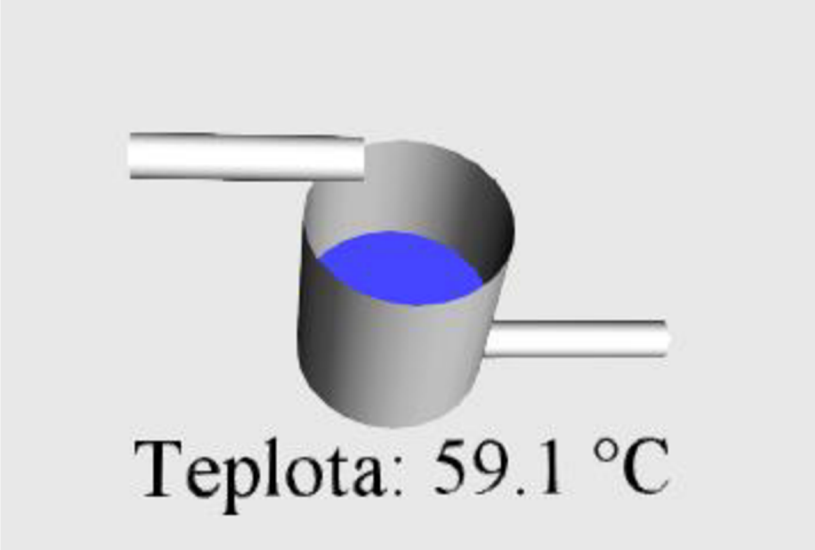
6.1.3 Otestování funkčnosti

3D model jsem otestoval tak, že jsem si vytvořil pomocný simulační obvod, který měl za úkol nasimulovat kladný jednotkový skok v čase jedna sekunda, který představoval spuštění čerpadla a záporný jednotkový skok v čase 40 sekund, který představoval vypnutí čerpadla. V čase 60 sekund jsem přivedl kladný jednotkový skok na „ovládání“ výpustného ventilu pro vypuštění tanku. Zaznamenaný průběh lze vidět na Obrázek 6-2.



Obrázek 6-2 Průběh výšky hladiny při testování

Na Obrázek 6-3 je vidět, že hladina ve 3D vizualizaci stoupla společně s výškou hladiny v simulaci. Pro otestování zobrazení teploty jsem použil pouze konstantu přivedenou na vstup bloku *VR Text Output*.



Obrázek 6-3 Testování 3D modelu

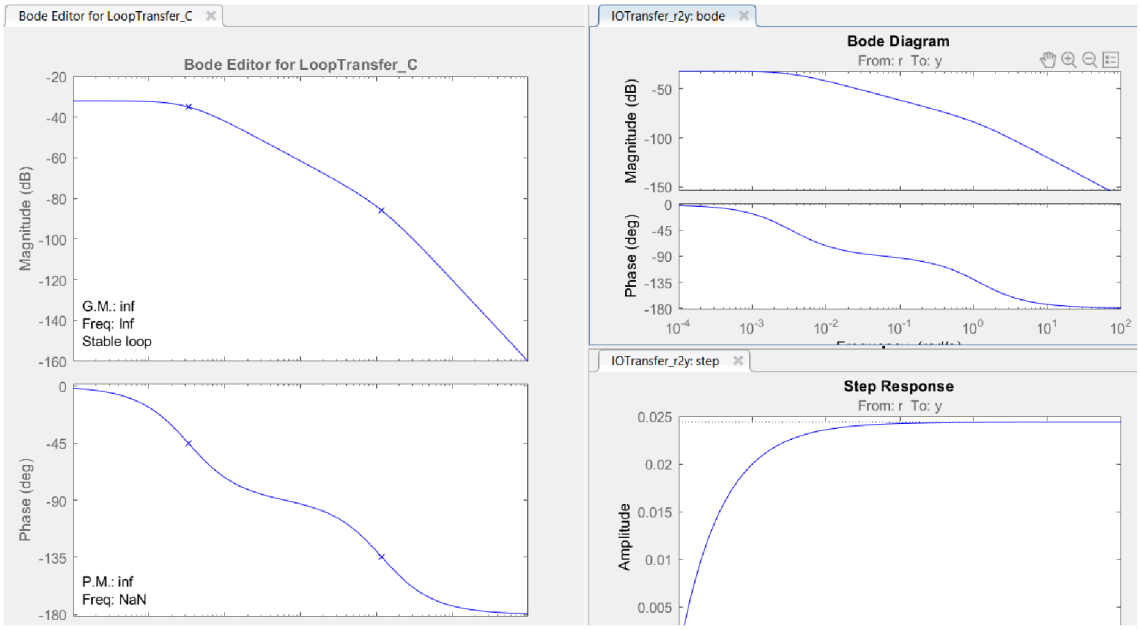
6.2 Návrh regulátoru teploty

Návrh regulátoru, který řídí teplotu lutru v kotli, jsem prováděl pro soustavu s přenosem

$$F_s(p) = \frac{9{,}87 \cdot 10^{-5}}{p^2 + 1.19p + 3{,}95 \cdot 10^{-3}},$$

tedy pro soustavu s kratšími časovými konstantami.

Před samotným návrhem jsem si stanovil, že maximální překmit aktuální teploty nad žádanou hodnotu může být maximálně 1 °C. Dále mám požadavky na to, aby požadovaná teplota byla nejdříve překročena za 60 sekund a také chci, aby se aktuální teplota ustálila na žádané hodnotě do 80 sekund. Využil jsem metody návrhu pomocí standardního tvaru

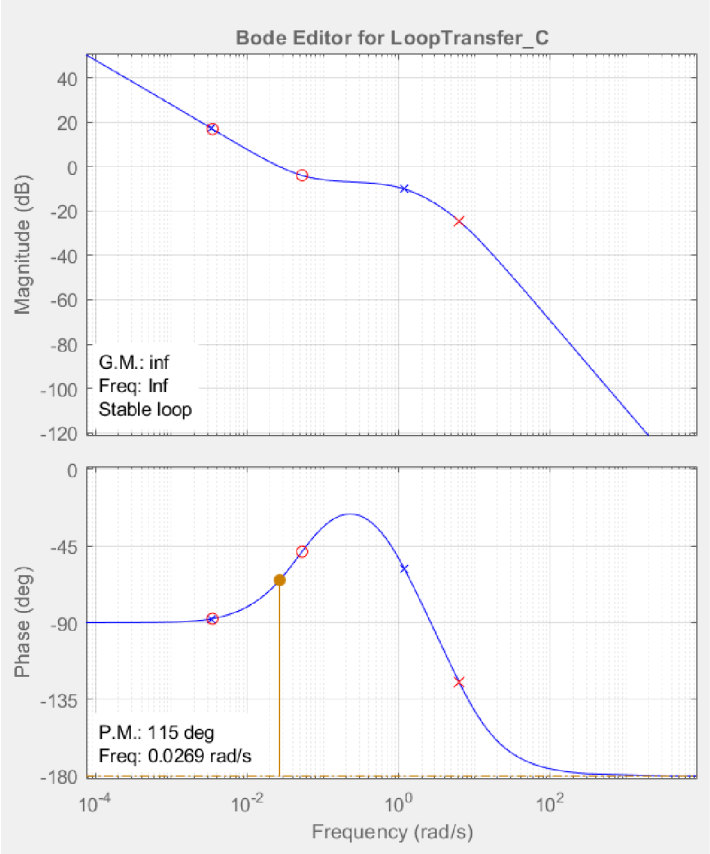


Obrázek 6-4 Frekvenční odezvy a přechodová charakteristika tepelné soustavy

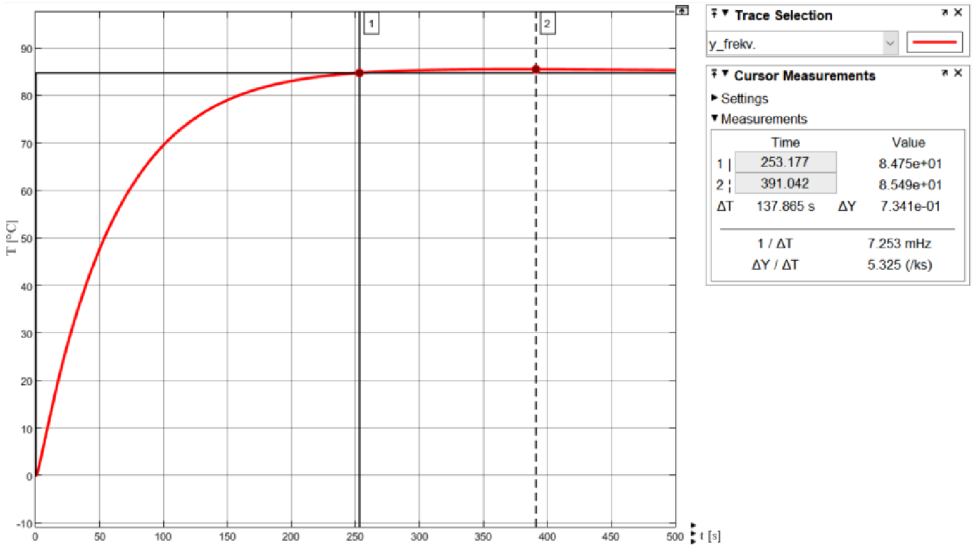
frekvenční charakteristiky otevřeného obvodu. Nejdříve jsem do nástroje sisotool, ve kterém lze navrhnout regulační obvod několika metodami, nahrál přenos soustavy. Pro úpravu frekvenčních charakteristik slouží bode editor. Prvním krokem bylo vložení přenosu soustavy. To se provádí v záložce Edit architecture → G, kde se vybere příslušný přenos z Base workspace. Po načtení a vykreslení frekvenční odezvy systému je vidět, že soustava obsahuje dva póly, což odpovídá. Jelikož návrh regulátoru je prováděn úpravou tvaru frekvenční charakteristiky otevřeného obvodu, což ovlivňuje frekvenční odezvu uzavřeného obvodu, tak jsem si zobrazil i bode diagram pro uzavřený obvod. Ten se přidá v záložce New plot → New Bode → IOTransfer_r2y, což je označení přenosu žádané hodnoty na výstup. Ve třetím grafu (přechodová charakteristika) je vidět za jak dlouho, případně jestli dosáhne systém požadované hodnoty. Jak je v tomto grafu vidět, tak samotná soustava dosáhne pouze 2,5 % žádané hodnoty za velmi dlouhý čas.

Regulátor jsem naladil s ohledem na kompromis mezi překmitem nad žádanou teplotu a rychlostí regulace. Parametry PID regulátoru jsem nastavil tak, že přenos regulátoru je $F_R(p) = 1 \cdot \frac{(19p+1)\cdot(2{,}9\cdot 10^2 p+1)}{p(0{,}42p+1)}$.

Nuly regulátoru jsem umístil na frekvence $5{,}31 \cdot 10^{-2}\, rad/s$ a $3{,}49 \cdot 10^{-3}\, rad/s$. Abych mohl vytvořit reálný PID regulátor a odsimulovat ho, tak jsem musel přidat realizační konstantu. Tento pól jsem umístil na frekvenci 6,24 rad/s, tj. cca dvě dekády od pravé nuly. Jak je vidět z Obrázek 6-5, amplitudová bezpečnost je nekonečno a fázová bezpečnost je 115 °.



Obrázek 6-5 Frekvenční charakteristika otevřeného obvodu



Obrázek 6-6 Simulace řízení teploty pomocí PID regulátoru

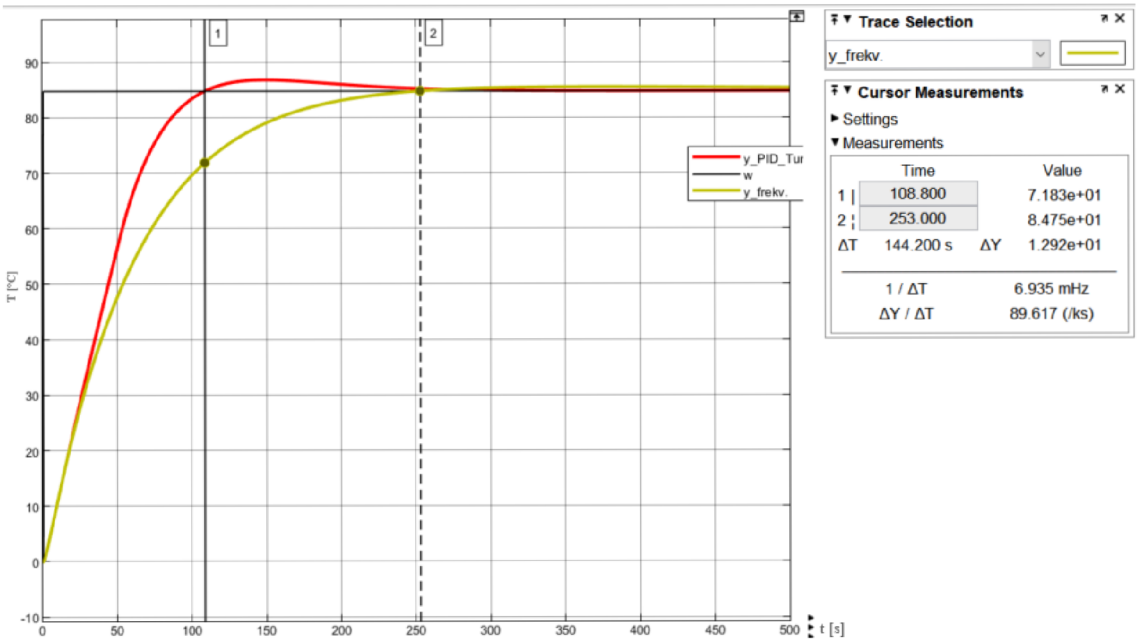
Z Obrázek 6-6 lze vyčíst, že regulátor dokázal dovést teplotu lihu uvnitř nádrže na 85 °C s překmitem 0,74 °C za zhruba 253 sekund.

6.2.1 Simulace PID regulátorů

Před samotnou implementací regulátorů do PLC jsem si průběhy teploty v tanku nasimuloval a v případě návrhu pomocí frekvenčních charakteristik doladil konstanty tak, aby se průběh výstupní hodnoty obvodu blížil mým požadavkům (velikost překmitu a doba regulačního děje). Pro získání potřebného tvaru přenosu regulátoru do prostředí TIA Portal, tak jak jsem ho navrhl metodou frekvenčních charakteristik, jsem využil funkce *pid*, do které jsem vložil tvar regulátoru ze *sisotool*.

Pro porovnání s mnou navrženým regulátorem jsem použil regulátor s automaticky nastavenými parametry pomocí funkce PID Tune App, s hodnotami popsanými na začátku této kapitoly. V bloku PID Controller jsem vybral metodu anti-windupu Clamping.

V Obrázek 6-7 lze vidět, že regulátor podle mého návrhu, viz kapitola 7, dosáhl žádané teploty za 253 sekund a překmitl o 0,74 °C. Automatický návrh překmitnul o 2 °C a na žádanou teplotu se vrátil až za 217 sekund. K aplikaci do procesu rektifikace lutru se více hodí první regulátor, protože je žádoucí, aby teplota narůstala pozvolně, a daly se tak od sebe oddělit jednotlivé složky lutru.



Obrázek 6-7 Porovnání návrhů regulátorů v simulaci

7. PROPOJENÍ MODELU S ŘÍZENÍM V PLC

Obslužné funkce pro komunikaci v PLC jsem vložil do hlavní smyčky kódu programu PLC. Pro PID regulátor jsem použil samostatnou cyklicky volanou funkci a pro automatický režim jsem napsal obslužnou funkci, která řídí proces podle stavového automatu, který jsem navrhnul.

Pro operátorský panel TP700 jsem vytvořil dvě obrazovky. Jednu pro běžnou obsluhu, druhou servisní, na které jsou další možnosti ovládání komunikace a regulátoru v PLC.

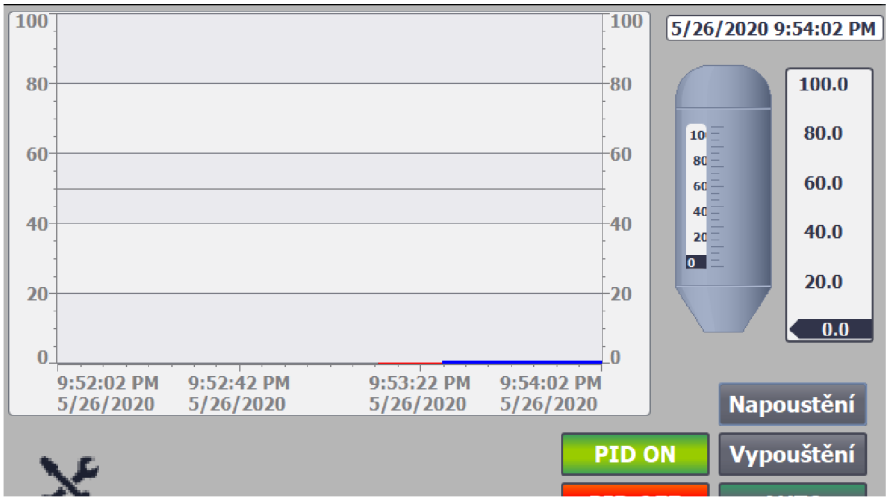
7.1 Popis řídícího systému

Zpracování programu v PLC probíhá cyklicky, kdy si PLC nejdříve uloží stavy vstupů do mezipaměti, potom vykoná program a pokud je potřeba, tak uloží stavy výstupů do mezipaměti a následně převede stavy výstupů z mezi paměti na fyzické výstupy, posledním krokem cyklu je kontrola pamětí, obsluhu požadavků na komunikaci. Tento cyklus se opakuje, dokud není PLC zastaveno. [21]

Jelikož se čas cyklu může měnit, např. v závislosti na počtu vykonávaných operací, tak se pro časově kritické aplikace (PID regulátor) využívají funkce s cyklickým přerušením hlavní smyčky, které zaručují, že potřební instrukce budou vykonávány v požadovaný čas.

7.2 Vizualizace

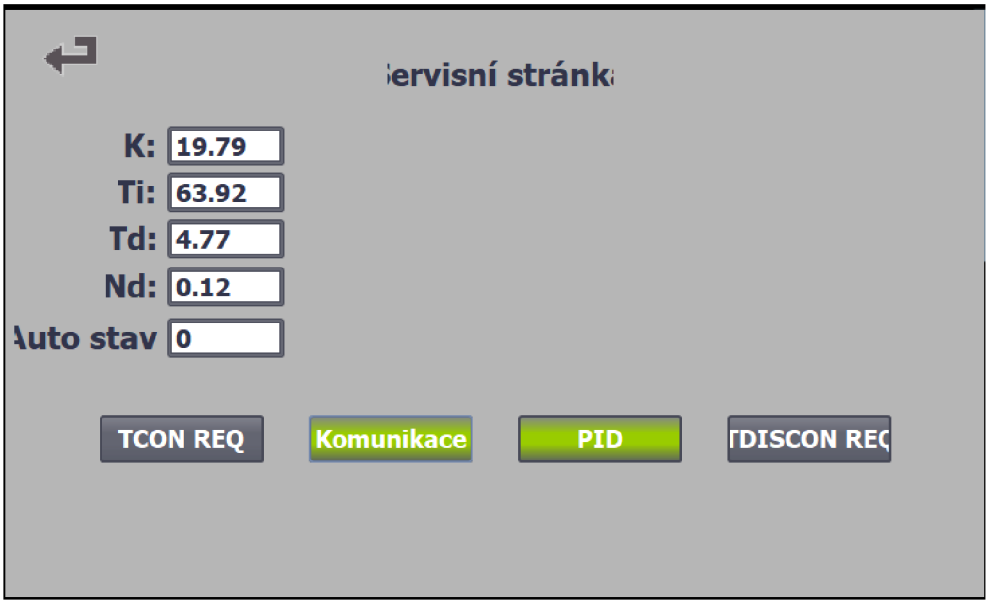
Pro ovládání simulace jsem připravil vizualizaci, která je zobrazena na operátorském panelu TP700 Comfort. Ve vizualizaci se nachází graf, ve kterém jsou zakresleny hodnoty požadované a aktuální hodnoty za posledních 100 sekund. Dále je zde naznačen tank, ve kterém se zobrazuje aktuální výška hladiny. Vedle něj se nachází sloupcový graf aktuální teploty, který slouží pro rychlou informaci obsluhy. V pravém dolním rohu se také



Obrázek 7-1 Vizualizace na operátorském panelu

nacházejí tlačítka pro ovládání regulátoru nebo napouštění a vypouštění nádrže. Náhled vizualizace je vidět na Obrázek 7-1.

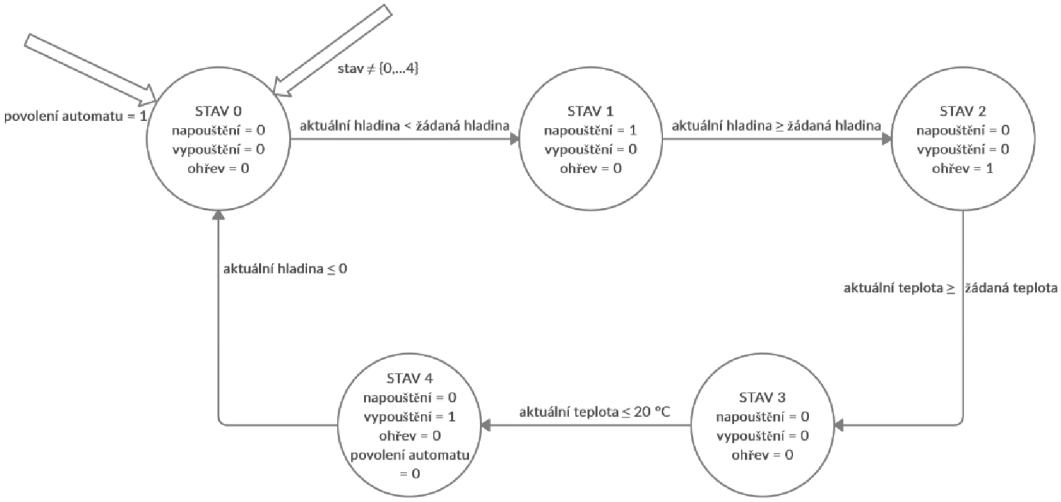
V levém dolním rohu se nachází tlačítko, které uživatele přesměruje na servisní stránku, kde jsou tlačítka pro ovládání komunikace a jsou zde také zobrazeny konstanty PID regulátoru, které jsou aktuálně nastaveny, viz Obrázek 7-2.



Obrázek 7-2 Vizualizace na operátorském panelu – servisní stránka

7.3 Automatický režim

Pro jednodušší ovládání celého destilačního programu jsem vytvořil obslužnou funkci, která má za cíl naplnit rektifikační tank lutrem, ohřát ho na danou teplotu a vypustit ho. Průběh funkce je popsán konečným stavovým automatem, viz Obrázek 7-3. Obslužná funkce čte a zapisuje přímo z, resp. do proměnných výšky hladiny, teploty lutru a ovládání ventilů. Povolování PID regulátoru je prováděno pomocí pomocné celočíselné proměnné *PID_Stavy*, která obsahuje číslo, kterému je přiřazen režim regulátoru blokem *PID_Temp* (0 → neaktivní, 3 → automatický režim). Tato proměnná je přivedena na vstup bloku regulátoru. Po změně stavu regulátoru je nutné provést změnu vstupního parametru regulátoru *ModeActivate* z log. „0“ na log. „1“. Následuje opačná změna, která zajistí, že je parametr „připraven“ pro další zavolání



Obrázek 7-3 Stavový automat automatické obsluhy rektifikace

8. PROPOJENÍ PLC A MATLAB/SIMULINK A OVĚŘENÍ FUNKČNOSTI

8.1 Obsluha komunikace pomocí SIMULINK

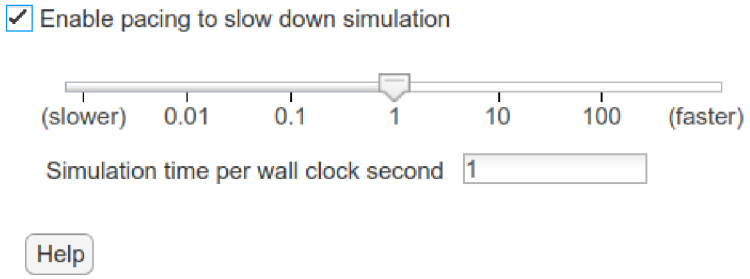
8.1.1 Inicializace

Před spuštěním simulace je potřeba nejprve inicializovat nastavení UDP parametrů. Pro účely inicializace jsem vytvořil m-file *komunikace_inicializace.m.* Tento soubor obsahuje příkaz pro vytvoření UDP objektu se správnými parametry, viz kapitola 2.3.2. Dále se inicializují pomocné proměnné, které využívají obslužných funkcích, jež jsou volané během simulace.

V tomto souboru jsou také hodnoty a vzorce pro výpočty parametrů rektifikačního tanku, jako je například objem vody a lutru, koeficienty přenosu tepla, specifická tepla kapalin apod.

8.1.2 Spuštění simulace

Simulace je spouštěna s nastavenými hodnotami parametrů *stop time* na *inf* (aby simulace běžela libovolně dlouho, dokud ji obsluha nevypne) a vzorkovací periodou *sample time*, která je určena proměnnou *tvz* z m-filu *komunikace_inicializace.m*. *Pacing* neboli krokování je povoleno a nastaveno na 1 (viz Obrázek 8-1), to znamená že jedna sekunda v simulaci je stejně dlouhá jako jedna sekunda ve skutečnosti. To zajišťuje



Obrázek 8-1 Krokování simulace

správnou vzorkovací frekvenci vzhledem k regulátoru v PLC.

Při běhu simulace se volají dvě obslužné funkce pro komunikaci. Jedná se o *MATLAB Function*, které mají jako vstup, resp. výstup vektory hodnot ze, resp. do simulace. Tyto vektory se skládají, resp. rozkládají pomocí multiplexorů a demultiplexorů (bloky *MUX* a *DMX*). Hodnota akčního zásahu pro tepelnou soustavu je přivedena na vstup bloku

přenosové funkce (*Transfer Fcn*). Žádané hodnoty se dají nastavit během simulace pomocí posuvníků.

8.1.3 Odesílání dat

Pro odesílání dat jsem naprogramoval funkci pro odesílání dat do PLC *send_Mfcn* má vstupní vektor složený z hodnot žádané a aktuální teploty a výšky hladiny. Data se následně upraví do tvaru, který vyhovuje PLC a pomocí evalin, assignin (viz podkapitola 8.5 Evalin a Assignin) a postupu popsaného v kapitole 4.5 Funkce fwrite.

8.1.4 Příjem dat

Příjem dat probíhá ve funkci *receive_Mfcn*, kterou jsem za tímto účelem vytvořil. Funkce má na výstupu jeden vektor, který obsahuje akční zásah pro změnu teploty a výšky hladiny. V této funkci je podmíněný cyklus, který má na starosti vyčítání dat ze vstupního bufferu *udp* objektu. Cyklus probíhá, dokud je počet bytů v buffer větší nebo roven počtu bytů jedné zprávy. Tento počet je určen počtem přijímaných hodnot × 4, jak bylo popsáno v kapitole 4.4 Funkce *fread*.

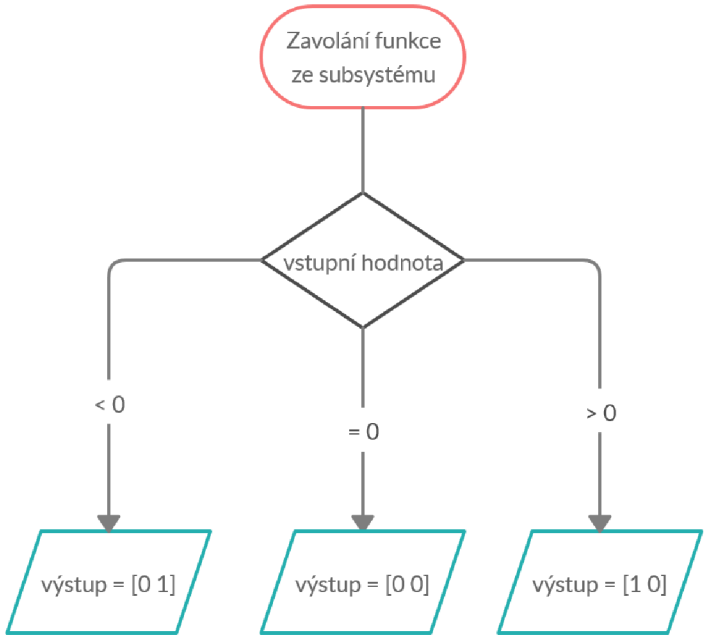
Jelikož regulátor v PLC má výstupní hodnoty v rozsahu 0–100 (%), tak podělím hodnotu právě přijatého akčního zásahu 100 a vynásobím ji hodnotou maximálního výkonu elektrického topidla a tuto hodnotu dám do výstupního vektoru. Hodnotu povelu pro ventily přepíšu taktéž do výstupního vektoru funkce.

8.1.5 Řízení výšky hladiny

Vstupem do subsystému pro modelování výšky hladiny jsou dvě hodnoty: povel pro ventily, který je výstupem z MATLAB funkce *receive_Mfcn*, a aktuální teplota na výstupu tepelné soustavy. Na výstup subsystému je přivedena aktuální výška hladiny, která se odesílá do PLC.

Výška hladiny je simulována integrátorem s omezením podle rozměrů vnitřního tanku. Do integrátoru vstupuje rozdíl mezi přitékajícím a odtékajícím lutrem. Zapnutí a vypnutí virtuálního čerpadla a otevření a zavření výpusti obstarává moje MATLAB funkce *fill_INOUT*, ve které se zanalyzuje vstupní hodnota a podle toho, jestli je větší než nula, tak se nádrž začne napouštět, pokud je menší než nula, tak se vypustí a pokud je rovna nule, tak se výška hladiny nezmění. Povely se uloží do výstupního vektoru, který se v subsystému rozdělí pomocí demultiplexoru, a jsou přivedeny na příslušná zesílení a do bloku VR Sink, kde slouží k naznačení napouštění a vypouštění nádrže.

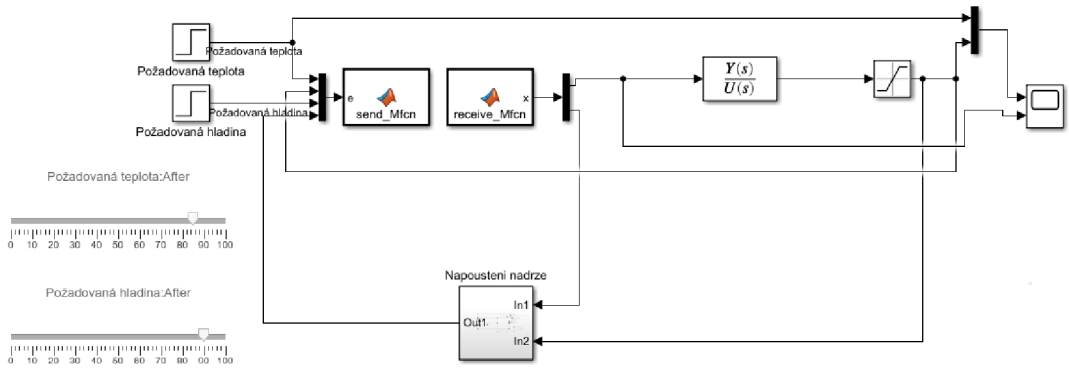
Princip spouštění/vypínání a otevírání/zavírání je znázorněn ve vývojovém diagramu na Obrázek 8-2.



Obrázek 8-2 Vývojový diagram funkce fill_INOUT

Hodnota teploty se zobrazí ve 3D světe pomocí bloku VR Text Output, který je popsán v kapitole 4.5.6 VR Text Output), viz Obrázek 6-1.

Blokové zapojení v simulaci jsem nahrnul jako regulační zpětnovazební obvod, kde regulátor reprezentují funkce pro komunikaci s PLC, viz Obrázek 8-3.



Obrázek 8-3 Blokové schéma regulačního obvodu

8.2 Obsluha komunikace pomocí PLC

8.2.1 Inicializace

Pro inicializaci komunikace pomocí UDP protokolu je zapotřebí přivést log. „1“ na vstup *REQ* bloku TCON buď v online režimu nebo pomocí vizualizace na operátorském panelu. Po úspěšné inicializaci a pokud je spuštěna simulace technologie v MATLABu na PC, je možné spustit komunikaci nastavením log. „1“ do proměnné *Comm_EN*, buď

v online režimu nebo pomocí vizualizace na operátorském panelu, která povoluje blok pro odesílání dat *TUSEND*. Při povolené komunikaci je možné posílat příkazy pro napouštění a vypouštění nádrže.

8.2.2 Regulátor

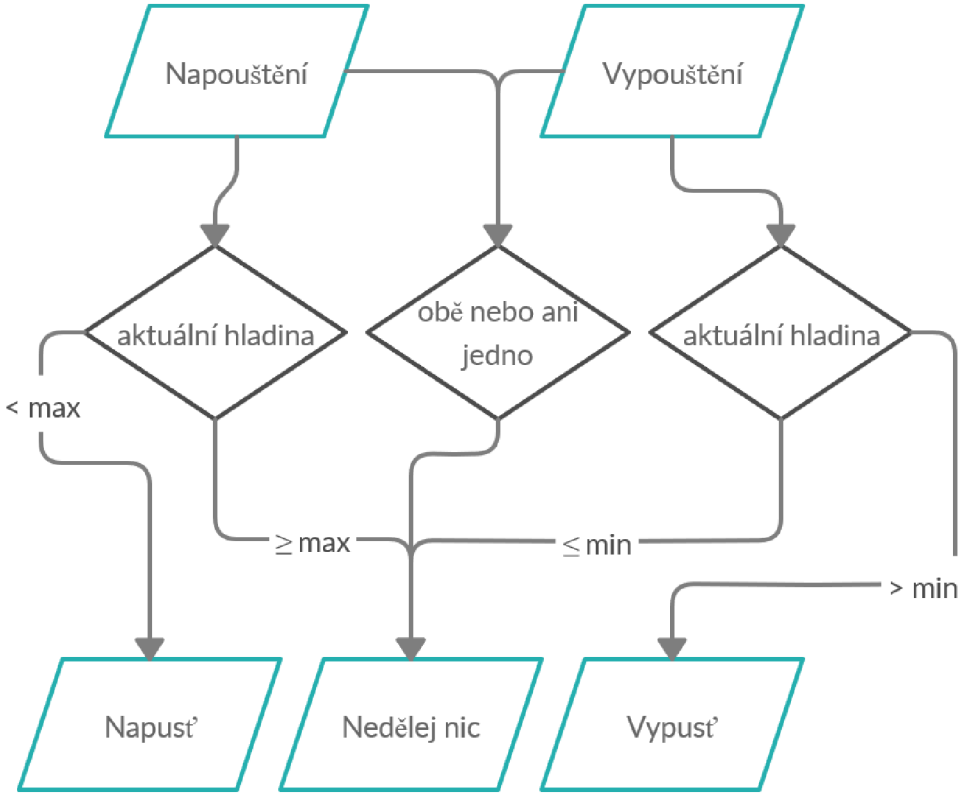
Blok PID regulátoru je umístěn v organizačním bloku, který je cyklicky volán s přesně danou periodou, která je stanovena v nastavení tohoto bloku. Toto řešení je použito z důvodu regulace, protože softwarově implementovaný PID regulátor musí být volán s danou periodou.

Do bloku vstupuje žádaná a aktuální hodnota teploty lutru v nádrži, výstupem je akční zásah, který je v rozsahu 0–100 % výkonu. V organizačním bloku se taktéž při každém zavolání bloku nastavuje pomocná proměnná, která v hlavním bloku odešle data do PC a kde se taktéž zresetuje.

Pokud je vyžadováno manuální spuštění regulace teploty pomocí regulátoru, tak je možné regulátor povolit nastavením log. „1“ do proměnné *PID_AktivovatStav*, buď v online režimu nebo pomocí stisknutí příslušného tlačítka ve vizualizaci na operátorském panelu.

8.2.3 Napouštění a vypouštění

Napouštění a vypouštění je možné ovládat pomocí vizualizace na operátorském panelu. Pro ovládání jsem připravil dvě tlačítka: *Napouštění* a *Vypouštění*, která nastavují příslušné proměnné v hlavním bloku programu, kde je vytvořena podmínka pro maximální, resp. minimální hladinu. Pokud je zmáčknuto tlačítko *Napouštění* a hladina je menší než maximální nastavená hodnota, tak se do proměnné, která se posílá do PC, uloží „1“. Při vypouštění je situace obdobná, jen se do dané proměnné uloží „-1“. Pokud hladina dosáhne maxima nebo minima, případně jsou-li stisknuta obě tlačítka současně nebo není stisknuté ani jedno, tak se do proměnné uloží „0“ a vyresetují se proměnné pro napouštění a vypouštění. Principiální popis funkce je znázorněn v Obrázek 8-4.



Obrázek 8-4 Vývojový diagram napouštění/vypouštění nádrže v PLC

.

8.3 Porovnání návrhů regulátorů

Pro porovnání průběhů teploty v závislosti na parametrech regulátoru jsem využil celkem tři návrhy PID regulátorů.

První návrh jsem vytvořil pomocí metody standardního tvaru frekvenční charakteristiky otevřeného obvodu podle postupu popsaného v kapitole 6.2.

Druhý návrh jsem vytvořil nástrojem PID Tuner App v programu Simulink v časové oblasti tak, že jsem nastavil Response time na 29,63 sekund a Transient Behavior na 0,9 (větší hodnota nejde nastavit, a proto nejde zcela eliminovat překmit, pokud chci zachovat Response time). Navržené konstanty jsem následně převedl funkcemi *pid* a *pidstd* pro převod do potřebného tvaru pro regulátor implementovaný v PLC.

Pro třetí návrh jsem využil automatického nastavení regulátoru v PLC, kde jsem použil možnost Fine Tuning.

Každý nástroj využívá trochu jiné tvary přenosu PID regulátoru a je nutné při přechodu mezi tvary jednotlivé konstanty případně přepočítat. Dále jsou uvedeny jednotlivé tvary.

Sériový tvar přenosu regulátoru, který je používán v TIA Portal [22]:

$$\mathrm{y} = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot p}(w - x) + \frac{T_D \cdot p}{a \cdot T_D \cdot p + 1}(c \cdot w - x) \right] \tag{8.1}$$

kde K_p – proporcionální zesílení

T_I – integrační konstanta

T_D – derivační konstanta

a – filtrační koeficient

b – váhování proporcionální složky

c – váhování derivační složky

DeadZone – necitlivost (pro nastavení anti-windup)

ControlZone – oblast řízení (šířka pásma okolo požadované hodnoty, kde je regulátor aktivní)

Sériový tvar přenosu regulátoru, který využívá Simulink [23]:

$$\mathrm{C} = P \left[1 + I \frac{1}{p} + D \frac{N}{1 + N \frac{1}{p}} \right] \tag{8.2}$$

kde P – proporcionální zesílení

I – integrační konstanta

D – derivační konstanta

N – filtrační koeficient

Paralelní tvar přenosu regulátoru, který využívá Simulink [23]:

$$\mathrm{C} = P + I \cdot \frac{1}{p} + D \frac{N}{\frac{N}{p} + 1} \tag{8.3}$$

kde P – proporcionální zesílení

I – integrační zesílení

D – derivační zesílení

N – časová konstanta filtru prvního řádu

Sériový tvar přenosu regulátoru, který využívá MATLAB ve funkci *pidstd* je [24]:

$$\mathrm{C} = K_p \left[1 + \frac{1}{T_I \cdot p} + \frac{T_D \cdot p}{\frac{T_D \cdot p}{N} + 1}\right] \tag{8.4}$$

kde K_p – proporcionální zesílení

T_I – integrační konstanta

T_D – derivační konstanta

N – filtrační koeficient

Paralelní tvar přenosu regulátoru, který využívá MATLAB ve funkci *pid* je [25]:

$$\mathrm{C} = K_p + \frac{K_I}{T_I \cdot p} + \frac{K_D \cdot p}{T_f \cdot p + 1} \tag{8.5}$$

kde K_p – proporcionální zesílení

K_I – integrační zesílení

K_D – derivační zesílení

T_f – časová konstanta filtru prvního řádu

8.4 Implementace PID regulátorů v PLC

Tabulka 8-1 Přehled parametrů regulátorů

Návrhová metoda / Parametr	Frekvenční charakteristiky	PID Tune App	Fine Tuning
K_p [-]	305,34	711,45	50,43
T_I [s]	305,34	11,13	3,47
T_D [s]	17,29	3390,98	0,77
a [-]	41,60	0,12	0,1
b [-]	1	1	0,58
c [-]	0	0	0

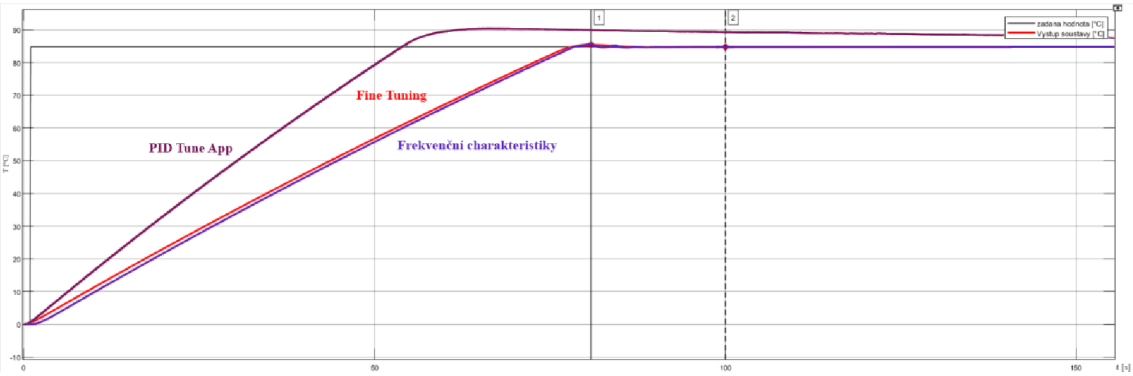
Tabulka 8-2 Přehled klíčových hodnot průběhů

Návrhová metoda / Parametr	Frekvenční charakteristiky	PID Tune App	Fine Tuning
Překmit [°C]	0,34	4,96	0,62
Překmit [%]	0,4	5,9	0,74
Počet viditelných překmitů	2	1	1

Doba dosažení požadované hodnoty [s]	78,5	54,3	77,8

V Tabulka 8-1 jsou vypsány parametry PID regulátoru ve tvaru, v jakém byly nastaveny v prostředí TIA Portal při implementaci do PLC. Pro všechny regulátory i soustavu byla nastavena perioda vzorkování v simulaci a běhu PLC programu na 0,1 sekundy. Dále jsem do Tabulka 8-2 uvedl hodnoty, které popisují některé parametry průběhy výstupních hodnot. Naměřené průběhy jsou vloženy do Příloha 1 -. Náhled těchto průběhů je zobrazen v Obrázek 8-5.

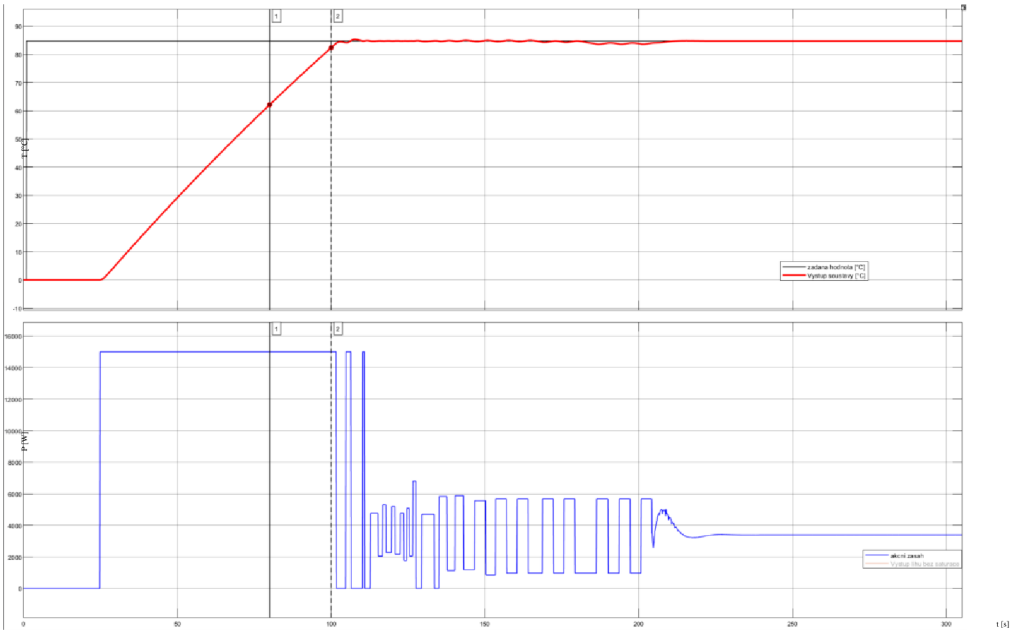
Z grafu v Obrázek 8-5 lze vidět, že nejrychleji požadované hodnoty dosáhl regulátor navržený v PID Tune App, nicméně tento regulátor měl také největší překmit a největší změny akčního zásahu. Průběh výstupní hodnoty řízené regulátorem, který byl navržený automatickým nastavením regulátoru pomocí programovacího prostředí TIA Portal, byl velmi podobný průběhu regulátoru, který jsem nastavil já pomocí frekvenčních charakteristik. Rozdíl mezi maximálními překmity byl necelých 0,3 °C, což je relativně malý rozdíl.



Obrázek 8-5 Demonstrace průběhů výstupní veličiny při řízení regulátory

Výhodou ručního nastavování regulátoru je, že lze nejdříve graficky nastavit odezvu uzavřeného obvodu, následně ji doladit na modelu a až poté implementovat do zařízení.

Automatické nastavení v PLC probíhá přímo na daném zařízení. Tento postup není vhodný tam, kde není žádoucí rozkmitání soustavy, případně kde nelze rozkmitat soustavu, protože návrhové prostředí zahlásí chybu, že nelze dosáhnout oscilace soustavy. Průběhy aktuální teploty a akčního zásahu při automatickém nastavování parametrů PID regulátoru jsou vidět v Obrázek 8-6. Výhodou automatického nastavení je úspora času oproti ručnímu návrhu.



Obrázek 8-6 Průběhy aktuální teploty a akčního zásahu při automatickém nastavování parametrů PID regulátoru v PLC

Mým požadavkem před návrhem regulátorů bylo, aby překmit nebyl větší než 1 °C Dále jsem měl požadavky na to, aby požadovaná teplota byla nejdříve překročena za 60 sekund a aby se aktuální teplota ustálila na žádané hodnotě během 80 sekund. Tyto požadavky splňuje mnou navržený regulátor pomocí frekvenčních charakteristik a automatický návrh z prostředí programu TIA Portal, viz Tabulka 8-2.

V porovnání se simulací je průběh výstupní veličiny (teplota lutru v rektifikačním kotli) při řízení regulátorem navrženým frekvenčními charakteristikami více než třikrát rychlejší a v případě řízení automaticky navrženým regulátorem dvakrát rychlejší. Překmity byly menší, než jaké vyšly v simulaci. Myslím si, že toto bylo způsobeno rozdílnou implementací algoritmu PID regulátoru včetně anti-windup technologie a také s problémy dodržet požadovanou periodu vzorkování na straně PC.

9. ZÁVĚR

Hlavním cílem této práce bylo vzdáleně řídit zjednodušený matematický model části výrobního procesu, za který jsem si vybral dvouplášťový rektifikační tank, pomocí regulátoru v PLC. Takový tank slouží ke druhé destilaci lihu (lutru), který vznikne první destilací ovocného kvasu. Skládá ze dvou válcových nádob, které jsou zasunuty do sebe a prostor mezi nimi vyplňuje voda, která ohřívá lutr. Objem vnitřní nádoby je zhruba 100 l. Po zanedbání některých jevů se jedná o tepelný systém druhého řádu, viz kapitola 5.3.

Prvním krokem bylo prostudovat možnosti komunikačních protokolů pro uskutečnění komunikace mezi PC se spuštěnou instancí MATLAB, ve které je nasimulován model rektifikačního tanku, a PLC S7-1500 a vybrat vhodný protokol pro propojení těchto dvou zařízení. Pro tento účel jsem původně zvolil protokol TCP, ale jelikož se mi ho nepodařilo po provedení několika kroků nastavení opětovně zprovoznit, využil jsem UDP protokol, který je pro tuto úlohu dostačující, i když nenabízí kontrolu doručení zprávy. Navíc jsou pro tento protokol připraveny funkce a bloky na obou zařízeních a na straně MATLABu.

I přes některé problémy se mi podařilo komunikaci zprovoznit a otestovat, viz kapitola 8. Obsluha komunikace na obou stranách probíhá automaticky. V programu MATLAB je pro komunikaci nutné spustit připravenou funkci, která nadeklaruje všechny potřebné proměnné a otevře simulaci, kterou stačí spustit. Na straně PLC lze komunikaci a řízení spustit ze servisní stránky ve vizualizaci na operátorském panelu. Hodnoty jsem mezi zařízeními předával pomocí polí, která jsou univerzální a dají se v případě potřeby pro další účely snadno rozšířit.

V dalším kroku jsem vytvořil 3D model rektifikačního kotle. Model má za cíl snadnější reprezentaci toho, co se v soustavě děje, tj. graficky zobrazuje výšku hladiny, napouštění a vypouštění a u nádrže se také zobrazuje aktuální teplota lutru.

K tomuto 3D modelu jsem připojil matematický model popisující teplotu a výšku hladiny Abych mohl demonstrovat funkčnost a bych mohl jednodušeji a rychleji implementovat různé PID regulátory, tak jsem navrhl podobnou soustavu, která ale měla kratší časové konstanty, takže regulační děj trval mnohonásobně kratší dobu oproti původnímu modelu.

Pro řízení teploty lutru jsem navrhl PID regulátor pomocí metody standardního tvaru frekvenční charakteristiky otevřeného obvodu. Tento regulátor jsem poté otestoval v simulaci. Poté jsem navrhl regulátor pomocí automatického systému PID Tune App, který je implementován v Simulinku. Následně jsem tyto dva regulátory porovnal. Pro danou aplikaci (rektifikace lutru) lépe vyhovoval mnou navržený regulátor, protože měl menší překmit a také pozvolněji dosáhl požadované hodnoty. To více vyhovovalo mým požadavkům, které jsem si stanovil před návrhem regulátoru

Po otestování regulátorů v simulaci jsem parametry těchto regulátorů nastavil v bloku PID regulátoru v prostředí programu TIA Portal. Tyto regulátory potom vzdáleně řídily

model soustavy, který byl simulován v PC. Díky možnosti automatického naladění regulátoru v TIA Portalu, jsem měl možnost porovnat celkem tři návrhy.

Nejméně ideální byl návrh pomocí PID Tune App, protože měl největší překmit a také měl nejvíce agresivní průběh akčního zásahu. To bylo nejspíše způsobeno rozdílnou implementací algoritmu regulátoru v Simulinku a MATLABu. Myslím si, že velké změny akčního zásahu mohl způsobovat jiné vyjádření přenosu regulátoru, a tím pádem jiný „význam“ parametrů. Dalším faktorem může být způsob, jakým je implementován anti-windup a také malá perioda vzorkování, kdy PC už nestačilo počítat teplotu na výstupu v dostatečně krátkém okamžiku vůči PLC. Mnou navržený regulátor byl o trochu pomalejší než automatický návrh z TIA Portalu, ale měl také menší překmit, což je lepší z hlediska mnou zadaný kritérií.

V PLC jsem ještě také navíc naprogramoval funkci, která automaticky napustí tank, ohřeje ho na požadovanou teplotu, potom vypne ohřev a počká, až líh vychladne. Následně pak tank vypustí. Tato automatická funkce lze spustit přes vizualizaci na operátorském panelu, kterou vytvořil.

Při dalším rozšíření bych se zaměřil na odhalení důvodu, proč nefungovala komunikace pomocí TCP/IP, dále bych porovnal model s reálnou soustavou, pokud by se naskytla taková možnost. Také bych detailněji popsal matematický model teploty lutru v nádrži, kde dochází k odpařování lihu a tím k úbytku objemu (hmotnosti) lihu, což ovlivňuje přenosovou funkci soustavy. Pokud by byl při zohlednění úbytku lihu velký rozdíl v chování soustavy, tak by bylo vhodné realizovat například gain scheduling regulátor. Co se týče automatického režimu, tak bych implementoval batch control podle standardu ISA-88. Vhodným rozšířením by mohlo být vytvoření grafického uživatelského prostředí v MATLABu, kde by bylo například možné nastavit IP adresy PLC, spuštění simulace, aktuální hodnoty, reset výšky hladiny a teploty lihu pro jednodušší testování atp.

Citovaná literatura

[1] SIEMENS. *Open TCP/IP Communication via Industrial Ethernet: A5E00711636-01* [online]. 12.2005. 2005 [cit. 2019-12-15]. Dostupné z: https://cache.industry.siemens.com/dl/files/612/22146612/att_113921/v1/t-bausteine_e.pdf

[2] MATHWORKS. *Tcpclient* [online]. MATHWORKS. b.r. [cit. 2019-12-15]. Dostupné z: https://www.mathworks.com/help/matlab/ref/tcpclient.html

[3] MATHWORKS. *Tcpip* [online]. MATHWORKS. b.r. [cit. 2019-12-15]. Dostupné z: https://www.mathworks.com/help/instrument/tcpip.html

[4] MATHWORKS. *Udp* [online]. MATHWORKS. b.r. [cit. 2019-12-16]. Dostupné z: https://www.mathworks.com/help/instrument/udp.html

[5] MATHWORKS A TRAEGER. *IP-S7-LINK* [online]. MATHWORKS A TRAEGER. b.r. [cit. 2019-12-16]. Dostupné z: https://www.mathworks.com/products/connections/product_detail/ip-s7-link.html

[6] MATHWORKS A SIEMENS. *SIMATIC Target 1500S* [online]. MATHWORKS A SIEMENS. b.r. [cit. 2019-12-17]. Dostupné z: https://www.mathworks.com/products/connections/product_detail/simatic-target-1500s.html

[7] MATHWORKS. *Instrument Control Toolbox* [online]. MATHWORKS. b.r. [cit. 2019-12-18]. Dostupné z: https://www.mathworks.com/products/instrument.html

[8] MATHWORKS. *Uint8* [online]. MATHWORKS. b.r. [cit. 2019-12-18]. Dostupné z: https://www.mathworks.com/help/matlab/ref/uint8.html?searchHighlight=uint8&s_tid=doc_srchtitle

[9] MATHWORKS. *Typecast* [online]. MATHWORKS. b.r. [cit. 2019-12-18]. Dostupné z: https://www.mathworks.com/help/matlab/ref/typecast.html?searchHighlight=typecast&s_tid=doc_srchtitle

[10] Single. *Mathworks* [online]. b.r. [cit. 2019-12-18]. Dostupné z: https://www.mathworks.com/help/matlab/ref/single.html?searchHighlight=single&s_tid=doc_srchtitle

[11] *Vizualizace dynamických systémů v prostředí virtuální reality: Učební texty k semináři* [online]. DANĚK, Jan. b.r. [cit. 2019-12-18].

[12] ZBOROVSKÝ, Vojtěch. *Propojení virtuálního modelu v MATLAB/Simulink s PLC* [online]. Brno, 2018, 59 s. [cit. 2019-12-20]. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií.

[13] MATHWORKS. *Navigate Using Orbisnap* [online]. MATHWORKS. b.r. [cit. 2019-12-21]. Dostupné z: https://www.mathworks.com/help/sl3d/navigate-with-orbisnap.html

[14] Jak pálit slivovici: Postup destilace. *Slovácké pálenice* [online]. b.r. [cit. 2020-05-14]. Dostupné z: https://slovacke-palenice.cz/jak-destilovat

[15] Pěstitelská pálenice: Úkapy, dokapy, vzorky z měřidel. *CENTEP* [online]. b.r. [cit. 2020-05-15]. Dostupné z: https://www.centep.cz/index.php?id=10005&lang=cze

[16] Přiboudlina. *WIKISKRIPTA* [online]. b.r. [cit. 2020-05-15]. Dostupné z: https://www.wikiskripta.eu/w/P%C5%99iboudlina

[17] SVA. *Sigma: pumpy Hranice* [online]. b.r. [cit. 2020-05-16]. Dostupné z: http://www.sigmapumpy.com/cerpadlo-sva-id431.html

[18] *Regulované soustavy* [online]. b.r. [cit. 2020-05-31]. Dostupné z: http://books.fs.vsb.cz/sipro/sippri2.htm

[19] JIRGL, Mirloslav. *Archiv Miroslava Jirgla* [online]. 2020 [cit. 2020-05-16].

[20] Ovocné destiláty: Destilace potravinářského lihu. *Mendelu.cz* [online]. b.r., 2018-08-17 [cit. 2020-05-24]. Dostupné z: http://web2.mendelu.cz/af_291_projekty2/vseo/stranka.php?kod=8594

[21] *What is process scan & process time in PLC* [online]. b.r. [cit. 2020-05-31]. Dostupné z: https://automationforum.in/t/what-is-process-scan-process-time-in-plc/3666

[2] SIEMENS. *Closed-Loop Control with "PID_Compact" V2.2: SIMATIC S7-1500* [online]. b.r., 2017-03 [cit. 2020-05-26]. Dostupné z: https://cache.industry.siemens.com/dl/files/707/79047707/att_915339/v2/79047707_PidCompactV2_2_DOC_V2_0_1_en.pdf

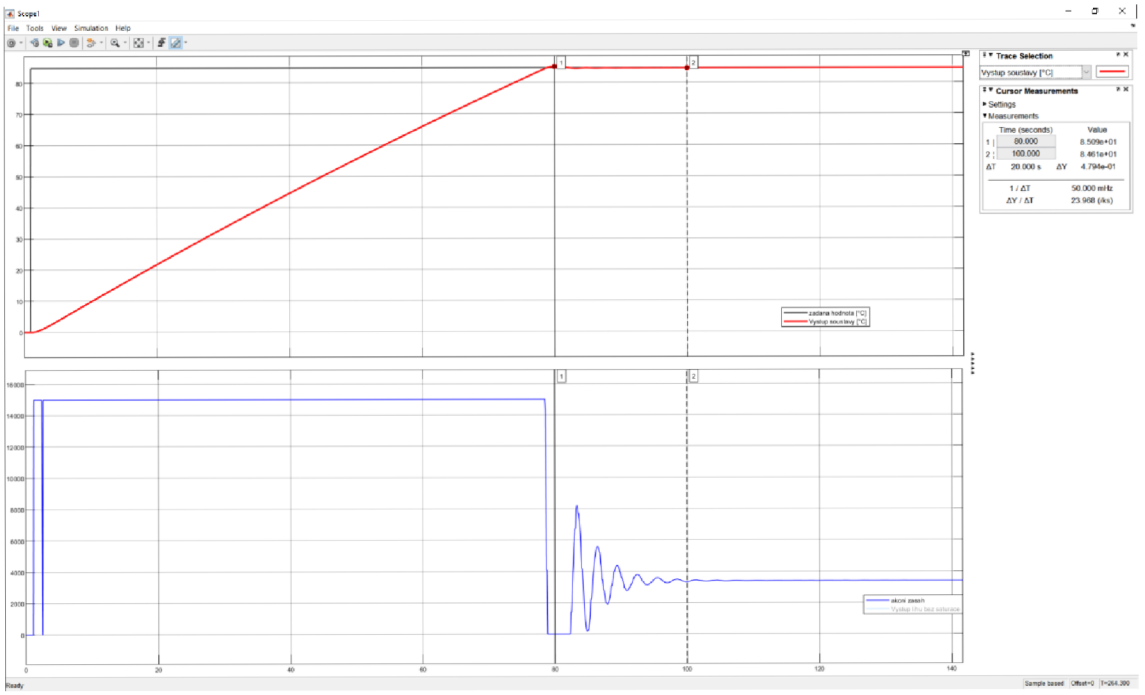
[3] MATHWORKS. *PID Controller: Continuous-time or discrete-time PID controller* [online]. b.r. [cit. 2020-05-26]. Dostupné z: https://www.mathworks.com/help/simulink/slref/pidcontroller.html?s_tid=srchtitle

[4] MATHWORKS. *Pidstd: Create a PID controller in standard form, convert to standard-form PID controller* [online]. b.r. [cit. 2020-05-26]. Dostupné z: https://www.mathworks.com/help/control/ref/pidstd.html

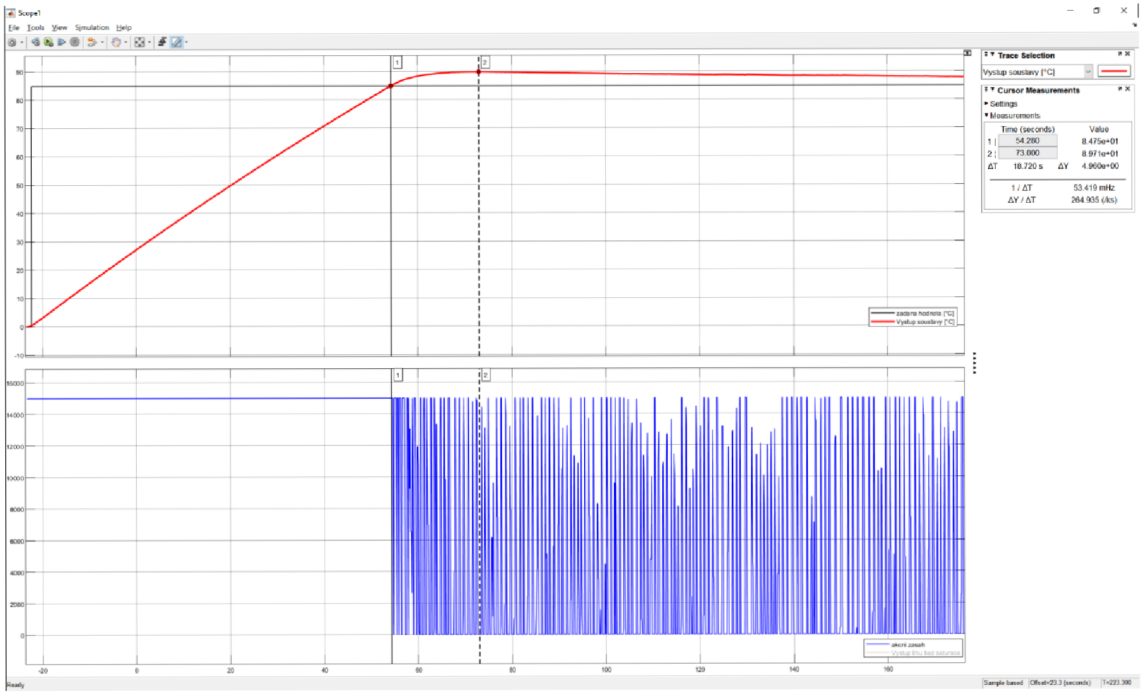
[5] MATHWORKS. *Pid: Create PID controller in parallel form, convert to parallel-form PID controller* [online]. b.r. [cit. 2020-05-26]. Dostupné z: https://www.mathworks.com/help/control/ref/pid.html

Seznam příloh

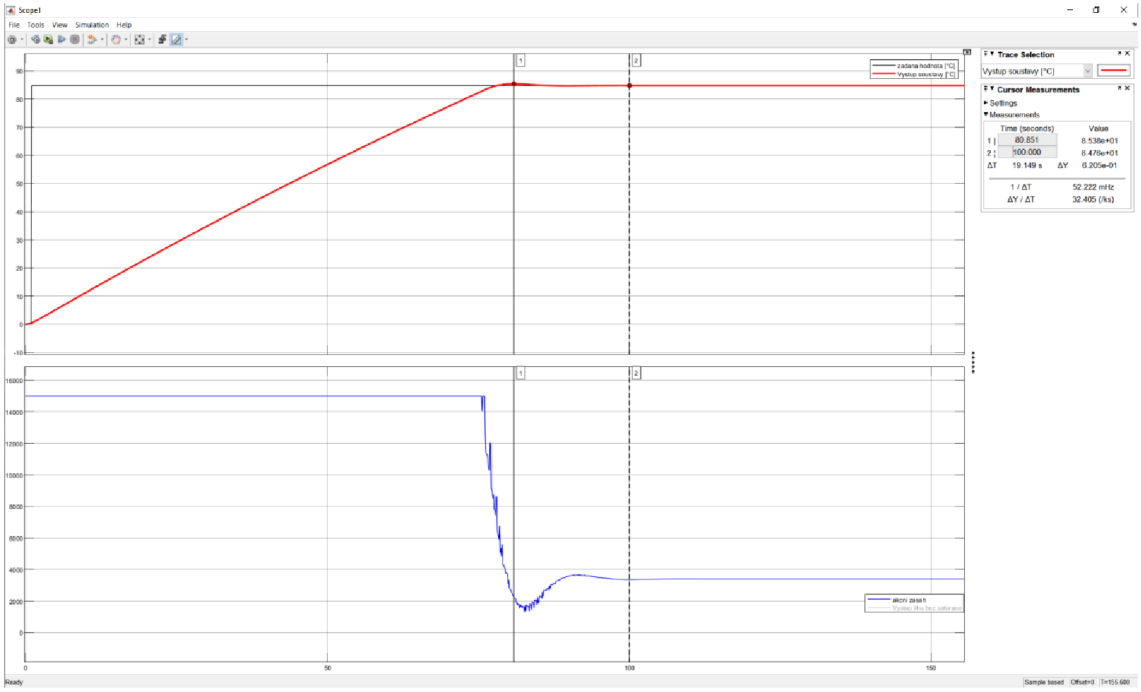
Příloha 1 - **Naměřené průběhy regulované veličiny různými regulátory**



Obrázek 9-1 Regulátor navržený pomocí frekvenčních charakteristik



Obrázek 9-2 Regulátor navržený pomocí PID Tune App



Obrázek 9-3 Regulátor navržený pomocí Fine Tuning v PLC

Příloha 2 - **Program pro MATLAB**

Zdrojový kód programu je uložen v systému

Příloha 3 - **Simulace modelu**

Zdrojový kód programu je uložen v systému

Příloha 4 - **Návrhy regulátorů v nástroji sisotool**

Zdrojový kód programu je uložen v systému

Příloha 5 - **3D model tanku**

Zdrojový kód programu je uložen v systému

Příloha 6 - **Projekt pro PLC**

Zdrojový kód programu je uložen v systému