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

PROPOJENÍ VIRTUÁLNÍHO MODELU V MATLAB/SIMULINK S PLC
CONNECTING THE MATLAB/SIMULINK VIRTUAL MODEL TO PLC

DIPLOMOVÁ PRÁCE
MASTER THESIS

AUTOR PRÁCE
AUTHOR
Bc. Vojtěch Zborovský

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Miroslav Jirgl, Ph.D

BRNO 2018

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. Vojtěch Zborovský ***ID:*** 162088

Ročník: 2 ***Akademický rok:*** 2017/18

NÁZEV TÉMATU:

Propojení virtuálního modelu v MATLAB/Simulink s PLC

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je propojení virtuálního modelu v prostředí MATLAB/Simulink, vytvořeného v jazyce VRML, s PLC S7-1500 využitím TCP/IP komunikace a následná realizace řídicího algoritmu.

1. Prostudujte problematiku tvorby virtuálních modelů v jazyce VRML a stručně popište.
2. Navrhněte a realizujte jednoduchý 3D model vybraného technologického zařízení.
3. Seznamte se s knihovnou 3D Animation v prostředí MATLAB/Simulink a stručně popište jednotlivé bloky.
4. Pro navržený 3D model zařízení vytvořte jeho zjednodušený fyzikální model.
5. V prostředí MATLAB/Simulink implementujte výsledný 3D model technologického zařízení a připojte příslušný fyzikální model.
6. Pro daný model navrhněte řízení a ověřte vlastnosti modelu pomocí simulace.
7. Přes TCP/IP komunikaci propojte model v prostředí MATLAB/Simulink s PLC SIEMENS S7-1500 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: 5.2.2018 ***Termín odevzdání:*** 14.5.2018

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

Konzultant:

. **Ing. Václav Jirsík, CSc.**

předseda oborové rady

Abstrakt

Diplomová práce popisuje tvorbu virtuální reality v jazyce VRML pomocí editoru VRML Pad a VRealm Builder. Obsahuje také popis objektů v knihovně simulink 3D Animation používaných pro tvorbu trojrozměrných virtuálních světů a propojení dynamiky z Matlab/Simulink. Součástí práce je návrh modelu a dynamiky technologického procesu a jeho implementace do programu Matlab/Simulink. Kompletní model je propojen přes TCP/IP k systému PLC S7-1500, ve kterém je navrženo a realizováno řízení technologického procesu. Proces je navíc vizualizován na HMI od firmy Siemens, které umožňuje jednoduché ovládání komunikace a technologického procesu.

Klíčová slova

PLC, HMI, MATLAB, TCP/IP, VRML

Abstract

Master's thesis describes creating of virtual reality in VRML Language with use of VRML Pad editor and VRealm Builder. The thesis consists of the described objects in Simulink 3D Animation Library which are used for 3D Virtual scenes and connection of dynamics from Matlab/Simulink. Dynamics is created in Matlab/Simulink and connected by TCP/IP protocol to system PLC S7-1500. In the PLC is program for control of technology process. Process is visualized by HMI by Siemens AG and supplied by basic setting of connected communication and technology process.

Key words

PLC, HMI, MATLAB, TCP/IP, VRML

Bibliografická citace:

ZBOROVSKÝ, V. *Propojení virtuálního modelu v MATLAB/Simulink s PLC*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2018. 59 s. Vedoucí diplomové práce Ing. Miroslav Jirgl, Ph.D..

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma Propojení virtuálního modelu v Matlab/Simulink s 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: **10. května 2018** .....
podpis autora

Poděkování

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

V Brně dne: **10. května 2018**

.....
podpis autora

Obsah

1. Úvod 12
2. Virtuální světy 13
2.1 Jazyk VRML 13
2.2 Souřadnicový systém 14
2.3 Příkazy v jazyce VRML 14
2.4 Tvorba virtuálních modelů s pomocí modelovacích programů 16
2.4.1 VRMLPad 17
2.4.2 VRealm Builder 18
2.5 Tvorba základních objektů pomocí nástrojů VRealm Builder a VRML Pad .. 19
2.5.1 Pozadí 20
2.5.2 Výchozí bod pozorování 20
2.5.3 Transform 21
2.5.4 Další objekty 24
3. Návrh 3D modelu technologického zařízení - Tanku 25
4. KnihovnA 3D Animation v prostředí MATLAB/Simulink 27
4.1 VR Sink 27
4.2 VR Source 28
4.3 VR Signal Expander 29
4.4 VR Placeholder 30
4.5 VR Tracer 30
4.6 Real Time Synchronization 30
5. Fyzikální model tanku 31
5.1 Model výšky hladiny 31
5.2 Model teploty 32
5.3 Návrh regulátoru pro model teploty. 33
6. Realizace fyzikálního modelu v prostředí Matlab/Simulink 35
6.1 Model výšky hladiny 35
6.2 Model teploty 36
7. Propojení virtuálního modelu s dynamikou v Matlab/Simulink 37
8. Využití PLC pro řízení 40

8.1 Model versus reálný systém 40
8.2 Řídicí program 41
8.3 Návrh HMI 43
9. Komunikace mezi PLC a Matlab 44
9.1 Nastavení komunikace pro propojení v PLC 44
9.2 Bloky v PLC pro komunikaci TCP/IP 45
9.2.1 Blok TCON 45
9.2.2 Blok TDISCON 45
9.2.3 Blok TSEND 45
9.2.4 Blok TRCV 46
9.2.5 Blok TDIAG 46
9.3 TCP/IP komunikace v prostředí Matlab/Simulink 46
10. Realizace komunikace 47
10.1 Tvorba a odesílání paketu dat 48
10.1.1 Matlab ->PLC 48
10.1.2 PLC -> Matlab 49
10.2 Přijímání dat z komunikace 49
10.2.1 PLC -> Matlab 50
10.2.2 Matlab -> PLC 51
10.3 PI regulátor v Matlab/Simulink 51
10.4 Regulátor PI v PLC 52
11. Ověření funkčnosti a výsledků 54
11.1 Porovnání simulace Matlab a naměřených dat z PLC 56
12. Závěr 57
Literatura 58

Seznam symbolů, veličin a zkratek

VRML	-	Virtual Reality Modeling Language	
VUT	-	Vysoké učení technické v Brně	
PLC	-	Programable Logic Controller	
VR	-	Virtuální Realita	
RT	-	Real Time, Reálný Čas	
HMI		Human Massive Interface	
A	-	Obsah kruhu	$[m^2]$
a	-	Průřez odtoku	$[m^2]$
w	-	Průtok přítoku	$[m^3 \cdot s^{-1}]$
ω	-	Kmitočet	$[rad \cdot s^{-1}]$
Ti	-	Časová konstanta regulátoru	[s]
K	-	Zesílení regulátoru	[-]
K_V	-	Ztrátový koeficient	$[Wm^{-3} \cdot K^{-1}]$
g	-	Gravitační konstanta	$[m^3 \cdot kg^{-1} \cdot s^{-2}]$
m_V	-	Hmotnost vody	[kg]
c_V	-	Specifické teplo vody	$[Ws \cdot kg^{-1} \cdot °C^{-1}]$
p	-	Příkon otopného systému	[W]
Q	-	Objem	$[m^3]$
t	-	Čas	[s]

Seznam obrázků

Obr. 2-1 Souřadnicový systém ve VRML 14
Obr. 2-2 Kartézský souřadnicový systém 14
Obr. 2-3 Základní struktura objektu Transform 15
Obr. 2-4 Našeptávač ve VRML Pad 17
Obr. 2-5 Textový editor VRML Pad 18
Obr. 2-6 V Realm Builder 19
Obr. 2-7 Pozadí virtuálního světa 20
Obr. 2-8 Viewpoit v textu 20
Obr. 2-9 Viewpoint ve VRealm Builder 20
Obr. 2-10 Zobrazený systém v Matlab 21
Obr. 2-11 Základní struktura objektu Transform 22
Obr. 2-12 Nastavení poloměru válce 23
Obr. 2-13 Nastavení barvy objektu 23
Obr. 2-14 Výběr Barvy z knihovny 24
Obr. 2-15 nastavení Transform v Textovém Editoru 24
Obr. 3-1 Návrh Vizualizace Tanku se Solid Edge 25
Obr. 3-2 Návrh zjednodušeného modelu tanku 26
Obr. 3-3 Návrh ovládacího ventilu 26
Obr. 4-1 VR Sink 28
Obr. 5-1 Nákres modelu výšky hladiny [3] 31
Obr. 5-2 Blokové schéma [3] 32
Obr. 5-3 Graf návrhu regulátoru LAFCH 34
Obr. 6-1 Model nádrže v MATLAB 35
Obr. 6-2 Model teploty MATLAB 36
Obr. 7-1 Úprava dynamiky pro model ve VRML 37
Obr. 7-2 Úprava dynamiky pro model ve VRML 38
Obr. 7-3 Volba vstupu do VR Sink 39
Obr. 8-1 Graf pro standardizaci teploty [6] 41
Obr. 8-2 Blokový diagram pro automatický režim 41

Obr. 9-1 Výměna dat mezi systémy 44
Obr. 10-1 Propojení komunikace HMI, PLC a PC 47
Obr. 10-2 Nastavení bloku TCON 47
Obr. 10-3 Vývojový diagram pro odesílání dat 48
Obr. 10-4 Vývojový diagram pro přijímání dat 50
Obr. 10-5 Nastavení PI regulátoru Matlab/Simulink 52
Obr. 10-6 Rovnice PID regulátoru v PLC Siemens 52
Obr. 10-7 Vstupní a výstupní parametry regulátoru 53
Obr. 10-8 Nastavení parametrů PID v PLC 53
Obr. 11-1 Signalizace stavu komunikace 54
Obr. 11-2 Ověření funkčnosti komunikace 55
Obr. 11-3 Příklad nastavení hodnoty a ověření komunikace 55
Obr. 11-4 Napouštění tanku v RT 55
Obr. 11-5 Průběh napouštění a vypouštění 56

1. ÚVOD

Virtuální světy a vizualizace nám umožnují zobrazovat jednoduchá i složitější zařízení do 3D nebo 2D prostorů. Pomocí kterých lze přehledně a jednoduše monitorovat, sledovat a ovládat procesy nebo technologie. K tomu nám slouží několik nástrojů, které umožňují převést naše představy do počítače nebo vizualizačního zařízení. Hlavní výhodou virtuálních modelů je simulace reálných modelů bez nutnosti vlastnit fyzické zařízení nebo technologii. Stačí znát fyzické parametry a požadovanou technologii virtualizovat a simulovat. Používání virtuálních modelů může ušetřit náklady a odhalit chyby ještě předtím, než se začne s výrobou skutečných technologických zařízení. Například při sestavování technologie nebo při tvorbě a testování řídícího programu. Pokud by se program testoval až na skutečném zařízení, mohlo by v některých případech dojít jeho zničení, což je nežádoucí.

Cílem diplomové práce je seznámení s prostředky pro tvorbu virtuálních modelů v jazyce VRML, následného navržení a vytvoření virtuálního modelu technologického procesu pro napouštění, ohřev a vypouštění tanku s vodou. Do virtuálního modelu implementovat jeho fyzikálních vlastnosti pomocí prostředí Matlab/Simulink. Poté pomocí knihovny 3D Animation implementované v Matlabu bude propojen fyzikální model a virtuální model v jeden celek a simulováno jeho chování. Následně model z programu Matlab/Simulink propojit s PLC S7-1500 firmy Siemens pomocí TCP/IP komunikačního protokolu a posílat mezi oběma zařízeními data v reálném čase. Součástí řešení je i vytvoření ovládání přes HMI - Simatic Touch panel, jenž je připojené k PLC.

2. VIRTUÁLNÍ SVĚTY

2.1 Jazyk VRML

Zkratka vychází z anglického Virtual Reality Modeling Language. Jazyk je určen k popisu obsahu virtuálních světů a jejich chování. Jazyk VRML definuje způsob zápisu virtuálních světů do souborů v textovém formátu, pomocí takzvaných uzlů (Nodes) a je současně předpisem pro tvorbu virtuální reality. Tento jazyk není výtvorem jediné firmy, ale byl stvořen firmami a odborníky z celého světa a je uznáván jako univerzální standard.

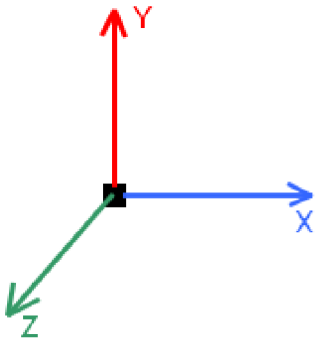
Tento jazyk vznikl v roce 1995 jako verze VRML 1.0. Poté probíhaly úpravy a koncem roku 1997 byl jazyk VRML uznán jako standard ISO s označením ISO/IEC 147721-1:1997. V dnešní době je znám pod mezinárodním názvem VRML97.

Výhodou virtuálních světů je propojení prostorových objektů ve 3D prostoru s animacemi a zvuky. Prvky pro tvorbu virtuálních světů nebo samotné virtuální světy lze propojovat z lokálních, ale také z externích úložišť kdekoliv na internetu. Přecházení mezi světy je plynulé z důvodu nízké grafické náročnosti samotných prvků, které VRML dokáže vytvořit. Pomocí několika málo bodů lze vytvořit jednoduchý svět kde výpočet pro zobrazení objektů v prostoru bude potřebovat velmi nízký výpočetní výkon. Pro popis statického i dynamického světa se používají stejné prostředky, a proto není nutné měnit statický prvek na pohyblivý, ale lze mu jen předepsat další vlastnosti.

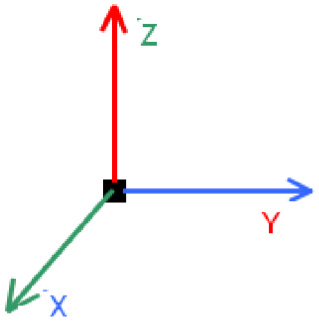
V prostředí se lze pohybovat všemi směry. Chodit, létat nebo zkoumat jednotlivé objekty. Umožňuje také propojení interakcí uživatele do VRML světa a propojení s dalšími programy nebo programovacími jazyky. Popis virtuálních světů je ukládán v textovém tvaru a jeho velikost lze snížit ještě kompresí pomocí dalších programů, aniž bychom ztratili informace, a lze je zase dekódovat zpět.[1][2]

2.2 Souřadnicový systém

Ve VRML je však pootočený souřadnicový systém[5] na rozdíl od klasického kartézského, kde osa Z míří kolmo vzhůru. V základním zobrazení ve VRML míří osa Z směrem k nám.



Obr. 2-1 Souřadnicový systém ve VRML



Obr. 2-2 Kartézský souřadnicový systém

2.3 Příkazy v jazyce VRML

Základními jednotkami v jazyce VRML jsou pro vzdálenosti – metry, pro úhly – radiány, čas je udávaný v sekundách, barvy jsou popsané v RGB formátu, a to v rozsahu 0-1 namísto standartního 0-255.

Další prvky a výrazy a názvosloví užívané ve virtuálních světech uvedu pro lepší orientaci ve VRML.

Avatar je náš virtuální dvojník. V podstatě to, co vidí on, vidím také včetně perspektivního zkreslení. My pak sedím u velitelského stanoviště a mohu jeho pohyby ovládat. Lze si jej představit jako válec na pomyslných nohách s definovanými parametry. [1],[2].

NavigationInfo je předpis vlastností avatara, který obsahuje parametry o velikosti avatara, osvětlení prostoru ve směru pohledu, vzdálenost dohledu a rychlost pohybu.

AvatarSize definuje velikost avatara pomocí parametrů [*r*, *e*, *s*], kde *r* je poloměr ochranného válce kolem výšky avatara, *e* je výška očí v ose y. Poslední parametr *s* udává maximální výšku překročitelné překážky avatarem. Všechny hodnoty jsou v metrech.

Dalším prvkem je ***headlight***, což je čelní světlo, s hodnotami zapnuto nebo vypnuto (TRUE/FALSE).

VisibilityLimit je dohled avatara. Pokud je hodnota 0,0 zobrazí se veškeré objekty v reálném světě. Pokud bude nastavena jiná hodnota, zobrazí se pouze objekty ve vzdálenosti určené touto hodnotou od pozorovatele. Ideální je dosah viditelnosti omezit z důvodu náročnosti vykreslování prvků a úspory výpočetního výkonu.[2]

Speed je rychlost, s jakou se avatar pohybuje, tedy po jakých krocích se posunuje. Jednotka je metr za sekundu.[2]

Type je styl pohybu, lze vybrat chůzi, let, nebo prozkoumávání, type může obsahovat parametry [“WALK”, ”FLY”, ”EXAMINE”, ”ANY”/ ”NONE”]. ANY umožňuje použití myši a přepínat mezi prvními třemi režimy pohybu a NONE zakazuje myš a přepínání mezi režimy pohybu. [2]

WorldInfo obsahuje informace o světě. Parametr title – název světa, info – může mít více řádků, obsahuje informace o autorovi, datu vytvoření a komentáře, které se nezobrazují ve světě. [2]

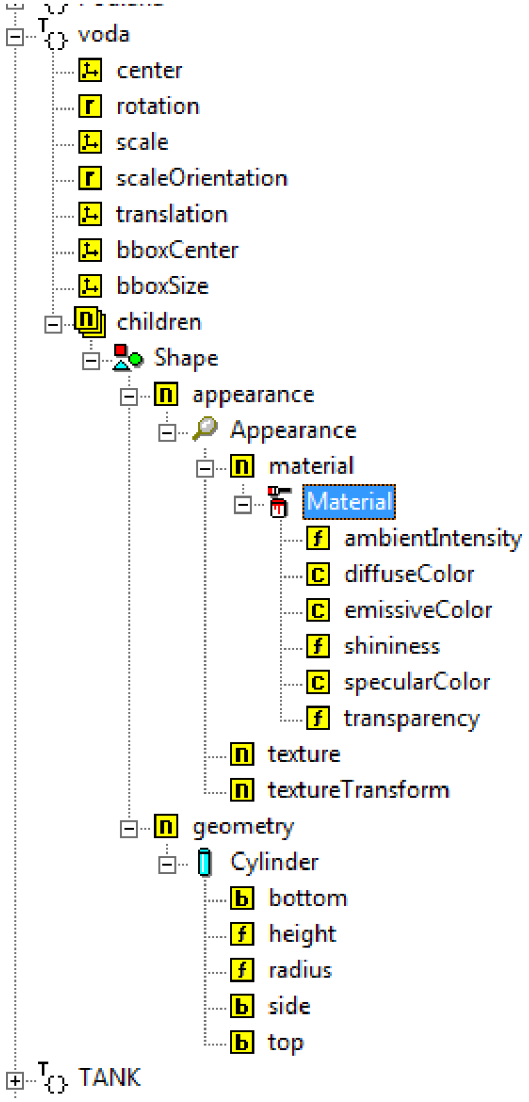
Viewpoint je bod pohledu, určuje pozici bodu, ve kterém se pozorovatel nachází, směr pohledu a natočení, a také určuje pole viditelnosti. Je to místo výchozího stanoviště, kam se teleportuje avatar. Může jich být i více.

Position- pozice v osách [x y z].

Orientation slouží k otočení pohledu ve vybrané ose x, y, nebo z o úhel v radiánech [x y z úhel] pozn. 1,57 radiánu = 90°.

DEF je klíčové slovo definice. Prvky, které jsou uvozeny touto zkratkou, jsou předpisem. Vlastnosti, které obsahuje, lze používat v dalších prvcích a jednoduše je přenášet.

USE je pro použití definice *DEF*. Pro zkrácení kódu je vhodné používat definice objektů. Pozor při používání ve zděděném



Obr. 2-3 Základní struktura objektu Transform

objektu (Transform). Změna vlastnosti v DEF mění vlastnosti u všech uzlů, které jsme vytvořili pomocí ***USE.***

Background – umožňuje definovat pozadí světa.

Transform – definice objektu zahrnující potomky (Children), které dědí parametry jako je rotace, měřítko, posunutí a atp.

Children je uzel, do kterého se umisťují Shape (tvary) nebo další prvky typu transform. Tímto způsobem lze vytvořit sestavy objektů, které se budou chovat jako jeden celek.

Shape je umožňuje přidání požadovaného geometrického objekt a definovat jeho vlastnosti, a barvu, průhlednost nebo materiál.

Základní tvary ve větvi geometry jsou BOX (krabice), CYLINDER (válec), SPHERE (koule) a kužely. Prvky mohou tvořit jeden celek a představovat tak jedno složitější těleso (Transform). Pokud chci seskupit více různých objektů dohromady, slouží nám k tomu příkaz ***Group***.

EXTERNPROTO se používá pro vložení dalších prvků takzvaných prototypů, které pochází z jiných světů a lze je vložit a libovolně používat.

2.4 Tvorba virtuálních modelů s pomocí modelovacích programů

Virtuální modely lze v dnešní době tvořit mnohem rychleji díky velkým výkonům výpočetní techniky. Pro tvorbu 3D modelů existuje mnoho programů. Ze softwarů jsou to například Solid Edge, SolidWorks, AutoCad 3D, Catia a několik dalších. Pro náročnější modelování virtuálních světů se používají náročnější a složitější aplikace například 3ds Max převážně pro filmové efekty nebo pro počítačové hry.

Jednou z variant je programování neboli tvoření virtuálních světů v jazyce VRML. Avšak tvořit složitější objekty přímo v tomto jazyce je náročné a neobratné. Proto je vhodné spojit jazyk VRML s programy, které jsem zmínil v předešlém odstavci. Například k vytvoření dutého válce s přesnou sílou stěny by ve VRML trvalo poměrně dlouho z důvodu potřeby definovat mnoho bodů a ty pak správně umístit a propojit.

Výhodou 3D CAD systémů je jednoduší tvorba složitějších objektů ve vektorové grafice, nevýhodou však zůstává objemný kód s mnoha vykreslovacími body při vykreslování v reálném čase. Objekty vytvořené CAD systémy jednoduše vyexportovat z těchto programů do jazyka VRML. Tento exportovaný model lze jednoduše vložit do jiného světa vytvořeném ve VRML a propojit např. s Matlabem.

Existují dvě možnosti, jak propojit exportovaný model do jiného souboru Virtuální Reality. První možností je zkopírováním bloku *Transform*, který je definicí objektu, a následným vložením textového kódu do jiného souboru. Druhou možností je pak použití externího prototypu, kde prototypem bude právě exportovaný objekt.

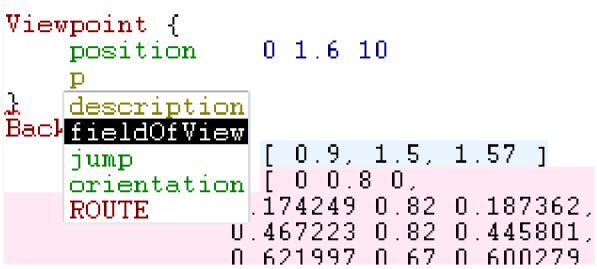
2.4.1 VRMLPad

VRMLPad je volně ke stažení jako 30ti denní licence na stránkách http://www.parallelgraphics.com/products/vrmlpad/download/

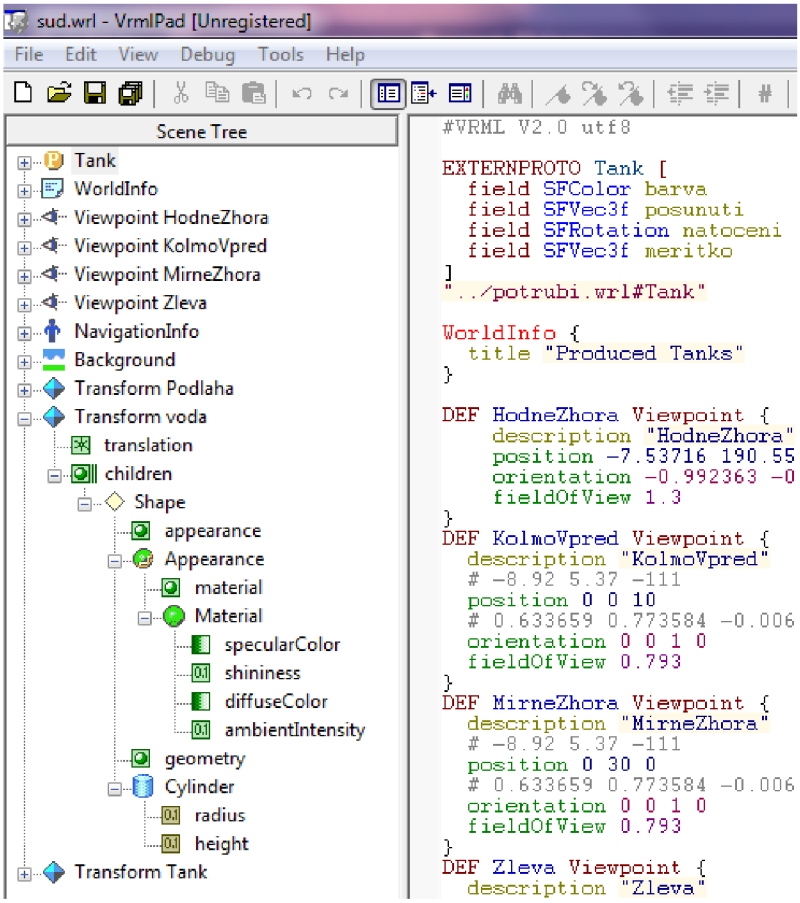
- Omezení volně dostupné zkušební licence:
- Nelze ukládat soubory větší než 64Kb
- Nelze kopírovat nebo vyřezávat výběry větší než 32Kb do schránky
- Debugovací sekvence je omezená na 2 minuty.
- Platnost zkušební licence je 30dní.

Textový editor VRMLPad je ideální pro tvorbu VRML souborů v textovém formátu. Editor VRML je rozdělen na 2 hlavní okna viz Obr. 2-5 Textový editor VRML Pad. V pravé části se nachází hlavní textová část, ve které se nachází kód naší virtuální reality a v levé části je strom, který se aktualizuje podle textu vpravo. Příkazy v Jazyce VRML jsou velice intuitivní, a struktura tvorby objektů je určena pravidly.

Pravidla jsou implementovaná do integrovaného našeptávače viz Obr. 2-4, který zobrazí uzly navazující na stávající strukturu VRML. Stačí napsat jen písmeno a vybrat požadovaný text.



Obr. 2-4 Našeptávač ve VRML Pad



Obr. 2-5 Textový editor VRML Pad

2.4.2 VRealm Builder

Další editor s názvem VRealm Builder pro jazyk VRML je programovací prostředí implementované přímo v prostředí Matlab, ale není součástí základního balíčku a je nutné si jej nejdříve doinstalovat. To dělám v příkazovém řádku přímo v Matlabu. Stačí tedy zavolat příkaz

>> vrinstall – install

a po dokončení tohoto příkazu pak jen zkontrolovat, zda je správně nainstalován

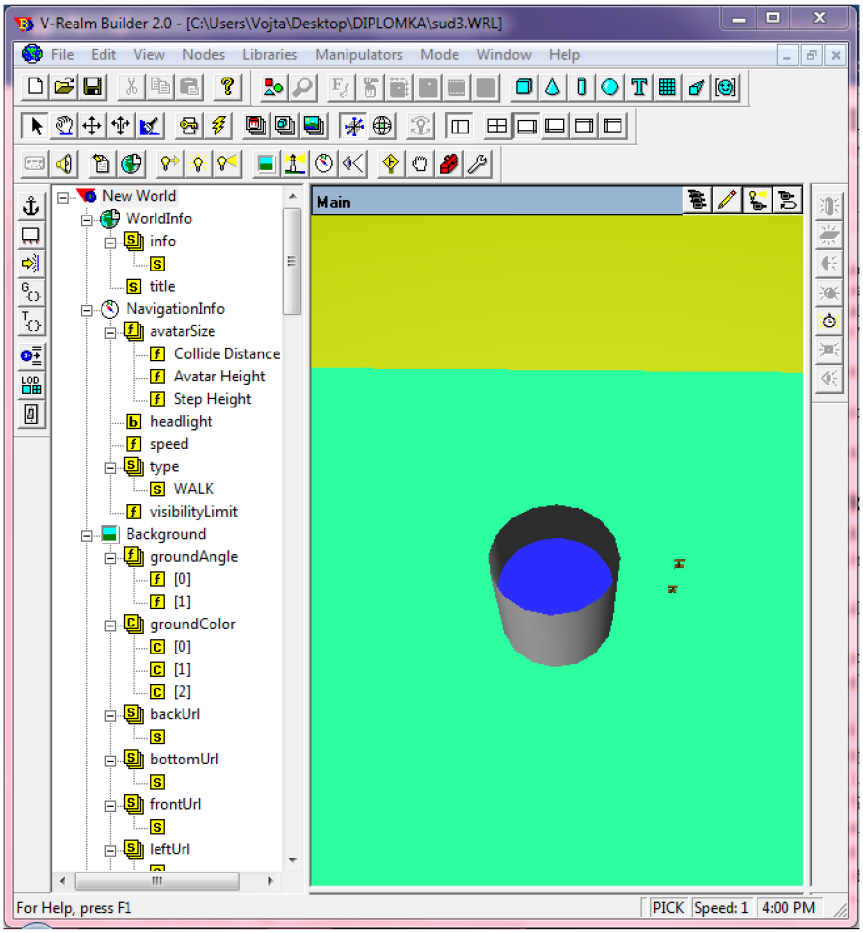
>> vrinstall -check

Pokud je vše v pořádku[1], nyní je nám umožněno pracovat s doinstalovaným balíčkem VRBuilder.

Obr. 2-6 je zobrazuje prostředí programu VRealm Builder a hlavní okno s vytvořeným virtuálním světem. Tento program je tvořen třemi základními částmi. Horní část editoru poskytuje příkazy VRML reprezentované ikonami a lištou se záložkami.

[1] Pokud nelze nastavit žádný parametr, tak je nutné VR Builder odinstalovat z PC a znovu jej nainstalovat.

Ikony jsou pro často užívané prvky. Kliknutím na ikonu se nám v části se stromovou strukturou vytvoří šablona odpovídající požadavku. Základní uzly používané pro tvorbu stromu se nachází v záložce *Nodes*. Editor nám zpřístupní jen uzly, které lze vložit do stávající struktury. Při ponechání kurzoru myši na kterékoli nezašedlé ikoně nám zobrazí název příkazu skrývajícího se pod ikonou.



Obr. 2-6 V Realm Builder

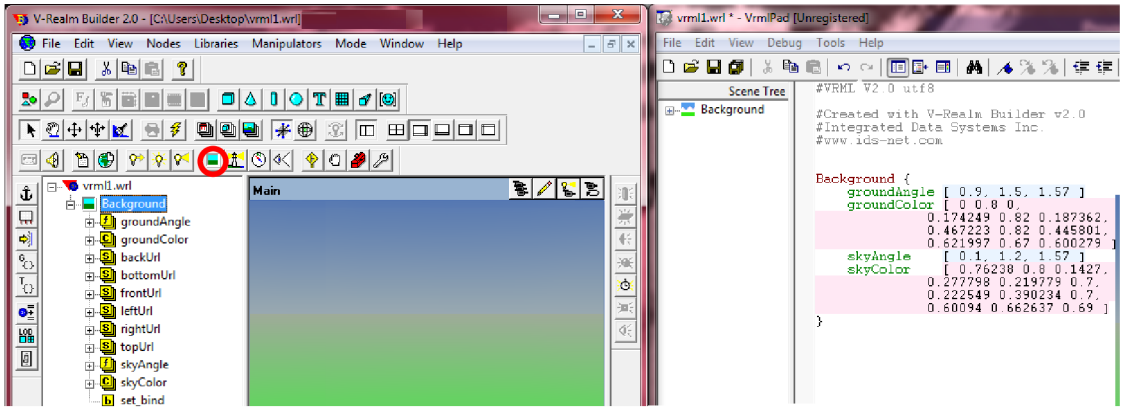
K tvorbě virtuálního světa je vhodné znát strukturu a chování uzlů, jež je vhodné propojovat. I když nám program omezí výběr zašednutím nepoužitelných uzlů, nemusí být dosaženo očekávaného výsledku.

2.5 Tvorba základních objektů pomocí nástrojů VRealm Builder a VRML Pad

V následující kapitole budou srovnány postupy tvorby základních prvků virtuálních scén pomocí dvou nástrojů s rozdílným přístupem. Srovnán bude textový editor VRML Pad a grafický nástroj VRealm Builder.

2.5.1 Pozadí

Pro tvorbu pozadí je na levé části na Obr. 2-7 je červeně zakroužkované tlačítko s názvem Background[2]. Jediné kliknutí nám vytvoří šablonu pozadí a naplní ji hodnotami základního rozložení barev modré a zelené, které lze libovolně měnit. Ve stromu vidíte vlastnosti groundColor, zobrazení země, a skyColor, barva pro nebe. Dále lze definovat zakřivení a další parametry. Na pravé části v textovém editoru jsou hodnoty barev nastaveny podle klíče RGB. Toto pozadí je tvaru koule, kde pohled pozorovatele je uvnitř koule. Prolínání barev v horizontu evokuje pohled do dálky. Pozadí tvoří pomyslnou bublinu okolo pozorovatele.



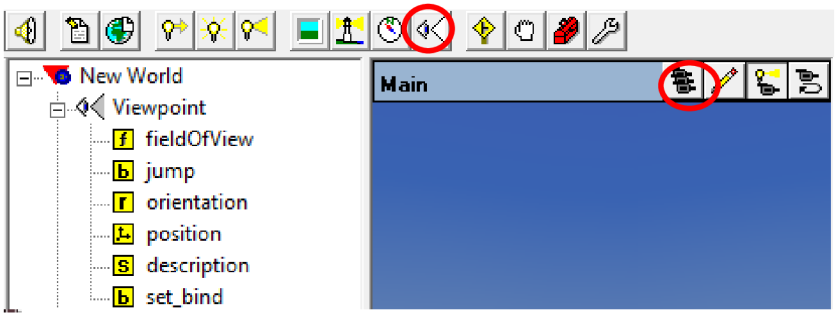
Obr. 2-7 Pozadí virtuálního světa

2.5.2 Výchozí bod pozorování

Lze definovat výchozí body pozorování (*Viewpoint* [2]) ve virtuální realitě podle Obr. 2-9. Tyto body jsou k dispozici v základním prohlížeči, který je implementován v Matlabu a je otevřen při spuštění simulace nebo při rozkliknutí VR Sink. Mezi těmito body lze rychle přecházet výběrem ze seznamu.

```
Viewpoint {
    position      0 1.6 10 }
```

Obr. 2-8 Viewpoit v textu



Obr. 2-9 Viewpoint ve VRealm Builder

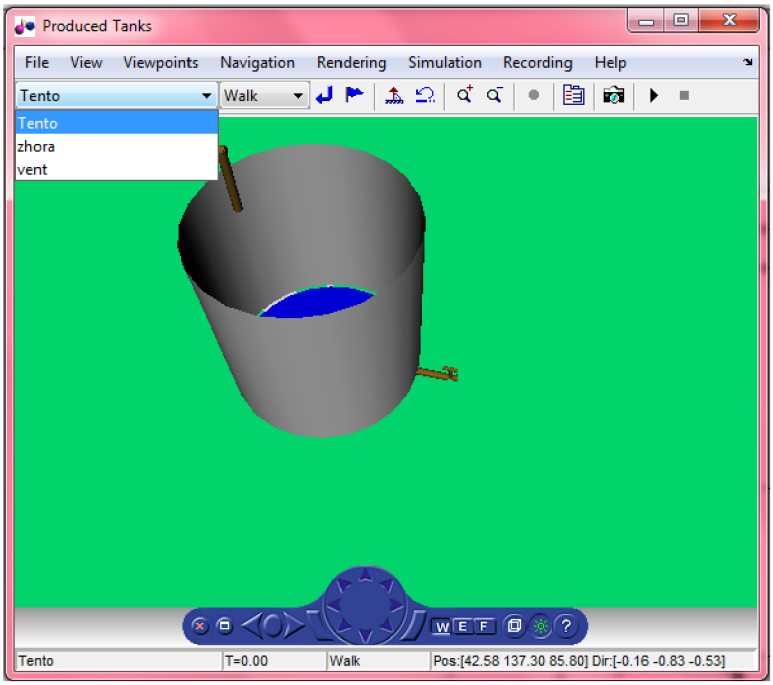
[2] Červeně zakroužkováno v obrázku

V levé části editoru jsou vlastnosti zobrazení, které lze nastavovat. Vlastnost fieldOfView je zorné pole, orientation je úhel pootočení pohledu, position je umístění v základním rozložení os x, y, z a Description je popis zobrazení. Pro přiřazení jména neboli definice pohledu stačí dvakrát klepnout na Viewpoint a přejmenovat jej. Tím se v textovém editoru přidá klíčové slovo DEF a název Viewpointu, který byl upraven nebo vytvořen.

Pozorovací body si lze nadefinovat libovolně buď přesnou polohou hodnot os X,Y,Z a úhlu nebo grafickou metodou, V editoru nastavím pohled jaký mi vyhovuje a ten si uložím do paměti kliknutím na tlačítko Viewpoint a tím se vytvoří nový pohled.

Tlačítko s kamerami otevře seznam existujících pozorovacích stanovišť, ze kterých lze vybrat a přesunout se na místo, které bylo vytvořeno a pojmenováno.

Lze vytvořit nový pohled na virtuální scénu v prohlížeči virtuální reality v Matlabu a uložit si tento pohled jako nový. Pomocí vlajky lze uložit aktuální pohled ve VR do souboru VRML, jenž prohlížíme.



Obr. 2-10 Zobrazený systém v Matlab

Na Obr. 2-10 je zobrazen prohlížeč virtuální reality v programu Matlab/Simulink. Ve výběru v levém horním rohu jsou k dispozici 3 pohledy, které byly vytvořeny.

Body usnadňují orientaci v prostoru, pokud se pozorovatel odchýlí od vytvořeného objektu, stačí se pak vybrat z vytvořených „Viewpointů“ požadovaný pohled.

2.5.3 Transform

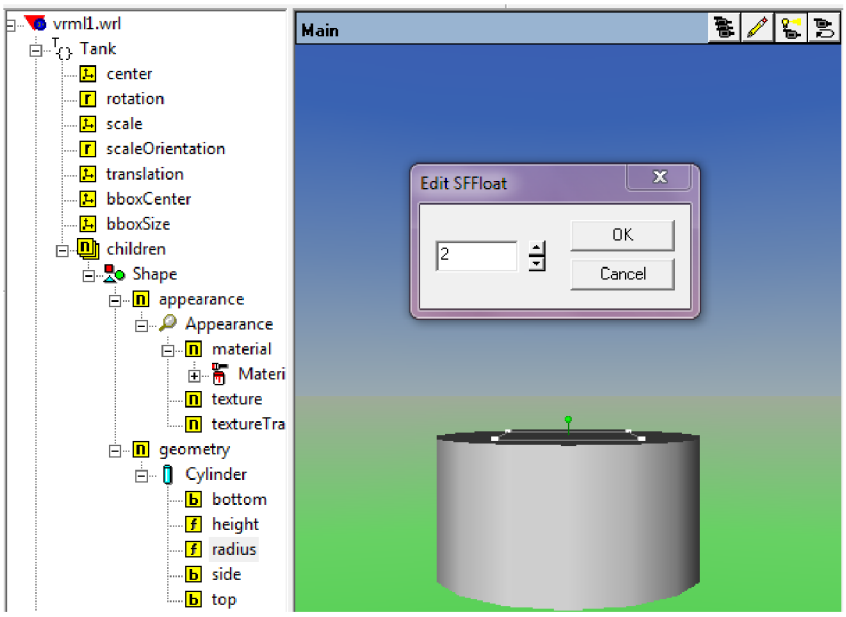
Abychom mohli vytvořit nějaký prvek ve 3D prostoru je nutné vytvořit základní strukturu pro jazyk VRML. Nejdůležitější je objekt Transform, protože jenom tyto prvky lze propojit s dynamikou (např. v Simulinku).

```
Transform {
    children Shape {
        appearance  Appearance {
            material    Material {
            }

        }

        geometry    Cylinder {
        }

    }
}
```

Obr. 2-11 Základní struktura objektu Transform

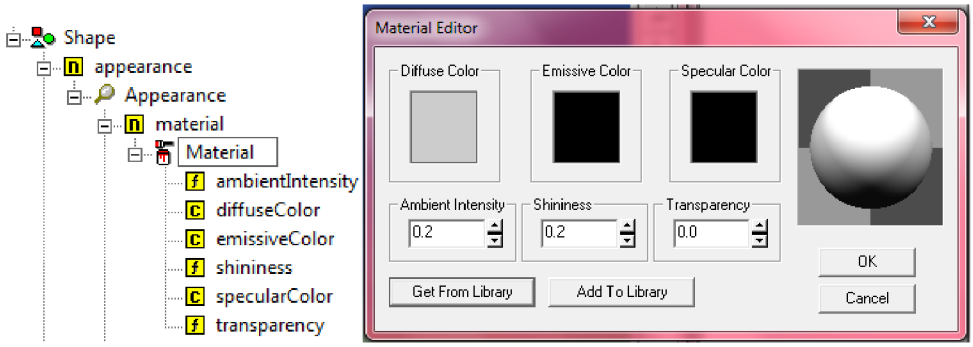
Pod definicí Transform v první závorce viz Obr. 2-1 jsou vlastnosti ovladatelné ze simulinku, cokoliv je dále za slovem Shape je zamčené a nelze je modifikovat přímo ze simulátoru matlabu. To znamená, že nelze jednoduše měnit velikost vytvořeného objektu (např. výšku válce). Je nutné však dodržovat následnost uzlů, tak jak jsou uvedeny na již zmíněném obrázku. Prvek Transform obsahuje vlastnosti, které je možné měnit, například posunutí, otočení, měřítko a další, také k němu patří uzel children. V uzlu children může být umístěn další prvek Transform, který lze opět například rotovat v jiné ose. Rotování probíhá vždy kolem bodu v uzlu translation. V uzlu children je definice pro uzel Shape. Na uzel Shape jsou vázány uzly appearance a geometry. V uzlu geometry je pak povoleno vložit tvary jako jsou válce, kužely, koule, čtverce a jiné. V uzlu appearance lze definovat materiál nebo textury a barvy. Ve VRealm Builder jsou dostupné knihovny s paletou před chystaných materiálů, textur a barev pro libovolné použití. Lze si vytvořit i vlastní kombinace barev nebo textur nebo je vkládat i externě ze souboru nebo také z internetu. Vkládání externích textur se provádí pomocí URL odkazů.

Ve větvi geometry definuji geometrický tvar Pokud tedy vytvořím válec, mohu upravovat parametry jako je výška válce (*height*), poloměr (*radius*), parametry *top* a *bottom* jsou čela válce a jsou boolovského typu (*TRUE/FALSE*). Objekty jsou duté a tvořeny pouze obálkou o velmi tenké vrstvě.

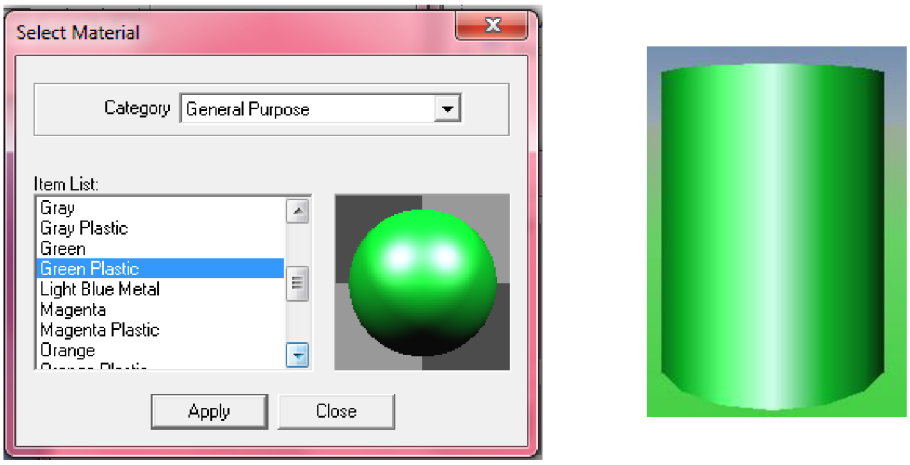


Obr. 2-12 Nastavení poloměru válce

Dvojklikem na kolonku Material se nám zobrazí Material editor, kde lze nastavovat barvy objektu. Zde opět lze využít knihovnu tlačítkem Get from Library a vybrat barvy z knihovny jak je ukázáno na a Obr. 2-1.



Obr. 2-13 Nastavení barvy objektu



Obr. 2-14 Výběr Barvy z knihovny

Na následujícím je popis objektu Transform v textovém editoru VRMLPad. Transparency je nastavení průhlednosti objektu v rozsahu 0 až 1, kde 0 značí neprůhlednost objektu a 1 je naprosto průhledný (neviditelný).

```
DEF Tank Transform {
    children Shape {
        appearance  Appearance {
            material    Material {
                ambientIntensity    0.1
                diffuseColor    0.0223017 0.8 0.1394
                emissiveColor   0 0 0
                shininess   0.2
                specularColor   1 1 1
                transparency    0
            }

        }

        geometry    Cylinder {
            height  5
            radius  2
        }

    }
}
```

Obr. 2-15 nastavení Transform v Textovém Editoru

2.5.4 Další objekty

Ostatní tvary jako jsou koule, válce, čtverce, a další prvky patřící pod objekt *Transform* jsou součástí editorů a je zbytečné popisovat každý zvlášť. Jediným rozdílem je, že v části geometry budou jiné parametry a s tím související vlastnosti konkrétních tvarů. Oba editory nám pomohou tyto prvky vytvořit. Složitější prvky z důvodu úspory času je výhodné tvořit v programech typu CAD a jejich exporty vložit do virtuální reality. To však nese s sebou větší náročnost na výpočetní výkon a složitost kódu.

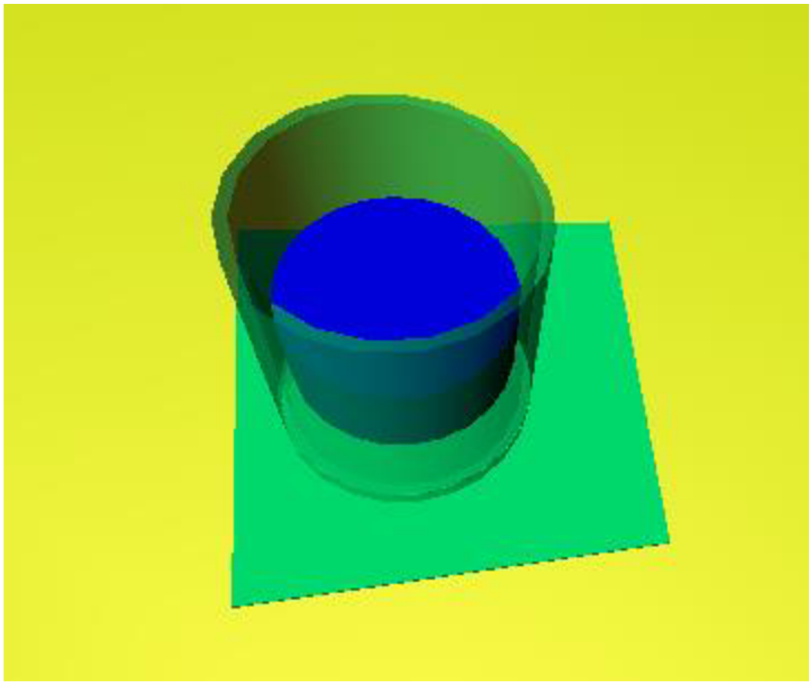
3. NÁVRH 3D MODELU TECHNOLOGICKÉHO ZAŘÍZENÍ - TANKU

Modelovaným technologickým zařízením bylo zvoleno napouštění a vypouštění tanku, pro získání zkušeností s tvorbou a propojením 3D Modelů v jazyce VRML a Matlab/Simulink.

Pro srovnání modelů tvořených pomocí CAD systému a klasického VRML byly vytvořeny dva modely, kde jeden složitější a výpočetně náročnější byl vytvořen v programu SoliEdge, a druhý model byl vytvořen v programu VRealm Builder.

Model tanku je tvořen dutým válcem vytvořeným v programu SolidEdge a ten exportován do jazyka na VRML. Z exportovaného souboru byl vyjmut prvek *Transform* představující válec a vložen do editoru VRMLPad. K válci byly připojeny další prvky *Transform* tak, aby byl vytvořen tank, který může být následně simulován. Výhodou využití CAD systémů je možnost tvořit reálnější a vizuálně podobnější realitě.

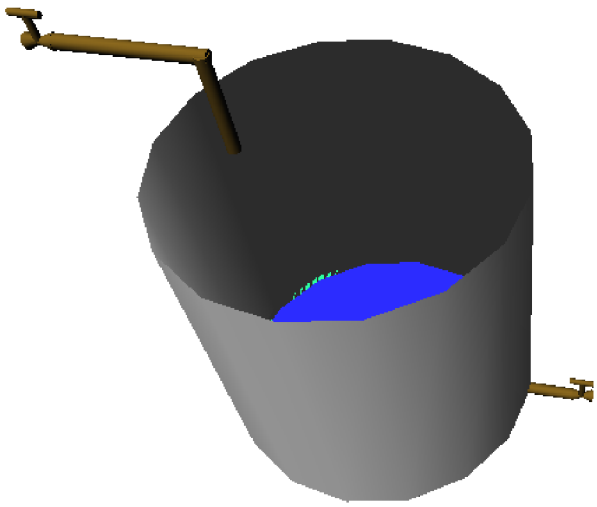
Nádoba je z hora otevřená a částečně průhledná pro viditelnost aktuálního objemu vody v nádrži. Nádoba by mohla být uzavřená s bezpečnostním ventilem proti přetlaku a podtlaku, aby nedošlo ke zničení nebo deformaci nádrže. Ve spodní části je dutý válec opatřen dnem.



Obr. 3-1 Návrh Vizualizace Tanku se Solid Edge

Jednodušší a méně graficky náročný tank je vytvořen v programu VRealm Builder bez použití CAD systému. Výhodou tohoto jednoduchého modelu je rychlost při vykreslování v reálném čase. Protože obsahuje jednoduchou grafiku pro vykreslování.

Nároky na vykreslování jsou minimální, nehrozí tedy zpoždění při běhu budoucí simulace v reálném čase a zdrojový kód virtuálního modelu bude krátký. Nevýhodou je nepřesná vizualizace jen přibližná podoba s reálným objektem.



Obr. 3-2 Návrh zjednodušeného modelu tanku

Pro model byl vytvořen ventil, kterému v simulaci lze měnit polohu uzávěru o 90°. Ventil je tvořen ze dvou prvků Transform. Prvním prvkem je ovládací kříž u přítoku pojmenovaný jako „Madlo_pritok“ a u druhého s názvem „Madlo_odtok“. Oba tyto prvky jsou oddělené, od ventilu abychom mohli měnit jejich polohu. Druhým prvkem Transform je tělo ventilu. Tyto ventily jsou v modelu dva, jeden pro odtok a jeden pro přítok. Jak je vidět na obrázku výše.

Obr. 3-3 Návrh ovládacího ventilu

4. KNIHOVNA 3D ANIMATION V PROSTŘEDÍ MATLAB/SIMULINK

Pro představení knihovny 3D Animation uvedu hlavní bloky používané při práci s převodem dat do virtuálních světů.

4.1 VR Sink

VR Sink umožnuje vstup signálů ze simulace v simulinku v prostředí Matlab do virtuálního světa vytvořeného ve VRML jazyce. Tento blok slouží pro ovládání virtuálního světa pomocí programu Matlab. Blok VR Sink zapíše hodnoty z jeho portů do polí virtuálních světů zadaných v dialogovém okně Parametry bloku. Po vložení bloku VR Sink na něj poklepejte, otevře se okno parametry viz .

VR Sink je ekvivalentní bloku VR To Video, s tím rozdílem, že parametr Show video output port pro blok VR Sink je ve výchozím nastavení vymazán. Blok VR Sink nemůže být kompilován softwarem Simulink Coder ™, ale může být použit jako zařízení pro zobrazení virtuálního světa v hostitelském počítači. Můžete jej zahrnout do modelů, které kompilujete pomocí softwaru Simulink Coder.

Do kolonky *Source File* připojím soubor s vytvořeným kódem v jazyce VRML - název souboru určující virtuální svět, který se připojí k tomuto bloku. Ve výchozím nastavení se v tomto textovém poli zobrazí úplná cesta k přidruženému 3D souboru ve virtuálním světě. Pokud v tomto poli zadáte pouze název souboru, software předpokládá, že soubor 3D virtuálního světa se nachází ve stejné složce jako soubor modelu. Můžete zadat soubor VRML nebo soubor X3D. Klepnutím na tlačítko Nový otevřete prázdný výchozí editor virtuálního světa. Když zadáte název zdrojového souboru nebo použijete tlačítko Procházet, tlačítko Nový se změní na tlačítko Upravit. Klepnutím na tlačítko Edit spustíte výchozí editor virtuálního světa se otevřeným zdrojovým souborem. Kliknutím na tlačítko View zobrazíte svět v prohlížeči Simulink 3D Animation Viewer nebo ve webovém prohlížeči. Klepnutím na tlačítko Reload znovu načte svět po jeho změně.

Automaticky otevřít prohlížeč - Pokud toto políčko zaškrtnete, zobrazí se výchozí prohlížeč virtuálního světa po načtení modelu Matlab/Simulink.

Povolit prohlížení z Internetu - Pokud zaškrtnete toto políčko, virtuální svět je přístupný pro zobrazení v klientském počítači. Pokud nezaškrtnete toto políčko, svět je viditelný pouze v hostitelském počítači. Tento parametr je ekvivalentní vlastnosti RemoteView objektu vrworld.

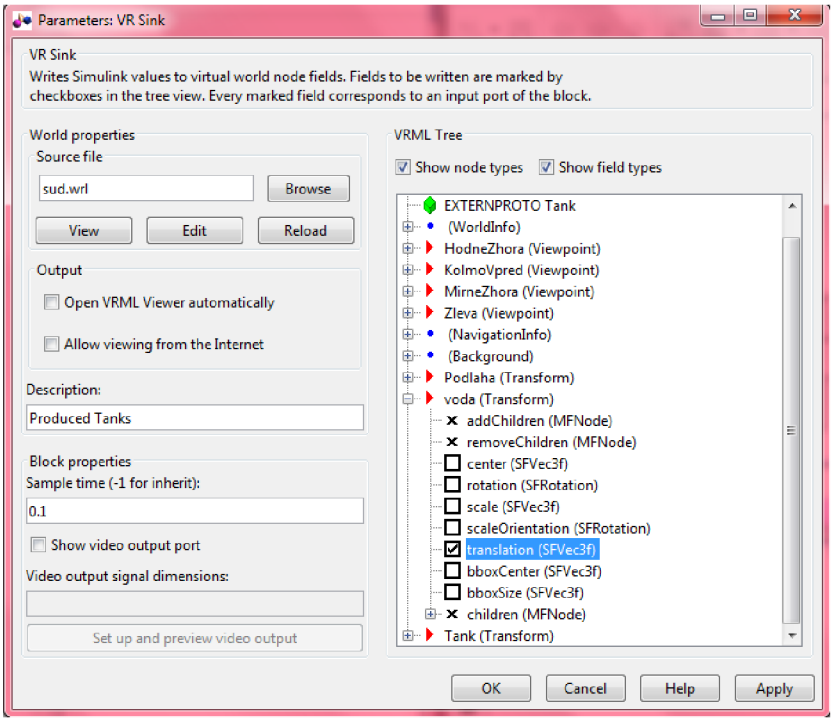
Description - Popis, který se zobrazuje ve všech výkazech objektů virtuální reality, v záhlaví prohlížeče Simulink 3D Animation Viewer.

Sample Time - Doba vzorkování zde zadejte vzorkovací čas nebo -1 pro zděděný vzorkovací čas. Pro dosažení hladké simulace MathWorks® doporučuje, abyste explicitně nastavili parametr Time sample. Hodnotu tohoto parametru můžete změnit.

Show video output port - Umožňuje portu výstup video streamu RGB pro další 2D video zpracování. Tento prvek je jediným rozdílem od VR to Video

VRML Tree - Toto pole zobrazuje strukturu virtuálního světového 3D souboru a virtuálního světa samotného. Uzly (nodes), které mají jména, jsou označeny červenými šipkami. K nim můžete přistupovat z rozhraní Simulink 3D Animation. Uzly bez jmen, ale jejichž děti jsou pojmenovány, jsou také označeny červenými šipkami. Tato schéma značení umožňuje vyhledání všech přístupných uzlů pomocí stromů pomocí šipky. Ostatní uzly mají před jmény modrou tečku. Pole s hodnotami, které jste nastavili, mají zaškrtávací políčka. Pomocí těchto políček vyberte políčka, jejichž hodnoty chcete aktualizovat software Simulink. Pro každé pole, které vyberete, je v bloku vytvořen vstupní port. Vstupní porty jsou přiřazeny vybraným uzlům a polím v pořadí, které odpovídá 3D souboru virtuálního světa. Pole, jejichž hodnoty nelze zapsat (protože jejich nadřazené uzly nemají jména nebo protože nejsou datovou třídou virtuálního světa eventIn nebo exposedField) mají ikonu ve tvaru "X".

Zobrazit typy uzlů (Show node types) - Pokud zaškrtnete toto políčko, typy uzlu se zobrazí ve stromu virtuálních scén. Zobrazení typů polí - Pokud zaškrtnete toto políčko, ve stromu virtuální scény se zobrazí typy polí.



Obr. 4-1 VR Sink

4.2 VR Source

Použijte zdrojový blok VR pro poskytnutí interaktivity mezi uživatelem navigujícím se ve virtuálním světě a simulací modelu Simulink. VR zdrojový blok registruje

uživatelské interakce s virtuálním světem a předá tato data modelu, což ovlivňuje simulaci modelu. Zdroj VR čte hodnoty z polí virtuálního světa zadané v dialogovém okně Parametry bloku a zadává jejich hodnoty modelu.

Pomocí VR Source lze získat data z virtuálního světa do modelu Matlab/Simulink, Pomocí dat senzorů z virtuálního světa můžete řídit simulaci. Tímto způsobem můžeme zajistit interaktivitu mezi navigací uživatelů, interakcí ve virtuálním světě a simulací modelu. Simulace bude reagovat na události virtuálního světa, jako jsou časové události nebo výstupy ze skriptů. Lze využít statické informace z virtuálního světa, například velikost krabice, pro ovládání simulace.

Můžete například zadat požadované hodnoty ve virtuálním světě tak, aby uživatel mohl určit umístění virtuálního světového objektu interaktivně. Simulace pak reaguje na změněné umístění objektu.

Ve výchozím nastavení blok VR Source neumožňuje signály s proměnnou velikostí. Pokud tento parametr povolíte, pak blok VR Source umožňuje proměnné velikosti signálů pro pole, která mohou během simulace měnit rozměry. Tato pole obsahují pole MFxxx, které mohou mít různý počet prvků (obvykle MFFloat nebo MFVec3f). SFImage je jediné pole SFxxx, které může mapovat signál s proměnnou velikostí. Podrobnosti o těchto typech dat naleznete v části Typy datových polí.

Uzly, které mají jména, jsou označeny červenými šipkami. K nim můžete přistupovat z rozhraní MATLAB®. Uzly bez jmen, ale jejichž děti jsou pojmenovány, jsou také označeny červenými šipkami. Toto schéma značení umožňuje vyhledání všech přístupných uzlů pomocí šipky. Ostatní uzly mají před jmény modrou tečku.

Políčka s čitelnými hodnotami mají zaškrtávací políčka. Pomocí těchto políček vyberte políčka, která chcete, aby software Simulink sledoval a použil k zadávání hodnot. Pro každé pole, které vyberete v poli Virtual Tree, vytvoří Simulink výstupní port v bloku VR Source. Simulink vytvoří výstupní porty ve stejném pořadí, ve kterém se vybraná pole objeví ve 3D souboru virtuálního světa.

4.3 VR Signal Expander

Blok VR Signal Expander vytvoří vektor s předdefinovanou délkou, pomocí některých hodnot ze vstupních portů a vyplnění zbytku s hodnotami signalizačního symbolu. Blok vysílače signálu VR přijímá a vydává signály typu double. Output width - Jak dlouhý by měl být výstupní vektor. Output signal indices - Vektor udávající polohu, na které se na výstupu objevují vstupní signály. Zbývající pozice jsou vyplněny signály VR Placeholder. Předpokládejme například, že chcete mít vstupní vektor se dvěma signály a výstupní vektor se čtyřmi signály, první vstupní signál v poloze 2 a druhý vstupní signál v pozici 4. V poli Output width zadejte 4 a v Output signal indices zadejte [2,4]. První a třetí výstupní signály nejsou blíže specifikovány a budou vyplněny podle základní definice ve VRML souboru.

4.4 VR Placeholder

Blok vysílá speciální hodnotu, která je interpretována jako "nespecifikovaná" blokem VR Sink. Když se tato hodnota objeví na vstupu VR Sink, ať již jako jediná hodnota nebo jako prvek vektoru, zůstane příslušná hodnota ve virtuálním světě nezměněna. Použijte tento blok pro změnu pouze jedné hodnoty z většího vektoru. Například pomocí tohoto bloku změníte pouze jednu souřadnici z 3-D pozice.

Hodnota výstupu bloku zástupného místa VR by neměla být modifikována před použitím v jiných blocích VR. Blok VR Placeholder vysílá signály typu double. Output Width - Délka vektoru obsahující hodnoty umístění.

4.5 VR Tracer

Trasování trajektorie objektu v přidružené virtuální scéně. Tento blok vytváří uzly značek v pravidelných časových krocích buď jako children zadaného nadřazeného uzlu (parametr nadřazeného uzlu) nebo v horní úrovni hierarchie scény (root).

Můžete zadat jeden ze tří typů značek:

- *Obecný tvar*
- *Řádky segmentů spojují polohy objektů v každém kroku*
- *Osy třídilé triády pro orientaci trajektorie v 3D prostoru*

Také můžete promítat sledované polohy objektů do roviny nebo do bodu. Vstup umístění objektu musí odpovídat umístění objektu v hierarchii scény. Pokud je sledovaný objekt umístěn jako podřízený objekt, definujte název nadřazeného objektu DEF v poli nadřazeného uzlu. Pokud je sledovaný objekt umístěn v horní části hierarchie scény (jeho poloha je definována v globálních souřadnicích scény), ponechte toto pole prázdné.

První vektorový vstup určuje polohu značky. Druhý vstup (pokud je aktivován parametrem pro výběr barvy značky) představuje barvu značky. Druhý nebo třetí blokový vstupní vektor (v závislosti na tom, zda je vektorový barevný vstupní vektor povolen) určuje souřadnice bodu projektu.

4.6 Real Time Synchronization

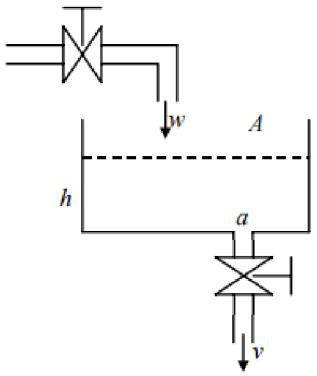
synchronizace s reálným časem umožňuje simulaci v chodu v reálném čase, tedy synchronizovaný s hodinami v PC. Bez této synchronizace bychom nemohli děj simulovat se vzdáleným zařízením, ale proběhl by výpočet simulace v co nejkratším čase. Tedy podle výpočetního výkonu procesoru v PC.

5. FYZIKÁLNÍ MODEL TANKU

Volím jednoduchý model tanku pro demonstrační úlohu v laboratořích. Uvažuji model 1m^3 vody a ohřev maximálně 100°C reálně však budu ohřívat například do 80°C. Pro zjednodušení v modelu zanedbám některé fyzikální děje a vlastnosti, které se projeví v reálném systému, avšak podstatu problému nám pomůže tato demonstrační úloha přiblížit. Soustava má velký objem, a proto bude časová konstanta vody v nádrži extrémně velká. Výkon otopného systému bude 80kW. Pro měření výšky hladiny jsem uvažoval ultrazvukový senzor pro měření vzdálenosti. Pro měření teploty postačí klasický senzor s Pt100 s dostatečným rozsahem teploty (-50°C až 150°C). V modelu teploty zanedbávám časovou konstantu teploměru a také časovou konstantu topného tělesa, vzhledem z velké hodnotě časové konstanty tvořenou objemem tekutiny v nádrži, si tyto malé časové konstanty dovolím zanedbat. Dalším elementem jsou ztráty do okolí, tyto ztráty volím na 200[Wm^{-3} K^{-1}] aby děj ochlazování byl rychlejší a viditelnější. Vzhledem k velké časové konstantě objemu vody a omezenému výkonu otopného systému, musím vzít v úvahu poměrně dlouhou dobu pro dosažení požadované teploty. Z tohoto důvodu by bylo potřeba systém změnit systém tak, aby regulační děj proběhl mnohem rychleji a úloha by byla realizovatelná maximálně v několika minutách.

5.1 Model výšky hladiny

Fyzikální model je navržen pro jednoduchou aplikaci [3] napouštění a vypouštění tanku, tento fyzikální model řeší množství přítoku a odtoku a fyzikální chování.

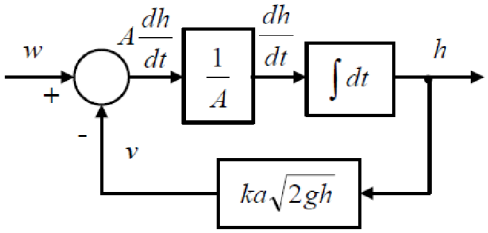


Obr. 5-1 Nákres modelu výšky hladiny [3]

Je to systémem prvního řádu, a proto obsahuje jeden integrátor. Podle fyzikálních zákonů se tlak na výstupu mění s výškou hladiny *h*. Přítok i odtok může být otevírán nezávisle na sobě změnou pootevření uzávěru. Uvažovaným médiem bude pitná voda.

Na Obr. 5-1 je uveden nákres tanku se dvěma ventily a s popisem důležitých veličin, kde *h* je výška hladiny a *A* je plocha hladiny vody v nádrži, která je konstantní, *a* je plocha odtoku, *w* je objem přítoku a *v* je objem vody, která odtekla.

Blokové schéma na Obr. **5-2** popisuje chování systému na množství vody v systému. Výstupem je výška hladiny v nádrži. Vstupem je přítok *w* do nádrže, od které je odečteno množství vody *v* odtékající ventilem. Ve zpětné vazbě je změna množství odtékající vody, která je závislá na změně otevření ventilu a výšce hladiny. Je to schéma pro změnu hydrostatického tlaku.



Obr. 5-2 **Blokové schéma [3]**

Pro výpočet simulace využívám bilanční rovnici (5.1)

$$\frac{dV}{dt} = A\frac{dh}{dt} = w - f(h) = w - \ ka\sqrt{2 \cdot g \cdot h} = w - ka\sqrt{2 \cdot 9.81 \cdot h} \tag{5.1}$$

w je množství přítoku [m^3sec^{-1}], *v* je množství odtoku [m^3sec^{-1}], *k* je hydrodynamický součinitel vody 0,94. Rovnice (5.2) je výpočet plochy hladiny A. Výpočet průřezu odtoku z nádrže je v rovnici (5.3). Průřez odtoku je však proměnný, proto je výpočet pro maximální možný nenucený odtok

$$A = \pi \cdot r_{hladina}^2 = \ \pi \cdot 0.5^2 = 0{,}7853\ m^2 \tag{5.2}$$

$$a = \pi \cdot r_{odtok}^2 = \ \pi \cdot 0{,}1^2 = 0{,}031415\ m^2 \tag{5.3}$$

5.2 Model teploty

Fyzikální model teploty vychází z kalorimetrické rovnice Rov. (5.4) [8] pro přenos tepla mezi látkami. V tomto případě uvažuji systém prvního řádu, a to přenos tepla mezi tankem a okolím, protože je to rozměrná nádoba a časová konstanta ohřevu vody v nádrži je mnohonásobně větší než doba ohřevu topného tělesa, proto jej v tomto případě neuvažuji.

$$m_V \cdot c_V \cdot \frac{d\vartheta_V(t)}{dt} + K_V \cdot \vartheta(t) = p(t) \tag{5.4}$$

Kde m_V *[kg]* je hmotnost vody, c_V *[Ws·kg^{-1}·°C^{-1}]* je specifické teplo vody, K_V je*[W·°C^{-1}]* koeficient přestupu tepla neboli tepelné ztráty, a *p*(t) *[W]* je příkon.

Pak platí pro časovou konstantu T Rov. (5.5) [8]

$$T = \frac{m_V \cdot c_V}{K_V}\ [sec] \tag{5.5}$$

A pro zesílení K Rov. (5.6) [8]

$$K = \frac{1}{K_V}\ [°C/W] \tag{5.6}$$

Po dosazení pak platí pro výpočet zesílení soustavy K_{tanku}:

$$K = \frac{1}{K_V} = \frac{1}{200} = 0{,}002\ [C/W] \tag{5.7}$$

Po dosazení pak platí pro výpočet zesílení soustavy T_{tanku}:

$$T = \frac{m_V \cdot c_V}{K_V} = \frac{1000 \cdot 4180}{200} = 20900\ [sec] \tag{5.8}$$

Pro přenos soustavy tanku vody platí přenos FS:

$$Fs = \frac{K_{tank}}{Tp + 1} = \frac{0{,}005}{20900p + 1} \tag{5.9}$$

Pokud přidám k soustavě ještě hořák se jmenovitým maximálním výkonem 80kW je jeho zesílení $K_{hořák}$ = 80000. tuto hodnotu je nutné podělit hodnotou 0,01, protože hodnota akčního zásahu, kterou v modelu uvažuji není 0-1 ale 0-100%.

Pak kompletní soustava tanku včetně hořáku bude mít přenos podle vzorce 5.10

$$Fs_{celk} = \frac{K_{hořák} \cdot K_V}{Tp + 1} = \frac{0{,}01 \cdot 80000 \cdot 0{,}005}{20900p + 1} = \frac{4}{20900p + 1} \tag{5.10}$$

Pro tuto soustavu lze navrhnout regulátor.

5.3 Návrh regulátoru pro model teploty.

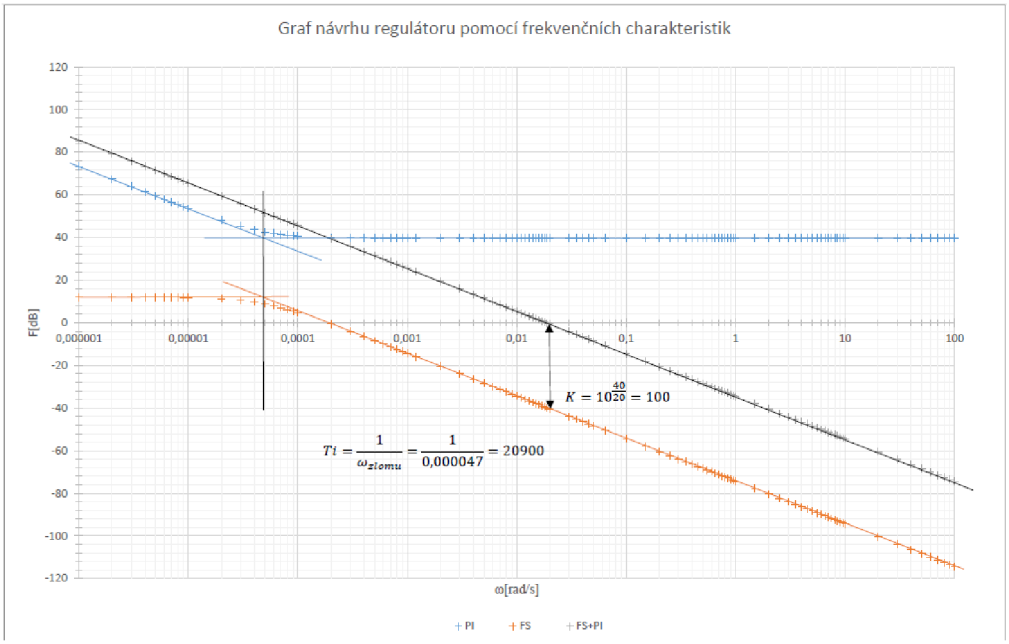
Pro návrh regulátoru podle amplitudové frekvenční charakteristiky převedl operátorový přenos soustavy do frekvenčního přenosu. Z tohoto přenosu byl úpravou do logaritmických souřadnic získán přenos soustavy. *[7]*

$$Fs_{db} = 20 \cdot \log(K) - 20 \cdot \log(\sqrt{T^2 \cdot \omega^2 + 1^2}) \tag{5.11}$$

Po dosazení:

$$Fs_{db} = 20 \cdot \log(4) - 20 \cdot \log(\sqrt{20900^2 \cdot \omega^2 + 1^2}) \tag{5.12}$$

Z této rovnice je sestavena do grafu charakteristika soustavy a k ní navržen regulátor. Vhodným regulátorem bude PI s časovou konstantou stejnou jako má soustava. A zesílením K=100.



Obr. 5-3 Graf návrhu regulátoru LAFCH

V grafu na Obr. 5-3 Graf návrhu regulátoru LAFCH vidím, že soustava s navrženým regulátorem se chová jako ideální integrátor omega řezu tedy mohu posunout libovolně daleko, ta je požadována co nejrychlejší, systém má omezení akčního zásahu, ten nebude nikdy větší než 100% maximálního výkonu akčního členu. Kvůli tomuto omezení nemá smysl nastavovat zesílení regulátoru příliš velké, a časová konstanta regulátoru by měla být přiměřená tak, abychom zajistili sklon 20db/dekádu alespoň 1 dekádu na každou stranu od omega řezu.

Pro PI regulátor platí

$$F_{PI} = r_0 + \frac{ri}{p} = K\left(1 + \frac{1}{Tip}\right) = P\left(1 + \frac{I}{p}\right) = k_r \frac{(T_r p + 1)}{p} \tag{5.13}$$

Z grafu zjistím hodnoty časové konstanty Ti a zesílení K

$$Ti = \frac{1}{\omega_{zlomu}} = \frac{1}{0{,}000047} = 20900s, \qquad K = 10^{\frac{40}{20}} = 100,$$
$$k_r = \frac{K}{Ti}$$

Po dosazení

$$F_{PI} = 100\left(1 + \frac{1}{20900p}\right) = 100\left(1 + \frac{0{,}000047}{p}\right) = 0{,}0047\left(\frac{20900p + 1}{p}\right) \tag{5.14}$$

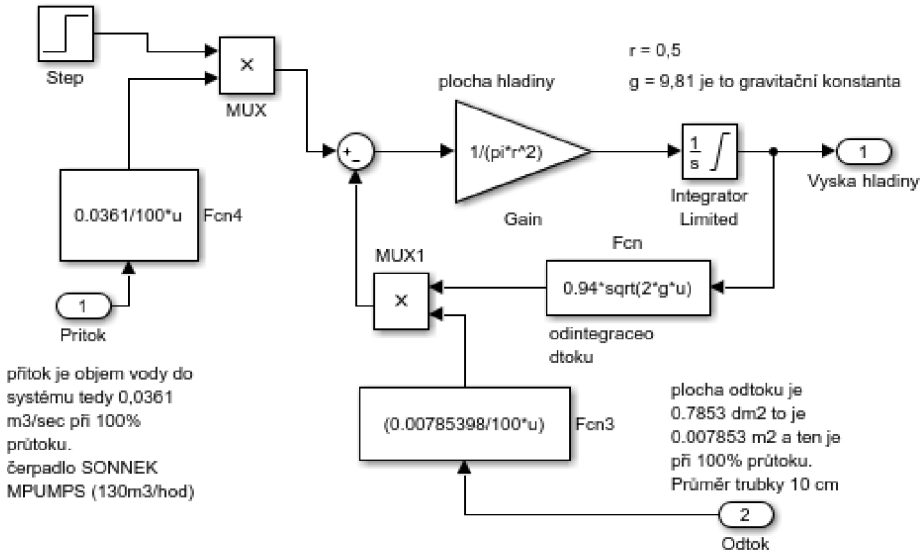
Tento navržený regulátor pak mohu implementovat do Matlab/Simulink i PLC.

6. REALIZACE FYZIKÁLNÍHO MODELU V PROSTŘEDÍ MATLAB/SIMULINK

K tvorbě modelu a jeho testování je využit program Matlab/Simulink verze 2017a. K realizaci modelu je nepostradatelnou součástí užití základních knihovních funkcí, které jsou integrované v programové knihovně.

6.1 Model výšky hladiny

Model nádrže vychází z hydrodynamické rovnice. Výstupem systému je výška hladiny, dále se na soustavu dívám jako na setrvačný článek prvního řádu. Proto obsahuje jeden integrátor. Vstupem systému je přítok a odtok. Pro přítok jsem použil čerpadlo od firmy SONNEK z kategorie M – PUMP, Toto čerpadlo je schopno dodat maximální průtok 130 m^3hod^{-1} při maximální teplotě 90°C. Přítok je určen otevřením přítokového ventilu, který je schopen do systému dodat 0,0361 m^3sec^{-1} což je 130 m^3hod^{-1} a maximální průřez odtoku, je 0,785 dm^2 to odpovídá průměru odtoku 10 cm. Pro ovládání využívám hodnot veličin napouštěcího a vypouštěcího ventilu. Oba ventily pracují v rozsahu 0-100% maxima. Uvažuji, že ventily mají lineární charakteristiku. Avšak reálný ventil lineární charakteristiku nemá.



Obr. 6-1 Model nádrže v MATLAB

Funkce Fcn4 slouží pro normalizaci přítoku do nádrže, ten je nastaven v maximální hodnotě na maximální průtok 130 m^3/hodinu to je: 0,0361 m^3sec^{-1}

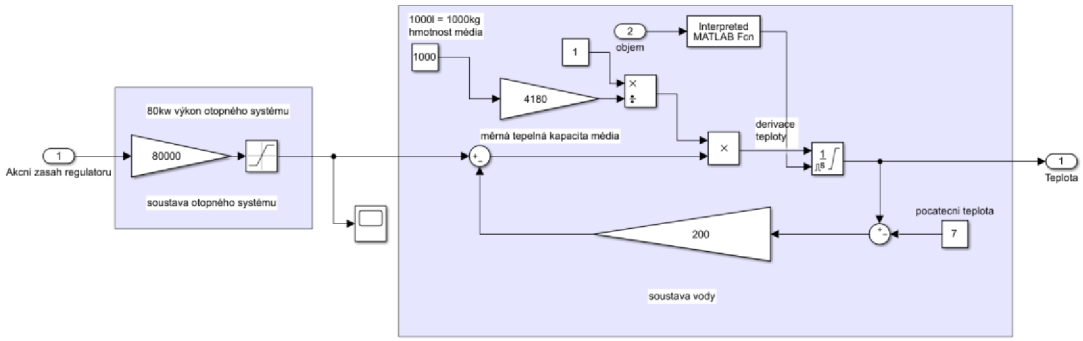
$$přítok\ w = \frac{\max přítok\ [m^3 \mathrm{sec}^{-1}]}{100} \cdot procenta \tag{6.2}$$

Funkce Fcn3 pro normalizaci průřezu odtoku z nádrže. Maximální hodnota odtoku je 0,785 dm^2 to je: 0,00785m^2

$$průřez\ odtoku\ a\ = \frac{0{,}00785[m^2]}{100} \cdot procenta \tag{6.3}$$

6.2 Model teploty

Pro navržený model tanku, je vhodné ještě zavést model teploty, jak již byl navržen dříve. Kalorimetrickou rovnici jsem implementoval do programu Matlab/Simulink viz Obr. 6-2 kde objem nádrže je stanoven na 1000 litrů. Předpokládám, že ohřev bude probíhat pouze, když bude nádoba plná, protože pokud by se měnil objem vody v nádobě, měnil by se zároveň přenos soustavy a parametry regulátoru by museli být proměnné. Tato skutečnost vede na řešení adaptivního regulátoru. Měrná tepelná kapacita je známá a hodnota je 4180. Tepelnou soustavu považuji za systém prvního řádu, proto obsahuje pouze jeden integrátor. Je to velice jednoduchý systém, ale pro použití v laboratoři jako demonstrační úloha je podle mého názoru dostatečný. Topný element má jmenovitý maximální výkon 80kW. Saturace otopného systému zajišťuje omezení maximálního výkonu, protože většího výkonu není schopen za normálního provozu dodat.



Obr. 6-2 Model teploty MATLAB

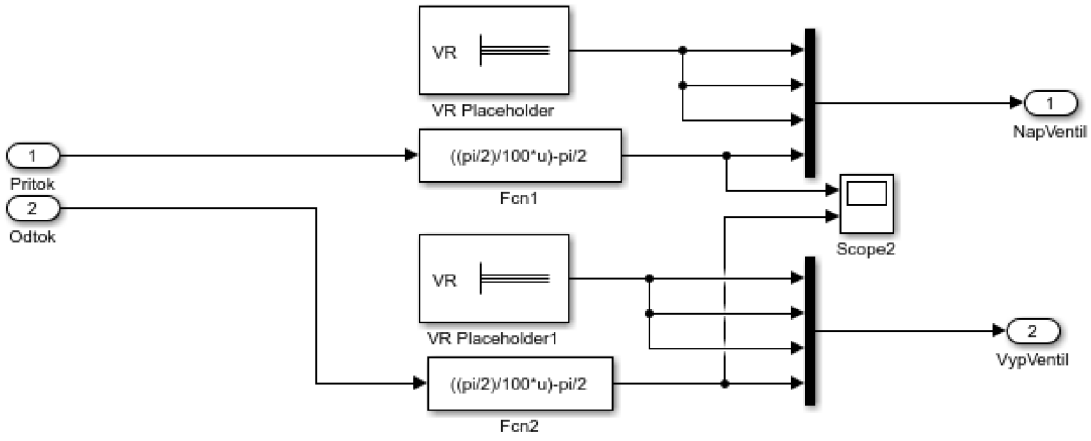
Aby byl systém řízen na požadovanou teplotu, zavedl jsem k němu i regulátor pro naši aplikaci postačí jednoduchý P regulátor, abych eliminoval ustálenou odchylku, mohu použít PI regulátor. Protože systém nádrže má jednu velkou časovou konstantu, považuji jej za systém prvního řádu. Pokud použiji PI regulátor, musí být I složka velmi malá, aby nedošlo k velkému zpožďování ale umožňuje podstatně přesnější doregulování na požadovanou hodnotu.

7. PROPOJENÍ VIRTUÁLNÍHO MODELU S DYNAMIKOU V MATLAB/SIMULINK

Model nádrže je poměrně velký, výška 1, 28m, průměr 0,5m a objem nádrže činí $1m^3$. Pro simulaci ve virtuálním modelu budu potřebovat upravit signály výšky hladiny z modelu a natočení jednotlivých ventilů. Nejprve se zaměřím na pohyb jednotlivých ventilů v animaci, jedná se o změnu úhlu rotace podle osy X, a to od 0 do pi/2, tedy 0-90°. Tímto pohybem simuluji otevření ventilu od 0%do 100%. Pro ovládání ventilu a tím i změnu průměru odtoku nebo přítoku potřebuji normalizovat požadované hodnoty na rozsah v procentech, který je výstupem jak z PLC, tak ovládacím prvkem Slider v Matlabu. podle rovnice (7.1) kde u je vstup a y je výstup funkce.

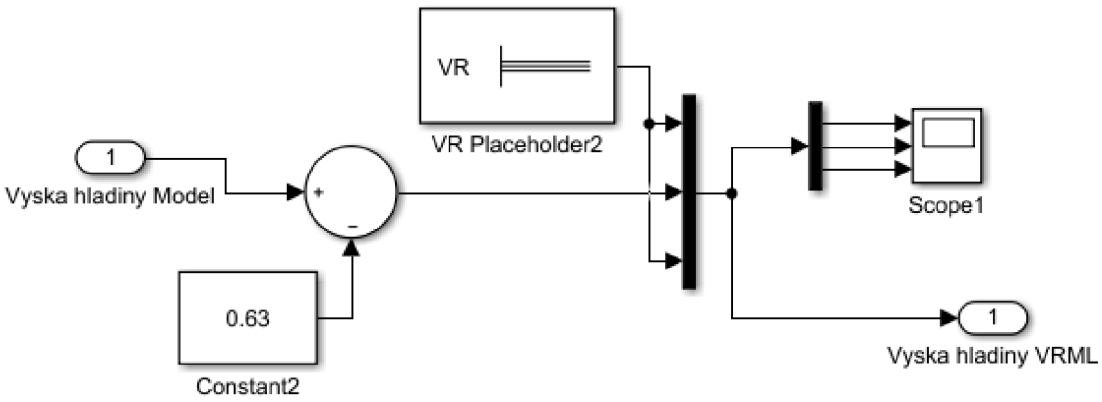
$$y = \left(\frac{pi/2}{100 \cdot u}\right) - pi/2 \tag{7.1}$$

Aby byly svázány hodnoty veličin přítoku a odtoku s ventily ve virtuální realitě, jsou vytvořeny funkce pro normalizaci z procent na natočení v úhlu. Pro ovládání musí být do VR posílán signál zahrnující všechny parametry. Proto musím sdružit signály jak úhlu, tak všech os. Abychom nezměnili výchozí nastavení orientace os, je nutné připojit na vstupy blok VR Placeholder. Na vstupech subsystému je umístěn přepínač, který přepíná mezi vstupem ovládaným ze simulace a vstupem z TCP/IP komunikace, tedy z PLC.



Obr. 7-1 Úprava dynamiky pro model ve VRML

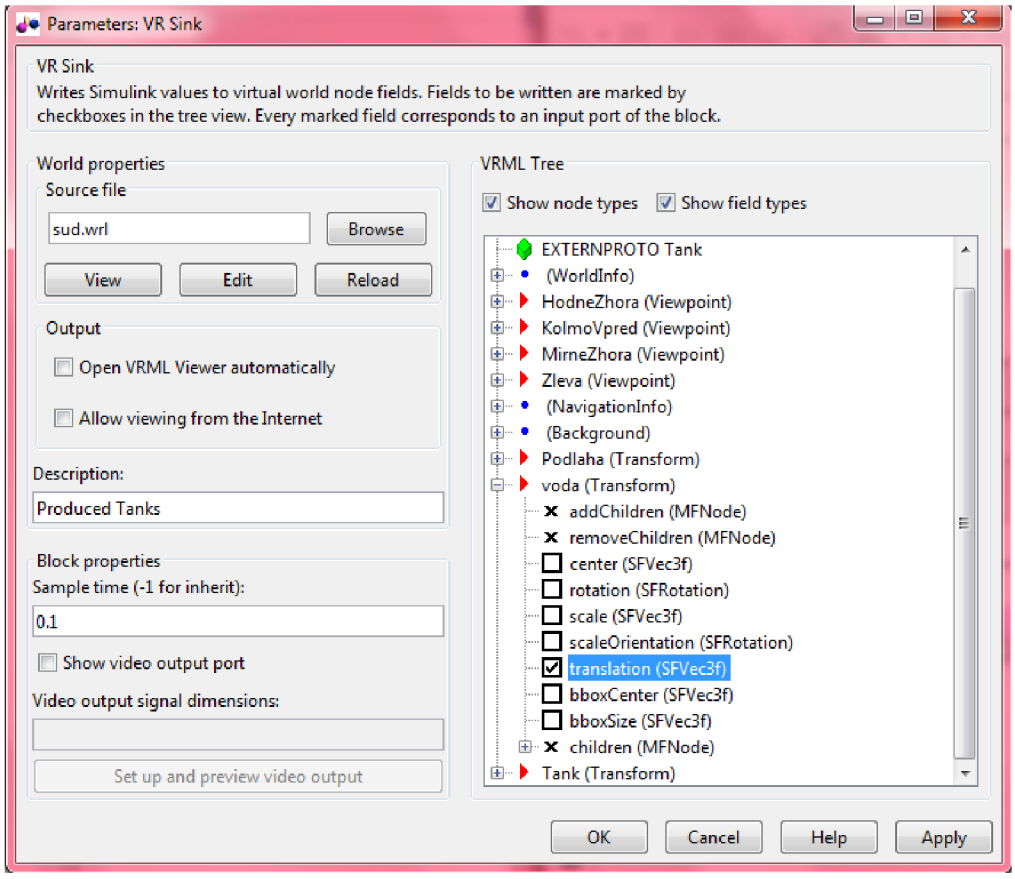
Pro zobrazení ve virtuální realitě je také nutné udělat drobnou úpravu pro simulaci výšky hladiny. Blok Cylinder (Válec) ve VRML je definován jeho výchozím bodem přesně uprostřed objektu. Tento výchozí bod však není vhodný pro simulaci z Matlab a je nutné posunout tento výchozí bod do jiného místa. Nejjednodušší variantou, jak k tomuto problému přistoupit, je odečtením poloviny vodního sloupce z maxima a o tuto hodnotu jej posunout níž. Tedy výška našeho vodního sloupce je 1,26m a tedy polovina je 0,63m



Obr. 7-2 Úprava dynamiky pro model ve VRML

Díky těmto přepočtům lze již k modelu ve virtuální realitě přistupovat ve stejných veličinách jako k matematickému modelu v programu Matlab/Simulink.

Posledním krokem je propojení signálů z Matlab/Simulink se signály do virtuální reality. Vložím do Simulinku blok VR Sink, tento blok je po vložení bez vstupu, proto musím nejdříve vytvořit vstupní porty pro tento blok. Ty vytvořím vybráním čtverce a označením fajfkou v příslušné části stromu zobrazeného v pravé části okna jako je na Obr. 7-3 Volba vstupu do VR Sink. Tímto vyberu všechny bloky prvky a jejich vlastnosti, které potřebuji během simulace měnit. Dále vyberu objekty „Madlo_pritok“ a „Madlo_odtok“ u obou zvolím parametr rotace (*rotation)*. Tímto postupem jsem vybral všechny potřebné vstupy do Virtuální Reality. Pozor na změnu osy rotace! Prvek *Transform* povoluje změnu rotace pouze v jedné ose. Pokud budu chtít změnit osu, ve které budu rotovat vytvořený objekt, je nezbytné si vytvořit vyšší prvek *Transform* a jeho potomek (*children*) bude již vytvořený Transform. Při této úpravě ale dbejte na kontrolu pozice (*Position*), protože rotace je prováděna vždy kolem bodu definovaném v parametru *Position*



Obr. 7-3 Volba vstupu do VR Sink

V pravé části parametrů bloku (strom) vyberu prvek voda, a ten rozbalím pomocí ikony +. Po rozvinutí vidím vstupy, které mohu vybrat a vytvořit jim vstup pro ovládání z prostředí Matlab/Simulink. Na obrázku 22 je vidět zatržená kolonka posunutí (translation). Nesmím však zapomenout, že tento budoucí vstup je stále vektorem a jeho vstupy jsou ve tvaru vektoru [x y z]. To znamená, že pokud budu-li chtít změnit hodnotu v jediném směru, je nutné použít multiplex signals a vkládat do vstupu sloučený vektor všech vstupních os. Hodnoty, které nechci měnit a byli stejné jako v definici virtuálního světa a neměnili se, je potřeba připojit tyto vstupy na výstup z bloku VR Placeholder Takto vybereme všechny vstupy se kterými chci v simulaci manipulovat.

8. VYUŽITÍ PLC PRO ŘÍZENÍ

Pro řízení modelu vytvořeného v prostředí MATLAB bylo použito PLC Siemens S7-1500. Je to jedno z nejnovějších PLC od firmy Siemens, programování PLC zajišťuje vývojové prostředí TIA Portal, V14, který obsahuje knihovny pro komunikaci, realizaci například PID regulátoru a mnoho dalších. Dále je možné k tomuto PLC připojit moderní dotykové obrazovky HMI, ze kterých lze ovládat technologický proces a vizualizovat ve 2D grafice děje probíhající v technologickém procesu.

8.1 Model versus reálný systém

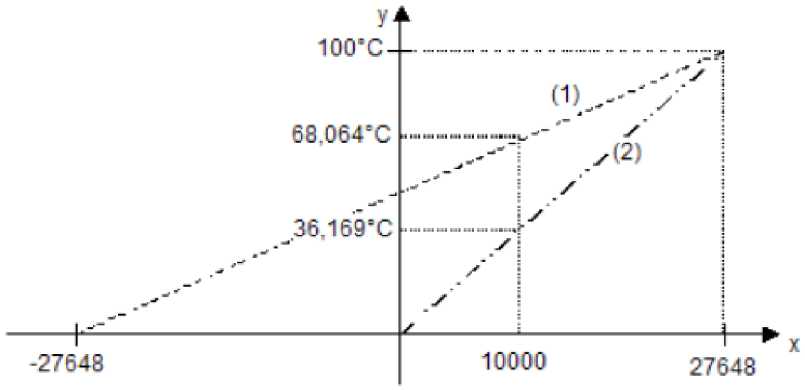
Model je určen k demosntračnímu účelu a oproti realitě je poněkud zjednodušen, protože je zanedbáno mnoho parazitních vlastností a dynamiky vyskytující se v reálném systmu. Model je navržen tak abychom ukázali jednoduché řízení, možnost propojení řízení s matematickým modelem dynamického systému, zobrazení tohoto systému ve virtuální realitě a zobrazení veličin do HMI a ovládání procesu z HMI.

Pokud bychom se chtěli co nejvíce přiblížit realitě, musím si nejprve uvědomit, že pokud je potřeba snímat fyzikální veličiny pomocí senzorů, potřebuji standardizaci vstupních signálů. Standardizace nám umožní například při použití senzoru pro měření teploty na vstupu v rozsahu (-50 až 150°C) a na jeho výstupu interpretuje změřenou hodnotu v rozsahu (4- 20mA). Tento senzor je pak připojen na příslušný analogový vstup, na tomto vstupu získám hodnoty 4-20mA které jsou A/D převodníkem interpretovány jako číslo v rozsahu 0-27648. toto číslo pak je nutné standardizovat na požadovanou veličinu například 0-100°C.

Analogový vstupní modul dává číselnou hodnotu typu INT, kterou představuje například měřená teplota. Tyto hodnoty jsou nazývány periferní. Tyto periferní hodnoty je nutné převést na inženýrské jednotky (tedy °C). Procedura vykonávající tuto funkci se nazývá standardizace. Tato procedura pro unipolární analogovou hodnotu se řídí podle rovnice 8.1

$$y = \frac{x}{27648} \cdot (MAX - MIN) + MIN \ \ [°C] \tag{8.1}$$

Kde x je vstupní proměnná ve °C periferní hodnota naměřená analogovým vstupním modulem je x =10000 a limity inženýrských jednotek je MIN = 0°C a MAX = 100°C



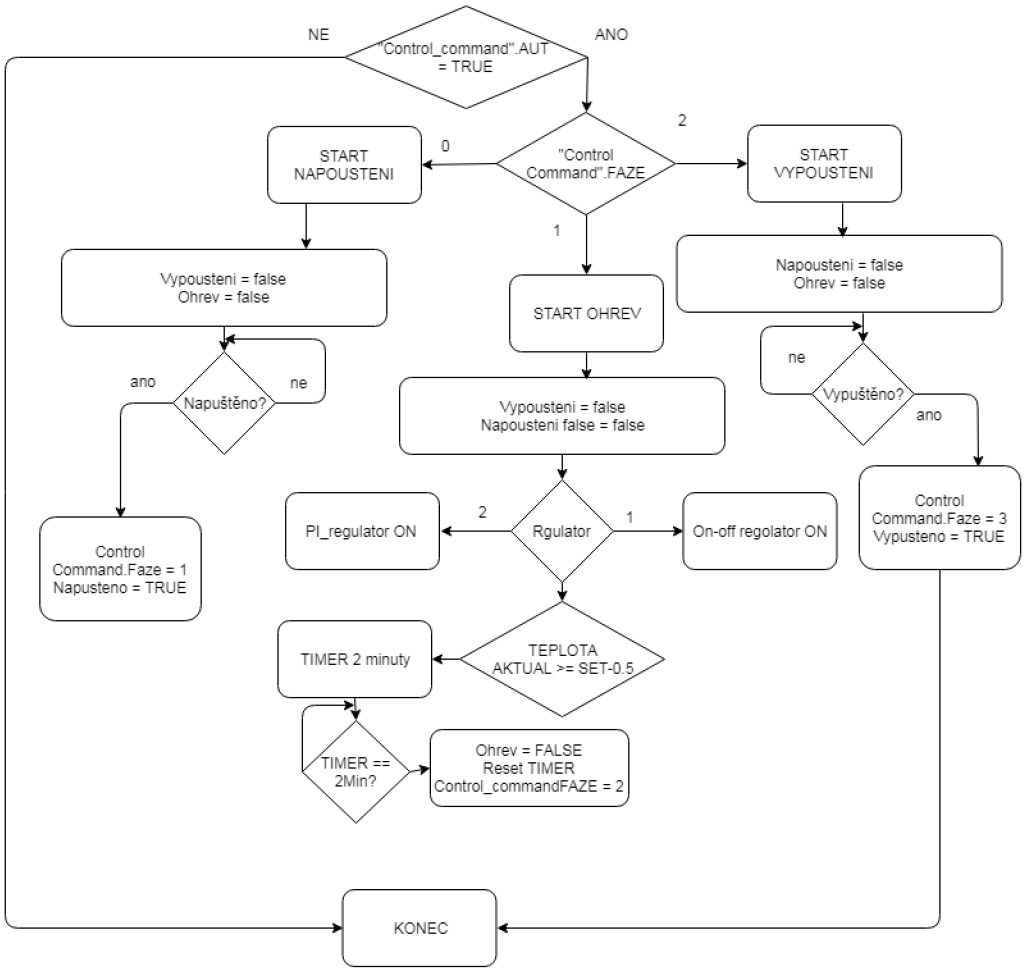
Obr. 8-1 Graf pro standardizaci teploty [6]

Destandardizace je inverzní operace standardizace. Tu řeším podle následující rovnice:

$$y = \frac{x - MIN}{MAX - MIN} \cdot 27648 \ [°C] \tag{8.2}$$

8.2 Řídicí program

Jakékoliv technologické zařízení je nutné ovládat manuálně nebo řídit automaticky.



Obr. 8-2 Blokový diagram pro automatický režim

Pro ovládání v manuálním režimu je požadováno zadávání hodnot do systému ideálně z vizualizačního zařízení. K tomuto účelu je použito HMI zadávat hodnoty například pro otevření a zavření ventilů, výšku hladiny a požadovanou hodnotu teploty pro ohřev tanku a spojení komunikace.

Pro automatické řízení jsem navrhl jednoduchý program podle obrázku na vývojovém diagramu s názvem „automat_process [FC4]“, jehož cílem je napustit tank do maximálního objemu, v polovině napouštění zapne míchání a po napuštění je procesem zadán ohřev na požadovanou hodnotu, kterou zadá uživatel z HMI. Jakmile dosáhne tekutina požadované teploty, je spuštěn časovač po dobu 2 minut. Po dočítání časovače, je vypnut ohřev a tekutina se začne vypouštět z tanku, asi v polovině vypouštění se zastaví míchání. Pro dosažení teploty je v programu implementován On - Off regulátor a PI regulátor. On – off regulátor pracuje tak, že dokud nedosáhne požadované hodnoty, tak topí plným výkonem. Po dosažení požadované hodnoty vypne dodávaný výkon a pokud teplota klesne pod 0,2°C požadované hodnoty, tak opět zapne. Pokud teplota vody v nádrži dosáhne odchylky 0,5°C požadované hodnoty spustí se 2 minutový časovač. Po 2 minutách se ohřívání vypne a přejde se k vypouštění tanku. Tento čítač je krátký kvůli testování. V technologickém procesu by byl určen recepturou při zpracování produktu.

V manuálním režimu lze nastavit průtok napouštěcího ventilu v numerickém bloku SET NapVentil a průřez vypouštěcího ventilu SET VypVentil. Nebo využít funkci Napouštění a Vypouštění v manuálním režimu. Pokud nastavíte hodnotu SET Objem a spustíte v manuálním režimu Napouštění, napustí se objem po tuto nastavenou hodnotu. Funkce Vypouštění vypustí tank. Míchání se spouští automaticky v obou režimech. V tomto režimu není k dispozici žádný z vytvořených regulátorů.

8.3 Návrh HMI

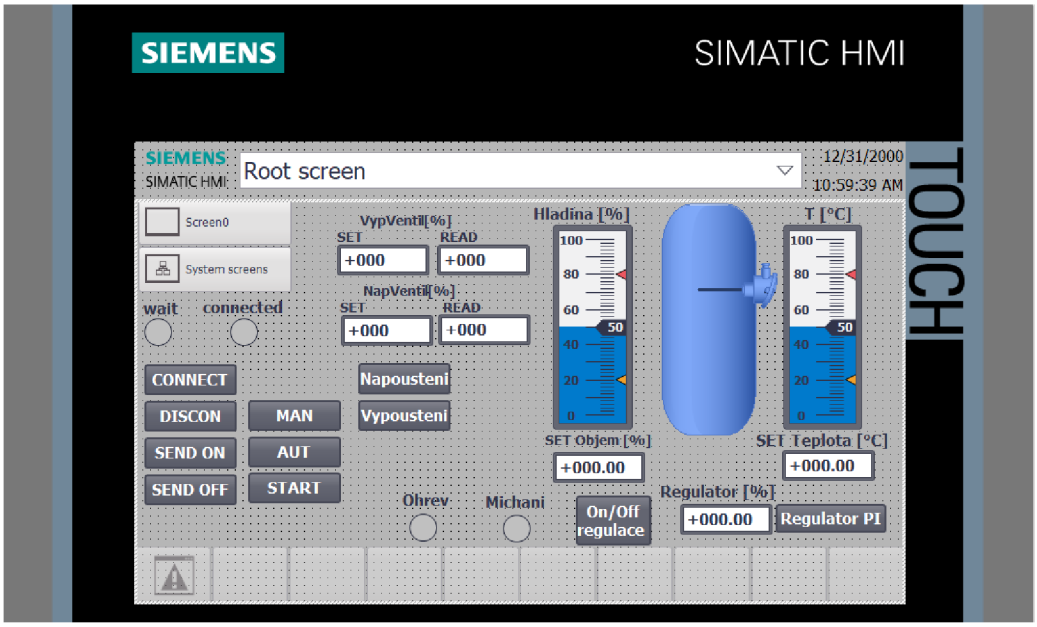
K veškerému ovládání slouží vizualizace v HMI od firmy Siemens. Vizualizaci jsem vytvořil v prostředí WinCC v TIA Portal. Vizualizační zařízení typu HMI Simatic TP700 Comfort Panel. Vizualizace tohoto panelu na Obr. 8-2 je vybavena tlačítky „CONNECT“, „DISCON“, „SEND ON“, „SEND OFF“ pro ovládání komunikace, indikačními prvky „wait“ a „connected“, tlačítky pro výběr režimu „MAN“ nebo „AUT“ a v neposlední řadě tlačítkem START pro spuštění automatického procesu. Dále vizualizace obsahuje číselná pole pro zobrazení a nastavení procesních veličin. Pole označená jako SET slouží pro nastavování, READ pro čtení hodnot v procesu. V Automatickém režimu můžu nastavit pouze Teplotu, protože ostatní hodnoty jsou nastavovány automaticky během vykonávání programu „automatic_process“.

Pro zobrazení aktivního míchání a ohřevu jsou umístěné ve spodní části obrazovky barevné indikační body. Pokud tyto body svítí červeně je daná operace neaktivní, pokud je operace spuštěna, rozsvítí se body zeleně.

V manuálním režimu lze nastavovat hodnoty ventilů a požadovanou výšku hladiny. Stisknutím tlačítka „Napousteni“ spustím funkci se stejným názvem a ta naplní tank do úrovně, nastaveného objemu v procentech. Tlačítkem „Vypousteni“ dám povel k vypuštění tanku.

Pro zobrazení aktuální výšky hladiny a aktuální teploty slouží vizualizační prvky Bargraf se stupnicí v rozsahu 0-100. Výška hladiny, je udávána v % a teplota ve °C s ohledem na rozsah teploty v procesu, která je také v rozsahu 0-100°C.

Talčítko „On/Off regulace“ povolí činnost On/off regulátoru, tento regulátor je nastaven jako výchozí pro automatický program. Tlačítko „Regulator PI“ deaktivuje On/Off regulátor a aktivuje PI regulátor. Mezi těmito tlačítky je zobrazovací prvek, ve kterém je aktuální hodnota akčního zásahu regulátoru.

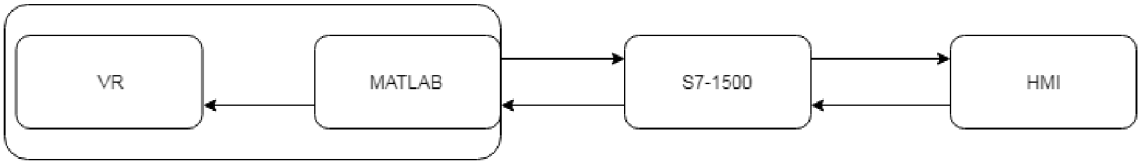


Obr. 8-2 HMI pro ovládání technologického procesu

9. KOMUNIKACE MEZI PLC A MATLAB

Komunikace mezi řídicím programem v PLC a virtuálním modelem v programu Matlab/Simulink, je nepostradatelnou částí pro propojení vizualizace s navrženým řídícím programem. Komunikace běží na protokolu TCP/IP a je zajištěna předem připravenými knihovnami a funkcemi v PLC i v prostředí Matlabu.

Tyto knihovní funkce zajišťují funkcionalitu, ale potřebují ještě vytvořit obslužnou část, pro přenášení dat umístěných v paketu v definovaném pořadím. Výhodou předpřipravené knihovny v PLC je že nemusím řešit operaci přenosu dat po bytech, pokud chci posílat jakýkoliv datový typ. Tyto knihovní funkce totiž režii s převodem datových typů má implementovanou v těchto funkcích. Jelikož jsou všechny přenášené proměnné v datovém typu REAL, který je tvořen 32bity. Budu přenášet pouze data o velikosti 4 byty. Na Obr. 9-1 je znázorněn přenos dat mezi jednotlivými systémy. Data jsou seřazena do paketu, který obsahuje identifikační hlavičku, platná data, a kontrolní součet.



Obr. 9-1 Výměna dat mezi systémy

9.1 Nastavení komunikace pro propojení v PLC

Abychom mohli připojit zařízení komunikující přes TCP/IP je nutné jej přidat do projektu. To lze provést v hlavním okně TIA Portalu kliknutím na „Add to device“ a vybrat „PC systems“ dále „PC general“ a toto zařízení vložit do projektu. Poté je nutné nastavit komunikaci, to lze provést po vložení a nastavení bloku TCON.

9.2 Bloky v PLC pro komunikaci TCP/IP

Pokud chceme přenášet data z PLC do jiného zařízení, můžeme využít v programu TIA Portál knihovnu *„Communication“* v podsložce *„Others“*, která obsahuje bloky TCON, TDISCON, TSEND a TRCV. **Chyba! Nenalezen zdroj odkazů.**

9.2.1 Blok TCON

Blok TCON slouží pro spojení komunikace mezi PLC a Matlab/Simulink, Otevře port nastavený ve vlastnostech tohoto bloku a je nutné nastavit IP adresy obou zařízení, které spolu budou komunikovat, jak PLC tak i vzdáleného PC. PLC je nutné nastavit jako server. Blok TCON musí být aktivní po celou dobu přenosu dat. Pokud chceme ukončit propojení, je nutné tento blok deaktivovat a poté aktivovat blok TDISCON, který uzavře otevřený port.

9.2.2 Blok TDISCON

Tento blok komunikaci uzavírá otevřenou komunikaci. Pro jeho aktivaci je nutné deaktivovat blok TCON. Dokud bude blok TDISCON aktivní, nepodaří se nám aktivovat TCON.

9.2.3 Blok TSEND

Tento blok odesílá data z PLC na linku do vzdáleného zařízení, v tomto případě PC. Data která přicházejí z proměnné „Data_Transmission“.Data_to_Matlab, jsou při každé vzestupné hraně signálu TSEND_Req odeslány na linku. Pokud dojde k odeslání celé délky datového formátu vstupní proměnné, potvrdí se přenos signálem TSEND_Done. Dokud nejsou přenesena všechny byte z proměnné Data_to_Matlab, Je aktivován signál Busy. Při odesílání dat musí být proměnná Data_to_matlab konstantní.

9.2.4 Blok TRCV

Blok Receive je určen pro čtení dat z komunikační linky ze vzdáleného zařízení. Data, která má Matlab poslat, musí být před odesláním upravena tak, aby je mohlo PLC přečíst. Je zde jiná souslednost odesílaných dat. Blok TRCV na nástupnou hranu přečte tolik byte, kolik udává datový typ na vstupu TRCV_DATA. Vstupem Data_from_Matlab, je určena velikost přijímaných dat. Tedy pokud zvolím datový typ REAL, bude přijímat paket o velikosti 4 byte. Signál TRCV_EN, povoluje čtení z linky. TRCV_DN říká, jestli byla data přijata. Signál Busy je aktivní doku není přijatá celá délka datového typu a TRCV_LEN nám dává informaci o tom, kolik byte bylo přijato z kompletní délky REAL, pokud je úspěšně přijat kompletní REAL, tato hodnota se automaticky vynuluje.

9.2.5 Blok TDIAG

Blok TDIAG slouží k diagnostice připojení, pomocí tohoto bloku lze identifikovat stavy připojení. Pokud je zařízení ve stavu Disconnect identifikuje chybu. Tento status lze propojit například na HMI Warnings a zobrazovat poruchy přímo na display HMI.

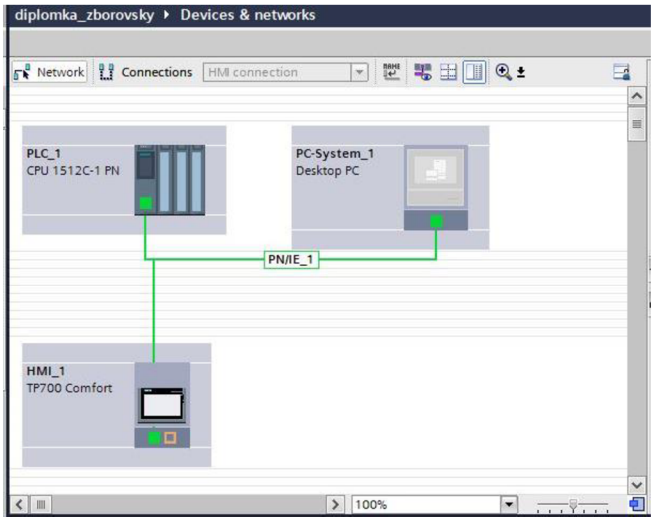
9.3 TCP/IP komunikace v prostředí Matlab/Simulink

Tento program má k dispozici jednoduché příkazy pro nastavení komunikace. V programu Matlab je k dispozici knihovna s názvem Instrument Control Toolbox, který obsahuje komunikaci TCP/IP a veškeré součásti potřebné k provozu této komunikace.
Příkazem `t=tcpip('192.168.1.201',2000,'NetworkRole','client');` nastavíme, IP zařízení ke kterému se připojíme, port zařízení na kterém komunikujeme, a roli zařízení (client/server). Po nastavení komunikace musíme ještě otevřít port pro komunikaci příkazem `fopen(t);` tímto příkazem propojíme obě zařízení. Dále už můžeme pomocí příkazů dvojice příkazů `fwrite(t,data);` odesílat a `read = fread(t,4); %kde 4 je délka dat v bytech` přijímat data.
Komuinikaci uzavíráme přikazem `fclose(t);`

10. REALIZACE KOMUNIKACE

Pro propojení softwaru Matlab/Simulink a reálného zařízení PLC, je potřeba nastavit komunikační linku. Nejprve v PLC vyberu ve stromu projektu ***„add new device“*** zvolím kategorii ***„PC systems“→„PC general“→“PC station***“ a ještě přidám HMI pro ovládání, „Simatic Comfort Panel“→ „7“Display“→ „TP700 Comfort“ a přidáme jej do projektu.

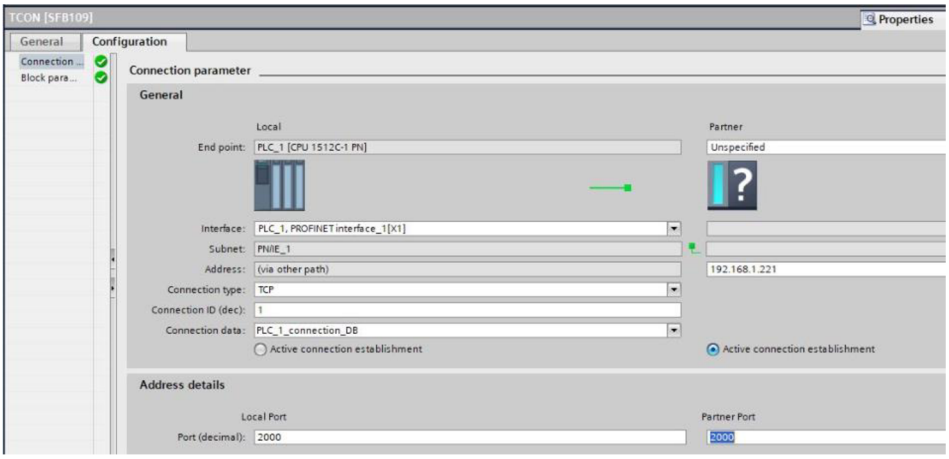
Poté propojím komunikaci podle obrázku Obr. 10-1. Ještě je potřeba nastavit IP adresy jednotlivých zařízení.



Obr. 10-1 Propojení komunikace HMI, PLC a PC

Vložením bloku TCON do bloku programu „Main [OB1]“ a jeho nastavením vytvořím po zapnutí signálu TCON_REQ vytvořím virtuální spojení, které čeká na připojení vzdáleného zařízení.

Na Obr. 10-2 jsou parametry bloku TCON, kde nastavím IP adresu PC a komunikační port PLC, ke kterému se bude připojovat Matlab.



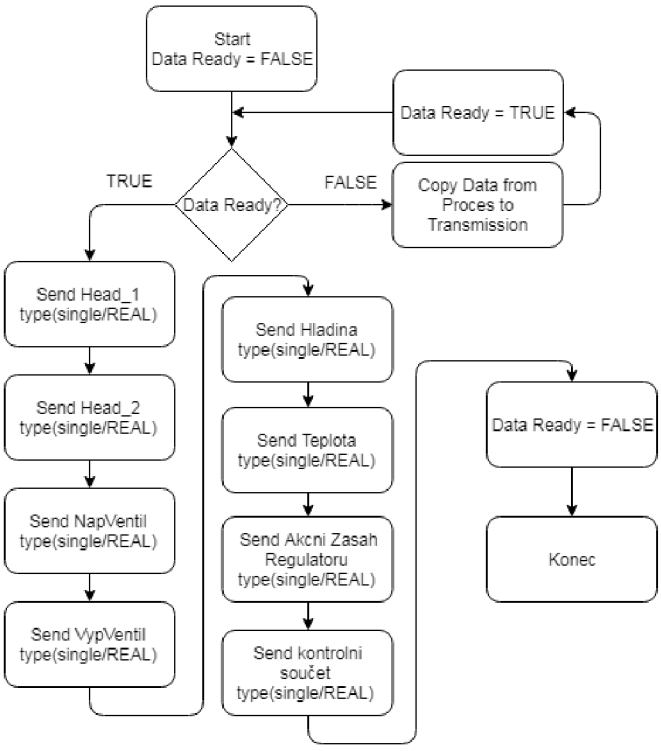
Obr. 10-2 Nastavení bloku TCON

Abychom mohli posílat data během simulace je nutné si vytvořit funkce pro přijímání a odesílání dat v obou zařízeních. Tato sériová komunikace je orientovaná po bytech a

opačně orientovaná posloupnost přicházejících bytů mezi Matlabem a PLC. Z tohoto důvodu je pro správnou reprezentaci přijímaných a odesílaných dat přidána ještě operace pro přehození bytů a přetypování dat. Ekvivalentní datový typ *„Real“ (PLC)* je *„single“ (Matlab)* oba typy mají datovou reprezentaci 32bitů.

10.1Tvorba a odesílání paketu dat

Pokud chci odesílat data z jednoho zařízení do druhého, musím data připravit, seřadit a odeslat. Ve druhém zařízení pak ve správném pořadí zase přijmout a přiřadit ke správným procesním proměnným. Pro odesílání postačí jednoduchý cyklus přepínače „*switch“* Program je navržen podle následujícího diagramu.



Obr. 10-3 Vývojový diagram pro odesílání dat

10.1.1 Matlab ->PLC

Pro odesílání dat slouží funkce SEND.m, pokud je tato funkce zavolána v Matlab/Simulink při spuštěné simulaci všechna data připojená na vstup této funkce se nemění a jsou připravena k odeslání. Ve funkci SEND.m podle vývojového diagramu na obrázku Obr. 10-3 jsou data reprezentována jako paket o přesné velikosti. Paket se skládá ze dvou proměnných typu „single“, které tvoří hlavičku a z dalších 5 užitečných dat typu „single“ potřebných pro řízení daného technologického procesu. Poslední odesílanou proměnnou jsou kontrolní 4 byty, které slouží pro validaci přijatých dat. Tyto čtyři byty jsou jednoduchým součtem dat, které jsou odesílány. Funkce pro odesílání dat z Matlabu

je naprogramována jako stavový automat, jenž postupně odesílá jednotlivé proměnné sekvenčně za sebou pomocí volání funkce fwrite(data). Který vystaví data na linku.

Před odesíláním dat na linku je nutné upravit data na osmibytový formát. Data, která budu odesílat, se nacházejí v proměnné *inFloat*. Proměnnou *inFloat* nejprve převedu na neznaménkový osmibytový typ příkazem: *„data = typecast(single(inFloat),'uint8')“*. Data jsou nyní v řádkovém vektoru a proto použiji příkaz pro prohození prvků na řádku *„sendData = fliplr(data)“* tato data pak mohu vyslat na linku příkazem *„fwrite(t,sendData)“* tato funkce se už interně stará o správné odeslání dat přes TCP/IP komunikaci. Pro odesílání dat z modelu v Matlab jsem zvolil cyklus periodicky opakovaný 0,1 vteřiny, k tomu jsem použil příkaz tic a toc, kdy se ptám na jestli toc je větší než 0,1s.

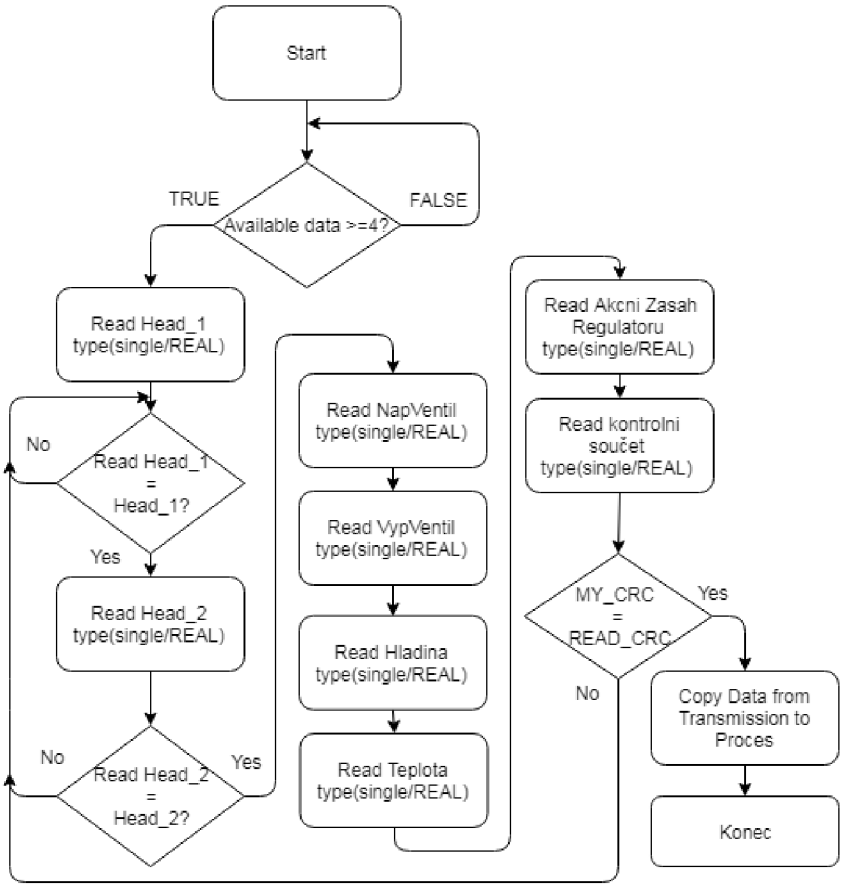
```
fwrite(t, fliplr(typecast(single(inFloat), 'uint8')));
```

10.1.2 PLC -> Matlab

Odesílání dat z PLC je řízeno blokem TSEND umístěném v bloku „Main [OB1]“, ve kterém je vstupní signál TSEND-REQ a ten povoluje odeslání dat na komunikační linku. Tento signál je nastavován ve funkci „sending [OB32]“. V této funkci se nejdříve dotazuji na připravenost dat. Základem je oddělení dat procesu, které se mohou měnit a připravených dat pro odeslání, Oddělení zajistím překopírováním dat z datového bloku „Process_data [DB7]“ do Data_Transmision[DB9]. Po této operaci jsou data připravena k odeslání a ještě si spočítám kontrolní součet připravených dat. Připravená data je nutné postupně vystavovat do proměnné „Data_Transmision“.Data_to_MATLAB, připojené na vstup bloku TSEND-DATA, data na vstupu bloku musí být konstantní během odesílání. Poté je nutné vyslat signál TSEND_REQ v logické 1, nástupná hrana tohoto signálu odešle data na linku. Identifikace správného odeslání dat umožňuje signál TSEND_DN. V bloku OB1 je vytvořen „Network 7: “ pro potvrzení odesílání. V něm výstup TSEND_DN resetuje signál TSEND_REQ. Signál TSEND_DN je pouze krátký puls, který je nutné zaznamenat. Z jednoho bloku TSEND lze odesílat pouze jeden datový typ, který je definován proměnnou připojenou právě na vstup TSEND-DATA. Funkce „sending“ postupně odesílá připravená data na linku, nejdříve vyšle dvě proměnné REAL představující hlavičku, poté 5 proměnných dat a nakonec vypočtený kontrolní součet. Po odeslání celého paketu je deaktivována validita dat a proces odesílání se opět opakuje. Perioda funkce „sending“ je 100ms.

10.2Přijímání dat z komunikace

Přijímání dat z komunikace je shodná v obou případech, jak v PLC tak v programu Matlab. Přijímání je navrženo podle následujícího blokového diagramu.



Obr. 10-4 Vývojový diagram pro přijímání dat

10.2.1 PLC -> Matlab

K přijímání dat z PLC do Matlabu v simulaci využívám funkci „RECEIVE.m“, která je napsaná podle diagramu uvedeného na Obr. 10-4. Pokud nejsou v bufferu komunikace k dispozici alespoň 4 byte, tak se stále dokola ptám, jestli jsou už alespoň 4 byte k dispozici, pokud ano, mohu začít s identifikací přijaté hodnoty. Ve funkci RECEIVE.m zavolám funkci „readFloat()“. Funkce „readFloat()“ zajistí vyzvednutí dat z interního bufferu komunikace příkazem `fread(t,4)`, a uložím si jeho výsledek do proměnné „read“. Výsledek předchozího příkazu vytvoří sloupcový vektor o velikosti 4 a jeho prvky reprezentují 4 byte. Je třeba zajistit upravit data do formátu odpovídajícímu v PLC. Nejprve přehodím pořadí přicházejících byte (sloupcový vektor) příkazem *„data=flipud(read)“* dále vektor přetypuji na typ single příkazem *„typecast(data,‘single‘)“* převedu na typ single a uložím výsledek do readFloat.

```
readFloat=typecast(uint8(flipud(read)),'single');
```

Nyní je proměnná readFloat stejná jako byla v PLC.

Ve funkci RECEIVE.m Postupně procházím přicházející data, a identifikuji začátek paketu, pokud souhlasí hlavička, kterou přijímám a očekávám, mohu bez starostí přijímat data a průběžně si je ukládat na dočasné proměnné. Na konci paketu přijatých dat je ještě kontrolní součet opět tvořen 4 byte, ve kterém je ukryta hodnota součtu odeslaných dat.

Pokud hodnota přijatého součtu souhlasí se součtem hodnoty přijatých dat, lze dočasná data nahrát do paměti pro proces a tato data jsou validní. Funkce RECEIVE.m je volána s periodou vzorkování simulace, to však není vhodné pro příjem dat. Proto při každém volání této funkce je vybráno tolik dat, kolik je k dispozici v bufferu, aby hodnota předaná do modelu byla vždy nejaktuálnější, stejně jako v PLC, které daný proces řídí.

10.2.2 Matlab -> PLC

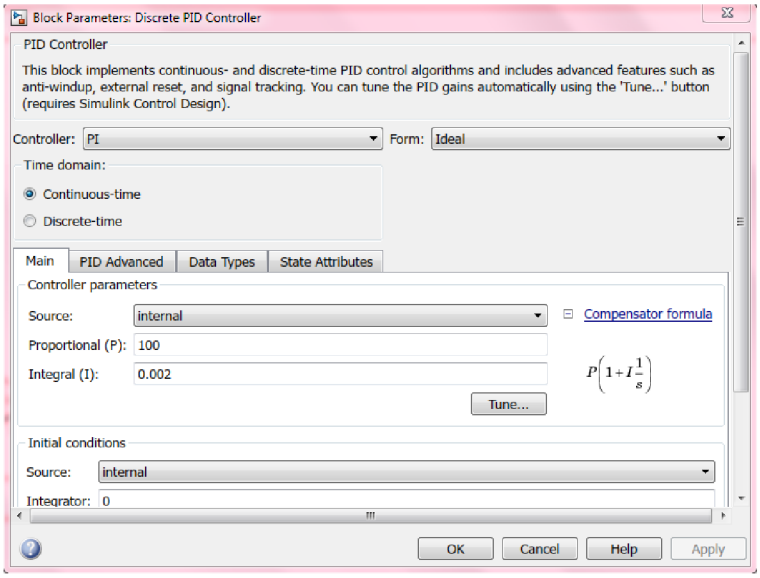
Přijímání dat z Matlabu do PLC s využitím bloku TRCV. Tento blok vybere data ze vstupního bufferu komunikace při aktivaci signálu TRCV_EN a uloží přijatá data do proměnné „Data_Transmision".Data_from_Matlab. Po uložení dat aktivuje výstup TRCV_NDR. Formát získaných dat je dán formátem vstupní proměnné „Data_Transmision".Data_from_matlab která je typu (REAL), to udává kolik byte bude vybráno z bufferu. Jakmile přečte všechny příchozí 4 byte a potvrdí přijetí signálem TRCV_NDR. Dokud není potvrzený signál NDR (new data ready), v proměnné zůstává hodnota před čtením nových dat.

Jestliže jsem získali signál TRCV_NDR, mám aktuální nová data v proměnné „Data_Transmision".Data_from_Matlab. Tato přijatá data porovnám s hodnotou představující hlavičku, kterou začíná každý paket, pokud jsem ji neobdržel, vyčítám data tak dlouho, dokud na hlavičku nenarazím. Jakmile je přijata dvakrát za sebou hlavička mohu bez starostí ukládat data do proměnných v bloku „Data_Transmision [DB9]". Kontrolou přijatých dat je srovnání přijaté hodnoty kontrolního součtu a přijatých dat. Pokud se součty rovnají, jsou přijaté hodnoty vystaveny do paměťového prostoru programu pro řízení do bloku „Process_Data [DB7]".

Pomocí bloku TRCV také identifikuji ztrátu komunikace. Pokud se například odpojí připojené zařízení, v mém případě Matlab/Simulink, blok TRCV zjistí chybu v při čtení z linky a nastaví TRCV_ERROR do logické 1 a TRCV_STATUS je roven hodnotě 16#80C4. Této vlastnosti využívám pro obnovení komunikace. Pokud dojde k této poruše je komunikace odpojena a je nutné ji znovu navázat.

10.3PI regulátor v Matlab/Simulink

Pro řízení teploty je v Matlab/Simulink vytvořen PI regulátor. Obdobný regulátor je implementován i do systému PLC. Je popsaný rovnicí v ideálním tvaru viz rovnice na Obr. 10-5. Perioda vzorkování je stejná jako perioda simulace, systém se chová jako spojitý vzhledem k regulátoru. Regulátor musí být omezen na akční zásah 0-100% proto v záložce „PID Advanced" je třeba zatrhnout „Output Limit" a nastavit maximum 100 a minimum 0. Nastavit proporcionální složku zesílení (P) a integrační složku (I), což je převrácená hodnota časové integrační konstanty Ti.



Obr. 10-5 Nastavení PI regulátoru Matlab/Simulink

10.4 Regulátor PI v PLC

Regulátor v PLC je popsán podobnou formou jako v Matlabu, kdy je zesílení vytknuto před všechny koeficienty PID regulátoru, jak můžete vidět na Obr. 10-6. v nastavení parametrů lze vybrat PID nebo PI regulátor

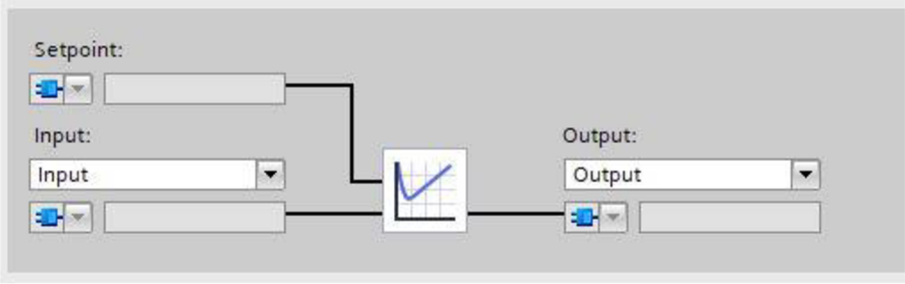
$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

y	Output value of the PID algorithm
K_p	Proportional gain
s	Laplace operator
b	Proportional action weighting
w	Setpoint
x	Process value
T_I	Integral action time
a	Derivative delay coefficient (derivative delay T1 = a × T_D)
T_D	Derivative action time
c	Derivative action weighting

Obr. 10-6 Rovnice PID regulátoru v PLC Siemens

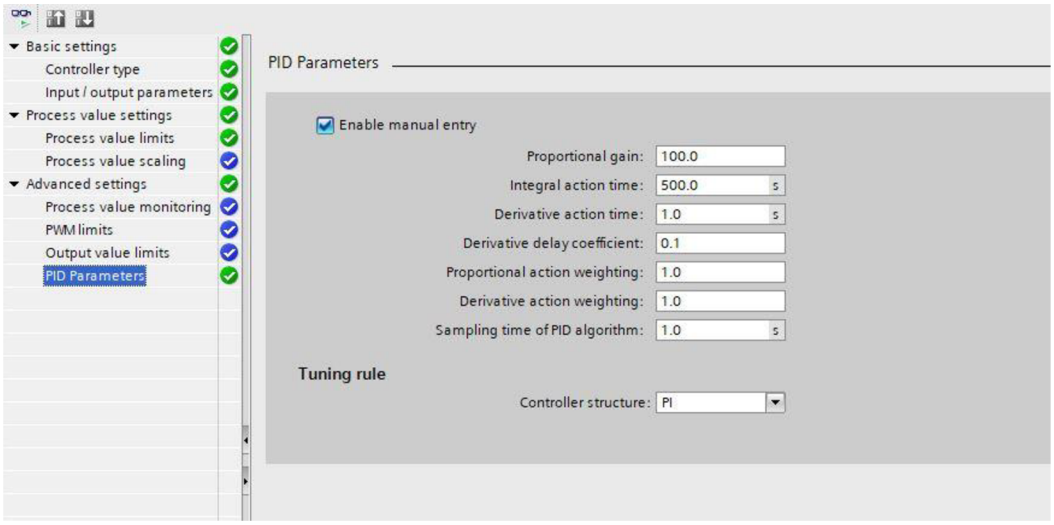
Ve stromu projektu ve složce *„Technology object“* vytvořím *„PID_compensator“* Nastavím mu vstupní a výstupní veličiny a parametry. v záložce *„Parameters“* Implementovaný model v Matlab/Simulink odesílá hodnoty v reálných veličinách. Hodnota aktuální teploty je odesílána z modelu ve stupních celsia stejně jako požadovaná hodnota teploty a výstup regulátoru je očekáván v procentech. Proto je výhodné použít parametry „*Input*“ a „*Output*“ pro propojení dat z modelu do regulátoru. Pokud bychom měřili hodnoty příchozí z reálných snímačů připojených na analogové vstupy PLC, bylo by nutné standardizovat z A/D převodníků a destandardizovat do AD převodníku na

výstupu viz kapitola 8.1 a použít vstupy „*Input_PER*“ a „*Output_PER*“. Tyto hodnoty jsou nastavovány v *„Basic Settings“*



Obr. 10-7 Vstupní a výstupní parametry regulátoru

V záložce *„PID_compensator“* → *„PID Parameters“* nastavím potřebné parametry. Nastavím hodnoty zesílení K= 100, časovou konstantu Ti = 500s, váhy s hodnotou 1.0, derivační konstanta se neuplatňuje, protože je zvolen typ PI regulátor. Regulátor musí být volán s požadovanou periodou vzorkování. (cyclic interrupt OB31) Pokud v bloku cyklického přerušení nastavím periodou volání 1 sec, je nutné nastavit i parametr *„sampling time“* v PID regulátoru na stejný vzorkovací čas. Parametry do PLC se uloží pouze v online režimu, a to pouze pokud spuštěném commissioning režimu!

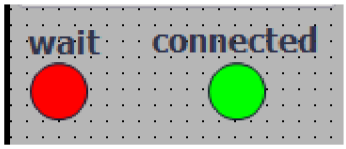


Obr. 10-8 Nastavení parametrů PID v PLC

Aby regulátor byl spouštěn automaticky, musí být nastaveny hodnoty v datovém bloku regulátoru [DB13]: „PI_regulator“.*Mode Activate* = *TRUE* a „PI_regulator“.Mode = 3.

11. OVĚŘENÍ FUNKČNOSTI A VÝSLEDKŮ

Nejdříve nahraji vizualizaci a program do PLC. Po úspěšném nahrání programu a vizualizace stisknu na obrazovce HMI tlačítko CONNECT, tím se nám otevře komunikační linka na straně PLC. Komunikace PLC čeká na připojení vzdáleného zařízení.

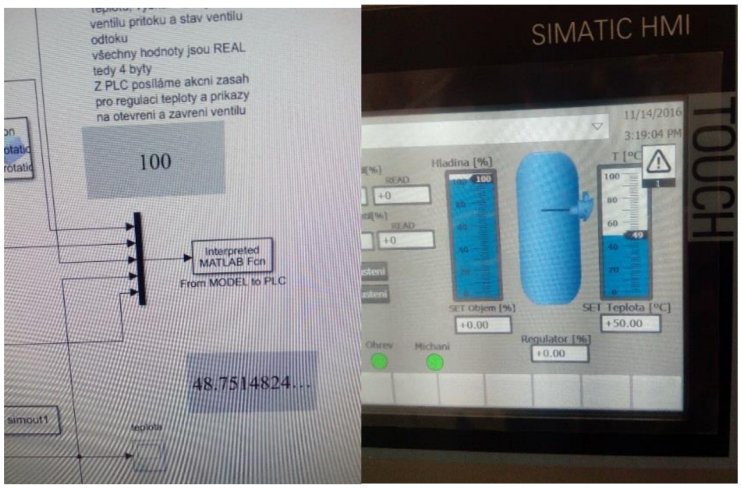


Obr. 11-1 Signalizace stavu komunikace

Nad tlačítkem CONNECT jsou k detekci spojení dva signalizační prvky „wait“ a „connected“. Pokud svítí zeleně „wait“, zařízení čeká na připojení vzdáleného zařízení. V následujícím kroku spustím v programu Matlab funkci „***Initialize.m***“, tímto se nám propojí komunikace mezi PLC a Matlabem. Pokud se tak stalo, měl by stav signalizačních bodů odpovídat zobrazení na Obr. 11-1. Pokud se podařilo propojit komunikaci v pořádku indikuje „connected“ zelenou barvou aktivní komunikaci. Nyní mohu otevřít model s vizualizací v Matlabu a spustit simulaci. Pro správnou simulaci nastavím fixní krok, Periodu vzorkování na 1 vteřinu a solver metodu *„ode4(Runge-Kutta)“*. Stop time bude mít hodnotu *„inf“*, to je nutné pro chod v reálném čase.

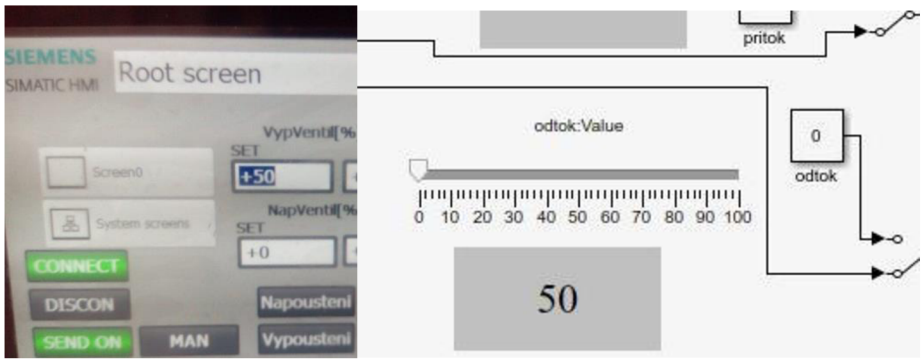
Po spuštění simulace odesílají data z Matlabu do PLC. Na HMI stisknu tlačítko „*SEND ON“*. Sigál aktivovaný tlačítkem povolí odesílání dat do Matlabu. Pokud komunikace běží, lze na HMI nastavit požadované hodnoty (teplotu, otevření ventilů, výšku hladiny) abychom mohli ovládat ventily jen nutné, aby byl aktivní manuální režim (zeleně svítí tlačítko „MAN“). Při aktivaci tlačítka „AUT“ je systém v automatickém režimu. V automatickém režimu stačí nastavit požadovanou teplotu, na které se řízení ustálí s odchylkou ±0,5°C. Poté lze stisknutím tlačítka START zahájit proces automatického programu. V prvním kroku je proměnná „Control_command“. FAZE = 0. Tato hodnota spustí napouštění tanku, po napuštění na maximální objem přechází do další fáze programu. Další fáze je „Control_command“.FAZE = 1. Tato část programu povolí ohřev vody v tanku. Jakmile dosáhne hodnota teploty ±0,5°C od požadované hodnoty, spustí se časovač na 2 minuty a po dvou minutách se ohřev vypne. V tomto momentu přechází automatický program do třetí fáze. Proměnná „Control_command“.FAZE = 2 povolí vypuštění tanku a jakmile je tank prázdný, zastaví se proces „Control_command“.FAZE = 3 a čeká na další spuštění podmínkou START.

Na Obr. 11-2 lze spatřit hodnotu výšky hladiny a teploty odesílanou z modelu do PLC při testování běhu programu a komunikace.

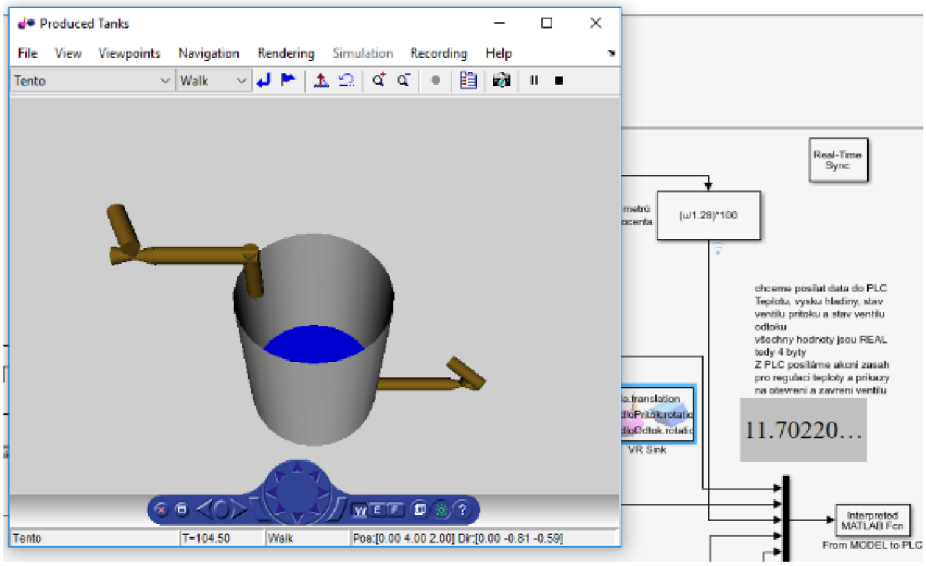


Obr. 11-2 Ověření funkčnosti komunikace

Na obrázku Obr. 11-3 je příklad přenosu informace z PLC do Matlab/Simulink.



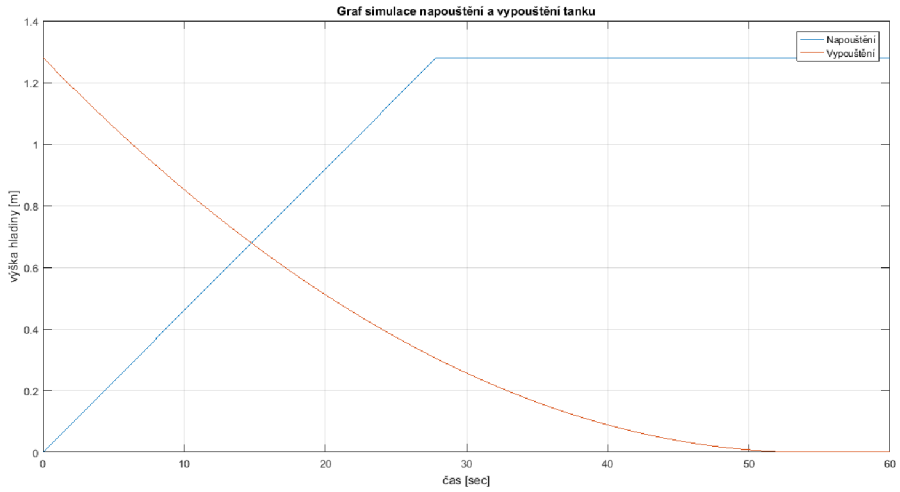
Obr. 11-3 Příklad nastavení hodnoty a ověření komunikace



Obr. 11-4 Napouštění tanku v RT

Na Obr. 11-5 je porovnání doby napouštění při nulovém odtoku a maximálním přítoku a vypouštění při nulovém přítoku a maximálním odtoku. Nádrž je napuštěna za 28s. což odpovídá výpočtu z rovnice Rov.(11.1) a vypuštěna za 51s.

$$t = \frac{Q}{w} = \frac{1}{0{,}0361} = 27{,}7\ s$$



Obr. 11-5 Průběh napouštění a vypouštění

11.1 Porovnání simulace Matlab a naměřených dat z PLC

Při realizaci regulátoru v PLC jsem narazil na problém s maximální možnou periodou vzorkování, kterou mi dovolí vzorkovat vstupní a výstupní signály připojené na vstupy a výstupy regulátoru Perioda vzorkování je stanovena blokem OB31 a to maximálně 60 vteřin. To je však problém, protože při takovém vzorkování je diskrétní ekvivalent soustavy na mezi stability a regulátor navržený na spojitou soustavu a testovaný v Matlab/Simulink kvůli tomu nefungoval. Realizaci regulátoru pro delší časové konstanty je třeba řešit jiným způsobem,což však nebylo z časových důvodů již možné realizovat.

Ověřením periody vzorkování jsem zjistil že soustavu při vzorkováním po 60s je diskretizovaná soustava s přenosem: $Fs_D = \frac{0{,}1147}{z-0.9971}$, což je pořád blízko meze stability. Proto nelze použít regulátor navržený metodou frekvenčních charakteristik viz kapitola 5.3. Řešením by bylo zmenšit časovou konstantu soustavy, nebo použít jinou vhodnější metodu.

12. ZÁVĚR

Hlavním cílem práce bylo nastudování problematiky tvorby virtuálních modelů a jazyka VRML. Vytvoření v jazyce VRML 3D model technologického zařízení a následného propojení fyzikálního modelu a vytvořeného 3D modelu. Dále propojit komunikaci mezi PLC Siemens S7-1500 a Matlab,/Simulink. V PLC implementovat řízení technologického procesu. Posledním bodem bylo ověření funkčnosti.

Byl zde popsán jazyk VRML a jeho základní uzly. Seznámil jsem se s jazykem VRML a ze získaných zkušeností s tímto jazykem jsem vytvořil jednoduchý virtuální svět a propojil jej s prostředím Matlab/Simulink. Seznámil jsem se také s knihovnou 3D Animation. Bylo ověřeno, že virtuální statický svět lze propojit s dynamikou vytvořenou v Simulinku. Fyzikální dynamika modelu byla navržena podle rovnic, které vyplývají z fyzikálních zákonů a jsou platné ve virtuálním světě podobně jako v reálném. Ačkoliv ve virtuálním modelu jsem některé vlastnosti a parazitní jevy zanedbal, je podstata reálného systému zachována i ve virtuálním světě. Podařilo se mi zprovoznit komunikaci mezi zařízeními tak, aby byla automatická, a možností rychlé identifikace chyby připojení a možností obnovy komunikace. Dále jsem navrhl přenosové funkce dat tak, aby nedocházelo k jejich ztrátám. Vždy je ověřováno, jestli jsou data validní a jestli paket dat byl přijat korektně. Dále jsem navrhl automatický program pro simulaci technologického procesu a také ovládání procesu v manuálním režimu. Byla ověřena funkčnost a bylo zjištěno několik nedostatků při navrhování regulátoru. Pro regulaci teploty tak velkého objemu vody, kdy časová konstanta je obrovská a při vzorkování 1 vteřiny je diskrétní ekvivalent soustavy na mezi stability, je nutné vzorkovat mnohem méně například po 100 a více sekundách. Během testování jsem se setkal s velkou náročností na výpočetní výkon. Při spuštění Programu v Matlab/simulink s běžící komunikací a vizualizací v Realtime režimu, se využití procesoru PC zvedlo na 100%. Tím byla dokázána vysoká náročnost na výpočetní výkon. Nevýhodou zvoleného systému je doba ohřevu velkého objemu, vhodnější by bylo model zmenšit a zkrátit dobu ohřevu na maximálně 3-5 minut, s ohledem na zmenšení příkonu otopného zařízení, aby nebyl neúměrně výkonný na zmenšený objem vody. Po tomto opatření by bylo možné použít tuto aplikaci ve výuce a umožnit studentům navrhovat regulátory přímo do PLC.

Literatura

[1] ZRZAVÝ, Jakub. VRML: Tvorba dokonalých www stránek, podrobný průvodce. Praha: Grada Publishing spol. s r.o., 1999. ISBN 80-7169-643-9.

[2] ŽÁRA, Jiří. VRML 97: laskavý průvodce virtuálními světy. Brno: Computer Press, 1999. ISBN 80-7226-143-6.

[3] PIVOŇKA, Petr. Výpočetní Technika v Automatizaci. Brno, 2003 [cit. 2018-01-02]. Elektronické Skriptum. VUT Brno. Autor textu: Prof.Ing.Petr Pivoňka,CSc.

[4] The MathWorks, Inc. https://www.mathworks.com/help/index.html. Brno, 2003 [cit. 2018-03-02].

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

[6] JIRGL, Miroslav. Laboratorní cvičení BPGA, cvičení se systémy Siemens, VUT FEKT Brno, únor 2018.

[7] BLAHA, Petr. VAVŘÍN, Petr. Řízení a regulace 1:Základy regulace lineárních systémů, spojité a diskrétní, VUT FEKT Brno.

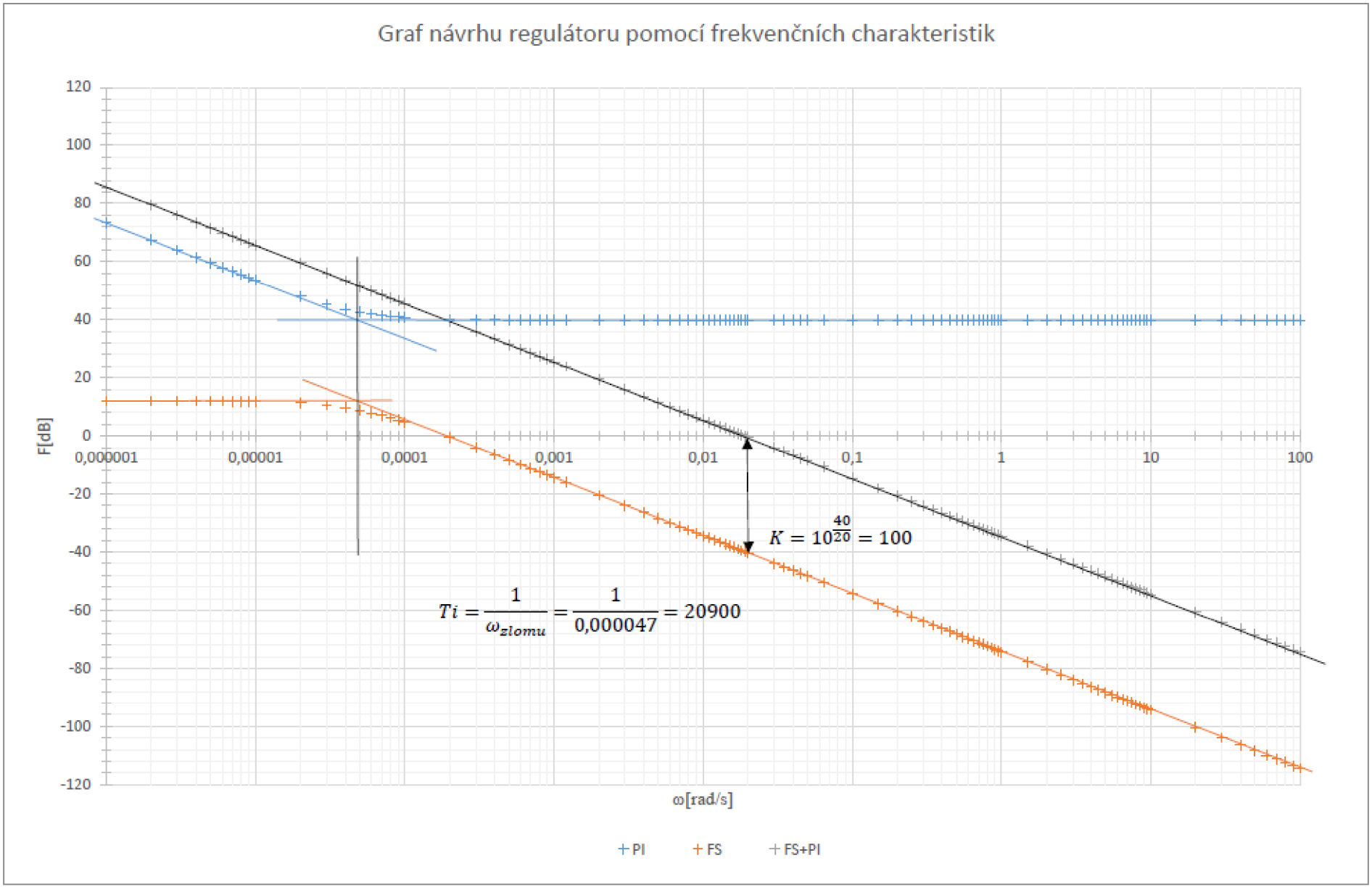
[8] JURA, Pavel. Signály a Systémy Část 2: Spojité systémy, VUT FEKT Brno, únor 2017.

[9] SIEMENS, Industry Online support. https://support.industry.siemens.com [cit. 2018-03-24].

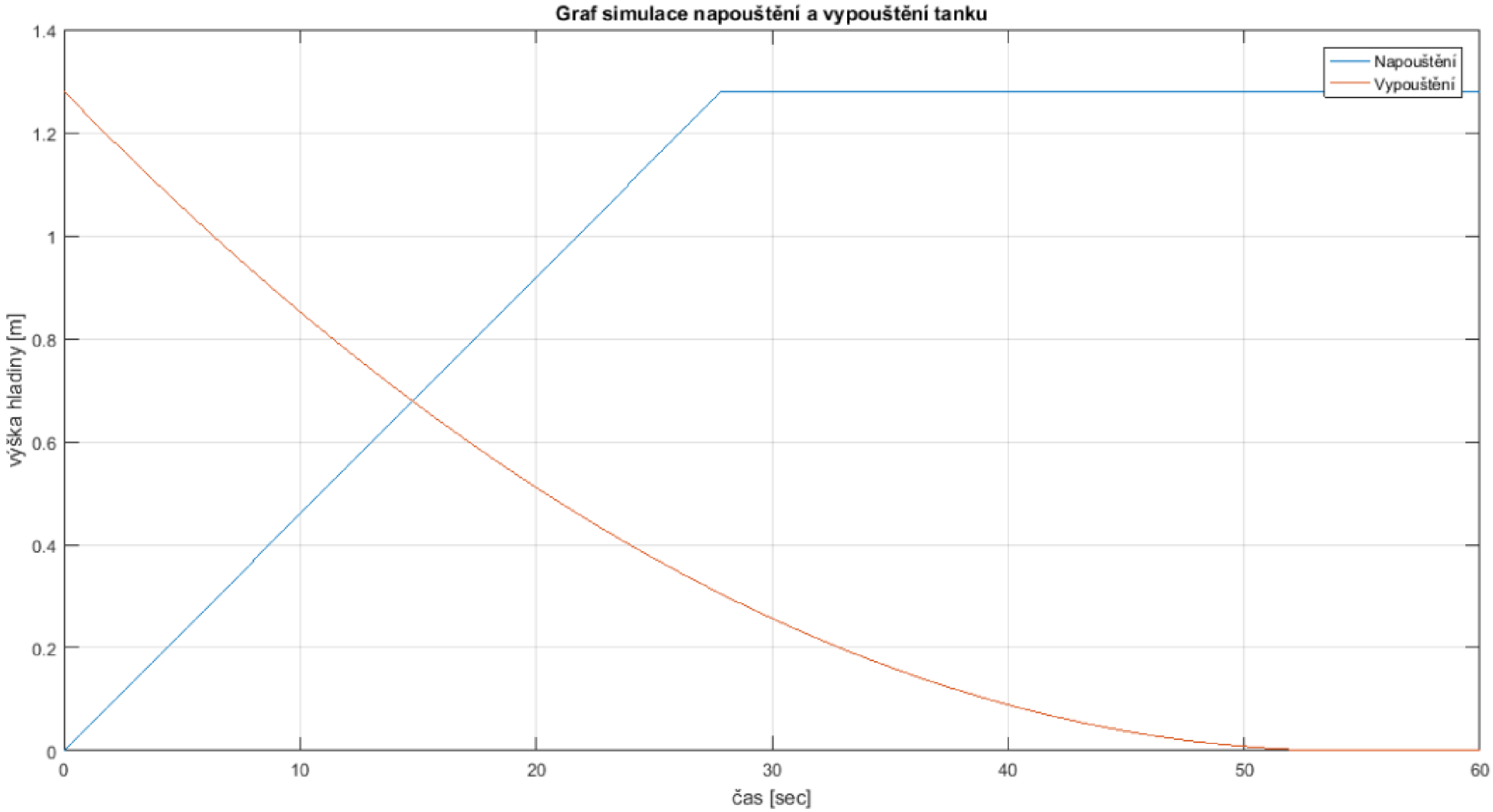
Seznam příloh

NÁVRH REGULÁTORU METODOU LAFCH I
SIMULACE NAPOUŠTĚNÍ A VYPOUŠTĚNÍ TANKU II
KOMPLETNÍ MODEL V MATLAB/SIMULINK III
CD S KOMPLETNÍ PŘÍLOHOU VČETNĚ PROJEKTU TIA PORTAL.....

NÁVRH REGULÁTORU METODOU LAFCH



SIMULACE NAPOUŠTĚNÍ A VYPOUŠTĚNÍ TANKU



KOMPLETNÍ MODEL V MATLAB/SIMULINK