TALLINNA TEHNIKAÜLIKOOL
INSENERITEADUSKOND
Elektroenergeetika ja mehhatroonika instituut

MASINÕPPEL PÕHINEVA KOORMUSE HINNAPÕHISE JUHTIMISALGORITMI UURIMINE NING VÄLJATÖÖTAMINE ENERGIAPAINDLIKKUSE SAAVUTAMISEKS

RESEARCH AND DEVELOPMENT OF A PRICE-BASED LOAD CONTROL ALGORITHM TO ACHIEVE ENERGY FLEXIBILITY BY IMPLEMENTING MACHINE LEARNING

MAGISTRITÖÖ

Üliõpilane: Martin Sarap

Üliõpilaskood: 183361AAVM

Juhendaja: Vahur Maask, nooremteadur

Tallinn 2021

AUTORIDEKLARATSIOON

Olen koostanud lõputöö iseseisvalt.
Lõputöö alusel ei ole varem kutse- või teaduskraadi või inseneridiplomit taotletud.
Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

“.....” ..... 202…..

Autor: .....
/ allkiri /

Töö vastab bakalaureusetöö/magistritööle esitatud nõuetele

“.....” ..... 202…..

Juhendaja: .....
/ allkiri /

Kaitsmisele lubatud

“.....”.....202… .

Kaitsmiskomisjoni esimees .....
/ nimi ja allkiri /

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina Martin Sarap

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Masinõppel põhineva koormuse hinnapõhise juhtimisalgoritmi uurimine ning väljatöötamine energiapaindlikkuse saavutamiseks“, mille juhendaja on Vahur Maask

1.1 reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.2 üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.

3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

07.06.2021

---

[1] *Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.*

LÕPUTÖÖ LÜHIKOKKUVÕTE

Autor: Martin Sarap *Lõputöö liik:* Magistritöö

Töö pealkiri: Masinõppel põhineva koormuse hinnapõhise juhtimisalgoritmi uurimine ning väljatöötamine energiapaindlikkuse saavutamiseks

Kuupäev: 82 *lk*
07.06.2021

Ülikool: Tallinna Tehnikaülikool

Teaduskond: Inseneriteaduskond

Instituut: Elektroenergeetika ja mehhatroonika instituut

Töö juhendaja: nooremteadur Vahur Maask

Sisu kirjeldus:

Antud magistritöö eesmärgiks on uurida ja välja töötada masinõppel põhineva juhtimisalgoritmi loomist energiapaindlikkuse saavutamiseks. Kuna elektrienergia turuhinnad on tugevalt seotud elektrisüsteemi koormuse ning tootmise hulgaga, siis piisab paindlikkuse pakkumiseks vaid hinnapõhisest käitumisest ehk tarbida elektrienergiat siis, kui turuhind on madal ning minimeerida kallimal ajal.

Töö raames loodi masinõppel põhinev juhtimisalgoritm, mille ülesanne on intelligentselt juhtida ventilatsiooniseadet vastavalt siseõhu mõõdetud süsinikdioksiidi tasemele ning elektrienergia hinnale. Juhtimisalgoritmi loomiseks kasutati stiimulõpet ehk masinõppe alavormi, mis suudab läbi õppimise leida parima käitumise ning seega luua efektiivse juhtimissüsteemi. Konkreetselt kasutati moodsat stiimulõppe meetodit nimega DDPG. Algoritmi testimiseks kasutati selleks sobivat demoruumi ning sealset juhitavat ventilatsiooniseadet.

Töö tulemusena programmeeriti masinõppe algoritm, mis suutis efektiivselt modelleerida reaalset ventilatsioonisüsteemi ning testimisega demonstreeriti, et see on võimeline standardsest mitteintelligentsest juhtimisalgoritmist efektiivsemalt töötada, olles samal ajal skaleeritav praktiliselt ükskõik kui suure süsteemi jaoks. Seega täideti töö eesmärk ning demonstreeriti masinõppe kasulikkust energeetikas.

Märksõnad: energiapaindlikkus, juhitavad koormused, masinõpe, stiimulõpe, DDPG, ventilatsioonisüsteemid, juhtimisalgoritm.

ABSTRACT

Author: Martin Sarap *Type of the work:* Master Thesis

Title: Research and development of price-based load control algorithm to achieve energy flexibility by implementing machine learning

Date: 07.06.2021 *82 pages*

University: Tallinn University of Technology

School: School of Engineering

Department: Department of Electrical Power Engineering and Mechatronics

Supervisor of the thesis: Early Stage Researcher Vahur Maask

Abstract:

The goal of this master's thesis is to research and develop a machine learning based control algorithm to achieve energy flexibility. Since the market prices of electricity are strongly correlated with the loads on the electric grid and the amount of available supply, price-based behavior is enough to offer flexibility for the grid. Essentially this means consuming electricity when it is priced lower and minimizing consumption when the prices are higher.

The thesis involved creating a machine learning based control algorithm which was tasked with controlling the ventilation of a room based on carbon dioxide levels and the current electricity prices. The algorithm was created using reinforcement learning, which is a subset of machine learning that can learn optimal behavior and thus create an effective control system. Specifically, a modern approach of reinforcement learning called DDPG was used. The algorithm was tested in a suitable demo room and the ventilation system in it.

The result of the thesis was a machine learning algorithm, which could effectively model a real ventilation system and its testing demonstrated its ability to achieve greater results than a traditional manually created control algorithm. It could achieve this while being practically infinitely scalable. Thus, the goal of the thesis was achieved and the applicability of machine learning in power engineering was demonstrated.

Keywords: energy flexibility, demand response, machine learning, reinforcement learning, DDPG, ventilation system, control algorithm.

LÕPUTÖÖ ÜLESANNE

Lõputöö teema:	**Masinõppel põhineva koormuse hinnapõhise juhtimisalgoritmi uurimine ning väljatöötamine energiapaindlikkuse saavutamiseks**
Lõputöö teema inglise keeles:	**Research and development of price-based load control algorithm to achieve energy flexibility by implementing machine learning**
Üliõpilane:	**Martin Sarap, 183361 AAVM**
Eriala:	**elektroenergeetika**
Lõputöö liik:	**magistritöö**
Lõputöö juhendaja:	**nooremteadur Vahur Maask**
Lõputöö ülesande kehtivusaeg:	**6 kuud**
Lõputöö esitamise tähtaeg:	**07. juuni 2021**

_____	_____	_____
Üliõpilane	Juhendaja	Õppekava juht

1. Teema põhjendus

Taastuvenergeetika kasutuselevõtuga tulenev tootmise ebaühtluse tõus toob endaga kaasa ka suureneva vajaduse energiapaindlikkuse järele, mida on võimalik saavutada olemasolevate tarbimisseadmete nutikuse tõstmisega, muutes need juhitavateks koormusteks. Üheks potentsiaalseks juhitavaks koormuseks on hoonete ventilatsioonisüsteemid, mida võib oma olemuse tõttu targalt juhtida, ilma et teenindatava ruumi siseõhu kvaliteet märkimisväärselt halveneks. Läbi hinnapõhise tarbimise, kus tarbimist püütakse ajastada kokku odavama elektrienergia turuhinnaga, on energiasüsteemi vabaturu olemuse tõttu hinnatundlik käitumine kasulik kogu süsteemi stabiilsusele. Seega on antud magistritöö eesmärgiks arendada võimekust utiliseerida potentsiaalset energiapaindlikkuse allikat, milleks on hoonete ventilatsioonisüsteemid. Kuna selline süsteem on matemaatiliselt keeruline ning sisaldab suurel hulgal elemente, ei ole reegli- või mudelipõhine lahendus praktiline ning parimate tulemuste saamiseks on vajalik kasutada masinõppe meetodeid.

2. Töö eesmärk

Töö eesmärgiks on masinõppe abil luua optimaalne hinnapõhine koormuse juhtimisalgoritm.

3. Lahendamisele kuuluvate küsimuste loetelu:

- Hoone ventilatsioonisüsteemi mudeli loomine.
- Võimalike masinõppe meetodite võrdlev analüüs valitud probleemi lahendamiseks.
- Optimaalse juhtimismeetodi ning -algoritmi väljatöötamine.
- Väljatöötatud lahenduse rakendatavuse hinnang reaalsetes süsteemides.

4. Lähteandmed

Mudeli loomiseks ning arendamiseks kasutatakse lähteandmetena asjakohaseid nõudeid, teooriat ning olemasolevaid ventilatsioonisüsteemide mudeleid. Need andmed leitakse erinevatest raamatutest ning teadusallikatest, kasutades vajadusel juhendaja abi. Hinnapõhise juhtimisalgoritmi arendamiseks on vajalikud ka ajaloolised elektrienergia hinnad, mis on võrgust lihtsasti kättesaadavad.

5. Uurimismeetodid

Tulemusteni jõutakse läbi eduka modelleerimise ning masinõppe meetodite rakendamisega, mis toimub Matlab tarkvara abil. Tulemuste kontrollimiseks on vajalik läbi viia ka mõõtmised reaalsete katseseadmetega.

6. Graafiline osa

Töö põhiosas on selgitamiseks ning andmete esitamiseks kasutatud erinevaid jooniseid ja skeeme.

7. Töö struktuur

1. MASINÕPPE KASUTAMINE KOORMUSE JUHTIMISEL
   - 1.1 Paindlikud koormused energiavõrgus
   - 1.2 Ventilatsioonisüsteemid
   - 1.3 Intelligentsed süsteemid
   - 1.4 Masinõpe
     - 1.4.1 Juhendatud õpe
     - 1.4.2 Juhendamata õpe
   - 1.5 Stiimulõpe
     - 1.5.1 MDP
   - 1.6 Tehisnärvivõrgud
   - 1.7 Sügavõpe, Q-õppimine, DDPG ja mudeli treenimine
     - 1.7.1 Q-õppimine
     - 1.7.2 DDPG
     - 1.7.3 Mudeli treenimine
2. MODELLEERIMINE
   - 2.1 Sisendandmete ettevalmistamine

2.1.1 Ruumis olevate inimeste modelleerimine

2.1.2 Keskkonna modelleerimine

2.2 Sammfunktsioon

2.3 Stiimulõppe algoritmi modelleerimine

2.3.1 Agendi treenimine

2.4 Tulemuste analüüs

2.4.1 Käsitsi loodud mudel

2.4.2 Stiimulõppe mudel

2.4.3 Võrdlus pikema aja peale

3. TESTIMINE REAALSEL KOORMUSEL

3.1 Testimise ettevalmistamine

3.2 Tulemuste analüüs ja algoritmi valideerimine

3.2.1 Simuleeritud ja reaalne CO_2 doseerimine

3.2.2 Testimise tulemused

3.2.3 Võrdlus lihtsa juhtimisstrateegiaga

3.2.4 Võrdlus simuleeritud ja testitud efektiivsuste vahel

3.3 Juhtimisalgoritmi rakendatavuse hinnang reaalsetes süsteemides

3.3.1 Mudeli edasi arendamine

4. MÕJU ELEKTRISÜSTEEMILE

4.1 Virtuaalne aku

4.1.1 Hinnapõhise juhtimise mõju süsteemi koormuskõverale

4.2 Koormuskõvera silumine

4.2.1 Päikesepaneelide tootmiskõveraga kohandumine

8. Kasutatud kirjanduse allikad

Allikatena on kasutatud erinevaid masinõppe, ventilatsiooni ning energeetika teemalisi raamatuid, teadusartikleid ning veebimaterjali. Samuti ka testseadmete dokumentatsioone ning standardit.

9. Lõputöö konsultandid

Lõputöö koostamisel konsulteeriti juhendajaga.

10. Töö etapid ja ajakava

Kirjanduse läbitöötamine (sügis - talv 2020)

Teoreetilise osa kirjutamine (sügis - talv 2020)

Modelleerimise teostamine (talv - kevad 2021)

Mudeli valideerimine ja mõõtmiste läbiviimine (märts - aprill 2021)

Järelduste ja kokkuvõtte koostamine (aprill 2021)

Töö lõplik versioon valmis (mai 2021)

Paranduste tegemine ja viimase peatüki lisamine (juuni 2021)

SISUKORD

EESSÕNA

Lõputöö teema andis välja Tallinna Tehnikaülikooli Elektroenergeetika ja mehhatroonika instituudi mikrovõrkude ja metroloogia uurimisrühm. Teema valiku põhjuseks oli isiklik huvi masinõppe ning intelligentsete juhtimissüsteemide vastu. Lõputöö koostamisel ja testimise läbiviimisel oli suureks abiks juhendaja Vahur Maask.

LÜHENDITE JA TÄHISTE LOETELU

Add	närvivõrgu kiht, mis liidab väärtused kokku (*addition*)
a_t	stiimulõppe mudeli tegevus ajasammul t (*action*)
C_G	soovitud CO_2 kontsentratsioon (ppm)
C'_G	soovitud CO_2 kontsentratsiooni suhteline hulk
C_{gen}	ühe inimese poolt toodetud CO_2 kogus
C_{max}	maksimaalne lubatud CO_2 kontsentratsioon (ppm)
C_O	välisõhu CO_2 kontsentratsioon (ppm)
C'_O	välisõhu suhteline CO_2 kontsentratsioon
C'_t	ruumi siseõhu suhteline CO_2 kontsentratsioon ajasammul t
$\Delta C'_{gen}$	CO_2 suhteline tõus ruumis inimese kohta
DDPG	töös kasutatud sügav stiimulõppe meetod (*Deep Deterministic Policy Gradient*)
E_t	elektrienergia reaalne maksumus ajasammul t (€)
E_{tmax}	ajasammule t vastava päeva maksimaalne elektrienergia hind (€/MWh)
E'_t	elektrienergia suhteline hind ajasammul t
E'_{t+1}	elektrienergia suhteline hind ajasammul $t+1$
E'_{t+3}	elektrienergia suhteline hind ajasammul $t+3$
E'_{t+8}	elektrienergia suhteline hind ajasammul $t+8$
E'_{t+12}	elektrienergia suhteline hind ajasammul $t+12$
I_t	ruumis olevate inimeste hulk ajasammul t
$I_t{}'$	ruumis olevate inimeste suhteline hulk ajasammul t
MDP	Markovi otsustusprotsess (*Markov Decision Process*)
MPC	mudelipõhine juhtimismeetod (*Model Predictive Control*)
P_{max}	ventilaatori maksimaalne võimsus (W)
P_{min}	ventilaatori minimaalne kiirus (W)
P_t	ventilaatori suhteline võimsus ajasammul t
ppm	osakest miljoni kohta (*parts per million*)
R_t	stiimulõppe mudeli kogupreemia ajasammul t
ReLU	närvivõrgu kiht, mis teisendab negatiivsed väärtused nulliks (*Rectified Linear Unit*)
r_t	stiimulõppe mudeli preemia ajasammul t (*reward*)
S'_t	normaliseeritud energiakulu ajasammul t
S_t	juhtimissignaal ajasammul t
s_t	stiimulõppe mudeli olek ajasammul t
t	ajasamm
t_s	stiimulõppe mudeli ajasammu pikkus

V	ruumis oleva õhu hulk (m^3)
ΔV_{max}	ventileeritava õhu hulk maksimaalsel võimsusel (m^3/h)
$\Delta V'_{max}$	ventileeritava õhu suhteline hulk maksimaalsel võimsusel
W_C	CO_2 kontsentratsiooni kaal preemia arvutamisel
W_E	elektrienergia maksumuse kaal preemia arvutamisel
α ja β	preemiafunktsiooni muutumise kiirust määravad tegurid
γ	stiimulõppe mudeli preemia diskontomäär (*discount factor*)
π	stiimulõppe juhtimisstrateegia (*policy*)
$\sum \Delta C_t'$	ajavahemiku summaarne C_t' erinevust etteantud CO_2 tasemest
$\sum E_t$	ajavahemiku summaarne elektrienergia maksumus

SISSEJUHATUS

Taastuvenergeetika laialdasema kasutuselevõtu üheks suurimaks väljakutseks on elektrienergia tootmise mittejuhitavus. Kuna tuul ja päikesepaiste ei ole inimese poolt kontrollitavad ning täpselt ette ennustatavad, siis ei ole võimalik teada, millal ja kui palju elektrienergiat toodetakse. Probleemi üheks lahenduseks on juhitavad koormused, mille raames viiakse juhtimine üle tarbija poolele, et kohaneda muutliku tootmisega ja seega pakkuda võrgule energiapaindlikkust. Tulenevalt elektrienergia kauplemise vabaturu olemusest, väljendub pakkumine ja nõudlus hinnas ning efektiivseks koormuse juhtimiseks piisab hinnapõhisest tarbimisest.

Hoonete ventilatsioonisüsteemid vahetavad välja ruumi siseõhu välisõhu vastu, et tagada vajalikud parameetrid nagu näiteks temperatuur ja CO_2 kontsentratsioon. Selleks kasutavad need süsteemid rohkem kui 2% kogu Euroopa Liidus tarbitavast elektrienergiast [1]. Kuna siseõhu parameetrite väärtused võivad paikneda üsna laias vahemikus ning vajavad muutumiseks aega, siis on ventilatsioonisüsteemi võimalik juhtida vastavalt elektrienergia turuhinnale, pakkudes nii võrgule energiapaindlikkust. Parimate tulemuste saavutamiseks on võimalik seadmete juhtimiseks kasutada intelligentseid masinõppel põhinevaid süsteeme.

Masinõppeks nimetatakse algoritme, mis suudavad andmetelt õppides oma sooritusvõimekust tõsta. Sedasorti õppimine eristab masinõppe meetodeid traditsioonilistest algoritmidest, kus kogu loogika on inimese poolt nö käsitsi kirjutatud. Keeruliste süsteemide juhtimiseks kasutatakse masinõppe alaliiki, mida nimetatakse stiimulõppeks. Sel juhul õpib algoritm õige käitumise positiivsete ja negatiivsete stiimulite kaudu, mis sarnaneb elusate olendite õppimisele. Masinõppe algoritmi tuumaks kasutatakse tihtipeale tehisnärvivõrku, mis sarnaneb olemuselt samuti looduses leitavatele süsteemidele.

Käesoleva töö eesmärgiks on luua stiimulõppel põhinev juhtimisstrateegia, mis suudaks hoone ventilatsioonisüsteemi juhtida selliselt, et saavutada sobiv siseõhu CO_2 kontsentratsioon, minimeerides sealjuures elektrienergia maksumust. Selleks kirjutati Matlab tarkvaras stiimulõppe algoritm, mida treeniti olemasolevate elektrienergia hindade alusel. Treenitud algoritmi testimiseks ühendati see reaalse ventilatsiooniseadmega, mida juhtides ning õhu CO_2 kontsentratsiooni jälgides veenduti algoritmi efektiivsuses. Tulemuste hindamiseks loodi ka lihtne süsteem, mis hoiab CO_2 taseme konstantse ning esindab traditsioonilist mitteintelligentset juhtimisstrateegiat taolistes süsteemides.

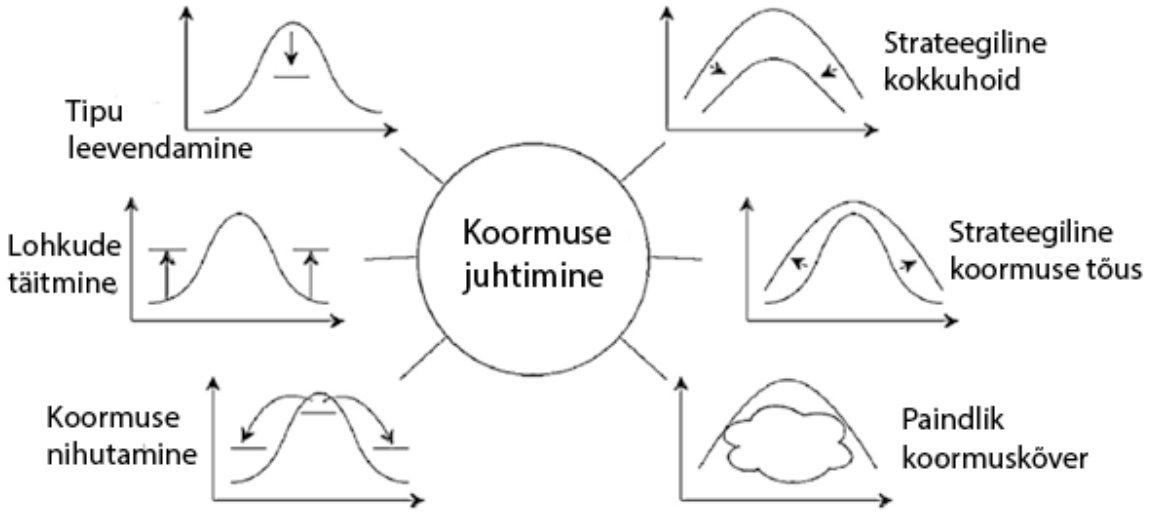
Töö esimeses peatükis antakse ülevaade tarbimise juhtimisest ning ventilatsioonisüsteemidest. Samuti esitatakse järk-järgult süvenedes informatsiooni masinõppe ning sellega seotud meetodite kohta, jõudes peatüki lõpuks antud töös kasutatava meetodi kirjelduseni. Teises peatükis kirjeldatakse stiimulõppe algoritmi jaoks sisendandmete ettevalmistamist, mudeli modelleerimist ja treenimist ning analüüsitakse sellega simuleeritud tulemusi. Töö kolmandas peatükis kirjeldatakse treenitud mudeli testimist reaalse ruumi ning ventilatsiooniseadmega ning hinnatakse taoliste süsteemide kasulikkust reaalsetes hoonetes. Viimases peatükis on analüüsitud mudeli potentsiaali paindlikkuse pakkujana ning hinnatud selle võimalikku mõju elektrisüsteemile. Lisades on esitatud töös kasutatud Matlab tarkvaras loodud koodid.

1. MASINÕPPE KASUTAMINE KOORMUSE JUHTIMISEL

1.1 Paindlikud koormused elektrisüsteemis

Taastuvenergeetika üheks suurimaks puuduseks on tootmise mittejuhitavus, kuna see sõltub suuresti ilmastikuoludest, mistõttu on tarbimise ja tootmise tasakaalu saavutamine igal ajahetkel raskendatud. Üheks lahenduseks on energia salvestamine, kuid üldjuhul osutub see hinna tõttu ebapraktiliseks. Üle jääb seega tarbimise juhtimine ehk paindlikud koormused, mille raames juhitakse süsteemis olevaid tarbimisseadmeid vastavalt tootmisele.

Tihtipeale on koormuse juhtimise peamiseks eesmärgiks vähendada tarbimist tipukoormuse ajal, et muuta koormuskõvera tippu lamedamaks. Kuid mõnel juhul võib lamedamale kõverale eelistatum olla kõver, mis ühtib tootmiskõveraga. Koormuse juhtimise abil on võimalik saavutada erinevaid eesmärke, nagu näiteks tipu leevendamine, lohkude täitmine, koormuse nihutamine, strateegiline kokkuhoid, strateegiline koormuse tõstmine ning paindlik koormuskõver [2]. Antud meetodid on graafiliselt esitatud joonisel 1.1.



Joonis 1.1 Erinevat tüüpi koormuse juhtimised [2]

Tarbimise juhtimine võimaldab läbi hinnasignaalide elektrienergia tarbimist dünaamiliselt muuta. See pakub mitmeid eeliseid. Energia hindu muutes on võimalik tarbimise koormust nihutada tipu aegadelt teistele perioodidele, vähendades nii tipu ja keskmise koormuse vahelist suhet. See omakorda parandab elektrisüsteemi efektiivsust, vähendab kulusid, tõstab kapitali tõhusust ning vähendab saasteainete loomist ning rikete ohtu [3]. Ajapõhiste hindade või muude rahaliste ajendite abil on võimalik klienti motiveerida oma tarbimist tipu aegadel vähendama. Seeläbi on neil

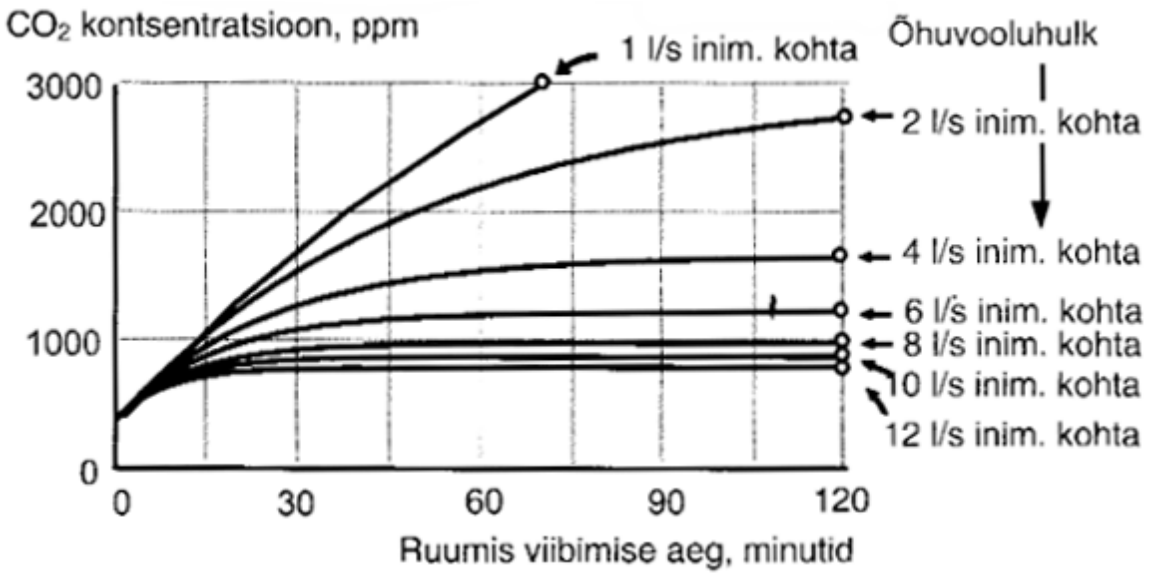
võimalik raha kokku hoida ning samal ajal vähendada koormust süsteemile. Kasutusel on ka lahendused, kus võrguettevõttel on otsene ligipääs tarbijate seadmetele, mis võimaldab neid vastavalt vajadusele sisse ja välja lülitada [4]. Kuid kõige lihtsam viis klientide kaasamiseks tarbimise juhtimisse on luua keskkond, kus energia tarbimine on kallim tipu aegadel ning odavam väiksema koormusega aegadel. Levinuim meetod selleks on otseselt siduda elektrienergia hind kliendile antud ajahetke turuhinnaga [5]. Seeläbi on tarbijal võimalik pakkuda elektrisüsteemile paindlikkusvõimekust vaid hinnale reageerides [6].

Reaalajas muutuva hinna puhul on tarbijal lõpmata palju valikuid otsustamaks, millal ja kui palju energiat tarbida. Optimaalne otsustamine nõuab praeguste ning tulevaste hindade arvestamist ning energia kulu võrdlemist tarbimise väärtusega. Kuna selline juhtimine on inimese jaoks võimatu, tuleb kasutusele võtta automatiseeritud energia majandamise süsteemid [3].

1.2 Ventilatsioonisüsteemid

Tavalise ventilatsioonisüsteemi saab jaotada kaheks osaks: kütte/jahutuselement ning ventilaator, mis tsirkuleerib õhku. Üldjuhul antakse süsteemile ette soovitud ruumi siseõhu temperatuuri väärtus ning seade püüab temperatuurianduri väljundi põhjal seda saavutada. Kuid lisaks temperatuurile reguleerib ventilatsioonisüsteem ka siseõhu suhtelist niiskussisaldust ning süsinikdioksiidi kontsentratsiooni. Antud töö raames käsitletakse neist viimast.

Süsinikdioksiidi ehk CO_2 sisaldus õhus on tavaliseks õhu kvaliteedi mõõteks ruumides, kus viibib inimesi. Tavaõhk sisaldab 21% hapnikku ning ligikaudu 0,04% (400 ppm) süsinikdioksiidi. Inimese poolt väljahingatud õhk sisaldab ligikaudu 18% hapnikku ja 1% süsinikdioksiidi. Seega suureneb hõivatud ruumis inimeste hingamise tõttu süsinikdioksiidi tase [7]. Joonisel 1.2 on esitatud süsinikdioksiidi kontsentratsiooni tõus inimestega asustatud ruumis erinevate õhuvoolu hulkade korral.



Joonis 1.2 CO_2 kontsentratsiooni tõus [7]

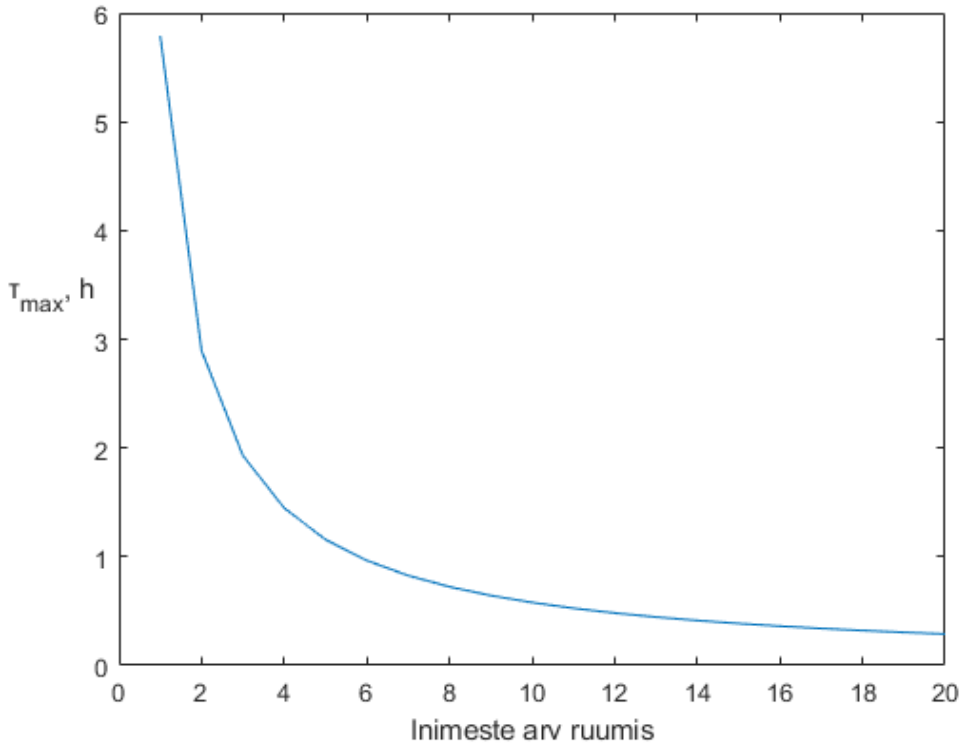
Nagu jooniselt näha, siis madala õhuvoolu puhul tõuseb ruumi siseõhu CO_2 tase kiiresti. Kuigi ohtlikuks peetakse alles kontsentratsioone üle 5000 ppm, siis üldiste soovituse kohaselt ei tohiks CO_2 kontsentratsioon ületada 1000 ppm piiri [8],mis on ka antud töö puhul võetud ülemiseks piiriks.

Maksimaalset aega τ_{max} (s), mis ventilatsioonisüsteem tohib olla välja lülitatud, saab leida järgmise valemiga [9]:

$$\tau_{max} = \frac{\Delta C \cdot V}{G} = \frac{\Delta C \cdot V}{I \cdot Q_{CO2,gen} \cdot 10^6} \tag{1.1}$$

Kus ΔC on maksimaalse lubatud CO_2 kontsentratsiooni erinevus algväärtusest (ppm),
V on ruumi siseõhu maht (m^3),
G on CO_2 tootmise kiirus (m^3/s),
$Q_{CO2,gen}$ on ühe inimese poolt toodetud CO_2 (m^3/s),
I on inimeste hulk ruumis.

Kui näiteks võtta antud töö raames uuritav ruum, mille ruumala V on 193 m^3, siis saab arvutada τ_{max} suuruse erinevate inimeste hulkade korral, mis on esitatud joonisel 1.3. Sealhulgas $Q_{CO2,gen}$ väärtuseks võetakse vastavalt EN 16798 standardile $5{,}56 \cdot 10^{-6}$ m^3/s ehk ligikaudu 20 L/h [10] ning ΔC 600 ppm (välisõhuks on võetud 400 ppm ning maksimaalseks lubatud kontsentratsiooniks 1000 ppm).



Joonis 1.3 CO_2 kontsentratsiooni tõus maksimaalsele tasemele ventileerimata ruumis

Graafikult on näha, et ilma ventilatsioonita tõuseb ruumi siseõhu CO_2 kontsentratsioon väga kiiresti üle maksimaalse lubatud taseme (tavaliselt on see arv küll suurem, kuna eeltoodud valemis arvestatakse täielikult õhukindla ruumiga). Kuna temperatuuri ja suhtelise niiskuse korral on analoogne aeg märkimisväärselt kõrgem [9], siis on automaatsüsteemi loomisel mõistlik lähtuda just sellest suurusest. Sel põhjusel on ka antud töö raames käsitletud vaid CO_2 kontsentratsiooni.

1.3 Intelligentsed süsteemid

Tööstuslike protsesse juhtivad süsteemid peavad jälgima ning optimeerima paljusid erinevaid parameetreid, kuid tihtipeale juhitakse neid manuaalselt või nõrkade kontrolleritega, mis ei suuda tagada piisavalt efektiivset juhtimist. Juhtimissüsteemide keerukuse, tõhususe ning nõutava paindlikkuse kasvuga on kaasnenud vajadus juhtsüsteemide vastu, mis tulevad toime mittelineaarsete, ajas muutuvate ning osaliselt tundmatute süsteemidega. Seega on vaja kasutusele võtta intelligentsed juhtimissüsteemid. Arvutite riist- ning tarkvara tehnoloogiate arengu tõttu on intelligentsed juhtimissüsteemid kogunud aina rohkem populaarsust, kuna need meetodid ei vaja enam võimsaid superarvuteid ning arendusi tehisintellekti tehnoloogias on võimalik rakendada üldotstarbeliste arvutitega. Isegi reaalajas juhtimine on saanud lihtsasti teostatavaks [11].

Eelnevate kogemuste ning teadmiste põhjal õppimine, mis propageerib ühest generatsioonist järgmisesse, on inimese intelligentsuse aluseks. Samuti edeneb teadus mudelite ja teooriate arendamisega, et selgitada eksperimentaalselt kogutud tõendeid. Ehk teisisõnu me kasutame õppimiseks alati andmeid ning andmete tüüp ja tõlgendamise meetod defineerib erinevad teadusharud. Andmete põhjal õppimist on uuritud aastakümneid paljudes erinevates suundades – statistika, andmekaeve, masinnägemine, bioinformaatika, mustrituvastus, signaali- ja pilditöötlus, masinnägemine, tööstuslik automatsioon jpm. Vaatamata erinevatele nimedele, põhinevad need kõik sarnastel meetoditel, mille võib koondada ühe mõiste alla – masinõpe. Sellest nimetusest võib järeldada, et arvuti õpib analoogselt inimese ajule. Mõnel juhul on meetodid saanud otseselt inspiratsiooni aju tööpõhimõtetest ning kasutatakse tehisnärvivõrke [12]. Seega masinõppimise eesmärk on välja töötada meetodeid, mis suudavad automaatselt leida mustreid andmehulkades ning kasutada neid tuleviku andmete või muude huvipakkuvate suuruste ennustamiseks [13].

Traditsiooniliselt on ventilatsioonisüsteeme juhitud mudeli- või reeglipõhiste meetoditega. Mudelipõhised juhtimismeetodid nagu näiteks MPC (*Model Predictive Control*) sobituvad komplekse termodünaamikaga ning suudavad saavutada üksiku hoone puhul primaarenergia kokkuhoidu. Kuid olemasolevale hoonele sellise süsteemi lisamine nõuaks termo-energeetilise mudeli loomist ja valideerimist. Mudelipõhised meetodid on seotud mudeli kvaliteediga ning täpse mudeli loomine on kallis ja keeruline. Mudelipõhist süsteemi kirjeldab kõrge alginvesteering, keerukus integreerida igapäevastesse ehitusprojektidesse ning raskused skaleerimisel [14]. Lisaks kui hoonet muudetakse energia efektiivsuse eesmärkidel või toimuvad ümberehitused muudel põhjustel, siis peab mudeli uuesti üles ehitama ning seadistama, mis osutub järjekordselt kalliks [15].

Reeglipõhised meetodid on moodsad mudelivabad juhtimismeetodid, mis on kujunenud tööstuses standardseks. Mudelivaba meetodit on potentsiaalselt võimalik üles skaleerida, kuna mudeli puudumine muudab süsteemi rakendamise erinevatele hoonetele lihtsamaks [15]. Reeglipõhiste lahenduste peamiseks miinuseks on raskused optimaalsete sätete leidmisel, kuna need ei ole piisavalt kohanduvad, et vastata hoonete ja termodünaamika keerukusele [16].

Hiljuti on populaarsust kogunud andmepõhised masinõppe algoritmid ning eelkõige stiimulõppe meetodid on ventilatsioonisüsteemide juhtimisel demonstreerinud edukust, kuna need tulevad stohhastilistes keskkondades hästi toime. Stiimulõppe eeliseks on nii mudelivaba kui ka kõrgelt kohanduv juhtimine. Algoritm suhtleb otse

ventilatsioonisüsteemiga ning kohaneb jooksvalt kontrollitava keskkonnaga läbi reaalajas kogutud andmete, vajamata selleks keerulist hoone termo-energeetilist mudelit. Seetõttu suudab stiimulõppel põhinev lahendus vähendada primaarenergia kasutust, olles samal ajal säästlik ning sobiv suuremahuliseks kasutuseks [16].

1.4 Masinõpe

Masinõppe meetodeid kasutatakse juhul kui K sisendi (x) ja väljundi (y) paari

$$(x_1, y_1), (x_2, y_2), \dots, (x_K, y_K) \tag{1.2}$$

M-dimensionaalsete sisenditega

$$x_k = (x_{k1}, x_{k2}, \dots, x_{kM}) \; ; \; k = 1 \dots K \tag{1.3}$$

ning vastavate N-dimensionaalsete väljunditega

$$y_k = (y_{k1}, y_{k2}, \dots, y_{kN}) \; ; \; k = 1 \dots K \tag{1.4}$$

on teada, kuid sisendite ja väljundite vahel seoseid loovad mudeli funktsioonid f_i

$$y_i = f_i(x_1, x_2, \dots, x_M) \; ; \; i = 1 \dots N \tag{1.5}$$

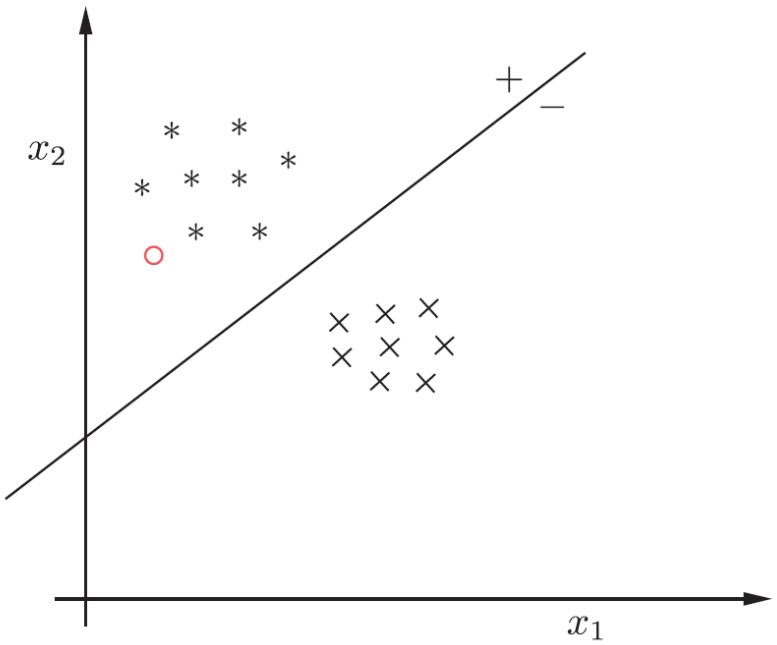
on täielikult teadmata. Masinõpe püüab neid funktsioone aproksimeerida, kasutades selleks sisend- ja väljundandmeid ehk treeningandmestikku [17]. Masinõppe meetodid jagatakse enamasti kolmeks [13]:

- juhendatud õpe
- juhendamata õpe
- stiimulõpe

1.4.1 Juhendatud õpe

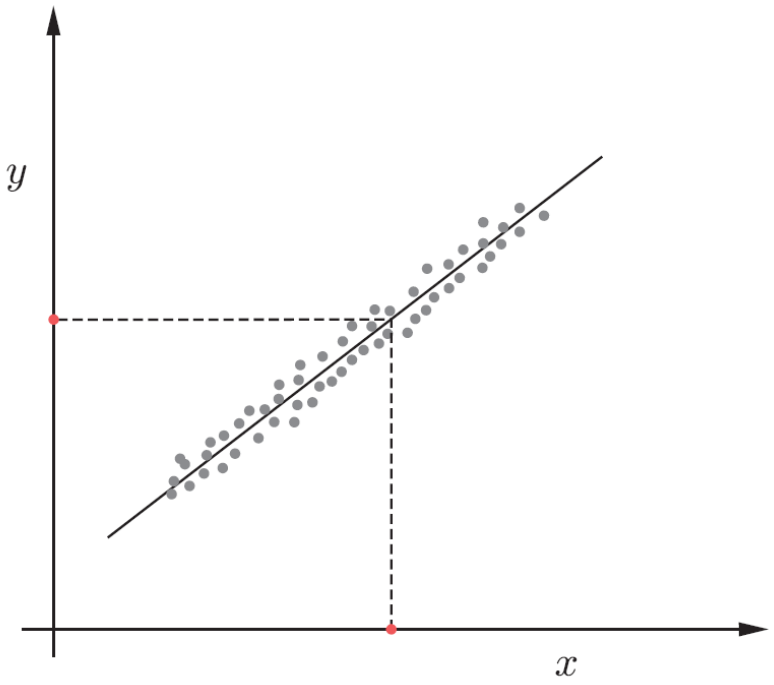
Lihtsaimal juhul on sisend x_k M arvust koosnev vektor, mille elemendid tähistavad konkreetseid andmepunkte, näiteks inimeste pikkuseid ja kaale. Neid nimetatakse tunnusteks või atribuutideks. Kuid üldiselt võib x_k olla keeruline struktuuriga objekt nagu näiteks pilt, lause, ajaseeria, molekuli kuju, graaf jne. Sarnaselt võib ka väljund y_k olla mistahes objekt, kuid tavaliselt on see mingi lõplikkusse hulka kuuluv element (näiteks mees või naine) või skalaar. Esimesel juhul on tegu klassifikatsiooniga ning teisel juhul regressiooniga. Mõlema variandi puhul on tegu juhendatud masinõppega [13].

Klassifikatsiooni ehk liigitamise puhul on eesmärgiks õppida seos sisendi x_k ja väljundi y_k vahel, kus $y \in \{1, \dots, C\}$ ning C on klasside arv. Kui $C = 2$, siis on tegu binaarse klassifitseerimisega ning kui $C > 2$, on tegu *multiclass* klassifitseerimisega [13]. Joonisel 1.4 on esitatud lihtsa lineaarse klassifikatsiooni väljundgraafik (klassifitseerijaks on siin diagonaalne joon). Antud juhul püüab mudel sisendandmed jaotada kaheks klassiks olenevalt sellest, kas andmepunkt asub klassifitseeriast positiivses või negatiivses suunas. Ristid ja tärnid esindavad treeningandmeid, mille põhjal klassifitseerija on loodud. Punane ring esindab tundmatut andmepunkti, mida püütakse klassifitseerida ning kuna see asub antud näites klassifitseerijast positiivses suunas, siis sildistatakse see tärniks [12].



Joonis 1.4 Lihtne klassifitseerija [12]

Teiseks juhendatud masinõppe liigiks on regressioon, mis erineb eelnevast meetodist peamiselt selle poolest, et väljund y_k ei ole diskreetne, vaid omab pidevat reaalarvulist väärtust (mõnel juhul võib väljund olla ka kompleksarv). Sisuliselt on regressiooni puhul tegu kõvera sobitamise ülesandega (*curve fitting*). Ette on antud mingi hulk treeningandmeid ning eesmärgiks on leida funktsioon f, mille graafik nendega võimalikult hästi sobib. Kui selline funktsioon on leitud, saab leida tundmatule sisendile vastava väljundi. Joonisel 1.5 on esitatud lineaarne väljundgraafik, kuid üldiselt kasutatakse masinõppe meetodeid sel juhul, kui on kasutada on vaja kõrgema dimensioonilist või muud moodi keerulisemat funktsiooni [12] [13].

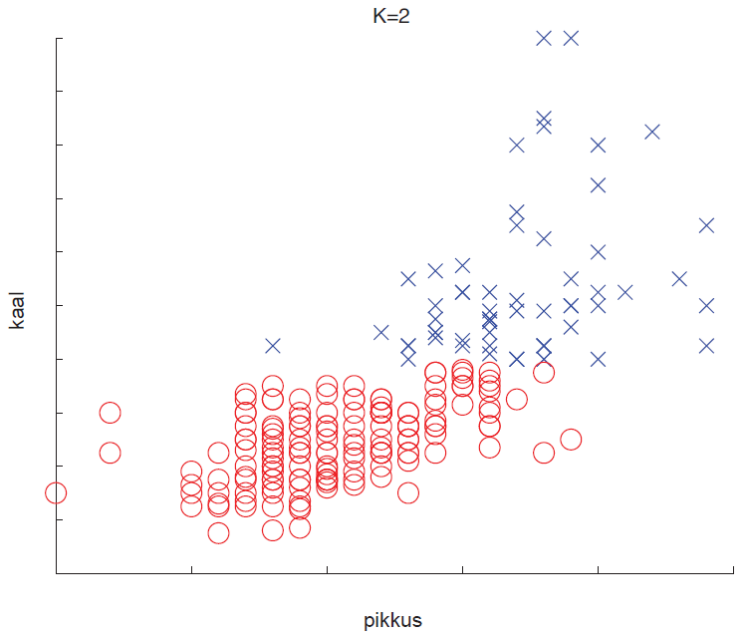


Joonis 1.5 Lihtne regressioon [12]

Mõlemal juhul hinnatakse valmis mudelit vea suuruse järgi, mille saab lihtsasti arvutada loodud funktsiooni väljundi y_i ning teadaoleva, ette antud väljundi erinevuse abil. Kuna treeningandmete kasutamine mudeli võimekuse hindamisel on ebaotstarbekas, siis kasutatakse selleks treenimiselt välja jäetud andmeid ehk valideerimisandmestikku [13].

1.4.2 Juhendamata õpe

Juhendamata masinõppe puhul on ette antud vaid sisendandmed x_k ilma vastavate väljunditeta ning eesmärgiks on nende seast leida “huvitavaid” mustreid. See on vähem defineeritud ülesanne, kuna ei ole ette antud, missuguseid mustreid otsida, ning puudub ka selge suurus, mida kasutada veana. Kõige levinum näide juhendamata õppe kohta on klasterdamine (*clustering*) ehk andmete grupeerimine. Joonisel 1.6 on esitatud graafik kaheks grupiks jagatud andmepunktidega. Selles näites on masinõppe algoritm jaganud pikkuse ja kaalu järgi inimesed meesteks ja naisteks. Antud juhul on selge, et moodustuvad kaks erinevat gruppi, kuid üldjuhul ei ole see teada ning gruppide arv tuleb ise määrata. Samuti ei ole kohe selge, millised sisendandmed mõjutavad gruppi kuulumist rohkem ja millised vähem, mistõttu annab töötav mudel informatsiooni ka omaduste “tähtsuse” kohta. Näiteks näotuvastusalgoritmide puhul võib juhtuda, et masin keskendub enim sellistele näo omadustele, mida inimesed näo ära tundmisel tingimata oluliseks ei pea [13].



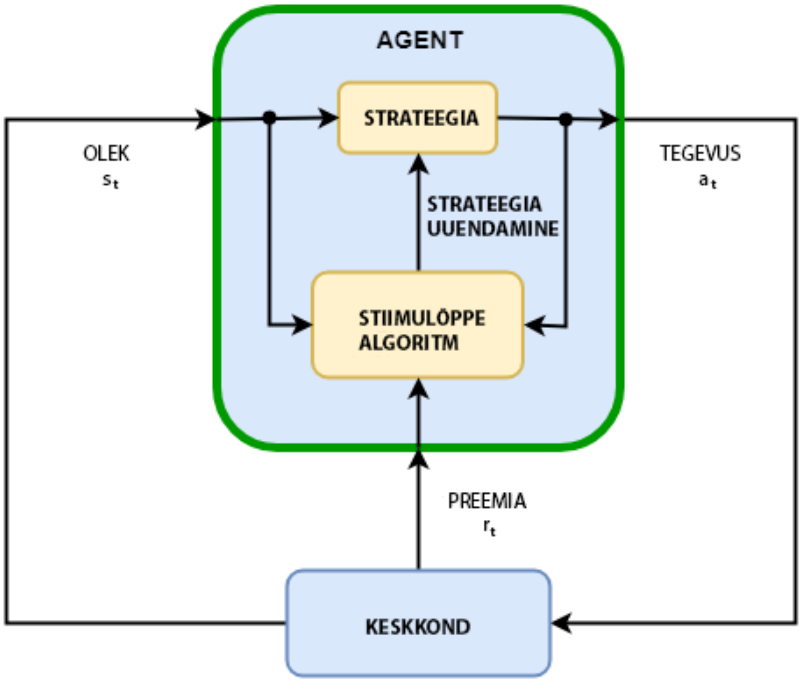
Joonis 1.6 Andmete grupeerimine [13]

Kolmandaks masinõppe liigiks on stiimulõpe, mida kasutatakse käitumise õppimiseks ning on käesolevas töös kasutatav meetod, mistõttu on see põhjalikumalt kirjeldatud järgmises alapeatükis.

1.5 Stiimulõpe

Stiimulõpe on tegevuse õppimine – kuidas kaardistada situatsioonid tegudele sellisel viisil, et maksimeerida preemiasignaal. Õppivale mudelile ei ole ette antud, missuguseid tegusid sooritada, vaid see peab neile vastava preemia väärtuse iseseisvalt katsetades leidma. Enamjaolt ei mõjuta ühe situatsiooni sooritus vaid antud ajahetke preemiat, kuid ka järgmisi situatsioone ning seeläbi ka järgmisi preemiaid. Taoline hilinenud preemia ning kasutatav katse-eksitus meetod on stiimulõppe kõige olulisemad omadused [18].

Stiimulõppe algoritm koosneb kahest peamisest elemendist – agent ning keskkond. See kujutab endast interaktsiooni agendi ja keskkonna vahel, millele järgneb hinnang ning preemia jagamine Agent on õppija ja otsuste läbiviija ning keskkond on see, millega agent suhtleb. Igal ajasammul t saab agent sisendiks oleku (*state*) s_t ning viib läbi tegevuse (*action*) a_t. Süsteemi olekuks nimetatakse agendi poolt mõõdetud keskkonna väärtuseid antud ajasammul. Tegevuse sooritamine liigutab keskkonna olekust s_t olekusse s_{t+1}, saades seeläbi preemia (*reward*) r_t, mis sõltub tegevuse efektiivsusest. Peale seda saab agent sisendiks oleku s_{t+1} ning algab järgmine tsükkel [19] [20]. Agent valib tegevuse juhtimisstrateegia (*policy*) $\pi: a_t = \pi(s_t)$ järgi, mis kirjeldab tegevust a_t oleku s_t puhul [21]. Stiimulõppe protsessi kirjeldav skeem on esitatud joonisel 1.7.



Joonis 1.7 Stiimulõppe protsess [22]

Efektiivsema strateegia jaoks arvestatakse ühe sammu kogupreemia R_t arvutamisel ka tulevaste sammude preemiaid, mida diskonteeritakse kaalu γ abil [20]:

$$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots = \sum_{\tau=t}^{\infty} \gamma^{\tau-t} r_t \tag{1.6}$$

Stiimulõppe eesmärgiks on õpetada agendile ülesande läbiviimist määramatus keskkonnas. Agent sisaldab kahte komponenti - strateegia ja õppimise algoritm [22]:

- Strateegia kaardistab tegevused keskkonnast saadud mõõtmistega. Üldjuhul on strateegiaks juhitavate parameetritega funktsiooni aproksimeerija, nagu näiteks tehisnärvivõrk;
- Õppimise algoritm uuendab pidevalt strateegia parameetreid, põhinedes tegevustele, vaatlustele ning preemiatele. Selle eesmärgiks on leida optimaalne strateegia, mis maksimeerib oodatud kumulatiivse pikaajalise preemia, mis ülesande käigus saavutatakse.

Olenevalt õppimisalgoritmile omab agent strateegia treenimiseks ühte või rohkemat funktsiooni aproksimeerijat, mida saab kasutada kahte moodi [22]:

- Kriitik (*critic*) annab konkreetse vaatluse ja tegevuse põhjal oodatud kumulatiivse pikaajalise preemia.
- Täitja (*actor*) annab konkreetse vaatluse põhjal väljundiks tegevuse, mis maksimeerib oodatud kumulatiivse pikaajalise preemia.

Tegevuse leidmiseks ainult kriitikut kasutavaid agente nimetatakse väärtuspõhisteks ning need kasutavad aproksimeerijat, et esindada väärtusfunktsiooni või Q-väärtusfunktsiooni (*Q-value function*). Üldiselt töötavad need agendid paremini diskreetsete tegevuse ruumidega ning pidevate tegevuste puhul osutuvad arvutuslikult kalliks [22].

Ainult täitjat kasutavaid agente nimetatakse strateegiapõhisteks. Üldjuhul on need agendid lihtsamad ning tulevad toime pideva väärtustega tegevustega, kuid algoritmi treenimine võib olla müra suhtes tundlik ning kohalikele miinimumidele koondumine on sage [22].

Tegevuse leidmiseks kriitikut ja täitjat kasutavaid agente nimetatakse täitja-kriitik (*actor-critic)* agentideks. Treenimise ajal õpib täitja parima tegevuse, kasutades selleks tagasisidet kriitikult (erinevalt otse preemia kasutamisest). Samaaegselt õpib kriitik preemia kaudu väärtusfunktsiooni, et see saaks täitjat adekvaatselt kritiseerida. Üldjuhul saavad sellised agendid hakkama nii diskreetsete kui ka pidevate tegevustega [22]. Peale agendi ja keskkonna saab stiimulõppe puhul välja tuua veel neli peamist alaelementi: strateegia, preemiasignaal (*reward signal*), väärtusfunktsioon ning keskkonna mudel [18].

Strateegia määrab agendi käitumise antud ajahetkel ehk kaardistab keskkonna olekud tegudega, mis vastavad neile olekutele. Psühholoogias nimetataks seda stiimul-reaktsioon reeglite koguks. Strateegia võib olla nii lihtne funktsioon või otsingutabel, kui ka suurt arvutusjõudlust nõudev protsess. Strateegia on stiimulõppe agendi tuum, kuna ainult sellest piisab, et määrata agendi käitumine. Üldiselt võivad strateegiad olla ka stohhastilised, mis juhul on väljundiks tegevuste tõenäosused [18].

Preemiasignaal defineerib stiimulõppe probleemi eesmärgi. Igal ajasammul saadab keskkond agendile arvu, mida kutsutakse preemiaks. Agendi ainueesmärk on maksimeerida pikas plaanis kogutud preemia koguarv. Seega määrab preemiasignaal selle, mis on agendi jaoks head ja halvad sündmused. Bioloogiliste süsteemide puhul saab analoogi tuua mõnu ja valu aistingutega. Preemiasignaal on põhiline strateegia muutmise mehhanism – kui valitud tegevus põhjustas preemia vähenemise, siis võib strateegia muutuda nii, et järgmine kord samas olukorras valitakse teistsugune tegevus. Preemiasignaalid võivad samuti olla stohhastilised funktsioonid, mis sõltuvad keskkonnast ja läbiviidud tegevusest [18].

Kui preemiasignaal määrab hea käitumise antud ajahetkel, siis väärtusfunktsioon määrab hea käitumise pikas plaanis. Mingi oleku väärtus on võrdne preemia koguhulgale, mida agent suudab tulevikus koguda, alustades sellest olekust. Seega väärtused näitavad olekute pikaajalist ihaldatavust, arvestades sellele järgnevaid olekuid ning nendega kaasnevat preemiat. Näiteks võib mingi olek alati genereerida madalat preemiat, kuid omada kõrget väärtust, kuna sellel olekule järgnevad kõrgema preemiatootlikkusega olekud. Seega peab tegevuste hindamisel arvestama just väärtusega, kuid kuna seda peab pidevalt jooksvate vaatluste põhjal tuletama, on see märkimisväärselt keerulisem preemia arvutamisest [18].

Viimane stiimulõppe süsteemide element on keskkonna mudel. See on matemaatiline süsteem, mis järgib keskkonna käitumist ehk lubab järeldusi tuua selle kohta, kuidas keskkond käitub. Teades antud olekut ja tegevust, suudab mudel ennustada järgmise oleku ning preemia. Mudeleid kasutatakse planeerimiseks. See tähendab, et tegevuste käik otsustatakse tulevaste olekute põhjal, ilma neid läbi kogemata. Selliseid süsteeme kutsutakse mudelipõhisteks , vastupidiselt lihtsamatele mudelivabadele süsteemidele, mis on rangelt katse-eksitus meetodil töötavad õppijad. Moodsad stiimulõppe süsteemid kasutavad mõlemat tüüpi meetodeid [18].

Stiimulõpe erineb juhendatud õppest, kuna ei kasutata eelnevalt teada näiteid õigest käitumisest. Interaktiivse keskkonna kohta on tihtipeale ebapraktiline koguda näiteid kõikvõimalikest situatsioonides, kus agendil on vaja käituda. Kaardistamata territooriumis, kus intuitiivselt on õppimisest enim kasu, peab agent suutma õppida iseenda kogemustest. Erinevus on ka juhendamata õppega, mille puhul otsitakse struktuure pealtnäha juhuslikust andmekogust. Esmapilgul võib tunduda, et ka stiimulõpe võib klassifitseeruda juhendamata õppeks, kuna ei ole ette antud õiget käitumist, kuid stiimulõpe ei püüa leida peidetud struktuure, vaid selle eesmärk on ainult maksimeerida preemia. Etteantud probleemi struktuuri leidmine võib kindlasti kasulikuks osutuda, kuid see ei lahenda iseenesest preemia maksimeerimise probleemi. Stiimulõpet peetakse seega juhendamata ning juhendatud õppe meetoditest eraldiseisvaks [18].

Formaalselt esitatakse stiimulõppe meetodeid kasutades ideid dünaamiliste süsteemide teooriast, eelkõige mittetäieliku informatsiooniga Markovi otsustusprotsesside optimaalse juhtimise teooriat.

1.5.1 MDP

Markovi otsustusprotsess (MDP) on matemaatiline raamistik otsuste modelleerimiseks, mis on sobilik mittetäieliku info korral. Tavaliselt väljendatakse MDP nelja omadusega [23]:

- S – keskkonna kõikvõimalike olekute hulk
- A – kõikvõimalike tegevuste hulk
- $p(.|s,a)$ – oleku ülemineku tõenäosus, $s_{t+1} \sim p(.|s_t, a_t)$
- $q(.|s,a)$ – preemia tõenäosusjaotus, $R(s_t, a_t) \sim q(.|s_t, a_t)$

S tähistab agendi poolt vaadeldavat maailma. Agendi õppimine jagatakse diskreetseteks ajaperioodideks ning peale igat ajaperioodi t liigub agent olekusse $s_t \in S$. Agent valib tegevuse $a_t \in A(s_t)$, kus $A(s_t)$ on kõikvõimalike tegevuste hulk keskkonna olekus s_t. Antud tegevuse läbiviimisele järgneb liikumine olekusse s_{t+1} ning arvväärtusega preemia $R(s_t, a_t)$. Oleku ülemineku tõenäosus $p(s_{t+1}|s_t, a_t)$ määrab tõenäosuse, millega liigutakse olekusse s_{t+1}, juhul kui viiakse läbi tegevus a_t olekus s_t. Preemia hulga määrab tõenäosusjaotus $q(s_{t+1}|s_t, a_t)$ ning osutab olekus s_t tegevuse a_t kasulikkusele [23].

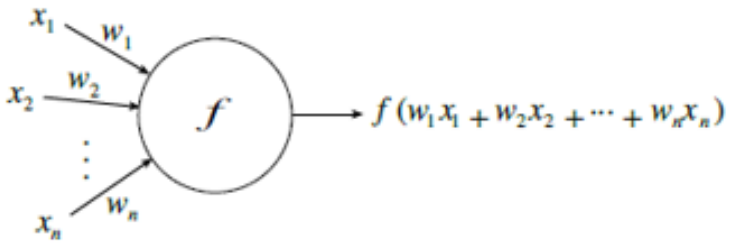
MDP lahendus annab väljundiks strateegia π, mis kaardistab omavahel tegevusi ja olekuid ning seega juhib agendi otsuseid läbi terve õppimisperioodi [23]. Eesmärgiks on leida strateegia, mis maksimeerib ajavahemikus 0...T oodatud preemia diskonteeritud kumulatiivse summa ehk $\sum_{t=0}^{T} \gamma^t R(s_t, a_t)$ [24].

Juhul kui on teada täielik keskkonna mudel ehk suurused (S, A, p, q) on täielikult leitavad, taandub MDP tavaliseks planeerimisprobleemiks, mida on võimalik lahendada traditsiooniliste dünaamilise programmeerimise meetoditega. Kuid juhul kui täielikku keskkonna mudelit ei eksisteeri, mis on päriselu probleemide puhul tüüpiline, siis on vaja püüda aproksimeerida puudulikku mudelit (mudelipõhine stiimulõpe) või otse väärtusfunktsiooni ja strateegiat (mudelivaba stiimulõpe). Mudelipõhised meetodid kasutavad statistilisi võtteid, et aproksimeerida puuduv mudel ning mudelivabad meetodid püüavad aproksimeerida juhtimisstrateegia otse [23].

1.6 Tehisnärvivõrgud

Tehisnärvivõrgud on õppimismasinad, mis koosnevad suurest hulgast kihtidena ühendatud sõlmedest ehk neuronitest. Õppimine saavutatakse ühenduste kaalude väärtuste seadistamisega [12].

Joonisel 1.8 on esitatud abstraktne neuron n sisendiga. Iga sisendi kanal i suudab edasi anda reaalse väärtuse x_i. Neuroni sees olev funktsioon f võib olla vabalt valitud. Üldjuhul on sisendi kanalitelega seotud kaal w_i ehk väärtus, millega sisendit korrutatakse. Neuronis pannakse kõik sisendid kokku (tavaliselt lihtsalt liidetakse kokku) ning arvutatakse funktsiooni väärtus. Sellistest neuronitest koosnev tehisnärvivõrk on seega vaid funktsioonide võrk ning närvivõrkude mudelid erinevad üksteisest peamiselt ühenduste skeemi, sisemiste funktsioonide ning informatsiooni edastuse ajastuse poolest [25].

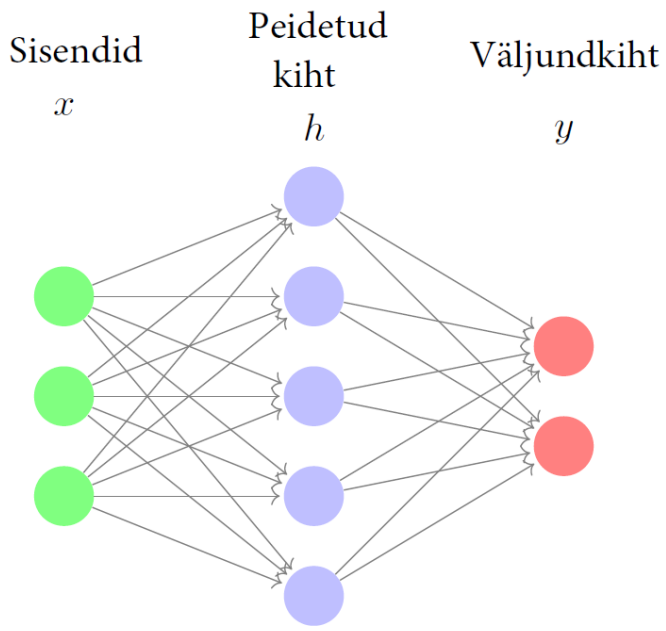


Joonis 1.8 Üksik neuron [25]

Loogiline neuron on matemaatiliselt lihtsasti väljendatav: see liidab kokku kaalutud sisendid eelmisest kihist, lahutab sellest tundlikkusläve θ ning viimaks edastab väärtuse aktivatsioonifunktsioonile A, et arvutada enda väljund a [25]:

$$a = A\left(\sum_i w_i x_i - \theta\right) \tag{1.7}$$

Kõige tuntum närvivõrgu mudel on kolmekihiline pärilevivõrk (joonis 1.9). Selline süsteem koosneb sisendkihist, ühest peidetud kihist (*hidden layer*) ja väljundkihist. Neuronid edastavad informatsiooni vaid pärisuunas ehk sisendkihis olev neuron on ühendatud vaid kõikide peidetud kihi neuronitega, mis tähendab, et informatsioon saab liikuda vaid sisendikihi poolt väljundkihi suunas [25].



Joonis 1.9. Lihtne närvivõrk [26]

Esimese ehk sisendkihi abil saadakse sisendandmed x (üldjuhul vektor pikkusega n_x ($n_x \in \mathbb{N}$)) närvivõrku. Sisendkihi neuronite arv on tavaliselt võrdne sisendina kasutatavate tunnuste arvuga. Klassikalistes pärilevivõrkudes on iga sisendkihi neuron ühendatud kõikide järgmises kihis olevate neuronitega, keerulisemate arhitektuuride puhul ei pea see nii olema. [27]. Järgmises kihis muundatakse sisendväärtused läbi mittelineaarse funktsiooni, mis kujutab endast maatrikskorrutist $n_h * n_x$ suuruse kaalude maatriksiga W_1, pluss suurusega n_h kaalude vektor b_1, millele järgneb mittelineaarne transformatsioon [26]:

$$h = A(W_1 * x + b_1) \tag{1.8}$$

kus A on mittelineaarne aktivatsioonifunktsioon.

Aktivatsioonifunktsioonina kasutatakse tihtipeale sigmoidfunktsiooni [17]:

$$A(x) = \frac{1}{1 + e^{-x}} \tag{1.9}$$

Mittelineaarne aktivatsioonifunktsioon annab võrgule võimekuse lahendada mittelineaarseid probleeme. Järgmisena sooritatakse maatrikskorrutis $n_y * n_h$ suuruse kaalude maatriksiga W_2 ning lisatud on ka vektor b_2, mis sisaldab peidetud ning väljundkihi vaheliste ühenduste kaalusid [26].

$$y = (W_2 * h + b_2) \tag{1.10}$$

Peidetud kihis olevad neuronid ning nende ühendustega seotud kaalud on see, mis andmehulgast informatsiooni eraldab. Peidetud kihid on vajalikud mittelineaarsete funktsioonide modelleerimiseks. Ühendades sisendkihi läbi kaalutud ühenduste otse väljundkihiga, oleks võimalik modelleerida vaid lineaarseid funktsioone [27].

Väljundkihist saab kätte sisendile vastava väljundi. Olenevalt närvivõrgu funktsioonist on see kas reaalarvuline väärtus (regressioon) või tõenäosuste hulk (klassifikatsioon). Seda kontrollib väljundkihi neuronite aktivatsioonifunktsioon [27].

Kogu võrku treenitakse eesmärgiga minimeerida empiirilist viga $I_S[f]$. Kõige populaarsem meetod närvivõrgu parameetrite optimeerimiseks põhineb gradient laskumisel läbi tagasilevi algoritmi. Lihtsamaimal juhul muudab algoritm igal iteratsioonil oma sisemisi parameetreid θ nii, et sobituda järgmise funktsiooniga [26]:

$$\theta \leftarrow \theta - \alpha \nabla_\theta IS[f] \tag{1.11}$$

kus α on õppimise samm ehk kiirus [26].

1.7 Sügavõpe, Q-õppimine, DDPG ja mudeli treenimine

Sügavõpe on masinõppe alavorm, kus õppimisel asetatakse rõhk kihthaaval andmetest aina tähendusrikkama andmete esituse leidmisele. Andmehulga mudeli kihtide arvu nimetatakse seega mudeli sügavuseks. Moodsad sügavõppe mudelid hõlmavad tihtipeale kümneid või sadu kihte, mis on kõik treeningandmete põhjal automaatselt loodud. Andmete esitusi modelleeritakse peaaegu alati läbi mitmekihiliste närvivõrkude [28].

Sügavad närvivõrgud erinevad tavalistest mitmekihilistest pärilevivõrkudest järgmiste omaduste poolest [27]:

- rohkem neuroneid
- keerulisemad kihtide vahelised ühendused
- märkimisväärselt suurem nõudlus arvutusjõudlusele
- automaatne tunnuste eraldamine (*feature extraction*)

Neuronite arv on ajas suurenenud keerulisemate mudelite tõttu. Lihtsate kihtide vaheliste ühenduste asemel kasutatakse lokaalseid saari, mis on ümbritsevate neuronitega ühendatud. Rohkem ühendusi tähendab, et võrgul on rohkem parameetreid, mida peab optimeerima. Seega suureneb ka vajadus arvutusjõudluse järele, mistõttu on sügavõpe alles viimase 20 aasta jooksul kiiresti populaarsust kogunud. Võimsamate arvutite ning keerukamate võrkude tõttu on moodsad

närvivõrgud võimelised tunnuseid eraldama veelgi intelligentsemalt ning see on võimaldanud näiteks suured edusammud pildituvastuses [27].

Sügav stiimulõpe on stiimulõppe vorm, kus närvivõrku kasutatakse universaalse funktsiooni aproksimeerijana. Selle meetodi miinusena kaovad võrgult piirded ning lahenduse koondumine ei ole garanteeritud. Kuid siiski selline närvivõrkude kasutamine on paljulubav ning pakub häid tulemusi [27].

1.7.1 Q-õppimine

Q-õppimine (*Q-learning*) on mudelivaba väärtuspõhinev stiimulõppe vorm, mille tööpõhimõte seisneb järkjärgulises tegevuste väärtuste hindamises. Selleks uuendatakse Q-funktsiooni, mis püüab hinnata kogu preemiat, mis antud tegevuse sooritamisega on võimalik saavutada. Iga võimaliku oleku jaoks määratakse igale võimalikule tegevusele väärtus, mis on funktsioon kohesest preemiast ning kogu tulevikus saadavast preemiast, mis saavutati tänu antud tegevusele. Q-õppimise puhul õpib agent keskkonnast läbi episoodide, ilma eelnevalt keskkonna kohta informatsiooni omamata. Agent omab tabelit $Q[S,A]$, kus S on kõikide olekute ning A kõikide tegevuste hulk. Algoritmi kirjeldab Q-uuenduse võrrandi üks samm: [29] [30]

$$Q(s,a) = Q(s,a) + \alpha[r + \gamma max a' Q(s',a') - Q(s,a)] \tag{1.12}$$

kus Q on oodatud väärtus tegevuse a sooritamisel olekus s,
s on olekute vektor,
a on tegevuste vektor,
r on preemia,
α on õppimise samm, mis kontrollib koondumist ja selle kiirust,
ning γ on diskontomäär.

Diskontomäär γ muudab varem saadud preemia väärtuslikumaks kui tulevikus saadud preemia. Meetod õpib optimaalse strateegia asemel kõikide tegevuste väärtused. See on küll arvutuslikult kallis, kuid toob kaasa ka eeliseid, kuna igal ajal saab sooritada iga tegevuse ning sellest on võimalik õppida [29].

Q-õppimist saab algoritmina esitada järgmiselt [29]:

- Võtta juhuslik $Q(s,a)$ algväärtus
- Kordus iga episoodi jaoks
    - Võtta s algväärtus (missuguses olekus keskkond on alustades).
    - Kordus iga ajasammu jaoks:
        - Valida a väärtus s ning Q järgi.

        - Viia läbi tegevus a, arvutada r ja s'.
        - $Q(s,a) = Q(s,a) + \alpha[r + \gamma max a' Q(s',a') - Q(s,a)]$.
        - $s \leftarrow s'$.
        - Kuni s jõuab lõppväärtuseni.
- Kuni on saavutatud piisav täpsus või maksimaalne episoodide arv.

Kui algoritmi väljundiks on lõplik hulk diskreetseid tegevusi, siis ei ole maksimumi arvutamine keeruline, kuna saab eraldi arvutada lihtsalt Q-väärtused iga tegevuse kohta ning neid omavahel võrrelda (nii saadakse automaatselt ka see tegevus, mis maksimeerib Q-väärtuse). Kuid kui tegevuste väärtuste ruum on pidev (sobivaid tegevuse väärtuseid on praktiliselt lõpmata palju), siis ei ole võimalik kogu ruumi täielikult läbi arvutada ning optimeerimisprobleemi lahendamine on keeruline [31].

1.7.2 DDPG

DDPG (*Deep Deterministic Policy Gradient*) on algoritm, mis õpib samaaegselt Q-funktsiooni ning strateegia. See lähenemine on väga sarnane Q-õppimisele ning on sarnaselt motiveeritud: kui optimaalne tegevuse väärtusfunktsioon $Q^*(s_t, a_t)$ on teada, siis on võimalik leida optimaalne tegevus $a^*(s)$ järgmise valemiga [31]:

$$a^*(s) = arg\ \max_a Q^*(s,a) \tag{1.13}$$

DDPG on täitja-kriitik tüüpi stiimulõppe algoritm, mis tuleb toime pidevate väärtustega tegevustega, kuna tegevus arvutatakse sügava tehisnärvivõrgu abil. DDPG koosneb kahest põhilisest võrgust – kriitiku ja täitja võrk. Nii kriitik kui ka täitja koosnevad omakorda kahest alavõrgust – *online* ja *target*. Kõik võrgud koosnevad kihtidest ning iga kiht sisaldab vastavaid parameetreid. Konkreetse võrgu kõiki parameetreid tähistatakse sümboliga θ. Kriitiku võrk on treenitud simuleerima reaalset Q-tabelit, kasutades selleks närvivõrku, mis elimineerib dimensionaalsuse probleemi. Täitja võrk on treenitud deterministliku strateegia genereerimiseks, mille väljundiks on konkreetne tegevus [32] [33].

Lisaks luuakse ka *target* võrk, mis uuendab aeglaselt täitja ja kriitik võrke, et suurendada treeningu stabiilsust. Tegevuste avastamise protsessi aitamiseks on loodud avastamisstrateegia μ', mis lisab algoritmile juhuslikku, ajas vähenevat müra ξ järgmise valemiga [33]:

$$\mu'(s_t) = \mu\left(s_t|\theta_t^{\mu}\right) + \xi_t \quad (1.14)$$

kus $\xi_{t+1} = r_d * \xi_t$, milles r_d on müra vähenemise kiirus
$\mu(s_J|\theta_j^{\mu})$ on täitja funktsioon, milles θ_t^{μ} on strateegia võrgu parameetrid.

1.7.3 Mudeli treenimine

Närvivõrgu mõju sisendandmetele on talletatud kihte ja neuroneid ühendavates kaaludes, mis on kokkuvõttes üks hulk arve. Ehk võib öelda, et kaalud on kihi poolt läbiviidava muundamise parameetrid. Seega närvivõrgu raames tähendab treenimine kaaludele selliste väärtuste leidmist, mis väljenduvad sisendi ja väljundi õige seosega [28]. Enamus andmehulkades on teatud omadused tugevamas korrelatsioonis teatud väljunditega (näiteks väline temperatuur ja elektrienergia tarbimine ajahetkel). Närvivõrgud õpivad neid korrelatsioone läbi juhusliku proovimise ning arvutavad seejärel tulemuse täpsuse. Edukalt treenitud tehisnärvivõrk koosneb kaaludest, mis võimendavad signaali ning summutavad müra. Suurem kaalu väärtus osutab signaali ja väljundi suuremale korrelatsioonile. Sisendid, mis on ühendatud läbi suurte kaalude, mõjutavad võrgu andmete tõlgendamist rohkem, kui väiksemate kaaludega ühendatud sisendid [27].

Tehisliku närvivõrgu treenimisega alustades määratakse võrgu kaaludeks juhuslikud väärtused, mistõttu on võrk vaid juhuslike arvutuste jada ning väljundi täpsus on madal. Kuid iga iteratsiooniga kohandatakse kaalude väärtuseid õiges suunas ning väljundi täpsus hakkab tõusma. See on treeningtsükkel, mis peale piisavat hulka iteratsioone genereerib kaalude väärtused, mis minimeerivad väljundi viga. Minimeeritud veaga võrk on selline, mille väljundid on võimalikult lähedal eesmärgile ning seda saab kutsuda treenitud võrguks [28]. Neid parameetreid, mis on seotud mudeli üldise ehitusega, nimetatakse hüperparameetriteks ning neid muutes saab parandada mudeli treenimise kiirust ning tulemuse kvaliteeti. Hüperparameetrite valik keskendub peamiselt ala- ja ülesobitamise vältimisele ning treenimise kiiruse maksimeerimisele [27].

Väljundi kontrollimiseks on vaja mõõta, kui kaugel see on oodatud väärtusest. Selleks kasutatakse kaofunktsiooni, mis võrdleb arvutatud ning tõelisi väljundeid ning leiab nende vahelise erinevuse ehk kauguse skoori (*distance score*). Treenimise olemus seisneb selle skoori kasutamises tagasiside signaalina, et nihutada närvivõrgu parameetreid selliselt, et minimeerida kaotuse skoori (*loss score*). Parameetrite väärtuste nihutamise eest vastutab optimeerija (optimeerijaid on erinevaid ning valik sõltub konkreetsest ülesandest), mis tugineb tagasilevi (*backpropagation*) algoritmile. See algoritm on sügavõppes kesksel kohal [28]. Mudeli kaofunktsioon premeerib võrku

heade ning karistab halbade pakkumiste eest. Seega kaofunktsioon nihutab võrgu parameetreid paremate otsuste ja ennustuste moodustamise suunas. Närvivõrgu parameetreid ei ole üldjuhul võimalik analüütiliselt leida, mistõttu kasutatakse kaofunktsioone, mis suudavad neid parameetreid iteratiivsel teel aproksimeerida [27].

Aktivatsioonifunktsiooni eesmärk on edastada eelmise kihi väljund järgmise kihini. Need funktsioonid võimaldavad võrgul modelleerida mittelineaarseid funktsioone. Kõige lihtsam variant on lineaarne aktivatsioonifunktsioon, mis praktikas tähendab seda, et signaal läbib neuroni muutumata kujul. Lineaarset funktsiooni kasutatakse närvivõrkude sisendkihis. Mittelineaarse aktivatsioonifunktsioonina kasutatakse tihti sigmoidfunktsiooni, mis suudab ükskõik kui laias vahemikus olevad väärtused muundada 0 ja 1 vahele, mis on tihtipeale lahenduse koondumisel vajalik. Sarnaselt kasutatakse ka hüperboolset tangensfunktsiooni ehk $tanh$, mis muundab sisendid vahemikku -1 kuni 1. Selle eeliseks on parem võimekus kasutada negatiivseid arve [27].

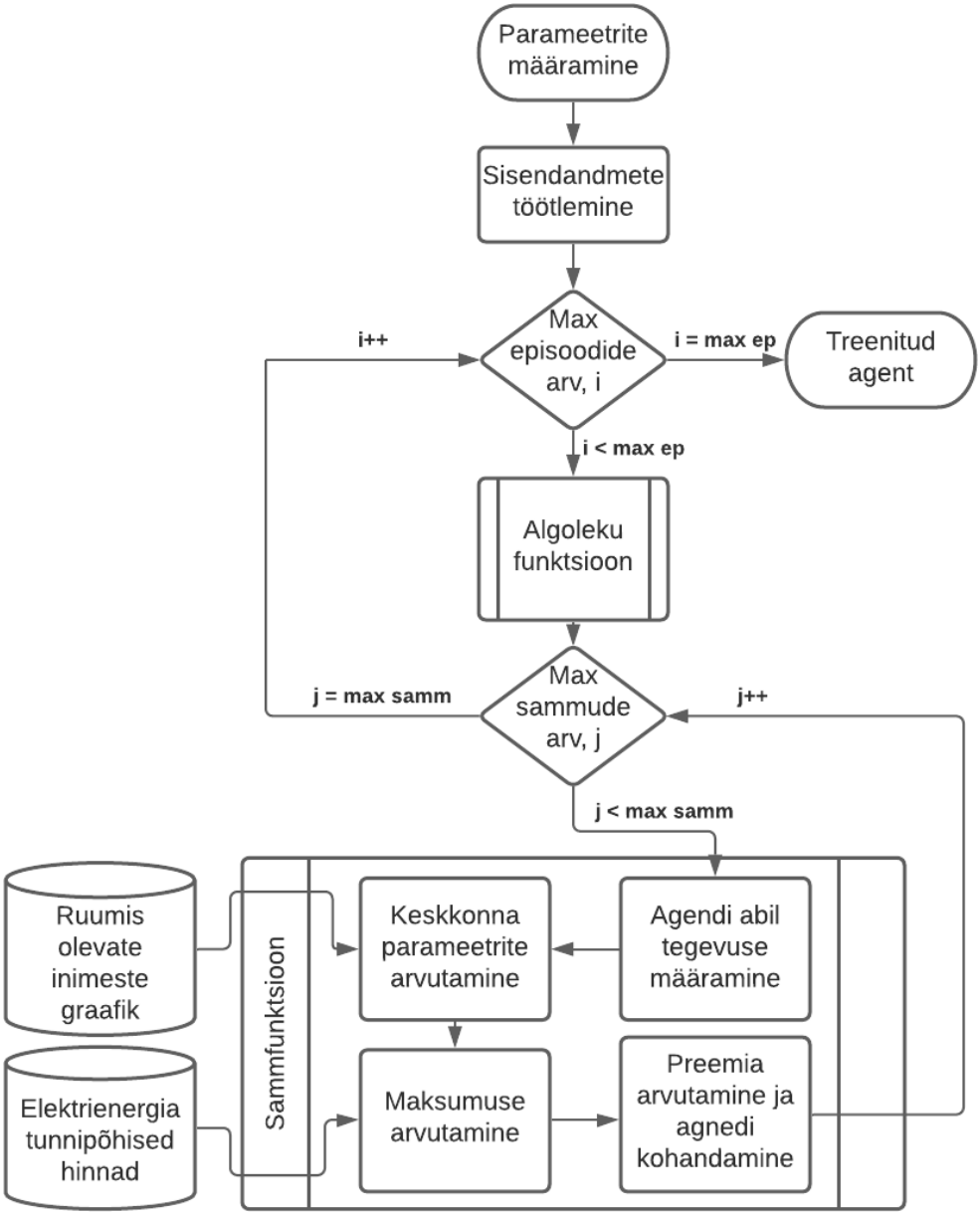
Õppimise samm mõjutab optimeerimise ajal võrgu parameetrite muutmise kiirust. See sõltub kasutatavast kaofunktsioonist ning antud iteratsiooni vea suurusest. Treenimise alguses on vead suured ning suurem õppimise sammu koefitsient aitab astuda suuremaid samme ning kiirendada treenimise protsessi. Kuid kui võrk on optimumpunktile lähedal, siis põhjustab liiga suur õppimise kiirus üleulatust ning võrk ei koondugi optimumile, vaid liigub selle ümbruses edasi-tagasi. Vastupidiselt madal õppimise kiirus juhib võrku väikeste sammudega optimumi poole, kuid selleks kulub palju aega [27]. Kuna suurte võrkude treenimiseks võib kuluda nädalaid, siis on treenimise kiirus kriitiline parameeter.

2. MODELLEERIMINE

Ventilatsioonisüsteemi modelleerimiseks kasutati Matlab tarkvara. Seal loodi keskkond ning vastav stiimulõppe agent, mille eesmärgiks on juhtida ventilatsiooniseadet selliselt, et saavutada võimalikult madal CO_2 kontsentratsioon ning energiakulud. Kuna need on vastuolulised eesmärgid, mis mõlemad sõltuvad juhuslike väärtustega sisenditest, siis on optimaalse juhtimise leidmine mittetriviaalne ülesanne.

Markovi otsustuspuul põhineva lahenduse kasutamine võimaldab efektiivselt toime tulla keskkonnast tuleneva juhuslikkusega. Ning kuna juhtimissüsteem on loodud tegutsema reaalses maailmas, siis ei ole täpse keskkonna mudeli loomine praktiline. Seega ei ole võimalik ka optimaalset strateegiat analüütiliselt leida ning vajalik on kasutada mudelivaba meetodit, nagu näiteks Q-õppimine [23]. Ning kuna antud süsteemi puhul on mõistlik kasutada pidevaid väärtuseid, siis osutub DDPG meetod sobivaks.

Matlab keskkonnas loodud programm arvutab iteratiivselt sammfunktsiooni, mis määrab vastavalt ventilaatori kiirusele ajasammu CO_2 taseme. Treenimise ajal otsustab stiimulõppe agent ventilaatori kiiruse ning saab selle eest preemiat, mille järgi agenti parandatakse, kuni saavutatakse piisavalt efektiivne mudel. Programmi käiku kirjeldav üldine plokkskeem on esitatud joonisel 2.1. Treenimisele kuluva aja määravad peamiselt maksimaalne sammude ning episoodide arv. Neist esimene defineerib ühe episoodi pikkuse ning on seotud ajasammu suurusega t_s. Antud töös kasutati viie minuti pikkust ajasammu, mis tähendab, et ühe sammfunktsiooni iteratsiooniga edendatakse simulatsiooni viie minuti võrra. Tunnipõhine elektrienergia hind muutub seega iga kaheteistkümne sammu tagant. Maksimaalne sammude arv määrab treenimisel simuleeritava ajavahemiku pikkuse. Võttes selleks näiteks 288, treenitakse agenti ühe ööpäeva andmetega. Episoodide maksimaalne arv piirab treenimise kestvust. Antud töö kontekstis ei olnud selleks vajadust ning treenimine katkestati manuaalselt. Ruumis olevate inimeste graafik ning elektrienergia tunnipõhised hinnad on eelnevalt määratud ning algoritm teisendab need ajasammu põhiseks. Algoleku funktsioon loob vajalikud muutujad ning annab neile algse väärtuse, et sammfunktsioon saaks neid esimese sammu arvutamisel kasutada.



Joonis 2.1 Stiimulõppe mudeli plokkskeem

2.1 Sisendandmete ettevalmistamine

Stiimulõppe algoritm vajab tegevuse otsustamiseks kahte sisendit – elektrienergia hind ning siseõhu mõõdetud CO_2 kontsentratsioon. Lisaks antud ajahetke hinnale antakse algoritmile ka järgnevate ajahetkede hinnad, et mudelil oleks võimalik otsuste tegemisel mingil määral tuleviku hindadest lähtuda. Käesolevas töös kasutatakse närvivõrgu sisenditena hindu E'_t, E'_{t+1}, E'_{t+3}, E'_{t+8}, ja E'_{t+12}. Kaugemale tulevikku vaatamine tõstab mudeli efektiivsust, kuid iga lisanduv sisend suurendab treenimise keerukust ning sellele kuluvat aega. Piiramatu arvutusjõudluse puhul oleks parim variant mudelile ette anda kõik teadaolevad tuleviku hinnad (ning isegi kalkuleeritud hinnangud mitte teadaolevate

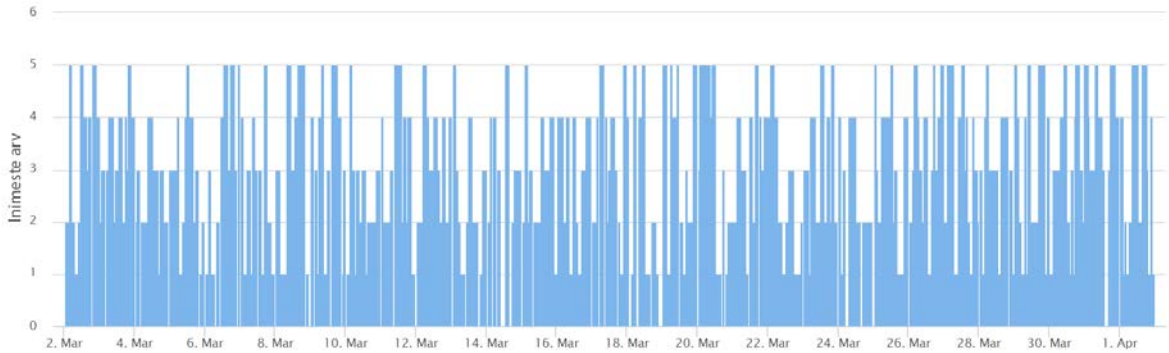
hindade kohta), kuid antud mudeli loomisel leiti, et just need viis sisendit annavad parima tulemuse.

Tulenevalt närvivõrgu ülesehitusest töötab see efektiivsemalt sel juhul, kui arvutustes kasutatakse normaliseeritud väärtuseid. Üldjuhul kasutatakse suuruseid vahemikus -1 ... 1, kuid kuna siseruumi ventilatsioonisüsteemi kontekstis ei oma negatiivsed väärtused praktilist tähendust, siis kasutatakse antud töös väärtuseid 0 ... 1.

Elektrienergia hindadena kasutatakse Nord Pooli tunnipõhiseid andmeid [34]. Võrguning muude säärased tasudega jäetakse antud töö raames arvestamata, kuna need on üldjuhul konstantse väärtusega ning ei mõjuta juhtimisstrateegiat. Kuna hinna absoluutväärtus ei oma stiimulõppe agendi jaoks tähtsust, siis lahutatakse antud ajahetke väärtusest selle päeva minimaalse tunni hind ning jagatakse päeva kõige kallima hinnaga, mille läbi saadakse normaliseeritud hind vahemikus 0 – 1. Sarnaselt käitutakse ka CO_2 kontsentratsiooni arvutamisel, kuid sel juhul määratakse minimaalne ning maksimaalne kontsentratsioon manuaalselt. Antud töös on minimaalseks väärtuseks võetud 400 ppm, mis vastab ligikaudu välisõhu tavalisele CO_2 tasemele. Maksimaalseks väärtuseks on võetud 1000 ppm [8].

2.1.1 Ruumis olevate inimeste modelleerimine

Mudeli treenimisel on vaja modelleerida ka CO_2 genereerimist, mis tuleneb ruumis olevate inimeste arvust. Mida sarnasem on see mudel reaalsele elule, seda paremini algoritm toime tuleb. Seega on tähtis ruumis olevate inimeste hulka täpselt modelleerida ning antud töös kasutati selleks *Occupancy Simulator* aplikatsiooni [35]. Seal loodi ruum, mida kasutab ööpäeva jooksul erinevate ajavahemikkudega 0 kuni 5 inimese suurune grupp. Seega sobivad andmed näiteks väikese koosoleku- või klassiruumi simuleerimiseks. Andmed loodi ühe aastase vahemiku jaoks ning graafik tüüpilisest nädalast on esitatud joonisel 2.2.



Joonis 2.2 Ruumis olevate inimeste arv [35]

Ruumi kasutamist modelleerimisel ei arvestatud tüüpilise tööpäeva pikkusega, vaid hoiti ruumi kasutamise tõenäosus sama igal kellaajal. Reaalses ruumi puhul on tüüpiline, et seda kasutatakse vaid teatud ajavahemikus ning näiteks öösel ei viibi ruumis kedagi. Antud lähenemine muudaks küll andmed sarnasemaks reaalse kontoriruumi graafikuga, kuid pidurdaks asjatult treenimise kiirust, kuna ilma CO_2 allikateta ei ole ventilaatori juhtimisel mõju. Seega kuluks suur osa treenimise ajast selliste andmete töötluseks, mis ei mõjuta juhtimisstrateegiat. Samadel põhjustel pikendaks see ka testimiseks vajava aja pikkust.

Kuna stiimulõppe agent kasutab sisendina siseõhu CO_2 kontsentratsiooni, siis on selle simuleerimiseks vajalik ruumis olevate inimeste hulk läbi korrutada ühe inimese poolt toodetud CO_2 koguse keskmise väärtusega C_{gen}. Antud töös on selleks võetud 20 L/h [10].

2.1.2 Keskkonna modelleerimine

Stiimulõppe algoritmi treenimiseks loodi keskkonna mudel, mille ülesanne on võimalikult täpselt modelleerida ventileeritavat ruumi ning seal kasutatavat ventilatsiooniseadet. Kuna antud töö käigus loodud algoritmi eesmärk on töötada konkreetses demoruumis, siis kasutati treenimisel selle parameetreid. Need on toodud välja tabelis 2.1.

Tabel 2.1 Parameetrite tähised ning väärtused

Tähis	Väärtus	Ühik	Seletus
t_s	5	min	ajasammu suurus treenimisel
C_{max}	1000	ppm	maksimaalne lubatud CO_2 kontsentratsioon
C_G	800	ppm	soovitud CO_2 kontsentratsioon
C_O	400	ppm	Välisõhu CO_2 kontsentratsioon
V	193	m^3	ruumis oleva õhu hulk
ΔV_{max}	230	m^3/h	ventileeritava õhu hulk maksimaalsel võimsusel
P_{max}	172	W	ventilaatori maksimaalne võimsus
P_{min}	83	W	ventilaatori minimaalne võimsus
C_{gen}	20	L/h	ühe inimese poolt toodetav CO_2 kogus

Närvivõrgu arvutustes osalevad ka keskkonna parameetrid, mistõttu on ka nende normaliseerimine vajalik. Valemitega 2.1 kuni 2.4 on leitud normaliseeritud ning ajasammu suurusega arvestav CO_2 suhteline tõus ruumis inimese kohta C'_{gen}, ventileeritava õhu suhteline hulk $\Delta V'_{max}$, soovitud CO_2 kontsentratsiooni suhteline väärtus C'_G ning välise õhu CO_2 kontsentratsiooni suhteline väärtus C'_O.

$$\Delta C'_{gen} = \frac{10^6 \cdot \Delta C}{V * \frac{60}{t_s} \cdot C_{max} \cdot 10^3} = \frac{10^6 \cdot 20}{193 \cdot \frac{60}{5} \cdot 600 \cdot 10^3} = 1{,}44 \cdot 10^{-2} \quad (2.1)$$

$$\Delta V'_{max} = \frac{\Delta V_{max}}{\frac{60}{t_s} \cdot V} = \frac{230}{\frac{60}{5} \cdot 193} = 9{,}93 \cdot 10^{-2} \quad (2.2)$$

$$C'_G = \frac{C_G}{C_{max}} = \frac{800}{1000} = 0{,}80 \quad (2.3)$$

$$C'_O = \frac{C_O}{C_{max}} = \frac{400}{1000} = 0{,}40 \quad (2.4)$$

2.2 Sammfunktsioon

Sammfunktsioon määrab süsteemi oleku antud ajasammul. See hõlmab endas keskkonda modelleerivaid valemeid (nagu näiteks CO_2 kontsentratsiooni ja seadme poolt tarbitud elektrienergia arvutamise kohta) ning ka preemia arvutamist stiimulõppe agendi jaoks. Seega on sammfunktsooni iteratiivne läbi arvutamine stiimulõppe algoritmi aluseks.

Igal ajasammul arvutatakse ruumi siseõhu normaliseeritud CO_2 kontsentratsioon C'_t valemi 2.5 järgi. Valem põhineb sarnastes töödes kasutuga [36], kuid on kohandatud stiimulõppe jaoks kasutamiseks.

$$C'_t = (1 - P'_t \cdot \Delta V'_{max}) \cdot \left(C_{t-1} + I_t \cdot C'_{gen}\right) + P'_t \cdot \Delta V'_{max} \cdot C'_O \quad (2.5)$$

kus P'_t on ventilaatori suhteline võimsus ajasammul t,
I_t on ruumis olevate inimeste hulk ajasammul t.

Ventilatsiooniseadme normaliseeritud võimsuse põhjal arvutatakse valemi 2.6 järgi normaliseeritud energiakulu S'_t, kus väärtus 1 on energiakulu sel juhul, kui ventilatsiooniseade töötab ajasammu vältel maksimaalsel võimsusel. Energiakulu on ventilaatori kiirusega kuupvõrdelises suhtes, kuna see tuleneb enamasti ventilaatori labade õhutakistusest [37].

$$S'_t = \frac{(P_{max} - P_{min}) \cdot P'^3_t + P_{min}}{P_{max}} \quad (2.6)$$

Ajasammule vastava reaalse elektrienergia maksumuse E_t (€) saab arvutada valemiga 2.7:

$$E_t = \frac{S_t' \cdot E_t' \cdot E_{tmax}}{\frac{60}{t_s} \cdot 10^6} \tag{2.7}$$

kus E_t' on elektrienergia suhteline maksumus ajasammul t,

E_{tmax} on ajasammule t vastava päeva maksimaalne elektrienergia hind (€/MWh).

Kuna stiimulõppe algoritmi poolt valitud tegevus mõjutas keskkonda, siis tuleb seda mõju hinnata ehk arvutada preemia. Preemia arvutamise viise on erinevaid ning analüütilised meetodid optimaalse valemi leidmiseks üldjuhul puuduvad [18], mistõttu on kõige levinum kasutada katse-eksitus meetodit, lähtudes sealjuures teatud reeglitest. Antud töös ka nii tehti ning ajasammu preemia r_t arvutatakse kahest sigmoidfunktsioonist koosneva valemiga:

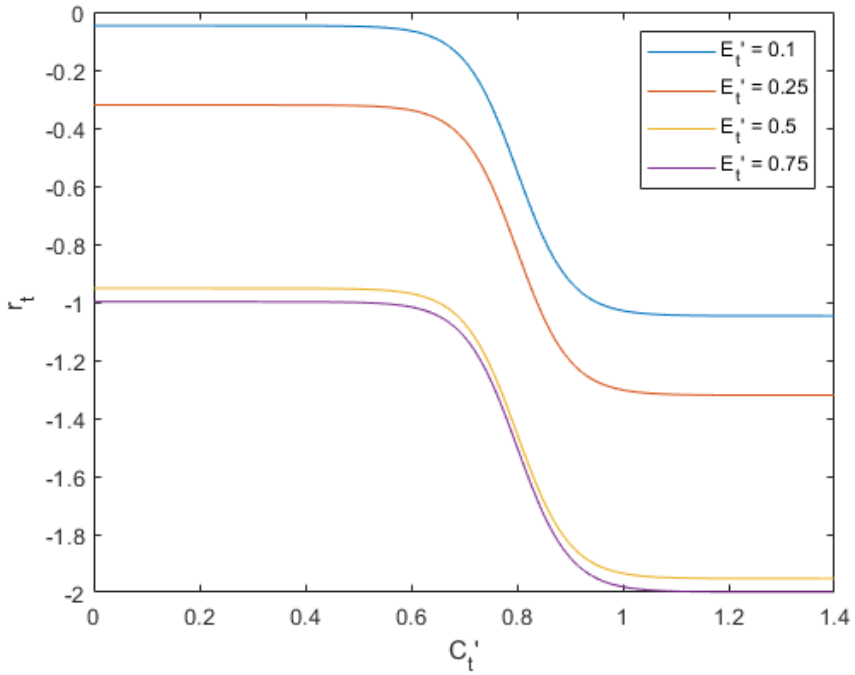
$$r_t(C_t', E_t') = -W_C \frac{1}{1 + e^{\alpha(C_G' - C_t')}} - W_E \frac{1}{1 + e^{\beta(0{,}3 - E_t')}} \tag{2.8}$$

Kus W_C on CO_2 kontsentratsiooni kaal preemia arvutamisel,

W_E on elektrienergia maksumuse kaal preemia arvutamisel,

α ja β määravad funktsiooni muutumise kiiruse soovitud väärtuste ümber.

Joonisel 2.3 on esitatud r_t graafik erinevate elektrienergia maksumuste E_t' ning väärtuste $\alpha = 20$, $\beta = 15$ ja $W_C = W_E = 1$ korral, mis illustreerib preemia sõltuvust CO_2 tasemest.



Joonis 2.3 Preemia sõltuvus CO_2 suhtelisest tasemest

Et hoida ära CO_2 tõusu üle maksimaalse lubatud taseme ($C_t' > 1$), siis kasutatakse selles olukorras teist preemia valemit, mis muudab agendi jaoks maksimumväärtuse ületamise veelgi kulukamaks:

$$r_t(C_t', E_t') = -C_t'^4 - W_E \frac{1}{1 + e^{\beta(0,3 - E_t')}} \tag{2.9}$$

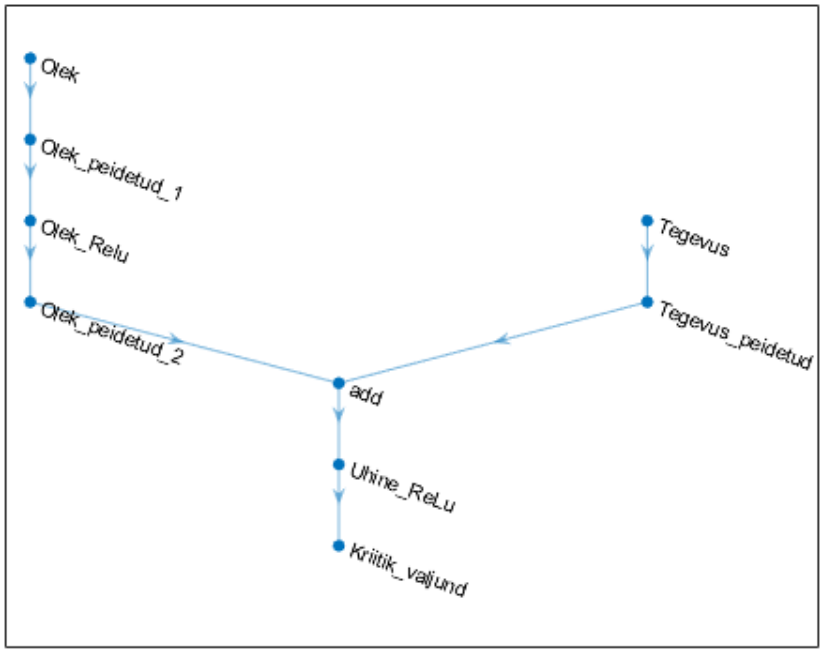
Antud valemid annavad vastuseks alati negatiivse väärtuse ning mida ihaldatum on tulemus (madal CO_2 kontsentratsioon ning elektrienergia maksumus) seda väiksem on preemia absoluutväärtuselt. Seega püüab stiimulõppe algoritm hoida treenimise ajal kogupreemiat võimalikult nullile lähedal. Negatiivne preemia taoliste süsteemide puhul tihtipeale efektiivne [38].

2.3 Stiimulõppe algoritmi modelleerimine

Stiimulõppe algoritmi lõppeesmärgiks on luua intelligentne agent, mis mõõdetud CO_2 kontsentratsiooni ning elektrienergia hinna järgi annaks väljundina antud olukorras optimaalse ventilatsiooniseadme võimsuse. Agendiks on siinjuhul tehislik närvivõrk ning selle treenimiseks kasutati DDPG algoritmi.

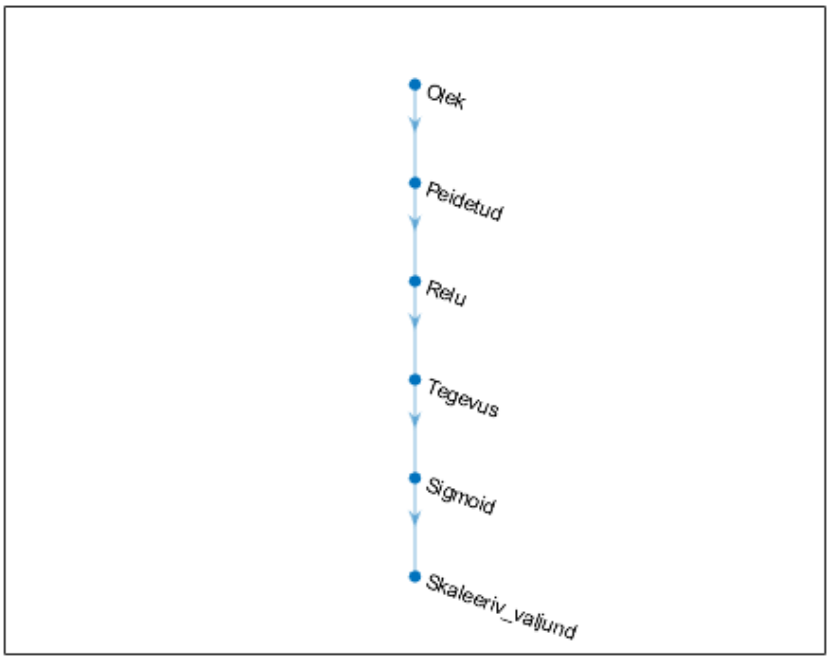
Sarnaselt preemia arvutamisele, puuduvad ka tehisliku närvivõrgu hüperparameetrite määramisel konkreetsed reeglid ning tihtipeale rakendatakse katse-eksitus meetodit [39]. Siiski on teada üldisemad reeglid, mis näitavad kätte suuna. Neid teadmisi järgides on kokku pandud DDPG algoritmile vajalikud närvivõrgud.

Esmalt on loodud kriitiku närvivõrk, millel on kaks sisendit (olek ja tegevus) ning üks väljund. Närvivõrk on jagatud kaheks osaks ehk teeks, kus mõlemad sisendid järgivad oma teed kuni saavad kokku ühisel teel ning moodustavad koos väljundi (joonis 2.4). Närvivõrgus olev ReLU (*Rectified Linear Unit*) kiht muudab iga nullist väiksema väärtuse võrdseks nulliga ning add (*addition*) kiht liidab element haaval kokku erinevad närvivõrgu kihid. Kriitiku oleku sisendkiht koosneb vastavalt sisendandmete arvule kuuest neuronist. Oleku esimene peidetud kiht sisaldab 50 ning teine 25 neuronit. Suurem neuronite arv garanteerib, et närvivõrk on võimeline saavutama optimaalset lahendust, kuid suurendab treenimisele kuluvat aega. Kriitiku tegevuse sisendkiht koosneb vastavalt väljundite arvule vaid ühe neuronit ning sellele järgnev peidetud kiht 25 neuronit. Ning kriitiku väljund koosneb samuti ühest neuronist. Täitja võrgu sisend koosneb sarnaselt kuuest neuronist, kuid selle peidetud kiht vajab vaid kolme neuronit.



Joonis 2.4 Kriitiku närvivõrgu skeem

Teisena loodi täitja närvivõrk (joonis 2.5), mille sisendiks on olek ning väljundiks ventilaatori normaliseeritud võimsus. Antud närvivõrk sisaldab enne väljundit ka sigmoid ning skaleerivat kihti. Neist esimene muudab väljundi väärtuse vahemikku 0 kuni 1. Teine korrutab selle suurusega $1 - \mathrm{MinP}$ ning nihutab MinP võrra üles, et saavutada ventilaatorile vastav minimaalne ja maksimaalne võimsus.



Joonis 2.5 Täitja närvivõrgu skeem

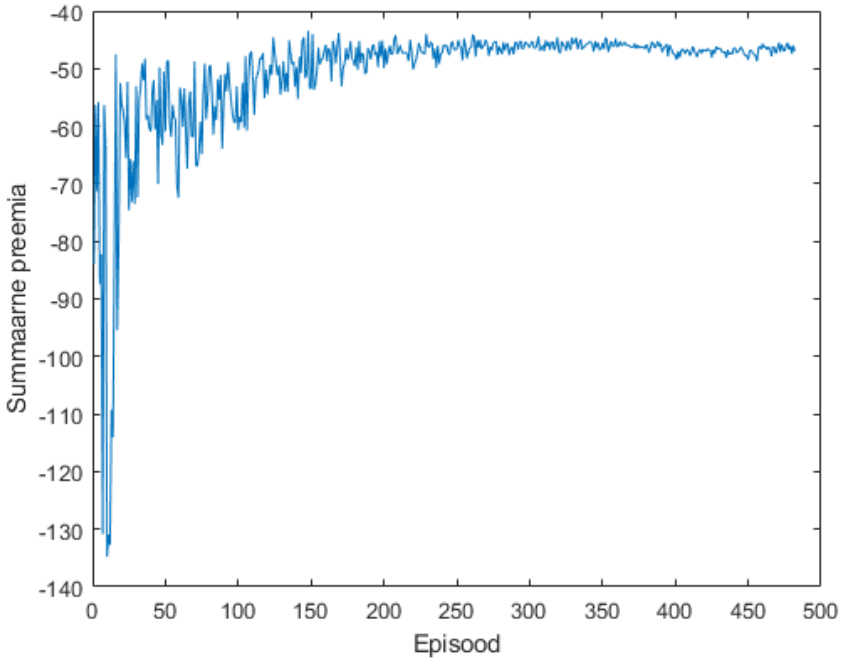
Edukalt treenitud mudeli saavutamiseks oli vaja määrata ka paljud teised parameetrid, mille täpsed väärtused on leitavad lisas 1 olevas Matlabi koodis.

Treenitud mudeli testimiseks ehk valideerimiseks loodi valideerimisandmestik, mis koosnes testimisest erinevate väärtustega, kuid olemuselt sarnastest elektrienergia hindadest ning ruumi kasutamise graafikust. Valideerimine on tähtis, et vältida ülesobitamist – mudelit mis tuleb toime vaid treenimisel kasutatud andmetega.

2.3.1 Agendi treenimine

Stiimulõppe agendi treenimine on üldjuhul edukam, kui maksimaalne sammude arv ehk episoodi pikkus on pikem. Nii on sellel rohkem võimalusi õppida ning vähendatakse tõenäosust, et hiljem töö käigus tekkiv olukord on agendile täielikult tundmatu. Kuid pikemate episoodide kasutamine nõuab ka rohkem aega, mistõttu on vaja leida väärtus, mis loob mõistliku ajaga efektiivse agendi. See sõltub suurel määral kasutatava riistvara jõudlusest ning treeningparameetritest. Antud töös leiti läbi katsetamise, et parimad tulemused on võimalik saada 34 560 sammu pikkuse episoodiga. Ehk treenimisel simuleeriti 120 ööpäeva pikkust ajavahemikku.

Treenimisel ei olnud vaja kasutada maksimaalset episoodide arvu, kuna jälgiti episoodide summaarse preemia kujunemist. Üldjuhul võib agendi lugeda treenituks, kui summaarne preemia koondub mingi väärtuse juurde. Seda jälgiti ning koondumise tuvastamisel treenimine katkestati. Antud töös kasutatud mudelitel kulus selleks 100 – 200 episoodi ning ligikaudu kümme tundi arvutusaega. Joonisel 2.6 on esitatud näide koondunud algoritmist. Koondumise kiirust mõjutavad ka muud tegurid, millest peamised on seotud müraga. Nende tegurite töös kasutatud väärtuseid saab näha lisas 1 esitatud koodis.



Joonis 2.6 Preemia koondumine lõplikule väärtusele

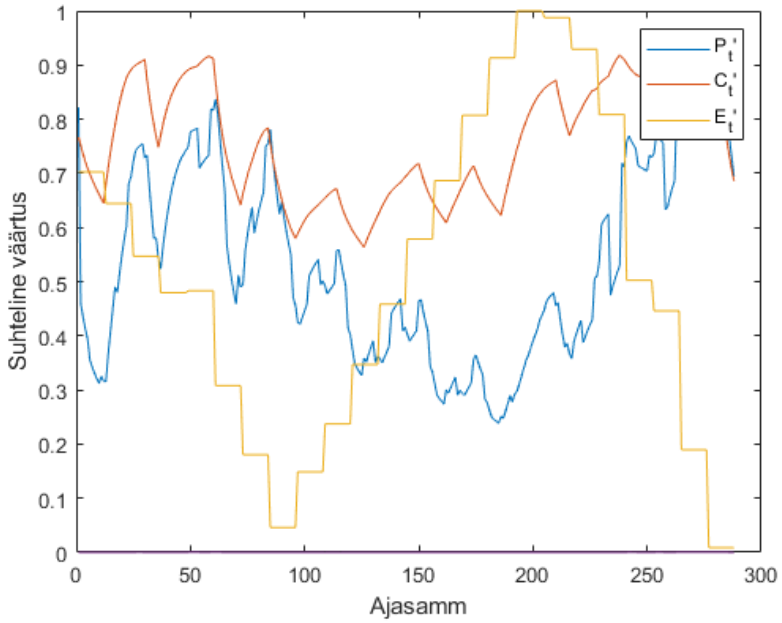
2.4 Tulemuste analüüs

Treenitud agendi testimiseks simuleeriti seda ühele aastale vastavate andmete alusel ning võrreldi käsitsi koostatud mudeli. Elektrienergia hindadena kasutati Nordpooli 2020 aasta EE andmeid [34]. Kuna suuruste W_C ning W_E muutmisega on võimalik treenida agente, mis arvestavad CO_2 kontsentratsiooni ja elektrienergia maksumust erinevalt (tabel 2.2), siis on võrdluseks välja toodud kaks erinevat mudelit. Üks neist proovib saavutada võimalikult madalat hinda ning lubab selleks veidi kõrgemat keskmist CO_2 taset. Teine hoiab pikaajalise CO_2 taseme sarnase konstantse mudeliga, kuid üritab seda saavutada odavamalt.

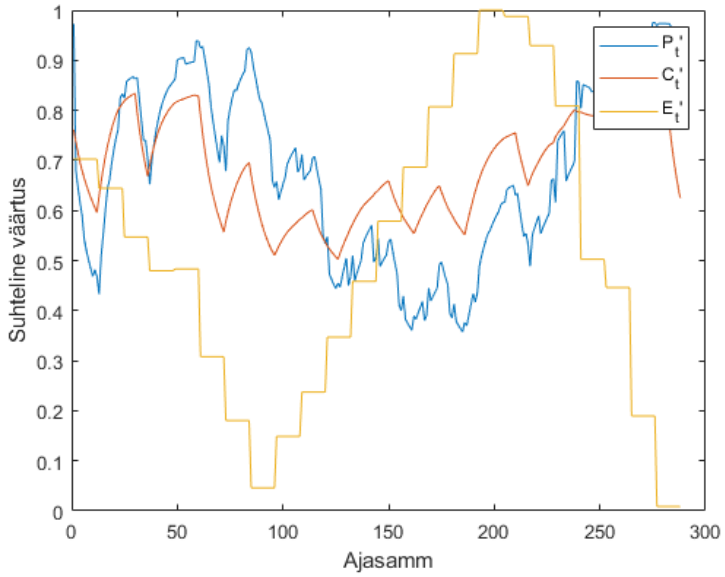
2.4.1 Stiimulõppe mudel

Joonistel 2.7 ja 2.8 on esitatud stiimulõppe väljundgraafikud sama ajavahemiku kohta, kuid erinevate W_C ja W_E väärtuste puhul. Esimesel puhul on kasutatud mudelit, mis on treenitud väärtustega $W_C = W_E = 1$. See suudab hoida CO_2 taseme etteantud taseme lähedal, hoides elektrienergia maksumuse madala. Teine mudel on treenitud väärtustega $W_C = 1{,}3$ ja $W_E = 1$ ning see hoiab CO_2 taseme etteantud tasemest madalamal, kuid saavutab selle suurema maksumuse arvelt.

Graafikutelt on näha, et CO_2 tase hoitakse etteantud väärtuse lähedal, kuid samas arvestatakse elektrienergia hinnaga. Kallimal ajal lubatakse CO_2 tasemel tõusta ning odavamal ajal suurendatakse ventilaatori võimsust, et CO_2 taset alandada. Ning kuna tegu on närvivõrguga, siis ei ole praktiliselt võimalik valemi kujul esitada seost hinna ja ventilaatori võimsuse vahel.



Joonis 2.7 Stiimulõppe mudeli väljund ühe ööpäeva kohta kui $W_C = W_E = 1$



Joonis 2.8 Stiimulõppe mudeli väljund ühe ööpäeva kohta kui $W_C = 1{,}3$ ja $W_E = 1$

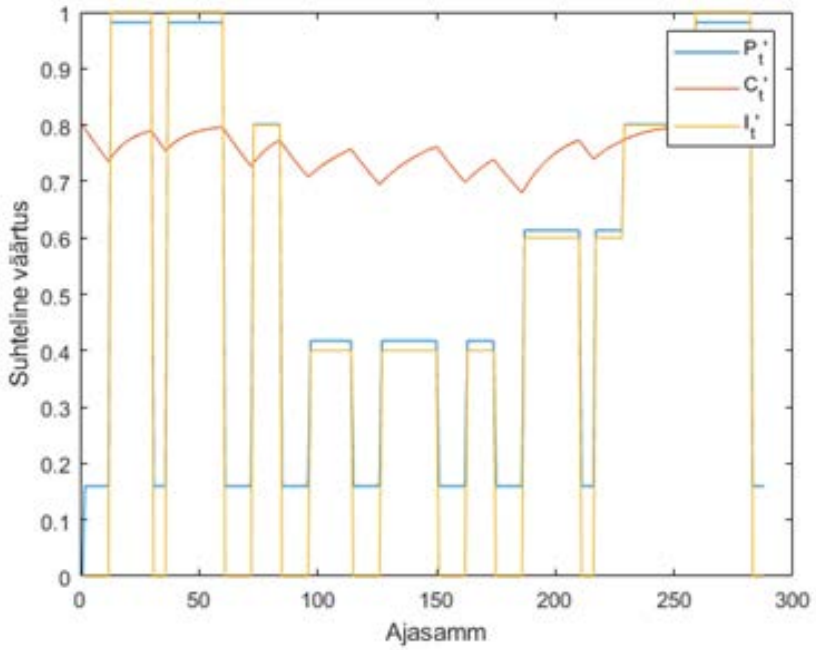
Kuigi stiimulõppe agendile antakse sisendina ette lisaks antud ajahetke hinnale ka järgnevate ajahetkede hinnad, siis ei ole väljundgraafiku järgi aru saada, et tulevaste hindadega reaalselt arvestataks. Kuna treenimise ajal avaldab nende sisenditega arvestamine preemia väärtusele üsna väikest mõju, siis ei ole need närvivõrgu käitumist tuntavalt mõjutanud. Sellest võib järeldada, et mudel on alatreenitud ning selle tõhusust saaks tõsta efektiivsema treeningu läbi. Kuid efektiivsem treening nõuaks ka suuremat arvutusjõudlust ning antud töö eesmärk oli demonstreerida masinõppe kasulikkust, mitte luua optimaalne mudel. Siiski oli tulevaste hindadega arvestamisest kasu, kuna sellise konfiguratsiooniga saavutati parim tulemus.

2.4.2 Käsitsi loodud mudel

Käsitsi koostatud mudeli all mõeldakse lihtsat juhtimisstrateegiat, mis leiab ventilaatori võimsuse analüütiliselt valemi abil. Kuna selle mudeli eesmärk on esindada traditsioonilist juhtimislahendust, siis määrab see ventilaatori võimsuse selliselt, et hoida ruumi siseõhu CO_2 kontsentratsiooni konstantsel tasemel (valem 2.10), mille väärtus antud juhul on võrdluse loomise eesmärgil võetud C_G . Seega on ventilatsioonisüsteemi võimsus tuletatud valemist 2.5.

$$P_t' = \frac{I_t \cdot \Delta C_{gen}'}{\Delta V_{max}' \left(C_G' + I_t \cdot C_{gen}'\right) - C_O' \cdot \Delta V_{max}'} \tag{2.10}$$

Joonisel 2.9 on esitatud käsitsi loodud süsteemi väljundgraafik. See näeb välja ootuspärane ehk ruumi siseõhu CO_2 kontsentratsioon püsib etteantud väärtuse lähedal olenemata elektrienergia hinnast. Suurus I_t' on siin ruumis olevate inimeste suhteline arv, kus väärtus 1 vastab ruumi maksimaalsele täitumusele ehk 5 inimesele.



Joonis 2.9 Käsitsi loodud mudeli väljund ühe ööpäeva kohta

2.4.3 Võrdlus pikema aja peale

Mõlema mudeli pikaajalise soorituse hindamiseks simuleeriti neid ka ühe aasta andmetega, mille tulemused on välja toodud tabelis 2.2. Tabelisse on lisatud ka erinevate W_C ning W_E väärtustega treenitud agendid, et demonstreerida süsteemi paindlikkust. Suurema W_C puhul arvestab agent rohkem CO_2 väärtusega ning W_E puhul pigem elektrienergia maksumusega. Neid suuruseid enne treenimist muutes on võimalik luua erineva karakteristikuga juhtimisstrateegiaid. Suurus $\sum \Delta C_t'$ tähistab kogu ajavahemiku summaarset C_t' erinevust etteantud CO_2 tasemest, milleks siinjuhul on 0,8. Suurus $\sum E_t$ (€) tähistab kogu ajavahemiku summaarset elektrienergia maksumust.

Tabel 2.2 Tulemused erinevate mudelitega

Mudel	$\sum \Delta C_t'$	$\sum E_t$, €
Manuaalne	-4,04·10^3	16,3
$W_C = 1$ $W_E = 1$	-5,84·10^3	10,3
$W_C = 1$ $W_E = 2$	-1,17·10^3	8,48
$W_C = 1{,}3$ $W_E = 1$	-1,41·10^4	17,4

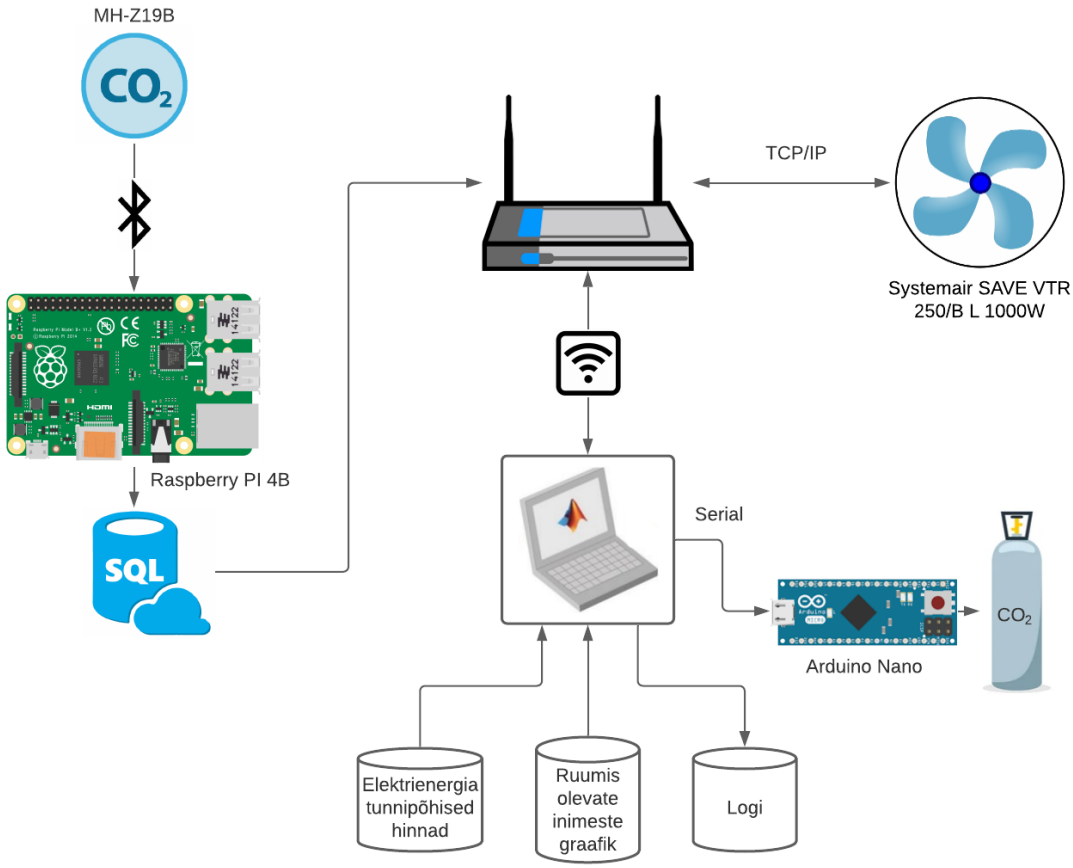
Tabelist on näha, et etteantud parameetrite järgi on võimalik luua juhtimisstrateegia, mis suudab oma ülesannet täita mitmel erineval moel. Vastavalt lõppkasutaja soovidele on võimalik hoida pea konstantset CO_2 taset madalama hinnaga või lubada sellel soovitud vahemikus võnkuda, et saavutada minimaalne elektrienergia maksumus. Kuna väärtuste $W_C = W_E = 1$ puhul saadi autori arvates parimad tulemused, siis otsustati edaspidi kasutada nende väärtustega treenitud mudelit.

3. TESTIMINE REAALSEL KOORMUSEL

Stiimulõppel põhineva juhtimisstrateegia valideerimiseks testiti seda ka reaalses ruumis sealse ventilatsiooniseadmega. Testimine viidi läbi Akadeemia tee 5a ühiselamu keldrikorrusel asuvas demoruumis, milles asub ventilatsiooniseade Systemair SAVE VTR 250/B L 1000W [40]. Kasutatava ruumi ning seadme parameetreid kasutati mudeli loomisel ning need on esitatud eelnevas peatükis.

3.1 Testimise ettevalmistamine

Algoritmi testimiseks loodi Matlab keskkonnas programm, mis kasutab ventilatsiooniseadme juhtimiseks treenitud stiimulõppe mudelit. Programm jooksis sülearvutis, mis oli ventilatsiooniseadmega seotud läbi Modbus TCP/IP ühenduse. Süsinikdioksiidi doseerimiseks ühendati sülearvutiga läbi Modbus Serial ühenduse gaasiballooni väljundit kontrolliv mikrokontroller. Ning CO_2 andmete lugemiseks kasutati andurit, mis salvestas mõõteinfo võrgus olevasse andmebaasi. Kõik kasutatud seadmed ning nendevahelised ühendused on esitatud joonisel 3.1.



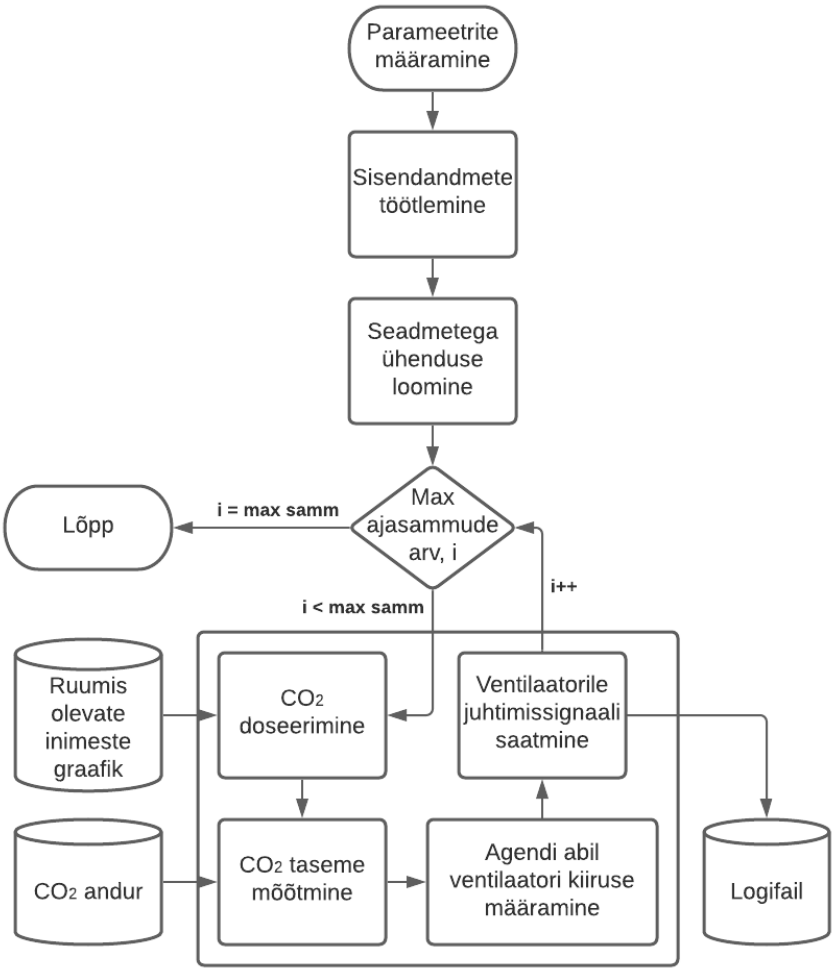
Joonis 3.1 Testimisel kasutatud seadmete skeem

Testimisel kasutati reaalset ajasammu pikkusega üks minut. See tähendab, et iga ühe minuti tagant sooritati mõõtmised ning saadeti seadmetele juhtimissignaal. Simulatsiooni ajasammu t_s väärtuseks võeti sarnaselt treenimisele viis minutit. Seetõttu oli võimalik testandmete kogumist kiirendada ning ühe ööpäeva simuleerimiseks kulus vaid 4,8 tundi. Lühemat ajasammu kasutades on võimalik testimist kiirendada ning sama ajavahemiku vältel rohkem andmeid koguda. Kuid liiga lühikese ajasammu puhul ei jõua andur tuvastada muutusi CO_2 taseme muutuses, mistõttu tekiks testandmetesse hulk juhtimiskäske, millel ei ole näiliselt mõju ruumi siseõhule.

Demoruumiga oli ühendatud ka hoone üldine ventilatsioonisüsteem, mille toimet ei olnud võimalik piisaval määral elimineerida. Seetõttu sai testimist edukalt läbi viia vaid öösel, kuna sel ajal üldine ventilatsioon demoruumi ei teenindanud ning praktiliselt kogu õhuvool ruumis oli juhitava ventilatsiooniseadme poolt põhjustatud. See piiras testimise kestvuseks 562 minutit, mis vastavalt ajasammule andis simuleeritud aja pikkuseks ligikaudu 47 tundi. Sellest piisab mudeli hindamiseks, kuna elektrienergia suhteline hind saavutab 24 tunni jooksul väärtused vahemikus 0 kuni 1, mistõttu on kontrollitud mudeli käitumine iga hinnataseme juures.

Testimise läbiviimiseks kirjutati Matlab keskkonnas programm, mis viib täpselt igal ajasammul läbi iteratsiooni, mille käigus see omandab vajalikud mõõteandmed ning edastab need stiimulõppe agendile, mis genereerib väljundi vahemikus 0,16 kuni 1. See teisendatakse sobivale kujule ning edastatakse ventilatsiooniseadmele. Tsükli sees vabastatakse ka doseerimisgraafiku järgi balloonist süsinikdioksiidi ning salvestatakse kõik vajalikud suurused logifaili. Plokkskeem antud programmi tööpõhimõttest on esitatud joonisel 3.2 ning täielik kood on olemas lisas 4.

Antud programm koosneb kiiresti täidetavatest käskudest, mis ei nõua palju arvutusjõudlust ning on võimelised jooksma ka mikroarvuti peal. See kehtib ka närvivõrgu abil väljundi leidmise kohta, mis koosneb peamiselt maatriksarvutustest. Seetõttu ei osutu taolise juhtimissüsteemi rakendamine vajaminevate arvutite poolest oluliselt kulukaks ega mõõtmetelt mahukaks.



Joonis 3.2 Testimisprogrammi plokkskeem

Elektrienergia tunnipõhised hinnad saadi Nord Pooli andmebaasist [34] ning kasutati 2021 aasta aprillikuu EE andmeid vahemikus 01.04.2021 00:00 kuni 03.04.2021 00:00.

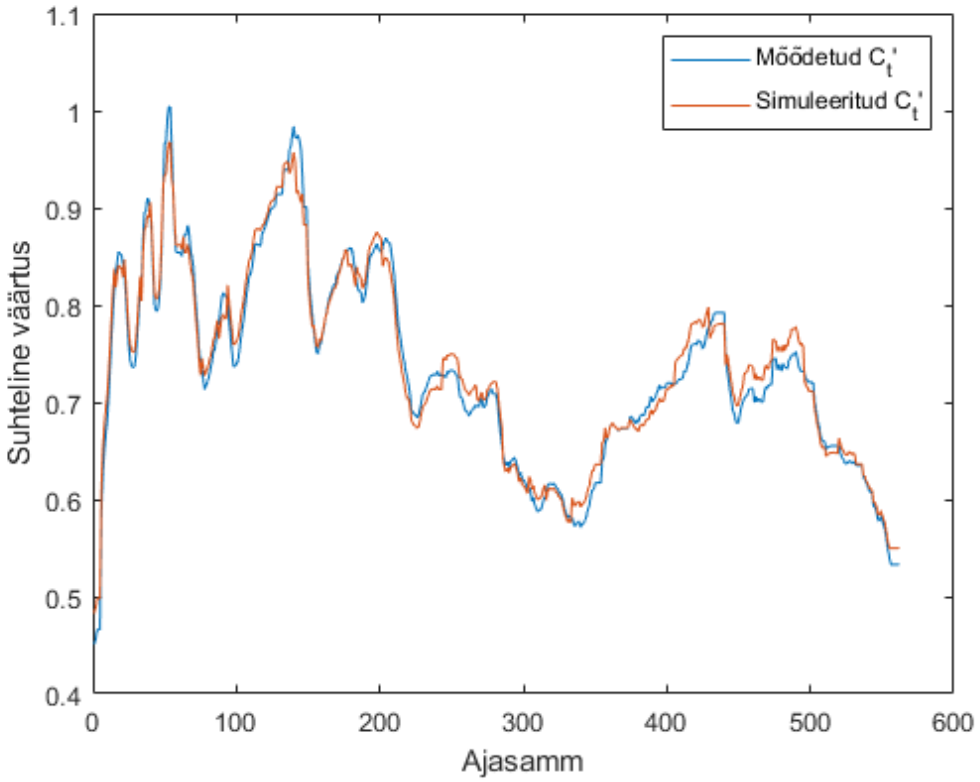
Süsinikdioksiidi doseerimiseks kasutati gaasiballooni, mille düüsi kontrolliti samm-mootori abil. Mootori juhtmoodul oli ühendatud Arduino Nano mikrokontrolleriga, millele oli läbi Serial-ühenduse otse Matlab tarkvarast võimalik juhtimissignaale saata. Ballooni väljund oli ühendatud pika voolikuga, millesse oli regulaarse vahemaa tagant puuritud augud, et hajutada gaasi doseerimine ühtlaselt üle ruumi. Enne testimise algust mõõdeti anemomeetri abil balloonist tuleva gaasi kogus erinevatel mootori positsioonidel. Testimise ajal saadeti mikrokontrollerile vastav signaal, et doseerida vajalik kogus süsinikdioksiidi. Sarnaselt algoritmi treenimisega, loodi ka testimise jaoks *occupancy simulator* [35] aplikatsiooni abil ruumi kasutamise graafik, mis sisaldab igale ajahetkele vastavat ruumis viibivate inimeste arvu. Inimeste simuleerimiseks vabastati igale inimesele vastavalt 20L/h süsinikdioksiidi. Ruumis viibivate inimeste suhteline kogus igal ajasammul I_t' on kujutatud joonisel 3.5.

Siseõhu CO_2 taset mõõdeti infrapuna anduriga MH-Z19B [41], mis oli bluetooth side kaudu ühendatud Raspberry PI 4B mikroarvutiga. See oli omakorda ühendatud võrguga ning salvestas andurilt saadud väärtused SQL andmebaasi, millest Matlab tarkvara suutis igal ajasammul lugeda ning väärtused stiimulõppe agendile edastada. Andur paigaldati ventilatsiooniseadme väljavõtule lähedale, et võimalikult efektiivselt mõõta ruumi keskmist CO_2 kontsentratsiooni [42].

3.2 Tulemuste analüüs ja algoritmi valideerimine

3.2.1 Simuleeritud ja reaalne CO_2 doseerimine

Simuleeritud mudelis määratakse õhu CO_2 tase matemaatiliselt ning selle muutus on kohene. See tähendab, et ruumis olevate inimeste poolt tekitatav süsihappegaas levib hetkega ning tasakaaluoleku saavutamiseks aega ei kulu. Reaalses elus on see aga teisiti, kuna tekkinud süsihappegaas vajab aega, et ruumis hajuda ning ventilaatori kiiruse reguleerimise mõju on nähtav alles mõne aja pärast. Ehk võib öelda, et õhu CO_2 tasemel on oma inerts. Ajaline nihe simuleeritud ning testitud CO_2 tasemete vahel on demonstreeritud joonisel 3.3. Simuleeritud CO_2 tase on siinjuhul arvutatud valemi 2.5 abil ning igal ajasammul on kasutatud sama CO_2 doseerimise ja ventilaatori kiiruse väärtuseid mis testimisel.

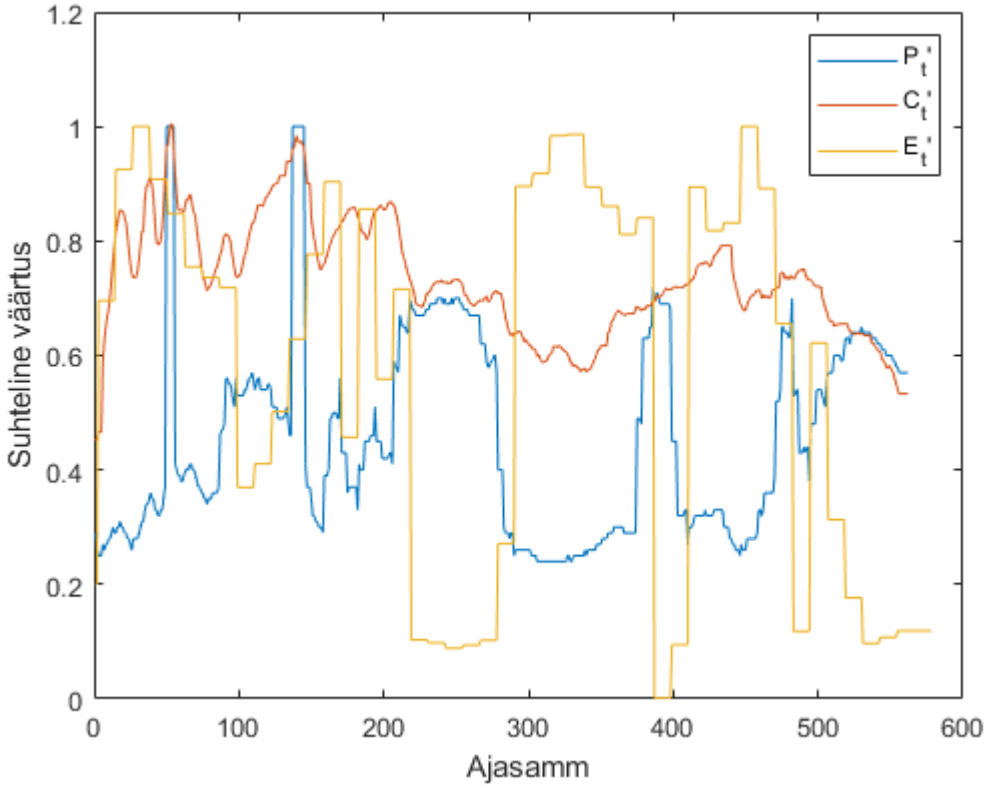


Joonis 3.3 Reaalne ja simuleeritud CO_2 doseerimine

Nagu näha, siis kõverad on kujult väga sarnased, kuid simuleeritud CO_2 väärtus on enamjaolt mõne ajasammu võrra ees. Kuna testimisel kasutati ühe minutilist ajasammu, siis võib antud ruumis CO_2 hajumise kiiruseks lugeda ligikaudu viis minutit. Kõverate sarnane kuju annab kinnitust, et mudelis kasutatud valemid on üsna praktilised ning et testruumi õhuvoog on pea täielikult ventilatsiooniseadme põhjustatud. Täpsema stiimulõppe mudeli saavutamiseks oleks mõistlik arvestada CO_2 taseme inertsiga.

3.2.2 Testimise tulemused

Eelmises alapeatükis demonstreeriti CO_2 taseme modelleeritud käitumise sarnasust testimise käigus mõõdetud väärtustele, mistõttu on mõistlik testimise tulemusi edasi analüüsida. Joonisel 3.4 on esitatud mudeli väljundgraafik testimise vältel. Sarnaselt eelmises peatükis olevate graafikutega, on sellel kujutatud igal ajasammul suhteline ventilaatori võimsus P_t', siseõhu CO_2 suhteline tase C_t' ning suhteline elektrienergia hind E_t'.



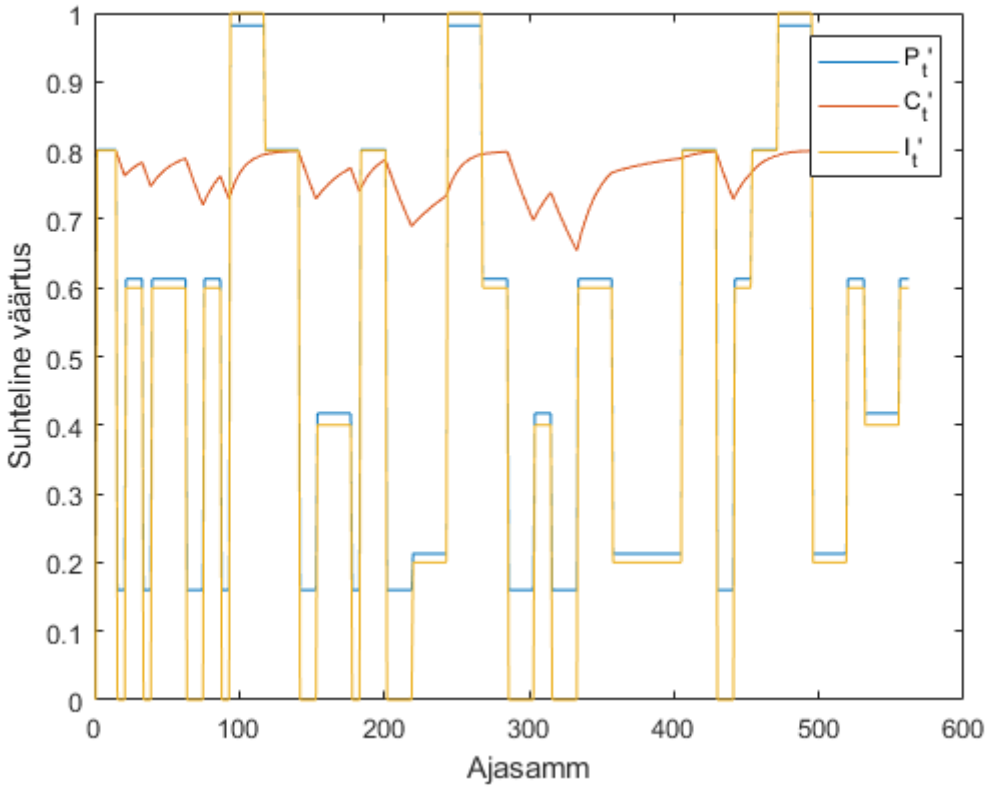
Joonis 3.4 Testimise väljundgraafik

Graafiku järgi saab stiimulõppe agendi juhtimisstrateegia lugeda edukaks, kuna vastavalt ootustele suurendatakse ventilaatori võimsust sel ajal, kui elektrienergia hind on madalam, hoides seejuures CO_2 taset piisavalt madalal. Kuid samas ei ole tegu lihtsa pöördvõrdelise funktsiooniga, vaid keeruka närvivõrgu arvutuskäiguga. Tulemuste võrdlemiseks on kasutati sama meetodit mis simulatsiooni korral ning leiti terve ajavahemiku kohta summaarne CO_2 taseme erinevus etteantud väärtusest ning elektrienergia kogumaksumus. Seega saadi suuruste $\sum \Delta C_t'$ ning $\sum E_t$ väärtusteks vastavalt -37,7 ning $5{,}7 \cdot 10^{-2}$ €.

Graafiku alguses on kahes punktis näha ventilaatori võimsuse hüppeline tõus maksimumi. See tuleneb koodis olenevast turvameetmest, kus väärtuse $C_t' > 0.96$ korral seatakse ventilaator maksimumkiirusele, et vältida liiga kõrget C_t' väärtust. Seda võib pidada stiimulõppe algoritmi veaks, kuna ideaalsel juhul ei oleks sellist turvameedet vaja. Selline käitumine tuleneb antud juhul preemiafunktsioonist (valemid 2.8 ja 2.9), kuna maksimaalse CO_2 taseme ületamisele ei vasta piisavalt suur negatiivne preemia. Kuid probleemi lahendus ei ole triviaalne, kuna kui funktsiooni sel juhul liiga suur negatiivne preemia lisada, siis muutub mudel liialt ettevaatlikuks ning hoiab CO_2 taset alati ebavajalikult madalal. Viga aitaks parandada preemiafunktsiooni täiustamine ning ka põhjalikum mudeli treenimine. Siiski aitab see viga demonstreerida reaalses elus testimise vajalikkust, kuna simulatsioonides sellist olukorda ei tekkinud.

3.2.3 Võrdlus lihtsa juhtimisstrateegiaga

Stiimulõppe mudeli efektiivsuse hindamiseks võrreldi testimise tulemusi ka eelmises peatükis tutvustatud käsitsi loodud juhtimisalgoritmiga, mis püüab igal ajahetkel hoida konstantset CO_2 väärtust (valem 2.10). Antud algoritmile anti ette iga ajasammu jaoks samad CO_2 doseerimise ning elektrienergia hindade andmed, mida kasutati testimisel. Nende andmetega algoritmi läbi arvutamine annab ligikaudsed väärtused, mis oleks saavutatud, kui seda oleks kasutatud stiimulõppe mudeli asemel. Algoritmi väljundgraafik on esitatud joonisel 3.5.

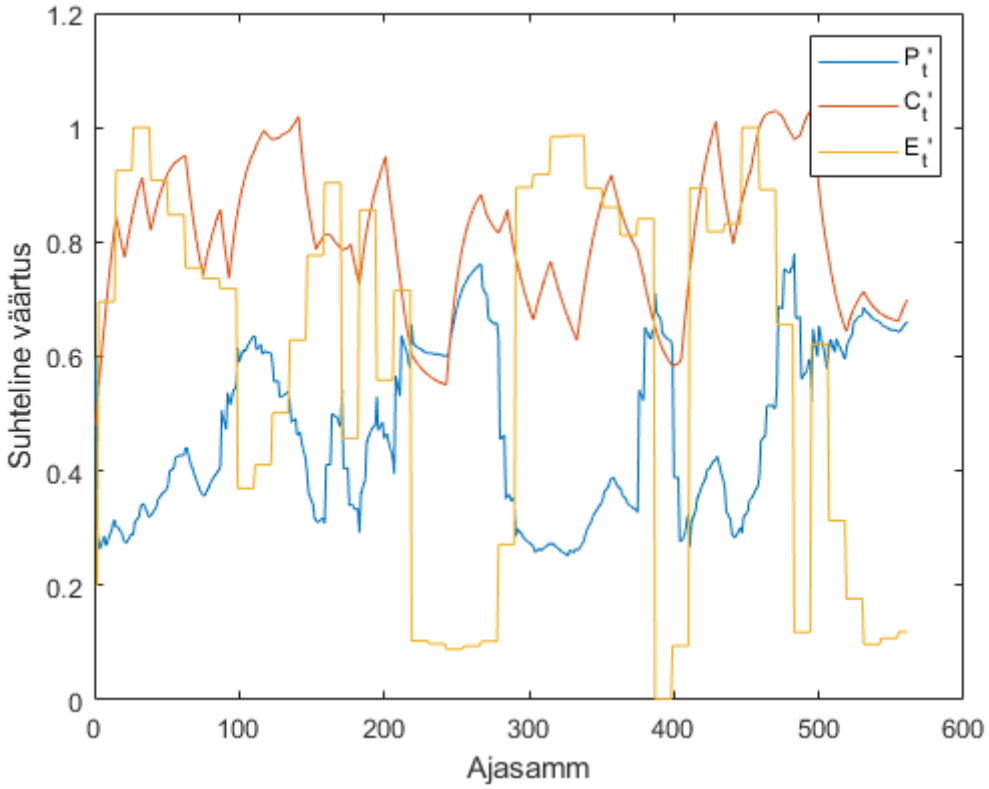


Joonis 3.5 Lihtsa juhtimisstrateegia väljundgraafik

Nagu ka eelmises peatükis, siis püüab antud juhtimisstrateegia hoida CO_2 kogust etteantud väärtuse juures ning ventilaatori kiirus ei sõltu elektrienergia hinnast. Väärtused $\sum \Delta C_t'$ ning $\sum E_t$ on antud juhul vastavalt -17,9 ning $9{,}8 \cdot 10^{-2}$ €. See tähendab, et stiimulõppe mudel suutis hoida keskmise CO_2 taseme märkimisväärselt madalamal, kulutades selleks elektrienergiale vaid 58,2% summast, mis oleks kulunud lihtsat juhtimisstrateegiat kasutades.

3.2.4 Võrdlus simuleeritud ja testitud efektiivsuste vahel

Andes stiimulõppe mudelile ette testimisel kasutatud CO_2 doseerimise kogused ning ajasammudele vastavad elektrienergiad, kuid lastes sellel ise arvutada ventilaatori võimsuse ning järgnevate ajasammude CO_2 tasemed, on võimalik hinnata mudeli võimekust ette ennustada reaalse süsteemi käitumist. Stiimulagendi väljundgraafik nende andmete põhjal on esitatud joonisel 3.6.



Joonis 3.6 Simulatsiooni väljundgraafik testimisel kasutatud sisendandmete korral

Simuleeritud graafik näeb kujult välja sarnane testimise tulemustega, kuid CO_2 tase on kogu testimise vältel kõrgem. $\sum \Delta C_t'$ ja $\sum E_t$ on siinjuhul vastavalt 3,5 ja $5{,}2 \cdot 10^{-2}$ €. Kuna kasutati stiimulõppe mudelit, mis on treenitud tähtsustama elektrienergia hinda üle CO_2 taseme, siis on ka mõistetav, et ideaalsetes oludes suudab see saavutada madalama kogumaksumuse. Kuid tähelepanuväärne on keskmise CO_2 taseme suur erinevus. Võib järeldada, et antud mudeli puhul on reaalsest elust tulenevad faktorid kasulikud CO_2 väärtuse madalal hoidmise puhul, kuid väiksel määral kahjulikud maksumuse minimeerimisel. Samuti on võimalik väita, et simulatsiooni abil on võimalik teatud määral ette ennustada ventilatsioonile kuluva elektrienergia maksumust, kuid mitte CO_2 taset samas ajavahemikus. Antud juhul on simulatsioonis ja testimisel kasutatud identseid ruumi kasutatavuse andmeid, mida reaalses elus ei ole üldjuhul võimalik täpselt ette teada.

3.3 Juhtimisalgoritmi rakendatavuse hinnang reaalsetes süsteemides

Eelnevalt esitatud graafikud ja väärtused demonstreerivad antud töös kasutatud stiimulõppe rakendatavust reaalsetes süsteemides üsna hästi. Kuna simuleeritud mudel käitub sarnaselt reaalsele süsteemile, siis on seda võimalik kasutada efektiivse

juhtimisstrateegia loomisel ning tulemuste prognoosimisel. Töös kasutatud stiimulõppe meetod ei ole optimeeritud juhtimisstrateegia loomiseks ning selle efektiivsus on piiratud treenimisel kasutatava riistvara madala jõudluse poolt. Kuid siiski saavutati mudelit kasutades head tulemused, mis kinnitavad masinõppe meetodite potentsiaali ventilatsiooniseadmete juhtimises ning energiapaindlikkuse pakkumises.

Masinõppel põhineva juhtimisstrateegia üheks eeliseks on selle skaleeritavus ja paindlikkus. Kui sama süsteemi oleks vaja rakendada teistsuguste parameetritega ruumis, siis piisaks vaid mõne suuruse muutmisest ning oleks võimalik treenida uus agent, mis suudaks ülesannet sooritada sarnase efektiivsusega. Samuti on võimalik töötavat mudelit testimise käigus kogutud andmete abil edasi treenida. Nii suudab mudel kohaneda muutuvate parameetritega ning paremini toime tulla raskesti modelleeritavate nähtustega nagu näiteks CO_2 taseme inerts.

3.3.1 Mudeli edasi arendamine

Testimise käigus selgus, et kasutatav mudel ei suuda arvestada CO_2 inertsiga, mis eksisteerib reaalsetes süsteemides. Seda ning ka teisi taolisi võimalikke vigu on võimalik parandada testandmete põhjal treenimisega. Nii on võimalik kasutada testimise käigus kogutud andmeid, et parandada matemaatilise mudeli puudujääke ning tõsta mudeli rakendatavust reaalsetes süsteemides. Selliseid meetodeid kutsutakse eluaegseks või mittestatsionaarseks stiimulõppeks [43].

Mudeli teiseks puuduseks on ebapiisav treenimine, mis tulenes kasutatava riistvara suhteliselt väikesest arvutusjõudlusest. Kuna antud töö eesmärgiks on demonstreerida masinõppe rakendatavust ventilatsioonisüsteemide juhtimisel, siis ei olnud optimaalse mudeli loomine hädavajalik. Kuid kommertslikult kasutatava süsteemi puhul oleks mõistlik kasutada pilveteenust, et suurendada kasutatavat arvutusjõudlust mitme suurusjärgu võrra. Selle tulemusena oleks võimalik luua mudel, mis suudab kasutada suuremal hulgal sisendeid ja väljundeid ning neid efektiivsemalt rakendada.

4. MÕJU ELEKTRISÜSTEEMILE

Juhtimisalgoritmi kasulikkuse hindamiseks on otstarbekas analüüsida selle poolt pakutavat paindlikkust elektrisüsteemile. Algoritmi loomisel lähtuti hinnapõhisest juhtimisest, mis on süsteemi stabiilsuse vaatest kasulik [44] [45], kuid mitte ainus võimalus energiapaindlikkuse pakkumiseks. Kuigi Eestis need laialdast kasutust leidnud ei ole, siis on populaarsust koguvad agregeerimisteenused, kus koormust juhitakse spetsiifilisema signaali järgi.

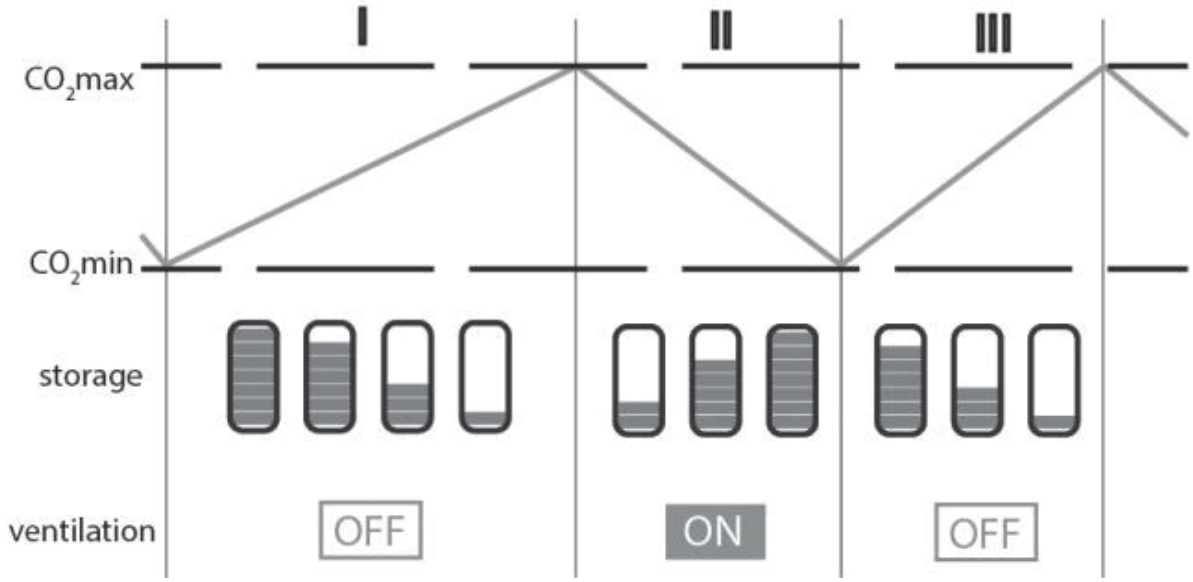
Käesolevas peatükis on töös loodud algoritmi poolt pakutavat energiapaindlikkust analüüsitud kolmel viisil. Esiteks on ventilatsioonisüsteemi käsitletud virtuaalse akuna, mis suudab energia talletamiseks kasutada puhast õhku. Aluseks on võetud testimise käigus saadud andmed, mille põhjal on igal ajahetkel arvutatud näiline energia hulk, mis kasutatud ruumis saadaval on. Teiseks on analüüsitud intelligentselt juhitud ventilatsiooniseadmete mõju elektrisüsteemi päevasele koormuskõverale. Kuna karakteerne kahe tipuga koormuskõver on süsteemi jaoks kadude allikas, siis on selle kitsendamine ihaldusväärne tegevus. Eesmärgi saavutamiseks kasutati stiimulõppe algoritmi sisendina hinna asemel koormuskõvera kujust tulenevat signaali. Viimaks hinnati algoritmi võimekust kohaneda päikesepaneelide tootmisgraafikuga, mis üldjuhul ei lange kokku tarbimisgraafikuga ning on seetõttu samuti kadude allikaks. Kõik kolm meetodit eeldavad konkreetse signaali olemasolu, mida algoritm on võimeline sisendina kasutama ning agregeerimisteenuse kasutamist.

4.1 Virtuaalne aku

Üks viis taastuvenergia allikatest tuleneva tootmise mittejuhitavuse kompenseerimiseks on akude kasutamine. Akud on hästi juhitavad, mistõttu sobivad need paljudeks rakendusteks nagu näiteks sageduse reguleerimine, pöörlev ja mittepöörlev reserv, energia tasakaalustamine, kriitilise koormuse toetamine ning tippude leevendamine. Kuid akudel põhinevate lahenduste hinnad on endiselt kõrged, mistõttu ei ole nende kasutamine elektrisüsteemides levinud [46]. Kuid energiat võivad salvestada ka hooned, mis toodavad puhast õhku.

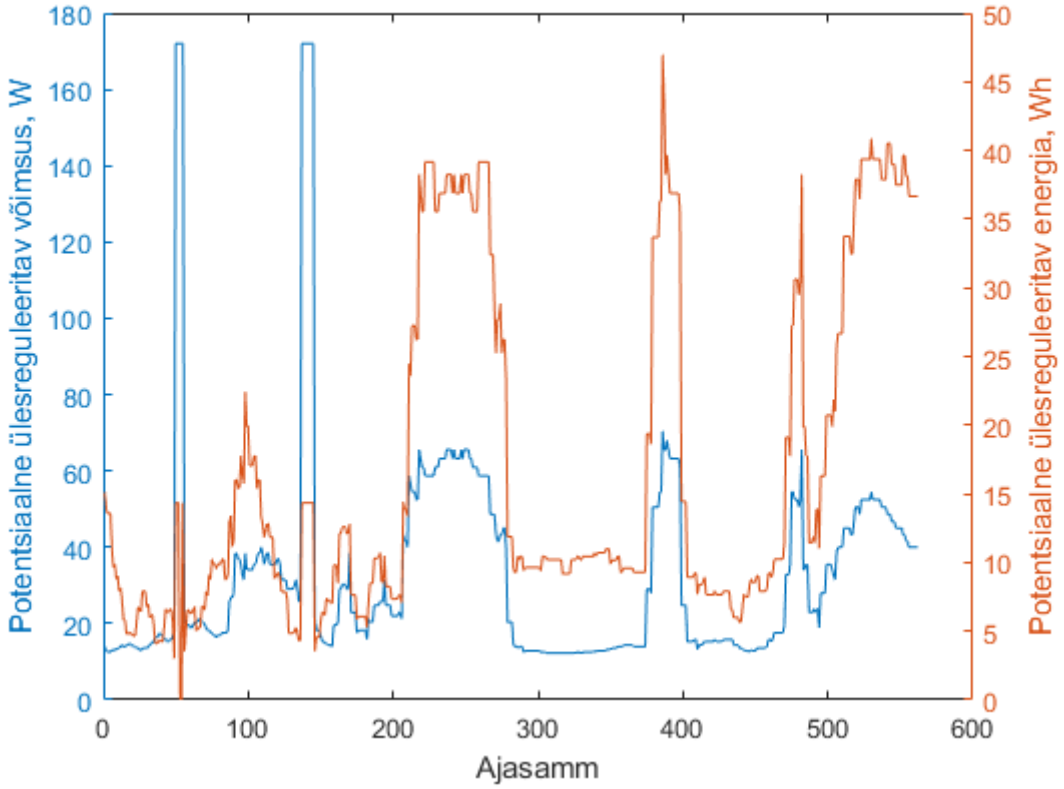
Ühendades ventilatsioonisüsteemi ruumiga, saab seda vaadelda virtuaalse akuna, mis talletab energiat puhtas õhus. Piltlikult öeldes on aku täis sel juhul, kui ruumi siseõhu CO_2 tase on minimaalne ning ventilaatori saab välja lülitada. See vähendab süsteemi koormust, mistõttu võib seda modelleerida kui akust toodetud energiat. CO_2 taseme tõusuga virtuaalne aku tühjeneb, kuna väheneb energia hulk, mida

ventilatsioonisüsteem näiliselt pakkuda suudab. Sellise aku poolt pakutav maksimaalne võimsus oleneb antud ajahetkel ventilaatori võimsusest, kuna see määrab ülesreguleerimise piiri. Ventilaator on võimalik täielikult välja lülitada, mis maksimeerib saadava võimsuse, kuid minimeerib kestvuse. Ventilaatori kiirust saab ka alandada osaliselt, et vähendada koormust pikema aja vältel. Analoogselt on võimalik ka koormuse suurendamine. Lihtne graafik sellise aku tööpõhimõttest on esitatud joonisel 4.1.



Joonis 4.1 Ventilatsioonil põhinev virtuaalne aku [47]

Antud töös kasutatava süsteemi poolt pakutavat energiahulka igal ajasammul on võimalik leida, kui valemi 2.5 iteratiivsel arvutamisel leitakse ajasammude hulk, mille korral CO_2 tase tõuseb uue ventilaatori võimsuse puhul maksimumini. Töös kasutatud näites on seda tehtud maksimaalse virtuaalse aku võimsuse juures ehk on eeldatud, et koormuse vähendamiseks lülitatakse ventilatsioon täielikult välja. Arvutamisel on kasutatud eelmises peatükis esitatud testimise käigus kogutud andmeid ja viie minutilist ajasammu ning graafik tulemustega on esitatud joonisel 4.2. CO_2 taseme tõusu kiiruse seisva ventilaatori puhul määrab ruumis olevate inimeste arv, mistõttu sõltub sellest ka pakutava energia hulk. Antud juhul on arvestatud halvima olukorraga ning igal ajahetkel eeldatakse, et ruumis on ventilatsiooni jaoks maksimaalne arv ehk viis inimest.



Joonis 4.2 Saadaval oleva energia ja võimsuse väärtused

Graafikus esitatud saadaolev võimsus on kujult identne joonisel 3.4 oleva ventilaatori võimsusega, kuna igal ajahetkel on võimalik koormust alandada ventilaatori hetkevõimsuse arvelt. Potentsiaalne ülesreguleeritav energia on arvutatud selle võimsuse põhjal ning on seda suurem, mida kauem on võimalik ventilaatorit väljalülitatuna hoida.

Vaadeldavas ajavahemikus on potentsiaalse elektrienergia keskmine väärtus 16,9 W/h ning keskmine võimsus 33,1 W. Kuna igal ajahetkel on arvestatud, et ruumis viibib maksimaalne hulk inimesi, siis saab pakutava energia koguses olla kindel. Reaalselt igal ajahetkel ruumis nii palju inimesi ei viibi, mistõttu on tegelik saadaoleva energia kogus suurem. Kuid kuna järgmiste ajahetkede info ei ole reaalses elus teada, siis saab kindla väärtuse anda vaid maksimumiga arvestades.

Kui paindlikkuse pakkumises olla agressiivsem ning maksimaalse inimeste arvu asemel arvestada keskmist ruumi kasutatavust (~2,4), siis on võimalik keskmiseks energia koguseks saada 34,1 W/h. See arv illustreerib paremini süsteemi üldist võimekust, kuid teatud ajahetkedel võib seda ülehinnata. Üheks lahenduseks oleks CO_2 taseme jälgimisega analüüsida ruumis olevate inimeste kogust ning seeläbi määrata tulevaste ajahetkede tõenäosuslik CO_2 genereerimise hulk.

Tuleb ka mainida, et antud suuruste puhul on rangelt arvestatud maksimaalse CO_2 kontsentratsiooni väärtusega 1000 ppm. Selle piiri lühiajaline ületamine üldjuhul probleeme ei tekita (tuntavad mõjud algavad 2000 ppm juures [48]), mistõttu on teatud oludes võimalik paindlikkust veelgi suurendada. Samuti eeldavad kasutatud andmed pidevat ruumi kasutamist, mistõttu on leitud tulemused rakendatavad vaid tööpäevale vastavale ajavahemikule. Suur osa ventilatsioonisüsteeme ei ole öösiti koormatud, mistõttu ei ole sel ajal kasu ka nende juhtimisest.

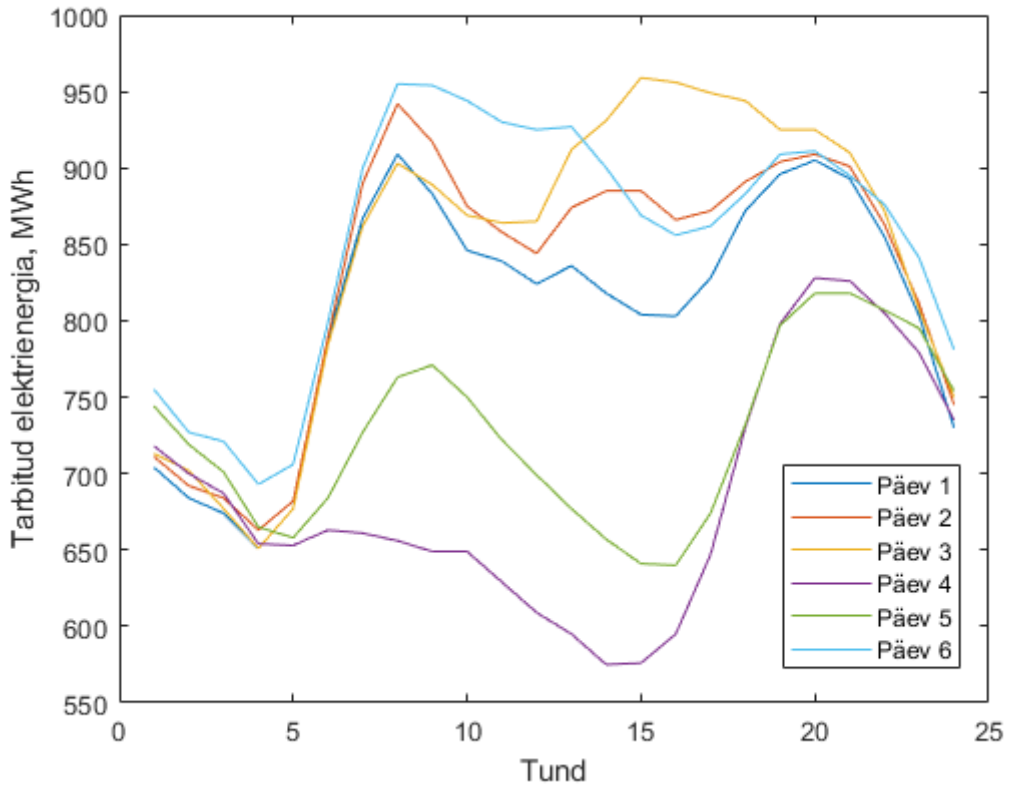
Töös kasutatava ventilatsiooniseadme ventilaatori maksimaalne võimsus on 172 W ning see suudab maksimaalselt teenindada viite inimest. Euroopa Liidus tarbivad ventilatsioonisüsteemid kokku 2% kogu elektrienergiast [1]. Kuna sarnast suurust ventilatsiooni koguvõimsuse jaoks saadaval ei ole, siis leiti antud töö raames Eesti ventilatsiooni koguvõimsus tarbitud energia kaudu. Et saadud väärtused oleks konservatiivsed, siis eeldati ventilaatori sajaprotsendilist kasutusaega ehk töös kasutatud ventilaatori aastaseks energiakuluks võeti $172 \cdot 8760 = 1{,}51 \cdot 10^6$ Wh. Võttes aastas Eestis kasutatava elektrienergia koguseks 8,44 TWh [49], saadakse 2% energiakasutust eeldades Eestis kasutatava ventilatsiooni koguvõimsuseks $172 \cdot 0.02 \cdot 8{,}44 \cdot 10^{12} / 1{,}51 \cdot 10^6 = 19{,}23 \cdot 10^6$ W ehk ligikaudu 20 MW. Tuleb meeles pidada, et antud väärtus ei ole leitud täpseks modelleerimiseks, vaid suurusjärgu hindamiseks. Üldjuhul ventilatsioon igal ajahetkel 100% võimsusega ei tööta, mistõttu on tegelik koguvõimsus märkimisväärselt suurem.

Võttes Eestis oleva ventilatsiooni koguvõimsuseks 20 MW, saab hinnata selle poolt salvestatava energia kogust. Eelnevalt leitud keskmise saadaoleva energia ja võimsuse suuruseid saab korrutada väärtusega $20 \cdot 10^6 / 172 = 116 \cdot 10^3$ ning leida energia kogus terve Eesti jaoks: $116 \cdot 10^3 \cdot 16{,}9 = 2{,}0$ MWh. Sarnaselt on võimalik leida ka keskmine saadaolev võimsus: $116 \cdot 10^3 \cdot 33{,}1 = 3{,}8$ MW. See tähendab, et kasutades taolist intelligentset juhtimist kõikide Eesti ventilatsiooniseadmete peal, on päeva jooksul keskmiselt saadaval lühiajaline salvestusvõimekus mahuga 2,0 MWh ning võimsusega 3,8 MW, mida on võimalik kasutada näiteks sageduse reguleerimiseks. Tuleb ka tähele panna, et need suurused on leitud kasutatud elektrienergia kogumaksumust minimeeriva algoritmi korral, mis ei ole optimeeritud maksimaalse võimsuse ega energia pakkumiseks.

4.1.1 Hinnapõhise juhtimise mõju süsteemi koormuskõverale

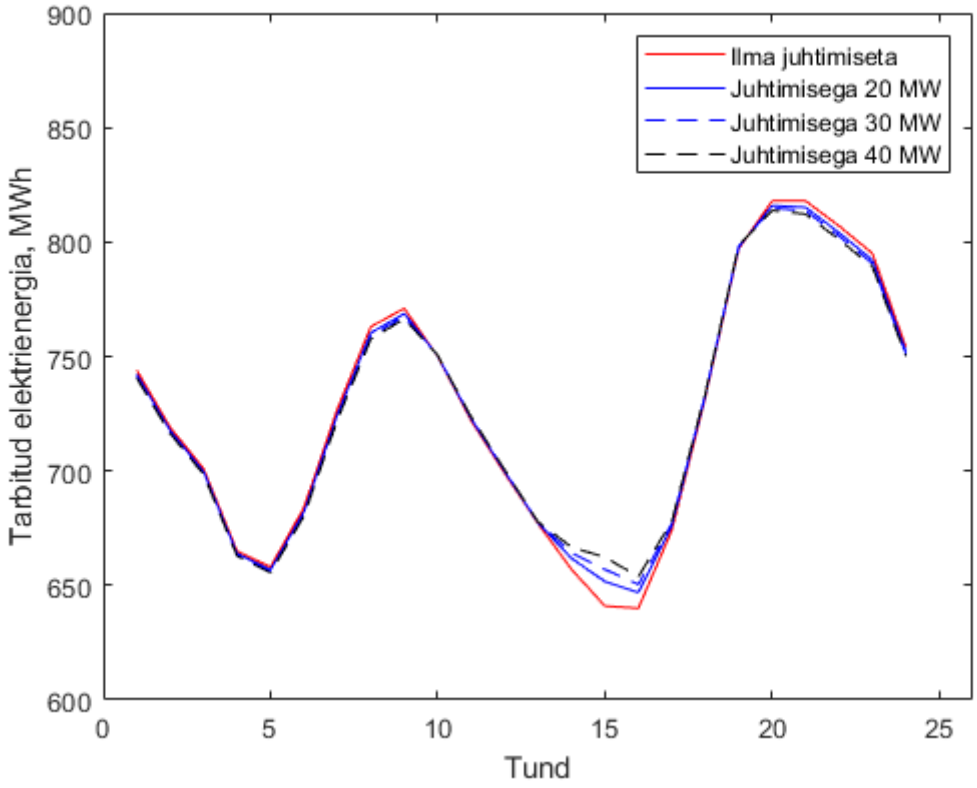
Üheks võimaluseks hinnata juhtimisalgoritmi mõju elektrisüsteemile on uurida selle efekti koormuskõverale. See kõver on karakteerse kujuga ning koosneb üldjuhul kahest tipust, mis tekivad enne tööpäeva algust ning peale tööpäeva lõppu. Elektrisüsteemi

seisukohalt oleks ideaalne, kui tarbimiskõver oleks lame, kuna see vähendaks eelkõige vajadust pakutavat võimsust kiiresti üles ja alla reguleerida. Joonisel 4.3 on esitatud kuue järjestikuse päeva tunnipõhine Eestis tarbitud elektrienergia graafik [50].



Joonis 4.3 Tunnipõhised koormuskõverad kuue järjestikuse päeva kohta

Mõju hindamiseks kasutati viienda päeva koormuskõverat ning sama päeva elektrienergia hindu. Joonisel 4.4 on esitatud mõjutatud koormusgraafikud erinevate ventilatsiooni koguvõimsuste juures. Graafikult on näha, et hinnapõhine juhtimine mõjutab peamiselt kõvera tööpäeva keskel asuvat lohku. See tuleneb sellele ajale vastavast madalast elektrienergia hinnast. Koormuskõvera maksimumid sisuliselt mõjutatud ei ole, kuna antud päeval need ei väljendunud piisavalt kõrgemates hindades. Kuid kui eesmärgiks võtta otseselt elektrisüsteemi koormuse mõjutamine, siis on paremate tulemuste jaoks võimalik kasutada hinnasignaali, mis on otseselt tuletatud koormuskõvera kujust. Seda võimalust uuriti järgmises alapeatükis.

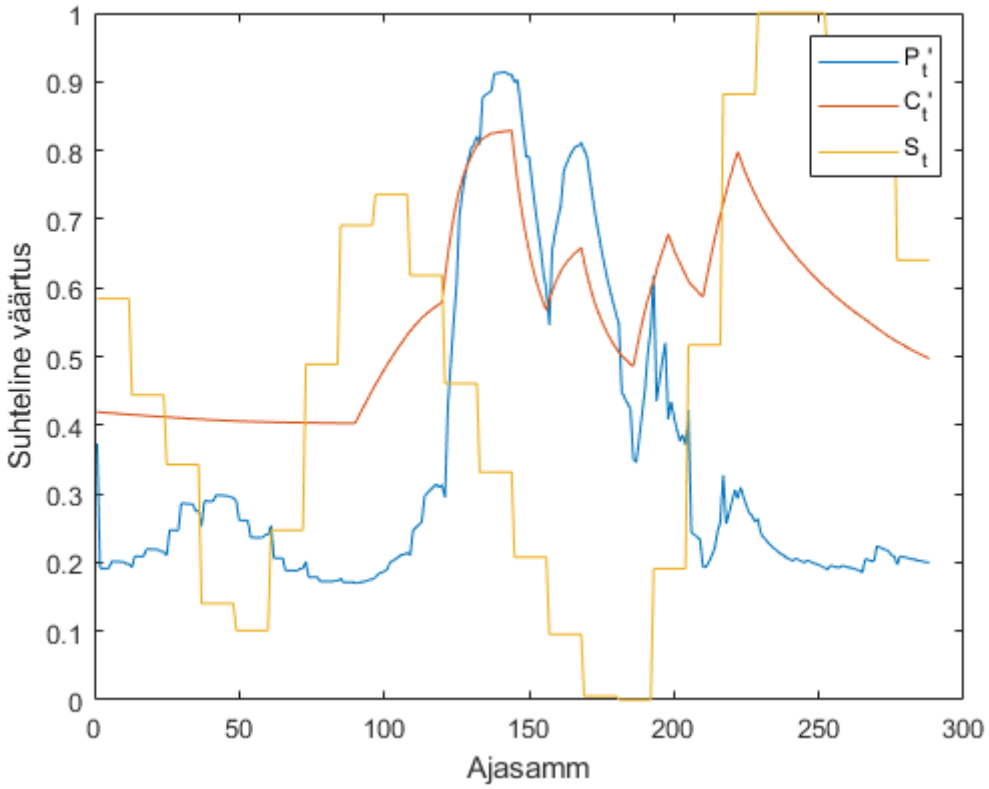


Joonis 4.4 Hinnapõhise juhtimise mõju elektrisüsteemi koormuskõverale

4.2 Koormuskõvera silumine

Energiapaindlikkuse saavutamiseks on võimalik reageerida ka hinnast erinevale süsteemilt tulevale signaalile. Üks näide sellest on tarbimiskõverale vastav signaal, mis püüab kõvera kuju siluda, et vähendada erinevust minimaalse ja maksimaalse koormuse vahel ning leevendada järsku koormuse muutumise kiirust [46]

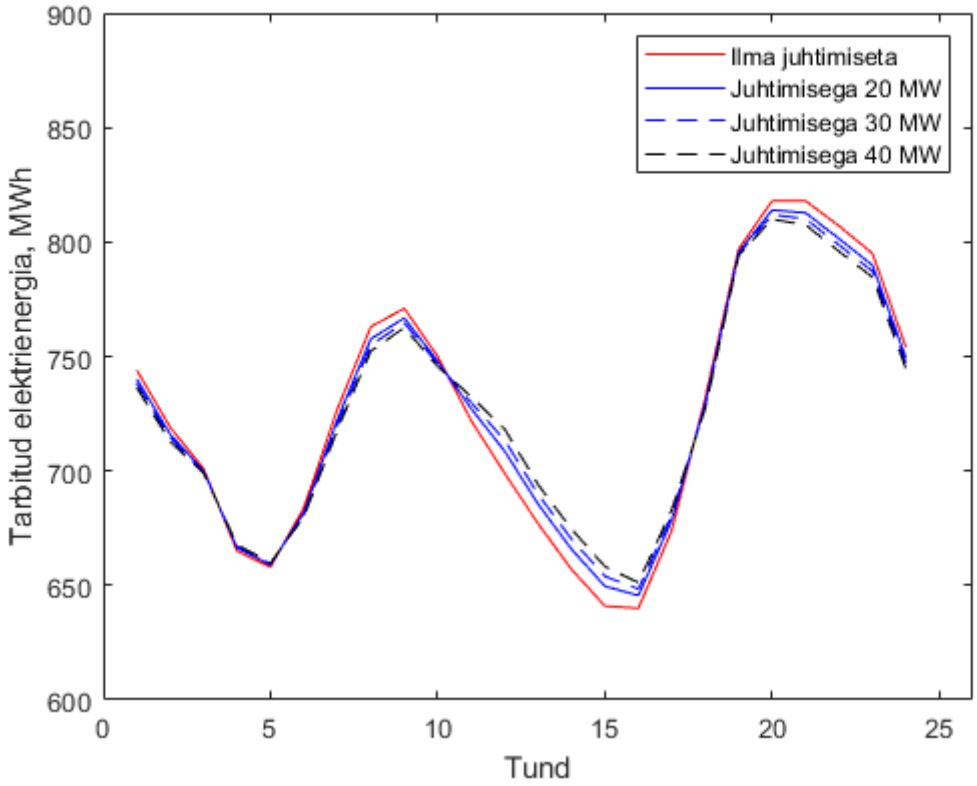
Seda on teatud määral võimalik saavutada energiapaindlikkuse abil, kui kõvera kuju käsitleda juhtimises sisendina. Kuna antud töö raames loodud juhtimisalgoritm kasutab sisendina hinnasignaali vahemikus 0 … 1, siis on võimalik seda kasutada ka otse koormuse silumisel, kui hind asendada koormuse kõverale vastava signaaliga. Demonstratsiooniks on seekordki välja valitud jooniselt 4.3 viienda päeva andmed ning selle põhjal on loodud simulatsioon, mille väljundgraafik on esitatud joonisel 4.5. Graafik sarnaneb eelnevates peatükkides esitatule, kuid seekord on normaliseeritud elektrienergia hinna E_t' asemel kasutatud juhtimissignaali S_t.



Joonis 4.5 Simulatsiooni väljundgraafik koormuskõverast tuleneva sisendi korral

Et hinnata taoliselt käituva mudeli mõju kogu Eesti elektrisüsteemile, on sarnaselt eelmise alapeatükiga eeldatud Eesti ventilatsiooni koguvõimsuseks 20 MW ning selle intelligentne juhtimine täies mahus. Kuna sisendina kasutati koormuskõverast tuletatud signaali, siis on eesmärgiks koormuse tasandamine. Joonisel 4.6 on võrreldud intelligentse juhtimisega saavutatud koormuskõverat baaskoormusega. Et illustreerida kõvera tasandamise efekti sõltuvust juhitava koormuse võimsusest, on graafikule lisatud ka 30 ja 40 MW koguvõimsusega ventilatsiooni stsenaariumid.

Intelligentset koormuse juhtimist kasutatava koormuskõvera arvutamiseks on baaskoormuselt lahutatud ventilatsioonile vastav tarbimine ning juurde lisatud simuleeritud ventilatsiooni võimsus. Kõverate alla jäävad pindalad on võrdsed, mis tähendab, et mõjutatud on vaid koormuse kuju ning tarbimise kogumaht on sama.



Joonis 4.6 Koormuspõhise juhtimise mõju süsteemi koormuskõverale

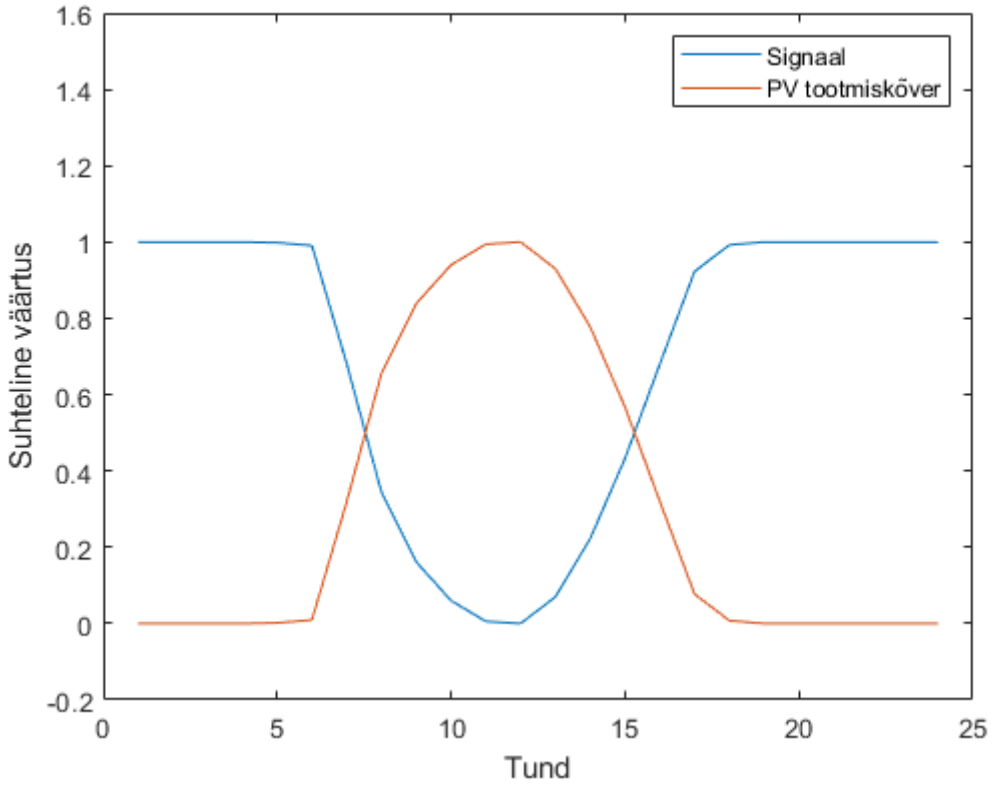
Graafikult on näha, et tänu intelligentsele juhtimisele on suudetud nihutada koormust tippudest lohkudesse. Ilma juhtimiseta asub ööpäeva tunnipõhine tarbimine vahemikus 640 kuni 818 MWh ning 20 MW võimsuse juhtimisega vahemikus 646 kuni 813 MWh ehk vahemik on 11 MWh võrra kitsam. Suurendades juhitava koormuse võimsust 40 MW peale, on vahemik algolekuga võrreldes 30 MWh võrra kitsam.

Koormuskõverast tuleneva juhtimissignaali kasutamine eeldab vastava teenuse olemasolu, kus stiimulite toimel on koormuskõvera järgi tarbimise juhtimine majanduslikult kasulikum eelnevalt käsitletud hinnapõhisest juhtimisest. Hetkeoludes üldjuhul taoline käitumine majanduslikult tasuv ei ole, kuid taastuvenergia suurema kasutuselevõtuga langeb süsteemi võimekus kiiresti muutuva koormusega toime tulla ning võrguettevõttel võib tekkida motivatsioon tarbijaid koormuskõverale reageerimise eest kompenseerida. Samuti on võimalik, et elektrienergia hind hakkab sarnanema rohkem süsteemi koormusega, mistõttu on hinnapõhisel juhtimisel funktsionaalselt identsed tulemused. Lisaks on süsteemile energiapaindlikkuse pakkumiseks koormuse asemel võimalik kasutada ka tootmisgraafikust tulenevat signaali.

4.2.1 Päikesepaneelide tootmiskõveraga kohandumine

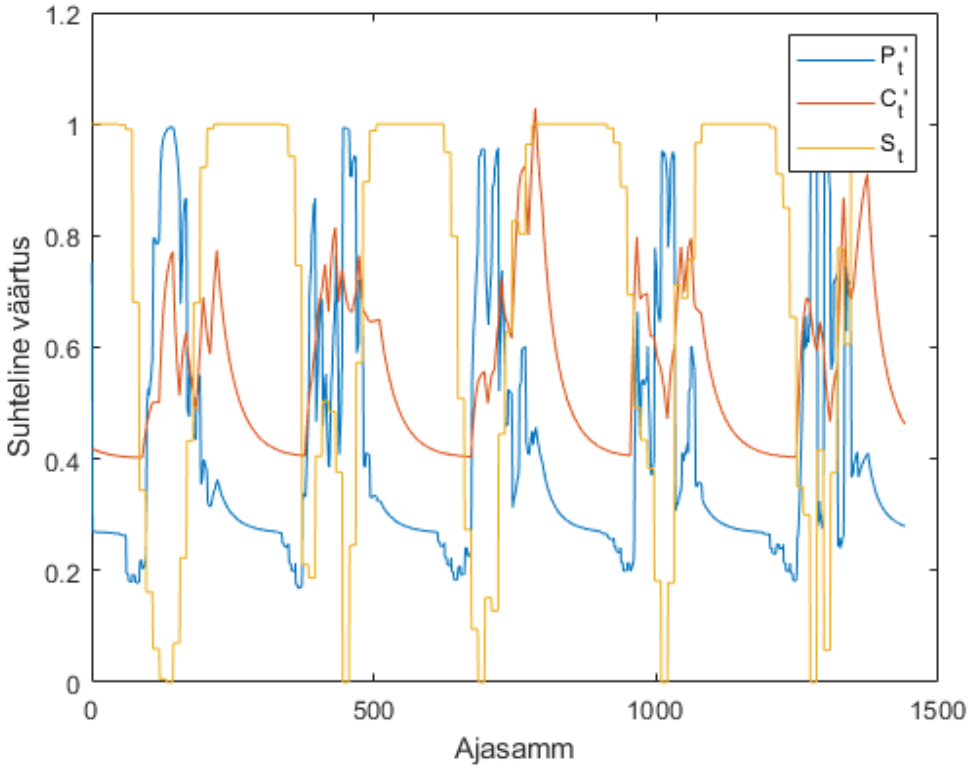
Kui süsteem sisaldab märkimisväärses koguses mittejuhitavat tootmist, siis võib mõistlik olla ka juhtida tarbimist vastava tehnoloogia tootmiskõvera järgi. Hea näide on päikesepaneelid, mille tootmiskõver on karakteerse kujuga ning pikas plaanis üsna etteaimatav. Kuna paneelide tootlikkus on seotud päikese asukohaga, siis on see suurim keskpäeval, mis ei lange kokku tarbimise tipuga. Süsteemi efektiivsuse vaatest on ideaalne, kui toodetud energia tarbitakse lähedal asuva tarbija poolt ära. Kuid kuna paneelide tootlikkus üldjuhul tarbimisega ei kattu, siis jääb sedasi toodetud elektrienergiat tihti peale piltlikult öeldes üle. Seega oleks kasulik, kui süsteem sisaldaks tarbijaid, mis suudaks koormust ajastada kokku päikesepaneelide tootlikkusega.

Sarnaselt eelmise alapeatükiga on ka siinjuhul stiimulõppe algoritmi jaoks kasutatud hinnast erinevat sisendit. Seekord on kasutatud tüüpilise päikeseenergial põhineva seadme päevase tootmiskõvera pöördväärtust, et luua algoritmile sobiv signaal vahemikus 0 … 1. Kasutatud suurused on esitatud joonisel 4.7. Tootmiskõver on genereeritud PVGIS tööriista abil [51] Eesti asukoha ning optimeeritud päikesepaneelide paigutuse alusel.



Joonis 4.7 Päikesepaneelide tüüpiline päevane tootmiskõver ja sellega pöördvõrdeline signaal

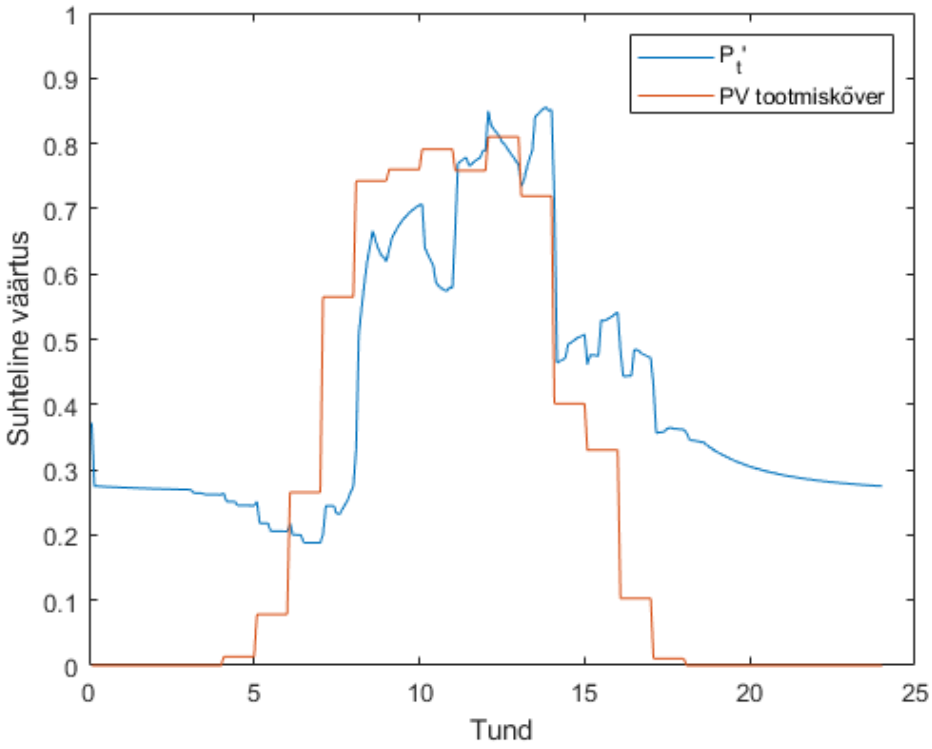
Tootmiskõvera pöörväärtust kasutamine tähendab, et suurus 1 vastab ajale, mil päikesepaneeli tootlikus on 0 ning algoritmi jaoks on eelistatum võimsuse minimeerimine. See sarnaneb algoritmi treenimisel kasutatud hinnasignaalile, kus väärtus 1 tähendab maksimaalset hinda. Analoogne signaal loodi viie järjestikuse päeva jaoks ning seda kasutati sisendina ventilatsiooniseadme simuleerimiseks viiel ööpäeval, eesmärgiga nihutada tarbimine vastavusse päikesepaneelide tootmiskõveraga. Simulatsiooni väljund on esitatud joonisel 4.8.



Joonis 4.8 Simulatsiooni väljundgraafik PV tootmisel põhineva signaaliga viiel ööpäeval

Antud graafikult on võimalik näha, et tarbimine kattub suurel määral madala signaaliga. See tähendab, et elektrienergiat tarbitakse rohkem sel ajal, millal päikesepaneelid seda toodavad. Tarbimise nihutamine toimub CO_2 taseme arvelt, mis tõuseb tööpäeva lõpus üsna kõrgele, kuna sel ajal on päikesepaneelide tootlikkus juba madal. Kuid välja arvatud ühel ajasammul, on CO_2 tase tugevalt piirides ning optimaalse juhtimise jaoks oleks soovitatav veelgi agressiivsemalt hinnasignaalile reageerida ning päikese loojudes ventilatsiooni võimsust alandada. Keskmine ööpäevane kogutarbimine antud signaali kasutamise tõttu võrreldes hinnapõhise juhtimisega praktiliselt ei muutunud.

Kattuvuse paremaks hindamiseks on arvutatud ventilatsiooni võimsuse viie päeva keskmine ning joonisel 4.9 on seda võrreldud sama perioodi keskmise päikesepaneelide tootmiskõveraga.



Joonis 4.9 Ventilatsiooni tarbimiskõvera võrdlus PV tootmiskõveraga

Viimaselt graafikult on näha, et viie päeva keskmist arvestades langeb ventilatsiooniseadme võimsus üsna hästi kokku päikesepaneeli tootmiskõveraga. Vaid tööpäeva lõppedes ei ole ventilaatori võimsust võimalik piisavalt kiiresti langetada, kuna sel ajal veel ruumi kasutatakse ning CO_2 tase on kõrge.

Tuleb ka märkida, et käesolevas peatükis kasutatud stiimulõppe mudel on treenitud elektrienergia hinna alusel, mistõttu ei ole see optimeeritud süsteemi koormuskõverast või päikesepaneelide tootmiskõverast tuletatud signaali kasutama. Kuid siiski on saavutatud tulemused head, mis demonstreerib masinõppe meetodite paindlikkust. Seetõttu on realistlik ka eeldada, et erinevate paindlikkusteenuste jaoks treenitud mudelid on üsna sarnased ning vastavalt süsteemi vajadustele on võimalik juhtimisstrateegia eesmärki kiiresti kohandada.

KOKKUVÕTE

Taastuvenergeetika kasutuselevõtuga kaasnevad tingimata ka väljakutsed elektrienergia tootmise juhtimises ning seetõttu tõuseb vajadus juhitava koormuse järele. Märkimisväärse osa energiast Euroopa Liidus tarbivad hooned, millest samuti suur osa on ventilatsioonisüsteemide arvelt. Kuid ventilatsioonisüsteem ei pea töötama pidevalt konstantsel võimsusel, vaid seda saab reguleerida keskkonnaparameetrite ning hinna järgi. Seetõttu on ventilatsioonisüsteemide intelligentsel juhtimisel suur potentsiaal tarbimise juhtimises.

Elektrienergiaga kauplemine põhineb vabal turul, mistõttu määratakse hind pakkumise ja nõudluse kaudu. See sisaldab ka mittejuhitavast tootmisest tulenevat pakkumist, mis võib näiteks päikesepaneelide suure tootlikkuse tõttu hea ilma korral elektrienergia hinna nullilähedaseks viia. Juhtides tarbimist hinna põhjal, on võimalik turu liikumisi ära kasutada ning ka turgu positiivselt mõjutada. See omakorda parandab elektrisüsteemi efektiivsust ning aitab vähendada elektrienergia tootmisega kaasnevat negatiivset mõju keskkonnale.

Masinõppe meetodid on tänapäeval populaarne võimalus intelligentsete süsteemide saavutamiseks. Automaatse treenimisprotsesside kaudu saab luua kordades keerulisema ja võimekama algoritmi, kui oleks võimalik manuaalselt koostada. Stiimulõppe abil on saanud reaalsuseks juhtimissüsteemid, mis on fundamentaalselt intelligentsed ning käitumiselt inimlikud. Tänu moodsate arvutite tohutule arvutusjõudlusele ning programmeerimiskeelte võimekusele on masinõppe rakendamine reaalsetes süsteemides jõukohane paljudele. Masinõppe meetodid on ka lihtsamini skaleeritavad, kuna süsteemi laiendamiseks on vaid vaja muuta keskkonna parameetreid ning treenimisalgoritm suudab üldjuhul minimaalsete muutustega leida uue süsteemi jaoks optimaalse lahenduse.

Käesolevas töös loodi stiimulõppel põhinev juhtimisstrateegia, mis suutis ruumi ventilatsioonisüsteemi juhtida efektiivsemalt ja säästlikumalt kui käsitsi loodud juhtimismeetod. Lisaks demonstreeriti nii ventilatsiooniseadme potentsiaali juhitava koormusena kui ka masinõppe kasutamise võimalusi energeetikas. Juhtimisstrateegiat testiti ventilatsiooni demoruumis, kus saavutati 58,2% väiksem elektrienergia maksumus ning madalam keskmine siseõhu CO_2 tase. Töös kirjeldati ka võimalusi lihtsasti erineva käitumisega mudelite loomiseks, mis suudavad asetada rohkem või vähem rõhku elektrienergia maksumusele ning seega pakkuda spetsiifilisi lahendusi.

Töö edasiarenduseks oleks mõistlik algoritmile lisada võimekus temperatuuri ja õhuniiskust jälgida ning juhtida. Sel juhul oleks võimalik terve hoone ventilatsioonisüsteemi juhtimine täielikult mudeli kätte anda. Tulenevalt masinõppe meetodist, nõuaks see vaid üsna triviaalset sisend- ja väljundparameetrite muutmist, kuna sama meetod suudaks ka selle mudeli valmis treenida. Lisaks saaks meetodi skaleeritavust demonstreerida suuremas keskkonnas, mis sisaldaks mitmeid ruume, ventilatsiooniseadmeid ning mõõteandmeid. Viimaks on võimalik testida antud meetodit erinevate koormuse juhtimise variantidega.

SUMMARY

The increasing application of renewable energy brings with it challenges in energy supply management and an increased need for demand side management. A significant part of energy in the EU is consumed by buildings and a large part of that by HVAC systems. But a ventilation system does not have to work at a constant power. Instead, it can be regulated based on environmental parameters and the current price of electricity. This gives the intelligent control of ventilation systems great potential for demand response.

The trading of electrical energy is based on the free market which means that the price is decided by supply and demand. This includes the supply caused by non-controllable sources, which for example could result in near zero electricity prices due to high supply from solar panels in sunny weather. By basing the demand of electricity on the price, the movements of the market can be taken advantage of and the market itself positively influenced. This would increase the effectiveness of the grid and help reduce harmful effects on the environment.

Machine learning methods are popular ways to achieve intelligent systems. Through automatic training processes it is possible to create a much more complicated and effective control algorithm than would be possible by manual programming. Reinforcement learning has made it possible to create control systems which act in a fundamentally intelligent and human way. Thanks to the huge capabilities of modern computers and programming languages machine learning methods have become accessible to many. Machine learning algorithms are more easily scalable as only changing the environmental parameters is required to expand the model and the core training algorithm can be kept mostly unchanged.

This thesis involved the creation of a reinforcement learning based control strategy which was able to control the ventilation system more effectively and cheaper than a manually programmed algorithm. In addition, the potential of ventilation systems in demand response applications was demonstrated. The control algorithm was tested in a ventilation demo room, where a 58,2% price decrease was achieved while keeping the average CO_2 levels lower. Also, the method for easily creating control strategies with different goals and behaviors was explained.

The work could be continued by adding temperature and humidity control capabilities to the model. In that case the model could control the entire HVAC system of a building. Due to the attributes of machine learning algorithms, this could be implemented fairly trivially by mostly adding the necessary inputs and outputs to the model, since the same method could be used for training. Furthermore, the scalability of the method could be demonstrated in a larger environment which would include many rooms, HVAC devices and measurement data. Finally, the method could be tested with different forms of demand side management.

KASUTATUD KIRJANDUSE LOETELU

[1] „European Comission,“ [Võrgumaterjal]. Available: https://ec.europa.eu/info/energy-climate-change-environment/standards-tools-and-labels/products-labelling-rules-and-requirements/energy-label-and-ecodesign/energy-efficient-products/ventilation-units_en. [Kasutatud 10. 03. 2021].

[2] L. Gelazanskas ja K. A. Gamage, „Demand side management in smart grid: A review and proposals for future direction,“ *Sustainable Cities and Society,* kd. 11, pp. 22-30, 2014.

[3] D. O'Neill, M. Levorato, A. Goldsmith ja U. Mitra, „Residential Demand Response Using Reinforcement Learning,“ %1 *2010 First IEEE International Conference on Smart Grid Communications*, Gaithersburg, MD, USA, 2010.

[4] „Energy.Gov,“ [Võrgumaterjal]. Available: https://www.energy.gov/oe/activities/technology-development/grid-modernization-and-smart-grid/demand-response. [Kasutatud 10. 03. 2021].

[5] „Energy.Gov,“ [Võrgumaterjal]. Available: https://www.energy.gov/oe/services/electricity-policy-coordination-and-implementation/state-and-regional-policy-assistanc-4. [Kasutatud 12 03. 2021].

[6] S. Tiptipakorn ja W.-J. Lee, „A Residential Consumer-Centered Load Control Strategy in Real-Time Electricity Pricing Environment,“ %1 *39th North American Power Symposium*, Las Cruces, NM, USA, 2007.

[7] E. Abel ja H. Voll, Hoonete energiatarve ja sisekliima, Tallinn: OÜ Presshouse, 2007.

[8] M. G. Apte, W. J. Fisk ja J. M. Daisey, „Associations Between Indoor CO2 Concentrations and Sick Building Syndrome Symptoms in U.S. Office Buildings: An Analysis of the 1994-1996 BASE Study Data,“ *Indoor Air,* kd. 10, pp. 246-257, 2000.

[9] V. Maask, T. Häring, R. Ahmadiahangar, A. Rosin ja T. Korõtko, „Analysis of Ventilation Load Flexibility Depending on Indoor Climate Conditions,“ %1 *IEEE International Conference on Industrial Technology*, Buenos Aires, Argentina, 2020.

[10] *EVS-EN 16798-1:2019.*

[11] A. Krijgsman, Artificial Intelligence in Real-Time Control, 1993.

[12] S. Theodoridis, Machine Learning A Bayesian and Optimization Perspective, London: Elsevier Ltd, 2015.

[13] K. P. Murphy, Machine Learning A Probabilistic Perspective, Cambridge, Massachusetts: The MIT Press, 2012.

[14] D. Sturzenegger, D. Gyalistras, M. Morari ja R. S. Smith, „Model Predictive Climate Control of a Swiss Office Building: Implementation, Results, and Cost–Benefit Analysis,“ *IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY,* kd. 24, nr 1, 2016.

[15] F. Ruelens, S. Iacovella, B. J. Claessens ja R. Belmans, „Learning Agent for a Heat-Pump ThermostatWith a Set-Back Strategy Using Model-Free Reinforcement Learning,“ *Energies,* kd. 8, pp. 2300-2318, 2015.

[16] A. Mastropietro, F. Castiglione, S. Ballesio ja E. Farizio, „Reinforcement Learning Control Algorithm for HVAC Retrofitting: Application to a Supermarket Building Model by Dynamic Simulation,“ %1 *Building Simulation*, 2019.

[17] A. Zielesny, From Curve Fitting to Machine Learning, Springer, 2011.

[18] R. S. Sutton ja A. G. Barto, Reinforcement Learning An Introduction, Cambridge, MA: The MIT Press, 2018.

[19] D. Ormoneit ja Š. Sen, „Kernel-Based Reinforcement Learning,“ *Machine Learning,* kd. 49, pp. 161-178, 2002.

[20] S. Heo, K. Nam, J. Loy-Benitez, Q. Li, S. Lee ja C. Yoo, „A deep reinforcement learning-based autonomous ventilation control system for smart indoor air quality management in a subway station,“ *Energy and Buildings,* kd. 202, 2019.

[21] T. Wei, Y. Wang ja Q. Zhu, „Deep Reinforcement Learning for Building HVAC Control,“ *54th ACM/EDAC/IEEE Design Automation Conference,* 2017.

[22] „Mathworks,“ [Võrgumaterjal]. Available: https://www.mathworks.com/help/reinforcement-learning/ug/create-agents-for-reinforcement-learning.html. [Kasutatud 14. 03. 2021].

[23] E. Barret ja S. L. Linder, „Autonomous HVAC Control, A Reinforcement Learning Approach,“ %1 *ECML*, Porto, 2015.

[24] M. Kyushik, K. Hayoung ja H. Kunsoo, „Deep Q Learning Based High Level Driving Policy Determination,“ %1 *IEEE Intelligent Vehicles Symposium*, 2018.

[25] R. Rojas, Neural Networks A Systematic Introduction, Springer, 1996.

[26] V. Francois-Lavet, R. Islam, J. Pineau, P. Henderson ja M. G. Bellemare, „An Introduction to Deep Reinforcement Learning,“ *Foundations and Trends in Machine Learning,* kd. 11, nr 3-4, 2018.

[27] J. Patterson ja A. Gibson, Deep Learning A Practicioner's Approach, Sebastopol, CA: O'Reilly Media, 2017.

[28] F. Chollet, Deep Learning With Python, Manning Publications Co., 2018.

[29] D. Pandey ja P. Pandey, „Approximate Q-Learning: An Introduction,“ %1 *Second International Conference on Machine Learning and Computing*, 2010.

[30] C. Shyalika, „towards data science,“ 2019. [Võrgumaterjal]. Available: https://towardsdatascience.com/a-beginners-guide-to-q-learning-c3e2a30a653c. [Kasutatud 09 03 2021].

[31] „OpenAI Spinning Up,“ [Võrgumaterjal]. Available: https://spinningup.openai.com/en/latest/algorithms/ddpg.html. [Kasutatud 11 03 2021].

[32] C. Qiu, Y. Hu, Y. Chen ja B. Zeng, „Deep Deterministic Policy Gradient (DDPG)-Based Energy Harvesting Wireless Communications,“ *IEEE INTERNET OF THINGS JOURNAL,* kd. 6, nr 5, 2019.

[33] J. Duan, D. Shi, R. Diao, H. Li, Z. Wang, B. Zhang, D. Bian ja Z. Yi, „Deep-Reinforcement-Learning-Based Autonomous Voltage Control for Power Grid Operations,“ *IEEE TRANSACTIONS ON POWER SYSTEMS,* kd. 35, nr 1, 2020.

[34] „Nord Pool,“ [Võrgumaterjal]. Available: https://www.nordpoolgroup.com/Market-data1/#/nordic/table. [Kasutatud 11. 03. 2020].

[35] „Occupancysimulator,“ [Võrgumaterjal]. Available: http://occupancysimulator.lbl.gov/. [Kasutatud 13. 03. 2021].

[36] S. Rotger.Griful, R. Jacobsen, D. Nguyen ja G. Sørensen, „Demand response potential of ventilation systemsin residential buildings,“ *Energy and Buildings,* kd. 121, pp. 1-10, 2016.

[37] A. Chatterjee, L. Zhang ja X. Xia, „Optimization of mine ventilation fan speeds according to ventilation on demand and time of use tariff,“ *Applied Energy,* kd. 146, pp. 65-73, 2015.

[38] T. Fuchida, K. T. Aung ja A. Sakuragi, „A study of Q-learning considering negative rewards,“ *Artificial Life and Robotics,* kd. 15, pp. 351-354, 2010.

[39] „towardsdatascience,“ [Võrgumaterjal]. Available: https://towardsdatascience.com/hyper-parameter-tuning-techniques-in-deep-learning-4dad592c63c8. [Kasutatud 11 03 2021].

[40] „Systemair,“ [Võrgumaterjal]. Available: https://shop.systemair.com/et-EE/save--vtr--250b--l--1000w/p403427. [Kasutatud 11. 03. 2021].

[41] „Winsen-Sensor,“ [Võrgumaterjal]. Available: https://www.winsen-sensor.com/d/files/infrared-gas-sensor/mh-z19b-co2-ver1_0.pdf. [Kasutatud 08 05 2021].

[42] „CO2meter,“ [Võrgumaterjal]. Available: https://www.co2meter.com/blogs/news/6056206-co2-sensor-location-where-to-mount-your-co2-iaq-monitor. [Kasutatud 13 05 2021].

[43] K. Khetarpal, M. Riemer, I. Rish ja D. Precup, „Towards Continual Reinforcement Learning: A Review and Perspectives“.

[44] J. Cigler, Z. Váňa, T. Mužík, J. Šulc ja L. Ferkl, „Usage of spot market prices prediction for demand side management,“ %1 *European Control Conference*, Aalborg, 2016.

[45] D. Zhang, S. Li, M. Sun ja Z. O'Neill, „An Optimal and Learning-Based Demand Response,“ *IEEE Transactions On Smart Grid,* kd. 7, nr 4, pp. 1790-1800, 2016.

[46] J. Wang, S. Huang, D. Wu ja N. Lu, „Operating a Commercial Building HVAC Load as a Virtual Battery through Airflow Control,“ *IEEE Transactions on Sustainable Energy,* 2020.

[47] A. Lösing, S. Baum ja I. Stadler, „Using Ventilation Systems As A Demand Response Technology,“ %1 *International Energy and Sustainability Conference*, Cologne, 2016.

[48] „Kane,“ [Võrgumaterjal]. Available: https://www.kane.co.uk/knowledge-centre/what-are-safe-levels-of-co-and-co2-in-rooms. [Kasutatud 03 06 2021].

[49] „Elering,“ [Võrgumaterjal]. Available: https://elering.ee/elektri-tarbimine-ja-tootmine. [Kasutatud 03 06 2021].

[50] „Nordpoolgroup,“ [Võrgumaterjal]. Available: https://www.nordpoolgroup.com/Market-data1/Power-system-data/Consumption1/Consumption/EE/Hourly/?view=table. [Kasutatud 04 06 2021].

[51] „EU SCIENCE HUB,“ [Võrgumaterjal]. Available: https://ec.europa.eu/jrc/en/pvgis. [Kasutatud 04 06 2021].

LISAD

```
% Keskkonna parameetrite määramine
KK.Ts = 5; %Ajasammu pikkus minutites
KK.WC = 0.8; %CO2 kaal
KK.WE = 2; %Elektrienergia hinna kaal
KK.CO = 400; %Väline CO2 [ppm]
KK.Cmax = 1000; %Maksimaalne lubatud CO2 kontsentratsioon [ppm]
KK.CG = 800; %Soovitud CO2 kontsentratsioon [ppm]
KK.dVmax = 230/(60/KK.Ts); %Ventileeritava õhu hulk maksimaalsel
võimsusel ajaühikus [m^3]
KK.V = 193; %Ruumi suurus [m^3]
KK.dVmaxN = KK.dVmax/KK.V; % Osa ruumi õhust, mis ventilaator suudab
vahetada ajaühikus maksimaalsel võimsusel
KK.CO2_Gen = 20/((60/KK.Ts)*1000); %Ühe inimese poolt toodetava CO2 kogus
[m^3/Ts]
KK.PPM = (10^6*(KK.CO2_Gen/KK.V))/(KK.Cmax); %CO2 ppm suhteline tõus
ruumis inimese kohta
KK.Pmax = 172; %Ventilaatori maksimaalne võimsus [W]
KK.Pmin = 10; %Ventilaatori minimaalne võimsus [W]
HinnadTund= importdata("2020Hourly.mat"); %Tunnip?hised hinnad [€/MWh]
KK.Inimesed = importdata("Occupancy4.xlsx"); %Inimeste hulk ruumis igal
ajasammul
% Hindade normaliseerimine
HinnadTemp = zeros(length(HinnadTund),1);
for i = 1:length(HinnadTund)/24
    pmin = min(HinnadTund(i*24-23:i*24));
    for j = 1:24
        HinnadTemp(j+(i-1)*24) = HinnadTund(j+(i-1)*24) - pmin;
        if HinnadTemp(j+(i-1)*24) == 0
            HinnadTemp(j+(i-1)*24) = 0.05; %0 hind võib segadusse ajada
        end
    end
    pmax = max(HinnadTemp(i*24-23:i*24));
    for j = 1:24
        HinnadTemp(j+(i-1)*24) = HinnadTemp(j+(i-1)*24) / pmax;
    end
end
% Hindade teisendamine ajasammu põhiseks
KK.Hinnad = zeros(length(HinnadTund)*60/KK.Ts,1);
KK.HinnadReal = zeros(length(HinnadTund)*60/KK.Ts,1);
for i = 1:length(HinnadTemp)
    for j = 1: 60/KK.Ts
        KK.Hinnad(j+(i-1)*60/KK.Ts) = HinnadTemp(i);
        KK.HinnadReal(j+(i-1)*60/KK.Ts) = HinnadTund(i)/1000000; % [€/Wh]
    end
end
% Treenimise pikkuse määramine
doTraining = 0;
MaxEp = 50000;
MaxStep = 1*24*60/KK.Ts; %Ühe episoodi pikkus

% Mudeli parameetrite määramine
ObservationInfo = rlNumericSpec([6 1],'LowerLimit',0','UpperLimit',1');
ObservationInfo.Name = 'Measurements';
ObservationInfo.Description = 'CO2, Et, Et+1, Et+3, Et+8, Et+12';

ActionInfo = rlNumericSpec([1 1],'LowerLimit',0,'UpperLimit',1);
ActionInfo.Name = 'Action';
numActions = ActionInfo.Dimension(1);
```

```
ResetHandle = @myResetFunction;
StepHandle = @(Action,LoggedSignals)
myStepFunction(Action,LoggedSignals,KK);

env = rlFunctionEnv(ObservationInfo,ActionInfo,StepHandle,ResetHandle);
numObservations = ObservationInfo.Dimension(1);
numActions = numel(ActionInfo);

% Kriitiku võrgu loomine
statePath = [

featureInputLayer(numObservations,'Normalization','none','Name','Olek')
    fullyConnectedLayer(50,'Name','Olek_peidetud_1')
    reluLayer('Name','Olek_Relu')
    fullyConnectedLayer(25,'Name','Olek_peidetud_2')];
actionPath = [
    featureInputLayer(numActions,'Normalization','none','Name','Tegevus')
    fullyConnectedLayer(25,'Name','Tegevus_peidetud')];
commonPath = [
    additionLayer(2,'Name','add')
    reluLayer('Name','Uhine_ReLu')
    fullyConnectedLayer(1,'Name','Kriitik_valjund')];

criticNetwork = layerGraph(statePath);
criticNetwork = addLayers(criticNetwork,actionPath);
criticNetwork = addLayers(criticNetwork,commonPath);
criticNetwork = connectLayers(criticNetwork,'Olek_peidetud_2','add/in1');
criticNetwork =
connectLayers(criticNetwork,'Tegevus_peidetud','add/in2');
criticOpts = rlRepresentationOptions('LearnRate',1e-
03,'GradientThreshold',1);

critic = rlQValueRepresentation(...
            'net',criticNetwork,...
            'obsname',ObservationInfo,...
            'actName',ActionInfo,...
            'Observation',{'Olek'},...
            'Action',{'Tegevus'});

% Täitja võrgu loomine
actorNetwork = [

featureInputLayer(numObservations,'Normalization','none','Name','Olek')
        fullyConnectedLayer(3, 'Name','Peidetud')
        reluLayer('Name','Relu')
        fullyConnectedLayer(numActions,'Name','Tegevus')
        sigmoidLayer('Name', 'Sigmoid')
        scalingLayer('Scale',0.84,'Bias',0.16,'Name','Skaleeriv_valjund')
        ];
actorOpts = rlRepresentationOptions('LearnRate',1e-
03,'GradientThreshold',1);
actor =
rlDeterministicActorRepresentation(actorNetwork,ObservationInfo,ActionInf
o,'Observation',{'Olek'},'Action',{'Skaleeriv_valjund'},actorOpts);
% DDPG agendi loomine
agentOpts = rlDDPGAgentOptions(...
    'SampleTime',1,...
    'TargetSmoothFactor',1e-3,...
    'ExperienceBufferLength',1e6,...
```

```
    'DiscountFactor',1,...
    'NumStepsToLookAhead',32,...
    'MiniBatchSize',64);
agentOpts.NoiseOptions.StandardDeviation = 0.12;
agentOpts.NoiseOptions.StandardDeviationDecayRate = 1e-5;
agent = rlDDPGAgent(actor,critic,agentOpts);
% Treenimise seaded
if doTraining
trainOpts = rlTrainingOptions(...
    'MaxEpisodes',MaxEp, ...
    'MaxStepsPerEpisode',MaxStep, ...
    'ScoreAveragingWindowLength',20, ...
    'Verbose',false, ...
    'Plots','training-progress',...
    'StopTrainingCriteria','EpisodeReward',...
    'SaveAgentCriteria','EpisodeReward',...
    'SaveAgentValue',0,...
    'StopTrainingValue',0);
    % Treenimiskäsk
    trainingStats = train(agent,env,trainOpts);
else
    % Kui treenitud agent on juba olemas, siis impordime
    agent = importdata("TrainedAgent.mat");
end
```

Lisa 2 Stiimulõppe agendi sammfunktsioon

```
function [NextObs,Reward,IsDone,LoggedSignals] = myStepFunction(Action,
LoggedSignals,envConstants)
% Sammfunktsioon
persistent Step
if isempty(Step)
    Step = 1;
else
    Step = Step + 1;
end
CO2 = LoggedSignals.State(1);
Et = KK.Hinnad(Step);
Et1 = KK.Hinnad(Step+1);
Et3 = KK.Hinnad(Step+3);
Et8 = KK.Hinnad(Step+8);
Et12 = KK.Hinnad(Step+12);
Inimesed = KK.Inimesed(Step);

CO2 = (1-Action*KK.dVmaxN)*(CO2 + Inimesed *
KK.PPM)+Action*KK.dVmaxN*KK.CO/KK.Cmax; % Uus suhteline CO2
kontsentratsioon
S = ((KK.Pmax-KK.Pmin)*Action^3+KK.Pmin)/KK.Pmax; % Tarbitud suhteline
võimsus
E = Et*S;% Tarbitud elektrienergia suhteline maksumus

LoggedSignals.State(1) = CO2;
LoggedSignals.State(2) = Et;
LoggedSignals.State(3) = Et1;
LoggedSignals.State(4) = Et3;
LoggedSignals.State(5) = Et8;
LoggedSignals.State(6) = Et12;
NextObs = LoggedSignals.State;

if (CO2 <=0.96)
    Reward = -(KK.WC/(1+exp(20*(KK.CG/KK.Cmax-CO2)))) -
(KK.WE/(1+exp(15*(0.3-E))));
else
    Reward = -CO2^4 - (KK.WE/(1+exp(15*(0.3-E))));
end
```

Lisa 3 Stiimulõppe agendi algfunktsioon

```
function [InitialObservation,LoggedSignal] = myResetFunction()
% Funktsioon algseisundi rakendamiseks
CO2 = 1;
Et = 0;
Et1 = 0;
Et3 = 0;
Et8 = 0;
Et12 = 0;
clear myStepFunction;
LoggedSignal.State = [CO2; Et; Et1; Et3; Et8; Et12];
InitialObservation = LoggedSignal.State;
end
```

```
% Andmete sisestamine
KK.Ts = 5; %Ajasammu pikkus minutites
HinnadTund= importdata("2021AprilHourly.mat"); %Tunnipõhised hinnad
[€/MWh]
% Hindade normaliseerimine
for i = 1:length(HinnadTund)/24
    pmin = min(HinnadTund(i*24-23:i*24));
    for j = 1:24
        HinnadTemp(j+(i-1)*24) = HinnadTund(j+(i-1)*24) - pmin;
        if HinnadTemp(j+(i-1)*24) == 0
            HinnadTemp(j+(i-1)*24) = 0.05; %0 hind võib segadusse ajada
        end
    end
    pmax = max(HinnadTemp(i*24-23:i*24));
    for j = 1:24
        HinnadTemp(j+(i-1)*24) = HinnadTemp(j+(i-1)*24) / pmax;
    end
end
% Hindade teisendamine ajasammu põhiseks
for i = 1:length(HinnadTemp)
    for j = 1: 60/KK.Ts
        KK.Hinnad(j+(i-1)*60/KK.Ts) = HinnadTemp(i);
        KK.HinnadReal(j+(i-1)*60/KK.Ts) = HinnadTund(i)/1000000; %[€/Wh]
    end
end

KK.Inimised = importdata("Occupancy4.xlsx");
agent = importdata("TrainedAgent.mat");
% Põhiprogramm
clear Ard
m = modbus('tcpip','192.168.1.3',502); %Ventilatsiooniseadmega ühenduse
loomine
Tmax = 3000; %Katse maksimaalne kestvus minutites
logi = string(0); h = 0;
filename = 'logifail.xlsx';
Ard = serialport("COM4",9600); %Arduinoga ühenduse loomine
for i = 1:Tmax
    c = clock; t1 = c(6);  %Kella sekundi väärtus tsükli alguses

    try %CO2 väärtus mõõteseadmest
        conn = database('Nimi','DBuser','SiseKliima');
        sqlquery = ['SELECT CO2 FROM andmed WHERE Nr=(SELECT MAX(Nr) FROM
andmed)'];
        CO2ppm = fetch(conn,sqlquery);
        CO2 = CO2ppm{:,:}/1000;
    catch
        %CO2 väärtust ei õnnestunud lugeda
    end

    %Doseerimise kogus balloonile
    Inimesed = KK.Inimesed(i);
    Cdoos = Inimesed * 33; %CO2 doseerimine (cL/min)
    doos = int2str(Cdoos);
    try
        write(Ard,doos,'string')
    catch
        %Ballooni kontrollerile ei õnnestunud signaali saata
    end
```

```
    E0 = KK.Hinnad(i); E1 = KK.Hinnad(i+1); E3 = KK.Hinnad(i+3); E8 =
KK.Hinnad(i+8); E12 = KK.Hinnad(i+12);
    obs = [CO2, E0, E1, E3, E8, E12]; %Vaatlusandmed [CO2, E, E+1, E+3,
E+8, E+12]
    aaction = getAction(agent,obs); %Tegevuse leidmine [ventilaatori
kiirus 0.16 - 1.0]
    Action = aaction{:,:}; %Matlabi tüütustega tegelemine
    if CO2 >= 0.96 & Action < 0.85
        Action = 1;
    end
    Action = double(uint16(round(Action,2)*100));

    %Ventilaatorile käsu saatmine
    try
        write(m,'holdingregs',13801,Action); %Sissepuhke kiirus
        write(m,'holdingregs',13802,Action); %Väljapuhke kiirus
    catch
        %Käsu saatmine ei õnnestunud
    end

    aeg = '%d:%d:%d'; aeg = sprintf(aeg,c(4),c(5),round(c(6)));
%Kellaaeg ühe stringina
    logi(i,1) = string(aeg); logi(i,2:7) = obs; logi(i,8) = Cdoos;
logi(i,9) = Action(1); logi(i,10) = vent;
    writematrix(logi(i,:),filename,'Sheet',1,'WriteMode','append')
    if mod(i,60) == 0 %Igal täistunnil luuakse eraldi logi selle tunni
kohta
        h = h + 1;
        formatSpec = 'backuplogi_%d.xlsx';
        filename2 = sprintf(formatSpec,h);
        writematrix(logi((i-
59):i,:),filename2,'Sheet',1,'WriteMode','append')
    end

    c = clock; t2 = c(6);
    At = t2 - t1; %Tsüklile kulunud aja leidmine
    pause(60-At); %Ooteaja lisamine, et igale tsüklile kuluks täpselt 1
minut
end
```