

**T.C.
ISTANBUL OKAN UNIVERSITY
GRADUATE SCHOOL**

**THESIS FOR THE DEGREE OF MASTER OF
AUTOMOTIVE MECHATRONICS
AND INTELLIGENT VEHICLES PROGRAM**

Mete YAVUZ

**MULTI-AGENT REINFORCEMENT LEARNING
BASED ENERGY MANAGEMENT
WITH P2P/V2G**

THESIS ADVISOR
Assoc. Prof. Dr. Ömer Cihan KIVANÇ

ISTANBUL, August 2023

**T.C.
ISTANBUL OKAN UNIVERSITY
GRADUATE SCHOOL**

**THESIS FOR THE DEGREE OF MASTER OF
AUTOMOTIVE MECHATRONICS
AND INTELLIGENT VEHICLES PROGRAM**

**Mete YAVUZ
203005009**

**MULTI-AGENT REINFORCEMENT LEARNING
BASED ENERGY MANAGEMENT
WITH P2P/V2G**

Submission Date of Thesis : 01.08.2023

Presentation Date of Thesis : 24.08.2023

Advisor : Assoc. Prof. Dr. Ömer Cihan KIVANÇ_____

Jury Members : Prof. Dr. Ramazan Nejat TUNCAY_____

Assoc. Prof. Dr. Salih Barış ÖZTÜRK_____

ISTANBUL, August 2023

**T.C.
ISTANBUL OKAN UNIVERSITY
GRADUATE SCHOOL**

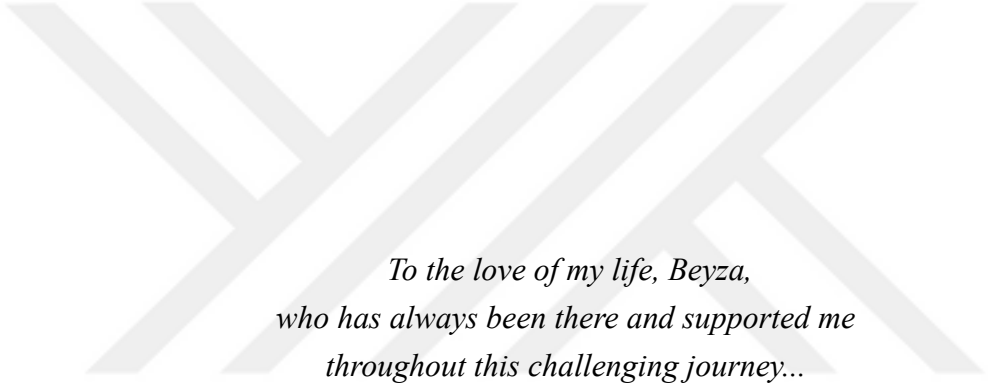
**THESIS FOR THE DEGREE OF MASTER OF
AUTOMOTIVE MECHATRONICS
AND INTELLIGENT VEHICLES PROGRAM**

**Mete YAVUZ
203005009**

**MULTI-AGENT REINFORCEMENT LEARNING
BASED ENERGY MANAGEMENT
WITH P2P/V2G**

THESIS ADVISOR
Assoc. Prof. Dr. Ömer Cihan KIVANÇ

ISTANBUL, August 2023



*To the love of my life, Beyza,
who has always been there and supported me
throughout this challenging journey...*

ACKNOWLEDGMENT

I would like to express my deep gratitude to all those who have supported me throughout the course of this research and the completion of this master's thesis.

First and foremost, I am deeply thankful to my advisor, Assoc. Prof. Dr. Ömer Cihan KIVANÇ, for his guidance, encouragement, and help during the entire project. Furthermore, I would like to thank Prof. Dr. Ramazan Nejat TUNCAY for taking his time to help and guide throughout my entire academic journey. A special gratitude should go out to all Engineering Faculty members whom allowed me to use necessary laboratories any time I need and providing me the necessary equipment.

Furthermore, I would like to thank Beyza Nur HACIOGLU, my love, my mother, my grandmother and especially our cat for not leaving me alone on this challenging journey.

Last but not least, so as to never forget, I dedicate this study to everyone who lost their lives and their loved ones in the earthquake that shook us deeply whose epicenter was Kahramanmaraş, Türkiye on February 6, 2023.

TABLE OF CONTENTS

1	INTRODUCTION	1
1.1	AIMS OF THE STUDY	4
1.2	LIMITATIONS OF THE STUDY	4
1.3	STRUCTURE & OUTLINE OF THE THESIS	5
2	ESS, EV, and P2P ENERGY TRADING in SMART MICRO GRIDS	6
2.1	ENERGY STORAGE SYSTEMS	6
2.1.1	Battery Energy Storage Systems (BESS)	6
2.1.1.1	Lithium-Ion Batteries	7
2.1.1.2	Lead-Acid Batteries	8
2.1.1.3	Battery Management System (BMS)	9
2.1.2	Supercapacitor Energy Storage Systems (SESS)	9
2.2	ELECTRIC VEHICLES and V2G	10
2.2.1	Classification of Electric Vehicle (EV)s	11
2.2.1.1	Battery Electric Vehicle (BEV)	11
2.2.1.2	Hybrid Electric Vehicle (HEV)	12
2.2.1.3	Fuel Cell Electric Vehicle (FCEV)	15
2.2.1.4	Plug-in Hybrid Electric Vehicle (PHEV)	15
2.2.2	Electric Machine Technology in Electric Vehicle (EV)s	16
2.2.3	Charging Technologies in Electric Vehicle (EV)s	21
2.2.4	Electric Vehicle (EV) Integration into Grid	22
2.3	P2P ENERGY TRADING	23
3	OPTIMISATION TECHNIQUES & GAME THEORY	26
3.1	REINFORCEMENT LEARNING METHODS on OPTIMISATION	26
3.1.1	Components of Reinforcement Learning (RL)	26
3.1.2	Exploration & Exploitation Dilemma	28
3.1.3	Model-Free & Model-Based Learning Processes	29
3.1.4	On-Policy & Off-Policy Learning Strategies	30

3.1.5	Reinforcement Learning (RL) Algorithms	32
3.1.5.1	Monte Carlo	32
3.1.5.2	Q-Learning	33
3.1.5.3	SARSA	34
3.1.5.4	Deep Q-Learning (DQN)	36
3.1.5.5	Deep Deterministic Policy Gradient (DDPG)	37
3.1.5.6	Asynchronous Advantage Actor Critic (A3C)	39
3.1.5.7	Trusted Region Policy Optimization (TRPO)	40
3.1.5.8	Proximal Policy Optimization (PPO)	43
3.1.5.9	Twin Delayed Deep Deterministic Policy (TD3)	45
3.1.6	Soft Actor-Critic (SAC)	46
3.2	GAME THEORY on REINFORCEMENT LEARNING	48
3.2.1	Nash Equilibrium	48
3.2.2	Game Types and Applications	49
3.2.2.1	Cooperative and Non-Cooperative	49
3.2.2.2	Symmetric and Asymmetric	49
3.2.2.3	Simultaneous And Sequential	50
3.2.2.4	Perfect and Imperfect Information	50
3.2.2.5	Constant Sum, Zero-Sum And Non-Zero-Sum	51
4	MATHEMATICAL MODELLING	52
4.1	ENERGY STORAGE SYSTEM MODEL	52
4.2	ELECTRIC VEHICLE MODEL	55
4.3	ENERGY TRADING & INVOICES MODEL	64
4.4	INTEGRATED HOMES MODEL	66
4.4.1	Before Energy Trading	67
4.4.2	Energy Trading Process	70
4.4.3	After Energy Trading	71
4.5	APARTMENT BUILDING MODEL	74
4.6	REINFORCEMENT LEARNING (RL) MODEL	78
4.6.1	States	78
4.6.2	Action Spaces	79
4.6.3	Objective (Reward) Function	80
4.6.4	Reset Function	83
4.6.5	Deep Neural Network (DNN) Design	83
4.6.6	Deep Q-Learning (DQN) Agent Design	85
4.6.7	Training Options	87
5	CASE STUDY and SIMULATION RESULTS	89

5.1	STATISTICAL ANALYSIS on TRAINING DATA	92
5.2	STATISTICAL ANALYSIS on TESTING DATA	101
5.3	TRAINING PROCESS & RESULTS	110
5.3.1	Training Process	110
5.3.2	Test Results	123
6	CONCLUSION and PROPOSALS	160
A	APARTMENT BUILDING MODEL	185
B	ELECTRIC VEHICLE MODEL	187
C	ENERGY STORAGE SYSTEM MODEL	190
D	HOME MODEL	192
E	INITIALIZATION BLOCK OF THE MATLAB	199
F	DDQN MODEL	201
G	P2P ENERGY TRADING MODEL	205
H	DESCRIPTIVE STATISTIC (UTILITY)	208
	VITA	209

ABSTRACT

MULTI-AGENT REINFORCEMENT LEARNING BASED ENERGY MANAGEMENT WITH P2P/V2G

An energy management structure based on Double Deep Q-Networks (DDQN) Reinforcement Learning (RL) algorithm with Peer-To-Peer (P2P) energy trading and Vehicle-To-Grid (V2G) functionalities is presented with this study. This study investigates the effectiveness of P2P energy trading and V2G functionalities on reducing the electricity bill of each houses and contributing to energy independence in an environment including multiple prosumers, Energy Storage System (ESS)s, and Electric Vehicle (EV)

Reinforcement Learning (RL) was applied to a cluster including 6 independent houses in a residential area. Load requirements and solar energy production data of each house were collected from different sources to create high diversity. Observation space was designed with environmental information such as load requirement, solar energy production, and EV status etc. Action space of the environment was designed to optimize the energy flow from the ESSs, and EVs to be used in P2P energy trading. In addition to energy flow action signals, price action signals to be used in P2P energy trading added to the action space. Simulink simulation was performed with given environment specifications, input data and learning parameters.

Simulation results showed that by implementing a P2P energy trading platform to a cluster could reduce electricity bill by 19% on average by optimizing the energy flow. Additionally, involving the batteries of EVs to the system via V2G could reduce main grid dependency by 110 *kWh* on average in the cluster.

This study and simulation results conclude that implementing a P2P energy trading platform and V2G capability of EVs with RL algorithms optimize the system energy flow by controlling the energy sources therefore reduces the both electricity bills and energy dependence on the main grid.

Key words: Reinforcement learning, Peer-to-Peer (P2P) Energy Trading, Vehicle-to-Grid (V2G), Energy Management, Multi Agent

Date: 01.08.2023

ÖZET

P2P/V2G İLE ÇOKLU-TEMSİLCİLİ PEKİŞTİRMELİ ÖĞRENME TABANLI ENERJİ YÖNETİMİ

Bu çalışma, Pekıştirmeli Öğrenme algoritmalarından "Double Deep Q-Network" algoritmasını temel alan, komşular arası enerji alışverişini ve elektrikli araçları enerji kaynağı olarak kullanan bir enerji yönetim yapısı sunmaktadır. Bu çalışma, birden çok üretici ve tüketici, enerji depolama sistemi ve elektrikli araç bardındıran bir ortamda, komşular arası enerji alışverişini ve elektrikli araçları enerji kaynağı olarak kullanmanın, elektrik faturalarındaki ve ana şebekeye olan bağımlılığı düşürme üzerindeki etkilerinin araştırılmaktadır.

İçerisinde 6 bağımsız ev bardındıran bir yerleşim bölgesi kümesine pekıştirmeli öğrenme uygulanmıştır. Yüksek çeşitlilik oluşturmak için her evin yük gereksinimleri ve güneş enerjisi üretim verileri farklı kaynaklardan toplanmıştır. Gözlem uzayı, yük gereksinimi, güneş enerjisi üretimi, elektrikli araçların durumu gibi bilgileri içermektedir. Aksiyon uzayı, komşular arası enerji alışverişinde kullanılacak, elektrikli araçlar ve enerji depolama sisteminde bulunan bataryalardaki enerji akışını optimize edecek şekilde tasarlanmıştır. Enerji akışında kullanılacak aksiyon sinyallerine ek olarak, komşular arası enerji alışverişinde kullanılacak fiyat aksiyon sinyalleri de aksiyon uzayına eklendi. Belirlenen çevre özellikleri, girdi verileri ve öğrenme parametreleri ile Simulink simülasyonu gerçekleştirilmiştir.

Simulasyon sonuçları, enerji akışını optimize etmenin ve komşular arası enerji alışverişi platformunu belirli bir kümeye dahil etmenin, elektrik faturalarında ortalama %19'luk düşüş sağlayabileceğini göstermiştir. Ayrıca, elektrikli araçlarda bulunan bataryaları sisteme dahil etmenin, ana şebekeye olan enerji bağımlılığını küme içinde ortalama 110 kWh düşürülebileceği gösterilmiştir.

Bu çalışma ve simülasyon sonuçları, enerji kaynaklarını ve akışını pekıştirmeli öğrenme algoritması yardımıyla optimize ederek hem elektrik faturalarını hem de ana şebekeye olan enerji bağımlılığını azalttığı göstermektedir.

Anahtar kelimeler: Pekıştirmeli Öğrenme, Komşular Arası Enerji Alışverişi, Arabadan Eve, Enerji Yönetimi, Çoklu Temsilci

Tarih: 01.08.2023

SYMBOLS

Δt	Sampling Time
ϵ	Greedy Exploration Factor
η	Efficiency
Ω	Ohm
π	Policy
A	Action
Ah	Battery Cell Capacity
C	Carbon
$CCCV$	Constant Current-Constant Voltage
Co	Cobalt
E	Energy
G	Cumulative Reward
H	Hydrogen
I	Current
kWh	Kilo Watt Hour
L	Loss Function
Li	Lithium
N_p	Number of Parallel Cells in Battery
N_s	Number of Series Cells in Battery
O	Oxygen
OCV_{lut}	Look-up-Table of Open Circuit Voltage (OCV)
P	Power
Pb	Lead

Q	Value Function
R	Reward
R_{lut}	Look-up-Table of Internal Resistance (R)
S	State
S'	Next State
t	Current Time
V	Volt
W	Watt



ABBREVIATIONS

A3C	Asynchronous Advantage Actor Critic
AC	Alternative Current
BESS	Battery Energy Storage Systems
BEV	Battery Electric Vehicle
BLDC	Brushless Direct Current
BMS	Battery Management System
CNN	Convolutional Neural Network
CPU	Central Process Unit
DC	Direct Current
DDPG	Deep Deterministic Policy Gradient
DDQN	Double Deep Q-Networks
DMMDP	Distributed Multi Motor Driven Power-Train
DNN	Deep Neural Network
DQN	Deep Q-Learning
DTC	Direct Torque Control
EM	Electric Machine
EM	Electric Motor
ESS	Energy Storage System
EV	Electric Vehicle
FCEV	Fuel Cell Electric Vehicle
FOC	Field Oriented Control
GPU	Graphics Processing Unit
HEV	Hybrid Electric Vehicle
ICE	Internal Combustion Engine
IM	Induction Machine
MDP	Markov Decision Process
ML	Machine Learning
MPC	Model Predictive Control
P2P	Peer-To-Peer
PE	Power Electronic
PHEV	Plug-in Hybrid Electric Vehicle

PM	Permanent Magnet
PMSM	Permanent Magnet Synchronous Machine
PMW	Pulse Width Modulation
PPO	Proximal Policy Optimization
RES	Renewable Energy Source
RL	Reinforcement Learning
SAC	Soft Actor-Critic
SESS	Supercapacitor Energy Storage Systems
SoC	State of Charge
SoH	State of Health
SOTA	State-Of-The-Art
SRM	Switch Reluctance Machine
TD3	Twin Delayed Deep Deterministic Policy
TRPO	Trusted Region Policy Optimization
US	United States
V2G	Vehicle-To-Grid
V2H	Vehicle-to-Home
V2V	Vehicle-to-Vehicle
WHO	World Health Organization

LIST OF FIGURES

Figure 1.1	Overall Structure of Smart Microgrid	3
Figure 2.1	Typical Charging/Discharging Curves of a Lithium-Ion Battery	7
Figure 2.2	Typical Charging/Discharging Curves of a Lead-Acid Battery	8
Figure 2.3	The Major Components of a Battery Electric Vehicle (BEV)	12
Figure 2.4	The Major Components of a Hybrid Electric Vehicle (HEV)	13
Figure 2.5	Configuration Topologies of HEV Design	14
Figure 2.6	The Major Components of a Fuel Cell Electric Vehicle (FCEV)	15
Figure 2.7	The Major Components of a Plug-in Hybrid Electric Vehicle (PHEV)	16
Figure 2.8	Example of In-Wheel Electric Machine	17
Figure 2.9	Power-Train Designs in Electric Vehicle (EV)s	18
Figure 2.10	Power-Train Designs in Electric Vehicle (EV)s (DMMDP)	19
Figure 2.11	Electric Machine (EM)s Used in Electric Vehicle (EV)s	19
Figure 2.12	Visual Representation of Virtual and Physical Layers	24
Figure 2.13	Overview look at the Peer-to-Peer Energy Trading Framework	25
Figure 3.1	Components of Reinforcement Learning	26
Figure 3.2	Markov Decision Process Example in Real World Applications	27
Figure 4.1	Overall Simulink Model	52
Figure 4.2	Simulink Model of Energy Storage System (ESS)	55
Figure 4.3	Simulink Model of Electric Vehicle (EV)	63
Figure 4.4	Simulink Model of Energy Trading System	66
Figure 4.5	Simulink Model of Before Energy Trading Section	70
Figure 4.6	Simulink Model of Energy Trading Process	71
Figure 4.7	Simulink Model of After Energy Trading Section	74
Figure 4.8	Simulink Model of Apartment Building Model	77
Figure 4.9	Comprehensive Simulink Model of Apartment Building Model	78
Figure 4.10	Environment in Simulink	82
Figure 4.11	Deep Neural Network Design of the Actor-Critic in DDQN	84

Figure 4.12 Deep Q-Network Agents (DQNs) in Simulink	87
Figure 5.1 Load Power Data (Training, Complete)	93
Figure 5.2 Load Power Data (Training, Daily)	93
Figure 5.3 Solar Energy Production Data (Training, Complete)	95
Figure 5.4 Solar Energy Production Data (Training, Daily)	95
Figure 5.5 Electric Vehicle Status Signal Data (Training, Complete)	97
Figure 5.6 Electric Vehicle Status Signal Data (Training, Daily)	97
Figure 5.7 Derivative Signal Data of Electric Vehicle Status (Training, Complete)	99
Figure 5.8 Derivative Signal Data of Electric Vehicle Status (Training, Daily)	99
Figure 5.9 Daily Energy Consumption Profile of Electric Vehicles (Training, Complete)	100
Figure 5.10 Daily Energy Consumption Profile of Electric Vehicles (Training, Daily)	101
Figure 5.11 Load Power Data (Testing, Complete)	102
Figure 5.12 Load Power Data (Testing, Daily)	102
Figure 5.13 Solar Energy Production Data (Testing, Complete)	104
Figure 5.14 Solar Energy Production Data (Testing, Daily)	104
Figure 5.15 Electric Vehicle Status Signal Data (Testing, Complete)	106
Figure 5.16 Electric Vehicle Status Signal Data (Testing, Daily)	106
Figure 5.17 Derivative Signal Data of Electric Vehicle Status (Testing, Complete)	108
Figure 5.18 Derivative Signal Data of Electric Vehicle Status (Testing, Daily)	108
Figure 5.19 Daily Energy Consumption Profile of Electric Vehicles (Testing, Complete)	109
Figure 5.20 Daily Energy Consumption Profile of Electric Vehicles (Testing, Daily)	110
Figure 5.21 Episode Reward Values of Agent #1 During the Training	111
Figure 5.22 Episode Reward Values of Agent #2 During the Training	111
Figure 5.23 Episode Reward Values of Agent #3 During the Training	112
Figure 5.24 Episode Reward Values of Agent #4 During the Training	112
Figure 5.25 Episode Reward Values of Agent #5 During the Training	113
Figure 5.26 Episode Reward Values of Agent #6 During the Training	113
Figure 5.27 Average Reward Values of Agent #2 During the Training	114
Figure 5.28 Average Reward Values of Agent #2 During the Training	114
Figure 5.29 Average Reward Values of Agent #3 During the Training	115
Figure 5.30 Average Reward Values of Agent #4 During the Training	115

Figure 5.31 Average Reward Values of Agent #5 During the Training	116
Figure 5.32 Average Reward Values of Agent #6 During the Training	116
Figure 5.33 Q0 Values of Agent #1 During the Training	117
Figure 5.34 Q0 Values of Agent #2 During the Training	117
Figure 5.35 Q0 Values of Agent #3 During the Training	118
Figure 5.36 Q0 Values of Agent #4 During the Training	118
Figure 5.37 Q0 Values of Agent #5 During the Training	119
Figure 5.38 Q0 Values of Agent #6 During the Training	119
Figure 5.39 Comparison of Episode Reward, Average Reward, and Q0 of Agent #1	120
Figure 5.40 Comparison of Episode Reward, Average Reward, and Q0 of Agent #2	120
Figure 5.41 Comparison of Episode Reward, Average Reward, and Q0 of Agent #3	121
Figure 5.42 Comparison of Episode Reward, Average Reward, and Q0 of Agent #4	121
Figure 5.43 Comparison of Episode Reward, Average Reward, and Q0 of Agent #5	122
Figure 5.44 Comparison of Episode Reward, Average Reward, and Q0 of Agent #6	122
Figure 5.45 Difference Between Load Input and Solar Input	124
Figure 5.46 ESS Battery Charging Curves (Case #1)	126
Figure 5.47 ESS Battery Charging Curves (Case #2)	126
Figure 5.48 ESS Battery Charging Curves (Case #3)	127
Figure 5.49 ESS Battery Charging Curves (Case #4)	127
Figure 5.50 ESS Battery Discharging Curves (Case #1)	128
Figure 5.51 ESS Battery Discharging Curves (Case #2)	129
Figure 5.52 ESS Battery Discharging Curves (Case #3)	129
Figure 5.53 ESS Battery Discharging Curves (Case #4)	130
Figure 5.54 ESS Battery Energy Sold Curves (Case #1)	131
Figure 5.55 ESS Battery Energy Sold Curves (Case #2)	131
Figure 5.56 ESS Battery Energy Sold Curves (Case #3)	132
Figure 5.57 ESS Battery Energy Sold Curves (Case #4)	132
Figure 5.58 EV Battery Charging Curves (Case #1)	133
Figure 5.59 EV Battery Charging Curves (Case #2)	134
Figure 5.60 EV Battery Charging Curves (Case #3)	134
Figure 5.61 EV Battery Charging Curves (Case #4)	135
Figure 5.62 EV Battery Discharging Curves (Case #1)	136
Figure 5.63 EV Battery Discharging Curves (Case #2)	136

Figure 5.64 EV Battery Discharging Curves (Case #3)	137
Figure 5.65 EV Battery Discharging Curves (Case #4)	137
Figure 5.66 EV Battery Energy Sold Curves (Case #1)	138
Figure 5.67 EV Battery Energy Sold Curves (Case #2)	139
Figure 5.68 EV Battery Energy Sold Curves (Case #3)	139
Figure 5.69 EV Battery Energy Sold Curves (Case #4)	140
Figure 5.70 Energy Bought from Grid (Case #1)	141
Figure 5.71 Energy Bought from Grid (Case #2)	141
Figure 5.72 Energy Bought from Grid (Case #3)	142
Figure 5.73 Energy Bought from Grid (Case #4)	142
Figure 5.74 Energy Sold to Grid (Case #1)	143
Figure 5.75 Energy Sold to Grid (Case #2)	144
Figure 5.76 Energy Sold to Grid (Case #3)	144
Figure 5.77 Energy Sold to Grid (Case #4)	145
Figure 5.78 Energy Bought via P2P (Case #1)	146
Figure 5.79 Energy Bought via P2P (Case #2)	146
Figure 5.80 Energy Bought via P2P (Case #3)	147
Figure 5.81 Energy Bought via P2P (Case #4)	147
Figure 5.82 Energy Sold via P2P (Case #1)	148
Figure 5.83 Energy Sold via P2P (Case #2)	149
Figure 5.84 Energy Sold via P2P (Case #3)	149
Figure 5.85 Energy Sold via P2P (Case #4)	150
Figure 5.86 SoC of the ESS Battery (Case #1)	151
Figure 5.87 SoC of the ESS Battery (Case #2)	151
Figure 5.88 SoC of the ESS Battery (Case #3)	152
Figure 5.89 SoC of the ESS Battery (Case #4)	152
Figure 5.90 SoC of the EV Battery (Case #1)	153
Figure 5.91 SoC of the EV Battery (Case #2)	153
Figure 5.92 SoC of the EV Battery (Case #3)	154
Figure 5.93 SoC of the EV Battery (Case #4)	154
Figure 5.94 Energy Usage Bills of Each Cases (Home #1)	155
Figure 5.95 Energy Usage Bills of Each Cases (Home #2)	155
Figure 5.96 Energy Usage Bills of Each Cases (Home #3)	156

Figure 5.97 Energy Usage Bills of Each Cases (Home #4)	156
Figure 5.98 Energy Usage Bills of Each Cases (Home #5)	157
Figure 5.99 Energy Usage Bills of Each Cases (Home #6)	157



LIST OF TABLES

Table 2.1	Comparison Between BESS and SESS Performance	10
Table 4.1	Input Arguments of the “ESS” Function	53
Table 4.2	Output Arguments of the “ESS” Function	53
Table 4.3	Input Arguments of the “ESSSoC” Function	54
Table 4.4	Output Arguments of the “ESSSoC” Function	55
Table 4.5	Output Arguments of the “Li-Ion look-up” Function	56
Table 4.6	Input Arguments of the “Battery” Function	57
Table 4.7	Output Arguments of the “Battery” Function	57
Table 4.8	Input Arguments of the “Drive SoC Calculator” Function	58
Table 4.9	Output Arguments of the “Drive SoC Calculator” Function	58
Table 4.10	Input Arguments of the “SoC” Function	59
Table 4.11	Output Arguments of the “SoC” Function	59
Table 4.12	Input Arguments of the “Charger” Function	59
Table 4.13	Output Arguments of the “Charger” Function	60
Table 4.14	Input Arguments of the “Discharger” Function	60
Table 4.15	Output Arguments of the “Discharger” Function	61
Table 4.16	Input Arguments of the “EV” Function	61
Table 4.17	Output Arguments of the “EV” Function	62
Table 4.18	Input Arguments of the “Trade with Equal Opportunities” Function . .	64
Table 4.19	Output Arguments of the “Trade with Equal Opportunities” Function .	64
Table 4.20	Input Arguments of the “Prepare Invoices” Function	65
Table 4.21	Output Arguments of the “Prepare Invoices” Function	65
Table 4.22	Input Arguments of the “Allocate Power for Energy Trading” Function	66
Table 4.23	Output Arguments of the “Allocate Power for Energy Trading” Function	66
Table 4.24	Input Arguments of the “Before Energy Trading” Function	67
Table 4.25	Output Arguments of the “Before Energy Trading” Function	68
Table 4.26	Input Arguments of the “Energy Trading Process” Function	70
Table 4.27	Output Arguments of the “Energy Trading Process” Function	71

Table 4.28	Input Arguments of the “After Energy Trading” Function	72
Table 4.29	Output Arguments of the “After Energy Trading” Function	72
Table 4.30	Input Arguments of the “Apartment Building” Function	75
Table 4.31	Output Arguments of the “Apartment Building” Function	76
Table 4.32	States of the Environment	79
Table 4.33	Actions of the Environment	80
Table 4.34	Formatted Actions Example to be used in learning	80
Table 4.35	Summary Table of DNN	85
Table 4.36	Agent Options of Built-in MATLAB DQN Agent Class	86
Table 4.37	Training Options of Built-in MATLAB “Train” Function	88
Table 5.1	Sample Data	90
Table 5.2	Main Grid Energy Buying/Selling Prices	90
Table 5.3	Battery Parameters of Energy Storage Systems	91
Table 5.4	Electric Vehicle and V2G Functionality Parameters	91
Table 5.5	Electric Vehicle Parameters	92
Table 5.6	Battery Parameters of Electric Vehicles	92
Table 5.7	Descriptive Statistical Analysis of Load Power Data (Training)	94
Table 5.8	Descriptive Statistical Analysis of Solar Energy Production Data (Training)	96
Table 5.9	Descriptive Statistical Analysis of Electric Vehicle Status Data (Training)	98
Table 5.10	Departure and Arrival Hours of Electric Vehicles (Training)	98
Table 5.11	Daily Energy Consumption Profile of Electric Vehicles (Training)	100
Table 5.12	Descriptive Statistical Analysis of Load Power Data (Testing)	103
Table 5.13	Descriptive Statistical Analysis of Solar Energy Production Data (Testing)	105
Table 5.14	Descriptive Statistical Analysis of Electric Vehicle Status Data (Testing)	107
Table 5.15	Departure and Arrival Hours of Electric Vehicles (Testing)	107
Table 5.16	Daily Energy Consumption Profile of Electric Vehicles (Testing)	109
Table 5.17	Cases used in Testing Process	123
Table 5.18	Descriptive Statistical Analysis of Load Power Data (Testing)	125
Table 5.19	Comparison of Total ESS Battery Charging Energy	128
Table 5.20	Comparison of Total ESS Battery Discharging Energy	130
Table 5.21	Comparison of Total Battery P2P Selling Energy	133

Table 5.22	Comparison of Total EV Battery Charging Energy	135
Table 5.23	Comparison of Total EV Battery Charging Energy	138
Table 5.24	Comparison of Total Electric Vehicle P2P Selling Energy	140
Table 5.25	Comparison of Total Energy Bought from Grid	143
Table 5.26	Comparison of Total Energy Sold to Grid	145
Table 5.27	Comparison of Total Energy Bought via P2P	148
Table 5.28	Comparison of Total Energy Sold via P2P	150
Table 5.29	Comparison of Total Energy Usage Bills	158
Table 5.30	Comparison of Total Energy Usage Bills by %	158
Table 5.31	Comparison of Total Energy Usage	159



CHAPTER 1 INTRODUCTION

In recent years, interests in the microgrids have been exponentially increased. Microgrids compose of the integration of distributed power sources, energy storage systems, digital monitoring equipments, and in some cases electric vehicles [1]. To fill the gap between energy-efficient grids and microgrid concepts, energy management algorithms and smart microgrid concept have been main focus of researchers lately. Increasing demand of countries on green energy, generation capacity of renewable energy sources, production of electric vehicles, and people interest in energy efficiency concept has enabled improvement and integration of machine learning and optimization algorithms in energy management systems.

With the increasing demand on distributed energy sources, residential households have been becoming prosumers, instead of being pure consumers [2]. Being a prosumer allows a household to participate in energy selling process with its surplus energy. In addition to selling surplus energy back to the main grid, prosumers can sell surplus energy, that are produced by distributed energy sources, to their neighbors through Peer-To-Peer (P2P) energy trading mechanism. Furthermore, batteries with relatively high energy capacity in Electric Vehicle (EV)s can be used in Peer-To-Peer (P2P) energy trading mechanism through Vehicle-to-Home (V2H), Vehicle-to-Vehicle (V2V), and Vehicle-To-Grid (V2G) integration.

Multiple distributed energy sources, energy storage systems, electric vehicles with V2X capability, and energy trading across neighbors cause optimization issues from energy management perspective. Proper use of limited energy sources can significantly increase energy efficiency. To overcome optimization issues, researchers all around the world have offered various optimization algorithms based on conventional and machine learning algorithms.

Appliance scheduling based optimization methods have been widely offered in the literature. Jinsoo Han and et al. proposed a Power Line Communication (PLC) based method to monitor renewable energy production and daily energy consumption profile to optimize energy usage by scheduling and controlling the appliance [3]. Mahmood Anzar and et al. proposed a Demand Side Management (DSM) in smart grid with renewable energy source integration to minimize the energy cost [4]. Furthermore, Hyunjeong Lee and et al. proposed similar solution with smart plugs to collect energy consumption data from appliances to control energy management through scheduling [5]. Other than conven-

tional optimization methods, Reinforcement Learning (RL) based optimization methods have recently been offered in the literature. Bin Zhang and et al. proposed Proximal Policy Optimization (PPO) based algorithm to control thermostatically controlled loads to minimize operation costs [1]. Aysegül Kahraman and et al. proposed 3 distinct RL algorithms namely Deep Q-Learning (DQN), Twin Delayed Deep Deterministic Policy (TD3), and Deep Deterministic Policy Gradient (DDPG) to minimize operational cost by scheduling usage of appliances [2]. M. K. Perera and et al. proposed RL implementation through Python environment to optimize energy storage system utilization and to minimize grid dependency [6]. Chong Aih Hau and et al. proposed Deep Q-Learning (DQN) based RL algorithm to automate energy trading with lithium-ion battery Energy Storage System (ESS) [7]. Nanqian Ruan and et al. proposed Soft Actor-Critic (SAC) based RL algorithm to overcome uncertainties of renewable energy generation and load demands to optimize energy management with taking constraints into account [8].

Deep examination of the current literature shows a significant gap of implementation Peer-To-Peer (P2P) and Vehicle-To-Grid (V2G) concepts. Additionally, most of the RL based algorithms have focused on single agent environments. Moreover, many researches have focused on scheduling the usage time of appliances. However, scheduling the usage of appliances causes lack of comfort issues throughout the day and introduce several uncertainties. To overcome aforementioned issues, this study focuses on multi-agent Deep Q-Learning (DQN) algorithm based energy management system including Peer-To-Peer (P2P) energy trading and Vehicle-To-Grid (V2G) concepts to optimize energy costs and to reduce grid dependency.

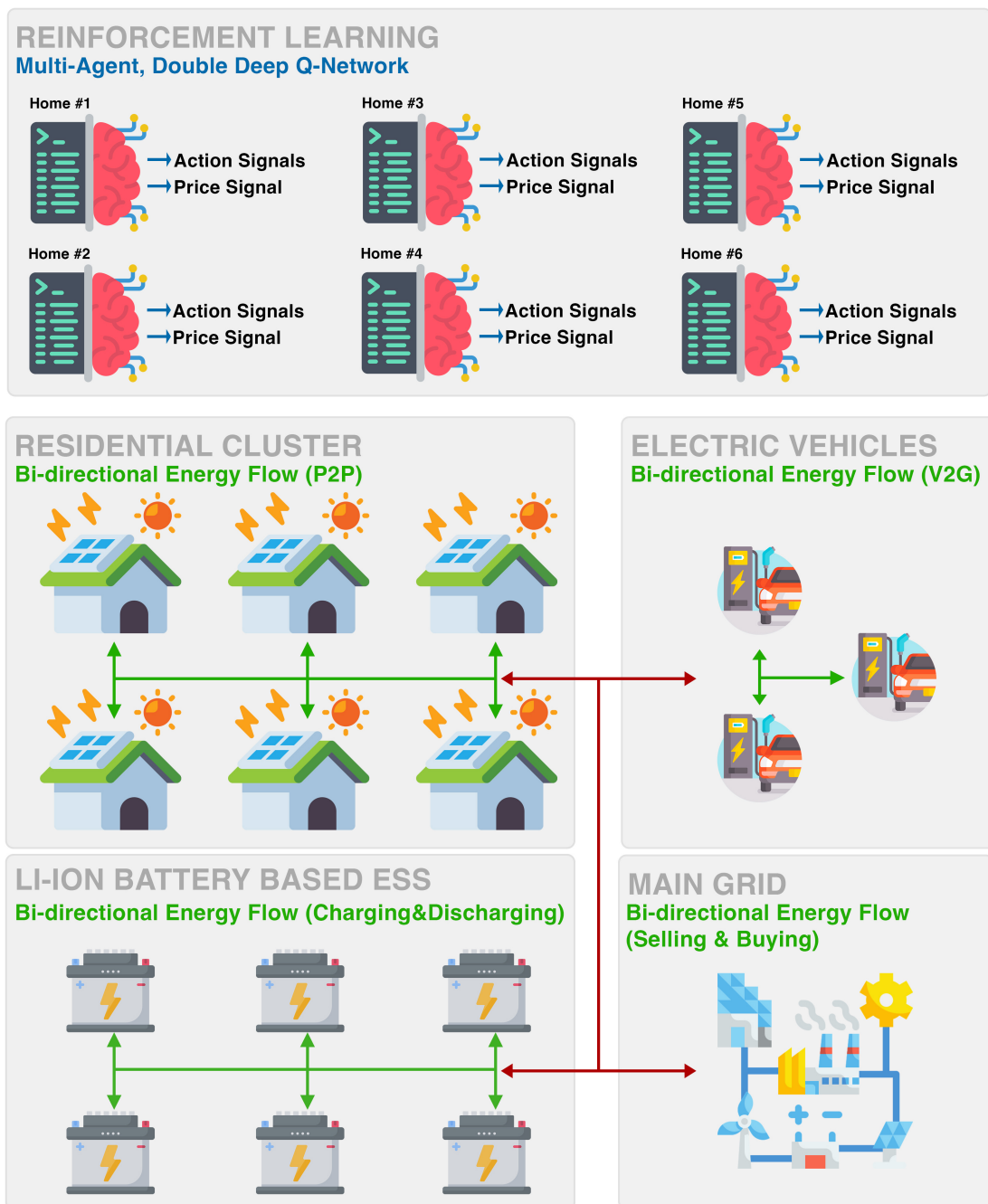


Figure 1.1: Overall Structure of Smart Microgrid

1.1 AIMS OF THE STUDY

The aim of the master thesis is to design a smart energy management structure and to optimize the energy management to be used in residential areas formed by detached house clusters or apartment buildings. Optimizing the charge/discharge curves of the batteries placed in the central Energy Storage System (ESS) that belong to each house in the cluster and optimizing the amount of energy to be used in Peer-To-Peer (P2P) energy trading through Reinforcement Learning (RL) are the two fundamental objectives of the study. The third fundamental objective of the study is to optimize the charge/discharge curves of the batteries placed in the Electric Vehicle (EV) that contain relatively high energy capacities. As in the batteries in the ESS, optimizing the amount of energy to be used in P2P is the fourth fundamental objective of the study. The last fundamental objective of the study is to optimize the price policy that will be used in P2P energy trading through RL and cooperative game theory by maximizing the long term profits of each houses. Fundamental objectives can be summarized as follows.

- To optimize the charge/discharge curves of the batteries placed in ESS and EV,
- To optimize the energy amount (ESS, EV, etc.) to be used in P2P energy trading with RL and Cooperative Game Theory,
- To optimize the price policy to be used in P2P energy trading by maximizing the long term profits of each houses.

Besides the fundamental objectives, reducing the electricity bills of each houses in the long term through the advantage of P2P and energy supply capability of EVs with Vehicle-To-Grid (V2G) is considered as the economic aspect of the study. Similarly, minimizing the dependency on grid via optimizing the energy flow and encouraging P2P energy trading inside the cluster is considered as a social-economic aspect of the study.

1.2 LIMITATIONS OF THE STUDY

Addressing the limitations of the study has importance to achieve a stable solution to an energy management structure that contains multiple electric vehicles, renewable energy sources, energy storage system, and prosumers. As mentioned earlier, optimizing the energy flow between multiple prosumers via P2P and V2G functionalities composes the aim of the study. Thus, P2P and V2G limitations during the implementation should be carefully examined.

Data collection process to be used in environmental inputs, such as load requirement and solar energy production, highly dependent on the geographical locations and consumer

behaviors. High diversity data were collected to build a stable learning environment during the simulation. However, different climate conditions and cultural consumer behaviors might require constant learning updates and different learning parameters.

Current State-Of-The-Art (SOTA), such as battery structures and designs of EVs and ESSs, Renewable Energy Source (RES)s availability, efficiency constraints were taken into account in the learning process. Technological advancements of the components used in the model might require updates to catch the current SOTA.

Training of Deep Neural Network (DNN) models requires relatively long time with current Central Process Unit (CPU)s and Graphics Processing Unit (GPU)s. Defining the episode and step numbers of the learning process were limited to the study duration. Therefore, results might not show the most accurate solutions for the given environment.

Uncertainties of departure and arrival times of EVs constitute the biggest challenge of the study. Throughout the study, assumption was made that EVs have predefined departure and arrival times to simplify the learning process and therefore to achieve more stable learning process.

1.3 STRUCTURE & OUTLINE OF THE THESIS

The remainder of this master thesis is organized as follows. Chapter 2 provides a comprehensive literature review, examining the current state-of-the-art technologies in Energy Storage System (ESS), Electric Vehicle (EV), and Vehicle-To-Grid (V2G) functionality, as well as their application in energy management systems. Chapter 3, as in Chapter 2, provides a comprehensive literature review, examining the current state-of-the-art technologies in Reinforcement Learning (RL), and Game Theory models, as well as several algorithm examples and game types in energy management systems. Chapter 4 describes the mathematical models used in MATLAB and Simulink to train and simulate the environment. Chapter 5 describes the examined case studies and shows the simulation results. In the final chapter, Chapter 6, conclusion of the study and future proposals are presented.

CHAPTER 2 ESS, EV, and P2P ENERGY TRADING in SMART MICRO GRIDS

2.1 ENERGY STORAGE SYSTEMS

Electricity demand around the whole world is assumed to be growing approximately 2.5-3% annually [9]. Therefore, the generation capability of electricity companies must be increased accordingly. As a result of the economic and pandemic crises in the world, increasing the traditional energy production capacities is getting more expensive day by day. As a consequence, energy production companies find the solution to benefit from renewable energy sources. One of the greatest challenges for energy generation companies is matching the demanded loads and operating smoothly [10]. For the purpose of smooth operations, traditional power generation is now being strengthened and supported by renewable energy sources. Such renewable energy sources contain solar, wind, geothermal, hydropower, ocean waves and bioenergy power generation [11, 12]. On the other hand, due to their nature of power generation, such sources can produce energy intermittently. As a result of this, renewable energy sources can not literally serve like traditional power generation methods and therefore can not fully meet the demanded loads [13]. To overcome this issue, the interest in Energy Storage Systems is increasing worldwide. Energy Storage System enables us to solve the problem of intermittent energy production by renewable energy sources by storing the excess energy to be scheduled to use in future demands where there is a lack of energy production [14]. Researchers aimed to develop and increase the variety of Energy Storage Systems. Different types of Energy Storage Systems, such as battery, hydrogen, flywheels, super capacitors, and thermal fluid storage, have been proposed and investigated often in literature [15–20]. In this chapter, due to the area of interest of the thesis, Electro-Chemical (Battery Energy Storage System) and Electrical (Super-Capacitor) storage systems will be investigated.

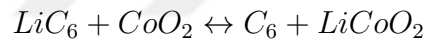
2.1.1 Battery Energy Storage Systems (BESS)

All around the world, to meet the demanded loads and to ensure the stability of power grid, electrochemical energy storage systems are widely used. Reasons for the electrochemical energy storage systems usage is their ability to be well integrated into grid applications. In fact, BESS will in the future become more attractive to further grid applications due to its decreasing price and increasing power/energy density. Typical electrochemical energy storage systems include Lithium-Ion and Lead-Acid, Sodium Sulphur, Metal Air and

Flow Battery systems [21, 22]. Different chemicals in the battery systems present different characteristics, such as cost, life cycle, power/energy density [22]. Batteries with such chemical reactions, regardless of the chemistry, can be charged/discharged within their charging/discharging cycle life. Functionality of charging/discharging cycle makes the storage process efficient. Throughout the history of BESS applications, various types of battery chemicals have been applied to BESS. However, Lithium-Ion and Lead-Acid batteries are the most frequently applied chemical in BESS applications.

2.1.1.1 Lithium-Ion Batteries

Lithium-Ion batteries have dominantly been used in consumer electronics. Moreover, due to its energy density, lack of memory effect and energy storage ability when not in use, Lithium-Ion batteries have been the most popular battery type which is used in electric vehicle, military and aerospace applications. Lithium-Ion batteries consist of one anode electrode and one cathode electrode. Those electrodes are separated by a separator. During the discharging process, lithium ions move from the negative electrode to the positive electrode. Likewise, during the charging process, lithium ions move from cathode electrode to the anode electrode. A typical chemical reaction in the Lithium-Ion battery is as follows.



Above chemical reaction represents the discharging (left to right) and the charging (right to left) process of a Lithium-Ion battery. The important characteristics, among the whole other chemical types, can be sorted as power/energy density, long life cycle, consistent charge and discharge process, stable performance in a wider temperature range and self-discharge when not in use [23, 24]. Typical charging/discharging curve of a Lithium-Ion battery can be seen in the figure below.

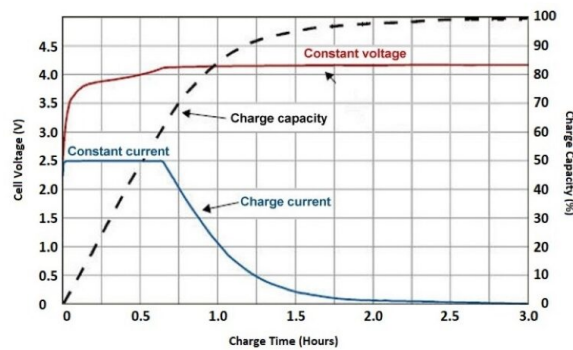


Figure 2.1: Typical Charging/Discharging Curves of a Lithium-Ion Battery

To achieve the desired voltage-current ratings, Lithium-Ion batteries must be assembled in series-parallel configurations [25]. Different types of Lithium-Ion batteries have been proposed in the literature. Such battery types include Lithium cobalt oxide (LiCoO₂), lithium manganese oxide (LiMn₂O₄), lithium iron phosphate (LiFePO₄), lithium nickel–manganese-cobalt oxide (LiNiMnCoO₂), lithium nickel cobalt aluminum oxide (LiNiCoAlO₂) and lithium titanate (Li₄Ti₅O₁₂) [26]. Different types of Lithium-Ion batteries represent slightly different chemical characteristics.

2.1.1.2 Lead-Acid Batteries

Lead-Acid Batteries have mostly been used in electric power storage for solar and wind power generations. Furthermore, in Internal Combustion Engine vehicles, Lead-Acid batteries are used to start the ignition system. Lead-Acid Batteries are the first type of rechargeable batteries. Lead-Acid batteries consist of a lead and lead oxide to convert chemical energy into electrical energy [27]. Just like in the Lithium-Ion batteries, Lead-Acid batteries consist of one anode electrode and one cathode electrode. During the discharging process, electrons move from the negative electrode to the positive electrode. Likewise, during the charging process, electrons move from the cathode electrode to the anode electrode. A typical chemical reaction in the Lead-Acid battery is as follows.



Above chemical reaction represents the discharging (left to right) and the charging (right to left) process of a Lead-Acid battery. The important characteristics of Lead-Acid batteries can be sorted as less expensive than other batteries, relatively low cost, safety and stable charge/discharge operation performance [28]. Typical charging/discharging curve of a Lead-Acid battery can be seen in the figure below.

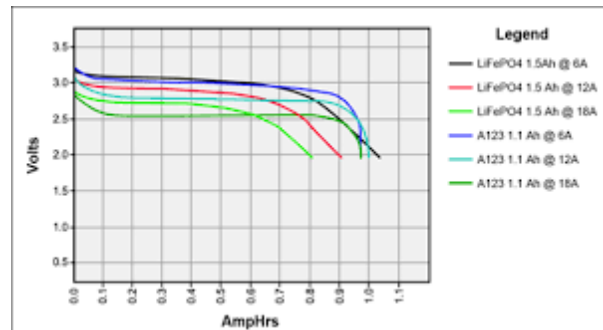


Figure 2.2: Typical Charging/Discharging Curves of a Lead-Acid Battery

2.1.1.3 Battery Management System (BMS)

In order to protect, monitor and maintain the safety of BESS, Battery Management System (BMS) are introduced. Due to their electrochemical characteristics, if not controlled, batteries can be over charged or over discharged [29]. Batteries are expected to achieve a certain amount of cycle life and thus to store its capacity at maximum. Some reasons causing batteries to not achieve the expected cycle life are manufacturing and production quality, material used during production. However, incorrect usage and monitoring are the most important factors affecting the cycle life of a battery. Incorrect usage and monitoring result in temperature issues, over charging and over discharging of battery [30,31]. These issues relatively accelerate the degradation of the battery. BMS can significantly reduce the root causes of degradation, thus can improve the cycle life and capacity retention. Model based BMSs are particularly important, where batteries are assembled to get battery packs, for electric vehicles and grid integrated energy storage applications [32]. Minimizing temperature diversity across the battery pack, monitoring each cell to protect against internal degradation, achieving the optimal charging curve for each cell and balancing the voltage levels of each cell throughout the battery pack are the main features of a BMS. Due to the electrochemical reactions on the battery, the internal chemical states of each cell can not be measured. The BMS can only measure the cell voltage, current and temperature. In addition to the limited measurement capability, the BMS can only control the amount of electrical current and ambient temperature by cooling/heating the ambient. Such battery internal states like State of Charge (SoC) and State of Health (SoH) can only be estimated by BMS according to measured voltage, current and temperature values. One of the features of the BMS is to record the data history to diagnose the issues of each cell. Accurate estimations of SoC and SoH can lead to improved cycle life and energy capacity of a battery. Estimation of SoC and SoH is an extensive research topic in the literature. Researchers have proposed such algorithms like Extended Kalman Filter (EKF) [33–35], Fuzzy Unscented Kalman Filter [36], Adaptive and Equivalent Circuit Model algorithm [37,38], and Neural Network algorithms [39]. With the development of the micro-computers and accelerated neural networks algorithm, and due to the significant increase in the use of electric vehicles and battery energy storage systems, SoC/SoH estimation is still quite an important and wide research topic today.

2.1.2 Supercapacitor Energy Storage Systems (SESS)

As mentioned earlier, interest in conventional energy production, such as fossil fuel, is decreasing day by day. Due to decreasing interest in conventional energy production, the interest in energy production moves toward sustainable and renewable energy sources [40]. Nowadays, with multiple production techniques, such as solar and wind, the importance

of renewable energy production has been increased [41]. BESSs have characteristics such as high energy density. On the contrary, BESSs are reasonably weak in power density. Weakness in power density raises problems such as ability to supply high charging and discharging power [42]. Batteries used in BESSs and Supercapacitor used in Supercapacitor Energy Storage Systems (SESS) show similar chemical characteristics generally. Energy storing and releasing principles are almost the same in terms of chemical reactions [40]. On the other hand, Supercapacitors can perform a greater amount of charging/discharging cycles whereas batteries are so limited to approximately a few thousand of charging/discharging cycles. Furthermore, Supercapacitors can supply a greater amount of charging/discharging currents than typical batteries [43].

Table 2.1: Comparison Between BESS and SESS Performance

Device Characteristics	BESS	SESS
Charging Time	$1 < t < 5 \text{ h}$	1 - 30 s
Discharging Time	$t > 0.3 \text{ h}$	1 - 30 s
Energy Density (Wh/kg)	10 - 100	1 - 10
Cycle Life	1000	1000000
Power Density (W/kg)	< 1000	10000
Charge/Discharge Efficiency	0.7 - 0.85	0.85 - 0.98

As can be seen in the above table, in general, Supercapacitors offer higher power density, lower energy density, higher charging/discharging efficiency, higher cycle life and rapid charging/discharging time against batteries [44]. Since the energy density of Supercapacitors are relatively lower than batteries, most applications require a combination of Battery-Supercapacitor hybrid energy storage systems [45]. Aim of using a hybrid energy storage system is to combine the benefits of Supercapacitor characteristics, such as long cycle life and power density, with the Battery characteristics, such as high energy density [46].

2.2 ELECTRIC VEHICLES and V2G

Quite long history lies behind the development of Electric Vehicle (EV)s. EVs have been around for 150 years after the invention of electric motors [47]. Back in from the 1920s to 1930s, EVs had been used as the primary transportation vehicle all around the world, especially for local and public transportation. After the late 1930s, due to its performance and friendly cost, EVs were replaced by Internal Combustion Engine (ICE) vehicles. In

the past, lack of reliable electric transmission, quite limited driving range, EVs were not primarily preferred by vehicle owners. Through the wider availability of gasoline, ICE vehicles were spreaded out all around the world. Furthermore, gasoline enabled consumers to carry the energy source in a container and extended range caused consumers to demand an ICE vehicle. Energy crisis in the United States (US) postal service brought back the EVs into attention in the early 1970s. However, as before, due to several issues, EVs development has been terminated. General Motors started to produce EV1 which managed to get people's attention back on EVs. Sequentially, Ford, Toyota and Honda followed General Motors on EVs production in early 2000s. With the support of governments, due to the green environment policies, almost all modern vehicle companies are producing different types of EVs nowadays.

According to a survey conducted by the WHO, more than 90% people are being exposed to pollution far beyond the limit set by the WHO [48]. It has been found that there is a positive correlation between pollution and cardiovascular disorders. A research conducted by Swedish researchers, showed that being exposed to pollution can lead to Diabetes Mellitus [49]. Air pollution does not only affect human health, but also causes global warming and thus climate change. Under these circumstances, EVs appear to be the most eco-friendly and nearly zero-emission option for transportation. Therefore, a tremendous revolution is expected in the automotive industry [50].

2.2.1 Classification of Electric Vehicle (EV)s

Batteries, Battery Management System (BMS), Power Electronic (PE) units to control energy conversion processes, and Electric Motor (EM) are the main components of EVs. Unlike in the ICE vehicles, EVs use electric motors to create propulsion force. Despite the fact that internal combustion engines have an excessive amount of emission release, whereas electric motors have almost no emission release. The major advantages of an electric motor are higher starting torque, higher efficiency and lower maintaining cost than ICEs [51]. Electric vehicles are classified according to energy sources and propulsion systems. There are four major types of EVs, namely Battery Electric Vehicle (BEV), Hybrid Electric Vehicle (HEV), Fuel Cell Electric Vehicle (FCEV), and Plug-in Hybrid Electric Vehicle (PHEV).

2.2.1.1 Battery Electric Vehicle (BEV)

As the name suggests, Battery Electric Vehicle (BEV)s use batteries as energy sources. BEV additionally benefit from the rapid charge/discharge capabilities of super-capacitors by combining with batteries as multiple energy sources [52]. Battery must be charged by the electric grid or renewable sources. Theoretically, BEVs have zero emission release

to nature. In addition to zero emission, they perform relatively high performance and efficiency. BEVs are independent of gasoline. On the other hand, the major disadvantages of BEVs are short range, approximately 250-350 km, and high initial cost due to batteries. However, with the non-existence of moving mechanical parts like in ICEs, BEVs have longer life span and lower maintenance cost [48]. The major components of a typical BEV are electric motor, power electronics unit, battery, cooling system, onboard charger, charging port, transmission lines. Some of the BEVs models produced are Tesla Model 3, BMW i3, Nissan Leaf, Tesla Model X.

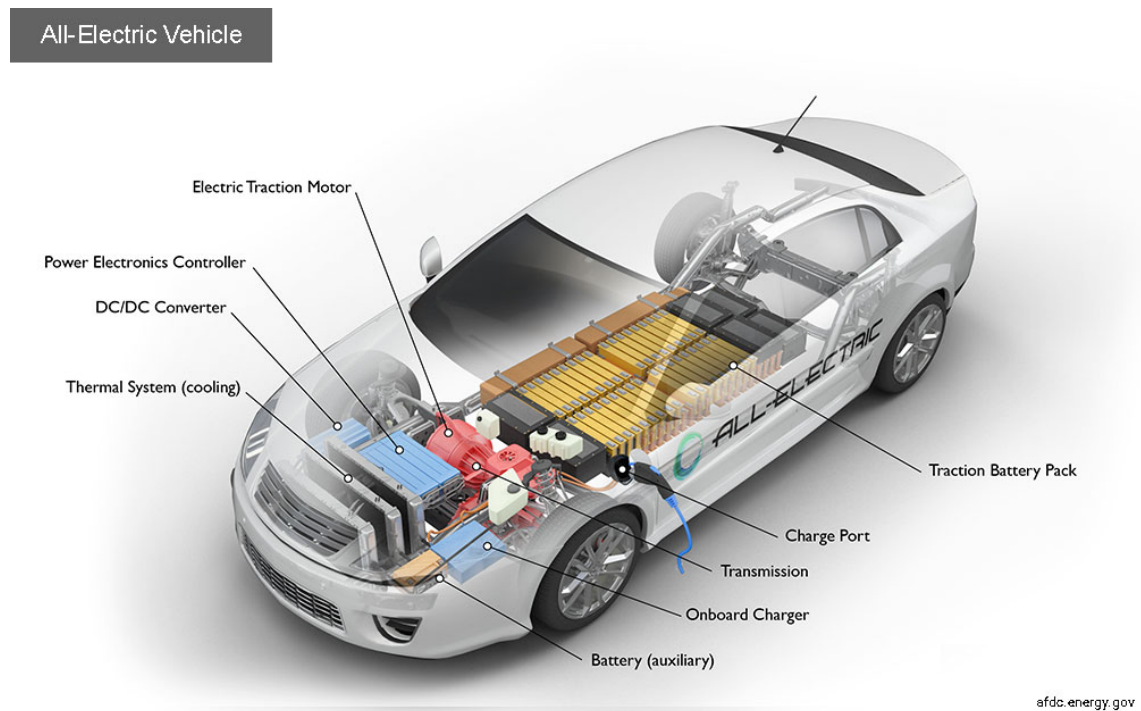


Figure 2.3: The Major Components of a Battery Electric Vehicle (BEV) [53]

2.2.1.2 Hybrid Electric Vehicle (HEV)

Unlike BEVs, HEVs use multiple energy sources. HEVs generally use batteries, super-capacitors and gasoline as energy sources. Similar to BEVs, HEVs use super-capacitors to supply energy during rapid energy demands. The batteries in the HEVs do not have to be charged from outside. HEVs can fulfill energy demand by gasoline stations. Therefore, HEVs are dependent on gasoline. HEVs release low emission to nature and significantly less emission than ICE vehicles. Owning batteries in HEVs leads to savings in gasoline fuel as compared to ICEs vehicles. HEVs perform better than BEVs in terms of range. However, due to the multiple energy sources, HEVs are worse than ICE vehicles in terms of cost. Furthermore, multiple energy sources cause optimization and management problems in terms of control. In addition to control problems, sizing and managing the battery

is more problematic than BEVs [54]. The major components of a typical HEV are an electric motor, power electronics unit, battery, cooling system, transmission lines, ICE, and gasoline fuel tank.

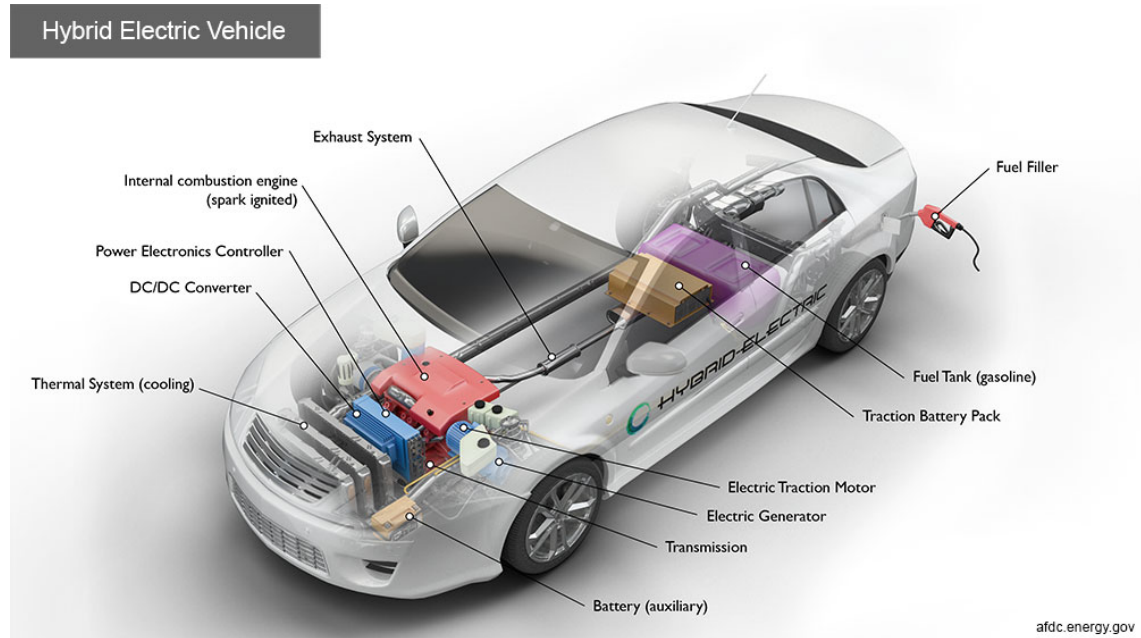


Figure 2.4: The Major Components of a Hybrid Electric Vehicle (HEV) [53]

The electric motor also performs generative action, as called regenerative braking, to capture the excess energy of the ICE [55]. To improve power performance and usage of ICE for fuel economy, different types of HEVs, such as series, parallel and series-parallel, have been developed [56–59].

In series HEVs, the traction force is supplied by the electric motor or ICE. The secondary electric motor acts as a generator to generate energy to charge the batteries by ICE power. The purpose of series power-train architecture is to extend the range of ICE by supplying extra battery energy. In series HEVs, batteries can alone power the vehicle via electric motor. Gasoline can power the vehicle via ICE. Combined traction action by both electric motor and ICE can be performed. Moreover, ICE can charge the battery in power split mode [56–59].

In parallel HEVs, the traction force is supplied by both electric motor and ICE. In this architecture, electric motor and ICE are mechanically coupled to the drive shaft via clutches. The traction power can be created by the ICE alone, by the electric motor alone or by both. In parallel configuration, the secondary electric motor is not used as a generator. Therefore, unlike the series configuration, parallel configuration only needs two propulsion systems. In parallel HEVs, batteries can alone power the vehicle via electric motor. Gasoline can power the vehicle via ICE. Combined traction action by both electric

motor and ICE can be performed. Moreover, ICE can charge the battery in power split mode [56–59].

In series-parallel HEVs, combinations of series and parallel configuration are assembled. Additional mechanical parts are required to assemble such a configuration. Construction cost of a series-parallel HEV is much higher compared to other configurations. Additionally, combining series and parallel configuration requires advanced control techniques and additional mechanical parts [56–59]. Some of the HEVs models produced are Honda Civic, Toyota Prius, Toyota Camry.

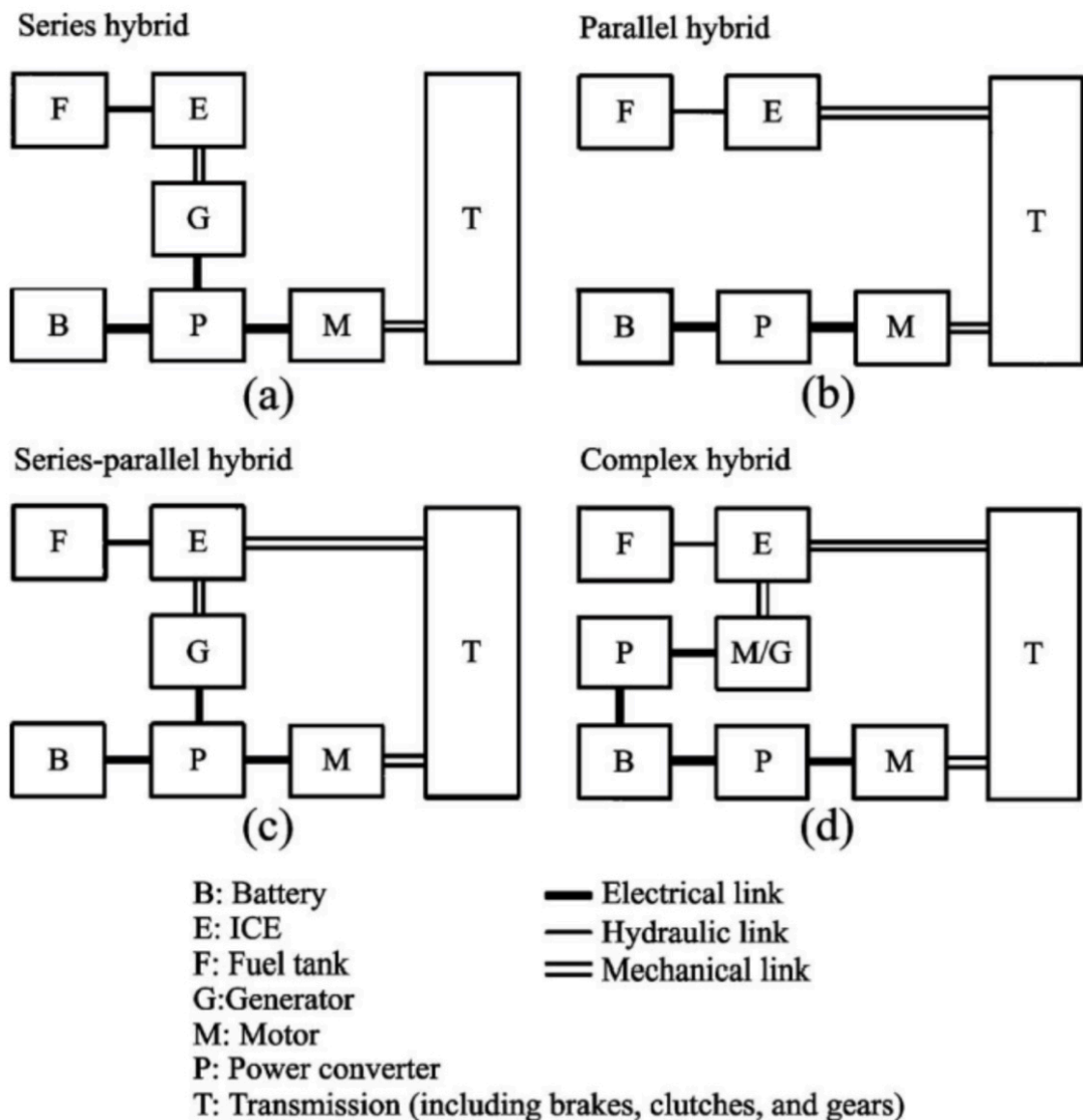


Figure 2.5: Configuration Topologies of HEV Design [55]

2.2.1.3 Fuel Cell Electric Vehicle (FCEV)

As the name suggests, Fuel Cell Electric Vehicle (FCEV)s use fuel cells as primary energy sources. In addition to fuel cells, FCEVs use batteries and super-capacitors to benefit high energy and high power density of batteries and super-capacitors, especially to overcome high power density needed for starting [60–62]. FCEVs require replacement or filling of a hydrogen tank by hydrogen when empty. Just like in BEVs, FCEVs release theoretically zero-emission or nearly zero-emission to nature. Hydrogen is an efficient energy source for vehicles. Moreover, FCEVs are independent of gasoline. FCEVs generally offer medium driving range. Due to its underdevelopment, hydrogen is a high cost energy source nowadays. On the other hand, the reliability of fuel cells is still being investigated by researchers [63–65]. Furthermore, complexity of hydrogen infrastructure is another downside of FCEVs. The major components of a typical FCEV are electric motor, power electronics unit, additional battery pack, hydrogen fuel tank, filling port, and transmission lines. Some of the FCEVs models produced are Hyundai Tucson, Hyundai Nexo, Honda Clarity and Toyota Mirai.

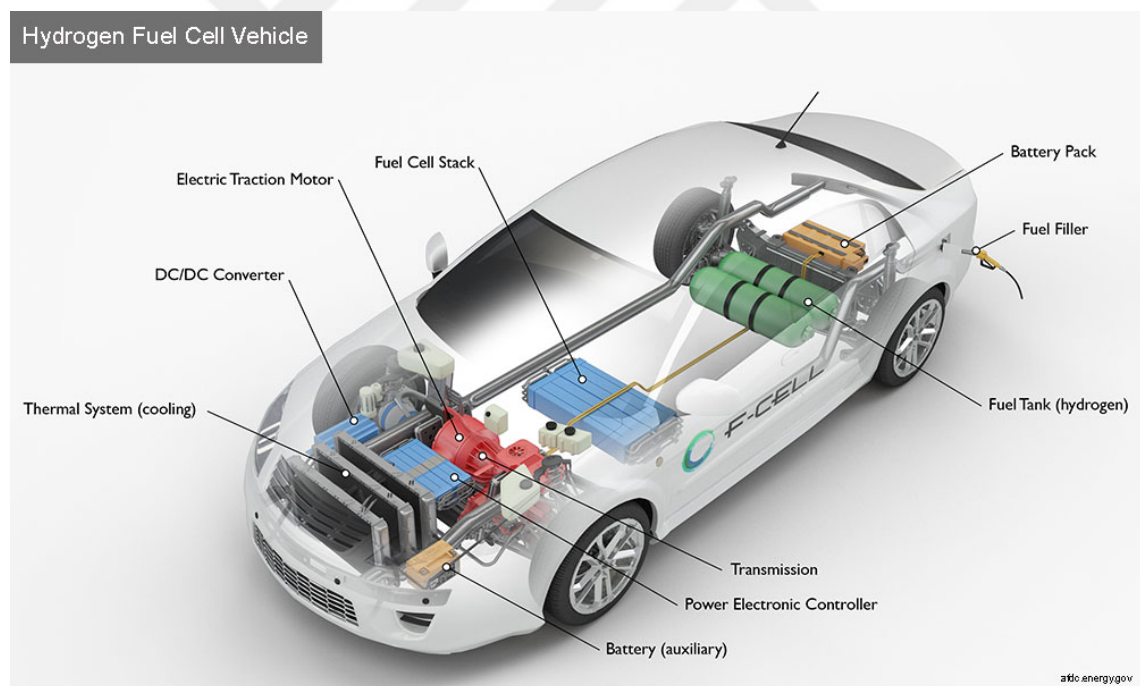


Figure 2.6: The Major Components of a Fuel Cell Electric Vehicle (FCEV) [53]

2.2.1.4 Plug-in Hybrid Electric Vehicle (PHEV)

In HEVs, regenerative braking and ICE supply energy back to the batteries. In addition to the same construction structure in HEVs, PHEVs have a larger battery pack and ability to charge the battery pack from outside. The major components of a typical PHEV are

Electric motor, power electronics unit, battery, cooling system, onboard charger, additional charging port to charge large battery packs, transmission lines, ICE, and gasoline fuel tank. Some of the PHEV models produced are Chevrolet Volt, Ford Fusion, Audi A3 E-Tron, BMW i8 and Kia Optima.

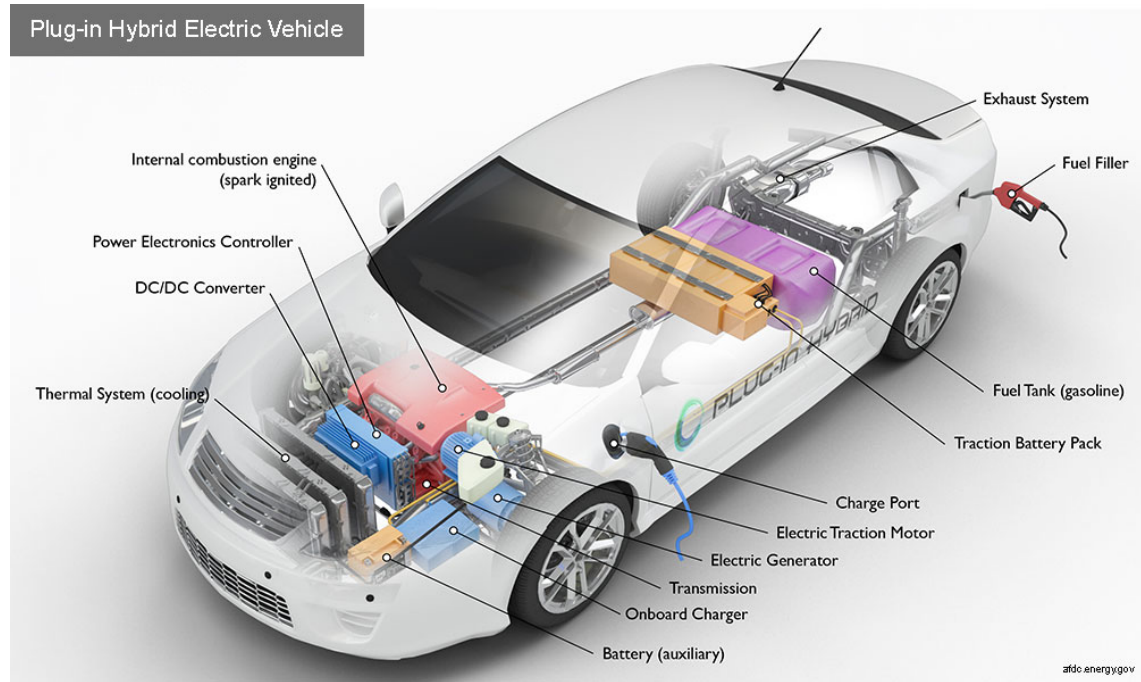


Figure 2.7: The Major Components of a Plug-in Hybrid Electric Vehicle (PHEV) [53]

2.2.2 Electric Machine Technology in Electric Vehicle (EV)s

Instead of having a single type of in-direct engine-wheel configuration like in ICE vehicles, EVs use both motor-shaft-wheel and direct motor-wheel drive configuration. Each configuration has advantages and disadvantages. In in-direct motor-wheel configuration, wheels are driven by the electric motor via differential gearbox. Thus, the number of mechanical parts, due to the differential gearbox, are higher than motor-wheel (in-wheel, hub motor) drive configuration. On the contrary, in-wheel drive configuration allows the vehicle to have increased space. Also, decreasing the number of mechanical parts on the vehicle gives advantages of more reliable and stable vehicle design. Furthermore, in-wheel drive configuration allows the vehicle control unit to control each wheel independently, thus giving flexible torque-speed control over each motor. However, powering each wheel via different motors can cause safety problems during failure [66]. Researchers have mainly focused on in-wheel drive configuration to overcome safety issues and to design high-torque, high-efficiency in-wheel drive electric motors for EVs [67–69].



Figure 2.8: Example of In-Wheel Electric Machine

The majority of companies that produce EVs use centralized single and double motor driven power-trains. Main idea behind the centralized manner is to place a differential to control the torque of the wheels. In single motor power-trains, wheels are powered by a single electric machine via differential. Single motor configuration leads to the compactness of power-train design. Less components are required to build a single motor power-train configuration. However, Single motor driven power-trains are not suitable for high power. Some EV models that contain single motor driven power-trains are Tesla Polestar 2, and Cupra Born. In dual motors power-trains, wheels are powered by two electric machines via two different differentials. Electric machines and differentials are separated as Front and Rear electric machines/differentials. Dual motors power-train can supply more horsepower (HP) and perform in higher efficiency range. Separation in the differentials give EV the ability to control both wheel pairs independently. Thus, dual motors power-trains can achieve more stable control in traction. Stable control impacts the effectiveness of the overall traction system. Some EV models that contain dual motor driven power-trains are Tesla Model 3. Having dual motors and differentials cause an increase in prices. Moreover, independent control of wheel pairs causes a complexity on the overall system.

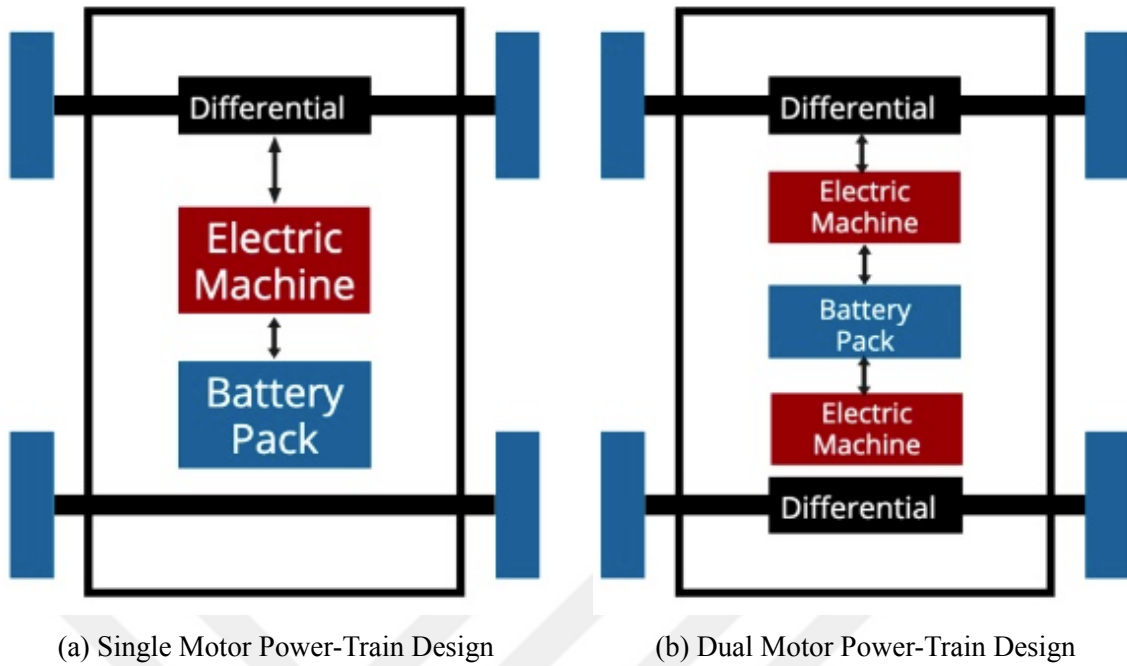


Figure 2.9: Power-Train Designs in Electric Vehicle (EV)s

Another power-train type is the Distributed Multi Motor Driven Power-Train (DM-MDP). DMMDPs mainly contain three or four electric machines. In triple motors configuration, EVs have two in-wheel and one centralized electric machine with a differential gearbox. In four motors configuration, EVs have four different in-wheel electric machines without any differential gearbox. Both configurations allow the all wheel driving architecture. Thus, handling stability increases proportionally. Electric machines can supply different positive/negative torque to each wheel. However, due to the number of electric machines and control units, both configurations are relatively more costly than centralized power-trains [70].

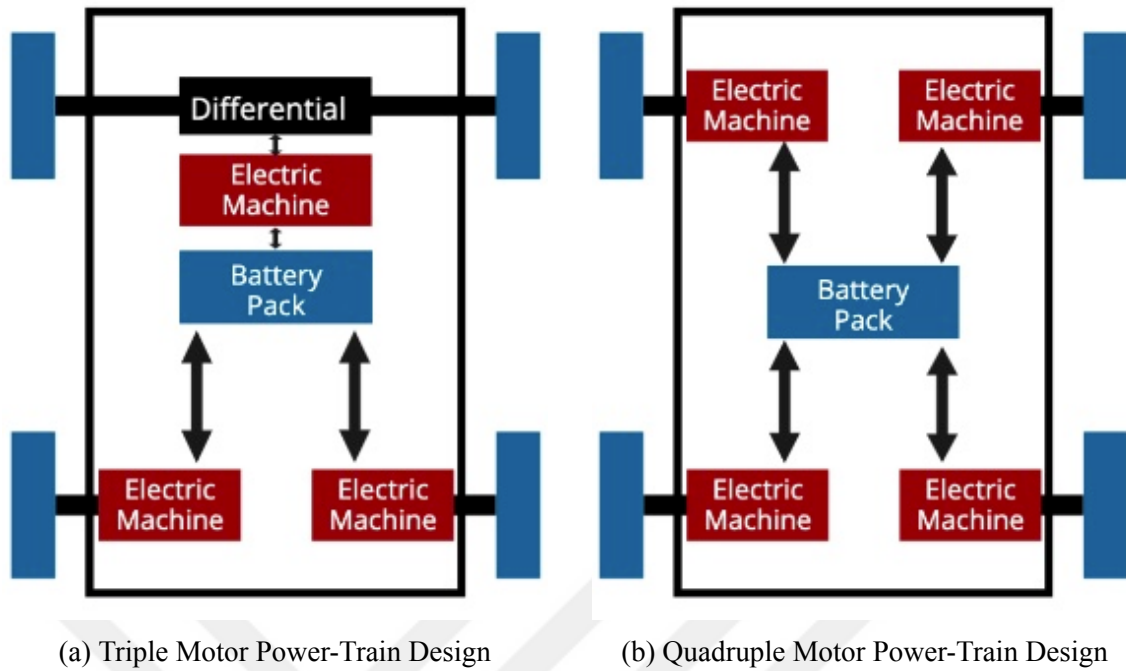


Figure 2.10: Power-Train Designs in Electric Vehicle (EV)s (DMMDP)

Over the last decades, several electric machine types have been proposed in the literature. Most proposed electric machine types in literature are Brushless Direct Current (BLDC) machines, Induction Machine (IM)s, Permanent Magnet Synchronous Machine (PMSM)s, and Switch Reluctance Machine (SRM)s. All different types of electric machines, used in EVs, offer different advantages and disadvantages.

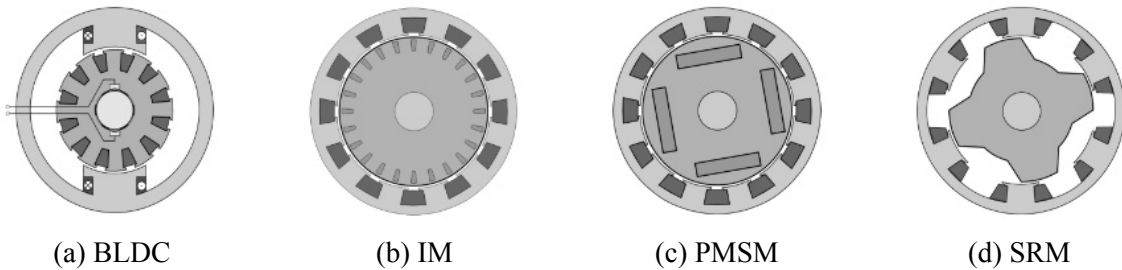


Figure 2.11: Electric Machine (EM)s Used in Electric Vehicle (EV)s

Development in the power electronics made the Brushless Direct Current (BLDC)s more attractive technology for EVs. Researchers have started to frequently investigate the BLDCs. Especially for EVs, BLDCs are strongly convenient electric machine technology. BLDCs have been recently adopted by EV manufacturers, due to high efficiency, high starting and continuous torque, better torque-speed curve, and less maintenance [71–74]. High demand for BLDCs forced manufacturers to create several designs, such as slotted and slotless BLDC [75]. General stator-rotor pair technique applies to BLDCs as well [71].

Stator core does not exist on the slotless BLDCs [76]. Advantages of BLDC machines are high torque density, lossless rotor copper, small in size and lighter, high power output capability, and reliable design. Disadvantages of BLDC machines are high cost due to Permanent Magnet (PM) rare earth material, high torque ripple, narrow range of constant power supply [70]. In the literature, to control a BLDC machine, several techniques such as Modified Pulse Width Modulation (PMW) control [77], current control [78], Model Predictive Control (MPC) [79], Direct Torque Control (DTC) [80–82], Field Oriented Control (FOC) [83–85] have been proposed. Some EV models that use BLDC are Toyota Prius.

Induction Machine (IM)s have been used in different applications such as lifting systems [86,87], cranes [88,89], pumps [90,91], vacuum cleaners [92] and etc. Improvement and development in the IMs design lead IM to be used in EVs as well. All around the world EV manufacturers use IMs as a propulsion system due to reliable and robust performance, low production cost and maintenance costs [93,94]. Advantages of IMs are high dynamic response and ease control of torque and speed, providing high peak torque and requiring almost no maintenance. Disadvantages of IMs are relatively low efficiency and low starting torque than other electric machines and high copper losses [70]. In the literature, to control a IM, several techniques such as voltage control [95], frequency control [96], electronically pole changing [97], slip power recovery [98], and direct torque control [99] have been proposed. Some EV models that use IM are Tesla Model X, Tesla Model S, Toyota Rav 4, and General Motors EV1.

Permanent Magnet Synchronous Machine (PMSM)s have been used in different applications such as air conditioner [100], refrigerator [101], AC compressors, washing machines [102], electric power steering in vehicles [103–105], robotics-aerospace and etc [106, 107]. PMSMs have similar working principles to IMs. Interaction on rotating and constant magnetic field between stator and rotor, turns the rotor of PMSMs. Lighter weight than IMs, more efficient operation performance and compactness, various control techniques make PMSMs an attractive choice for EV manufacturers. Ease and cleanliness in the design and manufacturing, low Electromagnetic Interference (EMI), safe and reliable operation make PMSMs a suitable electric machine type for EV applications. Advantages of PMSMs, as described, are high efficiency, high power density and compact/small structure. Disadvantages of PMSMs are costly magnetization material, demagnetization issues, and high iron loss during high speed operation in in-wheel drive applications [70]. In the literature, to control a PMSM, several techniques such as Field Oriented Control (FOC) [108, 109], direct torque control [110, 111] and sensorless MPC [112, 113] have been proposed. Some EV models that use PMSM are Nissan Leaf and Toyota Prius.

Switch Reluctance Machine (SRM)s have been used in different applications such

as pumps [114], washing machines [115] and etc. Most EV manufacturers use SRM in HEVs. Advantages of SRMs are fault tolerance, ease in control, rigid construction, low construction costs, and high speed. Disadvantages of SRMs are noisy operation, high ripple on torque, operation issues like vibration due to construction and low torque density [70]. In the literature, to control a SRM, several techniques such as current control [116, 117], instantaneous torque control [118, 119] have been proposed. Some EV models that use SRM are Lucas Chloride and Land Rover (PHEV).

2.2.3 Charging Technologies in Electric Vehicle (EV)s

Charging the battery in an EV is a necessary process to take advantage of a rechargeable (e.g. Li-ion) battery pack. Charging an EV can be defined as using a charger device, which converts Alternative Current (AC) grid energy to Direct Current (DC) battery energy, to charge a primary energy source (or battery pack) in an EV. Several charging station types are available such as home and public stations. Home type chargers mainly allow EV users to charge an EV in a few kW. In contrast, public stations generally have high power chargers available which boost an EV charging duration. All type chargers follow a common electrical procedure. Firstly, AC grid energy must be converted to DC energy. Then, converted DC energy must be controlled and regulated to satisfy EV and battery pack requirements. The charging cable establishes a connection between the EV power control unit and the charger control unit and is responsible for energy flow. The EV power control unit and the charger control unit are responsible to manage the energy flow, regulate the energy flow, and terminate the energy flow through control signals.

Conductive charging, inductive charging and battery replacement options are available to EV users. However, due to heating issues, inductive charging technique is not preferred [120]. Moreover, due to cost issues, battery replacement technique is not preferred. Nowadays, conductive charging techniques are the most suitable and preferred EV charging method all around the world.

Three charging levels are available in the conductive charging technology.

Level 1 charging (a.k.a “Home Charging”) is a method that uses an AC grid energy to charge an EV. Level 1 charging requires a single-phase grounded outlet. Thus, the power delivered by Level 1 charging is limited to 2.9 kW (220V, 13A). On-board charger on the vehicle manages the electric flow and charging process. Due to quite limited power, Level 1 charging is the slowest charging method [121]. The required connection cords are NEMA 5-15 and SAE J1772 [122].

Level 2 charging is a method that uses single-phase or three-phase AC grid energy to charge an EV. Level 2 charging requires a single-phase or a three-phase outlet. Thus, the power delivered by Level 2 charging is limited to 3.5 kW (220V, 16A) or 7.0 kW (220V, 32A). On-board charger and/or smart charger device manages the electric flow and

charging process. Level 2 charging is a medium-speed charging method and faster than Level 1 charging [121]. The required connection cords are SAE J1772 in both ends [122].

Level 3 charging, on the other hand, is DC fast charging. DC fast chargers are mainly located in gas stations or dedicated charging stations. DC fast charging enables rapid charging that takes less than 2 hours to fully charge a typical EV. DC fast charging power may be between 50 kW and up to 350 kW. The required connection cords are CCS and CHAdeMO. Some manufacturers, like Tesla, design their own connection cords [122].

2.2.4 Electric Vehicle (EV) Integration into Grid

Charging/discharging capability of EVs have made EVs as portable energy sources. Vehicle-to-Home (V2H), Vehicle-to-Vehicle (V2V) and Vehicle-To-Grid (V2G) applications have been the main interest of researchers to overcome issues that come with EVs. The literature proposes that EVs have become an ESS to buffer the main grid [123]. Thus, supporting the intermittent renewable energy source has become another flexible feature of EVs. Integration of EVs into the grid enables EV owners to send energy to the grid. Thus Vehicle-To-Grid (V2G) integration reduces the energy cost and supports the EV owner during peak energy demands [124]. Furthermore, Vehicle-To-Grid (V2G) integration acts as a backup energy source during absence of renewable energy production [125].

In the Vehicle-to-Home (V2H) concept, the vehicle provides energy to home outlets. For this purpose, the Electric Vehicle (EV) acts as a backup energy source and controllable load. To reduce the energy cost, Vehicle-to-Home (V2H) concept can help home appliances to shift their usage duration. At peak price time, the Electric Vehicle (EV) can sell excess energy back to the grid to reduce energy cost. With the same logic, the Electric Vehicle (EV) can buy required charging energy during the off-peak price time. The Electric Vehicle (EV) helps with energy management of homes in the Vehicle-to-Home (V2H) concept [126].

In the Vehicle-to-Vehicle (V2V) concept, the vehicle provides energy to other vehicles. For this purpose, the Electric Vehicle (EV) acts as an energy source to other vehicles. Multiple Electric Vehicle (EV)s exchange energy to increase charging/discharging efficiency. The Vehicle-to-Vehicle (V2V) concept isolates the vehicles and establishes a Vehicle-to-Vehicle (V2V) network between vehicles. Charging costs can be reduced via local energy exchanging. Moreover, linking the vehicles and grid smooths the grid operation and decreases the grid energy demand [126].

In the Vehicle-To-Grid (V2G) concept, the vehicle provides energy to the grid. The Electric Vehicle (EV) acts as a backup energy source to support the grid during peak-load demands. Furthermore, the Electric Vehicle (EV) is now a controllable load which can be charged/discharged. At peak price time, the Electric Vehicle (EV) can sell excess energy back to the grid to reduce energy cost. With the same logic, the Electric Vehicle

(EV) can buy required charging energy during the off-peak price time. Also, the Vehicle-To-Grid (V2G) concept stabilizes the grid operations [126]. In the literature, researchers have proposed that the Vehicle-To-Grid (V2G) concept can provide frequency regulation and stabilize the grid network [127]. Especially, during the absence of renewable energy production, Electric vehicles can regulate the frequency of the grid [128].

2.3 P2P ENERGY TRADING

In the last decade, small and medium scale distributed energy sources have excessively increased with the help of electric vehicles, energy storage systems and bidirectional energy flow [129]. To manage the energy demand effectively, home-scale energy producers (or proactive consumers - a.k.a. prosumers) must provide excess energy either back into the grid or into other consumers. Thus, home-scale energy producers, via renewable energy sources, have started to participate in energy trading. Prosumers can participate in Peer-To-Peer (P2P) energy trading by selling excess energy to fulfill other consumers energy demands [130]. However, selling excess energy as negative energy back into the grid can make a prosumer a participant in P2P energy trading [131]. Meanwhile, the grid benefits from the P2P energy trading concept by improving grid reliability [132], reducing investment and maintenance costs [133], and reducing peak-energy demand during peak hours [134].

Distributed network contains a sharing energy source network that P2P trades participate in. Peers who demand energy can directly access the selling energy in the network by other prosumers. Responsibility of a prosumer is to provide a service and offer a bid to consumers [135]. Researchers divide P2P energy trading into two layers. First layer is called the virtual layer. In the virtual layer, Offering mechanism, information exchanges, security constraints, and exchange orders happen via centralized or decentralized manners. Second layer is called the physical layer. In the physical layer, according to the requirement in the virtual layer, all the physical energy transactions, selling, buying and providing excess energy back into the grid, happen [136].

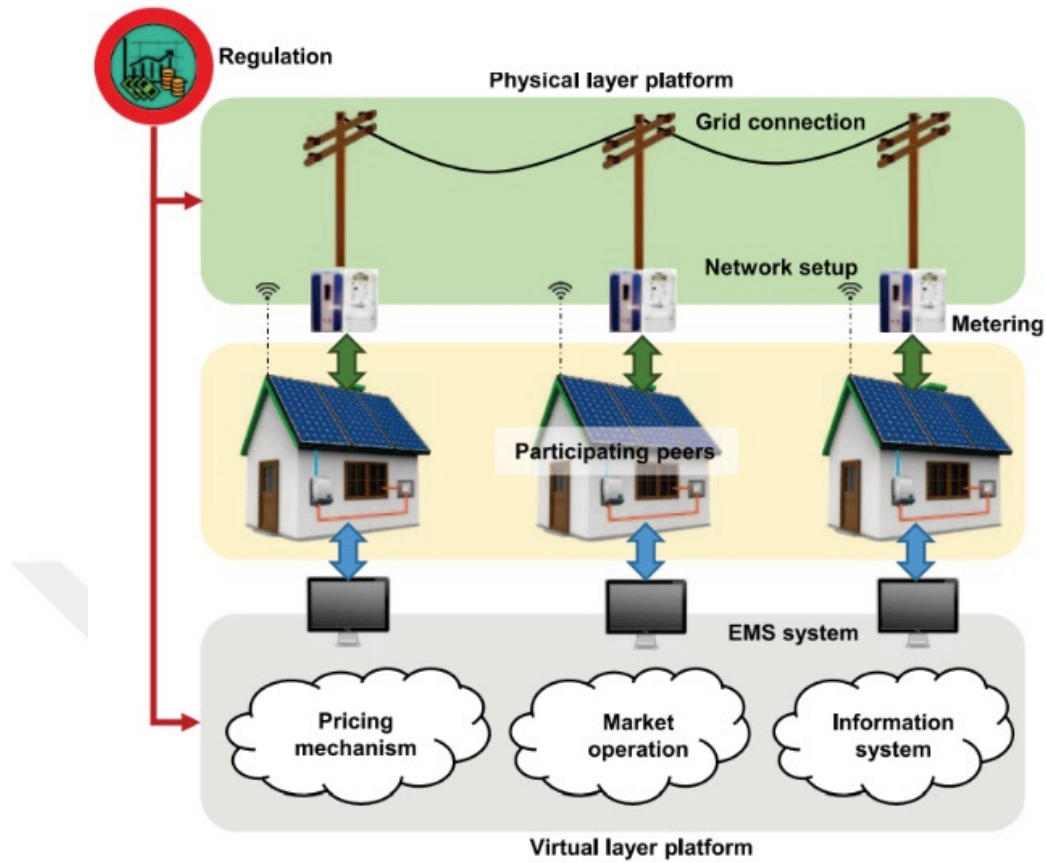


Figure 2.12: Visual Representation of Virtual and Physical Layers

The key components of a virtual layer are information system, market operation, pricing mechanism and energy management system. Information system, as described earlier, manages the access of participants, monitors the market operation systems and is responsible for virtual layer's security. Market operation ensures an efficient energy transaction by defining the bidding and payment rules. Pricing mechanism sets the energy transaction prices to fulfill the cost satisfaction demands of prosumers and consumers. Energy management system secures the transaction of energy between peers. Furthermore, EMS participates in bidding strategy on behalf of the prosumer and consumer [129].

Physical layer has grid connection, metering and communication infrastructure as key components. Grid connection enables prosumers to supply excess energy back into the grid. Moreover, grid connection enables consumers to provide demanded energy when energy trading is not possible. Thus, grid connection enables reliable energy trading. Metering in the P2P trading measures the bought and sold energy by peers. In addition to a regular meter, peers should have a transactive meter to log the transactions [136]. Bidirectional energy meter is required to measure sold energy back into the grid by a prosumer [129]. Communication infrastructure enables peers to communicate with other peers via structured, unstructured and hybrid communication structures [137].

Challenges of virtual layers in P2P energy trading are effectively reducing the cost of energy [138], balancing energy generation and load demand [138, 139], prosumers interest in energy trading [140, 141], defining a pricing mechanism [142, 143], elimination of uncertainties [144], establishing a secure transaction environment [145, 146].

On the other hand, challenges of physical layers in P2P energy trading are physical electric constraint [147], power losses during transactions [148] and network stability [149].

In the literature, different techniques such as auction theory, game theory and block chain technology have been proposed to create a common framework for P2P energy trading [129]. Game theory methods, such as cooperative-competitive Nash games, generalized Nash games, Stackelberg games have been developed to provide an optimal solution via both competitive and cooperative methods between peers [150–152]. Auction theory methods, such as single-double auction, have been developed to provide step-by-step methods to establish a stable interaction between peers [130, 153]. Block chain methods, such as Elecbay, ethereum, hyperledger, have been developed to provide secured and decentralized energy trading between peers [154, 155].

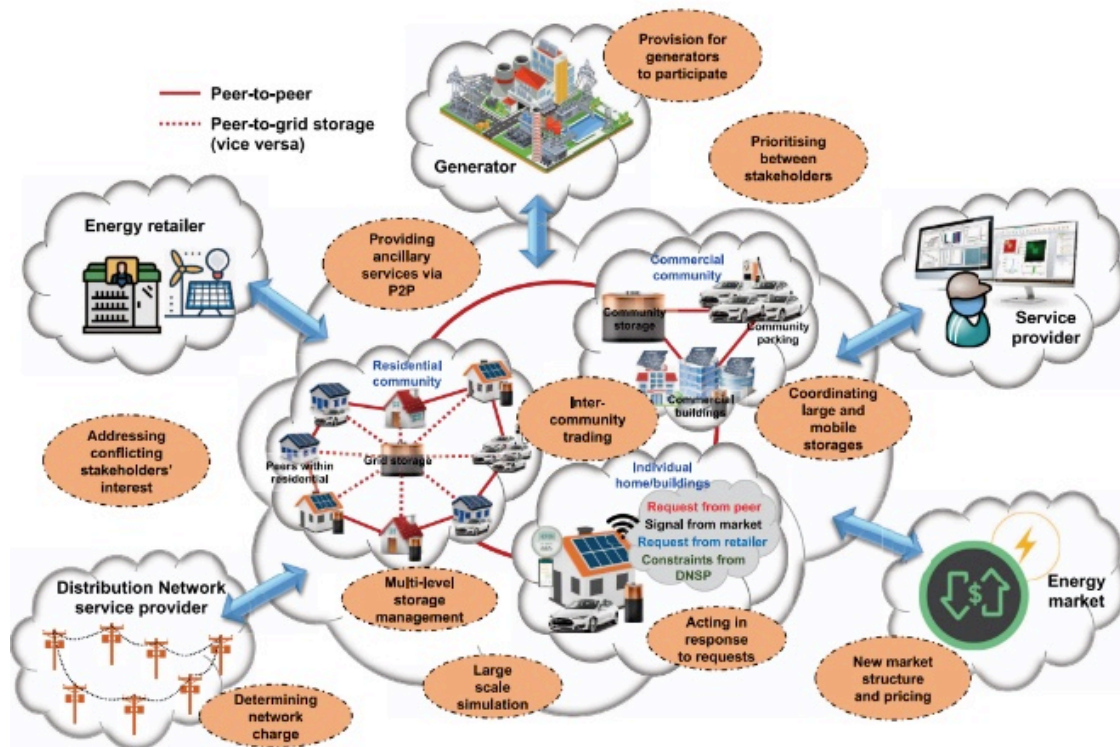


Figure 2.13: Overview look at the Peer-to-Peer Energy Trading Framework [129]

CHAPTER 3 OPTIMISATION TECHNIQUES & GAME THEORY

3.1 REINFORCEMENT LEARNING METHODS on OPTIMISATION

Reinforcement Learning (RL) is a sub-branch of Machine Learning (ML). The main concept behind the RL is to maximize a given numerical reward [156]. Maximizing a numerical reward happens to learn the environment with state-action constraints [157]. Reinforcement learning investigates, in a predefined environment, what action should be taken and how to map actions into different situations which refers to states in reinforcement learning [157]. Reinforcement learning shows similarities with human learning mechanisms [158]. Reinforcement learning was first proposed by Samuel in 1959. First type of RL was proposed to solve a checkers game to learn the value function, so actions could be guided via agents [159]. After Samuel's contribution, another value function iteration algorithm was proposed by Sutton in 1988. Sutton confirmed that the TD-Learning algorithm converges in the long run [157]. As an improvement on the TD-Learning algorithm, Watkins proposed another value function iteration algorithm in 1992 called Q-Learning [160]. After the Q-Learning, researchers and companies, such as Google, have accelerated the improvements on the RL and proposed different algorithms to solve Reinforcement Learning problems [158].

3.1.1 Components of Reinforcement Learning (RL)

In theory, Reinforcement Learning (RL) consists of several components to solve an optimization problem. Reinforcement Learning (RL) maximizes a given reward by learning the environment via the interaction between agent and environment.

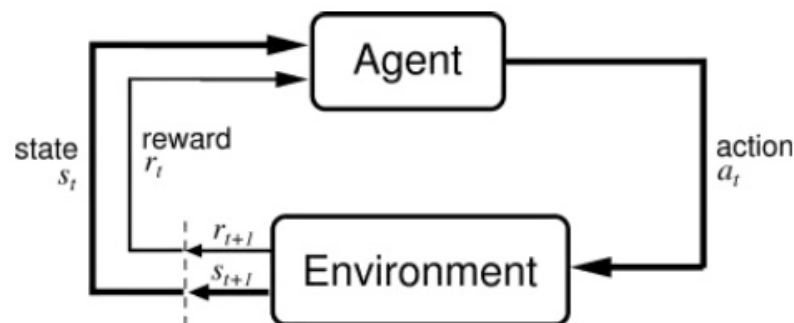


Figure 3.1: Components of Reinforcement Learning

Most cases, reinforcement learning algorithms are modeled with a Markov Decision Process (MDP) method. The Markov Decision Process consists of five-tuples [161]. Tuples are known as (S, A, P, R, r) .

1. S is the state of the environment. Set of S refers to all possible states in the environment.
2. A is the actions of the environment. Set of A refers to all possible actions that can be taken by an agent.
3. P is the probability transition matrix. P represents the probability of achieving the next state “ s ” after the agent takes an action “ a ” at the current state “ s ”.
4. R is the reward function of the environment. After the agent takes an action “ a ” at the current state “ s ”, the agent receives a reward at time “ t ”. This reward is represented by $R(a,s)$. Received reward directs the agent into the maximum profit state.
5. “ r ” is the discount factor on the reward R . The discount factor “ r ” determines the importance of the current reward and the future rewards.

The steps of a basic MDP is as follows [162]: The initial states of an agent, at time t , are determined from the environment. An action, from the set of “ A ”, chosen and executed by the agent. After the execution is held, next states, according to the probability transition matrix, are received. Furthermore, the agent receives a reward value from the predefined reward function R . The agent receives two reward values from the reward function. The first reward represents the current reward at time t . The second reward represents the future reward which is discounted by the discount factor “ r ”. After the $\langle S, A, P, R, r \rangle$ tuple is finished, the same tuple is executed with the next states. An example of a MDP tuple can be seen in below figure.

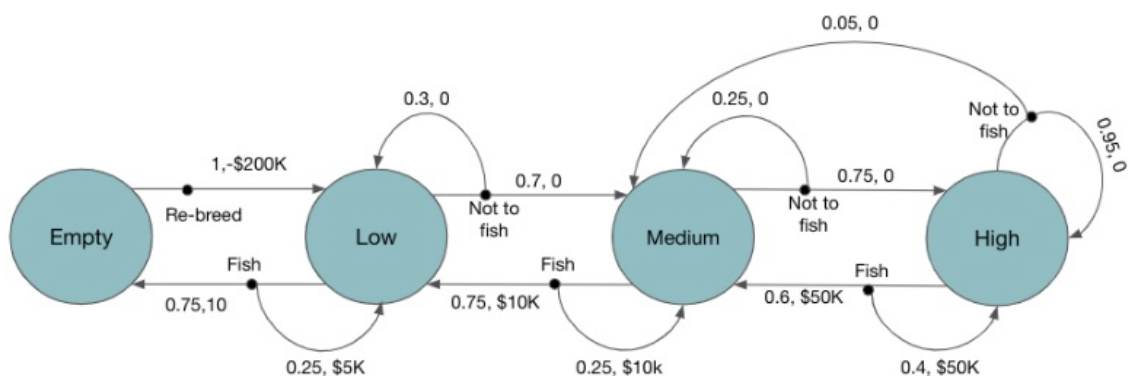


Figure 3.2: Markov Decision Process Example in Real World Applications [163]

In theory, in addition to the MDP tuples, Reinforcement learning consists of several other components. The first and the most fundamental component of the RL is an agent. In RL problems, an agent represents the learner. An agent makes an action decision from a given action space. An environment consists of all action and state spaces. Environment is the place where the agent learns by interacting with it. Action, as described previously on MDP, is a set of actions. The agent chooses an action from the action space and executes the chosen action. State is described as the state of the agent in the given environment. State can be numeric or any other variable types that describe the situation the agent deals with. As a consequence to agent's action selection, the agent receives a reward from the environment. Reward could be positive or negative according to the rules set by the environment. One of the most important components of the RL is policy. Policy is described as the control strategy that agent develops during interactions. Policy represents a network that maps states and actions. On the other hand, similar to the Reward, value is the expected long-term reward with predefined discount factor " γ " is applied. Value is calculated in the current state with current policy. Q-value, also known as action-value, is the long-term reward represented with current state, taking action and current policy. Difference between Value and Q-Value is that Q-Value also takes the action into consideration.

3.1.2 Exploration & Exploitation Dilemma

Performance of optimization algorithms relies on the parameter configuration. Configuring learning parameters changes bias and operational performance of the algorithm. In order to maximize the efficiency of the optimization process, parameters must be configured properly. Configuration of parameters is called algorithm configuration [164, 165]. With the increasing use of Machine Learning algorithms, algorithm configuration has become more important than ever. Researchers have developed different techniques for algorithm configurations [166]. The beginning of the search, the agent is not fully informed about the environment and states. Agents must learn the environment by exploring it. However, exploring the whole state of the environment can be a costly and time wasting process. To overcome these issues, agents should take the initiative to reduce the complexity of the process. With the previously explored data, agents must try to increase the expected reward by using the gained information. In the simplest term, exploration refers to collecting states of the environment, without trying to maximize the expected reward. In exploration, agents try to understand the environment and associate the collected information to individual states. On the contrary, exploitation refers to using the previously collected data to further increase the expected reward. In exploitation, agents use the collected information instead of collecting new information about the environment. The ratio between exploration and exploitation determines the quality of the optimization process. One should make a decision of this ratio by considering the needs and the complexity of

the environment.

Most of the reinforcement learning researchers have adapted an Epsilon-Greedy exploration approach to get the most available benefits of the environment [167]. The Epsilon-Greedy method combines the greedy and random searching methods to increase the efficiency of both methods. In a greedy method, gradually, agents will try to take the best action available at the moment. Although gaining the maximum reward from the best action seems the perfect solution, this could lead to insufficient environment exploration [168]. To overcome this issue, in a random exploration method, agents will try to take a random action from the action space. As the agents explore the states, combination of epsilon and random methods will give the sufficient and efficient environment exploration [169].

3.1.3 Model-Free & Model-Based Learning Processes

Reinforcement learning can be divided into two separate categories according to usage of model: Model free and Model based learning.

In the model-free RL, the environment does not contain any transition probability matrix. In addition to not having any transition probability matrix, model-free RL also does not have rewards. Model-free algorithms try to learn the environment based on the agent's actions through exploration. Some algorithms include Q-Learning, Policy Gradient, etc. Model-free algorithms execute the same action more than once to adjust the policy for maximum possible reward outcomes.

In Q-Learning, agents try to learn the optimal policy to map the action to the states. Q in the Q-Learning stands for the policy which contains action and state at the current time. Q is represented as $Q(s, a)$. Based on the action taken, the agent will get a reward from the environment. Mapping the action and the state serves to build a Q-table to store $Q(s, a)$. The main drawback of Q-Learning is the need for high memory to store all state-action pairs. To overcome this issue, researchers have proposed deep Q-Learning. In deep Q-Learning, algorithms will predict the $Q(s, a)$ with the help of a Neural network. Prediction of the $Q(s, a)$ prevents the storing of $Q(s, a)$, thus reducing the memory needs [170]. Deep Q-Learning has a wide variety of application areas in the literature. Hoffman and et al. have proposed a method to improve the muting pattern to improve cell-edge users throughput [171]. Chouiekh and et al. have proposed a method to control the traffic lights in the intersection to improve the efficiency of intersection usage [172]. Aikins and et al. have proposed a method to prevent cyberattacks to improve the safety of self-driving vehicles [173].

In Policy Gradient, policies are identified as $\pi Q(a|s)$. The purpose of the policy gradient is to optimize the Q value with gradient ascent. Such optimization can be achieved via maximization of the local approximation [170]. Policy gradient updates the policies step by step to optimize the target function. The main drawback of the Policy Gradient

is obtaining the optimal learning parameters. Non-optimal learning parameters can cause inefficient learning rate. To overcome this learning issue, Schulman and et al. proposed Trusted Region Policy Optimization (TRPO) [174]. Main idea behind the TRPO is to obtain constant increase or stable reward values. Researchers have different application areas such as Atari games, robotics, etc [170].

In the model-based RL, the environment is built upon a given model that includes transition probability matrix and reward. Predefined models can give the agent an intuition about the environment. Therefore, the agent can make a better picture about the environment and thus forecast the actions. Model-Based RL can obtain optimal control, with given restrictions, quicker than the model-free RL. The main idea behind the model-based RL is to use pre-sampled data to train the policy. Training is held in the virtual environment. The most known example of model-based RL is AlphaGo. In the AlphaGo scenario, an agent trained with 30 million pre-collected data from the professional Go player. Model-Based RL has a wide variety of application areas in the literature. Chen and et al. have proposed a control strategy based on learning-based algorithms to control the Heating Ventilation and Air Conditioning (HVAC) systems [175]. Park and et al. have proposed a control method to study a wireless channel access method in limited spectral resources [176]. Xiao and et al. have proposed a control algorithm based on a model to fulfill the passengers requirements in the public transit systems [177]. Some further applications, such as filling the missing data in the robotics environment to ensure the robots movement precision, have been proposed in the literature [178].

3.1.4 On-Policy & Off-Policy Learning Strategies

In Reinforcement Learning algorithms, policy is defined as the actions suggested by the agent which are considered to be the optimal actions for a specific state. When the whole state-action pairs are considered, policy is a probability distribution that an agent can take action from an action space that maximizes the reward. The policy is denoted by π in the mathematical term. Policies are divided into two types that are update policy and behavior policy, respectively. Update policy is defined as how an agent learns the optimal policy to find the optimal state-action pair per each state and action. On the other hand, behavior policy is defined as how an agent behaves in a particular state.

The difference between the on-policy and off-policy strategies lies on the idea of differentiation between update and behavior policy. In the on-policy algorithms, the same policy for update and behavior policies is used. Thus, an agent tries to improve the update and behavior policies simultaneously. This means that an agent chooses the optimal policy and uses the same policy to behave in the environment. Therefore, $Q(s, a)$ function is learned from the action that chosen from the optimal policy $\pi(a|s)$. The update function

used in on-policy algorithms is as follows:

$$Q(s, a) \leftarrow Q(s, a) + \alpha(r + \gamma Q(s', a') - Q(s, a))$$

where a' is the action taken by the policy π .

On the contrary, in the off-policy algorithms, update and behavior policies are different. Thus, an agent tries to improve the update policy from the behavior policies. In off-policy algorithms, $Q(s, a)$ function is learned from different actions that are not the same as taken at time t , such as random actions. The update function used in off-policy algorithms is as follows:

$$Q(s, a) \leftarrow Q(s, a) + \alpha(r + \gamma \max_{a'} Q(s', a') - Q(s, a))$$

where a' are all the actions possible in state s' .

SARSA algorithm is a type of an on-line policy learning algorithm. SARSA has a wide variety of application areas in the literature. Fang and et al. have proposed a learning method to solve the two-dimensional irregular pieces nesting problem for industries such as glass, and metal cutting etc [179]. Han and et al. have proposed a learning method to reduce the average transmission power in the communication area and improve the data throughput [180]. Arving and et al. have proposed a new learning method by modifying the SARSA with Multi-Layer Perceptron to avoid obstacles for autonomous vehicles [181]. Apart from the physical applications, Almalki and et al. have proposed a learning method to improve a user's playing snake habits to achieve improvement on the moving capability of snakes [182]. Mohan and et al. have proposed a method to overcome trajectory planning and obstacle avoidance of autonomous vehicles by combining on-policy and off-policy algorithms [183].

Q-Learning algorithm is a type of an off-line policy learning algorithm. Q-Learning has a wide variety of application areas in the literature. Cardarilli and et al. have proposed a novel algorithm called MaxReset Q-Learning (MR Q-Learning) to solve the inefficient hardware implementation such as RAM [184]. Kim and et al. have proposed a Q-Learning algorithm that collects real-time data in robotics environments to improve the mobile robot's navigation and obstacle avoidance abilities [185]. Similar to this, Pambudi and et al. have proposed a fuzzy logic integrated Q-Learning algorithm to solve the same navigation and obstacle problems [186]. Furthermore, Kim and et al. have proposed a unique Q-learning method using real-time data, rather than stored data, to improve an autonomous vehicle's path planning [187]. Healthcare systems are largely entitled to the domain in which machine learning algorithms are being adopted to improve their effectiveness and efficiency. As a result of the COVID-19 pandemic, it turned out that health-

care systems were far behind in today's technology. Raina and et al. have proposed a learning method to solve the healthcare systems issues by implementing a real-time Q-Learning algorithm [188]. Similar to on-line policy learning algorithms, off-line learning algorithms have diverse applications in the communication area. Liman and et al. have proposed a multi-agent deep Q-learning algorithm to solve mesh points issues that occur on the wireless backhaul networks [189]. Increasing demand on the defense industry has led to a high demand in reinforcement learning applications such as Unmanned Air Vehicles (UAV). Gao and et al. have proposed a method to improve the path planning of a UAV and increase an UAV's awareness for possible risks during the flights [190].

3.1.5 Reinforcement Learning (RL) Algorithms

3.1.5.1 Monte Carlo

Monte Carlo is a control algorithm that uses dynamic programming to find an optimal solution for an optimization problem. Monte Carlo is a model-free type of reinforcement learning framework. In the Monte Carlo algorithm, the agent collects sample data from its experience so that it can learn. The agent learns the state value $V(s)$ with the policy π by experiencing average return from the whole sampled episodes. The Monte Carlo algorithm can only be implemented to episodic environments where the environment has a stop signal.

Monte Carlo has relatively narrow application areas in Reinforcement Learning problems due to its basic and insufficient optimization structure. Wu and et al. have proposed a Monte Carlo based learning RL environment to solve the uncertainties between exploration and exploitation in learning algorithms by using Bayesian method [191]. He and et al. have proposed a Monte Carlo Tree Search based learning RL environment to improve automatic circuit routing on VLSI circuit designs [192]. Adsawinnawanawa and et al. have proposed a novel Monte Carlo algorithm called The Sharing of Similar Knowledge on Monte Carlo (SSKMC) algorithm to increase a trader's net worth on Cryptocurrency trading platforms such as Bitcoin [193]. In the power electronic and energy applications area, Shuai and et al. have proposed a Monte Carlo Tree Search algorithm to schedule a residential micro grid usage for optimum renewable energy trade off without the need of accurately forecasting the future renewable power generations [194]. Researchers, Li and et al., Huang and et al., and Meng et al. have proposed a Monte Carlo Tree Search algorithm to optimize a user's move in Go and Chess games [195–197].

Algorithm 1 Monte Carlo

```
1: procedure Monte Carlo
2:    $Q(s, a) \leftarrow 0, Returns(s, a) \leftarrow 0$  for all  $s \in S, a \in A$ 
3:    $\epsilon \leftarrow 1, k \leftarrow 1$ 
4:   loop
5:     Sample  $k_{th}$  episode  $(s_{k1}, a_{k1}, r_{k1}, s_{k2}, \dots, s_T)$  under policy  $\pi$ 
6:     for  $t = 1, \dots, T$  do
7:       if First visit to  $(s, a)$  in episode  $k$  then then
8:         Append  $\sum_{j=t}^T r_{kj}$  to  $Returns(s_t, a_t)$ 
9:          $Q(s_t, a_t) \leftarrow average(Returns(s_t, a_t))$ 
10:      end if
11:    end for
12:     $k \leftarrow k + 1, \epsilon = 1/k$ 
13:     $\pi_k = \epsilon - greedy$  with respect to  $Q$  (policy improvement)
14:  end loop
15: return  $Q, \pi$ 
16: end procedure
```

3.1.5.2 Q-Learning

Q-Learning is a reinforcement learning algorithm to find an optimal solution for an optimization problem. Q-Learning is a model-free and off-policy type of reinforcement learning framework. In Q-Learning, finding the optimal action in a particular state is the purpose. Due to off-policy property, a Q-Learning agent takes random actions to maximize the reward and does not follow a policy. Table called Q-Table is created after every taken step that pairs an action into a state. Until all criteria are met, the agent updates the Q-Table after each step. In the test process, the agent takes the best action from the Q-Table that maximizes the reward that was created during the training process.

Q-Learning has a wide variety of application areas in Reinforcement Learning problems due to its simple and efficient learning process. As mentioned earlier in the “On-Policy & Off-Policy Learning Strategies” chapter, Q-Learning has been adopted and integrated into robotics environments, such as robot motion planning, autonomous vehicles, and etc. Besides the physical applications, Q-Learning has been used in gaming, card playing, healthcare systems, advertising, finance, etc.

Algorithm 2 Q-Learning

```
1: procedure Q-Learning
2:    $Q(S, A; \Phi) \leftarrow \text{random } \Phi$ 
3:   for each training time step do
4:     Get the initial observation  $S$  from the environment
5:     For the current observation  $S$ , select a random action  $A$  with probability  $\epsilon$ .
6:     Otherwise, select the action for which the critic value function is greatest.
7:      $A \leftarrow \text{argmax} Q(S, A; \Phi)$ 
8:     Execute action  $A$ . Observe the reward  $R$  and next observation  $S'$ .
9:     if  $S'_i$  is terminal state then
10:        $y_i \leftarrow R_i$ 
11:     else
12:        $y_i \leftarrow R + \gamma \max Q(S', A; \Phi)$ 
13:     end if
14:     Compute the difference  $\Delta Q$  between the value function target and the current
        $Q(S, A, \Phi)_{\text{value}}$ 
15:      $\Delta Q \leftarrow y - Q(S, A; \Phi)$ 
16:     Update the critic using the learning rate  $\alpha$ 
17:     if table based critics then
18:        $Q(S, A) \leftarrow Q(S, A; \Phi + \alpha \Delta Q)$ 
19:     else
20:        $\Delta \Phi \leftarrow \frac{1}{2} \nabla_{\Phi} (\Delta Q)^2$ 
21:        $\Phi \leftarrow \Phi + \alpha \Delta \Phi$ 
22:     end if
23:     Set the observation  $S$  to  $S'$ .
24:   end for
25: return  $Q$ 
26: end procedure
```

3.1.5.3 SARSA

SARSA is a reinforcement learning algorithm to find an optimal solution for an optimization problem. SARSA is a model-free and on-policy type of reinforcement learning framework. In SARSA, the agent follows a policy (π) to find the optimal state-action pair.

SARSA has a wide variety of application areas in Reinforcement Learning problems due to its simple and efficient learning process like Q-Learning. As mentioned earlier in the “On-Policy & Off-Policy Learning Strategies” chapter, SARSA has been adopted and integrated into robotics environments, such as robot motion planning, autonomous

vehicles, and etc. Besides the physical applications, SARSA has been used in gaming, card playing, healthcare systems, advertising, finance, etc.

Algorithm 3 SARSA

```

1: procedure SARSA
2:    $Q(S, A; \Phi) \leftarrow \text{random } \Phi$ 
3:   for each training time step do
4:     Get the initial observation  $S$  from the environment
5:     For the current observation  $S$ , select a random action  $A$  with probability  $\epsilon$ .
6:     Otherwise, select the action for which the critic value function is greatest.
7:      $A \leftarrow \text{argmax} Q(S, A; \Phi)$ 
8:     Execute action  $A$ . Observe the reward  $R$  and next observation  $S'$ .
9:     For the current observation  $S'$ , select a random action  $A'$  with probability  $\epsilon$ .
10:    Otherwise, select the action for which the critic value function is greatest.
11:     $A' \leftarrow \text{argmax} Q(S', A'; \Phi)$ 
12:    if  $S'_i$  is terminal state then
13:       $y_i \leftarrow R_i$ 
14:    else
15:       $y_i \leftarrow R + \gamma \max Q(S', A; \Phi)$ 
16:    end if
17:    Compute the difference  $\Delta Q$  between the value function target and the current
       $Q(S, A, \Phi)$  value
18:     $\Delta Q \leftarrow y - Q(S, A; \Phi)$ 
19:    Update the critic using the learning rate  $\alpha$ 
20:    if table based critics then
21:       $Q(S, A) \leftarrow Q(S, A; \Phi + \alpha \Delta Q)$ 
22:    else
23:       $\Delta \Phi \leftarrow \frac{1}{2} \nabla_{\Phi} (\Delta Q)^2$ 
24:       $\Phi \leftarrow \Phi + \alpha \Delta \Phi$ 
25:    end if
26:    Set the observation  $S \leftarrow S'$ .
27:    Set the action  $A \leftarrow A'$ .
28:  end for
29: return  $Q$ 
30: end procedure

```

3.1.5.4 Deep Q-Learning (DQN)

Deep Q-Learning (DQN) is a reinforcement learning algorithm to find an optimal solution for an optimization problem. Deep Q-Learning is a model-free and off-policy type of reinforcement learning framework similar to Q-Learning. Instead of creating a Q-Table, Deep Q-Learning uses neural networks to find the optimal state-action pair. Neural networks approximate the Q values in DQN. The size of neural network layers depends on the complexity of the problem. In Deep Q-Learning, the weights in the neural networks are updated with gained reward, instead of updating the Q-Table. Thus, optimization problems that cannot be solved via Q-Learning can be optimally solved via Deep Q-Learning.

Similar to Q-Learning, Deep Q-Learning has a wide variety of application areas in Reinforcement Learning problems due to its ability to solve complex optimization problems. One of the most researched areas for Deep Q-Learning is traffic control. Yu and et al. have proposed a DQN algorithm to solve the control problems of the traffic signals on the high complex road networks. They use queue length and capacity of the roads to determine the proper and optimal traffic signals [198]. Abhishek and et al. have proposed a method to optimize traffic lights operation that dynamically control the lights by using Q-Learning and deep neural networks rather than using conventional methods such as timers [199]. Chen and et al. have proposed a DQN algorithm to solve autonomous vehicles' decision making problem that occurs on overtaking between two lanes. They built a Python program to train their agent and test their algorithm in a simulation environment [200]. Yadav and et al. have proposed a DQN method to model a primary users (PUs) behavior pattern in the distributed dynamic spectrum environment. In the results, their algorithm achieved higher successful transmission by reducing the interference [201]. With development in the game sector, Deep Q-Learning types of machine learning algorithms have been applied to games. He and et al. have proposed a method to learn from a Flappy Bird game model to improve user's behavior [202]. Moreno-Vera and et al. have proposed a Deep Q-Learning method to bring a computer the ability to play Atari games by learning the environment and optimal action-state pairs [203]. On the other hand, Tushar and et al. have proposed a method that uses Convolutional Neural Network (CNN) to control the snake autonomously in the snake game [204]. Sharma and et al. have proposed a DQN based control strategy to optimally evacuate people from the fire situations and tested their algorithm in a real building consisting of 91 rooms with a success [205].

Algorithm 4 Deep Q-Learning (DQN)

```
1: procedure Deep Q-Learning (DQN)
2:    $Q(S, A; \Phi) \leftarrow$  with random  $\Phi$ ,  $\Phi_t \leftarrow \Phi$ 
3:   for each training time step do
4:     Get the initial observation  $S$  from the environment
5:     For the current observation  $S$ , select a random action  $A$  with probability  $\epsilon$ .
6:     Otherwise, select the action for which the critic value function is greatest.
7:      $A \leftarrow \operatorname{argmax} Q(S, A; \Phi)$ 
8:     Execute action  $A$ . Observe the reward  $R$  and next observation  $S'$ .
9:     Store the experience  $(S, A, R, S')$  in the experience buffer.
10:    Sample a random mini-batch of  $M$  experiences  $(S, A, R, S')$  from the experience buffer.
11:    if  $S'_i$  is terminal state then
12:       $y_i \leftarrow R_i$ 
13:    else
14:       $y_i \leftarrow R + \gamma \max Q(S', A; \Phi)$ 
15:    end if
16:    Update the critic parameters by minimizing the loss  $L$ :
17:     $L \leftarrow \frac{1}{M} \sum (y_i - Q(S_i, A_i, \Phi))^2$ 
18:    Update the target critic parameters depending on the target update method.
19:    Update the probability threshold  $\epsilon$  for selecting a random action based on the decay rate.
20:  end for
21: return  $Q$ 
22: end procedure
```

3.1.5.5 Deep Deterministic Policy Gradient (DDPG)

Deep Deterministic Policy Gradient (DDPG) is a reinforcement learning algorithm to find an optimal solution for an optimization problem. DDPG is a model-free and off-policy reinforcement learning method. DDPG is based on an actor-critic learning algorithm that targets to maximize the expected cumulative long-term reward. Actor in DDPG chooses the optimal action that maximizes the long-term reward for a given state. Critic in DDPG computes the expected long-term reward with an action-state pair that is being chosen by an Actor.

DDPG has a wide variety of application areas due to its actor-critic learning algorithm that can be adapted to complex environments. Energy is a trending application area where machine learning algorithms are frequently applied. Zhang and et al. have proposed a

control scheme based on DDPG for energy distribution systems. They model the system based on generators, demanded load fluctuations and photovoltaics. They propose a learning method to solve the uncertainties of renewable energy sources to provide a steady control [206]. Zheng and et al. have proposed a energy management control scheme based on DDPG for fuel cell hybrid vehicles to dramatically decrease the fuel consumption [207]. Similar to Zhang and et al. studies, Das and et al. have proposed a control scheme based on DDPG to control the uncertainties in energy distribution processes with renewable energy production by considering the loss of life in battery energy storage systems [208]. Wu and et al. have proposed a DDPG based control scheme to stabilize DC bus voltage fluctuations that occur on the dual active bridge DC-DC converter [209].

Algorithm 5 Deep Deterministic Policy Gradient (DDPG)

```

1: procedure Deep Deterministic Policy Gradient (DDPG)
2:    $Q(S, A; \Phi) \leftarrow$  with random  $\Phi$ ,  $\Phi_t \leftarrow \Phi$ 
3:    $\pi(S; \theta) \leftarrow$  with random  $\theta$ ,  $\Theta_t \leftarrow \Theta$ 
4:   for each training time step do
5:     Select action  $A = \pi(S; \theta) + N$ , where  $N$  is noise
6:     Execute action  $A$ . Observe the reward  $R$  and next observation  $S'$ .
7:     Store the experience  $(S, A, R, S')$  in the experience buffer.
8:     Sample a random mini-batch of  $M$  experiences  $(S, A, R, S')$  from the experience buffer.
9:     if  $S'_i$  is terminal state then
10:        $y_i \leftarrow R_i$ 
11:     else
12:        $y_i \leftarrow R + \gamma \max Q(S', \pi(S'_i; \theta_t); \Phi_t)$ 
13:     end if
14:     Update the critic parameters by minimizing the loss  $L$ :
15:      $L \leftarrow \frac{1}{M} \sum (y_i - Q(S_i, A_i, \Phi))^2$ 
16:     Update the actor parameters by maximizing the expected discounted reward.
17:      $J \approx \frac{1}{M} \sum i = G_{ai} G_{\pi i}$ 
18:      $G_{ai} = \nabla Q(S_i, A_i; \phi)$ , where  $A = \pi(S_i, \theta)$ 
19:      $G_{\pi i} = \nabla_{\theta} \pi(S_i, \theta)$ 
20:     Update the target actor and critic parameters.
21:   end for
22: return  $Q$ 
23: end procedure

```

3.1.5.6 Asynchronous Advantage Actor Critic (A3C)

Asynchronous Advantage Actor Critic (A3C) learning method was firstly proposed by Google DeepMind Engineers to be used at games such as Atari 2600 games and TORCS Car Racing Simulator [210]. A3C is based on an actor-critic neural networks framework that uses multiple instances of the environment asynchronously. In overall, multiple agents with their own parameters interact with the environment to experience and collect data.

A3C has relatively narrow application of areas unlike other learning algorithms. Zhang and et al have proposed a controlling mechanism based on A3C to control the energy allocation ratio to reduce the cooperation costs in the environment that contains an integrated power, natural gas and heat system [211]. Similar study has been conducted by Dong and et al to reduce the daily average operating costs between 3%-8.7% [212]. On the power electronic side, Xu and et al have proposed a voltage regulating control method based on A3C to reduce the voltage fluctuations and to stabilize the power grid [213]. Biswas and et al have proposed an energy management system based on A3C for multimode hybrid electric powertrains to obtain better fuel economy for any type of unknown drive cycles [214].

Algorithm 6 Asynchronous Advantage Actor Critic (A3C)

```
1: procedure Asynchronous Advantage Actor Critic (A3C)
2:    $Q(S, A; \Phi) \leftarrow$  with random  $\Phi$ 
3:    $\pi(S; \theta) \leftarrow$  with random  $\theta$ 
4:   for each training time step do
5:     Generate  $N$  experiences by following the current policy
6:     Compute the cumulative reward  $G_t$  (including the discounted future reward).
7:      $G_t \leftarrow \sum_{k=t}^{ts+N} (\gamma^{k-t} R_k) + b\gamma^{N-t+1} V(S_{ts+n}; \phi)$ 
8:     Compute the advantage function  $D_t$ 
9:      $D_t \leftarrow G_t - V(S_t; \phi)$ 
10:    Compute the gradients for the actor network by following the policy gradient
        to maximize the expected discounted reward.
11:     $d\theta \leftarrow \sum_{t=1}^N \nabla_{\theta} \ln \pi(A|S_t; \phi) * D_t$ 
12:    Compute the gradients for the critic network by minimizing the mean squared
        error loss between  $V(S_t; \phi)$  and  $G_t$ 
13:     $d\phi \leftarrow \sum_{t=1}^N \nabla_{\phi} (G_t - V(S_t; \phi))^2$ 
14:    Update the actor parameters by applying the gradients.
15:     $\theta \leftarrow \theta + \alpha d\theta$ , where  $\alpha$  is the learning rate of the actor
16:    Update the critic parameters by applying the gradients.
17:     $\phi \leftarrow \phi + \beta d\phi$ , where  $\beta$  is the learning rate of the critic.
18:  end for
19: return  $Q$ 
20: end procedure
```

3.1.5.7 Trusted Region Policy Optimization (TRPO)

Trusted Region Policy Optimization (TRPO) is a reinforcement learning algorithm to find an optimal solution for an optimization problem. TRPO is a model-free and on-policy reinforcement learning method. TRPO is based on an actor-critic learning algorithm that uses policy gradients to update the policy parameters. Actions in TRPO are selected randomly based on the probability distribution. The agent computes probabilities of each action in the action space to maximize the reward.

TRPO has a wide variety of application areas due to its ability to prevent performance drops during the learning phase. Hide and seek game is an educational learning environment for machine learning algorithms. In reinforcement learning perspective, hide and seek game is a multi-agent environment where multiple agents collaborate to maximize the reward. Wafa and et al. have proposed a TRPO based reinforcement learning algorithm to a hide and seek game environment and then compare the results with Vanilla

Policy Gradient algorithm [215]. In the robotic field, Yang and et al. have proposed a TRPO based algorithm to control a 7 degree of freedom robot arm for two specific reasons. Firstly, they gather data and use the data to teach a robot arm to reach a specific target in an environment with multiple obstacles. Secondly, they teach a robot arm to reach a moving target which they call dynamic object task [216]. Liu and et al. have proposed an object discrimination algorithm based on TRPO for active object recognition from a camera [217].



Algorithm 7 Trust Region Policy Optimization (TRPO)

```
1: procedure Trust Region Policy Optimization (TRPO)
2:    $V(s; \phi) \leftarrow$  with random  $\phi$ 
3:    $\pi(A|S; \theta) \leftarrow$  with random  $\theta$ 
4:   for each training time step do
5:     Generate  $N$  experiences by following the current policy
6:     Compute the cumulative reward  $G_t$  and the advantage function.
7:     if Advantage estimation method is finite horizon then
8:        $G_t \leftarrow \sum_{k=t}^{ts+N} (\gamma^{k-t} R_k) + b\gamma^{N-t+1} V(S_{ts+n}; \phi)$ 
9:        $D_t \leftarrow G_t - V(S_t; \phi)$ 
10:    end if
11:    if Advantage estimation method is generalized advantage estimator then
12:       $D_t \leftarrow G_t + V(S_t; \phi)$ 
13:       $D_t \leftarrow \sum_{k=t}^{ts+N-1} (\gamma\lambda)^{k-t} \delta_k$ , where  $\lambda$  is smoothing factor
14:       $\delta_k \leftarrow R_t + b\gamma V(s; \phi)$ 
15:    end if
16:    for each mini-batches of learning experience epoch do
17:      Sample a random mini-batch data set of size  $M$  from the current set of
      experiences
18:      Update the critic parameters by minimizing the loss
19:       $L_{critic}(\Phi) \leftarrow \frac{1}{M} \sum i = (G_i - V(S_t; \Phi))^2$ 
20:      Update the actor parameters
21:       $g \leftarrow \nabla_{\theta} L_{actor}(\theta) = \nabla_{\theta} - \frac{1}{M} \sum i \left( \frac{\pi(A_i|S_i; \theta)}{\pi(A_i|S_i; \theta_{old})} \right) D_i + w_i(Q, S_i)$ 
22:      Apply the conjugate gradient (CG) method to find an approximate solu-
      tion.
23:       $x \approx H^{-1}g$ , where  $H$  is the Hessian of the KL-divergence between the old
      and new policies.
24:      Using a line search algorithm, find the largest  $\alpha$  that satisfies the following
      constraints.
25:       $\theta \leftarrow \theta_{old} + \alpha \sqrt{\frac{2\delta}{x^T H^{-1} x}} x$ 
26:       $L_{actor}(\theta) - L_{actor}(\theta_{old}) < 0$ 
27:      If a valid value of  $\alpha$  exists, update the parameters of the actor network to
       $\theta$ . If a valid value of  $\alpha$  does not exist, do not update the actor parameters.
28:    end for
29:  end for
30:  return  $Q$ 
31: end procedure
```

3.1.5.8 Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) is a reinforcement learning algorithm to find an optimal solution for an optimization problem. PPO is a model-free and on-policy reinforcement learning method. PPO is based on an actor-critic learning algorithm that uses stochastic gradient descent to update the policy parameters. PPO simplifies the TRPO algorithm's computing process. The main difference between PPO and TRPO is that TRPO performs a more robust learning process than PPO in deterministic environment dynamics.

PPO has a wide variety of application areas from autonomous navigation to communication. Toan and et al. have proposed a PPO based learning algorithm with convolutional neural networks to optimize a mobile robot's mapless autonomous navigation problems [218]. Zishen and et al. have proposed a PPO based control method for the secondary users to gain continuous power control to effectively share the common spectrum with the primary users in cognitive radio networks [219]. Esfandiari and et al. have proposed a PPO based control method for quadcopters to optimize its performance during take-off and hovering phases with 6 degrees of freedom [220].

Algorithm 8 Proximal Policy Optimization (PPO)

```
1: procedure Proximal Policy Optimization (PPO)
2:    $V(s; \phi) \leftarrow$  with random  $\phi$ 
3:    $\pi(A|S; \theta) \leftarrow$  with random  $\theta$ 
4:   for each training time step do
5:     Generate  $N$  experiences by following the current policy
6:     Compute the cumulative reward  $G_t$  and the advantage function.
7:     if Advantage estimation method is finite horizon then
8:        $G_t \leftarrow \sum_{k=t}^{ts+N} (\gamma^{k-t} R_k) + b\gamma^{N-t+1} V(S_{ts+n}; \phi)$ 
9:        $D_t \leftarrow G_t - V(S_t; \phi)$ 
10:    end if
11:    if Advantage estimation method is generalized advantage estimator then
12:       $D_t \leftarrow G_t + V(S_t; \phi)$ 
13:       $D_t \leftarrow \sum_{k=t}^{ts+N-1} (\gamma\lambda)^{k-t} \delta_k$ , where  $\lambda$  is smoothing factor
14:       $\delta_k \leftarrow R_k + b\gamma V(s; \phi)$ 
15:    end if
16:    for each mini-batches of learning experience epoch do
17:      Sample a random mini-batch data set of size  $M$  from the current set of
      experiences
18:      Update the critic parameters by minimizing the loss
19:       $L_{critic}(\Phi) \leftarrow \frac{1}{M} \sum i = (G_i - V(S_i; \Phi))^2$ 
20:      Update the actor parameters
21:       $g \leftarrow \nabla_{\theta} L_{actor}(\theta) = \nabla_{\theta} - \frac{1}{M} \sum i (\frac{\pi(A_i|S_i; \theta)}{\pi(A_i|S_i; \theta_{old})}) D_i + w_i(Q, S_i)$ 
22:      Apply the conjugate gradient (CG) method to find an approximate solu-
      tion.
23:       $x \approx H^{-1}g$ , where  $H$  is the Hessian of the KL-divergence between the old
      and new policies.
24:      Using a line search algorithm, find the largest  $\alpha$  that satisfies the following
      constraints.
25:       $L_{actor}(\theta) \leftarrow \frac{1}{M} \sum_{i=1}^M (-\min(r_i(\theta) D_i, c_i(\theta) D_i) + w H_i(Q, S_i))$ 
26:       $r_i(\theta) \leftarrow \frac{\pi(A_i|S_i; \theta)}{\pi(A_i|S_i; \theta_{old})}$ 
27:       $c_i(\theta) \leftarrow \max(\min(r_i(\theta), 1 + \epsilon), 1 - \epsilon)$ 
28:    end for
29:  end for
30:  return  $Q$ 
31: end procedure
```

3.1.5.9 Twin Delayed Deep Deterministic Policy (TD3)

Twin Delayed Deep Deterministic Policy (TD3) is a reinforcement learning algorithm to find an optimal solution for an optimization problem. TD3 is a model-free and off-policy reinforcement learning method. TD3 is based on an actor-critic learning algorithm. TD3 is an extension of DDPG that uses minimum value function for policy updates less frequently and uses noise for target action.

TD3 has a wide variety of application areas from energy flow to antenna applications. Woo and et al. have proposed a AC Optimal Power Flow (ACOPF) control method based on TD3 with Levenberg-Marquardt backpropagation descent to improve the computational performance of the energy flow process [221]. Hossain and et al. have proposed a voltage regulation control method based on TD3 to reduce the power loss and increase the stability of the power grid with uncertainties of distributed energy sources [222]. Zhang and et al. have proposed an array optimization based on TD3 to improve an antenna array's health and performance by modeling conformal array antennas as electromagnetic models [223].

Algorithm 9 Twin Delayed Deep Deterministic Policy (TD3)

```
1: procedure Twin Delayed Deep Deterministic Policy (TD3)
2:   Specify actor and critic numbers
3:    $Q_k(S, A; \phi_k)$  with random  $\phi_k, \phi_{tk} \leftarrow \phi_k$ 
4:    $\pi(S; \theta)$  with random  $\theta, \theta_t \leftarrow \theta$ 
5:   for each training time step do
6:     Select action  $A = \pi(S; \theta) + N$ , where  $N$  is stochastic noise
7:     Execute action  $A$ . Observe the reward  $R$  and next observation  $S'$ .
8:     Store the experience  $(S, A, R, S')$  in the experience buffer
9:     Sample a random mini-batch of  $M$  experiences  $(S_i, A_i, R_i, S'_i)$  from the experience buffer
10:    if  $S'_i$  is terminal state then
11:       $y_i \leftarrow R_i$ 
12:    else
13:       $y_i \leftarrow R_i + \gamma \min(Q_{tk}(S'_i, \text{clip}(\pi(S'_i; Q_t)) + \epsilon); \phi_{tk})$ 
14:    end if
15:    Update each critics parameters by minimizing the loss.
16:     $L \leftarrow \frac{1}{M} \sum (y_i - Q_{tk}(S_i, A_i; \phi_k))^2$ 
17:    At every update policy frequency, update each actors parameters.
18:     $\nabla_{\theta} J \approx \frac{1}{M} \sum (G_{ai} - G_{\pi i})$ 
19:     $G_{ai} \leftarrow \nabla_A \min(Q_k(S_i, A; \phi)), \text{ where } A = \pi(S_i; \theta)$ 
20:     $G_{\pi i} \leftarrow \nabla_{\theta} \pi(S_i; \theta)$ 
21:    At every update target frequency, update the target actor and critics
22:  end for
23: return  $Q$ 
24: end procedure
```

3.1.6 Soft Actor-Critic (SAC)

Soft Actor-Critic (SAC) is a reinforcement learning algorithm to find an optimal solution for an optimization problem. SAC is a model-free and off-policy reinforcement learning method. SAC is based on an actor-critic learning algorithm. SAC defines the optimal policy by both maximizing the long-term expected reward and its entropy. For a given state's policy uncertainty is defined as the policy entropy. Thus, increasing the policy entropy brings more exploration of the environment.

SAC has a wide variety of application areas due to its entropy maximizing mechanism. Zhong and et al. have proposed a learning algorithm based on SAC to improve anomaly detection processes in noisy observational environments [224]. Zhu and et al.

have proposed a SAC based learning algorithm to improve a railway train energy management system to reduce the energy cost of subway systems. To compare the results and effectiveness of SAC, they have made a comparison between DQN and SAC algorithms. Sun and et al. have proposed two different intelligent computation offloading algorithms based on SAC to reduce service latency of 6G mobile networks in multi-access edge computing [225].

Algorithm 10 Soft Actor-Critic (SAC)

```

1: procedure Soft Actor-Critic (SAC)
2:   Specify critic numbers
3:    $Q_k(S, A; \phi_k)$  with random  $\phi_k, \phi_{tk} \leftarrow \phi_k$ 
4:    $\pi(S; \theta)$  with random  $\theta, \theta_t \leftarrow \theta$ 
5:   Perform a warm start by taking a sequence of actions following the initial random
      policy in  $\pi(S)$ . For each action, store the experience in the experience buffer.
6:   for each training time step do
7:     Select action  $A = \pi(S; \theta)$ 
8:     Execute action  $A$ . Observe the reward  $R$  and next observation  $S'$ .
9:     Store the experience  $(S, A, R, S')$  in the experience buffer
10:    Sample a random mini-batch of  $M$  experiences  $(S_i, A_i, R_i, S'_i)$  from the experience
      buffer
11:    At every critic update frequency, update the critics parameters by minimizing
      the loss.
12:       $L \leftarrow \frac{1}{M} \sum (y_i - Q_{tk}(S_i, A_i; \phi_k))^2$ 
13:      if  $S'_i$  is terminal state then
14:         $y_i \leftarrow R_i$ 
15:      else
16:         $y_i \leftarrow R_i + \gamma \min(Q_{tk}(S'_i, A'_i; \phi_{tk}) - \alpha \ln \pi(S'_i; \theta))$ , where  $\alpha$  is the entropy
          loss weight
17:      end if
18:      At every update policy frequency, update the actor parameters.
19:       $J_\pi \leftarrow \frac{1}{M} \sum (-\min(Q_{tk}(S_i, A_i; \phi_{tk}) +) + \alpha \ln \pi(S'_i; \theta))$ 
20:      At every update policy frequency, update the entropy weight.
21:       $L_\alpha \leftarrow \frac{1}{M} \sum (-\alpha \ln \pi(S'_i; \theta)) - \alpha H$ , where  $H$  is the target entropy.
22:      At every update target frequency, update the target critics
23:    end for
24: return  $Q$ 
25: end procedure

```

3.2 GAME THEORY on REINFORCEMENT LEARNING

In the digital age, decision making has become more valuable due to rapidly changing dynamics of the world. Decision making generally causes conflict situations. Conflict situations can be rationally solved via Game Theory [226]. Game Theory has been widely used in areas such as economics, social sciences and engineering applications to create a model of the environment for decision making purposes. The concept of Game Theory was firstly proposed by John von Neumann and Oskar Morgenstern in 1944. Neumann and Morgenstern had published a book called “Theory of Games and Economic Behavior” [227]. The best-known application of Game Theory is “Prisoner’s Dilemma” which was created by Merrill Flood and Melvin Dresher in 1950. In the “Prisoner’s Dilemma” game, two agents are asked to choose action either by cooperating with each or non-cooperating with each other to maximize self outcomes [228].

An environment with the interaction of multiple agents can be mathematically modeled with Game Theory. Game Theory consists of players, a.k.a agents, players’ actions, and set of rules. The actions of a player is called strategy that represents a plan of actions that can be applied with probability according to current environmental states. The main idea lying behind the game theory is to maximize a player’s outcomes via utility function during the game. With the developing computational skills of computers, most of the learning problems that contain multiple agents have started to be modeled based on game theory. In this project, the learning environment is modeled based on the Game Theoretical approach.

3.2.1 Nash Equilibrium

In Game Theory, each agent makes a decision to optimally maximize their own outcomes. Outcomes of each agent are dependent on other agents’ outcomes. In non-cooperative games where agents compete against each other, an agent should try to maximize its own outcome by minimizing other agents’ outcomes. If an agent doesn’t have incentive to change its strategy in a current environmental state with given other agents’ strategies, an equilibrium state has been achieved. Achieved equilibrium state is defined as Nash Equilibrium. Nash Equilibrium was firstly proposed by John Nash in 1951 [229]. In theory, a Nash equilibrium state gives the optimal outcomes to each agent. In Nash equilibrium, if a Nash equilibrium exists and every agent is rational to take the optimal action, then all agents will converge to a steady state. Thus, not all games may have a Nash equilibrium state although some games may have more than one Nash equilibrium. Therefore, alternative outcomes should be preferred by all agents and must be feasible.

Nash equilibrium can be achieved via different strategies by agents. In pure-strategy Nash equilibrium, an agent cannot obtain a higher utility outcome by choosing a different

action with given every other agents' action. On the other hand, in mixed-strategy Nash equilibrium, an agent cannot obtain a higher utility than the given outcome. Therefore, the maximum utility outcome is defined by the environment.

3.2.2 Game Types and Applications

3.2.2.1 Cooperative and Non-Cooperative

Most of the researchers have been focused on cooperative and non-cooperative games in Game Theory literature. In cooperative games, all agents share a common strategy to maximize cumulative utility outcomes and execute that strategy as a group. On the contrary, in non-cooperative games, all agents follow their own strategy to maximize their own utility outcomes.

Cooperative game theory has a wide variety of application areas in the literature. Li and et al. have proposed a cooperative game structure to optimally make a jamming decision on wireless networks by modeling the environment as a Stackelberg game [230]. Li and et al. have proposed a cooperative game structure to achieve the optimal task allocation plan in a multi-robot environment [231]. Zhang and et al. have proposed a cooperative game structure to achieve the optimal strategy for power control and resource allocation of heterogeneous devices during broadcasting by establishing Signal to Noise Ratio (SNRs) and Signal to Interference Ratio (SIRs) utility functions [232].

Non-cooperative game theory has a wide variety of application areas in the literature. He and et al. have proposed a non-cooperative game theory structure to achieve the optimal loading balance scheme in Radio-over-Fibre (RoF) systems by applying differential game theory essentials [233]. Hui and et al. have proposed a non-cooperative game theory structure to optimize energy distribution in wireless sensor networks and to increase the lifetime of wireless sensor networks [234]. Duan and et al. have proposed a non-cooperative game theory structure to achieve an optimal energy distribution plan in integrated energy systems to increase the efficiency of the whole distribution system [235]. Yang and et al. have proposed a non-cooperative game theory structure to increase the efficiency between distributed energy generation systems and power users and to improve the economic effectiveness by reducing the usage costs [236].

3.2.2.2 Symmetric and Asymmetric

In symmetric games, specifically in short-term games, all agents share the same strategy based on the same environmental rules. That means all agents share the same action space. "Prisoner's Dilemma" is an example of symmetric game theory. On the contrary, in asymmetric games, all agents differentiate each other by adopting different strategies. Strategies of agents are determined by the agent's current state. Thus, the same action

may not be beneficial for all agents in the environment.

Symmetric game theory has a wide variety of application areas in the literature. El-hadef and et al. have proposed a symmetric game structure to diagnose faulty nodes based on an input syndrome by maximizing all agents' utility outcomes [237]. Zhao and et al. have proposed a symmetric game structure to achieve the optimal inventory coordination of manufacturers and suppliers by analyzing different strategies [238]. Roberto and et al. have proposed a symmetric game structure to solve the multi-hop problems that causes excessive resource consumption on VANETs [239].

Asymmetric game theory has a wide variety of application areas in the literature. Wen and et al. have proposed an asymmetric game structure to achieve the optimal channel selection in multi-channel wireless AD-HOC networks by computing the quality discrepancy between channels [240]. Dou and et al. have proposed an asymmetric game structure to analyze network effects in two-sided markets by applying the Stackelberg game [241].

3.2.2.3 Simultaneous And Sequential

In simultaneous games, agents choose actions simultaneously. Therefore, agents are not aware of the other agents' actions selection. On the contrary, in sequential games, agents choose actions sequentially. Therefore, agents are aware of the other agents' actions selection.

Simultaneous and sequential game theories have a relatively low variety of application areas in the literature. Ahmad and et al. have proposed a simultaneous game structure to schedule multi-core processors' tasks to minimize the energy consumption based on both simultaneous and cooperative games [242]. Haijie and et al. have proposed a sequential game structure to solve the byproducts and wastes price decision between buyer and seller based on both sequential and cooperative games [243].

3.2.2.4 Perfect and Imperfect Information

In perfect games, all agents can access all the same environmental and other information. Thus, all agents have complete information about the environment. Agents make decisions based on the complete information and are fully informed. On the contrary, in imperfect games, agents cannot access other agents' information and to information that is not shared by the environment. Thus, each agent has incomplete information about the environment. Therefore, agents make decisions based on incomplete information and are partially informed.

Perfect and imperfect game theories have a relatively low variety of application areas in the literature. Lu and et al. have proposed a perfect game structure to integrate game theory into parallel monte carlo tree search algorithm [244]. Xing and et al. have proposed

a perfect game structure to solve the flight slot allocation on airports to prevent flight delays and resource waste by modeling the situation as a Stackelberg game [245]. Farshchi and et al. have proposed an imperfect game structure to analyze risk dominance strategy on Nash equilibrium [246]. Nandini and et al. have proposed an imperfect game structure to improve a poker agent decision making process in such games as No Limit Texas Hold etc [247].

3.2.2.5 Constant Sum, Zero-Sum And Non-Zero-Sum

In constant sum games, the sum of utility outcomes of all the agents are constant even under different agents' action and individual outcomes. In zero sum games, the sum of utility outcomes of all the agents are zero. On the contrary, in non-zero sum games, the sum of utility outcomes of all the agents are different from zero and can be changed according to the agents' action and individual outcomes.

Constant Sum, Zero-Sum And Non-Zero-Sum game theories have a wide variety of application areas in the literature. Harris and et al. have proposed a constant sum game structure to build a platform for humans to compete against machine algorithms [248]. Yang and et al. have proposed a constant sum game structure to build an intrusion detection system for smartphone systems based on both non-cooperative and perfect information game theories [249]. Amadi and et al. have proposed a zero sum game structure to improve a server's defense system against distributed denial of service (DDoS) attacks by solving the game with linear programming [250]. Touazi and et al. have proposed a zero sum game structure to optimize feature selection and build a selection scheme for artificial intelligence applications [251]. Ji-jun and et al. have proposed a non zero sum game structure to analyze enterprises behaviors in supply chains based on both cooperative and non zero sum game theories [252]. An and et al. have proposed a zero sum game structure to optimize performance metrics of modular and reconfigurable robot (MRR) systems and to reduce the energy costs due to controller [253].

CHAPTER 4 MATHEMATICAL MODELLING

In this chapter, mathematical models of each environmental component are explained. Built-in MATLAB objects and Simulink blocks were used to simulate the learning process.

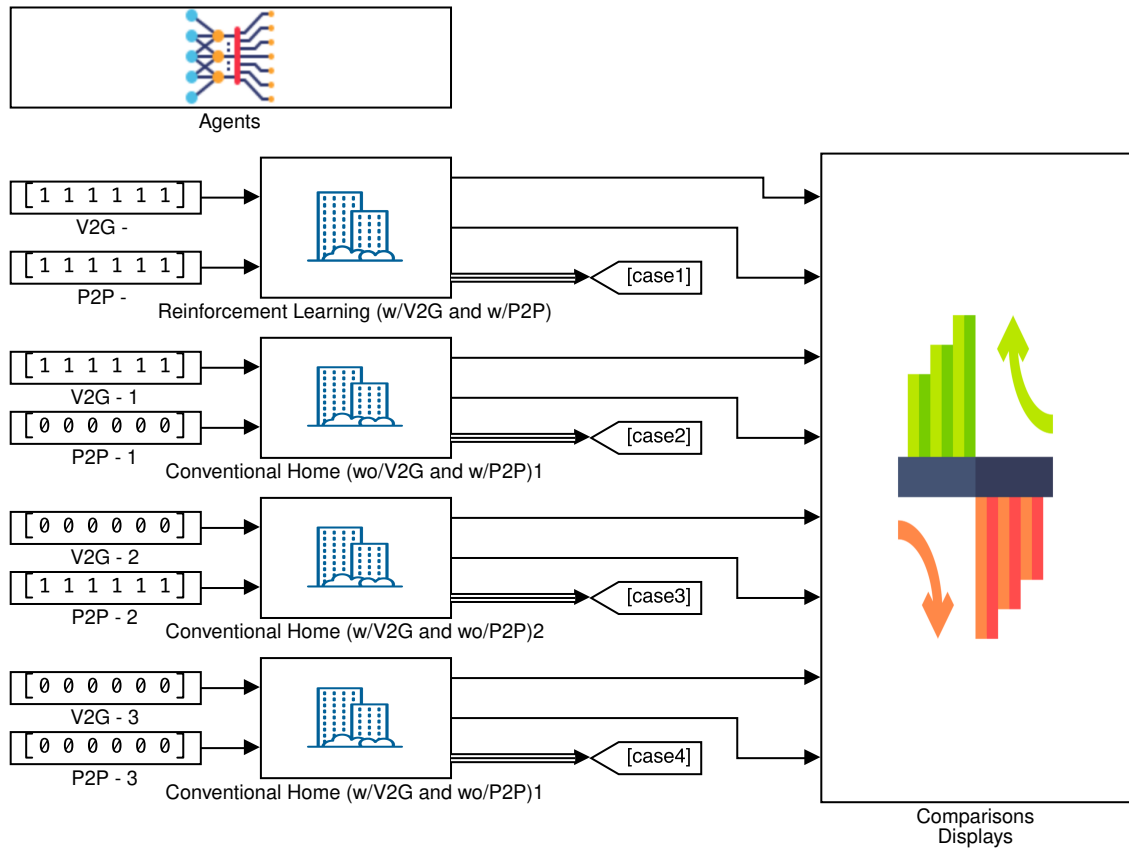


Figure 4.1: Overall Simulink Model

4.1 ENERGY STORAGE SYSTEM MODEL

A MATLAB model was developed to simulate the behavior and performance of an Energy Storage System (ESS) based on lead-acid batteries. The model was implemented using the C programming language within the MATLAB environment. The "ESS" was divided into two distinct MATLAB functions, namely "ESS" and "ESSSoC", which were encapsulated within a MATLAB class. The "ESS" function encapsulates the fundamental behavior of the Energy Storage System. It accepts three input arguments, namely power, mode, and the current State of Charge (SoC), and returns four outputs: output power, available power, output energy, and computed State of Charge (SoC). Output power of the ESS can be

expressed as follows.

$$P_n = \begin{cases} P_n \geq 0, & \frac{P_n}{\eta_{disc}} \\ P_n < 0, & P_n \cdot \eta_{chg} \end{cases}$$

where P_n is required power, η_{disc} is discharging efficiency, and η_{chg} is charging efficiency of ESS. To satisfy the constraints, following equation should be applied.

$$P_n = \min(MDP, \max(MCP, Power))$$

where MDP is maximum discharging power, and MCP is maximum charging power of ESS. Available power in ESS for P2P energy trading can be expressed as follows.

$$P_t^{ava} = (ESS_n^{capacity} \cdot SoC_t - ESS_n^{capacity} \cdot SoC_{min})$$

where P_{ava} is available power at time t , $ESS_n^{capacity}$ is capacity of ESS, SoC_t is current SoC, and SoC_{min} is minimum SoC that ESS can have.

Table 4.1: Input Arguments of the “ESS” Function

Argument	Description	Unit	Range
Power	The power input to the Energy Storage System	Watt	$-\infty$ to ∞
Mode	The operational mode of the Energy Storage System	-	0, 1
State of Charge	The current state of charge (SoC) of the battery	-	0 to 1

Table 4.2: Output Arguments of the “ESS” Function

Argument	Description	Unit	Range
Output Power	The power output from the Energy Storage System	Watt	$-\infty$ to ∞
Available Power	The available power for P2P trading	Watt	$-\infty$ to ∞
Output Energy	The energy output from the Energy Storage System	kWh	$-\infty$ to ∞
State of Charge	The computed state of charge (SoC) of the battery	-	0 to 1

Algorithm 11 ESS to represents the Energy Storage System behavior

```
1: procedure ESS (Power, mode, SoC)
2:   if not column matrix then
3:     Convert Power input to column matrix
4:     Convert mode input to column matrix
5:     Convert SoC input to column matrix
6:   end if
7:   Calculate maximum charging power
8:   Calculate maximum discharging power
9:   Compute Output Power, Available Power, Output Energy, and SoC
10: return OutputPower, AvailablePower, OutputEnergy, SoC
11: end procedure
```

The "ESSSoC" function calculates the updated state of charge (SoC) of the battery based on the provided energy input and the previous SoC value. This function accepts two input arguments and yields a single output. The input arguments consist of energy and the previous state of charge (SoC) respectively. The function output represents the computed state of charge (SoC). Computed SoC can be expressed as follows.

$$SoC_{t+1} = SoC_t - P_n \cdot \frac{\Delta t}{Q_n}$$

To protect the ESS from over-charging and over-discharging, following constraint should be applied.

$$SoC_{t+1} = \min(1, \max(0, SoC_{t+1}))$$

Table 4.3: Input Arguments of the "ESSSoC" Function

Argument	Description	Unit	Range
Energy	The energy input to compute the new SoC	kWh	$-\infty$ to ∞
Previous SoC	The previous state of charge (SoC) of the battery	-	0 to 1

Table 4.4: Output Arguments of the “ESSSoC” Function

Argument	Description	Unit	Range
Computed SoC	The calculated state of charge (SoC) of the battery	-	0 to 1

Algorithm 12 ESSSoC to compute the current SoC

```

1: procedure ESSSoC (Energy, Previous SoC)
2:   Compute SoC
3: return SoC
4: end procedure

```

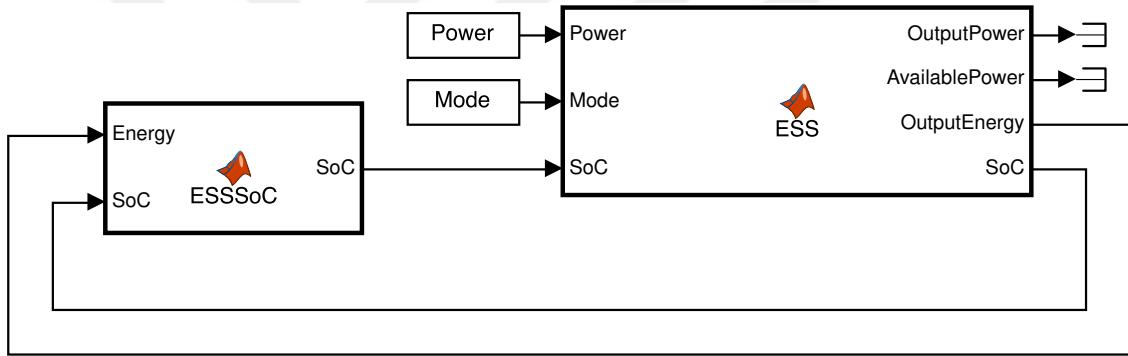


Figure 4.2: Simulink Model of Energy Storage System (ESS)

4.2 ELECTRIC VEHICLE MODEL

A MATLAB model was developed to simulate the behavior and performance of an Electric Vehicle (EV) powered by a Lithium-Ion battery. The model was implemented using the C programming language within the MATLAB environment. The Electric Vehicle was divided into seven distinct MATLAB functions, namely Li-Ion look-up table, Li-Ion Battery, Drive SoC Calculator, SoC, Charger, Discharger, and EV, which were encapsulated within a MATLAB class. The Li-Ion look-up function computes the Open Circuit Voltage (OCV) and internal resistance look-up tables for the Li-Ion battery. This function does not require any input arguments and produces two outputs: the OCV look-up table and the internal resistance look-up table.

Table 4.5: Output Arguments of the “Li-Ion look-up” Function

Argument	Description	Unit	Range
OCV	Open circuit voltage look-up table	V	$-\infty$ to ∞
Internal Resistance	Internal resistance look-up table	Ω	$-\infty$ to ∞

Algorithm 13 Li-Ion look-up to compute battery’s OCV and R

- 1: **procedure** Li-Ion look-up
 - 2: Initialize Open Circuit Voltage experimental results
 - 3: Initialize Internal Resistance experimental results
 - 4: Compute OCV Look-up Table by using polyfit function to calculate polynomial coefficients of 10th degree polynomial via experimental results
 - 5: Compute Internal Resistance Look-up Table by using polyfit function to calculate polynomial coefficients of 10th degree polynomial via experimental results
 - 6: **return** OCV Look-up Table, Internal Resistance Look-up Table
 - 7: **end procedure**
-

The ”Battery” function is responsible for computing the numerical values of the Open Circuit Voltage (OCV) and internal resistance based on a given state of charge. This function accepts three input arguments and produces two outputs. The input arguments consist of the current state of charge, OCV and internal resistance look-up tables respectively. Calculation of numerical OCV and internal resistance values can be expressed as follows.

$$OCV = \sum_{i=0}^{i=10} (SoC^{10-i} \cdot OCV_{lut}(i))$$

$$R = \sum_{i=0}^{i=10} (SoC^{10-i} \cdot R_{lut}(i))$$

Table 4.6: Input Arguments of the “Battery” Function

Argument	Description	Unit	Range
SoC	Current state of charge of the battery	-	0 to 1
OCV LuT	OCV Look-up table	-	-
Internal Resistance LuT	Internal Resistance Look-up table	-	-

Table 4.7: Output Arguments of the “Battery” Function

Argument	Description	Unit	Range
OCV	Computed numerical value of OCV	V	0 to ∞
Internal Resistance	Computed numerical value of internal resistance	Ω	0 to ∞

Algorithm 14 Battery to compute battery’s OCV and R numerical values

-
- 1: **procedure** Battery (OCV Look-up Table, R Look-up Table)
 - 2: Compute numerical OCV value
 - 3: Compute numerical R value
 - 4: **return** OCV numerical value, R numerical value
 - 5: **end procedure**
-

The ”Drive SoC Calculator” function is designed to calculate the state of charge (SoC) when the vehicle is driven, considering a specified daily energy requirement for the drive. This function requires three input arguments and produces a single output. The input arguments include the previous state of charge, a presence signal, and the required daily driving energy. The output of the function represents the computed state of charge value. Computed state of charge can be expressed as follows.

$$SoC_{t+1}^{drive} = SoC_t \left((EV_{rate} == 1) \cdot \frac{EV_{drive}}{(V_{nom} \cdot N_s \cdot N_p \cdot Ah \cdot \eta_{disc})} \right)$$

where EV_{rate} is derivative signal of departure of EV, EV_{drive} is daily driving energy in kWh, V_{nom} is nominal voltage of EV battery, N_s is number of series battery cell in EV battery, N_p is number of parallel battery cell in EV battery, Ah is battery cell capacity.

Table 4.8: Input Arguments of the “Drive SoC Calculator” Function

Argument	Description	Unit	Range
Previous SoC	Previous state of charge of the EV.	-	0 to 1
Presence Signal	Signal that represents whether the EV is at home or not.	-	0, 1
Daily driving energy	Daily driving energy data for daily vehicle ride.	kWh	0 to ∞

Table 4.9: Output Arguments of the “Drive SoC Calculator” Function

Argument	Description	Unit	Range
SoC	Computed current SoC of the EV.	-	0 to 1

Algorithm 15 Drive SoC Calculator to compute current SoC after the daily ride

- 1: **procedure** Drive SoC Calculator (Previous SoC, Presence Signal, Daily driving energy)
 - 2: Compute SoC
 - 3: **return** SoC
 - 4: **end procedure**
-

The ”SoC” function is responsible for calculating the state of charge (SoC) based on the provided input parameters. This function accepts two input arguments and produces a single output. The input arguments consist of the previous state of charge and the current. The output of the function represents the computed state of charge value. Computed SoC can be expressed as follows.

$$SoC_{t+1} = SoC_t - (I_n \cdot \frac{\Delta t}{60} \cdot \frac{Ah}{N_p})$$

where I_n is battery current. To protect the EV from over-charging and over-discharging, following constraint should be applied.

$$SoC_{t+1} = \min(1, \max(0, SoC_{t+1}))$$

Table 4.10: Input Arguments of the “SoC” Function

Argument	Description	Unit	Range
Previous Soc	Previous SoC of the EV	-	0 to 1
Current	Current output of the EV Battery	A	1 to ∞

Table 4.11: Output Arguments of the “SoC” Function

Argument	Description	Unit	Range
SoC	Computed current SoC of the EV	-	0 to 1

Algorithm 16 SoC to compute current SoC after charging/discharging cycle

```

1: procedure SoC (Previous SoC, Current))
2:   Compute SoC
3: return SoC
4: end procedure

```

The “Charger” function is responsible for calculating the necessary outputs during the charging cycle. This function accepts two input arguments and produces three output values. The input arguments consist of the power input and the previous state of charge (SoC). The output arguments include the current, output power, and grid power.

Table 4.12: Input Arguments of the “Charger” Function

Argument	Description	Unit	Range
Power	Required charging power of the EV	W	0 to ∞
Previous SoC	Previous state of charge of the EV	-	0 to 1

Table 4.13: Output Arguments of the “Charger” Function

Argument	Description	Unit	Range
Current	Computed current output of the EV Battery	A	0 to ∞
Output Power	Computed output power	W	$-\infty$ to ∞
Grid Power	Computed grid power	W	$-\infty$ to ∞

Algorithm 17 Charger to compute charging outputs of the EV

```

1: procedure Charger (Power, Previous SoC)
2:   Compute  $OCV_{lut}$  and  $R_{lut}$  using Li-Ion look-up function
3:   Compute numerical OCV and R values using Battery function
4:   Calculate Allowed Energy
5:   if  $Power > AllowedEnergy$  then
6:      $Power \leftarrow AllowedEnergy$ 
7:   end if
8:   Calculate Constant Current Constant Voltage (CCCV) state
9:   if CCCV then
10:     $Current \leftarrow CV * (SoC < MaxSoC)Energy$ 
11:  else
12:     $Current \leftarrow CC * (SoC < MaxSoC)Energy$ 
13:  end if
14: return Current, Output Power, Grid Power
15: end procedure

```

The “Discharger” function is responsible for calculating the required outputs during the discharging cycle. This function accepts two input arguments and produces two output values. The input arguments include the power input and the previous state of charge (SoC). The output arguments consist of the current and the output power.

Table 4.14: Input Arguments of the “Discharger” Function

Argument	Description	Unit	Range
Power	Required discharging power of the EV	W	0 to ∞
Previous SoC	Previous state of charge of the EV	-	0 to 1

Table 4.15: Output Arguments of the “Discharger” Function

Argument	Description	Unit	Range
Current	Computed current output of the EV Battery	A	0 to ∞
Output Power	Computed output power	W	$-\infty$ to ∞

Algorithm 18 Discharger to compute discharging outputs of the EV

-
- 1: **procedure** Discharger (Power, Previous SoC)
 - 2: Compute OCV_{lut} and R_{lut} using Li-Ion look-up function
 - 3: Compute numerical OCV and R values using Battery function
 - 4: Compute Current, Output Power
 - 5: **return** Compute Current, Output Power
 - 6: **end procedure**
-

The ”EV” function conducts a comprehensive simulation of the Electric Vehicle (EV) system. This function accepts five input arguments and produces five output values. The input arguments include the vehicle information, power, status information, mode and previous SoC. The output arguments consist of the output power, grid power, available power, current and SoC.

Table 4.16: Input Arguments of the “EV” Function

Argument	Description	Unit	Range
Vehicle Information	Signal represents whether Home has EV or not, Signal represents whether EV has V2G functionality	-	0, 1
Power	Required discharging power of the EV	W	$-\infty$ to ∞
Status Information	Signal represents whether EV is at home or not, Signal represents the change of Status signal, Signal represents daily driving energy data for daily vehicle ride	-	0, 1
Mode	Signal represents charging, idle or discharging modes	-	-1, 0, 1
Previous SoC	Previous state of charge of the EV	-	0 to 1

Table 4.17: Output Arguments of the “EV” Function

Argument	Description	Unit	Range
Output Power	Computed output power	W	0 to ∞
Grid Power	Computed grid power	W	$-\infty$ to ∞
Available Power	Computed available power	W	$-\infty$ to ∞
Current	Computed current output of the EV Battery	A	$-\infty$ to ∞
SoC	Computed current state of charge (SoC)	-	0 to 1

Algorithm 19 EV to compute entire Electric Vehicle simulation

```

1: procedure EV (Vehicle Information, Power, Status Information, Mode, Previous
   SoC)
2:   if Mode is Discharging then
3:     Compute Output Power and Current using Discharger function
4:   else if Mode is Idle then
5:     Set Output Power and Current to 0
6:   else if Mode is Charging then
7:     Compute Output Power and Current using Charger function
8:   end if
9:   Compute Available Power, Grid Power, and SoC by using SoC and Drive SoC
   Calculator functions
10: return Current, Output Power
11: end procedure

```

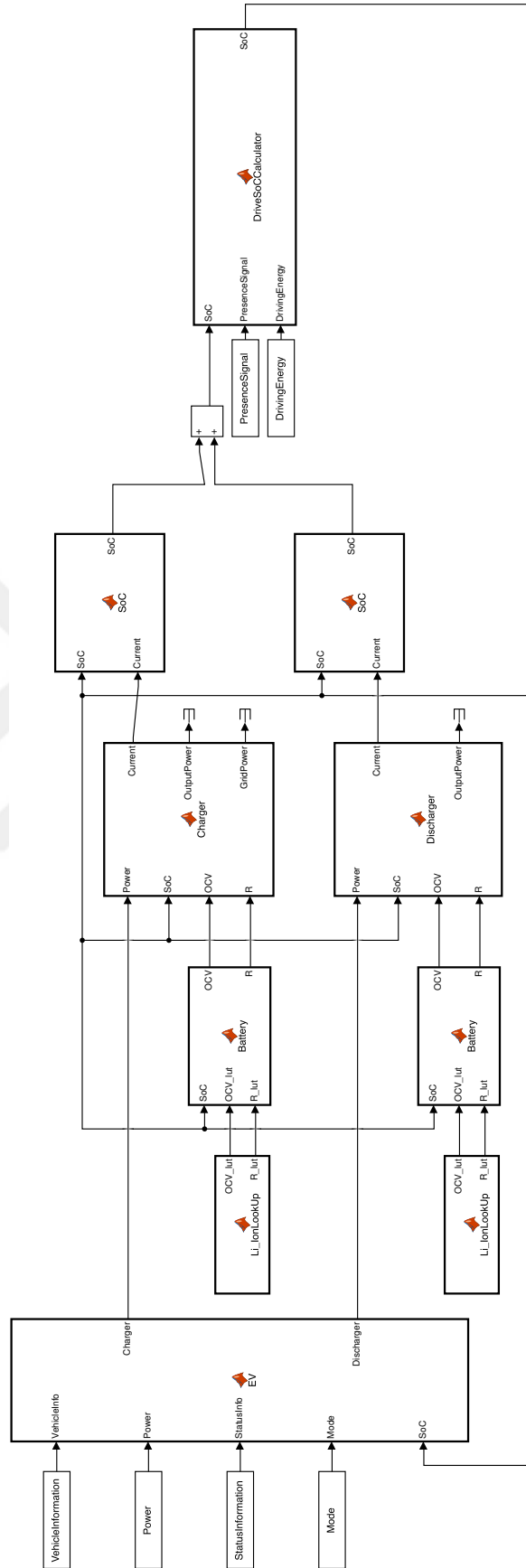


Figure 4.3: Simulink Model of Electric Vehicle (EV)

4.3 ENERGY TRADING & INVOICES MODEL

The model for the Equal Opportunities based Trading System was developed in MATLAB to simulate the behavior of Peer to Peer Energy Trading. The MATLAB model was constructed using the C programming language. The Trading System is composed of three distinct MATLAB functions, namely "Trade with Equal Opportunities," "Prepare Invoices," and "Allocate Power for Energy Trading," which are encapsulated within a MATLAB class. The "Trade with Equal Opportunities" function emulates the behavior of the entire Trading System. This function accepts two input arguments, namely the buyer and seller information, and produces a single output in the form of a transaction matrix. The transaction matrix includes transaction ID, buyer ID, seller ID, energy amount, and cost.

Table 4.18: Input Arguments of the "Trade with Equal Opportunities" Function

Argument	Description	Unit	Range
Buyer	Buyer information matrix	-	
Seller	Seller information matrix	-	

Table 4.19: Output Arguments of the "Trade with Equal Opportunities" Function

Argument	Description	Unit	Range
Transactions	The matrix includes the following information: transaction ID, buyer ID, seller ID, energy amount, and cost	-	

Algorithm 20 Trade with Equal Opportunities to compute Trading simulation

```
1: procedure Trade with Equal Opportunities (Buyer, Seller)
2:   for Number of Seller do
3:     while Number of Seller > 0 and Number of Buyer > 0 do
4:       Compute Transactions and set Seller-Buyer to current Seller-Buyer state
       by calculating necessary state
5:     end while
6:   end for
7: return Transactions
8: end procedure
```

The "Prepare Invoices" function is responsible for calculating the invoices associated with a given transaction for both buyers and sellers. This function accepts a single input argument and produces a single output in the form of an invoice matrix. The input argument is the transaction matrix, while the output argument is an invoice matrix that includes information such as home ID, bought amount, sold amount, total cost, and total profit.

Table 4.20: Input Arguments of the "Prepare Invoices" Function

Argument	Description	Unit	Range
Transactions	Transactions information matrix	-	

Table 4.21: Output Arguments of the "Prepare Invoices" Function

Argument	Description	Unit	Range
Invoices	The matrix includes the following information: home ID, bought amount, sold amount, total cost, and total profit	-	

Algorithm 21 Prepare Invoices to compute Trading invoices for each home

```

1: procedure Prepare Invoices (Transactions)
2:   for Number of Home do
3:     Compute Invoices by adding up all the Transactions values
4:   end for
5: return Invoices
6: end procedure

```

The "Allocate Power for Energy Trading" function is responsible for determining the energy allocation that occurs during the energy trading process. This function accepts two input arguments and produces a single output in the form of an allocated energy matrix. The input arguments consist of the invoices matrix and the amount of salable energy for each home. The output argument is an allocated energy matrix that includes information such as home ID, sold amount, sold amount from excessive energy, sold amount from Energy Storage System (ESS), and sold amount from Electric Vehicle (EV).

Table 4.22: Input Arguments of the “Allocate Power for Energy Trading” Function

Argument	Description	Unit	Range
Invoices	Invoices information matrix	-	
Salable Energy	Salable energy information matrix	-	

Table 4.23: Output Arguments of the “Allocate Power for Energy Trading” Function

Argument	Description	Unit	Range
Allocated Energy	The matrix includes the following information: home ID, sold amount from excessive energy, sold amount from ESS and sold amount from EV	-	

Algorithm 22 Allocate Power for Energy Trading to compute sold energy allocation for each home

- 1: **procedure** Allocate Power for Energy Trading (Invoices, Salable Energy)
 - 2: **for** Number of Seller **do**
 - 3: Compute Allocated Energy by using Invoice and Salable Energy matrices
 - 4: **end for**
 - 5: **return** Allocated Energy
 - 6: **end procedure**
-

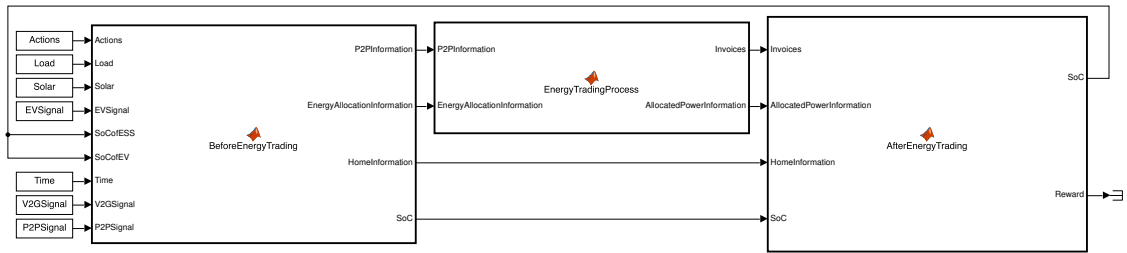


Figure 4.4: Simulink Model of Energy Trading System

4.4 INTEGRATED HOMES MODEL

The Integrated Home model conducts a comprehensive simulation of the Energy Management system. A MATLAB model was developed to simulate the behavior of an Integrated Home. The model was implemented using the C programming language within the

MATLAB environment. The Integrated Home model encompasses various functionalities, including Energy Storage Systems (ESS), Electric Vehicles (EV), Peer-to-Peer energy trading, and Vehicle-to-Grid capabilities.

The Integrated Home model was partitioned into three distinct sections: Before Energy Trading, Energy Trading Process, and After Energy Trading.

4.4.1 Before Energy Trading

In the initial section of the Integrated Home model, the energy consumption of the home is determined by considering the provided load and solar energy production inputs. Subsequently, the difference between the load and solar energy production inputs is calculated, and the charging/discharging functions for the Energy Storage System (ESS) and Electric Vehicle (EV) are computed. Following the computation of these charging/discharging functions, the available salable energy for Peer-to-Peer energy trading is calculated. The final step involves the calculation of EV charging costs for each home, if applicable.

Table 4.24: Input Arguments of the “Before Energy Trading” Function

Argument	Description	Unit	Range
Actions	Action information	-	-
Load	Load input of the home	W	0 to ∞
Solar	Solar energy production input of the home	W	0 to ∞
EV Signal	Signal represents whether EV is at home or not, Signal represents the change of Status signal, Signal represents daily driving energy data for daily vehicle ride	-	0 to 1
SoC of ESS	State of Charge information of ESS	-	0 to 1
SoC of EV	State of Charge information of EV	-	0 to 1
Time	Digital clock in real time	H	0 to ∞
V2G Signal	The signal represents whether the EV is equipped with Vehicle-to-Grid (V2G) functionality	-	0, 1
P2P Signal	The signal represents whether the Home is participated in Peer-to-peer (P2P) energy trading	-	0, 1

Table 4.25: Output Arguments of the “Before Energy Trading” Function

Argument	Description	Unit	Range
P2P Information	The matrix includes the following information: home ID, price, selling power, and buying power	-	-
Home Information	The matrix includes the following information: home ID, the difference, Load, Solar, Charged ESS Power, Discharged ESS Power, Charged EV Power, Discharged EV Power	-	-
Energy Allocation Information	The matrix includes the following information: home ID, total selling power, available ESS power, computed ESS power for selling, available EV power, computed EV power for selling	-	-
SoC	The matrix includes the following information: home ID, SoC of ESS, and SoC of EV	-	-

Algorithm 23 Before Energy Trading

```
1: procedure Before Energy Trading (Actions, Load, Solar, EV Signal, SoC of ESS,
   SoC of EV, Time, V2G Signal, P2P Signal)
2:   Compute main grid energy prices via “Supplier Price” function
3:   Calculate the difference between Load and Solar inputs
4:   Compute Energy Storage System (ESS) state via “ESS” function
5:   Calculate Electric Vehicle Mode:  $3 < Time < 6$ 
6:   if Vehicle Mode is Charging then
7:     Charge the EV via “EV.Charger” function
8:   else
9:     Discharge the EV via “EV.Discharger” function
10:  end if
11:  Compute Electric Vehicle (EV) charging cost via “Calculate EV Charging Cost”
   function
12:  return P2P Information, Home Information, Energy Allocation Information, SoC
13: end procedure
```

Algorithm 24 Supplier Price to compute main grid energy prices

```
1: procedure Supplier Price (Time))
2:   Compute main grid energy prices based on Time input
3:  return Selling Price, Buying Price
4: end procedure
```

Algorithm 25 Calculate EV Charging Cost to compute Electric Vehicle (EV) charging cost

```
1: procedure Calculate EV Charging Cost (Home Information))
2:   Compute Electric Vehicle (EV) charging cost based on Home Information input
3:  return EV Charging Cost
4: end procedure
```

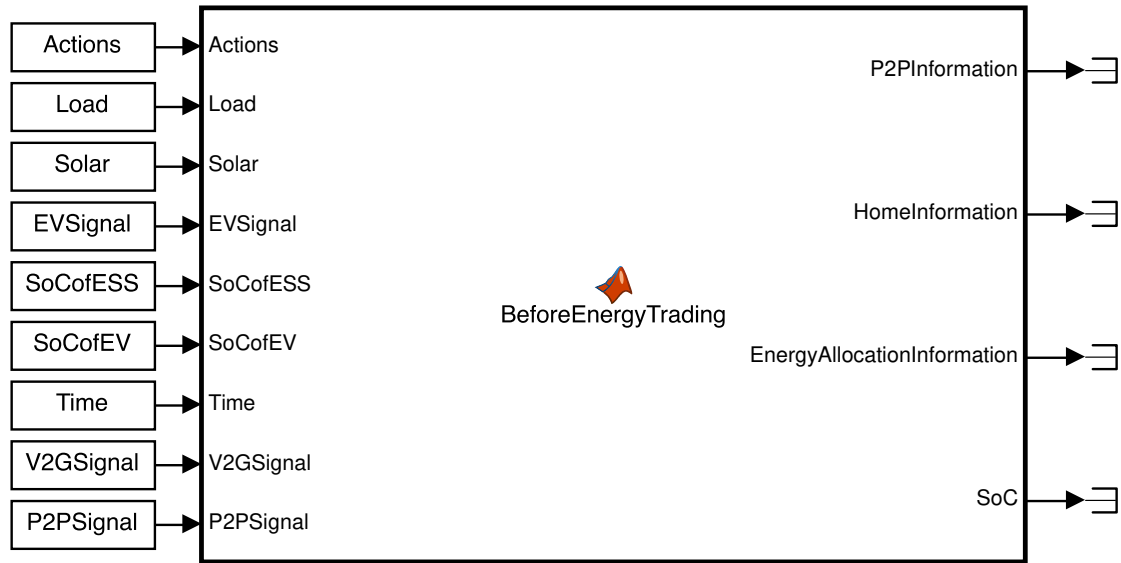


Figure 4.5: Simulink Model of Before Energy Trading Section

4.4.2 Energy Trading Process

In the second section of the Integrated Home model, Peer-to-Peer (P2P) energy trading process is evaluated by considering the provided inputs from the “Before Energy Trading”. Initially, the transaction matrix is obtained in the Energy Trading Process. After the transaction matrix is obtained, Invoices matrix is calculated based on the transaction matrix. The final step involves the power allocation for each home, if any trades have occurred.

Table 4.26: Input Arguments of the “Energy Trading Process” Function

Argument	Description	Unit	Range
P2P Information	The matrix includes the following information: home ID, price, selling power, and buying power	-	-
Energy Allocation Information	The matrix includes the following information: home ID, total selling power, available ESS power, computed ESS power for selling, available EV power, computed EV power for selling	-	-

Table 4.27: Output Arguments of the “Energy Trading Process” Function

Argument	Description	Unit	Range
Invoices	The matrix includes the following information: home ID, bought amount, sold amount, total cost, and total profit	-	-
Allocated Power Information	The matrix includes the following information: home ID, total sold power, sold power from excessive energy, sold power from ESS, and sold power EV	-	-

Algorithm 26 Energy Trading Process

- 1: **procedure** Energy Trading Process (P2P Information, Energy Allocation Information))
 - 2: Compute transaction matrix via “Trade With Equal Opportunities” function
 - 3: Compute invoices matrix via “Prepare Invoices” function
 - 4: Compute Allocated Power Information via “Allocate Powers for Selling” function
 - 5: **return** Allocated Power Information
 - 6: **end procedure**
-

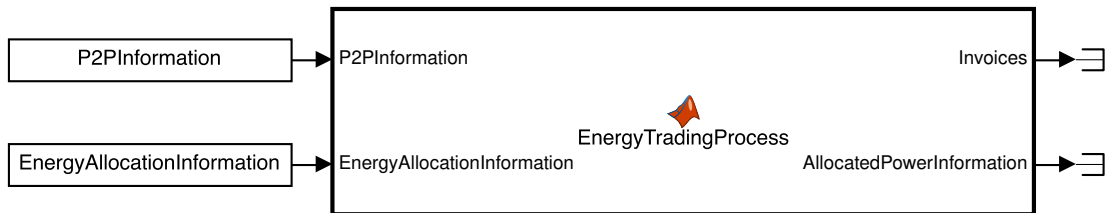


Figure 4.6: Simulink Model of Energy Trading Process

4.4.3 After Energy Trading

In the final section of the Integrated Home model, the new state of charge of ESS and EV are evaluated by considering the provided inputs from the “Energy Trading Process”. Initially, the energy bought from and sold to the main grid is calculated. After the main grid information is obtained, the bill of each house is calculated. The final step involves the new SoC and Reward calculation for each home.

Table 4.28: Input Arguments of the “After Energy Trading” Function

Argument	Description	Unit	Range
Invoices	The matrix includes the following information: home ID, bought amount, sold amount, total cost, and total profit	-	-
Home Information	The matrix includes the following information: home ID, the difference, Load, Solar, Charged ESS Power, Discharged ESS Power, Charged EV Power, Discharged EV Power	-	-
Allocated Power Information	The matrix includes the following information: home ID, total sold power, sold power from excessive energy, sold power from ESS, and sold power EV	-	-

Table 4.29: Output Arguments of the “After Energy Trading” Function

Argument	Description	Unit	Range
SoC	The matrix includes the following information: home ID, SoC of ESS, and SoC of EV	-	-
Reward	Numerical reward value for Reinforcement Learning computation	-	-

Algorithm 27 After Energy Trading

- 1: **procedure** After Energy Trading (Invoices, Home Information, Allocated Power Information)
 - 2: Compute bought energy from main grid via “Energy From Grid” function
 - 3: Compute sold energy to main grid via “Energy To Grid” function
 - 4: Compute the bill of energy usage via “Calculate Bills” function
 - 5: Compute the new SoC of ESS and EV via “SoC After Selling”
 - 6: Compute Numerical reward value for Reinforcement Learning computation via “Compute Rewards” function
 - 7: **return** SoC, Reward
 - 8: **end procedure**
-

Algorithm 28 Energy From Grid

- 1: **procedure** Energy From Grid (Home Information, Invoices)
 - 2: Compute bought energy from main grid using Home Information and Invoices
 - 3: **return** Bought Energy From Grid
 - 4: **end procedure**
-

Algorithm 29 Energy To Grid

- 1: **procedure** Energy To Grid (Home Information, Allocated Power Information)
 - 2: Compute sold energy to main grid using Home Information and Allocated Power Information
 - 3: **return** Sold Energy To Grid
 - 4: **end procedure**
-

Algorithm 30 Calculate Bills

- 1: **procedure** Calculate Bills (Invoices, Bought Energy From Grid, Sold Energy To Grid, EV Charging Cost, [Selling Price, Buying Price])
 - 2: Compute bill of energy usage using Invoices, Bought Energy From Grid, Sold Energy To Grid, EV Charging Cost, and [Selling Price, Buying Price]
 - 3: **return** Bill of Energy Usage
 - 4: **end procedure**
-

Algorithm 31 SoC After Selling

- 1: **procedure** SoC After Selling (SoC, Allocated Power Information, V2G Signal)
 - 2: Compute new SoC of Energy Storage System (ESS) via “ESS” function
 - 3: Compute new SoC of Electric Vehicle (EV) via “EV” function
 - 4: **return** SoC
 - 5: **end procedure**
-

Algorithm 32 Compute Rewards

- 1: **procedure** Compute Rewards (Home Information)
 - 2: Compute Numerical reward value for Reinforcement Learning computation
 - 3: **return** Reward
 - 4: **end procedure**
-

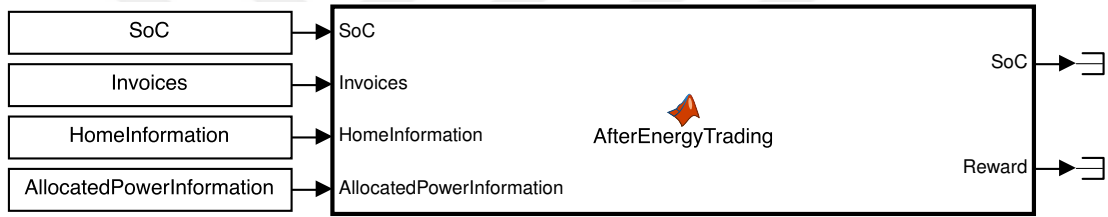


Figure 4.7: Simulink Model of After Energy Trading Section

4.5 APARTMENT BUILDING MODEL

The Apartment Building model includes “n” number of homes. All of the homes encapsulate the functionality of the “Integrated Home” model. A MATLAB and Simulink model was developed to simulate the behavior of “n” number of houses with Peer-to-Peer (P2P) energy trading. To ensure energy stability during renewable energy production uncertainties, the system includes an additional energy supplier model acting as the main grid. The Apartment Building model returns computed State of Charges of ESS and EV matrix, numerical Reward values matrix to be used in Reinforcement Learning computation, and Information matrix composed of required information to be used in report purposes. In this project, the number of homes in the “Apartment Building” model, “n”, was selected as 6.

Table 4.30: Input Arguments of the “Apartment Building” Function

Argument	Description	Unit	Range
V2G Signals	The matrix includes the following information: home ID, bought amount, sold amount, total cost, and total profit	-	-
P2P Signals	The matrix includes the following information: home ID, the difference, Load, Solar, Charged ESS Power, Discharged ESS Power, Charged EV Power, Discharged EV Power for each home	-	-
Actions	The matrix includes the following information: home ID, total sold power, sold power from excessive energy, sold power from ESS, and sold power EV for each home	-	-
Loads	The matrix includes the following information: home ID, load (W) for each home	-	-
Loads	Signal represents whether EV is at home or not, for each home, Signal represents the change of Status signal, for each home, Signal represents daily driving energy data for daily vehicle ride, for each home	-	-
SoCs	The matrix includes the following information: home ID, SoC of ESS, and SoC of EV for each home	-	-
Time	Digital clock in real time	-	-

Table 4.31: Output Arguments of the “Apartment Building” Function

Argument	Description	Unit	Range
Rewards	The matrix includes the following information: home ID, numerical Reward values for each home	-	-
Infos	The matrix includes the following information: home ID, load, solar production info, Charged ESS Power, Discharged ESS Power, Sold ESS Power, Charged EV Power, Discharged EV Power, Sold EV Power, Energy Bought from Grid, Energy Sold to Grid, Bill, Energy Bought via P2P, Energy Sold via P2P, SoC of ESS, SoC of EV, for each home	-	-
SoCs	The matrix includes the following information: home ID, SoC of ESS, and SoC of EV for each home	-	-

Algorithm 33 Apartment Building

- 1: **procedure** Apartment Building (V2G Signals, P2P Signals, Actions, Loads, Solars, EV Signals, SoCs, Time)
 - 2: **for all** home **do**
 - 3: Compute numerical Reward values, Home Information, and SoCs of ESS/EV
 - 4: **end for**
 - 5: **return** Rewards, Infos, SoCs
 - 6: **end procedure**
-

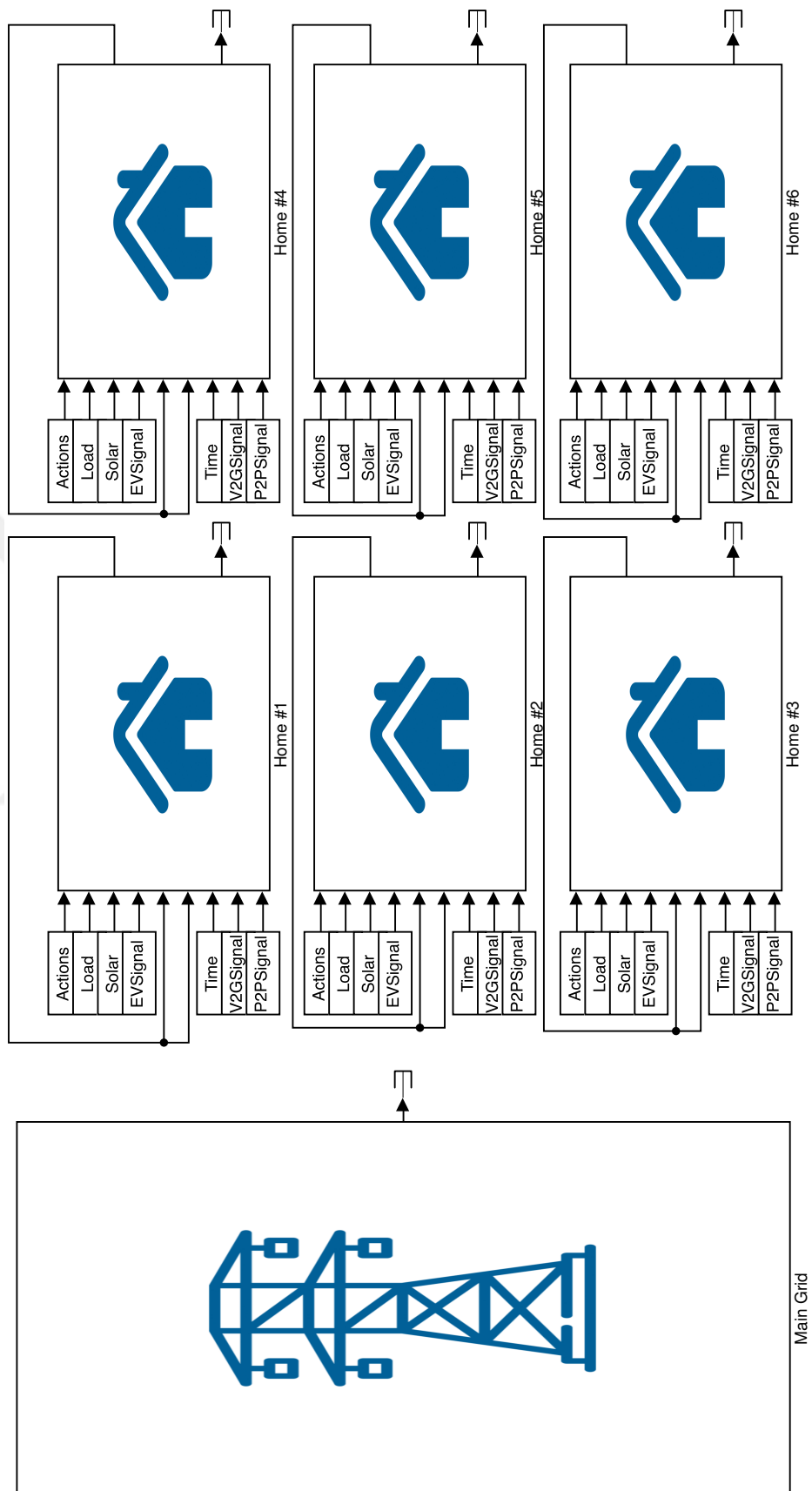


Figure 4.8: Simulink Model of Apartment Building Model

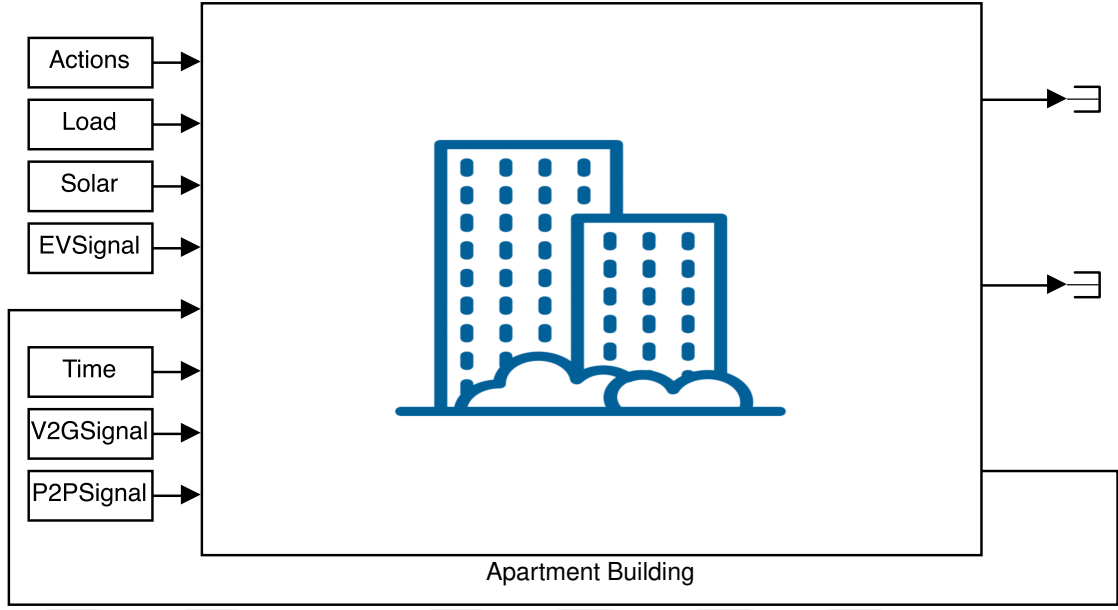


Figure 4.9: Comprehensive Simulink Model of Apartment Building Model

4.6 REINFORCEMENT LEARNING (RL) MODEL

The Reinforcement Learning (RL) model includes the Apartment Building model as the environment and as many agents as the number of homes “n”. As mentioned earlier, the Apartment Building model consists of 6 homes with P2P energy trading capability. Therefore, the Simulink model includes 6 “RL Agent” built-in blocks to be used in the learning process. Built-in RL Agent block takes 3 input arguments namely, “observation”, “reward”, and “isdone”. “Isdone” input takes a signal to terminate the learning process when all the requirements are satisfied. In this project, “isdone” input takes constant false (0) value due to environment specifications. “Reward” input takes the numerical reward values of each home in the environment through the “Apartment Building” model.

4.6.1 States

“Observation” input takes the states of each home in the environment through the “States” MATLAB function. “States” input represents the state of the environment during the learning process. Model includes 7 distinct information per home as the state of the environment. “rINumericSpec” built-in MATLAB function is used to represent continuous state information of each home.

$$S_t^n = \left\{ Load_t^n, Solar_t^n, SoC_t^{ESS,n}, SoC_t^{EV,n}, Price_{buying}, Price_{selling} \right\}$$

where $Load_t^n$ is load demand, $Solar_t^n$ is solar power generation, $SoC_t^{ESS,n}$ current SoC of ESS, $SoC_t^{EV,n}$ current SoC of EV, $Price_{buying}$ is energy buying price of main grid, and

$Price_{selling}$ is energy selling price of main grid at time t .

Table 4.32: States of the Environment

Argument	Description	Unit	Range
Load	Load input to the home	W	0 to ∞
Solar	Produced solar energy input to the home	W	0 to ∞
ESS SoC	Current state of charge of ESS	-	0 to 1
EV SoC	Current state of charge of EV	-	0 to 1
EV Presence Signal	Signal represents whether EV is at home or not	-	0 or 1
Buying Price	Main grid energy selling price	TRY/kWh	0 to ∞
Selling Price	Main grid energy buying price	TRY/kWh	0 to ∞

4.6.2 Action Spaces

Output of the built-in RL Agent block is a set of actions defined in a discrete domain. The actions from the built-in RL Agent feeds the input of the environment. Environment takes 5 distinct action inputs per home from each agent to simulate the ‘‘Apartment Building’’ model. ‘‘rlFiniteSetSpec’’ built-in MATLAB function is used to represent discrete action space of each home.

$$A_t^n = \left\{ A_{action,t}^{ESS,n}, A_{amount,t}^{ESS,n}, A_{action,t}^{EV,n}, A_{amount,t}^{EV,n}, A_{price,t} \right\}$$

where A_{action}^{ESS} is action signal of ESS, A_{amount}^{ESS} is action signal to determine amount of power delivery from ESS in P2P energy trading, A_{action}^{EV} is action signal of EV, A_{amount}^{EV} is action signal to determine amount of power delivery from EV in P2P energy trading, A_{price} is action signal to determine price multiplier to be used in P2P energy trading at time t .

Table 4.33: Actions of the Environment

Argument	Description	Unit	Range
Battery Signal	Signal that represents whether ESS will be participated in P2P	-	0 to 1
Battery Amount	Signal that represents how much energy will be allocated from ESS to be used in P2P (in %)	-	0.2 to 1.0
EV Signal	Signal that represents whether EV will be participated in P2P	-	0 or 1
EV Amount	Signal that represents how much energy will be allocated from EV to be used in P2P (in %)	-	0.2 to 1.0
Price Signal	Signal that represents multiplier of the main grid selling price (in %)	-	0.60 to 0.95

All the possible action combinations in the action space of each home were formatted as arrays due to MATLAB computation requirements. Formatted actions examples to be used in learning are as follows.

Table 4.34: Formatted Actions Example to be used in learning

#	Battery Signal	Battery Amount	EV Signal	EV Amount	Price Signal	Action Array
1	1	0.75	0	0.65	0.72	[1, 0.75, 0, 0.65, 0.72]
2	0	0.70	1	0.85	0.68	[0, 0.70, 1, 0.85, 0.68]
3	1	0.45	1	0.25	0.84	[1, 0.45, 1, 0.25, 0.84]

4.6.3 Objective (Reward) Function

Reward outcome of the environment is composed of a scalar value and computed by the “Apartment Building” model. In RL applications, the reward consists of penalties and rewards to maximize the utility function and make an agent know environmental boundaries.

$$R_t^n = \left[\left(0.7 * \left(\sum (bills) + bills_t^n \right) \right) - \left(0.3 * \left(\sum (grids) + grids_t^n \right) \right) \right]$$

The first part of the reward function represents maximization of the bill values in normalized range as reward. The first constant represents the weight of the maximization of the bill values. Therefore, 70% of the reward function consists of the bill values. To achieve equilibrium in game theory, summation of the bill values of each home were added to the reward function. The second part of the reward function represents minimization of energy bought from the main grid in normalized range as penalty. The first constant represents the weight of the minimization of the grid values. Therefore, 30% of the reward function consists of the grid values. To achieve equilibrium in game theory, summation of energy bought from the main grid of each home was added to the reward function.



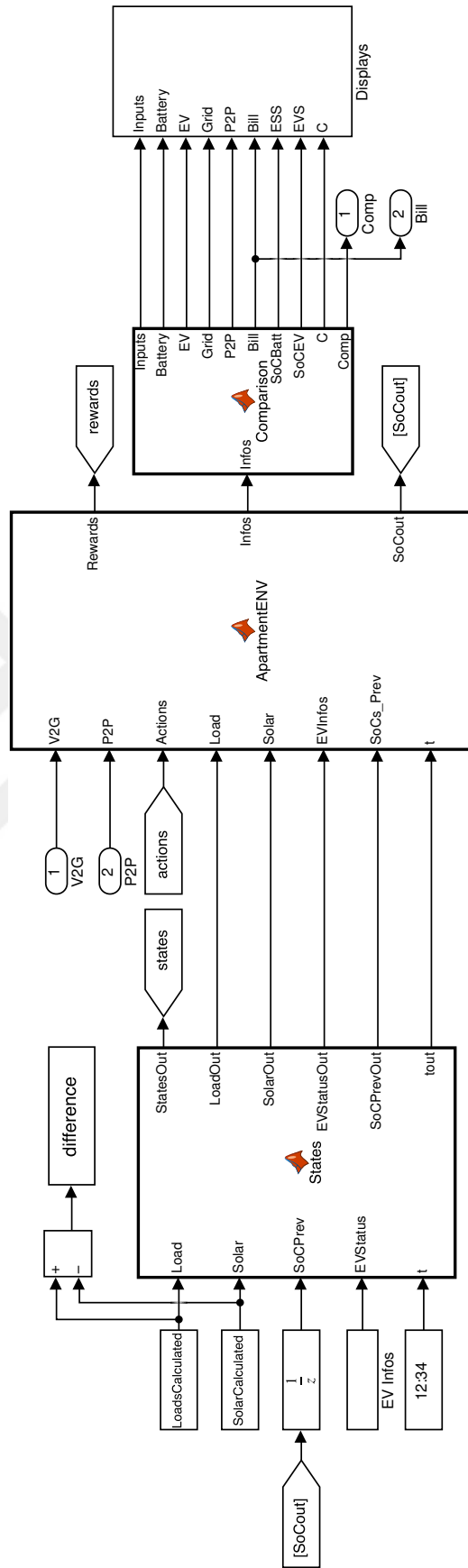


Figure 4.10: Environment in Simulink

4.6.4 Reset Function

Reset function resets the initial states to random states. Environment includes 2 random initial states namely SoC of ESS, and SoC of EV.

Algorithm 34 Reset Function

```
1: procedure Reset Function
2:   for all Home do
3:     Set SoC of ESS to a random value
4:     Set SoC of EV to a random value
5:   end for
6: return Transactions
7: end procedure
```

4.6.5 Deep Neural Network (DNN) Design

In the Double Deep Q-Network algorithm, actor-critic method uses Deep Neural Networks to update the critic. Critic updates the Q-Values through DNN. In the DDQN algorithm of the RL, DNN includes 4 fully connected hidden layers with Rectified Linear Unit (ReLU) activation function. DNN includes a feature input layer with the same input size as the state number of the environment. Output layer of the DNN is a fully connected layer with node number of total action size. Total action size is defined as all the possible action combinations from the action space.

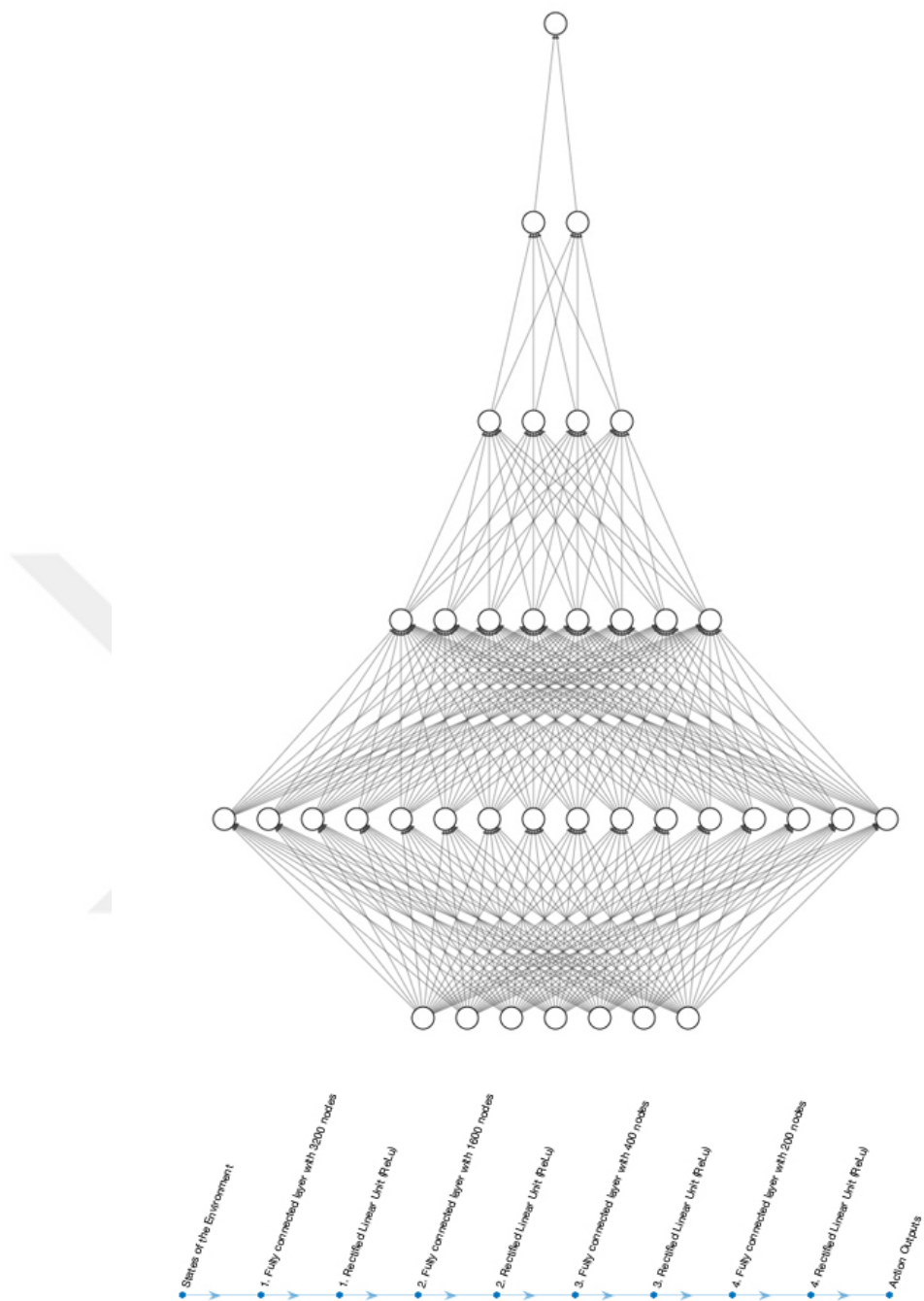


Figure 4.11: Deep Neural Network Design of the Actor-Critic in DDQN

Designed DNN includes 6,388,792 parameters to be trained through DDQN. Learning rate of DNN is defined as 0.001. Summary table of the designed DNN is as follows.

Table 4.35: Summary Table of DNN

#	Type	Nodes	Activation Function	Output Size	Trainable Parameters
1	Input Layer	7	-	7	-
2	Hidden Layer	3200	ReLu	3200	25,600
3	Hidden Layer	1600	ReLu	1600	5,121,600
4	Hidden Layer	400	ReLu	400	640,400
5	Hidden Layer	200	ReLu	200	80,200
6	Output Layer	2592	-	2592	520,992

4.6.6 Deep Q-Learning (DQN) Agent Design

Each agent in the environment is represented by a “rlDQNAgent” built-in MATLAB DQN agent object. MATLAB provides all the functionality of a DQN agent with adjustable options such as batch size, buffer length, discount factor etc. Each agent in the environment include exactly the same agent options. To start a DDQN algorithm, the critic $Q(s, a; \phi)$ should be initialized with random ϕ value. After critic initialization, a random set of actions from action space should be selected with ϵ through greedy exploration value for the current observation S_t . If the greedy exploration done, then select the set of actions that maximizes the critic value function.

$$A_t^n = \operatorname{argmax} Q(S_t^n, A^n; \phi)$$

After executing set of actions A_t^n , current states S_t^n , selected set of actions A_t^n , rewards R_t^n , new states S_{t+1}^n are stored in the experience buffer. Minibatch (M) (S_i, A_i, R_i, S'_i) from the experience buffer is sampled. The value function target y_i is computed through sampled minibatch.

$$A_{max} = \operatorname{argmax} Q(S'_i, A'; \phi)$$

$$y_i = R_i + \gamma Q_t(S'_i, A_{max}; \phi_t)$$

where γ is the discount factor of long-term reward. Loss function is calculated across all sampled experiences M .

$$L = \frac{1}{2M} \sum_{i=1}^M (y_i - Q(S_i, A_i; \phi))^2$$

In the final phase, randomly selected ϕ should be updated according to choose update method.

$$\phi_t = \tau\phi + (1 - \tau)\phi_t$$

where τ is smoothing factor. With the updated ϕ value, training should be repeated for the number of step. Agent options table of each agent is as follows.

Table 4.36: Agent Options of Built-in MATLAB DQN Agent Class

Property	Description	Value
Use Double DQN	Signal to perform learning with DDQN algorithm	True
Epsilon Min	Minimum value of Epsilon for epsilon-greedy exploration	0.1
Epsilon Decay	Decay rate of Epsilon for epsilon-greedy exploration	0.00045
Sample Time	Sample time of agent	1s
Target Smooth Factor	Number of steps required to update target critic of DNN	0.01
Discount Factor	Discount factor to control the impact of the future rewards in learning	0.965
Experience Buffer Length	Size of the buffer that stores the sampled states, actions, rewards, and next states	1,000,00
Mini Batch Size	Size of random experience that agent samples from the buffer	16

Visual presentation of agents in Simulink is as follows.

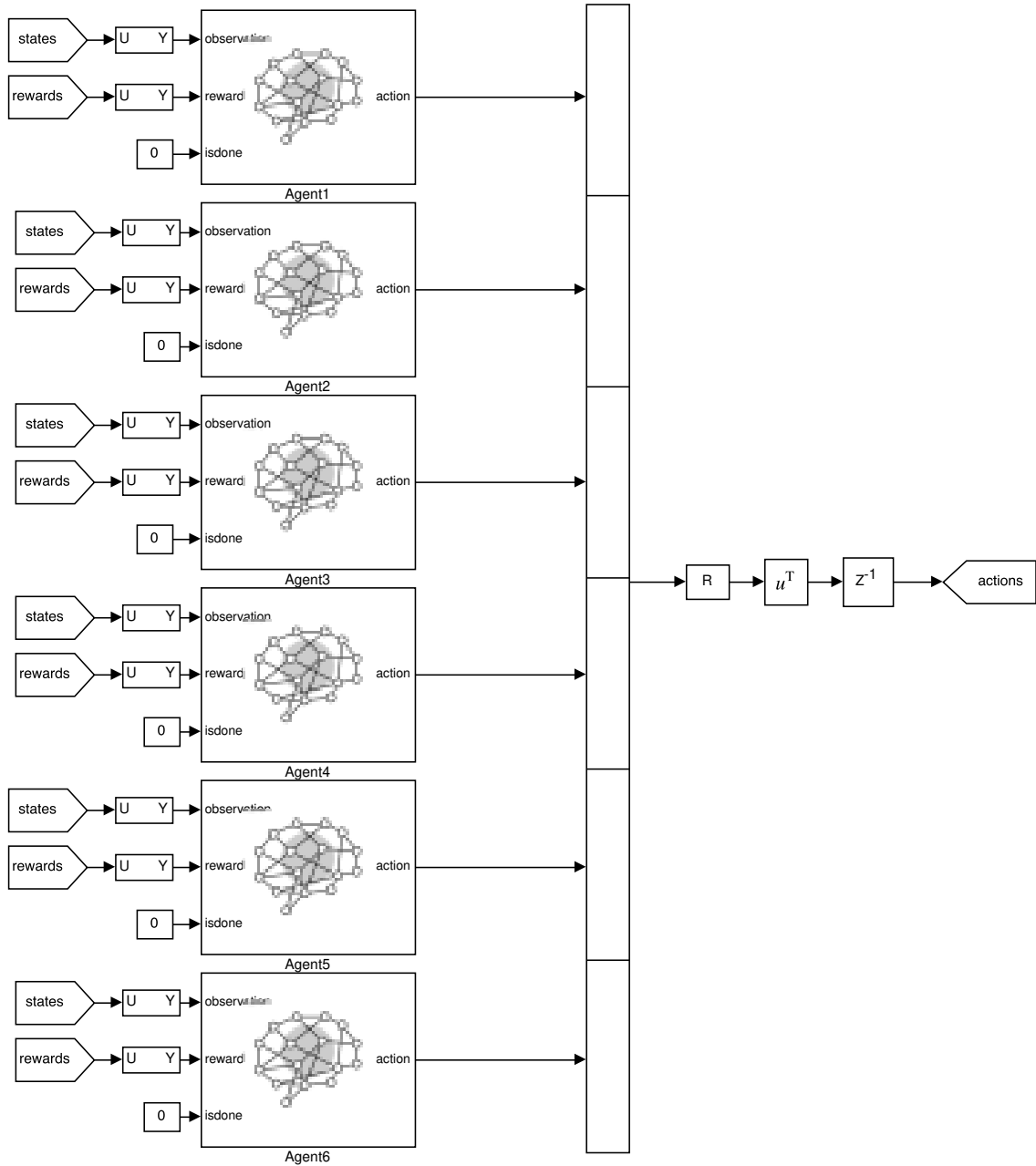


Figure 4.12: Deep Q-Network Agents (DQNs) in Simulink

4.6.7 Training Options

Built-in “train” function is used to train multiple agents environment in MATLAB. “Train” function accepts 3 arguments namely, agents, environment, and training options. Training options define training behaviors of learning via built-in “rlMultiAgentTrainingOptions” object. Training options table of learning is as follows.

Table 4.37: Training Options of Built-in MATLAB “Train” Function

Property	Description	Value
Agent Groups	Groups the agents into given positive integers	Auto
Learning Strategy	Defines the learning strategy of each agent as dependent or independent	Decentralized
Maximum Episodes	Maximum number of episodes to train the agents	2000
Max Steps per Episode	Maximum number of steps to run per episode	120
Score Averaging Window Length	Window length to be used in averaging the reward values of each agent	20
Stop Training Criteria	Training termination condition	Episode Count
Stop Training Value	Critical value of the training termination condition	2000

CHAPTER 5 CASE STUDY and SIMULATION RESULTS

To create diversity in the data to be used in training and testing phases, data were collected from multiple sources. Not-available data such as electric vehicle status signals were produced using statistical methods in accordance with the collected data. Collected data, that covers 1 year in total, were divided into 2 different sub datasets as 5 days of training data and 10 days of testing data. To show the effectiveness of the training, completely different testing dataset from the training dataset were selected. While 1 day dataset contains 24 separate sub-data, 5 day training data set contains 120, 10 day training data set contains 240 sub-data. Each data array includes load input information, solar energy production information, electric vehicle status signal information, derivative of electric vehicle status signal, and electric vehicle daily energy consumption information. The load input represents the power information required by the house at time " t " in *Watts*. Solar energy production information represents the power information produced by the solar panels at time " t " in *Watts*. The electric vehicle status signal represents the information whether the electric vehicle is at home at time t as $[0, 1]$, if any. The derivative signal of the electric vehicle status represents the electric vehicle departure status at time t as $[0, 1]$, if any. Electric vehicle daily energy consumption information represents the amount of energy (*kWh*) consumed by the electric vehicle in a single day during transportation, if any. Other data that belong to electric vehicles were produced using statistical methods in accordance with the collected data. Daily energy consumption information of electric vehicles were produced from the consumption information of vehicles with different size of batteries and electric motors in the current market. At *time* = 9, sample data of 6 different houses are as follows.

Table 5.1: Sample Data

Home #	Load (W)	Solar (W)	EV Status	Derivative Signal	EV Daily Energy Consumption (kWh)
1	751	300	0	1	6.44
2	1247	300	1	0	9.93
3	490	300	0	0	0
4	220	300	1	0	3.53
5	603	300	0	0	0
6	870	300	0	0	0

The Simulink sampling time of the simulation was set to 1 second, and the sampling time of the data was set to 60 minutes. In the simulation, the energy selling and buying prices of the local electricity producer company are used. Hourly selling/buying pricing information is as follows.

Table 5.2: Main Grid Energy Buying/Selling Prices

Start Time	End Time	Selling Price (TRY/kWh)	Buying Price (TRY/kWh)
06:00	17:00	1.95	0.32
17:00	22:00	2.86	0.32
22:00	06:00	1.22	0.32

The relevant parameters of the batteries of the energy storage system for each house are as follows.

Table 5.3: Battery Parameters of Energy Storage Systems

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Capacity (kWh)	4.8	4.8	4.8	4.8	4.8	4.8
Max SoC	1.0	1.0	1.0	1.0	1.0	1.0
Min SoC	0.1	0.1	0.1	0.1	0.1	0.1
Charging Efficiency (%)	95	95	95	95	95	95
Discharging Efficiency (%)	95	95	95	95	95	95
Maximum Discharging Power (W)	2000	2000	2000	2000	2000	2000
Maximum Charging Power (W)	5000	5000	5000	5000	5000	5000

The electric vehicle and V2G functionality parameters for each house are as follows.

Table 5.4: Electric Vehicle and V2G Functionality Parameters

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Electric Vehicle	1	1	0	1	0	0
V2G Functionality	1	1	0	1	0	0

The relevant parameters of electric vehicles for each house are as follows.

Table 5.5: Electric Vehicle Parameters

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Capacity (kWh)	36	4.8	-	4.8	-	-
Max SoC	0.8	1.0	-	1.0	-	-
Min SoC	0.2	0.1	-	0.1	-	-
Target SoC	0.65	0.65	-	0.65	-	-
Discharging Efficiency (%)	95	95	-	95	-	-
Charging Efficiency (%)	92	92	-	92	-	-
Maximum Discharging Power (W)	1000	1000	-	1250	-	-
Charging Power (W)	11000	7400	-	7400	-	-

The relevant parameters of the batteries of electric vehicles are as follows.

Table 5.6: Battery Parameters of Electric Vehicles

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Series Cells	96	96	-	92	-	-
Parallel Cells	35	40	-	38	-	-
Nominal Voltage (V)	3.6	3.6	-	3.6	-	-
Fully Charged Voltage (V)	4.2	4.2	-	4.2	-	-
Battery Capacity (Ah)	3.5	2.5	-	2	-	-

5.1 STATISTICAL ANALYSIS on TRAINING DATA

A 5-day (120 data) data set in random sequence was selected from the data pool to be used in the reinforcement learning training process. A fixed seed value has been assigned so that the training can be done with the same data throughout the learning process. The graph of the load input data selected to be used in the training process is as follows.

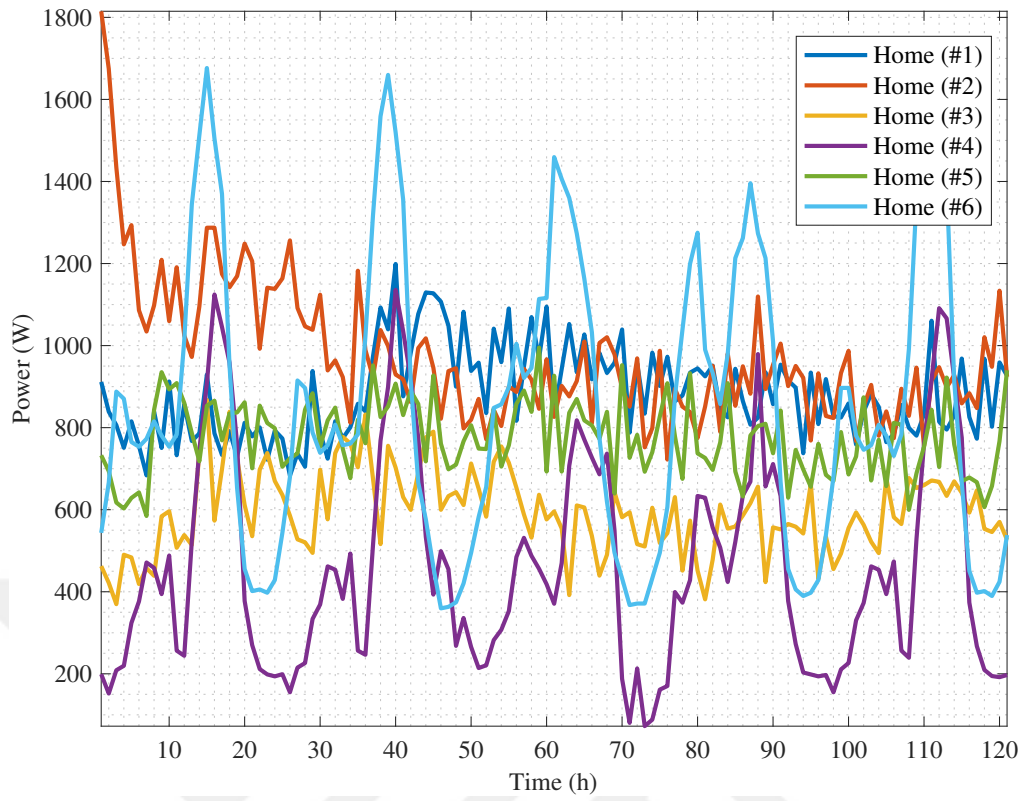


Figure 5.1: Load Power Data (Training, Complete)

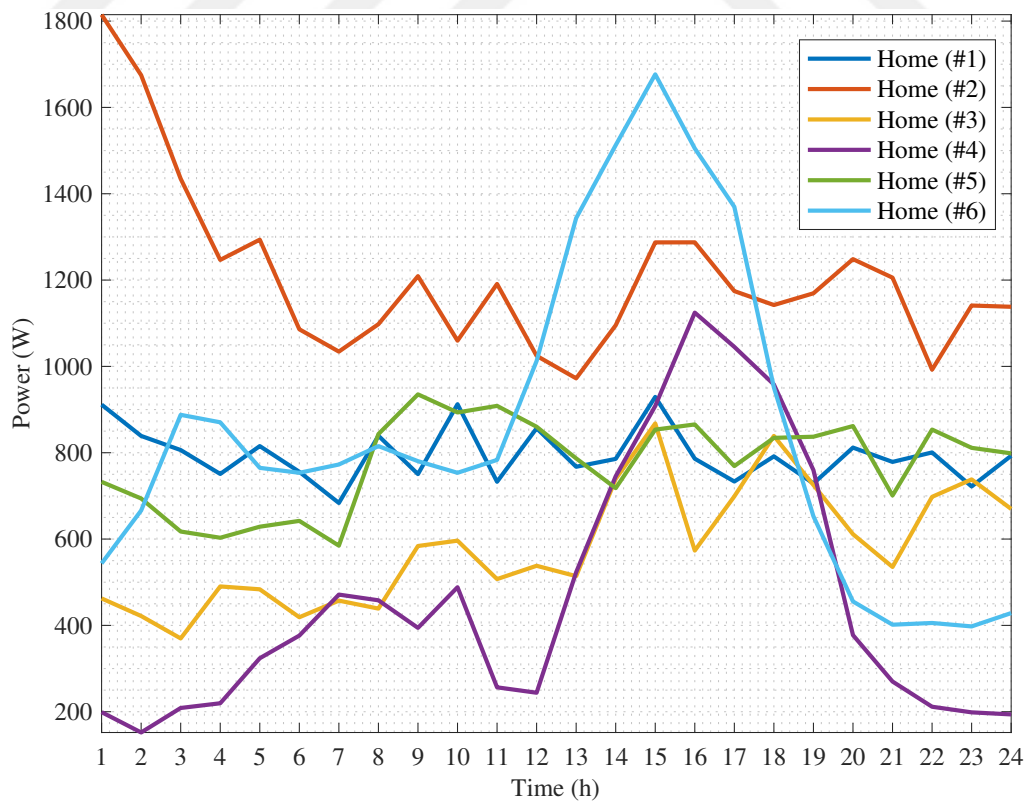


Figure 5.2: Load Power Data (Training, Daily)

The statistical analysis table of the load data used in the training process is as follows.

Table 5.7: Descriptive Statistical Analysis of Load Power Data (Training)

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Mean	794	1207	585	452	774	842
Median	786	1169	573	376	799	773
Max	929	1814	867	1124	935	1676
Min	683	972	369	151	584	197
Range	245	842	498	972	350	1278
Variance	3787	37779	18169	88410	10776	13920
Standard Devia- tion	64	194	134	297	103	373
Percentile (25%)	750	1092	478	210	699	543
Percentile (50%)	786	1169	573	376	498	772
Percentile (75%)	821	1258	698	578	855	966
Interquartile Ranges	70	465	219	368	156	422
Skewness	0.62847	1.7183	0.42303	1.0213	-0.3731	0.82012

As can be seen from the analysis results, the load data of the training set of each house show different characteristics. The second house with the highest mean and value range, and the fourth house load with the lowest mean and value range. As can be seen from the maximum and minimum load values, the data has high variability over a wide spectrum. High data diversity supports more stable decision making processes with less data in reinforcement learning algorithms.

The graph of the solar energy production data selected to be used in the training process is as follows.

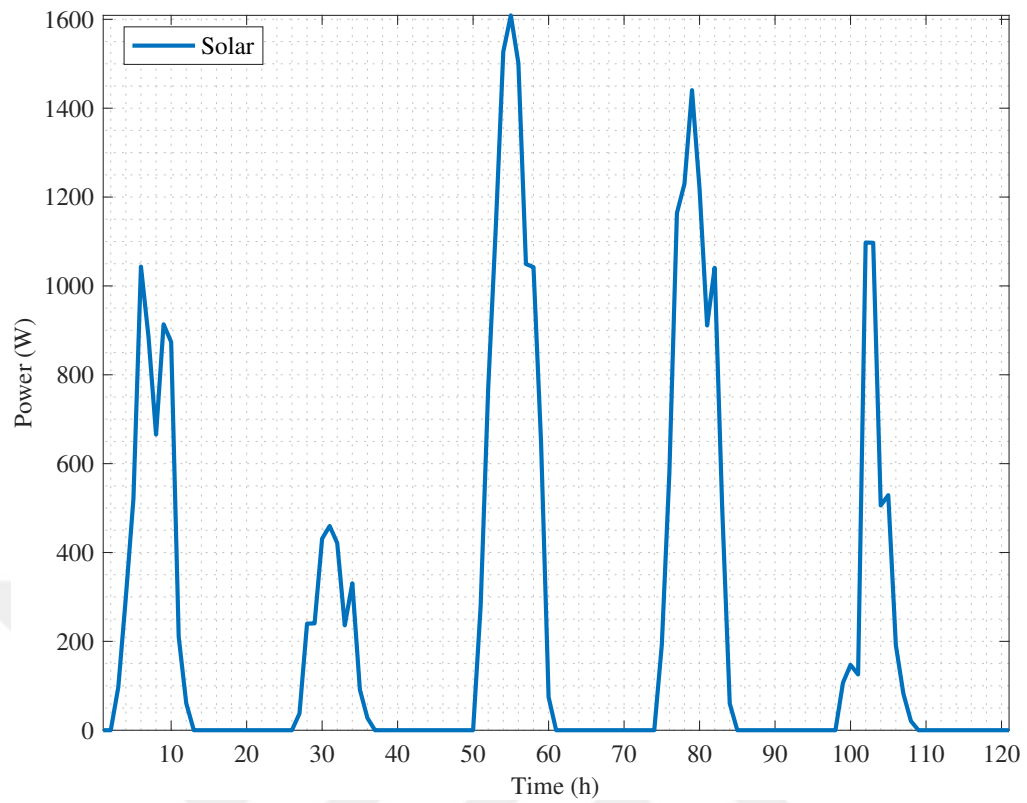


Figure 5.3: Solar Energy Production Data (Training, Complete)

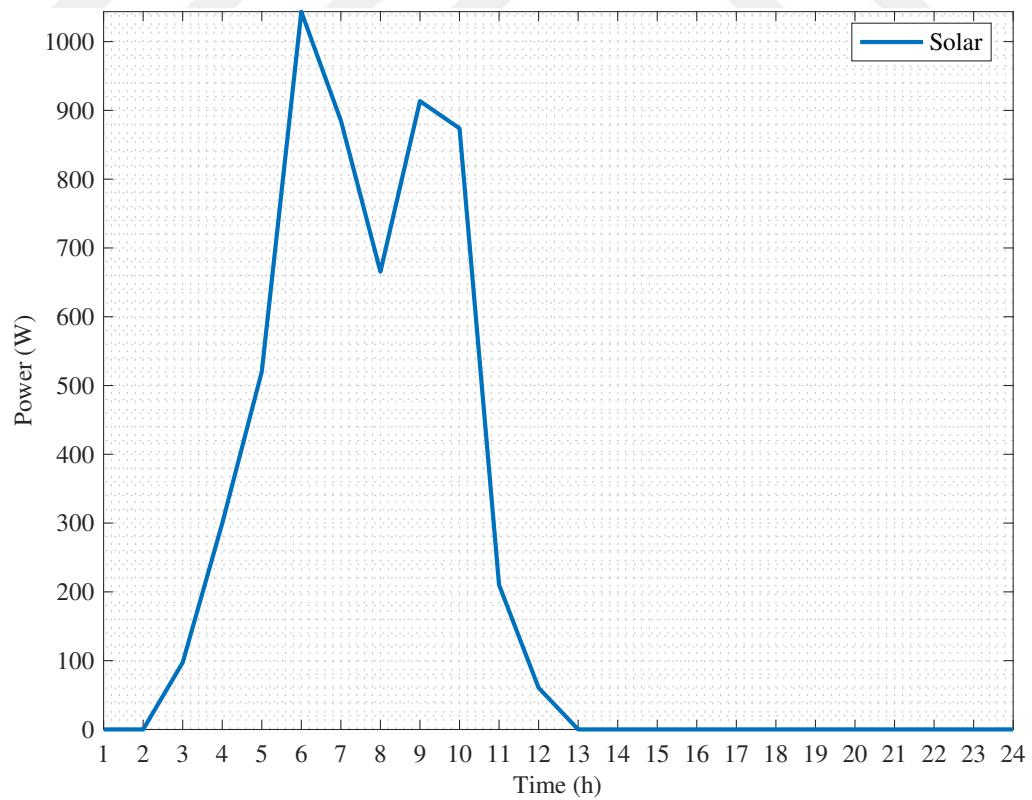


Figure 5.4: Solar Energy Production Data (Training, Daily)

The statistical analysis table of the solar energy production amount data used in the training process is as follows. It is assumed that each house benefits equally from the total amount of solar energy produced.

Table 5.8: Descriptive Statistical Analysis of Solar Energy Production Data (Training)

	Total Production (W)	Production Per Home (W)
Mean	1336	222
Median	0	0
Max	6260	1043
Min	0	0
Range	6260	1043
Variance	4.63e6	1.28e5
Standard Devia- tion	2153	358
Percentile (25%)	0	0
Percentile (50%)	0	0
Percentile (75%)	2128	354
Interquartile Ranges	2128	354
Skewness	1.2797	1.2797

As can be seen from the analysis results, the solar power generation data shows a typical solar panel generation characteristic. The generation characteristic has the lowest energy production average range in the morning and evening times, and the highest energy production range at noon. Values lower than the threshold value are set to 0 for ease of operation.

The graph of the "EV Status" data produced by statistical methods to be used in the training process, in accordance with other available data, is as follows.

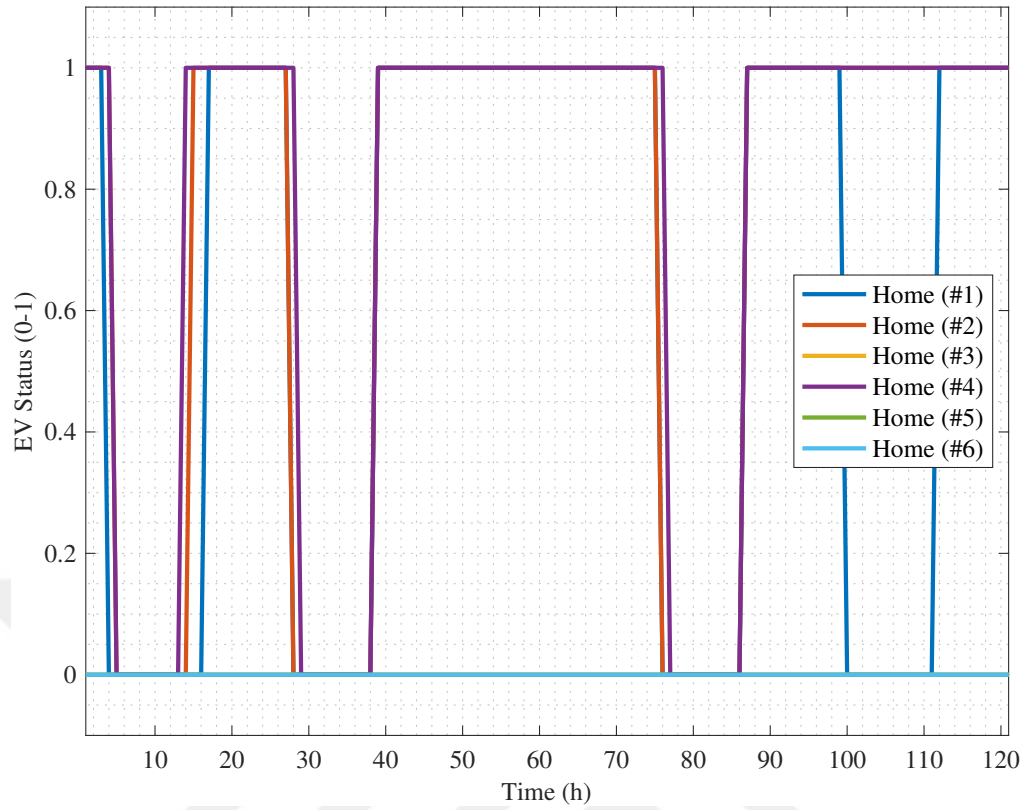


Figure 5.5: Electric Vehicle Status Signal Data (Training, Complete)

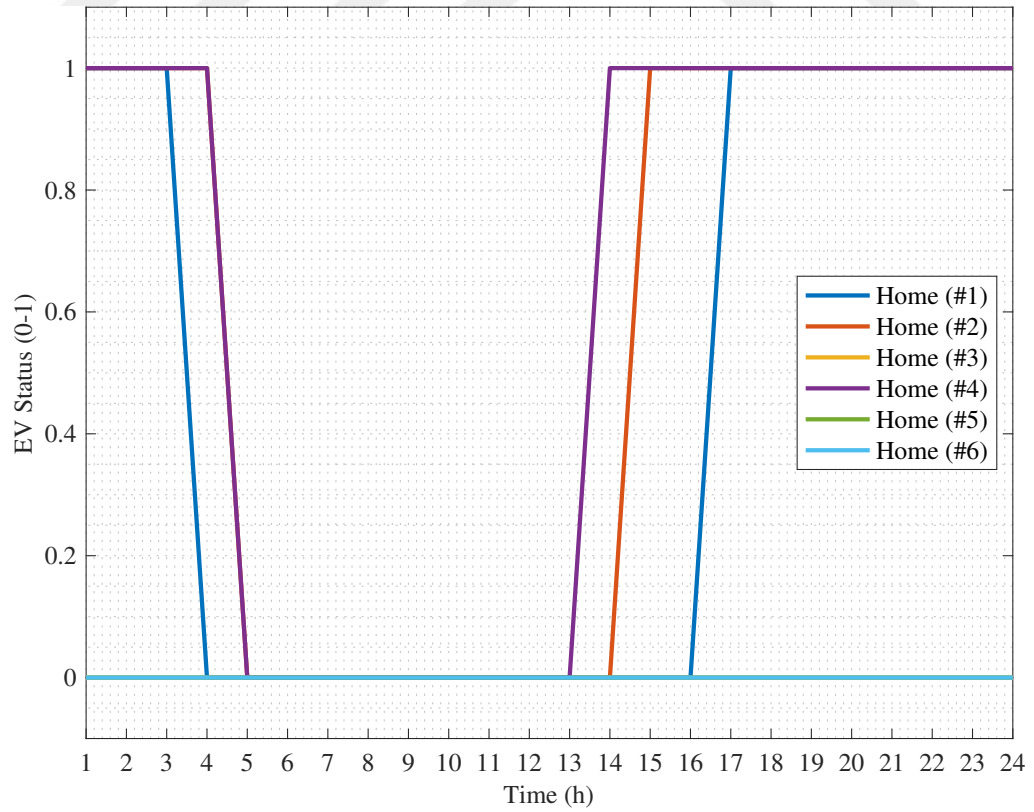


Figure 5.6: Electric Vehicle Status Signal Data (Training, Daily)

The table below shows the total hours count that electric vehicles are at home or not, if applicable.

Table 5.9: Descriptive Statistical Analysis of Electric Vehicle Status Data (Training)

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Status “1”	12	15	0	16	0	0
Status “0”	13	10	0	9	0	0

As can be seen from the table above, electric vehicle status signals, if any, were produced with an appropriate characteristic, based on city life and working hours, during a 1-day period. Another produced data represents the derivatives of the existing “EV Status” data. Among the houses with electric vehicles, there were no electric vehicles between 09:00 and 21:00 in house #1, between 10:00 and 19:00 in house #2, and between 10:00 and 18:00 in house #4.

Table 5.10: Departure and Arrival Hours of Electric Vehicles (Training)

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Departure Hour	09:00	10:00	00:00	10:00	00:00	00:00
Arrival Hour	21:00	19:00	00:00	18:00	00:00	00:00

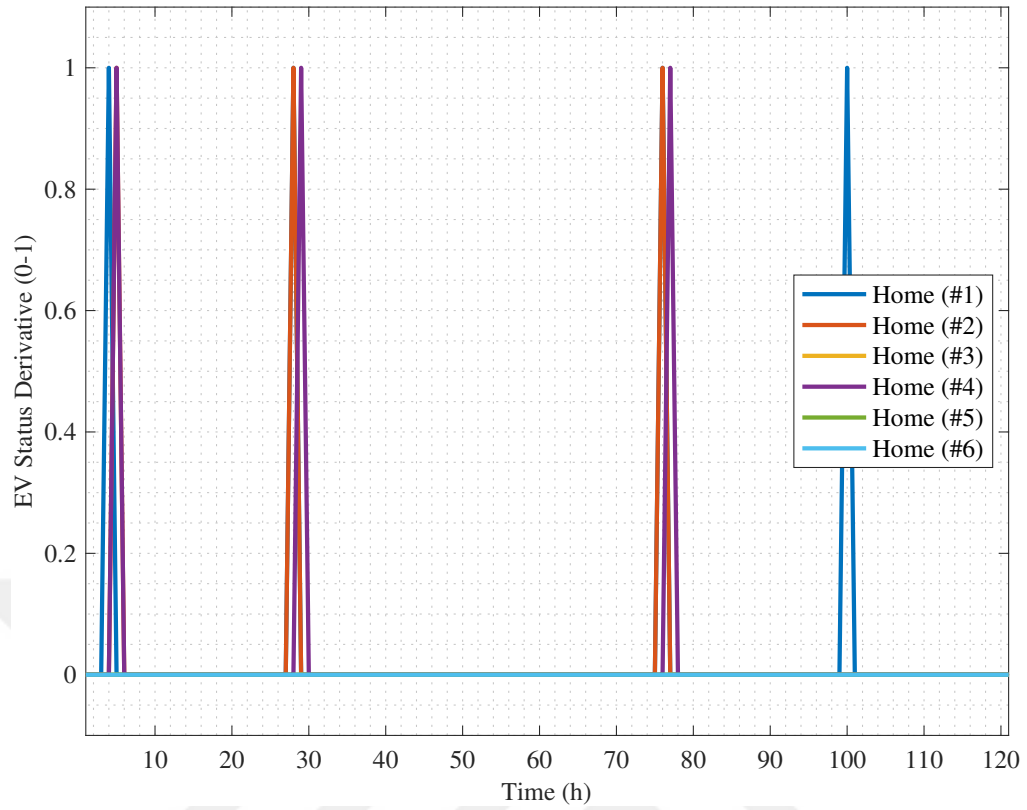


Figure 5.7: Derivative Signal Data of Electric Vehicle Status (Training, Complete)

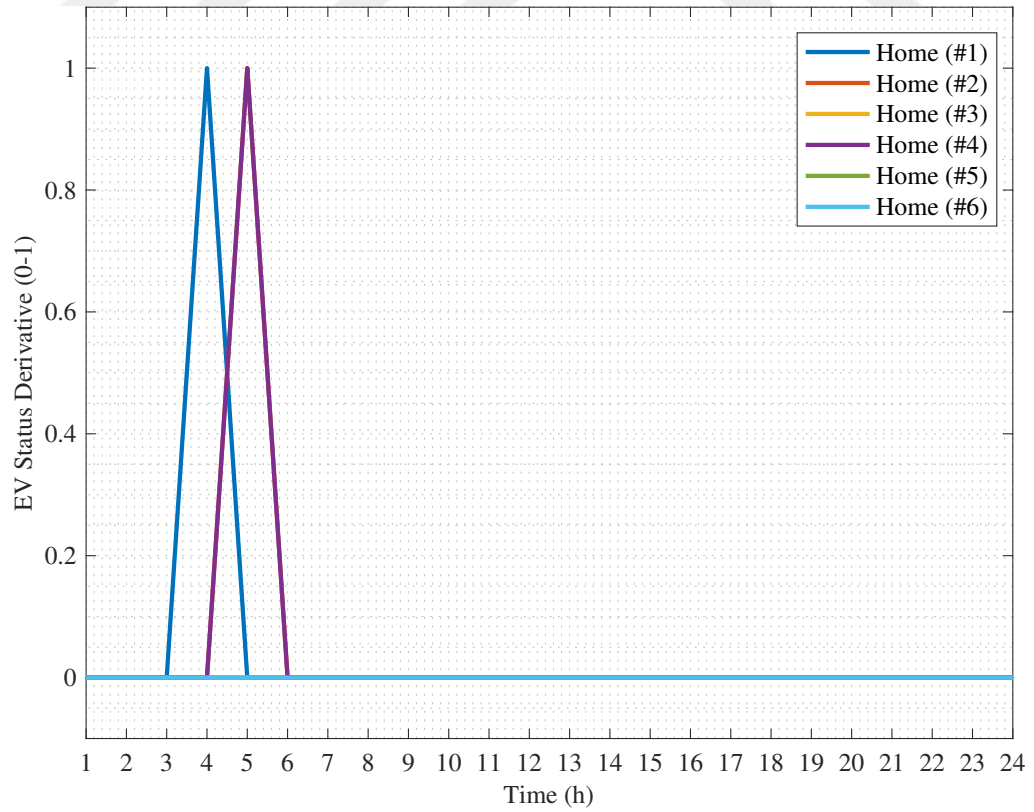


Figure 5.8: Derivative Signal Data of Electric Vehicle Status (Training, Daily)

The energy consumption data of the daily driving profiles of electric vehicles used in the training process, if available, are as follows.

Table 5.11: Daily Energy Consumption Profile of Electric Vehicles (Training)

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Energy Consumption (kWh)	6.44	9.93	0	3.53	0	0

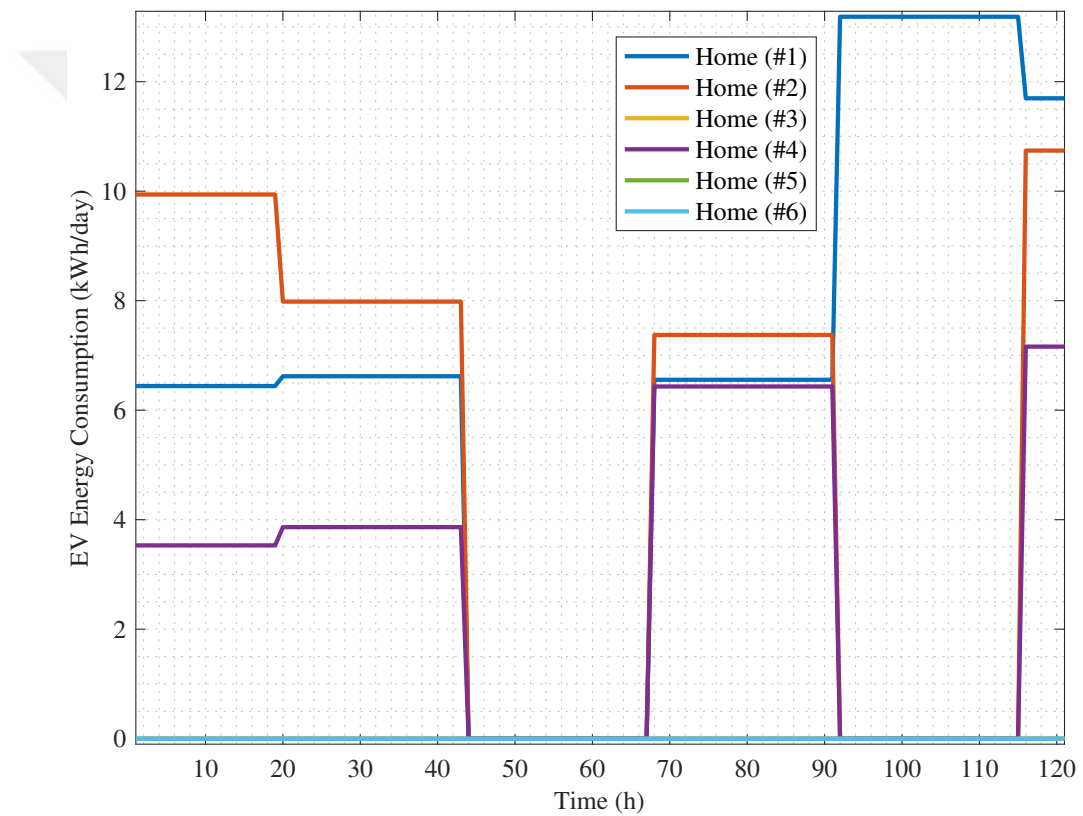


Figure 5.9: Daily Energy Consumption Profile of Electric Vehicles (Training, Complete)

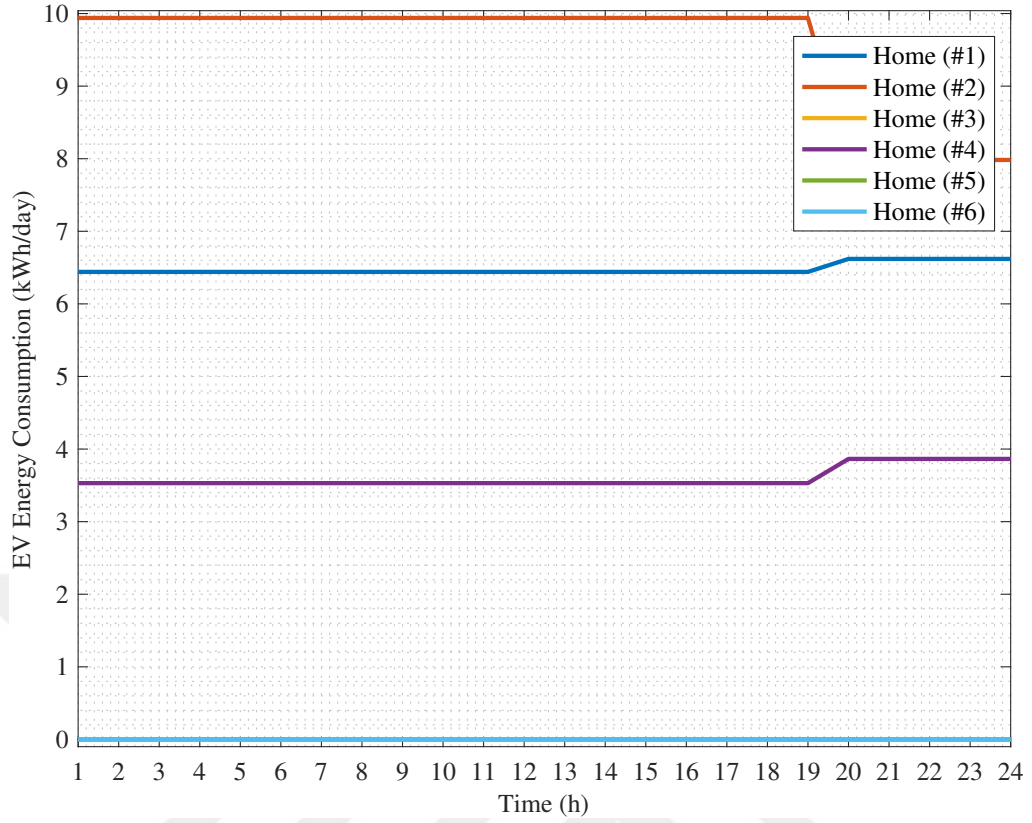


Figure 5.10: Daily Energy Consumption Profile of Electric Vehicles (Training, Daily)

5.2 STATISTICAL ANALYSIS on TESTING DATA

A 10-day (240 data) data set in random sequence was selected from the data pool to be used in the reinforcement learning testing process. A fixed seed value has been assigned so that the tests can be performed with the same data throughout the learning process. The graph of the load input data selected to be used in the testing process is as follows.

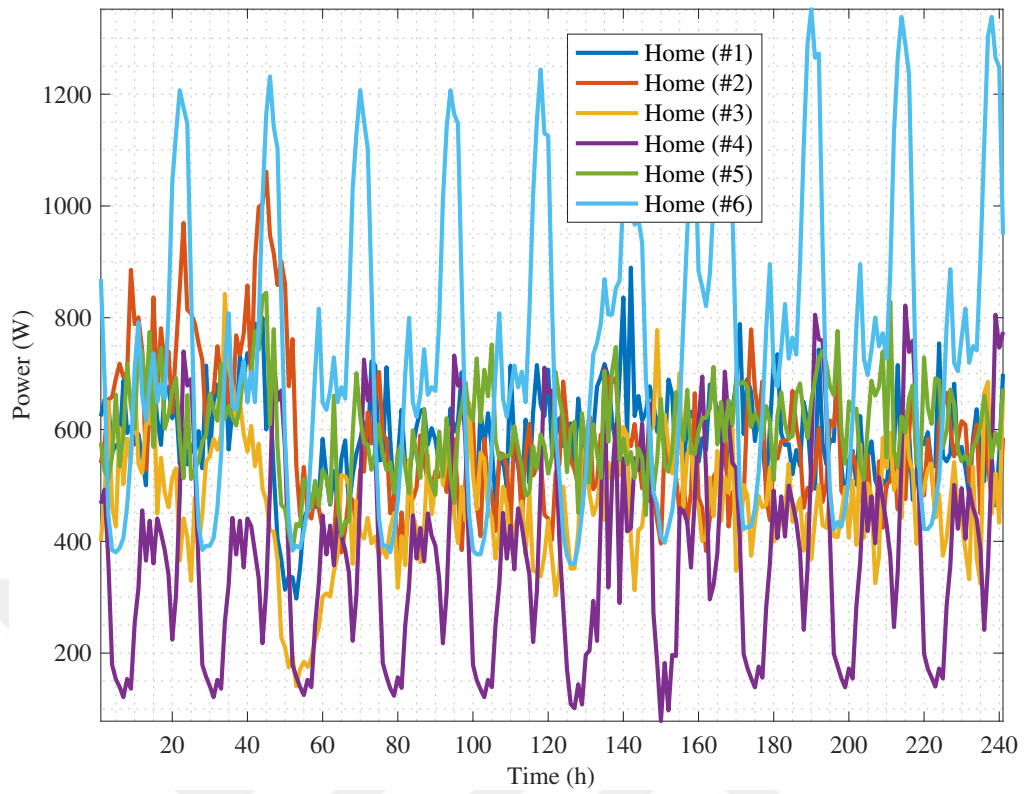


Figure 5.11: Load Power Data (Testing, Complete)

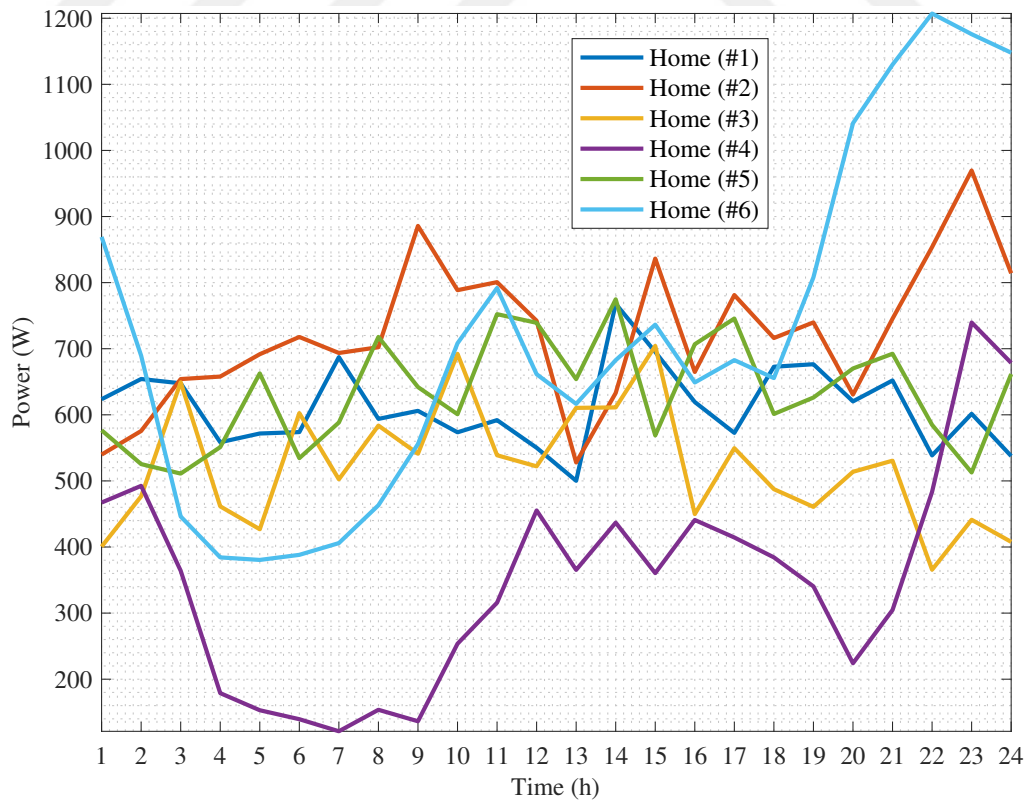


Figure 5.12: Load Power Data (Testing, Daily)

The statistical analysis table of the load data used in the testing process is as follows.

Table 5.12: Descriptive Statistical Analysis of Load Power Data (Testing)

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Mean	799	736	523	429	671	834
Median	783	745	520	414	659	856
Max	1001	891	723	801	798	1448
Min	693	528	336	71	527	352
Range	307	362	387	730	270	1095
Variance	6305	9929	7609	48205	5899	12441
Standard Devia- tion	79	99	87	219	76	352
Percentile (25%)	738	654	472	256	613	492
Percentile (50%)	783	745	520	414	659	856
Percentile (75%)	866	819	562	577	753	1103
Interquartile Ranges	128	165	89	320	139	610
Skewness	0.6807	-0.1132	0.3962	0.1949	0.2627	0.1273

As can be seen from the analysis results, the load data of each house's testing set show different characteristics. The sixth house has the highest mean and value range, and the fourth house has the lowest mean and value range. As with training data, the data is highly diverse over a broad spectrum. In addition, the testing data show similarities with the training data, but exhibit different characteristics.

The graph of the solar energy production data selected to be used in the testing process is as follows.

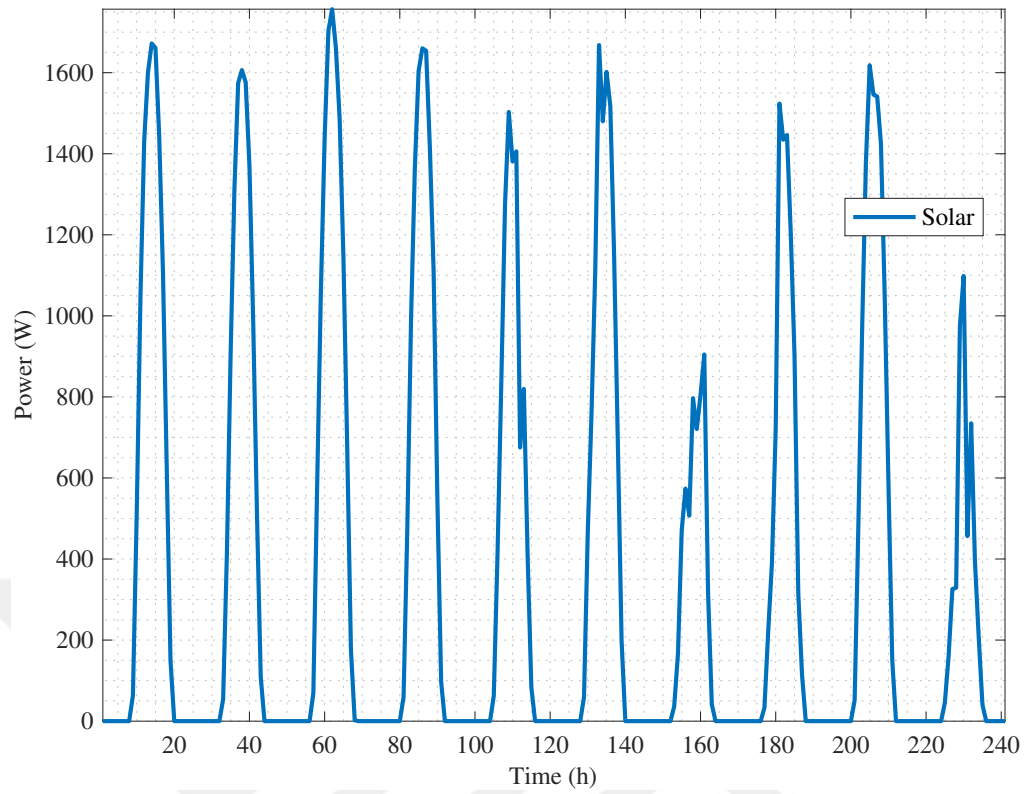


Figure 5.13: Solar Energy Production Data (Testing, Complete)

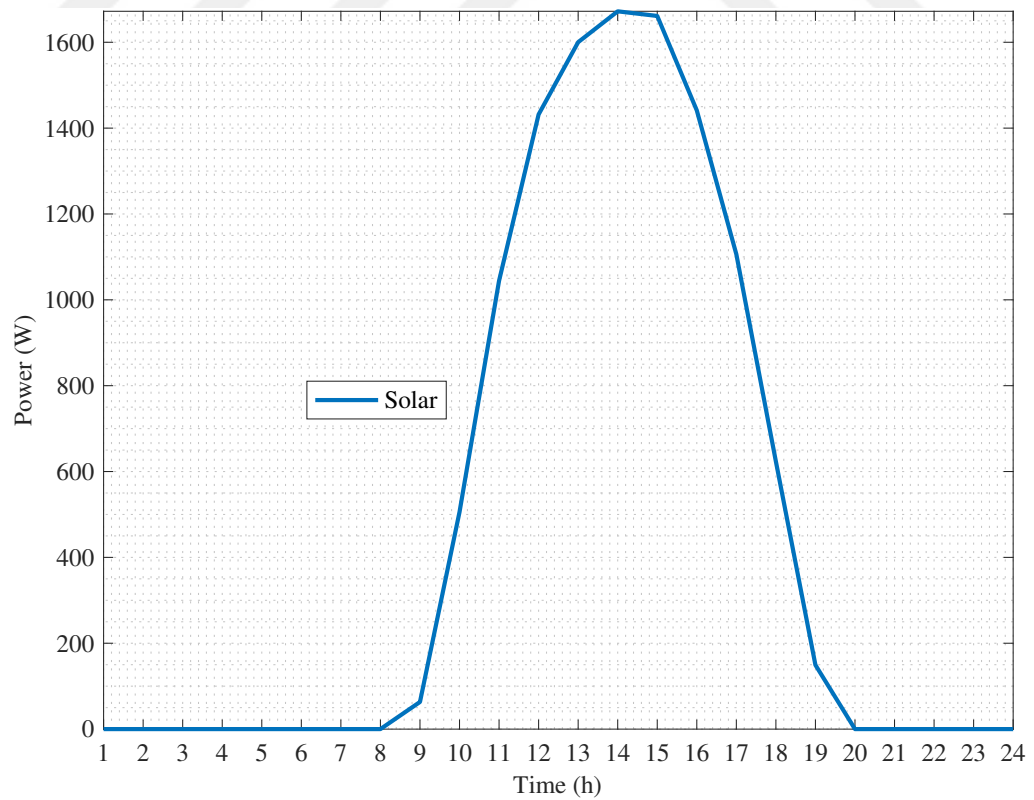


Figure 5.14: Solar Energy Production Data (Testing, Daily)

The statistical analysis table of the solar energy production data used in the testing process is as follows. It is assumed that each house benefits equally from the total amount of solar energy produced.

Table 5.13: Descriptive Statistical Analysis of Solar Energy Production Data (Testing)

	Total Production (W)	Production Per Home (W)
Mean	2113	352
Median	0	0
Max	8633	1439
Min	0	0
Range	8633	1439
Variance	1.07e7	2.97e5
Standard Devia- tion	3271	545
Percentile (25%)	0	0
Percentile (50%)	0	0
Percentile (75%)	4805	800
Interquartile Ranges	4805	800
Skewness	1.0891	1.0891

As can be seen from the analysis results, the amount of energy produced by the solar panels used in the testing phase has a higher average and total value than the training phase. The main reason for this is that the winter months are used in the training phase and the summer months are used in the testing phase. The reason for using the winter months in the training phase is to obtain a stable decision-making output even with lower renewable energy production. As in the training process, values lower than the threshold value were set to 0.

The graph of the "EV Status" data produced by statistical methods to be used in the testing process, in accordance with other available data, is as follows.

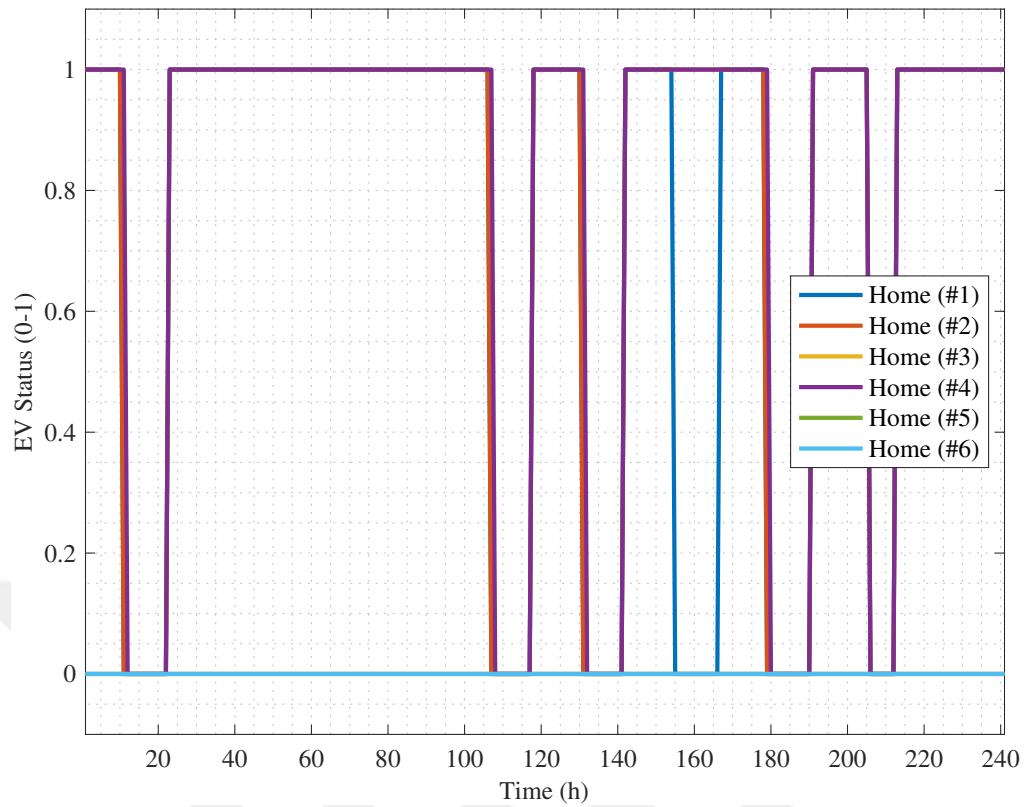


Figure 5.15: Electric Vehicle Status Signal Data (Testing, Complete)

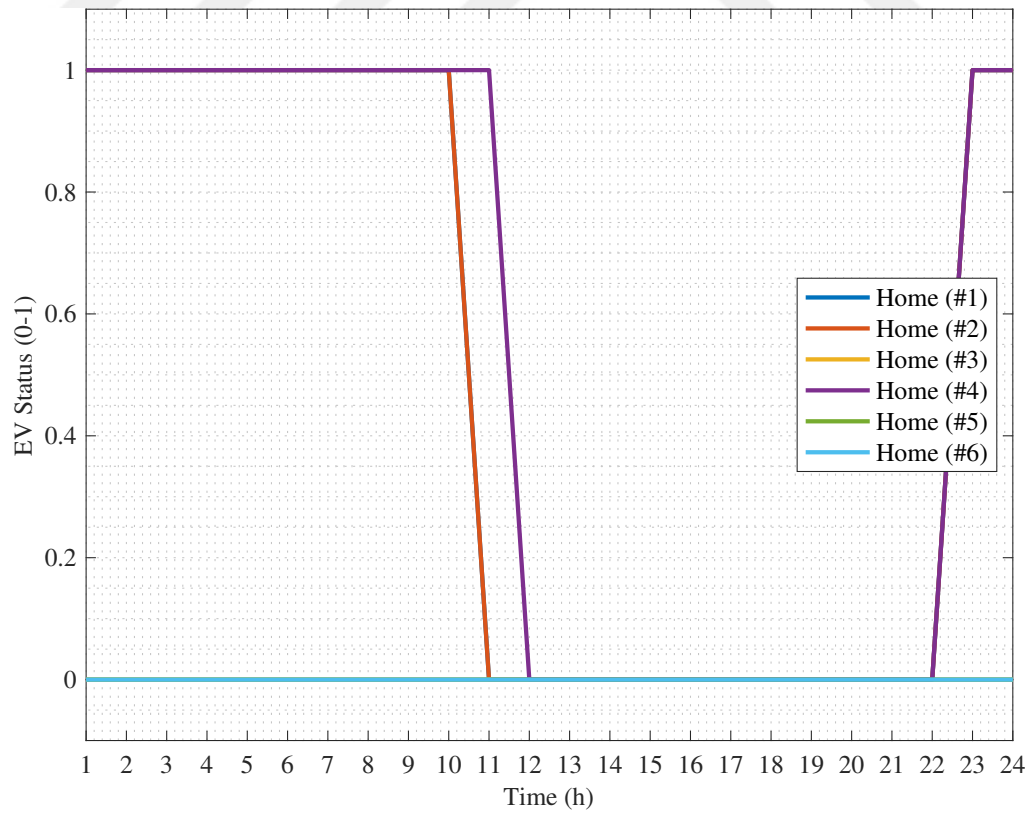


Figure 5.16: Electric Vehicle Status Signal Data (Testing, Daily)

The table below shows the total hours count that electric vehicles are at home or not, if applicable.

Table 5.14: Descriptive Statistical Analysis of Electric Vehicle Status Data (Testing)

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Status “1”	14	14	0	15	0	0
Status “0”	11	11	0	10	0	0

As can be seen from the table above, the electric vehicle status data used in the testing phase is in line with the data used in the training phase. Among the houses with electric vehicles, there are no electric vehicles between 09:00 and 19:00 in house #1, between 09:00 and 19:00 in house #2, and between 10:00 and 19:00 in house #4.

Table 5.15: Departure and Arrival Hours of Electric Vehicles (Testing)

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Departure Hour	09:00	09:00	00:00	10:00	00:00	00:00
Arrival Hour	19:00	19:00	00:00	19:00	00:00	00:00

As can be seen from the table above, the departure and arrival times of the electric vehicles used in the testing process are similar to the data used in the training process.

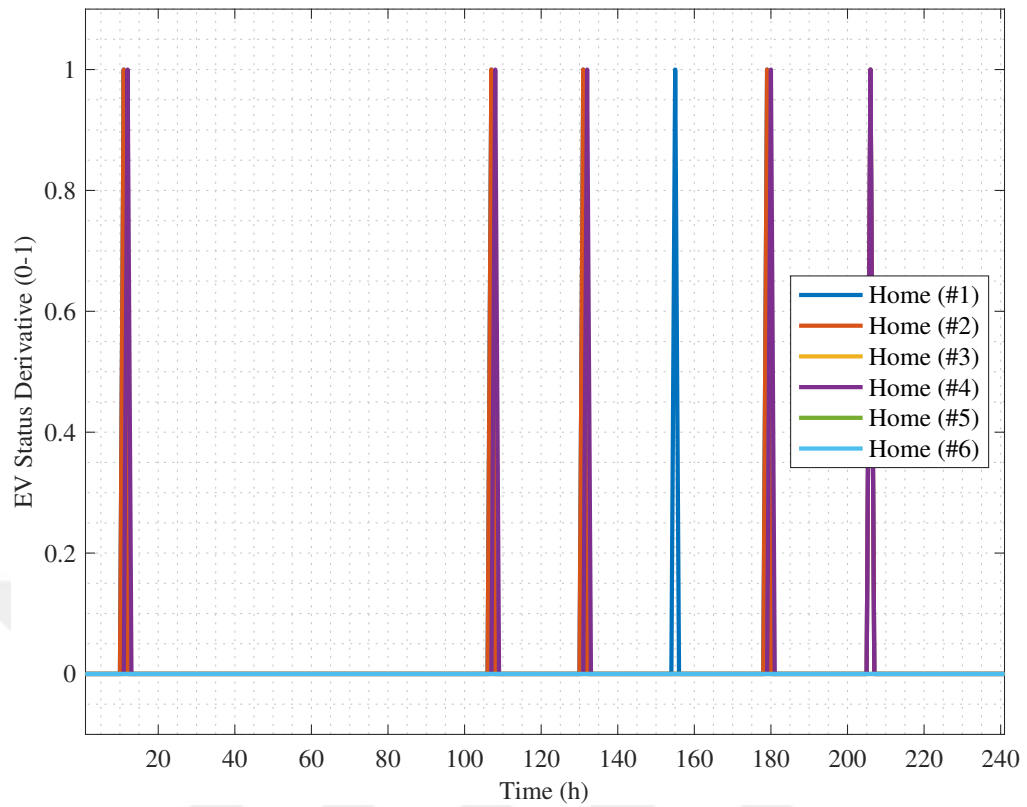


Figure 5.17: Derivative Signal Data of Electric Vehicle Status (Testing, Complete)

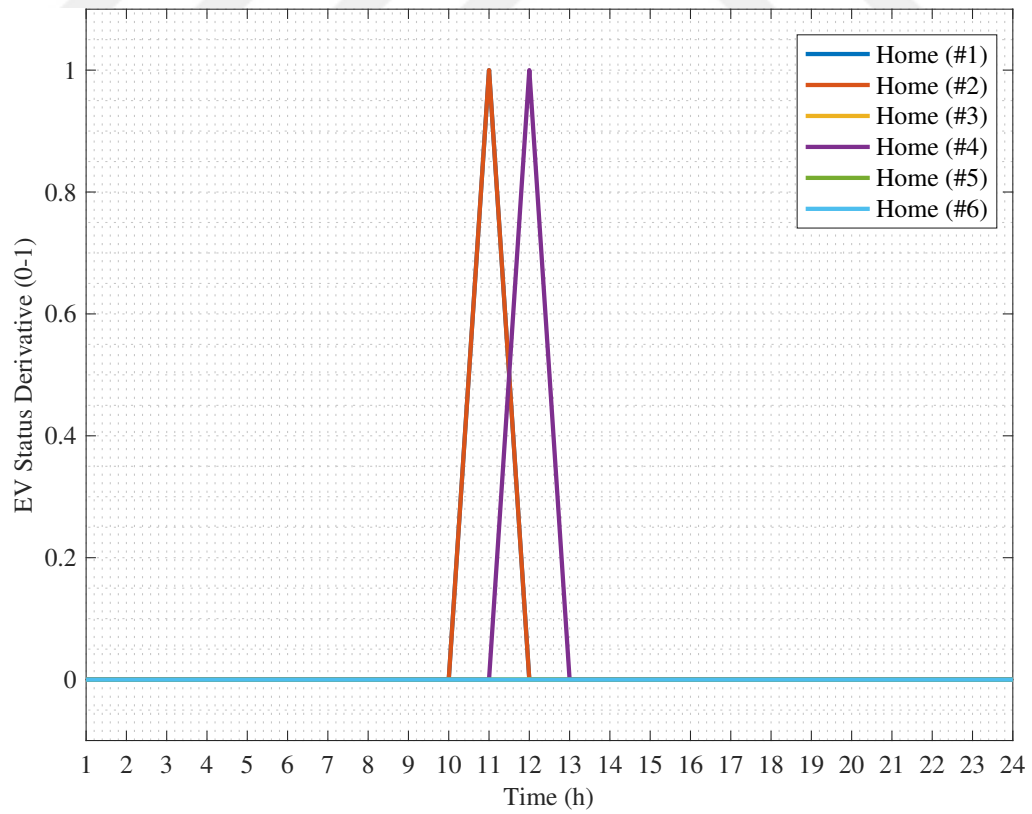


Figure 5.18: Derivative Signal Data of Electric Vehicle Status (Testing, Daily)

During the testing process, the energy consumption data of the daily driving profiles of electric vehicles, if available, are as follows.

Table 5.16: Daily Energy Consumption Profile of Electric Vehicles (Testing)

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Energy Consumption (kWh)	7.48	9.40	0	4.33	0	0

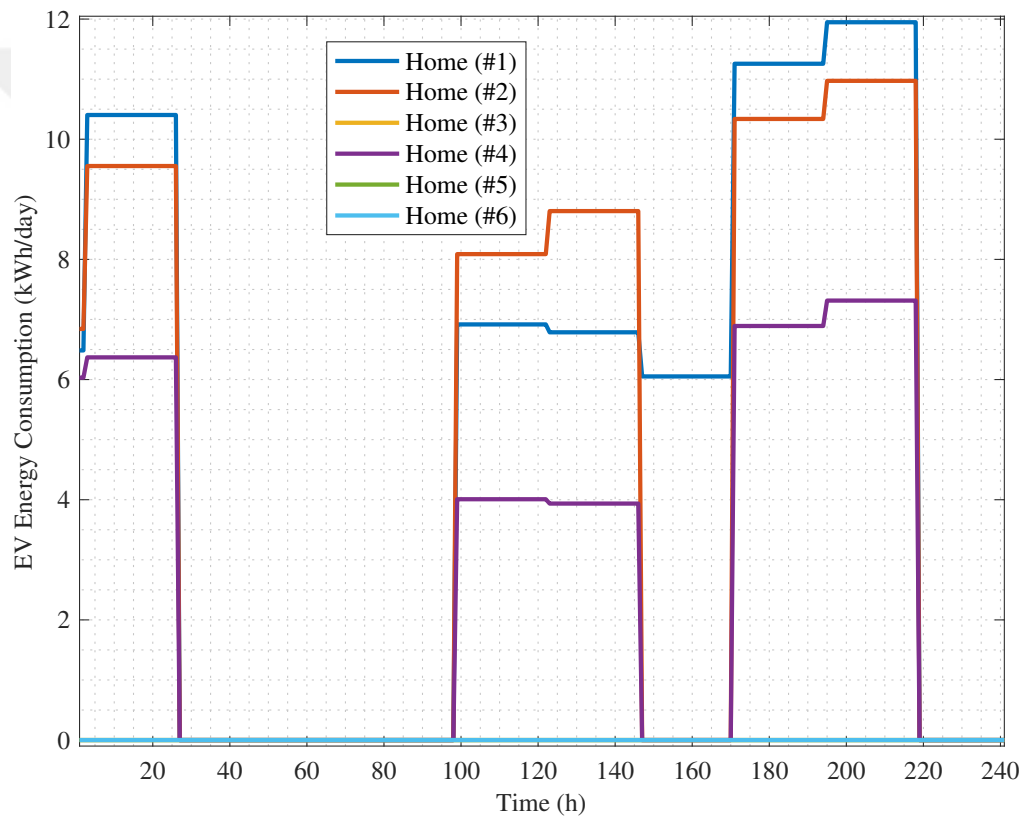


Figure 5.19: Daily Energy Consumption Profile of Electric Vehicles (Testing, Complete)

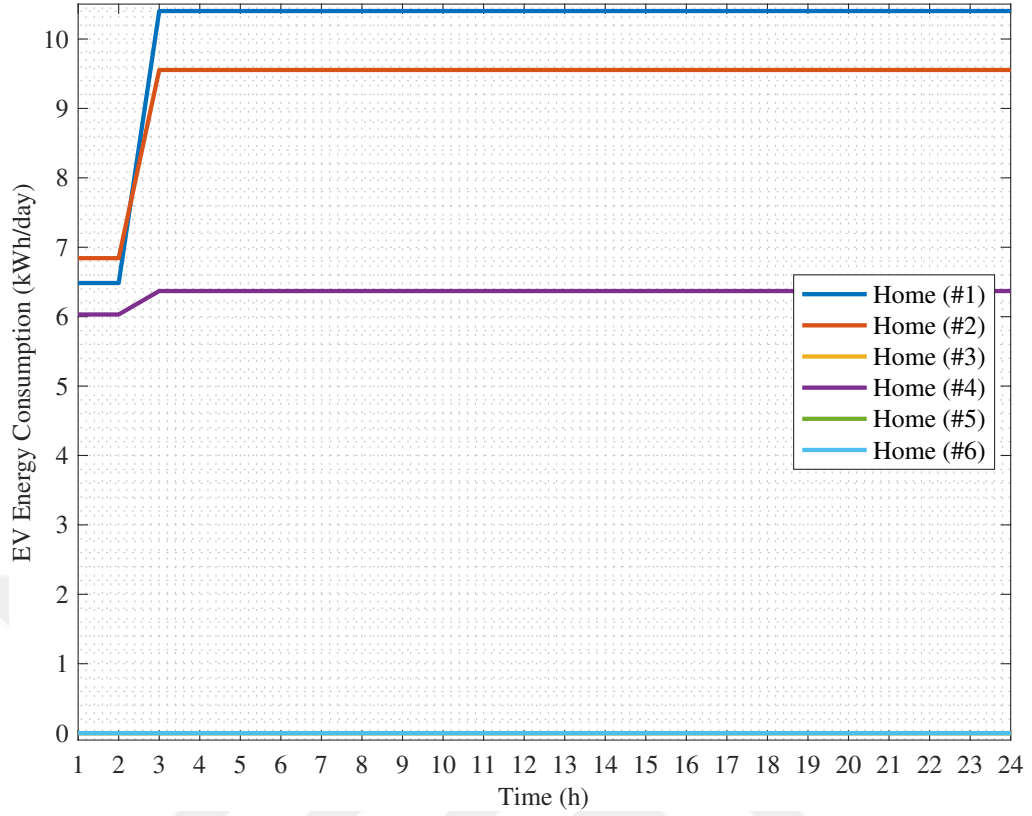


Figure 5.20: Daily Energy Consumption Profile of Electric Vehicles (Testing, Daily)

5.3 TRAINING PROCESS & RESULTS

5.3.1 Training Process

The model was trained with the built-in "train" function in MATLAB, with the current environment and action space, in a way that would have 5000 episodes and each episode would have 120 steps. Learning and training outputs of each agent were saved as a "mat" formatted file. The learning outcomes include the network weights used in deep learning in the action-state map of the relevant agent. On the other hand, the training outcomes include the episode reward value, the average reward value, and the Q0 value of the respective agent's training performance. The episode reward value represents the agent's current reward value in the relevant episode. The average reward value represents the average value of the agent's reward value in past episodes. The Q0 value represents the estimated value of the long-term rewards with the discount factor applied. The reward values of each agent are as follows.

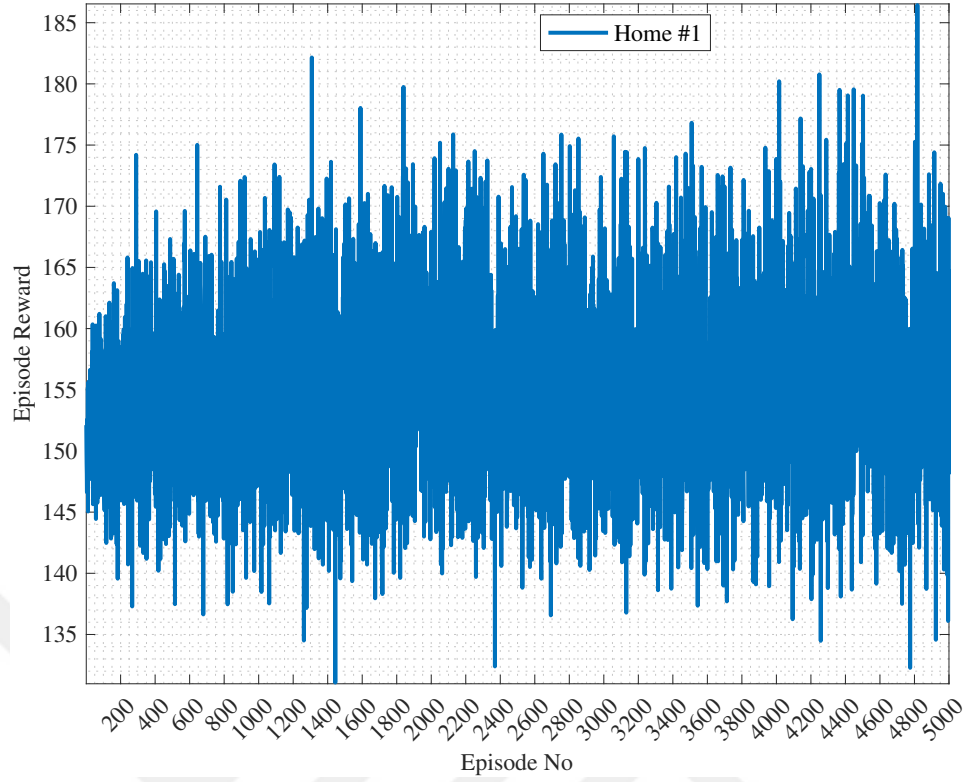


Figure 5.21: Episode Reward Values of Agent #1 During the Training

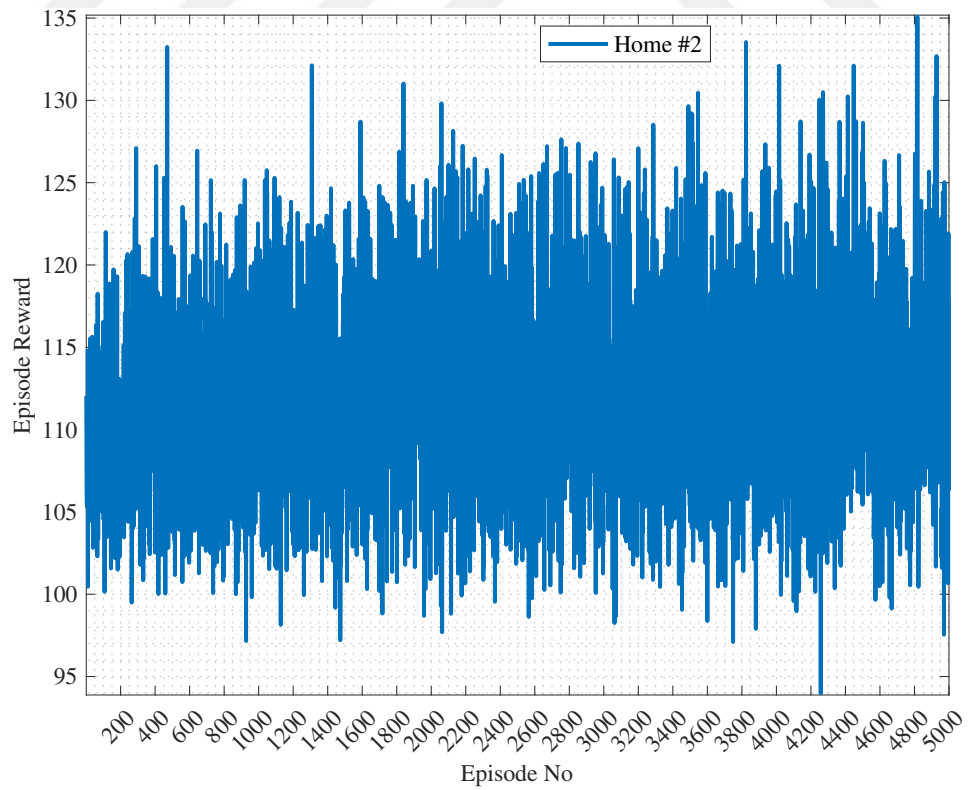


Figure 5.22: Episode Reward Values of Agent #2 During the Training

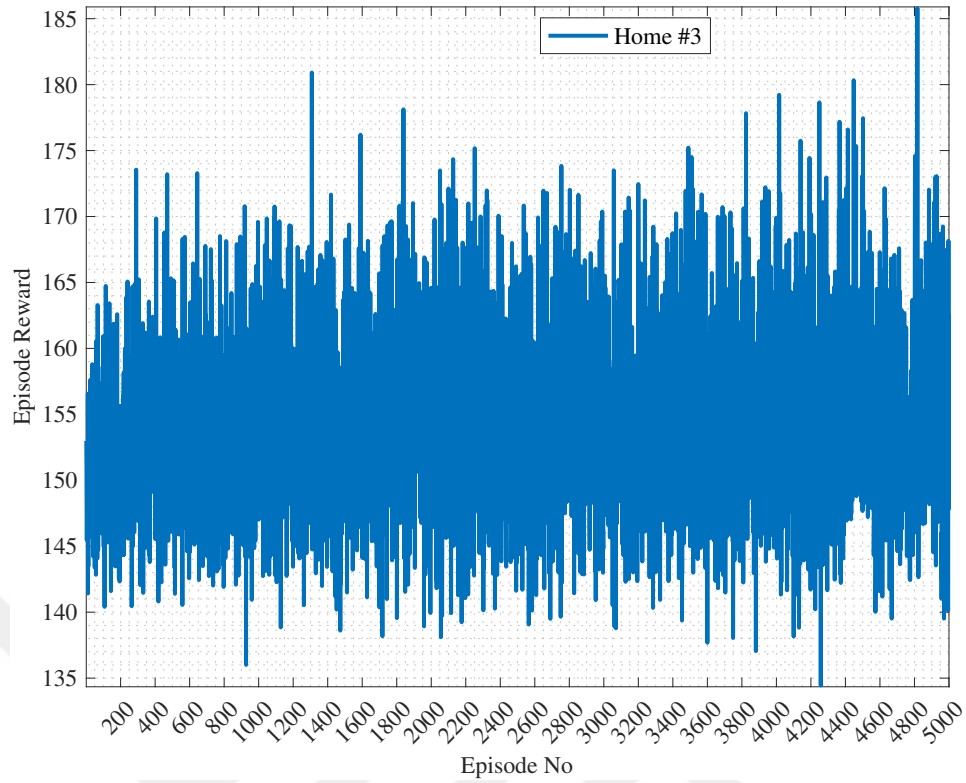


Figure 5.23: Episode Reward Values of Agent #3 During the Training

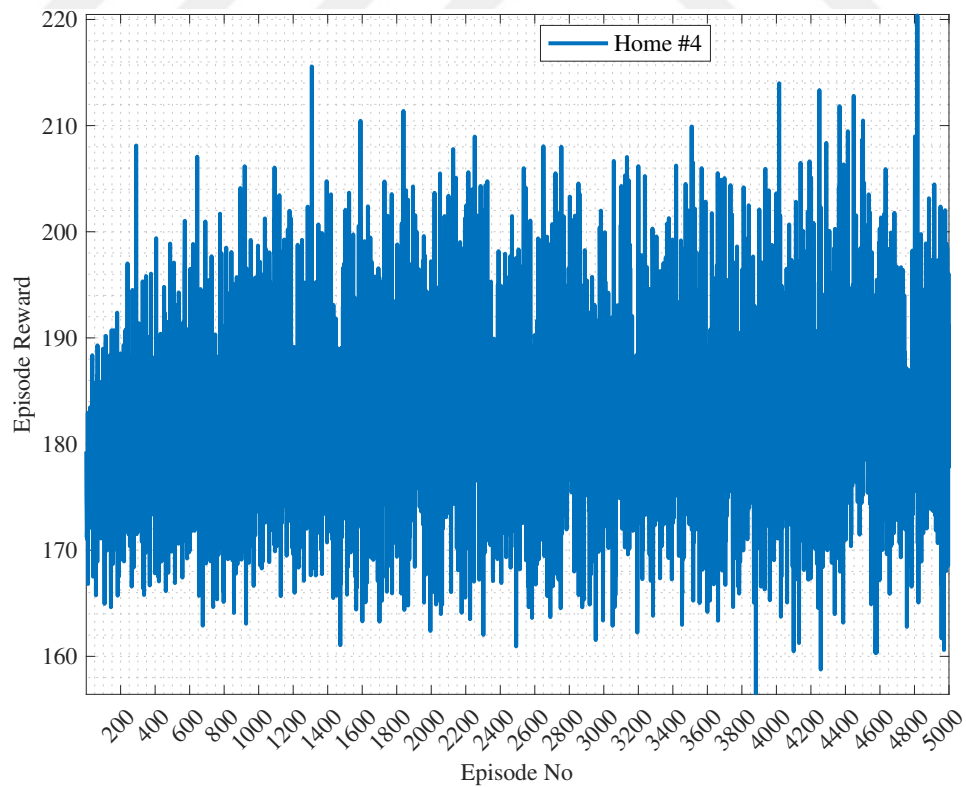


Figure 5.24: Episode Reward Values of Agent #4 During the Training

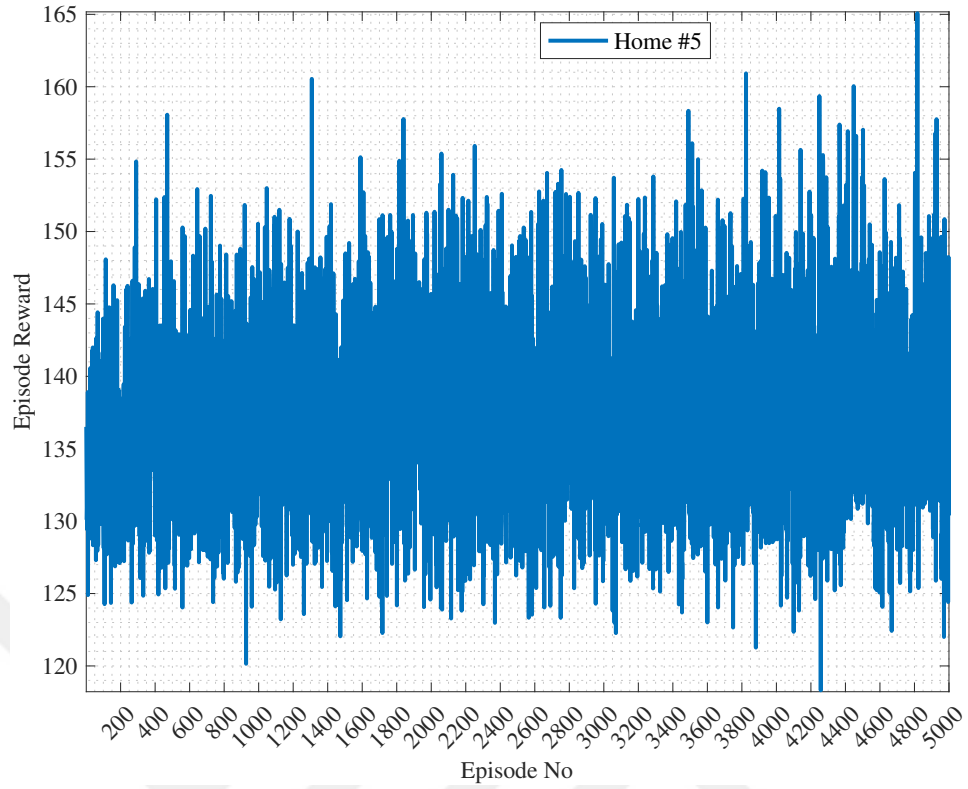


Figure 5.25: Episode Reward Values of Agent #5 During the Training

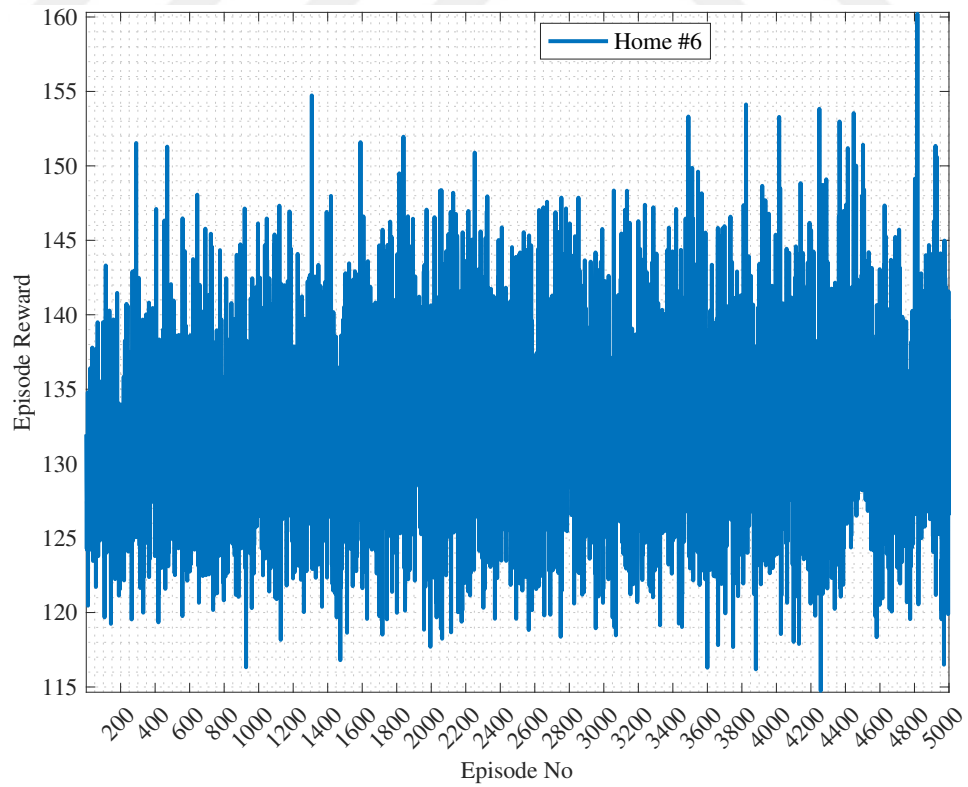


Figure 5.26: Episode Reward Values of Agent #6 During the Training

The average reward values of each agent, that were calculated according to the window value given, are as follows.

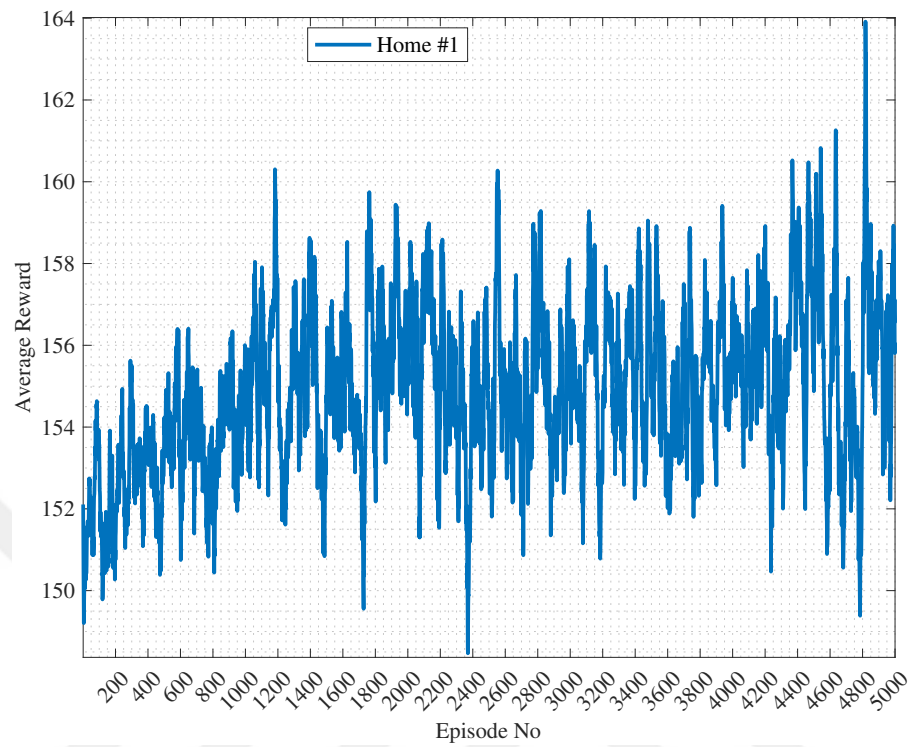


Figure 5.27: Average Reward Values of Agent #2 During the Training

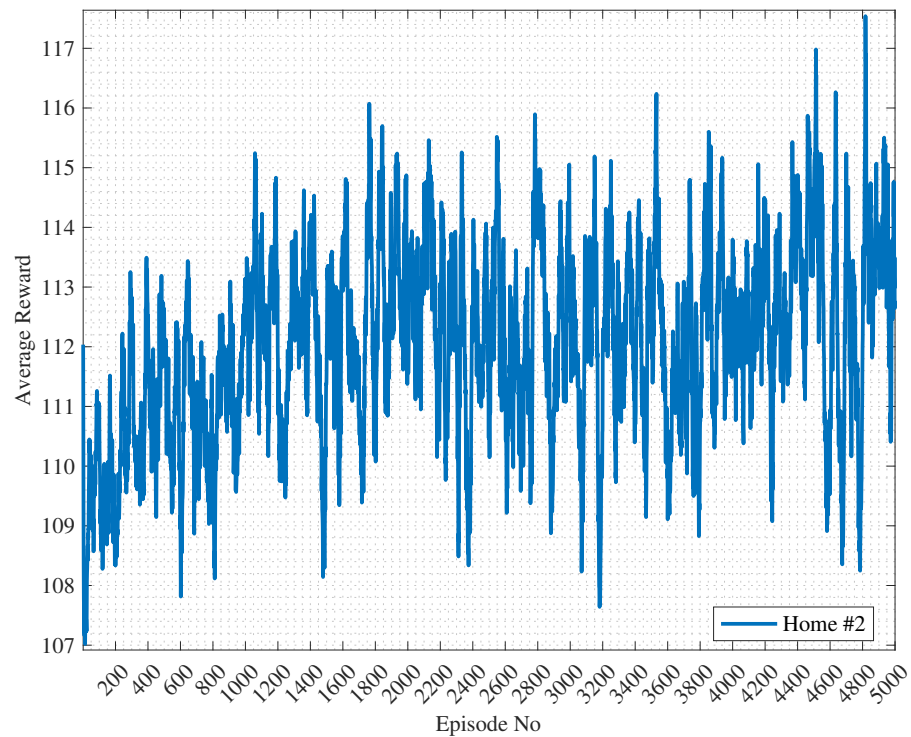


Figure 5.28: Average Reward Values of Agent #2 During the Training

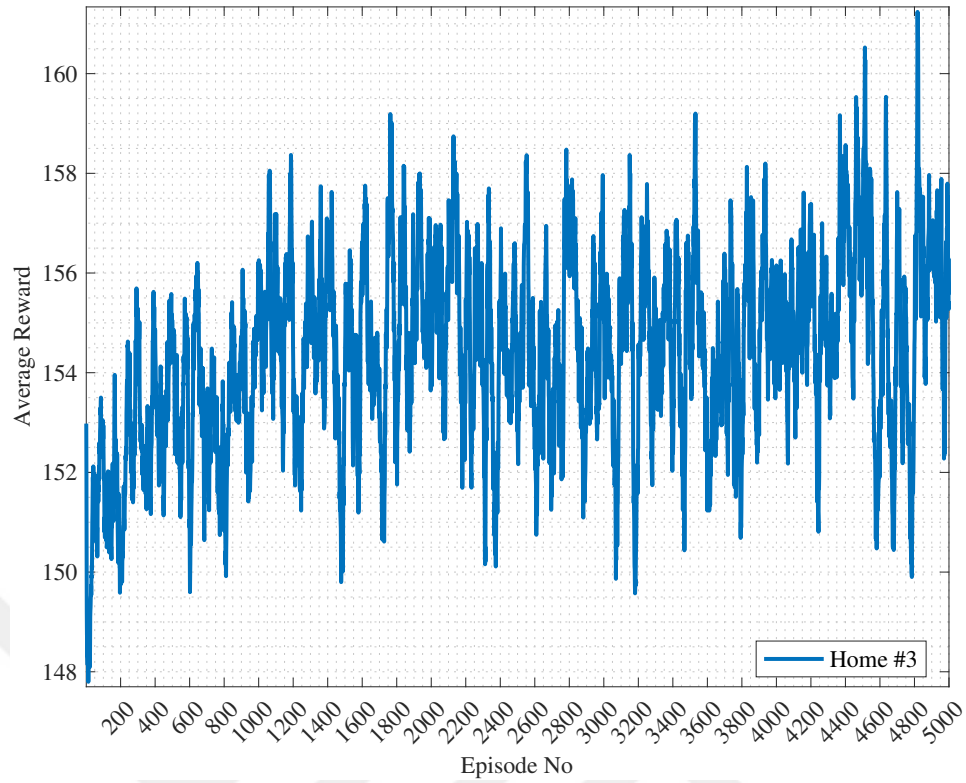


Figure 5.29: Average Reward Values of Agent #3 During the Training

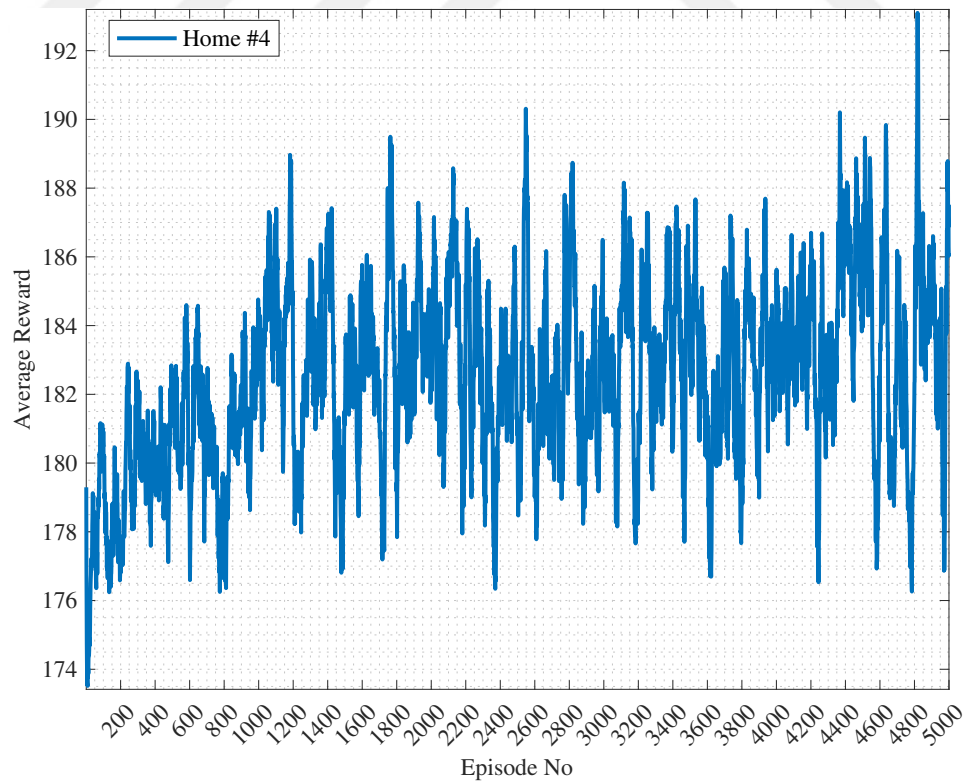


Figure 5.30: Average Reward Values of Agent #4 During the Training

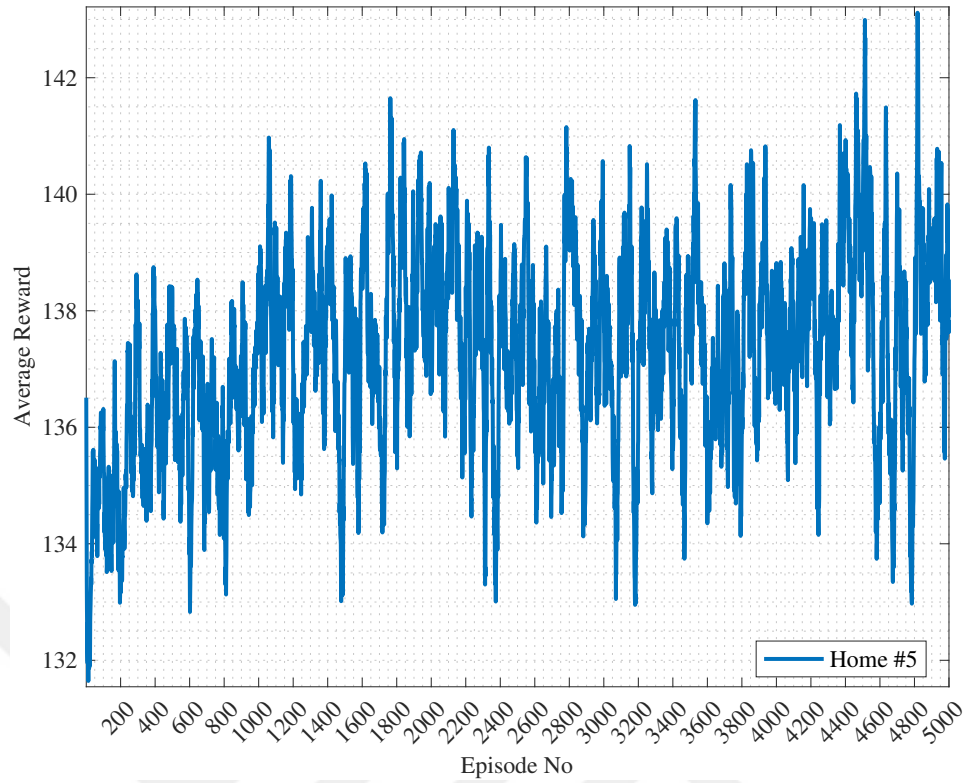


Figure 5.31: Average Reward Values of Agent #5 During the Training

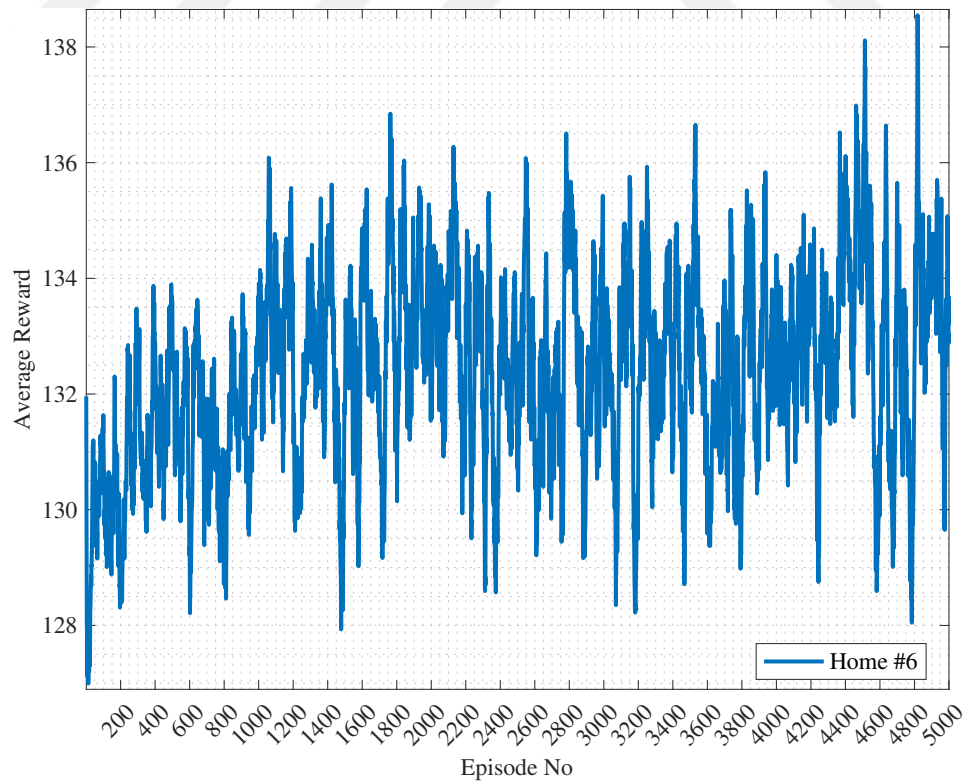


Figure 5.32: Average Reward Values of Agent #6 During the Training

The Q0 value graphs showing the efficiency and performance of each agent's neural network structure used in the training are as follows.

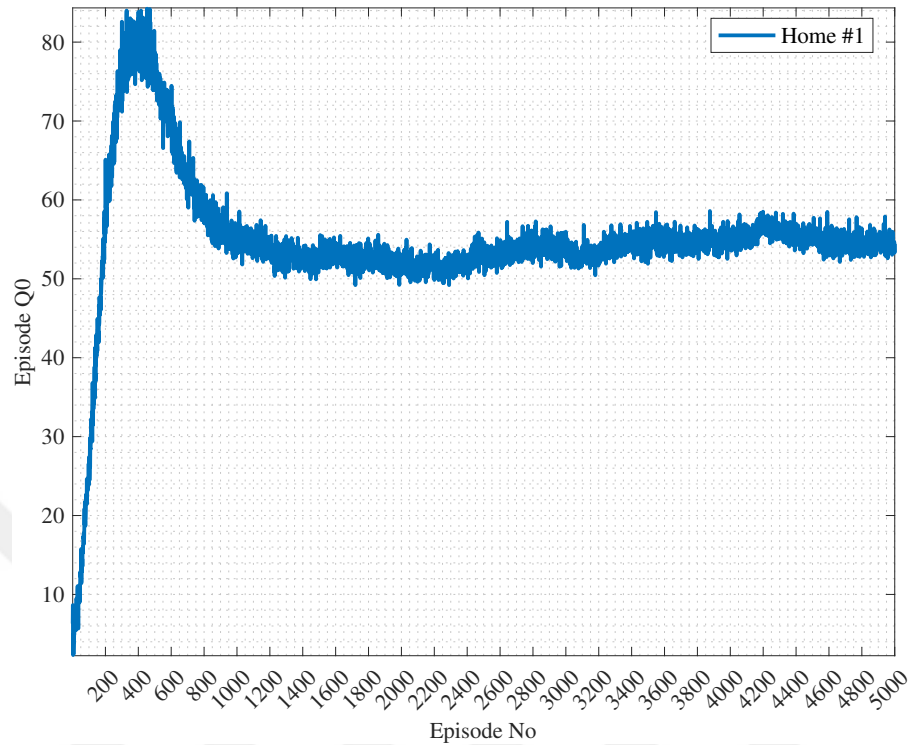


Figure 5.33: Q0 Values of Agent #1 During the Training

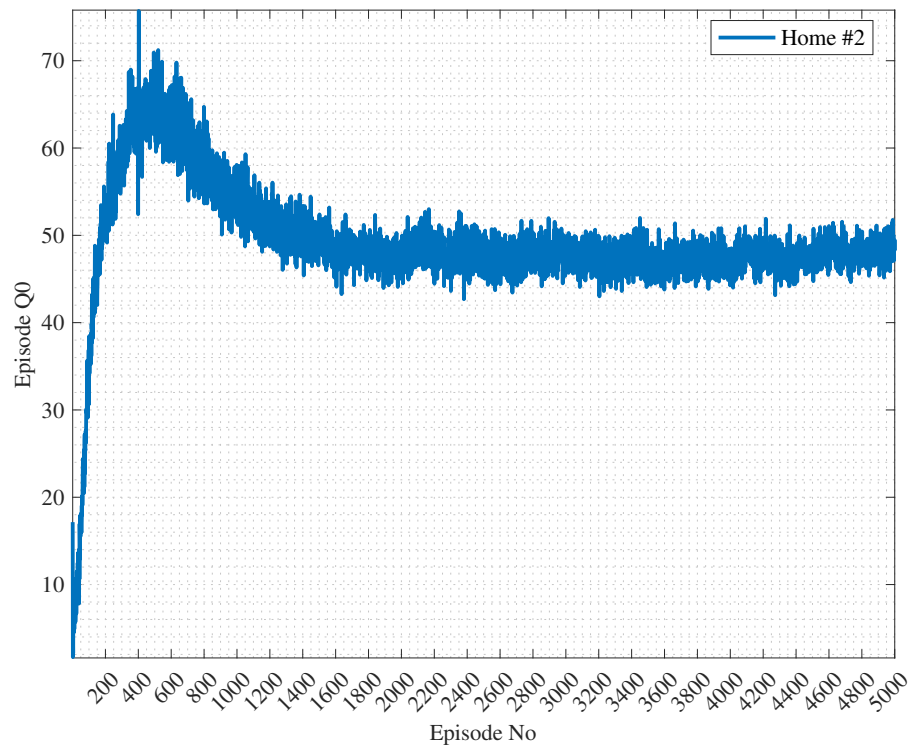


Figure 5.34: Q0 Values of Agent #2 During the Training

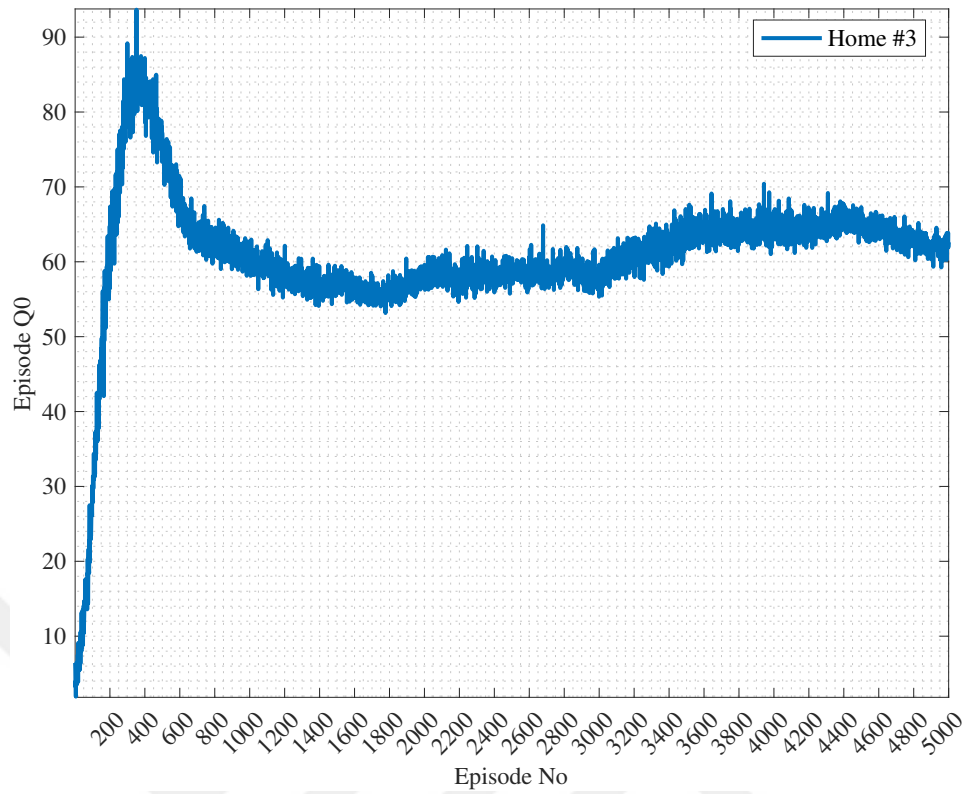


Figure 5.35: Q0 Values of Agent #3 During the Training

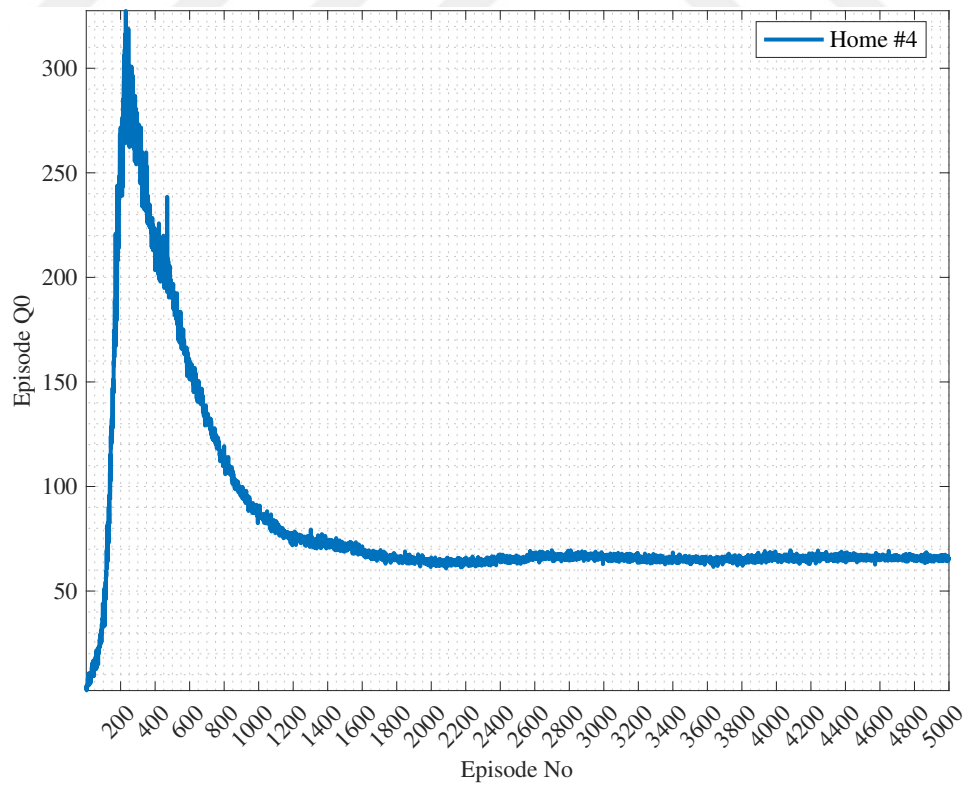


Figure 5.36: Q0 Values of Agent #4 During the Training

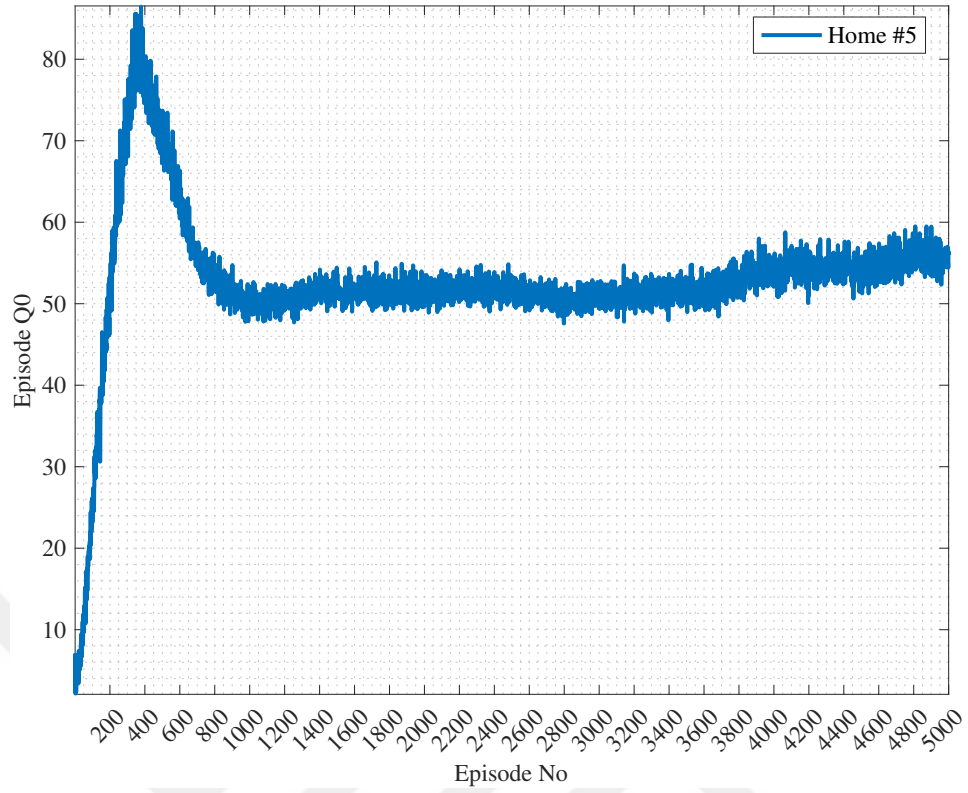


Figure 5.37: Q0 Values of Agent #5 During the Training

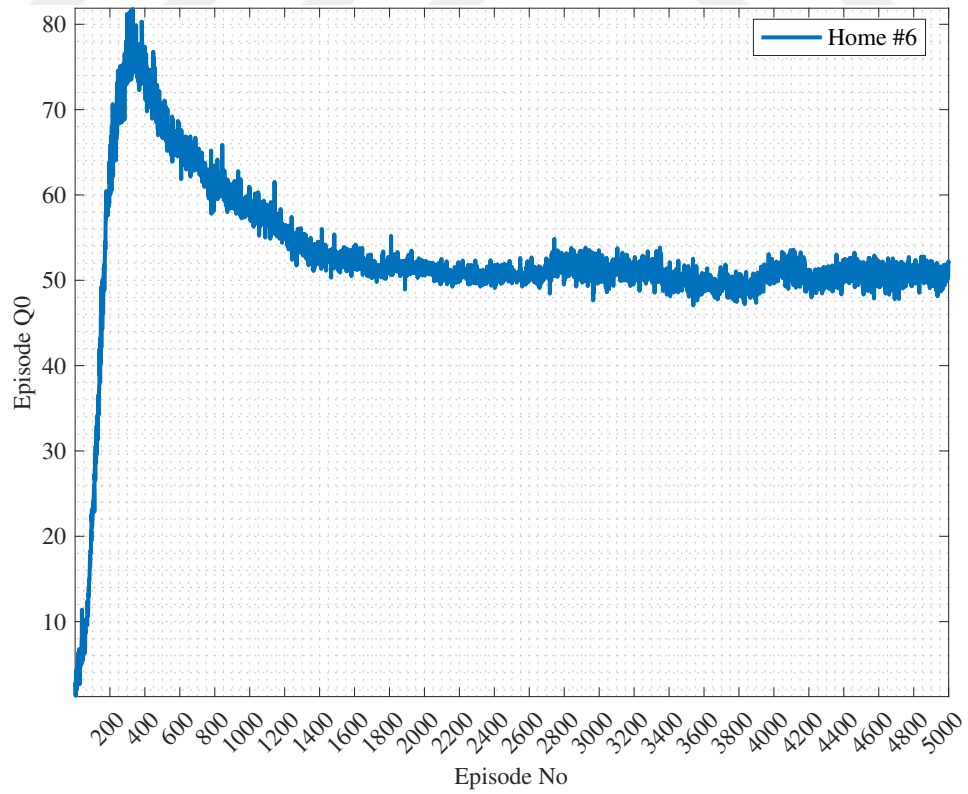


Figure 5.38: Q0 Values of Agent #6 During the Training

Comparative graphs of episode reward, average reward and Q0 values are as follows.

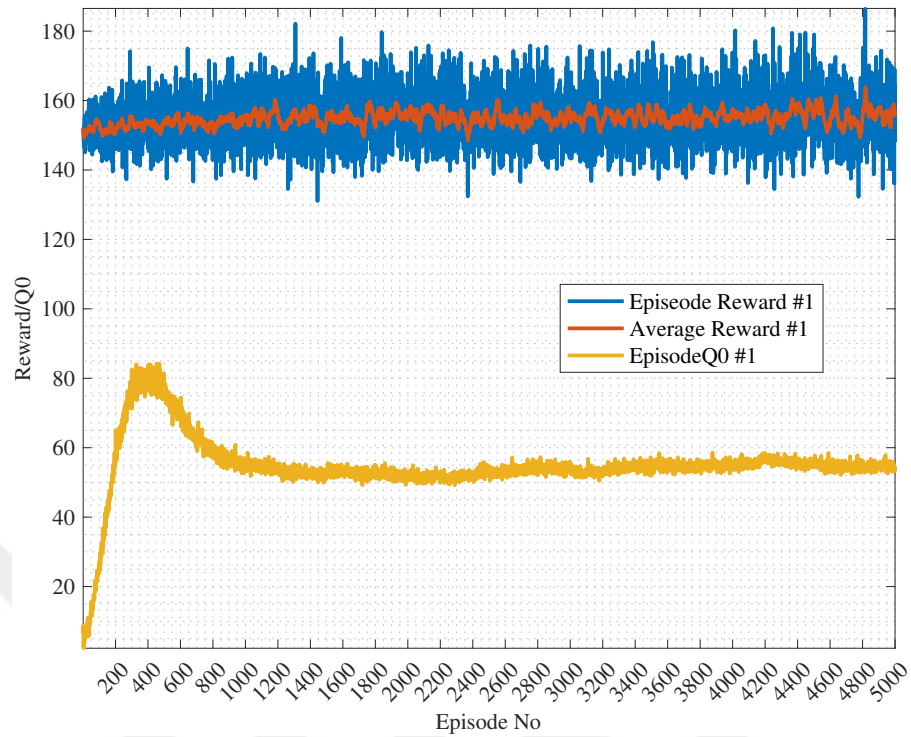


Figure 5.39: Comparison of Episode Reward, Average Reward, and Q0 of Agent #1

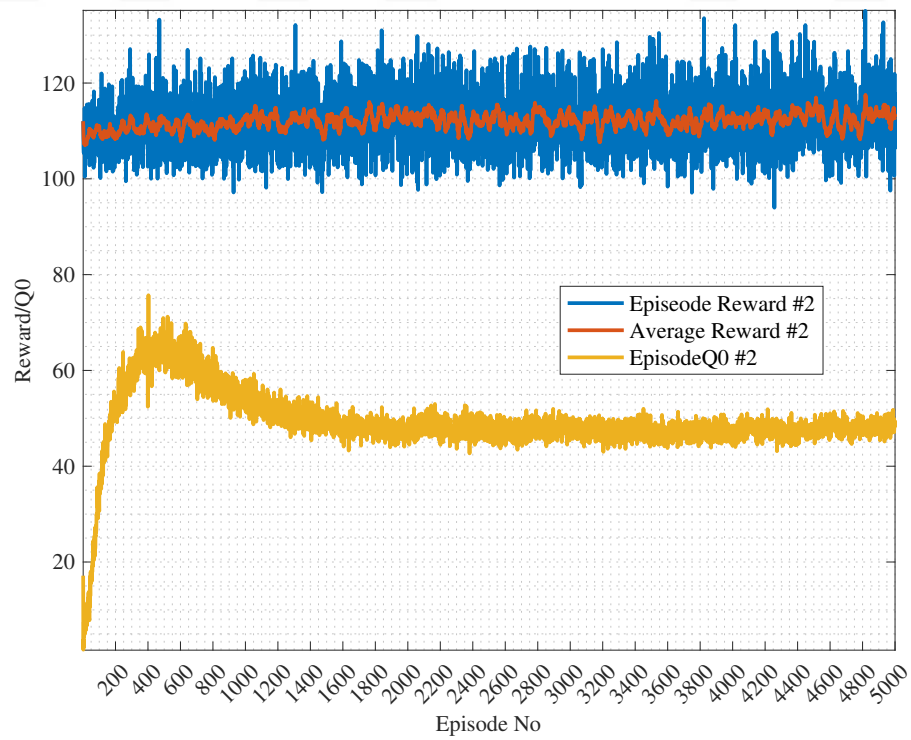


Figure 5.40: Comparison of Episode Reward, Average Reward, and Q0 of Agent #2

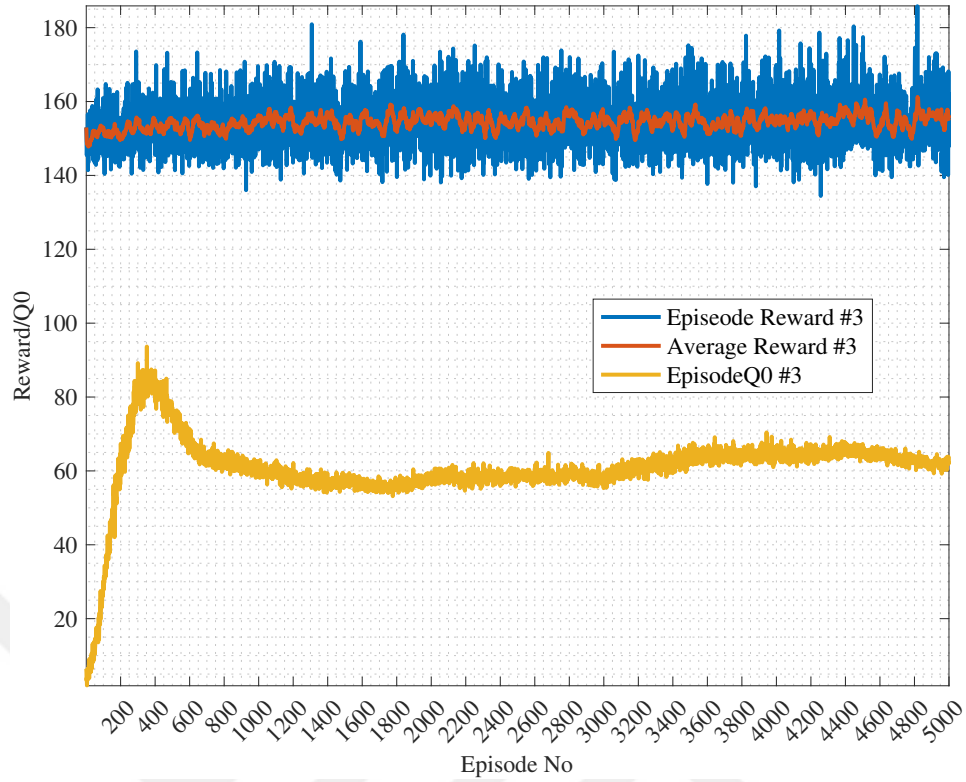


Figure 5.41: Comparison of Episode Reward, Average Reward, and Q0 of Agent #3

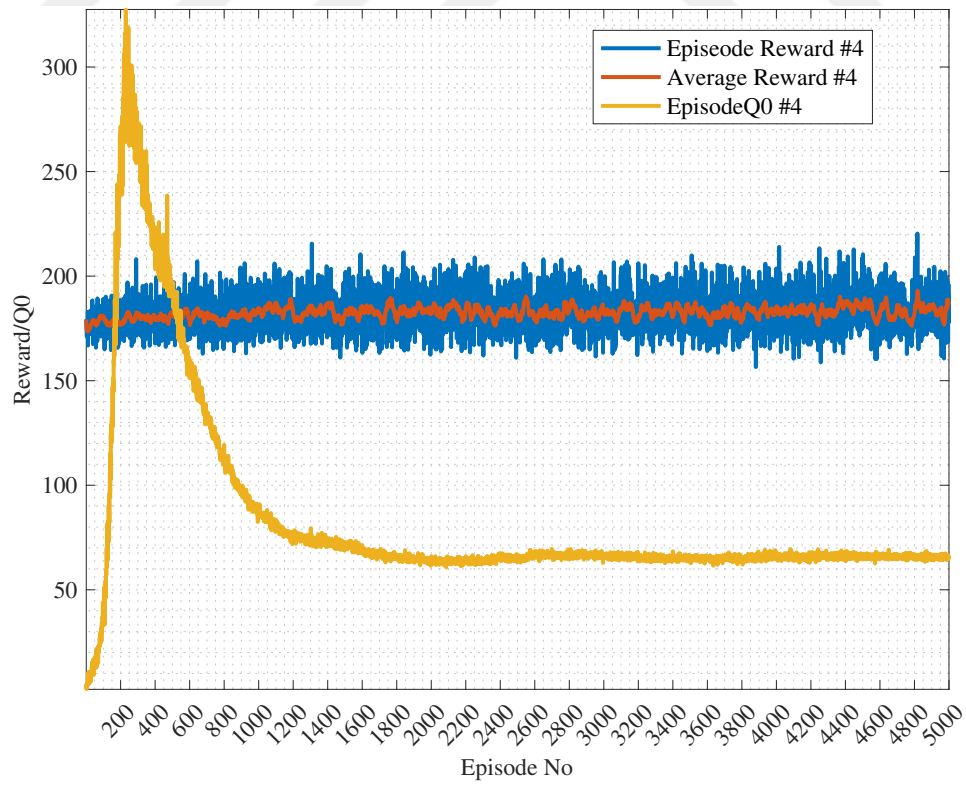


Figure 5.42: Comparison of Episode Reward, Average Reward, and Q0 of Agent #4

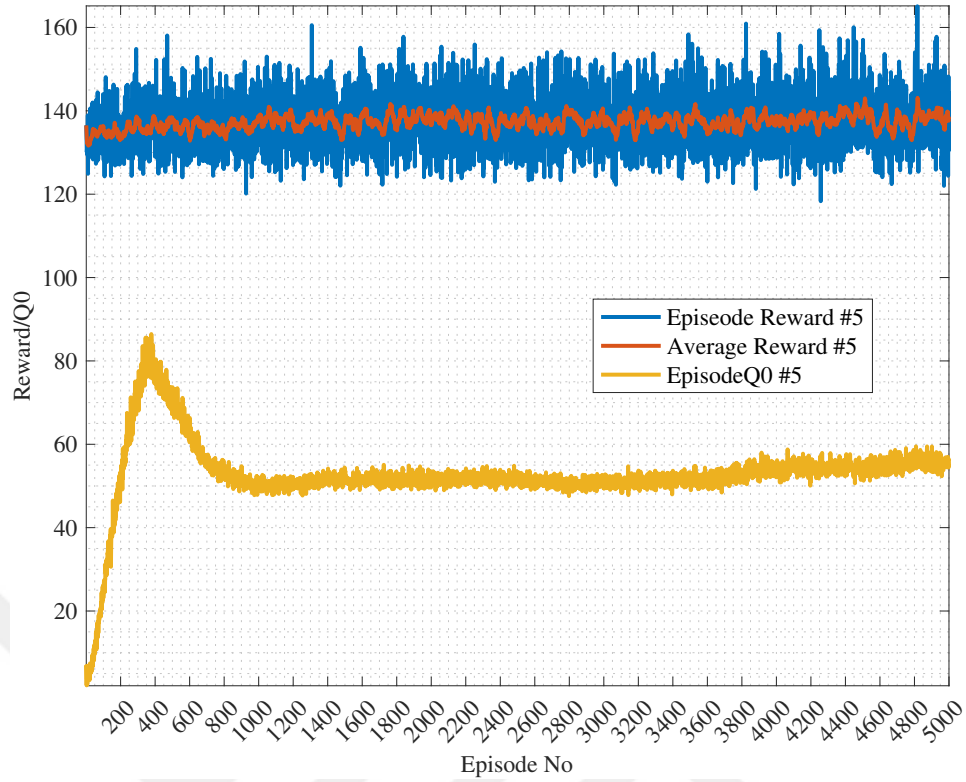


Figure 5.43: Comparison of Episode Reward, Average Reward, and Q0 of Agent #5

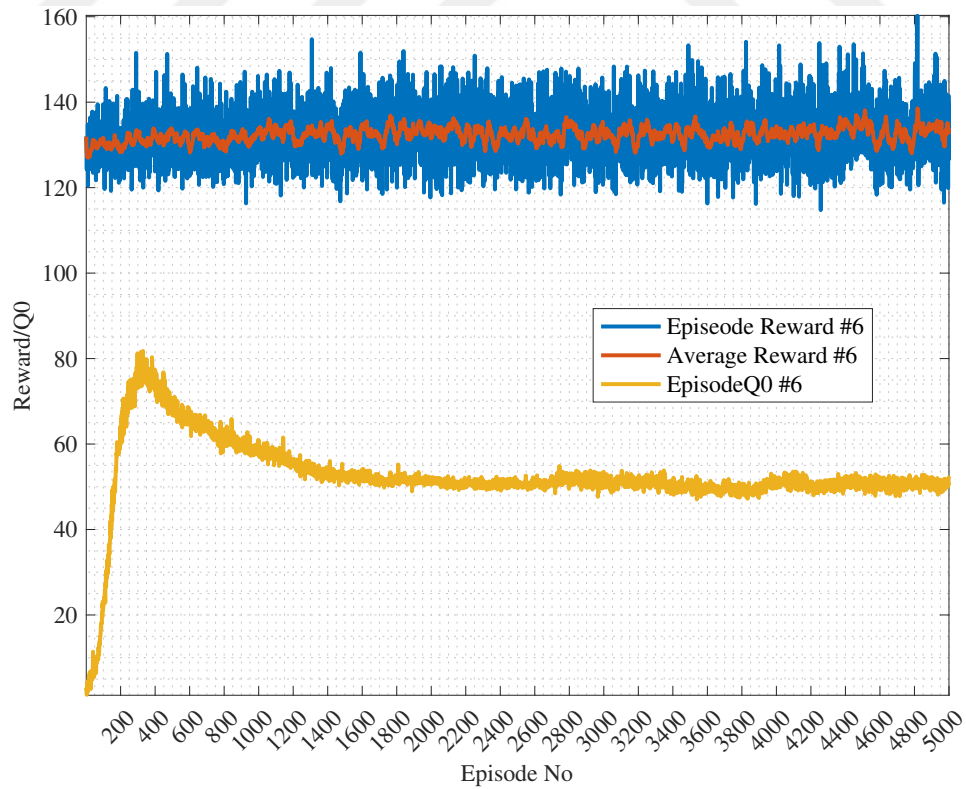


Figure 5.44: Comparison of Episode Reward, Average Reward, and Q0 of Agent #6

As can be seen from the figures, the increasing average reward value and the Q0 value that converges after a certain episode count shows that the training has reached the saturation point at the end of approximately 2200 episodes. The reason why the training was carried out in more than 2200 episodes is that it provides a more stable training by eliminating training uncertainties.

5.3.2 Test Results

Within the scope of the project, the results were tested in 4 different cases. The first of these cases is the environment where the training is fully used and the Vehicle-to-Grid and Peer-to-peer functionality is present. The second case is the environment with Vehicle-to-Grid functionality but no peer-to-peer functionality. The third case is the environment where there is no Vehicle-to-grid functionality but peer-to-peer functionality. The last case is a conventional environment with neither vehicle-to-grid nor peer-to-peer functionality.

Table 5.17: Cases used in Testing Process

	Case #1 (V2G-P2P)	Case #2 (V2G-P2P)	Case #3 (V2G-P2P)	Case #4 (V2G-P2P)
Home #1	Yes-Yes	Yes-No	No-Yes	No-No
Home #2	Yes-Yes	Yes-No	No-Yes	No-No
Home #3	No-Yes	No-No	No-Yes	No-No
Home #4	Yes-Yes	Yes-No	No-Yes	No-No
Home #5	No-Yes	No-No	No-Yes	No-No
Home #6	No-Yes	No-No	No-Yes	No-No

Load, solar, ev status, ev rate, ev idrive input values used in all cases consist of the same values. The value graph, which is the difference between the load value needed by the house and the amount of solar energy produced, which forms the basis of the decision-making mechanism, is as follows.

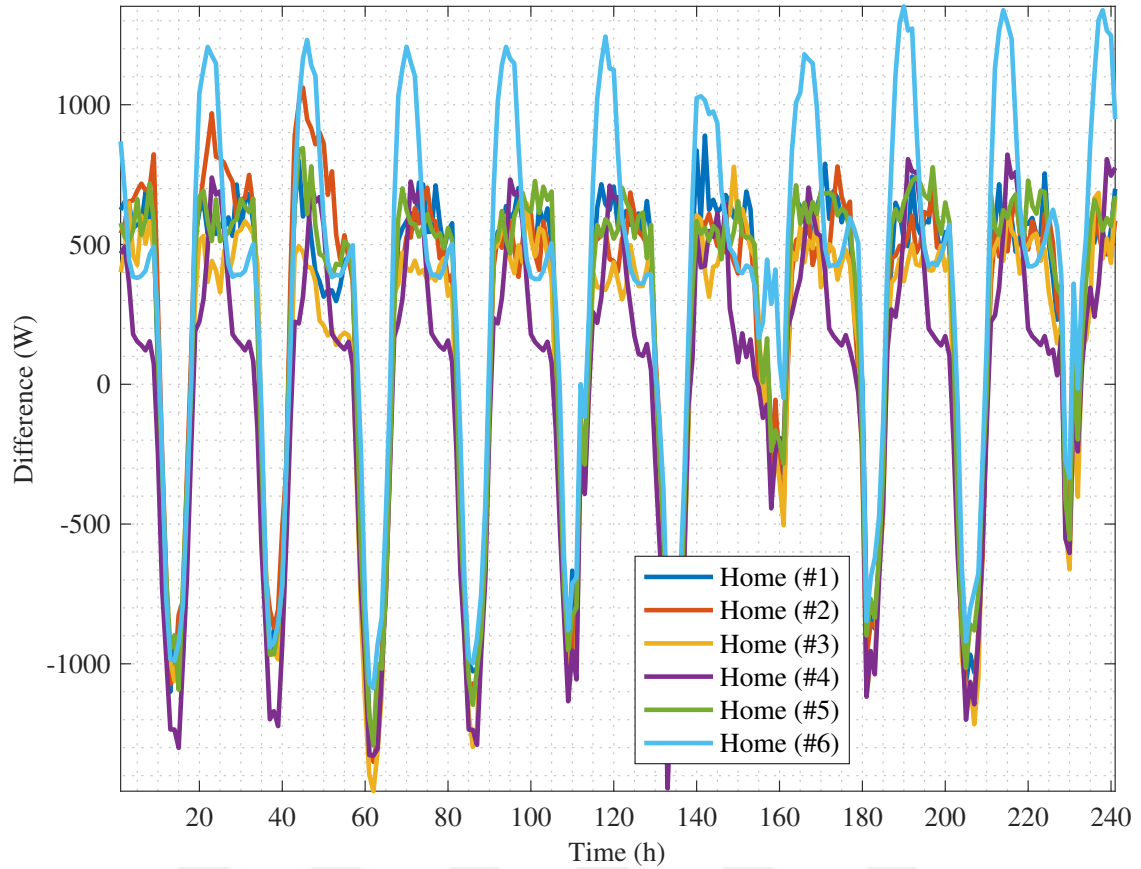


Figure 5.45: Difference Between Load Input and Solar Input

In the figure above, places above 0 indicate that the load is higher than the current solar production and the energy must be met from other sources, while places below 0 indicate that the load is lower than the current solar production and there is an excess of energy.

The total load, solar generation and difference values of the 10-day testing data, in terms of energy, are as in the table below.

Table 5.18: Descriptive Statistical Analysis of Load Power Data (Testing)

	Home #1	Home #2	Home #3	Home #4	Home #5	Home #6
Load (kWh)	196.79	193.83	137.46	107.20	166.75	194.20
Solar (kWh)	62.51	62.51	62.51	62.51	62.51	62.51
Difference (kWh)	134.28	131.32	74.94	44.68	104.24	131.69
Energy Shortage (kWh)	147.94	145.87	98.31	76.96	122.78	145.22
Energy Surplus (kWh)	13.66	14.55	23.36	32.28	18.55	13.53

As seen in the table, each house needs energy from outside in total. However, as can be seen from the previous figure, in some cases, energy demand occurs, and in some cases, energy surplus occurs. Resources that houses can supply energy in case of energy need,

1. The batteries in the central energy storage system (own resource),
2. The batteries of the electric vehicles (own resource), if there is an electric vehicle and if there is V2G functionality,
3. The neighbors, if there is P2P functionality.

In case of failure or insufficiency of these resources, the main grid supplies energy to the environment. In case of an excess of energy, the resources that the houses can transfer the excess energy to,

1. The batteries in the central energy storage system (own resource),
2. The batteries of the electric vehicles (own resource), if there is an electric vehicle and if there is V2G functionality,
3. The neighbors, if there is P2P functionality.

In cases of failure or insufficiency of these resources, houses can sell their energy surplus to the main grid.

Within the case studies, the charging graphics of the batteries in the central energy storage system of case #1, where reinforcement learning is applied, are as follows.

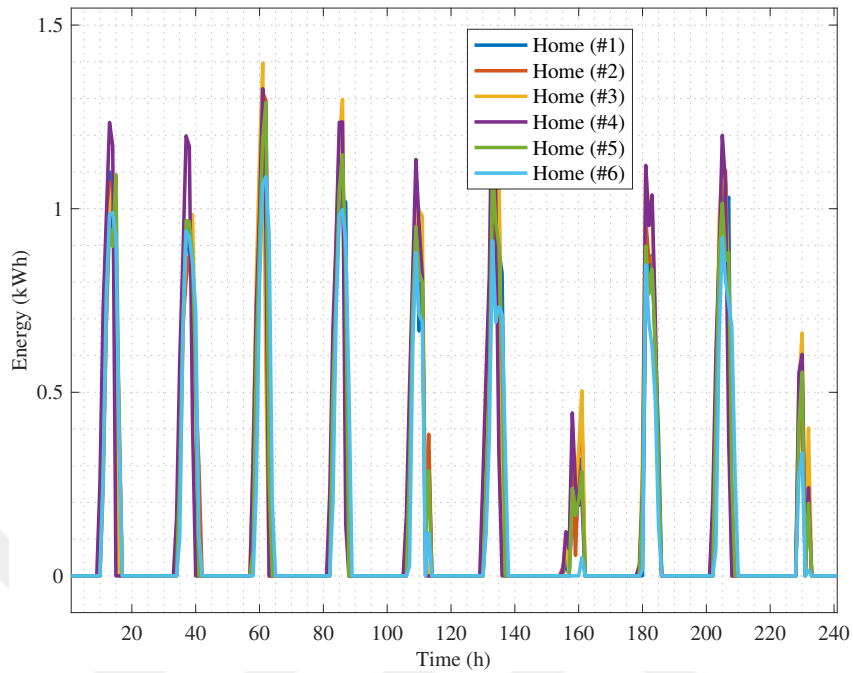


Figure 5.46: ESS Battery Charging Curves (Case #1)

The charging graphics of the batteries in the central energy storage system of case #2, where includes V2G functionality but no P2P functionality, are as follows.

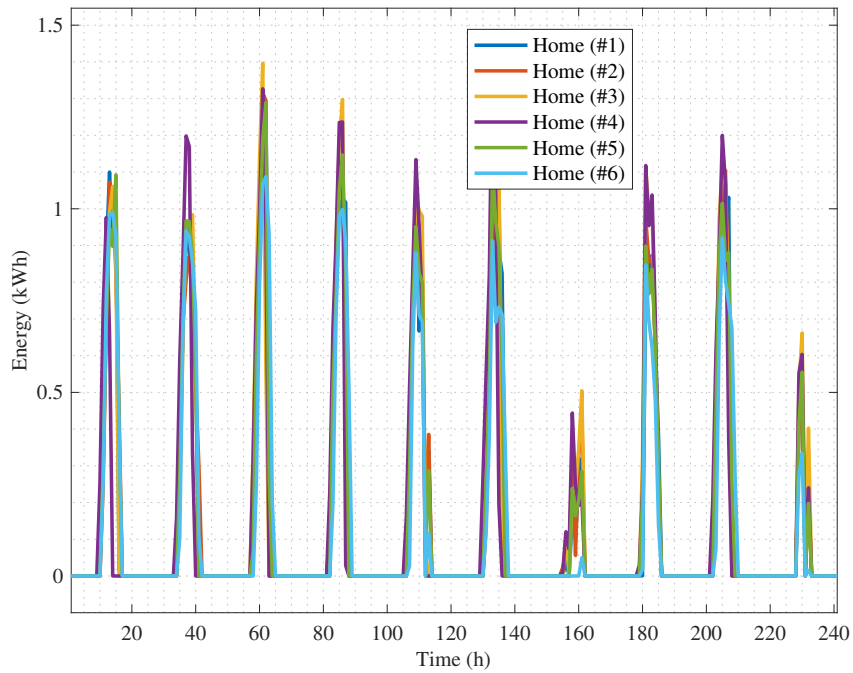


Figure 5.47: ESS Battery Charging Curves (Case #2)

The charging graphics of the batteries in the central energy storage system of case #3, where includes P2P functionality but no V2G functionality, are as follows.

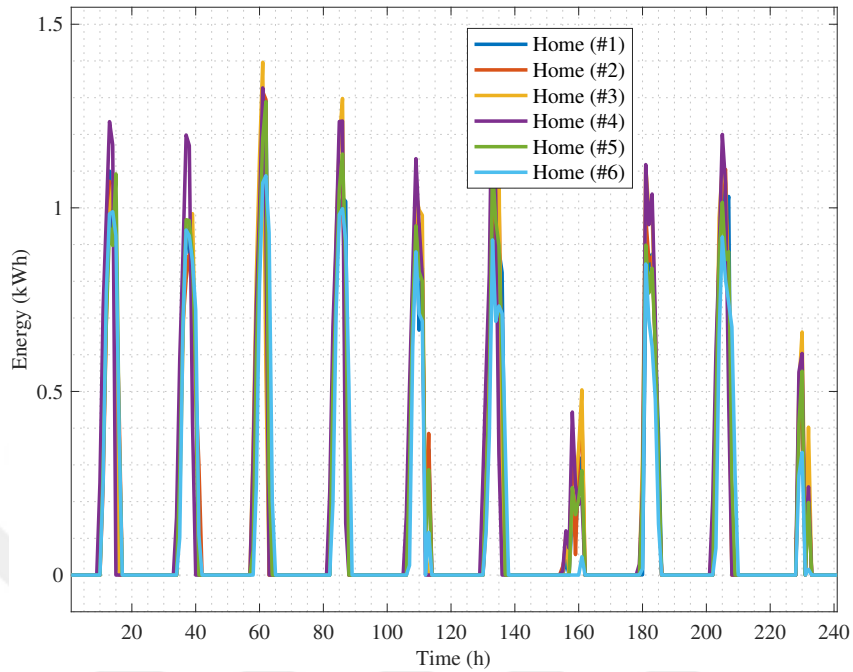


Figure 5.48: ESS Battery Charging Curves (Case #3)

The charging graphics of the batteries in the central energy storage system of case #4, where includes neither V2G functionality nor P2P functionality, are as follows.

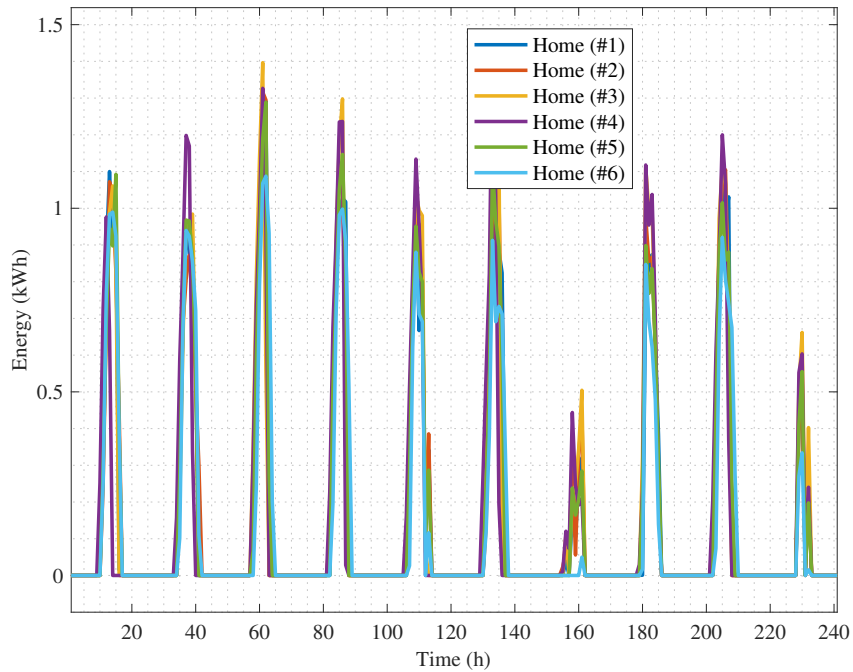


Figure 5.49: ESS Battery Charging Curves (Case #4)

The comparison table of the battery charge curves of each case is as follows.

Table 5.19: Comparison of Total ESS Battery Charging Energy

	Case #1 (kWh)	Case #2 (kWh)	Case #3 (kWh)	Case #4 (kWh)
Home #1	35,65	35,55	35,65	35,55
Home #2	36,21	36,21	36,21	36,21
Home #3	37,58	37,58	37,58	37,58
Home #4	37,77	35,61	37,89	35,61
Home #5	35,61	35,61	35,61	35,61
Home #6	31,83	31,83	31,83	31,83
Total	214,66	212,39	214,78	212,39

Within the case studies, the discharging graphics of the batteries in the central energy storage system of case #1, where reinforcement learning is applied, are as follows.

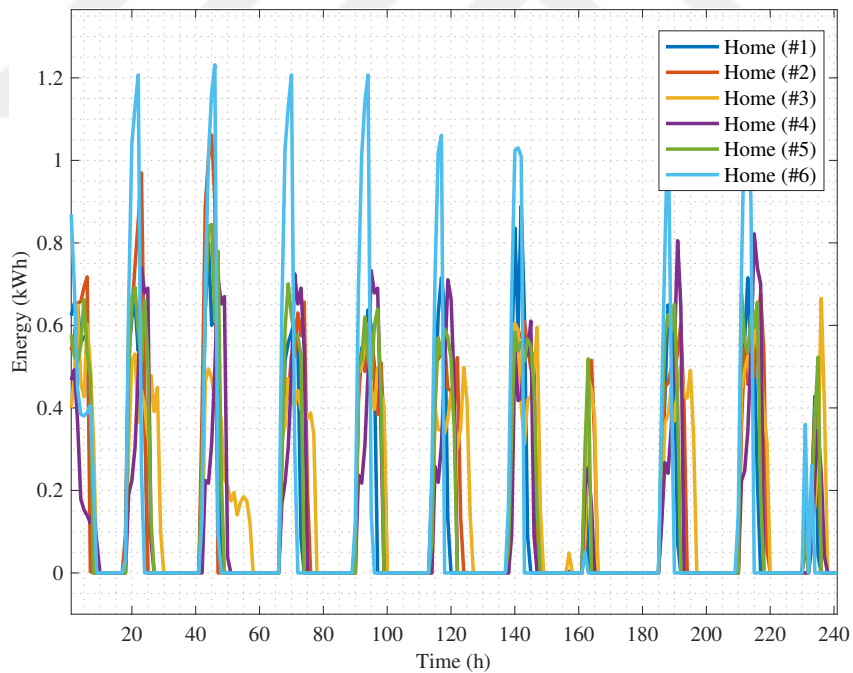


Figure 5.50: ESS Battery Discharging Curves (Case #1)

The discharging graphics of the batteries in the central energy storage system of case #2, where includes V2G functionality but no P2P functionality, are as follows.

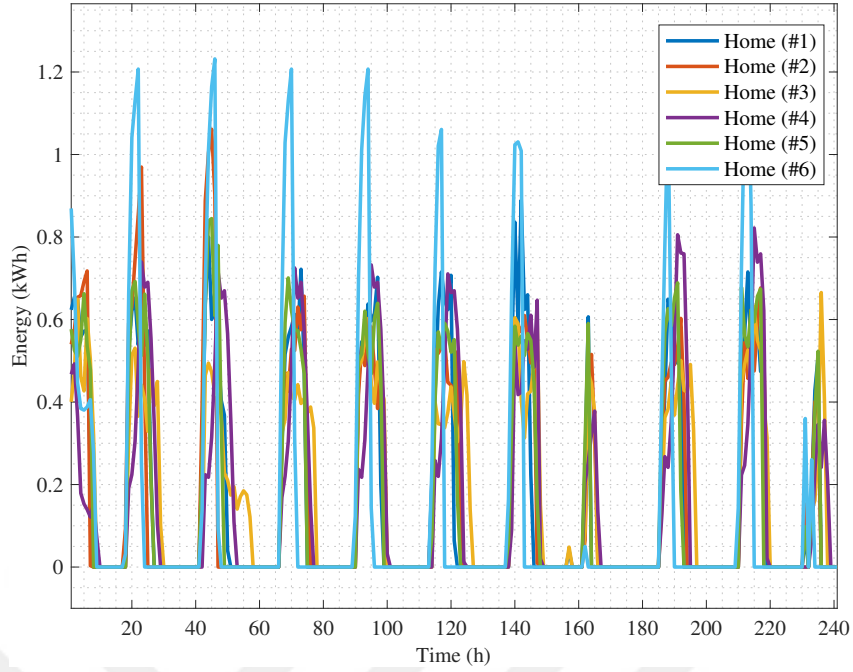


Figure 5.51: ESS Battery Discharging Curves (Case #2)

The discharging graphics of the batteries in the central energy storage system of case #3, where includes P2P functionality but no V2G functionality, are as follows.

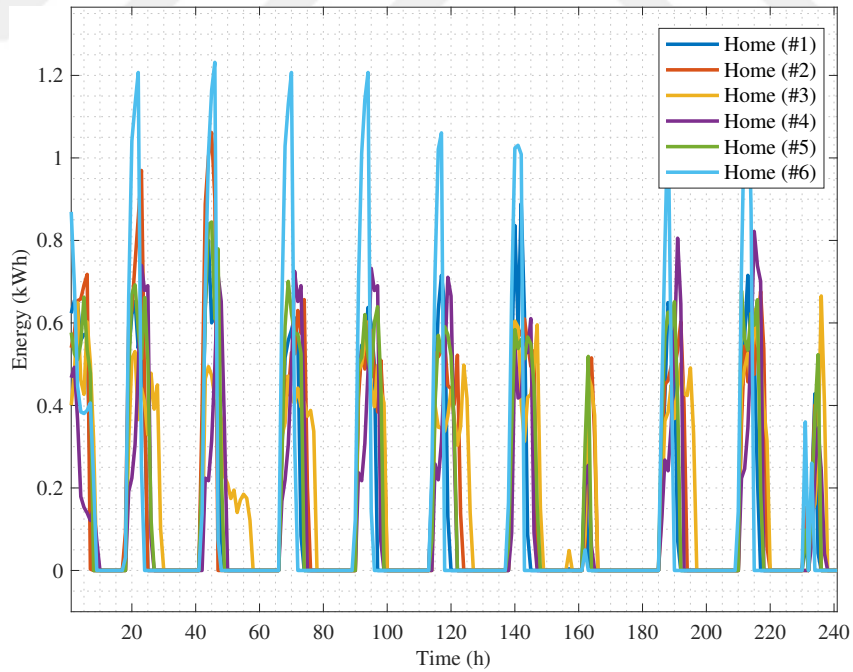


Figure 5.52: ESS Battery Discharging Curves (Case #3)

The discharging graphics of the batteries in the central energy storage system of case #4, where includes neither V2G functionality nor P2P functionality, are as follows.

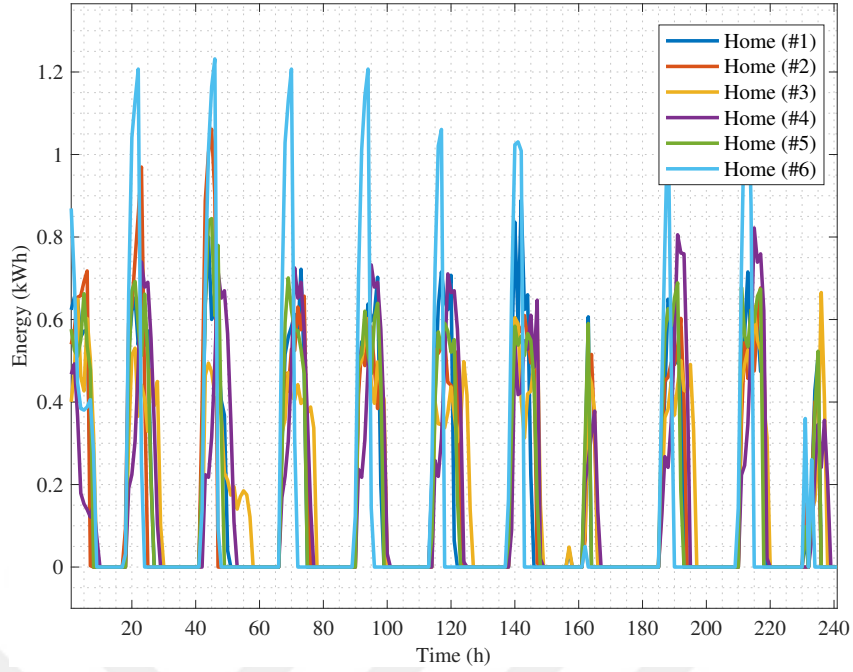


Figure 5.53: ESS Battery Discharging Curves (Case #4)

The comparison table of the battery discharge curves of each case is as follows.

Table 5.20: Comparison of Total ESS Battery Discharging Energy

	Case #1 (kWh)	Case #2 (kWh)	Case #3 (kWh)	Case #4 (kWh)
Home #1	30,09	39,39	30,09	39,39
Home #2	40,05	40,05	40,05	40,05
Home #3	41,42	41,42	41,42	41,42
Home #4	30,88	39,45	29,79	39,45
Home #5	37,41	39,45	37,41	39,45
Home #6	35,62	35,67	35,60	35,67
Total	215,47	235,43	214,35	235,43

Within the case studies, the energy selling graphics of the batteries in the central energy storage system during the P2P energy trading of case #1, where reinforcement learning is applied, are as follows.

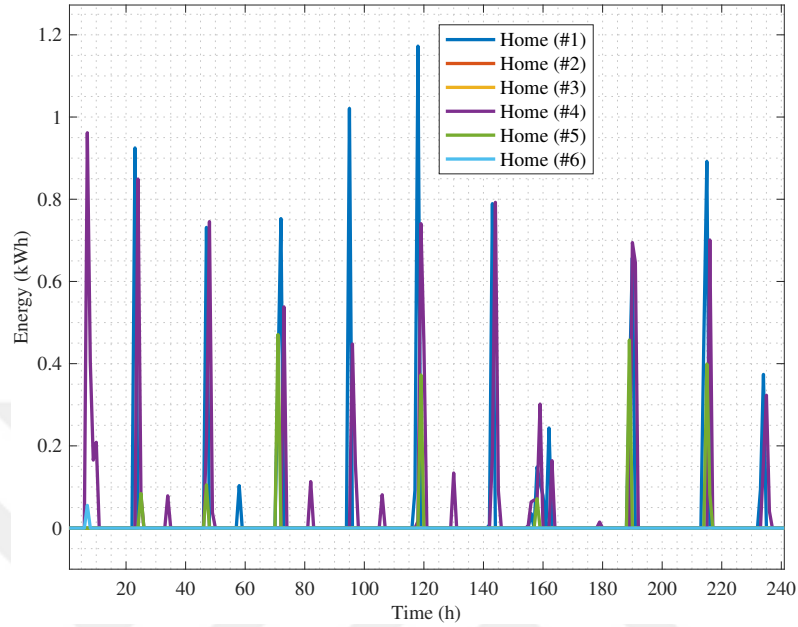


Figure 5.54: ESS Battery Energy Sold Curves (Case #1)

The energy selling graphics of the batteries in the central energy storage system during the P2P energy trading of case #2, where includes V2G functionality but no P2P functionality, are as follows.

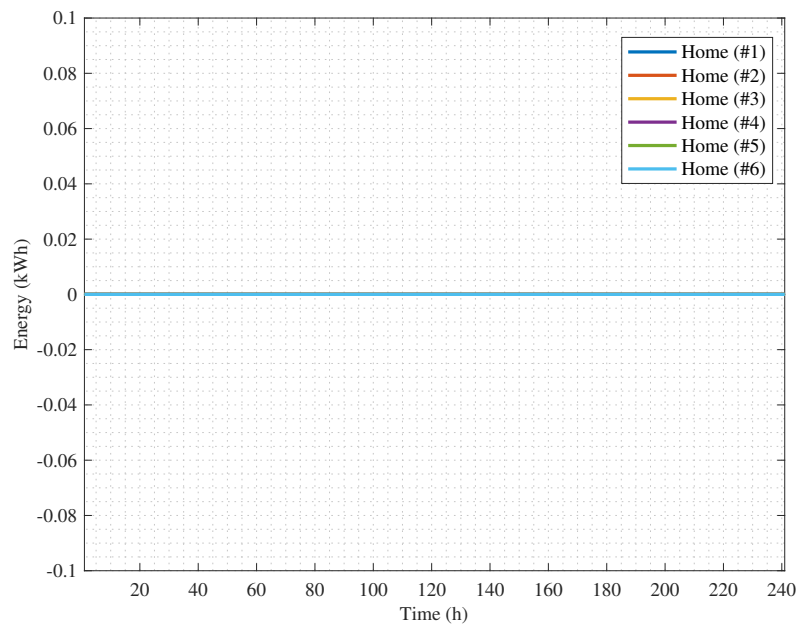


Figure 5.55: ESS Battery Energy Sold Curves (Case #2)

The energy selling graphics of the batteries in the central energy storage system during the P2P energy trading of case #3, where includes P2P functionality but no V2G functionality, are as follows.

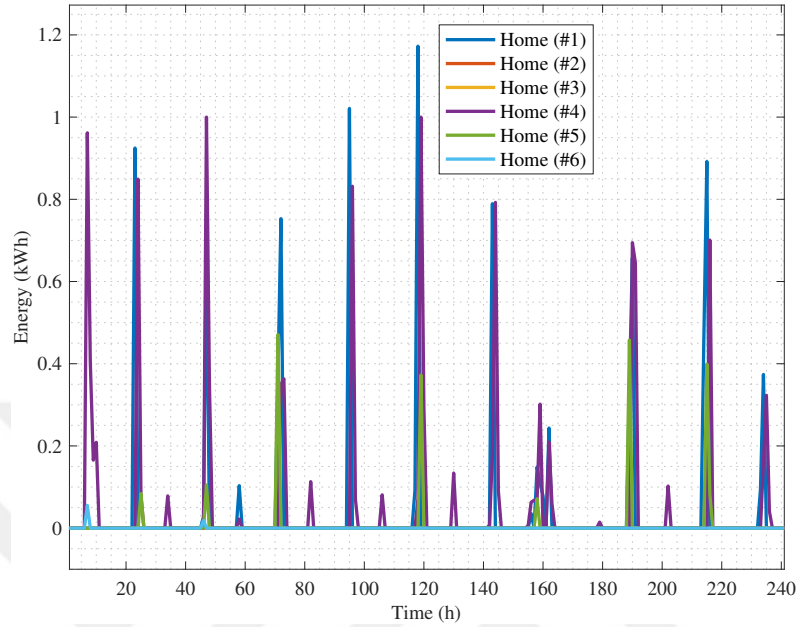


Figure 5.56: ESS Battery Energy Sold Curves (Case #3)

The energy selling graphics of the batteries in the central energy storage system during the P2P energy trading of case #4, where includes neither V2G functionality nor P2P functionality, are as follows.

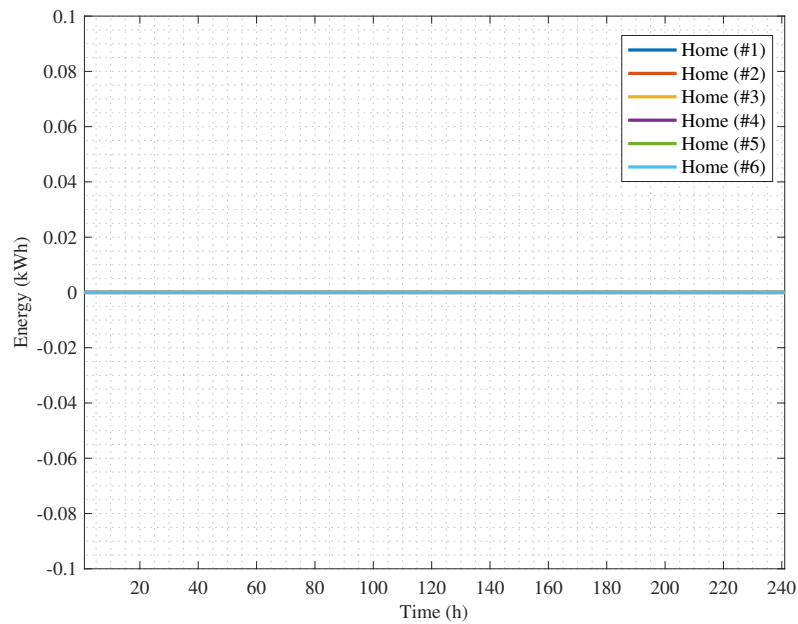


Figure 5.57: ESS Battery Energy Sold Curves (Case #4)

The comparison table of the energy selling curves of each case is as follows.

Table 5.21: Comparison of Total Battery P2P Selling Energy

	Case #1 (kWh)	Case #2 (kWh)	Case #3 (kWh)	Case #4 (kWh)
Home #1	9,40	0,00	9,41	0,00
Home #2	0,00	0,00	0,00	0,00
Home #3	0,00	0,00	0,00	0,00
Home #4	10,73	0,00	11,95	0,00
Home #5	2,04	0,00	2,04	0,00
Home #6	0,06	0,00	0,07	0,00
Total	22,23	0,00	23,47	0,00

Within the case studies, the charging graphics of the batteries in the electric vehicles of case #1, where reinforcement learning is applied, are as follows.

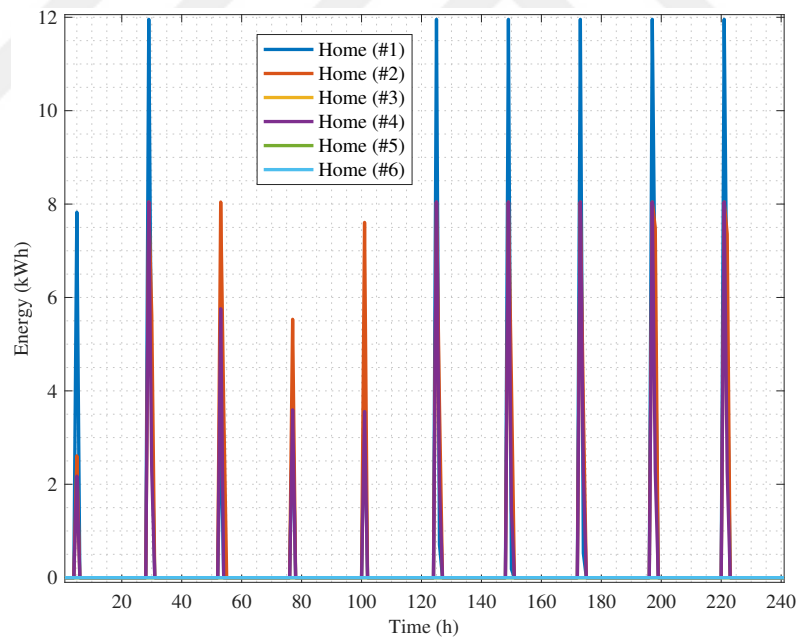


Figure 5.58: EV Battery Charging Curves (Case #1)

The charging graphics of the batteries in the electric vehicles of case #2, where includes V2G functionality but no P2P functionality, are as follows.

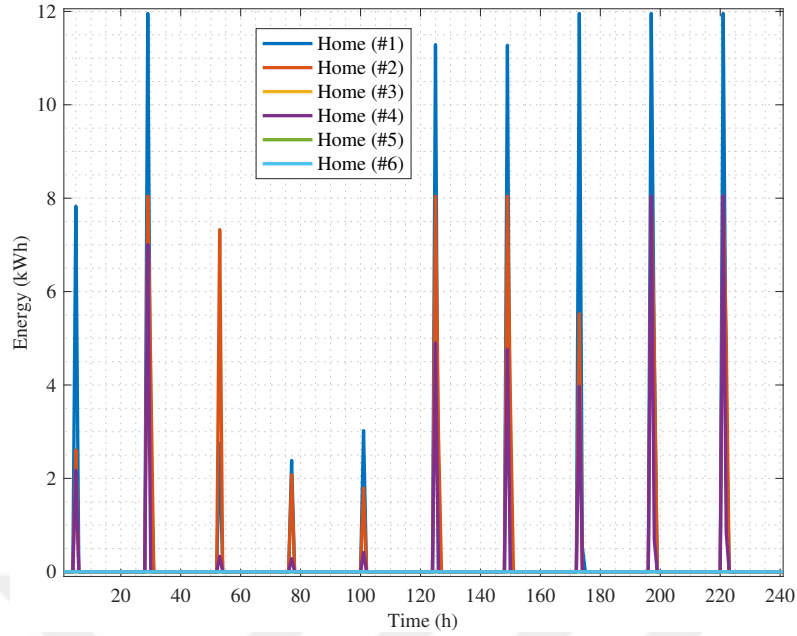


Figure 5.59: EV Battery Charging Curves (Case #2)

The charging graphics of the batteries in the electric vehicles of case #3, where includes P2P functionality but no V2G functionality, are as follows.

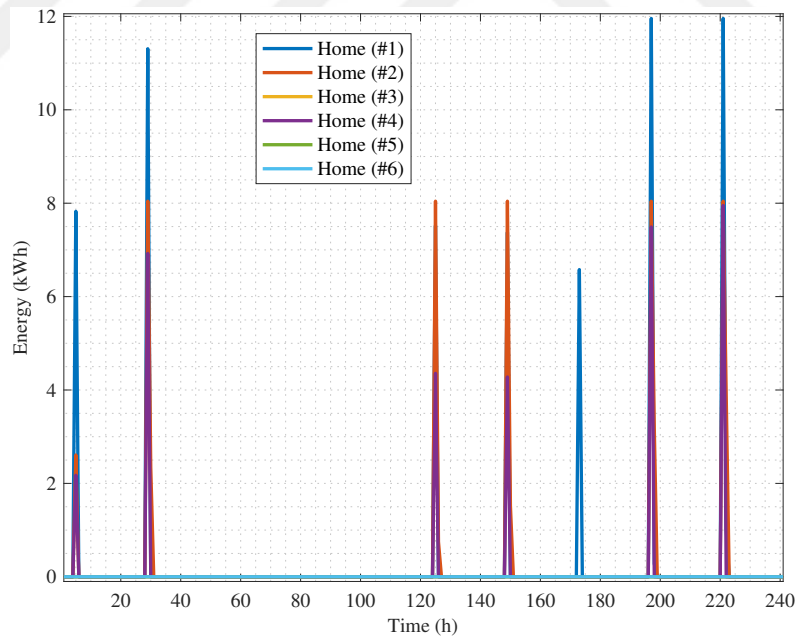


Figure 5.60: EV Battery Charging Curves (Case #3)

The charging graphics of the batteries in the electric vehicles of case #4, where includes neither V2G functionality nor P2P functionality, are as follows.

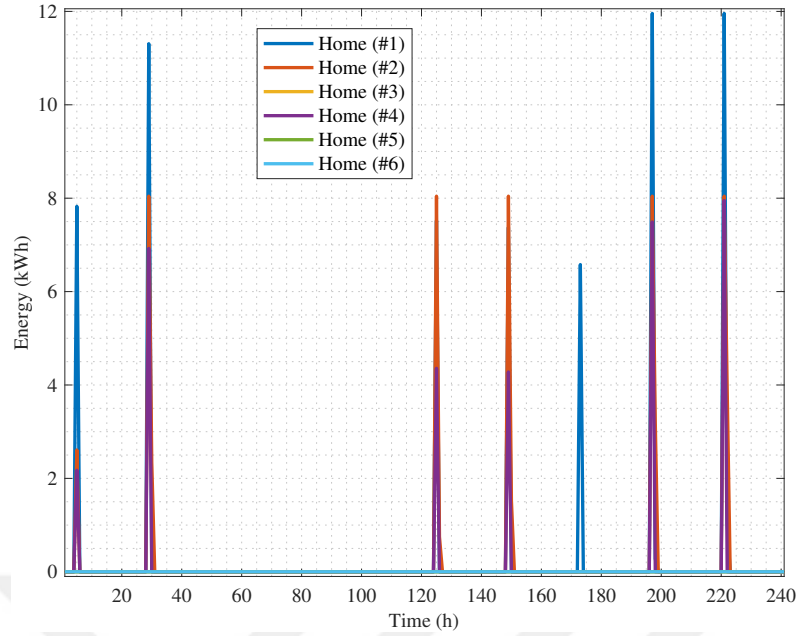


Figure 5.61: EV Battery Charging Curves (Case #4)

The comparison table of the electric vehicle charging curves of each case is as follows.

Table 5.22: Comparison of Total EV Battery Charging Energy

	Case #1 (kWh)	Case #2 (kWh)	Case #3 (kWh)	Case #4 (kWh)
Home #1	106,04	97,11	65,83	65,83
Home #2	107,82	80,34	54,52	54,52
Home #3	0,00	0,00	0,00	0,00
Home #4	75,94	41,44	33,17	33,17
Home #5	0,00	0,00	0,00	0,00
Home #6	0,00	0,00	0,00	0,00
Total	289,80	218,89	153,52	153,52

Within the case studies, the discharging graphics of the batteries in the electric vehicles of case #1, where reinforcement learning is applied, are as follows.

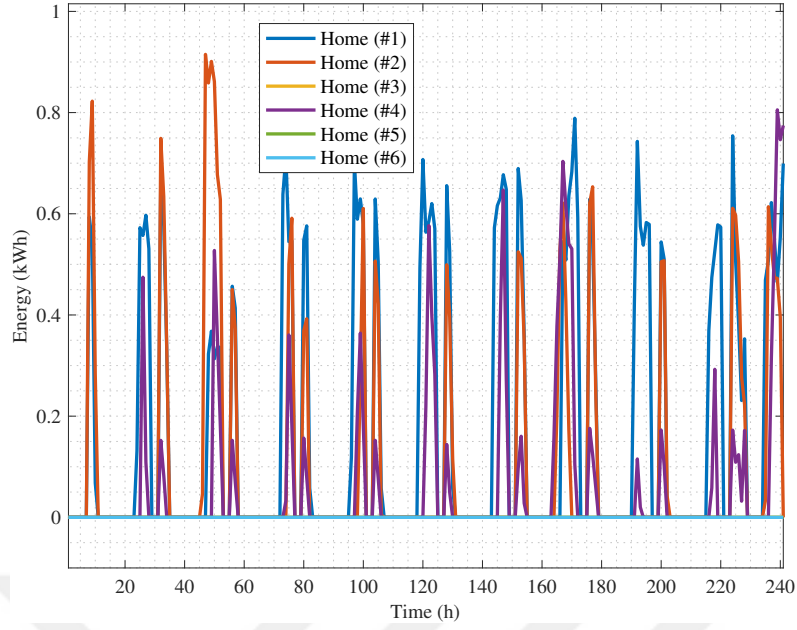


Figure 5.62: EV Battery Discharging Curves (Case #1)

The discharging graphics of the batteries in the electric vehicles of case #2, where includes V2G functionality but no P2P functionality, are as follows.

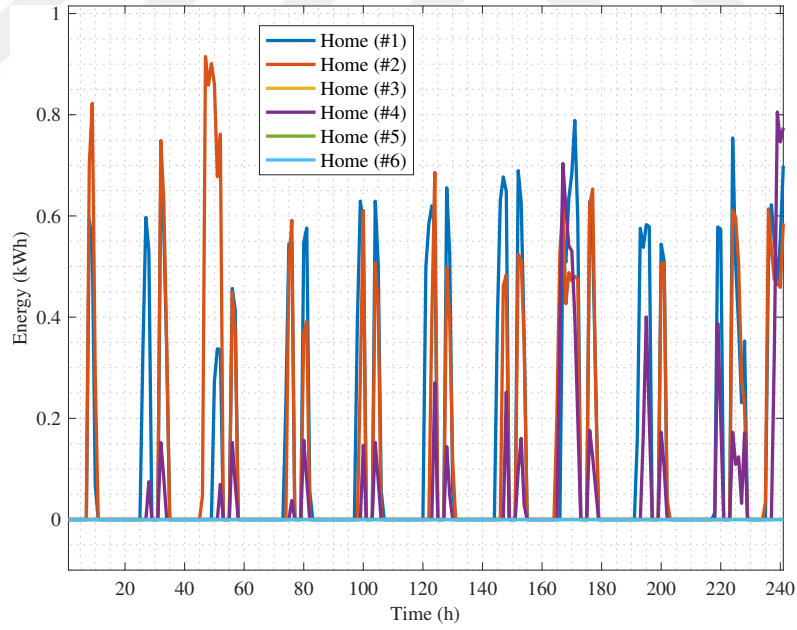


Figure 5.63: EV Battery Discharging Curves (Case #2)

The discharging graphics of the batteries in the electric vehicles of case #3, where includes P2P functionality but no V2G functionality, are as follows.

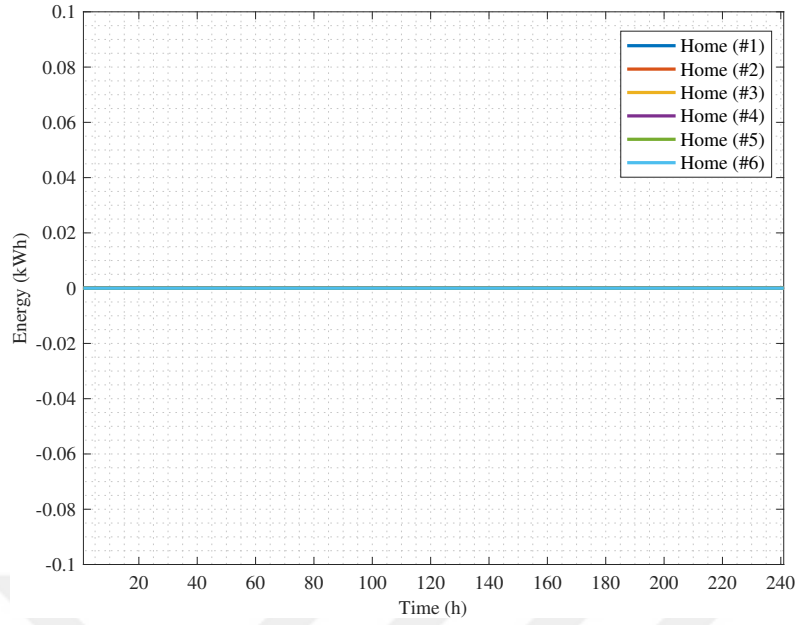


Figure 5.64: EV Battery Discharging Curves (Case #3)

The discharging graphics of the batteries in the electric vehicles of case #4, where includes neither V2G functionality nor P2P functionality, are as follows.

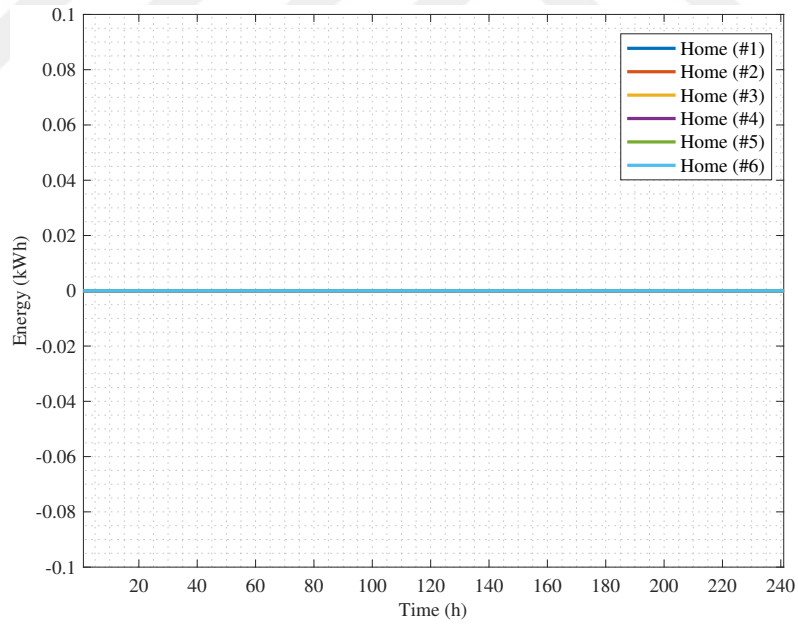


Figure 5.65: EV Battery Discharging Curves (Case #4)

The comparison table of the electric vehicle discharging curves of each case is as follows.

Table 5.23: Comparison of Total EV Battery Charging Energy

	Case #1 (kWh)	Case #2 (kWh)	Case #3 (kWh)	Case #4 (kWh)
Home #1	43,09	34,40	0,00	0,00
Home #2	24,59	29,14	0,00	0,00
Home #3	0,00	0,00	0,00	0,00
Home #4	15,62	10,88	0,00	0,00
Home #5	0,00	0,00	0,00	0,00
Home #6	0,00	0,00	0,00	0,00
Total	83,30	74,42	0,00	0,00

Within the case studies, the energy selling graphics of the batteries in the electric vehicles during the P2P energy trading of case #1, where reinforcement learning is applied, are as follows.

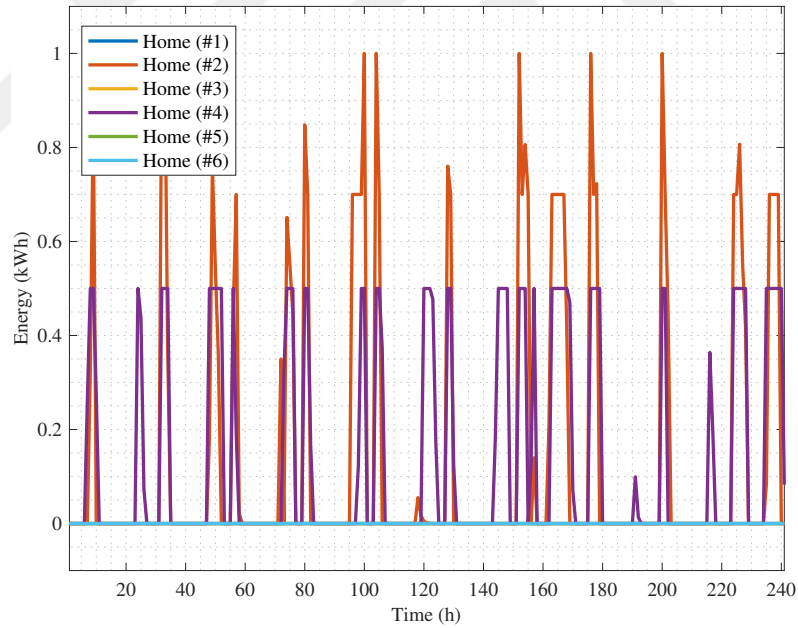


Figure 5.66: EV Battery Energy Sold Curves (Case #1)

The energy selling graphics of the batteries in the electric vehicles during the P2P energy trading of case #2, where includes V2G functionality but no P2P functionality, are as follows.

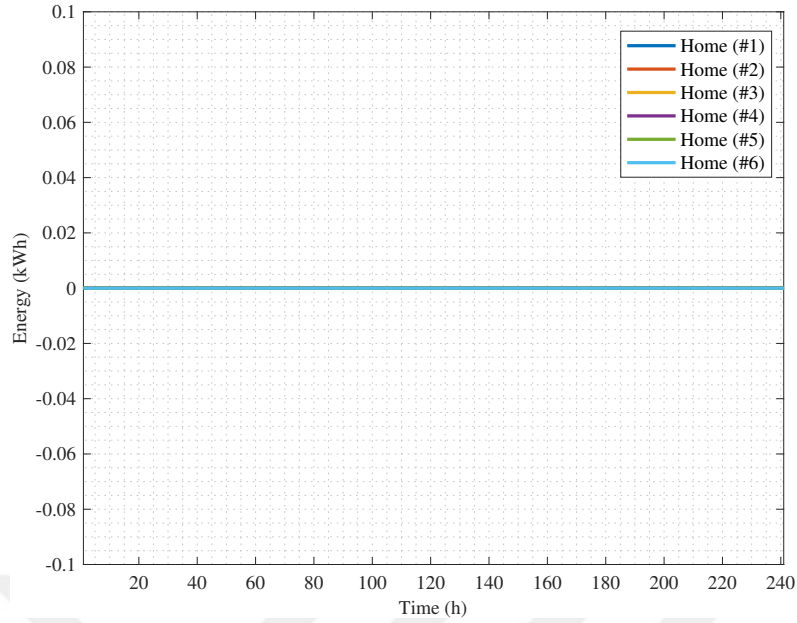


Figure 5.67: EV Battery Energy Sold Curves (Case #2)

The energy selling graphics of the batteries in the electric vehicles during the P2P energy trading of case #3, where includes P2P functionality but no V2G functionality, are as follows.

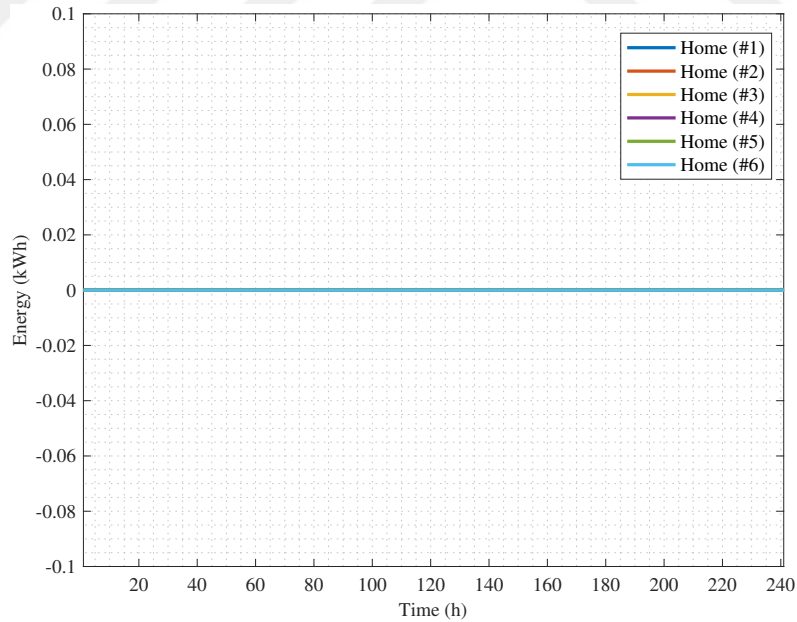


Figure 5.68: EV Battery Energy Sold Curves (Case #3)

The energy selling graphics of the batteries in the electric vehicles during the P2P energy trading of case #4, where includes neither V2G functionality nor P2P functionality, are as follows.

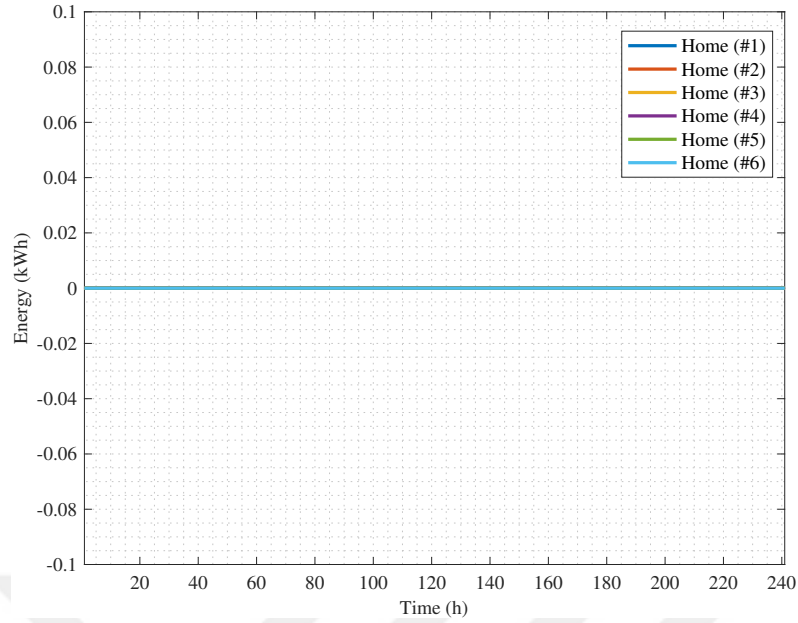


Figure 5.69: EV Battery Energy Sold Curves (Case #4)

The comparison table of the energy selling curves of each case is as follows.

Table 5.24: Comparison of Total Electric Vehicle P2P Selling Energy

	Case #1 (kWh)	Case #2 (kWh)	Case #3 (kWh)	Case #4 (kWh)
Home #1	0,00	0,00	0,00	0,00
Home #2	35,25	0,00	0,00	0,00
Home #3	0,00	0,00	0,00	0,00
Home #4	33,10	0,00	0,00	0,00
Home #5	0,00	0,00	0,00	0,00
Home #6	0,00	0,00	0,00	0,00
Total	68,35	0,00	0,00	0,00

Within the case studies, the graphics of bought energy from the main grid of case #1, where reinforcement learning is applied, are as follows.

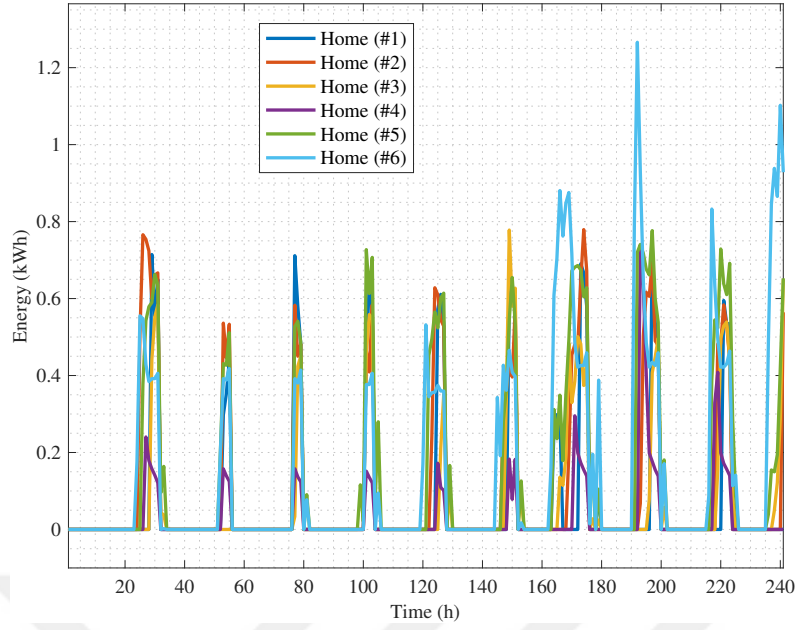


Figure 5.70: Energy Bought from Grid (Case #1)

The graphics of bought energy from the main grid of case #2, where includes V2G functionality but no P2P functionality, are as follows.

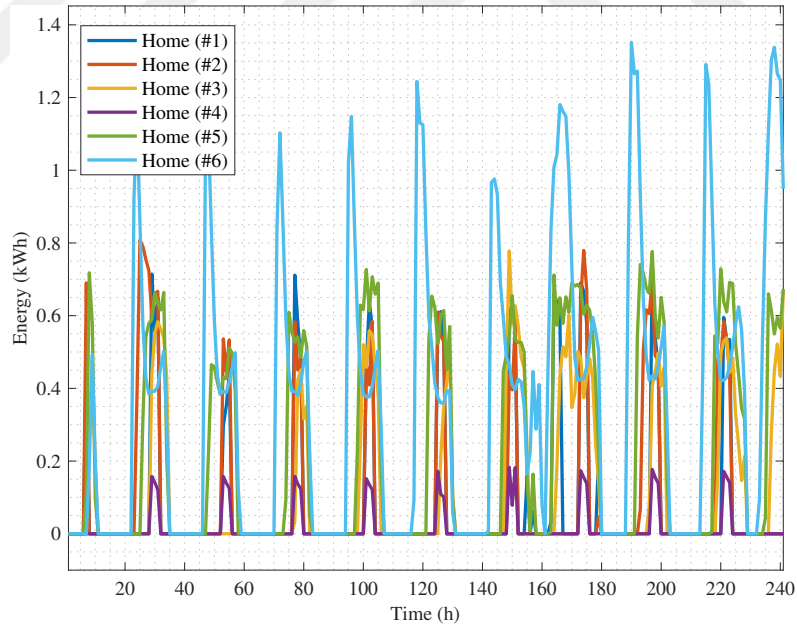


Figure 5.71: Energy Bought from Grid (Case #2)

The graphics of bought energy from the main grid of case #3, where includes P2P functionality but no V2G functionality, are as follows.

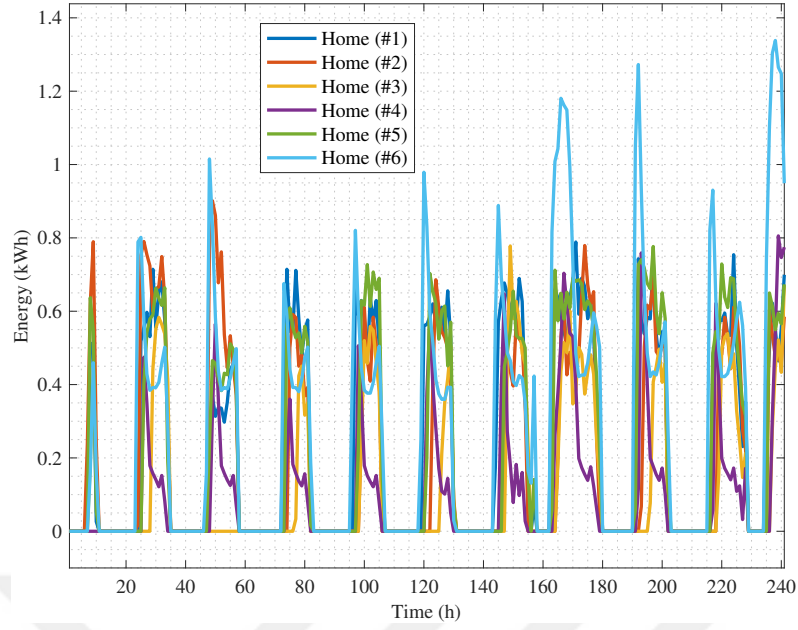


Figure 5.72: Energy Bought from Grid (Case #3)

The graphics of bought energy from the main grid of case #4, where includes neither V2G functionality nor P2P functionality, are as follows.

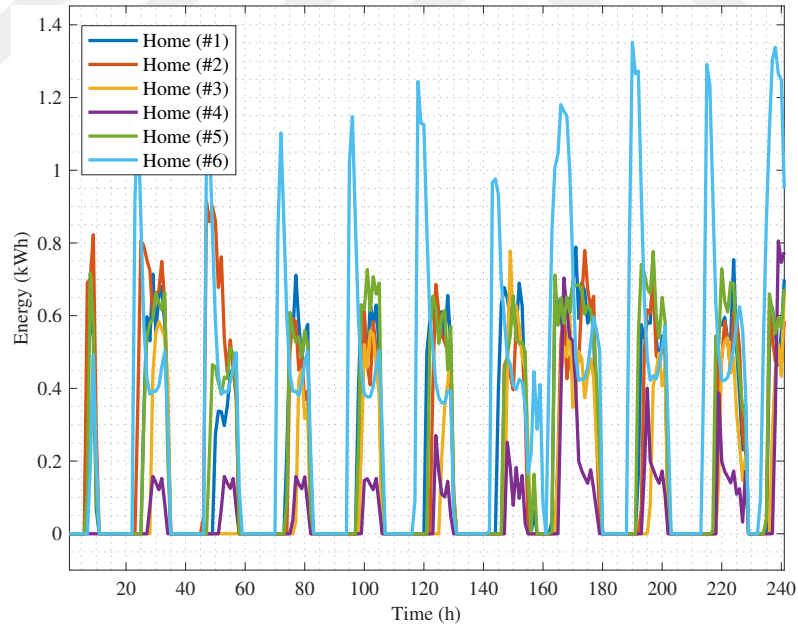


Figure 5.73: Energy Bought from Grid (Case #4)

The comparison table of the bought energy from the main grid of each case is as follows.

Table 5.25: Comparison of Total Energy Bought from Grid

	Case #1 (kWh)	Case #2 (kWh)	Case #3 (kWh)	Case #4 (kWh)
Home #1	16,25	18,13	58,66	52,54
Home #2	24,02	21,76	48,72	50,91
Home #3	15,54	28,82	28,61	28,82
Home #4	7,72	3,90	24,43	14,78
Home #5	33,34	54,64	55,87	54,64
Home #6	36,28	85,38	68,37	85,38
Total	133,15	212,64	284,67	287,06

Within the case studies, the graphics of sold energy to the main grid of case #1, where reinforcement learning is applied, are as follows.

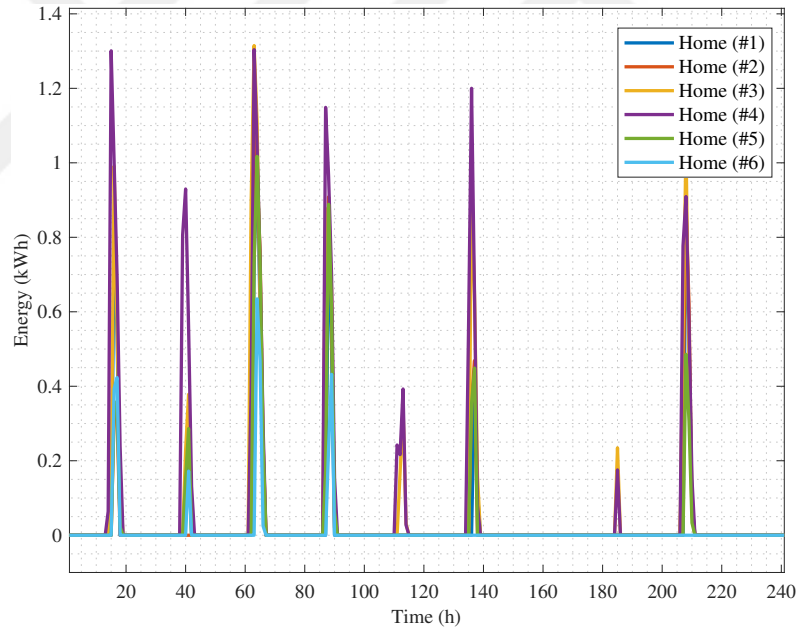


Figure 5.74: Energy Sold to Grid (Case #1)

The graphics of sold energy to the main grid of case #2, where includes V2G functionality but no P2P functionality, are as follows.

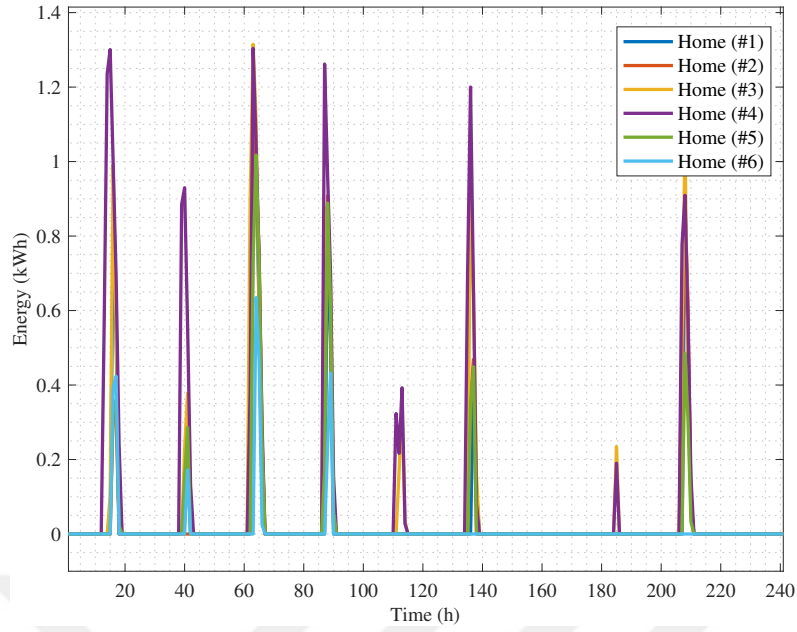


Figure 5.75: Energy Sold to Grid (Case #2)

The graphics of sold energy to the main grid of case #3, where includes P2P functionality but no V2G functionality, are as follows.

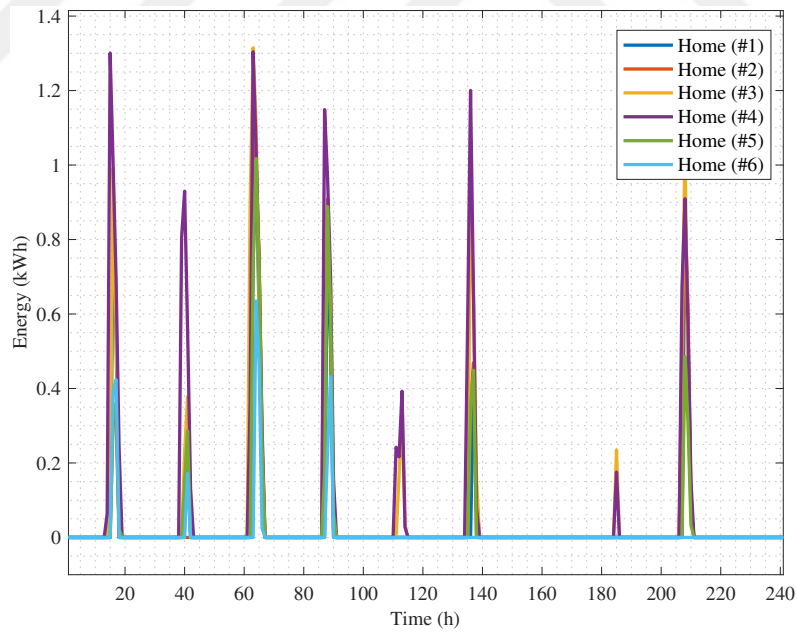


Figure 5.76: Energy Sold to Grid (Case #3)

The graphics of sold energy to the main grid of case #4, where includes neither V2G functionality nor P2P functionality, are as follows.

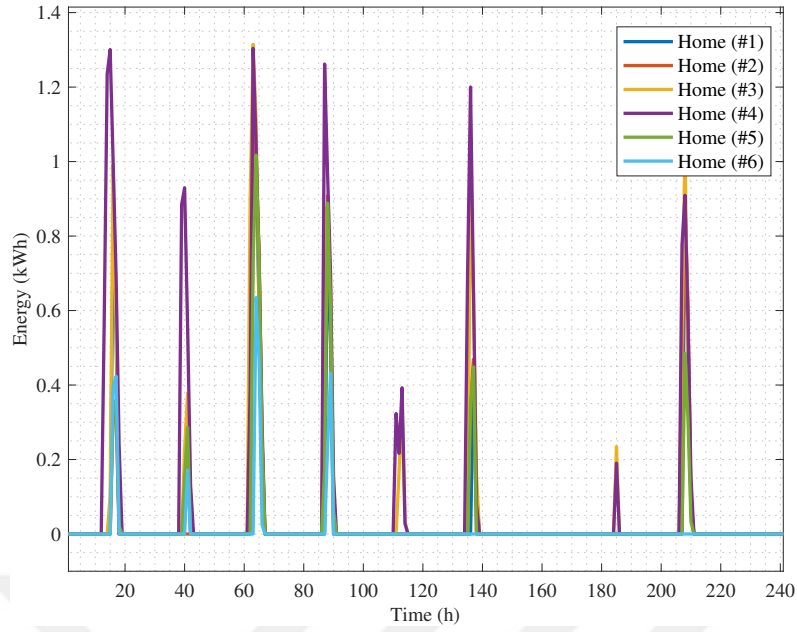


Figure 5.77: Energy Sold to Grid (Case #4)

The comparison table of the sold energy to the main grid of each case is as follows.

Table 5.26: Comparison of Total Energy Sold to Grid

	Case #1 (kWh)	Case #2 (kWh)	Case #3 (kWh)	Case #4 (kWh)
Home #1	7,50	7,60	7,50	7,60
Home #2	8,76	8,76	8,76	8,76
Home #3	13,56	13,56	13,56	13,56
Home #4	18,40	20,56	18,28	20,56
Home #5	7,24	7,24	7,24	7,24
Home #6	2,78	2,78	2,78	2,78
Total	58,23	60,50	58,11	60,50

Within the case studies, the graphics of bought energy during P2P energy trading of case #1, where reinforcement learning is applied, are as follows.

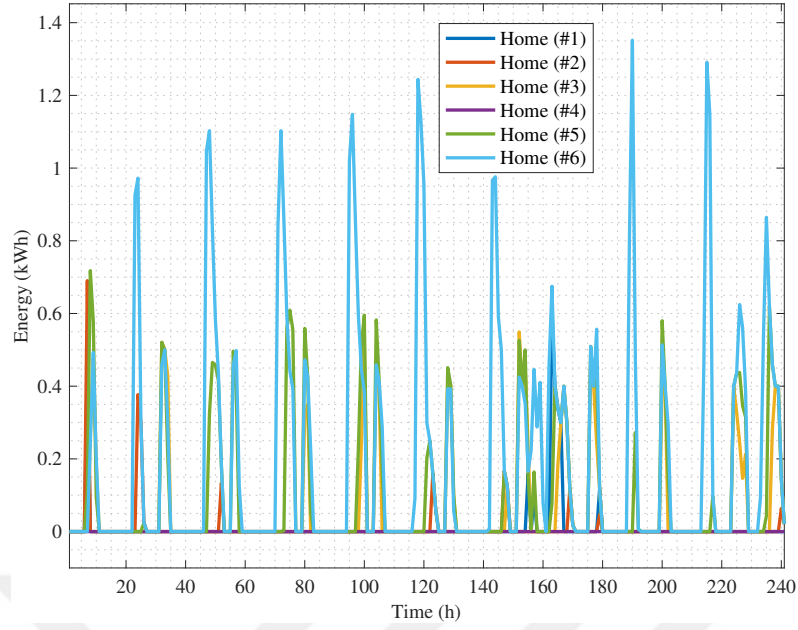


Figure 5.78: Energy Bought via P2P (Case #1)

The graphics of bought energy during P2P energy trading of case #2, where includes V2G functionality but no P2P functionality, are as follows.

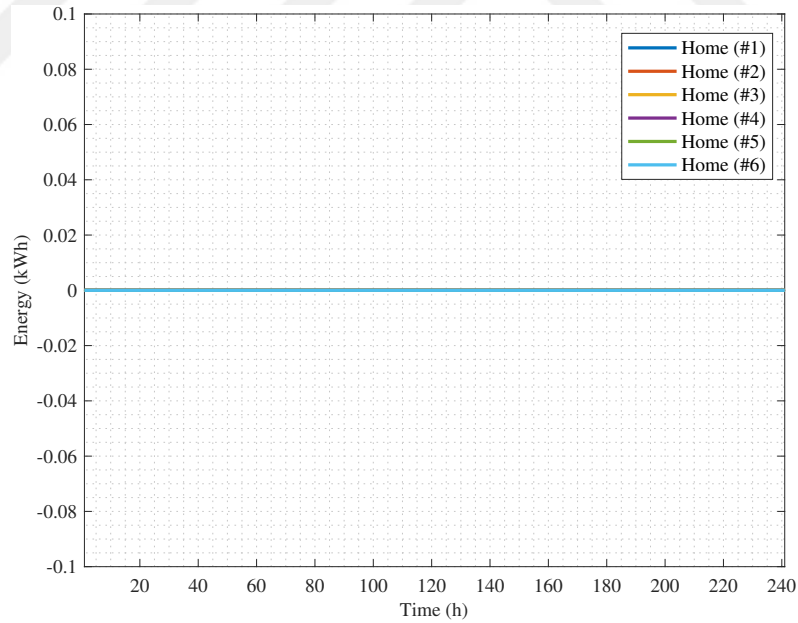


Figure 5.79: Energy Bought via P2P (Case #2)

The graphics of bought energy during P2P energy trading of case #3, where includes P2P functionality but no V2G functionality, are as follows.

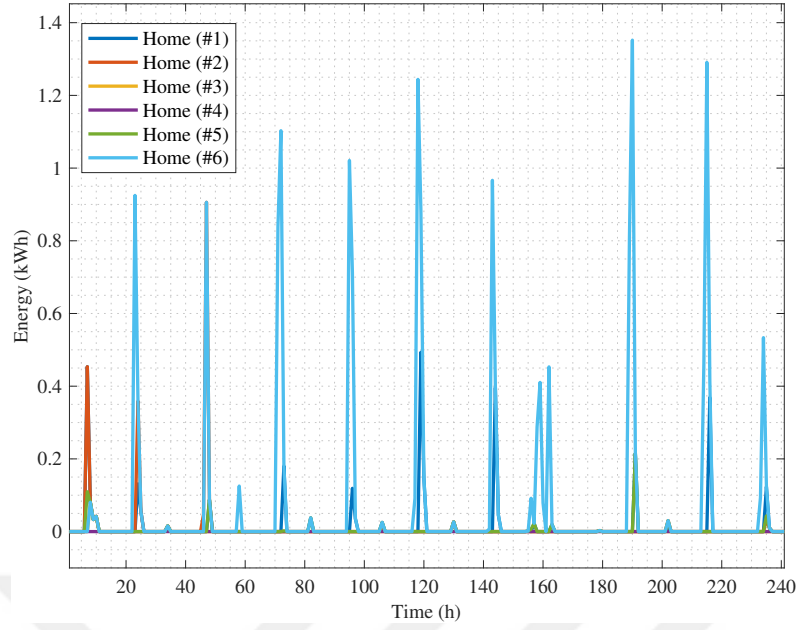


Figure 5.80: Energy Bought via P2P (Case #3)

The graphics of bought energy during P2P energy trading of case #4, where includes neither V2G functionality nor P2P functionality, are as follows.

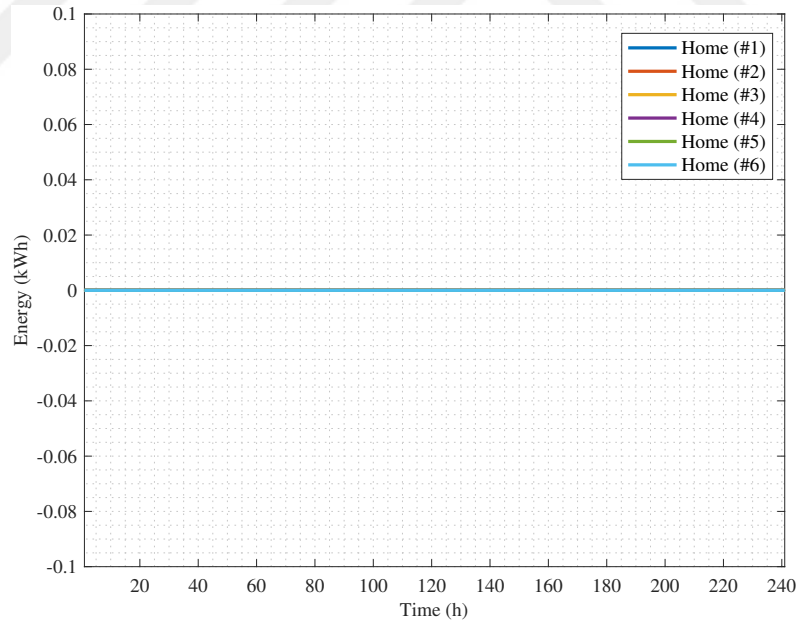


Figure 5.81: Energy Bought via P2P (Case #4)

The comparison table of the bought energy via P2P energy trading of each case is as follows.

Table 5.27: Comparison of Total Energy Bought via P2P

	Case #1 (kWh)	Case #2 (kWh)	Case #3 (kWh)	Case #4 (kWh)
Home #1	2,50	0,00	3,18	0,00
Home #2	2,60	0,00	2,19	0,00
Home #3	13,28	0,00	0,20	0,00
Home #4	0,00	0,00	0,00	0,00
Home #5	23,34	0,00	0,81	0,00
Home #6	49,16	0,00	17,08	0,00
Total	90,58	0,00	23,47	0,00

Within the case studies, the graphics of sold energy during P2P energy trading of case #1, where reinforcement learning is applied, are as follows.

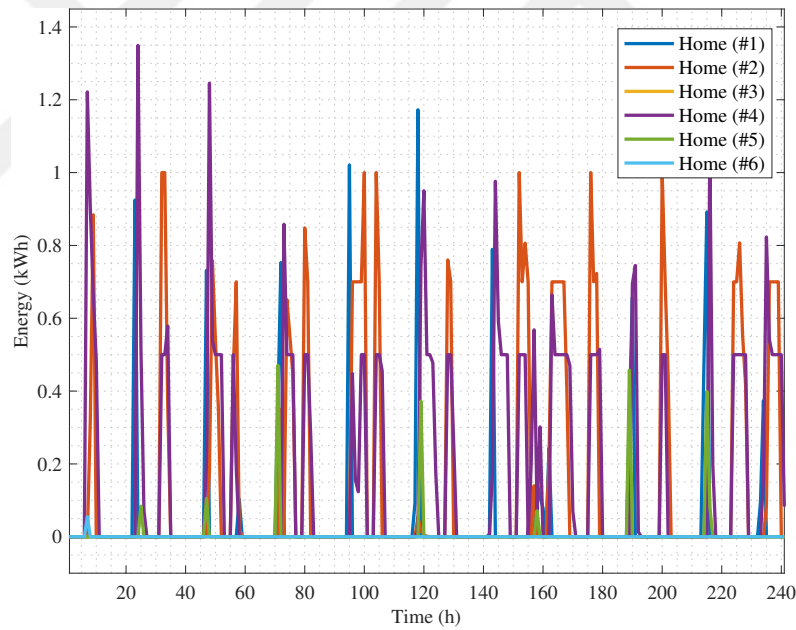


Figure 5.82: Energy Sold via P2P (Case #1)

The graphics of sold energy during P2P energy trading of case #2, where includes V2G functionality but no P2P functionality, are as follows.

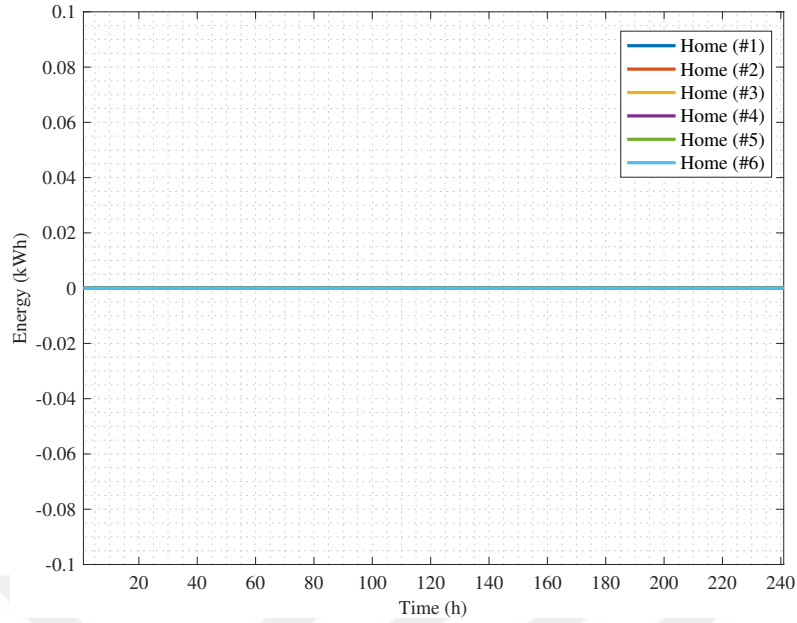


Figure 5.83: Energy Sold via P2P (Case #2)

The graphics of sold energy during P2P energy trading of case #3, where includes P2P functionality but no V2G functionality, are as follows.

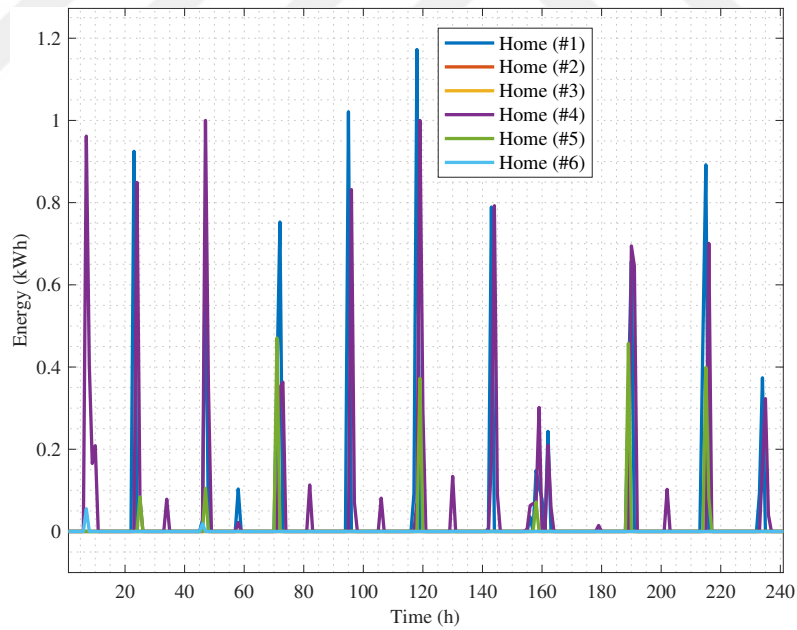


Figure 5.84: Energy Sold via P2P (Case #3)

The graphics of sold energy during P2P energy trading of case #4, where includes neither V2G functionality nor P2P functionality, are as follows.

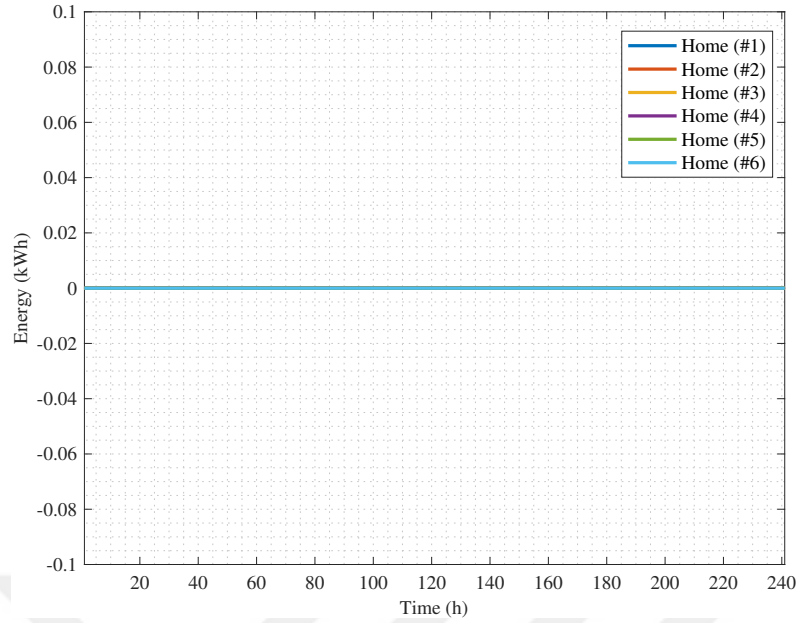


Figure 5.85: Energy Sold via P2P (Case #4)

The comparison table of the sold energy via P2P energy trading of each case is as follows.

Table 5.28: Comparison of Total Energy Sold via P2P

	Case #1 (kWh)	Case #2 (kWh)	Case #3 (kWh)	Case #4 (kWh)
Home #1	9,40	0,00	9,41	0,00
Home #2	35,25	0,00	0,00	0,00
Home #3	0,00	0,00	0,00	0,00
Home #4	43,83	0,00	11,95	0,00
Home #5	2,04	0,00	2,04	0,00
Home #6	0,06	0,00	0,07	0,00
Total	90,58	0,00	23,47	0,00

Within the case studies, the graphics of SoC of ESSs of case #1, where reinforcement learning is applied, are as follows.

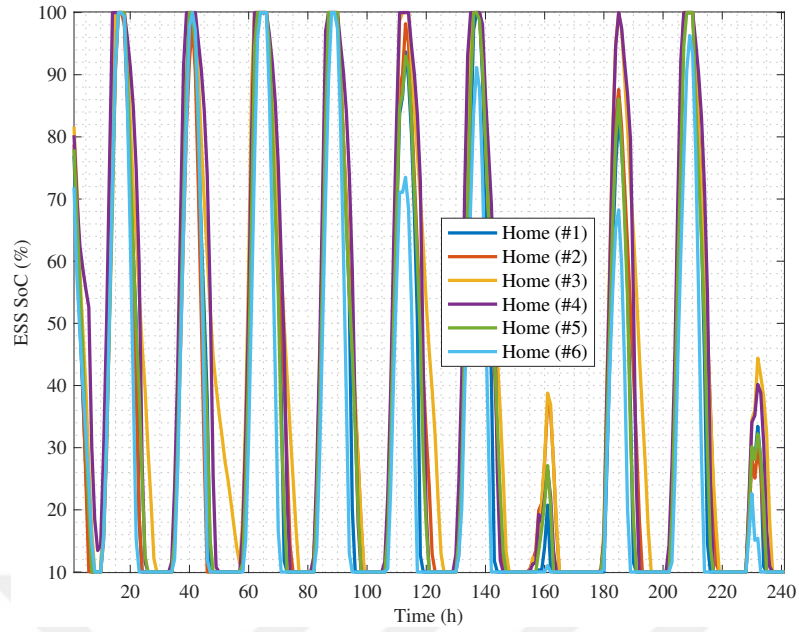


Figure 5.86: SoC of the ESS Battery (Case #1)

The graphics of SoC of ESSs of case #2, where includes V2G functionality but no P2P functionality, are as follows.

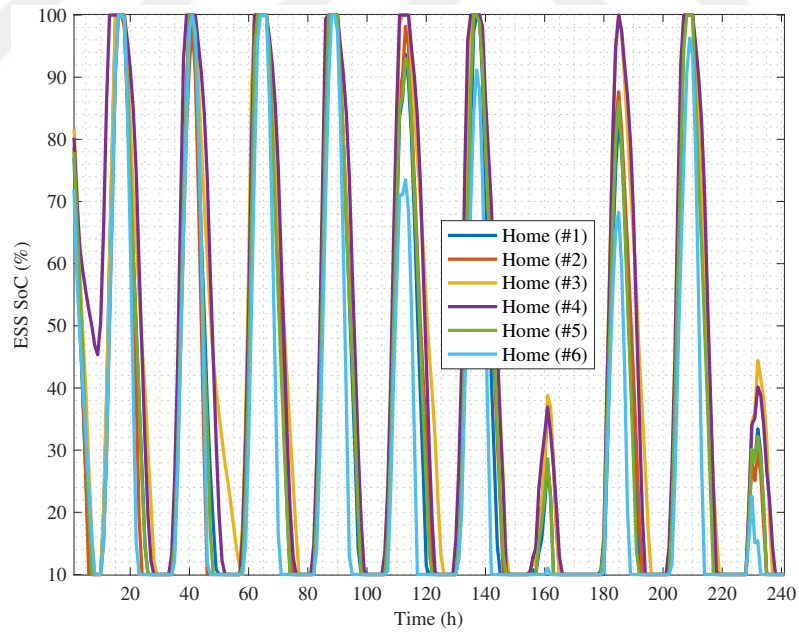


Figure 5.87: SoC of the ESS Battery (Case #2)

The graphics of SoC of ESSs of case #3, where includes P2P functionality but no V2G functionality, are as follows.

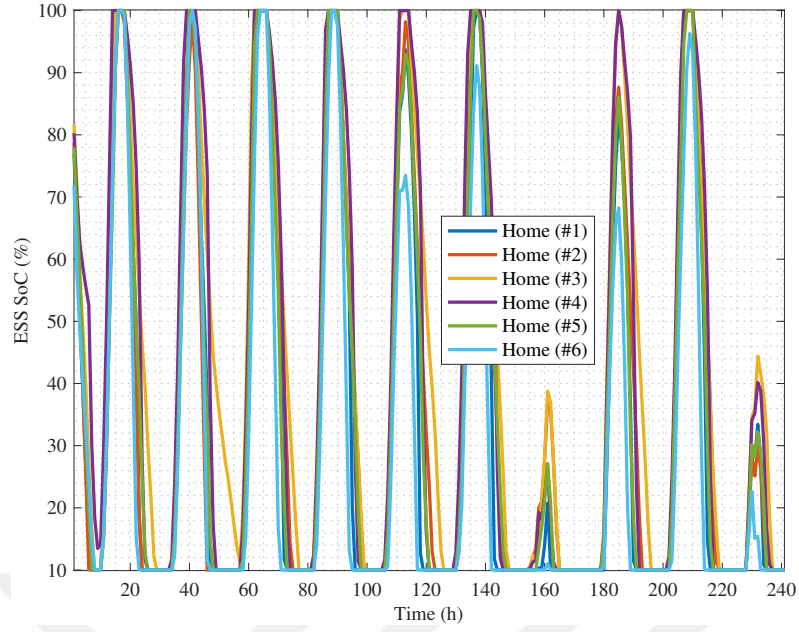


Figure 5.88: SoC of the ESS Battery (Case #3)

The graphics of SoC of ESSs of case #4, where includes neither V2G functionality nor P2P functionality, are as follows.

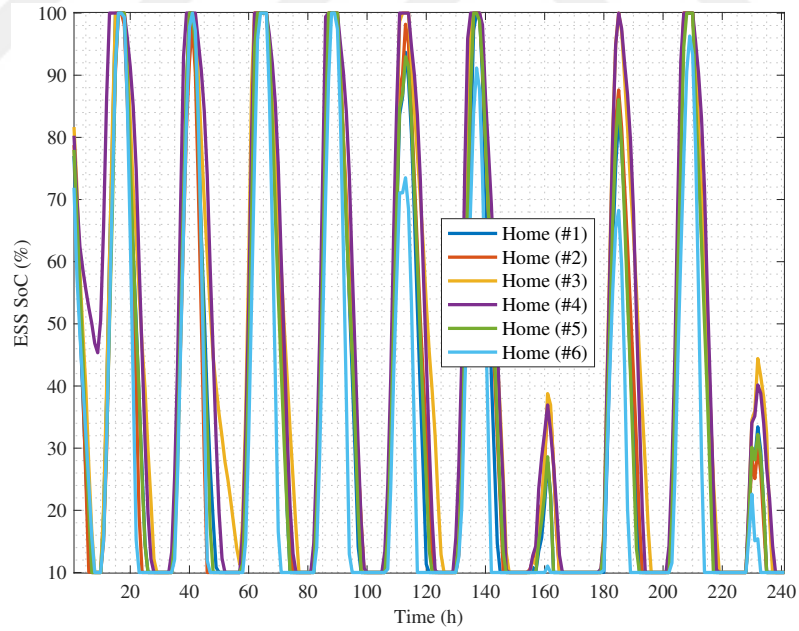


Figure 5.89: SoC of the ESS Battery (Case #4)

Within the case studies, the graphics of SoC of EVs of case #1, where reinforcement learning is applied, are as follows.

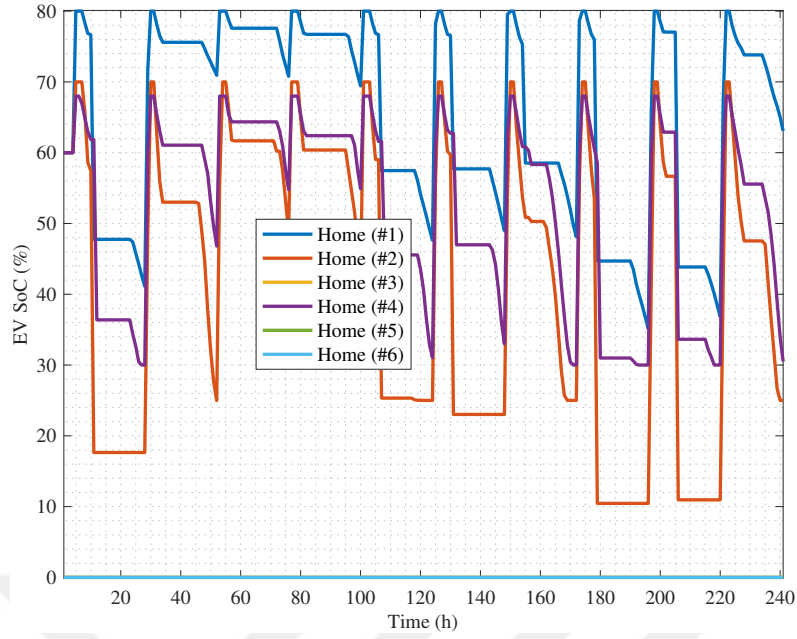


Figure 5.90: SoC of the EV Battery (Case #1)

The graphics of SoC of EVs of case #2, where includes V2G functionality but no P2P functionality, are as follows.

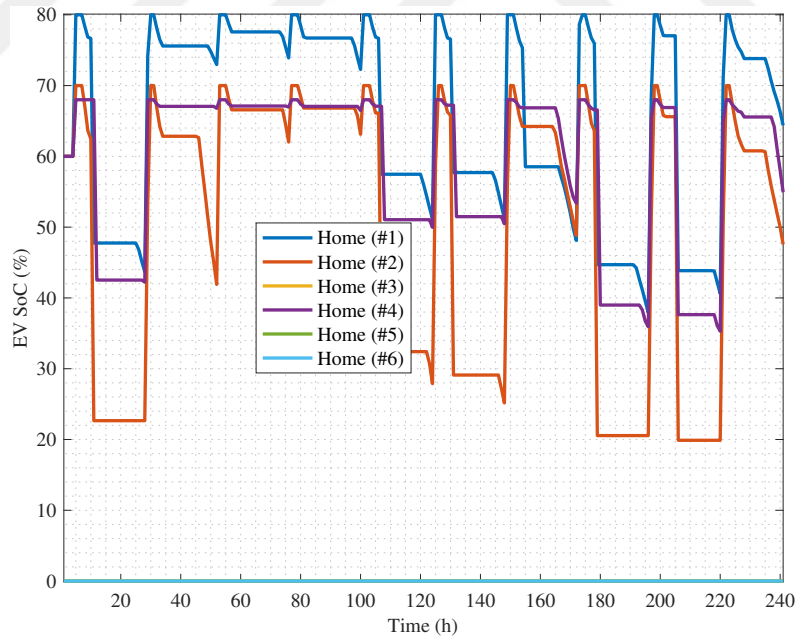


Figure 5.91: SoC of the EV Battery (Case #2)

The graphics of SoC of EVs of case #3, where includes P2P functionality but no V2G functionality, are as follows.

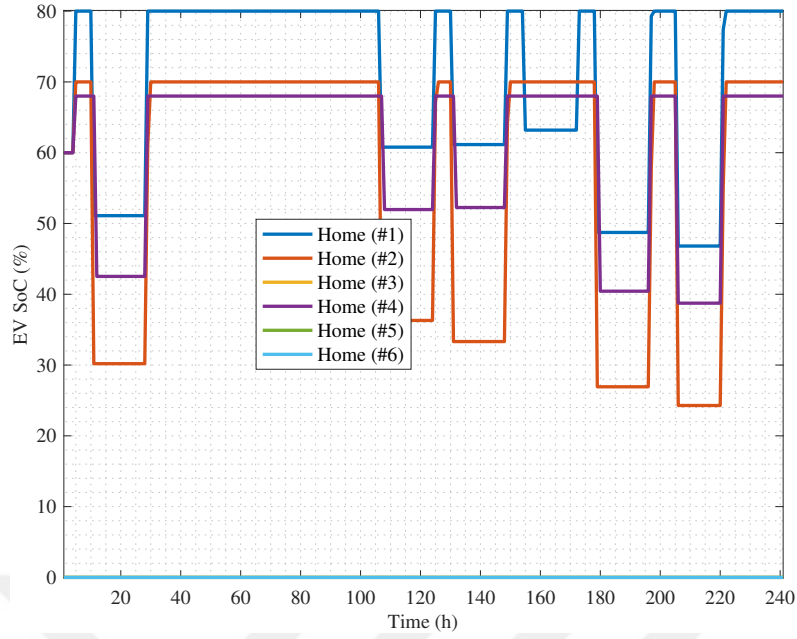


Figure 5.92: SoC of the EV Battery (Case #3)

The graphics of SoC of EVs of case #4, where includes neither V2G functionality nor P2P functionality, are as follows.

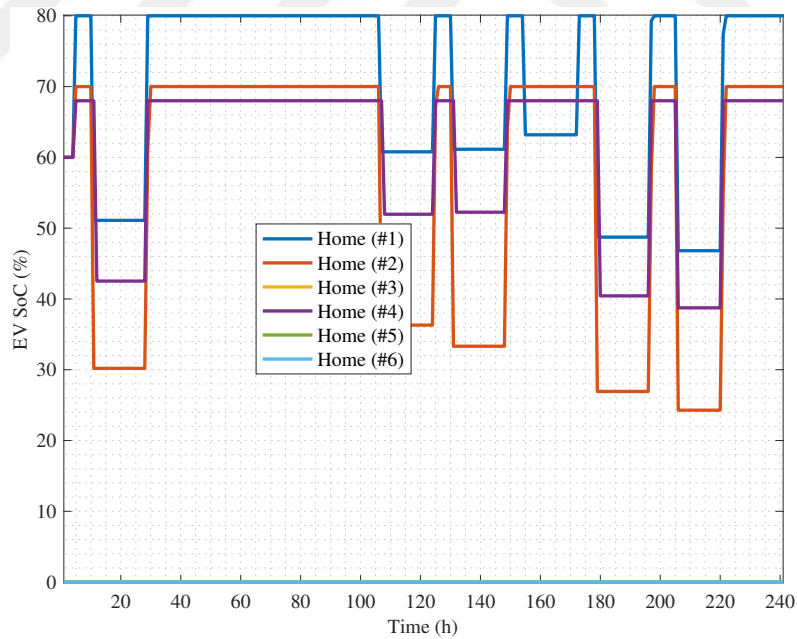


Figure 5.93: SoC of the EV Battery (Case #4)

Throughout the project scope, the energy usage bills to be used in the performance evaluation of the training model are as follows for each case. The negative slope trend of the invoice value shows the loss from energy usage, and the positive slope trend shows the profit from energy sales.

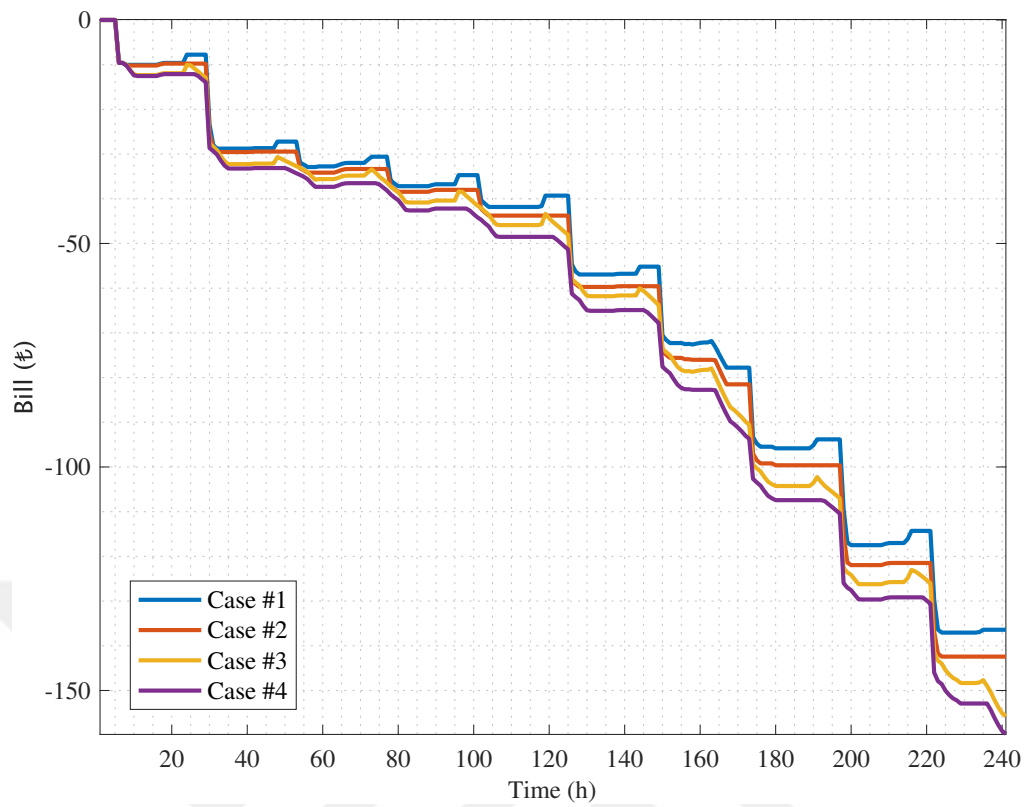


Figure 5.94: Energy Usage Bills of Each Cases (Home #1)

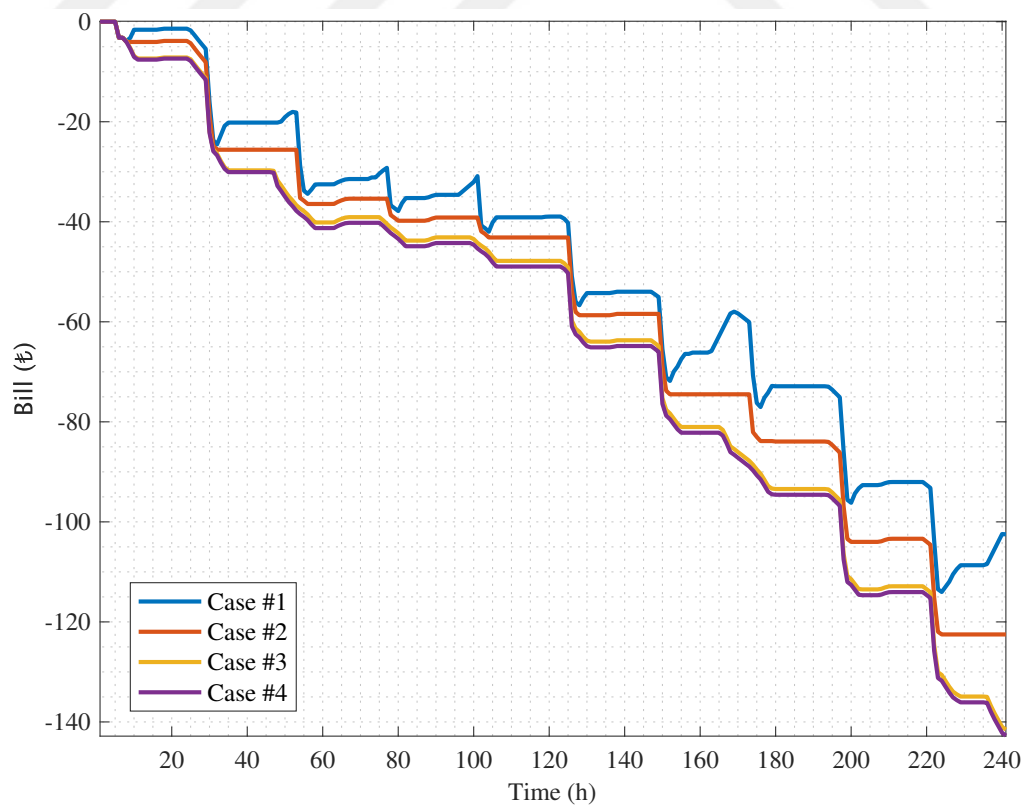


Figure 5.95: Energy Usage Bills of Each Cases (Home #2)

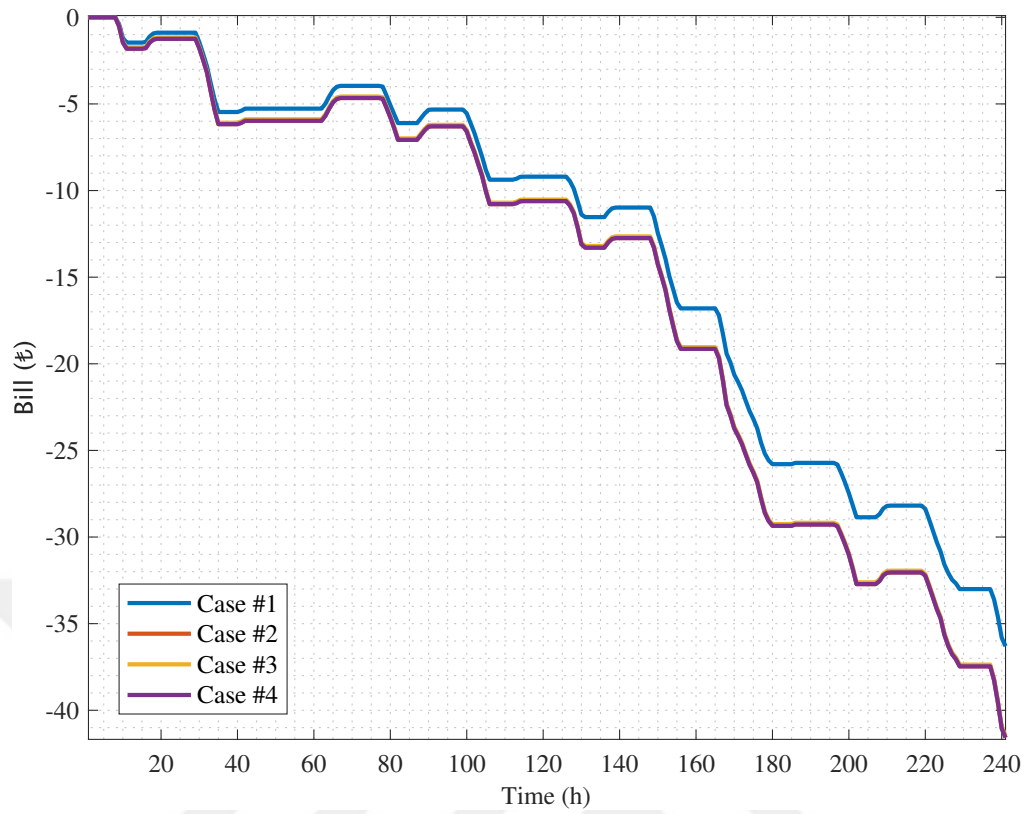


Figure 5.96: Energy Usage Bills of Each Cases (Home #3)

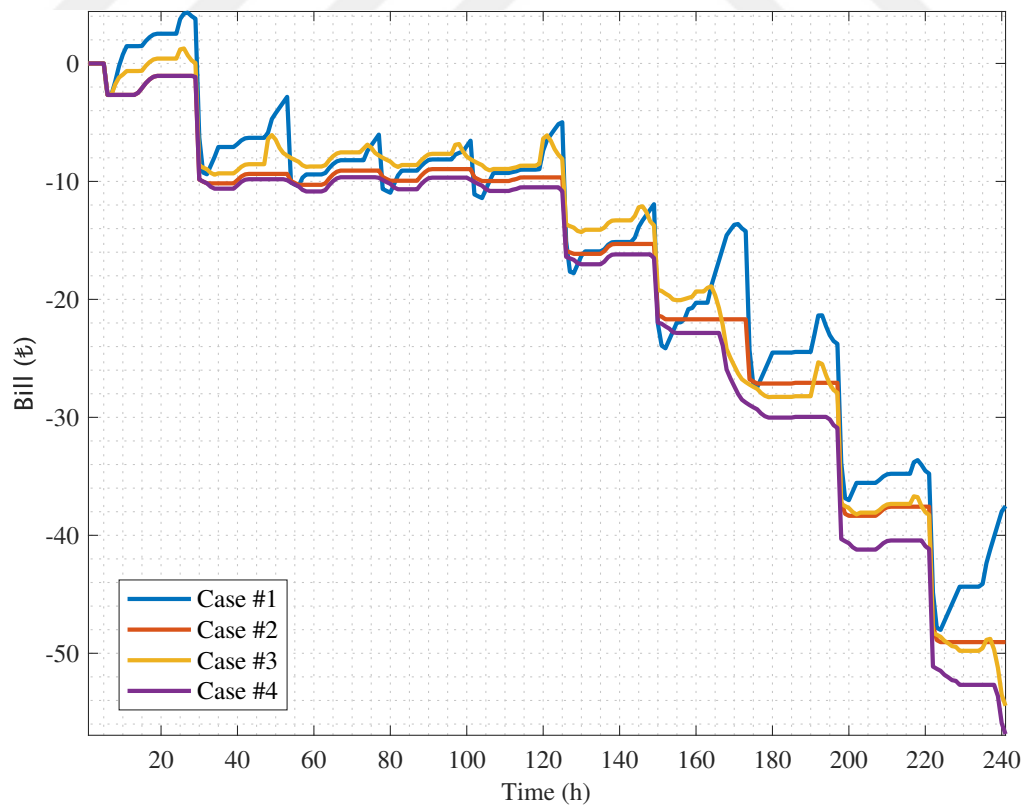


Figure 5.97: Energy Usage Bills of Each Cases (Home #4)

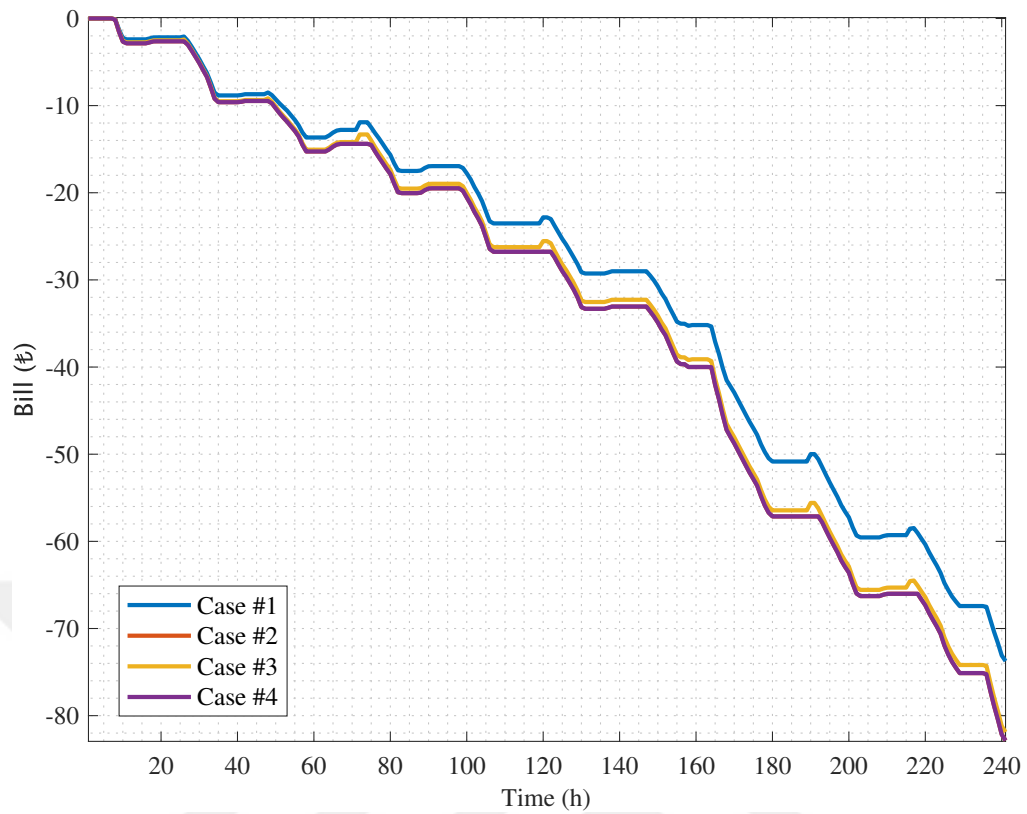


Figure 5.98: Energy Usage Bills of Each Cases (Home #5)

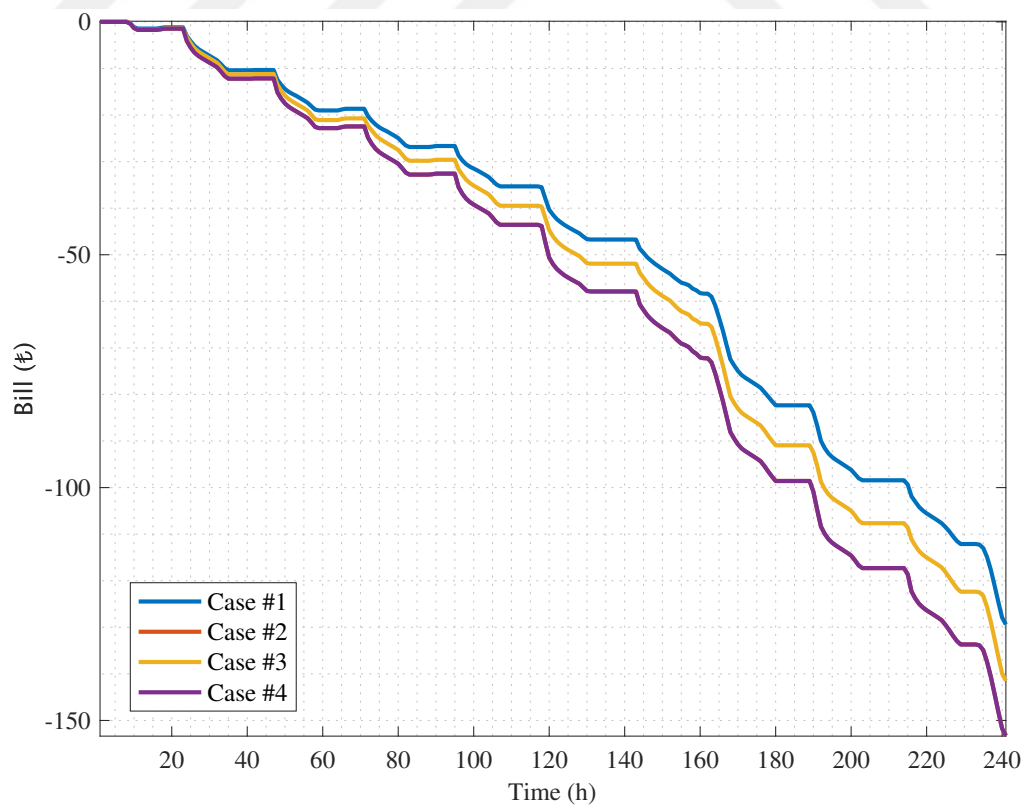


Figure 5.99: Energy Usage Bills of Each Cases (Home #6)

When the energy usage bills for each house are analyzed, in the case where learning takes place with V2G/P2P functionality, the energy bills do not always tend to increase but rather decrease relatively. The main reason for this is that the relatively large batteries of electric vehicles play an active role in the P2P (via V2G) process. The comparison table of energy usage bills for each case and house is as follows.

Table 5.29: Comparison of Total Energy Usage Bills

	Case #1 (TRY)	Case #2 (TRY)	Case #3 (TRY)	Case #4 (TRY)
Home #1	-136,41	-142,42	-156,61	-160,59
Home #2	-103,20	-122,51	-142,26	-143,46
Home #3	-37,12	-42,40	-42,30	-42,40
Home #4	-37,44	-49,05	-55,40	-57,80
Home #5	-74,56	-83,68	-82,71	-83,68
Home #6	-130,54	-154,43	-142,65	-154,43
Total	-519,27	-594,49	-621,93	-642,35

As can be seen in the table above, there has been a 19.16% decrease in the total invoice amount for each house, compared to the 1st and 4th cases. Comparison table of each houses and cases is as follows.

Table 5.30: Comparison of Total Energy Usage Bills by %

	Case #1 vs Case #2 (%)	Case #1 vs Case #3 (%)	Case #1 vs Case #4 (%)
Home #1	4,22	12,90	15,06
Home #2	15,76	27,46	28,06
Home #3	12,46	12,26	12,46
Home #4	23,67	32,42	35,22
Home #5	10,90	9,86	10,90
Home #6	15,47	8,49	15,47
Average	12,65	16,51	19,16

As can be seen in the table above, when all the cases are compared, the 1st case, the case where reinforcement learning was applied and V2G/P2P functionalities were available,

has successfully created a downward trend in energy usage bills against all other cases. The comparative table of energy usage statistics, which is another aim of the project, is as follows.

Table 5.31: Comparison of Total Energy Usage

	Case #1 (kWh)	Case #2 (kWh)	Case #3 (kWh)	Case #4 (kWh)
Battery Charged	214,66	212,39	214,78	212,39
Battery Dis- charged	215,47	235,43	214,35	235,43
Battery Sold	22,23	0,00	23,47	0,00
EV Charged	289,80	218,89	153,52	153,52
EV Discharged	83,30	74,42	0,00	0,00
EV Sold	68,35	0,00	0,00	0,00
Grid Bought	133,15	212,64	284,67	287,06
Grid Sold	58,23	60,50	58,11	60,50
P2P Bought	90,58	0,00	23,47	0,00
P2P Sold	90,58	0,00	23,47	0,00
Total Grid	422,95	431,53	438,19	440,58

There was no significant change in charge/discharge profiles in battery usage between cases. In all cases, the batteries in the central energy storage system showed approximately the same charge/discharge/sold characteristics. On the other hand, in the charging profiles of electric vehicles, the highest charge value is seen in case 1, which has V2G and P2P functionality. The second highest charge value is seen in case 2 where only V2G functionality is available. The main reason for the high use of electric vehicle batteries in Case 1 is the sale of energy to neighbors in the P2P process. During the learning, agents learned to charge electric vehicles in low energy prices and to make a profit by discharging electric vehicles and selling them to neighbors via P2P during high energy prices. At the same time, as can be seen from the grid bought/sold and P2P bought/sold values, the dependency on the main grid has been significantly reduced in case 1. In the amount of energy bought from the main grid, there has been a decrease of 9 kWh for case 1 vs 2, 16 kWh for case 1 vs 3, and 18 kWh for case 1 vs 4, over a 10-day period. As can be seen from the case studies, the V2G and P2P functionalities included in the system reduce the external dependency on the system and provide a significant reduction in energy bills.

CHAPTER 6 CONCLUSION and PROPOSALS

This study aimed to investigate the impact of V2G and P2P energy trading in a cluster including 6 houses with different energy consumption and same solar energy production profile through energy management perspective. Structure of the model was divided into multiple sections and subsections due to simplicity, effectiveness, and efficiency concerns. MATLAB class was created with multiple sub-classes to simulate a cluster behavior with P2P and V2G functionalities. MATLAB software was preferred due to its element-wise matrix operations and C programming language compatibility. Created MATLAB class was implemented in Simulink via built-in "MATLAB Function" blocks afterward. Simulink software was preferred due to its multi-agent environment capability for RL algorithms. To train multiple agents in the cluster environment, DDQN algorithm was used with "Double" flag on to benefit from DDQN algorithm's computational advantages.

Through online available databases, data were collected from different sources to provide diversity in the learning process. Collected data were analyzed with statistical methods to confirm diversity. Furthermore, data were cleaned and then transformed into MATLAB array variables. Unavailable data, such as EVs status, was created via statistical methods in line with the collected data.

Multi-Agent environment was trained in Simulink. Learning process was carried out in 5000 episodes and each episode was carried out in 120 steps. Training results clearly showed that P2P energy trading provides several benefits in terms of reducing energy dependency on the main grid. Combining P2P and V2G capabilities resulted in a significant energy bills and energy dependency reduction. Therefore, benefits arise such as:

- Reducing households energy dependency on the main grid,
- Increasing clean renewable-energy usage across the households,
- Improving energy quality due to lower energy losses on distribution lines,
- Lower maintenance cost on the grid side due to low energy requirement by households,
- Less structural complexity on the distribution lines
- Better energy allocation by energy production, companies to be used in industrial applications with high power requirements.

However, renewable energy production requires high initial cost due to component and installation prices. Initial cost of renewable energy production was not taken into account in this project. Furthermore, degradation in the battery of electric vehicles was not taken into account. However, today's electric vehicle battery technology requires careful computation in V2G due to the relatively high frequency charging/discharging cycles. Fortunately, V2G compatibility in modern EVs have been significantly improved over the last decades with the help of improvements on power electronic technologies.

Future research could explore the impact of multiple distributed renewable energy sources, such as wind turbines. Multiple distributed renewable energy sources might completely eliminate the dependency on the main grid, thus improving energy quality. As mentioned earlier, degradation and battery aging play a significant role in V2G technology. Further investigation is needed to explore the impact of battery aging of EVs in the V2G process. In this way, the benefits of the V2G technology could be further examined. One of the greatest challenges in combining V2G technology and machine learning is uncertainties of electric vehicle absence at any time. Such uncertainty might cause low battery charge issues in case of emergency. Throughout this project, assumptions were made that each agent knew when electric vehicles would be available or not. This assumption provides convenience in the learning process by eliminating the uncertainties. Future research must develop smart strategies to overcome such uncertainties. DDQN algorithm and discrete action space was preferred due to lower computational time and cost despite the fact that might not be the best preference. Future research might be based on continuous action space design and proper learning algorithms such as Deep Deterministic Policy Gradient (DDPG) to make more sensitive computations. Improvements on the CPU and GPU will significantly reduce weights updating time in DNN, thus the learning process could be made with more data to improve decision making in energy management.

REFERENCES

- [1] Bin Zhang, Zhe Chen, and Amer M. Y. M. Ghias. Deep reinforcement learning-based energy management strategy for a microgrid with flexible loads. In *2023 International Conference on Power Energy Systems and Applications (ICoPESA)*, pages 187–191, 2023.
- [2] Aysegül Kahraman and Guangya Yang. Home energy management system based on deep reinforcement learning algorithms. In *2022 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, pages 1–5, 2022.
- [3] Jinsoo Han, Chang-sic Choi, Wan-ki Park, Ilwoo Lee, and Sang-ha Kim. Smart home energy management system including renewable energy based on zigbee and plc. *IEEE Transactions on Consumer Electronics*, 60(2):198–202, 2014.
- [4] Mahmood Anzar, Rafiq Iqra, Anila Kousar, Shafaq Ejaz, Manuel S. Alvarez-Alvarado, and A. Khan Zafar. Optimization of home energy management system in smart grid for effective demand side management. In *2017 International Renewable and Sustainable Energy Conference (IRSEC)*, pages 1–6, 2017.
- [5] Hyunjeong Lee, Wan-Ki Park, and Il-Woo Lee. A home energy management system for energy-efficient smart homes. In *2014 International Conference on Computational Science and Computational Intelligence*, volume 2, pages 142–145, 2014.
- [6] M. K. Perera, K. T. M. U. Hemapala, and W. D. A. S. Wijayapala. Developing a reinforcement learning model for energy management of microgrids in python. In *2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, pages 68–73, 2021.
- [7] ChongAih Hau, Krishnanand Kaippilly Radhakrishnan, JunYen Siu, and Sanjib Kumar Panda. Reinforcement learning based energy management algorithm for energy trading and contingency reserve application in a microgrid. In *2020 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)*, pages 1005–1009, 2020.
- [8] Nanqian Ruan, Yinliang Xu, Wai Kin Chan, and Zixuan Wei. Real-time energy management of a low-carbon micro-grid based on deep reinforcement learning. In *2022 IEEE/IAS Industrial and Commercial Power System Asia (ICPS Asia)*, pages 1102–1107, 2022.
- [9] G Sathish Kumar, Agnes Idhaya Selvi, and Karuppaswamy Pandiyan. Design and analysis for energy storage systems for transport and grid applications using smes devices with pwm-csc. In *2019 IEEE International Conference on Clean Energy and Energy Efficient Electronics Circuit for Sustainable Development (INCCES)*, pages 1–4, 2019.

- [10] Matthew T. Lawder, Bharatkumar Suthar, Paul W. C. Northrop, Sumitava De, C. Michael Hoff, Olivia Leitermann, Mariesa L. Crow, Shriram Santhanagopalan, and Venkat R. Subramanian. Battery energy storage system (bess) and battery management system (bms) for grid-scale applications. *Proceedings of the IEEE*, 102(6):1014–1030, 2014.
- [11] Charbel Bou-Mosleh, Pierre Rahme, Peter Beaino, Rebecca Mattar, and Elie Abi Nassif. Contribution to clean energy production using a novel wave energy converter: Renewable energy. In *International Conference on Renewable Energies for Developing Countries 2014*, pages 108–111, 2014.
- [12] Batuhan Hangun, Onder Eyecioglu, and Murat Beken. Investigating the energy production trends of countries and its relationship between economic development. In *2022 11th International Conference on Renewable Energy Research and Application (ICRERA)*, pages 597–600, 2022.
- [13] L. Kang, H. Guo, J. Wu, and S. Chen. Characteristics of distributed generation system and related research issues caused by connecting it to power system. *Power Sys Technology*, 34:43–47, 11 2010.
- [14] P. Hu, R. Karki, and Roy Billinton. Reliability evaluation of generating systems containing wind power and energy storage. *Generation, Transmission & Distribution, IET*, 3:783 – 791, 09 2009.
- [15] Keno Oltmanns, Jan-Simon Telle, Thomas Steens, Benedikt Hanke, and Karsten von Maydell. Integration and dimensioning of battery storage systems in commercial building applications with renewable powerplants and battery electric vehicles. In *2021 Sixteenth International Conference on Ecological Vehicles and Renewable Energies (EVER)*, pages 1–10, 2021.
- [16] Xiangjun Li and Shangxing Wang. Energy management and operational control methods for grid battery energy storage systems. *CSEE Journal of Power and Energy Systems*, 7(5):1026–1040, 2021.
- [17] Maad Shatnawi, Nasir Al Qaydi, Nawf Aljaberi, and Maitha Aljaberi. Hydrogen-based energy storage systems: A review. In *2018 7th International Conference on Renewable Energy Research and Applications (ICRERA)*, pages 697–700, 2018.
- [18] K. Nigim, J. McQueen, and M. Persohn-Costa. Operational modes of hydrogen energy storage in a micro grid system. In *2015 IEEE Electrical Power and Energy Conference (EPEC)*, pages 473–477, 2015.
- [19] Ziming Wang, Hongyi Lin, Xiao Guo, Zhenyu Li, and Guozhu Chen. Super capacitor energy storage system’s charging design based on composite control mode. In *2021 6th International Conference on Power and Renewable Energy (ICPRE)*, pages 950–954, 2021.
- [20] Xin Wang and Yun Liu. Analysis of energy storage technology and their application for micro grid. In *2017 International Conference on Computer Technology, Electronics and Communication (ICCTEC)*, pages 972–975, 2017.

- [21] Mingming Shi, Juan Hu, Huachun Han, and Xiaodona Yuan. Design of battery energy storage system based on ragone curve. In *2020 4th International Conference on HVDC (HVDC)*, pages 37–40, 2020.
- [22] Felipe O. Ramos, Aline L. Pinheiro, Rafael N. Lima, Martins M. B. Neto, Washigton A.S. Junior, and Libina G. S. Bezerra. A real case analysis of a battery energy storage system for energy time shift, demand management, and reactive control. In *2021 IEEE PES Innovative Smart Grid Technologies Conference - Latin America (ISGT Latin America)*, pages 1–5, 2021.
- [23] Mahammad A. Hannan, Md. Murshadul Hoque, Aini Hussain, Yushaizad Yusof, and Pin Jern Ker. State-of-the-art and energy management system of lithium-ion batteries in electric vehicle applications: Issues and recommendations. *IEEE Access*, 6:19362–19378, 2018.
- [24] Debin Kong, Xianglong Li, Yunbo Zhang, Xiao Hai, Bin Wang, Xiongying Qiu, Qi Song, Quan-Hong Yang, and Linjie Zhi. Encapsulating v2o5 into carbon nanotubes enables the synthesis of flexible high-performance lithium ion batteries. *Energy Environ. Sci.*, 9:906–911, 2016.
- [25] Yong Li, Jian Song, and Jie Yang. A review on structure model and energy system design of lithium-ion battery in renewable energy vehicle. *Renewable and Sustainable Energy Reviews*, 37:627–633, 2014.
- [26] Chunli Gong, Zhigang Xue, Sheng Wen, Yunsheng Ye, and Xiaolin Xie. Advanced carbon materials/olivine lifepo4 composites cathode for lithium ion batteries. *Journal of Power Sources*, 318:93–112, 2016.
- [27] S.A.U.A. Madusanka, D.M.O.R. Mahadiulwewa, S.P.A.A.J. Samarakoon, K.A.H. Sandeepanie, and R.M.T. Damayanthi. Improving the performance of lead acid batteries using nano-technology. In *2019 Moratuwa Engineering Research Conference (MERCon)*, pages 589–593, 2019.
- [28] M. Taguchi, T. Sasaki, and H. Takahashi. Discharge-charge property of lead-acid battery using nano-scale pbo2 as cathode active material. In *Materials Transactions*, volume 55, pages 327–333, 2014.
- [29] Aixiang Zhang, Shizhan Song, Chuanyong Wang, Jian Zhang, Kun Wang, and Liwei Li. Research of battery management system for integrated power supply. In *2017 Chinese Automation Congress (CAC)*, pages 3178–3181, 2017.
- [30] P. Chen, X. Hu, and Q. Shung, G. Application of lithium iron phosphate battery on electric power engineering. In *Applied Mechanics and Materials*, volume 577, pages 560–563, 2014.
- [31] C. Dong, L. Li, W, and H. Zhang, W. Design of new lithium battery management system for electric vehicle. In *Communication Power Technology*, volume 3, pages 33–35, 2012.

- [32] Mark W. Verbrugge. *Adaptive Characterization and Modeling of Electrochemical Energy Storage Devices for Hybrid Electric Vehicle Applications*, pages 417–524. Springer New York, New York, NY, 2009.
- [33] Yunqiu Wang, Lei Li, Quansen Ding, Jiale Liu, and Pengwei Chen. Lithium-ion battery soc estimation based on an improved adaptive extended kalman filter. In *2021 IEEE 16th Conference on Industrial Electronics and Applications (ICIEA)*, pages 417–421, 2021.
- [34] Lei Lin, Masahiro Fukui, and Kyotsugu Takaba. An accurate soc estimation system for lithium-ion batteries by ekf with dynamic noise adjustment. In *2015 15th International Symposium on Communications and Information Technologies (ISCIT)*, pages 33–36, 2015.
- [35] Xiong Yonghua, Yang Yan, He Yong, Wu Min, and An Jianqi. A novel soc estimation method for li-ion batteries based on improved kalman filter with parameter online identification. In *Proceedings of the 31st Chinese Control Conference*, pages 6820–6825, 2012.
- [36] Xiaozhou Zhang and Ruiping Zhang. Estimation of lithium battery soc based on fuzzy unscented kalman filter algorithm. In *2021 IEEE/IAS Industrial and Commercial Power System Asia (I&CPS Asia)*, pages 200–204, 2021.
- [37] Min-Joon Kim, Sung-Hun Chae, and Yeon-Kug Moon. Adaptive battery state-of-charge estimation method for electric vehicle battery management system. In *2020 International SoC Design Conference (ISOCC)*, pages 288–289, 2020.
- [38] Youngbin Song, Minjun Park, Minhwan Seo, and Sang Woo Kim. Improved soc estimation of lithium-ion batteries with novel soc-ocv curve estimation method using equivalent circuit model. In *2019 4th International Conference on Smart and Sustainable Technologies (SpliTech)*, pages 1–6, 2019.
- [39] Ai-Yun Gao, Feng-Li Zhang, Zhu-Mu Fu, Zhi-Chao Zhang, and Hao-Di Li. The soc estimation and simulation of power battery based on self-recurrent wavelet neural network. In *2017 Chinese Automation Congress (CAC)*, pages 4247–4252, 2017.
- [40] Mustafa Ergin Şahin, Frede Blaabjerg, and Ariya Sangwongwanich. Modelling of supercapacitors based on simplified equivalent circuit. *CPSS Transactions on Power Electronics and Applications*, 6(1):31–39, 2021.
- [41] M. R. Palacín and A. de Guibert. Why do batteries fail? *Science*, 351(6273):1253292, 2016.
- [42] Francisco Díaz-González, Andreas Sumper, Oriol Gomis-Bellmunt, and Roberto Villafañila-Robles. A review of energy storage technologies for wind power applications. *Renewable and Sustainable Energy Reviews*, 16(4):2154–2171, 2012.
- [43] A. Rufer and P. Barrade. A supercapacitor-based energy-storage system for elevators with soft commutated interface. *IEEE Transactions on Industry Applications*, 38(5):1151–1159, 2002.

- [44] F. Rafik, H. Gualous, R. Gallay, A. Crausaz, and A. Berthon. Frequency, thermal and voltage supercapacitor characterization and modeling. *Journal of Power Sources*, 165(2):928–934, 2007. IBA – HBC 2006.
- [45] Liuchen Chang, Wenping Zhang, Shuang Xu, and Katelin Spence. Review on distributed energy storage systems for utility applications. *CPSS Transactions on Power Electronics and Applications*, 2(4):267–276, 2017.
- [46] Mamadou Baïlo Camara, Hamid Gualous, Frederic Gustin, and Alain Berthon. Design and new control of dc/dc converters to share energy between supercapacitors and batteries in hybrid vehicles. *IEEE Transactions on Vehicular Technology*, 57(5):2721–2735, 2008.
- [47] Lixin Situ. Electric vehicle development: The past, present & future. In *2009 3rd International Conference on Power Electronics Systems and Applications (PESA)*, pages 1–3, 2009.
- [48] Sesha Gopal Selvakumar. Electric and hybrid vehicles – a comprehensive overview. In *2021 IEEE 2nd International Conference On Electrical Power and Energy Systems (ICEPES)*, pages 1–6, 2021.
- [49] Ikenna C. Eze, Emmanuel Schaffner, Evelyn Fischer, Tamara Schikowski, Martin Adam, Medea Imboden, Ming Tsai, David Carballo, Arnold von Eckardstein, Nino Künzli, Christian Schindler, and Nicole Probst-Hensch. Long-term air pollution exposure and diabetes in a population-based swiss cohort. *Environment International*, 70:95–105, 2014.
- [50] Adriana Tintelecan, Anca Constantinescu Dobra, and Claudia Marțiș. Lca indicators in electric vehicles environmental impact assessment. In *2019 Electric Vehicles International Conference (EV)*, pages 1–5, 2019.
- [51] D. Naunin. Electric vehicles. In *Proceedings of IEEE International Symposium on Industrial Electronics*, volume 1, pages 11–24 vol.1, 1996.
- [52] Sebastian Rivera and Bin Wu. Electric vehicle charging station with an energy storage stage for split-dc bus voltage balancing. *IEEE Transactions on Power Electronics*, 32(3):2376–2386, 2017.
- [53] U.S. DEPARTMENT OF ENERGY. <https://afdc.energy.gov>.
- [54] Aishwarya Panday and Hari Bansal. A review of optimal energy management strategies for hybrid electric vehicle. *International Journal of Vehicular Technology*, 12 2014.
- [55] C. C. Chan. The state of the art of electric, hybrid, and fuel cell vehicles. *Proceedings of the IEEE*, 95(4):704–718, 2007.
- [56] Yasser A Alamoudi, Azzeddine Ferrah, Raghu Panduranga, Ahmed Althobaiti, and F Mulolani. State-of-the art electrical machines for modern electric vehicles. In *2019 Advances in Science and Engineering Technology International Conferences (ASET)*, pages 1–8, 2019.

- [57] Mehrdad Ehsani, Krishna Veer Singh, Hari Om Bansal, and Ramin Tafazzoli Mehrjardi. State of the art and trends in electric and hybrid electric vehicles. *Proceedings of the IEEE*, 109(6):967–984, 2021.
- [58] Alireza Khaligh and Zhihao Li. Battery, ultracapacitor, fuel cell, and hybrid energy storage systems for electric, hybrid electric, fuel cell, and plug-in hybrid electric vehicles: State of the art. *IEEE Transactions on Vehicular Technology*, 59(6):2806–2814, 2010.
- [59] Amin Mahmoudzadeh Andwari, Apostolos Pesiridis, Srithar Rajoo, Ricardo Martinez-Botas, and Vahid Esfahanian. A review of battery electric vehicle technology and readiness levels. *Renewable and Sustainable Energy Reviews*, 78:414–430, 2017.
- [60] Stefan Keller, Klaere Christmann, Miguel Sanchez-Arjona Gonzalez, and Adrian Heuer. A modular fuel cell battery hybrid propulsion system for powering small utility vehicles. In *2017 IEEE Vehicle Power and Propulsion Conference (VPPC)*, pages 1–4, 2017.
- [61] Xiangjun Li, Jianqiu Li, Liangfei Xu, and Minggao Ouyang. Power management and economic estimation of fuel cell hybrid vehicle using fuzzy logic. In *2009 IEEE Vehicle Power and Propulsion Conference*, pages 1749–1754, 2009.
- [62] Seyoung Kim and Sheldon S. Williamson. Modeling, design, and control of a fuel cell/battery/ultra-capacitor electric vehicle energy storage system. In *2010 IEEE Vehicle Power and Propulsion Conference*, pages 1–5, 2010.
- [63] Xian Wu and Haibin Li. The reliability work in fuel cell vehicle’s road test. In *2006 IEEE International Conference on Vehicular Electronics and Safety*, pages 481–484, 2006.
- [64] Boštjan Pregelj, Andrej Debenjak, Gregor Dolanc, and Janko Petrovčič. A diesel-powered fuel cell apu—reliability issues and mitigation approaches. *IEEE Transactions on Industrial Electronics*, 64(8):6660–6670, 2017.
- [65] Shinae Lee, Dao Zhou, and Huai Wang. Reliability assessment of fuel cell system - a framework for quantitative approach. In *2016 IEEE Energy Conversion Congress and Exposition (ECCE)*, pages 1–5, 2016.
- [66] C. J. Ifedi, B. C. Mecrow, J. D. Widmer, G. J. Atkinson, S. T. M. Brockway, and D. Kostic-Perovic. A high torque density, direct drive in-wheel motor for electric vehicles. In *6th IET International Conference on Power Electronics, Machines and Drives (PEMD 2012)*, pages 1–6, 2012.
- [67] K.M. Rahman, N.R. Patel, T.G. Ward, J.M. Nagashima, F. Caricchi, and F. Crescimbeni. Application of direct-drive wheel motor for fuel cell electric and hybrid electric vehicle propulsion system. *IEEE Transactions on Industry Applications*, 42(5):1185–1192, 2006.

- [68] Chunhua Liu, K. T. Chau, and J. Z. Jiang. A permanent-magnet hybrid in-wheel motor drive for electric vehicles. In *2008 IEEE Vehicle Power and Propulsion Conference*, pages 1–6, 2008.
- [69] Yee-Pien Yang, Jia-Yuan Liang, and Xian-Yee Xing. Design and application of axial-flux permanent magnet wheel motors for an electric vehicle. In *AFRICON 2009*, pages 1–5, 2009.
- [70] Deepak Mohanraj, Janaki Gopalakrishnan, Bharatiraja Chokkalingam, and Lucian Mihet-Popa. Critical aspects of electric motor drive controllers and mitigation of torque ripple—review. *IEEE Access*, 10:73635–73674, 2022.
- [71] Rahul Charles C M and Savier J S. Bidirectional dc-dc converter fed bldc motor in electric vehicle. In *2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, pages 1–6, 2021.
- [72] B.V. Ravi Kumar and K. Siva Kumar. Design of a new dual rotor radial flux bldc motor with halbach array magnets for an electric vehicle. In *2016 IEEE International Conference on Power Electronics, Drives and Energy Systems (PEDES)*, pages 1–5, 2016.
- [73] Adil Usman and Bharat Singh Rajpurohit. Time-efficient fault diagnosis of a bldc motor drive deployed in electric vehicle applications. In *2020 IEEE Global Humanitarian Technology Conference (GHTC)*, pages 1–5, 2020.
- [74] Anusha Vadde and S Sachin. Influence of rotor design in bldc motor for two - wheeler electric vehicle. In *2021 1st International Conference on Power Electronics and Energy (ICPEE)*, pages 1–6, 2021.
- [75] Rajnish Kumar, Prabhakar Kumar, Balmukund Kumar, Parshuram Singh, Rahul Kumar, and Amitesh Kumar. Design and analysis of high performance of a bldc motor for electric vehicle. In *2022 2nd International Conference on Emerging Frontiers in Electrical and Electronic Technologies (ICEFEET)*, pages 1–4, 2022.
- [76] Agus Mujianto, Muh. Nizam, and Inayati. Comparison of the slotless brushless dc motor (bldc) and slotted bldc using 2d modeling. In *2014 International Conference on Electrical Engineering and Computer Science (ICEECS)*, pages 212–214, 2014.
- [77] Cunshan Zhang and Dunxin Bian. A pwm control algorithm for eliminating torque ripple caused by stator magnetic field jump of brushless dc motors. In *2008 7th World Congress on Intelligent Control and Automation*, pages 6547–6549, 2008.
- [78] C.L. Chu, M.C. Tsai, and H.Y. Chen. Torque control of brushless dc motors applied to electric vehicles. In *IEMDC 2001. IEEE International Electric Machines and Drives Conference (Cat. No.01EX485)*, pages 82–87, 2001.
- [79] Zicheng Li and Shanmei Cheng. Torque ripple reduction in brushless dc motors based on model predictive control. In *2010 International Conference on Electrical and Control Engineering*, pages 4498–4501, 2010.

- [80] Qihang Sun, Zhenguo Li, Penglei Wang, and Xiaolei Duan. Direct torque control technology of brushless dc motor based on new four-level inverter topology. In *2020 7th International Conference on Information Science and Control Engineering (ICISCE)*, pages 1845–1849, 2020.
- [81] Sirichai Dangeam. Three-phase inverter with direct torque control for brushless dc motor. In *2021 18th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pages 543–546, 2021.
- [82] Tang Puhua, Lei Yuyong, Liu Kefu, and Jiang Daijun. New direct torque control of brushless dc motor. In *2010 International Conference on Electrical and Control Engineering*, pages 3902–3907, 2010.
- [83] M. Rivandi Fadli, M. Mushthofa Musyasy, Jihad Furqani, and Agus Purwadi. Modelling of field orientation control (foc) method in 120 kw brushless dc motor (bl dc). In *2019 6th International Conference on Electric Vehicular Technology (ICEVT)*, pages 383–389, 2019.
- [84] B. Pavan Kumar and Krishnan C.M.C. Comparative study of different control algorithms on brushless dc motors. In *2016 Biennial International Conference on Power and Energy Systems: Towards Sustainable Energy (PESTSE)*, pages 1–5, 2016.
- [85] Sheshadri Shekhar Rauth and Banshidhari Samanta. Comparative analysis of im/bl dc/pmsm drives for electric vehicle traction applications using ann-based foc. In *2020 IEEE 17th India Council International Conference (INDICON)*, pages 1–8, 2020.
- [86] M. Y. Chernenko, E. S. Kucher, and E. Y. Kamysheva. High-speed passenger lift model development. In *2018 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon)*, pages 1–4, 2018.
- [87] Vaibhav W. Nage and Sanjay M. Shinde. An elevator driven by single-sided linear induction motor (slim). In *2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)*, pages 376–379, 2016.
- [88] G. A. Fedyaeva, T. V. Smorodova, and R. V. Kovalev. Electromechanical system of the asynchronous electric drive of movement of a bridge crane modeling. In *2016 2nd International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*, pages 1–4, 2016.
- [89] Vladimir L. Kodkin and Alexandr S. Anikin. The reflection mode of the moment loads by an asynchronous electric drive with vector control and the features of this mode in the rotation drive of the tower crane. In *2020 International Ural Conference on Electrical Power Engineering (UralCon)*, pages 416–420, 2020.
- [90] O. A. Lysenko. The study of dynamic characteristics of an electric drive of centrifugal pump with taking into account tooth harmonics of induction motor. In *2016*

2nd International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), pages 1–4, 2016.

- [91] K.V Mahesh, Sachin Angadi, and A. B. Raju. Single stage pv water pumping system with mppt employing three phase induction motor. In *2018 International Conference on Current Trends towards Converging Technologies (ICCTCT)*, pages 1–5, 2018.
- [92] R.N. Tuncay, M. Yilmaz, and C. Onculoglu. The design methodology to develop new-generation universal-motors for vacuum cleaners. In *IEMDC 2001. IEEE International Electric Machines and Drives Conference (Cat. No.01EX485)*, pages 926–930, 2001.
- [93] Mounir Zeraoulia, Mohamed El Hachemi Benbouzid, and Demba Diallo. Electric motor drive selection issues for hev propulsion systems: A comparative study. *IEEE Transactions on Vehicular Technology*, 55(6):1756–1764, 2006.
- [94] B. Tabbache, A. Kheloui, and M.E.H. Benbouzid. Design and control of the induction motor propulsion of an electric vehicle. In *2010 IEEE Vehicle Power and Propulsion Conference*, pages 1–6, 2010.
- [95] Kentaro Suzuki, Suzuo Saito, Toshiaki Kudor, Atsushi Tanaka, and Yasuhiro Andoh. Stability improvement of v/f controlled large capacity voltage-source inverter fed induction motor. In *Conference Record of the 2006 IEEE Industry Applications Conference Forty-First IAS Annual Meeting*, volume 1, pages 90–95, 2006.
- [96] Aung Zaw Latt and Ni Ni Win. Variable speed drive of single phase induction motor using frequency control method. In *2009 International Conference on Education Technology and Computer*, pages 30–34, 2009.
- [97] Hayate Matsumoto and Kazuto Sakai. A motor capable of conversion between synchronous and induction motors with pole change for electric vehicles. In *2022 IEEE Energy Conversion Congress and Exposition (ECCE)*, pages 1–8, 2022.
- [98] Sita Ram, O.P. Rahi, Veena Sharma, and Ajay Kumar. Performance analysis of slip power recovery scheme employing two inverter topologies. In *Michael Faraday IET International Summit 2015*, pages 356–361, 2015.
- [99] Richa Pandey, Anup Kumar Panda, and Nishant Patnaik. Comparative performance analysis of dtc fed three-phase and five-phase induction motor. In *2020 5th International Conference on Devices, Circuits and Systems (ICDCS)*, pages 162–166, 2020.
- [100] Tae Uk Jung, Sung Ho Lee, Sung Jun Park, Cheol Ho Yun, and Yu Tao. Blde motor drive system of air-condition of hybrid electric vehicle. In *2007 International Conference on Electrical Machines and Systems (ICEMS)*, pages 636–641, 2007.
- [101] Vlad Grădinaru, Lucian Tutelea, and Ion Boldea. Hybrid analytical/fem optimization design of spmsm for refrigerator compressor loads. In *International Aegean Conference on Electrical Machines and Power Electronics and Electromotion, Joint Conference*, pages 657–662, 2011.

- [102] Andrei Oprea and Dan Florica. Efficient topology of powering a washing machine using a permanent magnet synchronous motor. In *2021 International Conference on Applied and Theoretical Electricity (ICATE)*, pages 1–4, 2021.
- [103] D. Iles-Klumpner, I. Serban, and M. Risticevic. Pmsm drive systems for electric active steering (eafs) application: a comparative characterization. In *2006 IEEE Vehicle Power and Propulsion Conference*, pages 1–6, 2006.
- [104] Hui Chen, Canlong Jin, Ping Jiang, Xiaoping Gong, and Xiangzhi Feng. Pmsm servo drive system for electric power steering based on two-degree-of-freedom torque control. In *2006 IEEE International Conference on Industrial Technology*, pages 2901–2906, 2006.
- [105] Raul-Octavian Nemes, Mircea Ruba, Sorina Ciornei, Horia Hedesiu, Claudia Martis, and Calin Husar. Real-time co-simulation of electric power steering system. In *2018 International Conference and Exposition on Electrical And Power Engineering (EPE)*, pages 0103–0106, 2018.
- [106] Pier Carlo Berri, Matteo Davide Lorenzo Dalla Vedova, Paolo Maggiore, and Matteo Scanavino. Permanent magnet synchronous motor (pmsm) for aerospace servomechanisms: Proposal of a lumped model for prognostics. In *2018 2nd European Conference on Electrical Engineering and Computer Science (EECS)*, pages 471–477, 2018.
- [107] Jinquan Xu, Boyi Zhang, Hao Fang, and Hong Guo. Guaranteeing the fault transient performance of aerospace multiphase permanent magnet motor system: An adaptive robust speed control approach. *CES Transactions on Electrical Machines and Systems*, 4(2):114–122, 2020.
- [108] R.G. Shriwastava, D.R. Bhise, and Pravin Nagrale. Comparative analysis of foc based three level dcml driven pmsm drive. In *2019 International Conference on Innovative Trends and Advances in Engineering and Technology (ICITAET)*, pages 26–31, 2019.
- [109] Li Yu, Chunyang Wang, Hongwei Shi, Ruihao Xin, and Lingxin Wang. Simulation of pmsm field-oriented control based on svpwm. In *2017 29th Chinese Control And Decision Conference (CCDC)*, pages 7407–7411, 2017.
- [110] Hesna Aberkane, Djamel Sakri, and Djamel Rahem. Improvement of direct torque control performances using fcs-mpc and svm applied to pmsm: Study and comparison. In *2018 International Conference on Electrical Sciences and Technologies in Maghreb (CISTEM)*, pages 1–6, 2018.
- [111] Wenming Wu, Fang Xie, Guoli Li, Kangkang Liang, Chenming Qiu, and Hong Jiang. Research on direct torque control based on rzvsvpwm of pmsm. In *2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pages 2507–2511, 2019.

- [112] Yifang Zhu, Gang Xu, Jian Yin, and Yuan Liu. Speed control of permanent magnet synchronous motor drives based on model predictive control. In *2018 IEEE 3rd International Conference on Cloud Computing and Internet of Things (CCIoT)*, pages 88–93, 2018.
- [113] Changliang Dang, Manfeng Dou, and Yuanlin Wang. Model predictive direct torque control for pmsm drives in m–t frame. In *2021 IEEE International Conference on Predictive Control of Electrical Drives and Power Electronics (PRECEDE)*, pages 138–141, 2021.
- [114] P.Niveditha Dheeraja and E.Shiva Prasad. Grid interfaced single stage solar water pump using srm. In *2021 6th International Conference on Communication and Electronics Systems (ICCES)*, pages 131–137, 2021.
- [115] M. Asgar, E. Afjei, A. Behbahani, and A. Siadatan. A 12/8 double-stator switched reluctance motor for washing machine application. In *The 6th Power Electronics, Drive Systems & Technologies Conference (PEDSTC2015)*, pages 168–172, 2015.
- [116] Erdal Şehirli, Feriha Erfan Kuyumcu, and Bekir Çakır. Current controlled srm fed by three-phase voltage source converter. In *2015 50th International Universities Power Engineering Conference (UPEC)*, pages 1–6, 2015.
- [117] Dong-Hee Lee, Jong Heon Lee, and Jin-Woo Ahn. Current control of a high speed srm with an advanced 4-level converter. In *8th International Conference on Power Electronics - ECCE Asia*, pages 109–114, 2011.
- [118] Jiangyong Liu, Lixiong Wang, Lingzhi Yi, Guanghui Zhu, and Xinjian Yin. Optimization of srm direct instantaneous torque control strategy based on improved firefly algorithm. In *2019 IEEE 3rd Conference on Energy Internet and Energy System Integration (EI2)*, pages 364–368, 2019.
- [119] Reza Shahbazi, Sayed Morteza Saghaiannezhad, and Amir Rashidi. A new converter based on ditc for improving torque ripple and power factor in srm drives. In *2020 11th Power Electronics, Drive Systems, and Technologies Conference (PEDSTC)*, pages 1–5, 2020.
- [120] Kentaro Furiya, Naoki Omura, Hideki Nagano, Kenji Nishimura, Akio Ueda, and Susumu Tokura. Parallel-connected bilayer coil for a 3.3-kw electric vehicle wireless charging system. In *2017 IEEE Wireless Power Transfer Conference (WPTC)*, pages 1–4, 2017.
- [121] B. Han. *Electric Vehicle Energy Management Considering Stakeholders' Interest in Smart Grids*. University of Liverpool, 2019.
- [122] Developing infrastructure to charge electric vehicles. https://afdc.energy.gov/fuels/electricity_infrastructure.html. Accessed: 2023-04-20.
- [123] Rwamurangwa Evode. Modeling of electric grid behaviors having electric vehicle charging stations with g2v and v2g possibilities. In *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICEC-CME)*, pages 1–5, 2021.

- [124] Reza Hemmati, Hasan Mehrjerdi, Nasser A Al-Emadi, and Elyas Rakhshani. Mutual vehicle-to-home and vehicle-to-grid operation considering solar-load uncertainty. In *2019 2nd International Conference on Smart Grid and Renewable Energy (SGRE)*, pages 1–4, 2019.
- [125] Hasan Mehrjerdi and Elyas Rakhshani. Vehicle-to-grid technology for cost reduction and uncertainty management integrated with solar power. *Journal of Cleaner Production*, 229:463–469, 2019.
- [126] Chunhua Liu, K. T. Chau, Diyun Wu, and Shuang Gao. Opportunities and challenges of vehicle-to-home, vehicle-to-vehicle, and vehicle-to-grid technologies. *Proceedings of the IEEE*, 101(11):2409–2427, 2013.
- [127] Timo Lehtola and Ahmad Zahedi. Vehicle to grid system in frequency regulation for securing electricity network stability. In *2015 IEEE PES Asia-Pacific Power and Energy Engineering Conference (APPEEC)*, pages 1–4, 2015.
- [128] Bhumika Shrimali, Jai Kumar Maherchandani, and Abrar Ahmed Chhipa. Vehicle to grid system integration for frequency regulation of renewable based microgrid. In *2021 International Conference on Sustainable Energy and Future Electric Transportation (SEFET)*, pages 1–6, 2021.
- [129] Wayes Tushar, Tapan Kumar Saha, Chau Yuen, David Smith, and H. Vincent Poor. Peer-to-peer trading in electricity networks: An overview. *IEEE Transactions on Smart Grid*, 11(4):3185–3200, 2020.
- [130] Wayes Tushar, Tapan Kumar Saha, Chau Yuen, Thomas Morstyn, Nahid-Al-Masood, H. Vincent Poor, and Richard Bean. Grid influenced peer-to-peer energy trading. *IEEE Transactions on Smart Grid*, 11(2):1407–1418, 2020.
- [131] Zejia Jing, Manisa Pipattanasomporn, and Saifur Rahman. Blockchain-based ne-gawatt trading platform: Conceptual architecture and case studies. In *2019 IEEE PES GTD Grand International Conference and Exposition Asia (GTD Asia)*, pages 68–73, 2019.
- [132] Morstyn T., Farrell N., and Darby S. Using peer-to-peer energy-trading platforms to incentivize prosumers to form federated power plants. In *Nat Energy*, volume 3, pages 94–101, 2018.
- [133] Esther Mengelkamp, Johannes Gärttner, Kerstin Rock, Scott Kessler, Lawrence Orsini, and Christof Weinhardt. Designing microgrid energy markets: A case study: The brooklyn microgrid. *Applied Energy*, 210:870–880, 2018.
- [134] Chenghua Zhang, Jianzhong Wu, Yue Zhou, Meng Cheng, and Chao Long. Peer-to-peer energy trading in a microgrid. *Applied Energy*, 220:1–12, 2018.
- [135] R. Schollmeier. A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. In *Proceedings First International Conference on Peer-to-Peer Computing*, pages 101–102, 2001.

- [136] Wayes Tushar, Chau Yuen, Hamed Mohsenian-Rad, Tapan Saha, H. Vincent Poor, and Kristin L. Wood. Transforming energy networks via peer-to-peer energy trading: The potential of game-theoretic approaches. *IEEE Signal Processing Magazine*, 35(4):90–111, 2018.
- [137] Jogunola O., Ikpehai A., Adebisi B., Hammoudeh M., Gacanin H., and Harris G. Comparative analysis of p2p architectures for energy trading and sharing. In *Energies*, volume 11, pages 1–20, 2018.
- [138] Fangyuan Si, Jinkuan Wang, Yinghua Han, Qiang Zhao, Peng Han, and Yan Li. Cost-efficient multi-energy management with flexible complementarity strategy for energy internet. *Applied Energy*, 231:803–815, 2018.
- [139] R. Alvaro-Hermana, Jesus Fraile-Ardanuy, Pedro J. Zufiria, Luk Knapen, and Davy Janssens. Peer to peer energy trading with electric vehicles. *IEEE Intelligent Transportation Systems Magazine*, 8(3):33–44, 2016.
- [140] Hannes Kirchhoff and Kai Strunz. Key drivers for successful development of peer-to-peer microgrids for swarm electrification. *Applied Energy*, 244:46–62, 2019.
- [141] Kaixuan Chen, Jin Lin, and Yonghua Song. Trading strategy optimization for a prosumer in continuous double auction-based peer-to-peer market: A prediction-integration model. *Applied Energy*, 242:1121–1133, 2019.
- [142] Wayes Tushar, Chau Yuen, David B. Smith, and H. Vincent Poor. Price discrimination for energy trading in smart grid: A game theoretic approach. *IEEE Transactions on Smart Grid*, 8(4):1790–1801, 2017.
- [143] Umit Cali and Ozan Çakir. Energy policy instruments for distributed ledger technology empowered peer-to-peer local energy markets. *IEEE Access*, 7:82888–82900, 2019.
- [144] Kelvin Anoh, Dragana Bajovic, Dejan Vukobratovic, Bamidele Adebisi, Dusan Jakovetic, and Mirsad Cosovic. Distributed energy trading with communication constraints. In *2018 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, pages 1–6, 2018.
- [145] Weigang Hou, Lei Guo, and Zhaolong Ning. Local electricity storage for blockchain-based energy trading in industrial internet of things. *IEEE Transactions on Industrial Informatics*, 15(6):3610–3619, 2019.
- [146] Zhetao Li, Jiawen Kang, Rong Yu, Dongdong Ye, Qingyong Deng, and Yan Zhang. Consortium blockchain for secure energy trading in industrial internet of things. *IEEE Transactions on Industrial Informatics*, 14(8):3690–3700, 2018.
- [147] Hamada Almasalma, Sander Claeys, and Geert Deconinck. Peer-to-peer-based integrated grid voltage support function for smart photovoltaic inverters. *Applied Energy*, 239:1037–1048, 2019.

- [148] Yinliang Xu, Hongbin Sun, and Wei Gu. A novel discounted min-consensus algorithm for optimal electrical power trading in grid-connected dc microgrids. *IEEE Transactions on Industrial Electronics*, 66(11):8474–8484, 2019.
- [149] Huajie Gu, Ruifeng Yan, and Tapan Saha. Review of system strength and inertia requirements for the national electricity market of australia. *CSEE Journal of Power and Energy Systems*, 5(3):295–305, 2019.
- [150] Woongsup Lee, Lin Xiang, Robert Schober, and Vincent W. S. Wong. Direct electricity trading in smart grid: A coalitional game analysis. *IEEE Journal on Selected Areas in Communications*, 32(7):1398–1411, 2014.
- [151] Amrit Paudel, Kalpesh Chaudhari, Chao Long, and Hoay Beng Gooi. Peer-to-peer energy trading in a prosumer-based community microgrid: A game-theoretic model. *IEEE Transactions on Industrial Electronics*, 66(8):6087–6097, 2019.
- [152] Amir Anees, Tharam Dillon, and Yi-Ping Phoebe Chen. A novel decision strategy for a bilateral energy contract. *Applied Energy*, 253:113571, 2019.
- [153] Weijia Liu, Donglian Qi, and Fushuan Wen. Intraday residential demand response scheme based on peer-to-peer energy trading. *IEEE Transactions on Industrial Informatics*, 16(3):1823–1835, 2020.
- [154] Fengji Luo, Zhao Yang Dong, Gaoqi Liang, Junichi Murata, and Zhao Xu. A distributed electricity trading system in active distribution networks based on multi-agent coalition and blockchain. *IEEE Transactions on Power Systems*, 34(5):4097–4108, 2019.
- [155] Sana Noor, Wentao Yang, Miao Guo, Koen H. van Dam, and Xiaonan Wang. Energy demand side management within micro-grid networks enhanced by blockchain. *Applied Energy*, 228:1385–1398, 2018.
- [156] Kunal Kashyap, Lalit, Daksh Shah, and Lokesh Gautam. From classical to quantum: A review of recent progress in reinforcement learning. In *2021 2nd International Conference for Emerging Technology (INCET)*, pages 1–5, 2021.
- [157] Montague P., R. Reinforcement learning: An introduction, by sutton, r.s. and barto, a.g. In *Trends in Cognitive Sciences*, volume 3, page 360, 1999.
- [158] Jingkai Jia and Wenlin Wang. Review of reinforcement learning research. In *2020 35th Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pages 186–191, 2020.
- [159] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229, 1959.
- [160] Christopher JCH Watkins and Peter Dayan. Technical note: Q-learning. *Machine learning*, 8:279–292, 1992.
- [161] Jinpei Cui. Resource optimization research of wireless multimedia service transmission under the mdp framework. In *Xidian University*, pages 20–22, 2017.

- [162] Su Meiling. Research on streaming media transmission in wireless networks based on markov decision process theory. In *West China Normal University*, pages 22–25, 2017.
- [163] Somnath Banerjee. Real world applications of markov decision process, Jan 9, 2021. <https://towardsdatascience.com/real-world-applications-of-markov-decision-process-mdp-a39685546026>.
- [164] Holger H. Hoos. *Automated Algorithm Configuration and Parameter Tuning*, pages 37–71. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.
- [165] Changwu Huang, Yuanxiang Li, and Xin Yao. A survey of automatic parameter tuning methods for metaheuristics. *IEEE Transactions on Evolutionary Computation*, 24(2):201–216, 2020.
- [166] Shengcai Liu, Ke Tang, Yunwei Lei, and Xin Yao. On performance estimation in automatic algorithm configuration. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(03):2384–2391, Apr. 2020.
- [167] Michel Tokic. Adaptive ϵ -greedy exploration in reinforcement learning based on value differences. In Rüdiger Dillmann, Jürgen Beyerer, Uwe D. Hanebeck, and Tanja Schultz, editors, *KI 2010: Advances in Artificial Intelligence*, pages 203–210, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.
- [168] Christopher Painter-Wakefield and Ronald Parr. Greedy algorithms for sparse reinforcement learning, 2012.
- [169] Nguyen Duc Toan and Kim Gon-Woo. Environment exploration for mapless navigation based on deep reinforcement learning. In *2021 21st International Conference on Control, Automation and Systems (ICCAS)*, pages 17–20, 2021.
- [170] Qingyan Huang. Model-based or model-free, a review of approaches in reinforcement learning. In *2020 International Conference on Computing and Data Science (CDS)*, pages 219–221, 2020.
- [171] Marcin Hoffmann and Pawel Kryszkiewicz. Radio environment map and deep q-learning for 5g dynamic point blanking. In *2022 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. IEEE, sep 2022.
- [172] Chaymae Chouiekh, Ali Yahyaouy, Abdellah Aarab, and Abdelouahed Sabri. Road traffic: Deep q-learning agent control traffic lights in the intersection. In *2022 International Conference on Intelligent Systems and Computer Vision (ISCV)*, pages 1–5, 2022.
- [173] Godwyll Aikins, Sagar Jagtap, and Weinan Gao. Resilience analysis of deep q-learning algorithms in driving simulations against cyberattacks. In *2022 1st International Conference on AI in Cybersecurity (ICAIC)*, pages 1–6, 2022.
- [174] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization, 2017.

- [175] Liangliang Chen, Fei Meng, and Ying Zhang. Mbrl-mc: An hvac control approach via combining model-based deep reinforcement learning and model predictive control. *IEEE Internet of Things Journal*, 9(19):19160–19173, 2022.
- [176] Jong In Park and Kae Won Choi. Model-based reinforcement learning for wireless channel access. In *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*, pages 1393–1395, 2022.
- [177] Meng Xiao, Jianbing Xiahou, and Meng Ge. A reinforcement-learning-based bus scheduling model. In *2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, volume 10, pages 923–927, 2022.
- [178] Nobuhiko Yamaguchi, Osamu Fukuda, and Hiroshi Okumura. Model-based reinforcement learning with missing data. In *2020 Eighth International Symposium on Computing and Networking Workshops (CANDARW)*, pages 168–171, 2020.
- [179] Jie Fang, Yunqing Rao, Wei Ding, and Ronghua Meng. Research on two-dimensional intelligent nesting based on sarsa-learning. In *2022 5th International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE)*, pages 826–829, 2022.
- [180] Kang Han and Chengyin Ye. Power control research for device-to-device wireless network underlying reinforcement learning. In *2022 Global Conference on Robotics, Artificial Intelligence and Information Technology (GCRAIT)*, pages 351–354, 2022.
- [181] C. S. Arvind and J. Senthilnath. Autonomous rl: Autonomous vehicle obstacle avoidance in a dynamic environment using mlp-sarsa reinforcement learning. In *2019 IEEE 5th International Conference on Mechatronics System and Robots (ICMSR)*, pages 120–124, 2019.
- [182] Ali Jaber Almalki and Pawel Wocjan. Exploration of reinforcement learning to play snake game. In *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 377–381, 2019.
- [183] Prajval Mohan, Lakshya Sharma, and Pranav Narayan. Optimal path finding using iterative sarsa. In *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 811–817, 2021.
- [184] Gian Carlo Cardarilli, Luca Di Nunzio, Rocco Fazzolari, Daniele Giardino, Dario Natale, Marco Re, and Sergio Spanò. “mr q-learning” algorithm for efficient hardware implementations. In *2021 55th Asilomar Conference on Signals, Systems, and Computers*, pages 1186–1190, 2021.
- [185] HoWon Kim and WonChang Lee. Dynamic obstacle avoidance of mobile robots using real-time q-learning. In *2022 International Conference on Electronics, Information, and Communication (ICEIC)*, pages 1–2, 2022.

- [186] Arga Dwi Pambudi, Trihastuti Agustinah, and Rusdhianto Effendi. Reinforcement point and fuzzy input design of fuzzy q-learning for mobile robot navigation system. In *2019 International Conference of Artificial Intelligence and Information Technology (ICAIIIT)*, pages 186–191, 2019.
- [187] HoWon Kim and WonChang Lee. Real-time path planning through q-learning’s exploration strategy adjustment. In *2021 International Conference on Electronics, Information, and Communication (ICEIC)*, pages 1–3, 2021.
- [188] Rashika Raina, Rakesh Kumar Jha, and Ashish Suri. Q-learning based intelligent and interactive healthcare system. In *2022 IEEE 11th International Conference on Communication Systems and Network Technologies (CSNT)*, pages 253–256, 2022.
- [189] Juliana Liman and Li-Hsing Yen. Reinforcement learning for channel and radio allocations to wireless backhaul links. In *2022 23rd Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pages 1–4, 2022.
- [190] Yang Gao, Yuankai Li, and Ziqi Guo. A q-learning based uav path planning method with awareness of risk avoidance. In *2021 China Automation Congress (CAC)*, pages 669–673, 2021.
- [191] Bo Wu and Yanpeng Feng. Monte-carlo bayesian reinforcement learning using a compact factored representation. In *2017 4th International Conference on Information Science and Control Engineering (ICISCE)*, pages 466–469, 2017.
- [192] Youbiao He, Hebi Li, Tian Jin, and Forrest Sheng Bao. Circuit routing using monte carlo tree search and deep reinforcement learning. In *2022 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, pages 1–5, 2022.
- [193] Ekkarat Adsawinnawanawa and Narongdech Keeratipranon. The sharing of similar knowledge on monte carlo algorithm applies to cryptocurrency trading problem. In *2022 International Electrical Engineering Congress (iEECON)*, pages 1–4, 2022.
- [194] Hang Shuai and Haibo He. Online scheduling of a residential microgrid via monte-carlo tree search and a learned model. In *2021 IEEE Power & Energy Society General Meeting (PESGM)*, pages 01–01, 2021.
- [195] Meiyang Li and Wenzhi Huang. Research and implementation of chinese chess game algorithm based on reinforcement learning. In *2020 5th International Conference on Control, Robotics and Cybernetics (CRC)*, pages 81–86, 2020.
- [196] Hebin Huang and Shuqin Li. The application of reinforcement learning in amazons. In *2019 International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)*, pages 369–372, 2019.
- [197] Ding Meng, Bo Jianbo, Qi Yizhong, Fu Yao, and Li Shuqin. Design of amazon chess game system based on reinforcement learning. In *2019 Chinese Control And Decision Conference (CCDC)*, pages 6337–6342, 2019.

- [198] Pengcheng Yu and Jie Luo. Minimize pressure difference traffic signal control based on deep reinforcement learning. In *2022 41st Chinese Control Conference (CCC)*, pages 5493–5498, 2022.
- [199] A Abhishek, Prathvik Nayak, Krishna P Hegde, A Lakshmi Prasad, and K S Nagegowda. Smart traffic light controller using deep reinforcement learning. In *2022 3rd International Conference for Emerging Technology (INCET)*, pages 1–5, 2022.
- [200] Wenbin Chen, Guo Xie, Wenjiang Ji, Rong Fei, Xinhong Hei, Siyu Li, and Jialin Ma. Decision making for overtaking of unmanned vehicle based on deep q-learning. In *2021 IEEE 10th Data Driven Control and Learning Systems Conference (DDCLS)*, pages 350–353, 2021.
- [201] Manish Anand Yadav, Yuhui Li, Guangjin Fang, and Bin Shen. Deep q-network based reinforcement learning for distributed dynamic spectrum access. In *2022 IEEE 2nd International Conference on Computer Communication and Artificial Intelligence (CCAI)*, pages 1–6, 2022.
- [202] Zhixuan He. Analysis on deep reinforcement learning with flappy bird gameplay. In *2022 5th International Conference on Information and Computer Technologies (ICICT)*, pages 95–99, 2022.
- [203] Felipe Moreno-Vera. Performing deep recurrent double q-learning for atari games. In *2019 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, pages 1–4, 2019.
- [204] Md. Rafat Rahman Tushar and Shahnewaz Siddique. A memory efficient deep reinforcement learning approach for snake game autonomous agents. In *2022 IEEE 16th International Conference on Application of Information and Communication Technologies (AICT)*, pages 1–6, 2022.
- [205] Jivitesh Sharma, Per-Arne Andersen, Ole-Christoffer Granmo, and Morten Goodwin. Deep q-learning with q-matrix transfer learning for novel fire evacuation environment. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(12):7363–7381, 2021.
- [206] Guangdou Zhang, Jian Li, Yankai Xing, Olusola Bamisile, and Qi Huang. A deep deterministic policy gradient based method for distribution system load frequency coordinated control with pv and ess. In *2022 4th Asia Energy and Electrical Engineering Symposium (AEEES)*, pages 780–784, 2022.
- [207] Chunhua Zheng, Wei Li, Yao Xiao, Dongfang Zhang, and Suk Won Cha. A deep deterministic policy gradient-based energy management strategy for fuel cell hybrid vehicles. In *2021 IEEE Vehicle Power and Propulsion Conference (VPPC)*, pages 1–6, 2021.
- [208] Avijit Das and Di Wu. Optimal coordination of distributed energy resources using deep deterministic policy gradient. In *2022 IEEE Electrical Energy Storage Application and Technologies Conference (EESAT)*, pages 1–5, 2022.

- [209] Han Wu, Jinjie Tian, Yanbing Jia, Xiangqi Meng, Junyi Shi, Pei Zhao, and Zhichao Yan. An innovative deep reinforcement learning controller for dc/dc dab converters based on deep deterministic policy gradient. In *2021 IEEE 5th Conference on Energy Internet and Energy System Integration (EI2)*, pages 665–669, 2021.
- [210] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning, 2016.
- [211] Bin Zhang, Weihao Hu, Di Cao, Qi Huang, and Zhe Chen. Asynchronous advantage actor-critic based approach for economic optimization in the integrated energy system with energy hub. In *2021 3rd Asia Energy and Electrical Engineering Symposium (AEEES)*, pages 1170–1176, 2021.
- [212] Jian Dong, Haixin Wang, Junyou Yang, Xinyi Lu, Liu Gao, and Xiran Zhou. Optimal scheduling framework of electricity-gas-heat integrated energy system based on asynchronous advantage actor-critic algorithm. *IEEE Access*, 9:139685–139696, 2021.
- [213] Zhengyuan Xu, Yan Zan, Chunlei Xu, Jin Li, Di Shi, Zhiwei Wang, Bei Zhang, and Jiajun Duan. Accelerated drl agent for autonomous voltage control using asynchronous advantage actor-critic. In *2020 IEEE Power & Energy Society General Meeting (PESGM)*, pages 1–5, 2020.
- [214] Atriya Biswas, Pier Giuseppe Anselma, and Ali Emadi. Real-time optimal energy management of multimode hybrid electric powertrain with online trainable asynchronous advantage actor-critic algorithm. *IEEE Transactions on Transportation Electrification*, 8(2):2676–2694, 2022.
- [215] Hani’ah Wafa and Judhi Santoso. Multiagent simulation on hide and seek games using policy gradient trust region policy optimization. In *2020 7th International Conference on Advance Informatics: Concepts, Theory and Applications (ICAICTA)*, pages 1–5, 2020.
- [216] Deyu Yang, Hanbo Zhang, and Xuguang Lan. Research on complex robot manipulation tasks based on hindsight trust region policy optimization. In *2020 Chinese Automation Congress (CAC)*, pages 4541–4546, 2020.
- [217] Huaping Liu, Yupei Wu, and Fuchun Sun. Extreme trust region policy optimization for active object recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 29(6):2253–2258, 2018.
- [218] Nguyen Duc Toan and Kim Gon Woo. Mapless navigation with deep reinforcement learning based on the convolutional proximal policy optimization network. In *2021 IEEE International Conference on Big Data and Smart Computing (Big-Comp)*, pages 298–301, 2021.
- [219] Fan Zishen, Xie Xianzhong, Shi Zhaoyuan, and Chen Xiping. Proximal policy optimization based continuous intelligent power control in cognitive radio network.

- In *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, pages 820–824, 2020.
- [220] Mohamadamin Esfandiari and M. A. Amiri Atashgah. Reinforcement learning control of an aerial robot based on a tuned proximal policy optimization in takeoff and hover phases. In *2022 10th RSI International Conference on Robotics and Mechatronics (ICRoM)*, pages 165–171, 2022.
 - [221] Jong Ha Woo, Lei Wu, Jong-Bae Park, and Jae Hyung Roh. Real-time optimal power flow using twin delayed deep deterministic policy gradient algorithm. *IEEE Access*, 8:213611–213618, 2020.
 - [222] Rakib Hossain, Mukesh Gautam, Mohammad Mansour Lakouraj, Hanif Livani, and Mohammed Benidris. Volt-var optimization in distribution networks using twin delayed deep reinforcement learning. In *2022 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, pages 1–5, 2022.
 - [223] Binchao Zhang, Cheng Jin, Buning Tian, Kaiqi Cao, Junwei Wang, and Pengyu Zhang. Electromagnetic-model-driven twin delayed deep deterministic policy gradient algorithm for stealthy conformal array antenna. *IEEE Transactions on Antennas and Propagation*, 70(12):11779–11789, 2022.
 - [224] Q. Zhu, S. Su, T. Tang, and X. Xiao. Energy-efficient train control method based on soft actor-critic algorithm. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 2423–2428, 2021.
 - [225] Chuan Sun, Xiongwei Wu, Xiuhua Li, Qilin Fan, Junhao Wen, and Victor C.M. Leung. Cooperative computation offloading for multi-access edge computing in 6g mobile networks via soft actor critic. *IEEE Transactions on Network Science and Engineering*, pages 1–1, 2021.
 - [226] Wenjie Liu, Jingfa liu, Mengmeng Cui, and Ming He. An introductory review on quantum game theory. In *2010 Fourth International Conference on Genetic and Evolutionary Computing*, pages 386–389, 2010.
 - [227] Mohamed Abdou Bouteldja, Mohamed Baadeche, and Mohamed Batouche. A novel approach for image denoising based on evolutionary game theory. In *2014 4th International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6, 2014.
 - [228] Fujiang Sun, Qiqing Song, and Ling Jing. The effects of inducing strategies on cooperation in prisoner’s dilemma games. In *2013 5th International Conference on Intelligent Human-Machine Systems and Cybernetics*, volume 2, pages 311–315, 2013.
 - [229] Abdelkrim Doufene and Daniel Krob. Pareto optimality and nash equilibrium for building stable systems. In *2015 Annual IEEE Systems Conference (SysCon) Proceedings*, pages 542–545, 2015.

- [230] Yan Li, Liang Xiao, Jinliang Liu, and Yuliang Tang. Power control stackelberg game in cooperative anti-jamming communications. In *The 2014 5th International Conference on Game Theory for Networks*, pages 1–6, 2014.
- [231] Man Li, Huaiyu Wu, and Lin Chen. Research on multi-robot cooperative task assignment based on cooperative game. In *2021 4th International Conference on Intelligent Autonomous Systems (ICoIAS)*, pages 424–429, 2021.
- [232] Shuili Zhang, Wen Ji, Bo-Wei Chen, and Yiqiang Chen. Research on cooperative video transmission over heterogeneous devices based on game theory. In *The 2014 5th International Conference on Game Theory for Networks*, pages 1–5, 2014.
- [233] Zhen He and Jianping Wang. Non-cooperative differential game based load balancing algorithm in radio-over-fibre system. *China Communications*, 11(14):79–85, 2014.
- [234] Li Hui, Liu Chun, and Li Hua. Power control based on non-cooperative game in wireless sensor networks. In *2013 6th International Conference on Intelligent Networks and Intelligent Systems (ICINIS)*, pages 135–138, 2013.
- [235] Jiandong Duan, Qian Sun, Chen Wu, Yao Yang, and Siyu Tu. A non-cooperative game based optimal dispatch model for integrated energy systems. *CSEE Journal of Power and Energy Systems*, pages 1–12, 2021.
- [236] Ying Yang, Wei Liu, Jizhou Zhou, Jinhui Zhou, and Junfang Zhang. Non-cooperative-game-based multi-agent collaborative planning method for distributed generations. In *2020 IEEE 4th Conference on Energy Internet and Energy System Integration (EI2)*, pages 273–278, 2020.
- [237] Mourad Elhadef. Using game theory to solve the fault diagnosis problem under symmetric comparison rules. In *2017 IEEE International Conference on Computer and Information Technology (CIT)*, pages 237–244, 2017.
- [238] Xiaohui Zhao, Wenhui Liu, Likun Zhu, and Jianfeng Liang. Inventory games of manufacturer and supplier under symmetric information. In *2009 International Conference on Management and Service Science*, pages 1–4, 2009.
- [239] Filipe Maciel Roberto, Joaquim Celestino Júnior, and Henning Schulzrinne. Using a symmetric game based in volunteer’s dilemma to improve vanets multihop broadcast communication. In *2011 IEEE 22nd International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 777–782, 2011.
- [240] Jingrong Wen, Muqing Wu, Lv Bo, Wang Dongyang, and Qinpeng Zhang. A game theory-based asymmetric multi-channel selection algorithm in wireless ad hoc networks. In *2012 IEEE 14th International Conference on Communication Technology*, pages 279–283, 2012.
- [241] Yifan Dou, Yongbo Xiao, and Jian Chen. Asymmetric pricing game in two-sided platforms. In *2009 6th International Conference on Service Systems and Service Management*, pages 921–926, 2009.

- [242] Ishfaq Ahmad, Sanjay Ranka, and Samee Ullah Khan. Using game theory for scheduling tasks on multi-core processors for simultaneous optimization of performance and energy. In *2008 IEEE International Symposium on Parallel and Distributed Processing*, pages 1–6, 2008.
- [243] Yu Haijie, Li Xiangyang, and Su Sheng. Price decision analysis based on game theory for two stages ecological industry chain. In *2007 IEEE International Conference on Control and Automation*, pages 1250–1254, 2007.
- [244] Junkai Lu, Xiaoyan Wang, Dayi Wang, and Yajie Wang. Parallel monte carlo tree search in perfect information game with chance. In *2016 Chinese Control and Decision Conference (CCDC)*, pages 5050–5053, 2016.
- [245] Zhiwei Xing, Yunxiao Tang, and Qian Luo. A flight slot allocation model based on game theory. In *2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, pages 364–369, 2016.
- [246] Seyyed Mohammad Reza Farshchi and Irene Khehvazadeh. Finding risk dominance strategy in imperfect game, a theory perspective. In *Proceedings of 2012 UKACC International Conference on Control*, pages 798–802, 2012.
- [247] C. Nandini, Sachin Kumar, and R Laasya. Ai poker agent based on game theory. In *2019 1st International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE)*, pages 201–204, 2019.
- [248] Christopher G. Harris. Developing a repeated multi-agent constant-sum game algorithm using human computation. In *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, volume 2, pages 390–394, 2012.
- [249] Feng Yang, Xuehai Zhou, Gangyong Jia, and Qiyuan Zhang. A non-cooperative game approach for intrusion detection in smartphone systems. In *2010 8th Annual Communication Networks and Services Research Conference*, pages 146–151, 2010.
- [250] E. C. Amadi, G. E. Eheduru, F. U. Eze, C. Ikerionwu, and H. O. Amuji. A zero-sum mitigation game model for distributed denial of service attack using linear programming. In *2017 IEEE 3rd International Conference on Electro-Technology for National Development (NIGERCON)*, pages 327–333, 2017.
- [251] Azzedine Touazi, Fatiha Mokdad, and Djamel Bouchaffra. Feature selection scheme based on zero-sum two-player game. In *2014 22nd International Conference on Pattern Recognition*, pages 1342–1347, 2014.
- [252] Xuelong Zhang, Hui Wang, Jiangxia Nan, Yuxi Luo, and Yanling Yi. Modeling and numerical methods of supply chain trust network with the complex network. *Symmetry*, 14(2):235, Jan 2022.

- [253] Tianjiao An, Xinye Zhu, Yuanchun Li, Hongwen Li, and Bo Dong. A non-zero-sum-based neural-optimal control method for modular and reconfigurable robot systems. In *2021 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pages 1402–1407, 2021.



APPENDIX A APARTMENT BUILDING MODEL

```
classdef Apartment
    methods (Static=true)
        function [Rewards, Infos, SoCout] = ApartmentHome(
            V2G, P2P, Actions, Load, Solar, EVInfos,
            SoCs_Prev, t)
            if ~iscolumn(V2G); V2G = transpose(V2G); end
            if ~iscolumn(P2P); P2P = transpose(P2P); end
            if ~iscolumn(Load); Load = transpose(Load); end
            [BuyingPrice, SellingPrice] = Home.
                Supplier_Price(t);
            %% Before Energy Trading
            [H, Infos, SoCs, SalableEnergy] = Home.
                HomeBeforeSelling(Actions, Load, Solar,
                EVInfos, SoCs_Prev(:,1), SoCs_Prev(:,2), t,
                V2G, P2P);
            evchargingcosts = Home.CalculateEVChargingCost(
                Infos, [BuyingPrice, SellingPrice]);
            %% Energy Trading Process
            transactions = Home.EnergySellingStockMarket(H,
                0);
            invoices = Trading.PrepareInvoices(transactions
                ,0);
            sold_allocated = Trading.
                AllocatePowersForSelling(invoices,
                SalableEnergy);
            %% After Energy Trading
            FromGrid = Home.ComputeEnergyFromGrid(Infos,
                invoices);
            ToGrid = Home.ComputeEnergyToGrid(Infos,
                sold_allocated);
            bills = Home.CalculateBills(invoices, FromGrid,
                ToGrid, evchargingcosts, [BuyingPrice,
                SellingPrice]);
            SoC = Home.CalculateSoCsAfterSelling(SoCs,
                sold_allocated, V2G);
            Infos = Home.GatherInformation(Load, Solar, Infos
                , sold_allocated, FromGrid, ToGrid, bills, [
                SoCs_Prev(:,1), SoCs_Prev(:,2)], SoC, invoices
                );
            SoCout = [SoC(:,2), SoC(:,3)];
        end
    end
end
```

```
        Rewards = Home.ComputeRewards(Infos);  
    end  
end  
end
```



APPENDIX B ELECTRIC VEHICLE MODEL

```

classdef ElectricVehicle
    methods (Static=true)
        function [Pout, Pgrid, Pava, SoC] = EV(P, m,
            SoC_Prev, EVInfos, V2G)
            if ~iscolumn(P); P = transpose(P); end
            if ~iscolumn(m); m = transpose(m); end
            if ~iscolumn(SoC_Prev); SoC_Prev = transpose(
                SoC_Prev); end
            EVStatus = EVInfos(:,1);
            Pdrive = Home.FromEnergykWhToPower(EVInfos(:,2)
                .* EVInfos(:,3));
            P = max(P, 0);
            [Pout_discharge] = ElectricVehicle.
                EVDischarger(P, SoC_Prev) .* (m == 1) .*
                EVStatus .* V2G;
            [Pout_charge, Pgrid] = ElectricVehicle.
                EVCharger(SoC_Prev);
            Pout_charge = Pout_charge .* (m == -1) .*
                EVStatus;
            Pgrid = Pgrid .* (m == -1) .*
                EVStatus;
            Pout = (Pout_discharge - Pout_charge) .*
                EVStatus;
            SoC = ElectricVehicle.EVSoC(Pout, SoC_Prev);
            SoC = ElectricVehicle.EVSoC(Pdrive, SoC);
            Pava = ElectricVehicle.EVAvailablePower(SoC) .*
                EVStatus;
        end
        function soc = EVDriveSoCCalculator(soc_prev,
            ev_rate, ev_idrive, EVBatteryParams)
            Vnominal = EVBatteryParams(:, 3);
            Ns = EVBatteryParams(:, 1);
            Np = EVBatteryParams(:, 2);
            Ah = EVBatteryParams(:, 5);
            soc_temp = ((ev_rate == 1) .* (ev_idrive ./ ((
                Vnominal .* Ns .* Np .* Ah * 85/100)/1000)));
            soc = soc_prev - soc_temp;
            soc = min(1, max(0, soc));
        end
        function [Pout] = EVDischarger(P, soc)

```

```

EVCapacity = Home.EVParams(:, 1);
MinSoC     = Home.EVParams(:, 3);
MaxPower    = Home.EVParams(:, 7);
P = max(P, 0);
max_discharge_power = Home.FromEnergykWhToPower
    ((EVCapacity .* soc) - (EVCapacity .* MinSoC
    ));
max_discharge_power = max(max_discharge_power,
    0);
max_discharge_power = min(max_discharge_power,
    MaxPower);
Pout = min(P, max_discharge_power) .* Home.
    Vehicle(:, 1) .* Home.Vehicle(:, 2);
end
function [Pout, Pgrid] = EVCharger(soc)
    Pcharge = Home.FromPowerToEnergykWh(Home.
        EVParams(:, 8));
    Efficiency = Home.EVParams(:, 5);
    Efficiency(Efficiency <= 0) = 1;
    EVCapacity = Home.EVParams(:, 1);
    MaxSoC      = Home.EVParams(:, 2);
    CurrentEnergy = EVCapacity .* soc;
    MaxEnergy    = EVCapacity .* MaxSoC;
    AllowedEnergy = max(MaxEnergy - CurrentEnergy
        , 0);
    Pgrid        = Home.FromEnergykWhToPower(min
        (Pcharge, AllowedEnergy) ./ Efficiency) .*
        Home.Vehicle(:,1);
    Pout          = Pgrid .* Efficiency .* Home.
        Vehicle(:,1);
end
function [soc] = EVSoC(P, soc_prev)
    EVCapacity = Home.EVParams(:, 1);
    EVCapacity(EVCapacity<=0) = 1;
    soc = soc_prev - (Home.FromPowerToEnergykWh(P)
        ./ EVCapacity);
    soc = min(1, max(0, soc)) .* Home.Vehicle(:,1);
end
function [Pava] = EVAvailablePower (soc)
    EVCapacity = Home.EVParams(:, 1);
    MinSoC     = Home.EVParams(:, 3);
    MaxPower    = Home.EVParams(:, 7);
    Pava = Home.FromEnergykWhToPower((EVCapacity .*
        soc) - (EVCapacity .* MinSoC));
    Pava = max(Pava, 0);
    Pava = min(Pava, MaxPower) .* Home.Vehicle(:,

```

```
1) .* Home.Vehicle(:, 2);  
    end  
end  
end
```



APPENDIX C ENERGY STORAGE SYSTEM MODEL

```

classdef ESSBattery
    methods (Static=true)
        function [Pout,Pava,Eout,soc] = ESS(Prequired,m,soc
        )
            if ~iscolumn(Prequired); Prequired = transpose(
                Prequired); end
            if ~iscolumn(m); m = transpose(m); end
            if ~iscolumn(soc); soc = transpose(soc); end
            ESSCapacity = Home.ESSParams(:,1); %
                Capacity of ESS (kWh)
            MaxSoC = Home.ESSParams(:,2); %
                Maximum SoC of ESS (0-1)
            MinSoC = Home.ESSParams(:,3); %
                Minimum SoC of ESS (0-1)
            MaxDischargePower = Home.ESSParams(:,6); %
                Maximum Discharge Power (kW)
            max_charge_power = -Home.
                FromEnergykWhToPower(((ESSCapacity.*MaxSoC)
                - (ESSCapacity.*soc)));
            max_discharge_power = Home.
                FromEnergykWhToPower(((ESSCapacity.*soc) - (
                ESSCapacity.*MinSoC)));
            max_charge_power = min(max_charge_power, 0);
            max_discharge_power = max(max_discharge_power,
                0);
            Pout = min(max_discharge_power, max(
                max_charge_power, Prequired)) .* m;
            Eout = Home.FromPowerToEnergykWh(Pout);
            soc = ESSBattery.ESSSoC(Eout,Home.ESSParams,
                soc);
            Pava = Home.FromEnergykWhToPower((ESSCapacity
                .* soc - ESSCapacity .* MinSoC));
            Pava = min(MaxDischargePower,Pava);
        end
        function soc = ESSSoC(batt_eout,ESSParams,soc_prev)
            ess_cap = ESSParams(:,1);
            soc = soc_prev - (batt_eout./ess_cap);
            soc = min(1, max(0, soc));
        end
    end
end

```

end



APPENDIX D HOME MODEL

```

classdef Home
    properties (Constant=true)
        Treal = 60; % -> Real time sampling time (
            Minutes)
        T = 1; % -> Simulation sampling time
        % Energy Storage System (ESS) Parameters -> [Max
            kWh, max soc, min soc, chg eff, dsc eff, max
            power, min power]
        ESSParams = [
            4.8, 1, 0.1, 0.95, 0.95, 2000, -5000
            4.8, 1, 0.1, 0.95, 0.95, 2000, -5000
            4.8, 1, 0.1, 0.95, 0.95, 2000, -5000
            4.8, 1, 0.1, 0.95, 0.95, 2000, -5000
            4.8, 1, 0.1, 0.95, 0.95, 2000, -5000
            4.8, 1, 0.1, 0.95, 0.95, 2000, -5000
        ]
        % Electric Vehicle (EV) Parameters -> [kWh, max_soc
            , min_soc, target_soc, chg_eff, dsc_eff,
            max_power, charging_power]
        EVParams = [
            36, 0.80, 0.20, 0.65, 0.92, 0.95, 1000,
                11000
            24, 0.70, 0.25, 0.65, 0.92, 0.95, 1000,
                7400
            0, 0.00, 0.00, 0.00, 0.00, 0.00, 0,
                0
            25, 0.68, 0.30, 0.65, 0.92, 0.95, 1250,
                7400
            0, 0.00, 0.00, 0.00, 0.00, 0.00, 0,
                0
            0, 0.00, 0.00, 0.00, 0.00, 0.00, 0,
                0
        ]
        % ESS Minimum selling SoC
        ESSMinSellingSoC = [
            0.15
            0.15
            0.15
            0.15
            0.15
        ]
    end
end

```

```

                                0.15
                                ]
% Vehicle Parameters -> [EV Available, V2G
    Available]
Vehicle = [
    1 1
    1 1
    0 0
    1 1
    0 0
    0 0
    ]
% Electric Vehicle (EV) Parameters -> [Max disch
    power, chg eff, disc eff, charging power, max
    soc, min soc]
EVParams2 = [
    2000,    0.95,    0.95,    11000,    0.8,
    0.2
    1500,    0.92,    0.93,    7400,    0.70,
    0.25
    0,    0,    0,    0,    0,
    0
    1600,    0.90,    0.90,    7400,    0.68,
    0.30
    0,    0,    0,    0,    0,
    0
    0,    0,    0,    0,    0,
    0
    ]
% Li-Ion Battery Parameters -> [Series, Parallel,
    Vnominal, Vfull, Vempty]
EVBatteryParams = [
    96, 35, 3.6,    4.2,    3.5
    96, 40, 3.6,    4.2,    2.5
    0, 0, 0,    0,    0
    92, 38, 3.6,    4.2,    2
    0, 0, 0,    0,    0
    0, 0, 0,    0,    0
    ]
EVTargets = [
    0.65
    0.65
    0
    0.65
    0
    0
    ]

```

```

    ]
    % Electricity Prices TL/kWh
    price_gunduz = 195.2907 / 100
    price_puant  = 286.0696 / 100
    price_gece   = 122.6951 / 100
    price_buying = 32.00000 / 100
    % Hours definitions
    hour_gunduz_start = 06*60 / Home.Treal
    hour_gunduz_end   = 17*60 / Home.Treal
    hour_puant_start  = 17*60 / Home.Treal
    hour_puant_end    = 22*60 / Home.Treal
    day_long = 1440 / Home.Treal
    % Number of homes
    n_home = 6
    % Create id array of homes
    ids_home = transpose(1:Home.n_home);
end
methods (Static=true)
function [selling_price,buying_price] =
    Supplier_Price(T)
    selling_price=0;
    buying_price = Home.price_buying;
    t = mod(T,Home.day_long);
    if (t>=0 && t<Home.hour_gunduz_start) || (t>=
        Home.hour_puant_end && t<Home.day_long)
        selling_price = Home.price_gece;
    elseif (t>=Home.hour_gunduz_start && t<Home.
        hour_gunduz_end)
        selling_price = Home.price_gunduz;
    elseif (t>=Home.hour_puant_start && t<Home.
        hour_puant_end)
        selling_price = Home.price_puant;
    end
end
function [H, Infos, SoCs, SalableEnergy] =
    HomeBeforeSelling(Actions, load, solar, EVInfos,
        ESSSoC, EVSoC, t, V2G, P2P)
    if ~iscolumn(V2G); V2G = transpose(V2G); end
    if ~iscolumn(P2P); P2P = transpose(P2P); end
    if ~iscolumn(load); load = transpose(load); end
    [BuyingPrice, ~] = Home.Supplier_Price(t);
    % Actions -> [BattAct, BattPercentage, EVAct,
        EVPercentage, PriceSignal, EVMode]
    BattAct      = Actions(:,1) .* P2P;
    BattPercentage = Actions(:,2) .* P2P;
    EVAct        = Actions(:,3) .* P2P .* V2G;

```

```

EVPercentage      = Actions(:,4) .* P2P .* V2G;
PriceSignal       = Actions(:,5) .* P2P;
ChargingTime = (mod(t,24*60/Home.Treal) >= 3 *
    60 / Home.Treal) && (mod(t,24*60/Home.Treal)
    < 6 * 60 / Home.Treal);
solar = solar * ones(Home.n_home,1);
diff = load - solar;
[BatteryHappened, BatteryAvailable, ~,
    ESSSoCOut] = ESSBattery.ESS(diff, ones(Home.
    n_home,1), ESSSoC);
EVRequired = diff - BatteryHappened;
EVMode = ones(Home.n_home, 1) .* ((ChargingTime
    ==0) - (ChargingTime==1));
[EVHappened, EVGrid, EVAvailable, EVSoCOut] =
    ElectricVehicle.EV(EVRequired, EVMode, EVSoC
    , EVInfos, V2G);
EVHappened = EVHappened .* (EVHappened > 0) .*
    V2G;
Result = EVRequired - EVHappened;
BuyingPower = Result .* (Result > 0) .* P2P;
BatteryAvailable_ = BatteryAvailable .* BattAct
    .* BattPercentage .* (BuyingPower <= 0);
EVAavailable_ = EVAavailable .* EVAct
    .* EVPercentage .* (BuyingPower <= 0);
SellingPower_ = abs(Result .* (Result<0)) +
    (BatteryAvailable_ + EVAavailable_) .* (
    BuyingPower <= 0);
BatteryChargedPower = abs(BatteryHappened)
    .* (BatteryHappened<0);
BatteryDischargedPower = abs(BatteryHappened)
    .* (BatteryHappened>0);
EVChargedPower = abs(EVGrid);
EVDischargedPower = abs(EVHappened) .* (
    EVHappened>0);
SalableEnergy = [Home.ids_home, SellingPower_,
    abs(Result.*(Result<0)), BatteryAvailable,
    BatteryAvailable_, EVAavailable, EVAavailable_
    ];
SalableEnergy(:,2:end) = SalableEnergy(:,2:end)
    .* P2P;
H = [Home.ids_home, PriceSignal .*
    BuyingPrice, SellingPower_, BuyingPower];
H(:,2:end) = H(:,2:end) .* P2P;
Infos = [Home.ids_home, Result, load, solar,
    BatteryChargedPower, BatteryDischargedPower,
    EVChargedPower, EVDischargedPower];

```

```

        SoCs      = [Home.ids_home, ESSSoCOut, EVSoCOut];
end
function [FromGrid] = ComputeEnergyFromGrid(Infos,
Invoices)
    temp_ = Home.FromPowerToEnergykWh((Infos(:,2) -
        Invoices(:,2)) .* ((Infos(:,2) - Invoices
            (:,2)) > 0));
    FromGrid = [Home.ids_home, temp_];
end
function [ToGrid] = ComputeEnergyToGrid(Infos,
sold_allocated)
    ToGrid = [Home.ids_home, abs(Home.
        FromPowerToEnergykWh((Infos(:,2) +
            sold_allocated(:,3)) .* ((Infos(:,2) +
                sold_allocated(:,3)) < 0)))];
end
function [bills] = CalculateBills(invoices,
FromGrid, ToGrid, EVChargingCost, SP)
    EnergyBuyingPrice = SP(1);
    EnergySellingPrice = SP(2);
    bills = [Home.ids_home, (invoices(:,5) + ToGrid
        (:,2).*EnergySellingPrice) - (invoices(:,4)
            + FromGrid(:,2).*EnergyBuyingPrice +
                EVChargingCost(:,2))];
end
function [SoC] = CalculateSoCsAfterSelling(SoC_Prev
, sold_allocated, V2G)
    SoC      = [Home.ids_home, zeros(Home.n_home,2)
        ];
    [~, ~, ~, SoC(:,2)] = ESSBattery.ESS(
        sold_allocated(:,4), ones(Home.n_home, 1),
        SoC_Prev(:,2));
    [~, ~, ~, SoC(:,3)] = ElectricVehicle.EV(
        sold_allocated(:,5), ones(Home.n_home, 1),
        SoC_Prev(:,3), [1,0,0], V2G);
end
function [Infos] = GatherInformation(Loads,Solar,
Infos_,sold_allocated,FromGrid,ToGrid,Bill,
SoCInit,SoCFinal,invoices)
    Infos = [Home.ids_home, Loads, ones(Home.n_home
        ,1).*Solar, Infos_(:,5), Infos_(:,6),...
        sold_allocated(:,4), Infos_(:,7), Infos_
            (:,8), sold_allocated(:,5), FromGrid
                (:,2), ToGrid(:,2),...
        Bill(:,2), invoices(:,2), invoices(:,3),
        SoCInit(:,1), SoCFinal(:,2), SoCInit

```

```

        (:,2), SoCFinal(:,3)];
end
function [EVChargingCost] = CalculateEVChargingCost
(Infos,SP)
    EVChargingCost = [Home.ids_home, Home.
        FromPowerToEnergykWh(Infos(:,7).*SP(1))];
end
function transactions = EnergySellingStockMarket(H,
displayInfo)
    % transactions -> ID, buyer, seller, bought,
    sold, price
    transactions = zeros(5000,5);
    sellers = sortrows(H(H(:,3)>0, [1,2,3]),2,"
        ascend");
    buyers = sortrows(H(H(:,4)>0, [1,4]),2,"
        descend");
    seller_count = size(sellers,1);
    buyer_count = size(buyers,1);
    if seller_count > 0 && buyer_count > 0
        [transactions] = Trading.
            TradeWithEqualOpportunities(buyers,
            sellers,0);
    end
    transactions = transactions(transactions(:,1)
        >0,:);
    if displayInfo
        disp(array2table(transactions,"
            VariableNames",{ 'id','Buyer ID','Seller
            ID','Power','Price'}));
    end
end
function Energy = FromPowerToEnergykWh(Power)
    Energy = (Power / 1000) * (Home.Treal / 60);
end
function Power = FromEnergykWhToPower(Energy)
    Power = (Energy * 1000) * (60 / Home.Treal);
end
function Rewards = ComputeRewards(Infos)
    bills = Home.NormalizeValues(Infos(:, 12));
    grids = Home.NormalizeValues(Infos(:, 10));
    Rewards = ( 0.7 .* ( sum(bills) + bills ) ) - (
        0.3 .* ( sum(grids) + grids ) );
end
function Normalized = NormalizeValues(values)
    Normalized = (values - min(values)) / (max(
        values) - min(values));

```

```
        Normalized(isnan(Normalized)) = 0;  
    end  
end  
end
```



APPENDIX E INITIALIZATION BLOCK OF THE MATLAB

```
load('data/home_ev_stat.mat');
SoCinit(:,1) = [0.90 0.90 0.90 0.90 0.90 0.90]; % SoC of
    ESS Battery
SoCinit(:,2) = [0.60 0.60 0.00 0.60 0.00 0.00]; % SoC of EV
    Battery
if Home.Treal == 10
    load('data/Load/loads_10min.mat');
    load('data/Solar/solar_10min.mat');
    load('data/EVStats/evstat_10min.mat');
    LoadsCalculated = LoadsCalculated_10min;
    SolarCalculated = SolarCalculated_;
    h_evstatus = h_evstatus_;
    h_evrate = h_evrate_;
    h_evidrive = h_evidrive_;
    clear LoadsCalculated_10min SolarCalculated_
        h_evstatus_ h_evidrive_ h_evrate_
elseif Home.Treal == 20
    load('data/Load/loads_20min.mat');
    load('data/Solar/solar_20min.mat');
    load('data/EVStats/evstat_20min.mat');
    LoadsCalculated = LoadsCalculated_20min;
    SolarCalculated = SolarCalculated_;
    h_evstatus = h_evstatus_;
    h_evrate = h_evrate_;
    h_evidrive = h_evidrive_;
    clear LoadsCalculated_20min SolarCalculated_
        h_evstatus_ h_evidrive_ h_evrate_
elseif Home.Treal == 30
    load('data/Load/loads_30min.mat');
    load('data/Solar/solar_30min.mat');
    load('data/EVStats/evstat_30min.mat');
    LoadsCalculated = LoadsCalculated_30min;
    SolarCalculated = SolarCalculated_;
    h_evstatus = h_evstatus_;
    h_evrate = h_evrate_;
    h_evidrive = h_evidrive_;
    clear LoadsCalculated_30min SolarCalculated_
        h_evstatus_ h_evidrive_ h_evrate_
```

```
else
    load('data/Load/loads_60min.mat');
    load('data/Solar/solar_60min.mat');
    load('data/EVStats/evstat_60min.mat');
    LoadsCalculated = LoadsCalculated_60min;
    SolarCalculated = SolarCalculated_;
    h_evstatus = h_evstatus_;
    h_evrate = h_evrate_;
    h_evidrive = h_evidrive_;
    clear LoadsCalculated_60min SolarCalculated_
          h_evstatus_ h_evidrive_ h_evrate_
end
```



APPENDIX F DDQN MODEL

```
%% Model Setting
clear;
mdl = 'Model';
open_system(mdl)
HowManyDaysToTrain = 5;
Initialization;
clc;
fprintf('Model 11tanmlamalar 1tamamland! -> \t%s\n',
    datetime)
%% Action Space
actBattAct = [0 1];
actBattPer = 0.2:0.1:1;
actEVAct = [0 1];
actEVPer = 0.2:0.1:1;
actPriceSignal = 0.60:0.05:0.95;
count = 1;
for a=1:length(actBattAct)
    for b=1:length(actBattPer)
        for c=1:length(actEVAct)
            for d=1:length(actEVPer)
                for e=1:length(actPriceSignal)
                    actHome{count} = ...
                        [actBattAct(a) actBattPer(b)
                        actEVAct(c) actEVPer(d)
                        actPriceSignal(e)];
                    count = count+1;
                end
            end
        end
    end
end
clear count a b c d e;
numActHome = length(actBattAct)*length(actBattPer)*length(
    actEVAct)*length(actEVPer)*length(actPriceSignal);
%% Observations
numObsHome = 7;
observations = rlNumericSpec([numObsHome 1],
    'LowerLimit',zeros(numObsHome,1),'UpperLimit',inf(
    numObsHome,1));
observations.Name = 'observationEV';
```

```

observations.Description      = 'load,solar,batt_soc,ev_soc,
    ev_stat,buying_price,selling_price';
actionInfoHome               = rlFiniteSetSpec(actHome);
actionInfoHome.Name          = 'Action';
actionInfoHome.Description   = 'Action information for homes
    ';
clear actBattAct actBattPer actEVAct actEVPer
    actPriceSignal actsHome
%% Environment
obsInfos = {observations,observations,observations,
    observations,observations,observations};
actInfos = {actionInfoHome,actionInfoHome,actionInfoHome,
    actionInfoHome,actionInfoHome,actionInfoHome};
agentblk = ["Model/Agents/Agent1","Model/Agents/Agent2","
    Model/Agents/Agent3","Model/Agents/Agent4","Model/Agents
    /Agent5","Model/Agents/Agent6"];
env = rlSimulinkEnv mdl,agentblk,obsInfos,actInfos,"
    UseFastRestart","on");
%% Environment Resets to initilaze variables
env.ResetFcn = @(in)localResetFcn(in);
rng('default')
%% DNN Design for homes
nHome = 1600;
dnnHome = [
    featureInputLayer(numObsHome,'Normalization','none','
        Name','States of the Environment')
    fullyConnectedLayer(nHome*2,'Name','1. Fully connected
        layer with 3200 nodes')
    reluLayer('Name','1. Rectified Linear Unit (ReLu)')
    fullyConnectedLayer(nHome,'Name','2. Fully connected
        layer with 1600 nodes')
    reluLayer('Name','2. Rectified Linear Unit (ReLu)')
    fullyConnectedLayer(nHome/4,'Name','3. Fully connected
        layer with 400 nodes')
    reluLayer('Name','3. Rectified Linear Unit (ReLu)')
    fullyConnectedLayer(nHome/8,'Name','4. Fully connected
        layer with 200 nodes')
    reluLayer('Name','4. Rectified Linear Unit (ReLu)')
    fullyConnectedLayer(numActHome,'Name','Action Outputs')
    ];
dnnHome = dlnetwork(dnnHome);
l_rate = 1e-3;
criticOpts = rlOptimizerOptions('LearnRate',l_rate,'
    GradientThreshold',1);
criticHome = rlVectorQValueFunction(dnnHome,observations,
    actionInfoHome);

```

```

agentOptions = rlDQNAgentOptions(...
    'SampleTime',Home.T,...
    'UseDoubleDQN',true,...
    'TargetSmoothFactor',1e-3,...
    'DiscountFactor',0.965 ,...
    'ExperienceBufferLength',1e6 ,...
    'MiniBatchSize',16, ...
    'CriticOptimizerOptions',criticOpts);
Tf = 24*60/Home.Treal*HowManyDaysToTrain;
maxepisodes = 2000;
maxsteps = ceil(Tf/Home.T);
agentOptions.EpsilonGreedyExploration.Epsilon = 1;
agentOptions.EpsilonGreedyExploration.EpsilonMin = 0.1;
agentOptions.EpsilonGreedyExploration.EpsilonDecay =
    (1-0.1)/maxepisodes*0.75;
agent1 = rlDQNAgent(criticHome,agentOptions);
agent2 = rlDQNAgent(criticHome,agentOptions);
agent3 = rlDQNAgent(criticHome,agentOptions);
agent4 = rlDQNAgent(criticHome,agentOptions);
agent5 = rlDQNAgent(criticHome,agentOptions);
agent6 = rlDQNAgent(criticHome,agentOptions);
trainingOpts = rlMultiAgentTrainingOptions(...
    "AgentGroups","auto",...
    "LearningStrategy","decentralized",...
    'MaxEpisodes',maxepisodes,...
    'MaxStepsPerEpisode',maxsteps,...
    "ScoreAveragingWindowLength",20,...
    'StopTrainingCriteria','EpisodeCount',...
    'StopTrainingValue',maxepisodes);
%% Training Phase
trainingStats = train([agent1,agent2,agent3,agent4,agent5,
    agent6],env,trainingOpts);
%% Save Training Results
a = datestr(datetime,30);
[status, msg, msgID] = mkdir('results/',a);
save(['results/',a,'/agent_DDQN1.mat'], "agent1");
save(['results/',a,'/agent_DDQN2.mat'], "agent2");
save(['results/',a,'/agent_DDQN3.mat'], "agent3");
save(['results/',a,'/agent_DDQN4.mat'], "agent4");
save(['results/',a,'/agent_DDQN5.mat'], "agent5");
save(['results/',a,'/agent_DDQN6.mat'], "agent6");
save(['results/',a,'/traininstatus.mat'], "trainingStats");
%% Reset function
function in = localResetFcn(in)
    ESSinits = randi([90,90],6,1)/100;
    EVinits = [randi([65,65],1,1)/100;randi([65,65],1,1)

```

```
        /100;0;randi([65,65],1,1)/100;0;0];  
SoCinit_ = [ESSinits,EVinits];  
in = setVariable(in,'SoCinit',SoCinit_);  
end
```



APPENDIX G P2P ENERGY TRADING MODEL

```

classdef Trading
    methods (Static = true)
        function [transactions] =
            TradeWithEqualOpportunities(Buyer,Seller,
            displayInfo)
            % transactions -> ID, buyer, seller, bought,
            cost
            transactions = zeros(5000,5);
            transactions_count = 0;
            Seller_ = Seller;
            Buyer_ = Buyer;
            for i = 1:size(Seller,1)
                while Seller_(i,3) > 0 && sum(Buyer_(:,2)
                >0) > 0
                    selling_power_per_home = Seller_(i,3) /
                    sum(Buyer_(:,2)>0);
                    Seller_(i,3) = Seller_(i,3) - sum(min(
                    Buyer_(:,2),selling_power_per_home))
                    ;
                    for b = 1:sum(Buyer_(:,2) > 0)
                        buyer_id = Buyer_(b,1);
                        seller_id = Seller_(i,1);
                        amount = min(Buyer_(b,2),
                        selling_power_per_home);
                        price_total = Home.
                        FromPowerToEnergykWh(amount) *
                        Seller_(i,2);
                        transactions_count =
                        transactions_count + 1;
                        transactions(transactions_count,:)
                        = [transactions_count,buyer_id,
                        seller_id,amount,price_total];
                    end
                    Buyer_(:,2) = Buyer_(:,2) - min(Buyer_
                    (:,2),selling_power_per_home);
                end
            end
            transactions = transactions(transactions(:,1)
            >0,:);
            if displayInfo

```

```

        disp('Transactions: ');
        disp(array2table(transactions,"
            VariableNames",{ 'id','Buyer ID','Seller
            ID','Power','Price'}));
    end
end
function [invoices] = PrepareInvoices(transactions,
displayInfo)
    % invoices -> Home ID, Bought, Sold, Cost,
    Revenue
    invoices = zeros(Home.n_home,5);
    invoices(:,1) = 1:Home.n_home;
    for id = 1:Home.n_home
        invoices(id,2) = sum(transactions(
            transactions(:,2)==id,4));
        invoices(id,3) = sum(transactions(
            transactions(:,3)==id,4));
        invoices(id,4) = sum(transactions(
            transactions(:,2)==id,5));
        invoices(id,5) = sum(transactions(
            transactions(:,3)==id,5));
    end
    if displayInfo
        disp('Invoices: ');
        disp(array2table(invoices,"VariableNames",{
            'id','Bought','Sold','Cost','Revenue'}))
        ;
    end
end
end
function [sold_allocated] =
AllocatePowersForSelling(invoices, SalableEnergy
)
    % sold_allocated -> 'id', 'TotalSold','Excess
    Sold', 'Battery Sold','EV Sold'
    sold_allocated = zeros(Home.n_home, 5);
    sold_allocated(:,1) = 1:Home.n_home;
    % SalableEnergy -> 'id', 'Salable Total', '
    Excess','BatteryAva','Battery','EVAva','EV'
    sold_amounts = invoices(invoices(:,3)
        >0,[1,3]);
    SalableEnergy = SalableEnergy(SalableEnergy
        (:,2)>0,:);
    for i=1:size(sold_amounts(:,1),1)
        sold_id = sold_amounts(i,1);
        sold_amount = sold_amounts(i,2);
        salable_excess = SalableEnergy(

```

```

        SalableEnergy(:,1)==sold_id,3);
    salable_battery = SalableEnergy(
        SalableEnergy(:,1)==sold_id,5);
    salable_ev      = SalableEnergy(
        SalableEnergy(:,1)==sold_id,7);
    sold_excess = min(sold_amount,
        salable_excess);
    sold_amount = sold_amount-sold_excess;
    sold_battery = min(sold_amount,
        salable_battery);
    sold_amount = sold_amount-sold_battery;
    sold_ev = min(sold_amount, salable_ev);
    sold_total = sold_excess + sold_battery +
        sold_ev;
    sold_allocated(sold_id,:) = [sold_id,
        sold_total, sold_excess, sold_battery,
        sold_ev];
    end
end
end
end

```

APPENDIX H DESCRIPTIVE STATISTIC (UTILITY)

```
function DataTable = Descriptive(data)
    mean_ = mean(data);
    median_ = median(data);
    max_ = max(data);
    min_ = min(data);
    range_ = range(data);
    variance_ = var(data);
    std_ = std(data);
    percentile_ = prctile(data,[25 50 75],1);
    percentile25 = percentile_(1,:);
    percentile50 = percentile_(2,:);
    percentile75 = percentile_(3,:);
    iqr_ = iqr(data);
    skewness_ = skewness(data);
    total_ones = sum(data(:,:)==1);
    total_zeros = sum(data(:,:)==0);
    DataTable = splitvars(table(...
        [mean_;median_;max_;min_;range_;variance_;std_;
        percentile25;percentile50;percentile75;iqr_;
        skewness_;total_ones;total_zeros], ...
        'RowNames',["Mean","Median","Max","Min","Range",
            "Variance","Std","P25","P50","P75","IQR","
            Skewness","Ones","Zeros"]));
end
```