

UNIVERSITY OF BAHRAIN

COLLEGE OF ENGINEERING

DEPARTMENT OF MECHANICAL ENGINEERING



**Integration Advanced Artificial Intelligence (AI) Tools in
Energy Management Systems in Enhancing Geographical
Information and Decision Based System**

A Thesis Submitted in Partial Fulfillment of the Requirements for the
Ph.D. Degree in Engineering Management

Prepared by

Aaed Khalil Shahada

University number: 20061469

Supervised by

Professor Ebrahim Abdulla Mattar

University of Bahrain

Kingdom of Bahrain

May 2024

بسم الله الرحمن الرحيم

Abstract

Effective energy management is crucial once related to constructing and licensing utilities, residential homes, and other civil structures. Due to the complexity of these interconnected and complicated processes, it has historically been challenging to appropriately manage these relationships. However, in recent years, artificial intelligence algorithms and models have been extensively utilized to help streamline and optimize energy management. To automate this process, a model using pretrained Deep Learning model is designed. Subsequent to training, validation and testing cycles were completed, 15 floorplans were inserted to the model and generated results in terms of decision-making. Manual calculations for identical floorplans were obtained by an expert engineer considering the length and the width for each floorplan using an AutoCAD file. The algorithm was designed to find out the areas in square meters of the outer surface of the floor plan for the opened surface such as windows or doors or for a solid surface such as block, concrete, steel, or wood. For attaining temperature deviation for buildings, hence, to calculate floor plan heat, real data temperature was downloaded which contain more than (1754 station) that send live temperature data each three hours of time. Heat gains were calculated for (15 floorplans) by using the data obtained earlier in the U-Values, outer surface areas and temperature deference's. Results of the algorithm were verified by comparing the algorithm output and the manual calculation by the Engineering office. All the results obtained by the algorithm were very accurate when comparing them with the manual generation by the engineering offices making the process automated and eliminating human interference. The results indicated an error of (0.35%) for the actual dimension and (0.69%) in the heat gain calculation for the 15 floorplans once compared them with manual calculation.

PUBLICATIONS AND PARTICIPATIONS

1. Deep Learning for Building Construction Floorplans

4th Smart Cities Symposium (SCS 2021)

Location: Conference, Bahrain, Conference date: 21-23 November 2021

ISBN: 978-1-83953-658-8, Conference number: CP791, Publisher: IET.

2. Deep Learning for Construction Floorplan Recognition

Date Added to IEEE Xplore: 29 May 2023

Electronic ISBN:978-1-83953-854-4, INSPEC Accession Number: 23026795

DOI: 10.1049/icp.2023.0650, Publisher: IET.

3. GIS Decision Based Systems in Energy Management Systems

DOI: 10.1049/icp.2024.1037

ISBN: 978-1-83953-993-0

Location: Hybrid Conference, Bahrain

Conference date: 3-5 December 2023, Publisher: IET

4. Artificial Intelligence and Energy Management Systems for Enhancing GIS and Decision Based System.

(Submitted for journal publication).

5. Exploring Integration of Energy Management Systems with GIS Systems.

(Submitted for journal publication).

Dedication

To my parents, wife, family, and friends

Acknowledgments

I would like to take this opportunity to express my deep thanks and gratitude to my supervisor, Professor Dr. Ebrahim Mattar, who continuously guided me throughout the research with unlimited support and considered me as one of his top priority tasks.

My sincere thanks to University of Bahrain and all staff members for their unlimited support during this journey.

My thanks to my parents who supported me from the start of the research until the end.

Many thanks also to my wife, family members, relatives, friends, colleagues, and to my department and college who supported me in one way or another.

My thanks go to my direct work supervisor in my working place Rasha Kameshki for her patience and continued support.

I would also like to thank Dr. Mayyadah Shahada for being always a side and giving me motivation and unlimited support throughout the research.

Special thanks go to My Father Khalil Shahada, for encouraging me to join the PhD Program at the University of Bahrain and his unlimited support.

My sincere thanks to my wife Lama Meshal for being very supportive and being always next to me throughout the research. Much appreciation to my kids Asim and Yasmeen for inspiring and motivating me.

Table of Contents

Thesis Discussion Committee Decision.....	
Abstract	B
Publications and Participations	C
Dedication	D
Acknowledgments.....	E
Table of Contents	F
List of Tables	H
List of Figures	J
Nomenclature	M
Chapter 1 Introduction.....	15
1.1 General.....	15
1.2 Research Problem	18
1.3 Research Questions.....	19
1.4 Research Objectives	19
1.5 Research Significance.....	20
1.6 Research Scope	22
1.7 Research Outline.....	22
Chapter 2 Literature Review.....	24
2.1 Learning Algorithm for Floorplan Recognition	24
2.2 GIS for Temperature extraction.....	34
2.3 Heat Transfer and U-Value.....	37
Chapter 3 Machine Learning Algorithms (Conventional Neural Network).....	41
3.1 Introduction	41
3.2 CapsuleNet.....	43
3.3 Overfitting.....	45
3.4 CNN architecture	46
3.5 Overall architecture	46
3.6 Convolutional layer	48
3.7 Pooling layer.....	53

3.8	Recipes.....	55
3.9	CNN layers Details	56
Chapter 4	Floorplans Dataset Topology	74
4.1	Introduction	74
4.2	Floorplans and existing datasets	74
4.3	CubiCasa5K: A Novel Floorplan Dataset	76
Chapter 5	Floor Plan Framework and Code Adherence Detection Process (Floorplan Recognition)	80
5.1	Introduction	80
5.2	Algorithm Generation Steps	81
5.3	Law and Regulation in Floorplan	86
5.4	Training of the Model.....	87
5.5	Model Execution and results generation	90
5.6	Actual dimensions of floorplan	102
5.7	Outer Surface Area Extraction	113
Chapter 6	Floor Plan Heat Loss Management by Using GIS Data	122
6.1	Heat Loss in Building and U-Value	122
6.1.1	Thermal Insulation Regulation in Bahrain	126
6.2	GIS and Temperature Deference	128
6.3	Floor plan Heat Loss Calculation	132
6.3.1	Overview Workflow	135
Chapter 7	Results and Discussion	138
7.1	Result Discussion	138
7.2	Outer Surface Area Extraction Error	156
7.3	Floor plan Heat Loss Calculation Error.....	172
7.4	Visual Management of the Heat Loss Factor	175
Chapter 8	Conclusions.....	178
8.1	Conclusions	178
References		183

List of Tables

Table 1. Areas for each room and the plot.....	85
Table 2. Number of cycles per each stage during training the model.....	90
Table 3. Actual dimension for plan no. 1.....	102
Table 4. Actual dimension for floor plan no. 2.....	103
Table 5. Actual dimension for floor plan no. 3.....	104
Table 6. Actual dimension for floor plan no. 4.....	105
Table 7. Actual dimension for floor plan no. 5.....	105
Table 8. Actual dimension for floor plan no. 6.....	106
Table 9. Actual dimension for floor plan no. 7.....	107
Table 10. Actual dimension for floor plan no. 8.....	107
Table 11. Actual dimension for floor plan no. 9.....	108
Table 12. Actual dimension for floor plan no. 10.....	109
Table 13. Actual dimension for floor plan no. 11.....	109
Table 14. Actual dimension for floor plan no. 12.....	110
Table 15. Actual dimension for floor plan no. 13.....	111
Table 16. Actual dimension for floor plan no. 14.....	111
Table 17. Actual dimension for floor plan no. 15.....	112
Table 18. Surface area of floor plan no. 1.....	113
Table 19. Surface area of floor plan no. 2.....	114
Table 20. Surface area of floor plan no. 3.....	114
Table 21. Surface area of floor plan no. 4.....	115
Table 22. Surface area of floor plan no. 5.....	115
Table 23. Surface area of floor plan no. 6.....	116
Table 24. Surface area of floor plan no. 7.....	116
Table 25. Surface area of floor plan no. 8.....	117
Table 26. Surface area of floor plan no. 9.....	117
Table 27. Surface area of floor plan no. 10.....	118
Table 28. Surface are of floor plan no. 11.	118
Table 29. Surface area of floor plan no. 12.....	119
Table 30. Surface area of floor plan no. 13.....	119
Table 31. Surface area of floor plan no. 14.....	120
Table 32. Surface area of floor plan no. 15.....	120
Table 33. Maximum of Thermal Transmittance Value U Value.	128
Table 34. Calculation of heat loss for the floor plans.	133
Table 35. Results compression for floor plan no. 1.	139
Table 36. Results for floor plan no. 2	140
Table 37. Results for floor plan no. 3.	141
Table 38. Results for floor plan no. 4.	142
Table 39. Results for floor plan no. 5.	143

Table 40. Results for floor plan no. 6.	144
Table 41. Results for floor plan no. 7.	145
Table 42. Results for floor plan no. 8.	146
Table 43. Results for floor plan no. 9.	147
Table 44. Results for floor plan no. 10.	148
Table 45. Results for floor plan no. 11.	149
Table 46. Results for floor plan no. 12.	150
Table 47. Results for floor plan no. 13.	151
Table 48. Results for floor plan no. 14.	152
Table 49. Results for floor plan no. 15.	153
Table 50. Average percentage of error for each floorplan.....	155
Table 51. Surface areas of floor plan no. 1.	156
Table 52. Surface areas of floor plan no. 2.	157
Table 53. Surface areas of floor plan no. 3.	158
Table 54. Surface areas of floor plan no. 4.	159
Table 55. Surface areas of floor plan no. 5.	160
Table 56. Surface are of floor plan no. 6.	161
Table 57. Surface areas of floor plan no. 7.	162
Table 58. Surface areas of floor plan no. 8.	163
Table 59. Surface areas of floor plan no. 9.	164
Table 60. Surface areas of floor plan no. 10.	165
Table 61. Surface areas of floor plan no. 11.	166
Table 62. Surface areas of floor plan no. 12.	167
Table 63. Surface areas of floor plan no. 13.	168
Table 64. Surface areas of floor plan no. 14.	169
Table 65. Surface areas of floor plan no. 15.	170
Table 66. Average percentage of error for each floorplan.....	172
Table 67. % of error in heat loss between algorithm output and actual areas.	174

List of Figures

Figure 1. The network can recognize walls of nonuniform thickness (see boxes 2, 4, 5), walls that meet at irregular junctions (see boxes 1, 2), curved walls (see box 3), and various room types in the layout. Zeng et al. (2019).	25
Figure 2. Illustrates the overview of the framework. Yang et. al (2019).	29
Figure 3. Workflow of the study, showing the input data, process, and results. Jang et. al. (2020).	30
Figure 4. Typical layout structure of an ANN. (Keiron et. al 2015).).	41
Figure 5. The complete CapsNet encoding and decoding processes (Alzubaidi et. al 2021).	44
Figure 6. simplified CNN architecture for MNIST classification. (LeCun et. al 1989).	47
Figure 7. Activation roadmaps (Nebauer et. al 1998).	49
Figure 8. The network will learn kernels that 'fire' when they observe a specific feature at a specific spatial place in the insight (Nebauer et. al 1998).	50
Figure 9. Pooling layer for the floorplan.	54
Figure 10. Frequent CNN architecture is to stack two convolutional layers before each pooling layer (Srivastava et. al 2013).	56
Figure 11. Deep neural network with architecture.	57
Figure 12. CNN layers visualization.	61
Figure 13. Steps of Convolution Layer.	63
Figure 14. Steps of Maximum Pooling layer.	65
Figure 15. Flatten and fully connected layers, (Jeong, 2019).	66
Figure 16. Sigmoid and ReLU functions, (Sharma, 2022).	68
Figure 17. Plot of exp, (Priya, n.d.).	69
Figure 18. CNN Model Layers (Jeong, 2019).	72
Figure 19. Sample from CubiCasa5K dataset.	77
Figure 20. Pre-processing workflow for area's extraction.	80
Figure 21. The row input as a floorplan with all the data attached.	81
Figure 22. Removing all text labels and symbols to represent architectural elements and furniture.	82
Figure 23. Finding corner points and removing scaling box.	83
Figure 24. Doors & windows are closed.	84
Figure 25. The floorplans are well recognized in terms of rooms.	85
Figure 26. Algorithm training process three key stages.	89
Figure 27. Heat loss calculation workflow.	126
Figure 28. Live real data uploaded on arcGIS.	129
Figure 29. Weather stations in Gulf region.	130
Figure 30. Weather stations in United States.	131
Figure 31. Information of temperature and time in Bahrain Intl Arp station.	131
Figure 32. Heat Loss Factor for each Floorplan.	134

Figure 33. Heat Loss Factor for each Floorplan.	136
Figure 34. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 1.	140
Figure 35. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 2.	141
Figure 36. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 3.	142
Figure 37. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 4.	143
Figure 38. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 5.	144
Figure 39. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 6.	145
Figure 40. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 7.	146
Figure 41. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 8.	147
Figure 42. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 9.	148
Figure 43. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 10.	149
Figure 44. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 11.	150
Figure 45. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 12.	151
Figure 46. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 13.	152
Figure 47. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 14.	153
Figure 48. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 15.	154
Figure 49. Percentage of error in the surface area for floorplan No. 1.	157
Figure 50. Percentage of error in the surface area for floorplan No. 2.	158
Figure 51. Percentage of error in the surface area for floorplan No.3.	159
Figure 52. Percentage of error in the surface area for floorplan No.4.	160
Figure 53. Percentage of error in the surface area for floorplan No.5.	161
Figure 54. Percentage of error in the surface area for floorplan No.6.	162
Figure 55. Percentage of error in the surface area for floorplan No.7.	163
Figure 56. Percentage of error in the surface area for floorplan No.8.	164
Figure 57. Percentage of error in the surface area for floorplan No.9.	165
Figure 58. Percentage of error in the surface area for floorplan No.10.	166
Figure 59. Percentage of error in the surface area for floorplan No.11.	167
Figure 60. Percentage of error in the surface area for floorplan No.12.	168

Figure 61. Percentage of error in the surface area for floorplan No.13.	169
Figure 62. Percentage of error in the surface area for floorplan No.14.	170
Figure 63. Percentage of error in the surface area for floorplan No.15.	171
Figure 64. % of error in the surface area for floorplan No.15.	176

Nomenclature

CNN	Convolutional Neural Network
ANN	Artificial Neural Networks
FCN	Fully Convolutional Networks
GEE	Google Earth Engine
cGAN	Conditional Generative Adversarial Networks
MCTS	Monte Carlo Tree Search
AI	Artificial Intelligence
YOLO	You-Only-Look-Once
MRIS	Moderate Resolution Imaging Spectroradiometer
IWM	Irrigation Water Management
GIS	Geographical Information System
MAE	Mean Absolute Error
RMSE	Residual Mean Square Error
BES	Building Energy Simulation
PE	Polyethylene
PC	Polycarbonate
PVS	Polyvinyl chloride
HG	Horticultural Glass
QA	Quality Assurance

Chapter 1

Introduction

Chapter 1 Introduction

1.1 General

The design of a home is a fascinating challenge. Floorplans, which are diagrams of the rooms that make up a building, and their spatial relationships to one another, are used to design homes. Rooms are just rooms that have been constructed with a certain purpose in mind. Kitchens, dining rooms, bedrooms, and other rooms are examples of rooms in floorplans. Each of these rooms is designed to serve a certain purpose or group of purposes. Floor plans are surprisingly complicated, as they must accommodate a wide range of functional and aesthetic requirements.

When fulfilling these goals, there are several explicit and implicit rules that must be observed. For instance, a floorplan must explicitly contain a certain number of rooms and walls that separate or compose such spaces in order to effectively depict spatial relationships between them. While there are no explicit restrictions on how these rooms should be set up, there are some implicit principles that must be followed.

The placement of a bedroom within a kitchen might be inconvenient. While a design must follow these criteria, there is a lot of flexibility in how they are followed. This openness allows for an enormous number of different floorplans to satisfy a set of requirements (Such as number of rooms, size, etc.). Creating such floor plans is a creative process that involves both imagination and math skills. People may be taught to do the job reasonably well, but it is still a physically demanding operation. Floorplans are generated in a professional context through an iterative revision process between a customer and an architect. The architect will create rough drawings of the

structure, show them to the customer, receive feedback, then revise the design. During this phase, the architect must draw on years of architectural knowledge and experience to examine a wide range of factors, including lighting, heating, and aesthetics. This procedure is repeated until a rough design is agreed upon, after which the building is further developed. In the world of layout design, several trials are typically the key to success. When there are no obvious objective measures to optimize, the ideal technique is to generate a plethora of options and let the client choose from them. Creating multiple designs for a single success, on the other hand, is inefficient, and the significant amount of human work required to generate a single design result in expensive costs. Reducing the amount of human time required for each iteration would cut the overall process time in half. As a result, reducing the amount of human time required to create a plan, or possibly automating the process entirely, is ideal. Home design is both a professional and a personal endeavor.

Many non-architects/designers are looking to rebuild or create their home outside of professional home development. Hiring a private designer, on the other hand, can be rather costly, and many people lack the skill or enthusiasm in home design. The ability to democratize this expertise would be extremely valuable. Software automation is the most effective way to accomplish this. Non-professionals would be able to receive normally expensive architectural services at a lower cost if software could automatically or artistically design dwellings. Furthermore, such software could lower the expense of the time-consuming iterative procedure used in professional settings. While technologies like AutoCAD, Revit, and SketchUp exist to assist architects in the creation of floorplans, none exist that actively work alongside the creator

to intelligently develop the plan. Because it must obey the explicit and implicit norms of floorplan design, this program would necessitate a large level of intelligence. Because these criteria are often based on a variety of circumstances and are difficult to define, hard coding them would be nearly impossible. Furthermore, even if the option existed, this task would be extremely time-consuming and subjective. Instead of hard coding these rules into the software, it would be preferable if it could learn them from existing examples. Computer vision applications, sequential models, and adversarial networks are some of the potential answers in the rapidly increasing field of machine learning. Finding and implementing the right answer, on the other hand, is not straightforward. While many individuals understand architectural standards, even the most essentials, such as the presence of walls, rooms, and doors, can be difficult for some machine learning models to comprehend.

A machine learning model must also understand nuances like common room adjacencies and how the sizes of different rooms relate to each other. Finally, due to the model's lack of knowledge of the semantics or intrinsic structure of floorplans, merely describing the problem in a form that a machine learning model can understand is problematic.

Utilizing models created for computer vision tasks as a starting point for using ML to solve this challenge is one option. These models make decisions based on visuals. A floor plan can be represented as a picture, but this offers several recognizing challenges that leading computer vision systems would struggle to address. To begin, the model must understand walls and how they are

placed to make rooms in a specific pattern (not all rooms contain four walls or are shaped uniformly). The model would next have to learn about the many types of rooms' attributes. Finally, the model must learn how different room types are placed in relation to one another. While learning the placements and characterizations of rooms is inescapable for any agent learning to build floorplans, it is particularly difficult when learning the concept of such objects purely from pixels.

A floorplan can also be expressed as a list of vectors, each of which carries information on one of the objects in the plan. The dimensionality of the object recognition problem is reduced with this approach. Rather than learning object characteristics by looking at hundreds of pixels, a model can learn object characteristics by looking at information contained in vectors such as size, end-point coordinates, object type, and so on.

This representation is promising, but it faces additional issues when considering the changing length of each floorplan and how the model will need to understand the positional relationships between each object, which may not be expressed simply through a list of vectors. To encode structural information in floorplans, use graphs.

1.2 Research Problem

How to design a deep learning model using a convolutional neural network (CNN) in order to be able to recognize floorplans for the automation of the process of licensing floorplans based on predefined criteria's related to local municipality standards and regulations. How significant is the effect of this model on the energy management of the building.

1.3 Research Questions

- What are the criteria to design a deep learning model?
- What is the impact in terms of duration on the current process of licensing floorplan after eliminating the human interference by automating this process?
- How accurate the results are after automating the process of reading floor plans?
- Can we eliminate human interference in the process of recognizing and licensing floorplans?
- Can we establish an early energy management tool that contributes positively in energy saving by taking an efficient discussion on floorplan design.

1.4 Research Objectives

- To create a model which is able to recognize and extract geometric information automatically from floorplans.
- To program the model to be able to take automated decisions related to the predefined standards and parameters on the floorplans.
- To eliminate all human errors and biases during reading floorplans.
- To improve the performance of governmental services by speeding up the process of reading and approving floorplans.
- To create a model that generates an early management tool for efficient decision making on the design of the floorplan.

1.5 Research Significance

Human error in critical decisions might cause major consequences in aspect of time and cost. Despite respected experience and many validations added to minimize human error, it is almost impossible to eliminate human error in processes where the human interaction is involved. Consider the scenario of approving incorrect contract drawing for a building where the client has started the construction work already. During the inspection stage to grant approvals, the client will be asked to do the physical modification to meet the correct standards. This scenario can be eliminated in case of using deep learning algorithms for licensing floor plans (Dong et. al. 2011).

The value of time now a day in project management is expressed with a considerable amount of money in case of delays. Penalties might be applied to the concerned delayed party and their reputation will be damaged severely. Nevertheless, introducing a deep learning algorithm to speed up the process of licensing floor plans and enhance the process can be of great value added to the system of building permits approval (Jamil et. al. 2017)

The capability of a human varies from one person to another. Some employees can work to achieve a significant number of applications such as reviewing floor plans and giving the needed comments with the decision of approving or rejecting the floor plans. Age, experience, number of employees and qualifications play a very important role in completing all the received floor plans applications on time with providing the required feedback. These criteria are mandatory, and they require extra cost to maintain their availability inside the entity. As a replacement, the deep learning algorithm can be introduced to eliminate the required criteria with less cost and higher efficiency (Christof et. al 2016).

Human biases are well-documented, from implicit association tests that demonstrate biases we may not even be aware of, to field experiments that demonstrate how much these biases can affect outcomes and make uncomfortable environment in the societies. Deep learning algorithm or any advanced computational methods can't identify the background of the customers unless they were programed for it. They treat all applications equally and eliminate any biases in their decision (Christof et. al 2016).

Moreover, the standardization factor in the decision is very important to ensure all the exceptional approvals were given equally. An exceptional decision can be given by one employee based on his understanding of the situation while the other employee might reject it. These discrepancies in the approvals for the exceptional cases affect the entity image on their customers. Advanced computational methods can overcome these discrepancies by unifying the exceptional approvals for all the cases (Aiswarya et. al. 2019).

Energy management is one of the important factors which needs to be reviewed in floorplans by engineers. Without having a full and comprehensive view of the energy management in each floorplan, energy saving can't be maintained.

When it comes to the reliability of the entity processes, it is very important to have a reliable sold system that gives the same results for deferent transactions. Human behavior varies depending on the current situation and status of the employee. Reliability related to human decision can't be achieved due to the different natural of humans. Introducing the advanced computational method can achieve the reliability of licensing building permits (Aiswarya et. al. 2019).

1.6 Research Scope

The scope of this research starts with identifying the criteria that are required for approving a new floorplan by the local municipality with all their standards and requirements. Considering energy saving as a part of those criteria in order to enhance the process of approving the floorplan by considering the heat gain in floorplan. Correspondingly, the design of the model using pretrained CNN model will take place by considering the predefined criteria. Training of the model by using a dataset of floorplans in order to enhance the ability of the model by viewing a significant number of floorplans and try to read all parameters in those floorplans such as rooms, halls, doors, walls, bathrooms and etc.. Verification of the model is required to insure the validity of the criteria during the functionality of the model. Finally, achieving the research objectives by evaluating each point and giving recommendations for further work and research.

1.7 Research Outline

The thesis comprises (eight) chapters. CHAPTER 1 outlines the introduction to the study. CHAPTER 2 discusses the theoretical framework and provides a literature review of deep learning in floorplans. CHAPTERS 3 AND 4 explain the CNN background and the datasets for this research. CHAPTER 5 describes the methods and procedures. The results and discussions are illustrated in CHAPTER 6, CHAPTER 7 provides the results and finally CHAPTER 8 stipulates the conclusion.

Chapter 2

Literature Review

Chapter 2 Literature Review

2.1 Learning Algorithm for Floorplan Recognition

Floor plans serve as more than just visual representations of room designs; they provide valuable information about room orientation, wall types, and furniture arrangement. Therefore, it is crucial to thoroughly understand floor plans before undertaking layout design. While humans can easily recognize semantic information in floor plans, automatically analyzing floor plans and identifying layout semantics is a challenging task in image interpretation and document analysis.

Zeng et al. (2019) and Kang et al. (2017) have explored the process of distinguishing room-boundary and room-type elements in floor plans. They have investigated the spatial relationships between floor plan elements, constructed a hierarchy of floor plan elements, and developed a multi-task network. By integrating a spatial contextual module with a room-boundary-guided attention mechanism, they utilized room-boundary features to guide room-type prediction, including the recognition of walls with non-uniform thickness, as depicted in Figure 1. Additionally, they created two datasets for floor plan recognition and extensively evaluated the network from various aspects, demonstrating its superior performance in terms of overall accuracy. Hsieh et al. (2017) have also suggested future research directions, including extracting dimension information from floor plan photos and learning to recognize text labels and symbols in floor plans.

Ferrando (2018) introduced a Machine Learning framework as an approach to infer architectural traits using quantifiable spatial elements. Other studies have

focused on determining the intimacy and seclusion of different rooms in a house plan. Patterns were discovered between spatial attributes and the



Figure 1. The network can recognize walls of nonuniform thickness (see boxes 2, 4, 5), walls that meet at irregular junctions (see boxes 1, 2), curved walls (see box 3), and various room types in the layout. Zeng et al. (2019).

perceived level of spatial closeness, as evaluated by a panel of experts in a validation exercise. Ferrando proposes that utilizing Machine Learning on a larger dataset could yield more precise predictions regarding the level of intimacy in floor plan areas. Building an extensive dataset of architectural spaces classified based on a comprehensive vector of spatial attributes, to train a Deep Neural Network for privacy classification, is a future advancement suggested by this study.

The Neural Network framework has a diverse impact on spatial analysis, as explained by Heras et al. (2014). Depending on the dataset used as input, Machine Learning algorithms can yield various types of new knowledge.

Analyzing a randomly selected dataset enables the inference of general principles and attributes that are consistent across cultures. Conversely, using a culturally or type-specific dataset leads to the acquisition of specialized knowledge about the specific culture or type. Therefore, the nature of the dataset must be carefully considered when interpreting the outcomes of a Machine Learning model. The analytical benefits of these new approaches primarily extend to academics, architects, and designers working with spatial analysis, with the potential for long-term utilization in the form of an intelligent design assistant.

To enhance the understanding of privacy levels in floor plans, two significant future steps are suggested. The first step involves incorporating 3D data into the analysis, which could be obtained automatically by a robotic device traversing a structure, either in real life or through simulation. This would provide a larger number of descriptors for the model. The second step entails gathering survey comments from a large sample of individuals regarding the perceived seclusion of rooms in a floor plan. According to Stahl et al. (2018), this approach could offer a less predictable method for determining the level of privacy in each room, replacing the reliance on a single expert evaluation with evaluations from a broader sample of people. These advancements have the potential to improve spatial analysis of floor plans and assist researchers in describing abstract architectural attributes using quantifiable spatial elements.

Du et al. (2020) conducted a comprehensive review of four advanced machine learning methods for spatial data classification and regression. These methods include kernel learning for solving nonlinear and high-dimensional problems, semi-supervised and active learning for addressing challenges with small

training sample sizes, ensemble learning for combining different learners' advantages and enhancing robustness, and deep learning for extracting hidden high-level features. Based on technical reviews and experimental analyses, it can be concluded that machine learning methods are suitable for overcoming challenges in spatial data handling and improving classification, regression, and inversion problem performance. These methods are expected to play an increasingly important role in the future.

Du et al. (2020) and Dodge et al. (2017) also highlight other directions in machine learning for spatial data handling that should be explored in the future. Firstly, embedding spatial laws and geographical knowledge in machine learning algorithms would enhance their capabilities. Secondly, improving the efficiency of learning and computing can be achieved through parallel computing or cloud computing platforms like Google Earth Engine (GEE). Lastly, fusing the advantages of multi-source data and different learning algorithms in open and complex geographical environments holds potential for further advancements.

Kim (2021) introduces a technique for automatically extracting vectorized indoor spatial information from floor plan photographs. This involves training a deep learning system on scanned floor plan images to extract important indoor elements such as room structures, junctions, walls, and openings. The technique is capable of handling complex floor designs that previous studies were unable to automatically extract. By comparing this newly discovered technology to manual digitization, Kim's research presents a methodology for evaluating its technical value. The study quantifies the hierarchical dimensions of technological value, such as substitutability, completeness, and supply and demand, using the analytical hierarchy process. The findings

demonstrate the technical value of automatic floor plan analysis using a deep learning algorithm, systematically and qualitatively comparing it to typical manual editing. Rosser et al. (2017) and Yang et al. (2011) explore the use of learned building measurements to create interior design plans for buildings. Their method focuses on building a predictive model by learning the geometry, orientation, and incidence of rooms from real-world building plan data. This predictive model is then used to estimate room dimensions and orientations, which are combined with the overall footprint of the building to iteratively create comprehensive interior layouts. The method is described as a semi-automatic approach that can handle limited data and generate likely representations of building plans using soft and hard constraints modeling. The study demonstrates the effectiveness of this method in predicting residential house designs with minimal prior information of room types and topology.

Yang et al. (2019) presents the DuLa-Net, a deep learning framework for predicting Manhattan-world 3D room layouts from a single RGB panorama presented in Figure 2. The method leverages two projections of the panorama, the equirectangular panorama-view, and the perspective ceiling-view, to improve prediction accuracy. The network architecture consists of two encoder-decoder branches for analyzing each of the two views, and a feature fusion structure connects the branches to jointly predict 2D floor plans and layout heights. To handle more complex room layouts, Chen et al. (2018) introduce the Realtor360 dataset, containing panoramas of Manhattan-world room layouts with varying numbers of corners. Experimental results demonstrate that the proposed approach outperforms state-of-the-art methods in terms of prediction accuracy and performance, particularly for rooms with

non-cuboid layouts classified according to a long enough vector of spatial attributes, in order to train a Deep Neural Network for the goal of privacy classification, is a future advancement of this study.

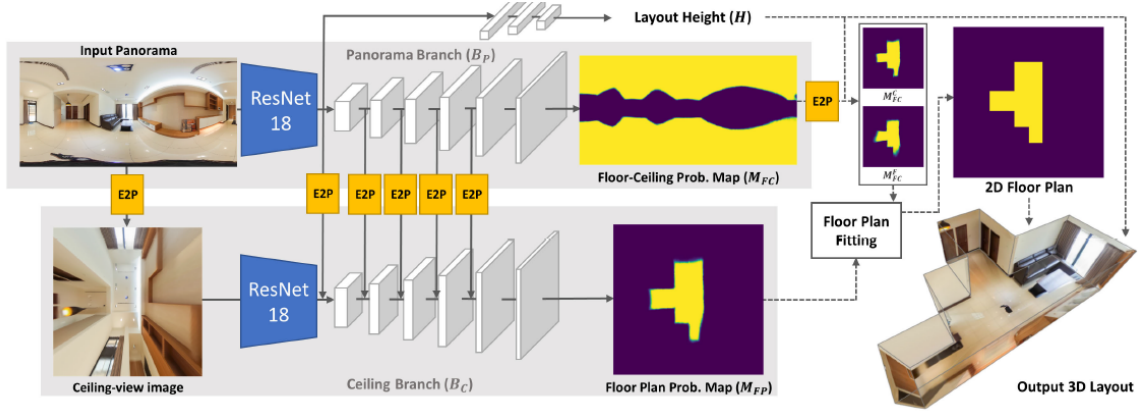


Figure 2. Illustrates the overview of the framework. Yang et. al (2019).

Jang et al. (2020) presented a method for automatically extracting vector data from floor plan images, which serves as the foundation for generating indoor spatial information. The paper's main contributions can be summarized in three parts. Firstly, they successfully recovered vector data from floor plan images, enabling the generation of indoor spatial information using CityGML or the cell space of the IndoorGML thick wall model. Previous studies were unable to achieve this using image-based approaches. Secondly, they employed a modern Convolutional Neural Network (CNN) architecture to generate high-quality segmentation results, producing reliable node-edge graph-formed vector data with thickness attributes. Figure 3 illustrates this. Thirdly, they developed centerline detection algorithms that allowed for the recovery of both orthogonal and diagonal lines.

In their study, Jang et al. focused on walls and doors as the primary objects of interest for the spatial data, which had already been extensively discussed in

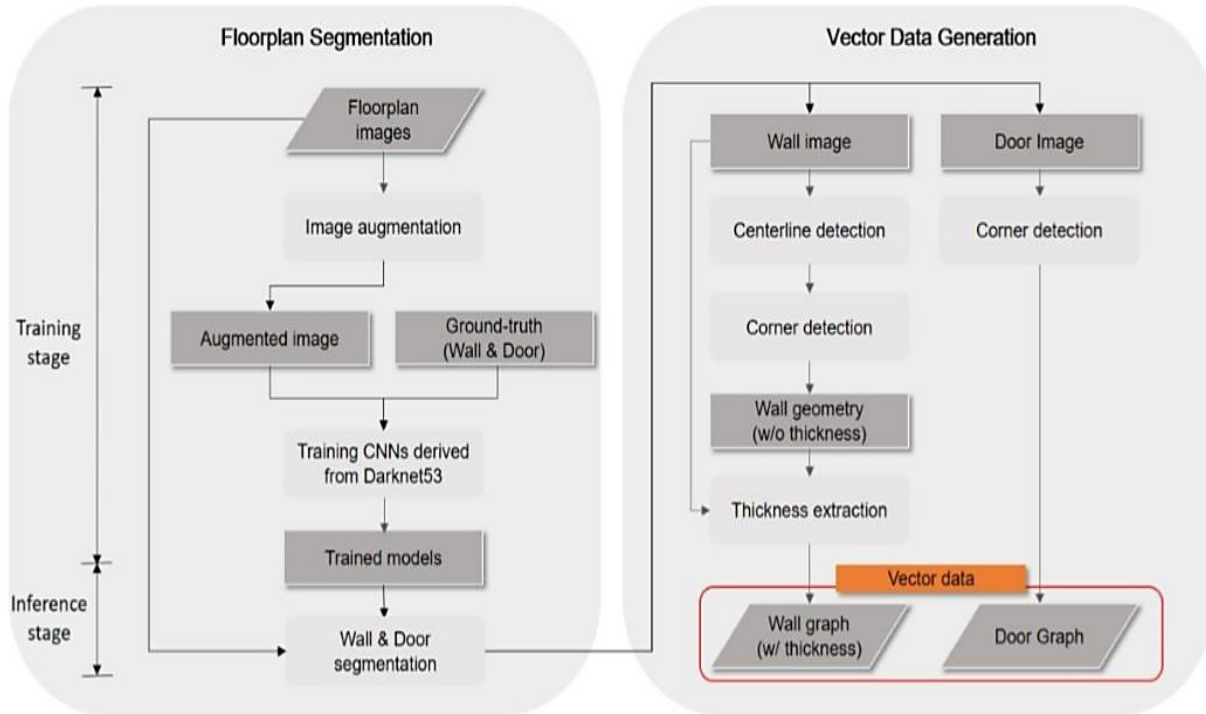


Figure 3. Workflow of the study, showing the input data, process, and results. Jang et. al. (2020).

Genova et al.'s (2020) research. However, they acknowledged that other objects such as windows and stairs could also play a crucial role and recommended their inclusion in future studies to generate semantically rich data. Additionally, they noted that the coordinate system used represented relative coordinates and couldn't be visualized in widely used viewers. They stressed the importance of considering and documenting the coordinate system that connects the real world and the generated data. From a technical perspective, they suggested removing nodes on diagonal edges to prevent the representation of single edges with more than two nodes, as mentioned in Hoiem et al. (2017).

Cho et al. (2020) described the use of deep-learning-based style transfer techniques to recognize floor plans by categorizing the plan's essential objects.

Previously, recognizing floor plans during the design and remodeling phases required significant time and professional interpretation. The goal was to enable computers to interpret the symbols in the plan and extract information from photographed architectural plans. However, due to the plan's varied conditions, computers faced challenges in directly obtaining information from them. The proposed approach employed Conditional Generative Adversarial Networks (cGAN) to convert existing plans into a coherent and easily understandable format, improving their readability. The dataset consisted of approximately 100 floor plans obtained from Sun et al. (2015), which were previously constructed by the Ministry of Land, Infrastructure, and Transport of Korea. The approach involved two steps: (1) defining the important objects in the floor plan for extraction and (2) using the defined objects as training input data for the cGAN style transfer model. In this paper, wall, door, and window objects were selected as the targets for extraction. The preexisting floor plans were segmented into each part and transformed into a consistent format, facilitating the automatic extraction of information for further utilization.

Ruwanthika et al. (2017) aimed to generate complete 3D models of houses for augmented reality applications. Their approach involved using the Harris corner detector to recognize walls and corners in blueprints. However, they encountered challenges when dealing with hollow outside walls, which were detected as two separate walls. To address this issue, the authors developed a special method to eliminate the unnecessary lines. Much of the research discussed in the review relied on classical image processing methods and simplified plans or scans, except for Ronneberger et al. (2015), which introduced a new deep learning model for extracting information from floor

plans, whether they were (2D) or (3D). The use of neural networks in image processing has seen a resurgence since 2017, but prior to that, traditional approaches were favored due to the limitations of AI during previous "AI winters."

Dodge et al. (2017) and Fukano et al. (2016) utilized Fully Convolutional Networks (FCN) to parse architectural floor plans, similar to the objective of Ruwanthika et al. Their focus was on handling simplified plans used in real estate websites, as traditional methods were unable to effectively extract geometric and semantic information from them. They evaluated different sizes of FCNs using a dataset provided alongside the paper to identify wall pixels. Additionally, they explored traditional and statistical approaches and employed the Google Vision API to remove text elements that could interfere with the analysis. The authors highlighted automatic 3D model creation as a potential application with relevance to real estate and interior design.

Liu et al. (2017) used a ResNet-based CNN instead of an FCN to detect wall junctions and other points of interest, icons, walls, and functionally defined areas such as living rooms and bedrooms. The resulting heat maps were post-processed and vectorized, although the technique was limited to Manhattan geometry without angled walls. The geometry was represented as a linear programming problem, where each room required a closed loop around the junction points that formed its walls.

Kalervo et al. (2019) built upon the work of Liu et al. by retraining the model using a larger labeled archive called CubiCasa5k, which they also made publicly available. They rewrote all the code in Python and included a post-processing algorithm limited to Manhattan geometry, although it is not specified whether there were algorithmic changes. Their dataset included

5000 colorful and monochrome raster scans, which had been manually recreated and labeled in SVG format. They provided a tool for converting these labeled SVGs into image tensors for training and evaluating the model.

Divya et al. (2017) focused on the automatic retrieval of previous building layouts from a repository for design reuse and faster project completion. They proposed using a deep architecture for layouts and introduced DANIEL, which utilized a deep learning framework to retrieve related floor plan layouts from a repository. The authors analyzed the performance of different layers of deep convolutional neural networks for floor plan retrieval and developed a complex dataset called ROBIN (Repository of Building Plans) consisting of three main categories and 510 actual floor plans. They conducted extensive studies on ROBIN and compared their findings with eight different state-of-the-art approaches. Sinisa et al. (2021) proposed a unique technique for reconstructing floor layouts from noisy 3D point clouds. They projected the input point cloud to a top view, generated a density map, and extracted room proposals from it. The method simultaneously optimized the polygonal shapes of the room proposals to fit the density map, resulting in an accurate vectorized floor map. They employed the Monte Carlo Tree Search (MCTS) algorithm, originally developed for game learning, to efficiently maximize an objective function based on fitness and predicted density maps from a deep neural network. The authors also introduced an MCTS refinement stage to modify the shapes of the room proposals using a differentiable method. The technique demonstrated significant improvements over state-of-the-art methods when tested on challenging datasets.

Alireza et al. (2022) and Chen et al. (2018) focused on symbol identification on digital architectural floor plans using a deep learning-based system.

Traditional symbol spotting techniques faced challenges due to graphical notation variability, poor intra-class symbol similarity, occlusion, clutter, and the complexity of symbols. The authors addressed these issues by modifying a You-Only-Look-Once (YOLO) architecture-based object detection framework and utilizing deep learning advancements. They used tiles as the basis for the training method to overcome challenges specific to DL-based object identification networks. The proposed technique successfully detected architectural symbols with varying graphical complexity and low intra-class similarity, even in the presence of opacity and clutter. The strategy outperformed existing symbol detection techniques and could handle different levels of deterioration and noise.

Chuhang et al. (2019) and Ruipeng et al. (2014) conducted further testing on the public SESYD dataset to detect architectural symbols on floor plans. Their techniques demonstrated successful symbol detection in various scenarios.

2.2 GIS for Temperature extraction

El Kenawy et al. (2010) conducted a study to generate accurate climate maps for Egypt by modeling and mapping monthly maximum and minimum temperatures as well as total precipitation. They developed a multivariate linear regression model with spline interpolation and collected climate data from 40 quality-controlled and homogenized series spanning the years 1957 to 2006. The predictors, including geographical data and remote sensing indices from the Moderate Resolution Imaging Spectroradiometer (MODIS), were integrated as raster layers in a Geographical Information System (GIS) environment. The inclusion of remote sensing indices, such as the Normalized

Difference Vegetation Index and the Normalized Difference Temperature Index, improved prediction accuracy. The model explained 76.5% and 51.7% of the spatial variability in maximum and minimum temperatures, respectively, when integrating geographic and remote sensing indices. For precipitation, the model explained 60.2% of the geographical variability throughout the year and 70.7% from September to April. The models' accuracy was evaluated using cross-validation, and the results, including coefficient of determination (R^2), Mean Absolute Error (MAE), and Willmott's D, showed excellent performance for maximum temperature (average MAE = 1.03°C) and total precipitation (average MAE = 2.73 mm).

Jahangiri et al. (2016) discussed the advantages of renewable energy over fossil fuels, such as affordability, accessibility, and low pollutant output. However, one drawback of renewable energy systems is the intermittent nature of power generation. To address this, the authors proposed the use of multiple renewable energy systems together or in synchronization with a generator to ensure continuous power supply. They also highlighted the limited accessibility of renewable energy systems in certain geographic locations. The study aimed to identify suitable sites for hybrid solar-wind power plants in the Middle East using a Boolean model in GIS software. The Boolean method, known for its strictness, was employed to ensure the selection of locations with high energy potential. Monthly weather data from 400 weather stations in the Middle East and RETScreen4 software were used for analysis. The findings aimed to facilitate the development of sustainable energy systems based on natural resources and support national power transmission and environmental regulations.

Andales et al. (2014) focused on improving irrigation water management (IWM) on irrigated farmland in the United States. They aimed to develop a mobile, scalable, device independent IWM system for Colorado, which could help conserve water, prevent pollution, and increase crop yield. The authors collaborated with crop growers, researchers, and advocates to create a prototype IWM system for Colorado, which was made available online. The system was integrated into the eRAMS GIS platform, providing online modeling services. It utilized daily soil water deficit estimates (Dc) to assist irrigators in planning irrigations in near real-time. The estimates were based on reference evapotranspiration (ET_r) and daily rainfall data from multiple networks in Colorado. Mobile applications were developed to display IWM data and interact with the eRAMS tool. The IWM approach was tested for three growth seasons (2010-2012) on a sprinkler-irrigated maize field in Colorado, and the daily root zone DC was determined. A comparison between estimated and measured values showed relative inaccuracies ranging from 2% to 31%, with an average Residual Mean Square Error (RMSE) of 16 mm of water. These errors were considered acceptable, and corrections based on measured profile soil water content were suggested to improve accuracy.

Matouq et al. (2013) additionally conducted a study to analyze weather data, including temperature and rainfall, from 1979 to 2008 in Jordan. They used geographical information systems (GIS) and the MATLAB program to simulate and visualize the data on geographical maps. The study examined the annual mean maximum temperature, annual mean minimum temperature, and annual mean rainfall for the specified time period. The analysis also included forecasts for the years 2009 to 2018. The results indicated that the mean annual rainfall in Jordan's northern and eastern regions remained unchanged,

while it increased in the central region. Temperatures in Jordan showed a significant increase since 1992, with a rise of 1.5 to 2 degrees. The study also observed significant changes in maximum and minimum temperatures since 1991, suggesting a turning point in Jordan's climate. The authors utilized GIS effectively to analyze the data and create user-friendly maps for understanding the impact of global warming, which was a first application of its kind in Jordan. The study's findings were expected to provide valuable insights for addressing climate change and developing appropriate strategies in the region.

2.3 Heat Transfer and U-Value

Infrared (IR) thermography is a reliable method for obtaining spatially resolved distributions of surface temperatures. It has a wide range of applications and can be used in various research issues and measuring settings with different physical configurations. Paris et al. (2011) presents the findings of using IR thermography to measure the overall heat transfer coefficient (U-Value) for building envelopes. The measured U-Values are validated using thermohygrometer measurements for both winter and summer seasons, as well as hypothetical values based on the applicable EN standard. The study also discusses the challenges posed by non-steady heat transport processes in building shells and conducts a sensitivity analysis to identify factors that significantly affect measurement accuracy (Paris et al. 2011).

Energy audits and performance assessments are increasingly important for existing buildings to conserve energy. Measuring the U-value of buildings is crucial in this context, and several methods are available for this purpose.

Design data, comparisons to similar structures, and field analyses like endoscopic investigations or logs can be used to measure the U-value more accurately. Heat flow meters can also be employed for in-place U-value testing, although they have longer execution times and greater measurement uncertainty. This work presents the results of an experimental campaign that examined the metrological performance of heat flow meters and their sensitivity to the surrounding environment. The actual U-values are compared to those estimated from design data and field analyses. The study shows that heat flow meters perform well when tests are conducted following ISO 9869, but their accuracy can be influenced by measurement settings and envelope component features (Giorgio et al., 2015).

A Building Energy Simulation (BES) model based on TRNSYS was developed to evaluate the U-value of greenhouse coverings made of different materials. The model considered the effects of inside-to-outside temperature differential, wind speed, and night sky radiation on the U-values of polyethylene (PE), polycarbonate (PC), polyvinyl chloride (PVC), and horticultural glass (HG). The inside and outside convective heat transfer coefficients were measured, and the model was calibrated using published data. The model's results were validated by comparing them to experimental results for a single layer of PE film. The study found that the U-value significantly changed with weather conditions and the use of single- or double-layer greenhouse covers. PC, PVC, and HG had lower U-values compared to PE, and there was a 34% decrease in heat gain with double glazing. The U-value increased with wind speed when the temperature was the same. The findings support previous research and indicate the accuracy

and potential usefulness of the provided model for greenhouse energy modeling (Adnan et al., 2017).

The selection of external convective heat transfer coefficients has a significant impact on building energy studies. Sensitivity analyses have shown that energy demand estimates can vary from 20% to 40% depending on these coefficients. This study used computational fluid dynamics simulations to predict convective heat transfer coefficients at the outside surfaces of a low-rise building with a simple design. The effects of wind speed, wind direction, and surface-to-air temperature disparities were analyzed. The results demonstrated that wind velocity has a significant impact on the convective heat transfer coefficient, wind direction affects vertical walls and roofs, and surface-to-air temperature differential has little influence on wind speeds above 2 m/s (Lee et al., 2016 & Mahaptra, 2015).

Chapter 3

Machain Learning Algorithms

(Convolutional Neural Networks)

Chapter 3 Machine Learning Algorithms (Conventional Neural Network)

3.1 Introduction

Artificial Neural Networks (ANNs) emulate the functioning of natural nerve systems, such as the human brain, in computational systems. ANNs consist of interconnected computing units called neurons, which collaborate in a distributed manner to process inputs and optimize their overall output.

The typical structure of an ANN, illustrated in Figure 4, can be represented by a model. Input data, usually in the form of a multidimensional vector, is received and propagated through hidden layers. These hidden layers make decisions based on information from preceding layers and evaluate how internal stochastic changes impact or enhance the final output, known as learning. Once vast multiple hidden layers are stacked together, it is commonly referred to as deep learning topology.

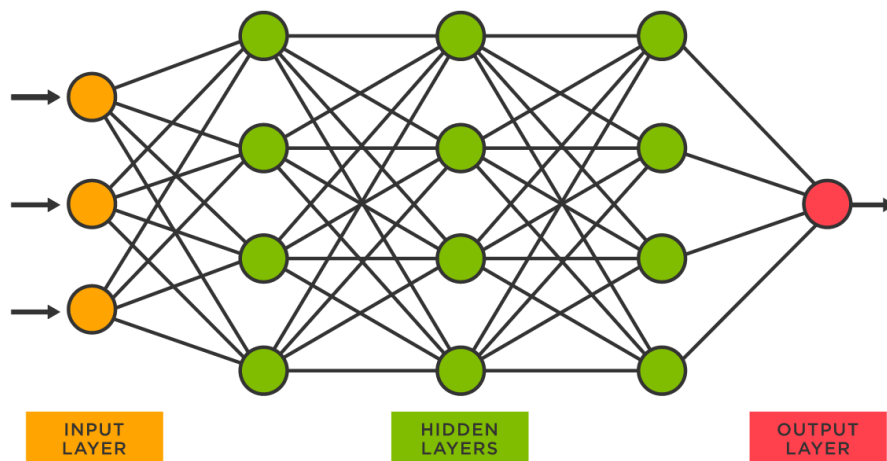


Figure 4. Typical layout structure of an ANN. (Keiron et. al 2015)

Image processing tasks often involve monitoring both supervised and unsupervised learning. Supervised learning refers to the use of pre-labeled inputs as targets. Each training sample consists of input values (vectors) and one or more corresponding output values. The objective of supervised learning is typically to accurately classify training examples based on their output values.

On the other hand, unsupervised learning differs from supervised learning in that the training set lacks labels. The success of unsupervised learning is often determined by the network's ability to minimize or maximize a connected cost function. However, it is important to note that most image-based pattern recognition tasks rely on supervised learning for classification.

Convolutional Neural Networks (CNNs) share similarities with regular ANNs in terms of their neuron-based self-optimization. Each neuron receives input and performs operations, such as scalar products followed by non-linear functions, which form the basis of many ANNs. The network expresses a perceptual score function from input image vectors to the final output, which is typically a weight. The final layer usually contains loss functions associated with different classes, and conventional techniques and guidelines for classical ANNs still apply (Keiron et al., 2015).

The main distinction between CNNs and regular ANNs lies in their application to image recognition tasks. CNNs incorporate image-specific features into their architecture, making them well-suited for image-related tasks while reducing the number of required parameters.

One significant limitation of older ANN types is their difficulty in handling the computational complexity associated with image data. Benchmark

datasets like the MNIST database, which consists of hand-written digits and has a relatively small image dimensionality of 28×28 , can be effectively used with most ANN types. In the case of MNIST, each neuron in the first hidden layer has 784 weights ($28 \times 28 \times 1$, considering the normalization to monochromatic values).

However, when dealing with larger, high-resolution image inputs, such as (64×64) , the number of weights on a single neuron in the first layer increases significantly to (12,288). Handling this level of input requires a considerably larger network compared to classifying normalized MNIST values, highlighting the limitations of employing these models.

3.2 CapsuleNet

CNN is an efficient technique for detecting object features and achieving well-behaved recognition performance in comparison with innovative handcrafted feature detectors. A few restrictions related to CNN are present, meaning that CNN does not consider certain relations, orientation, size, and perspectives of features. For instance, when considering a face image, the CNN does not count the various face components (such as mouth, eyes, nose, etc.) positions, and will incorrectly activate the CNN neurons and recognize the face without taking specific relations (such as size, orientation etc.) into account. At this point, consider a neuron that has probability in addition to feature properties such as size, orientation, perspective, etc. A specific neuron/capsule of this type has the ability to effectively detect the face along with different types of information. Thus, many layers of capsule nodes are used to construct the capsule network.

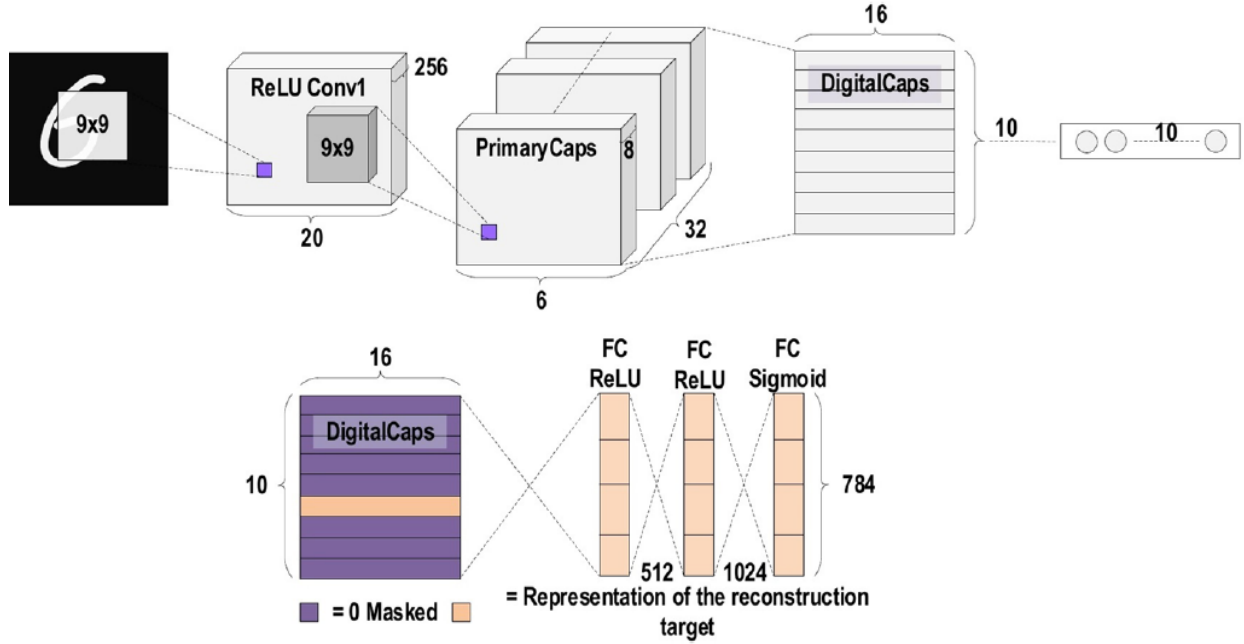


Figure 5. The complete CapsNet encoding and decoding processes (Alzubaidi et. al 2021).

An encoding unit, which contains three layers of capsule nodes, forms the CapsuleNet or CapsNet (the initial version of the capsule networks). For example, the MNIST architecture comprises 28×28 images, applying 256 filters of size (9×9) and with stride 1. The $28 - 9 + 1 = 20$ is the output plus 256 feature maps. Next, these outputs are input to the first capsule layer, while producing an 8D vector rather than a scalar; in fact, this is a modified convolution layer. Note that stride 2 with (9×9) filters is employed in the first convolution layer. Thus, the dimension of the output is $(20 - 9)/2 + 1 = 6$. The initial capsules employ 8×32 filters, which generate $32 \times 8 \times 6 \times 6$ (32 for groups, 8 for neurons, while 6×6 is the neuron size).

Figure 5 represents the complete CapsNet encoding and decoding processes. In the CNN context, a max-pooling layer is frequently employed to handle the translation change. It can detect the feature moves in the event that the feature is still within the max-pooling window. This approach has the ability to detect the overlapping features this is highly significant in detection and segmentation operations, since the capsule involves the weighted features sum from the preceding layer.

3.3 Overfitting

Can't we simply increase the number of hidden layers and neurons in the network? The straightforward answer to this question is generally no. There are two main reasons for this. Firstly, there are limitations in terms of processing power and training time, which prevent us from training extremely large ANNs. Secondly, we need to address the issue of overfitting. Overfitting occurs when a system fails to learn effectively due to various factors. It is a crucial concept in many machine learning algorithms, and it is important to take precautions to mitigate its effects. If our models exhibit signs of overfitting, it becomes more challenging to generalize the learned features not only to the training dataset but also to new analysis and prediction tasks. That is why it is essential to keep our ANNs as simple as possible. By minimizing the number of parameters required for training, we reduce the likelihood of overfitting, which, in turn, improves the predictive capability of the model (Krizhevsky et al., 2012).

3.4 CNN architecture

As mentioned earlier, CNNs specifically cater to the nature of image inputs, allowing their architecture to be tailored for optimal handling of this specific type of data.

One of the notable distinctions is the organization of neurons within CNN layers, which takes into account three dimensions: the spatial dimensions of the input (height and width) and the depth. The concept of depth is particularly interesting as it refers to the third dimension of any activation volume, rather than the total number of layers in the ANN. Unlike traditional ANNs, where neurons in a layer are connected to all neurons in the preceding layer, CNN neurons are only connected to a small portion of the previous layer. By employing this approach, the input "volume" takes on a dimensionality of $64 \times 64 \times 3$ (height, width, and depth), resulting in a final output layer with a dimensionality of $1 \times 1 \times n$ (where n represents the maximum number of classes). Effectively, the full input dimensionality is condensed into a smaller volume of class scores distributed across the depth dimension (LeCun et al., 1989).

3.5 Overall architecture

CNNs consist of three types of layers: convolutional layers, pooling layers, and fully connected layers. These layers are stacked together to form the CNN architecture. Figure 6 illustrates a simplified CNN architecture for MNIST classification. The basic functionality of the CNN in the example can be divided into four distinct parts:

The input layer stores the pixel values of the image, similar to other types of ANNs. Neurons in the convolutional layer compute the scalar product between their weights and the corresponding input region. The output of these neurons, often passed through an activation function like ReLU, captures features related to local regions of the input.

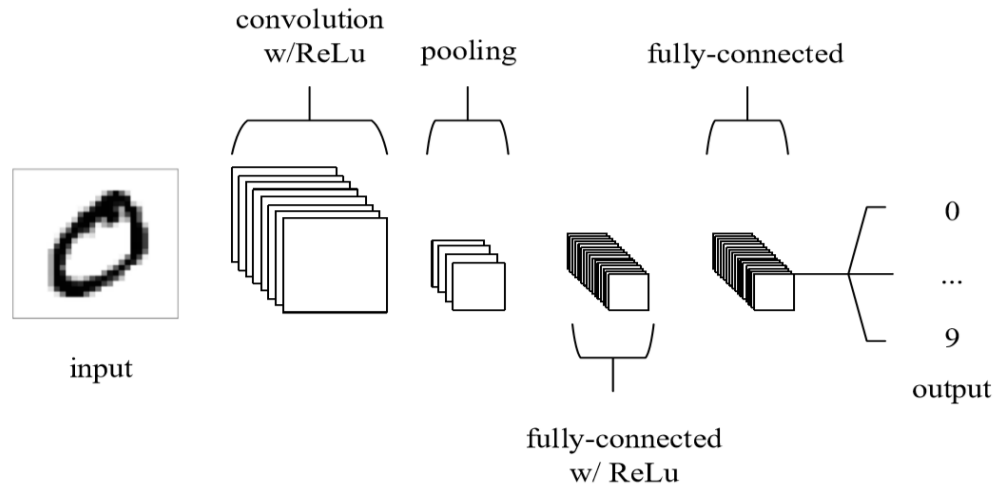


Figure 6. Simplified CNN architecture for MNIST classification. (LeCun et. al 1989).

The pooling layer down samples the spatial dimensionality of the input, reducing the number of parameters in the activation. Fully connected layers perform the same functions as regular ANNs, aiming to generate class scores for classification. It is recommended to use ReLU activation between these layers to enhance performance. By applying convolution and down sampling operations to the initial input layer, CNNs can transform it into class scores for classification and regression tasks. However, it is important to emphasize that understanding the overall structure of CNNs is not enough. Designing and optimizing these models can be time-consuming and challenging. In the following discussion, we will delve into each layer in detail, covering hyperparameters and connectivity considerations (LeCun et al., 1998).

3.6 Convolutional layer

The convolutional layer plays a crucial role in CNNs, as the name suggests. It utilizes learnable kernels with focused parameters. These kernels have a low spatial dimensionality but cover the entire depth of the input. When the data passes through a convolutional layer, each filter is convolved across the spatial dimension of the input, resulting in a 2D activation map. Figure 7 provides a visual representation of these activation maps.

For each value in the input, the scalar product is computed with the corresponding kernel. This process enables the network to learn kernels that "fire" when they detect specific features at specific spatial locations in the input. These occurrences are referred to as activations (Nebauer et al., 1998).

Each kernel generates its own activation map, and these maps are stacked together to form the complete output volume of the convolutional layer. As mentioned earlier, training ANNs on image data leads to models that are difficult to train effectively due to the fully connected nature of typical ANN neurons. This prevents each neuron in a convolutional layer from being partially connected to the input volume. The receptive field size of a neuron describes the dimensionality of this region.

The depth of the input is typically related to the level of connectivity in the depth dimension. For instance, if the input to a convolutional layer is a $64 \times 64 \times 3$ image (a 64×64 RGB colored image), and the receptive field size is $6 \times 6 \times 6$, each neuron in the convolutional layer would have a total of 108 weights ($6 \times 6 \times 3$, where 3 represents the connectivity in the depth dimension).

In contrast, a typical neuron in other types of ANNs would have 12,288 weights each. Convolutional layers effectively optimize the model's complexity by significantly reducing the number of weights. This optimization is achieved through three hyperparameters: depth, stride, and zero-padding.

The output volume's depth in convolutional layers can be manually adjusted by changing the number of neurons in each layer, corresponding to the same region of the input. This concept can also be observed in other types of ANNs, where all neurons in a hidden layer are directly connected to every neuron in the preceding layer. Reducing this hyperparameter can considerably decrease the total number of neurons in the network, but it may also slow down the model's pattern recognition abilities.

In general, each layer in a CNN performs an operation represented by the equation $\text{output} = f(W * \text{input} + b)$, where f is the activation function. To illustrate these techniques, let's consider a classification model with two hidden layers situated between the input and output layers.

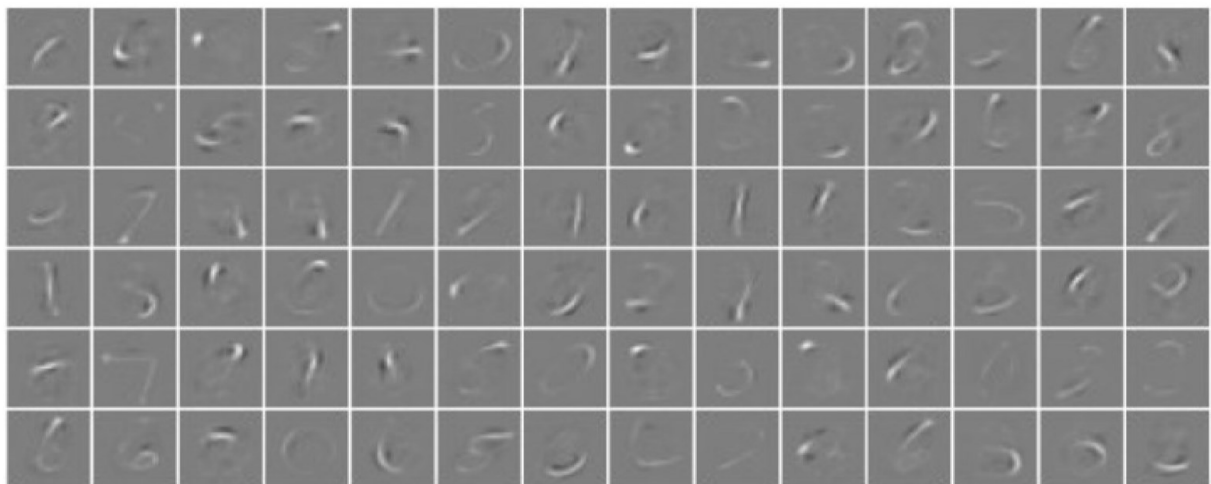


Figure 7. Activation roadmaps (Nebauer et. al 1998).

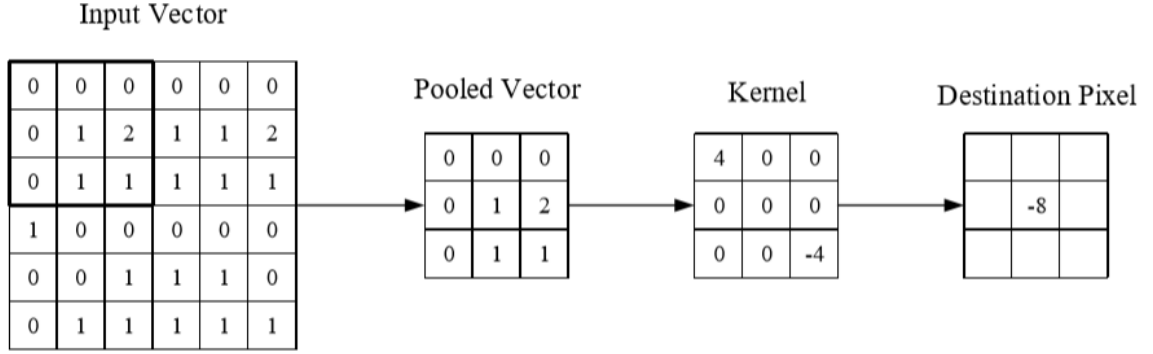


Figure 8. The network will learn kernels that 'fire' when they observe a specific feature at a specific spatial place in the insight (Nebauer et. al 1998).

These are notations for our quick example in this section:

m : number of input samples used for one iteration (batch size*)

p : number of nodes in the input layer (number of input features)

$h1$: number of nodes in hidden layer

$h2$: number of nodes in hidden layer

k : number of nodes in the output layer (number of classes)

i. INPUT $\rightarrow$ HIDDEN

$$X_{\text{batch}} \cdot W^{[1]} + b^{[1]} = z^{[1]}, \quad a^{[1]} = \text{activation}(z^{[1]})$$

$$\begin{array}{c}
 \begin{bmatrix} x_{1,1} & \cdots & x_{1,p} \\ \vdots & & \vdots \\ x_{m,1} & \cdots & x_{m,p} \end{bmatrix} \cdot \begin{bmatrix} w_{1,1} & \cdots & w_{1,h_1} \\ \vdots & & \vdots \\ w_{p,1} & \cdots & w_{p,h_1} \end{bmatrix} + \begin{bmatrix} b_1 \\ \vdots \\ b_{h_1} \end{bmatrix} = \begin{bmatrix} z_{1,1} & \cdots & z_{1,h_1} \\ \vdots & & \vdots \\ z_{m,1} & \cdots & z_{m,h_1} \end{bmatrix} \xrightarrow{f} \begin{bmatrix} a_{1,1} & \cdots & a_{1,h_1} \\ \vdots & & \vdots \\ a_{m,1} & \cdots & a_{m,h_1} \end{bmatrix} \quad (1)
 \end{array}$$

$\times \begin{matrix} (m,p) \\ (p,h_1) \end{matrix} \quad (1,h_1) \quad (m,h_1) \quad (m,h_1)$

A bias vector in the linear combination of each layer has a length h_i , which is equal to the quantity of nodes in layer (i). The form of the matrix $W[i]$ is $(m; h_1)$. The bias vector's transposition is shared among each row of the matrix in order to combine those two terms. The bias is added element-by-element to all m rows of the matrix, and each row corresponds to the h_i calculated values of each observation in a mini-batch sample. Broadcasting is another name for this algorithm, which entails expanding one matrix to have the same dimension as another. Keep in mind that broadcasting is only possible when an axis' length in two matrices is the same (Matouq et. al. 2013).

ii. HIDDEN [1] $\rightarrow$ HIDDEN [2]

$$a^{[1]} \cdot W^{[2]} + b^{[2]} = z^{[2]}, \quad a^{[2]} = \text{activation}(z^{[2]})$$

$$\begin{aligned} & \begin{bmatrix} a_{1,1} & \cdots & a_{1,h_1} \\ \vdots & & \vdots \\ a_{m,1} & \cdots & a_{m,h_1} \end{bmatrix}^{[1]} \cdot \begin{bmatrix} w_{1,1} & \cdots & w_{1,h_2} \\ \vdots & & \vdots \\ w_{h_1,1} & \cdots & w_{h_1,h_2} \end{bmatrix}^{[2]} + \begin{bmatrix} b_1 \\ \vdots \\ b_{h_2} \end{bmatrix}^{[2]'} = \begin{bmatrix} z_{1,1} & \cdots & z_{1,h_2} \\ \vdots & & \vdots \\ z_{m,1} & \cdots & z_{m,h_2} \end{bmatrix}^{[2]} \\ & \xrightarrow{f} \begin{bmatrix} a_{1,1} & \cdots & a_{1,h_2} \\ \vdots & & \vdots \\ a_{m,1} & \cdots & a_{m,h_2} \end{bmatrix}^{[2]}. \end{aligned} \quad (2)$$

$$\times \begin{matrix} & (m, h_1) \\ (h_1, h_2) & (1, h_2) & (m, h_2) & (m, h_2) \end{matrix}$$

iii. HIDDEN [2] $\rightarrow$ OUTPUT

iv. LOSS

Calculate a loss for each observation i and take an average of m losses in the current mini batch. Each loss is a negative log likelihood given the observed vector (y_i) and estimated vector of probabilities $\hat{p}_i$.

$$\text{LOSS}^i = -\sum_k y_k^i \cdot \log \hat{p}_k^i \quad (3)$$

$$E = \frac{1}{m} \sum_{i=1}^m \text{LOSS}^i \quad (4)$$

We can additionally provide the stride in which we established the depth across the input's spatial dimensions in order to locate the receptive field. For example, if we set the stride to 1, we will have a substantially overlapped receptive field with tremendously massive activations. Setting the stride to a higher value, on the other hand, will minimize the amount of overlapping and result in smaller spatial dimensions.

$$\begin{aligned}
a^{[2]} \cdot W^{[3]} + b^{[3]} &= z^{[3]}, \quad \hat{p} = \text{softmax}(z^{[3]}) \\
\begin{bmatrix} a_{1,1} & \cdots & a_{1,h_2} \\ \vdots & & \vdots \\ a_{m,1} & \cdots & a_{m,h_2} \end{bmatrix}^{[2]} &\cdot \begin{bmatrix} w_{1,1} & \cdots & w_{1,k} \\ \vdots & & \vdots \\ w_{h_2,1} & \cdots & w_{h_2,k} \end{bmatrix}^{[3]} + \begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix}^{[3]'} = \begin{bmatrix} z_{1,1} & \cdots & z_{1,k} \\ \vdots & & \vdots \\ z_{m,1} & \cdots & z_{m,k} \end{bmatrix}^{[3]} \\
(m, h_2) &\quad \times \quad (h_2, k) \quad (1, k) \quad (m, h_2) \\
\text{softmax} \rightarrow \begin{bmatrix} \frac{\exp(z_{1,1})}{\sum_i \exp(z_{1,i})} & \cdots & \frac{\exp(z_{1,k})}{\sum_i \exp(z_{1,i})} \\ \vdots & & \vdots \\ \frac{\exp(z_{m,1})}{\sum_i \exp(z_{m,i})} & \cdots & \frac{\exp(z_{m,k})}{\sum_i \exp(z_{m,i})} \end{bmatrix}^{[3]} & \quad (5) \\
(m, h_2) &
\end{aligned}$$

Zero-padding is a basic approach for padding the input's border, and it's a useful strategy for increasing the dimensionality of the result volumes. It is critical to recognize that by employing a few strategies, a new spatial dimensionality can be obtained for the convolutional layer's result. The following formula can be used:

$$\frac{(V-R)+2Z}{S+1} \quad (6)$$

Where V stands for the input volume size (height, breadth, and depth), R stands for the receptive field size, Z stands for zero padding set, and S stands for the stride. When the calculated result from this formula is not a complete integer, the stride has already been set wrongly, because the neurons will not fit neatly over the specified insight. Regardless of how hard we try, whenever we use an image input with any actual dimensionality, our models will still be massive. Nonetheless, strategies for drastically reducing the overall number of parameters within the convolutional layer have already been discovered. The idea behind parameter sharing is that if one region feature is valuable to compute in one spatial region, it is likely to be useful in another. We can drastically reduce the number of parameters produced by the convolutional layer by constraining each activation map within the away put volume to the same dumbbells and bias. As a result, when the backpropagation stage happens, each neuron in the result will represent the overall gradient, which may be total led across the depth - therefore only upgrading one set of weights rather than all of them.

3.7 Pooling layer

Pooling layers are utilized in CNNs to reduce the number of parameters, decrease the dimensionality of the representation, and simplify the computational complexity of the model. Typically, pooling layers employ the "MAX" function, where each feature map in the input is divided into non-overlapping regions. These regions, often with a 2×2 dimensionality, are pooled using max-pooling operations with a stride of 2 along the spatial dimensions of the input. This pooling technique reduces the activation map to 25% of its original size while maintaining the depth of the volume.

In most cases, max pooling with a stride and filter size of 2x2 is commonly observed due to the down sampling nature of the pooling layer. Alternatively, overlapping pooling can be employed by setting the stride to 2 and the kernel size to 3. However, it is important to be cautious when using larger kernel sizes, as they can significantly degrade the model's performance due to the destructive nature of pooling. It is worth noting that, in addition to max-pooling, CNN architectures may incorporate other types of pooling, such as general-pooling. General pooling layers allow pooling neurons to perform various common operations, including L1/L2 normalization and average pooling. However, the focus of this discussion will primarily be on the effective use of max-pooling (Simard et al., 2003).

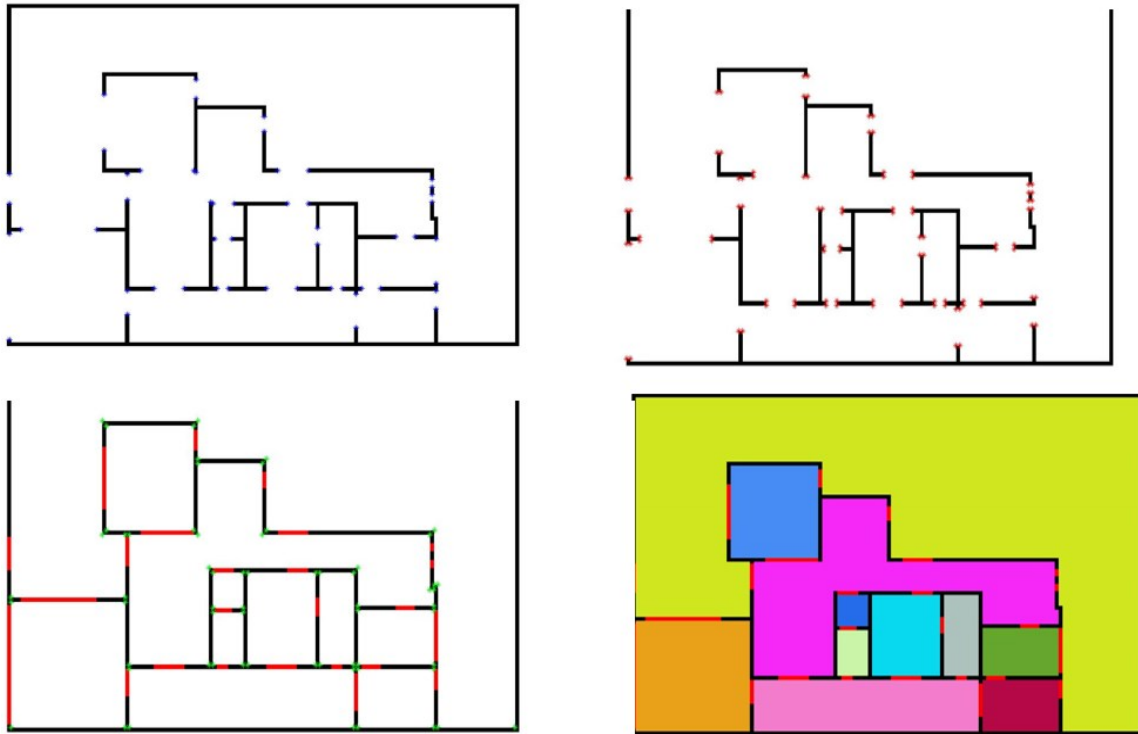


Figure 9. Pooling layer for the floorplan.

3.8 Recipes

While CNNs typically involve a few layers, there is no one-size-fits-all approach to designing a CNN architecture. Simply combining layers without careful consideration would be unwise. CNN architectures often follow a common structure, as evidenced by existing literature. Figure 10 illustrates a standard architecture where convolutional layers are stacked, followed by pooled layers, before feeding forward to fully connected layers (Srivastava et al., 2013).

Another common CNN architecture, as shown in Figure 10, is to stack two convolutional layers before each pooling layer. This practice is highly recommended as stacking multiple convolutional layers allows for capturing more complex input features.

It is also advisable to split large convolutional layers into multiple smaller ones to reduce computational complexity. For instance, if you stack three convolutional layers on top of each other with a receptive field of 3×3 , each neuron in the first convolutional layer perceives the input vector in 3D. Subsequently, the input vector is seen by neurons in the second convolutional layer with a 5×5 view, and then by neurons in the third convolutional layer with a 7×7 view (Szarvas et al., 2005). Non-linearities are present within these layer stacks, enabling the representation of stronger input features with fewer parameters. However, it is important to note that this approach can lead to memory allocation issues, particularly when using backpropagation. The input layer's dimensions should be recursively divisible by two, such as $(32 \times 32, 64 \times 64, 96 \times 96, 128 \times 128, \text{ or } 224 \times 224)$, which are commonly used sizes.

When using smaller filters, it is recommended to set the stride to one and employ zero-padding to maintain the input's dimensionality. The amount of zero-padding can be calculated by subtracting one from the receptive field size and dividing by two. CNNs with activation functions are powerful machine learning algorithms, but they can be resource-intensive (Szegedy et al., 2013).

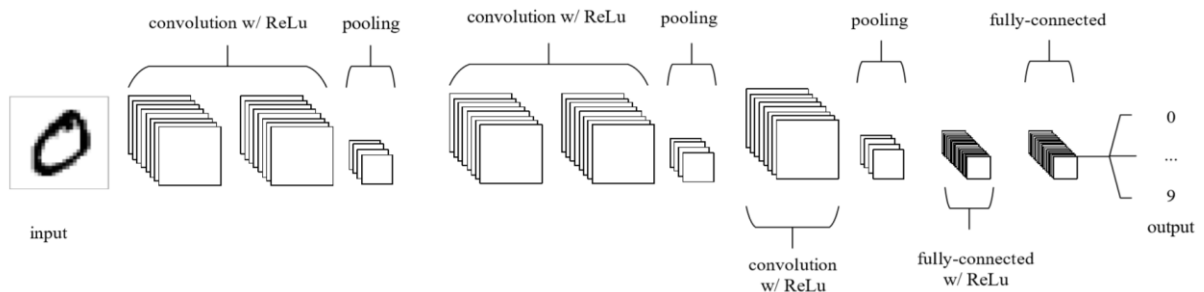


Figure 10. Frequent CNN architecture is to stack two convolutional layers before each pooling layer (Srivastava et. al 2013).

3.9 CNN layers Details

The CNN architecture consists of a number of layers (or so-called multi-building blocks).

Each layer in the CNN architecture, including its function, is described in detail below.

Convolutional Layer:

- **Kernel definition:** A grid of discrete numbers or values describes the kernel.
- **Convolutional Operation:** Initially, the CNN input format is described. The

Pooling Layer:

Activation Function (non-linearity) Mapping the input to the output is the core function.

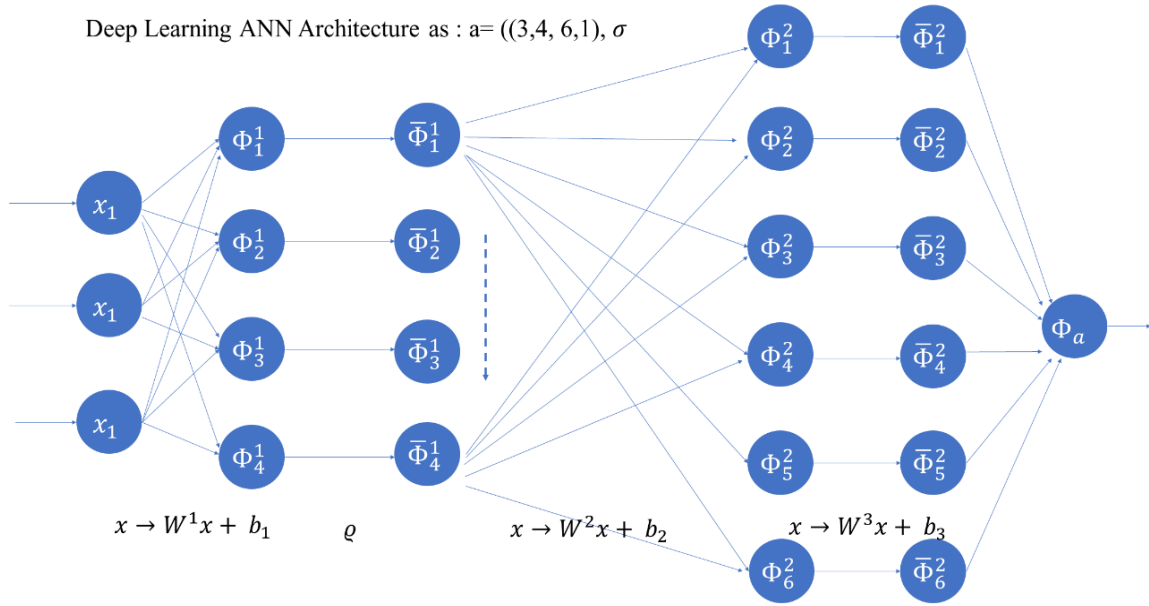


Figure 11. Deep neural network with architecture.

Artificial Neurons Models:

$$f(x_1, \dots, x_n) = \rho(\sum_{i=1}^n x_i w_i - b) = \rho(\langle X, W \rangle - b), \quad (7)$$

where

$$w = (w_1, \dots, w_n) \text{ and } x = (x_1, \dots, x_n)$$

There are a number of standard activation functions:

Heaviside function: $\varrho(x) = \begin{cases} 1, & x > 0, \\ 0, & x \leq 0, \end{cases}$

The sigmoid function:

$$\varrho(x) = \frac{1}{1+e^{-x}} \quad (8)$$

Rectifiable Linear Unit: $(\text{ReLU})\varrho(x) = \max\{0, x\}$

Affine Linear Maps and Weights:

Remark: Concatenating artificial neurons leads to compositions of affine linear maps and activation functions. Example: The following part of a neural network is given by:

$$\phi: \mathcal{R}^3 \rightarrow \mathcal{R}^2, \dots \quad \varphi(x) = W^{(2)}\varrho(W^{(1)}x + b^{(1)}) + b^{(2)} \quad (9)$$

$$W^{(1)} = \begin{pmatrix} w_{11}^1 & w_{12}^1 & 0 \\ 0 & 0 & w_{23}^1 \\ 0 & 0 & w_{33}^1 \end{pmatrix}$$

$$W^{(2)} = \begin{pmatrix} w_{11}^2 & w_{12}^2 & 0 \\ 0 & 0 & w_{23}^2 \end{pmatrix}$$

More definition of a Deep Neural Network:

$$\theta = \theta^{(l)}_{l=1}^L = ((W^{(l)}, b^{(l)}))_{l=1}^L \in {}_{l=1}^L X(\mathcal{R}^{N^\ell \times N_{l-1}} \times \mathcal{R}^{N^\ell}) \cong \mathcal{R}^{P(N)} \quad (10)$$

hence,

$$\begin{aligned} \phi_a(x, \theta) &= W^1 x + b^1 \\ \hline \phi^l(x, \theta) &= \varrho(\phi^{(l)}(x, \theta)), \dots \dots l \in [L - 1] \end{aligned}$$

$$\bar{\phi}^l(x, \theta) = \varrho(\phi^{(l)}(x, \theta)), \dots \dots l \in [L - 1] \quad \text{in addition, ...}$$

$$\phi^{(l+1)}(x, \theta) = W^{(l+1)} \bar{\phi}^{(l)}(x, \phi) + b^{(l+1)} \quad \text{and } l \in [L - 1]$$

we also refer to the followings:

$W^{(l)} \in \mathfrak{R}^{N_l \times N_{l-1}} \dots$ and $b^{(l)} \in \mathfrak{R}^{N_l}$ as the wights matrix and the bias vectors. In addition, $\bar{\phi}^{(l)}$ and the $\phi^{(l)}$ as the activations and pre-activations of the N_l neurons in the $l - th$ layer.

3.9.1 Training of Deep Neural Networks:

High-Level Set Up:

Samples : $(x_i, f(x_i))_{i=1}^m$ of a function, such as $f: \mathcal{M} \rightarrow \{1, 2, 3, \dots, \kappa\}$.

Training- and test data set. Then we need to select a ANN architecture, ...

$a = (N, \sigma)$ of a deep ANN,

Then to learn the weight matrices $W^{(l)} \in \mathfrak{R}^{N^{(l)} \times N^{(l-1)}}$. In addition, the basis vectors are: $b^{(l)} \in \mathfrak{R}^{N^{(l)}}$ by:

$$\min_{\theta} \sum_{i=1}^m l(\phi_a(x_i, \theta), f(x_i, \theta), f(x_i)) + \lambda \rho(\theta) \quad (11)$$

Yielding the ANN ϕ_a , This is often done by stochastic gradient descent. This is often done by stochastic gradient descent algorithms:

$$\text{Goal : } \phi_a(., \theta) \approx f$$

3.9.2 Training the model:

In this step we are now ready to train the model. A neural network type called a convolutional neural network, or CNN or Conv Net, is particularly adept at processing input with a grid-like architecture, like an image. A binary representation of visual data is a digital image. It is made up of a grid-like arrangement of pixels, each of which has a pixel value to indicate how bright and what color it should be (Mishra, 2020).

The moment we perceive an image, the human brain begins processing a massive amount of data. Every neuron has a distinct receptive field and is coupled to other neurons so that they collectively cover the whole visual field. Each neuron in a CNN processes data only in its receptive field, like how each neuron in the biological vision system responds to stimuli only in the constrained area of the visual field known as the receptive field. The layers are set up so that simpler patterns are detected early on and more complicated patterns later. One can enable sight to computers by employing a CNN (Mishra, 2020).

Convolution is a mathematical process that involves multiplying two functions to create a third function that expresses how the form of one function is altered by the other. The term "convolution" is used in CNN to refer to this

mathematical activity. To extract features from a picture, two images that can be represented as matrices are multiplied to produce an output (Gurucharan, 2022).

```
# making the deep learning function
def model():
    model = Sequential()
    model.add(Conv1D(filters=64, kernel_size=6, activation='relu',
                     padding='same', input_shape=(14, 1)))
    model.add(BatchNormalization())
    model.add(MaxPooling1D(pool_size=(3), strides=2, padding='same'))
    model.add(Flatten())
    model.add(Dropout(0.1))
    model.add(Dense(64, activation='relu'))
    model.add(Dense(64, activation='relu'))
    model.add(Dense(5, activation='softmax'))
    model.compile(loss='categorical_crossentropy', optimizer='adam', met-
rics=['accuracy'])
    return model
```

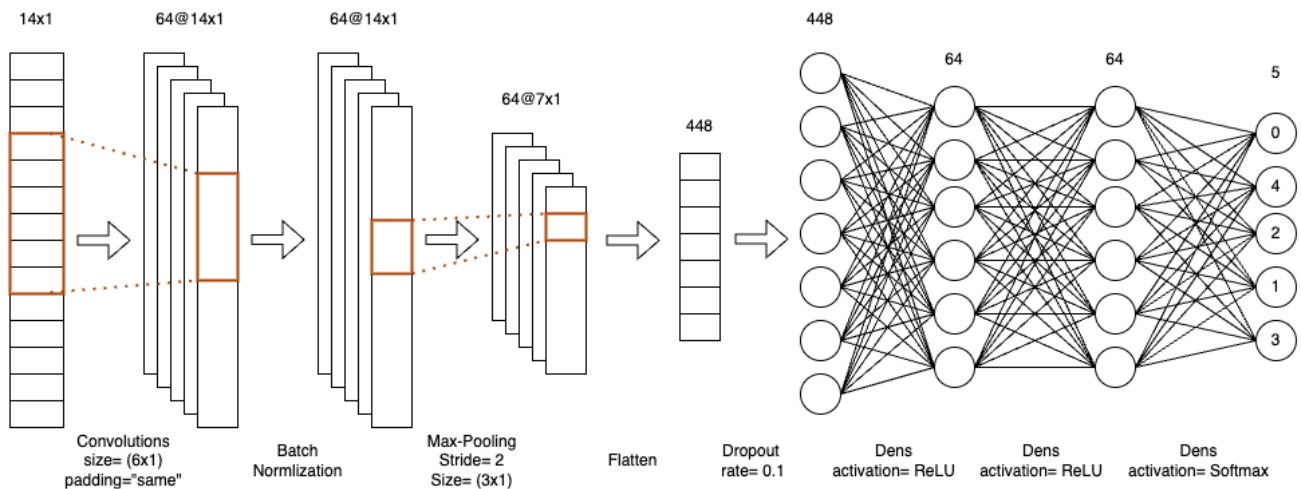


Figure 12. CNN layers visualization.

CNN is made up of three different kinds of layers: convolutional, pooling, and fully connected (FC). A CNN architecture is created when these layers are

stacked (Gurucharan, 2022). The used layers in our model are visualized in Figure 12. Layers are discussed below with steps:

1. 1D Convolutional layer:

The foundational component of the CNN is the convolution layer. It carries most of the computational load on the network (Mishra, 2020). The first layer utilized to extract the different features from the input photos is this one. The input image and a filter of a specific size (MxM) are convolution mathematically in this layer; in our scenario, the input image will be of size (14x1). The dot product between the filter and the portions of the input image regarding the size of the filter (MxM), which is in our example (6x1), is taken by swiping the filter over the input image (Gurucharan, 2022).

$$W_{out} = \frac{W-F+P}{S} + 1 \quad (12)$$

From Formula (12), we can calculate the size of the output volume if we have an input of size (W×W×D), D_{out} number of kernels, spatial/kernel size F (a single integer or set of single integers specifying the 1D convolution window's length), stride S (the kernel's movable size), and amount of padding P (When you pad, the output size is equal to the input size because of the same convolution) (Nkomo, 2022; Mishra, 2020). In our case the input shape is (None,14,1) so W=14, and the kernel size is 6 so F=6, and padding is 'same' that indicates we want the output to be same size of our input, so $W=W_{out}=14$, and S is 1 by default.

$$output = (None, 14, 64)$$

Figure 13 illustrates the steps of Convolution layer.

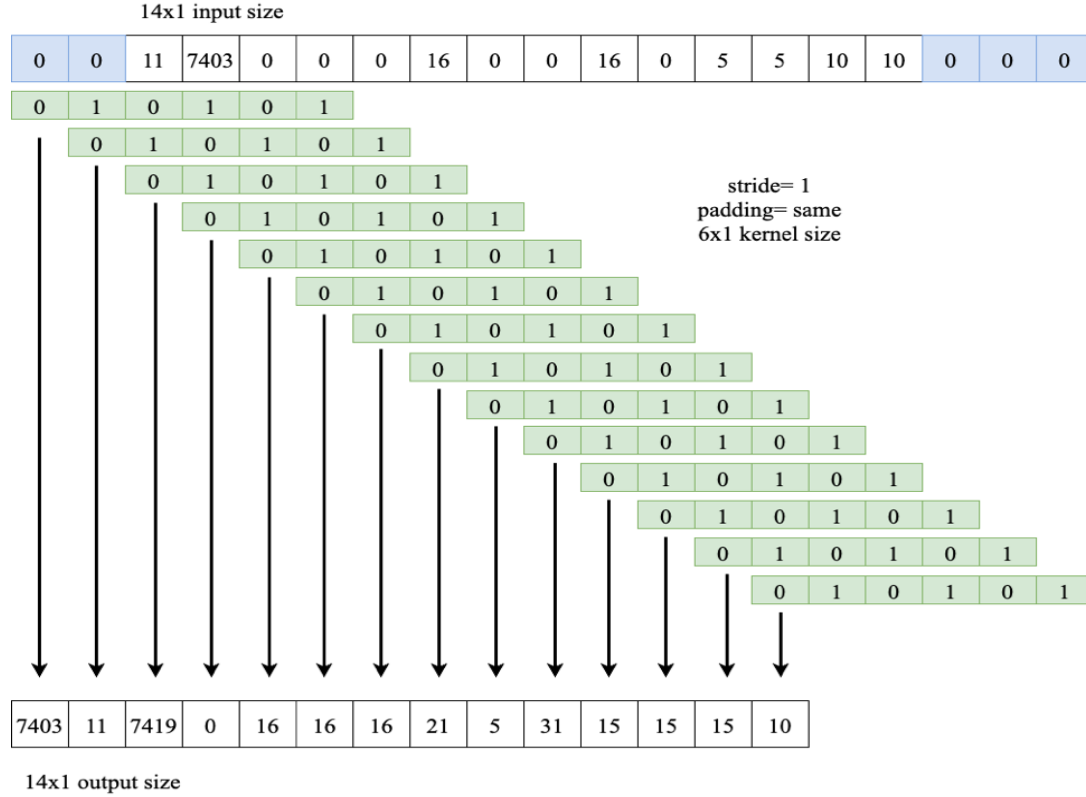


Figure 13. Steps of Convolution Layer.

2. Batch Normalization Layer

Instead of conducting the normalization in the raw data, batch normalization is done between neural network layers. Instead of utilizing the complete data set, mini batches are used. It accelerates training and uses faster learning rates to promote learning (Riva, 2022). The output dimension is the same as the input dimension.

$$output = (None, 14, 64)$$

The theory behind the normalization layer is demonstrated in the proceeding equations, (Peccia, 2019). The batch mean is calculated with the batch variance, Formula (13), and Formula (14).

$$\mu_B = \frac{1}{m} \sum_{i=1}^m x_i \quad (13)$$

$$\sigma_B^2 = \frac{1}{m} \sum_{i=1}^m (x_i - \mu_B)^2 \quad (14)$$

Next step is applying normalization with Formula (15),

$$\bar{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \quad (15)$$

The last step is applying scale and shift to get the output of the layer with Formula (16).

$$y_i = \gamma \bar{x}_i + \beta \quad (16)$$

3. 1D Max Pooling Layer

By obtaining a summary statistic of the surrounding outputs, the pooling layer substitutes for the network's output at specific locations. The representation's spatial size is lowered as a result, which lowers the calculation and weighting requirements. Every slice of the representation that the pooling operation touches is treated separately (Mishra, 2020).

A weighted average based on the distance from the central pixel is one of the pooling functions, along with the average of the rectangular neighborhood and the L2 norm of the rectangle neighborhood. However, max pooling, which reflects the highest output from the neighborhood, is the most well-liked method (Mishra, 2020).

The greatest component in Max Pooling is derived from a feature map. The elements inside a predetermined sized Picture slice are averaged using the average pooling technique. Sum Pooling calculates the

aggregate sum of the components in the predefined section. Convolutional Layer and Fully Connected Layer are typically connected by the Pooling Layer (Gurucharan, 2022).

$$W_{out} = \frac{W-F}{S} + 1 \quad (17)$$

The output volume can be calculated using Formula (18) if we have an activation map of size ($W \times W \times D$), a pooling kernel of spatial size F , and stride S (Mishra, 2020), in our case here the output of the previous layer is to be used as the input in this layer, so $W=14$, $F=3$, and $S=2$. However, using the "same" padding option produces the following output shape: $output\ shape = \frac{input\ shape}{strides}$

$$W_{out} = \frac{14}{2} = 7 \quad (18)$$

$$output = (None, 7, 64)$$

Figure 14 illustrates the steps of Max-Pooling layer.

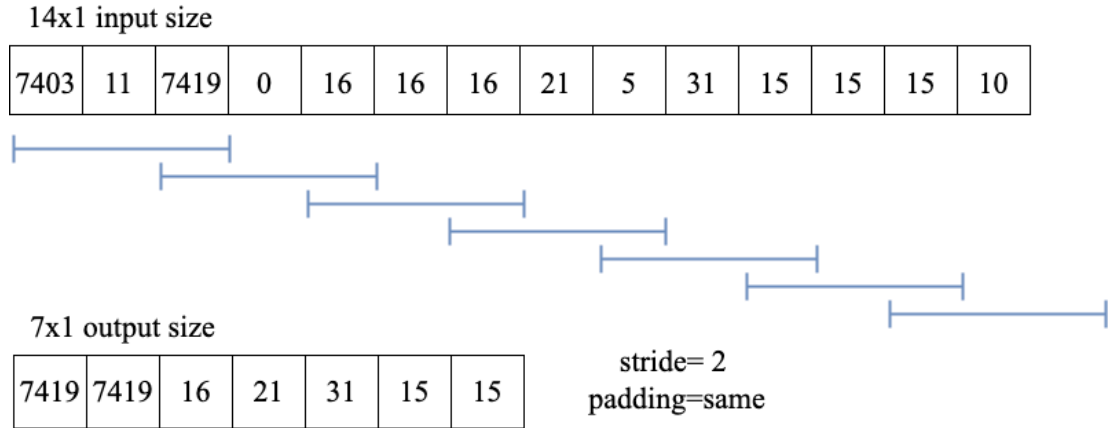


Figure 14. Steps of Maximum Pooling layer.

4. Flatten Layer

We have flattened and completely linked layers at the end of CNN. It must take the form of a one-dimensional linear vector. Direct inputs are not allowed to be rectangular or cubical in form. As a result, we need layers that are fully connected and flattened.

When data is converted into a 1-dimensional array for input into the next layer, it is flattened. We flatten the output of the convolutional layer to create a feature vector. It is also linked to the previous classification model, which is known as a fully connected layer, as shown in Figure 15. In other words, we connect the last layer to a single line that contains all the pixel data (Jeong, 2019).

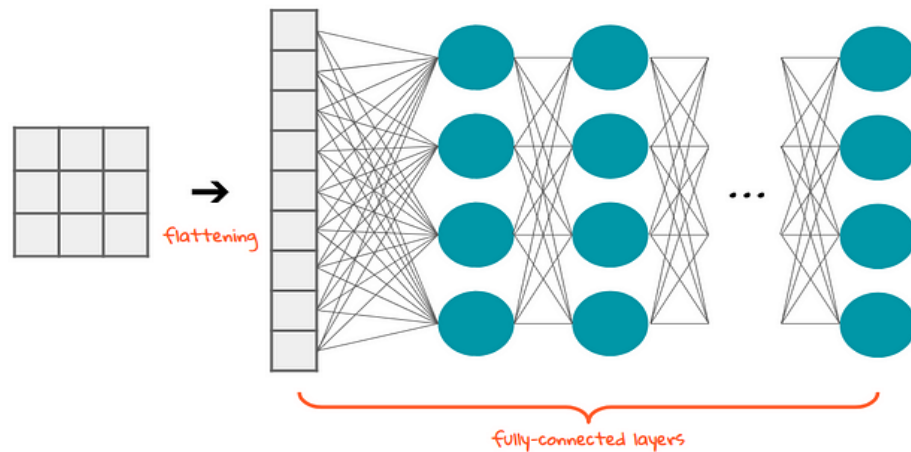


Figure 15. Flatten and fully connected layers, (Jeong, 2019).

By taking the previous output as an input for this layer, we got the following output:

$$output = (None, 7 \times 64) = (None, 448)$$

5. Dropout layer

A dropout layer is a particular kind of regularization method that is used to avoid overfitting and enhance generalization error in machine learning and neural network models. A dropout layer randomly removes a part of its input units during each training cycle. As a result, because it cannot rely on a single input unit to always be available, the model is compelled to acquire more reliable and generalizable properties. Any layer in a neural network model can have dropout layers added, and a hyperparameter can be used to adjust how many units will really drop out of the network (Brownlee, 2019).

$$output = (None, 448)$$

6. Dens Layer

Each neuron in a dense layer that is connected to the layer above it in a neural network signifies that layer is connected to the layer above it. This layer is the one that is most typically used in artificial neural network networks. Each neuron in the layer above delivers signals to the neurons in the dense layer, which multiply matrices and vectors, in a model. During matrix vector multiplication, the row vector of the output from the preceding layers is identical to the column vector of the dense layer. To multiply matrices using vectors, the row vector must have the same number of columns as the column vector (Verma, 2023). One of the most fundamental and important parameters of the Keras dense layer, which determines the size of the dense layer's output, is units. Since it reflects the output vector's dimensions, it must be a

positive integer; in this layer, the units are 64. The activation function in neural networks is a function that transforms the input values of the neurons. It adds nonlinearity to neural network networks so that the networks can figure out how input and output values relate to one another. In this layer, we are using the ReLU activation function (Verma, 2023).

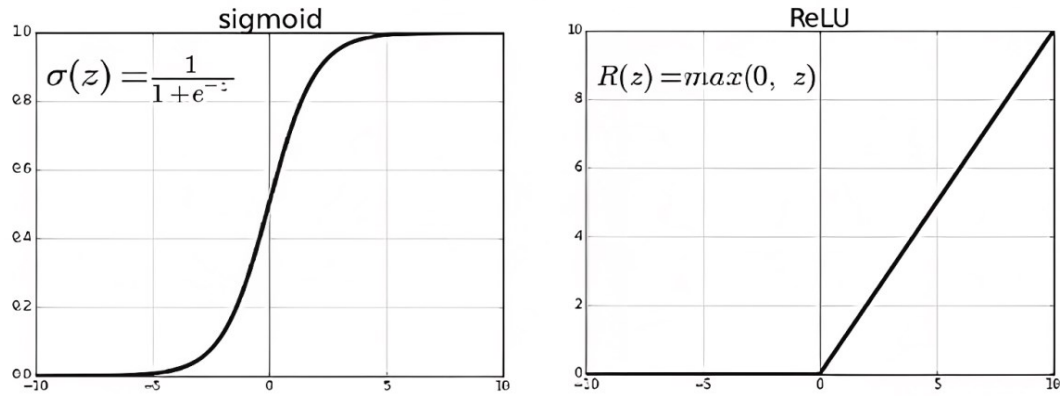


Figure 16. Sigmoid and ReLU functions, (Sharma, 2022).

Nowadays, the ReLU is the activation function that is employed the most globally. Figure 16 illustrates how the ReLU is only partially rectified (from bottom). When z is less than zero, $f(z)$ equals zero, and when z is more than or equal to zero, $f(z)$ equals z . Hence, the Range is $[0 \text{ to infinite})$ (Sharma, 2022).

output = (None, 64)

7. Dens Layer

output = (None, 64)

8. Dens Layer

The neural network's raw outputs are input into the SoftMax activation function, which then produces a vector of probability scores (Priya, n.d.).

- The neural network's raw outputs are represented by the vector z .
- 2.718 is the value of e .
- The expected probability that the test input will belong to class i is represented by the i^{th} entry in the SoftMax output vector $\text{SoftMax}(z)$.

The function of SoftMax is represented below:

$$\text{softmax}(z)_i = \frac{e^{z_i}}{\sum_{j=1}^N e^{z_j}} \quad (19)$$

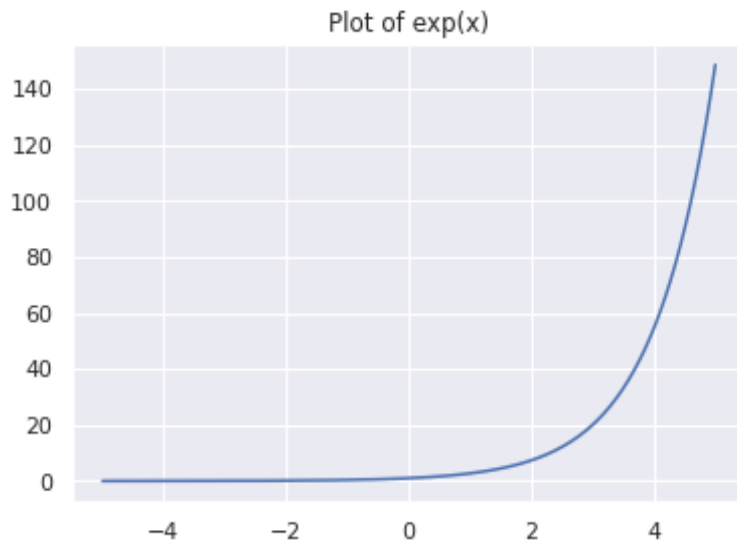


Figure 17. Plot of exp, (Priya, n.d.).

It's clear from the plot of e^x that e^x is always a positive value, regardless of whether the input x is positive, negative, or zero, Figure 17, (Priya, n.d.). By assuming the given neural network's raw outputs, represented by the vector $z = [0.3, 1.15, -0.7, 5.2, 1.4]$. To elaborate more on how the calculation process is done, the following calculations are done with referral to (Priya, n.d.):

1. Determine each entry's exponent.
2. Subtract the outcome of step 1 from the total exponents of all the entries.

$$\frac{e^{0.3}}{e^{0.3} + e^{1.15} + e^{-0.7} + e^{5.2} + e^{1.4}} = 0.00709$$

$$\frac{e^{1.15}}{e^{0.3} + e^{1.15} + e^{-0.7} + e^{5.2} + e^{1.4}} = 0.01659$$

$$\frac{e^{-0.7}}{e^{0.3} + e^{1.15} + e^{-0.7} + e^{5.2} + e^{1.4}} = 0.00260$$

$$\frac{e^{5.2}}{e^{0.3} + e^{1.15} + e^{-0.7} + e^{5.2} + e^{1.4}} = 0.95239$$

$$\frac{e^{1.4}}{e^{0.3} + e^{1.15} + e^{-0.7} + e^{5.2} + e^{1.4}} = 0.02130$$

$$\begin{bmatrix} 0.3 \\ 1.15 \\ -0.7 \\ 5.2 \\ 1.4 \end{bmatrix} \rightarrow \text{Softmax} \rightarrow \begin{bmatrix} 0.00709 \\ 0.01659 \\ 0.00260 \\ 0.95239 \\ 0.02130 \end{bmatrix}$$

The following conclusions can be drawn from the SoftMax output above: The greatest element in the SoftMax output vector, 0.952, is represented by the maximum value of 5.2 in the vector z of raw outputs.

SoftMax's application thus maintains the relative ranking of scores. The SoftMax output vector's entries are all between 0 and 1. See how the entries of the SoftMax output aggregate to 1 in a multiclass classification problem where the classes are mutually exclusive (Priya, n.d.): $0.0071 + 0.0166 + 0.00260 + 0.9524 + 0.02130 = 1$.

By checking the SoftMax output for our case as a following step.

Figure 18 demonstrates the complete layers of CNN model.

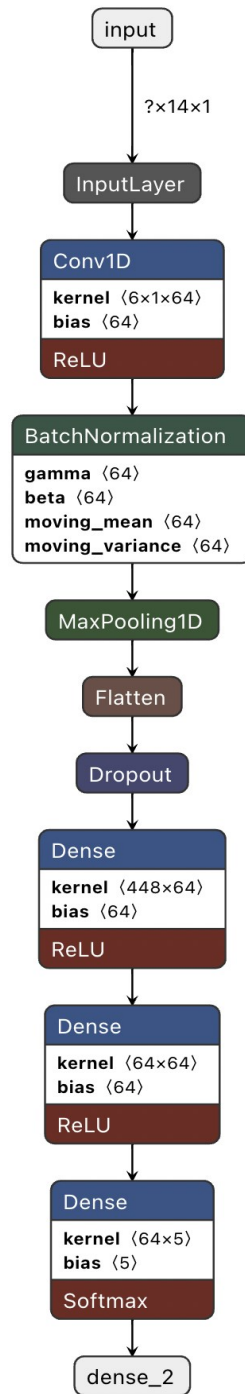


Figure 18. CNN Model Layers (Jeong, 2019).

Chapter 4

Floorplans Dataset Topology

Chapter 4 Floorplans Dataset Topology

4.1 Introduction

A collection of records that are given a specific name is referred to as a data set. Datasets are utilized by programs running on a system to store various types of information, such as medical records or insurance records. They can also be used to store data required by applications or the operating system itself, such as source programs, macro libraries, or system variables and parameters. Data sets that contain readable text can be printed or displayed on a console, although some data sets may contain binary data or load modules that are not easily printable. Cataloging data sets allows them to be referenced by name without specifying their storage location.

In this research, the training dataset will consist of a collection of floor plans. The existing literature on automatic floor plan conversion lacks representative large-scale data sets. Currently, the largest publicly available annotated data set contains fewer than (1,000 samples). The diversity of objects, such as different room and icon types, and the consistency and accuracy of their annotations, such as wall thickness, are limited. This indicates that there is an opportunity for further studies to explore the advantages of using larger and more content-rich data sets for training deep convolutional neural networks (CNNs).

4.2 Floorplans and existing datasets

The analysis and understanding of floorplan images have been a subject of research in automatic document analysis, which is a branch of computer

vision. Floorplans are scaled drawings that depict the structure of a building or apartment from a top-down perspective. They convey structural information and related semantics to the viewer. Typical elements found in floorplans include rooms, walls, doors, windows, and fixed furniture. They can also include technical details like building materials, electrical wiring, or plumbing lines. While floorplans are often initially created using computer-aided design (CAD) software in a vector graphics format, they are often converted to raster images for printing or digital publication for marketing purposes, such as selling or renting properties. However, this rasterization process discards the structured geometric and semantic information, making it difficult to utilize the floorplan drawings for applications like 3D real estate virtual tours or floorplan-based 3D model creation. Recovering the lost information from rasterized floorplan images is a non-trivial task.

Current state-of-the-art models in automatic floorplan image analysis are based on deep convolutional neural networks (CNNs). A promising approach involves training a single network backbone with multiple multi-task heads to recover the lost objects, structure, and semantics. While the results are already promising, the available datasets for training and benchmarking are still relatively small, with fewer than 100 samples. It is well-known that deep learning models require large amounts of data to achieve optimal performance, and increasing the dataset size would likely lead to better results. A breakthrough in this context was the proposal of an automatic floorplan parsing approach based on CNNs. Instead of relying on low-level image processing and subsequent heuristics, a regular fully convolutional network was trained to label objects (rooms, icons, and openings) and localize wall

joints. The extracted low-level information was then processed to recover the original floorplan objects as 2D polygons using a post-processor. The model was jointly optimized for segmentation and localization tasks. The key finding was that deep neural networks can serve as an effective precursor to the final post-processing heuristics for restoring floorplan elements, including their geometry and semantics. This method significantly improved the state-of-the-art and has inspired recent research in the field.

In this thesis, a novel floorplan image dataset is proposed, consisting of 5000 images with dense and accurate ground-truth annotations represented as polygons. The proposed dataset is more than five times larger than the previously largest dataset and offers more precise shape and directional information of objects. It also exhibits greater variability in terms of apartment types and drawing styles (Liu et al., 2017).

4.3 CubiCasa5K: A Novel Floorplan Dataset

The CubiCasa5K dataset is generated as a result of an online floor plan vectorization pipeline, which involves a combination of manual and automated processes. The pipeline primarily focuses on converting real estate marketing materials from the Finland region. The primary objective of this dataset is to provide the research community with resources to develop more accurate and automated models for applications related to real estate and other fields. The dataset comprises (5000 floorplans), each with manual annotations as shown in Figure 19. These floorplans were carefully selected and reviewed from a larger collection of (15,000 floorplan) images, predominantly sourced from Finland. The floorplans are categorized into three subcategories: high

quality architectural, high quality, and colorful. These subcategories consist of (3732, 992, and 276 floorplans), respectively. In order to train robust machine learning models, the dataset is randomly divided into training, validation, and test sets. The training set contains (4200 floorplans), while the validation and test sets contain (400 floorplans) each. The annotations in the dataset are provided in per-image SVG vector graphics format. Each annotation includes both semantic and geometric information for all the elements present in the corresponding floorplan. This comprehensive dataset aims to facilitate advancements in the development of machine learning models for floor plan analysis and related tasks (Schönfelder et al., 2017).

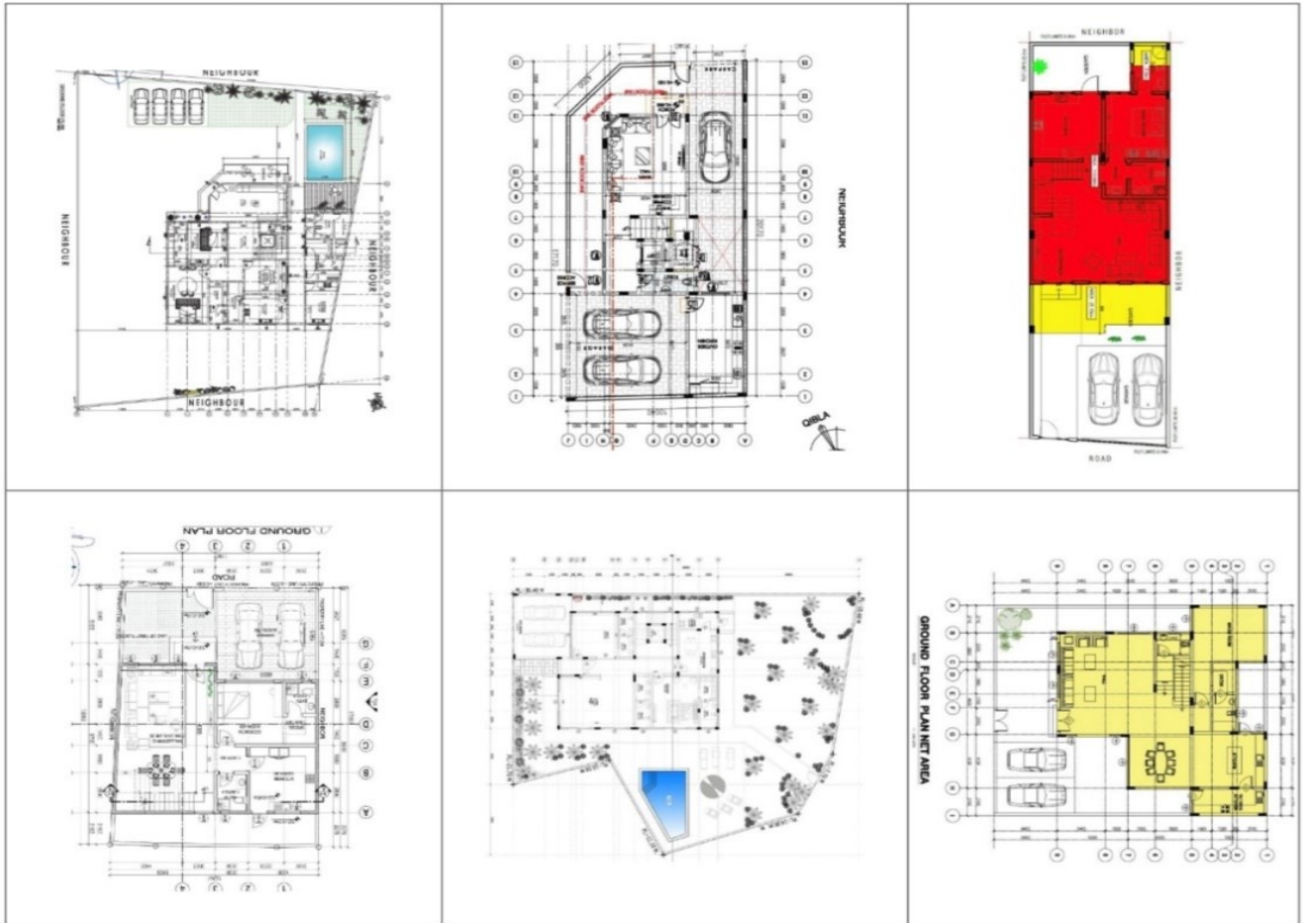


Figure 19. Sample from CubiCasa5K dataset.

4.3.1 Annotations and their consistency.

The proposed dataset has undergone an annotation pipeline that resulted in vectorized floorplan images with detailed annotations. The input for each sample is a raster scan, typically a scanned copy of the original floor plan drawing. Skilled human annotators manually performed the annotations, with the time taken ranging from 5 to 120 minutes, depending on factors such as complexity, clarity of the source, and the number of floors. An annotation protocol, described by Kippers et al. (2021), was followed for annotating each floorplan. This protocol emphasizes utilizing information from previously annotated elements, such as using walls as boundaries for rooms. A specialized computer-aided design (CAD) tool tailored for drawing floorplans was used for the annotation process.

To ensure consistency in the annotations, a Quality Assurance (QA) process with two stages was implemented for each annotated sample image. In the first stage, the annotator thoroughly reviewed and corrected any possible errors in the annotated floor plan. The second stage involved a separate QA person performing the same checking procedure as the initial annotator, identifying and correcting any errors that may have been missed in the first round.

Considering all these details, it can be concluded that the CubiCasa5K dataset is currently the largest and most versatile publicly available floorplan dataset.

Chapter 5

Floor Plan Framework and Code Adherence Detection Process (Floorplan Recognition)

Chapter 5 Floor Plan Framework and Code Adherence Detection Process (Floorplan Recognition)

5. 1 Introduction

Building a demonstration software in Matlab to automatically extract, represent, and analyze graph representations from floor plan photos is the technical aim of this section. Figure 20 will illustrate the preprocessing stage for area's extraction and the filtration steps that will be conducted on the floorplan to ensure the accuracy of the results.

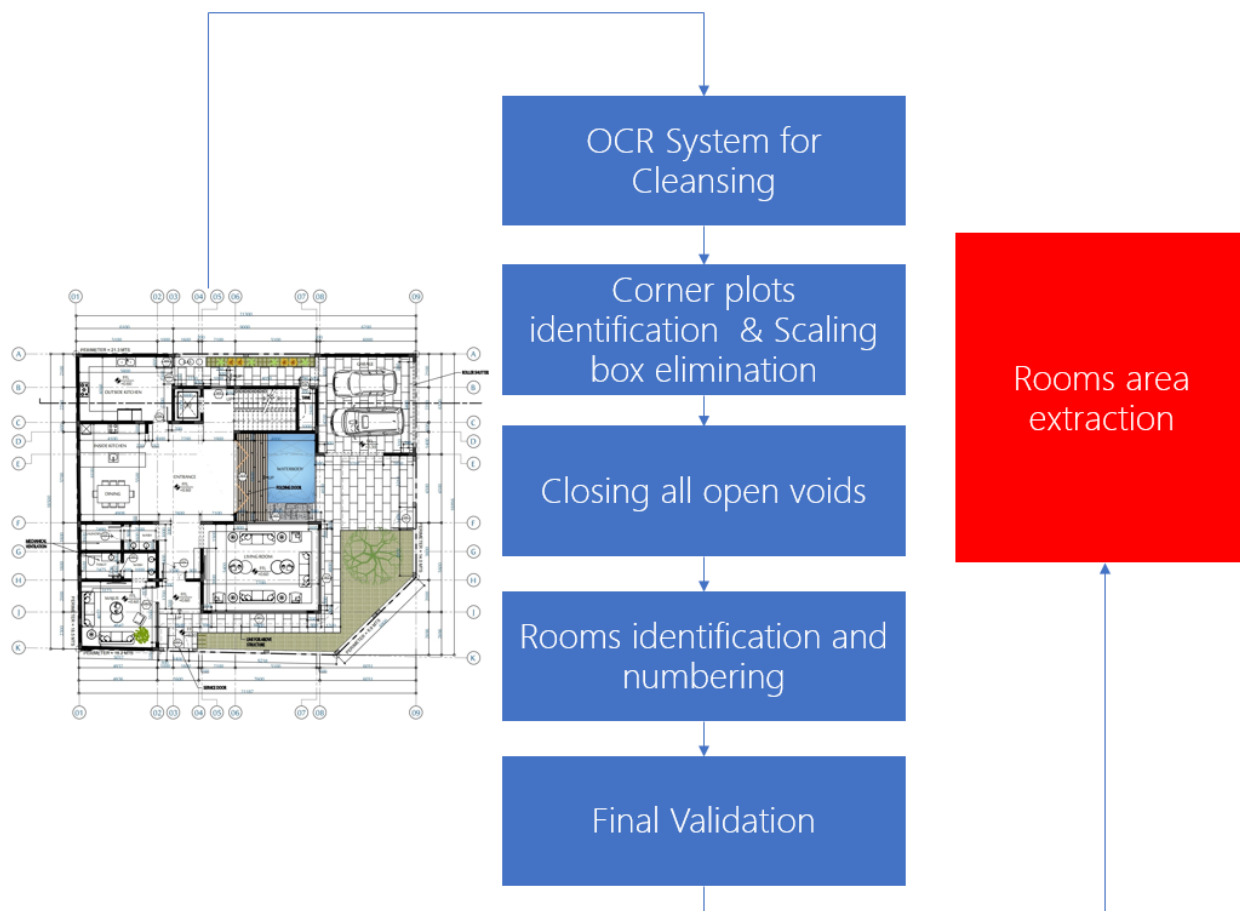


Figure 20. Pre-processing workflow for area's extraction.

Beginning with the unprocessed image of a building's floor plan Figure 21, each step of the graph extraction process is discussed in depth. The algorithmic techniques utilized to get around the primary challenges are also described in detail. Along with the code script, a graphic description of the technical processes is attached. The program is expected to receive a floorplan which will be entered as an input. The program then must read and recognize the floorplan, evaluate its criteria, extract the required information and then take a decision whether this floorplan is expected based on some predefined rules.

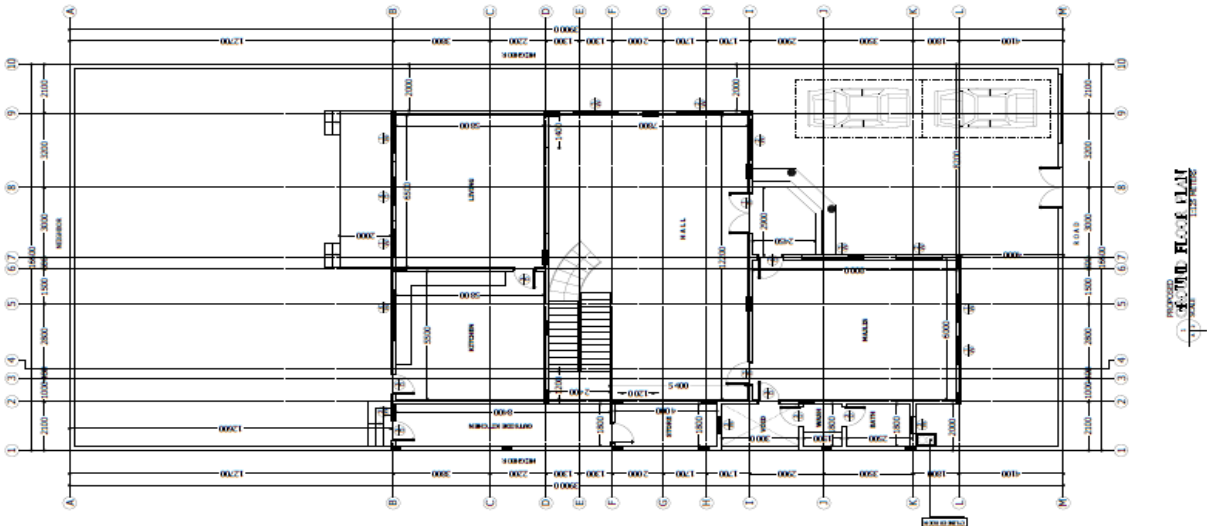


Figure 21. The row input as a floorplan with all the data attached.

5.2 Algorithm Generation Steps

The first task is the pre-processing of the floor plan image. The reason why this is necessary is that the only semantic elements that we need to extract from the image are walls, the door voids, the window voids, and background. There are other semantic elements, such as text labels, symbols to represent

architectural elements and furniture, and other drawing symbols that we need to rule out in the first place. An OCR system is therefore used to remove text labels. To control background noise, the image is then processed with a morphological blur filter and finally binarized to a black-and-white image as shown in Figure 22.

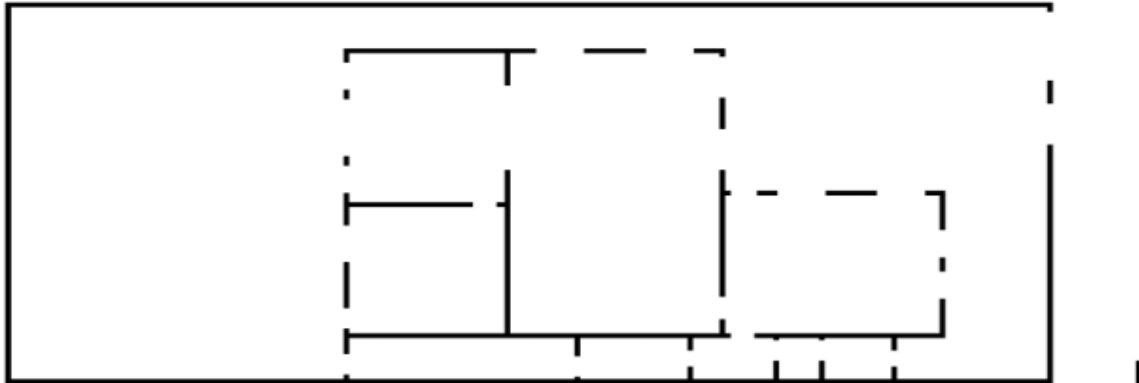


Figure 22. Removing all text labels and symbols to represent architectural elements and furniture.

We are now left with the recovered wall segments (1 valued pixels) over background following the pre-processing stage. The interest points, such as the corner points defining the doors, are not yet recorded in any data structures because this is still an image. Now the task is to locate where the doors are. After the cleansing process, it is very important to find out and identify the corner points of the floor plan. In order to find the corner plots, Harris corner detection technique will be used. This leaves us with numerous corners that were seen on the plan, many of which are not doors. The algorithm will identify all corner without distinguish the outer or the inner points. The plot corner points are included in this process too.

In addition to that, corner points for the scaling box are identified too as shown in Figure 23. That's why it is important after calculating the scale of the walls and the areas to remove the scaling box to avoid any confusion in the floorplan recognition.

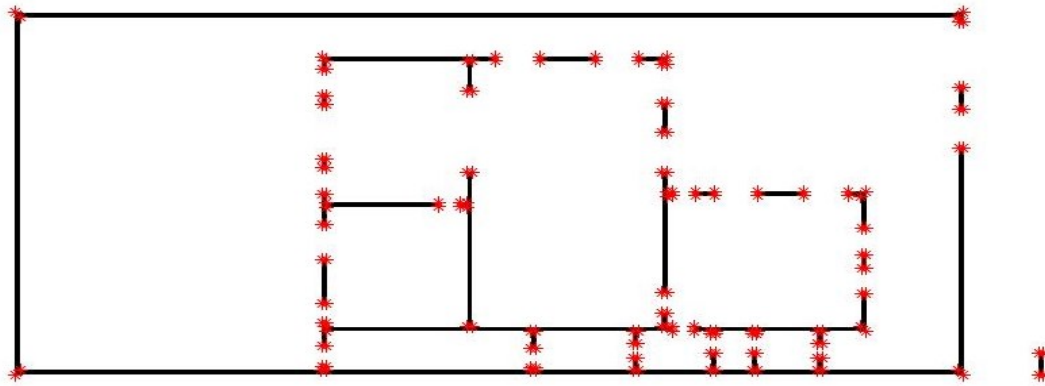


Figure 23. Finding corner points and removing scaling box.

The next step is to connect the doors and windows opening to form a solid floorplan. Now edges that each have two extremely close Harris corners surround each entrance. My goal is to completely cover the door space with color in order to clearly demarcate the several rooms, a step that is required for all following assignments. An algorithm calculates the midpoint between each pair of close points defining a door edge in order to accomplish this. It does this by first isolating the point pairs that are associated with a door. By applying a pixel condition on a mask centered at the door edge and calculating the ratio of 0-pixels to 1-pixels, the first goal is attained.

Since we have located the edge points of each door, we can move on to linking them to create lines. We can take advantage of the fact that door edges typically share the same x-coordinate or the same y-coordinate in order to

choose the appropriate door point pairs on the design. There are situations when doors are neither horizontally nor vertically oriented in relation to the drawing, thus this is obviously a simplification. Setting constraints for the door's dimensions on the plan and identifying the pairs that satisfy those conditions would be one way to get around this. But for the purposes of this example application, we suppose that every door will be specified by two edge points that either have the same x-coordinate or y-coordinate.

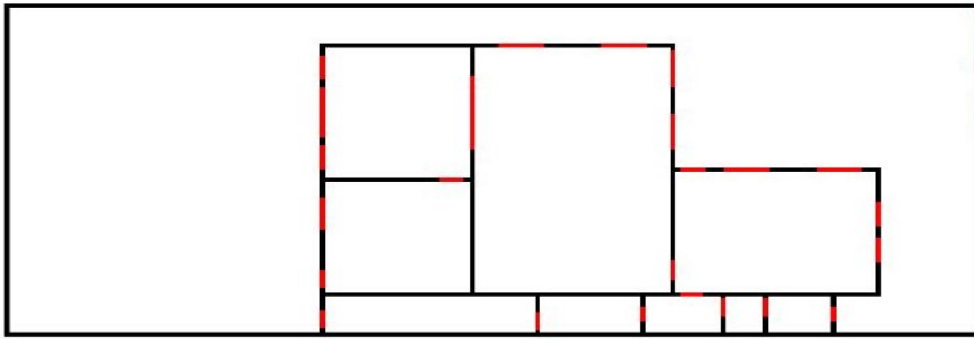


Figure 24. Doors & windows are closed.

Once chosen, all the couples that have a common coordinate are linked. The doors are now closed, allowing us to divide the image into its closed portions. We accomplish this by using Matlab's connected components and region characteristics function to acquire the division of the room as in Figure 24. The background is referred to as (room 0). The next step is to find the number of rooms and their boundary defining corner points. Each room defining corner points are then plotted. Subsequently, corner defining points are sorted for each room ascendingly and randomize a color for each room internal pixels. The rooms are then colored with that randomized color (currently the

seed value set is 0, if different colors are desired, a different seed value must be selected) as shown in Figure 25.

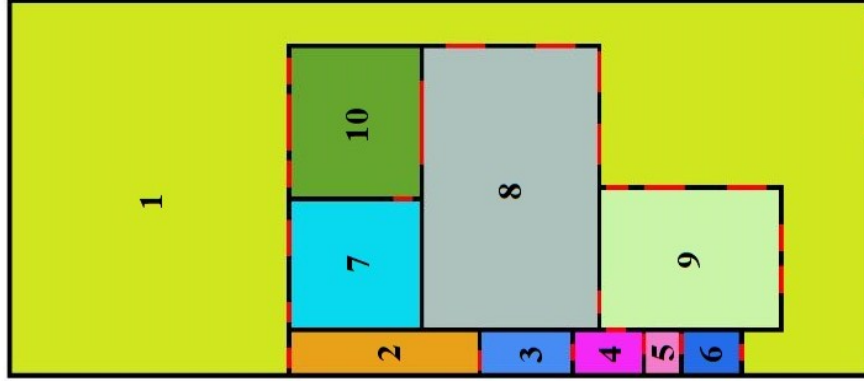


Figure 25. The floorplans are well recognized in terms of rooms.

Additionally, the algorithm is designed to find the areas for each room in the floor plan and the area of the plot. The difficulty faced in this step is defining the room which represents the plot and other rooms. The plot was defined as the outer room that has no other rooms inside it. Table 1 defines the output of the model with areas for each room and the plot.

Table 1. Areas for each room and the plot.

Room Number	Area in mm2
	From MATLAB
1	354721450.6
2	14888888.89
3	7086419.753
4	5308641.975
5	2641975.309

6	4419753.086
7	31697530.86
8	94629629.63
9	47722608.02
10	37395061.73
Plot	645828703.7

The final step in the model generation is to design the algorithm to be able to give its final decision regarding the entered floor plan whether it meets the regulation or no. section 5.3 will talk about the regulation of licensing a floor plan.

5.3 Law and Regulation in Floorplan

To understand the requirements of the current law and regulations related to licensing a new building permit by the concerned ministry agent, it is imperative to list all the steps that are being considered currently before issuing any new license. To perform that, and for the requirement of this research, the criteria of zoning Residential Area (RA) (Benayat 2022) will be considered. Multiple actions and steps during the audit of a new building permit with related floorplan are taking place before issuing the final approval. Zoning RA is the most popular zoning category in Bahrain since it is related for domestic houses. However, the following are the regulation for this zoning (Benayat 2022):

1. The building percentage shall not be more than 180% (one hundred and eighty percent) of the land area.

2. Built-up area of any floor shall not exceed 60% (sixty percent) of the land area.
3. Front Recession: Building shall be permitted on the ground floor at a distance at least 3 meters from the road.
4. Rear and Side Recessions: Building shall be permitted on the ground floor at a distance of at least 2 meters. From the roads.
5. Attaching the main building to one side of the boundary wall with a % not exceeding 35% from the length of the wall.

The above regulations were published on decision No. (55) for the year 2016 to amend some of the regulatory construction requirements in all areas of the Kingdom, promulgated by decision No. (28) for the year 2009. Decision No. (28) for the year 2009 to issue the regulatory constructions requirements in all areas of the Kingdom (Benayat 2022). The time required from an audit to review the floorplan to identify whether the floorplan is meeting the required criteria for zoning type RA might be very long. Beside this, human errors might occur during the audit of the floorplan.

5.4 Training of the Model

When the deep learning algorithm is trained, it properly interprets data and learns from it to perform the tasks with accuracy. Just like with humans, this takes time and patience (just consider all those worksheets had to be completed when learning multiplication tables back in grade school). Only by training the deep learning algorithm to correctly perceive information and

make accurate decisions based on the information provided, can then ensure it will perform the way it's intended.

First and foremost, algorithm training starts with data. The data to be used in this training is The CubiCasa5K dataset which was described in detail in the previous chapter. The CubiCasa5K dataset has high-quality, well-annotated data which will help the algorithm to learn effectively and perfume an accurate result. As shown in Figure 26, the algorithm training process has three key stages:

1. Training

In the initial training step, the model is given a set of training data (20 floor plans) and asked to make decisions based on that information. The decisions were asked is to find whether the Front recession and rear recession is matching the standards or no. mistakes at this stage is predictable and adjustment of the model is taking place in order to come up with better results learning from the floor plans entered. Accordingly, the number of floor plans is increased in the second stage (200 floor plans) and mistakes in this stage will be continued in finding the front and back recession. The number of floor plans increased in the third stage to 2000 floor plans, and the results at this stage were more accurate. Thus, the full set of data was entered into the model to complete the training of the deep learning algorithm.

In this cycle (the 4 stages) in case any error occurred in any of the cycles, the process then should start from beginning to ensure the accuracy of the model. For the case of this model, 26 times errors were observed, and the process restarted from beginning after doing the needed correction.

One issue to try and avoid is overfitting. That's the name of a common problem that happens when the model is aligned so closely with a specific dataset that it's memorized it rather than learned from it. In this instance, the model would be unlikely to interpret new data correctly. To avoid this problem, for every stage of the training, a deferent set of data was used to create variety and diversity.

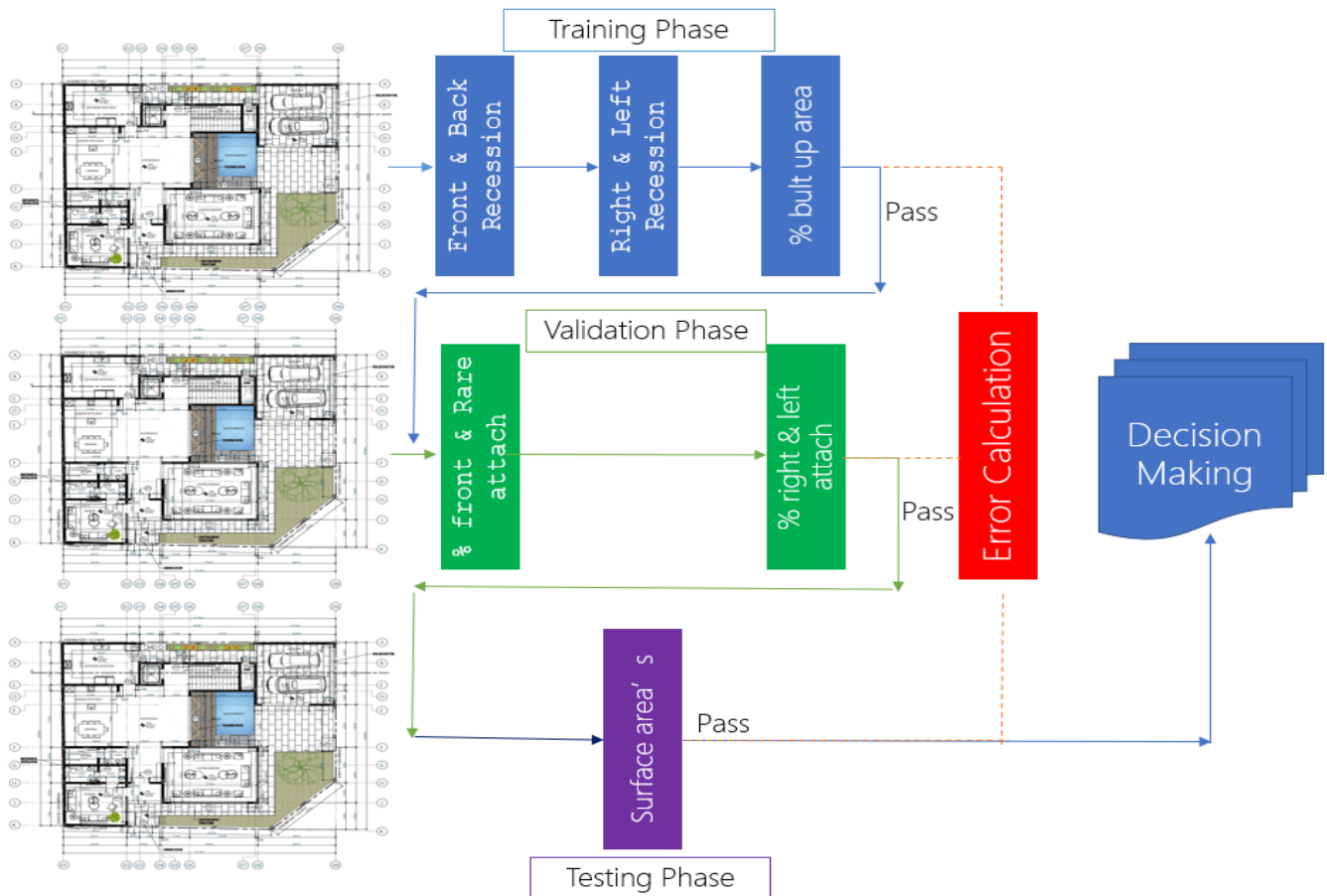


Figure 26. Algorithm training process three key stages.

2. Validation

In this phase, the model will be evaluated about how well it will perform using a new set of data. Therefore, new floor plans were entered, and the model were asked to give a decision to make sure that the results are accurate, and the

model is behaving as expected, and account for any new variables that might not have considered previously. Any issues with overfitting will likely become evident during the validation stage.

If any errors occurred in this stage, the training process should go back to the 1st stage to retrain the model with the required remedial correction of the model itself. Validation failed for 15 cycles and the process went back to stage 1 for the 3 times.

3. Testing

At this stage, the model should be ready to give the most accurate result by entering a complicated floor plan. The model is given a different set of floor plans and were asked for decisions. If the model can make accurate decisions based on this unstructured information, it is ready to go live! But if not, it's back to the training stage and the process is repeated until accurate results are obtained, and the model is performing as expected. For the third stage, errors occurred 7 times and the process started from the beginning.

Table 2. Number of cycles per each stage during training the model.

Stages	Training	Validation	Testing
Number of cycles	48	22	7

5.5 Model Execution and results generation

After training, validation and testing cycles are completed, it is time to insert floorplans to the model and generate the results in terms of decision-making

whether those floor plan are meeting Bahrain municipality regulations for zoning RA or no and can this floor plan be accepted and licensed or modification must take place based on the predefined criteria.

For this reason, an actual floor plans from Kingdom of Bahrain need to be used to feel the practicality and of the algorithm. Accordingly, a sample of 15 floor plans were selected randomly from approved and licensed consultant by The Council for Regulating the Practice of Engineering Professions (CRPEP) named as Sultan Engineering after obtaining the owners verbal approvals (appendix B). The samples were chosen from zoning RA and in the initial stage of licensing. i.e., not yet audited or approved by Bahrain Municipalities. The samples were received in AutoCAD format which will enable the extraction of any needed data for compression at a later stage. All labels in the floor plans were removed from the AutoCAD file and then the floor plans were saved as an image. Before inserting the sample floor plans to the model, OCR was used to remove any balance labels as mentioned in section 5.2.

The following are the output of the algorithm after inserting the floor plan as an input:

Floor plan No. 1:

- The percentage of built-up area is: (42.14%)
- The front recession is: (4m)
- The left side recession is: (2m)
- The right-side recession is: (2m)
- The rear recession is: (12.5m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (52.96%)

➤ The percentage of attached main building to right side boundary is: (0%)

➤ The percentage of attached main building to rear boundary is: (0%)

%% The Constraint/s which were not satisfied %

➤ The left side percentage of attached main building length should not exceed 35%, currently it is: (52.96%)

%% Decision %

➤ Floor plan No. 1 can't be accepted as it does not meet all the regulations.

Floor plan No. 2:

➤ The percentage of built-up area is: (76.05%)

➤ The front recession is: (3m)

➤ The left side recession is: (2m)

➤ The right-side recession is: (2m)

➤ The rear recession is: (2m)

➤ The percentage of attached main building to front boundary is: (0%)

➤ The percentage of attached main building to left side boundary is: (32.41%)

➤ The percentage of attached main building to right side boundary is: (0%)

➤ The percentage of attached main building to rear boundary is: (0%)

%% The Constraint/s which were not satisfied %

➤ Nil

%% Decision %

➤ Floor plan No. 2 can be accepted as it meets all the regulations.

Floor plan No. 3:

- The percentage of built-up area is: (66.86%)
- The front recession is: (3m)
- The left side recession is: (2m)
- The right-side recession is: (2m)
- The rear recession is: (2m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (41.95%)
- The percentage of attached main building to right side boundary is: (0%)
- The percentage of attached main building to rear boundary is: (0%)

%% The Constraint/s which were not satisfied %

- The left side percentage of attached main building length should not exceed 35%, currently it is: (41.95%)

%% Decision %

- Floor plan No. 3 can't be accepted as it does not meet all the regulations.

Floor plan No. 4:

- The percentage of built-up area is: (81.52%)
- The front recession is: (3m)
- The left side recession is: (2m)
- The right-side recession is: (2m)
- The rear recession is: (2m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (0%)

- The percentage of attached main building to right side boundary is: (51.62%)
- The percentage of attached main building to rear boundary is: (22.04%)

%% The Constraint/s which were not satisfied %%

- The right-side percentage of attached main building length should not exceed 35%, currently it is: (51.62.95%)
- The building should be attached to one side of the boundary walls, currently two sides.

%% Decision %%

Floor plan No. 4 can't be accepted as it does not meet all the regulations.

Floor plan No. 5:

- The percentage of built-up area is: (66.41%)
- The front recession is: (3m)
- The left side recession is: (2m)
- The right-side recession is: (2m)
- The rear recession is: (2m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (0%)
- The percentage of attached main building to right side boundary is: (34.44%)
- The percentage of attached main building to rear boundary is: (0%)

%% The Constraint/s which were not satisfied %%

- Nil

Decision

Floor plan No. 5 can be accepted as it meets all the regulations.

floor plan No. 6:

- The percentage of built-up area is: (78.58%)
- The front recession is: (3m)
- The left side recession is: (2m)
- The right-side recession is: (2m)
- The rear recession is: (2m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (0%)
- The percentage of attached main building to right side boundary is: (0%)
- The percentage of attached main building to rear boundary is: (25.20%)

The Constraint/s which were not satisfied

- Nil

Decision

Floor plan No. 6 can be accepted as it meets all the regulations.

floor plan No. 7:

- The percentage of built-up area is: (39.01%)
- The front recession is: (6m)

- The left side recession is: (2m)
- The right-side recession is: (2m)
- The rear recession is: (2m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (39.15%)
- The percentage of attached main building to right side boundary is: (0%)
- The percentage of attached main building to rear boundary is: (0%)

%% The Constraint/s which were not satisfied %%

- The left side percentage of attached main building length should not exceed 35%, currently it is: (39.15%)

%% Decision %%

- Floor plan No. 7 can't be accepted as it does not meet all the regulations.

floor plan No. 8:

- The percentage of built-up area is: (34.5%)
- The front recession is: (4.67m)
- The left side recession is: (2m)
- The right-side recession is: (3.2m)
- The rear recession is: (5m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (0%)
- The percentage of attached main building to right side boundary is: (12.95%)
- The percentage of attached main building to rear boundary is: (0%)

%% The Constraint/s which were not satisfied %

- Nil

%% Decision %

- Floor plan No. 8 can be accepted as it meets all the regulations.

floor plan No. 9:

- The percentage of built-up area is: (46.87%)
- The front recession is: (5.21m)
- The left side recession is: (2m)
- The right-side recession is: (2m)
- The rear recession is: (4m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (0%)
- The percentage of attached main building to right side boundary is: (0%)
- The percentage of attached main building to rear boundary is: (43.40%)

%% The Constraint/s which were not satisfied %

- The rear side percentage of attached main building length should not exceed 35%, currently it is: (43.40%)

%% Decision %

- Floor plan No. 9 can't be accepted as it is not meeting all the regulations.

floor plan No. 10:

- The percentage of built-up area is: (70.01%)
- The front recession is: (3m)
- The left side recession is: (2m)
- The right-side recession is: (1.9m)
- The rear recession is: (1.8m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (0%)
- The percentage of attached main building to right side boundary is: (17.54%)
- The percentage of attached main building to rear boundary is: (39.35%)

%% The Constraint/s which were not satisfied %%

- The rear side percentage of attached main building length should not exceed 35%, currently it is: (39.35%)
- The building should be attached to one side of the boundary walls, currently two sides.
- The right-side recession is less than 2 m: (1.9m)
- The rear recession is less than 2 m: (1.8m)

%% Decision %%

- Floor plan No. 10 can't be accepted as it is not meeting all the regulations.

floor plan No. 11:

- The percentage of built-up area is: (52.55%)
- The front recession is: (4.32m)
- The left side recession is: (2m)
- The right-side recession is: (4.2m)
- The rear recession is: (6.4m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (19.91%)
- The percentage of attached main building to right side boundary is: (0%)
- The percentage of attached main building to rear boundary is: (0%)

%% The Constraint/s which were not satisfied %%

- Nil

%% Decision %%

- Floor plan No. 11 can be accepted as it meets all the regulations.

floor plan No. 12:

- The percentage of built-up area is: (32.15%)
- The front recession is: (3.11m)
- The left side recession is: (2.3m)
- The right-side recession is: (2.2m)
- The rear recession is: (2m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (0%)
- The percentage of attached main building to right side boundary is: (32.15%)

➤ The percentage of attached main building to rear boundary is: (0%)
%% The Constraint/s which were not satisfied %%%

➤ Nil

%% Decision %%%

➤ Floor plan No. 12 can be accepted as it meets all the regulations.

floor plan No. 13:

- The percentage of built-up area is: (48.61%)
- The front recession is: (3m)
- The left side recession is: (2.05m)
- The right-side recession is: (2.1m)
- The rear recession is: (1.98m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (47.21%)
- The percentage of attached main building to right side boundary is: (0%)
- The percentage of attached main building to rear boundary is: (9.02 %)

%% The Constraint/s which were not satisfied %%%

- The left side percentage of attached main building length should not exceed 35%, currently it is: (39.35%)
- The building should be attached to one side of the boundary walls, currently two sides.
- The rear recession is less than 2 m: (1.98m)

Decision

- Floor plan No. 13 can't be accepted as it does not meet all the regulations.

floor plan No. 14:

- The percentage of built-up area is: (68.46%)
- The front recession is: (3m)
- The left side recession is: (2.02m)
- The right-side recession is: (2.01m)
- The rear recession is: (2.01m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (0%)
- The percentage of attached main building to right side boundary is: (29.25%)
- The percentage of attached main building to rear boundary is: (0%)

The Constraint/s which were not satisfied

- Nil

Decision

- Floor plan No. 14 can be accepted as it is meeting all the regulations.

floor plan No. 15:

- The percentage of built-up area is: (58.35%)
- The front recession is: (4.74m)
- The left side recession is: (2.95m)
- The right-side recession is: (2.01m)

- The rear recession is: (4.52m)
- The percentage of attached main building to front boundary is: (0%)
- The percentage of attached main building to left side boundary is: (26.47%)
- The percentage of attached main building to right side boundary is: (0%)
- The percentage of attached main building to rear boundary is: (12.54%)

%% The Constraint/s which were not satisfied %%%

- The building should be attached to one side of the boundary walls, currently two sides.

%% Decision %%%

- Floor plan No. 15 can't be accepted as it does not meet all the regulations.

5.6 Actual dimensions of floorplan

The engineering office were asked to calculate the dimensions of each floorplan on manual bases by measuring length and width for each floorplan using an AutoCAD file. The dimension calculation was done by the office engineers and verified by his superior in order to ensure the accuracy of those dimensions. Table 3 to 17 shows the actual dimensions for each floorplan.

Table 3. Actual dimension for plan no. 1.

Detail	Actual Dimension
% of built-up area	42.11%
Front recession (m)	4m
Left side recession (m)	2m
Right side recession (m)	2m

Rear recession (m)	12.6
% of front attach	0%
% of left attach	52.82%
% of right attach	0%
% of rear attach	0%

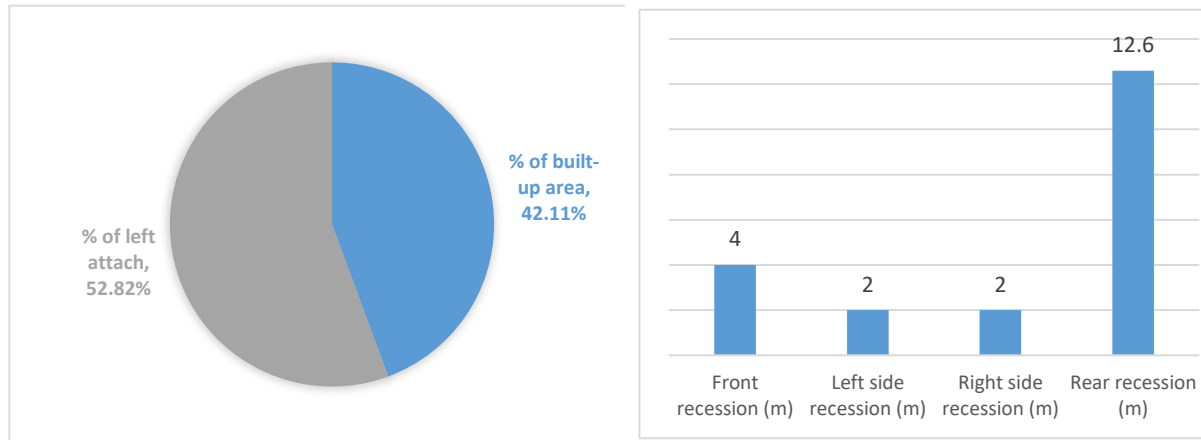


Table 4. Actual dimension for floor plan no. 2.

Detail	Actual Dimension
% of built-up area	76.12%
Front recession (m)	3.02
Left side recession (m)	2
Right side recession (m)	1.99
Rear recession (m)	1.97
% of front attach	0%
% of left attach	32.40%
% of right attach	0%
% of rear attach	0%

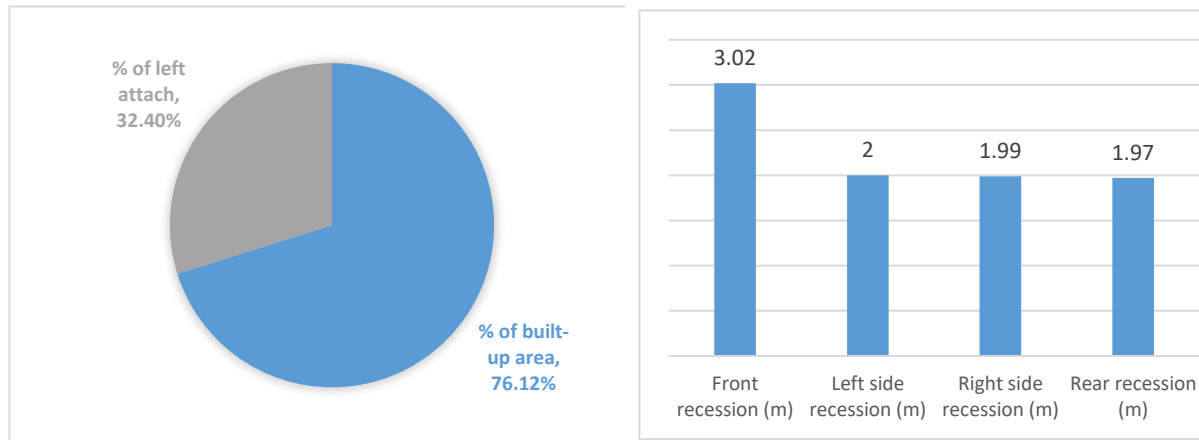


Table 5. Actual dimension for floor plan no. 3.

Detail	Actual Dimension
% of built-up area	66.84%
Front recession (m)	3
Left side recession (m)	2
Right side recession (m)	2
Rear recession (m)	2
% of front attach	0%
% of left attach	41.98%
% of right attach	0%
% of rear attach	0%

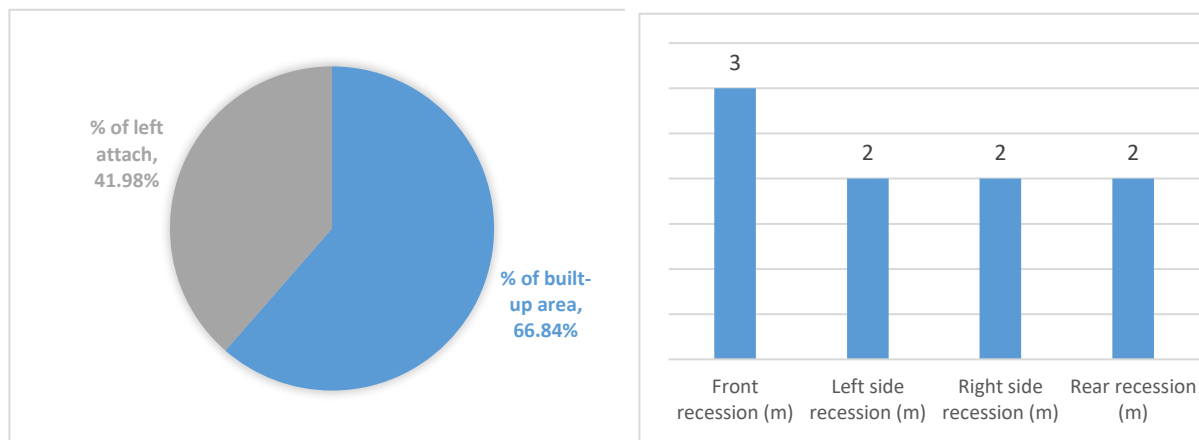


Table 6. Actual dimension for floor plan no. 4.

Detail	Actual Dimension
% of built-up area	81.32%
Front recession (m)	2.95
Left side recession (m)	2.07
Right side recession (m)	1.98
Rear recession (m)	2.06
% of front attach	0%
% of left attach	0%
% of right attach	51.58%
% of rear attach	22.08%

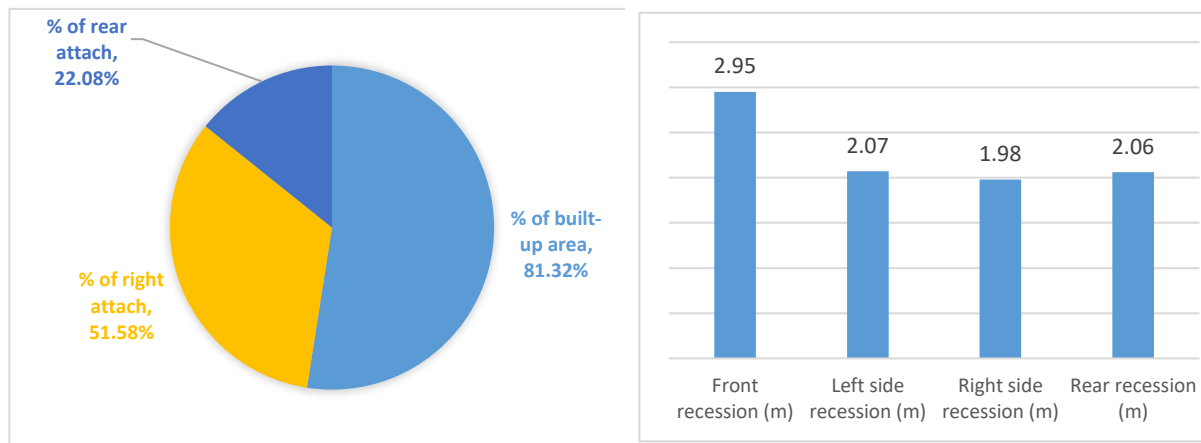


Table 7. Actual dimension for floor plan no. 5.

Detail	Actual Dimension
% of built-up area	66.42%
Front recession (m)	3
Left side recession (m)	2.01
Right side recession (m)	2
Rear recession (m)	2
% of front attach	0%
% of left attach	0%
% of right attach	34.42%
% of rear attach	0%

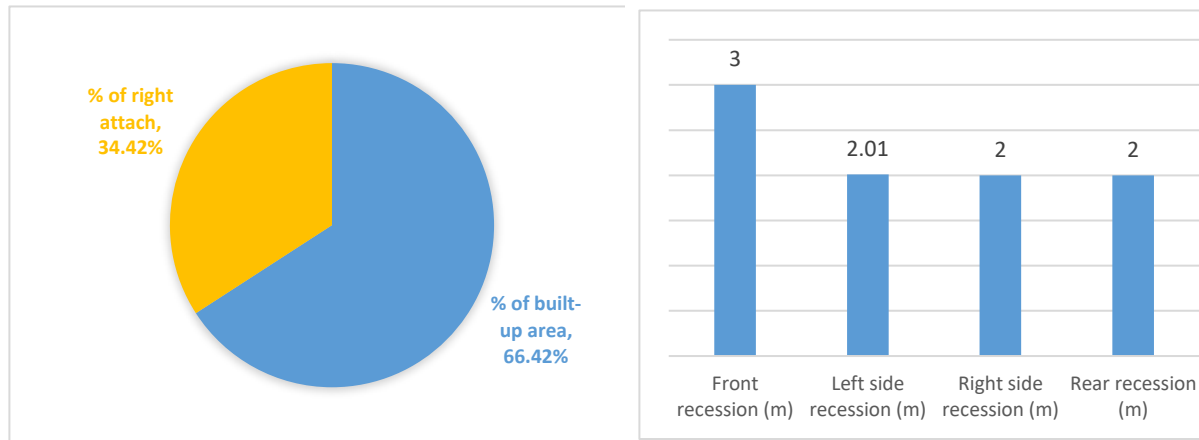


Table 8. Actual dimension for floor plan no. 6.

Detail	Actual Dimension
% of built-up area	78.57%
Front recession (m)	3
Left side recession (m)	2
Right side recession (m)	2
Rear recession (m)	2
% of front attach	0%
% of left attach	0%
% of right attach	0%
% of rear attach	25.23%

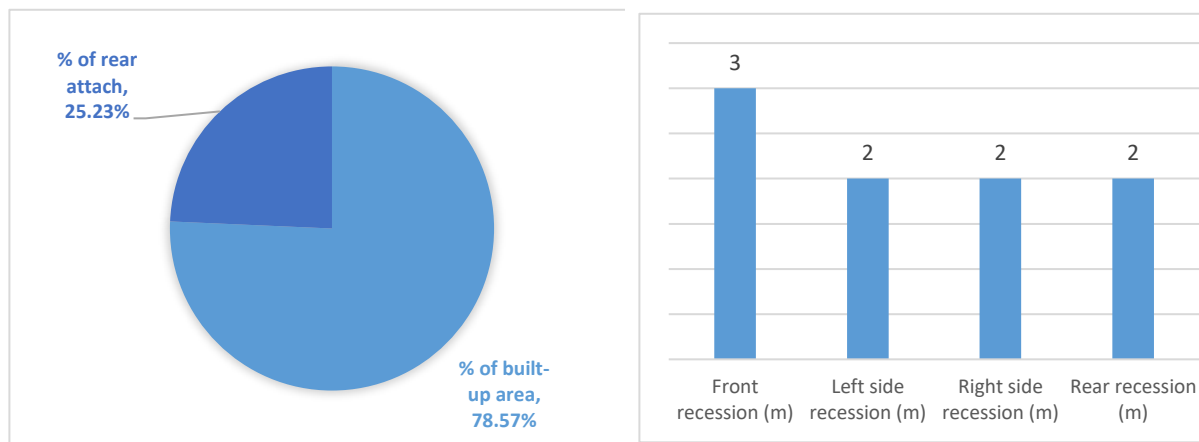


Table 9. Actual dimension for floor plan no. 7.

Detail	Actual Dimension
% of built-up area	38.99%
Front recession (m)	6
Left side recession (m)	2
Right side recession (m)	2
Rear recession (m)	1.98
% of front attach	0%
% of left attach	39.20%
% of right attach	0%
% of rear attach	0%

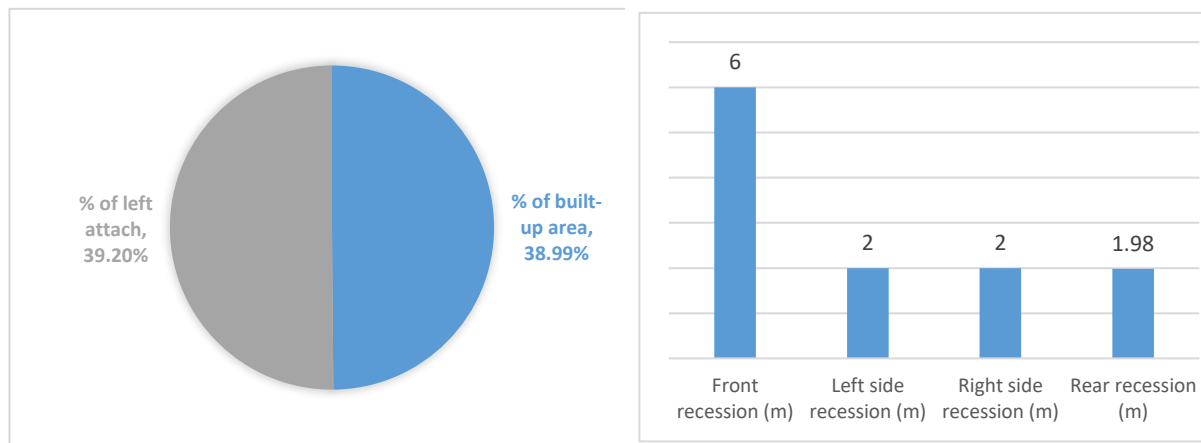


Table 10. Actual dimension for floor plan no. 8.

Detail	Actual Dimension
% of built-up area	34.45%
Front recession (m)	4.65
Left side recession (m)	2
Right side recession (m)	3.19
Rear recession (m)	5
% of front attach	0%
% of left attach	0%
% of right attach	12.93%
% of rear attach	0%

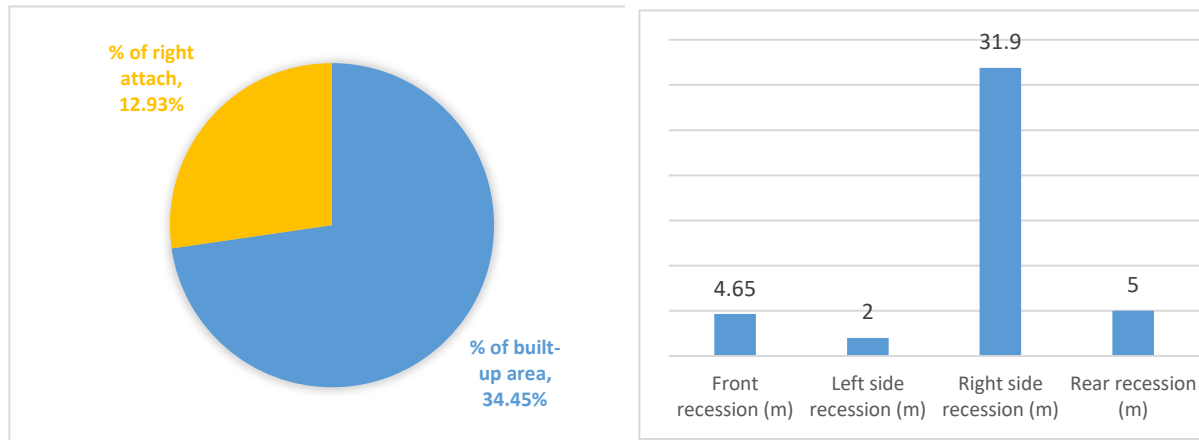


Table 11. Actual dimension for floor plan no. 9.

Detail	Actual Dimension
% of built-up area	46.79%
Front recession (m)	5.19
Left side recession (m)	2
Right side recession (m)	2
Rear recession (m)	0
% of front attach	0%
% of left attach	0%
% of right attach	0%
% of rear attach	43.44%

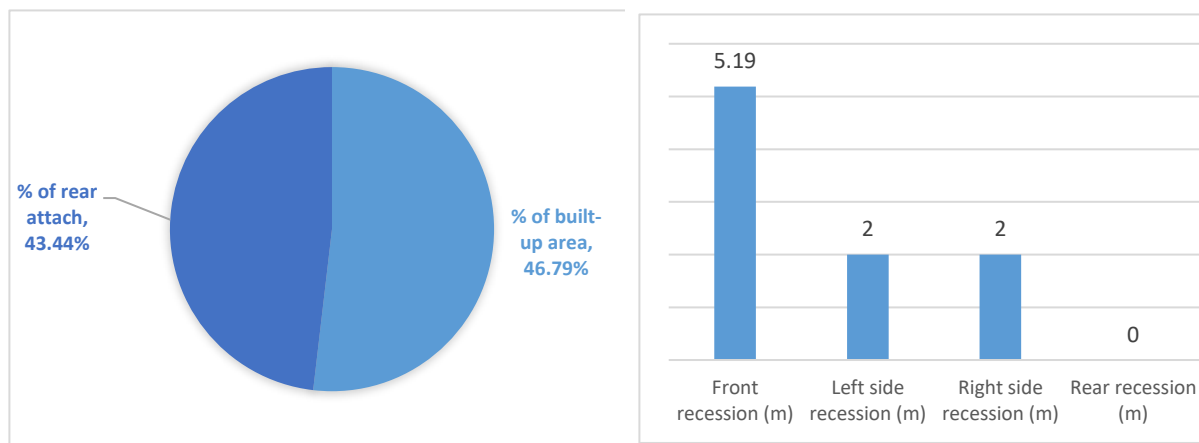


Table 12. Actual dimension for floor plan no. 10.

Detail	Actual Dimension
% of built-up area	70.06%
Front recession (m)	3
Left side recession (m)	1.97
Right side recession (m)	1.89
Rear recession (m)	1.84
% of front attach	0%
% of left attach	0%
% of right attach	17.69%
% of rear attach	39.28%

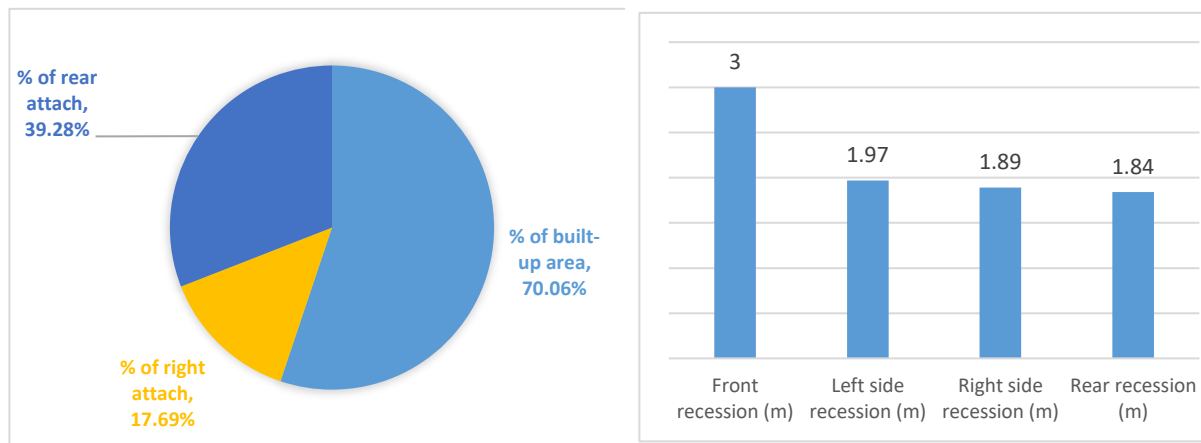


Table 13. Actual dimension for floor plan no. 11.

Detail	Actual Dimension
% of built-up area	52.38%
Front recession (m)	4.21
Left side recession (m)	2.14
Right side recession (m)	4.17
Rear recession (m)	6.44
% of front attach	0%
% of left attach	19.84%
% of right attach	0%
% of rear attach	0%

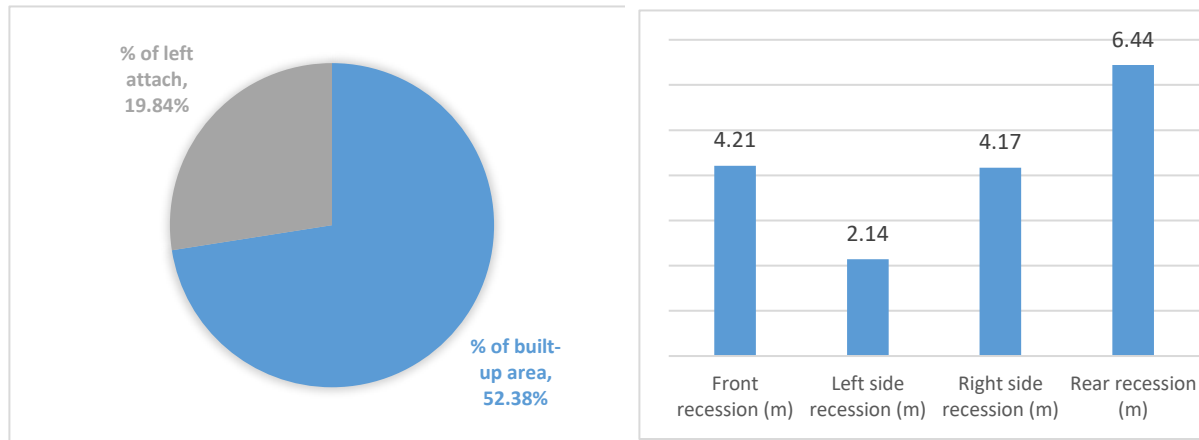


Table 14. Actual dimension for floor plan no. 12.

Detail	Actual Dimension
% of built-up area	32.14%
Front recession (m)	3.08
Left side recession (m)	2.21
Right side recession (m)	2.2
Rear recession (m)	2
% of front attach	0%
% of left attach	0%
% of right attach	32.14%
% of rear attach	0%

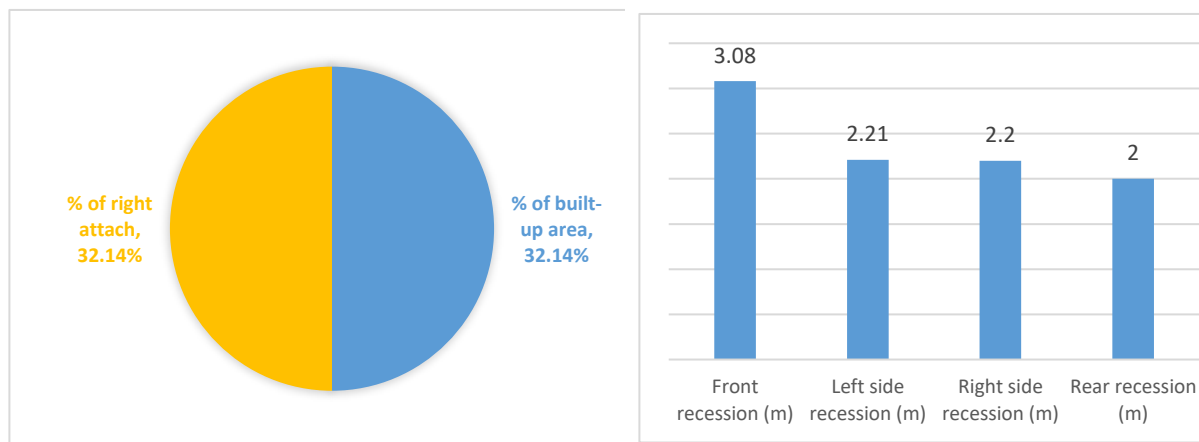


Table 15. Actual dimension for floor plan no. 13.

Detail	Actual Dimension
% of built-up area	58.66%
Front recession (m)	3
Left side recession (m)	2.03
Right side recession (m)	2.1
Rear recession (m)	2
% of front attach	0%
% of left attach	47.22%
% of right attach	0%
% of rear attach	8.97%

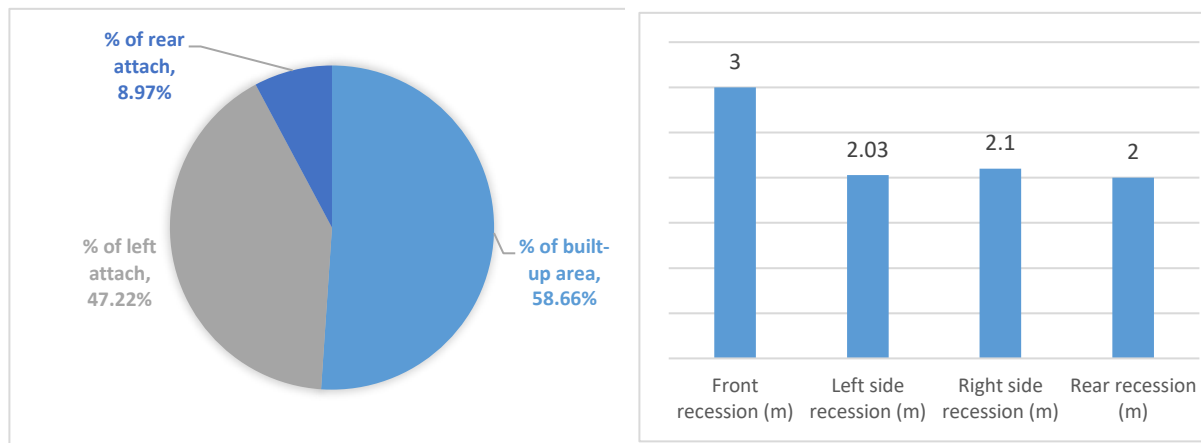


Table 16. Actual dimension for floor plan no. 14.

Detail	Actual Dimension
% of built-up area	68.41%
Front recession (m)	3
Left side recession (m)	2
Right side recession (m)	2
Rear recession (m)	2
% of front attach	0%
% of left attach	0%
% of right attach	29.24%
% of rear attach	0%

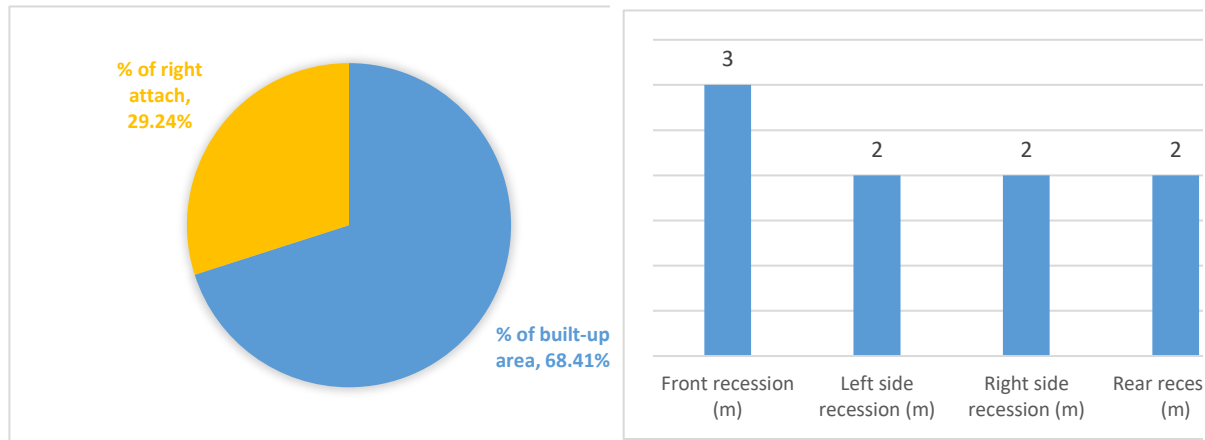
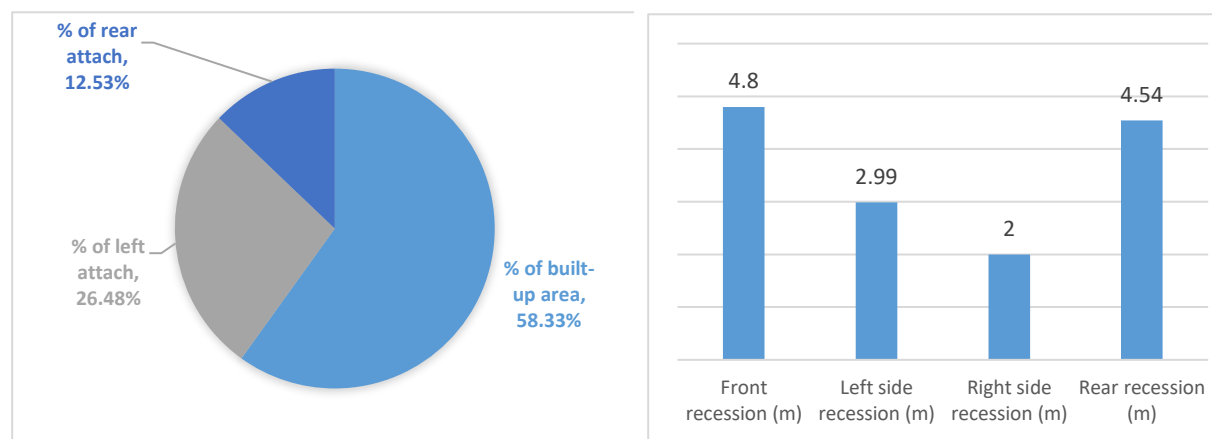


Table 17. Actual dimension for floor plan no. 15.

Detail	Actual Dimension
% of built-up area	58.33%
Front recession (m)	4.8
Left side recession (m)	2.99
Right side recession (m)	2
Rear recession (m)	4.54
% of front attach	0%
% of left attach	26.48%
% of right attach	0%
% of rear attach	12.53%



5.7 Outer Surface Area Extraction

The algorithm was designed to find out the areas in square meters of the outer surface of the floor plans. The outer surface area has two types in any floor plan. It could be flexible opened surface such as windows or doors or it might be a solid surface such as block, concrete, steel, or wood. Subsequently, the results of the algorithm output for the 15 floorplans are shown in Table 18 to Table 32. Since the algorithm can detect only two dimensions (X & Y) and can't detect Z dimensional, the height of each floor was assumed to be 3 meters fixed for each of the floor plans. This assumption was introduced in the algorithm calculation and it could be changed for any other floor plan which has different floor plan heights.

Table 18. Surface area of floor plan no. 1.

Details	Algorithm output (m^2)
Total area of external walls	220.5
Area of external wall openings	68.8
Area of external wall blocks	151.7
Area of the Roof	217

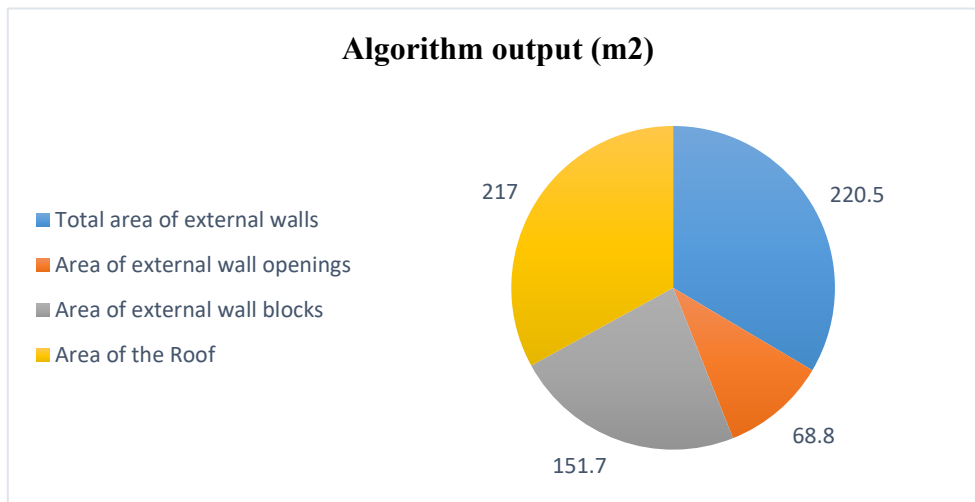


Table 19. Surface area of floor plan no. 2.

Details	Algorithm output (m^2)
Total area of external walls	324.7
Area of external wall openings	70.4
Area of external wall blocks	254.3
Area of the Roof	215

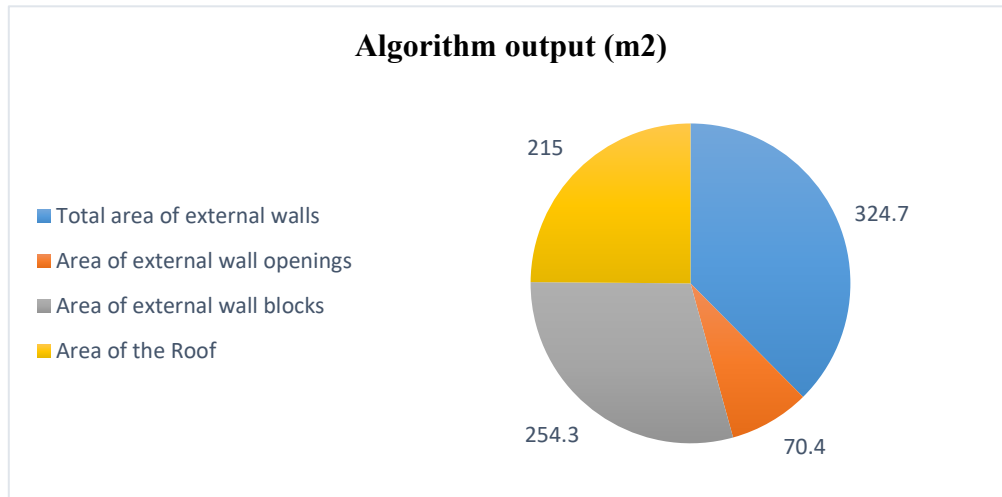


Table 20. Surface area of floor plan no. 3.

Details	Algorithm output (m^2)
Total area of external walls	498.6
Area of external wall openings	104.2
Area of external wall blocks	394.4
Area of the Roof	301

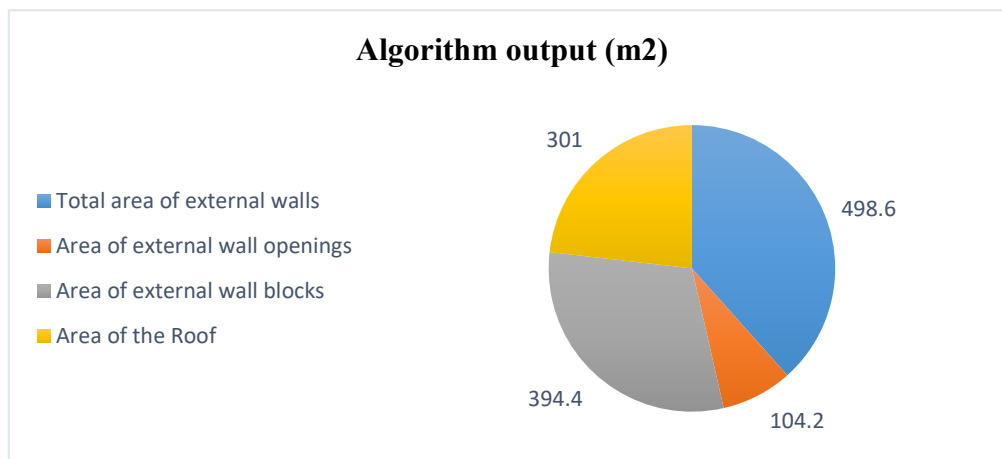


Table 21. Surface area of floor plan no. 4.

Details	Algorithm output (m^2)
Total area of external walls	275.2
Area of external wall openings	78.7
Area of external wall blocks	196.5
Area of the Roof	201

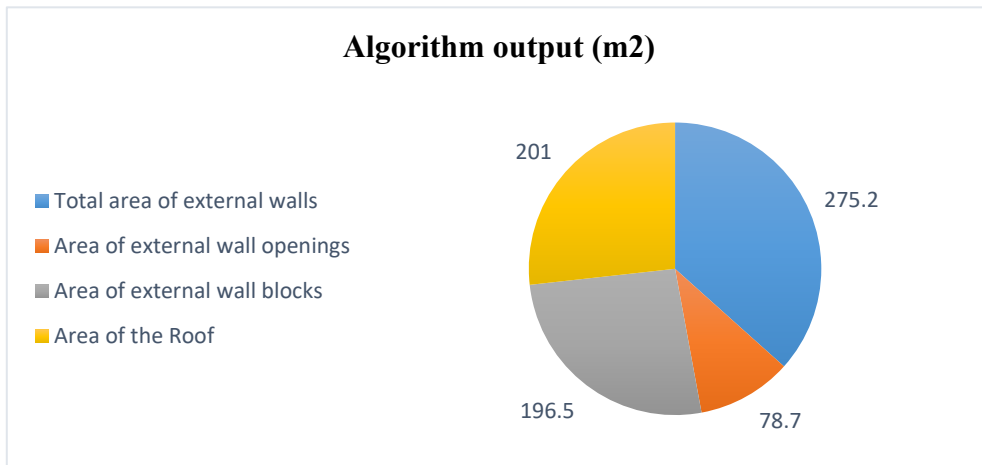


Table 22. Surface area of floor plan no. 5.

Details	Algorithm output (m^2)
Total area of external walls	422
Area of external wall openings	108.2
Area of external wall blocks	313.8
Area of the Roof	332

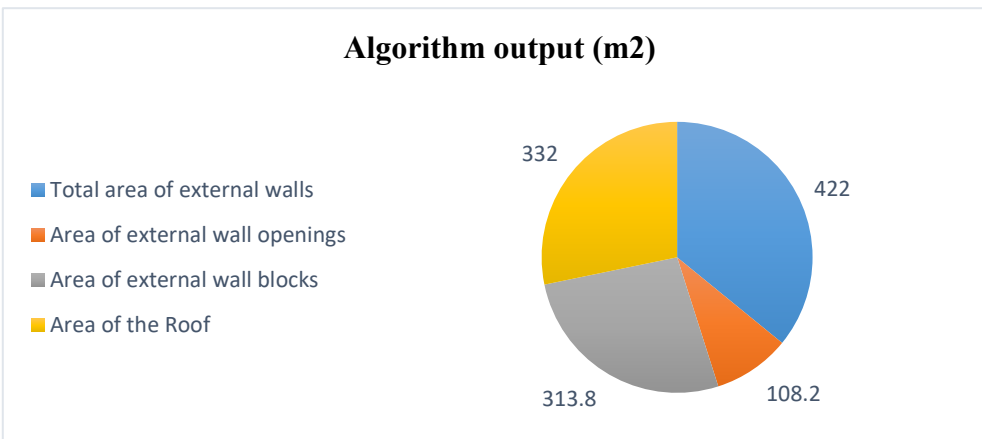


Table 23. Surface area of floor plan no. 6.

Details	Algorithm output (m^2)
Total area of external walls	379.7
Area of external wall openings	94.4
Area of external wall blocks	285.3
Area of the Roof	307

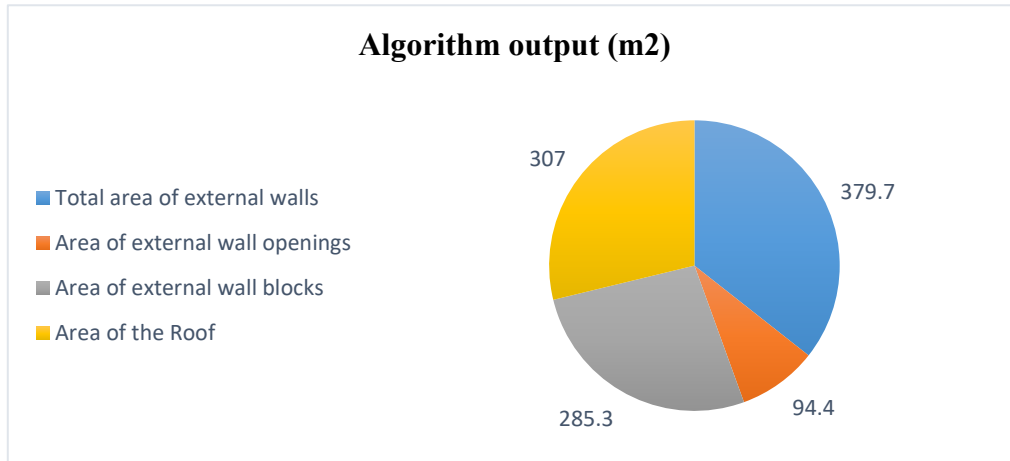


Table 24. Surface area of floor plan no. 7.

Details	Algorithm output (m^2)
Total area of external walls	293.2
Area of external wall openings	94.7
Area of external wall blocks	198.5
Area of the Roof	258

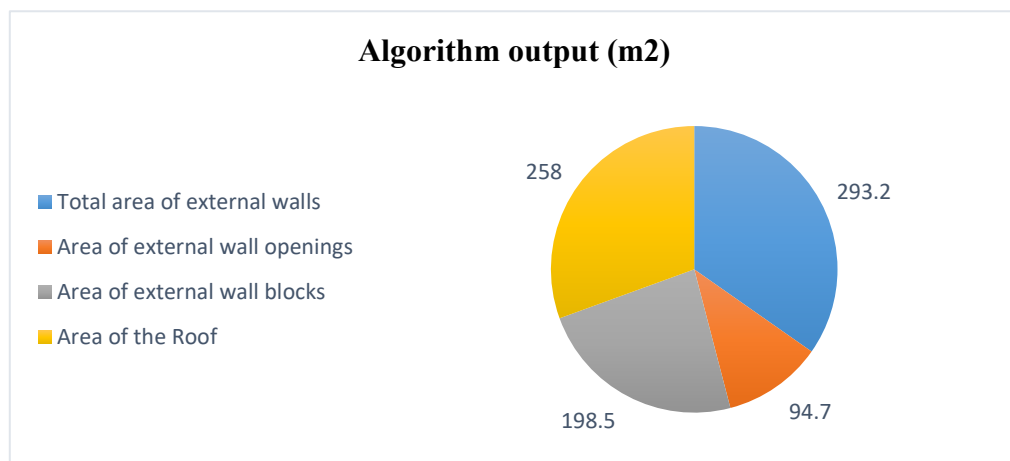


Table 25. Surface area of floor plan no. 8.

Details	Algorithm output (m^2)
Total area of external walls	444.5
Area of external wall openings	112.7
Area of external wall blocks	331.8
Area of the Roof	327

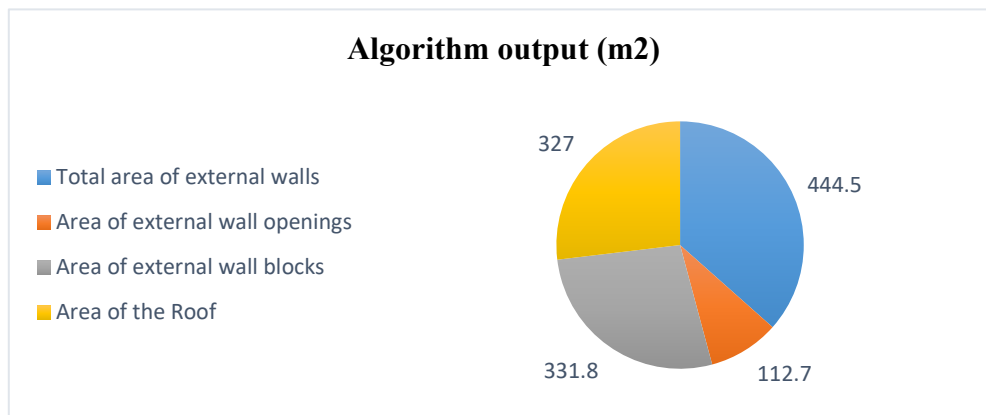


Table 26. Surface area of floor plan no. 9.

Details	Algorithm output (m^2)
Total area of external walls	330.7
Area of external wall openings	94.6
Area of external wall blocks	236.1
Area of the Roof	296

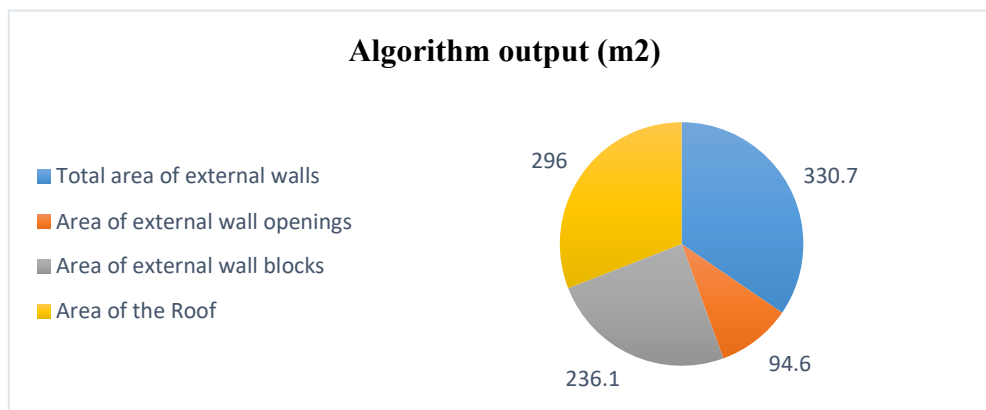


Table 27. Surface area of floor plan no. 10.

Details	Algorithm output (m^2)
Total area of external walls	266.2
Area of external wall openings	63.8
Area of external wall blocks	202.4
Area of the Roof	234

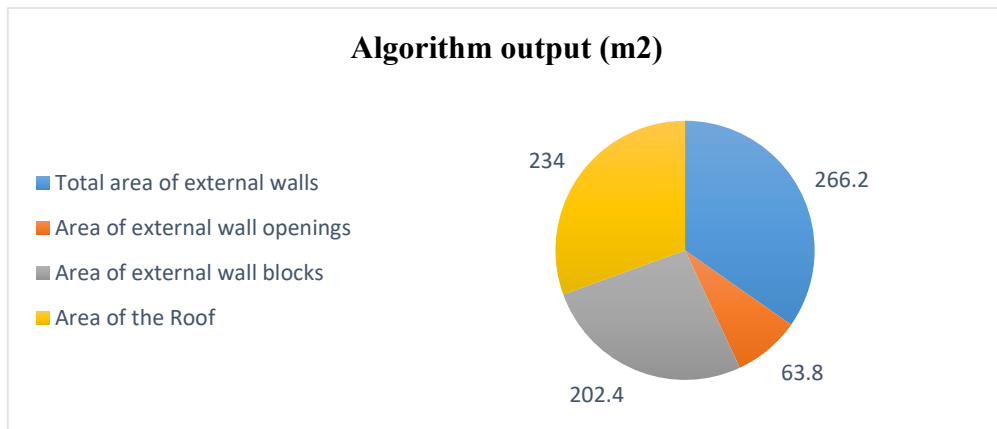


Table 28. Surface are of floor plan no. 11.

Details	Algorithm output (m^2)
Total area of external walls	421.5
Area of external wall openings	117.6
Area of external wall blocks	303.9
Area of the Roof	317

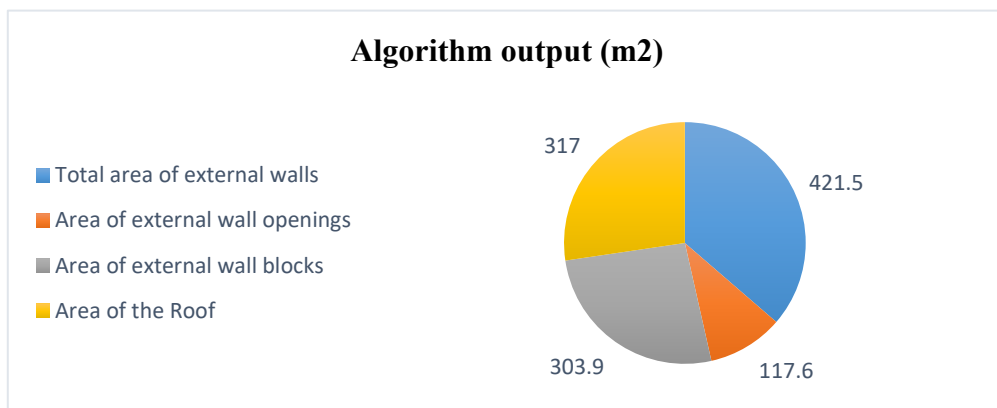


Table 29. Surface area of floor plan no. 12.

Details	Algorithm output (m^2)
Total area of external walls	346
Area of external wall openings	83.1
Area of external wall blocks	262.9
Area of the Roof	262

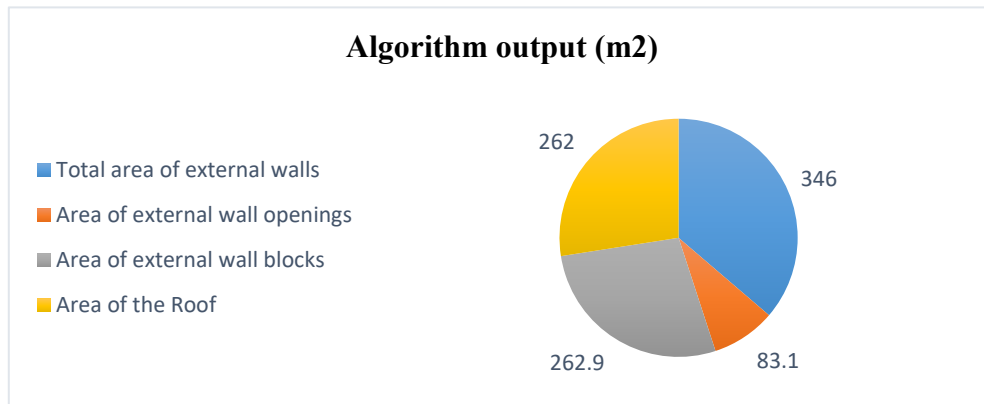


Table 30. Surface area of floor plan no. 13.

Details	Algorithm output (m^2)
Total area of external walls	283.1
Area of external wall openings	57.8
Area of external wall blocks	225.3
Area of the Roof	189

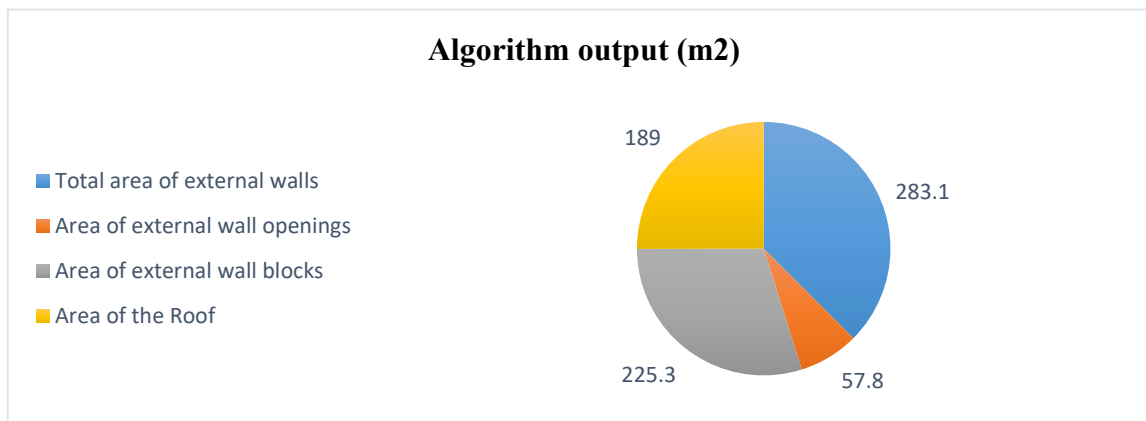


Table 31. Surface area of floor plan no. 14.

Details	Algorithm output (m^2)
Total area of external walls	331.1
Area of external wall openings	80.3
Area of external wall blocks	250.8
Area of the Roof	197

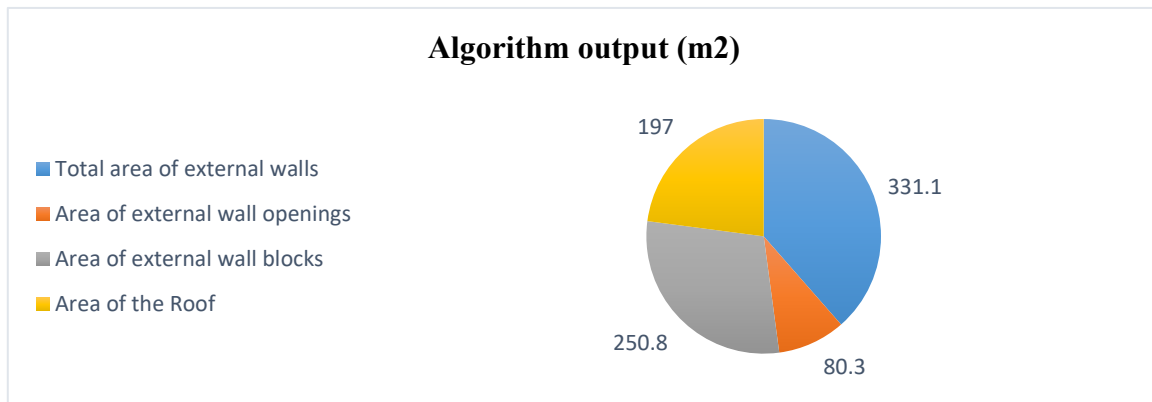
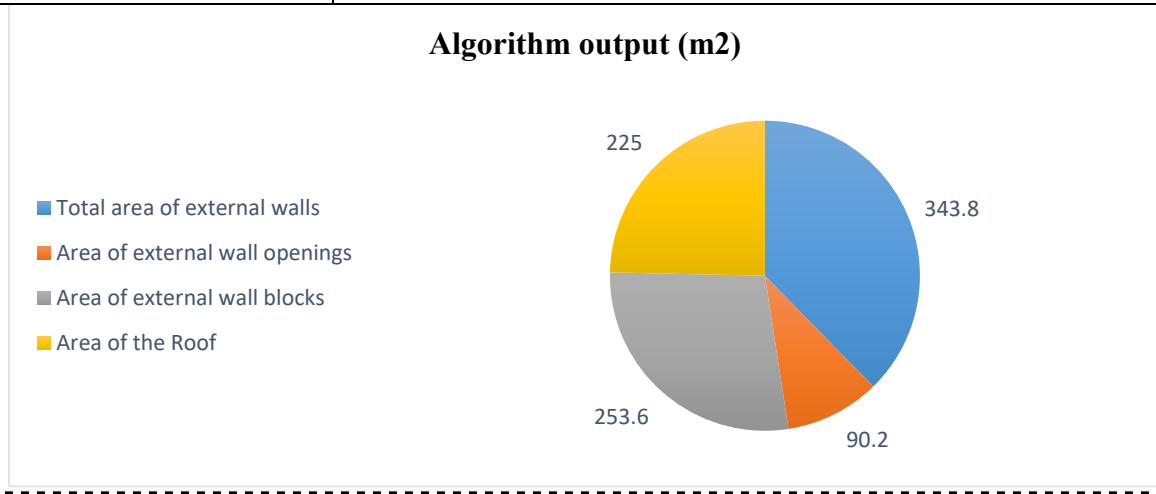


Table 32. Surface area of floor plan no. 15.

Details	Algorithm output (m^2)
Total area of external walls*	343.8
Area of external wall openings	90.2
Area of external wall blocks	253.6
Area of the Roof	225



Chapter 6

Floor Plan Heat Gain Management

by Using GIS Data

Chapter 6 Floor Plan Heat Gain Management by Using GIS Data

6.1 Heat Gain in Building and U-Value

The entire amount of heat that moves through a building's structure, whether from inside to outside which is known as heat loss or from outside to inside through conduction, convection, radiation, or any combination of these, is known as heat gain. The energy needed to maintain a space at a certain temperature on the coldest days is represented by this heat loss or gain figure, which is typically expressed in kW or BTUs.

Understanding how heat gain might impact an air conditioning cooling installation is crucial because a system must offer enough cooling to be used as the primary source of cooling and to boost comfort when used in conjunction with other cooling systems.

As mentioned, heat gain is a function of heat transfer from inside the building to the outside and is expressed in kW or BTUs.

U-values are used to measure the rate of heat transfer through walls, floors, and roofs. The U-value, which represents the building's overall heat transfer efficiency, shows how well various structural components conduct heat. A building with a low U-value has good insulation, while one with a high value has poor thermal performance.

U values are crucial since they serve as the foundation for any norm relating to energy or carbon reduction. These metrics assess if a building complies

with legal requirements. Any construction project must receive regulatory permission, typically from the Bahraini Electricity and Water Authority or local governments elsewhere in the world.

The composition and thickness of each building component need to be known, such as a wall, floor, or roof, to determine the U value of that component. The measurement of each building material's conductivity or heat transfer is the other essential property you need to acquire.

Buildings use energy for ventilation, lighting, cooking, cleaning, and recreation. More than 65% of energy is used by air conditioning, according to reports and research by Bahrain's Electricity and Water Authority (EWA 2023). Three key factors affect the rate of energy consumption by air conditioning:

1. Thermal performance of the building, which is affected by several factors, such as building form, building orientation, glazed surface areas, and thermophysical properties of building materials of the envelope of the building.
2. User's behavior in terms of controlling air-conditioning, lighting and other equipment.
3. General policy of the nation with respect to energy cost, building rules and regulations.

There is no doubt that the energy required to meet the requirements for indoor thermal comfort is significantly influenced by the shape, layout, design, and thermophysical qualities of the building materials utilized. As a result, it is the responsibility of architects and designers to carry out the necessary analytical studies that enable them to take advantage of the climatic elements with the

aim of minimizing the need for air conditioning and maximizing the use of natural illumination.

According to studies (EWA 2023), one of the key factors affecting the overall thermal performance is the thermal properties of the building envelope. A code for thermal insulation is established as a result. In order to lessen heat movement through the building envelope, the code addresses the thermal properties of roofs, outside walls, and glazed surfaces. To achieve this, the U-value for external walls and roofs is restricted, and the type of glass used in windows and other openings is specified. When it comes to lowering the rate of heat transmission from outside to inside during the hot summer and from inside to outside during the cold winter, the use of insulating materials is seen to be the most effective.

Heat transfer to and from the building takes place through the following:

1. Walls.
2. Windows and glazed surfaces.
3. Roof.

Between 60 and 70% of heat is transferred through walls throughout the summer. Air conditioning should be used to eliminate this level of heat. As a result, using insulating materials for walls and roofs is crucial for preserving energy.

There are various benefits to thermal insulation, including:

1. Reducing the energy consumption required for cooling and heating.
2. Reducing the capacity of air-conditioning equipment and hence reducing capital costs.

3. Protecting the building components from thermal stress because of expansion and contraction.
4. Reducing the electricity bill for consumers.
5. The provision of comfortable indoor thermal environment.
6. Protecting the surrounding environment from harmful gases emitted by electric power plants.

With relation to heat transfer through conduction and radiation, windows and glazed surfaces are thought to be the weakest locations. Decrease the area of glass surfaces that are exposed to the elements and direct sunlight instead and promote the use of double glazing and high-performance glass.

The general heat gain Formula is:

$$Q = U * A * \Delta T \quad (20)$$

U is the $U - Value$ of the materials used.

A is the surface area of a building.

ΔT is the deference in the temperature from inside the building to the outside of the building. In this research, the inner temperature will be assumed to be 25 Celsius degrees. Figure 27 illustrates the overview for heat gain calculations in this research.

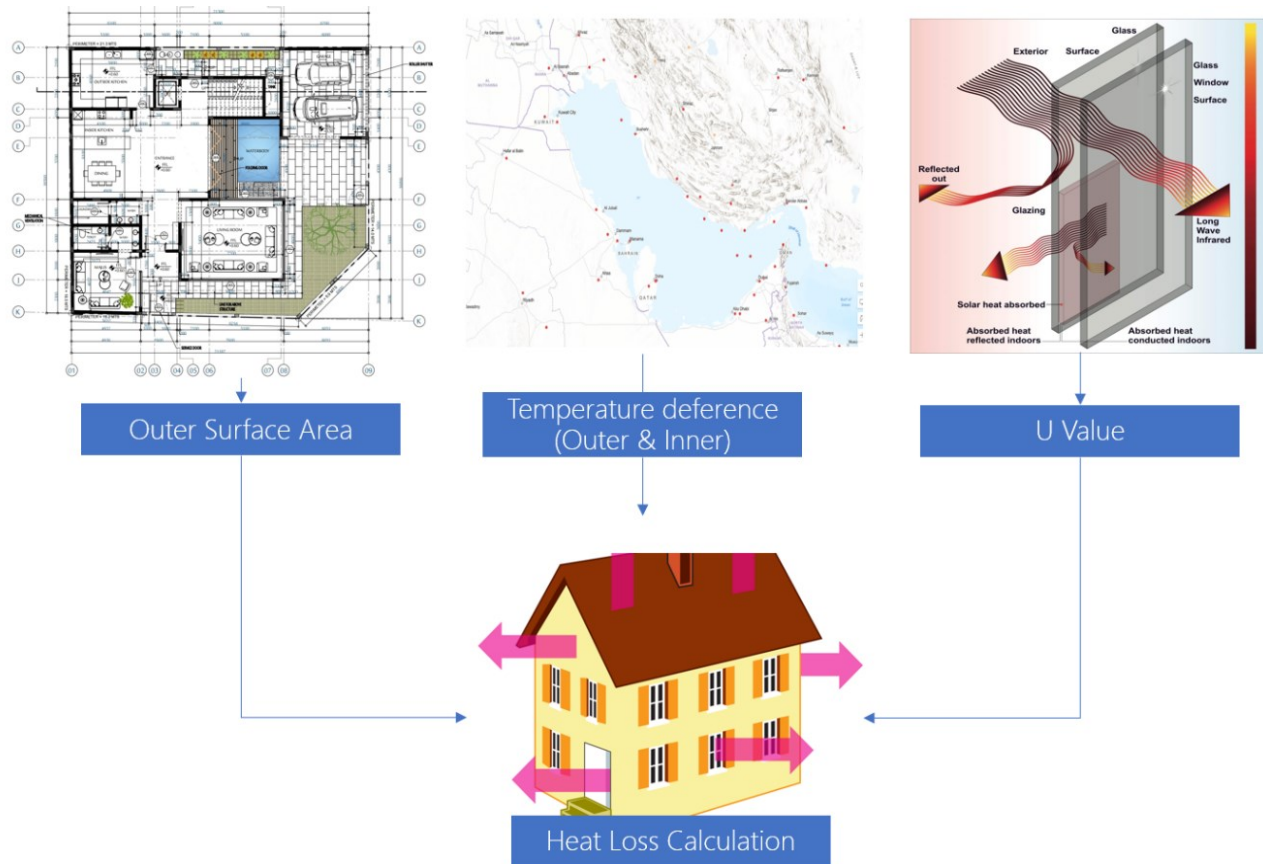


Figure 27. Heat gain calculation workflow.

6.1.1 Thermal Insulation Regulation in Bahrain

Thermal Insulation Order # 8/1999: Was issued in 1999 by H.E. The Minister of Housing & Municipality making it compulsory to provide thermal insulation for all buildings in Kingdom of Bahrain from 5 floors and above, which require air-conditioning. The Order stipulates the following requirements:

A. Thermal insulation materials should be used for walls of all buildings which require air-conditioning according to the following:

1. The overall U-value for external walls should not exceed 0.57 W/m² - oC.
2. High performance glass should be used for all buildings with more than three floors or if the area of the glazed surfaces ranges between 10-20% of the total external surface area of the building envelope. On the other hand, if the glazed area is more than 20%, double glazing should be used.

B. This regulation is applicable to all new buildings, which need air conditioning, and for existing buildings that need to be reconstructed or refurbished. The order has been implemented for buildings above four floors.

Thermal Insulation Order # 63/2012: Was issued in 2012 by H.E. The Minister of Housing & Municipality making it compulsory to provide thermal insulation for all residential buildings, facilities, warehouses, and stores that need cooling/air-conditioning. Accordingly, all buildings below 5 floors which were not covered in the previous thermal insulation order (8/1999) are now required to be provided with thermal insulation. This order has come effective on 1st September 2013, and the following are the requirements of thermal insulation:

- a. Thermal insulation shall be provided for all external walls including exposed columns, beams, staircases, and light wells/shafts. External walls of the building abutting adjoining building(s) if any shall also be insulated.
- b. Floors and walls of air-conditioned spaces exposed to non-air-conditioned spaces like car park/service areas in the building should be insulated.
- c. Spandrel areas of curtain walls should be insulated.

Thermal Insulation Order # 149/2018: Was issued on 31st of October 2018 by H.E. Minister of Works, Municipalities Affairs and Urban Planning. It was published in the official Gazette on 1st of November 2018. The order is legally binding effective 1st March 2019. The main clauses of the technical regulation attached to the order are summarized in table no. 33.

Table 33. Maximum Thermal Transmittance Value (U-Value).

Details	Maximum of Thermal Transmittance Value U Value (W/m².C)
Solid walls	0.57
Glass percentage less than or equal 40%	2.1
Glass percentage more than 40%	1.9
Doors	1.9
Roof	0.3

For this research, the values in table 33 will be considered for the calculation of the heat gain in the floor plans.

6.2 GIS and Temperature Deference

ArcGIS is a powerful geographic information system (GIS) software developed by Esri. It provides a comprehensive suite of tools and capabilities for working with spatial data, allowing users to analyze, visualize, and manage geographic information effectively. ArcGIS is widely used in various industries and sectors, including government, environmental management, urban planning, natural resource exploration, transportation, and many others.

ArcGIS Online has a cloud-based platform that enables users to access and share GIS data, maps, and applications from anywhere. It provides a collaborative environment for creating and sharing maps, conducting analysis, and publishing data on the web.

In continuation of finding the temperature deference for the building in order to calculate the heat gain of the floor plan, it is must to upload live temperature data on the ArcGIS software to examine the location of the floor plan and find the nearest weather station which can provide the outside temperature. The real data temperature was downloaded from (ArcGIS 2023) which contain more than 1754 station that send live temperature data every three hours. The data were uploaded on the ArcGIS as shown in Figure 28.

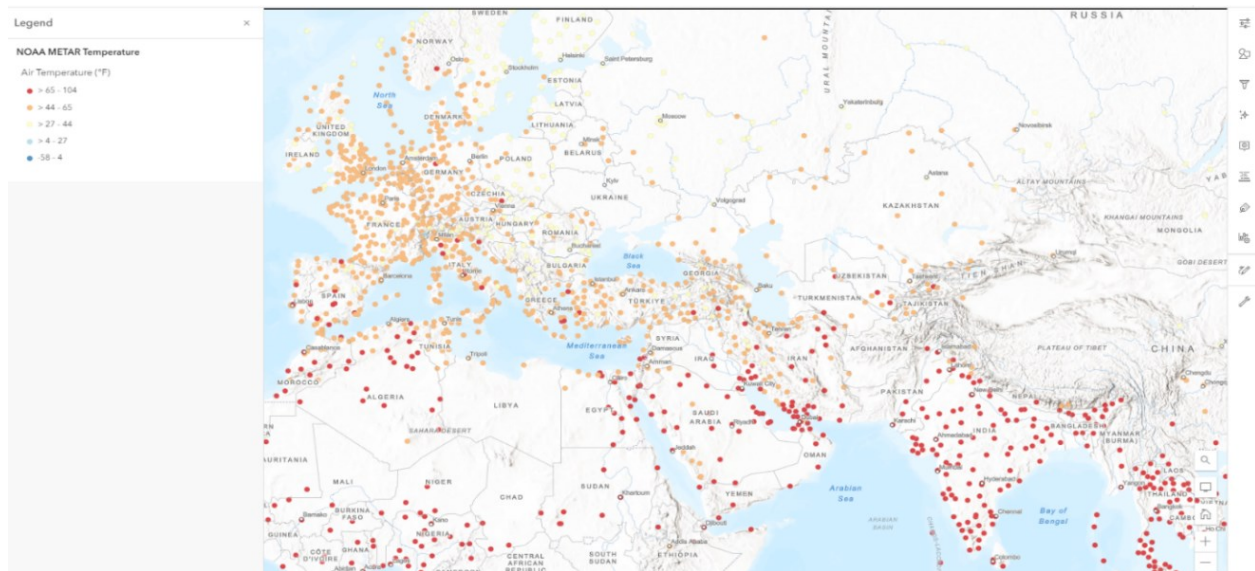


Figure 28. Live real data uploaded on arcGIS.

For our region, GCC region, especially in Bahrain, number of stations are very minimum but the variation of temperature over the geographical distance is very minimum too. In Figure 29, it shows that there are very few weather

stations comparing to the large distance. However, the number of stations is increasing significantly in the west side of the world as shown in Figure 30.

In Figure 31, the information of temperature and the time are provided and updated every three hours for all stations including Bahrain international airport station.



Figure 29. Weather stations in Gulf region.

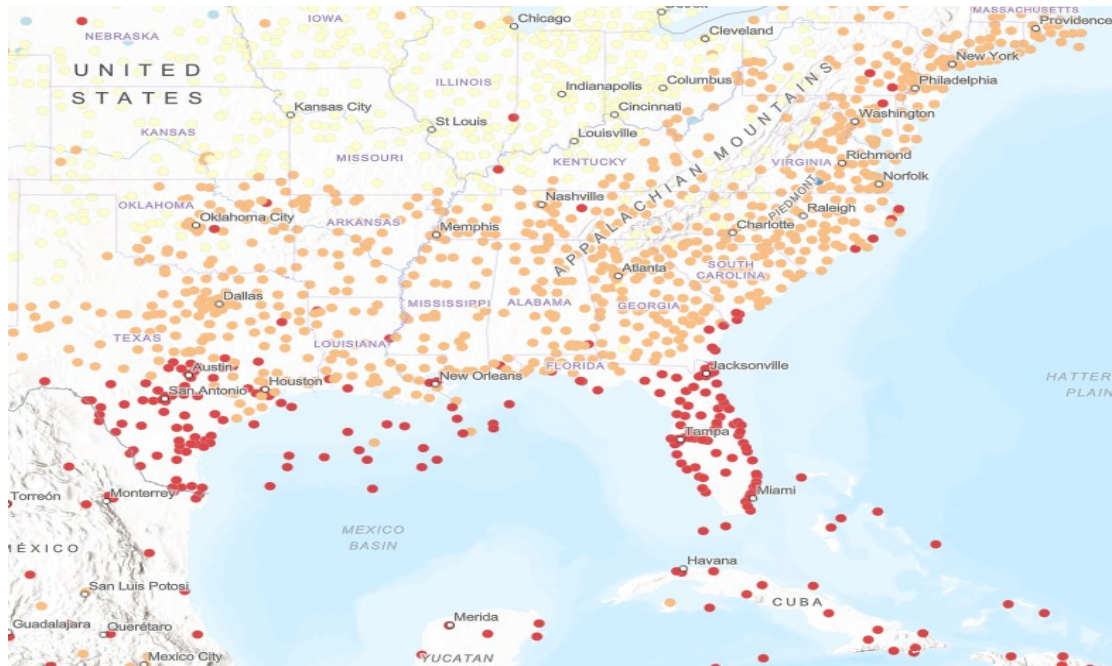


Figure 30. Weather stations in the west side of the world.

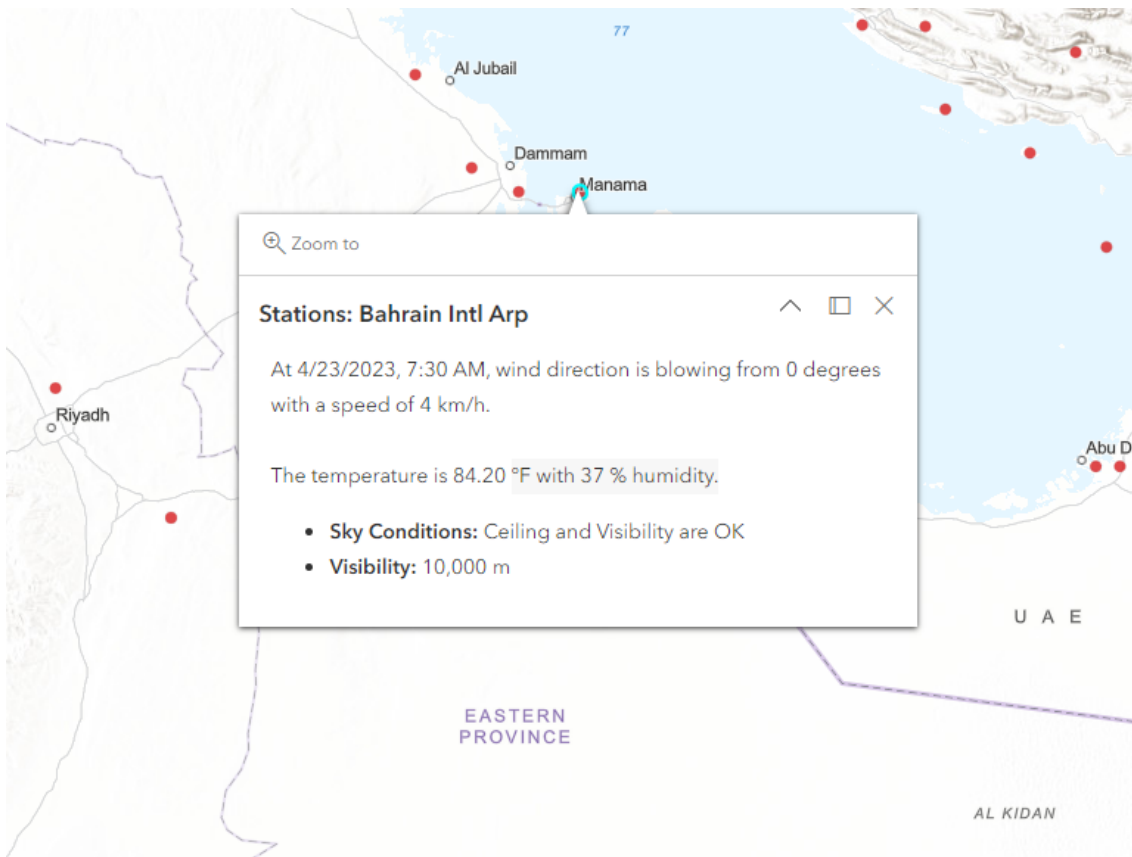


Figure 31. Information of temperature and time in Bahrain International Airport station.

6.3 Floor plan Heat Gain Calculation

After obtaining the surface area of each floorplan using a specialized algorithm, the next step in the workflow involves determining the maximum U-Value based on the regulations and standards set by Bahrain. The U-Value represents the rate of heat transfer through a particular building element, such as walls, windows, or doors. Adhering to the specified U-Value ensures that the building meets the required energy efficiency standards in Bahrain.

To calculate the heat gain factor for each floorplan, the algorithm requires additional data, specifically the ΔT (temperature difference) values. These values are obtained from ArcGIS live data, which provides accurate and real-time temperature information for the region. The ΔT represents the difference in temperature between the interior and exterior temperatures and is a crucial factor in calculating heat gain.

Once the algorithm has access to the surface area, maximum U-Value, and ΔT values, it performs the necessary calculations to determine the heat gain factor for each floorplan. This factor quantifies the amount of heat that is lost through the building envelope and helps assess the energy efficiency of the floorplan design.

The results obtained from the algorithm, including the heat gain factor for each of the 15 floorplans, are compiled and presented in Table 34. This table serves as a comprehensive summary of the heat gain characteristics for each floorplan, allowing stakeholders to compare and evaluate the energy performance of different designs.

By analyzing the results in Table 34, architects, engineers, or project managers can identify floorplans with higher heat gain factors, indicating areas that require improvement or optimization. This information can guide decisions related to insulation upgrades, window modifications, or other energy-saving measures, ultimately leading to more energy-efficient buildings and reduced heating costs.

The availability of accurate and up-to-date data from ArcGIS, combined with the algorithm's calculations and the regulatory framework of Bahrain, ensures that the heat gain factors obtained from this process are reliable and aligned with the regional context.

Table 34. Calculation of heat gain for the floor plans.

Floor plan No.	Surface Area for solid walls (m2)	Surface area for openings (m2)	Area of roof (m2)	U-Value for Solid wall (W/m2.C)	U-Value for openings (W/m2.C)	U-Value for roof (W/m2.C)	ΔT c	Heat gain factor
1	151.7	68.8	217	0.57	1.9	0.3	10	2822.89
2	254.3	70.4	215	0.57	1.9	0.3	10	3432.11
3	394.4	104.2	301	0.57	1.9	0.3	10	5130.88
4	196.5	78.7	201	0.57	1.9	0.3	10	3218.35
5	313.8	108.2	332	0.57	1.9	0.3	10	4840.46
6	285.3	94.4	307	0.57	1.9	0.3	10	4340.81
7	198.5	94.7	258	0.57	1.9	0.3	10	3704.75
8	331.8	112.7	327	0.57	1.9	0.3	10	5013.56
9	236.1	94.6	296	0.57	1.9	0.3	10	4031.17
10	202.4	63.8	234	0.57	1.9	0.3	10	3067.88
11	303.9	117.6	317	0.57	1.9	0.3	10	4917.63
12	262.9	83.1	262	0.57	1.9	0.3	10	3863.43

13	225.3	57.8	189	0.57	1.9	0.3	10	2949.41
14	250.8	80.3	197	0.57	1.9	0.3	10	3546.26
15	253.6	90.2	225	0.57	1.9	0.3	10	3834.32

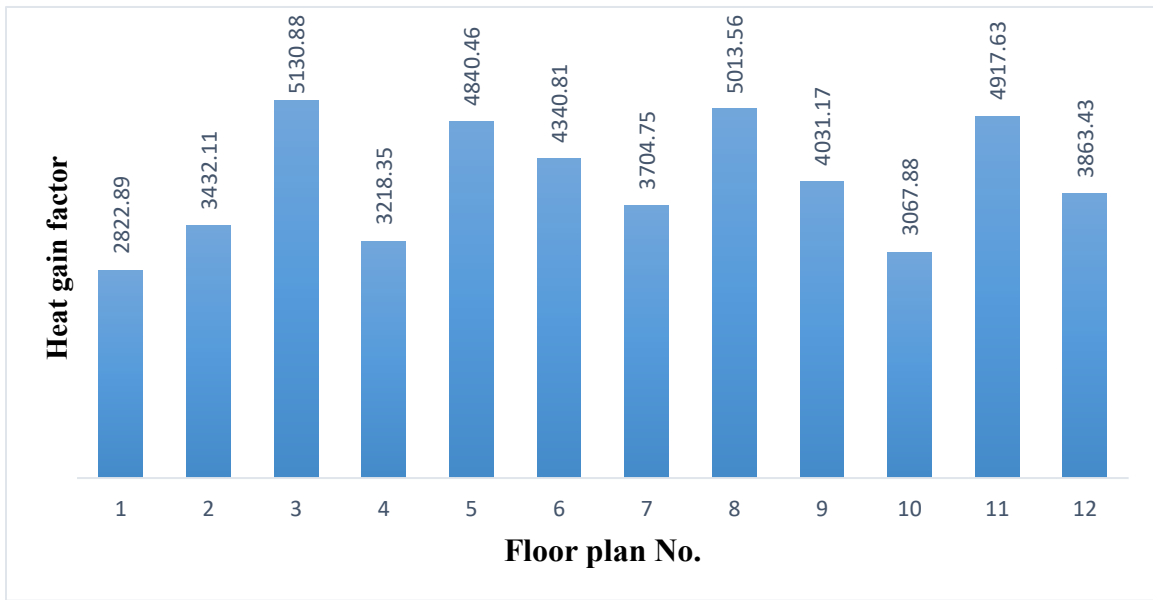


Figure 32. Heat Gain Factor for each Floorplan.

6.3.1 Overview Workflow

Figure 33 represents the comprehensive process workflow for analyzing heat gain factors in a floorplan. The workflow begins with the initial step of inserting the floorplan into the analysis algorithm.

Once the floorplan is inserted, the algorithm proceeds to analyze the various elements and characteristics of the floorplan that contribute to heat gain. This includes factors such as walls, windows, and doors, and overall area. Besides, the algorithm will determine whether this floorplan meets Bahrain regulation or not.

The software then performs complex calculations and simulations based on established heat transfer principles to determine the heat gain factor for the floorplan. This factor represents the amount of heat that is lost through the building envelope and indicates the overall efficiency of the floorplan in retaining heat.

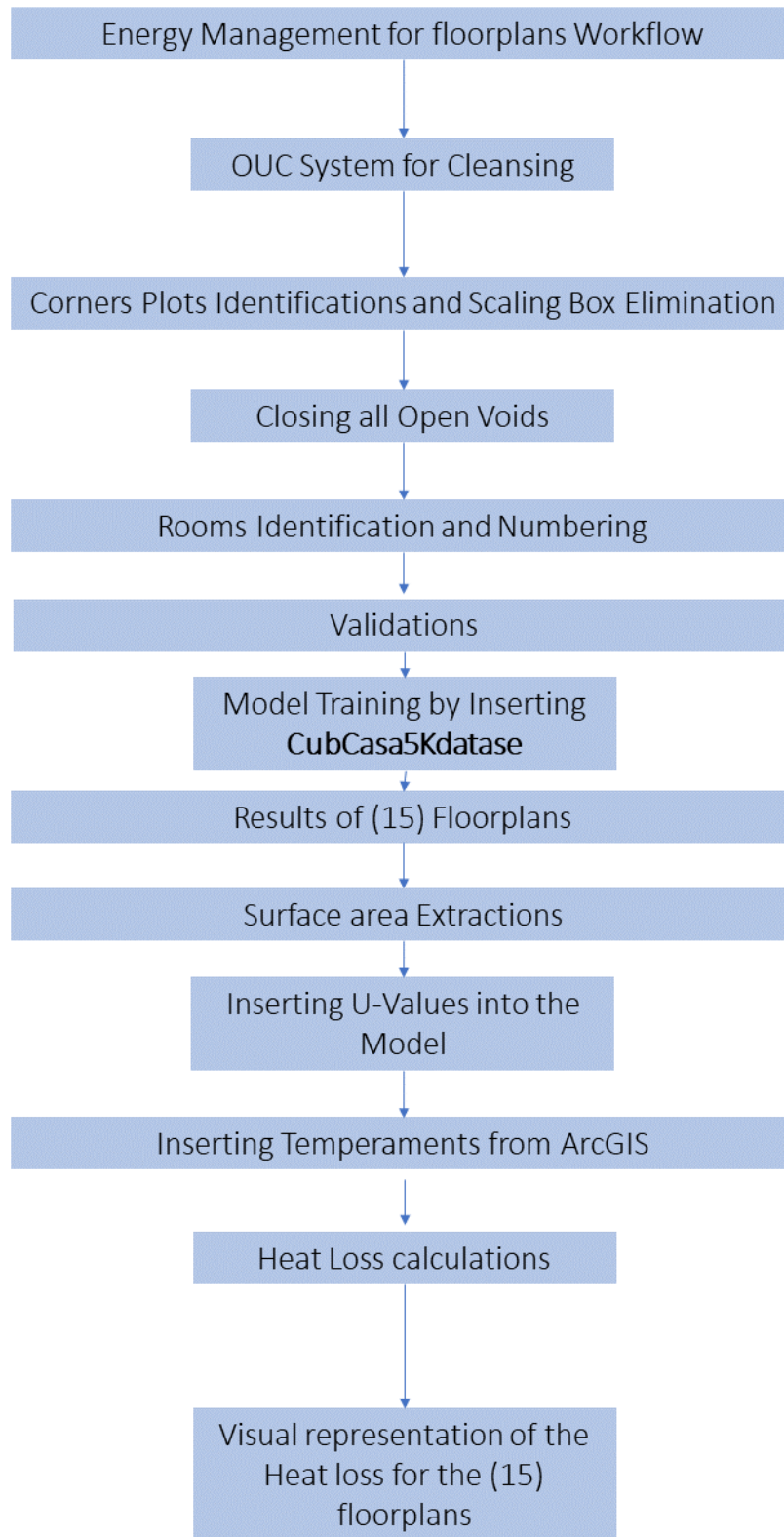


Figure 33. Heat Gain Factor for each Floorplan.

Chapter 7

Results and Discussion

Chapter 7 Results and Discussion

7.1 Result Discussion

To ensure the accuracy and reliability of the algorithm's results, a crucial step in the verification process involves comparing the dimensions of the floorplan obtained from the AutoCAD file with the dimensions generated by the algorithm. This step helps identify any discrepancies or errors in the algorithm's calculations.

By comparing the floorplan dimensions, including lengths, widths, and areas, between the AutoCAD file and the algorithm output, it becomes possible to assess the level of agreement between the two sets of data. Any variations or inconsistencies can be analyzed to determine the source of error and improve the algorithm's performance.

To quantify the accuracy of the algorithm, the percentage of error is calculated for each detail by comparing the algorithm's output values with the actual calculations performed by the engineering office. This comparison involves considering factors such as (surface area), (U-Values), and (temperature differentials).

Tables (35-49) provide a comprehensive breakdown of the result comparison between the algorithm's output and the actual calculations conducted by the engineering office. These tables present the relevant details, such as surface areas, U-Values, temperature differentials, and the corresponding values obtained from both the algorithm and the engineering office.

Additionally, (Figure 34-Figure 48) illustrates the percentage of error for each detail, showcasing the deviations between the algorithm's output and the engineering office's calculations. These figures offer a visual representation of the accuracy levels achieved by the algorithm across different aspects of the analysis.

By examining the percentage of error presented in these figures, it becomes possible to assess the algorithm's overall performance and identify specific areas where improvements may be needed. This analysis allows for adjustments and refinements to enhance the algorithm's accuracy and reliability, ensuring more precise heat gain factor calculations for future floorplans.

Table 35. Results compression for floor plan no. 1.

Detail	Algorithm output	Actual
% of built-up area	42.14%	42.11%
Front recession (m)	4	4m
Left side recession (m)	2	2m
Right side recession (m)	2	2m
Rear recession (m)	12.5	12.6
% of front attach	0%	0%
% of left attach	52.96%	52.82%
% of right attach	0%	0%
% of rear attach	0%	0%

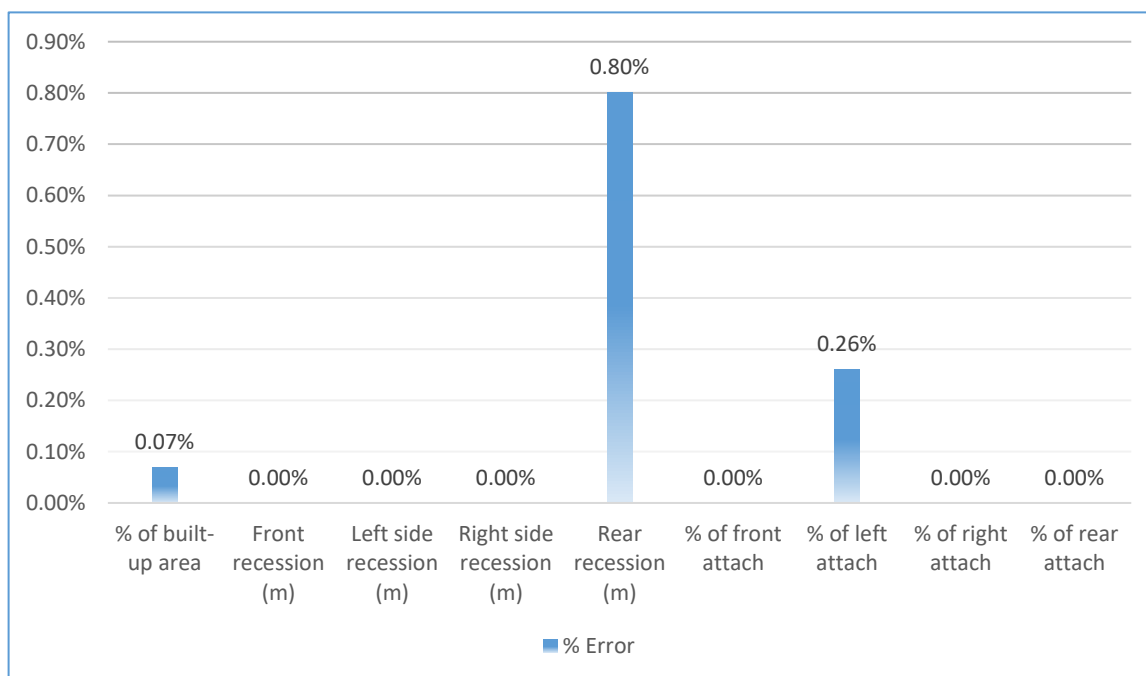


Figure 34. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 1.

Table 36. Results for floor plan no. 2

Detail	Algorithm output	Actual
% of built-up area	76.05%	76.12%
Front recession (m)	3	3.02
Left side recession (m)	2	2
Right side recession (m)	2	1.99
Rear recession (m)	2	1.97
% of front attach	0%	0%
% of left attach	32.41%	32.40%
% of right attach	0%	0%
% of rear attach	0%	0%

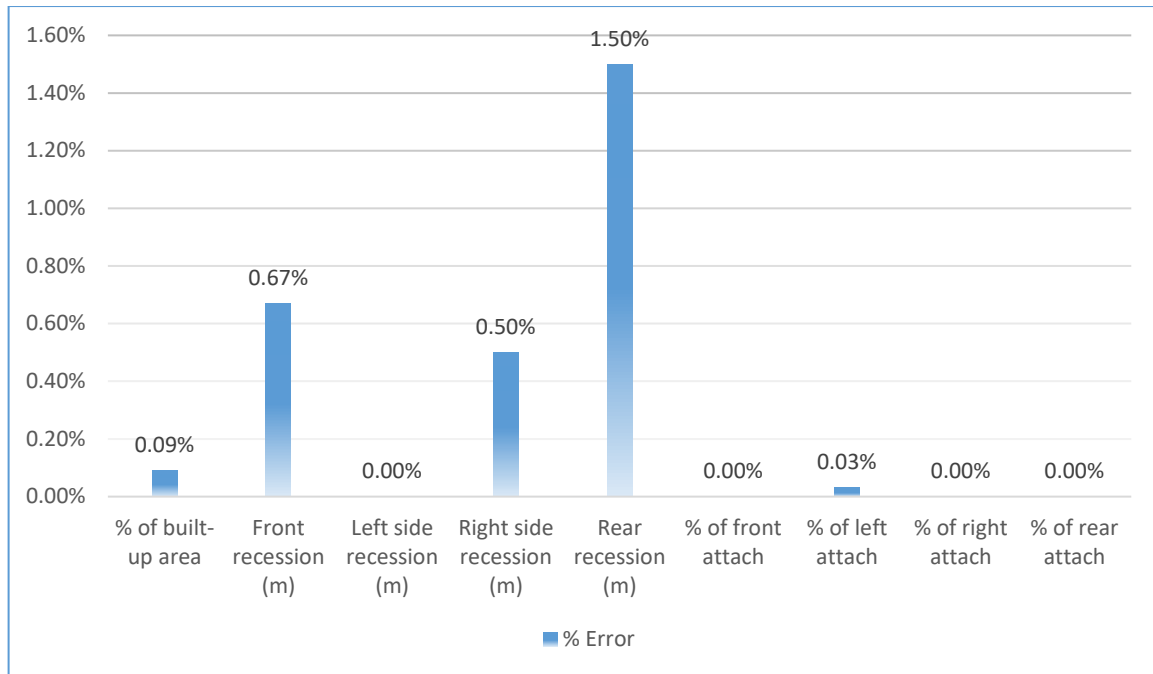


Figure 35. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 2.

Table 37. Results for floor plan no. 3.

Detail	Algorithm output	Actual
% of built-up area	66.86%	66.84%
Front recession (m)	3	3
Left side recession (m)	2	2
Right side recession (m)	2	2
Rear recession (m)	2	2
% of front attach	0%	0%
% of left attach	41.95%	41.98%
% of right attach	0%	0%
% of rear attach	0%	0%

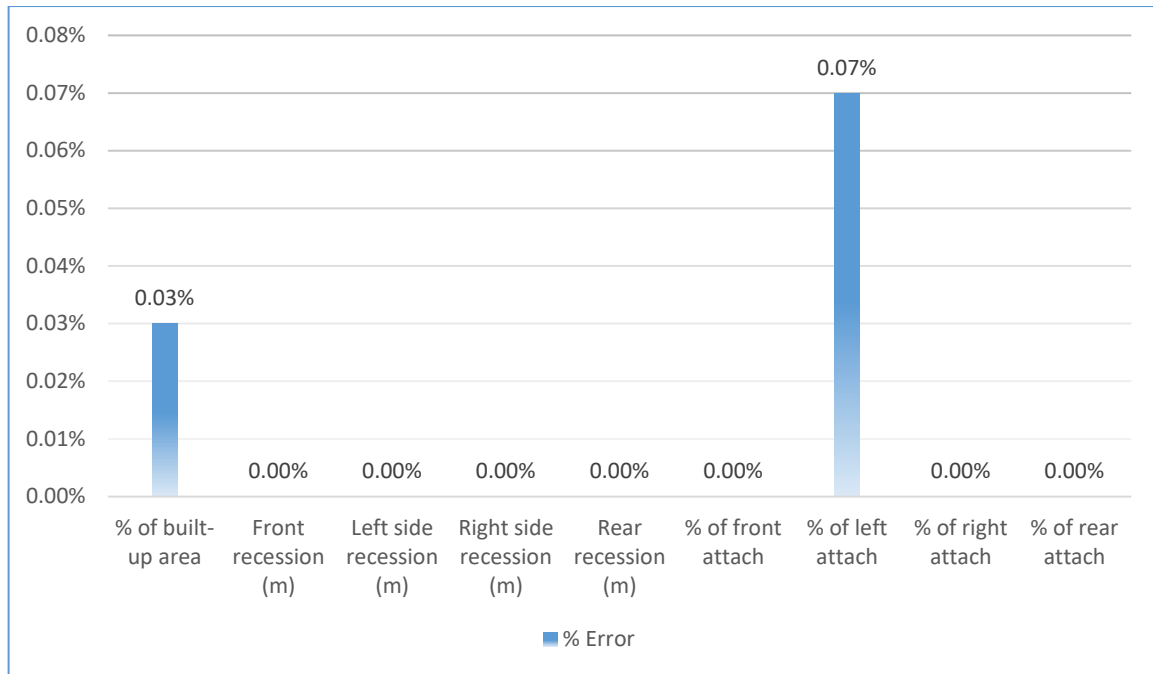


Figure 36. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 3.

Table 38. Results for floor plan no. 4.

Detail	Algorithm output	Actual
% of built-up area	81.52%	81.32%
Front recession (m)	3	2.95
Left side recession (m)	2	2.07
Right side recession (m)	2	1.98
Rear recession (m)	2	2.06
% of front attach	0%	0%
% of left attach	0.00%	0%
% of right attach	51.62%	51.58%
% of rear attach	22.04%	22.08%

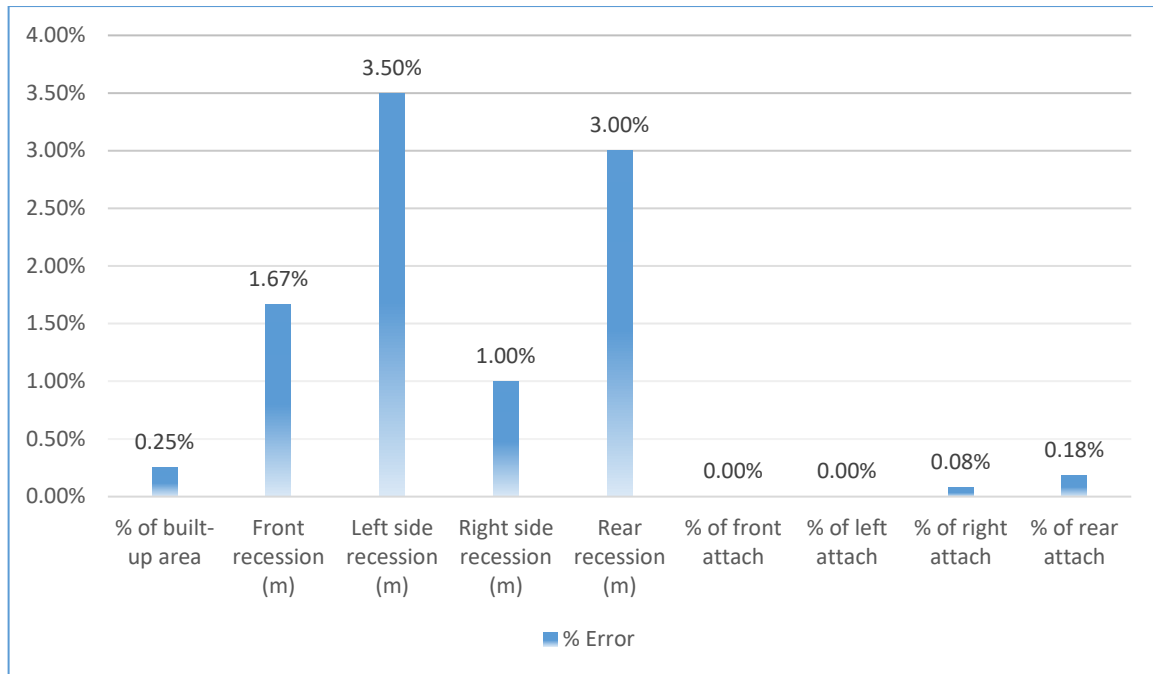


Figure 37. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 4.

Table 39. Results for floor plan no. 5.

Detail	Algorithm output	Actual
% of built-up area	66.41%	66.42%
Front recession (m)	3	3
Left side recession (m)	2	2.01
Right side recession (m)	2	2
Rear recession (m)	2	2
% of front attach	0%	0%
% of left attach	0.00%	0%
% of right attach	34.44%	34.42%
% of rear attach	0%	0%

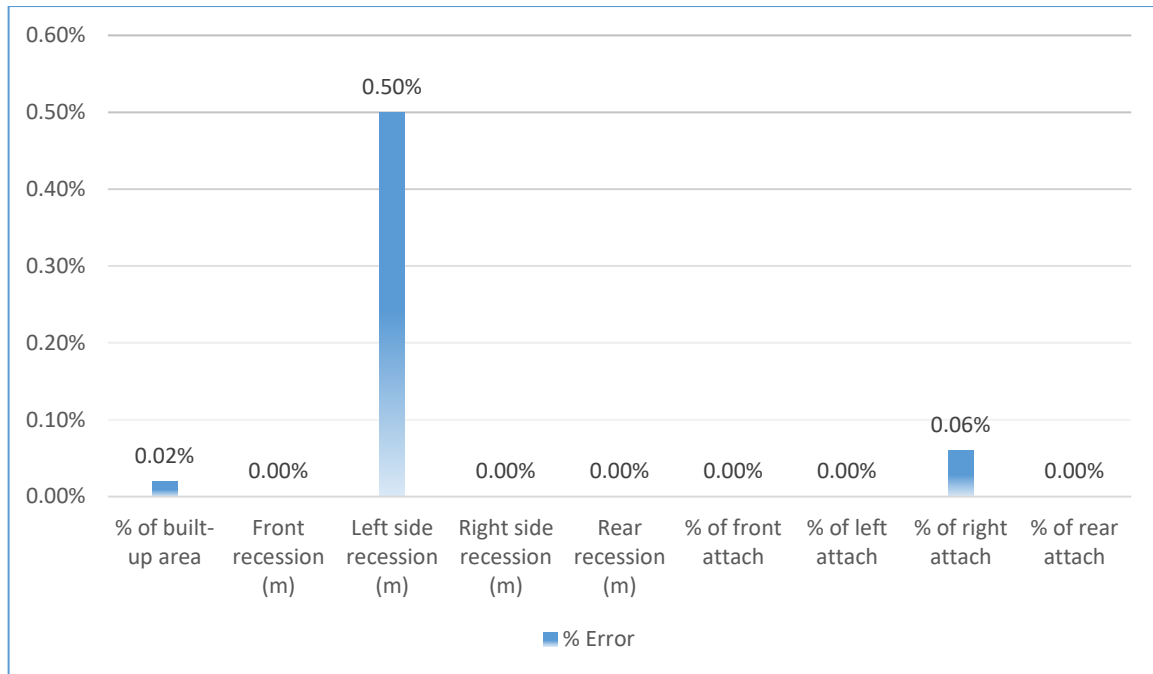


Figure 38. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 5.

Table 40. Results for floor plan no. 6.

Detail	Algorithm output	Actual
% of built-up area	78.58%	78.57%
Front recession (m)	3	3
Left side recession (m)	2	2
Right side recession (m)	2	2
Rear recession (m)	2	2
% of front attach	0%	0%
% of left attach	0.00%	0%
% of right attach	0%	0%
% of rear attach	25.20%	25.23%

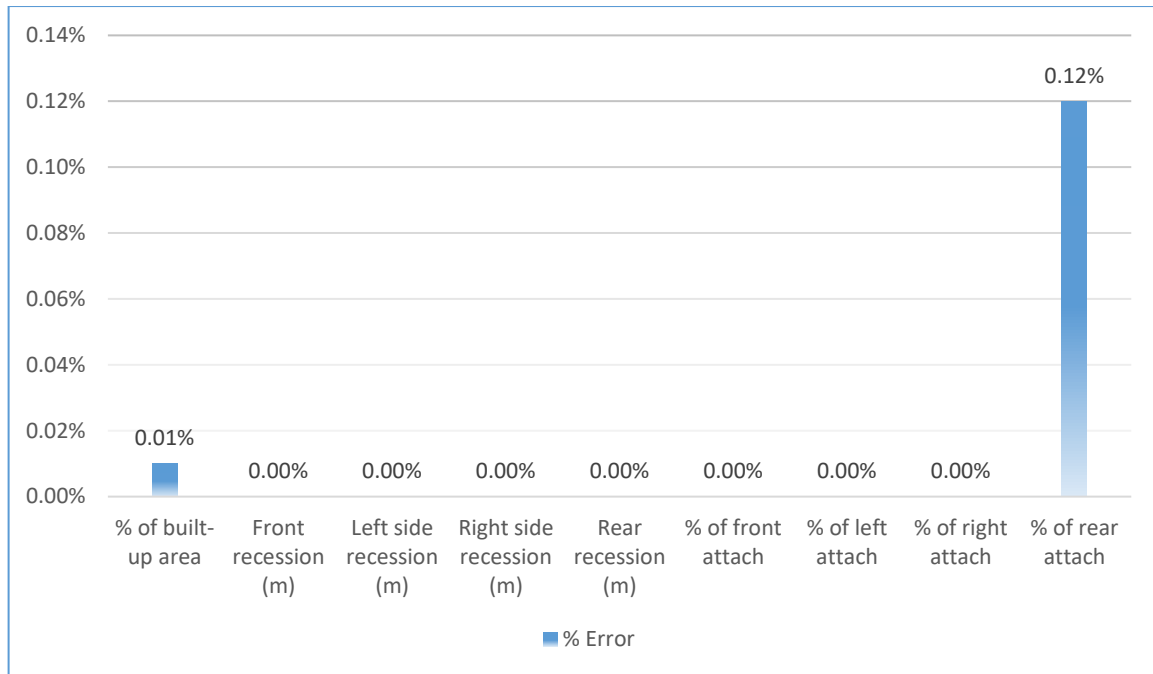


Figure 39. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 6.

Table 41. Results for floor plan no. 7.

Detail	Algorithm output	Actual
% of built-up area	39.01%	38.99%
Front recession (m)	6	6
Left side recession (m)	2	2
Right side recession (m)	2	2
Rear recession (m)	2	1.98
% of front attach	0%	0%
% of left attach	39.15%	39.20%
% of right attach	0%	0%
% of rear attach	0%	0%

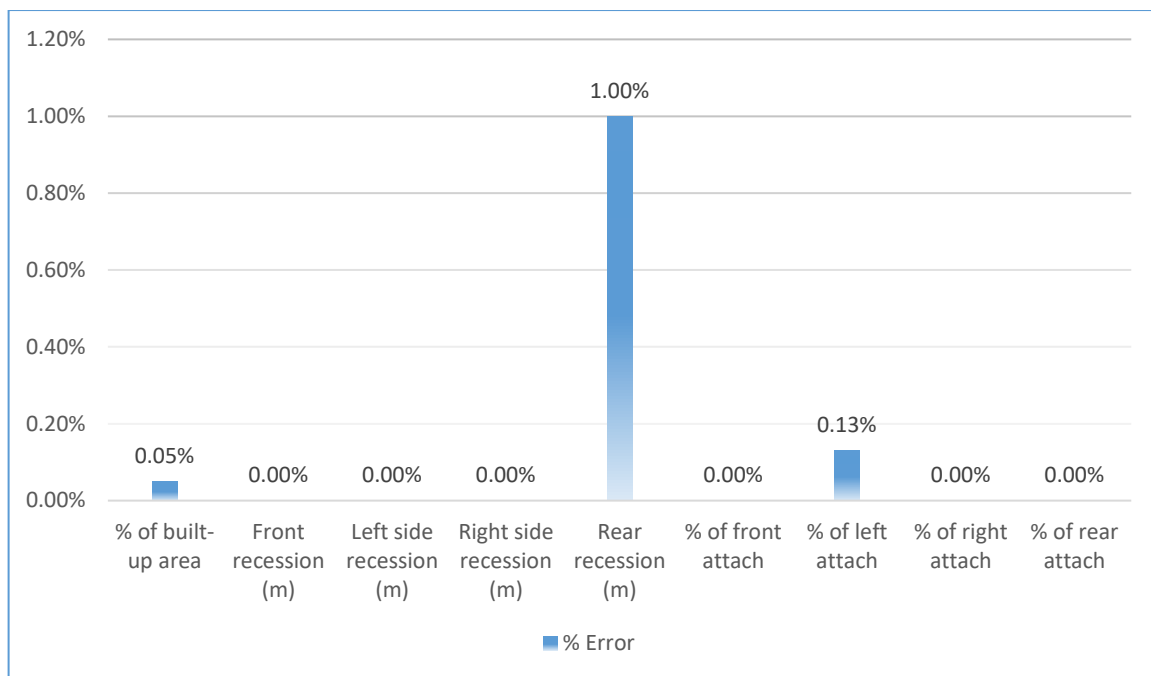


Figure 40. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 7.

Table 42. Results for floor plan no. 8.

Detail	Algorithm output	Actual
% of built-up area	34.50%	34.45%
Front recession (m)	4.67	4.65
Left side recession (m)	2	2
Right side recession (m)	3.2	3.19
Rear recession (m)	5	5
% of front attach	0%	0%
% of left attach	0.00%	0%
% of right attach	12.95%	12.93%
% of rear attach	0%	0%

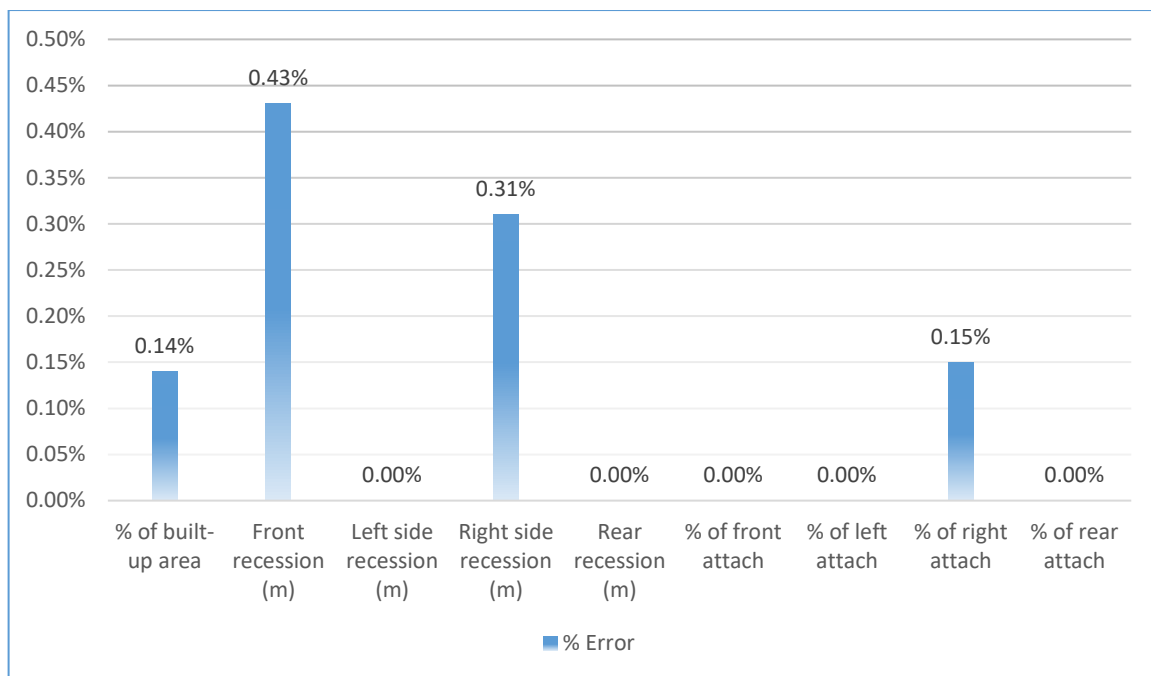


Figure 41. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 8.

Table 43. Results for floor plan no. 9.

Detail	Algorithm output	Actual
% of built-up area	46.87%	46.79%
Front recession (m)	5.21	5.19
Left side recession (m)	2	2
Right side recession (m)	2	2
Rear recession (m)	0	0
% of front attach	0%	0%
% of left attach	0.00%	0%
% of right attach	0%	0%
% of rear attach	43.40%	43.44%

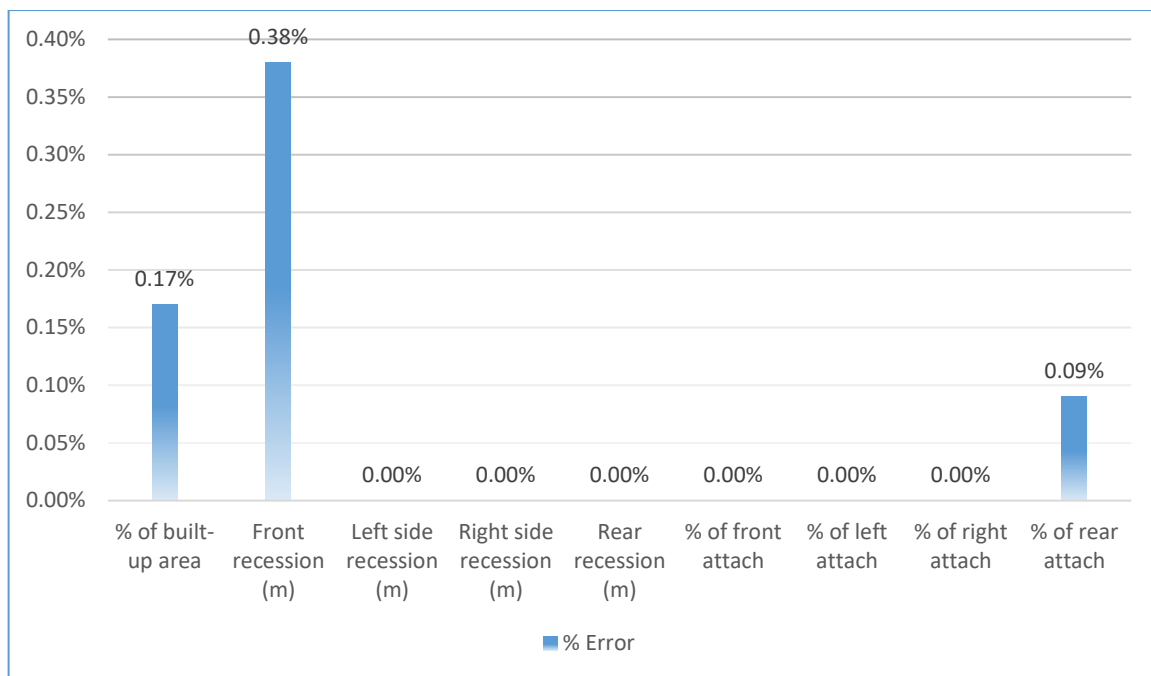


Figure 42. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 9.

Table 44. Results for floor plan no. 10.

Detail	Algorithm output	Actual
% of built-up area	70.01%	70.06%
Front recession (m)	3	3
Left side recession (m)	2	1.97
Right side recession (m)	1.9	1.89
Rear recession (m)	1.8	1.84
% of front attach	0%	0%
% of left attach	0.00%	0%
% of right attach	17.57%	17.69%
% of rear attach	39.35%	39.28%

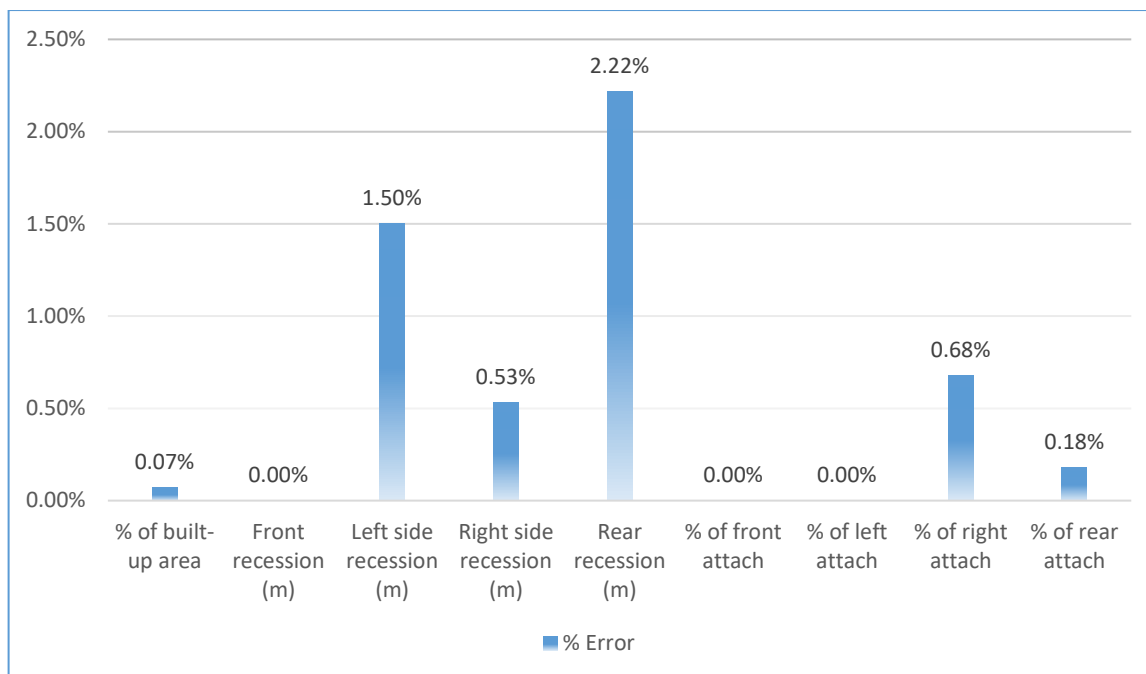


Figure 43. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 10.

Table 45. Results for floor plan no. 11.

Detail	Algorithm output	Actual
% of built-up area	52.55%	52.38%
Front recession (m)	4.32	4.21
Left side recession (m)	2	2.14
Right side recession (m)	4.2	4.17
Rear recession (m)	6.4	6.44
% of front attach	0%	0%
% of left attach	19.91%	19.84%
% of right attach	0%	0%
% of rear attach	0%	0%

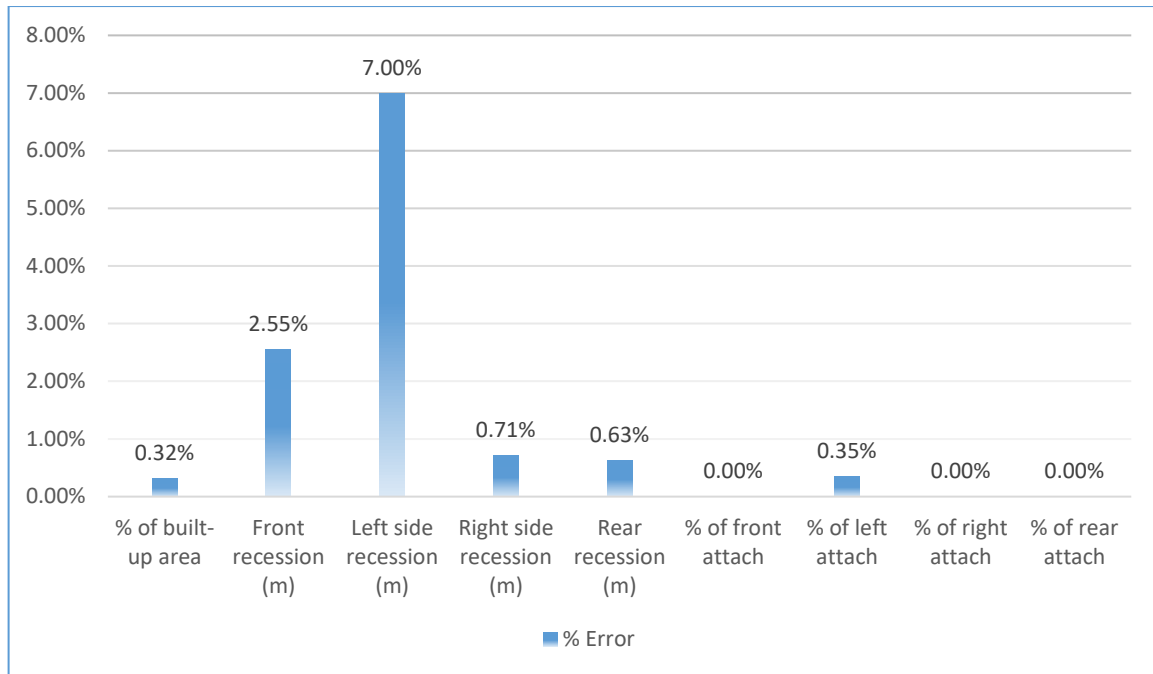


Figure 44. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 11.

Table 46. Results for floor plan no. 12

Detail	Algorithm output	Actual
% of built-up area	32.15%	32.14%
Front recession (m)	3.11	3.08
Left side recession (m)	2.3	2.21
Right side recession (m)	2.2	2.2
Rear recession (m)	2	2
% of front attach	0%	0%
% of left attach	0.00%	0%
% of right attach	32.15%	32.14%
% of rear attach	0%	0%

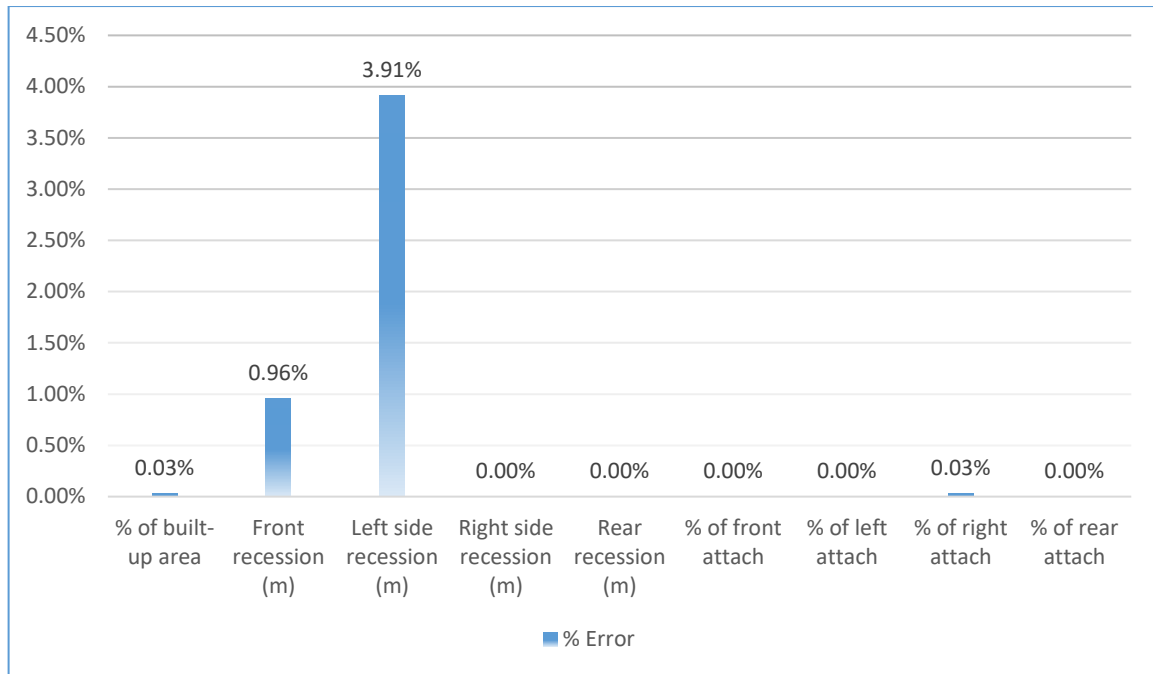


Figure 45. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 12.

Table 47. Results for floor plan no. 13

Detail	Algorithm output	Actual
% of built-up area	58.61%	58.66%
Front recession (m)	3	3
Left side recession (m)	2.05	2.03
Right side recession (m)	2.1	2.1
Rear recession (m)	1.98	2
% of front attach	0%	0%
% of left attach	47.21%	47.22%
% of right attach	0.00%	0%
% of rear attach	9.02%	8.97%

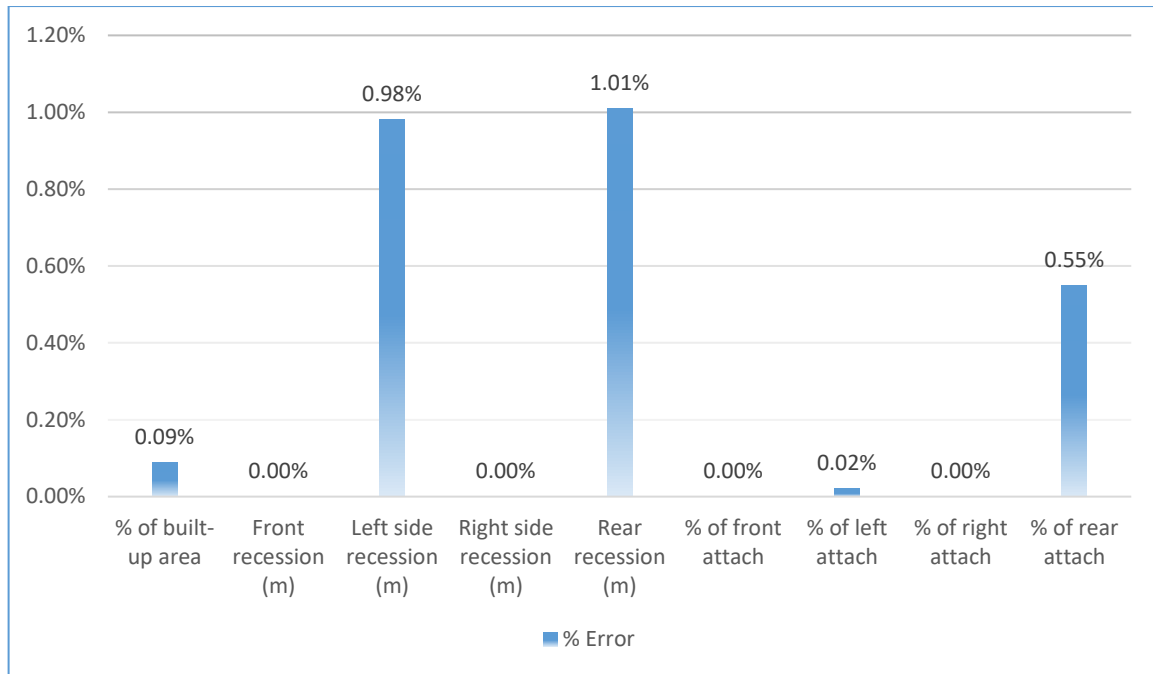


Figure 46. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 13.

Table 48. Results for floor plan no. 14.

Detail	Algorithm output	Actual
% of built-up area	68.46%	68.41%
Front recession (m)	3	3
Left side recession (m)	2.02	2
Right side recession (m)	2.01	2
Rear recession (m)	2.01	2
% of front attach	0%	0%
% of left attach	0.00%	0%
% of right attach	29.25%	29.24%
% of rear attach	0%	0%

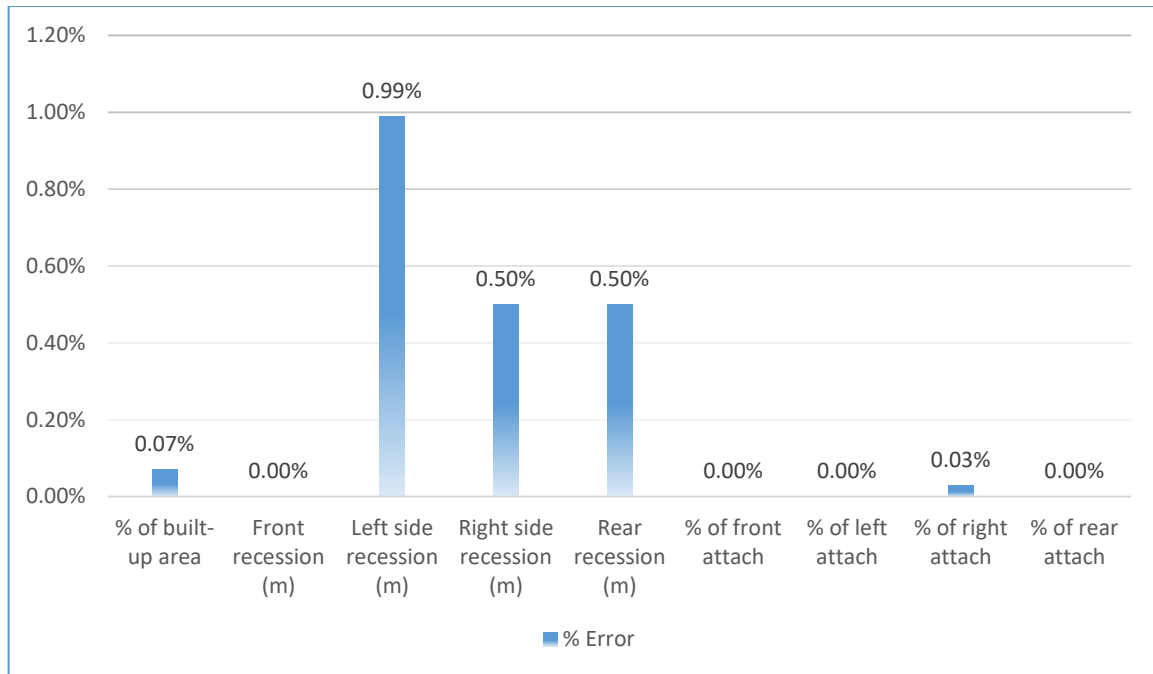


Figure 47. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 14.

Table 49. Results for floor plan no. 15.

Detail	Algorithm output	Actual
% of built-up area	58.35%	58.33%
Front recession (m)	4.74	4.8
Left side recession (m)	2.95	2.99
Right side recession (m)	2.01	2
Rear recession (m)	4.52	4.54
% of front attach	0%	0%
% of left attach	26.47%	26.48%
% of right attach	0%	0%
% of rear attach	12.54%	12.53%

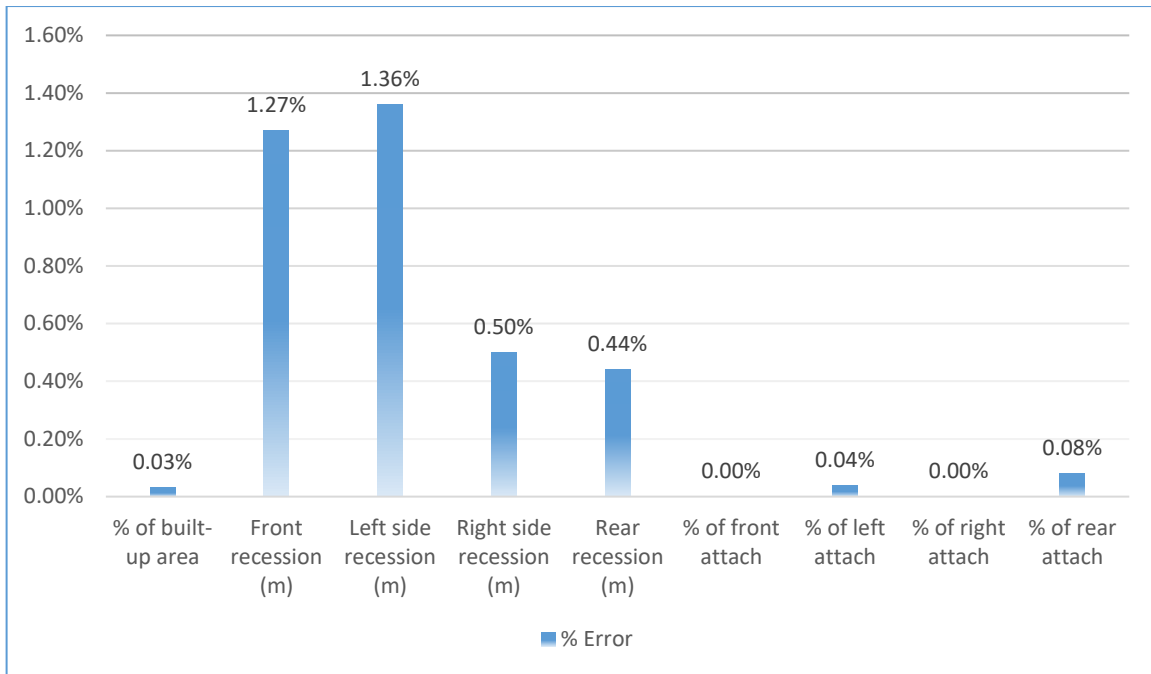


Figure 48. Percentages of error by comparing the results of actual output and algorithm output for floorplan No. 15.

Table 50 serves as a valuable summary of the average percentage error for each floorplan, providing insights into the overall accuracy of the algorithm's calculations. By examining the average percentage error, we can evaluate the algorithm's performance across multiple floorplans and assess its overall reliability.

Upon analyzing Table 50, it becomes evident that for floorplan no. 4 and floorplan no. 11, the error percentage exceeded 1%. This indicates that the algorithm's calculations for these specific floorplans deviated slightly from the actual calculations performed by the engineering office.

While the error percentages for these two floorplans are higher compared to others, it is important to note that the overall impact of these errors on the decision-making process remains minimal. The error percentages fall within

allowable tolerance levels, meaning they do not significantly affect the conclusions or recommendations drawn from the analysis.

The presence of minimal error percentages in the remaining floorplans indicates that the algorithm generally performs well in accurately calculating the heat gain factors. This reinforces the confidence in the algorithm's reliability and its ability to provide meaningful insights for energy efficiency improvements.

It is worth noting that the algorithm's performance can be further improved by identifying the specific reasons behind the higher error percentages for floorplan no. 4 and floorplan no. 11. By conducting a detailed analysis of the calculations and comparing them with the engineering office's calculations, it becomes possible to pinpoint areas where adjustments or refinements may be necessary.

However, given that the overall error percentages remain minimal and fall within acceptable tolerance levels, they do not significantly impact the decision-making process. The algorithm's results can still be considered reliable and trustworthy for informing energy efficiency decisions, building design optimizations, and other related considerations.

Table 50. Average percentage of error for each floorplan

Floorplan No.	Average percentage of error
1	0.13%
2	0.31%
3	0.01%
4	1.08%
5	0.06%
6	0.01%

7	0.13%
8	0.11%
9	0.07%
10	0.58%
11	1.28%
12	0.55%
13	0.29%
14	0.23%
15	0.41%

7.2 Outer Surface Area Extraction Error

To verify if the surface area extraction by the algorithm matches with the correct dimensions, a comparison between the actual measurements from the AutoCAD file and the algorithm output with % of error are showing in Table 51 to Table 65.

Accordingly, the percentage of error for each floorplan is presented in Figure 49 to Figure 63.

Table 51. Surface areas of floor plan no. 1.

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	220.5	222
Area of external wall openings	68.8	69.6
Area of external wall blocks	151.7	152.4
Area of the Roof	217	218

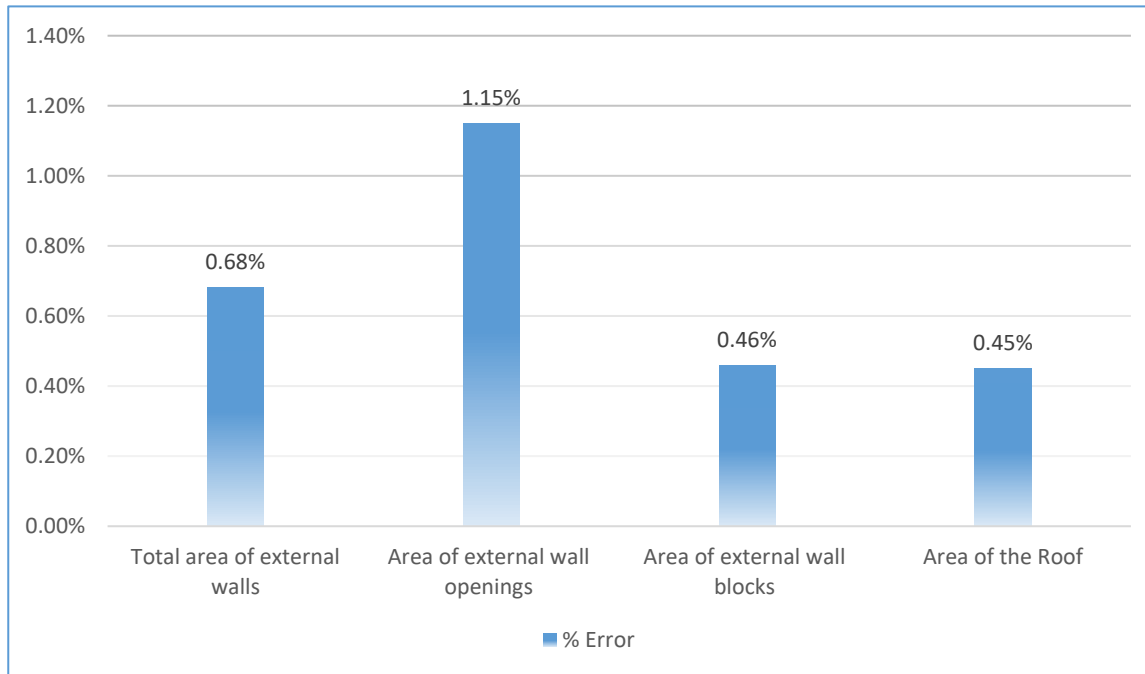


Figure 49. Percentage of error in the surface area for floorplan No. 1.

Table 52. Surface areas of floor plan no. 2

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	324.7	322.4
Area of external wall openings	70.4	69.2
Area of external wall blocks	254.3	253.2
Area of the Roof	215	219

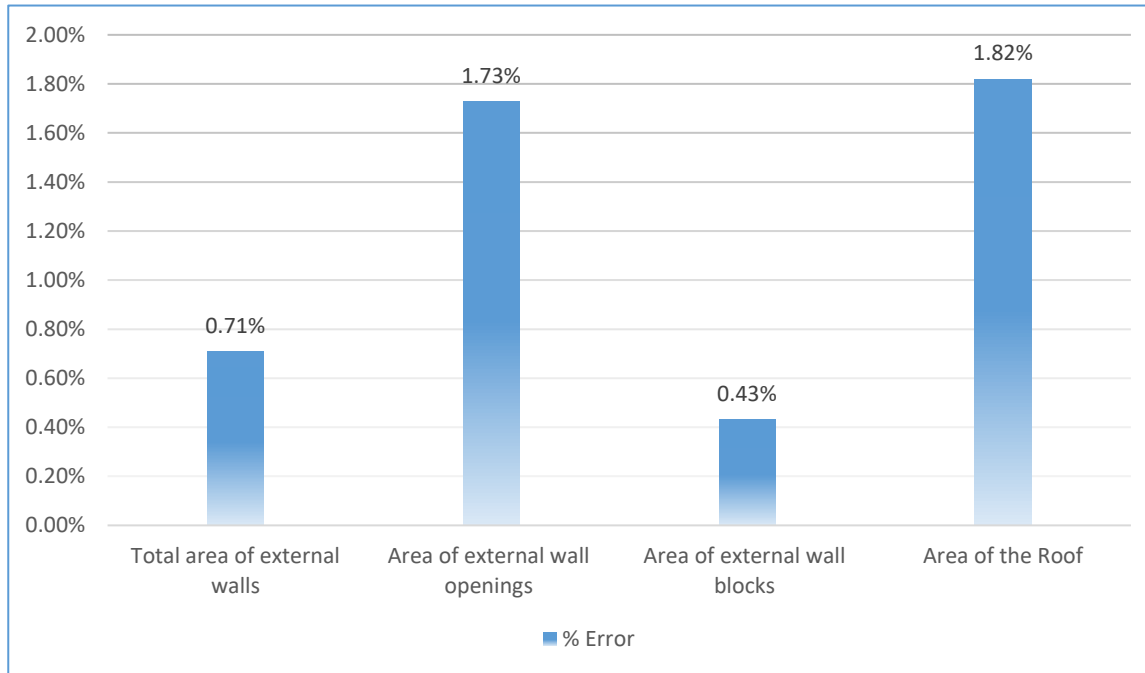


Figure 50. Percentage of error in the surface area for floorplan No. 2.

Table 53. Surface areas of floor plan no. 3

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	498.6	496.9
Area of external wall openings	104.2	102.9
Area of external wall blocks	394.4	394
Area of the Roof	301	306

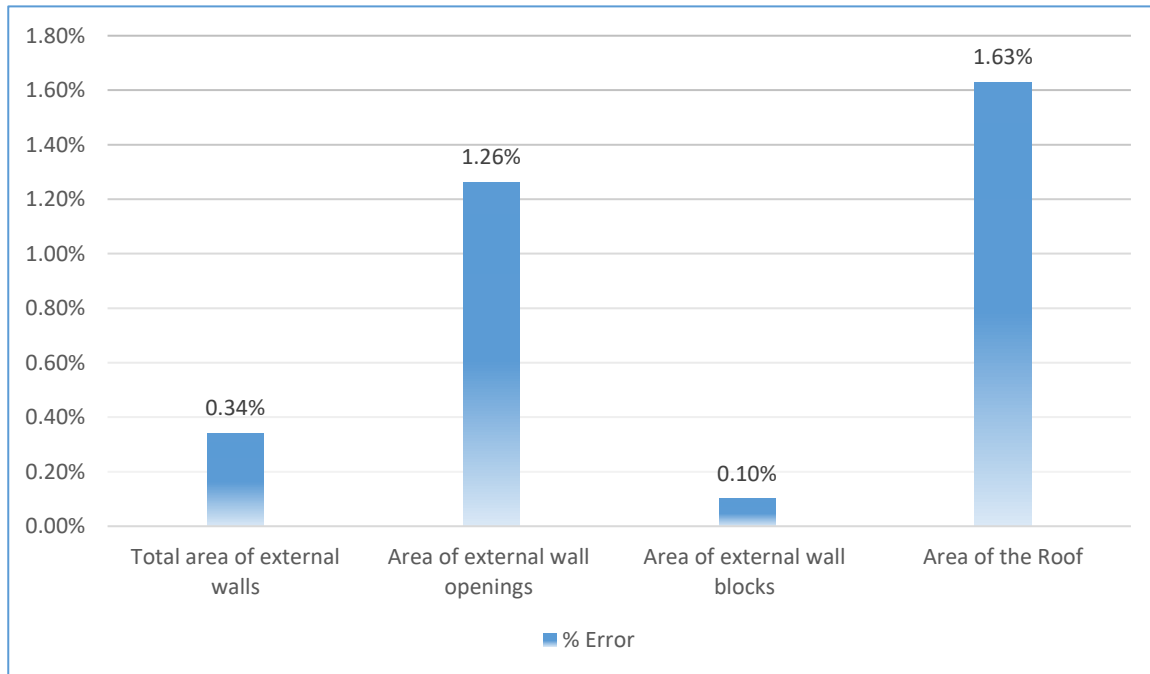


Figure 51. Percentage of error in the surface area for floorplan No.3.

Table 54. Surface areas of floor plan no. 4

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	275.2	275.4
Area of external wall openings	78.7	79.3
Area of external wall blocks	196.5	196.1
Area of the Roof	201	206

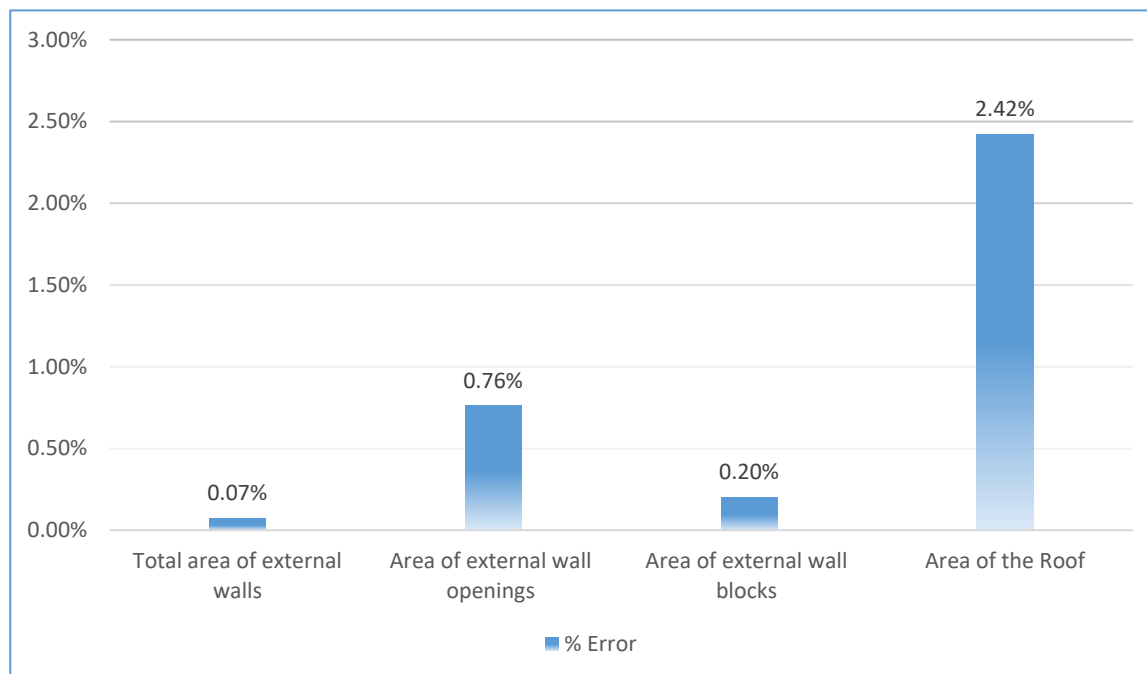


Figure 52. Percentage of error in the surface area for floorplan No.4.

Table 55. Surface areas of floor plan no. 5

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	422	421.2
Area of external wall openings	108.2	106.9
Area of external wall blocks	313.8	314.3
Area of the Roof	332	338

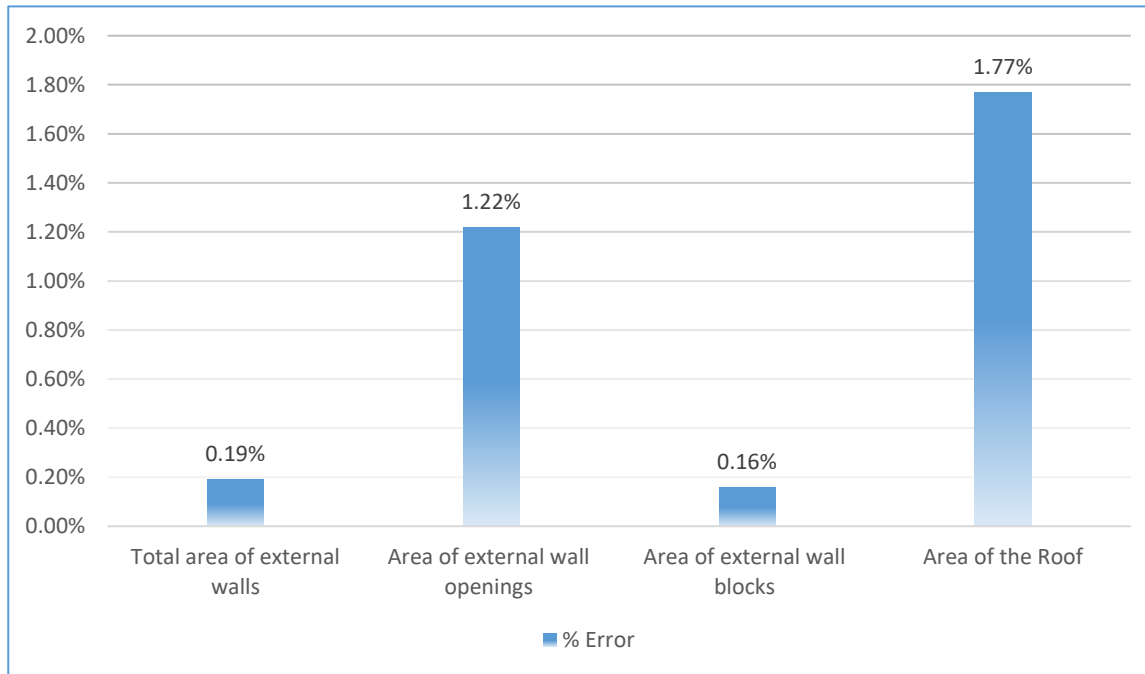


Figure 53. Percentage of error in the surface area for floorplan No.5.

Table 56. Surface are of floor plan no. 6.

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	379.7	383.4
Area of external wall openings	94.4	95.5
Area of external wall blocks	285.3	287.9
Area of the Roof	307	302

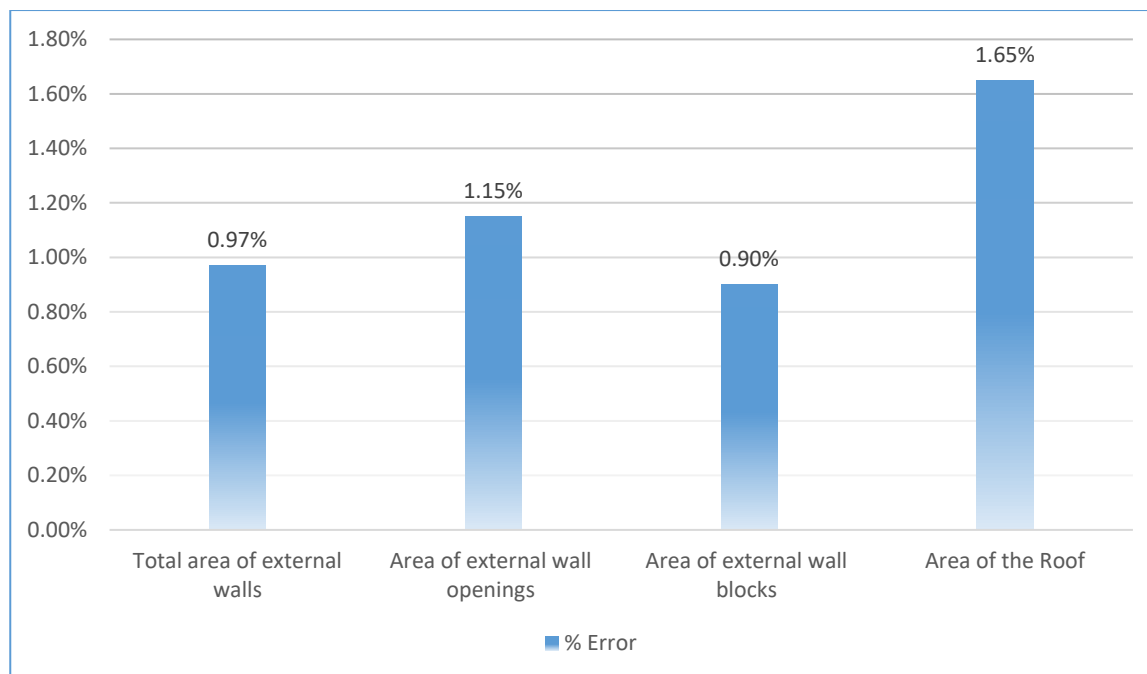


Figure 54. Percentage of error in the surface area for floorplan No.6.

Table 57. Surface areas of floor plan no. 7.

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	293.2	293.1
Area of externa wall openings	94.7	93.5
Area of external wall blocks	198.5	199.6
Area of the Roof	258	250

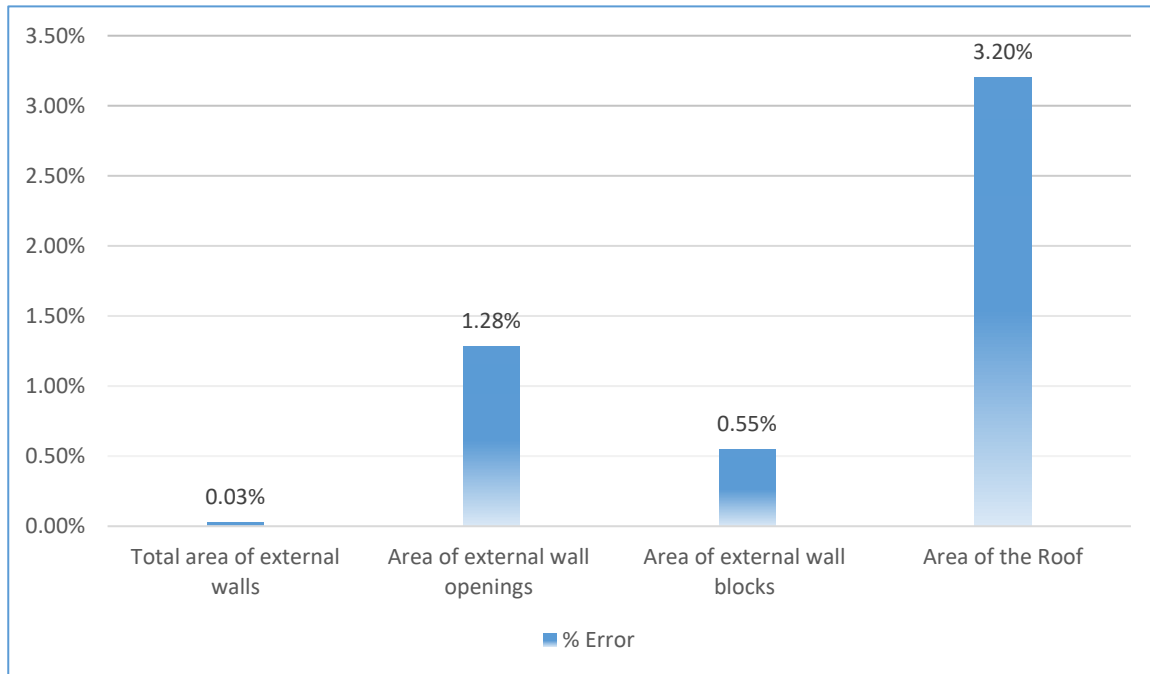


Figure 55. Percentage of error in the surface area for floorplan No.7.

Table 58. Surface areas of floor plan no. 8.

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	444.5	443.3
Area of external wall openings	112.7	110.8
Area of external wall blocks	331.8	332.5
Area of the Roof	327	328

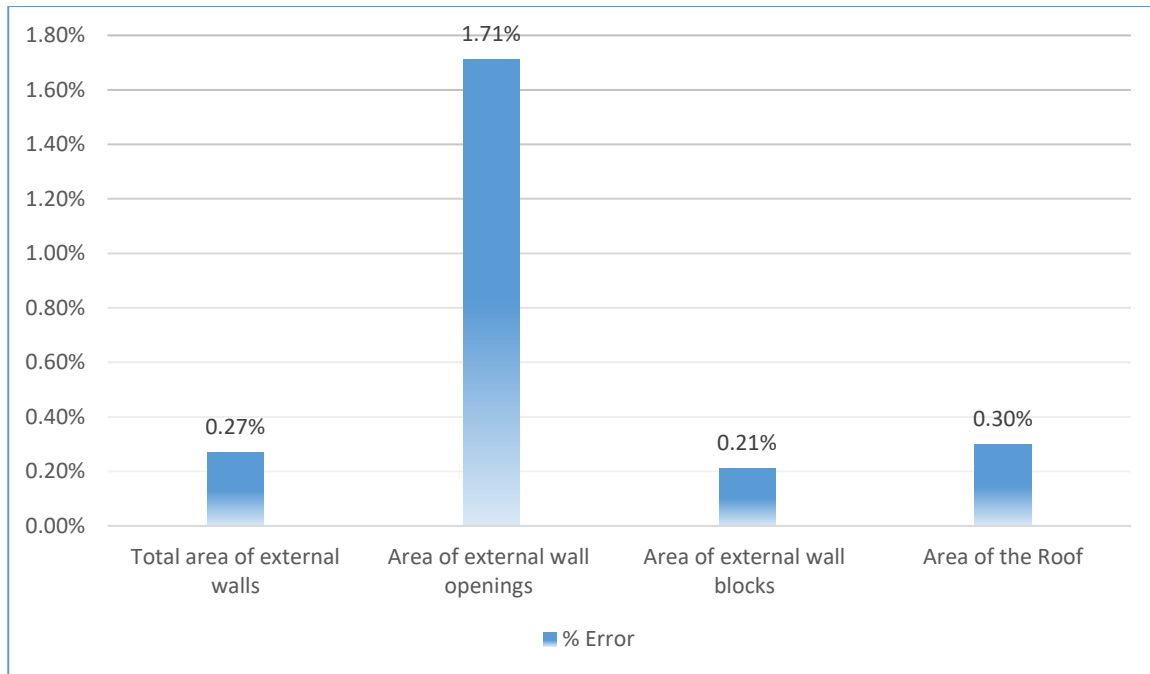


Figure 56. Percentage of error in the surface area for floorplan No.8.

Table 59. Surface areas of floor plan no. 9.

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	330.7	331.6
Area of external wall openings	94.6	92.9
Area of external wall blocks	236.1	238.7
Area of the Roof	296	299

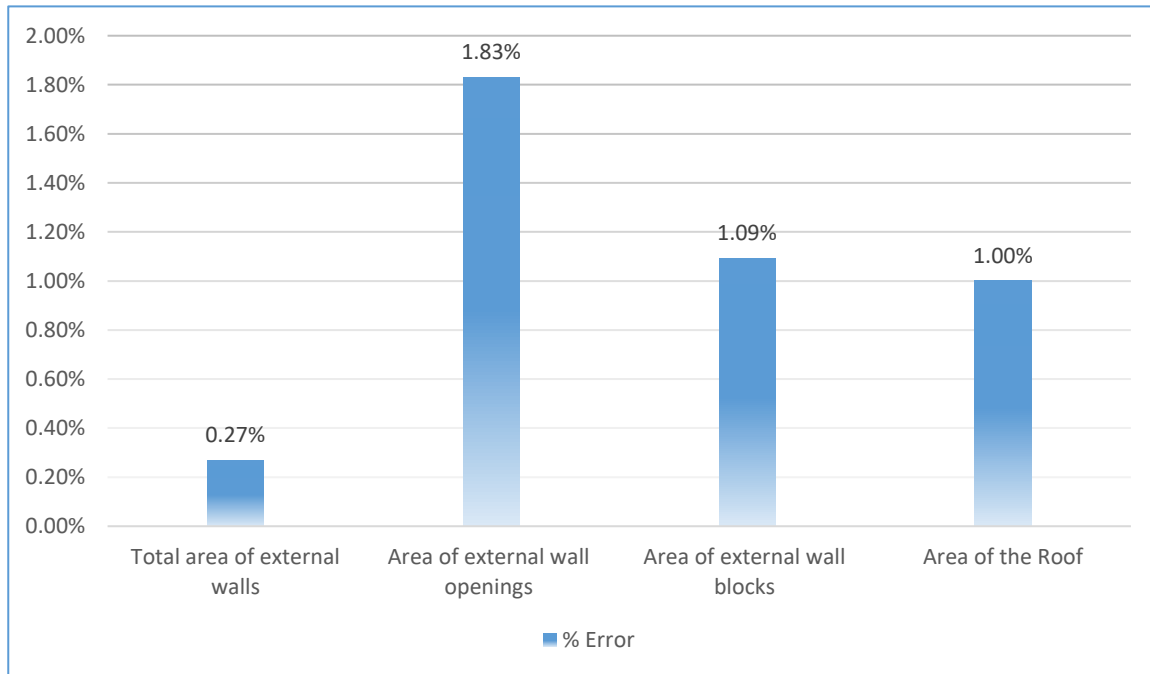


Figure 57. Percentage of error in the surface area for floorplan No.9.

Table 60. Surface areas of floor plan no. 10.

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	266.2	270.6
Area of external wall openings	63.8	67.3
Area of external wall blocks	202.4	203.3
Area of the Roof	234	231

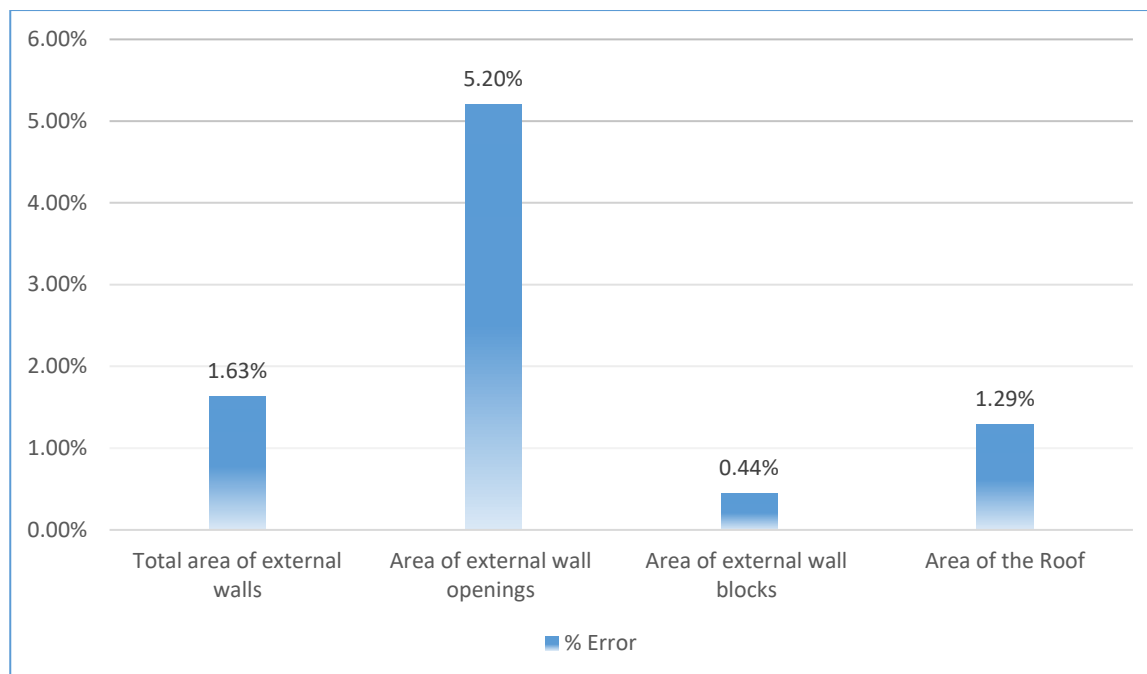


Figure 58. Percentage of error in the surface area for floorplan No.10.

Table 61. Surface areas of floor plan no. 11.

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	421.5	418.2
Area of external wall openings	117.6	115.7
Area of external wall blocks	303.9	302.5
Area of the Roof	317	321

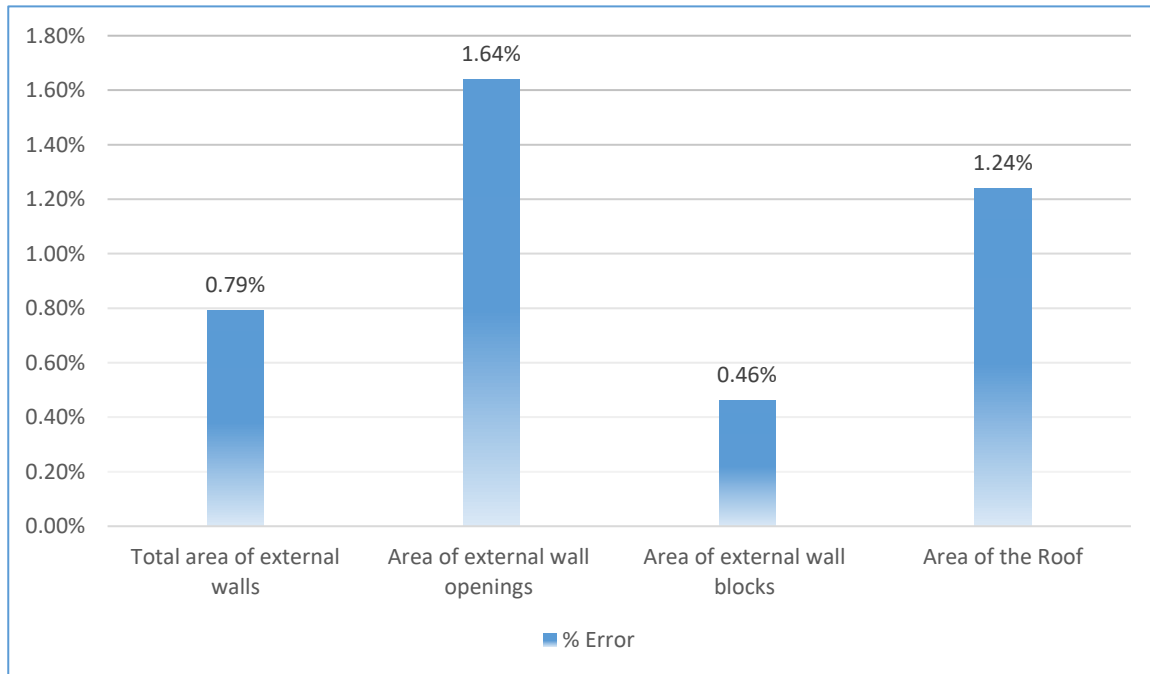


Figure 59. Percentage of error in the surface area for floorplan No.11.

Table 62. Surface areas of floor plan no. 12.

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	346	346
Area of external wall openings	83.1	82.3
Area of external wall blocks	262.9	263.7
Area of the Roof	262	258

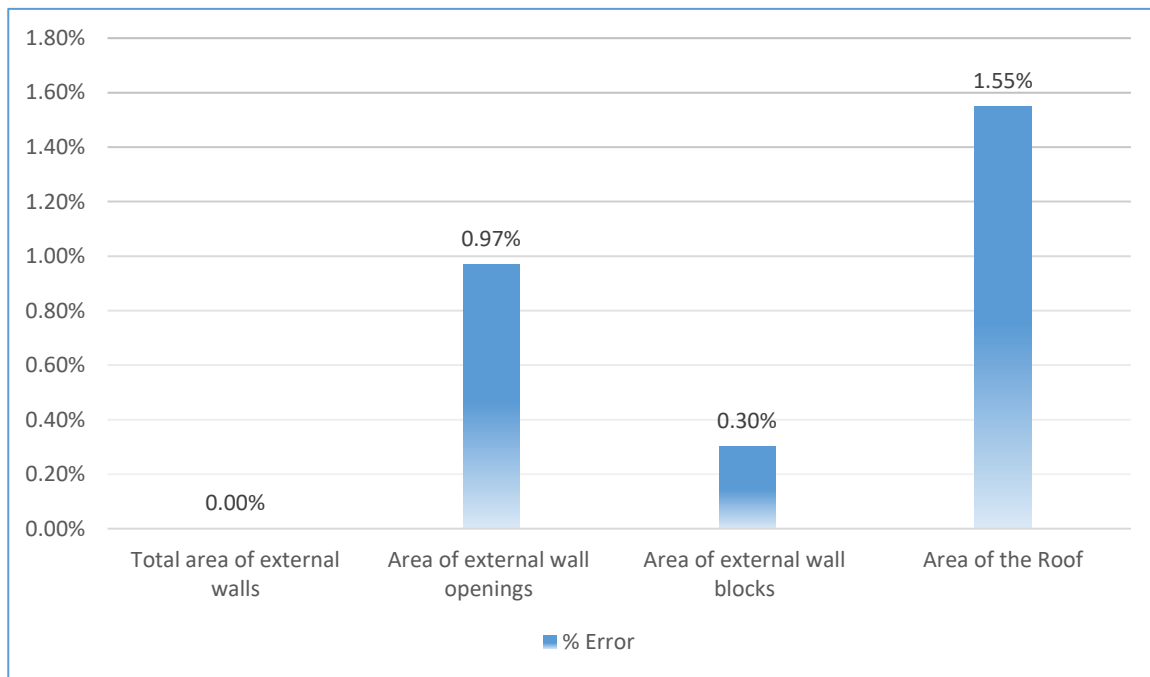


Figure 60. Percentage of error in the surface area for floorplan No.12.

Table 63. Surface areas of floor plan no. 13.

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	283.1	283.2
Area of external wall openings	57.8	58.5
Area of external wall blocks	225.3	224.7
Area of the Roof	189	182

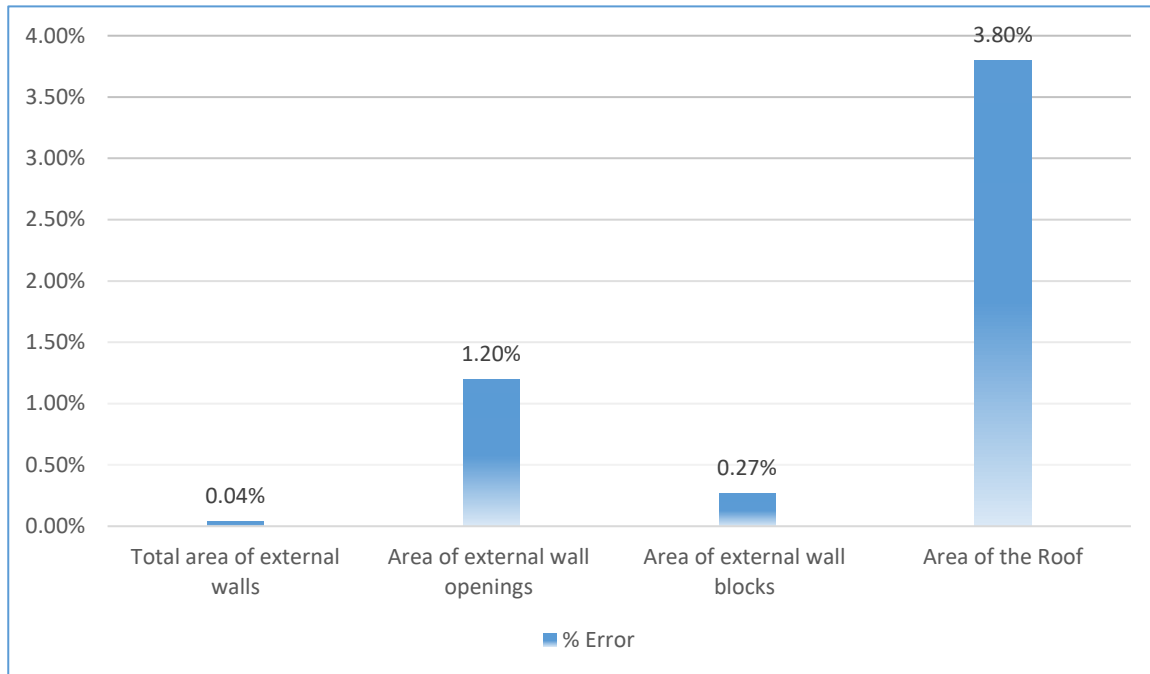


Figure 61. Percentage of error in the surface area for floorplan No.13.

Table 64. Surface areas of floor plan no. 14.

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls	331.1	327.8
Area of external wall openings	80.3	77.2
Area of external wall blocks	250.8	250.6
Area of the Roof	197	199

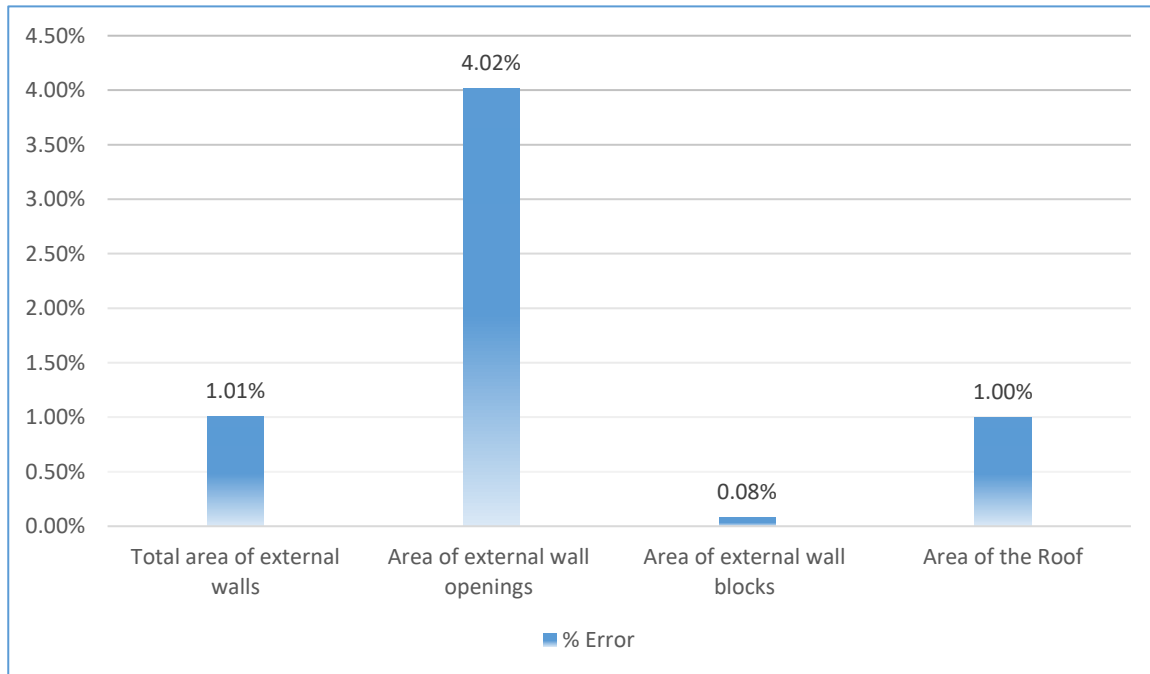


Figure 62. Percentage of error in the surface area for floorplan No.14.

Table 65. Surface areas of floor plan no. 15.

Details	Algorithm output (m^2)	Actual (m^2)
Total area of external walls*	343.8	345.2
Area of external wall openings	90.2	89.8
Area of external wall blocks	253.6	255.4
Area of the Roof	225	221

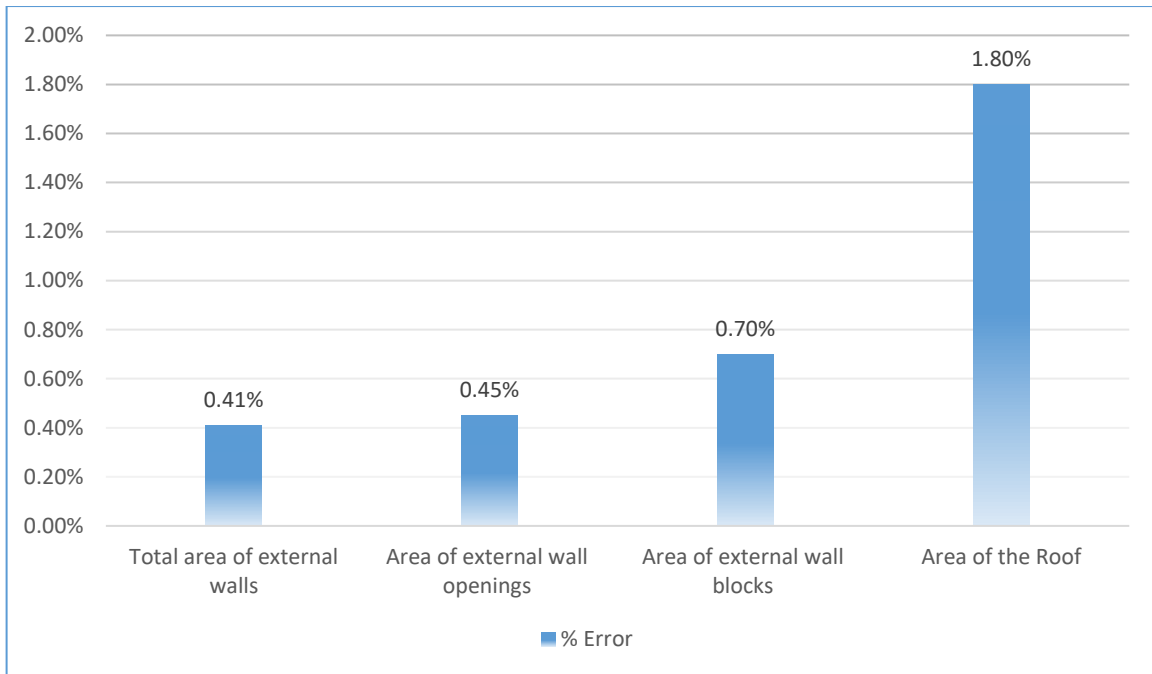


Figure 63. Percentage of error in the surface area for floorplan No.15.

Table 66 indicates the average percentage error for each floorplan, specifically concerning the surface area calculations. This table allows us to evaluate the accuracy of the algorithm's surface area calculations and assess its performance across different floorplans.

Upon analyzing Table 60, it is observed that floorplan no. 8 has the lowest percentage error, indicating that the algorithm's surface area calculation for this particular floorplan is highly accurate. On the other hand, floorplan no. 10 exhibits the highest percentage error, implying that the algorithm's calculation for this floorplan deviates more significantly from the actual surface area calculations.

Despite the variation in error percentages across different floorplans, it is important to note that the overall impact of these errors on the decision-making process remains minimal. The percentages fall within allowable

tolerance levels, meaning they do not significantly influence the conclusions or recommendations derived from the analysis.

The presence of minimal error percentages in the remaining floorplans suggests that, in general, the algorithm performs well in accurately calculating the surface areas.

Table 66. Average percentage of error for each floorplan

Floorplan No.	Average percentage of error
1	0.68%
2	1.17%
3	0.83%
4	0.86%
5	0.84%
6	1.17%
7	1.26%
8	0.62%
9	1.05%
10	2.14%
11	1.03%
12	0.71%
13	1.33%
14	1.52%
15	0.84%

7.3 Floor plan Heat Gain Calculation Error

Table 67 provides valuable information regarding the error in calculating the heat gain factor, comparing the results obtained by the algorithm with those

obtained by the engineering offices. This table allows for a comprehensive assessment of the algorithm's accuracy in determining the heat gain factors for the different floor plans.

The calculation of the heat gain factor by the engineering offices requires a significant amount of time and effort. In this case, the duration required for the engineer to obtain results for the 15 floor plans on a continuous working schedule is reported as 17 hours. This emphasizes the time-consuming nature of the manual calculations involved in determining the heat gain factor for each floor plan using traditional methods.

In contrast, the algorithm is designed to automate this process and significantly reduce the time required to obtain the results. By running the algorithm, the heat gain factors for all 15 floor plans can be calculated within a short span of 15 seconds. This remarkable reduction in processing time showcases the efficiency and effectiveness of the algorithm in streamlining the analysis and decision-making process.

The ability of the algorithm to generate results rapidly can have a significant impact on project timelines and resource allocation. The quick turnaround time allows architects, engineers, and other stakeholders to access essential information promptly, enabling them to make informed decisions and take necessary actions promptly.

Moreover, the algorithm's speed and efficiency contribute to increased productivity and cost savings. By minimizing the time spent on manual calculations, engineers can allocate their time and resources more effectively, focusing on higher-level tasks that require their expertise.

Table 67. % of error in heat gain between algorithm output and actual areas.

Floor plan No.	Heat gain factor by Algorithm	Heat gain factor by engineering office	% Error
1	2822.89	2845.1	0.78%
2	3432.11	3415	0.50%
3	5130.88	5118.9	0.23%
4	3218.35	3242.5	0.74%
5	4840.46	4836.6	0.08%
6	4340.81	4361.5	0.47%
7	3704.75	3664.2	1.11%
8	5013.56	4984.5	0.58%
9	4031.17	4009.4	0.54%
10	3067.88	3130.5	2.00%
11	4917.63	4885.6	0.66%
12	3863.43	3840.8	0.59%
13	2949.41	2938.3	0.38%
14	3546.26	3492.2	1.55%
15	3834.32	3826.9	0.19%

In conclusion, in the light of the results obtained above and the percentages of errors, the output indicates a very minor deference's between the algorithm output and between the engineering offices output. When it comes to the main research questions, the automation of floorplan licensing by using a deep learning algorithm will result in elimination of human interference, accurate results, and shorter duration.

7.4 Visual Management of the Heat Gain Factor

The primary objective in analyzing heat gain factors is to achieve lower values, as this directly translates into energy conservation. By reducing heat gain, buildings can maintain comfortable indoor temperatures while minimizing the need for excessive heating, resulting in energy savings and reduced environmental impact.

Figure 55 is a visual representation of the average heat gain across the 15 floorplans, with an average value of 3914 units. This figure provides a clear overview of the heat gain levels across the different floorplans, allowing for easy identification of those with higher or lower heat gain.

By referring to Figure 55, engineering offices can quickly determine where a specific floorplan stands in terms of its heat gain factor compared to others. This visual tool is valuable in guiding decision-making processes and effectively communicating with customers.

For floorplans identified with higher heat gain on Figure 55, engineering offices can provide informed recommendations and strategies to help mitigate this issue. These strategies may include minimizing opening areas, improving insulation, optimizing the placement of windows and doors, or suggesting alternative villa orientations that can reduce heat gain.

Conversely, for floorplans with lower heat gain values, engineering offices can highlight their energy efficiency and use them as examples for other customers. This can inspire and guide future designs to aim for similarly low heat gain factors, fostering a culture of energy-conscious construction and sustainable building practices.

The visual representation of heat gain factors in Figure 55 allows engineering offices to effectively communicate the importance of energy conservation to their customers. By demonstrating the range of heat gain values among the floorplans, customers can better understand the significance of achieving a lower heat gain factor and the potential energy savings associated with it.

Additionally, Figure 64 serves as a tool for benchmarking and setting targets. Engineering offices can compare the heat gain factor of a specific floorplan to the average value or those of other floorplans. This comparison helps customers and designers evaluate the relative energy performance of a building and make informed decisions to improve its energy efficiency.

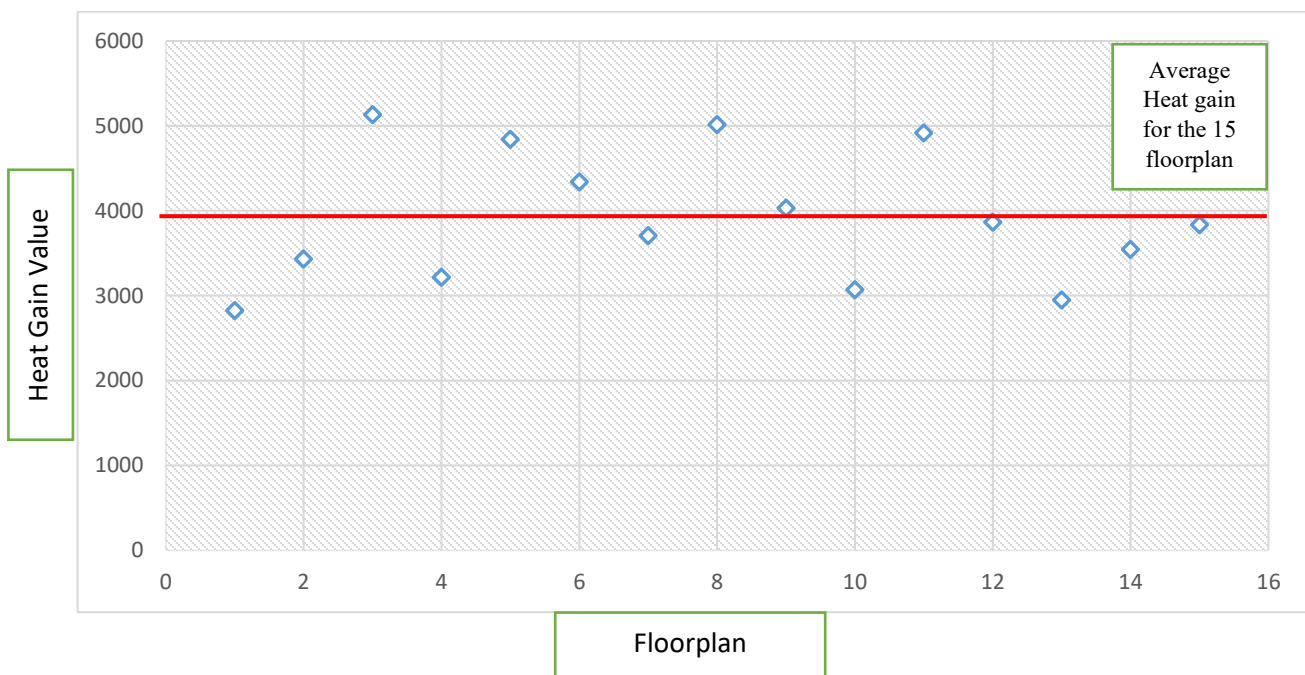


Figure 64. % of error in the surface area for floorplan No.15.

Chapter 8

Conclusions

Chapter 8 Conclusions

8.1 Conclusions

The goal of this study is to find the standards and requirements that Bahrain local municipalities must meet in order to approve a new layout. including energy conservation as one of those requirements to speed up the floorplan approval procedure. As a result, the model was created using a pre-trained CNN model. The model is trained using a dataset of floorplans to improve its performance by viewing a large number of floorplans and attempting to read all of their parameters, including rooms, halls, doors, walls, and bathrooms. Verification of the model to ensure that the criteria are valid while the model is in use. attaining the research goals by critically assessing each point.

The text labels on the floorplans were removed using an OCR technique. After using a morphological blur filter to reduce background noise, the image is binarized to create a black-and-white version. The data structures contained records of the interesting points, such as the corner points defining the doorways.

The floor plan's corner points immediately following cleaning. The Harris corner detection approach was utilized to locate the corner plots. The program then located every corner without differentiating between the outer and inner points. This technique also took the plot's turning points into consideration.

Corner points for the scaling box were also determined because it is crucial to remove the scaling box after figuring out the scale of the walls and the areas to prevent any confusion in the floorplan recognition process.

The doors and windows were joined to create a solid layout that entirely covers the entrance space with color to distinguish the various rooms. After that, the midway was located by separating the point pairs connected to doorways. by computing the proportion of (0-pixels) to (1-pixels) while applying a pixel condition to a mask centered at the door edge.

By assuming that every door will be defined by two edge points with the same (x-coordinate) or (y-coordinate), the voids were connected.

The number of rooms and the corners that defined their borders were identified. The corner points for each room are then mapped. The corner defining points are then sorted for each room in increasing order, and internal pixel colors are generated at random for each room. The plot was established as the outer room that has no other rooms inside it in order to identify the area for each room. Then the rooms are colored with that randomized color (now the seed value set is 0, if different colors are needed, a different seed value must be used).

The floorplans were inserted into the model after the training, validation, and testing cycles were finished. This produced results in terms of decision-making regarding whether those floorplans are meeting Bahrain municipality regulations for zoning RA and can be accepted and licensed or modification must take place based on the predefined criteria.

The engineering office was instructed to manually determine the length and width of each floorplan using an AutoCAD file in order to calculate the

dimensions of each floorplan. To ensure the accuracy of such dimensions, the office engineers calculated the dimensions, and his superior verified them.

Then, the algorithm was designed to find out the areas in square meter of the outer surface of the floor plan for the flexible opened surface such as windows or doors or for a solid surface such as block, concrete, steel, or wood.

The real data temperature, which contains more than (1754) stations that give live temperature data every three hours, was downloaded in order to determine the temperature differential for the building and compute the heat gain of the floor design.

Utilizing the information previously collected regarding the U-Values, exterior surface areas, and temperature differences, the heat gain was computed for 15 floorplans.

By comparing the floorplan's dimensions from the AutoCAD file with the algorithm output and the percentage of inaccuracy, the results of the algorithm were confirmed.

In order to confirm the accuracy of the results, the surface area that the algorithm recovered was contrasted with the actual measurements from the AutoCAD file. The accuracy of the task was then evaluated by calculating the percentage error.

The findings of the algorithm and the manual calculation made by the engineering offices were compared in order to confirm the conclusions of heat gain.

In conclusion, the algorithm produced very accurate results as compared to the manual generation performed by engineering offices, automating the process and removing human error.

References

References

- AASHTO. (2010). AASHTO LRFD Bridge Design Specifications (5th ed.). Washington: American Association of State Highway and Transportation Officials.
- Academic. (2019, January 10). Box girder bridge. Retrieved from Academic Dictionaries and Encyclopedias: <http://enacademic.com/dic.nsf/enwiki/726784>
- Academic Dictionaries and Encyclopedias. (2019). Box girder bridge. Retrieved January 10, 2019, from Academic Dictionaries and Encyclopedias: <http://enacademic.com/dic.nsf/enwiki/726784>
- ADO. (2019, January 10). Bridges & Flyovers. Retrieved from ADO Additives: <https://www.adochemicals.com/portfolio/bridges-flyovers/>
- ADO Additives. (2019). Bridges & Flyovers. Retrieved January 10, 2019, from ADO Additives: <https://www.adochemicals.com/portfolio/bridges-flyovers/>
- Ahsan, R., Rana, S., & Ghani, S. N. (2012). Cost optimum design of posttensioned I-girder bridge using global optimization algorithm. *Journal of Structural Engineering*, 138, 273-284. doi:10.1061/(ASCE)ST.1943-541X.0000458
- Akay, B., & Karaboga, D. (2009). Parameter tuning for the artificial bee colony algorithm. In N. T. Nguyen, R. Kowalczyk, & S. -M. Chen (Ed.), *ICCCI 2009*, LNAI 5796 (pp. 608-619). Heidelberg: Springer.
- Akin, A., & Saka, M. (2015). Harmony search algorithm based optimum detailed design of reinforced concrete plane frames subject to ACI 318-05 provisions. *Computers and Structures*, 147, 79-95.
- Application Programming Interface. (2018, January 15). Retrieved from CSI Computers and Structures, Inc: <https://www.csiamerica.com/application-programming-interface>
- Aydin, Z., & Ayvaz, Y. (2010). Optimum topology and shape design of prestressed concrete bridge girders using a genetic algorithm. *Struct Multidisc Optim*, 41, 151-162. doi:10.1007/s00158-009-0404-2
- Aydin, Z., & Ayvaz, Y. (2013). Overall cost optimization of prestressed concrete bridge using genetic algorithm. *KSCE Journal of Civil Engineering*, 17(4), 769-776. doi:10.1007/s12205-013-0355-4

- Aydogdu, I., & Saka, M. P. (2012). Ant colony optimization of irregular steel frames including elemental warping effect. *Advances in Engineering Software*, 44, 150-169.
- Aydogdu, I., Akin, A., & Saka, M. (2016). Design optimization of real world steel space frames using artificial bee colony algorithm with levy flight distribution. *Advances in Engineering Software*, 92, 1-14.
doi:10.1016/j.advengsoft.2015.10.013
- Azad, S. K., Azad, S. K., & Hasancebi, O. (2016). Structural optimization using big bang-big crunch algorithm: a review. *International Journal of Optimization in Civil Engineering*, 6(3), 433-445.
- Baldomir, A., Hernandez, S., Nieto, F., & Jurado, J. (2010). Cable optimization of a long span cable stayed bridge in La Coruña (Spain). *Advances in Engineering Software*, 41, 931–938. doi:10.1016/j.advengsoft.2010.05.001
- Barker, R. M., & Puckett, J. A. (2013). *Design of Highway Bridges: An LRFD Approach* (3rd ed.). New Jersey: John Wiley and Sons.
- BCSA, & SCI. (2019, January 9). Tied-arch bridges. Retrieved from SteelConstruction.info: https://www.steelconstruction.info/Tied-arch_bridges
- Belevicius, R., Juozapaitis, A., & Rusakevicius, D. (2018). Parameter study on weight minimization of network arch bridges. *Periodica Polytechnica Civil Engineering*, 62(1), pp. 48-55. doi:10.3311/PPci.10036
- Bruno, D., Lonetti, P., & Pascuzzo, A. (2016). An optimization model for the design of network arch bridges. *Computers and Structures*, 170, 13-25.
- Chen, W.-F., & Duan, L. (2014). *Bridge Engineering Handbook: Superstructure Design* (2nd ed.). Boca Raton: CRC Press.
- Cheng, J. (2010). Optimum design of steel truss arch bridges using a hybrid genetic algorithm. *Journal of Constructional Steel Research*, 66, 1011-1017.
doi:10.1016/j.jcsr.2010.03.007
- Cheng, M.-Y., & Prayogo, D. (2014). Symbiotic organisms search: a new metaheuristic optimization algorithm. *Computers and Structures*, 139, 98-112.
doi:10.1016/j.compstruc.2014.03.007
- CSI Computers and Structures, I. (2018). Application Programming Interface. Retrieved January 15, 2018, from CSI Computers and Structures, Inc: <https://www.csiamerica.com/application-programming-interface>
- CSI Knowledge Base. (2018). CSI Knowledge Base SAP2000 Home. Retrieved January 15, 2018, from CSI Knowledge Base: <https://wiki.csiamerica.com/display/sap2000/Home>

- CSI Knowledge Base SAP2000 Home. (2018, January 15). Retrieved from CSI Knowledge Base: <https://wiki.csiamerica.com/display/sap2000/Home>
- Daloglu, A. T., Artar, M., Ozgan, K., & Karakas, A. I. (2018). Optimum design of braced steel space frames including soil-interactions via teaching-learning-based optimization and harmony search algorithms. *Advances in Civil Engineering*, 2018, 1-16. doi:10.1155/2018/3854620
- Dlouhý, L., & Lepš, M. (2012). Optimization of tendon geometry of post-tensioned concrete bridges using genetic algorithms. Stirlingshire: Civil-Comp Press.
- Ermopoulos, I. C., & Ioannidis, S. S. (1985). Optimum rise design of steel arch bridges. *J. Construct. Steel Research*, 5, 303-310.
- Erol, O. K., & Eksin, I. (2006). A new optimization method: big bang-big crunch. *Advances in Engineering Software*, 106-111. doi:10.1016/j.advengsoft.2005.04.005
- Fu, G. (2013). *Bridge Design and Evaluation: LRFD and LRFR* (1st ed.). New Jersey: John Wiley & Sons.
- Fu, K.-C., Zhai, Y., & Zhou, S. (2005). Optimum design of welded steel plate girder bridges using a genetic algorithm with elitism. *Journal of Bridge Engineering*, 10(3), 291-301. doi:10.1061/(ASCE)1084-0702(2005)10:3(291)
- Garcia-Segura, T., Yepes, V., Alcala, J., & Perez-Lopez, E. (2015). Hybrid harmony search for sustainable design of post-tensioned concrete box-girder pedestrian bridges. *Engineering Structures*, 92, 112-122. doi:10.1016/j.engstruct.2015.03.015
- Garg, H. (2014, July). Solving structural engineering design optimization problems using an Artificial Bee Colony algorithm. *Journal of Industrial and Management Optimization*, 10(3), 777-794. doi:10.3934/jimo.2014.10.777
- Hasancebi, O., & Azad, S. K. (2012). An exponential big bang-big crunch algorithm for discrete design optimization os steel frames. *Computers and Structures*, 110-111, 167-179. doi:10.1016/j.compstruc.2012.07.014
- Hasancebi, O., Çarbas, S., Dogan, E., Erdal, F., & Saka, M. (2009). Performance evaluation of metaheuristic search techniques in the optimum design of real size pin jointed structures. *Computers and Structures*, 87, 284-302.
- Hasancebi, O., Çarbas, S., Dogan, E., Erdal, F., & Saka, M. (2010). Comparison of non-deterministic search techniques in the optimum design of real size steel frames. *Computers and Structures*, 88, 1033-1048.
- Hassan, M. (2013). Optimization of stay cables in cable-stayed bridges using finite element, genetic algorithm, and B-spline combined technique. *Engineering Structures*, 49, 643–654.

- Hassan, M., Nassef, A., & El Damatty, A. (2012). Determination of optimum post-tensioning cable forces of cable-stayed bridges. *Engineering Structures*, 44, 248-259.
- He, Y., & Aref, A. J. (2013). An optimization design procedure for fiber reinforced polymer web-core sandwich bridge deck systems. *Composite Structures*, 60, 183-195. doi:10.1016/S0263-8223(02)00311-2
- Islam, N., & Ahsan, R. (2010). Optimization of hanger arrangement of network arch bridges. *IABSE-JSCE Joint Conference on Advances in Bridge Engineering-II*, (pp. 107-119). Dhaka, Bangladesh. Retrieved April 7, 2018, from <http://www.iabse-bd.org>
- Islam, N., Rana, S., Ahsan, R., & Ghani, S. N. (2014). An optimized design of network arch bridge using global optimization algorithm. *Advances in Structural Engineering*, 17(2), 197-210.
- Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization [Technical Report-TR06]. Kayseri: Erciyes University, Engineering Faculty, Computer Engineering Department.
- Karaboga, D. (2010). Artificial bee colony algorithm. *Scholarpedia*. 5(3), 6915. doi:10.4249/scholarpedia.6915
- Karaboga, D., & Akay, B. (2011). A modified artificial bee colony (ABC) algorithm for constrained optimization problems. *Applied Soft Computing*, 11, 3021-3031. doi:10.1016/j.asoc.2010.12.001
- Karaboga, D., Aslantas, V., Karaboga, N., Basturk, B., Okdem, S., Ozturk, C., . . . Boudardara, F. (2018, May 21). Artificial Bee Colony (ABC) Algorithm Homepage. Retrieved from Artificial Bee Colony (ABC) Algorithm: <https://abc.erciyes.edu.tr/>
- Karakas, A. I., Ozgan, K., & Daloglu, A. T. (2016). A consistent FEM-vlasov model for hyperbolic cooling towers on layered soil under unsymmetrical wind load. *Wind and Structures*, 22(6), 617-633. doi:10.12989/was.2016.22.6.617
- Karas, S. (2016, October 12). History of Sustainable Bridge Solutions. *Structural Bridge Engineering*. (S. Shahidan, S. N. Mokhtar, M. H. Wan Ibrahim, N. Jamaluddin, Z. M. Jaini , & N. Ali, Eds.) IntechOpen. doi:10.5772/63461
- Kaveh, A., Bakhshpoori, T., & Barkhori, M. (2014). Optimum design of multi-span composite box girder bridges using cuckoo search algorithm. *Steel and Composite Structures*, 17(5), 705-719. doi:10.12989/scs.2014.17.5.705
- Kaveh, A., Maniat, M., & Naeini, M. A. (2016). Cost optimum design of post-tensioned concrete bridges using a modified colliding bodies optimization algorithm.

- Advances in Engineering Software, 98, 12-22.
doi:10.1016/j.advengsoft.2016.03.003
- Kim, J. B., Kim, R. H., & Eberle, J. R. (2013). Simplified LRFD Bridge Design (1st ed.). Boca Raton: CRC Press.
- Kumar, D. R., Bhikshma, V., Devi, C. U., & Suresh, S. (2014). Structural shape optimization of a bridge pier. Proceedings of the Twelfth International Conference on Computational Structures Technology (pp. 1-11). Stirlingshire: Civil-Comp Press.
- Lute, V., Upadhyay, A., & Singh, K. K. (2009). Computationally efficient analysis of cable-stayed bridge for GA-base optimization. Engineering Applications of Artificial Intelligence, 22, 750-758. doi:10.1016/j.engappai.2009.04.001
- Lute, V., Upadhyay, A., & Singh, K. K. (2011). Genetic algorithms-based optimization of cable stayed bridges. J. Software Engineering and Applications, 4, 571-578. doi:10.4236/jsea.2011.410066
- Marti, J. V., & Gonzalez-Vidoso, F. (2010). Design of prestressed concrete precast pedestrian bridges by heuristic optimization. Advances in Engineering Software, 41, 916-922. doi:10.1016/j.advengsoft.2010.05.003
- Marti, J. V., Gonzalez-Vidoso, F., Yepes, V., & Alcala, J. (2013). Design of prestressed concrete precast road bridges with hybrid simulated annealing. Engineering Structures, 48, 342-352. doi:10.1016/j.engstruct.2012.09.014
- Martinez, F. J., Gonzalez-Vidoso, F., Hospitaler, A., & Alcala, J. (2011). Design of tall bridge piers by ant colony optimization. Engineering Structures, 33, 2320-2329. doi:10.1016/j.engstruct.2011.04.005
- Martinez, F. J., Gonzalez-Vidoso, F., Hospitaler, A., & Yepes, V. (2010). Heuristic optimization of RC bridge piers with rectangular hollow sections. Computers and Structures, 88, 375-386. doi:10.1016/j.compstruc.2009.11.009
- Martinez-martin, F. J., Gonzalez-Vidoso, F., Hospitaler, A., & Yepes, V. (2012). Multi-objective optimization design of bridge piers with hybrid heuristic algorithms. Journal of Zhejiang University-SCIENCE, 13(6), 420-432. doi:10.1631/jzus.A1100304
- Namin, A. A. (2012). Structural evaluation of tied-arch and truss bridges subjected to wind and traffic loading (Master's thesis). Gazimagusa: Eastern Mediterranean University.
- Ney, L., de Goyet, V. d., & Maquoui, R. (1991). Optimum bracing of the arches of tied-arch bridges. Construction Steel Research, 18, 239-249.
- Peng, D. M., & Fairfield, C. A. (1999). Optimal design of arch bridges by integrating genetic algorithms and the mechanism method. Engineering Structures, 21, 75-82.

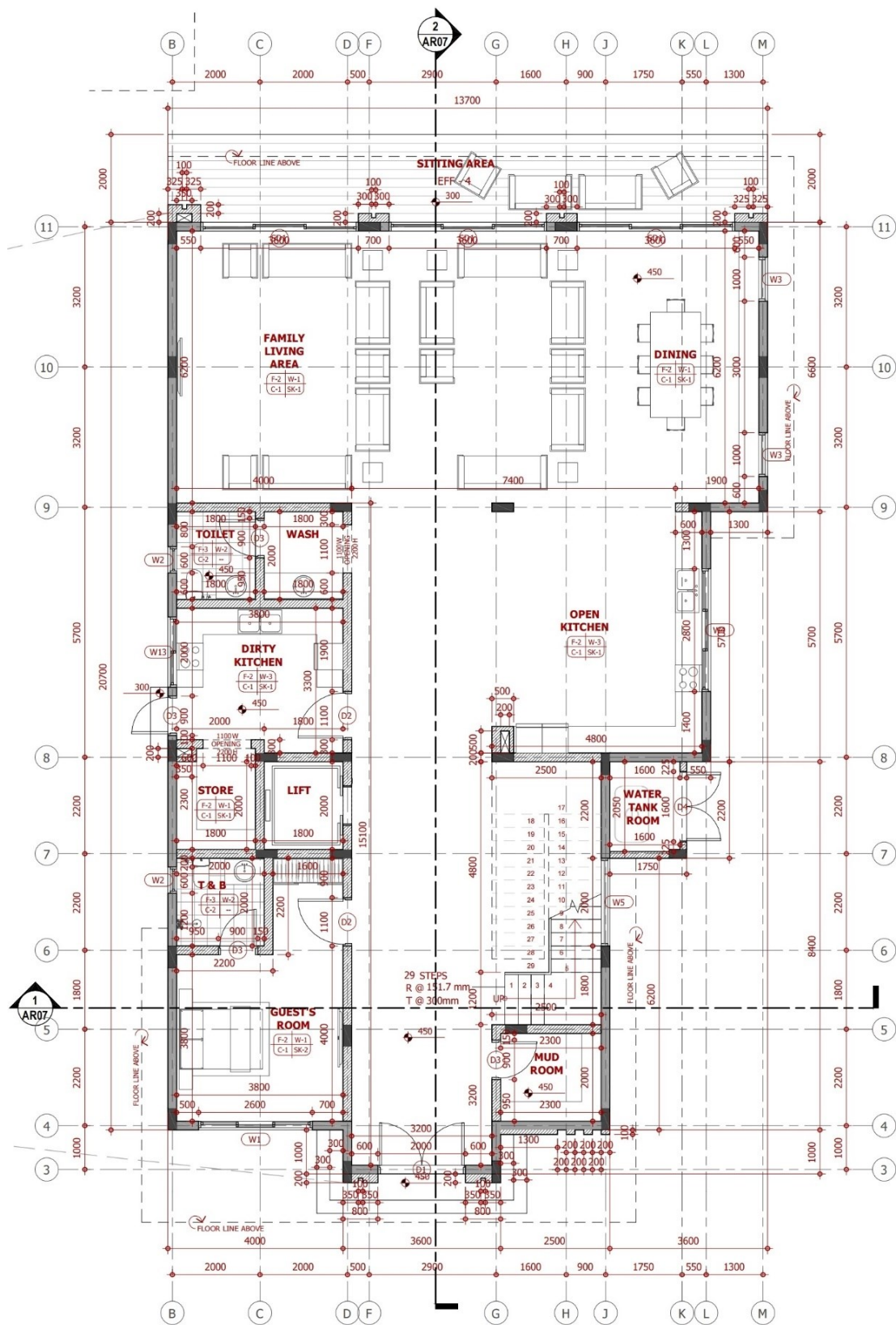
- Perea, C., Alcala, J., Yepes, V., Gonzalez-Vidoso, F., & Hospitaler, A. (2008). Design of reinforced concrete bridge frames by heuristic optimization. *Advances in Engineering Software*, 39, 676-688. doi:10.1016/j.advengsoft.2007.07.007
- Pouraminian, M., & Ghaemian, M. (2015). Shape optimisation of concrete open spandrel arch bridges. *Gradevinar*, 67(12), 1177-1185. doi:10.14256/JCE.1223.2015
- Prayogo, D. (2016, October 8). File Exchange: SOS.m. Retrieved from Mathworks: <http://www.mathworks.com>
- Rana, S., Islam, N., Ahsan, R., & Ghani, S. N. (2013). Application of evolutionary operation to the minimum cost design of continuous prestressed concrete bridge structure. *Engineering Structures*, 46, 38-48. doi:10.1016/j.engstruct.2012.07.017
- Rao, S. S. (2009). *Engineering Optimization Theory and Practice*. John Wiley and Sons.
- Saka, M. P. (2007). Optimum design of steel frames using stochastic search techniques based on natural phenoma: a review. In B. Topping (Ed.), *Civil Engineering Computations: Tools and Techniques* (pp. 105–147). Stirlingshire: Saxe-Coburg Publications.
- Saka, M. P., & Geem, Z. W. (2013). Mathematical and metaheuristic applications in design optimization of steel frame structures: an extensive review. *Mathematical Problems in Engineering*, 2013, 1-33.
- Saka, M. P., Dogan, E., & Aydogdu, I. (2013). Analysis of swarm intelligence-based algorithms for constrained optimization. In X.-S. Yang, Z. Cui, R. Xiao, A. H. Gandomi, & M. Karamanoglu (Eds.), *Swarm Intelligence and Bio-Inspired Computation: Theory and Applications* (pp. 25-48). London, Waltham: Elsevier.
- Sextos, A. G., & Balafas, G. K. (2011). Using the new SAP2000 open application programming interface to develop an interactive front-end for the modal pushover analysis of bridges. *3rd International Conference on Computational Methods in Structural Dynamics and Earthquake Engineering*, (pp. 1-14). Corfu. Retrieved from <http://congress.cimne.com>
- Simon, D. (2008, December). Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation*, 12(6), 702-713. doi:10.1109/TEVC.2008.919004
- SteelConstruction.info. (2019). Tied-arch bridges. Retrieved January 9, 2019, from SteelConstruction.info: https://www.steelconstruction.info/Tied-arch_bridges
- Talatahari, S. (2016). Symbiotic organisms search for optimum design of frame and grillage systems. *Asian Journal of Civil Engineering (BHRC)*, 17(3), 299-313.
- Taly, N. (2015). *Highway Bridge Superstructure Engineering: LRFD Approaches to Design and Analysis* (1st ed.). Boca Raton: CRC Press.

- Wikipedia. (2018, May 16). Biogeography-based optimization. Retrieved May 21, 2018, from Wikipedia: https://en.wikipedia.org/wiki/Biogeography-based_optimization
- Wikipedia. (2019, January 9). Truss bridge. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Truss_bridge
- Xing, B., & Gao, W.-J. (2014). Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms (Vol. 62). Cham, Bavaria, Germany: Springer.
- Yang, X.-S. (2010). Nature-Inspired Metaheuristic Algorithms (2nd ed.). Frome: Luniver Press.
- Yepes, V., Marti, J. V., & Garcia-Segura, T. (2015). Cost and CO2 emission optimization of precast-prestressed concrete U-beam road bridges by a hybrid glowworm swarm algorithm. *Automation in Construction*, 49, 123-134. doi:10.1016/j.autcon.2014.10.013

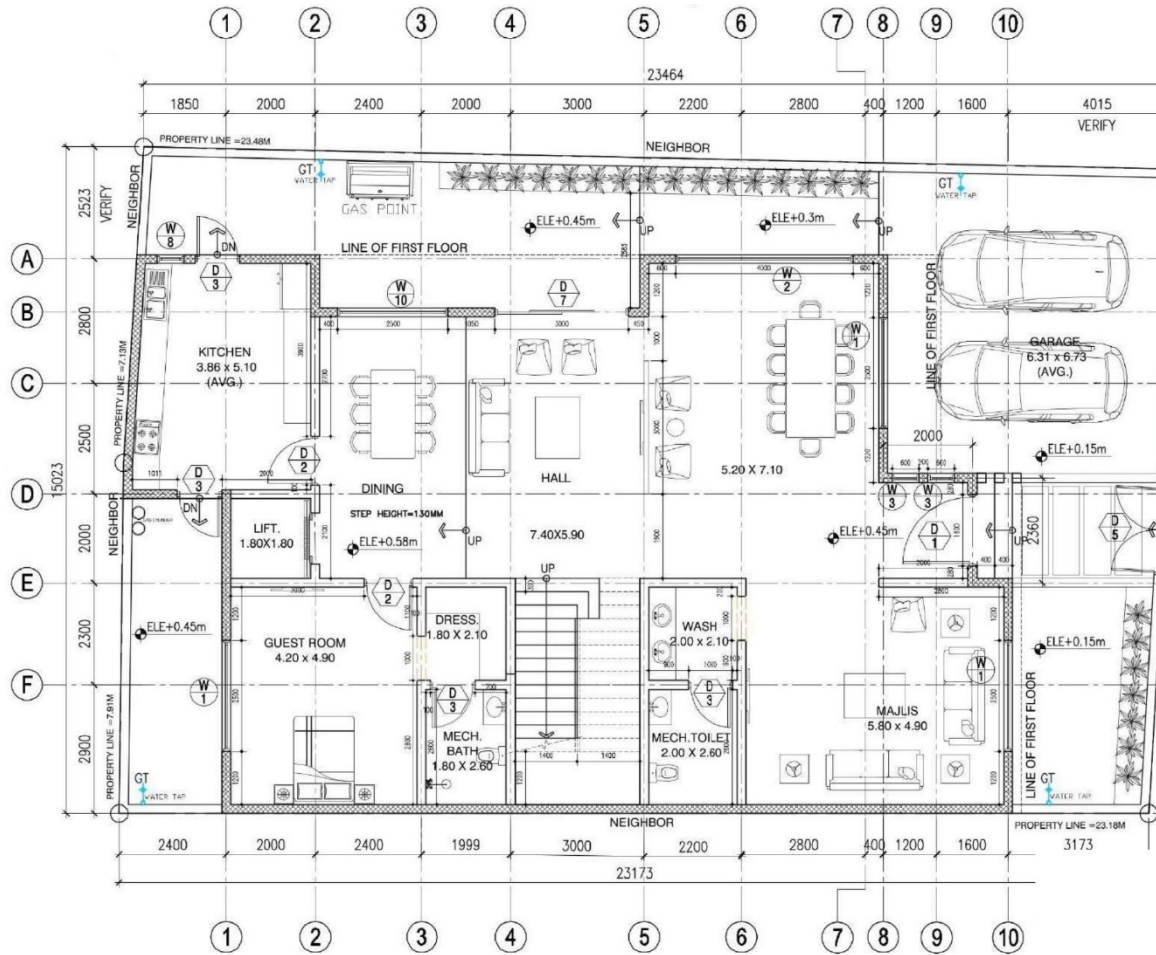
Appendix A

Floorplans received from Sultan Engineering office.

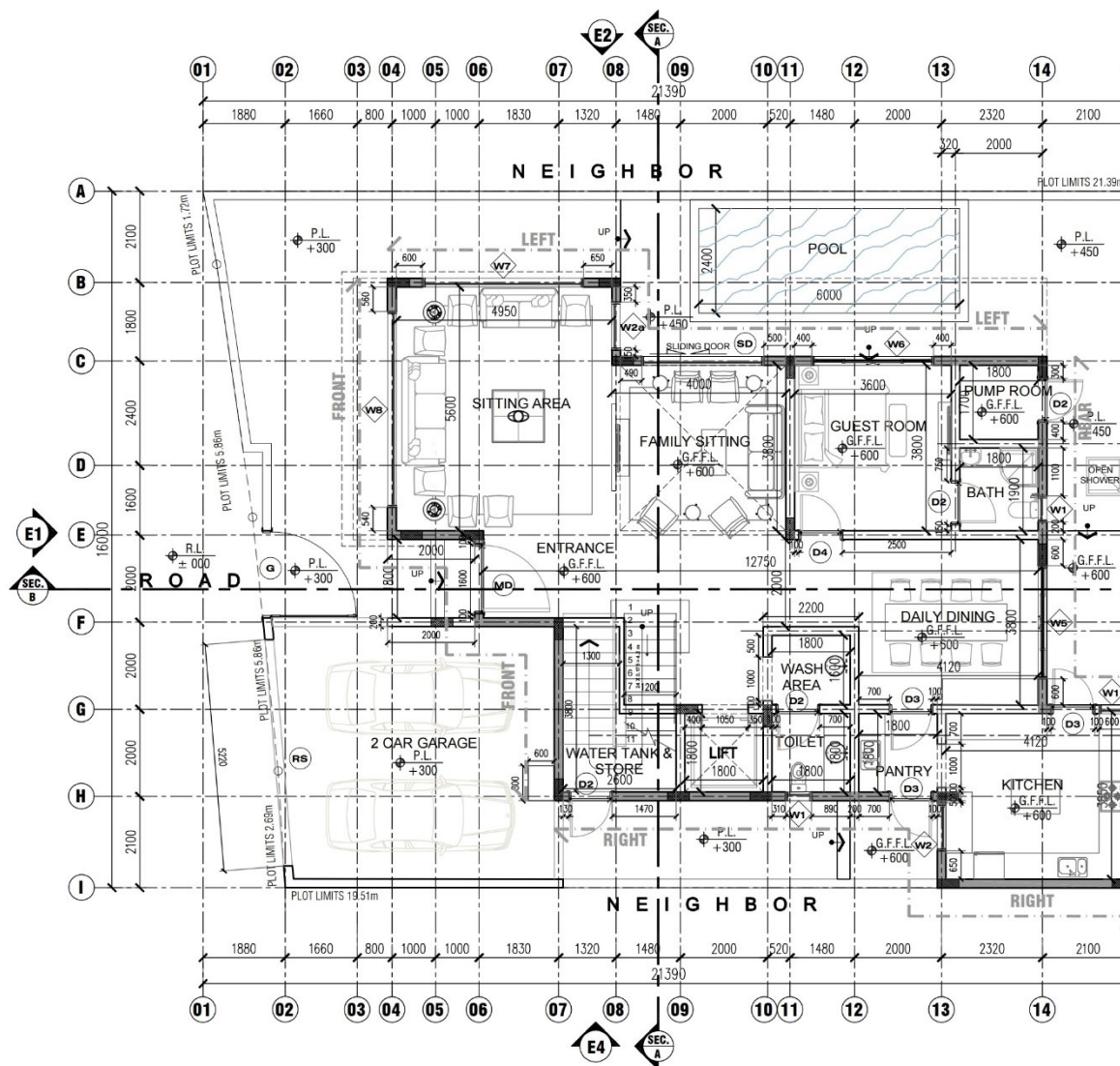
Floor Plan No. 1



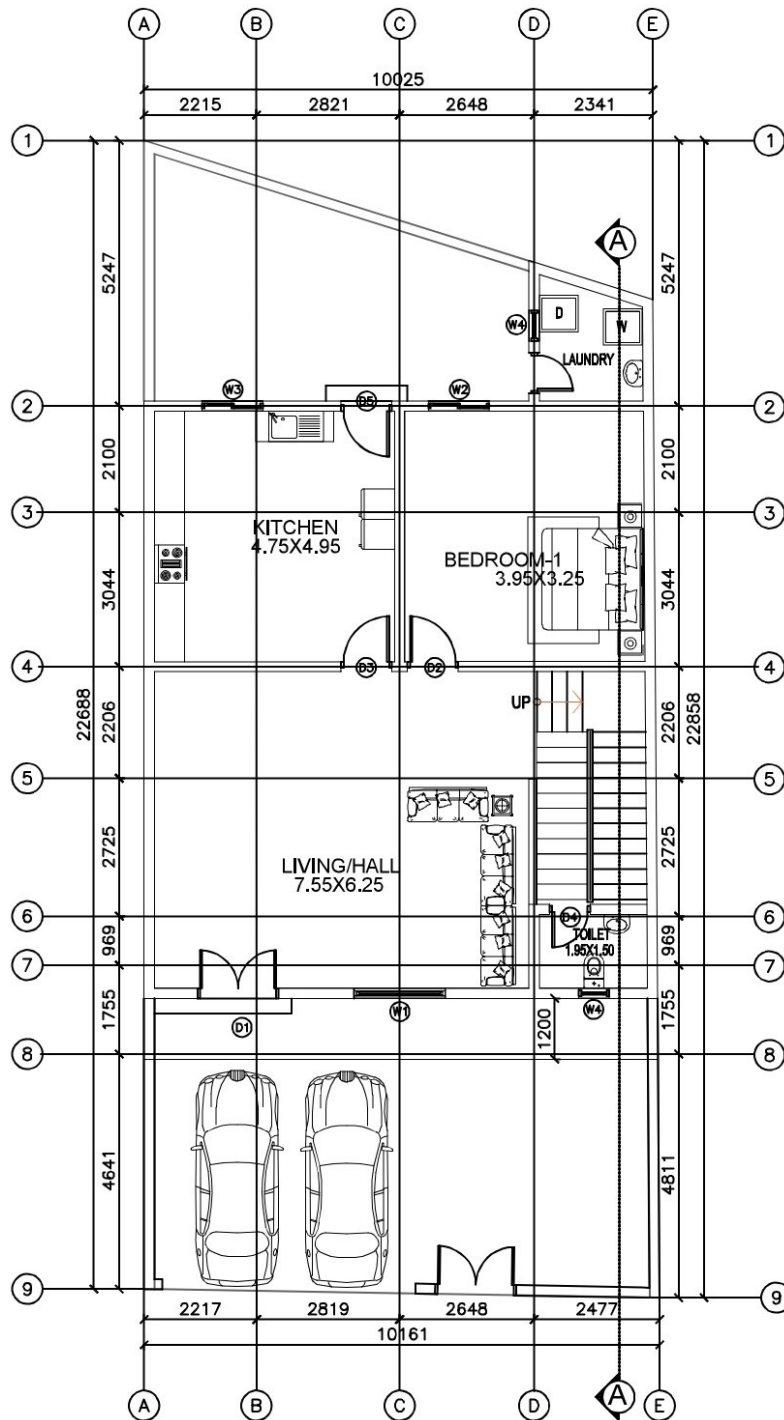
Floor Plan No. 2



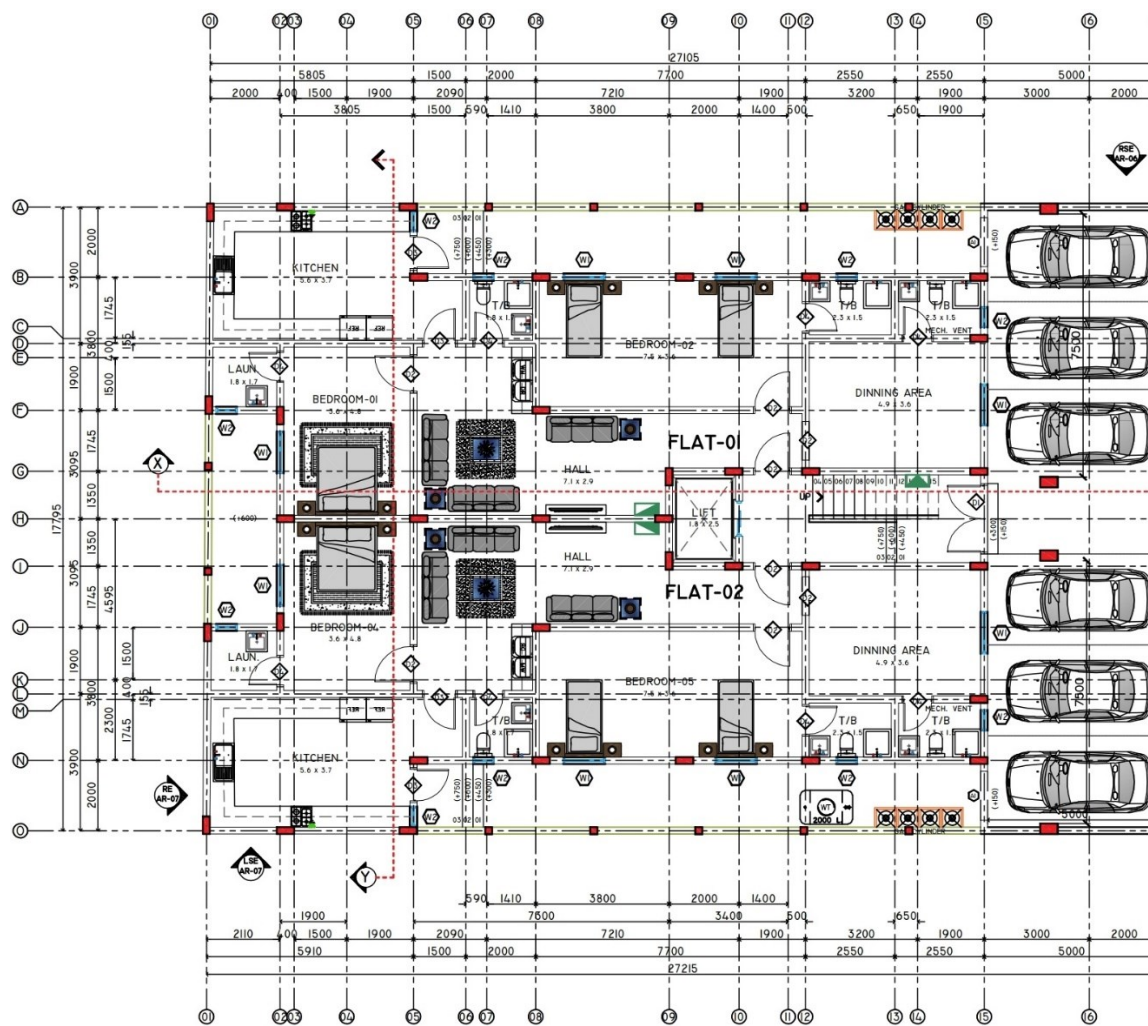
Floor Plan No. 3



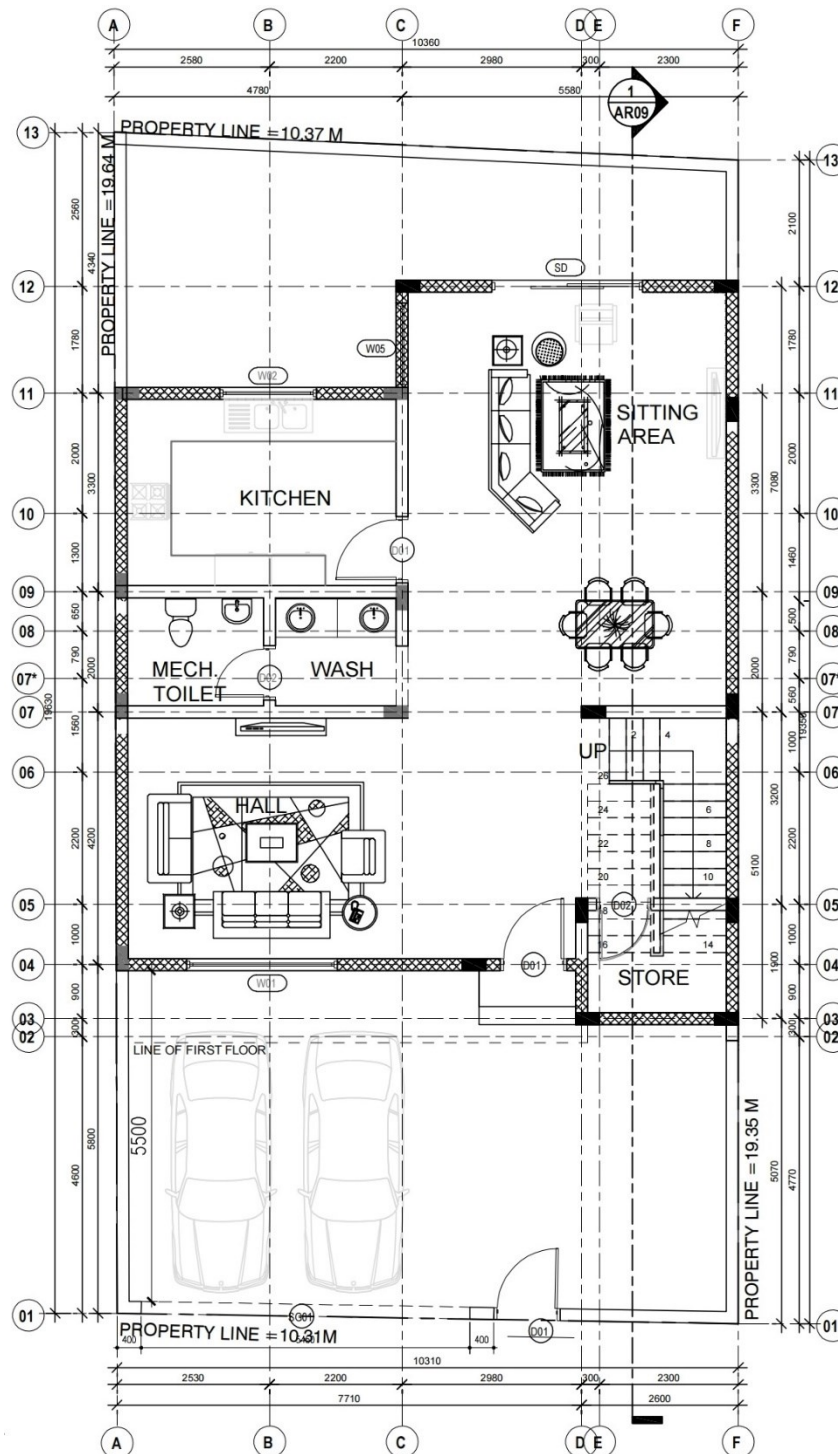
Floor Plan No. 4



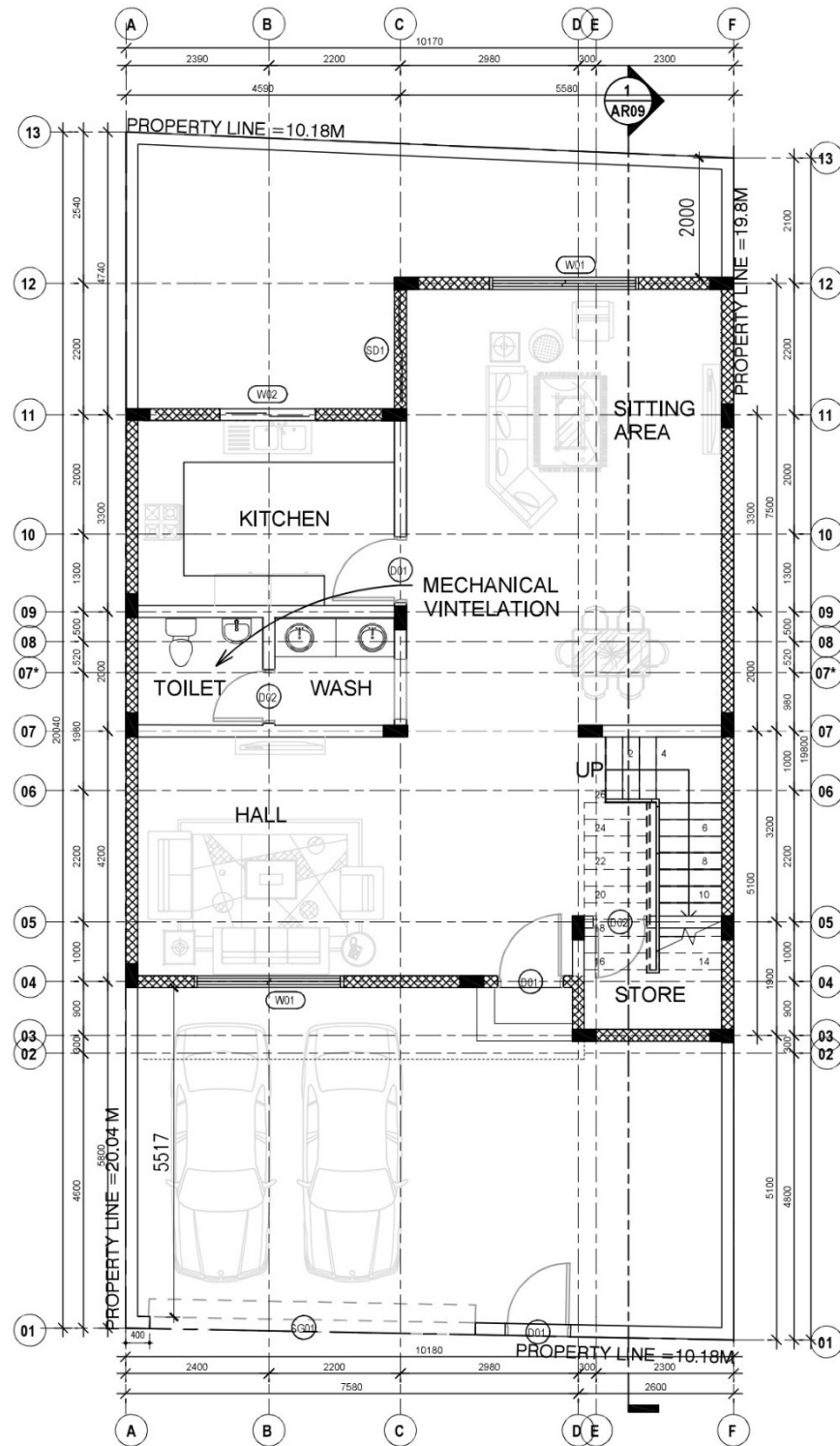
Floor Plan No. 5



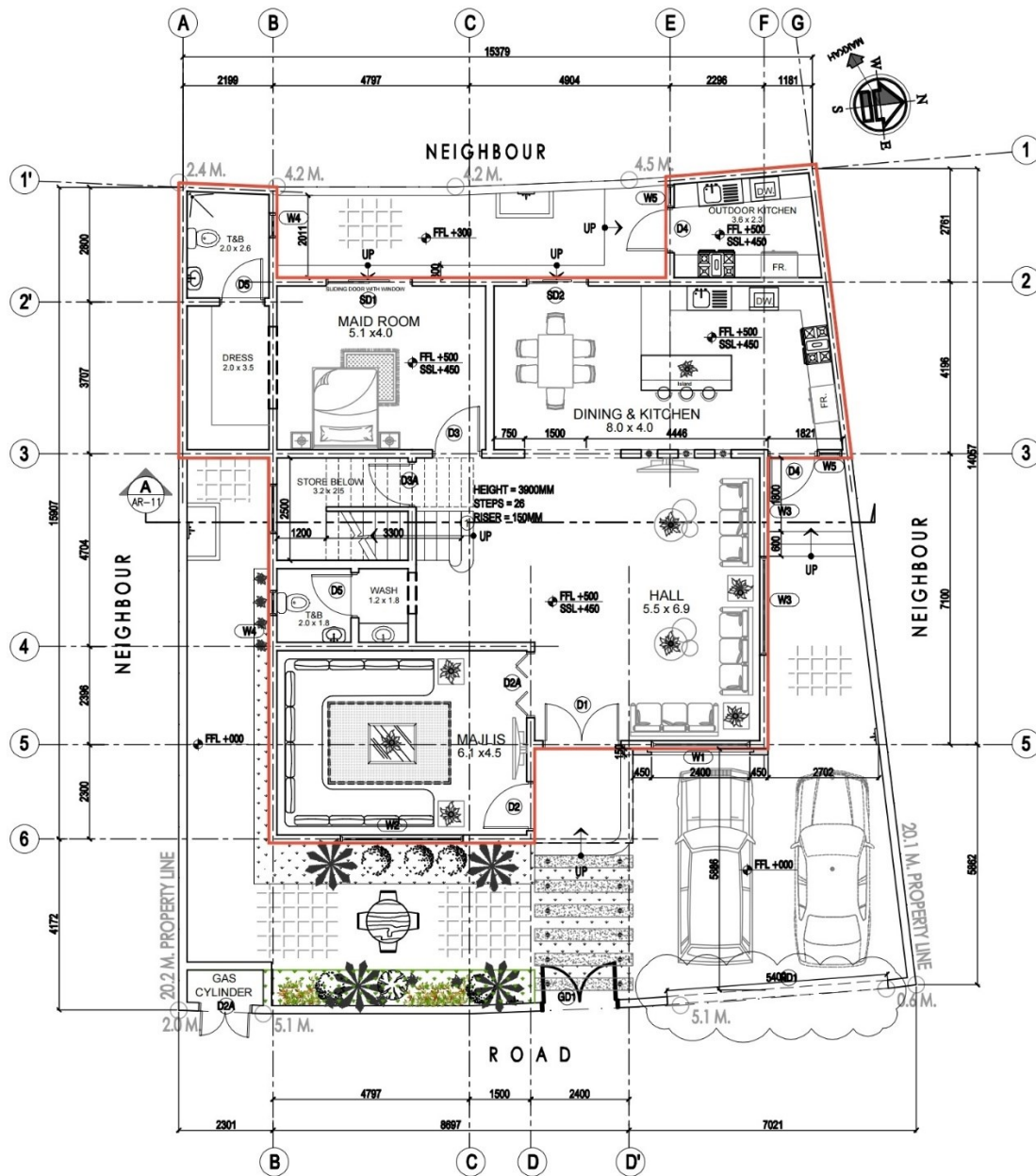
Floor Plan No. 6



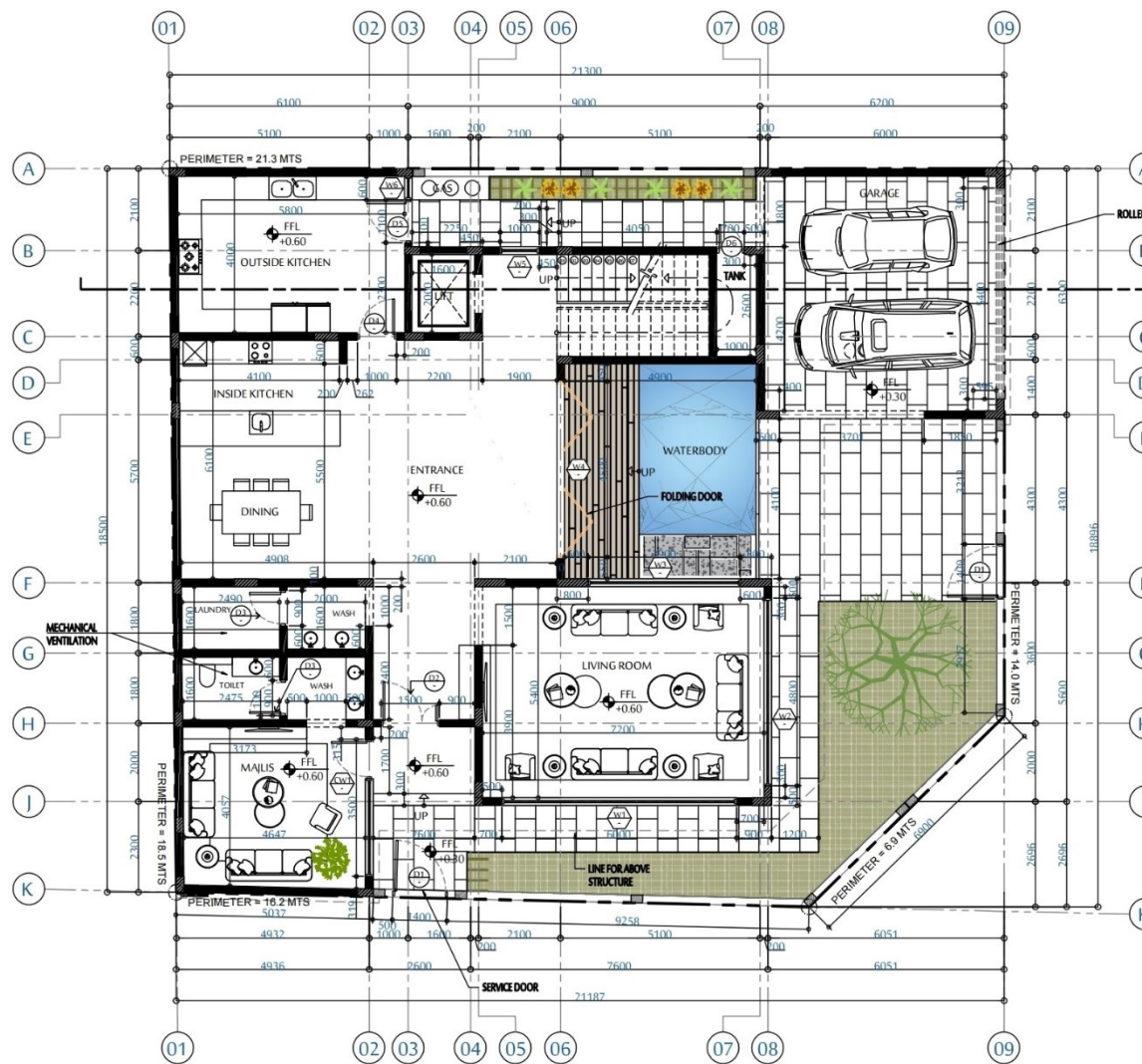
Floor Plan No. 7



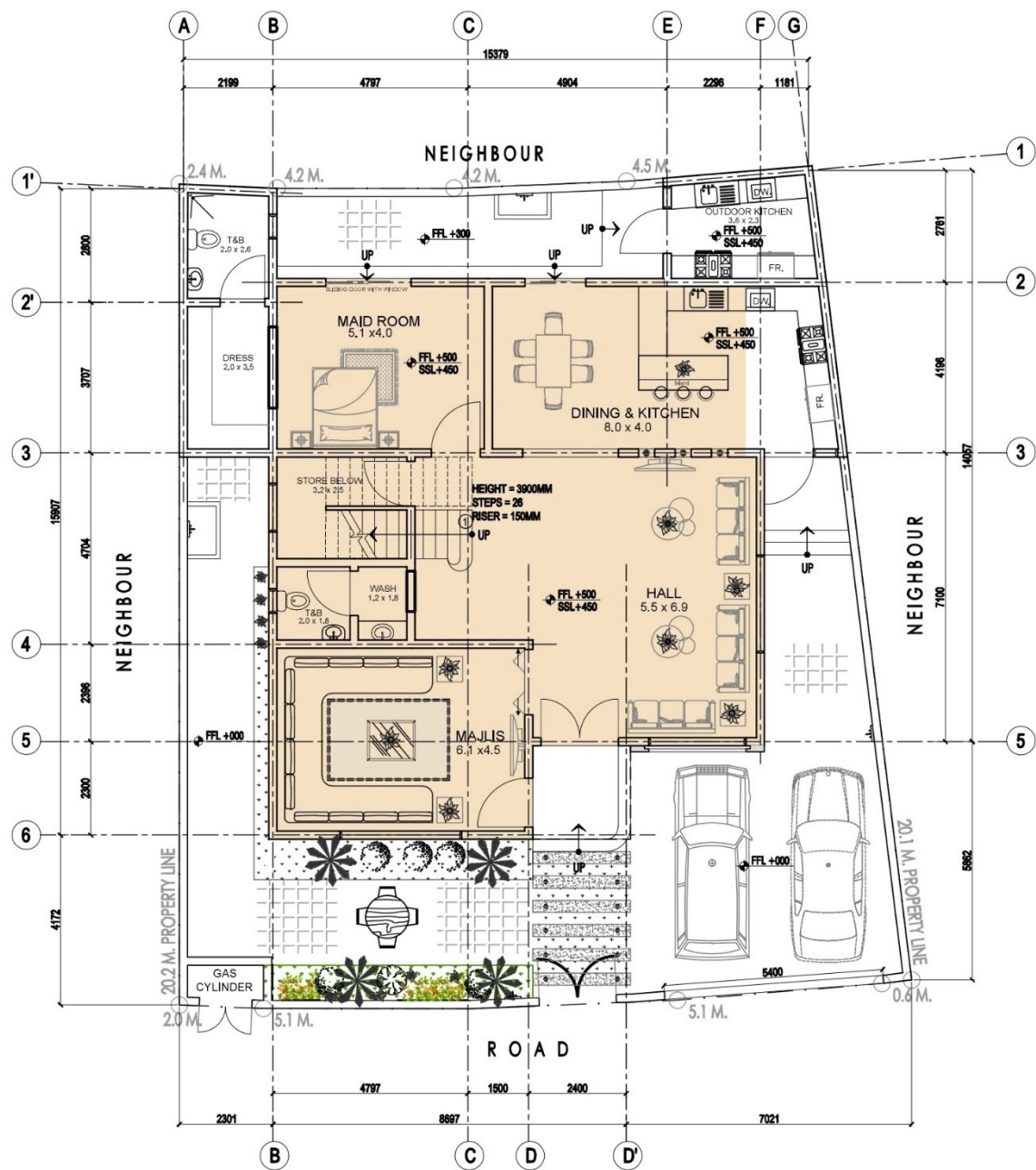
Floor Plan No. 8



Floor Plan No. 9



Floor Plan No. 10



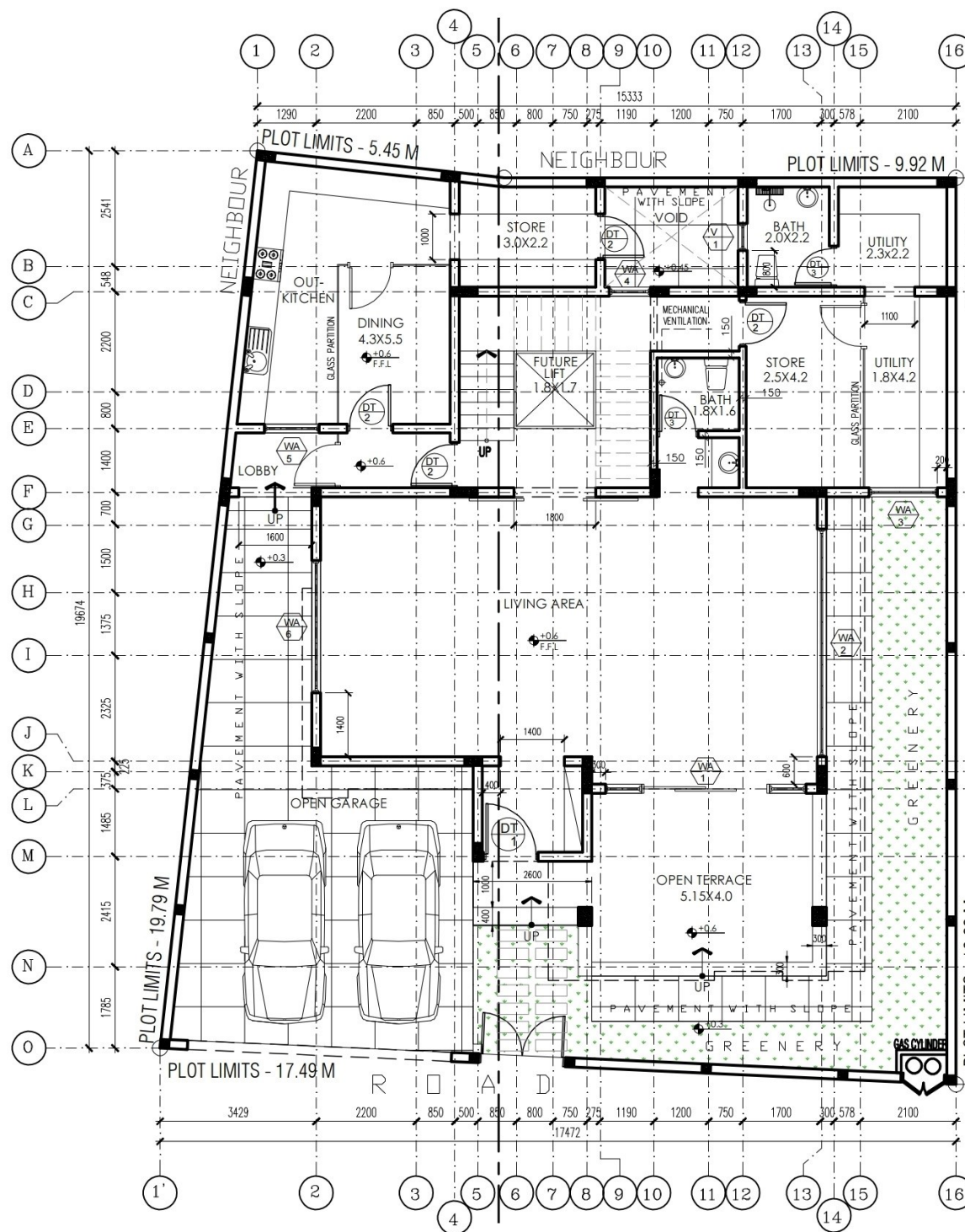
The architectural floor plan depicts a 2-story residential building with a central hall and various rooms. The plan is overlaid with a grid system (1-15 horizontally, A-N vertically) and includes various dimensions and annotations. Key features include:

- Rooms and Areas:** Dining Area, Kitchen (Inside and Outside), Living Room, Bath, Bedroom, Hall, Toilet, Wash, Laundry, and a Carport with two cars.
- Structural Elements:** A central hall with a staircase labeled "UP", an elevator, and a roller shutter.
- Dimensions and Annotations:** Numerous dimensions are provided for rooms and overall sections. Annotations include "FFL +0.60", "2 NOS. 130 MM HIGH RISER", and "GAS CYLINDERS".
- Grid System:** The plan is divided into a grid of 15 horizontal sections (1-15) and 14 vertical sections (A-N).

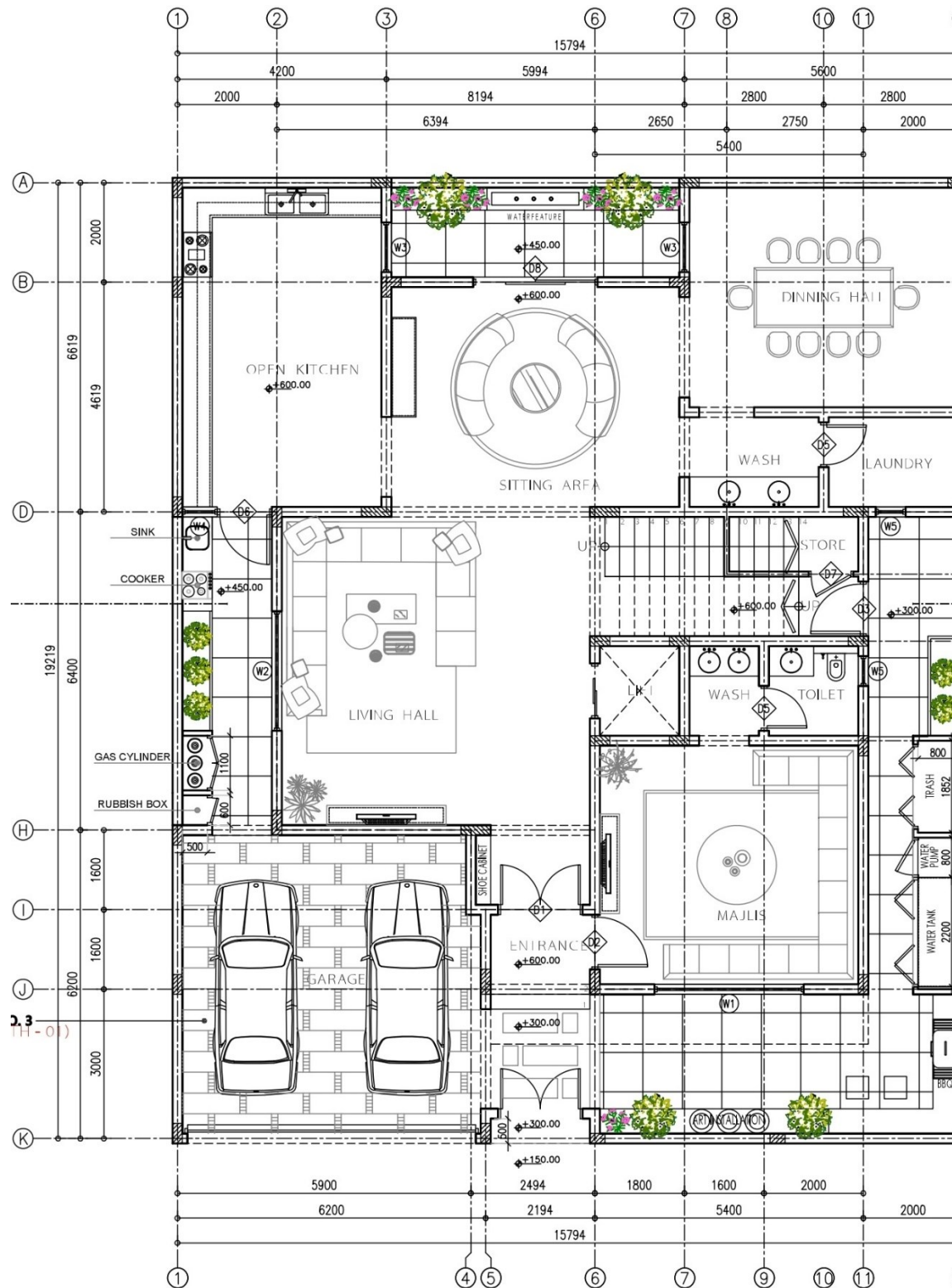
The floor plan illustrates a residential development with the following components:

- Units and Rooms:**
 - Unit 1 (Top Left):** Includes a MAID ROOM, UNDER GROUND WATER TANK, and a curved staircase.
 - Unit 2 (Top Right):** Features two TOILET rooms, a WASH area, and a TV.
 - Unit 3 (Middle Right):** Contains MALE MAJLIS and FEMALE MAJLIS rooms.
 - Unit 4 (Bottom Left):** Includes a DINING area, FAMILY HALL, KITCHEN, and two STORE rooms.
 - Unit 5 (Bottom Right):** Features an OUT KITCHEN and a Foyer.
- Common Areas and Features:**
 - Staircase:** A central curved staircase with 'UP' and 'DOWN' directions.
 - Garage:** An OPEN GARAGE with a car icon, measuring 5600 units.
 - Yard:** A REAR YARD with a car icon, measuring 7400 units.
 - Plot Limits:** Clearly marked on all four sides of the building.
- Dimensions and Orientation:**
 - Horizontal Dimensions:** 2700, 1800, 1000, 2100, 900, 400, 2300, 500, 500, 1300, 2750, 1450, 1120, 2880, 1200, 1900.
 - Vertical Dimensions:** 2100, 1800, 1800, 2200, 1750, 400, 1190, 800, 1000, 400, 800, 1800, 1800, 380, 630, 400, 1600, 2200, 4900, 2700.
 - Orientation:** North arrow pointing towards the top right.

Floor Plan No. 13



Floor Plan No. 14





Floor Plan No. 15

الملخص

في هذه الأطروحة، تم تصميم نموذج باستخدام نموذج التعلم العميق المُدرَّب مسبقًا من أجل أتمتة عملية ترخيص مخططات البناء والتخلص من التدخل البشري في هذه العملية. بعد الانتهاء من دورات التدريب والتحقق من الصحة والاختبار للنموذج، تم ادخال مجموعة من مخططات البناء في النموذج واستخلاص النتائج من حيث اتخاذ القرار سواء كانت مخططات البناء هذه تتوافق مع لوائح بلدية البحرين أم لا، وهل يمكن قبول هذه المخططات وترخيصها أو يجب أن يتم التعديل بناءً على المعايير المحددة مسبقًا. تم الاستعانة بمكتب هندسي معتمد وذلك لحساب أبعاد كل مخطط بشكل يدوي من خلال قياس الطول والعرض لكل مخطط باستخدام ملف أوتوكاد. كما تم حساب الأبعاد من قبل مهندسي المكتب والتحقق منها من قبل رئيسه للتأكد من دقة تلك الأبعاد. وعلى إثره، تم تطوير الخوارزمية لقياس المساحات بالمتر المربع من الاسطح للمخططات البناء. للعثور على الاختلاف الحراري للمبنى لحساب الفقد الحراري للمخطط الطابقي، تم تنزيل بيانات درجة الحرارة الحقيقية والتي تحتوي على أكثر من 1754 محطة ترسل بيانات درجة الحرارة الحية كل ثلاث ساعات. تم حساب قيم كسب الحرارة لـ 15 مخططًا باستخدام البيانات التي تم الحصول عليها مسبقًا في قيم U ومساحات السطح الخارجي واختلاف درجة الحرارة. تم التحقق من نتائج الخوارزمية من خلال مقارنة مخرجات الخوارزمية والحساب اليدوي من قبل المكتب الهندسي وكانت جميع النتائج التي حصلت عليها الخوارزمية دقيقة للغاية عند مقارنتها بالحسابات اليدوية التي تمت من قبل المكتب الهندسي مما جعل عملية ترخيص مخططات البناء آلية وبعيدة عن تدخل العنصر البشري.

جامعة البحرين

كلية الهندسة

قسم الهندسة الميكانيكية



دمج أدوات الذكاء الاصطناعي المتقدمة مع نظام المعلومات
الجغرافية لتعزيز إدارة الطاقة واتخاذ القرار

أطروحة مقدمة كجزء من متطلبات الحصول على درجة
دكتوراه الفلسفة في الإدارة الهندسية

إعداد

عائد خليل شكري شحادة

الرقم الجامعي: 20061469

إشراف

أ.د. إبراهيم عبدالله مطر

أستاذ

جامعة البحرين

مملكة البحرين

مايو، 2024