A Project Report

On

FLOOR LAYOUT PLANNING USING ARTIFICAL INTELLIGENCE TECHNIQUE

Submitted in partial fulfillment of the requirement of
University of Mumbai for the Degree of

Bachelor of Engineering
In
Computer Engineering

Submitted By
Mitali Chavan
Nidhi Nirmal Menon
Ritika Kumar
Shivani Rana

Supervisor
Prof. Varunakshi Bhojane

Department Of Computer Engineering
PILLAI COLLEGE OF ENGINEERING
New Panvel – 410 206
UNIVERSITY OF MUMBAI
Academic Year 2016 – 17

DEPARTMENT OF COMPUTER ENGINEERING
Pillai College of Engineering
New Panvel – 410 206

CERTIFICATE

This is to certify that the requirements for the report entitled **‘Floor Layout Planning using Artificial Intelligence’** have been successfully completed by the following students:

Name	**Roll No.**
Mitali Chavan	B813
Nidhi Nirmal Menon	B835
Shivani Rana	B845
Ritika Kumar	B848

in partial fulfillment of Bachelor of Engineering of Mumbai University in the Department of Computer Engineering, Pillai College of Engineering, New Panvel – 410 206 during the Academic Year 2016 – 2017.

Supervisor
(Prof. Varunakshi Bhojane)

Internal Examiner	**External Examiner**
Head of Department **(Dr. Madhumita Chatterjee)**	**Principal** **(Dr. R.I.K Moorthy)**

Table of Contents

Abstract

In the era of e-commerce while buying furniture online the customers obviously feel the need for visual representation of the arrangement of their furniture. Even when doing interiors of the house it's difficult to just rely on assumptions about best layouts possible and professional help may become quite expensive. It is also seen that the professional consulted may not be good enough to give efficient solutions or may give only one solution that may not cater to the customer's needs. To tackle such issues and give the users their peace of mind and to meet their satisfaction levels we are proposing a system, PeraNoto that aims to meet the customer needs. Currently its scope is such that it works only for one section of the house, the master bedroom. It aims at placing the basic furniture elements in it with accuracy while also providing the user with options of different possible layouts in each run. In this project, we make use of Genetic Algorithm (GA) to make a system that helps designing layouts for a master bedroom. GA is an evolutionary design model i.e. a generic population-based meta-heuristic optimization algorithm**.** Genetic algorithm iteratively follows a set of certain steps to reach its goal. These include usage of random population, generation of chromosomes for each possible solution from the set of inputs, calculation of fitness values after every operation and crossover between chromosomes in each generation until an optimum fitness function is reached. If even after this an optimum solution is not obtained modification in chromosome representation may also be done for better results. This modification implies performing a mutation function on a particular chromosome which is very close to the best solution possible. The only aim of the system is to deliver best possible layout to the user. And for this, PeraNoto will generate different layout designs with four being the maximum number of best layouts to be displayed at a time and one being the minimum. If at all the system is unable to generate a layout the user is prompted to try again thereby making it user friendly. It also interactive as it uses the sense of basic mathematics for checking the validity of the user's input and thus checks for errors and prompts the user. In all, PeraNoto is an attempt in using the field of evolutionary algorithm for development of systems for similar problems.

List of Figures

List of Tables

Chapter 1

INTRODUCTION

1.1 Fundamentals

In architecture and building engineering, a floor plan is a drawing to scale, showing a view from above, of the relationships between rooms, spaces and other physical features at one level of a structure. Floor plans may include notes for construction to specify finishes, construction methods, or symbols for electrical items. The floor planning can be classified into two categories either an equal area layout problem or an unequal area layout problem. The equal area layout problem is to determine how to assign a set of distinct facilities, to a set of distinct locations, so that each facility can be assigned to a single location called as one-to-one assignment problem. The unequal area layout problem is to regulate the allocation of all facilities within a block plan (or available area).

Optimizing furniture arrangement into a realistic and functional indoor configuration involves considerable complexity, taking into account various interacting factors, such as pair wise furniture relationships, spatial relationships with respect to the room, and other human factors. An effective representation that captures the necessary spatial relationships is needed. Some of the attributes to be considered while deciding furniture placement are - Bounding surfaces, Centre and orientation, Accessible space, Viewing frustum, Hierarchical relationships between different furniture objects.

Problems that deal with the identification of the optimum location of a number of objects in a predefined area are called Layout planning problems. In many frameworks the solution of layout planning problems has been a frequent issue. For example, construction site management, building design, the location of facilities in urban planning and the design of manufacturing plants.

When doing interiors of the house it's difficult to just rely on assumptions about best layouts possible and professional help may become quite expensive. Relevant floor plans can be generated before planning the layout of furniture pieces. Although interior design involves creative solutions that can be fairly subjective, a set of quantifiable design criteria has long been accepted in the industry. Specifically, such criteria determine whether the design is functional and suitable for human inhabitants. These human factors have been noted to be the “prime determinants” of interior design, emphasizing that while average measurements should be used, flexibility should be exercised to satisfy specific user needs. In optimization terms, such guidelines can be interpreted as soft constraints. The importance of accessibility in furniture placement is noted, which is a common consideration in decorating rooms with different purposes.

Considering the spatial relationships between different furniture objects, our goal is to integrate this information into an optimization framework with a properly defined cost function quantifying the quality of the furniture arrangement. Given an arbitrary room layout populated by furniture objects, the synthesized arrangement should be useful for virtual environment modelling in games and movies, interior design software, and other applications. The search space of our problem is highly complex as objects are interdependent in the optimization process. The furniture positions and orientations depend on numerous factors, such as whether the object should be visible or accessible. It is very difficult to have a global optimization scheme or a closed-form solution that yields a unique optimum.

In this project, we make use of Genetic Algorithm (GA) which is an Artificial Intelligence technique to display various optimal arrangements of furniture. The basic idea behind using GA is developing an evolutionary design model. This is done by generating chromosomes for each possible solution and then performing a crossover between them in each generation until an optimum fitness function is reached. Modification in chromosome representation may also be done for better results. GA uses the feedback from the evaluation process to select the fitter designs, generating new designs through the recombination of parts of the selected designs.

1.2 Objectives

The objective of this work is as follows:

1. To establish that Genetic Algorithm is a suitable technique for designing layout.
2. To discuss the mechanism of Genetic algorithm and to compare itwith other placement algorithms and site the scenarios when it has proved to be more efficient.
3. To understand method of feature extraction for furniture placement system and various constraints that should not be violated while designing the arrangement of furniture.
4. To identify evaluation metrics used for performance analysis of proposed system.

1.3 Scope

The proposed system will generate multiple layout designs for the furniture keeping inconsideration the structure of a master bedroom. Its dimensions will be given as input by the user. The master bedroom will have a window and two doors. The size of the doors and the window will be predefined and not subject to any changes but their position would be given by the user. The furniture elements are Wardrobe, Bed and Table. The size of these elements will also be fixed. The input given will be taken from the user in text format via a GUI and the output will be multiple possible arrangements of the furniture elements represented in 2D format.

1.4 Organization of the Report

The report is organized as follows:

The introduction is given in Chapter 1.It describes the fundamental terms used in this project. It motivates to study and understand the different techniques used in this work. This chapter also presents the outline of the objective of the report.

The Chapter 2 describes the review of the relevant various techniques in the literature systems. It describes the pros and cons of each technique.

The Chapter 3 presents the Theory and proposed work. It describes the major approaches used in this work.

The societal and technical applications are mentioned in Chapter 4.

Chapter 5 consists of the summary of the project.

Chapter 2

Literature Survey

2.1 Overview

This chapter consists of the literature survey that we have conducted on the various existing systems on floor planning. The following chapter explains the systems that we surveyed.

2.2 Literature Review

Jun H. Jo and John S. Gero [1] in their paper ‘Space Layout Planning Using an Evolutionary Approach’ describe a design method based on constructing a genetic/evolutionary design model whose idea is borrowed from natural genetics. They touch the major issues while using genetic designing model like the formulation of knowledge representation set and the genetic communication. A set of design elements, is in the form of design rules and needs to be interpreted into the language of the genetic search system. The principle for the schema translation is defined as components of a design rule schema which are distinct and need to be manipulated by the genetic search process are active and translated to those of the design gene schema. The other components are inactive and not translated but kept in the interpretation knowledge. To prevent an activity from being used more than once, they propose a reordering function that makes all activities of a genotype unique. As an example the model was implemented for an office layout problem, where the problem configurations and the evaluation criteria were drawn from the literature. The results show the usefulness of this design process model. On the basis of the advantages of genetic evolutionary design process and the results of the implementation, they concluded that the coupling of an evolutionary search technique with a design process can produce very good results, especially for large-scale problems which are at present computationally difficult. A comparison is also done between processes in Liggett’s Space Layout System and the EDGE system. Liggett uses a probability scheme to decide the next unit to assign. The EDGE system does not require any special heuristics to generate and evolve solutions and evaluates a whole solution instead of its component details.

The paper 'Generic Chromosome Representation and Evaluation for Genetic Algorithms' by Kristian Guillaumier [2] explores the role of chromosomes in genetic algorithms. The breakdown of the basic genetic algorithm is given by the author. According to the author, this algorithm may be implemented in any high-level programming language. However, in conventional implementations, most parameters, the fitness function, chromosome representation, and genetic operators are usually hard-coded. If the nature of the problem varies slightly or critical parameters change, the original code is revised – sometimes substantially- as per the need of the problem. Moreover, that the user may not be even computer literate and not prepared to deal with issues such as algorithm design, programming and debugging, is another point to be noted. The author illustrates this with a very simple time-tabling problem where each of 5 lectures has to be assigned one of 3 timeslots and where no lecture given by the same tutor may occur at the same time. The chromosomes are represented using a direct representation scheme. Once encoding is done, basic single point crossover and mutation is applied. The crossover fitness is decided as per the problem. In a time-tabling problem, it becomes no. of non-clashing schedules and if no schedules clash then the penalty value is returned as '0'. According to the fitness value the generations of chromosomes are checked iteratively. The generation that is closest to the fitness value or the generation at which the limit is achieved is returned as the optimal solution. This technique has been successfully implemented as a high-level modeling language called OPML (Optimization Problem Modeling Language) together with a general-purpose Genetic Algorithm-based runtime.

MohdSaberi Mohammad, Safaai Deris, Safie Mat Yatim, Muhammad Razib Othman[3] in their paper 'Feature selection method using genetic algorithm for the classification of small and high dimension data', propose an efficient feature selection method that finds and selects informative features from small or high dimension data thereby maximizing the efficiency of the classification process. They apply genetic algorithm to search out and identify the potential informative features combinations for classification and then use the classification accuracy from the support vector machine classifier to determine the fitness in genetic algorithm. The first proposed approach is called GASVM- a combination of GA with SVM without any modification in chromosome representation. The second proposed approach is that of New-GASVM which will modify the model of chromosome representation. The two main components in this approach are GA for features subset selection and SVM as classifier. On implementation along with the current best classifiers, the New-GASVM is the best performer and the GASVM turns out as the second best.

The shortcoming in GASVM is that it may not be able to support high dimension data whereas New-GASVM can.

Ryota Tachikawa and Yuko Osana [4] in their paper, ‘Office Layout Support System using Genetic Algorithm– Generation of Room Arrangement Plans for Polygonal Space’ propose an office layout support system using genetic algorithm. The proposed system has two phases: (1) generation of room arrangement plans and (2) generation of layout plans for workspace. The working is such that some conditions on rooms and furniture are given by a user and some room arrangement plans which satisfy the conditions are generated by Genetic Algorithm (GA). After one of the generated room arrangement plans is selected by a user, then some layout plans for workspace which satisfy the conditions are generated. The room arrangement plan is expressed in the form of a gene that consists of 4 parts: (1) rule for room size decision, (2) relation between rooms, (3) arrangement method in each wall and (4) angle of rooms in each wall. In order to leave better genes to the next generation, the degree of fitness is calculated, and genes are chosen according to this degree of their fitness. A combined GA which is based on the Adaptive Genetic Algorithm (AGA) and the GA with Search Area Adaption (GSA) is employed in this system in order to improve the convergence speed and the diversity of layout plans. For the generation of layout plans for workspace, layout plans are expressed in the form of genes in a similar fashion and further evaluation and computer experiments are done that included transition of fitness and variation of generated plans &execution time of proposed system compared with normal genetic algorithm, AGA and GSA and it is found that the proposed system (combined genetic algorithm) gives the best results.

Eugénio Rodrigues, Adélio Rodrigues Gaspar and Álvaro Gomes [5] in their paper ‘An evolutionary strategy enhanced with a local search technique for the space allocation problem in architecture, Part 1: Methodology’ propose an enhanced hybrid evolutionary computation scheme that couples an Evolutionary Strategy (ES) with a Stochastic Hill Climbing (SHC) technique to generate a set of floor plans to be used in the early design stages of architectural practice. The algorithm used presents a hybrid behavior by combining ES with an SHC technique. It takes advantage of both the global search capabilities of the ES and the local search characteristics of the SHC technique. This algorithm is named Evolutionary Program for Space Allocation Problem (EPSAP) and uses adaptive operators to perform the geometric transformations of the rooms, their walls and connections, and openings according to previously stored information. It begins with initiating the first stage (ES search) by randomly generating the population. Followed by initiating SHC search, here individuals are evaluated and ranked according to the objective function. Its main purpose is to be able to generate different candidate floor plans to be used in the early stages. The ES operators are carried out in each ES generation. These operators act transversely to different floor plan elements, for instance the alignment of walls of different spaces. These alignment operators act on the individual to remove incoherencies and make the floor plan emerge as a final design. The operators are invoked in the order: Wall Alignment Operator and Void Remover Operator. The former operator, the Wall Alignment Operator, works by making two lists, one with all vertices x-coordinates and the other with y-coordinates. If the ordinates are within the range of predefined values, they are substituted by their average.

Brian L.Thomas, Aaron S. Crandall and Diane J. Cook's [6] work ‘A Genetic Algorithm approach to motion sensor placement in smart environments’, explores techniques to identify "reasonable" sensor placement plans within smart home context. The goal here is to decide the strategic location of sensors over a given restricted area such as a kitchen or a living room. It is relevant to our research as it discusses the various algorithms to generate different plans for object placements with minimum user input. The paper compares various placement algorithms for sensor layouts such as 1) Human Intuition-Based 2) Monte Carlo-Based 3) Two-Dimensional Uniform Placement (Grid) 4) Hill Climbing and 5) Genetic Algorithm (GA) to prove that GA gives most accurate results. The results of three baseline algorithms, human, Monte Carlo and grid, exhibited notably different approaches to sensor placement than the GA and Hill Climbing approaches. The GA and Hill Climbing algorithm on other hand focus on optimal arrangement of objects around each other based on some predefined conditions (in this case sensors were clustered in areas having maximum activity). This is because Hill Climbing algorithm has a significant branching factor in the search space with each iteration, to the order of *(*Number of locations − Number of objects*)* * (All possible Combinations). It takes weeks of running to generate these results, as compared to much faster evaluations with the other algorithms. The paper thus concluded that the GA algorithm’s layouts are more efficient to generate than Hill Climbing, in addition to being significantly more accurate than the baseline algorithms tested.

Lap-Fai Yu, Sai-Kit Yeung, Chi-Keung, Demetri Terzopoulos, Tony F. Chan, and Stanley J. Osher [7] in their work ‘Make it Home: Automatic Optimization of Furniture Arrangement’, conducted a perceptual study to demonstrate that their system can synthesize multiple realistic furniture arrangements. The paper efficiently discusses Furniture Relationship Extraction, Furniture Arrangement Optimization and various Constraints that should not be violated. Furniture Relationship Extraction has two parts Object Representation and Learning Prior Relationships. Object Representation denotes various object properties like bounding surfaces - top, bottom, back surfaces (surfaces closest to any wall) and non-back surfaces, Centre and Orientation - denoted by (p,□) describing location and angle of object with respect to nearest wall, accessible space - which is used to measure how deep other objects can penetrate into the space during optimization etc. Learning of Prior Relationship involves Spatial relationships - The key prior relationships are the distance of an object to its nearest wall d and its relative orientation to the wall □, Hierarchical relationships:Given two objects A and B, object A is defined as the parent of B (and B as the child of A) if A is supporting B by a certain surface and Pair wiserelationships - certain objects, such as a television and a sofa or a dining table and chairs, interact with each other in pairs subject to pair wise orientation and distance constraints. The search space of our problem is highly complex as objects are interdependent in the optimization process. Thus we resort to stochastic optimization methods, specifically, simulated annealing. The Constraints mentioned are Accessibility - A furniture object must be accessible in order to be functional. Visibility - Some objects, such as a television or a painting, impose strict requirements on the visibility of their frontal surfaces, since their fundamental functionality is compromised if their fronts are blocked by another object. Pathway Connecting Doors - The placement of furniture objects such that they block doors should obviously be inhibited.

2.3 Summary of Literature Survey

Table 2.3 Literature survey summary

Sr. No	Name of the paper	Year	Research papers details	Remarks
1	Generic Chromosome Representation and Evaluation for Genetic Algorithms	2002	The author explores the role of chromosomes in genetic algorithms. The author explains the breakdown of the basic genetic algorithm.	The proposed system has been successfully implemented as a high-level modeling language called OPML (Optimization Problem Modeling Language) together with a general-purpose Genetic Algorithm-based runtime.
2	Feature selection method using genetic algorithm for the classification of small and high dimension data	2004	The authors propose an efficient feature selection method that finds and selects informative features from small or high dimension data thereby maximizing the efficiency of the classification process using genetic algorithm and support vector machine.	They proposed two approaches, GASVM- a combination of GA with SVM without any modification in chromosome representation and New-GASVM which will modify the model of chromosome representation. On comparing the two, New-GASVM is a better system.
3	Space Layout Planning using an Evolutionary Approach	2006	The authors describe a design method based on constructing a genetic/evolutionary design model whose idea is borrowed from natural genetics.	The authors propose a reordering function to get unique genotype. Also, comparison is done between processes in Liggett's Space Layout System and the EDGE system.
4	Office Layout Support System using Genetic Algorithm	2010	The author has proposed the system to work using combined genetic algorithm based on GSA and AGA.	The proposed system is compared with normal genetic algorithm, AGA and GSA and it is found that the proposed system (combined genetic algorithm) gives the

				best results.
5	Make it Home: Automatic Optimization of Furniture Arrangement	2011	The authors conducted a study to show that their system could synthesize multiple furniture arrangement. It discusses Furniture Relationship Extraction, Furniture Arrangement Optimization and various Constraints that should not be violated.	The authors resort to stochastic optimization methods, specifically, simulated annealing. The constraints taken into account for the proposed system are Accessibility, Visibility and Pathway Connecting Doors.
6	An evolutionary strategy enhanced with a local search technique for the space allocation problem in architecture	2013	The proposed system couples an Evolutionary Strategy (ES) with a Stochastic Hill Climbing (SHC) technique i.e. uses Evolutionary Program for Space Allocation Problem (EPSAP).	It is able to generate different candidate floor plans to be used in the early stages and then uses operators like Wall Alignment Operator and Void Remover Operator for the final output.
7	A Genetic Algorithm approach to motion sensor placement in smart environments	2015	This paper deals with deciding the strategic location of sensors over a given restricted area. A comparison is made between efficient algorithms.	The baseline algorithms concentrate on covering more of the space rather than intelligent arrangement of the objects. The GA and Hill Climbing algorithm focus on optimal arrangement of objects around each other based on some predefined conditions. Although the results of Hill Climbing algorithm were very similar to that of GA they are quite expensive to obtain. The result of comparison is that genetic algorithm gives the best results.

2.4 Inference of literature survey

- On referring the 7 papers is we have concluded that genetic algorithm is the best approach for our system, though it can be improved by clubbing it with another algorithm like Hill Climbing or Support Vector Machine.
- On the basis of the advantages of genetic evolutionary design process and the results of the implementation, it can be concluded that the coupling of an evolutionary search technique with a design process can produce very good results, especially for large-scale problems which are at present computationally difficult.
- There are two approaches regarding chromosome representation, with modification or without modification. It is concluded that with chromosome representation modification (as in new-GASVM) better results are obtained as compared to without modification (as in the case of GASVM) as GASVM may not be able to support high dimension data whereas New-GASVM can.
- The arrangement plan can expressed in the form of a gene and in order to leave better genes to the next generation, their degree of fitness is calculated, and genes are chosen accordingly. Thus using combined GA (based on AGAand GSA) can be employed to improve the convergence speed and the diversity of layout plans.
- For a highly complexsearch space out of the different optimization techniques stochastic optimization is supposed to give a better result.

Chapter 3

PeraNoto

3.1 Overview

In Javanese “Furniture” is known as “Perabot” and “Arrangement” is known as “Noto”. Thus combining the two, the proposed system is named “PeraNoto”. In this chapter we discuss about the working of the existing system and PeraNoto, the proposed system.

3.1.1 Existing System

The existing systems for floor layout planning work on the concept of drag and drop mechanism. The user is shown a layout of the room or is asked to choose a layout from the options provided which seem similar to the layout of the room they wish to plan and then from the set of objects provided they are expected to use the drag and drop facility. There is generally a set of symbols, realistic images or clip art images provided for the different kinds of furniture elements and the user is expected to only drag them on the room layout they find similar to their room and drop it on the location they wish to place it. The whole planning and layout produced can be in 2D or 3D format. Though this advantage of a 3D view of the room is given to the user it can also be seen that the existing system merely allows him to visualize how his thought-of room layout would look like or how he could think of a little alterations to his plan to result in a different layout. This system also has room for human errors of planning for instance, not leaving enough space for the door of the room to open and placing a table in front of it. Even an error of a centimeter is enough to make a layout inefficient.

Drawbacks:

- Though the user is able to see how exactly his layout will look like there are chances of him missing out more efficient layouts that the proposed system can provide.
- A user might always not find a room layout similar to his room for selecting and placing the furniture elements and might have to compromise with the output.
- The furniture elements used may not be scalable to the room dimensions when placed and thus, may create difficulty for the user.

3.1.2 PeraNoto

Here, we would discuss the system architecture of the proposed system. Input to the system is the position of doors and window and the dimensions of the room, in text format through a graphical user interface. The first unit is the input unit wherein the input as text format is first cleaned and then represented as chromosomes. The next unit is the processing unit where Genetic algorithm with the required constraints is applied to the chromosome and a total of four solutions are chosen. The final unit is the output unit wherein the selected solution chromosomes are represented in a user-friendly format. Thus the output is in 2-dimensional format where the user gets a layout of the furniture in the top view of the room.

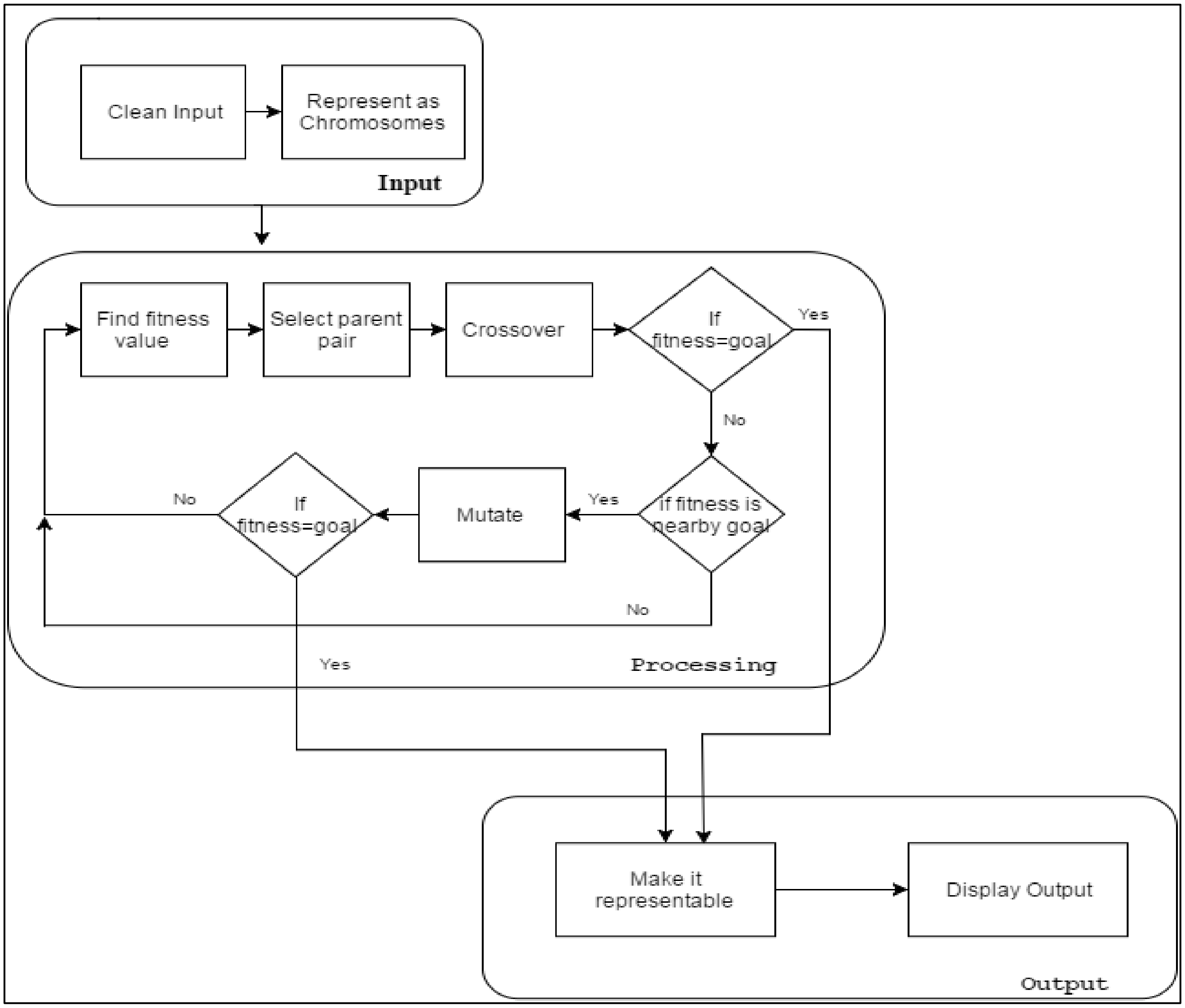


Fig. 3.1: System Architecture of PeraNoto

The overview of the modules in the architecture shown above is given below:

- **Clean input:** In this module, we clean the input i.e., we take the values from the different fields in the graphical user interface. The input is in the form of positions, dimensions and other constraints by the user, along with pre-defined constraints. This module selects only the relevant information and passes it to the next module for further processing.
- **Represent as Chromosomes:** In this module, the relevant information is to be represented as chromosomes. A chromosome (also sometimes called a **genotype**) is a set of parameters which define a proposed solution to the problem that the genetic algorithm is trying to solve. The set of all solutions is known as the *population*. The chromosome is often represented as a binary string; although a wide variety of other data structures are also used. As of now, we are taking the measurements in foot and converting them into inches for computation. The chromosome has 3 major divisions depicting each furniture element and 9 subdivisions in all with the x and y coordinates of the left bottom corner of the furniture element and its orientation. This chromosome can be represented as the following diagram
- **Find fitness value:** The fitness value of the given input is calculated. Fitness value is the method of figuring out how close a given design solution is to achieving the set aims or the goal state. In our case, the fitness value is the number of non-colliding furniture. This module compares the chromosome's fitness value for comparison with the goal value. The fitness values of all furniture are computed separately and then summed up for comparison with the goal fitness value.
- **Find parent pair:** The parent pair that is the preceding generation of the particular chromosome is selected after the calculation of fitness value. After the parent pair is selected, this input is then passed onto the next module. The system uses the steady state selection algorithm. This is the fitness proportionate selection where in the individual is selected on the basis of fitness. The probability of an individual to be selected as a parent increases with the fitness of the individual greater or less than its competitor's fitness.
- **Crossover:** Crossover is a genetic operator used to vary the programming of a chromosome or chromosomes from one generation to the next. It is the process of taking more than one parent solutions and then producing a child solution from them. The system

uses the two point crossover technique. After the crossover is done, the child solution is given to the next block.

- **Decision of fitness and goal state equality:** The fitness value of the child solution that is received from the crossover is compared with the predefined goal state. If the goal state is achieved then the child solution is given to the output module. Else, it is passed on to the next block.
- **Decision, whether fitness is nearby to goal state:** The fitness value of the solution is checked to see if it is nearby to the goal state. If it is, it is sent to the 'Mutate' block for further processing. Else, it is sent in a loop, back to the calculation of the fitness value.
- **Decision of fitness and goal state equality:** The fitness value of the child solution that is received from the crossover is compared with the predefined goal state. If the goal state is achieved then the child solution is given to the output module. Else, it is passed on to the next block.
- **Make it representable:** The solution is in chromosome form, which is not understandable by the user. Hence, this module converts it into a form that is more intelligible.
- **Display Output:** The output is displayed to the user in 2D format with the position of the furniture indicated by boxes and labels.

3.2 Implementation Details

The implementation of the paper is carried out using genetic algorithm and applying the required constraints to it. In this section we would explain the implementation by explaining the detailed methodology used with flowcharts and diagrams.

3.2.1 Methodology

The proposed system follows all the stages of Genetic algorithm as shown in the flow chart below.

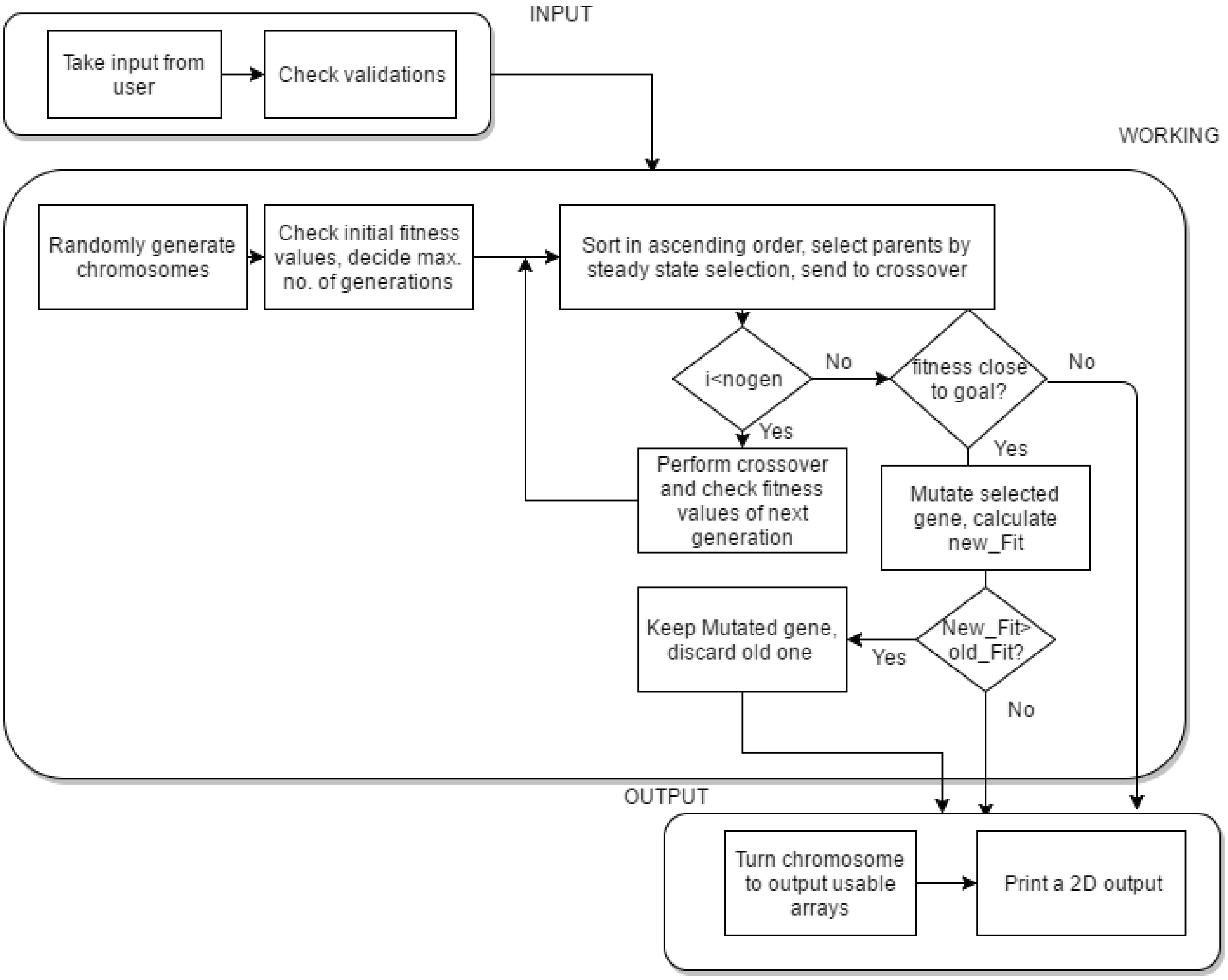
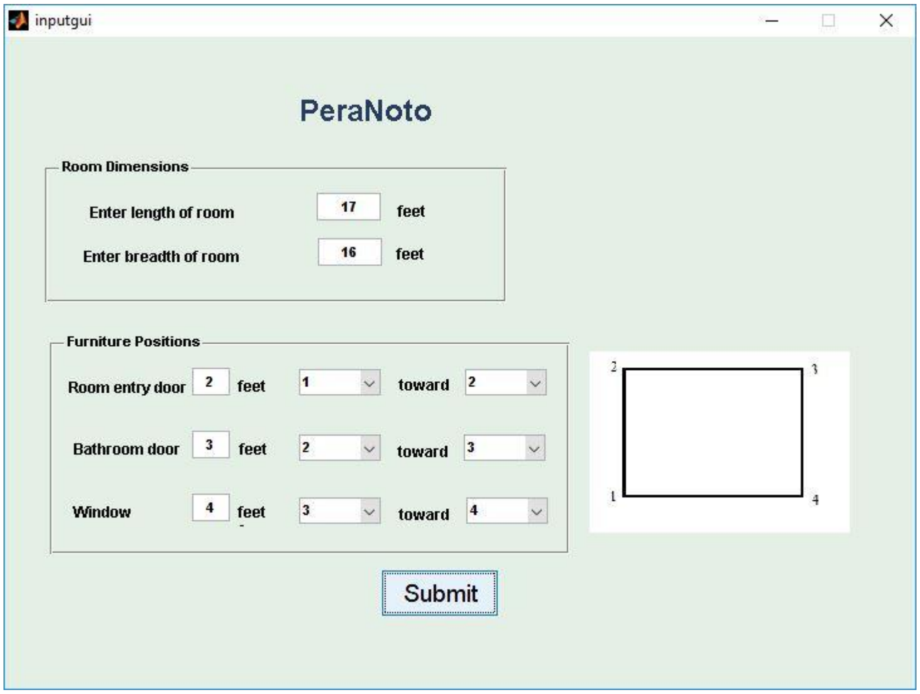

INPUT
Take input from user
Check validations
WORKING
Randomly generate chromosomes
Check initial fitness values, decide max. no. of generations
Sort in ascending order, select parents by steady state selection, send to crossover
i<nogen
No
Yes
fitness close to goal?
No
Yes
Perform crossover and check fitness values of next generation
Mutate selected gene, calculate new_Fit
Keep Mutated gene, discard old one
New_Fit> old_Fit?
Yes
No
OUTPUT
Turn chromosome to output usable arrays
Print a 2D output

Fig 3.2: Methodology Flowchart

The input is taken from the user through the GUI where the user enters length, breadth and the distance of the doors and window from one point towards another point on a wall. The GUI has a rectangle displayed on the right which number the corner points to help the user measure the distances correctly as shown below. The construction standards are according to the Indian Standards given by the Civil Engineering Department of Bureau of Indian standards.



inputgui

PeraNoto

Room Dimensions

Enter length of room 17 feet

Enter breadth of room 16 feet

Furniture Positions

Room entry door 2 feet 1 toward 2

Bathroom door 3 feet 2 toward 3

Window 4 feet 3 toward 4

Fig 3.3: GUI

The system then generates a randomized population of chromosomes. These chromosomes contain information about the coordinates of left bottom corner and also their alignment i.e. horizontal or vertical.

X_B	Y_B	A	X_T	Y_T	A	X_W	Y_W	A
1	2	3	4	5	6	7	8	9

Fig 3.4: Chromosome Structure

The above figure gives representation of a single chromosome. Each furniture object has a designated sub-range of the array to code information specifically about it. The designated ranges are as follows

- sub- range (1:3) represents Bed ; X_B and Y_B are coordinates of the bottom left corner of the Bed
- sub-range (4:6) represents Table; X_T and Y_T are coordinates of the bottom left corner of the Table
- sub-range (7:9) represents Wardrobe; X_W and Y_W are coordinates of bottom left corner of the Wardrobe

Each of the positions 3, 6 and 9 represent the alignment for the furniture represented by the respective sub-ranges to which these positions belong. These positions are set to 1 is alignment is horizontal and 2 if the alignment is vertical. The remaining two positions in each of the sub ranges give the coordinates of the bottom left corner of the furniture object that they represent. The coordinates are generated randomly and the maximum upper limit of these randomly generated numbers is the maximum value among length and breadth of the room whichever is greater. The alignment of these furniture elements; whether 1 or 2 is also random. The next step is to calculate fitness value of each of the chromosome in the population.

The fitness value of each chromosome is then calculated based on various constraints such as - the furniture objects do not overlap, they do not exceed the room dimensions, arrangement of furniture nearer to the walls is favoured, no furniture is in front of the two doors and wardrobe should never be in front of the window.

In addition to the score there are also flags included to notify certain conditions so that occurrences of false positives where a chromosome shows high fitness value score even after not fulfilling some basic conditions are identified. The scores assigned for each object were decided on the basis of trial and error method by experimenting with various random values and are subject to change as per the needs of the application. The fitness scores assigned accordingly are stated below.

Constraints	Wardrobe				Bed				Table			
	Min Score	Max Score	Flags		Min Score	Max Score	Flags		Min Score	Max Score	Flags	
			Yes	No			Yes	No			Yes	No
Overlap with Bed	0	10	1	0	-	-	-	-	-	-	-	-
Overlap with Table	0	10	1	0	0	10	1	0	-	-	-	-
Within room dimensions	0	10	1	0	0	10	1	0	0	10	1	0
In front of room door	0	10	1	0	0	10	1	0	0	10	1	0
In front of washroom door	0	10	1	0	0	10	1	0	0	10	1	0
In front of window	0	10	1	0	5	10	-	-	7	10	-	-
Near to any walls	5	10	-	-	5	10	-	-	5	10	-	-
Total Max/Min Score	5	70			10	60			12	50		

Table 3.1: Fitness Score

As evident from the table individual fitness of each object is calculated based on certain parameters.

- Fitness score of Wardrobe is calculated out of 70
- Fitness score of Bed is calculated out of 60
- Fitness score of Table is calculated out of 50

Maximum Fitness Score of a Chromosome =∑ Maximum Fitness score of each Furniture object
= Maximum score of (Wardrobe + Bed + Table)
= 70+60+50
= 180

Thus, Total Fitness score of a chromosome is always calculated out of 180.

Next we discuss flags; the flags give information about occurrence of conditions such as overlap, furniture exceeding room dimensions, furniture in front of any of the doors and wardrobe in front of the window. If a condition occurs then the flag is set to 1 otherwise set to 0. These conditions are checked for each object and then finally the flags for a chromosome are set by performing logical OR operation between the values of flags obtained for each object.

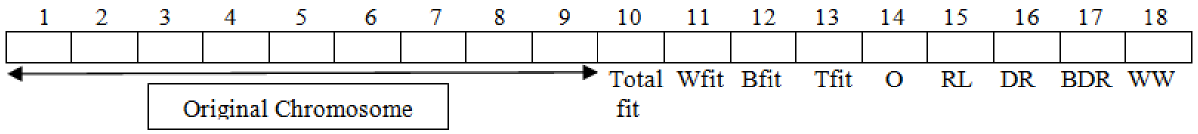


Fig 3.5: Chromosome after Fitness Evaluation

The above image represents the structure of the chromosome after its fitness has been calculated and related values are appended at the end of the chromosome. The flags are represented at positions 14,15,16,17 and 18.

Flag O	-	Flag is set to 1 if Overlap condition occurs
Flag RL	-	Flag is set to 1 if furniture objects exceeds room dimensions
Flag DR	-	Flag is set to 1 if furniture objects are in front of room entry door
Flag BDR	-	Flag is set to 1 if furniture objects are in front of bathroom door
Flag WW	-	Flag is set to 1 if Wardrobe is in front of window

To decide whether these flags should be set to 0 or 1 first the above flag conditions are checked for each furniture object and accordingly local flags are set to 1 or 0 then logical OR operation is performed on these local flags to set the chromosome flags O, RL, DR, BDR thus these flags will be set to 1 even if anyone furniture object breaches any constraint. The last flag WW only displays exclusive information about Wardrobe and hence does not include any other operations.

The positions 10, 11, 12 and 13 represent Total fitness score (Totalfit), Wardrobe fitness score (Wfit), Bed fitness score (Bfit) and Table fitness score (Tfit) respectively.

- Totalfit - Fitness value scored out of 180 for the entire chromosome
- Wfit - Fitness value scored out of 70 for the arrangement of Wardrobe
- Bfit - Fitness value scored out of 60 for the arrangement of Bed
- Tfit - Fitness value scored out of 50 for the arrangement of Table

Totalfit = $\sum$ Fitness value scored for each furniture object

= Wfit + Bfit + Tfit (Equation 1)

The next step after evaluating fitness score is to select parent chromosomes for crossover. The selection operator used here is Steady State selection. This selection operator sets a minimum threshold fitness value and all the chromosomes with a higher fitness value than the threshold is allowed to participate in the crossover. For the system, not only do the chromosomes have to possess a Totalfit value higher than the threshold but also all the above mentioned flags should be set to 0 for them to qualify for the crossover. This additional measure helps to be assured that only good chromosomes are passed on to create next generation.

The crossover performed here is Two-Point crossover. The chromosomes are sorted in ascending order according to their Tfit value and then crossover is performed between a chromosome and its immediately next chromosome. Since, crossover should be performed only within 3 furniture objects and the internal coordinates or alignment of each object should not be disturbed sub-ranges for Bed(B), Table(T), and Wardrobe(W) are each treated as separate blocks

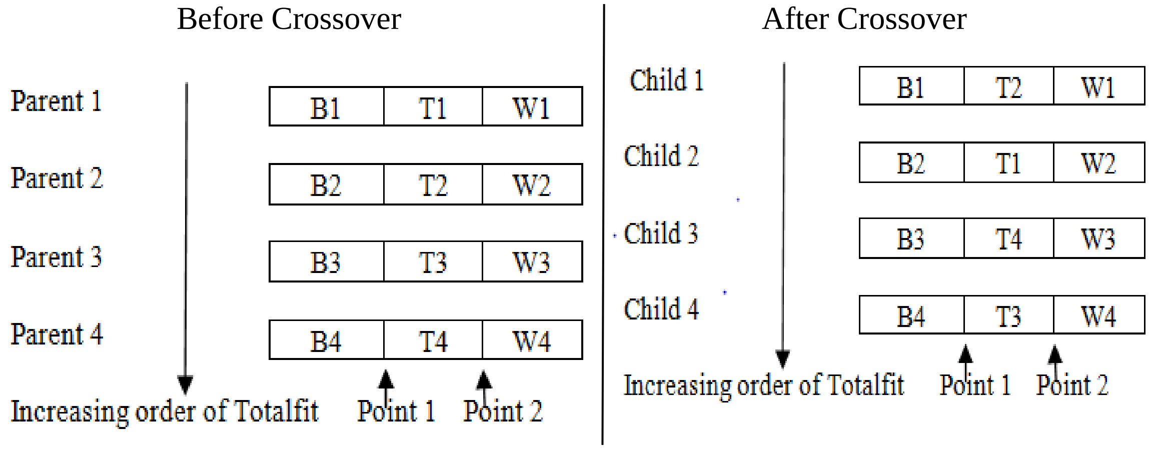


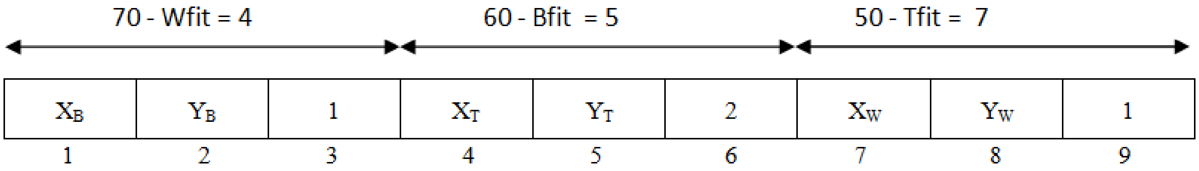
Fig 3.6: Crossover

As seen in the above images, Two-Point crossover is done by replacing only the Table blocks between the two simultaneous chromosomes. The fitness value of these Child chromosomes is again calculated similarly and then selected ones are again sent for crossover, this process continues over a few generations. The child chromosomes of the last generation are expected to have good fitness values the results generated from which can be displayed.

If a chromosome has fitness value very near to the goal fitness value then that chromosome can be mutated to reach the goal value. The probability of any chromosome undergoing mutation is very less and almost negligible. In the system if any chromosome undergoes mutation then only the alignment of any of the object is changed to reach the goal value. To decide which furniture object's alignment will change the difference between its goal fitness value and current fitness value is calculated. The alignment of object with maximum difference is then changed.

Object undergoing change in alignment = Object with max (difference (goal fitness value, current fitness value

= Max ((70-Wfit), (60-Bfit), (50-Tfit)) (Equation 2)



X_B	Y_B	1	X_T	Y_T	2	X_W	Y_W	1
1	2	3	4	5	6	7	8	9

Fig 3.7(a): Before Mutation

In the above image since the difference between 50 - Tfit = 7 is maximum thus the alignment of Table is changed during Mutation. Thus after mutation the chromosome changes to the below image

X_B	Y_B	1	X_T	Y_T	2	.X_W	Y_W	2
1	2	3	4	5	6	7	8	9

Fig 3.7(b): After Mutation

The mutation does not guarantee that the resulting chromosome will have a fitness value greater than its previous value. Hence, the fitness value is again calculated. If the new fitness value is greater than the previous value only then the mutated chromosome is included in the result otherwise the previous un-mutated chromosome is part of the result.

3.1 Activity Diagram

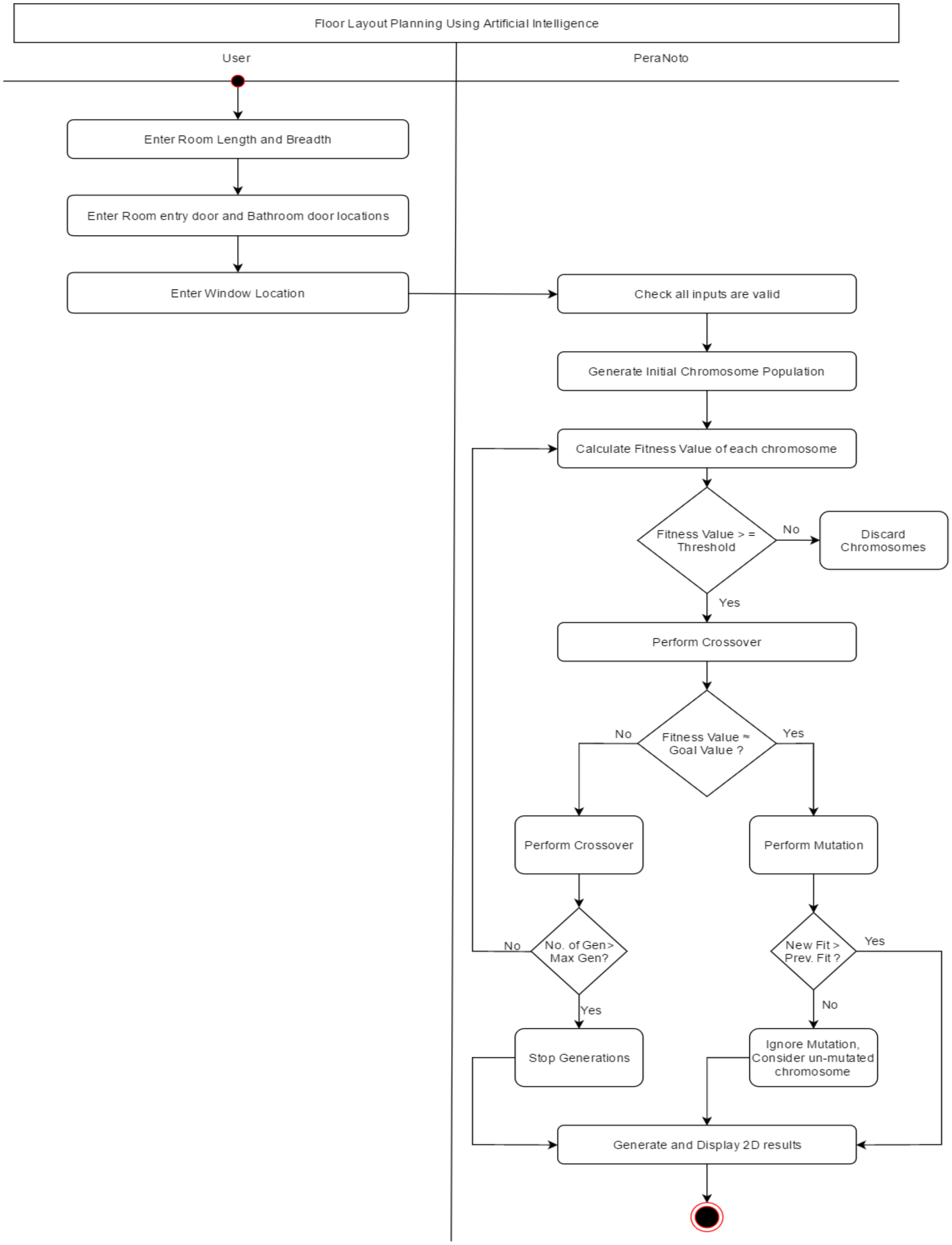

Floor Layout Planning Using Artificial Intelligence
User
PeraNoto
Enter Room Length and Breadth
Enter Room entry door and Bathroom door locations
Enter Window Location
Check all inputs are valid
Generate Initial Chromosome Population
Calculate Fitness Value of each chromosome
Fitness Value > = Threshold
No
Discard Chromosomes
Yes
Perform Crossover
No
Fitness Value ≈ Goal Value ?
Yes
Perform Crossover
Perform Mutation
No
No. of Gen > Max Gen?
Yes
New Fit > Prev. Fit ?
Yes
No
Stop Generations
Ignore Mutation, Consider un-mutated chromosome
Generate and Display 2D results

Fig 3.8: Activity Diagram

3.2 Hardware and Software Specifications

Hardware (For the programmer):

Processor: Any Intel or AMD x86 processor supporting SSE2 instruction set (Windows/Linux), All Intel based Macs with Intel Core 2 or later (Macintosh)

RAM: 2048 MB

Storage: 1 GB

Software (For the programmer):

Operating System: Windows/Linux/Macintosh

For the user:

Since an .exe file of the system has been created for distribution, any hardware and software configurations can be used.

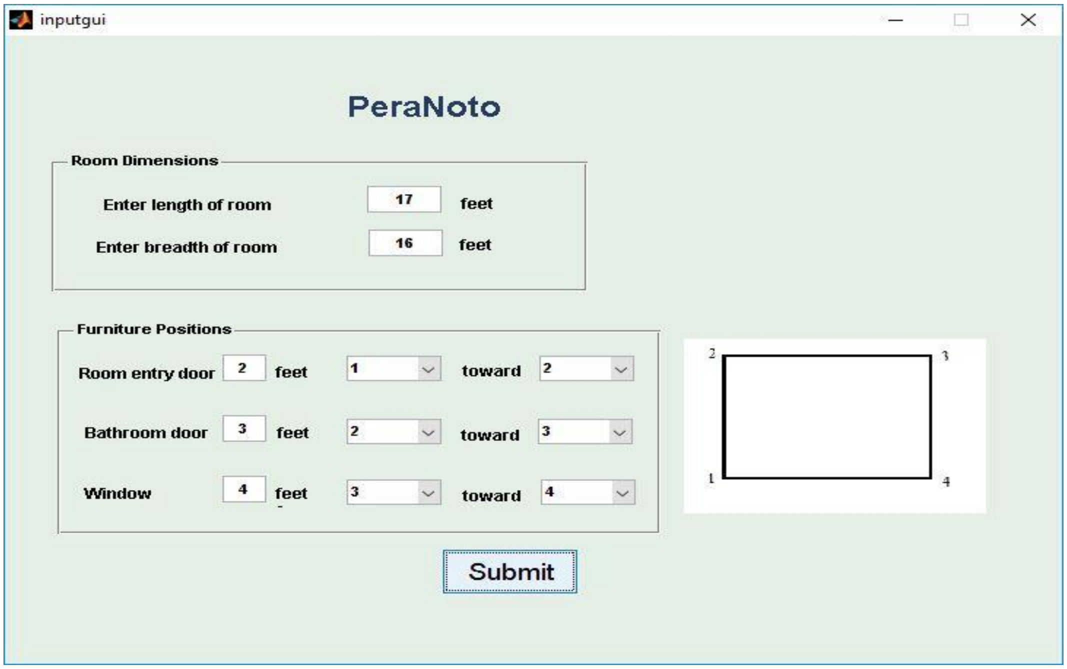
Chapter 4

Result and Discussion

4.1 Input and Output Screenshots

The following images are the screenshots of the inputs and their respective outputs:

1. For a room of dimensions 17x16:



inputgui

PeraNoto

Room Dimensions

Enter length of room 17 feet

Enter breadth of room 16 feet

Furniture Positions

Room entry door 2 feet 1 toward 2

Bathroom door 3 feet 2 toward 3

Window 4 feet 3 toward 4

Submit

Fig 4.1.1: Input (a)

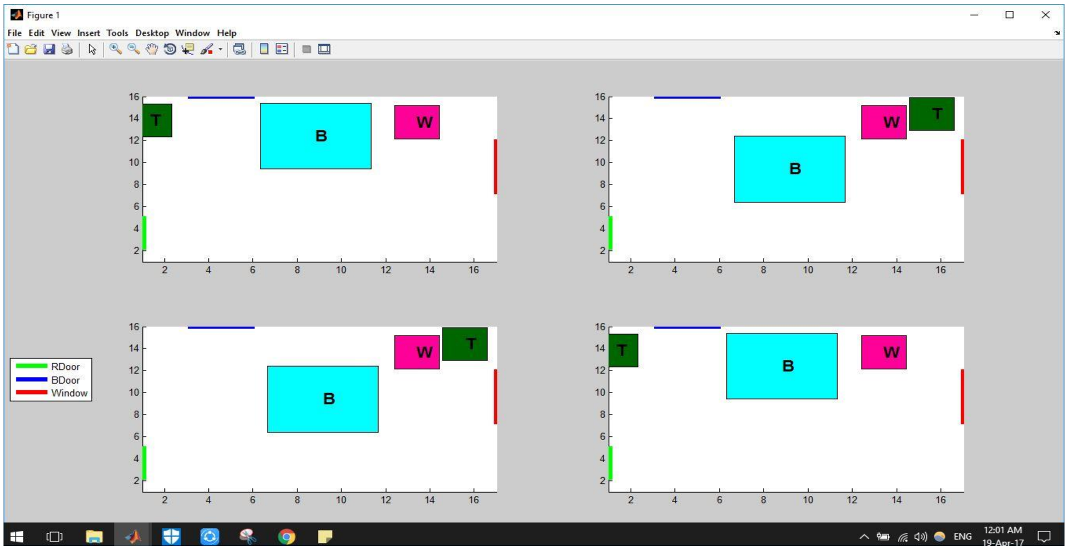
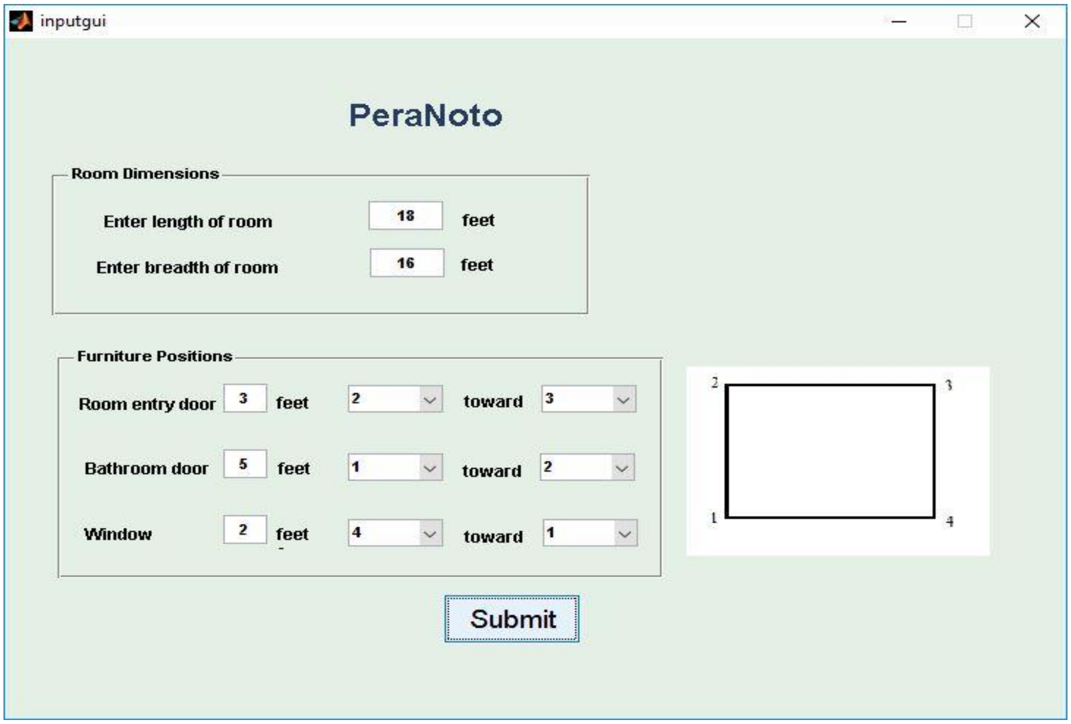


Fig 4.1.2: Output (a)

2. For a room of dimensions 18x16:



PeraNoto

Room Dimensions

Enter length of room 18 feet

Enter breadth of room 16 feet

Furniture Positions

Room entry door 3 feet 2 toward 3

Bathroom door 5 feet 1 toward 2

Window 2 feet 4 toward 1

Submit

Fig 4.2.1: Input (b)

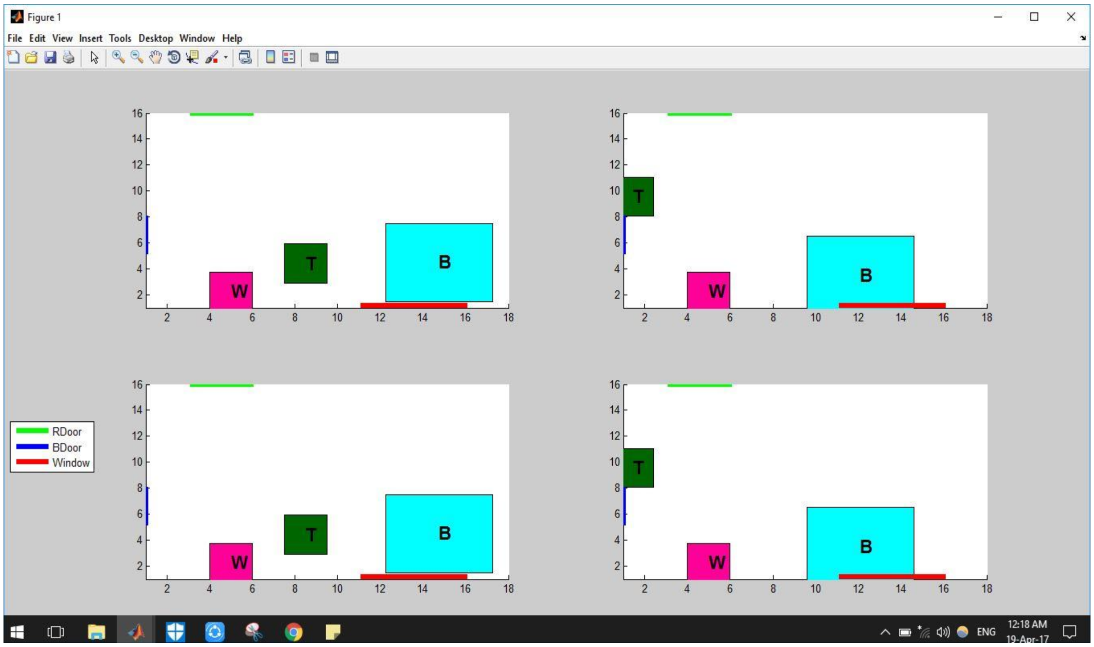
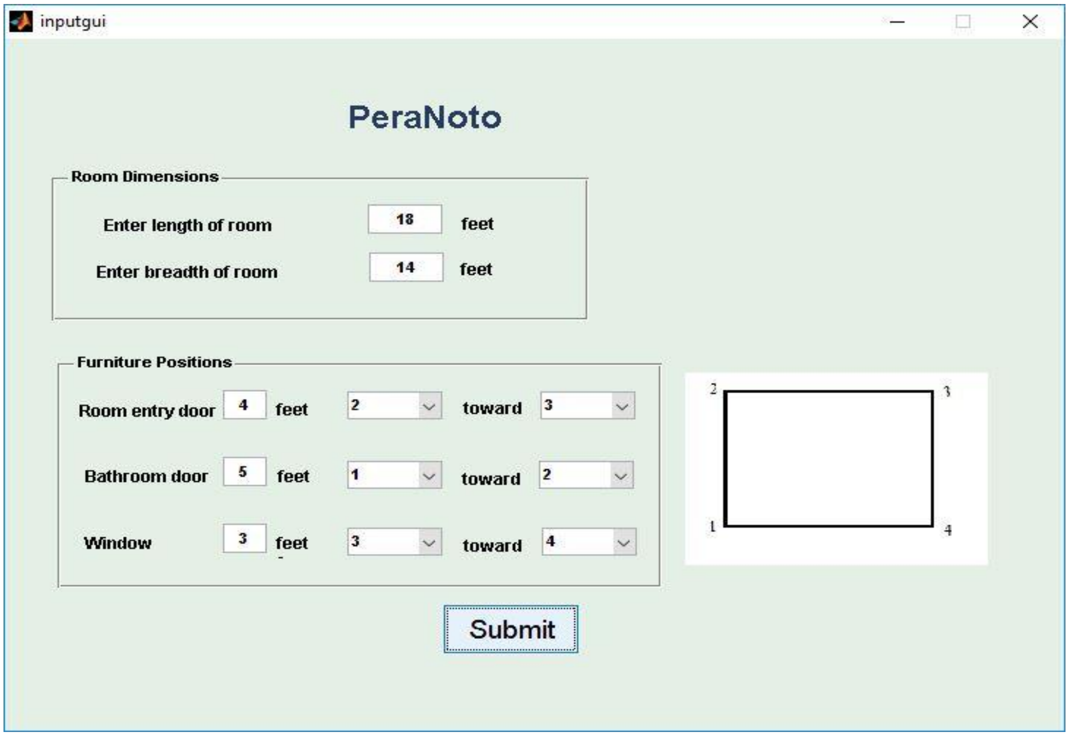


Fig 4.2.1: Output (b)

3. For a room of dimensions 18x14:


inputgui

PeraNoto

Room Dimensions

Enter length of room 18 feet

Enter breadth of room 14 feet

Furniture Positions

Room entry door	4	feet	2	toward	3
Bathroom door	5	feet	1	toward	2
Window	3	feet	3	toward	4

Fig 4.3.1: Input(c)

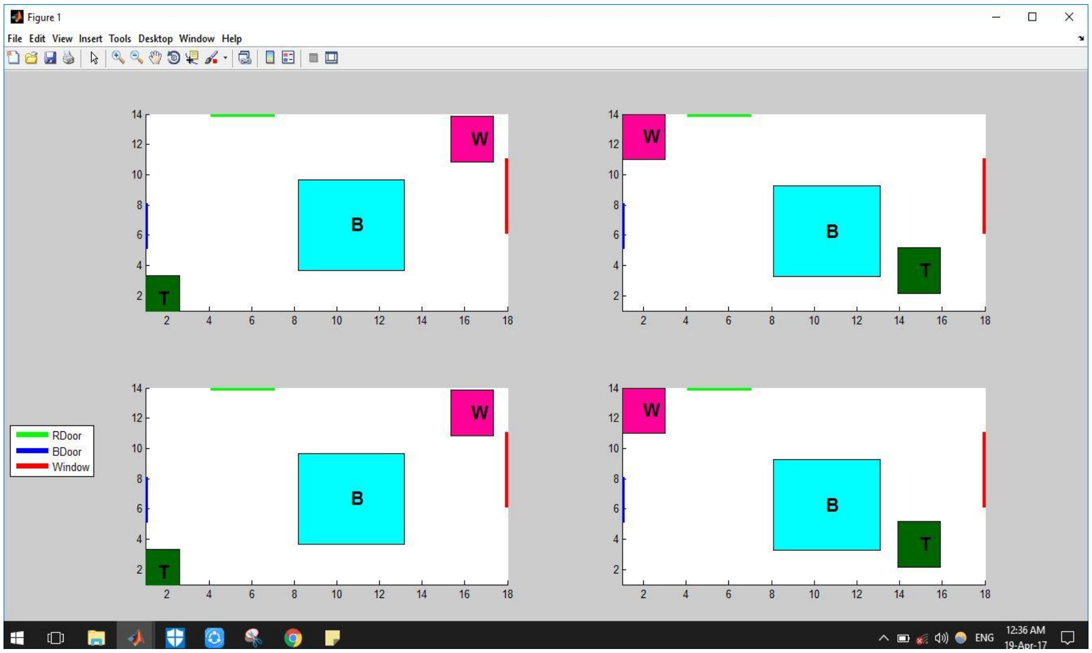
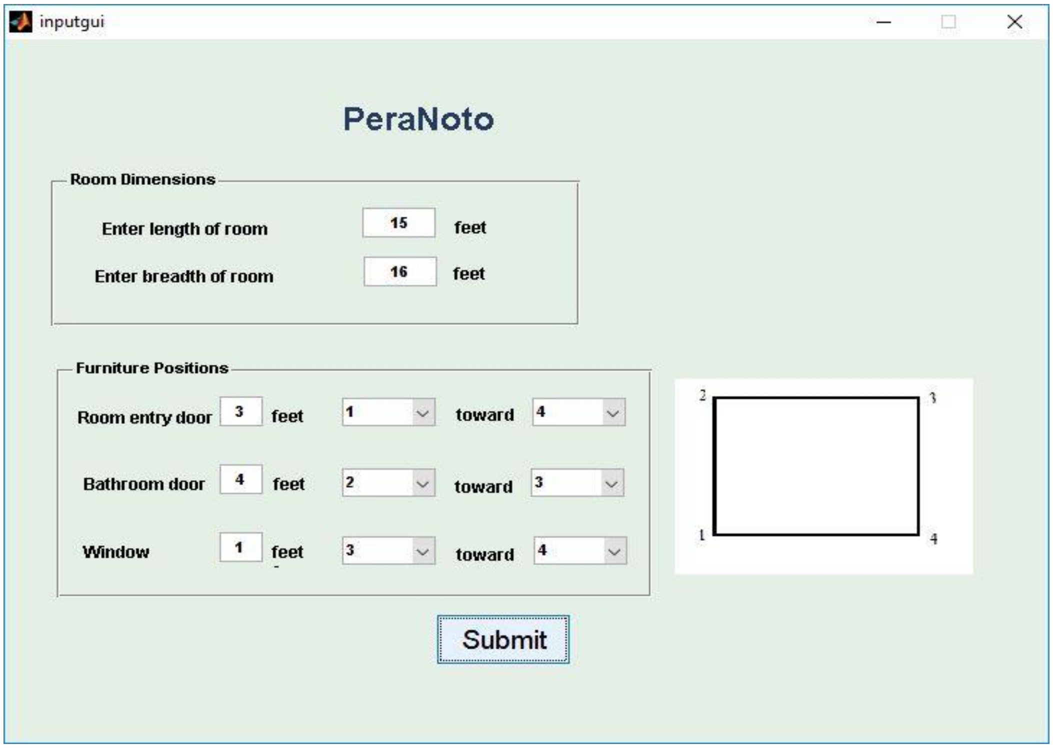


Fig 4.3.2: Output(c)

4. For a room with dimensions 15x16



PeraNoto

Room Dimensions

Enter length of room 15 feet

Enter breadth of room 16 feet

Furniture Positions

Room entry door 3 feet 1 toward 4

Bathroom door 4 feet 2 toward 3

Window 1 feet 3 toward 4

Fig 4.4.1: Input (d)

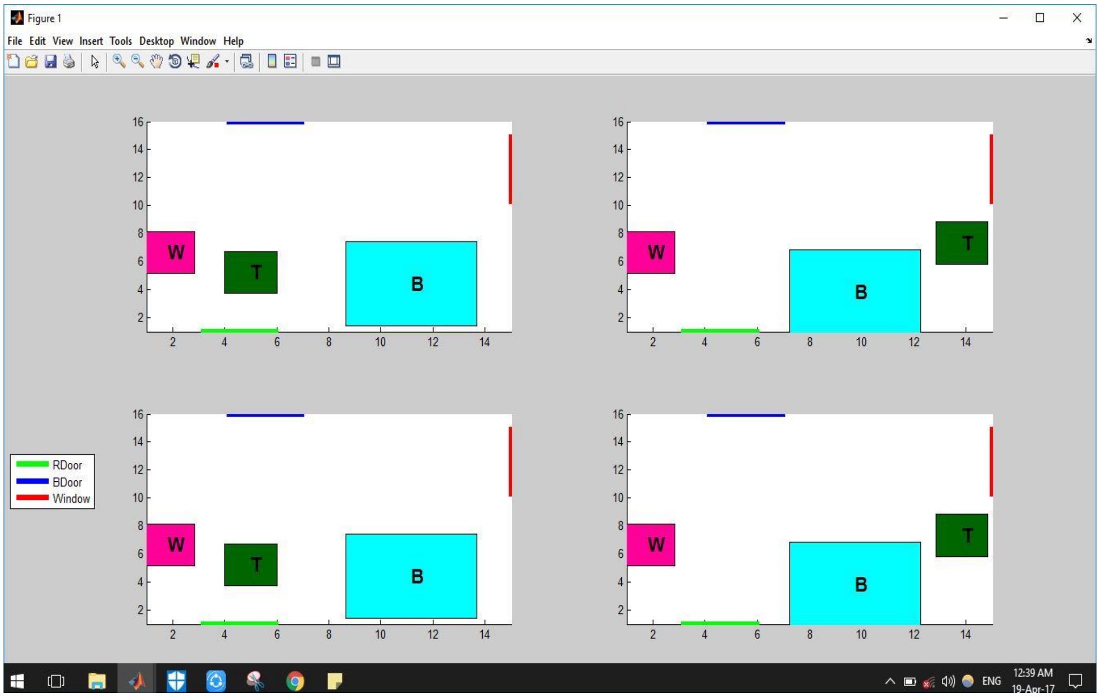
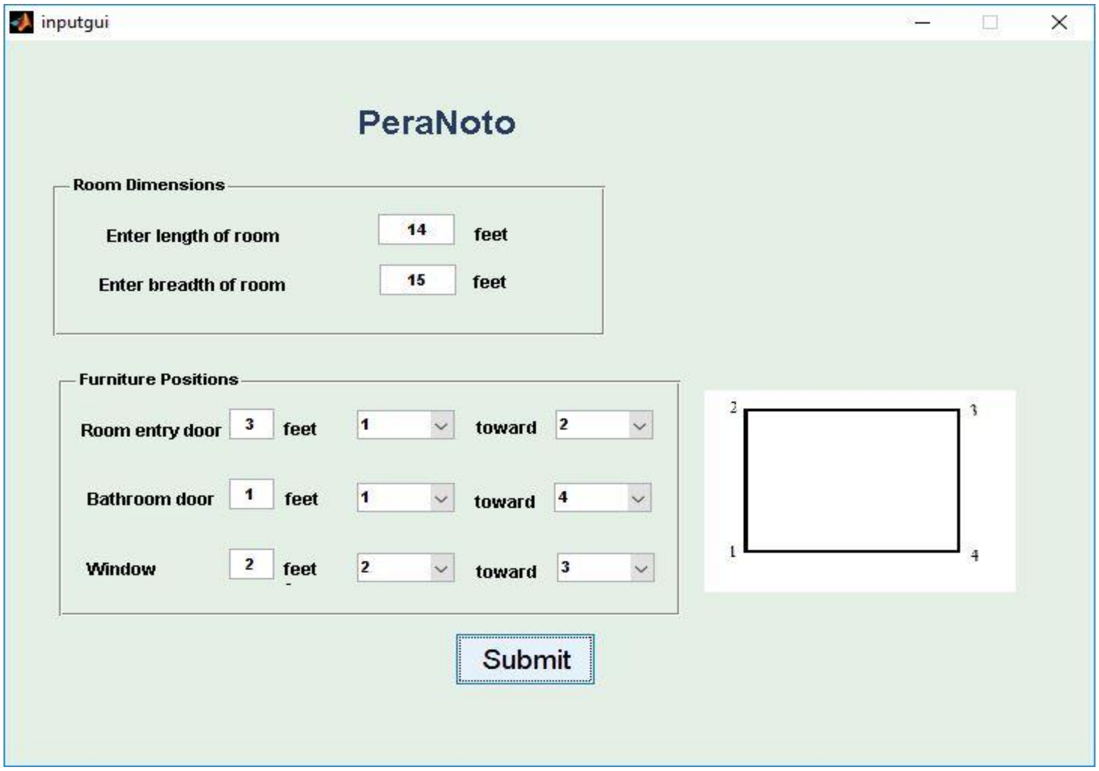


Fig 4.4.2: Output (d)

5 For a room of dimensions 14x15:

In case of some inputs, GA gives only one proper output. The screenshots below show such a case.



inputgui

PeraNoto

Room Dimensions

Enter length of room 14 feet

Enter breadth of room 15 feet

Furniture Positions

Room entry door 3 feet 1 toward 2

Bathroom door 1 feet 1 toward 4

Window 2 feet 2 toward 3

Submit

Fig 4.5.1: Input (e)

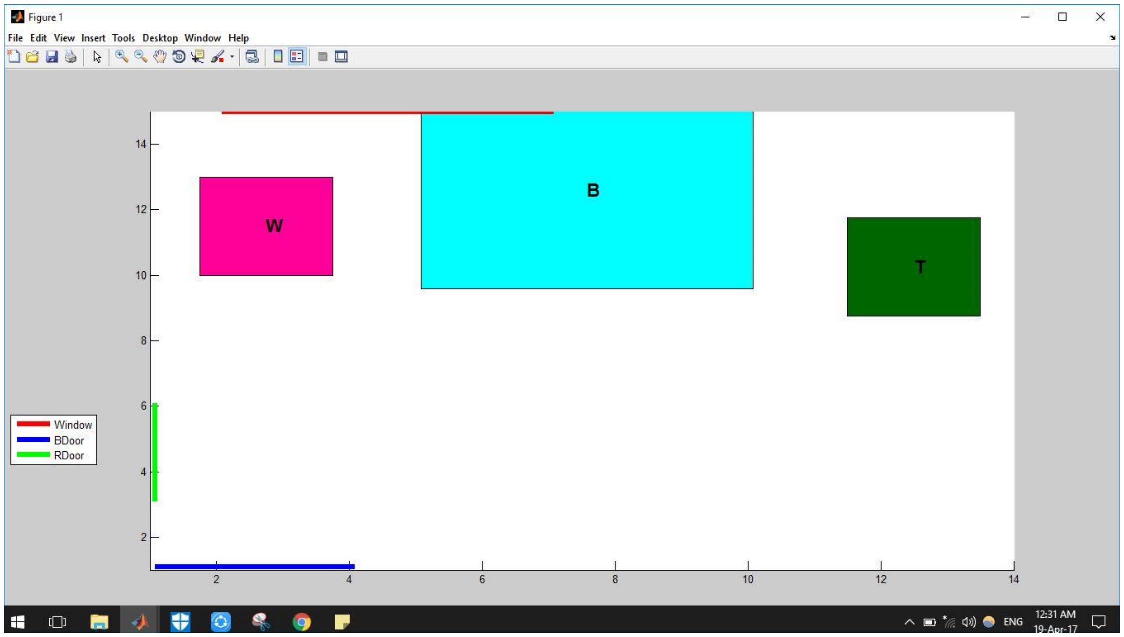


Fig 4.5.2: Output (e)

4.2 Evaluation Parameters

For a project to be analyzed and judged it is important for the system to go through the evaluation phase. In this phase the project is analyzed on the basis of key factors namely, relevance, efficiency and accuracy.

- **Relevance:** The criterion of relevance is used to assess whether the project fulfils an important function from a development perspective and whether its design was fundamentally suited to achieving the goals associated with the project.
- **Effectiveness:** The criterion of effectiveness is used to assess whether a development project achieved its goals. This can be evaluated by checking the system a number of times with different constraints in its working.
- **Efficiency:** The criterion of efficiency is performing or functioning in the best possible manner with the least waste of time and effort. For this factor to be analyzed the system needs to checked for the time it takes to execute the different test cases.
- **Accuracy:** Accuracy is defined as how the measured values are close to the target value. The system is to be checked for the number of times the result obtained is satisfactory when compared to how a human would do the same manually.

4.3 Performance Evaluation

- **Relevance:**

  The analysis of relevance of a system can be explained as: the proposed system is relevant as it is one of its kinds. Being an NP-Complete problem it hasn’t been touched by people yet and so this system gives scope for future development on this system as well as other similar systems by using this approach.
- **Efficiency, effectiveness and accuracy:**

  Analysis of the efficiency, effectiveness and accuracy of Genetic algorithm depends mostly on two factors:

  1. Initial population size
  2. Number of generations

Depending on the particular requirement, an optimum size has to be selected. When the values of accuracy for a population size were plotted on a graph, it was observed that as the size of the initial population was 20,000 the system generated optimum results as compared to other sizes.

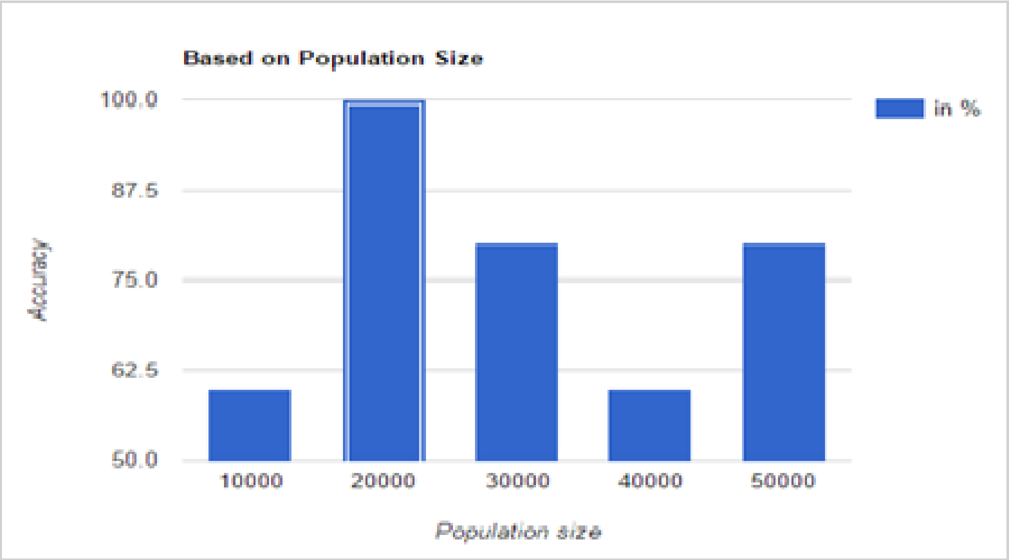


Fig 4.6: Accuracy v/s Population Size

One of the outputs when the population size is 20,000 is shown as below:

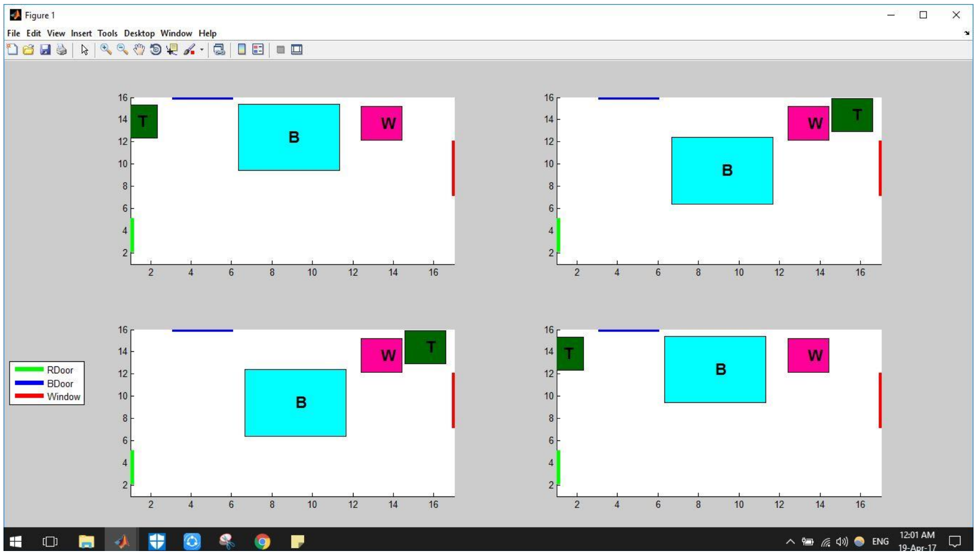


Fig 4.7 Output for 20,000 Population Size

The above result follows all the mentioned constrained and also produces an aesthetically pleasing arrangement since it places the Wardrobe and Table nearer to the walls and the Bed is also positioned in a relatively good position.

Comparably, the output for other sizes is not as satisfactory:

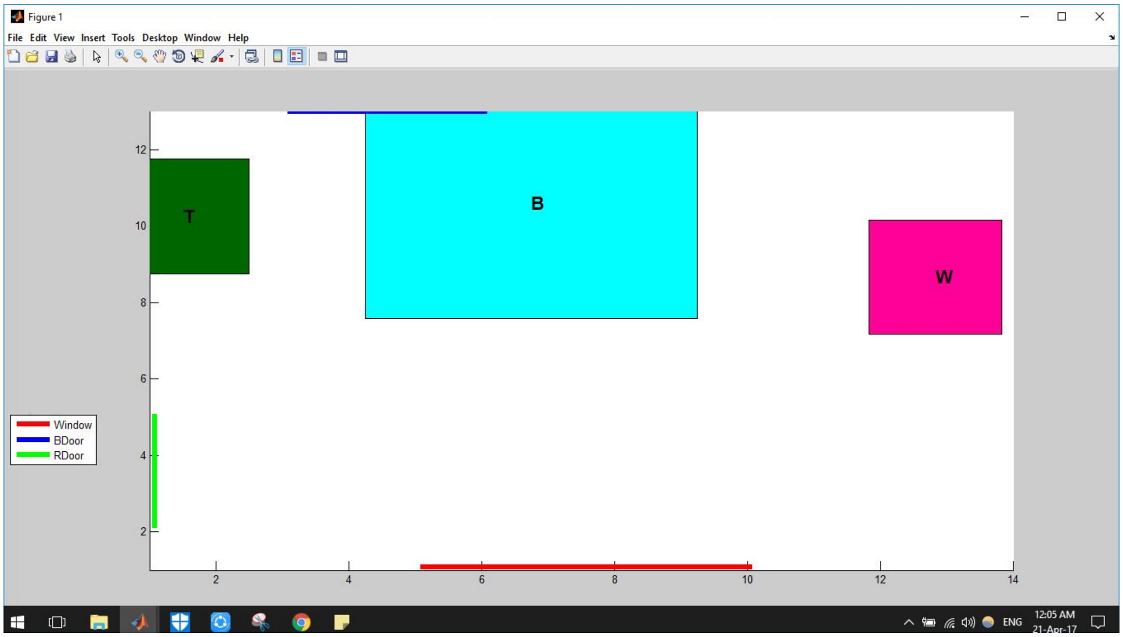


Fig 4.8: Output for Miscellaneous Population Sizes

Though the above image follows all the constraints of furniture arrangement applied to the system, the output generated is almost repetitive with hardly any appreciable difference in the results. Also, the Wardrobe is placed very near to if not in front of the window which is not very desirable.

On varying the number of generations, we observed that best outputs were received for 5 generations. In genetic algorithm, if the system runs out of chromosomes mid-generation then the rest of the generations are useless computations.

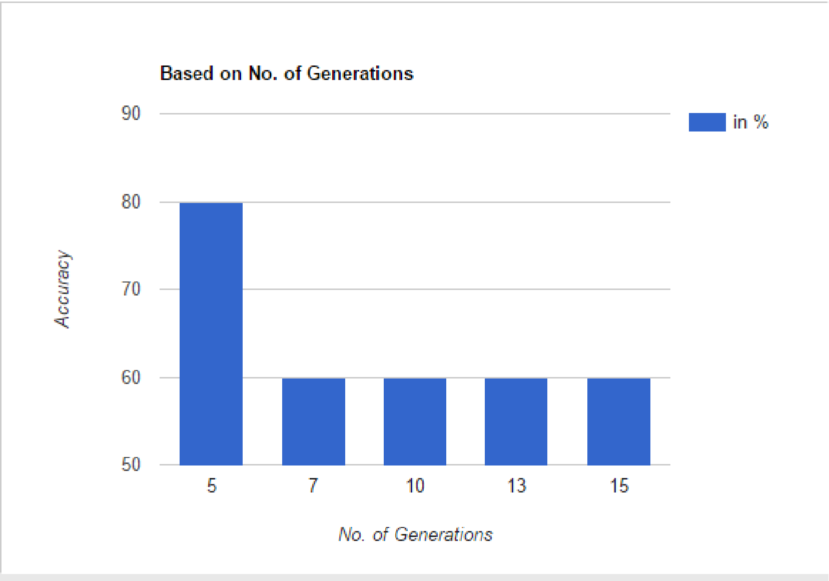


Fig 4.9: Accuracy v/s No. of Generations

One of the outputs received when number of generations was kept 5 is as follows:

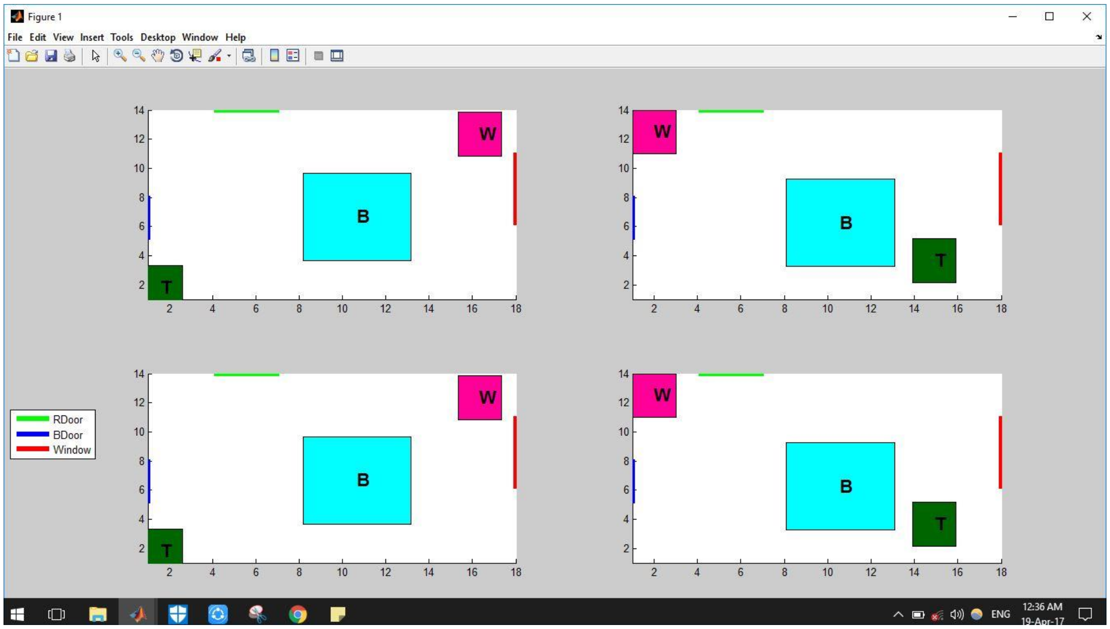


Fig 4.10: Output with 5 Generations

The above image gives an acceptable result of the Bed being in almost center of the room while the Wardrobe is in the corner or adjacent to the wall and Table is placed nearer to the Bed.

Comparably, for other cases, the output is not as desirable:

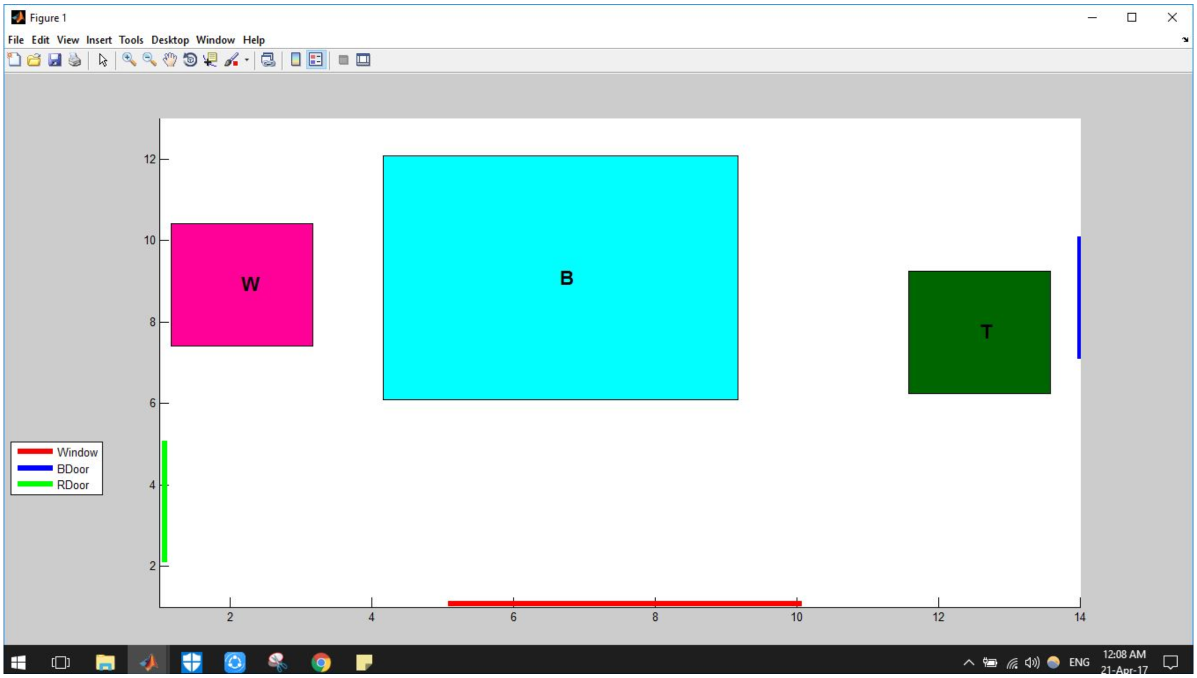


Fig 4.11: Output for Miscellaneous Generations

Based upon above observations, for our system, the number of generations has been decided as 5, for an initial population size of 20,000.

The relation of population size with time required is directly proportional. Upon varying the population size, the time required for the system to generate an output increases exponentially. The graph of time required vs. population size is as follows:

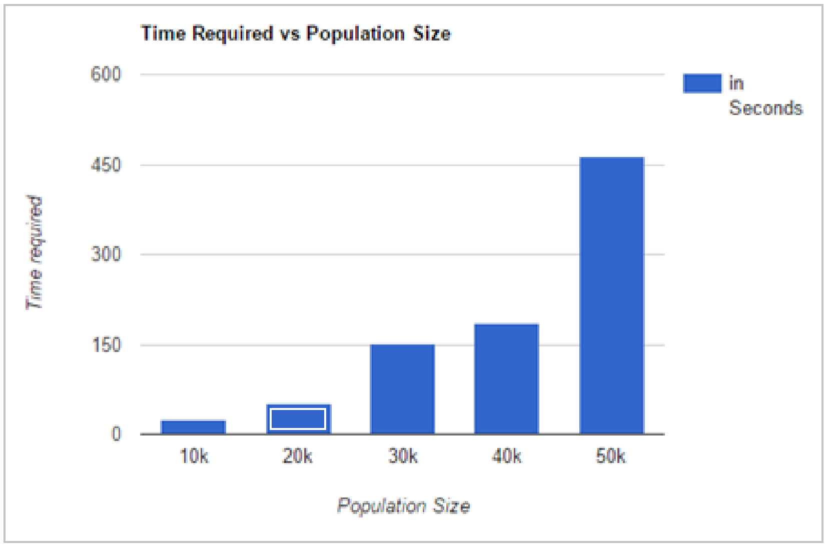


Fig 4.12: Time Required v/s Population Size

As the size of the initial population increases, the system has to deal with more genes, resulting in increased computations required for processing the genes.

However, there doesn’t seem to be any discernible relation between the time required and number of generations. There were no significant changes observed as the number of generations was increased or decreased, affirming that the time required for this system to generate a solution solely depends on the population size.

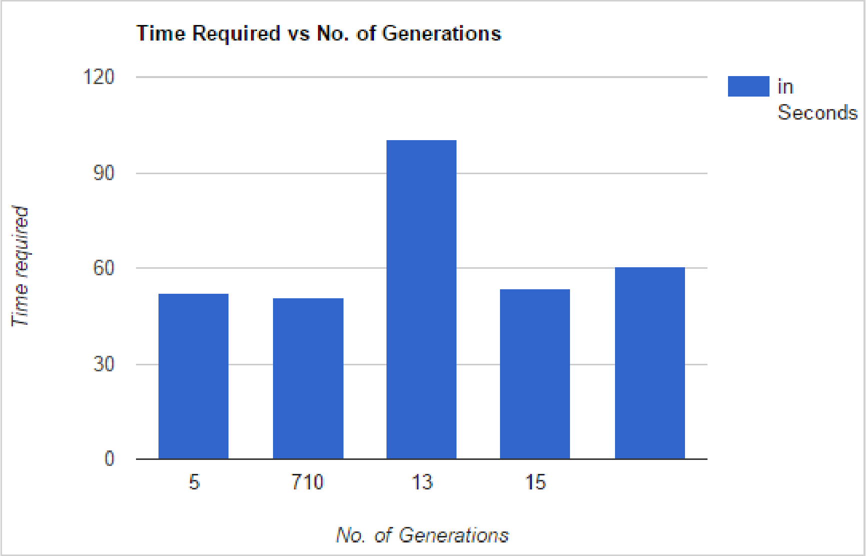


Fig 4.13 Time Required v/s No. of Generations

Chapter 5

Conclusion and Future Scope

5.1 Conclusion

PeraNoto uses genetic algorithm, a technique under the shade of artificial intelligence, to produce furniture layout plans for the user. PeraNoto requires basic input that includes size of the room followed by the position of doors and window from the any wall as per the user's convenience. These types of layout generations can be extended to include the scope of township planning, motion sensor placement in smart environments and in the generation of realistic indoor scenes in floor planning. The system developed is efficient and an extension of it would also be possible and not a problem.

5.1 Future Scope

5.1.1 Township Planning

The concept of planned optimized arrangement of objects can be applied in the case of Township planning. Using Genetic Algorithm (GA) the arrangement of residential buildings keeping in mind their proximity to schools, markets and hospitals can be planned. It can also be used to plan the layouts of roads and bridges as well as various connecting lanes in a township. Layout of various public places can also be planned in similar way.

5.1.2 3D Output

In the proposed system the generated output will be represented in 2D or as text format. A more convenient representation of the output for the user would be a 3D visualization of the furniture elements in their different arrangements. This can be achieved by using augmented reality concepts for generating 3D outputs.

5.1.3 Interior Designing

The concept of floor layout planning with the help of genetic algorithm can be extended to a broader scale when planning interiors of a whole office or a luxury hotel. Same principles that will be applied in the proposed system can be modified according to the size of the place to be designed and number of elements to be considered.

Appendix

Given below is the publication of this paper in the International Journal of Innovative Research in Science, Engineering and Technology:

IJIRSET

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Website: www.ijirset.com

Vol. 6, Issue 4, April 2017

Floor Layout Planning Using Artificial Intelligence Technique

Mitali Chavan, Nidhi Menon, Ritika Kumar, Shivani Rana

U.G. Students, Department of Computer Engineering,Pillai College of Engineering, New Panvel, Maharashtra, India

Abstract:In the era of e-commerce while buying furniture online the customers obviously feel the need for visual representation of the arrangement of their furniture. Even when doing interiors of the house it's difficult to just rely on assumptions about best layouts possible and professional help may become quite expensive. In this project, we make use of Genetic Algorithm (GA) which is an Artificial Intelligence technique to display various optimal arrangements of furniture. The basic idea behind using GA is developing an evolutionary design model. This is done by generating chromosomes for each possible solution and then performing a crossover between them in each generation until an optimum fitness function is reached. Modification in chromosome representation may also be done for better results. The proposed system will generate different layout designs for the furniture keeping in consideration the structure of a master bedroom.

Keywords:chromosome, genetic algorithm, genetic operators, layout generation

I. INTRODUCTION

In architecture and building engineering, a floor plan is a drawing to scale, showing a view from above, of the relationships between rooms, spaces and other physical features at one level of a structure. Floor plans may include notes for construction to specify finishes, construction methods, or symbols for electrical items. The floor planning can be classified into two categories either an equal area layout problem or an unequal area layout problem. The equal area layout problem is to determine how to assign a set of distinct facilities, to a set of distinct locations, so that each facility can be assigned to a single location called as one-to-one assignment problem. The unequal area layout problem is to regulate the allocation of all facilities within a block plan (or available area).In the era of e-commerce while buying furniture online the customers obviously feel the need of visual representation of the arrangement of their furniture. Even when doing interiors of the house it's difficult to just rely on assumptions about best layouts possible and professional help may become quite expensive. The proposed system will allow users to view multiple possible solutions of given pieces of furniture and let them decide what they like the best. This kind of help with multiple furniture arrangements will also be very useful while developing real life indoor scenes for gaming or animation purposes.Optimizing furniture arrangement into a realistic and functional indoor configuration involves considerable complexity, taking into account various interacting factors, such as pair wise furniture relationships, spatial relationships with respect to the room, and other human factors. An effective representation that captures the necessary spatial relationships is needed. Some of the attributes to be considered while deciding furniture placement are - Bounding surfaces, Centre and orientation, Accessible space, Viewing frustum, Hierarchical relationships between different furniture objects.

The objective of this work is as follows:

1. To establish that Genetic Algorithm is a suitable technique for designing layout.
2. To discuss the mechanism of Genetic algorithm and to compare it with other placement algorithms and site the scenarios when it has proved to be more efficient.
3. To understand method of feature extraction for furniture placement system and various constraints that should not be violated while designing the arrangement of furniture.To identify evaluation metrics used for performance analysis of proposed system

II. **RELATED WORK**

Jun H. Jo and John S. Gero [1] in their paper ‘Space Layout Planning Using an Evolutionary Approach’ describe a design method based on constructing a genetic/evolutionary design model. They touch the major issues while using genetic designing model like the formulation of knowledge representation set and the genetic communication. A set of design elements, is in the form of design rules and needs to be interpreted into the language of the genetic search system. On the basis of the advantages of genetic evolutionary design process and the results of the implementation, they concluded that the coupling of an evolutionary search technique with a design process can produce very good results, especially for large-scale problems which are at present computationally difficult.

The paper ‘Generic Chromosome Representation and Evaluation for Genetic Algorithms’ by Kristian Guillaumier [2] explores the role of chromosomes in genetic algorithms.In conventional implementations, most parameters, the fitness function, chromosome representation, and genetic operators are usually hard-coded. If the nature of the problem varies slightly or critical parameters change, the original code is revised sometimes substantially as per the need of the problem. The chromosomes are represented using a direct representation scheme. Once encoding is done, basic single point crossover and mutation is applied. According to the fitness value the generations of chromosomes are checked iteratively and the one closest to the fitness value or at which the limit is achieved is returned as the optimal solution. This technique has been implemented as a high-level modeling language called OPML (Optimization Problem Modeling Language) together with a general-purpose Genetic Algorithm-based runtime.

MohdSaberi Mohammad, Safaai Deris, Safie Mat Yatim, Muhammad Razib Othman[3] in their paper ‘Feature selection method using genetic algorithm for the classification of small and high dimension data’, propose feature selection method that finds and selects informative features from small or high dimension data thereby maximizing the efficiency of the classification process. They apply genetic algorithm to search out and identify the potential informative features combinations for classification and then use the classification accuracy from the support vector machine classifier to determine the fitness in genetic algorithm. The first proposed approach is called GASVM- a combination of GA with SVM without any modification in chromosome representation. The second proposed approach is that of New-GASVM which will modify the model of chromosome representation. On implementation along with the current best classifiers, the New-GASVM turns out to be the best performer.

Ryota Tachikawa and Yuko Osana [4] in their paper, ‘Office Layout Support System using Genetic Algorithm– Generation of Room Arrangement Plans for Polygonal Space’ propose an office layout support system using genetic algorithm. The proposed system has two phases: (1) generation of room arrangement plans and (2) generation of layout plans for workspace. The working is such that some conditions on rooms and furniture are given by a user and some room arrangement plans which satisfy the conditions are generated by Genetic Algorithm (GA). After one of them is selected by a user, some layout plans for workspace which satisfy the conditions are generated. A combined GA which is based on the Adaptive Genetic Algorithm (AGA) and the GA with Search Area Adaption (GSA) is employed in this system. For the generation of layout plans for workspace, layout plans are expressed in the form of genes in a similar fashion and further evaluation and computer experiments are done. When compared with normal genetic algorithm, AGA and GSA, it is found that the proposed system gives the best results.

Eugénio Rodrigues, Adélio Rodrigues Gaspar and Álvaro Gomes [5] in their paper ‘An evolutionary strategy enhanced with a local search technique for the space allocation problem in architecture, Part 1: Methodology’ propose an enhanced hybrid evolutionary computation scheme that couples an Evolutionary Strategy (ES) with a Stochastic Hill Climbing (SHC) technique to generate a set of floor plans to be used in the early design stages of architectural practice. The algorithm used presents a hybrid behavior by combining ES with an SHC technique. This algorithm is named Evolutionary Program for Space Allocation Problem (EPSAP) and uses adaptive operators to perform the geometric transformations of the rooms, their walls and connections, and openings according to previously stored information.

Brian L.Thomas, Aaron S. Crandall and Diane J. Cook's [6] work ‘A Genetic Algorithm approach to motion sensor placement in smart environments’, explores techniques to identify "reasonable" sensor placement plans within smart home context. The goal here is to decide the strategic location of sensors over a given restricted area such as a kitchen or a living room. The paper compares various placement algorithms for sensor layouts such as 1) Human Intuition-Based 2) Monte Carlo-Based 3) Two-Dimensional Uniform Placement (Grid) 4) Hill Climbing and 5) Genetic Algorithm (GA) to prove that GA gives most accurate results. The paper thus concluded that the GA algorithm’s layouts are more efficient to generate than Hill Climbing, in addition to being significantly more accurate than the baseline algorithms tested.

Lap-Fai Yu, Sai-Kit Yeung, Chi-Keung, Demetri Terzopoulos, Tony F. Chan, and Stanley J. Osher [7] in their work ‘Make it Home: Automatic Optimization of Furniture Arrangement’, conducted a perceptual study to demonstrate that their system can synthesize multiple realistic furniture arrangements. The paper efficiently discusses Furniture Relationship Extraction, Furniture Arrangement Optimization and various Constraints that should not be violated. Furniture Relationship Extraction has two parts Object Representation and Learning Prior Relationships. Stochastic optimization methods, specifically, simulated annealing are used. The constraints mentioned are accessibility**,** visibility and pathway connecting doors**.**

III. Scope of the Research

The proposed system will generate multiple layout designs for the furniture keeping inconsideration the structure of a master bedroom. Its dimensions will be given as input by the user. The master bedroom will have a window and two doors; one for entering the room and one for the bathroom. The size of the doors, window and furniture elements considered will be predefined but the position of doors and windows would be given by the user. The furniture elements are Wardrobe, Bed and Table. The input given will be taken from the user in text format via a GUI and the output will be a collection of four best possible arrangements of the furniture elements. This will be represented in 2D format. The system design helps in maintaining basic relationship between the furniture elements and also with respect to the room. The output to be generated is the top view of the room. The proposed system is a NP-Complete problem since there is no known solution to it. The only system existing close to the proposed system involves the use of drag and drop of furniture elements in either a blank room layout or by selecting one of the room layouts already provided, that seem closest to their room. Since this doesn’t always fulfil the desires of the user and is likely to miss out or suffer human errors the proposed system proves its significance by automatically generating the best possible solutions keeping in mind the aesthetics and relationship among the different furniture elements.

IV. Proposed Methodology

The system follows all the stages of Genetic algorithm as shown in the fig. 4.1.

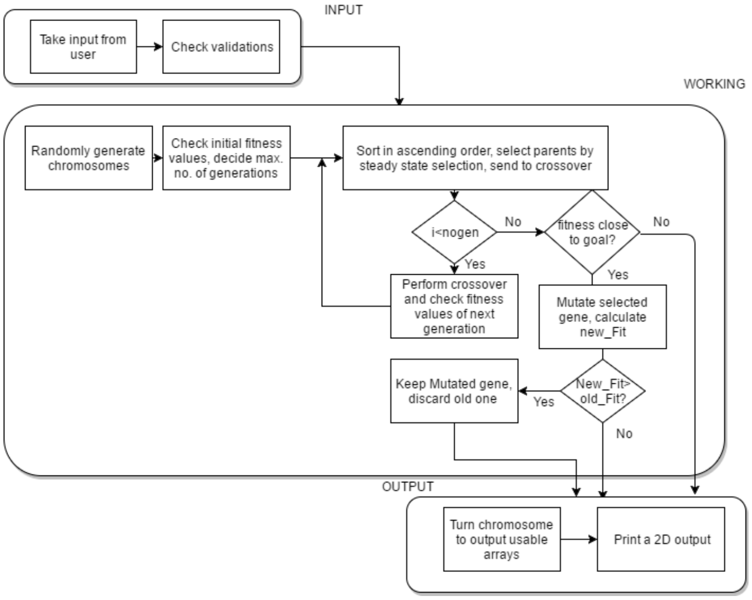
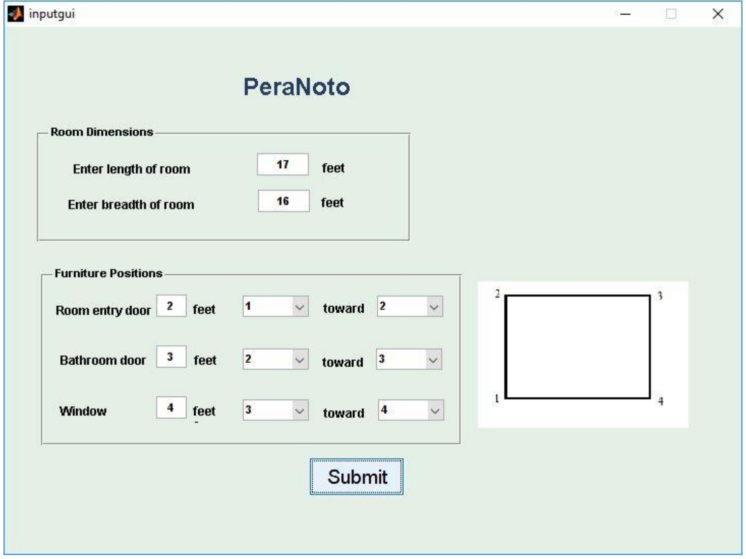

INPUT
Take input from user
Check validations
WORKING
Randomly generate chromosomes
Check initial fitness values, decide max. no. of generations
Sort in ascending order, select parents by steady state selection, send to crossover
i<nogen
No
Yes
fitness close to goal?
No
Yes
Perform crossover and check fitness values of next generation
Mutate selected gene, calculate new_Fit
Keep Mutated gene, discard old one
New_Fit> old_Fit?
Yes
No
OUTPUT
Turn chromosome to output usable arrays
Print a 2D output

Fig.4.1: Methodology Flowchart

The input is taken from the user through the GUI where the user enters length, breadth and the distance of the doors and window from one point towards another point on a wall. The GUI has a rectangle displayed on the right which number the corner points to help the user measure the distances correctly as shown in fig. 4.2.


inputgui

PeraNoto

Room Dimensions

Enter length of room 17 feet

Enter breadth of room 16 feet

Furniture Positions

Room entry door 2 feet 1 toward 2

Bathroom door 3 feet 2 toward 3

Window 4 feet 3 toward 4

Fig.4.2: GUI

The system then generates a randomized population of chromosomes. These chromosomes contain information about the coordinates of left bottom corner and also their alignment i.e. horizontal or vertical.

X_B	Y_B	A	X_T	Y_T	A	X_W	Y_W	A
1	2	3	4	5	6	7	8	9

Fig 4.3: Chromosome Structure

The fig.4.3 gives representation of a single chromosome. Each furniture object has a designated sub-range of the array to code information specifically about it. The designated ranges are as follows

- sub- range (1:3) represents Bed ; X_B and Y_B are coordinates of the bottom left corner of the Bed
- sub-range (4:6) represents Table; X_T and Y_T are coordinates of the bottom left corner of the Table
- sub-range (7:9) represents Wardrobe; X_W and Y_W are coordinates of bottom left corner of the Wardrobe

Each of the positions 3, 6 and 9 represent the alignment (A) for the furniture represented by the respective sub-ranges to which these positions belong. These positions are set to 1 if alignment is horizontal and 2 if alignment is vertical. The remaining two positions in each of the sub ranges give the coordinates of the bottom left corner of the furniture object that they represent. The coordinates are generated randomly. The alignment of these furniture elements is whether 1 or 2, is also random. Next, calculation of fitness values of each chromosome is done based on various constraints such as - the furniture objects do not overlap, they do not exceed the room dimensions, arrangement of furniture nearer to the walls is favoured, no furniture is in front of the two doors and wardrobe should never be in front of the window. The fitness scores assigned accordingly are stated in table 4.1.

Constraints	Wardrobe				Bed				Table			
	Min Score	Max Score	Flags		Min Score	Max Score	Flags		Min Score	Max Score	Flags	
			Yes	No			Yes	No			Yes	No
Overlap with Bed	0	10	1	0	-	-	-	-	-	-	-	-
Overlap with Table	0	10	1	0	0	10	1	0	-	-	-	-
Within room dimensions	0	10	1	0	0	10	1	0	0	10	1	0
In front of room door	0	10	1	0	0	10	1	0	0	10	1	0
In front of washroom door	0	10	1	0	0	10	1	0	0	10	1	0
In front of window	0	10	1	0	5	10	-	-	7	10	-	-
Near to any walls	5	10	-	-	5	10	-	-	5	10	-	-
Total Max/Min Score	5	70			10	60			12	50		

Table 4.1: Fitness scores

In addition to the score there are also flags included to notify certain conditions so that occurrences of false positives where a chromosome shows high fitness value score even after not fulfilling some basic conditions are identified. The scores assigned for each object were decided on the basis of trial and error method by experimenting with various random values and are subject to change as per the needs of the application.

- Fitness score of Wardrobe is calculated out of 70
- Fitness score of Bed is calculated out of 60
- Fitness score of Table is calculated out of 50

Hence Maximum Fitness Score of a Chromosome = ∑ Maximum Fitness score of each Furniture object
= Maximum score of (Wardrobe + Bed + Table)
= 70+60+50
= 180

Thus, Total Fitness score of a chromosome is always calculated out of 180.

Next we discuss flags; the flags give information about occurrence of conditions such as overlap, furniture exceeding room dimensions, furniture in front of any of the doors and wardrobe in front of the window. If a condition occurs then the flag is set to 1 otherwise set to 0. These conditions are checked for each object and then finally the flags for a chromosome are set by performing logical OR operation between the values of flags obtained for each object.

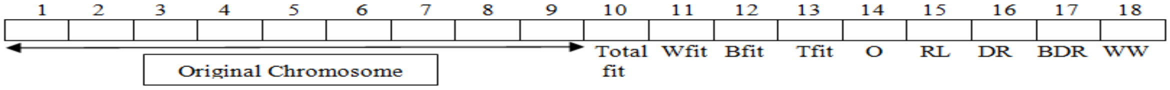


Fig. 4.4: Chromosome after Fitness Evaluation

The fig.4.4 represents the structure of the chromosome after its fitness has been calculated and related values are appended at the end of the chromosome. The flags are represented at positions 14,15,16,17 and 18.

Flag O - Flag is set to 1 if Overlap condition occurs
Flag RL - Flag is set to 1 if furniture objects exceeds room dimensions

Flag DR - Flag is set to 1 if furniture objects are in front of room entry door
Flag BDR - Flag is set to 1 if furniture objects are in front of bathroom door
Flag WW - Flag is set to 1 if Wardrobe is in front of window

To decide whether these flags should be set to 0 or 1 first the above flag conditions are checked for each furniture object and accordingly local flags are set to 1 or 0 then logical OR operation is performed on these local flags to set the chromosome flags O, RL, DR, BDR thus these flags will be set to 1 even if anyone furniture object breaches any constraint. The last flag WW only displays exclusive information about Wardrobe and hence does not include any other operations.

The positions 10,11,12 and 13 represent Total fitness score(Totalfit), Wardrobe fitness score(Wfit), Bed fitness score(Bfit) and Table fitness score(Tfit) respectively.

- Totalfit - Fitness value scored out of 180 for the entire chromosome
- Wfit - Fitness value scored out of 70 for the arrangement of Wardrobe
- Bfit - Fitness value scored out of 60 for the arrangement of Bed
- Tfit - Fitness value scored out of 50 for the arrangement of Table

Totalfit = ∑ Fitness value scored for each furniture object
= Wfit + Bfit + Tfit

The next step after evaluating fitness score is to select parent chromosomes for crossover. The selection operator used here is Steady State selection. This selection operator sets a minimum threshold fitness value and all the chromosomes with a higher fitness value than the threshold is allowed to participate in the crossover. For the system, not only do the chromosomes have to possess a Totalfit value higher than the threshold but also all the above mentioned flags should be set to 0 for them to qualify for the crossover. Being an NP- Complete problem there is a wide range of chromosomes that can be created but are undesirable and thus there is a need for this extra condition to be fulfilled in order to get meaningful output in less time. Thus, this additional measure helps to be assured that only good chromosomes are passed on to create next generation.

The crossover performed here is Two-Point crossover. The chromosomes are sorted in ascending order according to their Totalfit value and then crossover is performed between a chromosome and its immediately next chromosome. Since, crossover should be performed only within 3 furniture objects and the internal coordinates or alignment of each object should not be disturbed sub-ranges for Bed(B), Table(T), and Wardrobe(W) are each treated as separate blocks.

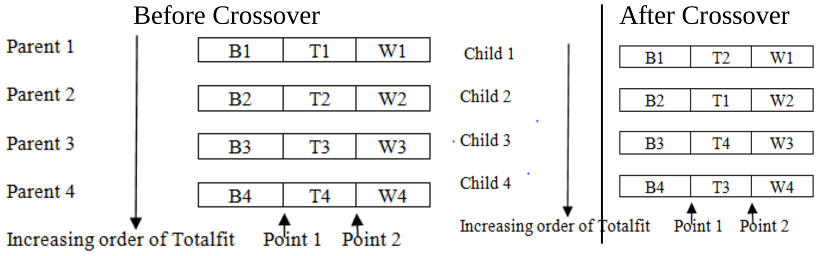


Fig.4.5: Before and after Crossover

As seen in fig.4.5, Two-Point crossover is done by replacing only the Table blocks between the two simultaneous chromosomes. The fitness value of these Child chromosomes is again calculated similarly and then selected ones are again sent for crossover, this process continues over a few generations. The child chromosomes of the last generation are expected to have good fitness values. These are the generated chromosomes that make up the result and are displayed to the user.

If a chromosome has fitness value very near to the goal fitness value then that chromosome can be mutated to reach the goal value. The probability of any chromosome undergoing mutation is very less and almost negligible. In the system if any chromosome undergoes mutation then only the alignment of any of the element is changed to reach the goal value. To decide which furniture element's alignment will change the difference between its goal fitness value and current fitness value is calculated. The alignment of element with maximum difference is then changed.

Element undergoing change in alignment = Element with max (difference (goal fitness value, current fitness value)
= Max ((70-Wfit), (60-Bfit), (50-Tfit))

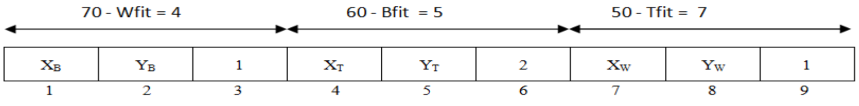


Fig.4.6: Before Mutation

In the fig.4.6, for instance, the difference between 50 - Tfit is maximum when compared to the other two differences, thus the alignment of Table is changed during Mutation. Thus, after mutation the chromosome changes to the fig.4.7.

X_B	Y_B	1	X_T	Y_T	2	.X_W	Y_W	2
1	2	3	4	5	6	7	8	9

Fig.4.7: After Mutation

The mutation does not guarantee that the resulting chromosome will have a fitness value greater than its previous value. Hence, the fitness value is again calculated. If the new fitness value is greater than the previous value only then the mutated chromosome is included in the result otherwise the previous un-mutated chromosome is part of the result.

V. Analysis and Research

Analysis of the efficiency and accuracy of Genetic algorithm depends mostly on two factors:

1. Initial population size
2. Number of generations

Depending on the particular requirement, an optimum size has to be selected. When the values of accuracy for each population size were plotted on a graph as shown in fig. 5.1, it was observed that as the size of the initial population was 20,000 the system generated optimum results as compared to other sizes.

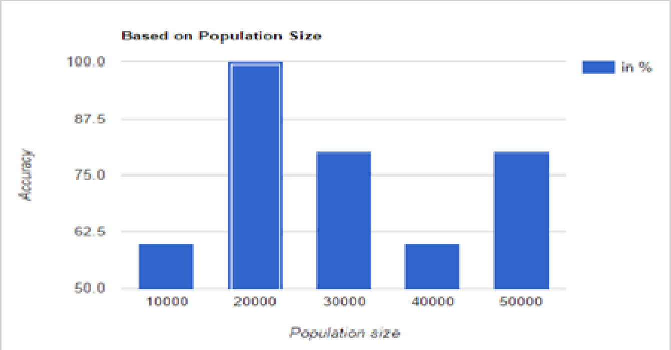


Fig. 5.1: Accuracy v/s. Population Size

In all the output images, Rectangle B represents Bed, Rectangle T represents Table and Rectangle W represents Wardrobe.

One of the outputs when the population size is 20,000 is shown in fig. 5.2. The arrangement follows all the mentioned constrained and also produces an aesthetically pleasing arrangement since it places the Wardrobe and Table nearer to the walls and the Bed is also positioned in a relatively good position.

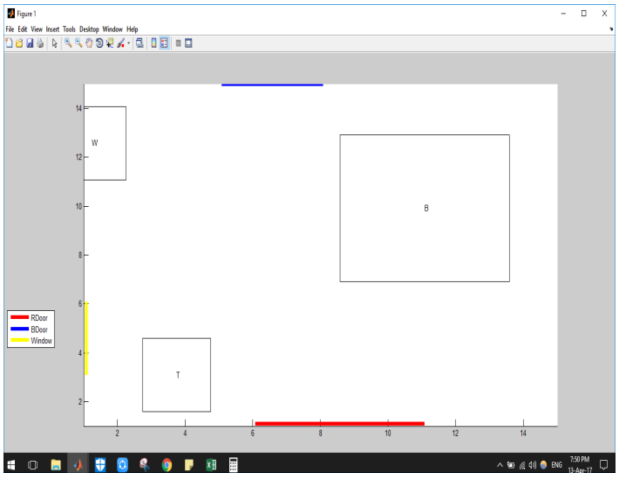


Fig. 5.2: Output with Population Size 20,000

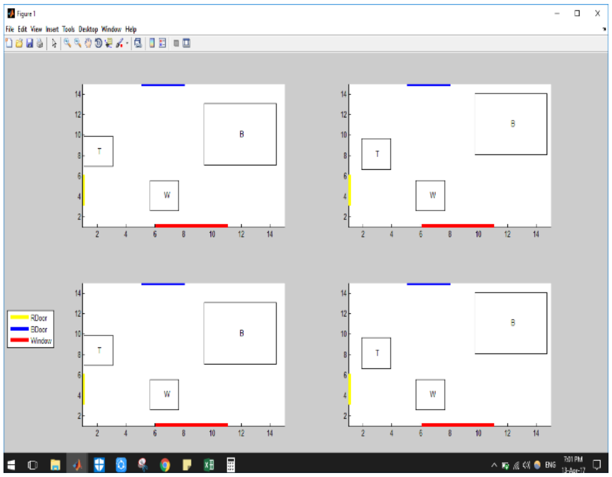


Fig. 5.3: Output with Miscellaneous Population Size

Comparably, the output for other population sizes is not as satisfactory. Though fig. 5.3 follows all the constraints of furniture arrangement applied to the system, the output generated is almost repetitive with hardly any appreciable difference in the results. Also, the Wardrobe is placed very near to if not in front of the window which is not very desirable.

On varying the number of generations, we observed that best outputs were received for 5 generations. In genetic algorithm, if the system runs out of chromosomes mid-generation then the rest of the generations are useless computations. This can be proved by plotting a graph of number of generations v/s Accuracy achieved as shown in fig 5.4.

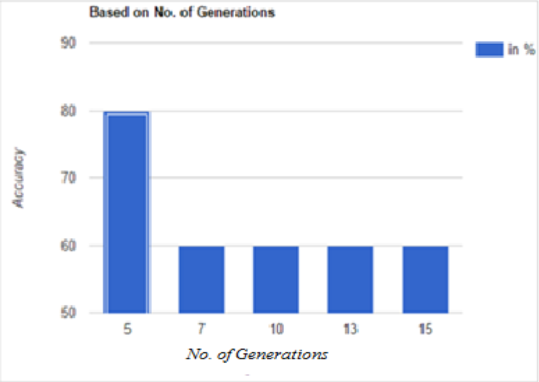


Fig. 5.4: Accuracy vs. No. of Generations

One of the outputs received when number of generations was kept 5 is as shown in fig 5.5. It gives an acceptable result of the Bed being in almost center of the room while the Wardrobe is in the corner or adjacent to the wall and Table is placed nearer to the Bed.

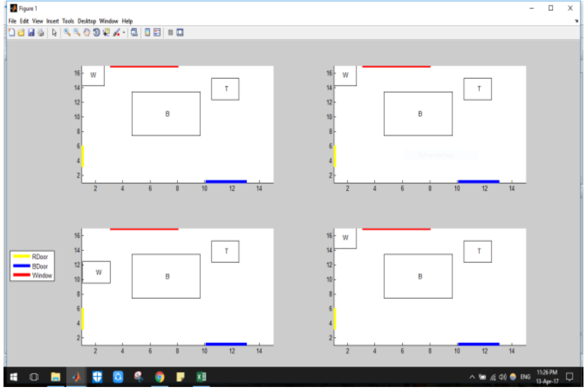


Fig. 5.5: Output with 5 Generations

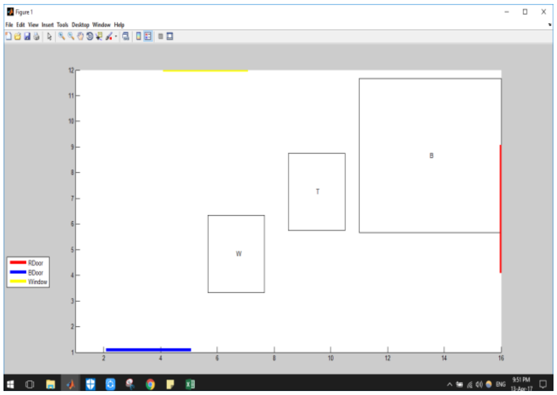


Fig. 5.6: Output Otherwise

For other no. of generations, the output arrangement is not very favorable as shown in fig 5.6. The Wardrobe is placed in the center of the room which is not desirable and hence not an optimal solution. Based upon above observations, for our system, the number of generations has been decided as 5, for an initial population size of 20,000.

The relation of population size with time required is directly proportional. Upon varying the population size, the time required for the system to generate an output increases exponentially. The graph of time required vs. population size is as shown in fig 5.7. As the size of the initial population increases, the system has to deal with more genes, resulting in increased computations required for processing the genes and hence increase in the time required to generate output.

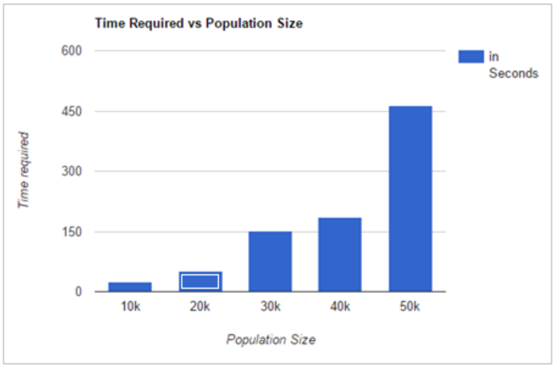


Fig. 5.7: Time required v/s. Population Size

However, there doesn't seem to be any discernible relation between the time required and number of generations. There were no significant changes observed as the number of generations was increased or decreased as can be observed from the graph of time required v/s no. of generations fig. 5.8, affirming that the time required for this system to generate a solution solely depends on the population size.

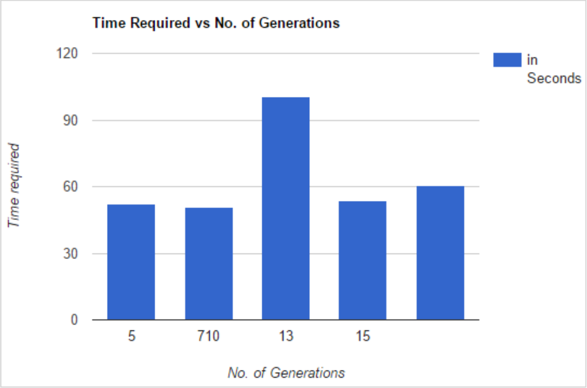


Fig. 5.8: Time required v/s. No. of Generations

VI. CONCLUSION

The system uses genetic algorithm, a technique under the shade of artificial intelligence, to produce furniture layout plans for the user. It is an NP- complete problem and thus it grows with growth in the number of constraints and furniture elements. The proposed system aims at taking a step towards automation in planning of the room layouts. It requires basic input that includes size of the room followed by the position of doors and window from the any wall as per the user's convenience. The input goes through the various constraints and functions of crossover and mutation to present the best possible outputs for the user. Also, this type of layout generations can be extended to include the scope of township planning, motion sensor placement in smart environments and in the generation of realistic indoor scenes in floor planning. Another extension possible is the generation of a 3D model of the layout suggested by the system to give a better understanding of it. The system developed is efficient and since it offers multiple solutions, it makes its existence more significant.

REFERENCES

[1] Jun H. Jo and John S. Gero, "Space Layout Planning using an Evolutionary Approach", Artificial Intelligence in Engineering, Volume 12, Issue 3, July 1998, pg. 149-162

[2] Kristian Guillaumier, "Generic Chromosome Representation and Evaluation for Genetic Algorithms", Proceedings of Computer Science Annual Workshop (CSAW '03), November 2003, pg. 64-67

[3] MohdSaberi Mohammad, Safaai Deris, Safie Mat Yatim, Muhammad Razib Othman, "Feature Selection Method Using Genetic Algorithm for the Classification of Small and High Dimension Data", First International Symposium on Information and Communications Technologies, October 2004

[4] Ryota Tachikawa and Yuko Osana, "Office Layout Support System using Genetic Algorithm– Generation of Room Arrangement Plans for Polygonal Space", International Symposium on Nonlinear Theory and its Applications, September 2010

[5] Eugénio Rodrigues, Adélio Rodrigues Gaspar and Álvaro Gomes, "An evolutionary strategy enhanced with a local search technique for the space allocation problem in architecture, Part 1: Methodology", Computer-Aided Design, Volume 45, Issue 5, May 2013, pg. 898-910

[6] Brian L.Thomas, Aaron S. Crandall and Diane J. Cook, “A Genetic Algorithm approach to motion sensor placement in smart environments”, Journal of Reliable Intelligent Environments, Volume 2, Issue 1, April 2016, pg. 3-16

[7] Lap-Fai Yu, Sai-Kit Yeung and Chi-Keung, “Make it Home: Automatic Optimization of Furniture Arrangement”, ACM Transactions on Graphics (TOG) – Proceedings of ACM SIGGRAPH, Volume 30, July 2011, article no.86

BIOGRAPHY

Mitali Chavan B.E. (pursuing)

Nidhi Nirmal B.E. (pursuing) is a member of IEEE and IEEE computer society

Ritika Kumar B.E. (pursuing)

Shivani Rana B.E. (pursuing)

References

[1] Jun H. Jo and John S. Gero, “Space Layout Planning using an Evolutionary Approach”, Artificial Intelligence in Engineering, Volume 12, Issue 3, July 1998, pg. 149-162

[2] Kristian Guillaumier, “Generic Chromosome Representation and Evaluation for Genetic Algorithms”, Proceedings of Computer Science Annual Workshop (CSAW ’03), November 2003, pg. 64-67

[3] MohdSaberi Mohammad, Safaai Deris, Safie Mat Yatim, Muhammad Razib Othman, “Feature Selection Method Using Genetic Algorithm for the Classification of Small and High Dimension Data”, First International Symposium on Information and Communications Technologies, October 2004

[4] Ryota Tachikawa and Yuko Osana, “Office Layout Support System using Genetic Algorithm–Generation of Room Arrangement Plans for Polygonal Space”, International Symposium on Nonlinear Theory and its Applications, September 2010

[5] Eugénio Rodrigues, Adélio Rodrigues Gaspar and Álvaro Gomes, “An evolutionary strategy enhanced with a local search technique for the space allocation problem in architecture, Part 1: Methodology”, Computer-Aided Design, Volume 45, Issue 5, May 2013, pg. 898-910

[6] Brian L.Thomas, Aaron S. Crandall and Diane J. Cook, “A Genetic Algorithm approach to motion sensor placement in smart environments”, Journal of Reliable Intelligent Environments, Volume 2, Issue 1, April 2016, pg. 3-16

[7] Lap-Fai Yu, Sai-Kit Yeung and Chi-Keung, “Make it Home: Automatic Optimization of Furniture Arrangement”, ACM Transactions on Graphics (TOG) – Proceedings of ACM SIGGRAPH, Volume 30, July 2011, article no.86

[8] http://www.bis.org.in/sf/pow/powced.pdf: The list of standards by the Bureau of Indian Standards, Civil Engineering Department

List of Publications

Research paper published in Call for Papers, Volume 6, Issue 5, May 2017 of International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)

Acknowledgement

We remain immensely obliged to Prof. Varunakshi Bhojane for her continuous help and invaluable support in garnering resources for us. It is her guidance and supervision which made this project happen.

We also express our gratitude to Prof. Ranjita Gaonkar for the inspirations at the beginning of our project and also for giving her enlightening comments till the end of it.

We would also like to thank the H.O.D. of Computer Department, Dr. Madhumita Chatterjee for her invaluable support throughout the project and also for giving us the approval for it. It is because of her we got new insights and immense motivation to complete our project.

We would also like to thank our principal Dr. R.I.K.Moorthy for giving us this opportunity and trusting us to build a project that has no existing system.

Here we would also like to add that working on this project has indeed been a fulfilling experience for all of us. It has taught us to think out of the box. It also made us understand that anything big has a small beginning and one should be brave enough to initiate that small beginning and try their level best to make it as successful because at the end all that matters is the satisfaction of seeing knowledge turn into products and efforts turn into reality.

Mitali Chavan
Nidhi Nirmal Menon
Ritika Kumar
Shivani Rana