Int. J. Simul. Multisci. Des. Optim. 2014, **5**, A01

DOI: 10.1051/smdo/2013004

IJSMDO

Available online at:
www.ijsmdo.org

ARTICLE OPEN ACCESS

A genetic algorithm based optimization framework to visualize, evaluate, and modify 3D space configurations in Desktop VR

Magesh Chandramouli[1,*] and Gary R. Bertoline[2]

[1] School of Technology, Purdue University, Calumet 46323, USA
[2] School of Technology, Purdue University, West Lafayette 47907, USA

Received 15 August 2013 / Accepted 12 December 2013 / Published online 4 February 2014

Abstract – This paper presents the design and implementation of a Desktop VR (Virtual Reality) framework for generating and evaluating Pareto-optimal alternate 3D spatial configurations using GA (genetic algorithms). The 3-tier framework involves the generation of the Pareto-optimal plans using GA which are subsequently visualized first using a Java-based 2D Interface and finally in the form of a 3D VR scene. The search spaces (function domains) are extremely large in today's multifaceted interior design situations, and the optimization procedure involves conflicting objective functions, and limitations in the form of constraint functions. The interior space allocation problem is formulated and implemented as the "optimal configuration of artifacts". When using GAs, a group of Pareto-optimal solutions (Pareto set) are available for the planners and decision-makers, wherefrom one solution ought to be picked. Therefore, this study applies a tool to not only visually evaluate the plans, but also to interact with those plans to develop them further if needed. Besides enabling the optimal spatial configuration of the scene elements, this framework also facilitates evaluation and interaction via the 3D VR worlds. The framework aids the proactive exploration, analysis, and finalization of design aspects such as color, size, lighting, etc. of the various elements prior to the actual construction. The results demonstrate the robust performance of the GA and the final 3D VR environment with dynamic interactive capabilities. This final interface facilitates "GA-Compliant" transformations and scene modifications thereby facilitating the exploration and examination of alternative scene configurations.

Key words: Multiobjective optimization, Design optimization, Genetic algorithms, Desktop virtual reality.

1 Introduction

The optimal utilization of valuable interior spaces is impeded by the lack of efficient procedures to sift through the voluminous search spaces and support the generation/evaluation of alternate 3D spatial configurations. Only a very small fraction of the total search space is explored while solving interior space configuration problems involving MOO (Multiobjective Optimization). The design effort should pay critical attention to the notion of "the function" of the space i.e., the function this interior space is going to serve. From the seminal works of notable authors like Ballast, Pile, and Piotrowski [1–3] the importance of interior design becomes lucid and it is also evident that the interior space not only directly and indirectly influences our activities at work and home, but also influences several other daily activities such as shopping, dining, etc. One frequently overlooked, but fundamental element in the design of efficient interior spaces is the optimal arrangement of the artifacts. Vosinakis et al. [4] opine that effective interior space designs are typically purpose-driven, implying that the design is very mindful of the need of the clients. This innovative approach that integrates GA-based MOO with CG-based virtual scene rendering techniques facilitates design generation and evaluation. Systematic configuration of spaces and the arrangement of spatial components are absent or minimal either because it is considered trivial (which is incorrect) or due to the computational complexity involved in formulating and solving such MOO problems. Instead of following a logical procedure, the arrangement of artifacts in interior spaces continues to follow, by and large, a whimsical approach that has barely explored or evaluated the multitudes of options available.

There is a clear lack of the explicit articulation of objectives based on the "function" (purpose) of space. This is not to imply that planning is not at all involved in such interior spaces, but is not done in a methodical/systematic manner. While the service of interior designers is solicited for large-scale projects, the arrangement of the artifacts within the constructed spaces (e.g., libraries) rarely even tends to explore the "search space".

*e-mail: mchandr@purduecal.edu

The GA-based method exhaustively explores the search space to find the optimal solution. The rest of the paper is organized as follows: Section 2 is the literature review that addresses the related work. Section 3 explains the methodology involving the 3-tier framework for creating the Desktop VR scenes. Section 4 covers results and discussion and finally, Section 5 provides the conclusion.

2 Related work

Numerous examples of GA within computer graphics are found. Sims [5] applied evolutionary techniques to generate complex structures and textures. Notable works include those by Sims [5], Dawkins [6], Graf and Banzhaf [7], and Tood and Latham [8]. Chen et al. [9] employed GA to pick the best possible feature set for discerning computer graphics from digital photographic images. Yu et al. [10] used GA for automatically extracting features and processing stereo images. Nishino et al. [11] developed an interactive graphics optimizer using immune algorithms and continuing their work on using immune algorithms in Computer Graphics, Nishino et al. [12] developed an interactive 3D graphics modeler by simulating the human immune system.

Genetic algorithms were chosen for the MOO step in this study because of the following reasons:

(i) GAs have demonstrated higher probability of convergence within the parametric space;
(ii) GAs do not get trapped in the local optimum and try to find the global optimum;
(iii) GAs offer a diverse set of increasingly viable solutions (preservations of useful variations);
(iv) GAs have been efficiently employed for combinatorial optimization problems with extremely large search space of the order of 10^{100} or higher;
(v) GAs simultaneously evaluate many points in the parameter space, more likely to yield the global optimum GO;
(vi) By employing elitism, GAs try to retain the solutions with the highest fitness (so that the best solutions are not lost).

Stewart et al. [13] state that considering the inherent complex nature of modern design problems, the definitions of many objectives are not always nonlinear or additive. There are innumerable applications within the domain of construction graphics and interior design wherein MOO can be employed for solving problems involving combinatorial optimization [5]. Various MOO techniques including biologically inspired algorithms such as SA (Simulated Annealing), PSO (Particle Swarm Optimization), IWD (Intelligent Water Drops), CSS (Charged System Search), and ANT colony optimization were explored from the perspective of their suitability to this research. According to Balling et al. [14], one important reason that genetic algorithms have been exceptionally effective in solving design problems with multiple conflicting objectives and constraints is the contradictory nature of the two fundamental operations of crossover and mutation. Crossover intends to sustain the goodness in current generation, while, mutation aims to induce modifications or changes in the chromosomal structure (reverse effect).

Nevertheless, as GA commonly offers a pool of optimal Pareto plans or designs, advanced techniques are needed to evaluate and expedite the decision making process while reducing subjectivity, wherein visualization performs a vital role. Visualizing the candidate solutions of the design space is a lucid way to evaluate these optimal alternatives whilst minimizing abstraction. Therefore there is an imminent need for a tool to explore, analyze and evaluate the Pareto solutions. Chandramouli et al. [15] demonstrated that by using 3D visual scene renderings, planners who are experts in the fields of design planning can identify desirable or undesirable patterns. Karlen [16] states that "space planning is not a simple process involving a single category of information; rather, it is a complex dovetailing of several processes involving knowledge in many categories of information". Such information is of diverse nature ranging from Architectural Graphics applications by Ching [17] and Graphics Communication works by Bertoline et al. [18] to Kopec's work on Environmental Psychology [19], Kilmer's work on functional/visual design concepts [20], and Panero and Zelnik's work on human dimension [21].

When designing interior spaces, Rengel [22] precisely states that "... we claim space and subdivide in particular ways to suit the needs of the project". A judicious configuration of the artifacts is an essential component in the design of efficient interior spaces. According to Kubba [23], "an understanding of the use and allocation of space is the cornerstone in many design disciplines". Rengel [22] states that, "the number, size, and placement of large pieces of furniture, fixtures, and equipment, has a major impact on the interior space". Such earlier works have overlooked the "exploration and exploitation of search space" as evident from the detailed survey of software tools used by interior designers by Lok [24]. However, artifacts are largely arranged as and when the need arises and is dictated by the expertise of the person(s) involved in the process without necessarily following a standard protocol. This is in sharp contrast to the construction process which follows a standard procedure in accordance with a well laid-out set of rules, which must not be violated. Works by Maher et al. [25] and Whyte et al. [26] illustrate that computer based three-dimensional (3D) tools have evidently reduced the design time and facilitated concept evaluation.

3 Methodology

3.1 Research framework

There are three major components (Figure 1) in the design and implementation of the prototype for generating virtual representations for the alternate interior space configurations (Pareto plans). The first component in the 3-tier framework is the GA-based MOO process that results in the Pareto-optimal plans. The second part involves taking the numerical results of the GA-based computation and presenting them in the form of a Java-based 2D Graphic Interface. The third component

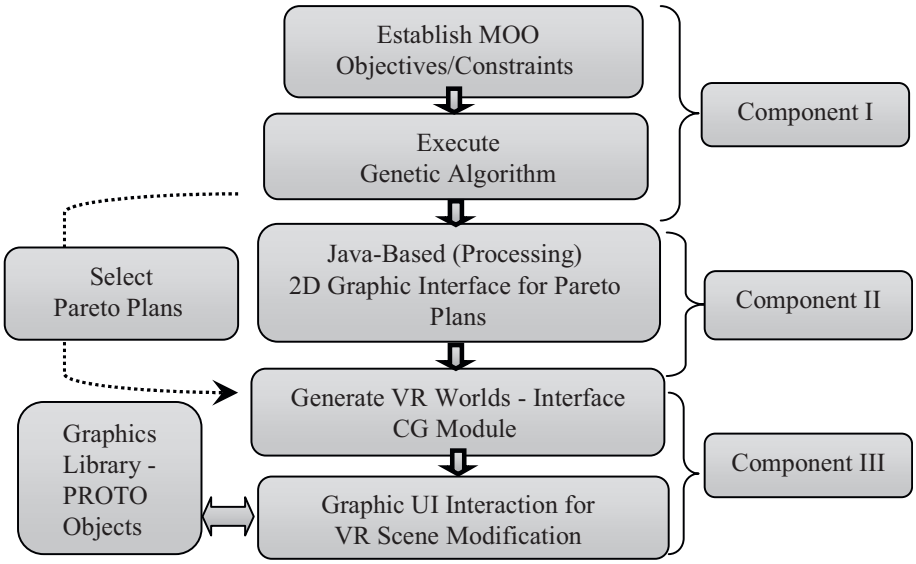

Figure 1. Framework for creating GA-based VR scenes.

involves the CG module wherein the virtual worlds are generated for viewing the plans in desktop Virtual Reality. The following sections provide a detailed description of the individual components of the aforementioned framework.

3.1.1 GA-based multiobjective optimization

Classically, a MOO is expressed as follows:

$$\text{Optimization } [f(x), or, y = (f_1(x), f_2(x) \cdots f_n(x))]$$
$$\text{Subject to } C(x) = \left(C_1(x), C_2(x), \cdots, C_p(x)\right)$$

n – Number of objective functions;
p – Number of constraints.

The set of values $(x_1, x_2, \ldots, x_n) \in X$ refers to the decision space and the set of values $(y_1, y_2, \ldots y_n) \in Y$ represents the objective space. The set of constraints i.e., $C(x) \leq 0$, limits the feasible set of solutions. GA-based MOO overcomes the limitations of traditional methods owing to its efficiency in solving non-linear, non-additive optimization problems without having to re-articulate or reformulate the design problem.

3.1.2 GA formulation

The phenotype in this study is the physical layout of the library and the appropriate genotype has to be formulated (Figure 2) for this space. The study area considered here is the floor space corresponding to a library. Also, other interior spaces such as a restaurant, a space, and office space are demonstrated to illustrate other areas of application. The objective is to generate an optimal layout, which is represented as a raster grid of cells. These cells in the raster grid (layout) are divided into categories. Within this genetic framework depicting the study area, every variable (changeable) zone is denoted by a "gene" (integer value). A linear array of values is used to represent the layout.

… C_{91}	… C_{92}	… C_{93}	… C_{94}	… C_{95}	… C_{96}	… C_{97}	… C_{98}	… C_{99}	… C_{100}
… C_{81}	… C_{82}	… C_{83}	… C_{84}	… C_{85}	… C_{86}	… C_{87}	… C_{88}	… C_{89}	… C_{90}
… C_{71}	… C_{72}	… C_{73}	… C_{74}	… C_{75}	… C_{76}	… C_{77}	… C_{78}	… C_{79}	… C_{80}
… C_{61}	… C_{62}	… C_{63}	… C_{64}	… C_{65}	… C_{66}	… C_{67}	… C_{68}	… C_{69}	… C_{70}
… C_{51}	… C_{52}	… C_{53}	… C_{54}	… C_{55}	… C_{56}	… C_{57}	… C_{58}	… C_{59}	… C_{60}
… C_{41}	… C_{42}	… C_{43}	… C_{44}	… C_{45}	… C_{46}	… C_{47}	… C_{48}	… C_{49}	… C_{50}
… C_{31}	… C_{32}	… C_{33}	… C_{34}	… C_{35}	… C_{36}	… C_{37}	… C_{38}	… C_{39}	… C_{40}
… C_{21}	… C_{22}	… C_{23}	… C_{24}	… C_{25}	… C_{26}	… C_{27}	… C_{28}	… C_{29}	… C_{30}
… C_{11}	… C_{12}	… C_{13}	… C_{14}	… C_{15}	… C_{16}	… C_{17}	… C_{18}	… C_{19}	… C_{20}
9 C_{01}	3 C_{02}	11 C_{03}	8 C_{04}	2 C_{05}	2 C_{06}	… C_{07}	… C_{08}	… C_{09}	… C_{10}

Figure 2. Grid representing physical layout.

Figure 2 illustrates the layout of the interior space in x and y dimensions. This is shown along with the genotypic representation using a single-dimensional array (Figure 3) The genes in the chromosome (G_1, G_2, …, G100) correspond to the cells in the grid representing the layout (C_1, …, C100). Index value represents the artifact element (Table 1). In both the genotype and phenotype in the (Figure 3) the "values" (integers representing the artifact type have been highlighted in green. It should be noted that the cells whose values are highlighted in yellow match (correspond between the linear integer array, the genotype and the layout, the phenotype). In the physical layout, the origin is at the bottom left.

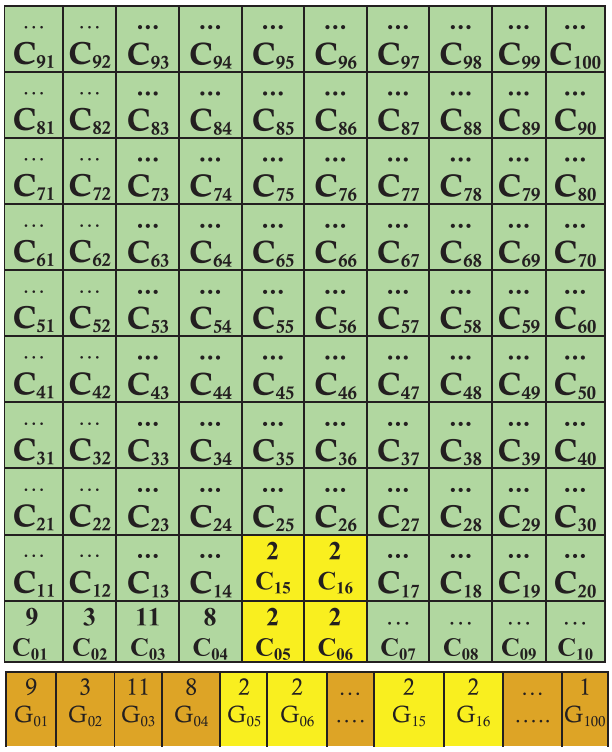

Figure 3. Encoding an artifact in a layout.

Table 1. Artifact elements and dimensions.

Index	Description	*X* dim.	*Y* dim.
1	IT lab	2	2
2	Washroom	2	2
3	Recreational space	2	2
4	Study Rm. Ind	1	1
5	Study Rm. Grp	3	3
6	Conf. room	4	4
7	Table/Ch. 1	1	1
8	Furniture 1	1	1
9	Furniture 2	1	1
10	Book-shelf	2	2
11	Ref. desk	1	1
12	Safety/emergency	4	4

The genotype conventions are as follows:

- G_n symbolizes the *n*th gene. Hence, G9 represents the 9th gene on a genotype with 100 genes.
- $G_{x\text{-}y}$ symbolizes the genotype component representing the *y*th occurrence of the artifact no. *x* (Index Value – *x*). Hence, $G_{1\text{-}2}$ symbolizes the 2nd occurrence of artifact number 1.
- min($G_{x:y}$) symbolizes the first gene of the genotype component representing the *y*th occurrence of the artifact no. *x* (Index Value – *x*). So, $G_{1\text{-}2}$ symbolizes the 2nd occurrence of artifact number 1.
- max($G_{x:y}$) symbolizes the last gene of the genotype component representing the *y*th occurrence of the artifact no. *x* (Index Value – *x*). Thus, $G_{1\text{-}2}$ symbolizes the 2nd occurrence of artifact number 1.
- C_n symbolizes the nth cell on the phenotype. Hence represents the 11th cell on the raster grid representing the actual layout of the interior space.

An artifact element occupying an area of 2 cells × 2 cells might seem continuous phenotypically, but on the genotype, they are not actually continuous (Figure 3).

3.1.3 Feasible set

Typically, for the first generation, each gene in the chromosome (representing the interior space) is designated a random value. One generation represents the size of the GA for a single run corresponding to 100 plans for the interior space layout. The suitability or "fitness" of each plan in the generation is calculated, using a fitness function. The plans that meet the constraints of the GA collectively constitute what is called the "feasible set". After the initial generation is obtained, the selection, recombination, and mutation processes are performed to create the subsequent generation. In this study, the cell location of the emergency facilities such as fire extinguisher and emergency door should not be changed. Hence, those artifacts continue to remain in their original location throughout the iterations. Also, every plan must have a minimum proportion of IT space and minimum amount of space for public utilities.

(a) Cells with index values 9 and 10 (emergency shelter and fire extinguisher) are not to be changed.
(b) The percentage of IT space and space for public utilities should have to be greater than 15% and 10%.

3.2 GA operations

3.2.1 Selection

The selection process signifies the choice of the individuals that are "fit-enough" for transmitting to the next generation and is based on the Darwinian notion of "survival of the fittest". Generally, this step is based on the "goodness" of a solution which is measured by the fitness of the individual. From the sorted pool of genotypes (Figure 4), the top 10 ($n = (Sel_{Rate} \times Pop_{Gen})$) highly ranked genotypes (with high fitness values) are selected for passing on the next generation. The Sel_{Rate} selected for the GA is 0.1 (10%). The remaining 90 genotypes ($n - (Sel_{Rate} \times Pop_{Gen})$) are created using the processes of crossover and mutation.

3.2.2 Crossover

The GA operation of recombination corresponds to the crossover (or recombination) process in the evolutionary cycle that involves combining the genetic information from two parent chromosomes (to produce the offspring. In this stage, "parent genotypes" are chosen and these are subjected to the crossover operation to generate children or "offspring" genotypes. To maintain the stochastic element in this process, a likelihood ratio documented in GA terminology as the "crossover probability" is employed. In this research, single point crossover is employed as explained here. The genetic material from

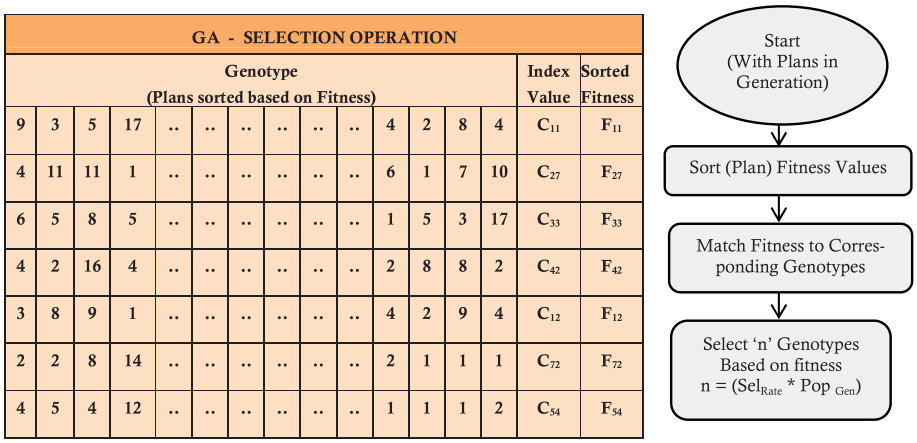

Figure 4. Fitness-based sorting of plans in a generation for selection.

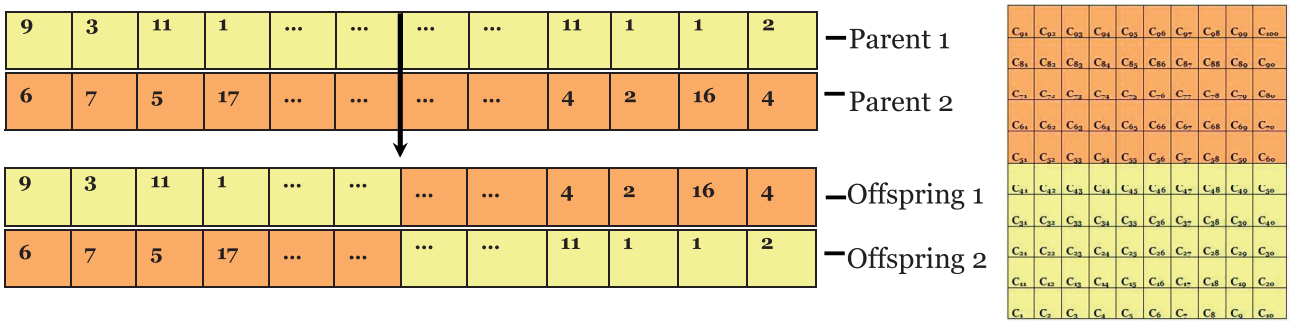

Figure 5. Crossover and resultant offspring genotype.

Parent 1 genotype (on the left of the crossover point) and from Parent 2 genotype (on the right of the crossover point) are joined to create Offspring1 (Figure 5).

The method of selection employed in this study for crossover is called tournament selection. In this type of selection, a small subsection of chromosomes from the parent generation is selected at first. From this the two chromosomes with the greatest fitness values are chosen. This method hence includes both a fitness based component as well as a non-fitness based component. Hence, it includes a stochastic element, while still considering the fitness of the parent chromosomes, which is why researchers consider this a blend of random as well as fitness-biased methods.

In the case of an artifact element occupying cells that are spread on more than a single row, the genotypic representation is not continuous. Hence, the most important consideration during the process of crossover is to prevent the splitting or breaking (of cells containing artifacts). Breaking refers to a single artifact element occurring at different locations, instead of being continuously represented at a single location. In order that unbroken artifact elements are obtained as a result of the crossover operation, other than performing the actual recombination between 50 and 51, care needs to be exercised in the GA formulation stage itself. For this reason, in the original chromosomes (representing the plans for the interior spaces) themselves, an artifact element ends before gene 50 or begins after 51 (Figure 6). Artifact element 6 (index value – 6) occupies four cells on the actual interior space layout (Table 1), the sub-section on the genotype representing this artifact element should occupy four genes.

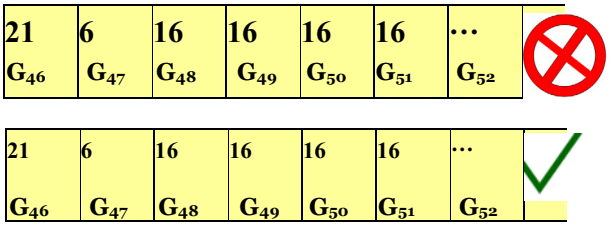

Figure.6. Correct and incorrect genotypes.

Let us consider the case when genes 48–51 represent this artifact element (G_{48}–G_{51}). During crossover, the other chromosomes 51st gene need not necessarily have the value 16. Consequently, the resultant plan of the interior space produced will have an area that is a mismatch for accommodating the corresponding artifact element. Thus, in the original chromosomes, by having an artifact element end before gene 50 or begin after 51. The general rule for any artifact element occupying more than one cell is as follows: $\max(Gx : y) < 50$ and $\min(Gx : y) > 51$, where x value can range from 1 to 12 (Index values of the elements)and the maximum value of y is the maximum number of times an artifact element occurs in

the layout. In generic terms, the above can be stated as follows:

$$\min(Gx : y) < \frac{n}{2} \quad \text{and} \quad \max(Gx : y) > \frac{n}{2} + 1,$$

where x value can range from 1 to $i_{\max}$ ($i_{\max}$ – no. of artifacts) and the max. Value of y, $\max(y) = \frac{n}{2}$, where n is the total no. of genes in the genotype.

3.2.3 Mutation

When the crossover operations are finished, the next generation has its population of chromosomes. At this point in time, random mutations are initiated in the population of chromosomes with a two-fold purpose: to prevent premature convergence and to enable the search process to diversify. These random mutations are induced at selected points (genes) located anywhere along the length of the chromosome. A value of 0.05 is selected as the probability for the mutation operation. A random value in the range of 0–1 is generated for each gene in the two offspring resulting in the crossover operation, the value of which determines whether mutation occurs or not. The integer value of the gene is subjected to change within the range of 1–12 if the mutation probability is higher than the random number so generated. Else, the gene is not subjected to mutation. The mechanism of the genetic algorithm rests on two fundamental operations namely selection and variation. Typically, when the GA proceeds with searching through the search space, it is desired to accrue the "good" traits or characteristics and prevent retaining the bad traits. In other words, it is required to preserve the good or "fit" individuals from generation to generation and eliminate those individuals or plans with poor fitness values (not suitable). Also, in order to facilitate obtaining diverse solutions, GA search process should be directed to explore the unexplored regions in the search space. All these are ensured by the GA operations of selection, crossover, and mutation.

3.3 Algorithms

The key algorithms (expressed as subroutines in the program) in the MOO framework are as follows:

3.3.1 Accidental re-allocation deterrent algorithm (ARDA)

During the process of creation of genotypes for a generation, a random function is used to generate integers in the range of 1–12 corresponding to the 12 artifact elements considered in this study. Following is the Matlab function used for this random generation of 100 chromosomes with 100 genes each: *randi* (12,100,100); However, as will be seen in the algorithms discussed subsequently here, there are other procedures that allot values to the cells based on the dimensions of the artifact and based on the constraints. This leads to the possibility that a cell whose value has been allocated (on a specific purpose) might be accidentally overwritten by a subsequent algorithm. In order to prevent this, the ARDA (Accidental Re-allocation Deterrent Algorithm) is implemented, which ensures that a cell whose value has been set on purpose by a specific function will not be accidently modified subsequently. From the earlier sections, it is known that every cell in the raster grid matches an artifact element which is represented in the genotype by the corresponding integer (index value).

Let us consider that for instance, an artifact element (index value x) occupies four cells (2×2), the genotypic depiction is not continuous as it is denoted by (G_6, G_7) and (G_{16}, G_{17}). ($G_6 = G_7 = G_{16} = G_{17} = x$). However, if later one of the cells preceding $G_6/G_7/G_{16}/G_{17}$ occupies more than one cell, then the subsequent corresponding cell will be overwritten with a value other than x. To prevent this from happening, besides the array containing the integer representing the index value for each cell in the genotype, another array of "dummy variables" is created. This dummy variable is given a default value of "1". When a gene value is changed "on purpose" by another function, say DVA explained below, then this dummy variable's value is changed to "2". Now, whenever a cell whose dummy variable value equals 2 is encountered it implies that this cell's value has been already set and should not be modified. In essence, the ARDA ensures the following: A cell's value can be changed only if $\text{dummy}_{\text{var}}$ = "1".

3.3.2 Dimension validator algorithm (DVA)

The actual function of this DVA is evident from the name of this algorithm. The purpose of the DVA is to ensure that all the genes in the genotype/all the cells in the physical layout are allocated the values in accordance to their dimensional attributes mentioned in the dimension table. In other words, if artifact × has dimensions $m \times n$, then DVA's role is to ensure that the corresponding number of genes in the genotype are allotted in accordance with the positional requirements. Earlier, it was noted that "However, if later one of the cells preceding $G_6/G_7/G_{16}/G_{17}$ occupies more than one cell, then the subsequent corresponding cell will be overwritten with a value other than x". The vice-versa is what DVA2 is concerned about. If a gene value, say G_{15}, is found to be equal to an index value corresponding to an artifact occupying more than one cell and the genes following it are not available for allotment G_{16}/G_{17}, then the value has to be changed to one corresponding to single-celled artifacts.

3.3.3 Constraint validator algorithm (CVA)

Once again, as the name indicates this algorithm ensures that the GA constraints are imposed. It was noted that cells with index values 18 and 19 (Emergency Shelter and Fire Extinguisher) are not to be changed. Throughout the MOO process, this CVA ensures that the corresponding cells/genes always continue to have the same values as required by the constraints.

3.3.4 Optional GA modifier algorithm (GMA)

This is an optional algorithm that should be executed if the same interior space project is carried with slightly modified layout. For instance, for the same library interior space allocation project discussed earlier, if for various reasons, it is decided that book shelves are located on the northern corner of the layout, then cells 91–100 need not be actually included in the GA process. This is in a way a modified CVA algorithm that ensures

that the MOO process is carried out according to this corresponding layout.

3.4 Fitness calculation and objectives

The following three objectives are considered:

1. Maximizing IT usability (labs, individual work stations),
2. Maximizing reading space (table/chairs, study rooms, and conference rooms), and
3. Maximizing public amenities (phones, ATMs, etc.).

Phenotypically, these objectives are innately conflicting since an enlargement in one space will lead to the shrinking of the space available for the other objectives. Also, to ensure that some minimum requirements are satisfied, the following constraints were imposed:

(a) Cells with index value 12 (emergency or safety features) are constant
(b) IT space and space for public utilities should be greater than 15% and 10%.

Balling's [27] Maximin function is used for calculating the fitness values of the genotypes. This function was originally applied for geospatial problems and has been employed in earlier studies involving spatial optimization [15, 28]. After taking into consideration several fitness evaluation strategies, this particular function described here has been found to be very suitable for GA studies involving a spatial component. The process of objective normalization is carried out using a basic scaling procedure. This is done using the formula shown in equation (1) for each objective k for a set of plans in a generation and then performing re-scaling subsequently. In equation (2), val_{ik} is the value of the objective k when assuming plan i, $\mathrm{val}_{\min,k}$ is the least value of objective k among all the plans in the current generation, and $\mathrm{val}_{\max,k}$ is the highest value of objective k among all the plans in a generation. The objectives of each plan were compared with other plans in the generation to find the fit ones in the current generation. Hence, when an objective of a plan i is compared with that of another plan j, plan j is better than plan i if the difference between j and i is positive (Eq. (2)).

$$\mathrm{Obj}_k(\mathrm{Plan}_i) = \frac{\mathrm{val}_{ik} - \mathrm{val}_{\min,k}}{\mathrm{val}_{\max,k} - \mathrm{val}_{\min,k}},\ k = 1, 2, 3\ i = 1, \ldots, 100$$

$$\mathrm{Obj}_k(\mathrm{Plan}_j) - \mathrm{Obj}_k(\mathrm{Plan}_i) = \frac{\mathrm{val}_{jk} - \mathrm{val}_{\min,k}}{\mathrm{range}_k} - \frac{\mathrm{val}_{ik} - \mathrm{val}_{\min,k}}{\mathrm{range}_k} = \frac{\mathrm{val}_{jk} - \mathrm{val}_{ik}}{\mathrm{range}_k}$$

$$f_i = 1 - \max\left(\min \times \left(\sum\left(\frac{\mathrm{Obj1}_j - \mathrm{Obj1}_i}{\mathrm{range}_1}, \frac{\mathrm{Obj2}_j - \mathrm{Obj2}_i}{\mathrm{range}_2}, \ldots, \frac{\mathrm{Obj}n_j - \mathrm{Obj}n_i}{\mathrm{range}_n}\right)\right)\right)^p.$$

The fitness of each plan in a generation is calculated with regard to the other plans in the same generation. Range corresponds to the difference between the maximum and the minimum values of that particular objective. Considering two plans $plan_j$ and $plan_i$ in a particular generation, $plan_i$ is dominated by $plan_j$ if *$plan_j$ surpasses the other plan in all the objectives*. The fitness is calculated as explained in equation (3). Based on the fitness formula described above, the Pareto-optimal plans were selected from a generation based on the fitness values obtained. While dominated plans had a fitness value between 0 and 1, Pareto-optimal plans had fitness values greater than 1 [14, 27]. Besides, in this study, a higher value for p is employed. This value was chosen to pursue Pareto-optimality more vigorously. The fitness of plans with fi more than 1 gets higher, and the fitness of those plans with fi values less than 1 gets further lower.

3.5 Pareto-optimality and visualization

A brief discussion of the notion of "Pareto-optimality", especially Pareto-optimal plans is very important from the perspective of this study on the whole. The GA iterations engenders a set of optimal solutions (Pareto-optimal set), which signifies the pool of solutions that denote the optimal trade-off [13] for the interior space problem with the three objectives subject to the constraints. For a plan to be considered as a Pareto plan it should not be inferior to or underperform any other plan from that generation. Stated otherwise, a plan may outclass the Pareto plan when considering one objective and a different plan may be better with regards to another objective; however, a "single plan" must not outdo a Pareto plan in all the objectives considered for the MOO. Balling, Taber, Brown, and Day [27] clearly state that the "*Pareto set is independent of the relative importance of the objectives*". The above discussion evinces that plans do not belong to the Pareto set (non-Pareto plans) are "dominated", as a Pareto plan that is better (or that which dominates) has been found. The fitness function used in this study is the one proposed by Balling et al. [14, 27].

However, the challenge now is to select one solution or plan for implementation. For any design optimization problem, a single satisfactory solution that can be implemented is required. As suggested by Chandramouli et al. [28], the use of genetic algorithms for design optimization entails meticulous scrutiny of the differences among the candidate solutions to obtain a better knowledge of the basic processes and the fulfillment of the objective functions. The process of choosing one single solution over others entails exhaustive domain knowledge. As noted by Seixas et al. [29], typically, many GA-based design optimization procedures make the final choice of the solution (from the Pareto set) based on some "higher level information".

However, when it comes to design and planning, it is not possible to implement all the conflicting solutions in the Pareto set (resulting from the GA process). Hence, a virtual reality based visualization framework is generated to explore and study the alternative solutions.

3.6 Visualizing GA results

As explained by the previous section, while using GAs for MOO, the procedure literally stops with the pool of the Pareto solutions and thereafter, one solution from the Pareto set is chosen based on some "higher-level or expert knowledge". Arguably, these involve several subjective measures and this seriously undermines the efforts to objectively select a plan without prejudice from the various stakeholders involved. Therefore there is an imminent need for a tool to explore, analyze and evaluate the Pareto solutions. Desktop VR-based graphic visualization can meet such need by facilitating not only presented information, but also enabling seeing and understanding of hidden information among datasets. Connolly [30] defines virtual reality as the application of an artificial environment generated by computer technology to simulate some targeted activity. However, directly creating virtual representations from the results of the GA process is a tedious task. Hence, a processing (Java-variant) 2D graphic interface is used to obtain the numerical array from the GA and output a 2D Graphic layout, which then serves as a reference for building the virtual worlds.

3.7 2D Graphic interface

In order to facilitate the transition from the numerical GA results to virtual representations, a 2D Graphic Interface is built in this study. For generating the 2D graphic interface, a Java-variant called Processing is employed. Processing, an open source programming language built by Reas and Fry [31], is largely used for graphic programming. The Pareto-optimal plan obtained from the GA-based iterations is a layout representing an optimal configuration of artifacts in the interior space. This is in the form an array of integers. This integer array contains 100 genes corresponding to the 100 cells in the raster grid representing the interior space layout (Figure 7). However, when constructing the 3D virtual scenarios corresponding to the Pareto-optimal plans, using this numerical array of integers can be extremely cumbersome.

Hence, the numerical array (Pareto plan) is provided as the input in the processing file (.pde), which produces an output in the form of a graphic layout. Figure 8 shows the iconized graphic interface for another Pareto plan. This iconized graphic interface can now be used as a reference for the 3D Virtual world scenarios depicting Pareto plans.

3.8 Virtual worlds for visualization

The tool used for visualizing the scenarios is the virtual reality modeling language (VRML/X3D) specifications laid out by Ames et al. [32]. The primary reasons for using this tool are the web advantage it offers and the programmable external

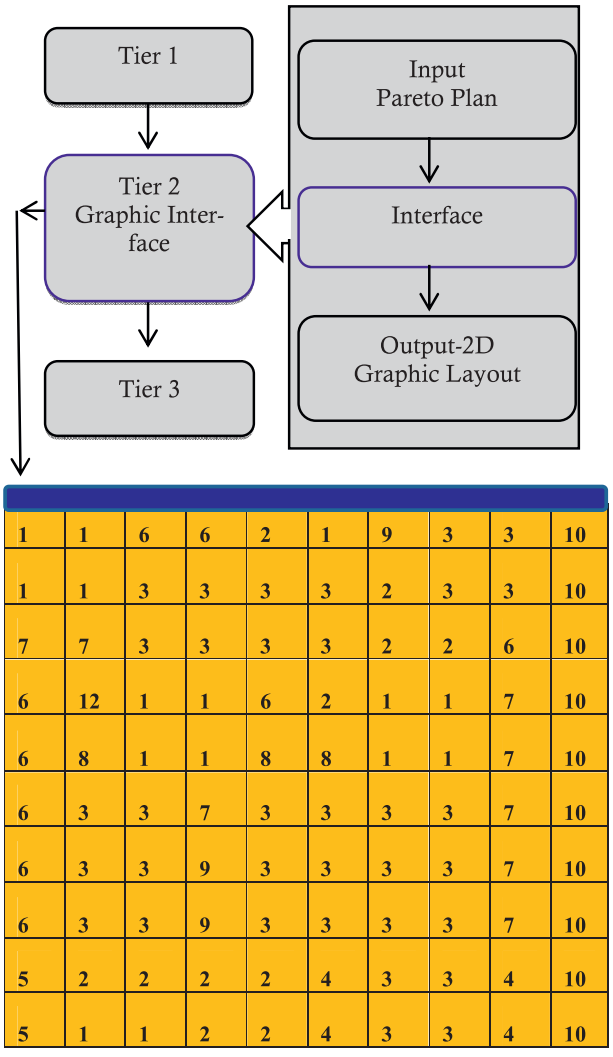

Figure 7. Tier II: Graphic interface.

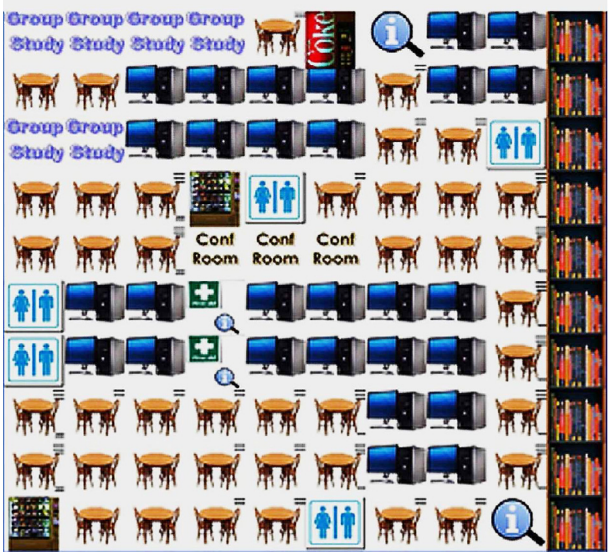

Figure 8. Tier II: Iconized graphic interface.

authoring interface (EAI). VR scenes thus created can be hosted online directly without much further modification. Users can view these desktop virtual environments on their computer using standard web browsers enabled with simple, open-source VRML plugins. Also, these are easily convertible into the latest web standard for 3D worlds, x3D.

Using a VR-based representation offers immense advantages. The users can navigate within these virtual worlds, move the objects in the worlds, rotate or scale them, and transform them in multiple ways. These virtual worlds facilitate user interaction with the 3D objects and provide a sense of immersion. In this study visualization is used as a tool to evaluate the Pareto plans objectively.

3.9 Virtual world generation

The scene is defined by "nodes" (X3D/VRML). A visualization scene can be considered to be composed of objects with properties. In the preceding sentence, the phrase "objects with properties" is highlighted or emphasized, since these properties determine how an objects looks and/or behaves. Considering a sample scene, say a library, as considered in this study. If further broken down into smaller fragments, the elements that result include tables, chairs, lamps, washrooms, etc. The furniture may be of a particular material, color, and dimensions. All these are the attributes of the furniture. Similarly, each element has its own characteristic features or attributes.

A scene can be viewed as being composed of elements or objects, each of which has its own properties or attributes. A parent object can include any number of children, which can be grouped or assembled to function as one single entity. This sort of hierarchical arrangement helps in the step-by-step design of the object and also understanding the framework at any later stage. A scene-tree construction is used in virtual scene renderings. The root or the parent object consists of whole scene grouped together and all the other components are grouped under this parent object using "parent-child" relationships. Individual scene elements corresponding to each floor type were created positioned according to their corresponding positions as per the Pareto plan obtained in the previous step.

For complex objects including multiple parts, various object parts are grouped to form parent objects, leading to complete objects that are combined and re-positioned to create the final 3D scene. Another advanced way of customizing the virtual objects is by the use of PROTO nodes, which the following section addresses.

3.10 Virtual objects and customizable nodes

Virtual world objects are described as shapes with geometry and appearance. All features such as buildings, rooms, artifacts, etc. can be modeled as shapes which can be grouped together and transformed (translated or rotated) within the coordinate system within which they are built. A vast number of VR objects, however complex they might be, are built using the fundamental shape node with the principal fields namely geometry and appearance. The geometry field is used to describe the geometric properties of the object and the appearance field is used to describe how the object looks. In order to be able to develop a framework that can interface the GA results with a virtual environment, the objects in the virtual environment should be mutually compatible with the GA. This implies that the scene or the objects that are formulated in the GA should be in a form that can be represented and programmed within the virtual environment (GA-compatible). A significant step towards this goal involves generating customizable nodes, the PROTOS. PROTOS represents reusable scene objects and these can be referenced from within the file and externally (EXTERNPROTO). The PROTO library created as part of this study can serve as an extremely useful addition that can be extended into several other object oriented applications as well. The framework also provides functionalities to dynamically change graphical attributes such as diffuseColor /specularColor/ transparency to generate various appearances of the same object and also dynamically manipulate the size of the object.

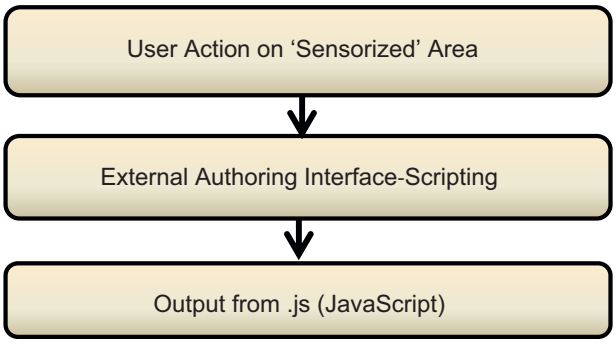

Figure 9. I/O sequence (event-response) framework.

3.11 Programmed interface: user interaction

One of the primary reasons for implementing the graphic library using PROTOS is to program the objects created. The exposedFields can be compared with Input and Output jacks and can be wired to create "ROUTES". Figure 9 below demonstrates the Input/Output flow corresponding to this. The key attributes of a graphics scene include geometry and appearance. There are numerous fields and exposed fields that cumulatively control the shape and appearance of objects.

By routing user events through appropriate Input and Output jacks, these fields can be controlled "dynamically" to manipulate geometry and appearance, such as these:

- Dynamically changing diffuseColor/specularColor/ transparency for alternate appearances of object(s);
- Dynamically changing the size of the object.

While these can be achieved using commercial software, these are numerous restrictions to what can be done within such IDEs. However, these programmed objects offer innumerable possibilities whereby their behavior and attributes can be controlled. By way of creating custom-made PROTOS and EXTERNPROTOS, the behavior of the scene and the constituent objects can be controlled very precisely to generate desired actions. Next, controlling the position and orientation by manipulating transformational attributes is discussed.

The virtual world/scene is divided into pixels/voxels. Objects are embedded within these grids (2D/3D) with built-in sensors to recognize user actions. When the user clicks within the perimeter/boundary of the pixel/voxel, the position (geometric center) is highlighted. The highlighted object always exists in the scene, however, initially in the un-activated state (visibility set to zero). Subsequently, using JavaScript functions, the user click event is used to activate the object (by modifying the material node's transparency value). In order to facilitate the user clicking, a buffer is provided whereby the user can clicking within a radius of the geometric center of the grid and the object is still selected. Ultimately, the crux of this interaction and manipulation involves building sensors and programming them to sense user actions.

Once the virtual scene environments have been built for the corresponding Pareto-optimal plans, further modifications without affecting the constraints can be performed as shown above wherein the decision makers can apply changes to attributes including position and appearance to "envision" alternative scenarios. This offers a tremendous capability whereby once a VR design is generated and the user wants to manipulate the final design. Some parts are programmed to be translated or rotated or scaled, while others are fixed or stationary. The GA constraints require them to be at that fixed position (for design criteria) and hence these cannot be changed.

Earlier sections covered in detail the methodology for the design and implementation of the GA-enabled visualization framework to generate design scenarios involving the arrangement of artifacts in interior spaces. The framework involves three major elements namely GA-based MOO, a 2D graphic interface, and finally a VR-based visualization environment. This discussion of the results of processes of MOO and visualization is especially from the perspective of the interior space application.

The GA-based MOO in conjunction with the VR-based visualization described previously was tested on a common interior space e.g., library. The interactive and navigable virtual worlds generated are discussed in detail here. Such visualizations can efficiently illustrate design scenario(s) than a conventional (paper-based) or PC-based 2D scene representations. However, unless the GA data is converted into the 3D format it is not of significant use to planners and decision makers, since analyzing voluminous numerical data and inferring from them is a cumbersome task, which may not ultimately yield the desired results. Visualization greatly facilitates understanding the design scenario and comprehending its function from diverse perspectives; this is especially important when multiple stakeholders with varying interests are involved.

4 Results and discussion

4.1 Important considerations while interpreting results

As mentioned in the earlier sections, the "primary focus" of this study is the amalgamation of the GA-based MOO with the Desktop VR based CG component to design and implement an innovative framework to generate alternative interior space configurations. However, the essential innovation involves the actual integration of GA with CG. The emphasis here is the demonstration of the usefulness of the visualization scenarios resulting from this integrated framework and the analysis of the environments as a tool to facilitate "informed decision-making". From this point of view, the GA metrics are not as significant as the process of interpreting the visual results. It was explained in the literature review that earlier research applying GA-based MOO to the domain of interior space design using a framework similar to that of this study could not be found; this is one of the major reasons justifying the innovative aspect of this research.

From the above discussion the need for extensive research in this area of "GA metrics for MOO in interior space planning" is evident. This in itself can be a subject matter deserving extensive research spanning over another graduate research. Hence, even though this section discusses the GA metrics, the focus is essentially to elucidate the visualization results and demonstrate the usefulness of the research outcomes in generating as well as evaluating alternative scenarios. Hence, the essential aspects of the GA parameters and GA metrics are discussed subsequently. The functionalities of the visual interface for viewing are demonstrated using not only the library setup, but various other possible interior space scenarios are used to demonstrate the various capabilities of the desktop virtual environments generated for this study. But, before looking at the visual interface, let us look at the GA-based MOO component.

4.2 Crucial GA operations for experiments

At the core of the GA based optimization are the important processes of crossover and mutation. Typically, the term "GA parameters" refers to the probabilities of crossover and mutation. However, before getting to the crossover probability and the mutation probability, the method of selecting the parent individuals for crossover needs to be discussed. The type of selection employed in this study for selecting the parent chromosomes for "mating" or crossover is called "tournament selection". This method involves selecting the parent chromosomes from a smaller subset of individuals from a generation. Typically, there is a stochastic element involved in this selection of the few chromosomes to serve as the pool for "parents-to-be". From this subset, the selection of the parent is classically based on its fitness value. Hence, the process of tournament selection is, in a way, characterized by stochastic as well as deterministic traits. One important feature in this process of tournament selection is the tournament size. When the tournament size is increased, the probability that an unfit chromosome (individual with lesser fitness value) is selected gets minimized. The tournament size employed in this study is 5. At random, five plans are selected from the parent generation. From this subset, two plans with the highest fitness values are selected as parent individuals for the subsequent step of crossover (mating). This is implemented as follows:

1. Random set containing five integer values are selected,
2. The five integer values are used to select the subset of individuals,

3. From this subset, individuals with highest fitness are selected as parents.

To prevent recurrence, a sub-routine (function) is executed in the program to ensure that the same chromosome is not selected again, as this would result in the children chromosomes being the same as the parents. As noted earlier, single-point crossover was employed for the reproduction process. Splitting refers to the case when a single artifact element occurring at non-continuous locations. The crossover point (after gene 50, G_{50}) represents the location where the portions are interchanged to generate the offspring.

The values (genes) in the first offspring before the crossover point are identical to the parent Individual no. 1 and the values after the crossover point are identical to the values in the parent individual (PI) no. 2 (51–100). The first part of the second offspring has values similar to PI2 and the other part replicates PI1's values. Subsequent to the reproduction operation, the chromosomes in the resultant population must be subjected to mutation. The mutation operation is carried out by comparing the result of the random value generated with the probability of mutation. In this study, a mutation probability of 0.05 is employed. In the natural evolution, mutations are considered to be a sporadic or rare occurrence and hence the mutation probability is assigned a low value. A random value (Ran_{Val}) in the range of 0–1 is produced corresponding to each gene in the two children chromosomes (offspring). The "crux" of the mutation process involves comparing this value hence generated with the "probability of mutation" (Mut_{prob}). If ($Mut_{prob} - Ran_{val} > 0$), then the integer value corresponding to the gene is changed to another random value between 1 and 12. Else, it remains non-mutated or integer value corresponding to the gene is not changed. It should not be the other way ($Ran_{val} - Mut_{prob} > 0$), as the probability of mutation is very low and then this may lead a very high frequency of mutation which contradicts the natural process of mutation. The process of mutation executed needs to be "constrained". This implies that the mutation process should not result in the "splitting of artifacts" mentioned earlier and it should not allot elements in the reserved cells (for safety features/emergency purposes).

4.3 GA validation

Kemenade et al. [33] examined the role of the parameters in the GA operational efficiency, their impact on the outcomes, and the various schemas for GA operations. Parameters of the GA such as the Generation size, tournament size, and the probabilities of crossover and mutation have a strong correlation with the overall performance of the GA. In this study, the GA was executed for varying values for the probability for mutation and tournament size and finally a medium tournament size of 5 and a low mutation probability (0.05) is employed.

The GA results are explained here from the perspective of the fitness of the generations and "Pareto-optimality". For repeated runs of the GA for at least 10 times, the average fitness of the generations showed consistent improvement explaining that the GA has significantly improved the fitness of the individuals. This implies that the final plans, whilst satisfying the constraints, have maximized the objectives incorporated within the GA. The result of the GA iterations is the Pareto-optimal set, which consists of the non-dominated solutions. However, practically only one solution can be implemented and may be more suitable for the specific problem on hand when taking in to consideration other factors that were not included in the GA. When there is an inventory list containing 12 elements, the random generation of the chromosomes cannot be expected to produce a gene allocation that matches the artifact attributes (or dimensions). Hence, a step is implemented before the actual execution of the genetic algorithms known as "selection of feasible solutions" is implemented to ensure that the genotypic representation corresponds to the realistic representation.

4.4 Visualization results

The GA results are input into a .pde file (Processing program) which converts these numerical results into the cells of a graphic interface (reference). Hence, the numerical array (Pareto plan) is provided as the input in the Processing file (.pde), which produces a output in the form of a graphic layout, which can now be used as a reference for constructing the 3D Virtual world scenarios depicting the Pareto-optimal plans. The floor is superimposed over this layout and the transparency value of the floor is adjusted so as to reveal the layout. In order to facilitate this process of further generating the virtual worlds from this 2D layout, the x, y, z axes are used initially as a reference and also a grid is superimposed over the scene to facilitate for properly situating the elements within the VR program code.

A virtual environment or a scene is composed of various scene objects, where the scene itself can be considered to be the root object. All the objects can be envisioned in the form of "parent-child" associations. Besides, as mentioned earlier, to be able to develop a framework that can facilitate visualizing the MOO results and further interacting with it generating customizable nodes, the PROTOS, is a crucial step. Table 2 illustrates selected artifact elements on the layout space and what virtual scene elements are corresponding to these index values. A more complex virtual artifact element leads to greater branching in terms of the scene hierarchy.

When evolutionary procedures are employed for multiobjective design optimization, it is imperative that the differences among the candidate solutions are meticulously evaluated to consider their suitability to the actual problem on hand. More importantly, it is not practically possible to include all possible or desired features in the form of objectives or constraints. A whole set of factors that have not been expressed as either objectives or constraints can be evaluated using the visual medium. Visualization empowers the decision makers to view and comprehend even hidden or ineffable information. The 3D virtual scenes besides enabling navigation also provide a sense of immersion whereby the planners can explore and evaluate the designs (Figure 10).

This study provides interactive capabilities within the virtual environment. In this research, Desktop VR is used for

Table 2. Index values of selected elements and scene hierarchies.

Artifact classification virtual scene	Level 1 of scene hierarchy	Level 2 of scene hierarchy
Index value – 1	IT lab	1. PC terminals 2. Furniture 1
Index value – 3	Refreshments/Rec. space	1. Vending machine 2. Snacks machine
Index value – 5	Group study room	1. Furniture 1 2. Table.Ch.1
Index value – 11	Ref/Info desk	1. Furniture 1 2. Table.Ch.1
Index value – 12	Book-shelf	1. Shelf

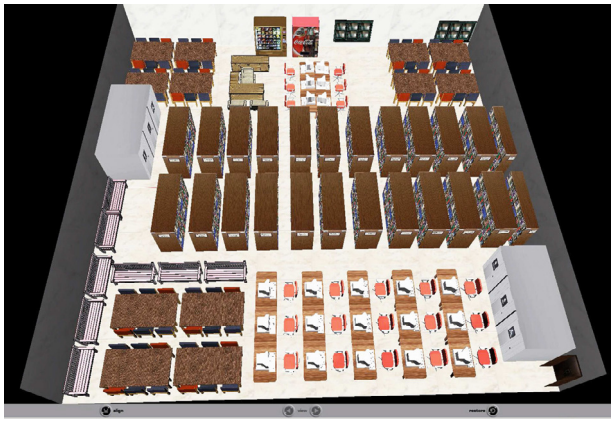

Figure 10. Desktop VR scene for a Pareto plan.

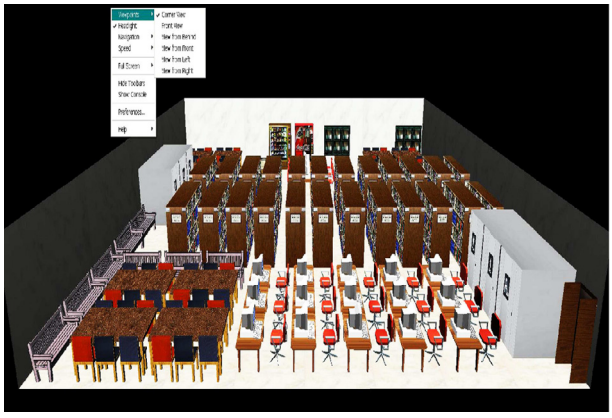

Figure 11. Multiple POVs (points of view).

the visualization component. By using 3D visual scene renderings, planners who are experts in the fields of design planning can identify desirable or undesirable patterns. It is very important to interact with the VR environment to perceive the unique aspects of the plan under study. The "point of view" is a feature that indicates the use of visualization to study the same scenario from different points of view (Figure 11).

The viewpoint is a preprogrammed spot within the VR environment that takes the viewer to the specific position and orientation wherefrom the scene can be viewed. This is very helpful when evaluating a scene with respect to a specific problem, so that the pros and cons of the design can be studied in detail. Also, another salient advantage while programing virtual scene representations is the capability to build the same by employing varying levels of detail. For instance, when viewing from a distance, the bigger or finer details need not be rendered for the smaller objects that are not quite as obvious. This notion can be used to efficiently model the scene so as to facilitate faster and smoother scene rendering. The virtual environment can be rendered in accordance with the viewer's position with the scene. Table 3 highlights few salient features provided by VR settings and explains how the VR visualization of Pareto plans enhances evaluating the various plans with respect to the objectives.

4.5 Interaction with the virtual scene

Virtual environments can be rendered interactive using different techniques. The EAI (External Authoring Interface) provides a window to access the nodes that constitute the various shapes within the virtual scene.

Typically, a VRML based virtual environment is composed of scene elements that can be broken up into nodes that constitute these shapes. These nodes (in these shapes) control how a scene object looks by programming their geometry and appearance. Interaction necessarily involves a change of state of the objects involved. The node that has to be selected and modified depends on the type of interaction desired. While the capabilities of the standalone virtual environment are limited, a plethora of possibilities can be opened up by employing the External Authoring Interface (EAI). Two important functionalities used in this study, from a design perspective, include "sensors" and "EAI". Sensors refer to the ability of a virtual environment to "sense" and respond to user actions. The primary means of interaction for a user with a desktop virtual environment involves the mouse and the keyboard. Examples of common mouse actions are as follows:

- Hover,
- Left-click,
- Right-click,
- Drag.

Theoretically, innumerable alternatives exist for a design problem, which is why an efficient tool such as GA is used to narrow down on a subset of solutions. It should however be noted that it is difficult to say that with 100% accuracy that the solution obtained is the "best" solution. This is so because

Table 3. Salient advantages of VR scene visualization of Pareto plans.

Feature	Advantage
Point of view (POV)	The scene can be evaluated from innumerable points of view.
Levels of detail (LOD)	The same scene can be created and viewed using varying levels of detail.
Navigation	Different browser/plug-in combinations offer various methods of navigation such as study, flying, and walk-through.
Controlling scene attributes	The scene attributes and the attributes of the VR scene objects such as Geometry Appearance can be controlled in multiple ways.

the final solution is the result of the interplay of a whole lot of factors such as GA formulation, constraints applied, various GA parameters such as rate of selection, probability of mutation, fitness function, and so on.

A different set of values for the aforementioned characteristics might lead to totally diverse set of solutions. Let X denotes the complete search space for a problem on hand. For design problems, depending on the number of various factors including objectives involved, the nature of objectives (e.g., conflicting), and constraints imposed etc. the search space may be enormous. The rationale behind using tools such as GA is to narrow down and identify the "optimal" solution.

Assuming that a robust GA has been employed with suitable fitness function and appropriate parameters, the user is still left with a pool of solutions (Pareto-optimal plans). From the above explanation it follows that for a different combination of factors, a different solution (or set of solutions) could have been attained. Also, most literature refer to terms such as "expert knowledge" or "domain-specific knowledge" or "higher level information" [29] that plays a crucial role in the selection of the final plan or design for implementation. This, quite frequently involves "intangible" or "ineffable" factors in the sense that some aspects may become obvious to the expert upon visualizing the design. Whilst, this can always be "expressed" once it has been "envisioned", it may be difficult to express prior to that.

The ultimate purpose of all these tools is to facilitate the process of "envisioning" the best or optimal design. The visual representation generated portrays a possible solution to the problem considered. However, when this visual environment lacks interactive capabilities, it is a major handicap as the user or the expert is not able to envision "What if" scenarios. It is extremely important that any transformation (translation or rotation) can occur only within the "constraints" already set in the GA. A transformation should not violate the GA constraints; else the entire process of MOO using the GA will be undermined. The following pseudo-code expresses this notion:

```
do{
// Object transformation within the Virtual Scene
} while (GAConstraintsMet == true);
```

In order to ensure that the conditions are always met, the virtual environment (and hence, the objects within the virtual scene) are programmed as follows:

- Not all objects are allowed to translate. Repositioning is enabled only for select objects within virtual scene.
- Areas where objects cannot be moved into are programmed in such a way that a "visual warning" (a transparent red object) is displayed when any infringement occurs.

The use of scripting here is to identify specific areas within the virtual environment that should not be transgressed or infringed upon (as the GA constraints are so formulated). In this process there are specific mouse activities that are performed such as hovering, clicking etc. When the mouse hovers upon the a specific scene area that has been categorized as a "restricted zone" (e.g., emergency exit as mentioned earlier), this area should undergo a change of state that is "visible". In other words, if the restricted area gets highlighted in a different color (e.g., red/transparent red), that indicates that the user should not move any object into this colored area.

In order to get this effect, a shape object should be embedded within this area and the actual work occurs in the MATERIAL node of the shape object. Let us assume that "Box" geometry is used to indicate the demarcated area. The MATERIAL node of this object should change the color to red when it "senses" mouse movement. This implies that, when the user hovers the mouse over this area, the object (upon sensing the mouse) should change the color of the Box object from "invisible" to "red". The state of invisibility is accomplished by turning the transparency to 100%. Inside the code snippet this is represented between values of 1 and 0, 1 being 100%. Any value in between is represented using corresponding decimals. In this study, JavaScript is used for scripting. A whole lot of interactions can be done with the virtual scene hence generated. These include transformations as well as modifying appearance related attributes.

On the whole, all the modifications can be classified either as geometry related modifications, appearance related modifications, and position related modifications. The last one also includes modifications related to the orientation and scale. Modifications to position, orientation, and scale are all grouped as transformational changes. In the virtual environment, the color that is seen or is visible finally is the result of various programmatic elements such as color, transparency, ambient intensity, etc. These can be controlled using a menu-driven environment, in which visible sensors can be activated to control the various values such as the transparency, color, shininess etc. This can be especially useful for designers focusing on the various design aspects involving light and those who study the impact of applying various -modulations to the light and color in the interior environment. Also, the ability to vary the intensity provides a valuable functionality to determine the most

suitable form and intensity of lighting for the specific environment under study.

5 Conclusion

This research integrated the notion of GA-based MOO within the discipline of interior space planning. he results evinced the robustness of the GA and its efficiency in generating the Pareto plans. The study goes beyond presenting the results in the form of a Pareto set with a pool of candidate solutions, by providing a means to visualize potential solutions using a virtual reality imaging, and by experimenting with those plans through an active interaction capability. Experimenting indicates producing alternative representations corresponding to a different set of appearance and geometry variables and the subset of variables influencing the final VR Scene. The subset of variables mentioned above includes a wide range of values such as diffuseColor, transparency, specularColor, etc. Visualizing and manipulating the plans in this manner provides a means of more effectively exploring and navigating the potential solutions. This greatly facilitates the practice of informed decision-making and selecting the optimum plan, and allows for the inclusion of many more stakeholders in the final decision process.

The generation of the graphic designs is based on a web-friendly framework. Designs are generated using OpenSource software that are web-compatible so that the results can be hosted online (World Wide Web). For this reason the research has accomplished all programming procedures using web-compatible OpenSource software such as VRML, Java, and JavaScript. In order to more up-to-date with the web standards, the code snippets can also be easily converted to X3D if it is so required. This conversion is a seamless and straightforward process as X3D is considered the successor to VRML and is the XML-based 3D authoring language. Programming the EAI (External Authoring Interface) of the virtual worlds (.wrl or .x3d) with Java and JavaScript opens up the key graphics attributes include geometry and appearance.

References

1. Ballast DK. 2002. Interior construction & detailing for designers and architects. Professional Publications: Belmont, CA.
2. Pile JF. 2007. Interior design, 4th edn. Prentice Hall, Upper Saddle River, NJ.
3. Piotrowski CM. 2002. Professional practice for interior designers, 3rd edn. John Wiley & Sons: New York.
4. Vosinakis S, Azariadis P, Sapidis N, Kyratzi S. 2007. A virtual reality environment supporting the design & evaluation of interior spaces, 4th INTUITION Conf. on Virtual Reality & Virtual Environments, Athens.
5. Sims K. 1991. Artificial evolution for computer graphics. Computer Graphics, 24(4), 319–328.
6. Dawkins R. 1986. The blind watchmaker. Longman: Essex.
7. Graf J, Banzhaf W. 1995. Interactive evolution for simulated natural evolution, artificial evolution, in European Conf. (AE'95) Selected Papers, Alliot JM, Editor. Springer-Verlag: Berlin, Germany. p. 259–272.
8. Todd S, Latham W. 1992. Artificial life or surreal art? Proceedings of 1st European Conf. on Artificial Life. MIT Press, Cambridge, MA. p. 504–513.
9. Chen W, Shi YQ, Xuan G, Su W. 2008. Computer graphics identification using genetic algorithm. IEEE International Conference on Pattern Recognition (ICPR08). Tampa, Florida, USA.
10. Yu TT, Yang M, Chen CS. 2005. Automatic feature extraction & stereo image processing with genetic algorithms for lidar data. Proceeding of Computer Graphics, Imaging & Vision: New Trends, 307–309.
11. Nishino H, Sueyoshi T, Kagawa T, Utsumiya K. 2007. An interactive graphics rendering optimizer based on immune algorithm. Lecture Notes in Computer Science, Applications of Evolutionary Computing, Springer, 4448, 459–469.
12. Nishino H, Sueyoshi T, Kagawa T, Utsumiya K. 2008. An interactive 3D graphics modeler based on simulated human immune system. Journal of Multimedia, 3(3), 51–60.
13. Stewart TJ, Janssen R, van Herwijnen M. 2004. A genetic algorithm approach to multiobjective l & use planning. Computers & Operations Research, 31(14), 2293–2313.
14. Balling R, Powell B, Saito M. 2004. Generating future l&-use & transportation plans for high-growth cities using a genetic algorithm. Computer-Aided Civil & Infrastructure Engineering, 19(3), 213–222.
15. Chandramouli M, Huang B, Xue L. 2009. Spatial change optimization: integrating GA with visualization for 3D scenario generation. Journal of the American Society of Photogrammetry & Remote Sensing, 75(8), 1015–1023.
16. Karlen M. 1993. Space planning basics. Van Nostr & Reinhold Publishers: New York.
17. F. Ching. 2009. Architectural graphics, 5th edn. Wiley.
18. Bertoline GR, Wiebe EN, Hartman NW, Ross WA. 2009. Technical graphics communication, 6th edn. McGraw-Hill: Boston, MA.
19. Kopec D. 2006. Environmental psychology for design. Fairchild Publications, NY.
20. Kilmer R, Kilmer WO. 1992. Designing interiors. Thomson Learning, CT.
21. Panero J, Zelnik M. 1979. Human dimensions & interior space. Whitney Library of Design/Watson Guptill Publications: New York.
22. Rengel R. 2003. Shaping interior space. Fairchild Publications: New York.
23. Kubba S. 2003. Space planning for commercial & residential interiors. McGraw-Hill, New York.
24. Lok L. 2004. A critical survey of software packages for use by interior designers. MSc Thesis, Department of Computer Science, University of Wales.
25. Maher ML, Bilda Z, Gül LF. 2006. Impact of collaborative virtual environments on design behavior, in Design Computing & Cognition '06. Gero JS, Editor. Dordrecht, The Netherlands: Springer, p. 305–321.
26. Whyte J, Bouchlaghem N, Thorpe A, McCaffer R. 2000. From CAD to virtual reality: modeling approaches, data exchange, & interactive 3d building design tools. Automation in Construction, 10, 43–55.
27. Balling RJ, Taber JT, Brown MR, Day K. 1999. Multiobjective urban planning using genetic algorithm. Journal of Urban Planning & Development, 125, 86–99.
28. Chandramouli M, Bertoline R, Connolly P. 2009. Generating alternative engineering design by integrating desktop VR with

genetic algorithms. Engineering Design Graphics Journal, 73(3), 1–12.

29. Seixas J, Nunes JP, Louren P, Lobo F, Condado P. 2005. GeneticLand: modeling land use change using evolutionary algorithms. Proceedings of the 45th Congress of the European Regional Science Association, Land Use and Water Management in a Sustainable Network Society, Vrije Universiteit: Amsterdam, p. 23–27.
30. Connolly PE. 2005. Virtual reality and immersive technology in education. International Journal of Information & Communication Technology Education, 1(1), 12–18.
31. Reas C, Fry B. 2003. Processing: a learning environment for creating interactive web graphics, in Proceedings of the SIGGRAPH 2003 Conf. on Web Graphics, SESSION: Java applications. San Diego: ACM.
32. Ames AL, Nadeau DR, Moreland JL. 1997. VRML 2.0 sourcebook , 2nd edn. John Wiley & Sons: NY.
33. van Kemenade C, Kok J, Eiben A. 1995. Raising GA performance by simultaneous tuning of selective pressure & recombination disruptiveness, in Proceedings of the 2nd IEEE Conference on Evolutionary Computation. IEEE Press, p. 346–651.

Cite this article as: Chandramouli M & Bertoline GR: A genetic algorithm based optimization framework to visualize, evaluate, and modify 3D space configurations in Desktop VR. Int. J. Simul. Multisci. Des. Optim., 2014, **5**, A01.