Sampling CAD Models via an Extended Teaching-Learning-Based Optimization Technique

Shahroz Khan[*,1] Erkan Gunpinar[1]
[1]Istanbul Technical University

Abstract

The Teaching-Learning-Based Optimization (TLBO) algorithm of Rao et al. has been presented in recent years, which is a population-based algorithm and operates on the principle of teaching and learning. This algorithm is based on the influence of a teacher on the quality of learners in a population. In this study, TLBO is extended for constrained and unconstrained CAD model sampling which is called Sampling-TLBO (S-TLBO). Sampling CAD models in the design space can be useful for both designers and customers during the design stage. A good sampling technique should generate CAD models uniformly distributed in the entire design space so that designers or customers can well understand possible design options. To sample *N* designs in a predefined design space, *N* sub-populations are first generated each of which consists of separate learners. Teaching and learning phases are applied for each sub-population one by one which are based on a cost (fitness) function. Iterations are performed until change in the cost values becomes negligibly small. Teachers of each sub-population are regarded as sampled designs after the application of S-TLBO. For unconstrained design sampling, the cost function favors the generation of space-filling and Latin Hypercube designs. Space-filling is achieved using the Audze and Eglais' technique. For constrained design sampling, a static constraint handling mechanism is utilized to penalize designs that do not satisfy the predefined design constraints. Four CAD models, a yacht hull, a wheel rim and two different wine glasses, are employed to validate the performance of the S-TLBO approach. Sampling is first done for unconstrained design spaces, whereby the models obtained are shown to users in order to learn their preferences which are represented in the form of geometric constraints. Samples in constrained design spaces are then generated. According to the experiments in this study, S-TLBO outperforms state-of-the-art techniques particularly when a high number of samples are generated.

Keywords: Teaching-Learning-Based Optimization, Design Sampling, Computer-Aided Design, Generative Design

1. Introduction

Engineering and industrial product design is a goal-oriented, constraint-based decision making process. The product obtained after this process should satisfy the consumers' needs, not just in its functional performance, but also its external appearance. The design process becomes more complex and time consuming if the products appearance is valuable to the consumers. At the initial stage of the design process, there may be no or few CAD models available for prototyping or design validation analysis. Design engineers form the design space for the CAD model using design specifications. The CAD model is modified to obtain geometric variations, some of which are inspected visually and validated based on the computer simulations or customer preferences. We think that a CAD model sampling technique can be beneficial for the designers during the design and validation stages. Moreover, CAD models sampled uniformly throughout the design space can form a better response surface, which can alleviate the computational burden of the computer-aided analysis (CAE) at the design validation stage[1].

The design space of a CAD model is defined by a set of design parameters, each with lower and upper bounds. Each design parameter represents a dimension in the design space. CAD model sampling is a high-dimensional

[*] **Email:** shahrozkhan2020@gmail.com, khansh@itu.edu.tr **Adress:** Istanbul Technical University, School of Mechanical Engineering, Inonu Caddesi, No: 65, Gumussuyu, 34437, Istanbul, TURKEY, **Tel:** +90-212-2931300 **Fax:** +90-212-2450795

Nomenclature

n	Number of subjects/parameters
N	Number of designs to be sampled in the design space
X_k	k^{th} learner
M_j	Mean result of the learner in the j^{th} subject (dimension)
X_{best}	Teacher (learner with the best fitness when considering all the subjects)
T_F	Teaching factor
$x^l_{m,j}$	Lower bound for the j^{th} design parameter for the design m
$x^u_{m,j}$	Upper bound for the j^{th} design parameter for the design m
P	Population
p_g	g^{th} sub-population of P
s	Size of each sub-population
T	Vector consists of teachers of sub-population
$\overline{x}_{p,j}$, $\overline{x}_{q,j}$	Scaled parameter values for the j^{th} subject of the teachers p and q in T
$\overline{U}(T)$	Cost function
$U_1(T)$	Potential energy (term for obtaining space-filling design)
$U_2(T)$	Term for obtaining non-collapsing designs
$U_3(T)$	Term for constraint handling mechanism
α	User-define parameter adjusting weight for the $U_2(T)$ term
λ_{pq}	Number of coordinates that two designs, p and q share
β	Weight for the $U_3(T)$ term
ϕ_t	t^{th} geometric constraint
n_c	Number of geometric constraints

constrained optimization problem as there are generally a high number of design parameters and design constraints (i.e., relations between the parameters).

In this work, the *space-filling* criterion [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 4, 13, 14, 15] is considered during sampling; this criterion favors the generation of CAD models spreading throughout the design space. Such sampling can give a better understanding of the design options in the design space for the designers. However, it tends to place a large portion of the designs at the corners and on the boundaries of the design space [5], especially in the case of high-dimensional sampling problems. To overcome this issue, designs are searched as far as possible in the class of Latin hypercube (LH) designs [16, 17, 4, 18, 6, 7, 8, 19, 11, 20, 21]. Therefore, the *non-collapsingness* criterion is also considered, so that the sampled designs do not share any design parameter values. Furthermore, non-collapsing designs are preferable in sampling CAD models, which are further tested in the validation step via computer simulations (i.e., CAE). The designs of experiments are crucial in CAE, which has the major goal of determining which design parameters have the greatest effects on the simulation results. Most CAE analyses are computationally expensive, and running the analysis for collapsing designs would ultimately result in unnecessary computational costs [4, 18]. As a result, this work takes the space-filling and non-collapsing properties into account while sampling CAD models. The degree of non-collapsingness is controlled with a weight term, which gives the designer the ability to create either completely non-collapsing or semi-non-collapsing designs.

Teaching-Learning-Based Optimization (TLBO) [22, 23] is a recently proposed evolutionary optimization technique which mimics the classic teaching learning philosophy of a class. In the past few years, TLBO became the rising star among the meta-heuristics optimization algorithms (such as genetic algorithm (GA) [24], particle swarm optimization (PSO) [25], artificial bee colony (ABC) [26], differential evolution [27] and CMA-ES [28], etc). According to the experiments done by Rao [23] and Rao and Waghmare [29], TLBO outperforms the optimization techniques such as GA, ABC, PSO and CMA-ES. Therefore, due to the effective performance and simplicity of the TLBO, the authors of the current research extend TLBO to generate space-filling designs for high-dimensional constrained design spaces. A new technique called *Sampling-TLBO* (S-TLBO) is proposed for the automatic search and generation of design variations for a CAD model in its predetermined design space. The criterion of space-filling enables the maintenance of the diversity of the sampled designs. Figure 1 shows the illustration of the outcomes of the proposed

S-TLBO technique.

Four major contributions are made into the standard TLBO approach in order to extend its ability for design sampling which are listed below:

(1) The population is divided into N number of sub-populations. Teachers of sub-populations will be regarded as the sampled designs, which are obtained by the S-TLBO approach.
(2) A cost (fitness) function is integrated into the TLBO technique in order to sample designs that spread evenly in the design space.
(3) A penalization technique is introduced in the TLBO to obtain designs having non-collapsing (NC) property as much as possible.
(4) A static constraint handling mechanism [30] is utilized in the S-TLBO technique to sample designs in the constrained design space so that sampled designs satisfy the predefined design constraints.

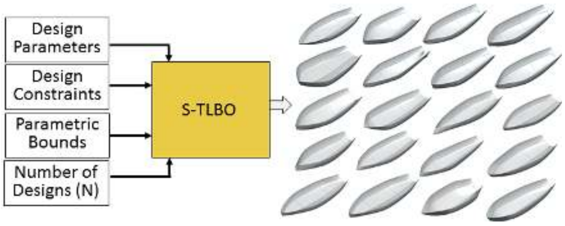

Figure 1: Illustration of the outcomes of proposed S-TLBO

The remainder of this paper is organized as follows: Section 1.1 reviews relevant literature. Formulation of original TLBO algorithm is described in Section 2. Section 3 discusses the proposed approach for generating new designs. Section 4 illustrates test models that are used for the experiments. The numerical results of the proposed approach are given in Section 5. Concluding remarks and opportunities for future work are presented in Section 6.

1.1. Related works

Our approach mainly depends on Teaching-Learning-Based Optimization (TLBO) and on the sampling of space-filling designs. Below, we discuss some previous works done by different researchers in these fields.

TLBO: TLBO was proposed by Rao et al. [22] and several improvements have been made on the technique in order to improve its performance and to expend its application in different fields. Rao and Patel [31] introduced the concept of elitism in the original TLBO to solve a series of constrained optimization problems. Another variation of TLBO called *OD-based TLBO* was introduced by Satapathy et al. [32]. OD-based TLBO combines the concept of traditional TLBO and orthogonal designs to improve its convergence performance. Chen et al. [33] enhanced the search ability of TLBO by introducing the concept of local learning and self-learning. A perturbation based scheme was introduced by Ouyang et al. [34] in order to prevent TLBO from getting trapped in the local optima. Patel and Savsani [35] and Yu et al. [36] incorporated the concept of Pareto dominance into TLBO to solve multi-objective constrained-based optimization problems.

TLBO and its variations have also been implemented to different fields such as science and engineering, including manufacturing and mechanical engineering, data clustering [37] and feature selection [38] and combinatorial optimization [39, 40, 41]. In manufacturing and mechanical engineering, TLBO has been used for cloud manufacturing [42], advanced finishing processes [43], heat exchangers [44], parameter optimization of selected casting processes [45], optimization of aerodynamic shapes [46] and mechanical design optimization [22].

Space-filling designs: There are several approaches proposed by different researchers for the optimal and space-filling designs. Johnson et al. [10] proposed a technique to obtain space-filling designs. In this method, designs to be sampled using LH are obtained while maximizing the minimum distance between designs (*maximin* criterion). Sacks et al. [47] introduced a method for the minimization of the integrated mean square error (IMSE). Shewry and Wynn [12] performed selection of designs based on entropy maximization. Bates et al. [21] and Husslage et al. [8] used an approach for potential energy minimization of designs.

Another algorithm for space-filling designs was proposed by Cioppa and Lucas [11]. However, their algorithm is computationally expensive because it requires long run times. A *sliced space-filling* design technique was introduced by Qian and Wu [9], where the design space is further sub-divided into small design spaces and sampling is then performed in the sub-spaces. Prescott [20] performed complete and partial enumeration searches to investigate the space-filling properties of orthogonal-column Latin Hypercube. Lekivetz and Jones [3] introduced an algorithm for contracting non-collapsing space-filling designs for higher dimension unconstrained input regions. Grosso et al. [18] proposed heuristic algorithms based on the Iterated Local Search (ILS) to create the maximin LH designs. Khan et al. [48] developed a spatial simulated annealing based sampling technique, and Khan and Gunpinar [49] proposed an Extended Latin Hypercube Sampling (ELHS) technique for automatic sampling and generation of CAD models. In ELHS to sample designs in the class of Latin Hypercube enumeration is preformed in each stratum of hypercube and variety of designs are selected based on a similarity constraint. ELHS has the ability to perform sampling in constrained and unconstrained spaces but it does not have good space-filling property.

There is a considerable amount of research that has been done on optimal selection of space-filling designs. However, most works done by researchers are proposed for the unconstrained design spaces. The research problem becomes more complicated when a selection of designs has to be performed in a constrained and high-dimensional design space like in the research of this paper. Myšáková and Lepš [50] proposed a technique to perform sampling for constrained spaces. This technique is based on the triangulation of admissible space by Delaunay Triangulation method. Although this technique has good space-filling property, it is just applicable to only two dimensional constraint problems. A constrained Latin Hypercube technique to solve constrained problems was demonstrated by Petelet et al. [19] and Benková et al. [14]. Draguljić et al. [17] proposed a CoNcaD algorithm for constructing non-collapsing and space-filling designs for bounded nonrectangular design spaces. Trosset [15] and Stinstra et al. [13] used maximin criterion for the construction of space-filling designs in the constrained 10-dimensional design space. The technique proposed by Stinstra et al. [13] does not guarantee the sampled designs to be non-collapsing. Fuerle and Sienz [4] proposed a method to produce Latin Hypercube designs in constrained spaces. This method has some drawbacks such that it cannot be implemented for high-dimensional sampling problems more than $3D$. Furthermore, it does not produce good results for a design space where infeasible designs are spread irregularly. S-TLBO, proposed in this paper, has the ability to obtain space-filling designs in more than $3D$ design spaces. S-TLBO also ensures sampling space-filling designs in the constrained design spaces where distribution of infeasible designs is highly irregular.

2. Teaching-Learning-Based Optimization (TLBO)

TLBO starts the optimization process by randomly generating a population of initial solutions for a given size within the defined design space. The solution is improved by undergoing two phases: teacher phase and learner phase. A population denoted by P is considered as a class of learners. Design variables are analogous to the subjects offered to learners in the class. In the teacher phase, the best solution in the entire class is selected as a teacher and this teacher tries to improve the quality of learners by sharing his/her knowledge with them. The quality of the teacher has significant influence on the quality of learners. In the learner phase, learners try to improve their quality by interacting with other learners. The detailed description of TLBO can be found in [22, 51]. The teacher and learner phases of TLBO are summarized in the subsequent sections.

2.1. Teacher phase

In the teacher phase, the teacher is selected based on the fitness value of cost function and attempts to increase the mean result of the entire class. Suppose there are n number of subjects (i.e. design parameters) offered to s number of learners. The k^{th} learner is denoted by $X_k = [x_{k,1}, x_{k,2}, x_{k,3}, \dots, x_{k,n}]$. At any sequential teaching or learning iteration, i, $M_{j,i}$ is the mean result of the learners in a particular subject j ($j = 1, 2, \dots, n$). The learner with the best fitness when considering all the subjects is chosen as a teacher (denoted by $X_{best,i}$) for that iteration. The position of each learner in the i^{th} iteration is updated using Equation (1).

$$x'_{k,j,i} = x_{k,j,i} + r_i \times (x_{best,j,i} - T_F \times M_{j,i}) \tag{1}$$

$x'_{k,j,i}$ is the updated value of $x_{k,j,i}$, and $x_{best,j,i}$ is the result of the best learner (i.e., teacher) in the subject j. r_i is a random number in the range $[0 - 1]$. T_F denotes the teaching factor whose value can be either 1 or 2 and computed using the following formula: $T_F = round[1 + rand(0, 1)\{2 - 1\}]$. $X'_{k,i}$ is accepted if it gives a better fitness value than $X_{k,i}$ in the i^{th} iteration.

2.2. Learner phase

In the learner phase, learners increase the quality of their knowledge by interacting with each other. A learner learns from other learners if the other learner has better knowledge than him/her. Two learners, S and R, are randomly selected such that $X'_{S,i} \neq X'_{R,i}$, where $X'_{S,i}$ and $X'_{R,i}$ denote the updated values for $X_{S,i}$ and $X_{R,i}$, respectively, at the end of the teacher phase. Learners S and R at the i^{th} iteration can be donated as $X_{S,i} = [x_{S,1,i}, x_{S,2,i}, x_{S,3,i}, \ldots, x_{S,n,i}]$ and $X_{R,i} = [x_{R,1,i}, x_{R,2,i}, x_{R,3,i}, \ldots, x_{R,n,i}]$, respectively, where $x_{S,j,i}$ and $x_{R,j,i}$ donates the j^{th} coordinate of the learners S and R, respectively.

If $X'_{S,i}$ gives a better fitness value, the position of the learner S is updated using Equation (2). Otherwise, Equation (3) is used to update its position.

$$x''_{S,j,i} = x'_{S,j,i} + r_i \times (x'_{S,j,i} - x'_{R,j,i}) \tag{2}$$

$$x''_{S,j,i} = x'_{S,j,i} + r_i \times (x'_{R,j,i} - x'_{S,j,i}) \tag{3}$$

3. Proposed Algorithm

This section presents details of the proposed S-TLBO technique that samples N designs for a given design space. We first outline the core idea behind the S-TLBO approach. Sampling designs in constrained and unconstrained spaces using S-TLBO will be then explained.

3.1. The S-TLBO approach

Basic terminologies are described first in relation to problem setting. A design or CAD model can be represented by design parameters. Let $x_{m,1}, x_{m,2}, x_{m,3}, \ldots, x_{m,j}$ be the design parameters for the design m. Lower and upper bounds for the j^{th} design parameter are denoted by $[x^l_{m,j}]$ and $[x^u_{m,j}]$, respectively. Design space is formed by the design parameters and their bounds. Each design parameter defines a dimension in the design space.

The standard TLBO provides a single optimal solution by guiding the initial population of learners to an optimum position. The objective of S-TLBO is to obtain N optimum solutions (or designs). The search process of S-TLBO starts by creating the random initial population P consisting of N sub-populations (see Equation (4)). p_g denotes the g^{th} sub-population of P (i.e., $g = 1, 2, 3, \ldots, N$). Each sub-population consists of s learners which is shown in Equation (5).

$$\mathrm{P} = \begin{bmatrix}(p_1)_{s\times j} & (p_2)_{s\times j} & (p_3)_{s\times j} & \ldots (p_N)_{s\times j}\end{bmatrix}^T \tag{4}$$

with

$$p_m = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_s \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & \ldots & x_{1,j} \\ x_{2,1} & x_{2,2} & \ldots & x_{2,j} \\ \vdots & \vdots & \vdots & \vdots \\ x_{s,1} & x_{s,2} & \ldots & x_{s,j} \end{bmatrix} \tag{5}$$

The population P can be considered as a school which consists of $p_1, p_2, p_3, \ldots, p_N$ number of classes. The ranking of a school mainly depends upon the quality of its teaching staff (i.e. teachers) and the quality of the students (i.e., learners). The school consisting of N classes should have N teachers, one for each class. Like standard TLBO, teachers are the best learners in the school having a minimum cost (fitness) value among all the learners.

The cost utilized in S-TLBO, which is given in Equation 15 and will be explained in the next section, is computed based on the teachers of sub-populations. Let $T = [T_1, T_2, \ldots, T_N]$ denotes a set of teachers and T_p is the teacher for the p^{th} sub-population. For the selection of N initial teachers, there are s^N number of teacher combinations, which can result in a high computational cost if N and s are assigned to a larger value. Therefore, a greedy-selection strategy is chosen for the selection of N initial teachers. In this strategy, N is first set to 2 in the cost function, and the best two learners of the first two sub-population which minimizes the cost function are selected as teachers T_1 and T_2. Afterwards, a learner, which minimizes the cost function, is selected as teacher T_3 from the third sub-population under the consideration of the preselected teachers T_1 and T_2 by setting $N = 3$. The selection process is repeated in a similar manner until N teachers for the N number of sub-populations are determined. Note that the greedy-selection strategy checks $s^2 + \sum_2^N s$ learners' combinations to select N initial teachers.

In S-TLBO, the learning process in any iteration is completed by performing N number of sub-iterations, one for each sub-population. Each sub-population goes through the teacher and learner phases individually while keeping the teachers of other sub-populations the same. During these phases, a new position for a learner is found using Equations (1), (2) and (3). Let X_k and X'_k be the current and new positions of a learner in the first sub-population, respectively. The new position is accepted if the cost value $T'_k = [X'_k, T_2, \ldots, T_N]$ is less than $T_k = [X_k, T_2, \ldots, T_N]$. The teachers T are updated after each sub-iteration for the sub-population. The teacher is a learner having a minimum cost value (computed with the teachers of other sub-populations) among the other learners in the same sub-population. Sub-iterations in other sub-populations are performed in the same way. S-TLBO is terminated when change in the cost values after two consecutive iterations performed in the population become negligibly small. Figure 2 shows the flow chart for the S-TLBO algorithm.

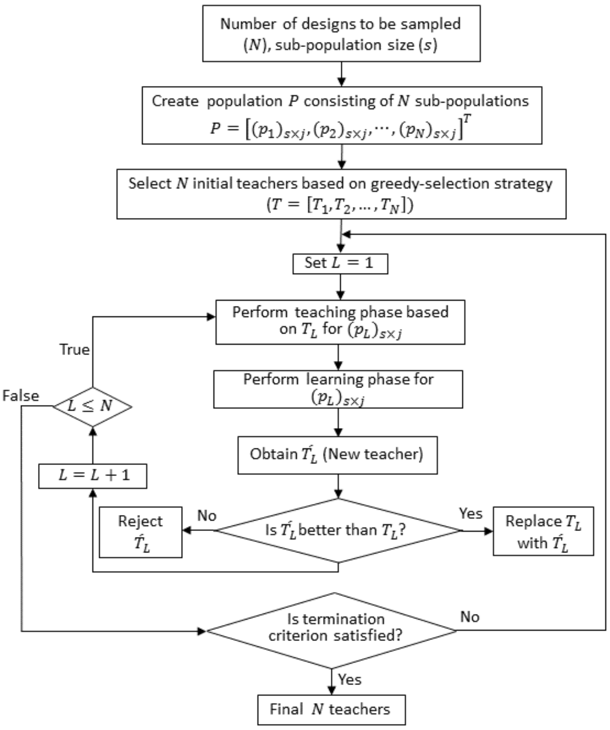
Number of designs to be sampled (N), sub-population size (s)
Create population P consisting of N sub-populations
$P = [(p_1)_{s\times j}, (p_2)_{s\times j}, \cdots, (p_N)_{s\times j}]^T$
Select N initial teachers based on greedy-selection strategy ($T = [T_1, T_2, \ldots, T_N]$)
Set $L = 1$
Perform teaching phase based on T_L for $(p_L)_{s\times j}$
Perform learning phase for $(p_L)_{s\times j}$
Obtain $\acute{T_L}$ (New teacher)
Is $\acute{T_L}$ better than T_L?
No
Yes
Reject $\acute{T_L}$
Replace T_L with $\acute{T_L}$
$L = L + 1$
$L \leq N$
True
False
Is termination criterion satisfied?
No
Yes
Final N teachers

Figure 2: Flow chart of S-TLBO

3.2. Sampling designs with S-TLBO in unconstrained spaces

Our objective, as stated before, is to generate different design variations of a given CAD product in a predetermined design space. During sampling, the sampled designs should be in different positions in the design space. Therefore, a cost function based on the technique of Audze and Eglais [2] is utilized, which follows a physical analogy: *Molecules in a space exert repulsive forces on each other that lead to potential energy in a space. These molecules are in equilibrium in case of minimum potential energy.* The designs in the design space can be considered as molecules and the cost function (see Equation (6)) for the S-TLBO is formulated as follows:

$$U_1(T) = \sum_{p=1}^{N-1} \sum_{q=p+1}^{N} \frac{1}{\sum_{j=1}^{n}(\overline{x}_{p,j} - \overline{x}_{q,j})^2} \tag{6}$$

$\overline{x}_{p,j}$ and $\overline{x}_{q,j}$ are the *scaled* parameter values for the j^{th} subject (dimension) of the teachers (designs) p and q in T which are computed by scaling parameter values between 0 (i.e., lower bound for the parameter) and 1 (i.e., upper bound for the parameter). These bounds form the design space, which is called *scaled design space.* Recall that N is the number of designs to be sampled and n is the number of dimensions in the design space. The potential energy in Equation (6) has to be minimized. Figure 3 (a) shows 25 samples (i.e., $N = 25$) generated using Equation (6) for a two-dimensional problem. It is seen that samples spread evenly in the design space, therefore space-filling designs can be obtained.

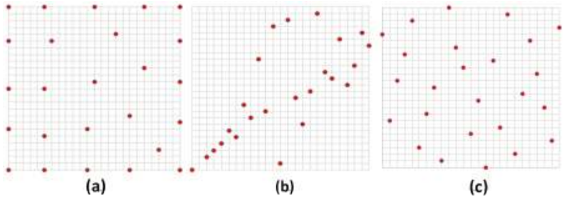

Figure 3: (a) Space-filling designs (b) Non-collapsing designs (c) Space-filling and non-collapsing designs.

Minimization of the potential energy $U_1(T)$ favors placement of the designs at the maximum separating distance from each other. However, this energy function itself locates designs at the corners and on the boundaries of the design space particularly for the high-dimensional sampling problems [5]. For the CAD model sampling, it is desired to spread designs evenly also in the inner portions of the design space. Therefore, non-collapsing property (i.e., designs not sharing same parameter values) for the sampled designs should be satisfied as much as possible. Figure 3 (a) and (b) show collapsing and non-collapsing designs, respectively. Sampling designs with only considering non-collapsing property for the constrained design spaces may result in designs with poor space-filling property as seen in Figure 3 (b). When both space-filling and non-collapsing properties are considered, better sampling can be achieved (see Figure 3 (c)). Note that the samples in Figure 3 are obtained using the S-TLBO approach with $N = 25$ and $s = 10$.

A penalization technique is introduced in order to sample non-collapsing designs in the design space. A new term, $U_2(T)$, is included in the cost function, which is as follows:

$$U(T) = U_1(T) + \alpha \times U_2(T) \tag{7}$$

where

$$U_2(T) = \sum_{p=1}^{N-1} \sum_{q=p+1}^{N} \lambda_{pq} \tag{8}$$

$$\lambda_{pq} = \sum_{j=1}^{n} f_1(y_{p,j}, y_{q,j}) \tag{9}$$

$$\begin{aligned} f_1(y_{p,j}, y_{q,j}) &= 1 \quad \text{if} \quad y_{p,j} = y_{q,j} \\ f_1(y_{p,j}, y_{q,j}) &= 0 \qquad \text{otherwise} \end{aligned} \tag{10}$$

α is a user-defined parameter adjusting weight for the $U_2(T)$ term. The parameter values for designs are first converted into integer values. $y_{p,j}$ and $y_{q,j}$ in Equation (9) are the corresponding integer coordinate values for $x_{p,j}$ and $x_{q,j}$ in the j^{th} dimension (subject), respectively. λ_{pq} denotes the number of coordinates that two designs, p and q share. The range of each design parameter is partitioned into N equal intervals (levels) as follows: $[x^{l}_{m,j} = x^{1}_{m,j}, x^{2}_{m,j}, \ldots, x^{N}_{m,j} = x^{u}_{m,j}]$. Parameter values are converted to the integer values using Equation (11) where r is an integer number varying from 1 to N. λ_{pq} for the designs p and q is then computed using the piecewise function f_1 shown in Equation (10).

$$\begin{aligned} \text{if } x^r_{p,j} \leq x_{p,j} < x^{r+1}_{p,j} \text{ then } y_{p,j} = r \\ \text{if } x^r_{q,j} \leq x_{q,j} < x^{r+1}_{q,j} \text{ then } y_{q,j} = r \\ \text{with } 1 \leq r \leq N \end{aligned} \tag{11}$$

The term $U_2(T)$ is based on the degree of violation for the non-collapsing property. Maximum value for this term can be $j \times {}^N C_2$. ${}^N C_2$ represents the combinations between designs which is as follows: ${}^N C_2 = \frac{N!}{2!(N-2)!}$. Setting the parameter α to larger values will lead to producing more non-collapsing designs. On the other hand, the designs sampled may have poor space-filling as seen in Figure 3 (b). As a result, α should be tuned carefully which will be discussed in the "Results and Discussion" section.

3.3. Sampling designs with S-TLBO in constrained spaces

Design space consists of feasible and infeasible regions in the case of constrained spaces. Feasible regions consist of feasible designs that satisfy the predefined design constraints. Infeasible designs are located in the infeasible regions. We transform the constrained problem to an unconstrained one by using a static constraint handling mechanism. This concept was introduced in genetic algorithms by Homaifar et al. [30] and has been utilized in many works [52, 53, 54, 55]. In this mechanism, designs are penalized if they violate any constraint. The cost function $\overline{U}(T)$ is utilized for constrained spaces which is written below:

$$\overline{U}(T) = U(T) + \beta \times U_3(T) \tag{12}$$

where

$$U_3(T) = \sum_{p=1}^{N} \sum_{t=1}^{n_c} f_2(p,t) \tag{13}$$

$$\begin{aligned} f_2(p,t) = 1 \;\; &\text{if the constraint } \phi_t \text{ is violated by the design } p \\ f_2(p,t) = 0 \;\; &\text{otherwise} \end{aligned} \tag{14}$$

The cost function $\overline{U}(T)$ consists of the terms $U(T)$, $U_3(T)$ and a parameter, β that adjusts the weight of $U_3(T)$. Let n_c be the number of geometric constraints and ϕ_t represent a constraint, where $t = 1, \ldots, n_c$. The piecewise function f_2 gets a value of 1 if a design, p, violates the constraint ϕ_t (see Equation (14)). Otherwise, it is 0. The term $U_3(T)$ is computed using Equation (13).

The cost function $\overline{U}(T)$ is overall composed of three terms, $U_1(T)$, $U_2(T)$ and $U_3(T)$ and two parameters; α and β, which is shown below:

$$\begin{aligned} \overline{U}(T) = \sum_{p=1}^{N-1} \sum_{q=p+1}^{N} \frac{1}{\sum_{j=1}^{n} (\overline{x}_{p,j} - \overline{x}_{q,j})^2} \; + \\ \alpha \times \sum_{p=1}^{N} \sum_{q=p+1}^{N} \lambda_{pq} \; + \; \beta \times \sum_{p=1}^{N} \sum_{t=1}^{n_c} f_2(p,t) \end{aligned} \tag{15}$$

In case of constrained design space, where there exists a infeasible region in the design space and no design is allowed to be selected from that region, some designs might not be able to satisfy geometric constraints and be non-collapsing at the same time, especially when the infeasible region is large or spread irregularly in the design space.

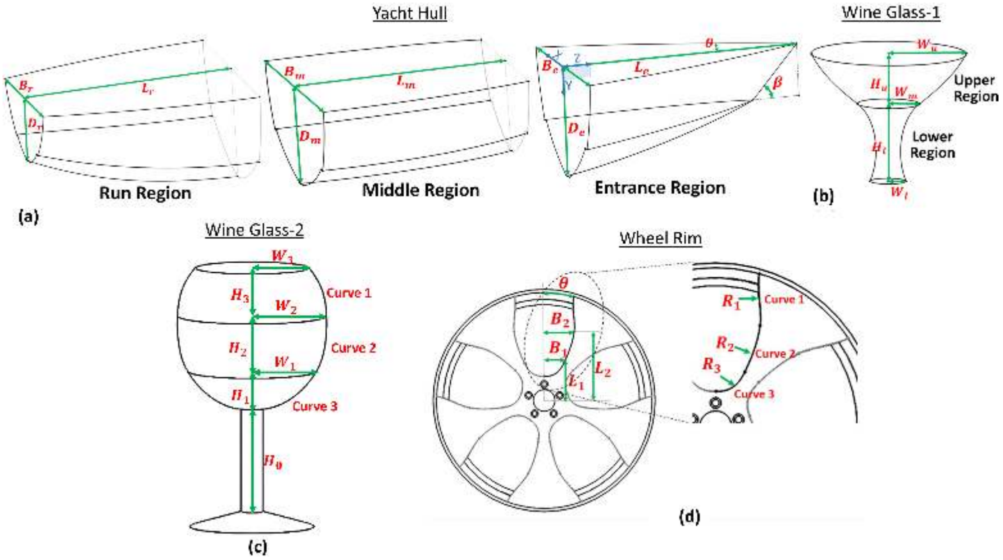

Figure 4: CAD model of a yacht hull (a), a sherry-type wine glass without base (glass-1) (b), a pinot noir-type wine glass (glass-2) (c) and a wheel rim (d) with their design parameters.

4. CAD Models for Study's Experiments

S-TLBO can generate a variety of designs for a given CAD model in the constrained and unconstrained design spaces. Both the design specifications and user preferences can be represented by constraints. To validate the performance of S-TLBO, four different CAD models were utilized, as follows: a yacht hull, wheel rim, sherry-type wine glass without a base (glass-1), and pinot noir-type wine glass (glass-2). These models are shown in Figure 4. The constraints for the yacht hull and glass-1 models were formulated by incorporating the user preferences. For the glass-2 model, the design specifications, such as the glasss capacity to store a certain amount of liquid, were given as constraints. The constraints for the yacht hull and glass-1 models were determined via one-to-one interviews with the participants; thus, to some extent, they reflect customer preferences related to the product. We call this *customer-centered design sampling*. A customer-centered design is developed considering the customers perspectives by understanding their preferences [56, 57, 58]. The design sampling is first carried out without constraints, and the sampled designs are shown to the participants to learn their preferences. By including these preferences as S-TLBO constraints, sampling can be performed in the constrained space.

For higher design variations, we divided the yacht hull model into three regions, as follows: the entrance, middle, and run. The glass-1 model had lower and upper regions, while the profiles of the wheel rim and glass-2 models were divided into three curves. The $3D$ surface models for these models were obtained by lofting between the Bezier curves separately in each region, and the design parameters were defined on the curves. The surface modification of the models was performed using the technique proposed by Khan et al. [59]. The design parameters L_e, L_m, and L_r represent the lengths of the entrance, middle, and run regions, respectively, of the yacht hull. Moreover, B_e,B_m,B_r and D_e,D_m,D_r denote the beams and widths of regions, respectively, while θ is the entrance angle and β is the bow angle. For the glass-1 model, the heights of the lower and upper regions are denoted by H_l and H_u, respectively. Moreover, W_l and W_u are the widths of the lower and upper regions, respectively, while W_m denotes the width at the connection between the upper and the lower regions. For the wheel rim model, parameters L_1,L_2 and B_1,B_2 denote the length and width of curves 1 and 2, respectively. The parameter θ adjusts the position of curve 3 on the outer circle of the wheel rim. The minimum radii of the curvature of curves 1, 2, and 3 are represented by parameters R_1, R_2, and R_3, respectively. The design parameters H_1,H_2,H_3 and W_1,W_2,W_3 denote the lengths and widths, respectively, of curves 1, 2, and 3, and H_0 is the vertical length of glass-2's stem. Each design parameter described represents a dimension in the design space. Therefore, 11-, 5-, 7-, and 8-dimensional design spaces were formed for the yacht hull, glass-1, glass-2, and wheel rim, respectively. Tables 1, 2, and 3 show the parameter ranges (i.e., lower and upper bounds for the design parameters) that were used in this study's experiments for the yacht hull, glass, and wheel rim models, respectively.

The design constraints for the yacht hull and glass-1 can be seen in Tables 1 and 2, respectively. S-TLBO first sampled designs in an unconstrained space; following this, participants' preferences concerning the appearance of these designs were learned via comprehensive one-to-one interviews. There were 30 and 50 samples obtained for the yacht hull and glass-1 models, respectively, as shown in Figures 5 (a) and 6 (a). Three participants attended the interview for the yacht hull model, whereas only one participant was interviewed for the glass-1 model. The CAD models generated by S-TLBO were shown to the participants one by one to learn whether they liked or disliked the models. Next, they were asked about the reasons for their preferences concerning the models. The following questions were asked when a model was shown to the participants:

- Familiarize yourself with this model by viewing it from different perspectives.
- Do you like or dislike it?
- If you like it, which features of this model do you like?
- If you dislike it, which features of this model do you dislike?

In this way, the participants' preferences were quantized and represented by geometric constraints. For instance, the participant disliked the designs in Figure 6 (a), numbered in red as 1, 7, 11, 13, 14, 16, 20, 27, 28, 29, 31, 35, 41, 47, 48, and 50. When these models are visually inspected, it can be recognized that the parameter W_m is approximately equal to or higher than W_u. The participant preferred designs with W_u parameter values of at least twice the W_m parameter values. As the objective of this study was to perform sampling in a predetermined design space, the constraints in Tables 1 and 2 are not all explained here in detail. Based on the predetermined constraints, S-TLBO was utilized once again in constrained design spaces for the generation of customer-centered designs.

Table 1: Parameter bounds and geometric constraints for the yacht hull model

Parameter Bounds (in meters)		
$6 \leq L_e \leq 12$	$6 \leq L_m \leq 12$	$2.5 \leq L_r \leq 8$
$4.80 \leq B_e \leq 10.8$	$4.64 \leq B_m \leq 10.64$	$3 \leq B_r \leq 9.0$
$2.34 \leq D_e \leq 4.34$	$2.21 \leq D_m \leq 4.21$	$1.75 \leq D_r \leq 3.50$
$30 \leq \beta \leq 120$	$25 \leq \theta \leq 70$	

Geometric Constraints			
No.	Participant 1	Participant 2	Participant 3
ϕ_1	$L_r \geq 2L_m/3$	$-1.2 \leq L_e - L_m \leq 2.4$	$-2.4 \leq L_e - L_m \leq 3.6$
ϕ_2	$B_e \geq B_m/2$	$L_r \geq L_m/2$	$L_m \geq L_r \geq L_m/2$
ϕ_3	$B_m \geq B_r$	$0 \leq B_e - B_m \leq 2.4$	$-1.8 \leq B_e - B_m \leq 3$
ϕ_4	$D_e \geq D_m \geq D_r$	$B_m \geq B_r$	$B_m \geq B_r$
ϕ_5	$50 \leq \beta \leq 120$	$0 \leq D_e - D_m \leq 0.7$	$0 \leq D_e - D_m \leq 0.7$
ϕ_6		$D_m \geq D_r$	$D_m \geq D_r$
ϕ_7		$40 \leq \beta \leq 70$	$110 \leq \beta \leq 80$
ϕ_8		$45 \leq \theta \leq 70$	if ($\beta \geq 110$) then $\theta \geq 55$

5. Results and Discussion

In this study, S-TLBO was tested for both constrained and unconstrained design sampling. Figures 5 (a), 6 (a), 7 (a), and 8 show the yacht hull, glass-1, glass-2, and wheel rim models generated by S-TLBO without using any geometric constraints, respectively. The models in Figures 5 (b, c, d), 6 (b), and 7 (b) were obtained after the application of S-TLBO with the geometric constraints. The designs in Figure 7 (b) were generated based on the geometric constraint of $volume \leq 200$. As stated above, the designs in Figures 5 (b, c, d) and 6 (b) were generated

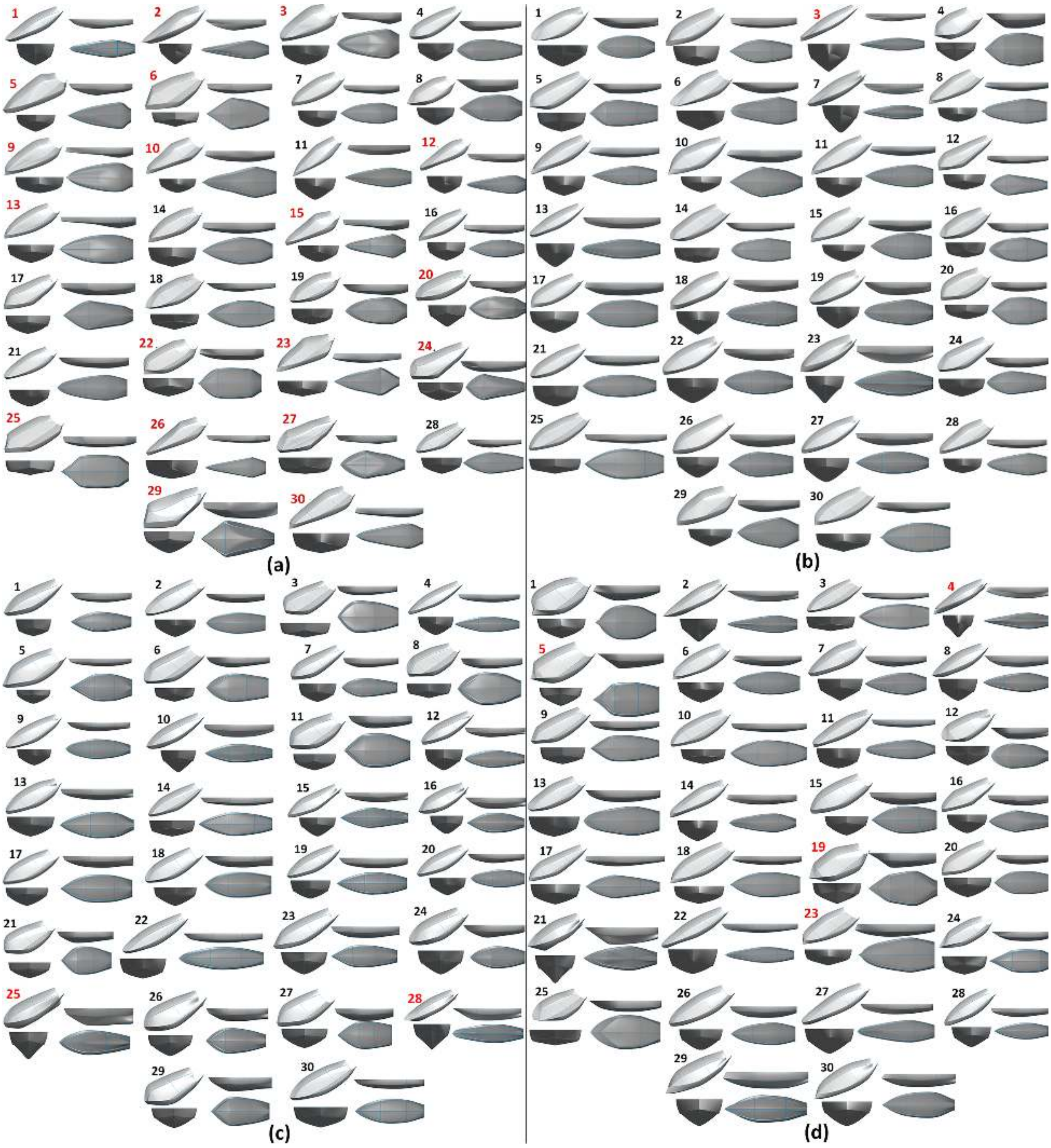

Figure 5: (a) Designs obtained using S-TLBO without the geometric constraints. Designs obtained with the geometric constraints, quantified for participants 1 (b), 2 (c) and 3 (d) for the yacht hull model. (The participants disliked the designs numbered in red).

based on user preferences, which were learned via one-to-one interviews. We also measured the success rate of S-TLBO for the yacht hull in terms of the model satisfaction of the interviewed participants. They were asked again whether they like the generated models. The satisfaction rate is the percentage rate of the number of CAD models that the participants liked over the number of sampled models. Satisfaction rates of 96.6%, 93.3%, and 86.6% were achieved for participants 1, 2, and 3, respectively.

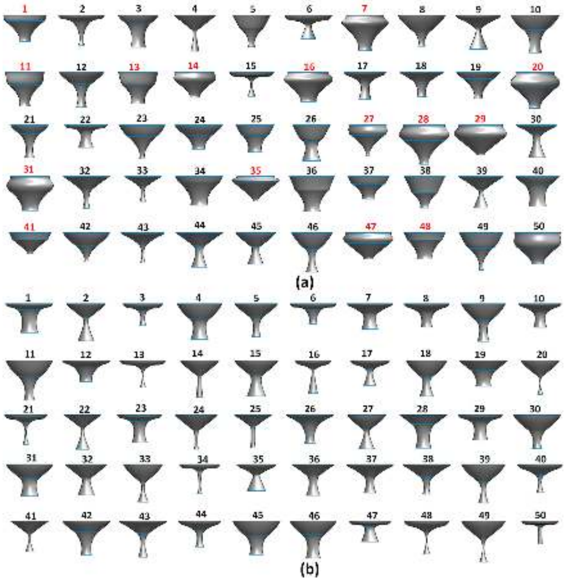

Figure 6: Designs obtained using S-TLBO without (a) and with (b) the geometric constraints quantified for the participant for the glass-1 model. (The participant disliked the designs numbered in red).

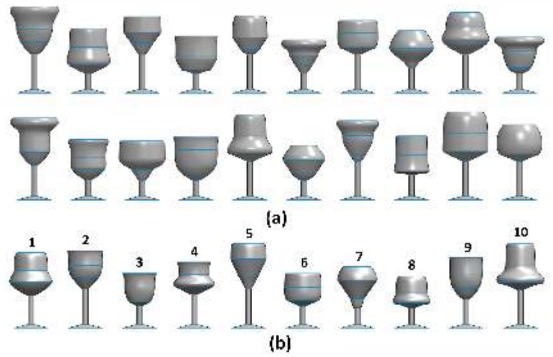

Figure 7: Glass-2 designs obtained using S-TLBO without (a) and with (b) the geometric constraint.

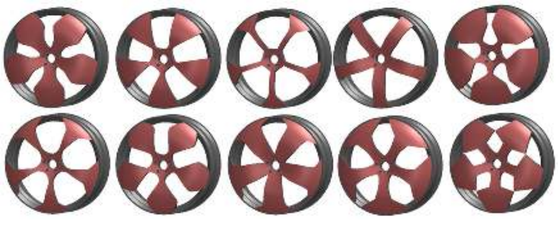

Figure 8: Designs obtained for the wheel rim models using S-TLBO.

5.1. Computational Time

A PC with an i7 6700 Intel Core, 3.4-GHz processor, and 16-GB memory was used for the experiments in this study; the C++ programming language was employed for the implementation of the S-TLBO algorithm. The computational times taken to obtain the sampling results in Figures 5, 6, 7, and 8 are shown in Table 4; they varied between 1.73 and 29.36 minutes. It has been observed that the processing time mainly depends on the number of designs to

Table 2: Parameter bounds and geometric constraints for glass-1 and glass-2

	Wine Glass-1		Wine Glass-2	
Parameter Bounds (in millimeters)				
	$40 \le H_u \le 120$	$90 \le H_l \le 170$	$50 \le H_0 \le 100$	$90 \le H_1 \le 115$
	$130 \le W_u \le 170$	$15 \le W_m \le 160$	$130 \le H_2 \le 155$	$175 \le H_3 \le 200$
	$15 \le W_l \le 65$		$17.5 \le W_1 \le 60$	$37.5 \le W_2 \le 60$
			$32.5 \le W_3 \le 60$	
Geometric Constraints				
ϕ_1	$H_l \ge H_u$		$volume \le 200$	
ϕ_2	$W_m \le W_u/2$			
ϕ_3	$W_m \le W_l$			

Table 3: Parameter bounds for the wheel rim model

Parameter Bounds (in millimeters)		
$50 \le B_1 \le 100$	$120 \le L_1 \le 270$	$0.5 \le B_2 \le 210$
$280 \le L_2 \le 400$	$3.5 \le \theta \le 30$	$0 \le R_1 \le 10$
$0 \le R_2 \le 50$	$0 \le R_3 \le 50$	

be sampled (N), dimensionality of the design space (n), and size of the subpopulations (s). Increases in the values of these parameters will increase the S-TLBO's processing time. Due to the larger value of s, the sampled designs in Figure 5 (d) had the highest computational time compared with the sampled designs in Figure 5 (a), (b), and (c). Furthermore, S-TLBO takes more time when it is used in constrained spaces. For instance, the designs in Figure 5 (b) were obtained after a higher processing time of S-TLBO than those obtained in Figure 5 (a), even with the same parameter settings for N, n, and s. The computational complexity of TLBO for a single iteration is $O(s \cdot n)$. Since S-TLBO has N subpopulations, its computational complexity for a single iteration is $O(s \cdot n \cdot N)$. Let i_{max} be the total number of iterations after the termination of S-TLBO; the computational complexity of S-TLBO is $O(s \cdot n \cdot N, i_{max})$.

Table 4: Processing time taken to obtain the sampling results in Figures 5, 6, 8, and 11

	N	n	s	$\overline{U}(T)$	$U_1(T)$	Number of Collapsing Models	Computational Time (minute)
Results in Figure 5 (a)	30	11	60	241.78	245.31	0	16.08
Results in Figure 5 (b)	30	11	60	23113.49	269.62	16	16.46
Results in Figure 5 (c)	30	11	70	13127.32	261.00	18	21.50
Results in Figure 5 (d)	30	11	80	11111.18	250.61	16	29.36
Results in Figure 6 (a)	50	5	30	1885.94	1734.16	0	10.63
Results in Figure 6 (b)	50	5	30	6757.20	2071.37	30	10.91
Results in Figure 7 (a)	20	7	20	167.04	137.85	20	3.60
Results in Figure 7 (b)	10	7	20	372.22	42.22	10	1.73
Results in Figure 8	10	8	20	28.91	30.62	0	3.28
Results in Figure 11	30	11	–	–	421.86	28	2.58

5.2. Algorithm Convergence

The S-TLBO stops the optimization process when there is no improvement in the cost function $\overline{U}(T)$ for some consecutive iterations (i); at this point, the design reaches the optimal position, and the algorithm is considered to converge to its optimality. The plots in Figure 9 show the cost value versus the number of S-TLBO iterations for the designs in Figures 5, 6, 7, and 8. Many iterations were performed for the models in Figures 5, 6, 7, and 8 to analyze the convergence of the S-TLBO algorithm (see the plots in Figure 9). No improvements were observed in the cost function ($\overline{U}(T)$) after some consecutive iterations. There was no change in the cost function after approximately the 60^{th} iteration for the models in Figures 5 (a), 5 (d), 5 (c), 6, 7, and 8. For the models in Figure 5 (b), S-TLBO converged at $i = 80$. The convergence rate depends on the number of designs that are sampled (N), dimensionality of design space (n), and total number of constraints (n_c). Due to its low dimensionality and the presence of fewer geometric constraints, S-TLBO converges faster for the sampling problem in Figure 6 than that in Figure 5.

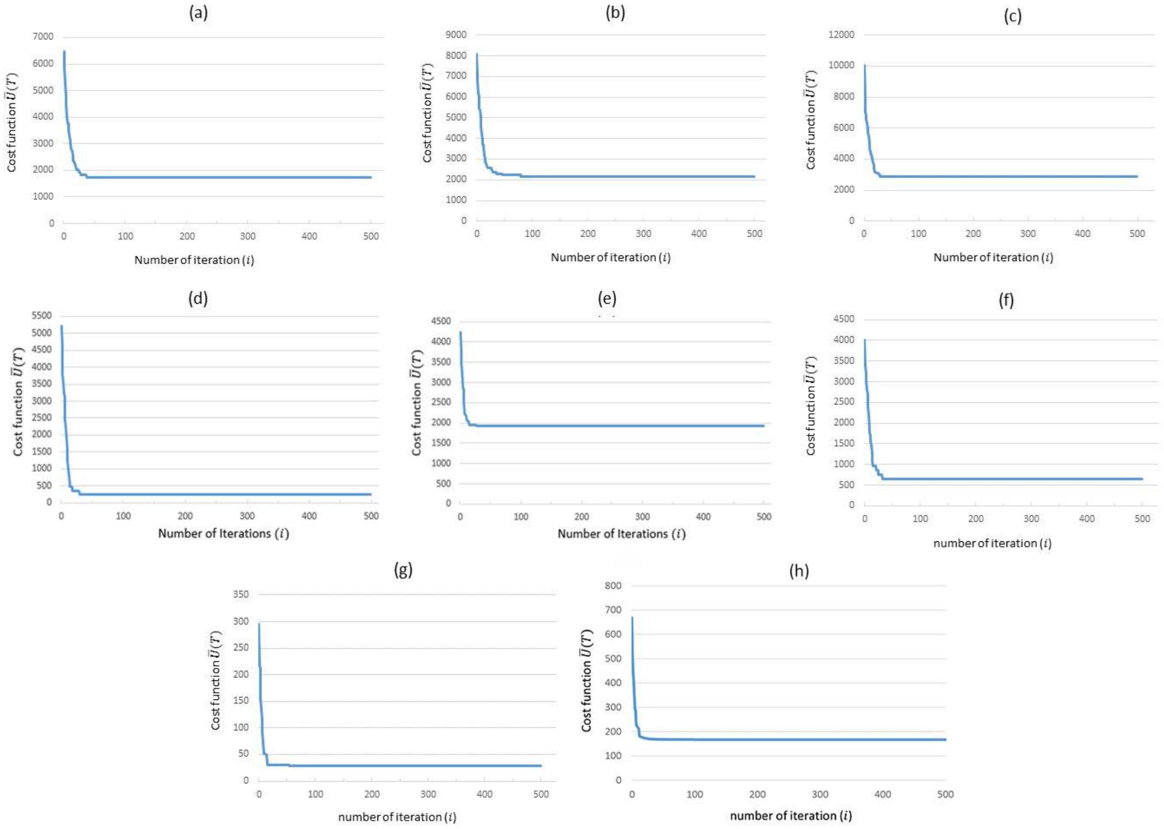

Figure 9: (a), (b), (c), (d) Plots showing the cost value versus number of S-TLBO iterations for the models in Figure 5 (a), (b), (c), and (d), respectively. (e), (f), (g), (h) Plots showing the cost value versus number of S-TLBO iterations for the models in Figures 6 (a), 6 (b), 8, and 7 (a), respectively.

5.3. Comparison with Existing Works

We compared the performance of S-TLBO with the existing state-of-the-art techniques in the literature that have been proposed to generate optimal space-filling designs in experiments. First, we compared the performance of S-TLBO with Fuerle and Sienz's [4] technique. For the comparison, we utilized the two-dimensional examples 2 and 7 in the paper [4], which are also shown in Figure 10 (a) and (c), respectively. Examples 2 and 7 utilize Equations 16 and 17, respectively, as constraints.

$$\begin{aligned}(x_1 - 0.5)^2 + (x_2 - 0.5)^2 &\geq 0.25\\ 0 \leq x_1 &\leq 1\\ 0 \leq x_2 &\leq 1\end{aligned} \tag{16}$$

$$\begin{aligned} (x_1 - 0.5)^2 + (x_2 - 0.5)^2 &\geq 0.1 \\ (x_1)^2 + (x_2 - 1)^2 &\geq 1/3 \\ (x_1 - 1)^2 + (x_2)^2 &\geq 1/3 \\ 0 \leq x_1 &\leq 1 \\ 0 \leq x_2 &\leq 1 \end{aligned} \tag{17}$$

Fifteen design points (i.e., $N = 15$) were sampled under the above constraints via S-TLBO. These points are shown in Figure 10 (b) and (d), which are plotted after conversion into integer coordinates using Equation 11. The designs in Figure 10 (b) were obtained with the $\alpha = 50, \beta = 2000$, and $s = 40$ settings. The designs in Figure 10 (d) were obtained with the following settings: $\alpha = 30, \beta = 2000$, and $s = 40$. The sampled designs generated using Fuerle and Sienz's [4] method are shown in Figure 10 (a) and (c). The values of potential energy $U_1(T)$ for the design points in Figure 10 (a), (b), (c), and (d) are were about 610, 555, 618, and 518, respectively. The potential energies of the sampled design points generated by S-TLBO were lower than those produced by Fuerle and Sienz's technique.

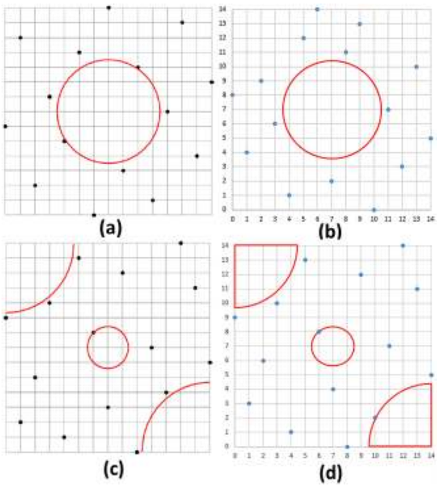

Figure 10: Design points are generated based on the constraints defined in Equation 16 using (a) Fuerle and Sienz's [4] technique and (b) S-TLBO. The design points are obtained based on the constraints defined in Equation 17 using (c) Fuerle and Sienz's [4] method and (d) S-TLBO (images (a) and (c) are taken from [4]).

We also compared the performance of S-TLBO with random sampling and other techniques, such as the L_p optimization technique proposed by Torsset [15], recently proposed CoNcaD technique of Draguljić et al. [17], and NPL algorithms SFDP* and SFDP** proposed by Stinstra et al. [13]. These techniques utilized the *maximin distance* proposed by Johnson et al. [10] for creating space-filling designs (see Equation 18). Therefore, we used Equation 18 instead of Equation 6 to allow comparison of our work with these methods. Equation 18 spreads out the designs in the design space by maximizing the minimum interpoint distance between the pair of designs, which is expressed as follows:

$$\underset{T}{max} \quad \underset{X_p, X_q \in T}{min} \quad \sqrt{\sum_{j=1}^{n} (\overline{x}_{p,j} - \overline{x}_{q,j})^2} \tag{18}$$

The quadrant/ball region example was also tested; this was defined by the following constraints: $x_j \geq 0$ where $j = 1, 2, \ldots, n$ and $\sum_{j=1}^{n}(x_j^2) \leq 0$. For $n = 2$ and $n = 10$, the designs were obtained using S-TLBO's constrained formulation with the maximin distance in Equation 18 for the settings of $\alpha = 0, \beta = 2000$, and $i = 500$. Table 5 shows the maximin distances for the design points produced by L_p optimization, SFDP* and SFDP**, CoNcaD , and random sampling. It should be noted that the data in Table 5 are taken from Table 2 in Draguljić et al. [17] except for random sampling, S-TLBO, and S-TLBO ($\alpha = 0$) . To compare the S-TLBO results with those of random sampling for the quadrant/ball region example, the designs were randomly selected from the design space. The samples violating the constraint(s) were rejected, and random sampling was repeated until all N samples satisfied the constraints. As mentioned in [13], SFDP* cannot be used for sampling problems with a high number of constraints. As shown in Table 5, except for $n = 10$ and $N = 10$, the design points generated by S-TLBO have a higher maximin distance value (i.e., better sampling quality) than those produced by the SFDP** and CoNcaD techniques. It should be noted that random sampling and L_p produced the poorest results compared with the other techniques. Especially, L_p generated the worst samples in high-dimensional design spaces (i.e., $n = 10$). According to the experiments, we observed that S-TLBO generated samples with better sampling quality, especially for large values of n and N. It is noteworthy that SFDP* and SFDP** do not consider the non-collapsing property during sampling, and therefore, these techniques should also be compared with S-TLBO when the parameter α is set to zero. In Table 5, S-TLBO ($\alpha = 0$) is the S-TLBO algorithm when $\alpha = 0$. It is shown in Table 5 that S-TLBO ($\alpha = 0$) produces better results than SFDP* and SFDP**.

Table 5: The maximin distance (in Equation 6) of the designs produced by different algorithms for $n = 2$ and $n = 10$ using the quadrant/ball example (NA: Not Applicable)

n	N	Random Sampling	SFDP*	SFDP**	L_P	CoNcaD	S-TLBO	S-TLBO($\alpha = 0$)
2	10	0.0444	0.3587	0.3522	0.2554	0.3400	0.3565	0.3630
2	20	0.0537	0.2394	0.1966	0.2212	0.2124	0.2292	0.2391
2	50	0.0063	0.1403	0.1188	0.1336	0.1204	0.1283	0.1409
2	100	0.0046	NA	0.0486	0.0660	0.0789	0.0822	0.0893
2	200	0.0037	NA	0.0259	0.0070	0.0539	0.0566	0.0584
10	10	0.3606	1.4142	1.2167	0.5017	1.3027	1.0882	1.4358
10	20	0.3095	1.0323	0.9591	0.4138	0.8364	0.9877	1.0461
10	50	0.2658	0.8811	0.7588	0.2952	0.6747	0.7917	0.8852
10	100	0.2203	NA	0.6248	0.2753	0.5838	0.6770	0.7095
10	200	0.1367	NA	0.5235	0.2559	0.5160	0.5689	0.5817

To compare the results of S-TLBO with random sampling, we generated the yacht hull model in Figure 4 (a) via random sampling under the constraints quantified from participant 1. Thirty yacht hull models were generated via random sampling. A model violating the constraint(s) was rejected, and a new model was sampled. The sampling was repeated until all 30 models satisfying the constraints were obtained, as shown in Figure 11. These 30 randomly sampled models were compared with the models in Figure 5. It can be seen from Table 4 that the random sampled models have a poorer space-filling property than those obtained from S-TLBO.

5.3.1. Statistical Analysis of Results in Table 5

We also analyzed the statistical performance of the proposed approach over the other techniques in Table 5. The Shapiro-Wilk and Kolmogorov-Smirnov normality tests were performed to check the distribution of the data in the table 5. The significance values for these tests were smaller than 0.05, which implies that the data have a non-normal distribution. The histogram plot in Figure 12 also confirmed this, which is obtained using the results of the proposed and benchmark techniques in Table 5. The data distribution is left-skewed with a skewness value of 1.032, therefore the data have a non-normal distribution. Thus, the Friedman test [60], a nonparametric statistical test, was used. It should be noted that the nonparametric approaches have been employed by researchers to compare the performance of swarm and evolutionary algorithms [61, 62, 63].

In the Friedman test, two hypotheses–the null hypothesis $H0$ and alternative hypothesis $H1$–are defined. The null hypothesis states that there is no significant difference between the algorithms, whereas the alternative hypothesis anticipates the presence of a significant difference between the algorithms [61]. chi^2 and p-Values provide information about whether a statistical hypothesis test is true. If the p-value is less than 0.05 and the value of chi^2 is greater than the critical value of chi^2, then there is strong evidence against $H0$. To carry out this test, two datasets were formed, one by excluding (the first dataset) and other by including (the second dataset) the results of SFDP* as some results

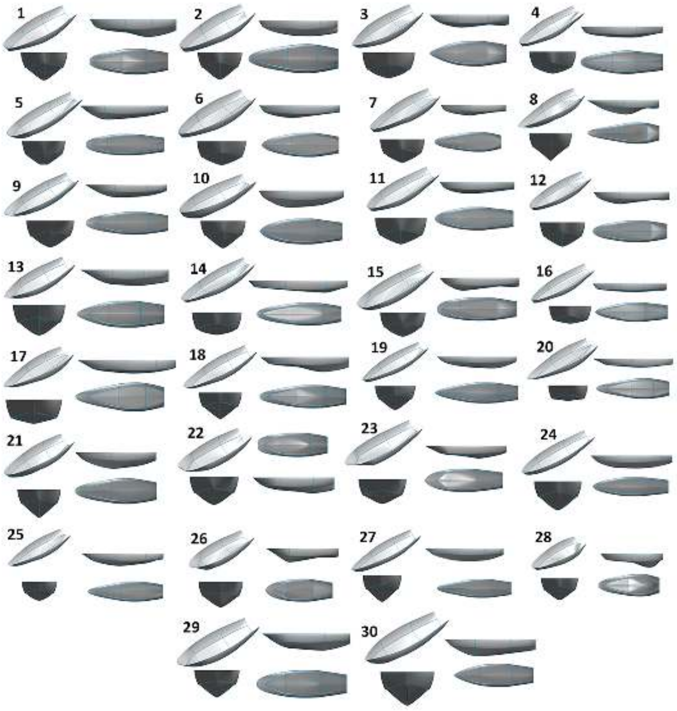

Figure 11: Yacht hull models generated via random sampling under the constraints quantified from participant 1

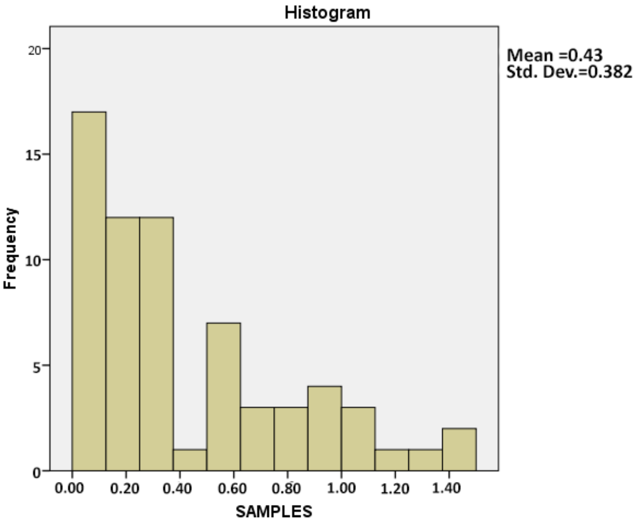

Figure 12: Histogram plot for the data in Table 5

in Table 5 were not provided due to the incapability of SFDP*.

We performed two types of Friedman test using the datasets, namely multiple and pairwise comparison. The multiple comparison examined the multiple algorithms at once, including S-TLBO and other techniques in Table 5. On the other hand, individual comparisons between two different algorithms were performed in the pairwise comparison in which each algorithm was compared to all other algorithms. The *p*-value and *chi*^{2} of the first dataset were $5.156e - 08$ and 42.286, respectively, for the multiple comparison. The values were $1.38e - 06$ and 37.543, respectively, for the second dataset. The critical chi^2 was 12.59 for both datasets. As the p-values were less than 0.05, and the chi^2 values were greater than the critical chi^2 (12.59), these results indicated strong evidence against $H0$. Similarly, the results showed strong evidence against $H0$ for the pairwise comparison.

Furthermore, pairwise comparison analyses were also performed for the first and second datasets, and their results can be seen in Tables 6 and 7, respectively. It should be noted that only S-TLBO and CoNCaD consider the non-collapsing criterion. Therefore, results of these techniques should be compared. On the other side, the results of random sampling, SFDP*, SFDP** and L_p should be compared with S-TLBO ($\alpha = 0$) as these techniques do not consider the non-collapsing criterion. The computed p-values for S-TLBO-CoNcaD and S-TLBO($\alpha = 0$)-others (others include random, SFDP*, SFDP ** and Lp) are all less than 0.05, and the chi^2 values are greater than the critical chi^2 (3.84) in Table 6. This proves that the null hypothesis $H0$ is rejected for the present analysis. Similarly, the computed p- and chi^2 values for S-TLBO-CoNcaD and S-TLBO($\alpha = 0$)-others in the second dataset are shown in Table 7, which are again less than 0.05 and the critical chi^2 value, respectively. These results imply that the null hypothesis H0 is rejected. This concludes that the results shown in Table 5 obtained using S-TLBO are significantly different from those obtained using the other alternative techniques. Consequently, S-LTBO has statistically better performance (i.e., space-fillingness) than the other alternative techniques.

Consequently, the Friedman test shows that the results given in Table 5 obtained using S-TLBO are significantly different from those obtained using the other alternative techniques. As S-TLBO has better performance (i.e. better space-filling property) according to the experiments in Table 5, it can be concluded that S-TLBO has statistically better performance than the other alternative techniques.

Table 6: p-values and chi^2 values for the first dataset obtained from the Friedman test in pairwise comparison

	Random Sampling	SFDP**	L_p	CoNcaD	S-TLBO	S-TLBO (α = 0)
			p-values			
Random Sampling	NaN	0.001	0.001	0.001	0.001	0.001
SFDP**	0.001	NaN	0.205	1	0.011	0.001
L_p	0.001	0.205	NaN	0.057	0.011	0.001
CoNcaD	0.001	1	0.057	NaN	0.011	0.001
S-TLBO	0.001	0.011	0.011	0.011	NaN	0.001
S-TLBO(α = 0)	0.001	0.001	0.001	0.001	0.001	NaN
			chi^2 values			
Random Sampling	NaN	10	10	10	10	10
SFDP**	10	NaN	1.6	0	6.4	10
L_p	10	1.6	NaN	3.6	6.4	10
CoNcaD	10	0	3.6	NaN	6.4	10
S-TLBO	10	6.4	6.4	6.4	NaN	10
S-TLBO(α = 0)	10	10	10	10	10	NaN

5.3.2. Comparison of S-TLBO with NSGA-II's non-dominated sorting and crowding distance phenomenon

We compared the weighted combination of objectives ($U_1(T)$, $U_2(T)$, and $U_3(T)$) with nonparameterized approaches, such as NSGA-II [64], which relies on the Pareto dominance and crowding distance to prioritize solutions. Zou et al. [65] proposed an improved TLBO called multi-objective teaching-learning-based optimization (MOTLBO). In MOTLBO, Zou et al. [65] used an external archive to retain the best solutions. The concept of nondominated sorting and the mechanism of crowding distance computation are incorporated into the algorithm, specifically in the teacher phase. Like in Zou et al.s [65] work, we incorporated the nondominated sorting concept and mechanism of crowding distance in S-TLBO, and we call the revised approach S-TLBO*.

S-TLBO* first sorts each subpopulation based on the nondominated sorting and crowding distance. The sorted subpopulations are then placed in an external archive. Following this, S-TLBO* applies the teacher and learner phases to each subpopulation to create a new one. It then combines the new subpopulation with its respective external archive. Note that after this combination, the total size of each population becomes 2s. Then, s solutions are selected from the

Table 7: p-values and chi^2 values for the second dataset obtained from the Friedman test in pairwise comparison

	Random Sampling	SFDP*	SFDP**	L_p	CoNcaD	S-TLBO	S-TLBO ($\alpha = 0$)
			p-values				
Random Sampling	NaN	0.527	0.001	0.001	0.001	0.001	0.001
SFDP*	0.527	NaN	0.527	0.527	0.527	0.527	0.011
SFDP**	0.001	0.527	NaN	0.205	1	0.011	0.001
L_p	0.001	0.527	0.205	NaN	0.057	0.011	0.001
CoNcaD	0.001	0.527	1	0.057	NaN	0.011	0.001
S-TLBO	0.001	0.527	0.011	0.011	0.011	NaN	0.001
S-TLBO ($\alpha = 0$)	0.001	0.011	0.001	0.001	0.001	0.001	NaN
				chi^2 values			
Random Sampling	NaN	0.4	10	10	10	10	10
SFDP*	0.4	NaN	0.4	0.4	0.4	0.4	6.4
SFDP**	10	0.4	NaN	1.6	0	6.4	10
L_p	10	0.4	1.6	NaN	3.6	6.4	10
CoNcaD	10	0.4	0	3.6	NaN	6.4	10
S-TLBO	10	0.4	6.4	6.4	6.4	NaN	10
S-TLBO ($\alpha = 0$)	10	6.4	10	10	10	10	NaN

$2s$ solutions based on the nondominated sorting and crowding distance for the next generation. In S-TLBO*, each initial teacher is selected from its external archive, and a nondominated solution in the first Pareto front can be used as the teacher. If there are two or more nondominated solutions in the first Pareto front, then the solution with the highest crowding distance value is selected as the teacher.

A solution X_1 is said to be Pareto dominated by the other solution X_2 if $U_1(X_1) \leq U_1(X_2) \wedge U_2(X_1) \leq U_2(X_2) \wedge U_3(X_1) \leq U_3(X_2)$ and$U_1(X_1) < U_1(X_2) \vee U_2(X_1) < U_2(X_2) \wedge U_3(X_1) < U_3(X_2)$. This dominating criterion is initiated to make designs that guarantee the validation of the geometric constraints in each iteration. During the teacher and learner phases, a new learner, X_1', replaces the previous learner X_1 if X_1' dominates X_1. Moreover, X_1' is rejected if X_1 dominates. When X_1' and X_1 do not dominate each other, we randomly select one to add to the population. As in S-TLBO, teachers are selected as the sampled designs at the end of the last iteration.

The performance of S-TLBO* was compared with that of S-TLBO for the wheel rim, glass-1, and yacht hull models for the settings of $i = 30$, $s = 10$, and $N = 10$. Table 8 shows that the values of $U_1(T)$ for the wheel rim and yacht samples in the case of S-TLBO were lower than those obtained using S-TLBO*. For all the test models, the values of $U_2(T)$ and $U_3(T)$ for the samples obtained using S-TLBO were less than those generated by S-TLBO*. The computational cost of S-TLBO* is approximately two times higher than that of S-TLBO.

5.4. User Study

We conducted a new user study that included 14 participants to compare the results obtained from S-TLBO and random sampling. Four participants had more than 3 years of professional design experience, and the others were chosen from the Amazon Mechanical Turk platform. First, designs were generated for the glass-2 model, which are shown in Figure 13. These designs were compared with the designs in Figure 7 (a), which are generated using S-TLBO. We asked the participants to rate designs based on a Likert scale, with anchors ranging from ”very poor” to ”very good” (1: very poor, 2: poor, 3: fair, 4: good, 5: very good). It should be noted that the survey participants did not have any information about the techniques used to generate the models. Table 9 summarizes the user study results. It was observed that the average ratings given by the users to the designs generated with random sampling were mostly lower than those obtained using S-TLBO. Ten out of 14 participants preferred the designs generated using S-TLBO, including the four experienced designers. Furthermore, the designs generated using random sampling had poor space-filling and non-collapsing compared to those generated using S-TLBO. When visually inspected, the designs generated using random sampling looked extremely similar. In contrast, the designs generated using S-TLBO looked highly different, as these designs were well spread in the design space (i.e., better space-filling designs).

Table 8: Comparison of S-TLBO with S-TLBO* for the wheel rim, glass, and yacht hull models.

	Wheel Rim	Wine Glass	Yacht Hull
	STLBO		
$U_1(T)$	28.70	58.90	22.40
$U_2(T)$	0	9	8
$U_3(T)$	–	0	0
Computational Time (minutes)	3.28	1.72	3.09
	S-TLBO*		
$U_1(T)$	33.34	56.20	29.20
$U_2(T)$	36	13	34
$U_3(T)$	–	12	15
Computational Time (minutes)	7.38	3.83	5.95

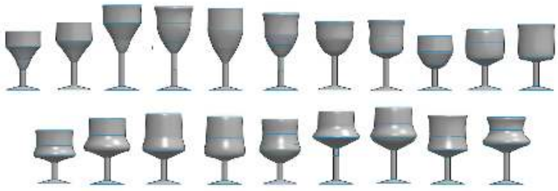

Figure 13: Glass-2 models generated using random sampling.

A *t*-test was also utilized to statistically examine the S-LTBO performance. The users data were normally distributed, as the skewness value was close to zero, and their mean/median values were approximately equal. The null hypothesis states that the ratings given to the designs generated using random sampling and S-LTBO will be similar. The p-value obtained from the t-test was less than the significance level of 0.05, so the null hypothesis was rejected.

5.5. Parameter Tuning

S-TLBO utilizes the static constraint handling method to favor the sampling of feasible designs. Here, β is a parameter in the cost function $\overline{U}(T)$ that adjusts the weight of the term $U_3(T)$, while α is a parameter in the cost function $\overline{U}(T)$ that adjusts the weight of the term $U_2(T)$. Experiments were conducted to observe the behavior of S-TLBO at different values of α.

Figure 14 (a), (b), and (c) show plots of the number of designs violating the geometric constraints versus β for the yacht hull model for participants 1, 2, and 3. Figure 14 (d) shows a plot for the number of designs violating the geometric constraints versus β for the glass-1 model. No generated CAD models violated any design constraint when parameter β was set to values greater than 600. Therefore, β was set to $2,000$ in this study's experiments. Figure 15 (a) and (b) show plots of the number of collapsing designs versus α. There were 49 collapsing designs when α was set to 0. When α increased, the number of collapsing designs decreased, and this reached zero when $\alpha = 40$. Therefore, α was set to 100 in this study, except for the designs in Figure 7.

A $3D$ surface plot, shown in Figure 16, was also obtained to validate the ideal values for α and β. It can be observed that the value of $U_1(T)$ was minimal when α and β were equal to zero. When α was zero, most of the sampled designs were collapsing, and thus, they spread more uniformly in the design space. When α increased, the sampled designs tended to be non-collapsing, and therefore, $U_1(T)$ increased. According to the plot, $U_1(T)$ started to increase rapidly until $\alpha = 40$; after this point, it became almost constant. A similar trend was observed for the parameter β. When β was zero, no geometric constraints were considered in the S-TLBO algorithm, and thus, the designs spread more uniformly in the design space. With the increase in β (i.e., favoring the satisfaction of the geometric constraints), the designs could not spread freely in the design space due to the presence of the constraints. It can be observed from the plot that $U_1(T)$ increased until $\beta = 600$.

Table 9: Results of the second user study

	Random Sampling	S-TLBO
User	Average Grade	
1	2.85	3.50
2	3.65	4.15
3	3.75	4.40
4	2.65	4.30
5	2.25	2.55
6	3.10	2.70
7	4.15	3.85
8	3.45	3.55
9	4.15	3.85
10	3.15	2.9
11	2.85	3.20
12	2.60	2.80
13	3.35	3.55
14	2.70	3.00
$U_1(T)$	226.23	137.84
$U_2(T)$	88.00	32.00
σ	1.024	1.070
μ	3.190	3.450
$Median$	3.00	3.00
$Skewness$	-0.04	-0.180
p-value	0.003365	

The size of the subpopulations (s) also plays an important role for the generation of space-filling designs. Higher values of s create a diverse initial solution for S-TLBO, which facilitates its search of the design space for the global optimum solution. In contrast, the application of S-TLBO with higher values of s can result in a higher computational cost. We recommend setting s to a value higher than 20. Table 4 provides values of s, $\overline{U}(T)$, and the number of collapsing designs for the designs in Figures 5, 6, 7, 8, and 11.

6. Conclusions and Future Works

This paper proposes a sampling method for CAD models. Teaching-Learning-Based Optimization (TLBO) is extended for design sampling in constrained and unconstrained spaces. The S-TLBO approach randomly generates sub-populations and improves learners in these sub-populations using teaching and learning phases. To obtain distinct designs in the design space, designs with space-filling and non-collapsing properties are favored during sampling. Designs, that do not satisfy predefined design constraints are penalized using a constraint handling mechanism. After S-TLBO terminates, teachers of sub-populations are regarded as the sampled designs.

As a future work, we would like to integrate simulated annealing approach [66] into the S-TLBO algorithm. We believe that this can enable the generation of better samples (i.e., space-filling and non-collapsing samples). Furthermore, the application of teaching and learning phases via teachers and learners in different populations will be investigated. Finally, Jaya Algorithm [67], and CMA-ES [28] will be studied in order to enhance their abilities to sample CAD models and a separate user preference learning system will also be integrated in the proposed approach.

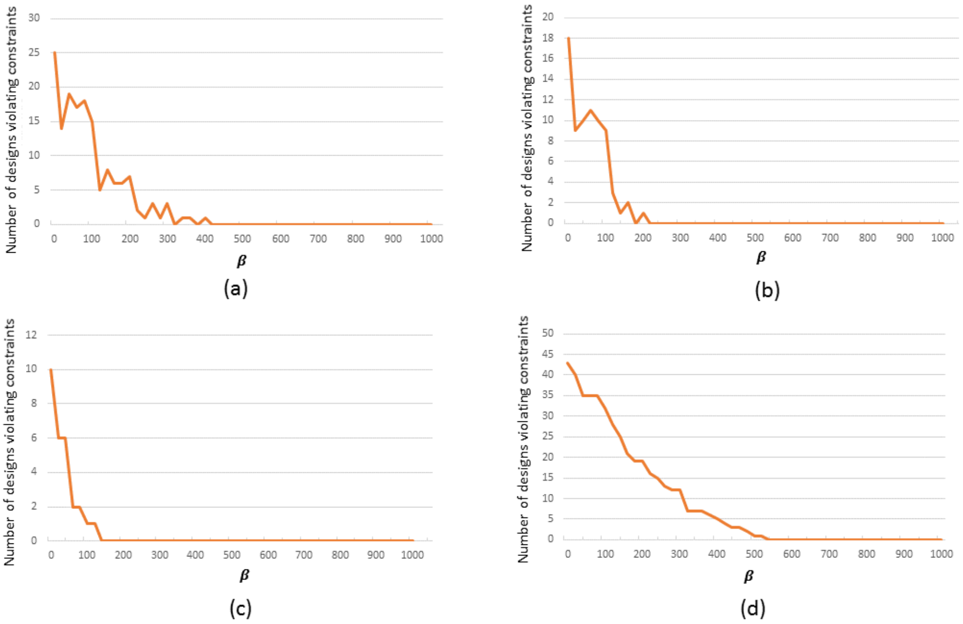

Figure 14: (a), (b), (c), (d) Plots showing the number of designs violating the geometric constraints versus β for the models in Figures 5 (b), (c), and (d) and 6 (b), respectively.

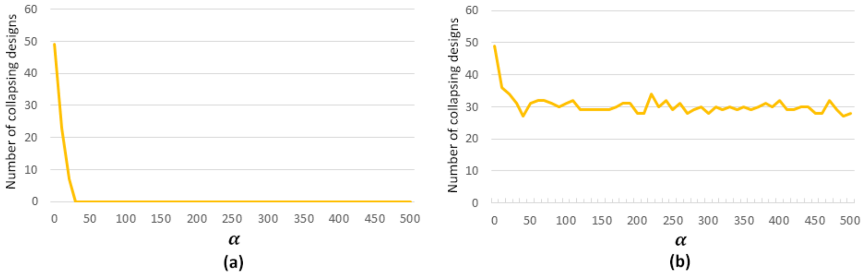

Figure 15: Plot of the number of collapsing designs versus α when $\beta = 0$ (a) and $\beta = 2,000$ (b).

Acknowledgment

This work is supported by the Scientific and Technological Research Council of Turkey (Project Number: 214M333). The authors would like to thank Dr. Serkan Gunpinar and Mr. Kemal Mert Dogan for their help about statistical analysis.

References

[1] I. Kaymaz, C. A. McMahon, A response surface method based on weighted regression for structural reliability analysis, Probabilistic Engineering Mechanics 20 (1) (2005) 11–17.

[2] P. Audze, V. Eglais, New approach for planning out of experiments, Problems of dynamics and strengths 35 (1977) 104–107.

[3] R. Lekivetz, B. Jones, Fast flexible space-filling designs for nonrectangular regions, Quality and Reliability Engineering International 31 (5) (2015) 829–837.

[4] F. Fuerle, J. Sienz, Formulation of the audze–eglais uniform latin hypercube design of experiments for constrained design spaces, Advances in Engineering Software 42 (9) (2011) 680–689.

[5] V. R. Joseph, E. Gul, Maximum projection designs for computer experiments, Biometrika 102 (2) (2015) 371–380.

[6] P. Ranjan, N. Spencer, Space-filling latin hypercube designs based on randomization restrictions in factorial experiments, Statistics & Probability Letters 94 (2014) 239–247.

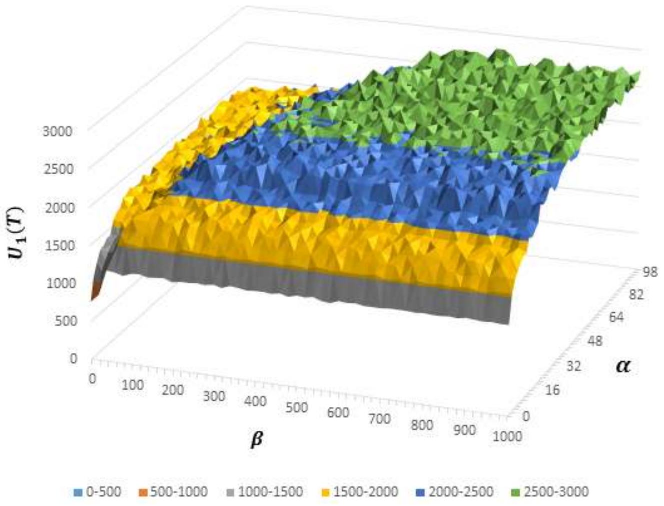

Figure 16: Surface plot of $U_1(T)$ versus β versus α.

[7] G. Damblin, M. Couplet, B. Iooss, Numerical studies of space-filling designs: optimization of latin hypercube samples and subprojection properties, Journal of Simulation 7 (4) (2013) 276–289.

[8] B. G. Husslage, G. Rennen, E. R. van Dam, D. den Hertog, Space-filling latin hypercube designs for computer experiments, Optimization and Engineering 12 (4) (2011) 611–630.

[9] P. Z. Qian, C. J. Wu, Sliced space-filling designs, Biometrika (2009) 945–956.

[10] M. E. Johnson, L. M. Moore, D. Ylvisaker, Minimax and maximin distance designs, Journal of statistical planning and inference 26 (2) (1990) 131–148.

[11] T. M. Cioppa, T. W. Lucas, Efficient nearly orthogonal and space-filling latin hypercubes, Technometrics 49 (1) (2007) 45–55.

[12] M. C. Shewry, H. P. Wynn, Maximum entropy sampling, Journal of applied statistics 14 (2) (1987) 165–170.

[13] E. Stinstra, D. den Hertog, P. Stehouwer, A. Vestjens, Constrained maximin designs for computer experiments, Technometrics 45 (4) (2003) 340–346.

[14] E. Benková, R. Harman, W. G. Müller, Privacy sets for constrained space-filling, Journal of Statistical Planning and Inference 171 (2016) 1–9.

[15] M. W. Trosset, Approximate maximin distance designs, in: Proceedings of the Section on Physical and Engineering Sciences, 1999, pp. 223–227.

[16] M. D. McKay, R. J. Beckman, W. J. Conover, A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Technometrics 42 (1) (2000) 55–61.

[17] D. Draguljić, T. J. Santner, A. M. Dean, Noncollapsing space-filling designs for bounded nonrectangular regions, Technometrics 54 (2) (2012) 169–178.

[18] A. Grosso, A. Jamali, M. Locatelli, Finding maximin latin hypercube designs by iterated local search heuristics, European Journal of Operational Research 197 (2) (2009) 541–547.

[19] M. Petelet, B. Iooss, O. Asserin, A. Loredo, Latin hypercube sampling with inequality constraints, AStA Advances in Statistical Analysis 94 (4) (2010) 325–339.

[20] P. Prescott, Orthogonal-column latin hypercube designs with small samples, Computational Statistics & Data Analysis 53 (4) (2009) 1191–1200.

[21] S. J. Bates, J. Sienz, D. S. Langley, Formulation of the audze–eglais uniform latin hypercube design of experiments, Advances in Engineering Software 34 (8) (2003) 493–506.

[22] R. V. Rao, V. J. Savsani, D. Vakharia, Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems, Computer-Aided Design 43 (3) (2011) 303–315.

[23] R. V. Rao, Teaching learning based optimization algorithm and its engineering applications, Springer International Publishing, Switzerland, 2016.

[24] J. H. Holland, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence, MIT press, 1992.

[25] R. Poli, J. Kennedy, T. Blackwell, Particle swarm optimization, Swarm intelligence 1 (1) (2007) 33–57.

[26] D. Karaboga, B. Basturk, On the performance of artificial bee colony (abc) algorithm, Applied soft computing 8 (1) (2008) 687–697.

[27] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, Journal of global optimization 11 (4) (1997) 341–359.

[28] C. Igel, N. Hansen, S. Roth, Covariance matrix adaptation for multi-objective optimization, Evolutionary computation 15 (1) (2007) 1–28.

[29] R. Rao, G. Waghmare, Solving composite test functions using teaching-learning-based optimization algorithm, in: Proceedings of the Interna-

tional Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA), Advances in Intelligent Systems and Computing, Vol. 199, Springer, 2013, pp. 395–403.

[30] A. Homaifar, C. X. Qi, S. H. Lai, Constrained optimization via genetic algorithms, Simulation 62 (4) (1994) 242–253.

[31] R. Rao, V. Patel, An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems, International Journal of Industrial Engineering Computations 3 (4) (2012) 535–560.

[32] S. C. Satapathy, A. Naik, K. Parvathi, A teaching learning based optimization based on orthogonal design for solving global optimization problems, SpringerPlus 2 (1) (2013) 130.

[33] D. Chen, F. Zou, Z. Li, J. Wang, S. Li, An improved teaching–learning-based optimization algorithm for solving global optimization problem, Information Sciences 297 (2015) 171–190.

[34] H.-b. Ouyang, L.-q. Gao, X.-y. Kong, D.-x. Zou, S. Li, Teaching-learning based optimization with global crossover for global optimization problems, Applied Mathematics and Computation 265 (2015) 533–556.

[35] R. Rao, V. Patel, A multi-objective improved teaching-learning based optimization algorithm for unconstrained and constrained optimization problems, International Journal of Industrial Engineering Computations 5 (1) (2014) 1–22.

[36] K. Yu, X. Wang, Z. Wang, Constrained optimization based on improved teaching–learning-based optimization algorithm, Information Sciences 352 (2016) 61–78.

[37] S. C. Satapathy, A. Naik, Data clustering based on teaching-learning-based optimization, in: International Conference on Swarm, Evolutionary, and Memetic Computing, Springer, 2011, pp. 148–156.

[38] S. C. Satapathy, A. Naik, K. Parvathi, Unsupervised feature selection using rough set and teaching learning-based optimisation, International Journal of Artificial intelligence and Soft Computing 3 (3) (2013) 244–256.

[39] Y. Xu, L. Wang, S.-y. Wang, M. Liu, An effective teaching–learning-based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time, Neurocomputing 148 (2015) 260–268.

[40] A. Baykasoğlu, A. Hamzadayi, S. Y. Köse, Testing the performance of teaching–learning based optimization (tlbo) algorithm on combinatorial problems: Flow shop and job shop scheduling cases, Information Sciences 276 (2014) 204–218.

[41] Z. Xie, C. Zhang, X. Shao, W. Lin, H. Zhu, An effective hybrid teaching–learning-based optimization algorithm for permutation flow shop scheduling problem, Advances in Engineering Software 77 (2014) 35–47.

[42] J. Zhou, X. Yao, Hybrid teaching–learning-based optimization of correlation-aware service composition in cloud manufacturing, The International Journal of Advanced Manufacturing Technology (2017) 1–19.

[43] R. Venkata, D. P. Rai, Optimisation of advanced finishing processes using a teaching-learning-based optimisation algorithm, in: Nanofinishing Science and Technology: Basic and Advanced Finishing and Polishing Processes, CRC Press, 2017, pp. 475–498.

[44] R. V. Rao, V. Patel, Multi-objective optimization of heat exchangers using a modified teaching-learning-based optimization algorithm, Applied Mathematical Modelling 37 (3) (2013) 1147–1162.

[45] R. V. Rao, V. Kalyankar, G. Waghmare, Parameters optimization of selected casting processes using teaching–learning-based optimization algorithm, Applied Mathematical Modelling 38 (23) (2014) 5592–5608.

[46] X. Qu, R. Zhang, B. Liu, H. Li, An improved tlbo based memetic algorithm for aerodynamic shape optimization, Engineering Applications of Artificial Intelligence 57 (2017) 1–15.

[47] J. Sacks, S. B. Schiller, W. J. Welch, Designs for computer experiments, Technometrics 31 (1) (1989) 41–47.

[48] S. Khan, E. Gunpinar, M. Moriguchi, Customer-centered design sampling for cad products using spatial simulated annealing, in: Proceedings of CAD17, Okayama, Japan, 2017, pp. 100–103.

[49] S. Khan, E. GÜNPINAR, An extended latin hypercube sampling approach for cad model generation., Anadolu University of Sciences & Technology-A: Applied Sciences & Engineering 18 (2).

[50] E. Myšáková, M. Lepš, Method for constrained designs of experiments in two dimensions, Engineering Mechanics (2012) 227.

[51] R. Rao, V. J. Savsani, D. Vakharia, Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems, Information Sciences 183 (1) (2012) 1–15.

[52] Ö. Yeniay, Penalty function methods for constrained optimization with genetic algorithms, Mathematical and Computational Applications 10 (1) (2005) 45–56.

[53] Á. E. Eiben, R. Hinterding, Z. Michalewicz, Parameter control in evolutionary algorithms, IEEE Transactions on evolutionary computation 3 (2) (1999) 124–141.

[54] R. Sarker, C. Newton, A genetic algorithm for solving economic lot size scheduling problem, Computers & Industrial Engineering 42 (2) (2002) 189–198.

[55] D. Powell, M. M. Skolnick, Using genetic algorithms in engineering design optimization with non-linear constraints, in: Proceedings of the 5th International conference on Genetic Algorithms, Morgan Kaufmann Publishers Inc., 1993, pp. 424–431.

[56] J. C. Kelly, P. Maheut, J.-F. Petiot, P. Y. Papalambros, Incorporating user shape preference in engineering design optimisation, Journal of Engineering Design 22 (9) (2011) 627–650.

[57] K. Holtzblatt, Customer-centered design for mobile applications, Personal and Ubiquitous Computing 9 (4) (2005) 227–237.

[58] H. Beyer, K. Holtzblatt, Contextual design: defining customer-centered systems, Elsevier, 1997.

[59] S. Khan, E. Gunpinar, K. M. Dogan, A novel design framework for generation and parametric modification of yacht hull surfaces, Ocean Engineering 136 (2017) 243–259.

[60] M. R. Sheldon, M. J. Fillyaw, W. D. Thompson, The use and interpretation of the friedman test in the analysis of ordinal-scale data in repeated measures designs, Physiotherapy Research International 1 (4) (1996) 221–228.

[61] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm and Evolutionary Computation 1 (1) (2011) 3–18.

[62] J. Derrac, S. García, S. Hui, P. N. Suganthan, F. Herrera, Analyzing convergence performance of evolutionary algorithms: A statistical approach, Information Sciences 289 (2014) 41–58.

[63] V. K. Patel, V. J. Savsani, A multi-objective improved teaching–learning based optimization algorithm (mo-itlbo), Information Sciences 357 (2016) 182–200.

[64] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: Nsga-ii, IEEE transactions on evolutionary computation 6 (2) (2002) 182–197.

[65] F. Zou, L. Wang, X. Hei, D. Chen, B. Wang, Multi-objective optimization using teaching-learning-based optimization algorithm, Engineering Applications of Artificial Intelligence 26 (4) (2013) 1291–1300.

[66] L. Lamberti, An efficient simulated annealing algorithm for design optimization of truss structures, Computers and Structures 86 (1920) (2008) 1936 – 1953.

[67] R. Rao, Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems, International Journal of Industrial Engineering Computations 7 (1) (2016) 19–34.