A Shape Sampling Technique via Particle Tracing for CAD Models

Erkan Gunpinar [a] Serkan Gunpinar [b]
[a] *Istanbul Technical University, Turkey*
[b] *Minneapolis, Minnesota, USA*

Abstract

In this paper, a shape sampling approach is proposed for CAD products that can be used to suggest innovative product shapes to designers and consumers. These shapes are intended to inspire designers and can be employed during the design process. For a given set of geometric parameters defining the product shapes, parameter relationships (i.e., geometric constraints), and parameter ranges, a particle tracing (PT) algorithm is proposed to find product shapes that satisfy the defined geometric constraints in the shape space. Particles are placed at points in the shape space by minimizing the Audze-Eglais potential energy of the particle positions using a permutation genetic algorithm. They then move until one of the predetermined stopping criteria is met. Particle movement is achieved using a cost function that favors movement towards feasible shapes. By iteratively running the PT algorithm, feasible shapes are obtained. Representatives of these shapes are identified using a k-medoids clustering approach, and such representatives can be used by designers or shown to consumers to customize the product according to their preferences. In this paper, eight CAD models (e.g., car hood, yacht hull, wheel rim) are utilized to validate the performance of the proposed sampling technique. We also compare our technique with related methods.

Key words: Computer-Aided Design, Generative Design, Constraint-based Sampling, Shape Space

1. Introduction

In today's market, products should attract customers not only by their performance or functionality, but also their external appearance. A product's external shape space consists of many shape variations. However, designers may not always anticipate all possible options, which may impede their ability to produce the most desirable product designs for consumers. We believe that a tool that performs the sampling of a product's shape space can help both designers and consumers in identifying the most appealing product shapes. If these shape variations are obtained automatically by sampling in the shape space, designers can utilize them to develop more visually appealing designs.

The design stage in engineering starts with determining the design specifications. There may be no CAD models or few such models available in the company model database. The designer first creates an initial CAD model if none are available and forms its design space using design specifications. The model is then modified to obtain as many variations as possible, which are validated through computer simulations or based on customer preferences. We think that it is preferable for designers to explore as many design variations of a CAD model as possible in advance before or during the design stage. Figure 1 (a) illustrates a design specification for a car hood, with its boundary in black. A CAD model is generated, as shown in Fig. 1 (b); the model is then modified to obtain its variations (see Fig. 1 (c)). The objective of this study is to perform sampling of a product's external shape within its shape (or design) space. To accomplish this, the product is first represented by geometric parameters; geometric constraints (i.e., relations between geometric parameters) are then determined. Finally, the geometric parameter ranges are identified. The shape space for the CAD model is formed using these geometric parameters and their ranges.

In this research, sampling quality is measured using two properties, namely *space-filling* and the computational time of the algorithm. The samples should have a good space-filling property so that they spread evenly in the constrained shape space to the greatest possible extent. Achieving such sampling in a shorter time is an important benefit in product design as well. The particle tracing (PT) algorithm, which was motivated by the motorcycle graph algorithm developed by Eppstein et al. [1], finds the shape variations of a given CAD product by inserting particles and moving them in the shape space. Specific tracing rules are introduced and applied for the particle movements. By

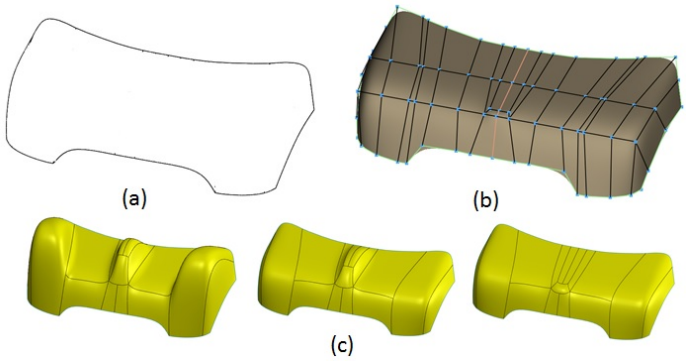

Fig. 1. After determining the design specifications (a) of a car hood (i.e., its boundary), its CAD model is generated (b). Distinct designs can be obtained by modifying the CAD model (c).

placing many particles in the n-dimensional shape space while minimizing the Audze-Eglais potential energy [2,3], different shape variations of the product can be generated. Such a particle placement strategy enables the even insertion of the particles into the shape space. The obtained product shapes are clustered using a k-medoids approach to identify a representative product shape of each cluster, which then can be shown to designers or consumers.

In summary, we present two main contributions in this paper, as follows:

- A sampling technique to obtain distinct CAD models that can be used in the design stage; and
- A particle tracing algorithm working jointly with the particle insertion method using Audze-Eglais potential energy.

The remainder of this paper is organized as follows: Section 2 reviews relevant literature. Section 3 discusses the proposed approach for generating new CAD models. The numerical results of the proposed approach are given in Section 4. Finally, concluding remarks and opportunities for future work are presented in Section 5.

2. Related Works

The research in this paper is more relevant to the generative design and shape exploration research fields; thus these fields represent the main focus here.

Generative design. Generative product design involves classes of design techniques that first examine design goals and then explore all possible permutations of a product. Generative design methods generally employ constraint-driven parametric search, and genetic and evolutionary algorithms. Yu et al. [4] proposed a constraint-based cooperative, interactive design method using genetic algorithms. In this approach, the design process is entirely driven by the designer and it represets the combination of constraint optimization and modeling. Caladas [5] developed an evolution-based generative design system for sustainable architecture called as GENE_ARCH. She demonstrated that building performances can be enhanced by the integration of parametric generative schemes and building simulation software for the evaluation of thermal performance. Bentley [6] first specified a phenotype in the design space, and the genotype was defined based on the solution space. To evolve the solutions, genetic algorithms were then utilized. Shape grammar [7] can also be used to represent geometry in a generative form, allowing a homogeneous model to be used for the design representation and the tools to generate it. Hornby and Pollack [8] developed L-system-based generative grammatical encoding. Their work showed that generative encoding of the genetic model can create significantly appropriate solutions. Gu et al. [9] suggested a neural network-based approach for exploring the designer's preference in selecting designs. Moreover, Krish [10] proposed a generative CAD-based design exploration method for complex multi-criterion design problems. First, a genotype of the design was built. Distinct designs were then obtained by randomly varying parameters within pre-defined limits. However, none of these approaches in the generative design field can perform automatic sampling while taking the space-filling property into account, which would enable the generation of distinct designs. Thus, we think that the techniques listed above can be enhanced by using the sampling method in this paper.

Shape Exploration. Forming a shape space and sampling from it has recently attracted a lot of attention in the computer graphics community. Ovsjanikov et al. [11] presented an approach for learning variability within a set of similar shapes. A deformation model is extracted automatically, which is then deformed through a set of intuitive deformation controls. Kalogerakis et al. [12] suggested a technique to generate new shapes by identifying new plausible combinations of segmented components from existing shapes. Chaudhuri et al. [13] proposed a technique to create visual content using relative semantic attributes (e.g., adjectives, such as "dangerous" or "scary") for design components. During the interactive design stage, different combinations of the components are selected and then assembled, and finally, novel designs are obtained. Kleiman et al. [14] proposed a dynamic map of shapes, where similar shapes are placed next to each other. Users browse shapes by dragging the map on the screen, which reveals new shapes that are comparable to shapes in the dragging direction. Fish et al. [15] introduced a meta-representation depicting a family of shapes, which can be used for exploration of shape repositories. Huang et al. [16] introduced a framework for computing consistent functional maps within shape collections. Many joint-shape analysis tasks, such as co-segmentation and shape exploration, are possible in this representation. Averkiou et al. [17] analyzed unorganized model collections to embed the models into low-dimensional spaces. Users can then explore the parametrized space to develop new models by probing the empty regions. In addtiton, Yumer et al. [18] proposed a method to create lower-dimensional and generative representations of high-dimensional procedural models using autoencoders. A shape design system was introduced for the generation of novel procedural models using an explore-and-select interaction. Gao et al. [19]

proposed an interactive exploration procedure for large 3D model repositories. The user labels the preferred models or parts so that recommended models are shown in the user interface. Recently, Schulz et al. [20] developed tools allowing interactive exploration and optimization of parametric CAD data. The above-mentioned techniques need model sets to be available in database to perform shape sampling. However, it is not always possible to have a proper data set of a product in the design phase. There can be few or no CAD models in the initial design stage. The proposed technique in this work allows us to generate design alternatives without having a proper model set. Furthermore, it is also possible to generate distinct designs via the proposed sampling technique, as a space-filling property is integrated into the technique.

Several works in the literature have explored shapes in architectural geometry. Yang et al. [21] presented a computational framework for shape exploration of constrained planar quad and circular meshes. Moreover, Zhao et al. [22] proposed intuitive tools to create new architectural free form shapes from an input design while conforming a set of prescribed constraints. However, these approaches are not applicable in our setting, as our approach is based on CAD models. There are also sampling algorithms in the literature, which can be adapted for shape sampling. Morris and Mitchell [23] generated maximin distance Latin hypercube designs via the spatial simulated annealing approach, taking sampling constraints into account. The Hit-and-Run (HAR) algorithm [24] samples over a convex polytope defined by linear constraints. The Bake-and-Shake (BAS) algorithm [25] performs sampling from the boundary of a convex polytope defined by a set of linear constraints. The introduced sampling technique in this paper exhibits better shape sampling performance in terms of space-filling, as we demonstrate later.

3. Particle Tracing (PT) Algorithm

The PT algorithm is inspired by the mesh segmentation techniques [1,26–28] that have become available in the literature. The motorcycle graph algorithm [1] developed by Eppstein et al. is introduced for quadrilateral mesh segmentation. Motorcycles or particles are inserted on the extraordinary vertices where the number of neighboring vertices is not four, except on the mesh boundaries. These particles move outward from the extraordinary vertices and trace the mesh along the edges. By doing this, quadrilateral partitions are obtained, where the traced mesh edges are on the partition boundaries.

3.1. *Method overview*

The proposed method consists of three main steps. After determining the processing time (t) and number of desired shapes (k) to generate, particles are inserted into the shape space. Geometric constraints and parameter ranges are given to the PT algorithm as input, and the particles are moved until reaching a position at which the geometric constraints are satisfied. The particle insertion and tracing steps are iteratively executed, and iterations terminate when the total processing time (t) reaches the user-defined execution time t in seconds. Finally, k representative shapes are computed using a clustering-based technique. Figure 2 shows the flow of the proposed method.

3.2. *Basic terminology and research goal*

The *shape* of a CAD product is represented and can be generated using *geometric parameters*. *Geometric constraints* define the mathematical relationships between geometric parameters via equalities and inequalities. Shape space S is an n-dimensional space where each geometric parameter is represented by a dimension in S. The shape space is formed by geometric parameters and their value ranges (i.e., lower and upper bounds). In this research, we assume that geometric parameters, geometric constraints, and parameter value ranges for a product's shape have been already determined in advance.

A *shape* is a point, and it is represented by coordinates $(\alpha_1, \alpha_2, ..., \alpha_n)$ of each dimension in S. These coordinates are the values of the shape's geometric parameters. The lower and upper bounds for the geometric parameters $(\alpha_1, \alpha_2, ..., \alpha_n)$ are denoted by $(\tau_1^l, \tau_2^l, ..., \tau_n^l)$ and $(\tau_1^u, \tau_2^u, ..., \tau_n^u)$, respectively. $\overline{S}$ represents the scaled shape space, where the coordinates $(\overline{\alpha}_1, \overline{\alpha}_2, ..., \overline{\alpha}_n)$ are the scaled parameter values varying between 0 and 1; these represent the lower and upper bounds, respectively, of the parameters $(\alpha_1, \alpha_2, ..., \alpha_n)$. In addition, $(\phi_1, \phi_2, ..., \phi_i)$ denote a set of geometric constraints where i is an integer. In fact, shape space consists of *feasible* and *infeasible* shapes. A shape is feasible if all geometric constraints are satisfied for its geometric parameters. Otherwise, it is infeasible.

The goal of this research is to perform shape sampling and find feasible shape alternatives with distinct geometries in the shape space. Enumerating these alternatives one-by-one in the shape space is impractical if n is large. For example, if three parameter values for each dimension within geometric parameter ranges are assigned in a 30-dimensional shape space, 3^{30} feasible and infeasible shape alternatives can be found, which is computationally expensive. As a result, a greedy approach is suggested to find feasible shapes in the shape space. The tracing methodology and particle insertion of the proposed method, which are employed to find feasible shapes in the shape space, will be elaborated later in the following subsections.

3.3. *Insertion of particles*

A single particle can only trace a small portion of the shape space S. Particles inserted at source points in different locations of S enable the generation of different feasible shapes. A careful selection of the source points is of pri-

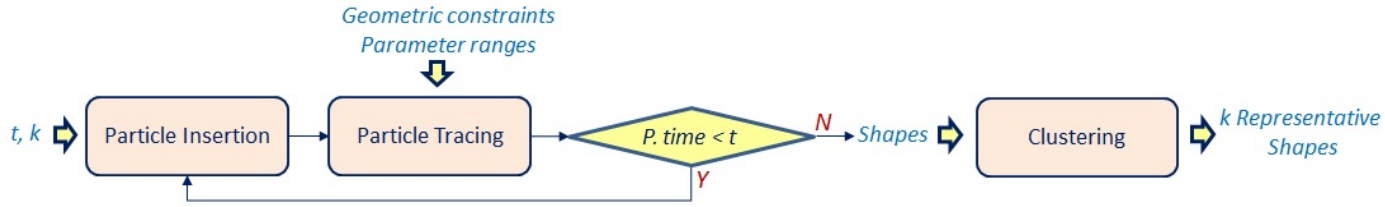

Fig. 2. The particle insertion and tracing algorithms are executed iteratively until reaching the user-defined execution time (t). k representative shapes are found next by means of a clustering technique.

mary importance. Source points should be evenly spread (i.e., *space-filling*) in the shape space to have a global exploration of the space. The method of Audze and Eglais [2,3] is utilized for the particle insertion, which is based on the analogy of minimizing forces between charged particles. Particles are in equilibrium when the potential energy is at the minimum. For the insertion of Y particles in the shape space, the potential energy E is introduced as follows:

$$E = \sum_{p=1}^{Y} \sum_{q=p+1}^{Y} \frac{1}{D_{pq}^2}, \tag{1}$$

where

$$D_{pq} = \sqrt{\sum_{m=1}^{n} (\overline{\alpha}_m^p - \overline{\alpha}_m^q)^2}. \tag{2}$$

Here, D_{pq} is the scaled distance in $\overline{S}$ between the source points p and q, whose coordinates are denoted by $\overline{\alpha}^p$ and $\overline{\alpha}^q$, resp.

Beyond the space-filling property, source points are chosen to be *non-collapsing* [29,3], which means that two points with a common parameter value should not exist. Without the non-collapsing property, the source points may spread over the shape space boundary for high dimensional space problems [30]. Because the energy function E is minimized when source points reside on the shape space boundary. The non-collapsing property can enable the generation of source points on the inner portions of the shape space. Figure 3 (a) illustrates a $2D$ shape space. The four source points in red are generated using only the space-filling property, and the points are located at the corners of the space.

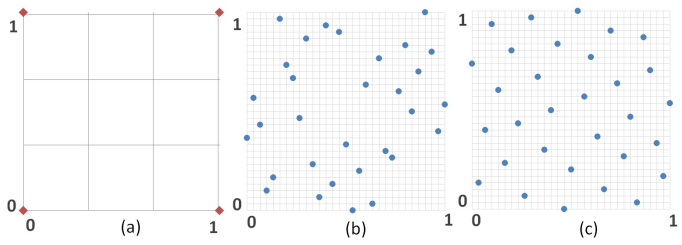

Fig. 3. (a) Generation of source points ($Y = 4$) considering only space-filling property; (b) source point generation considering only the non-collapsing rule ($Y = 30$); (c) source point generation considering both the space-filling and non-collapsing properties ($Y = 30$)

The optimization problem for the generation of source points requires the minimization of the potential energy E and can be formulated as:

$$min \rightarrow E, \tag{3}$$

with

$$0 \leq \overline{\alpha}_m^z \leq 1 \quad \text{for} \quad m = 1, \ldots, n \quad \text{and} \quad z = 0, \ldots, Y-1, \tag{4}$$

$$\psi_m^p \neq \psi_m^q. \tag{5}$$

The range of each geometric parameter is divided into $Y-1$ equally probable intervals. The scaled parameter value $\overline{\alpha}_m^z$ is obtained using Eq. 6 where ψ_m^z is the corresponding integer coordinate value between 0 and $Y-1$ for $\overline{\alpha}_m^z$ in the m^{th} dimension. Recall that $\overline{\alpha}_m^z = 0$ and $\overline{\alpha}_m^z = 1$ denote the lower and upper bounds, respectively, for the geometric parameter α_m^z.

$$\overline{\alpha}_m^z = \frac{\psi_m^z}{Y-1}, \tag{6}$$

where

$$0 \leq \psi_m^z \leq Y-1. \tag{7}$$

To find source points in the shape space, points are first randomly sampled while considering the non-collapsing property. These samples are then improved via a permutation genetic algorithm [31] to minimize the energy E (i.e., space-filling property). Single coordinates of two source points are swapped via cross-over operator if E decreases. Figure 3 (b) illustrates randomly sampled source points ($Y = 30$). Source points are evenly distributed after several swapping operations (c).

3.4. *Tracing methodology*

In a n-dimensional shape space, we take a similar particle tracing approach as that used in the previously suggested techniques [1,26–28] to find feasible shapes. As stated above, note that searching for feasible shape alternatives in the shape space is computationally expensive for larger n values. Thus, a greedy approach is adopted to find feasible shapes in the shape space. A particle is first inserted at a *source point* in the shape space. The particle is moved by

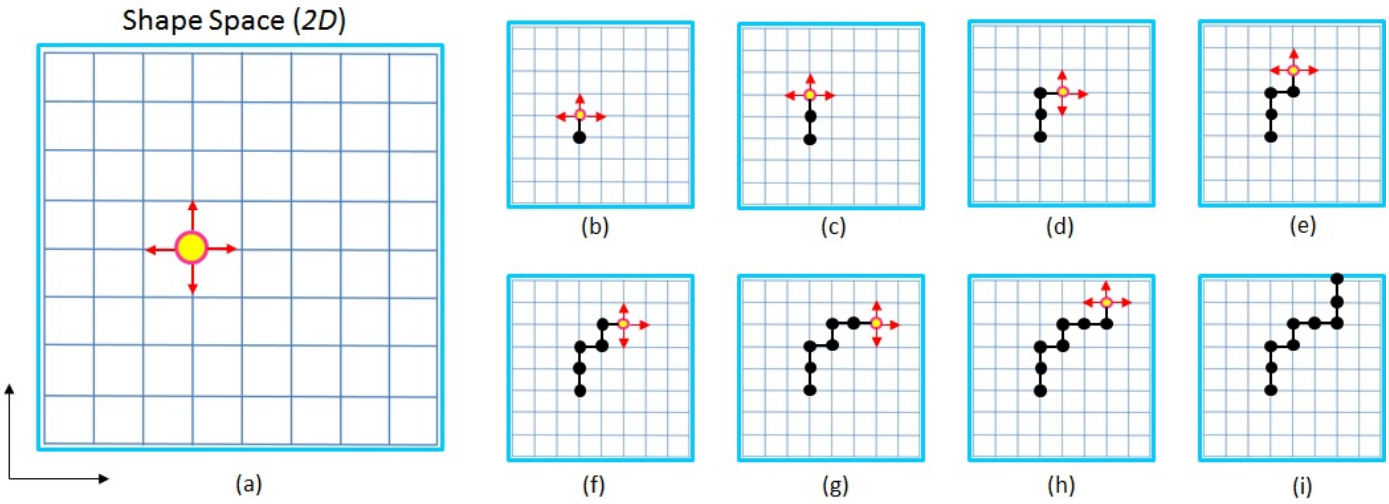

Fig. 4. *Two-dimensional particle tracing*: A particle (in yellow) is inserted at the source point and can be moved along four directions in the shape space(a). One of these directions is selected according to a cost function and moved along this direction. For the next particle tracing operations (b-i), there are at most three moving directions, and the particle is moved in a similar manner. The particle stops when it reaches the boundary of the shape space in light blue. Traced points are shown with black circles, some of which can represent feasible shapes.

some amount in a direction from this point and traces another point in each move in the shape space. If the traced point satisfies all the geometric constraints, it is regarded as *feasible shape*. Otherwise, it is *infeasible*. Particle tracing continues until the particle stops. Finally, one or more feasible shapes can be obtained. Note that the inserted particle does not always find a feasible shape in the shape space.

3.4.1. *Particle tracing rules*

In the n-dimensional shape space, a particle can move in $2n$ different directions. Movement direction is determined using a cost function, which is introduced below. The particle moves along one of 2n directions by an amount (i.e., step size) to minimize the cost function. Figure 4 demonstrates a particle movement in a two-dimensional shape space. For the sake of simplicity, tracing rules are explained in $2D$. These rules are similar for an n-dimensional shape space. As shown in Fig. 4 (a), there are four ($2n$) directions for the initial movement after inserting the particle into the source point. The particle moves along one of these directions, in which the cost function receives the lowest value. For the next movements (b-i), there are at most three ($\leq 2n-1$) movement directions. The particle moves until it stops. The traced points are stored and then checked in terms of whether they are feasible or infeasible shapes.

Particles are moved along the direction with a specified step size, which is computed based on an interval division constant ρ and a positive integer. The step size is computed separately for each dimension of the shape space. In the m^{th} dimension (where $m = 1, \ldots, n$), the step size s_m is calculated as follows:

$$s_m = (\tau_m^u - \tau_m^l)/\rho. \tag{8}$$

Particles moving in the shape space stop based on some *PT rules*. Figure 5 shows the termination criteria, which are detailed as follows:

(a) **Cross collision:** Two particles (in yellow and orange) meet at a point where moving directions are perpendicular to each other. One of these particles stops and other one continues tracing in the shape space;

(b) **Parallel collision:** Two particles (in yellow and orange) collide on a point where moving directions are parallel to each other. Both particles stop;

(c) **Meeting a previously traced point:** A particle (in orange) reaches a point that has already been traced by another particle (in yellow). The particle in orange stops, and the other one continues tracing;

(d) **Unimproved cost value:** A particle reaches a point where the cost value does not decrease further in any of the moving directions. In this case, the particle stops; and

(e) **Exceeding the shape space boundary:** Particle movement outward to the shape space boundary is restricted. If a particle exceeds the shape space boundary (light blue in Fig. 5), it terminates.

Note that the termination criteria (a), (b), and (c) are unlikely to occur in high-dimensional spaces.

3.4.2. *Cost function*

There are penalty function methods [32,33] that approximate a constrained problem by an unconstrained problem in such a way that minimization favors satisfaction of the constraints. These techniques add a penalty term to the objective function; this consists of a penalty parameter multiplied by a measure of violation of the constraints, which is nonzero when the constraints are violated and zero where they are not violated. Here, we take a similar approach. A cost function (F) is introduced based on the geometric

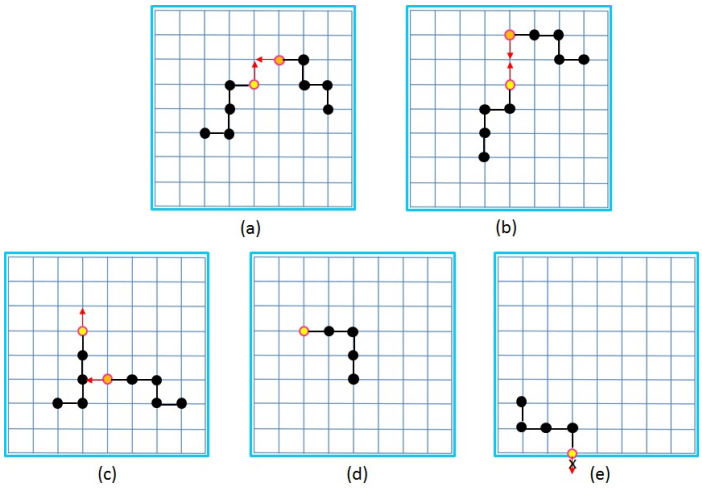

Fig. 5. *Termination criteria for particles*: (a) cross collision, (b) parallel collision (c) meeting a previously traced point, (d) unimproved cost value, (e) reaching shape space boundary.

constraints. F is computed separately for each particle position (i.e., point in the shape space) to determine particle movement directions. The function is formulated as follows:

$$F = \sum_{t=0}^{i} C(\phi_t). \tag{9}$$

Here, ϕ_t denotes a geometric constraint and the function $C(\phi_t)$ penalizes a shape or a point in the shape space S if it does not satisfy the constraint ϕ_t. Both equality and inequality geometric constraints can be handled in the cost function F. Let $f(\phi_t)$ denotes the equation for ϕ_t which can be 0 (greater/smaller than 0) for the equality (inequality) constraints (i.e., $f(\phi_0) = \alpha_1 - \alpha_3$, see Table 13). $C(\phi_t)$ is 0 if the equality or inequality constraint is satisfied. Otherwise, it is the absolute value of the $f(\phi_t)$ equation. Note that F is 0 for a feasible shape in the shape space. With this function, particle movements toward feasible shapes are favored. In each particle move, the cost value decreases and the particle moves closer to the feasible shapes.

3.5. *Proliferation of new particles*

Particles inserted at the source points move based on the PT rules, and some reach feasible points before termination. There can be more feasible points around a feasible point. Therefore, new particles are generated from a particle (called *parent particle*) reaching a feasible point in a process called *spawning*. Newly generated particles are called *spawn particles*. A parent particle spawns Y spawn particles.

Spawn particles can reside in a reduced shape space denoted by H, which represents a *hypercube*, where the center is the feasible point at which the parent particle is located. The maximum side length of H is denoted by r. The hypercube H is a sub-space in the shape space S, where $H \subset S$, and it is formed by computing lower and upper bounds

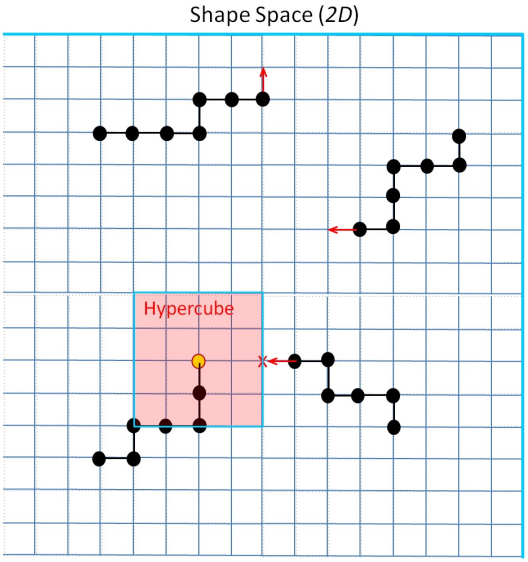

Fig. 6. A particle moves towards a feasible point (yellow dot) and stops. A hypercube (depicted with a pink rectangle) is formed around the feasible point where spawn particles will be inserted and move. Another particle on the right reaches the hypercube that has formed and terminates.

in each dimension using the center and side length. Spawn particles are located inside the hypercube while minimizing the potential energy E (in Equation 3) between these particles using the methodology described above.

Figure 6 illustrates a part of a shape space in $2D$, where the boundary is in blue. Four particles are inserted, and one of them reaches the feasible point shown by the yellow dot. A hypercube, depicted with a pink rectangle, is formed around this point. Spawn particles are inserted in this hypercube and move using PT rules until termination. Some of these particles are expected to reach feasible points. Note that all other particles will pause and wait for the termination of the spawn particles. The other particles then continue their movement in the shape space. A new PT rule is introduced here for the particles other than spawn particles.

(f) **Reaching a hypercube boundary:** If a particle reaches a hypercube boundary, it terminates (see the particle on the bottom right in Fig. 6), as it is assumed that spawn particles already performed a search there to find feasible points.

3.6. *Multiple runs of the PT algorithm*

The proposed PT algorithm generates feasible points, and the number of such points depends on the number of inserted particles Y. With the increase in Y, the number of feasible points obtained by the PT algorithm is expected to increase. However, it has been confirmed that the computational time taken for the particle insertion step is high [34] for larger values of Y; this is due to the swapping operations between coordinates of Y source points described previously. Therefore, we recommend a more practical us-

age scenario in this section.

The PT algorithm is run multiple times with lower values of Y. In each run, hypercubes formed during particle tracing are stored in a hypercube set denoted by H. Particles are inserted as far as possible from the hypercubes in the set H. Therefore, a new potential energy is introduced for the particle insertion that must be minimized, and this is defined as follows:

$$E_t = E + \theta * E_c, \tag{10}$$

where

$$E_c = \sum_{0}^{Y} \frac{1}{d_z^2}, \tag{11}$$

$$d_z = \begin{cases} 1e\text{-}9, & \text{if } p \cap \mathrm{H} \neq \emptyset \\ \min\limits_{p} D_{pb}, & \text{otherwise.} \end{cases} \tag{12}$$

The potential energy E_t has two terms and a coefficient. The function E is the previously defined potential energy in Section 3.2. The function E_c computes the proximity of source points to the hypercubes in H, and its weight is adjusted by the coefficient θ, which is set to 1.0 in this study. The empty set is denoted by $\emptyset$ in Eq. 12. Moreover, the value of d_z is close to 0 if the source point p is inside any hypercube in the set H. Otherwise, it is the distance D_{pb} denoting the scaled distance in $\overline{S}$ between the source point p and the closest hypercube boundary b in H. By including this parameter, we penalize source points that are in close proximity to the hypercubes in H. Recall that a feasible point search has been already performed in these hypercubes by the previous runs of the PT algorithm. Figure 7 illustrates particle insertion while considering hypercubes formed after the previous algorithm runs. New particles (in yellow) are inserted as far as possible from the hypercubes (in pink), as shown in Fig. 7 (b). As a result, different portions of the shape space can be searched via the PT algorithm. Finally, to store only distinct shapes, the parameter β is introduced, and this enables the storage of feasible shapes that have a minimum distance β (in the scaled shape space) from the previously traced feasible shapes. The parameter β is called the *sampling density*. If a distance between a feasible shape and the previously traced feasible shape is smaller than β, the one that is close to its nearest-neighbor is removed.

3.7. *Finding Representative Shapes*

A single run of the particle insertion and tracing algorithms does not guarantee to produce Y shapes, particularly for the constrained shape spaces (Y is user-defined). In the constrained spaces, a single run of the particle insertion and tracing steps may generate feasible shapes that are not very well distributed in the shape space as infeasible regions may spread irregularly. Via multiple runs of these steps, feasible shapes are obtained and only some *representative shapes* are chosen, which can be shown to the users. When the PT algorithm terminates after reaching the user-defined run time, the feasible shapes obtained are grouped into clusters. The most k representative shapes of each cluster, called *medoids*, are computed (see Fig. 2) via the $k-$medoids algorithm. We think that the cluster medoids are prototypes of their clusters and can represent them to some extent.

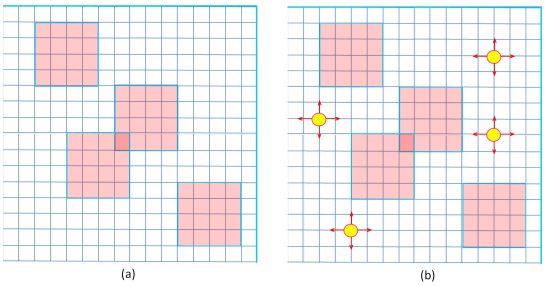

Fig. 7. Particle insertion considering the hypercubes formed after the previous algorithm runs.

Given a set of feasible points $(f_1, f_2, \ldots, f_s)$, k-medoids clustering aims at partitioning the s feasible points into k feasible point sets (i.e., clusters) $C = C_1, C_2, \ldots, C_k$ where $k \leq s$, $C_y \cap C_z = \emptyset$, $1 \leq y, z \leq k$, and k, s, y, and z are integers. The objective is to minimize the sum of the scaled distance (recall Eq. 4) in $\overline{S}$ from each point in the cluster C_i in relation to the cluster medoid μ_i. In other words, dissimilarity between any point and a medoid within a cluster is minimized so clusters are well represented by their medoids. The objective function is written as follows:

$$M = \sum_{i=1}^{k} \sum_{f \in C_i} \|f - \mu_i\|. \tag{13}$$

The partitioning around medoids (PAM) algorithm is used for k-medoid clustering, which is summarized below.

1. Select randomly k feasible points as the medoids.
2. Associate each feasible point with the closest medoid.
3. For each medoid μ_i and each feasible point f associated with μ_i, swap μ_i and f. Compute the objective function M. Select the feasible point with the lowest M value as the medoid of the cluster.
4. Repeat steps 2 and 3 until there is no change in M.

The k representative shapes are chosen to be the cluster medoids generated by the PAM algorithm.

4. Experiments and Discussion

4.1. *Test models*

For the validation of the proposed technique, eight test models are created, as follows: a ewer main body, a car body, a car hood, a yacht hull, a wheel rim, a wine glass, a bottle, and a park shed. These models, their geometric pa-

rameter with their ranges (i.e., lower/upper bounds) and their geometric constraints are outlined in Appendix. Recall that the objective of this study is to perform shape sampling for a given set of geometric constraints and parameter ranges, which are defined heuristically in this work. We initially define a set of geometric constraints and see the shapes generated after sampling. If implausible shapes are generated, new constraints are included in the sampling algorithm. By doing this iteratively, a set of geometric constraints is obtained. Parameter ranges are determined in a similar manner. Note that geometric constraints and parameter ranges must be selected carefully to form a better shape space containing many feasible shapes. The feasible shape space becomes narrower if redundant geometric constraints are introduced. In contrast, with fewer geometric constraints, shapes designated implausible by designers or consumers may be generated. A future work will be carried out to identify the proper set of geometric constraints for a given CAD product.

4.2. *Results*

Figures 8 and 9 show representative instance models for the eight CAD models generated by multiple runs of the PT algorithm for the $Y = 10$ and $t = 1200$ settings. It has been observed that distinct shapes in terms of appearance are obtained after application of the PT algorithm. For larger values of Y (i.e., number of inserted particles), the number of shapes N_S obtained is higher, as shown in Table 2. Computations are performed for different Y settings when the hypercube length r and the sampling density β are set to 0.5. N_S is squared exponentially proportional to Y. Recall that after a particle reaches a feasible point, Y spawn particles are generated from the particle and continue tracing in the sub-shape space. Y particles can generate at most Y^2 feasible points. Note that the swapping operation that is applied to distribute source points evenly is not utilized for the $Y = 50$, $Y = 100$, and $Y = 200$ settings due to its high computational cost [34]. Single run of the PT algorithm with a lower Y and multiple runs with a larger Y were tested. Former test was performed under a setting where Y is equal to 20. Its processing time was then set in the latter run and solution qualities were compared with different settings. Table 1 shows results of these runs when k value was chosen 10 in the clustering algorithm. It was observed that the potential energy (E) of the representatives in the multiple runs was generally less than those in the single run. Therefore, the representatives in the multiple runs mainly achieved better space-filling. Table 3 shows the number of shapes generated in a single run with $Y = 10$ for the ewer main body model. It can be observed that the number of shapes obtained differs for each run and varies between 22 and 49. This is because of the randomization process that is used for finding the initial source point positions before the swapping operation. As a result, we recommend multiple runs of the PT algorithm for the generation of more feasible shapes.

Table 1
Single and multiple runs of the PT algorithm (s: seconds)

Model	Single Run			Multiple Runs		
	t_p	N_s	E	t	N_s	E
Ewer	460 s	156	8.46	460 s	580	8.98
Car body	252 s	84	16.74	252 s	173	9.02
Hood	290 s	130	40.95	290 s	1134	39.06
Yacht Hull	157 s	21	53.11	157 s	179	43.53

N_S	45	50	22	49	32	34
t_p	19.1	20.7	19.5	21.8	16.6	14.6

Table 3
Performance of the proposed algorithm for the ewer main body with the $Y = 10$, $r = 0.5$ and $\beta = 0.5$ settings

Computational time: A PC with an Intel Core $i7$ 6700 3.4 GHz processor and 16 GB memory is used for the experiments in this study, and the implementation is single-threaded. Tables 2 and 3 show the computation time taken for the PT algorithm. The processing time t_p for the ewer main body model varies between 15 and 1451 seconds, depending on the Y settings. For the car side silhouette model, the time taken for the proposed method is between 9 ($Y = 10$) and 1057 ($Y = 200$) seconds. Since inserting many particles in the shape space increases the number of particle tracing operations, the processing time increases for larger values of Y. Recall that setting large values for Y enables the generation of distinct shapes in the shape space. In contrast, for larger values of Y, such as 1000, the PT algorithm is computationally expensive, as the algorithm is exponentially proportional to Y values. Therefore, we recommend the application of multiple algorithm runs using smaller Y values. The computation time also depends on the interval division constant ρ, which is based on the step size selected for the PT algorithm. In our experiments, ρ is set to 1000. Setting it to larger values will increase the processing time, as particles move with smaller steps. However, setting smaller values will decrease the number of shapes generated.

PT algorithm runs: The PT algorithm is executed multiple times to obtain more shapes. It has been observed that the number of shapes generated increases with the computation time of the algorithm. In other words, the longer the algorithm runs, the greater the number of feasible shapes generated. Let Ω be the nearest-neighbor distances for each feasible point in the feasible shape set ζ: $\Omega = \{\min D_{pq} : p \in \zeta\}$. Nearest-neighbor distances are computed and stored in Ω for every point in the set ζ. Recall that D_{pq} is computed using Eq. 4. D_{ave} denotes the average distance of the distance set Ω, while D_{max} is the maximum distance in Ω. Upon a completion of each run, D_{ave} and D_{max} are computed. Figure 10 shows that these values for the ewer body, yacht hull and park shed models. Both D_{ave} and D_{max} get closer to be stabilized after approximately 150, 200 and 400 runs, respectively. To some

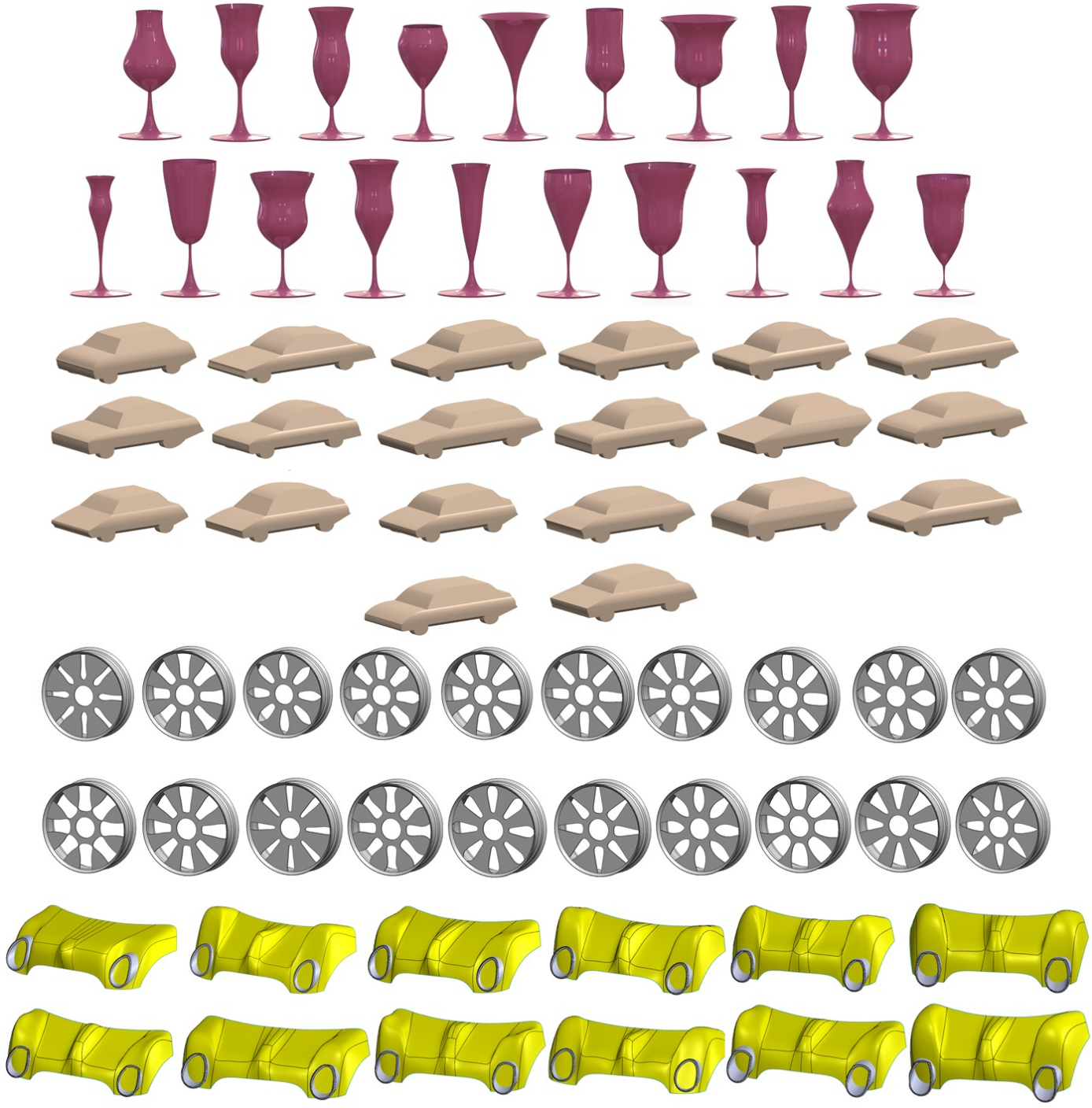

Fig. 8. Wine glass, car body, wheel rim, and car hood models generated by the proposed technique.

Parameter	Ewer Main Body					Car Body					Car Hood	Yacht Hull	Wheel Rim	Wine Glass	Bottle	Park Shed
Y	10	20	50	100	200	10	20	50	100	200	10	10	10	10	10	10
N_S	40	150	1202	4431	18110	11	45	390	1768	7441	56	15	38	56	62	11
t_p	15.0	424.4	106.5	369.4	1451.2	9.3	150.9	64.9	295.8	1057.9	3.5	2.1	2.5	4.1	3.6	2.1

Table 2
Performance of the proposed algorithm with different Y settings when $r = 0.5$, $\beta = 0.5$ (N_S = Number of shapes obtained, t_p = Computational time in seconds)

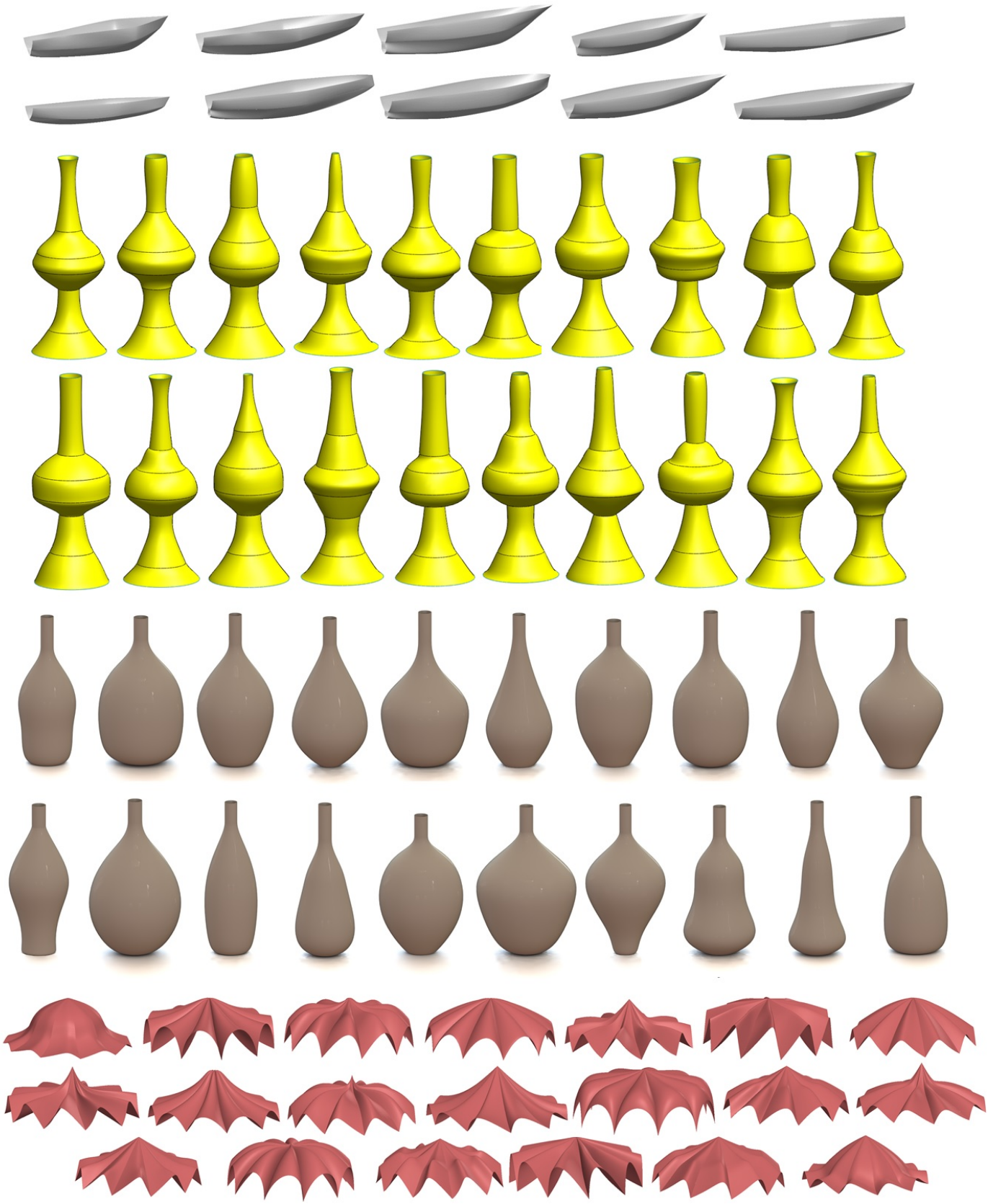

Fig. 9. Yacht hull, ewer main body, bottle, and park shed models obtained in the proposed work.

extent, this is an indication that the feasible shapes are well distributed in the shape space.

Setting algorithm parameters: Since the shape space is n-dimensional, the diagonal of the scaled shape space $\overline{S}$ is $\sqrt{n}$. For the CAD models used in this work, the diagonal length of their scaled shape space is $\sqrt{38}$. In light of this information, β is assigned a value of 0.5 in the experiments. When β is set to smaller values, the number of shapes obtained by the proposed algorithm increases, and shapes that are similar in appearance will exist in the obtained shape set. The hypercube length r is also set to 0.5 in the experiments. Assigning higher values, such as 2.0, for this parameter has been attempted, but few shapes are obtained by the PT algorithm, as the computed source points always reside in the generated hypercubes. Therefore, the particles inserted there stop according to the tracing rule (f) above.

4.3. *Comparing the proposed work with the existing methods*

Here, each step of the proposed technique (i.e., particle insertion, particle tracing and representatives selection) is evaluated separately.

Alternative methods for particle insertion: Random sampling (RS) and low discrepancy sequence (LDS) of Halton [35] are utilized to insert particles evenly in the shape space. Table 4 shows the experimental results with the $Y = 10$ and $Y = 20$ settings for all eight test models. It was observed that potential energies for the particles generated by RS and LDS are higher than those of the proposed insertion technique, and therefore, the particles obtained by our technique have a better space-filling property.

Alternative techniques for moving particles: The particle tracing step of the proposed method is compared with a nonlinear solver, which brings an infeasible point to a nearby feasible point. In short, a feasible point is found that is closest to the particle placed in the particle insertion step while the cost function F in Eq. 9 is forced to zero at the feasible point (i.e., all geometric constraints are satisfied). The nonlinear solver, SNOPT of Gill et al. [36,37], applies a sparse sequential quadratic programming algorithm, using limited-memory quasi-Newton approximations to the Hessian of the Lagrangian. SNOPT was replaced with the particle tracing technique and the results were analyzed in terms of two metrics: the potential energy, E, between resulting k-representatives and the variance measuring how well the representative represents the cluster. Variance (var) is computed using the below equation:

$$Var = \frac{\sum_{i=1}^{k} \sum_{f \in C_i} (\|f - \mu_i\|)^2}{s}. \qquad (14)$$

Here, C_i and μ_i denote, respectively, the cluster and cluster representative where i varies between 1 and k. $\|f - \mu_i\|$ is the Euclidean distance between the feasible shape f in C_i and the cluster center μ_i of C_i. s denotes the total number of feasible shapes in the clusters.

Table 5 shows the results with the $k = 10$ and $t = 600/1200$ settings for the ewer body, car body, car hood and yacht hull models. We observed that the PT algorithm mainly generated better results (i.e., less potential energy and less variance) than those of the nonlinear solver. Less potential energies of the representatives for the PT algorithm were achieved except in case of the yacht hull model experiment with the $t = 600$ setting. Finally, in terms of variance, the PT algorithm generated lower variance values for the ewer body, car body and yacht hull models.

Instead of moving a particle in a single direction during particle tracing, the particle is moved in the gradient descent (GD) direction, which is computed using finite differences. g denotes the GD direction at a particle location, which is as follows: $g = [g_1, g_2, \cdots, g_n]^T$, where n is the number of dimensions in the shape space. Recall that $\alpha = (\alpha_1, \alpha_2, ..., \alpha_n)$ denotes the particle location and F_α represents the constraint violation function (see Eq. 9) for α. Let α^{m+} and α^{m-} be, respectively, the forward and backward positions when the particle moves, respectively, in the positive and negative m^{th} directions by the step size s. The gradient g_m in the m^{th} direction is computed as follows: $g_m = (F_{\alpha^{m+}} - F_{\alpha^{m-}})/2s$. Let $\overline{g}$ be the normalized vector of g. The new location α' of a particle is found using the following equation: $\alpha' = \alpha + \overline{g} * s$. In the GD method, the particle moves until F is zero or greater than its previous position. To validate the performance of this method, s is set to 0.1, 0.01, 0.001 and 0.0001. Table 7 (c) shows the results for the six test models. Note that the GD method was replaced with the particle tracing technique and all other steps (i.e., the particle insertion and representatives' selection stages) remained same. It was observed that the GD method produced less number of feasible shapes compared to the particle tracing algorithm. In most cases, the PT algorithm produced more evenly sampled shapes (i.e., minimization of the potential energy E) in the shape space compared to the GD technique.

Alternative techniques for shape representative selection: There are several methods in the literature that can be used for the selection of shape representatives. k-means and mean shift algorithms are appropriate for these selections, and thus, they are analyzed here. The results are evaluated using two metrics, as follows: space-filling (E) and variance (var).

Table 6 shows the clustering results in terms of space-filling and variance. Experiments are performed for all eight models, and parameter settings are shown in the table. The same number of clusters are obtained for all the algorithms. Mean shift has a window radius (h_w) parameter and appropriate values of h_w for the test models to obtain same number of clusters listed in Table 6. According to these results, representatives obtained by the k-medoids have better space-filling properties than those of mean shift and k-means algorithms. Based on the experiments conducted, there were cases where k-medoids performed both better

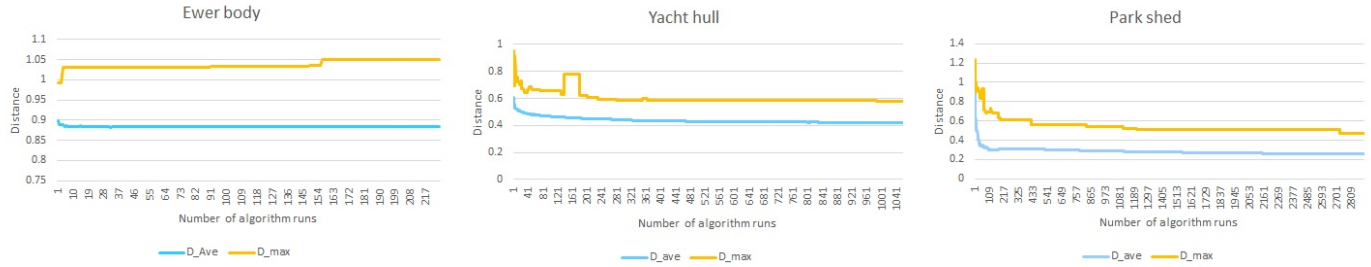

Fig. 10. D_{ave} and D_{max} are, respectively, the average and maximum nearest-neighbor distances for the feasible shapes obtained. D_{ave} and D_{max} are close to stable values after running the proposed method several times.

Table 4
Comparison of the proposed particle insertion technique with alternative methods (E: potential energy, t_p: processing time in seconds)

Model	$Y=10$						$Y=20$					
	RS		LDS		Our Technique		RS		LDS		Our Technique	
	E	t_p	E	t_p	E	t_p	E	t_p	E	t_p	E	t_p
Ewer body	27.82	<0.1s	9.54	<0.1s	**5.28**	0.76s	121.58	<0.1s	36.62	<0.1s	**25.71**	21.6s
Car body	27.22	<0.1s	9.25	<0.1s	**5.2**	0.64s	121.78	<0.1s	37.2	<0.1s	**26.18**	33.06s
Car hood	102.24	<0.1s	32.64	<0.1s	**18.14**	0.12s	470.16	<0.1s	133.28	<0.1s	**94.73**	6.8s
Yacht hull	104.62	<0.1s	31.93	<0.1s	**17.66**	0.17s	462.11	<0.1s	129.29	<0.1s	**90.94**	7.52s
Wheel rim	118.2	<0.1s	35.97	<0.1s	**19.48**	0.16s	533.9	<0.1s	148.19	<0.1s	**100.48**	6.16s
Wine glass	67.32	<0.1s	22.16	<0.1s	**12.2**	0.23s	309.4	<0.1s	101.37	<0.1s	**62.36**	14.3s
Bottle	68.64	<0.1s	23.92	<0.1s	**12.45**	0.2s	294.56	<0.1s	93.28	<0.1s	**62.0**	5.43s
Park shed	159.18	<0.1s	46.84	<0.1s	**24.59**	<0.1s	680.2	<0.1s	199.52	<0.1s	**126.36**	6.6s

Table 5
Comparing the PT algorithm with a nonlinear solver (PT: The PT algorithm, NLS: The nonlinear solver SNOPT)

Model	t=600				t=1200			
	PT		NLS		PT		NLS	
	E	var	E	var	E	var	E	var
Ewer body	**9.544**	3.589	9.752	3.670	**10.298**	3.597	10.301	3.668
Car body	**11.286**	2.495	12.684	2.819	**12.726**	2.602	13.595	2.686
Car hood	**36.547**	0.756	41.559	0.651	**39.147**	0.759	42.167	0.648
Yacht hull	56.236	0.502	**52.579**	0.581	**48.570**	0.515	52.431	0.605

and worse than mean shift and k-means algorithms in terms of variance. We have observed that computational time for the mean shift algorithm is higher than those of other two techniques (see Table 6). As a result, we recommend using $k-$medoids technique to obtain shape representatives.

Comparing the proposed method with the spatial simulated annealing technique: We compare our work with spatial simulated annealing (SSA) technique [38–40], which is implemented and adapted to the research problem discussed in this paper. Sampling quality is measured based on the computational time and the space-filling property (the Audze-Eglais potential energy E, Equation 3). The PT algorithm generates many shapes; thus, we choose k samples among them that minimizes E via a greedy approach.

The restricting distance H will be shortened as the simulated annealing temperature decreases and the initial length of H is taken as 1.0. The initial temperature is set to 80.0, and the algorithm stops when it becomes less than 0.00008. S denotes the shortening rate for T, and different S values are assigned for the experiments. The Markov chain length L is set to 10 times greater than the sample number Y, as suggested by Chen et al. [39]. For a given k samples, a new solution S_{i+1} is generated from S_i by means of a random perturbation in one of the samples in S_i. Chen et al. moved the randomly chosen point over a vector $\vec{h}$, with the direction chosen randomly. The length $\left|\vec{h}\right|$ of $\vec{h}$ takes a random value between 0 and H. When such perturbation is applied to the problem in this paper, the sample being perturbed is often invalid, as it is easy for it to flee out of the sampling region (shape space). Therefore, $\vec{h}$ is set to be one-directional, and only one coordinate of the sample is perturbed. As a fitness function, the Audze-Eglais potential energy E (see Eq. 1) is used in the SSA

Table 6
Comparing k-medoids with other clustering techniques (s =seconds, m =minutes, hr =hour)

Model	Mean shift			k-means			k-medoids			# of Shapes	Parameter values
	E	var	t_p	E	var	t_p	E	var	t_p		
Ewer body	43.394	2.525	5.53 m	46.873	3.313	0.059s	**36.081**	3.232	0.224 s	976	$Y = 10, t = 1200, k = 20, h_w = 1.7895$
Car body	53.36	1.804	2.46 m	49.962	2.370	<0.05 s	**48.609**	2.292	<0.05 s	467	$Y = 10, t = 1200, k = 20, h_w = 1.585$
Car hood	157.582	0.758	4.41 hr	55.873	0.741	0.109 s	**43.116**	0.722	0.531 s	2054	$Y = 10, t = 1200, k = 10, h_w = 0.829$
Yacht hull	89.68	0.598	2.79 m	55.495	**0.508**	<0.05 s	**49.696**	0.515	<0.05 s	437	$Y = 10, t = 1200, k = 10, h_w = 0.689$
Wheel rim	632.556	0.746	1.83 hr	247.167	0.634	0.094 s	**162.538**	0.622	0.35 s	1742	$Y = 10, t = 1200, k = 20, h_w = 0.773$
Wine glass	137.433	1.156	34.7 hr	131.904	1.051	0.10 s	**106.403**	1.061	0.30 s	1764	$Y = 10, t = 600, k = 20, h_w = 0.9735$
Bottle	181.1	1.1	53.4 m	118.943	1.255	0.09 s	**98.579**	1.166	0.31 s	1604	$Y = 10, t = 400, k = 20, h_w = 1.062$
Park shed	694.726	0.68	1.5 hr	382.242	0.535	0.10 s	**189.524**	0.493	0.28 s	1907	$Y = 10, t = 400, k = 20, h_w = 0.639$

approach. The initial solution (S_0) consists of only feasible samples (satisfying predefined geometric constraints) that are obtained using a random perturbation of samples. If a feasible sample does not satisfy the geometric constraints after perturbation, that perturbation is not performed.

To compare the PT algorithm with the SSA approach, the car body model with the geometric ranges and constraints (see Tables 15 and 16) is used. Table 7 (a) shows the computational time t_p of the SSA approach and E for different values of S and k. As S nears 1, E decreases with the increase in t_p. Table 7 (b) depicts the performance of the PT algorithm using multiple runs with $Y = 10$ and different t settings. For $k = 10$, the processing time t_p is 3004.3 seconds, and the potential energy E of the k samples is 5.46. The potential energy E for the SSA approach is 6.5 with a processing time t_p of 4869.0 seconds. For $k = 100$, the PT algorithm generates samples with lower potential energies than those of the SSA approach. As a result, the PT algorithm generates samples with better space-fillingness than those of the SSA approach in a shorter time.

We further tested all eight models using the PT algorithm with the $Y = 10$ and $k = 20$ settings. T and S were set to 0.00008 and 0.9999, respectively, for the SSA approach. We first executed the SSA algorithm and the parameter t in our method was set to the processing time of the SSA approach. Table 7 (c) shows the performances of both methods. According to the experimental results, k shapes obtained by our technique have lower potential energies than those produced by the SSA algorithm. In other words, our technique generates shapes that are more evenly spread throughout the shape space.

Comparing the proposed work with the random sampling and low discrepancy sequence: The proposed work is also compared with the random sampling (RS) and the low discrepancy sequence (LDS) of Halton [35]. No subsequent optimization was utilized in these approaches. The RS method was executed until desired number (i.e., 10) of feasible shapes is obtained, while the proposed work run only once with the $Y = 10, k = 10$ settings, and the number of samples to generate was set to a number (i.e., Y) in the LDS approach. Table 8 shows the results for all three methods. Results show that the proposed approach generated feasible shapes with potential energies less than those RS and LDS. Additionally, when the LDS algorithm was executed to generate 1000 shapes, it could not find feasible shapes in some cases (i.e., ewer body, car body and yacht hull models).

Table 8
Comparison of the proposed work with the random sampling (RS) and low discrepancy sequence (LDS) (t_p = processing time, NA = Not Available).

Model	Proposed Work ($Y = 10, k = 10$)		RS		LDS		
	E	t_p	E	t_p	E	t_p	Y
Ewer body	**8.01**	14.1s	22.41	2.96s	NA	0.82s	1000
Car body	**12.31**	19.44s	36.32	4.82s	NA	0.81s	1000
Car hood	**19.12**	2.31s	29.64	<0.1s	29.07	<0.1s	30
Yacht hull	**38.25**	1.25s	50.99	0.66s	NA	0.5s	1000
Wheel rim	**19.63**	1.73s	33.57	<0.1s	36.31	<0.1s	10
Wine glass	**16.87**	2.0s	26.39	0.22s	36.39	<0.1s	50
Bottle	**14.47**	2.64s	20.28	<0.1s	19.8	<0.1s	14
Park shed	**26.28**	1.24s	43.92	<0.1s	43.9	<0.1s	10

Additional experiments: The potential energy E is also considered while moving particles in the shape space. A new cost function is employed as follows: $F' = F + c * E$. c is a user-defined positive integer, and set to 0, 0.1, 0.5 and 1 in the experiments. Table 9 shows results for six test models where the proposed algorithm runs once as well as multiple times ($t = 300$). Both Y and k are set to 10. In most cases, better results (i.e., minimization of E) were obtained when c is set to 0. It is probably because setting c to a value different from 0 gives a lower freedom to the particles to move towards the feasible shapes during the particle tracing step. Notice that less numbers of feasible shapes (i.e., N_s) were obtained when c is greater than 0. Based on these experiments, we recommend utilizing only the cost function F in the particle tracing stage.

Table 7
Performance of approaches considered in this study. (a) The spatial simulated annealing approach, (b) multiple runs of the PT algorithm for the car body model, (c) The SSA algorithm, the proposed approach with the PT algorithm and with the gradient descent (GD) method (t_p, t: respectively, computational and processing time in seconds, N_s: number of shapes after the PT algorithm).

(a)

Parameter	The Spatial Simulated Annealing Approach						
k	10	10	10	100	100	100	100
S	0.99	0.9999	0.999999	0.99	0.999	0.9999	0.99999
t_p	1.6	58.2	4869.0	633.0	1111.2	4812.8	10827.4
E	10.6	10.3	6.5	1212.4	1189.4	1100.2	735.5

(b)

Parameter	Proposed Work ($Y = 10$)									
k	10	10	10	10	10	100	100	100	100	100
t_p	65.3	124.9	241.2	1006.6	3004.3	123.8	245.2	1009.2	3012.2	10007.8
E	6.84	6.71	5.99	5.65	5.46	1027.6	961.9	776.2	722.1	684.6

(c)

Model	SSA ($k = 20$)		Proposed Work ($Y = 10, k = 20$)			GD ($s = 0.1$)			GD ($s = 0.01$)			GD ($s = 0.001$)			GD ($s = 0.0001$)		
	E	t_p	E	t	N_s	E	t	N_s	E	t	N_s	E	t	N_s	E	t	N_s
Ewer body	33.581	127s	**26.63**	127s	328	*28.24*	127s	87	28.48	127s	101	29.86	127s	79	40.52	127s	31
Car body	42.304	145s	**32.04**	145s	196	*36.09*	145s	50	37.08	145s	72	60.22	145s	22	-	255s	5
Car hood	109.727	79s	**78.87**	79s	837	84.06	79s	452	81.83	79s	791	83.83	79s	543	94.68	79s	44
Yacht hull	157.786	71s	**109.86**	71s	164	127.52	71s	95	119.17	71s	132	*118.13*	71s	136	135.12	71s	63
Wine glass	73.745	60s	56.24	60s	806	56.64	60s	425	**55.79**	60s	505	56.0	60s	575	56.23	60s	475
Bottle	62.033	63s	**50.53**	63s	913	57.32	63s	86	56.26	63s	109	56.57	63s	68	56.11	63s	70
Wheel rim	99.84	57s	**83.48**	57s	450	-	-	-	-	-	-	-	-	-	-	-	-
Park shed	115.131	103s	**69.36**	103s	481	-	-	-	-	-	-	-	-	-	-	-	-

Table 9
New cost function performance during particle tracing (i.e., $F' = F + c * E$)

Model	Single run								t=300							
	$c = 0$		$c = 0.1$		$c = 0.5$		$c = 1.0$		$c = 0$		$c = 0.1$		$c = 0.5$		$c = 1.0$	
	E	N_s	E	N_s	E	N_s	E	N_s	E	N_s	E	N_s	E	N_s	E	N_s
Ewer body	**9.26**	38	9.39	34	9.43	30	12.84	16	**5.59**	544	5.77	139	9.03	34	8.75	23
Car body	**12.28**	21	-	0	-	0	-	0	**6.73**	155	-	0	-	0	-	0
Car hood	**20.71**	64	32.9	28	33.0	26	33.0	26	**16.56**	1563	16.97	942	17.26	938	16.94	734
Yacht hull	**51.31**	16	-	5	-	5	-	1	**20.13**	336	26.3	61	36.47	17	-	9
Wine glass	**18.93**	54	23.6	26	41.88	12	41.88	12	**11.4**	2098	12.11	749	11.61	675	11.96	649
Bottle	20.81	41	**20.79**	42	**20.79**	42	**20.79**	42	**10.68**	2129	10.9	1931	10.9	1931	10.9	1931

4.4. *User study*

A user study involving ten users was conducted with the wheel rim and car body models to highlight the efficacy of the proposed sampling algorithm. First five users had professional experience, with an average of three years in product design and the other five were students who had design experience in industrial projects. First, models were randomly sampled in the wheel rim and car body shape space, respectively. They were then shown to the users to familiarize them with the models in the shape space. Two tasks were given to the users for each model. In the first task, a CAD model was provided to the users so that they could make modifications to it by changing control point positions. Each user was then asked to develop five diverse models of their liking using a CAD tool. Next, the models sampled with $Y = 10$ and $t = 2400$ (30 models for the car body and 20 models for the wheel rim) were shown to users. As a second task, they were asked to select five diverse models among the sampled models according to their preference.

We evaluated the results based on the time taken to design or select five models and user grading. Each user graded the five models separately using a Likert scale and the average was calculated. Table 10 (a) summarizes the findings. For the wheel rim model, average satisfaction score was 3.94 (μ) with a standard deviation (σ) of 0.4 in the first task, which is lower than in the second task (4.2 ± 0.45). Furthermore, average time to develop five diverse designs was 11.4 minutes, and the total time to select designs among the sampled ones was about 3.4 minutes. Satisfaction scores for the first and second tasks were almost similar when the car

body model is utilized. However, time for selection in the second task was lower than the design development time in the first task. Following the design selection session, each user was asked to complete a survey including three questions based on a Likert scale. The results indicated that the users have had positive feedback about the proposed sampling technique. Table 10 (b) summarizes the results.

Another user study was conducted to compare the proposed approach with the SSA algorithm and Krish's generative design technique [10] on the wine glass models. Each of these technique produced 20 glass models without geometric constraints. By setting the creativity value in Genoform generative design software of Krish to 50%, the parametric bounds in Table 12 were generated, and utilized in our sampling technique and in SSA executed approximately 120 seconds. Seven users without any design experience were first asked to choose five diverse glass models they like among the 20 models. They were then scored the models based on a Likert scale. Note that the survey participants did not have any information about the techniques that were used to generate the models. Furthermore, the user studies were conducted with the help of two persons who did not know about the user study goals.

According to the results, our technique generated models that are more evenly distributed (i.e., a lesser potential energy E) in the shape space compared to those obtained by other techniques (see Table 11). Furthermore, average scores of the participants scoring on five preferred wine glass shapes were, respectively, 4.09, 4.17 and 4.11, for the models generated by Genoform, our technique and the SSA approach. The standard deviations are 0.53, 0.57 and 0.59. It should be noted that Genoform can hardly be used in the constrained shape spaces. Figure 12 shows the wine glass models generated using Genoform, our technique and the SSA approach.

Finally, we performed a statistical test on the user study results shown in Tables 10 (a) and 11, which are summarized in Fig. 11 including p-values obtained from Friedman tests in pair-wise analysis, and Box and Whisker plots. The Friedman test is appropriate as the sample size of the user studies are small. The null hypothesis in Friedman test states no significance in difference between the tasks or approaches. If the p-value is less than 0.05, there is a strong evidence that the null hypothesis is rejected. Thus, results of the first and second tasks for the wheel rim model are different as the p-value is close to 0.05. However, the null hypothesis holds true for the other user study results.

Table 11
Comparison of the proposed technique with the alternative techniques based on the energy E, survey results and usability in the constrained spaces (Survey scoring - 1: Poor, 2: Fair, 3: Good, 4: Very good, 5: Excellent. μ: Average score of the participants scoring five preferred wine glass shapes and σ: Standard deviation))

Method	E	μ	σ	Ability to utilize in constrained spaces
Genoform	89.16	4.09	0.53	No
Our Technique	48.51	4.17	0.57	Yes
SSA	63.57	4.11	0.59	Yes

Table 12
Lower and upper bounds for the wine glass model used in the user study shown in Table 11

Bounds	α_0	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9	α_{10}	α_{11}	α_{12}	α_{13}	α_{14}	α_{15}
Lower	0.38	0.58	0.5	3.3	0.36	7.25	5.85	10.55	4.24	14.5	2.54	18	6.25	20.75	6.65	47.2
Upper	1.12	1.76	1.5	9.9	1.08	21.75	17.55	31.65	12.7	43.5	7.6	54	18.75	62.25	19.95	47.8

5. Conclusions and Future Works

This paper proposed a generative design method which consists of three algorithm steps and produces variations of a CAD model. The shape space for a product is defined by geometric parameters, parameter ranges, and geometric constraints. Particles are inserted in the shape space considering space-filling property, which is achieved using the Audze-Eglais potential energy between particles. Particle tracing algorithm with specific rules then finds feasible shapes in the shape space. Finally, K-medoids algorithm is employed to find shape representatives, which can be shown to users.

In future work, a combination of shape features such as light field descriptors with geometric parameters will be employed in the sampling method. We think that this enables us to generate visually more distinct models. In addition, the determination of geometric parameter ranges will be studied further, which is crucial to forming a better shape space that contains many more feasible and interesting shapes. Setting an appropriate number of proper geometric constraints will also be investigated. Finally, the choices of a more relevant distance metric will be studied as Euclidean distance is less meaningful for high-dimensional data [41].

Acknowledgements

The authors would like to thank Shahroz Khan for drawing the CAD models in this work, and the Scientific and Technological Research Council of Turkey for supporting this research (Project Number: 315M077).

Appendix

Ewer main body. Figure 13 (a) shows a ewer main body [42] which is represented with six cubic Bezier curves (*C1, C2, ..., C6*), as depicted in (b), and lies on the XY plane. There are $G0$ geometric continuities between the connection points of these curves. $G1$ continuities exist at the connection points between the curves *C2* and *C3*, *C3* and *C4*, and *C5* and *C6*. These six curves are rotated around the Y axis by 2π, and the surface model of the ewer main body in Fig. 13 (b) is obtained. Each curve is represented with four control points, and there are 19 $(24 - 5)$ control points for the ewer main body representation. Note that five control points reside between the connections of

Table 10
(a) Design satisfaction outcomes (S. Outcome, 1: Poor, 2: Fair, 3: Good, 4: Very good, 5: Excellent. μ: Average score and σ: Standard deviation) and design development/selection time (D. Time/ S. Time) in minutes (b) Survey responses collected from the participants (1: Strongly Disagree, 2: Disagree, 3: Neutral, 4: Agree, 5: Strongly Agree)

(a)

User	Wheel Rim				Car Body			
	Task 1		Task 2		Task 1		Task 2	
	S. Outcome	D. Time	S. Outcome	S. Time	S. Outcome	D. Time	S. Outcome	S. Time
1	4.4	8.2	5	1.3	4.6	14.7	3.8	4.0
2	3.6	7.9	4	2.2	3.2	15.4	3.2	3.5
3	4	10.4	3.8	2.2	3.8	17.2	4.0	2.5
4	3.4	8.2	3.9	2.1	4	11.3	3.6	1.4
5	4.2	8.3	4.7	3.6	3.9	12	4.2	3.2
6	4	12	4.8	4.2	3.8	14.8	4.0	2.3
7	3.4	7.5	3.8	2.3	3	24.7	3.8	5.0
8	4.6	15.2	4.8	4.3	4.4	20.6	4.4	5.0
9	4.2	19.6	3.8	5.6	4.2	18.2	3.8	4.4
10	3.6	17.6	4	7.4	3.2	15.8	3.8	5.0
μ	3.9	11.4	4.2	3.4	3.8	16.3	3.8	3.6
σ	0.4	4.3	0.45	1.8	0.5	4.0	0.3	1.25

(b)

User	The proposed sampling technique can be useful in product design	Overall, I am satisfied with the models generated by the sampling technique	The models generated by the sampling technique give a better understanding of design alternatives
1	5	5	5
2	5	5	5
3	5	4	4
4	3	2	5
5	4	4	4
6	5	5	5
7	5	4	4
8	5	4	5
9	4	4	5
10	5	4	5
μ	4.6	4.1	4.7
σ	0.66	0.83	0.46

neighboring curves. The control points start from the top portion of the curve $C1$ and end at the bottom portion of the curve $C6$. Each control point is represented with two coordinates, one for the X and one for the Y coordinate. Therefore, 38 geometric parameters are obtained for the ewer main body model. In addition, $\alpha_0, \alpha_1, ..., \alpha_{37}$ denote the geometric parameters. For example, α_0 and α_1 represent the X and Y coordinates, respectively, of the first control point for the curve $C1$. Moreover, α_2 and α_3 represent the X and Y coordinates, respectively, of the second control point for the curve $C1$. Other geometric parameters are ordered in a similar manner (see Fig. 13 (c)).

Table 13 shows that 30 geometric constraints are defined for the ewer main body model. The constraints from ϕ_0 to ϕ_{26} are linear inequality constraints. The nonlinear equality constraints ϕ_{27}, ϕ_{28}, and ϕ_{29} provide $G1$ continuities at the connection points between the curves *C2* and *C3*, *C3* and *C4*, and *C5* and *C6*, respectively. In addition, v_a denotes a $2D$ vector from the point $(\alpha_{12}, \alpha_{13})$ to the point $(\alpha_{10}, \alpha_{11})$, and v_b is the $2D$ vector from the point $(\alpha_{14}, \alpha_{15})$ to the point $(\alpha_{12}, \alpha_{13})$. Furthermore, $\Lambda(v_a, v_b)$ denotes the angle between v_a and v_b in radians; v_c and v_d are the $2D$

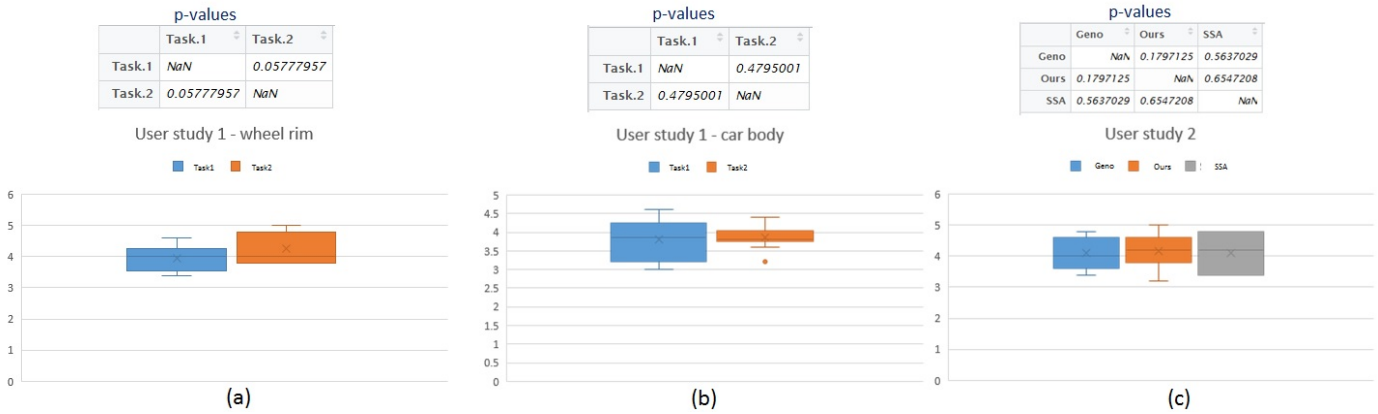

Fig. 11. p-values obtained from Friedman tests in pair-wise analysis, and Box and Whisker plots for the user study results.

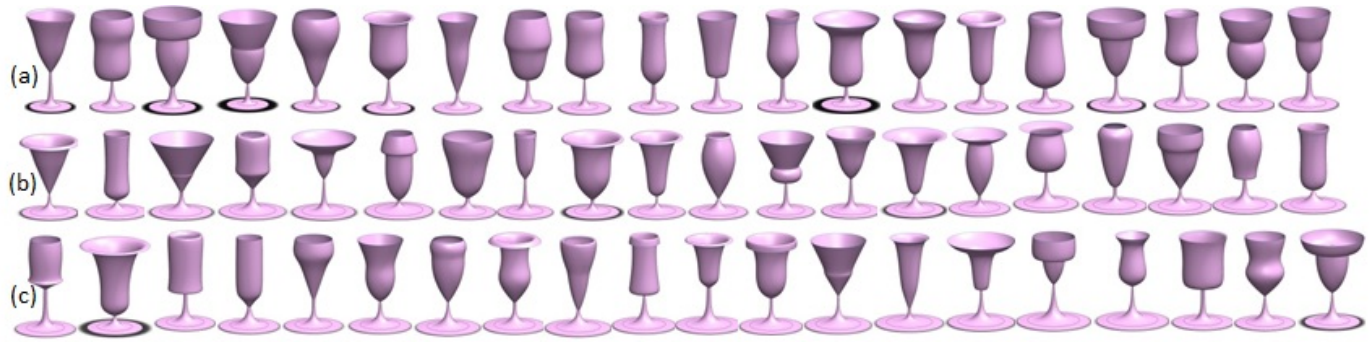

Fig. 12. Wine glass models generated using (a) Genoform of Krish [10] (a), (b) the proposed technique, (c) the SSA approach

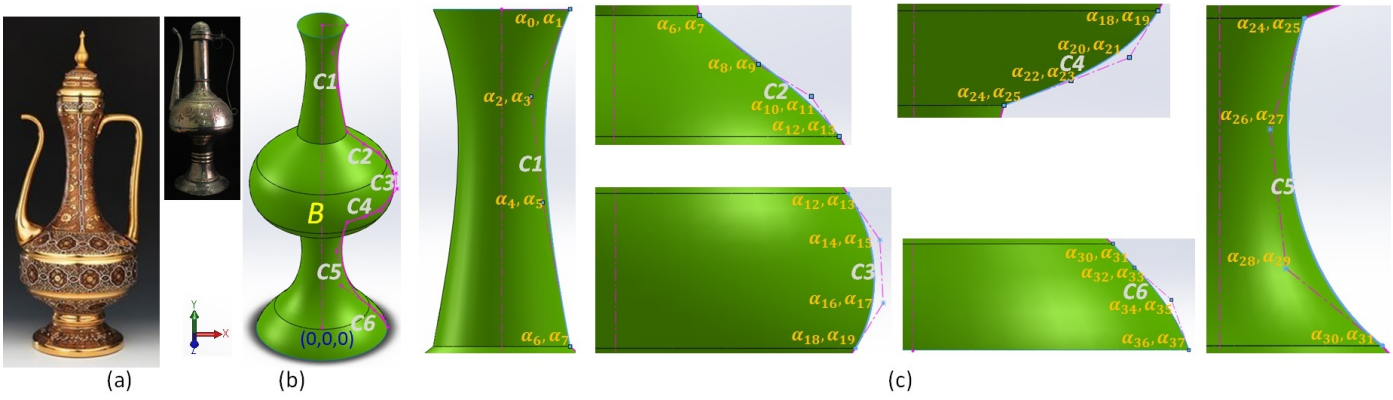

Fig. 13. (a) Two ewer main body models with different shapes (left from [42]. (b, c) Shape of a ewer main body represented with six curves (*C1, C2, ..., C6*)

vectors from the point $(\alpha_{18}, \alpha_{19})$ to the point $(\alpha_{16}, \alpha_{17})$ and from the point $(\alpha_{20}, \alpha_{21})$ to the point $(\alpha_{18}, \alpha_{19})$, respectively. Similarly, v_e and v_f are the $2D$ vectors from the point $(\alpha_{30}, \alpha_{31})$ to the point $(\alpha_{28}, \alpha_{29})$ and from the point $(\alpha_{32}, \alpha_{33})$ to the point $(\alpha_{30}, \alpha_{31})$, respectively. The value ranges for the geometric parameters for the ewer main body are displayed in Table 14.

Car body. Figure 14 (a) shows a sedan car side silhouette that is represented using eight Bezier curves and its lower portion. The lower portion is automatically adjusted based on the Bezier curves. Each curve has its own coordinate system, with the origin shown in (b), and the car side

Geometric Constraints		
$\phi_0 : f(\phi_0) = \alpha_1 - \alpha_3 \geq 0$	$\phi_1 : f(\phi_1) = \alpha_3 - \alpha_5 \geq 0$	$\phi_2 : f(\phi_2) = \alpha_5 - \alpha_7 \geq 0$
$\phi_3 : f(\phi_3) = \alpha_7 - \alpha_9 \geq 0$	$\phi_4 : f(\phi_4) = \alpha_9 - \alpha_{11} \geq 0$	$\phi_5 : f(\phi_5) = \alpha_{11} - \alpha_{13} \geq 0$
$\phi_6 : f(\phi_6) = \alpha_{13} - \alpha_{15} \geq 0$	$\phi_7 : f(\phi_7) = \alpha_{15} - \alpha_{17} \geq 0$	$\phi_8 : f(\phi_8) = \alpha_{17} - \alpha_{19} \geq 0$
$\phi_9 : f(\phi_9) = \alpha_{19} - \alpha_{21} \geq 0$	$\phi_{10} : f(\phi_{10}) = \alpha_{21} - \alpha_{23} \geq 0$	$\phi_{11} : f(\phi_{11}) = \alpha_{23} - \alpha_{25} \geq 0$
$\phi_{12} : f(\phi_{12}) = \alpha_{25} - \alpha_{27} \geq 0$	$\phi_{13} : f(\phi_{13}) = \alpha_{27} - \alpha_{29} \geq 0$	$\phi_{14} : f(\phi_{14}) = \alpha_{29} - \alpha_{31} \geq 0$
$\phi_{15} : f(\phi_{15}) = \alpha_{31} - \alpha_{33} \geq 0$	$\phi_{16} : f(\phi_{16}) = \alpha_{33} - \alpha_{35} \geq 0$	$\phi_{17} : f(\phi_{17}) = \alpha_{35} - \alpha_{37} \geq 0$
$\phi_{18} : f(\phi_{18}) = \alpha_8 - \alpha_6 \geq 0$	$\phi_{19} : f(\phi_{19}) = \alpha_{10} - \alpha_8 \geq 0$	$\phi_{20} : f(\phi_{20}) = \alpha_{12} - \alpha_{10} \geq 0$
$\phi_{21} : f(\phi_{21}) = \alpha_{20} - \alpha_{18} \leq 0$	$\phi_{22} : f(\phi_{22}) = \alpha_{22} - \alpha_{20} \leq 0$	$\phi_{23} : f(\phi_{23}) = \alpha_{24} - \alpha_{22} \leq 0$
$\phi_{24} : f(\phi_{24}) = \alpha_{32} - \alpha_{30} \geq 0$	$\phi_{25} : f(\phi_{25}) = \alpha_{34} - \alpha_{32} \geq 0$	$\phi_{26} : f(\phi_{26}) = \alpha_{36} - \alpha_{34} \geq 0$
$\phi_{27} : f(\phi_{27}) = \Lambda(v_a, v_b) = 0$	$\phi_{28} : f(\phi_{28}) = \Lambda(v_c, v_d) = 0$	$\phi_{29} : f(\phi_{29}) = \Lambda(v_e, v_f) = 0$

Table 13
Geometric constraints for the ewer main body model

Parameter ranges			
$1.0 \leq \alpha_0 \leq 5.0$	$66.0 \leq \alpha_1 \leq 72.0$	$1.0 \leq \alpha_2 \leq 5.0$	$58.0 \leq \alpha_3 \leq 68.0$
$1.0 \leq \alpha_4 \leq 5.0$	$50.0 \leq \alpha_5 \leq 62.0$	$3.0 \leq \alpha_6 \leq 5.0$	$43.0 \leq \alpha_7 \leq 52.0$
$3.0 \leq \alpha_8 \leq 8.0$	$41.0 \leq \alpha_9 \leq 47.0$	$6.0 \leq \alpha_{10} \leq 10.0$	$39.0 \leq \alpha_{11} \leq 42.0$
$8.0 \leq \alpha_{12} \leq 13.0$	$37.0 \leq \alpha_{13} \leq 40.0$	$10.0 \leq \alpha_{14} \leq 15.0$	$33.0 \leq \alpha_{15} \leq 38.0$
$10.0 \leq \alpha_{16} \leq 15.0$	$31.0 \leq \alpha_{17} \leq 35.0$	$8.0 \leq \alpha_{18} \leq 13.0$	$29.0 \leq \alpha_{19} \leq 32.0$
$10.0 \leq \alpha_{20} \leq 13.0$	$27.0 \leq \alpha_{21} \leq 30.0$	$5.0 \leq \alpha_{22} \leq 11.0$	$25.0 \leq \alpha_{23} \leq 28.0$
$3.0 \leq \alpha_{24} \leq 6.0$	$22.0 \leq \alpha_{25} \leq 26.0$	$3.0 \leq \alpha_{26} \leq 5.0$	$15.0 \leq \alpha_{27} \leq 24.0$
$3.0 \leq \alpha_{28} \leq 5.0$	$8.0 \leq \alpha_{29} \leq 18.0$	$6.0 \leq \alpha_{30} \leq 10.0$	$4.5 \leq \alpha_{31} \leq 11.0$
$9.0 \leq \alpha_{32} \leq 11.0$	$3.0 \leq \alpha_{33} \leq 6.0$	$10.0 \leq \alpha_{34} \leq 13.0$	$1.0 \leq \alpha_{35} \leq 3.8$
$12.0 \leq \alpha_{36} \leq 15.0$	$0.0 \leq \alpha_{37} \leq 2.2$		

Table 14
Geometric parameter ranges for the ewer main body model

silhouette is obtained by combining these curves. Quadratic or cubic curves are utilized, and the geometric parameters are the X and Y coordinates of their control points. A car side silhouette is defined by the 38 geometric parameters $\alpha_0, \alpha_1, ..., \alpha_{37}$ (see Fig. 14 (b)). Figure 14 (a) shows a simplified CAD model (in brown) for a car body, which is generated after sweeping the side silhouette in the Z direction and performing cut sweep using the blue sketch in the front profile.

Twenty-six inequality geometric constraints are defined for the car side silhouette shown in Table 15. Nine of them are nonlinear and others are linear inequality constraints. The geometric constraints for the car hood line (the $C1$ curve) are introduced here. In this case, ϕ_2 and ϕ_3 denote the linear inequality constraints for the hood line. Without these constraints, the curve formed can have a sharp bend as seen in Figure 15 (a). Such hood line shapes are unusual enough to be considered infeasible. The constraints ϕ_4 and ϕ_5 enable the generation of convex hood line shapes. Concave hood lines are abnormal for a car side silhouette, as shown in Fig. 15 (b). All other geometric constraints are determined in a similar way. Table 16 shows the value ranges of the geometric parameters for the car side silhouette.

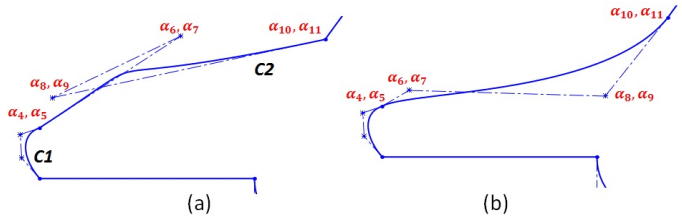

Fig. 15. Unusual hood line shapes can appear if the geometric constraints ϕ_2, ϕ_3 (a) and ϕ_4, ϕ_5 (b) are not included in the PT algorithm.

Car hood. A car hood is represented using a B-spline surface with 40 control points, as seen in Fig. 16, which is mirrored across the XZ plane. The control points P_1-P_{21} are specified in advance (i.e. with fixed positions), and they can be considered design specifications of a specific car. The control points P_{22}-P_{29} are stationary points in the X and Y axes, and incorporating them avoids unusual hood

Geometric Constraints	
$\phi_0 : f(\phi_0) = \alpha_3 - \alpha_1 \geq 0$	$\phi_1 : f(\phi_1) = \alpha_5 - \alpha_3 \geq 0$
$\phi_2 : f(\phi_2) = \alpha_8 - \alpha_6 \geq 0$	$\phi_3 : f(\phi_3) = \alpha_9 - \alpha_7 \geq 0$
$\phi_4 : f(\phi_4) = \alpha_7 - (\alpha_6 * \alpha_{11}/\alpha_{10}) \geq 0$	$\phi_5 : f(\phi_5) = \alpha_9 - (\alpha_8 * \alpha_{11}/\alpha_{10}) \geq 0$
$\phi_6 : f(\phi_6) = \alpha_{13} - (\alpha_{12} * \alpha_{15}/\alpha_{14}) \geq 0$	$\phi_7 : f(\phi_7) = \alpha_{18} - \alpha_{16} - (\alpha_{20}/4) \geq 0$
$\phi_8 : f(\phi_8) = \alpha_{20} - \alpha_{18} \geq 0$	$\phi_9 : f(\phi_9) = \alpha_{24} - \alpha_{22} \geq 0$
$\phi_{10} : f(\phi_{10}) = \alpha_{23} - (\alpha_{22} * \alpha_{25}/\alpha_{24}) \geq 0$	$\phi_{11} : f(\phi_{11}) = \alpha_{27} - (\alpha_{26} * \alpha_{29}/\alpha_{28}) \geq 0$
$\phi_{12} : f(\phi_{12}) = \alpha_{31} - (\alpha_{30} * \alpha_{33}/\alpha_{32}) \geq 0$	$\phi_{13} : f(\phi_{13}) = \alpha_{35} - (\alpha_{34} * \alpha_{37}/\alpha_{36}) \geq 0$
$\phi_{14} : f(\phi_{14}) = \alpha_{11} - \alpha_9 \geq 0$	$\phi_{15} : f(\phi_{15}) = \alpha_{15} - \alpha_{13} \geq 0$
$\phi_{16} : f(\phi_{16}) = \alpha_{17} - \alpha_{21} \geq 0$	$\phi_{17} : f(\phi_{17}) = \alpha_{19} - \alpha_{21} \geq 0$
$\phi_{18} : f(\phi_{18}) = \alpha_{28} - \alpha_{26} \geq 0$	$\phi_{19} : f(\phi_{19}) = \alpha_{36} - \alpha_{34} \geq 0$
$\phi_{20} : f(\phi_{20}) = \alpha_{32} - \alpha_{30} \geq 0$	$\phi_{21} : f(\phi_{21}) = \alpha_{24} - \alpha_{22} \geq 0$
$\phi_{22} : f(\phi_{22}) = \alpha_{13} - (2 * \alpha_{12} * \alpha_{15}/\alpha_{14}) \leq 0$	$\phi_{23} : f(\phi_{23}) = \alpha_{16} - (\alpha_{20}/4) \geq 0$
$\phi_{24} : f(\phi_{24}) = \alpha_{18} - (3 * \alpha_{20}/4) \leq 0$	$\phi_{25} : f(\phi_{25}) = \alpha_{23} - (2 * \alpha_{22} * \alpha_{25}/\alpha_{24}) - (\alpha_{24}/2) \leq 0$

Table 15
Geometric constraints for the car body model

Parameter ranges			
$-10.0 \leq \alpha_0 \leq 0.0$	$0.0 \leq \alpha_1 \leq 20.0$	$-10.0 \leq \alpha_2 \leq 0.0$	$0.0 \leq \alpha_3 \leq 20.0$
$-10.0 \leq \alpha_4 \leq 0.0$	$10.0 \leq \alpha_5 \leq 20.0$	$0.0 \leq \alpha_6 \leq 80.0$	$0.0 \leq \alpha_7 \leq 20.0$
$0.0 \leq \alpha_8 \leq 80.0$	$0.0 \leq \alpha_9 \leq 20.0$	$40.0 \leq \alpha_{10} \leq 80.0$	$10.0 \leq \alpha_{11} \leq 20.0$
$0.0 \leq \alpha_{12} \leq 20.0$	$0.0 \leq \alpha_{13} \leq 25.0$	$20.0 \leq \alpha_{14} \leq 25.0$	$15.0 \leq \alpha_{15} \leq 25.0$
$10.0 \leq \alpha_{16} \leq 110.0$	$-5.0 \leq \alpha_{17} \leq 20.0$	$10.0 \leq \alpha_{18} \leq 110.0$	$-5.0 \leq \alpha_{19} \leq 20.0$
$80.0 \leq \alpha_{20} \leq 120.0$	$0.0 \leq \alpha_{21} \leq 5.0$	$0.0 \leq \alpha_{22} \leq 30.0$	$-25.0 \leq \alpha_{23} \leq 0.0$
$15.0 \leq \alpha_{24} \leq 30.0$	$-25.0 \leq \alpha_{25} \leq -13.0$	$0.0 \leq \alpha_{26} \leq 20.0$	$-4.0 \leq \alpha_{27} \leq 0.0$
$10.0 \leq \alpha_{28} \leq 20.0$	$-4.0 \leq \alpha_{29} \leq -2.0$	$0.0 \leq \alpha_{30} \leq 10.0$	$-20.0 \leq \alpha_{31} \leq 0.0$
$5.0 \leq \alpha_{32} \leq 10.0$	$-20.0 \leq \alpha_{33} \leq -10.0$	$0.0 \leq \alpha_{34} \leq 6.0$	$-10.0 \leq \alpha_{35} \leq 0.0$
$3.0 \leq \alpha_{36} \leq 6.0$	$-10.0 \leq \alpha_{37} \leq -5.0$		

Table 16
Geometric parameter ranges for the car body model

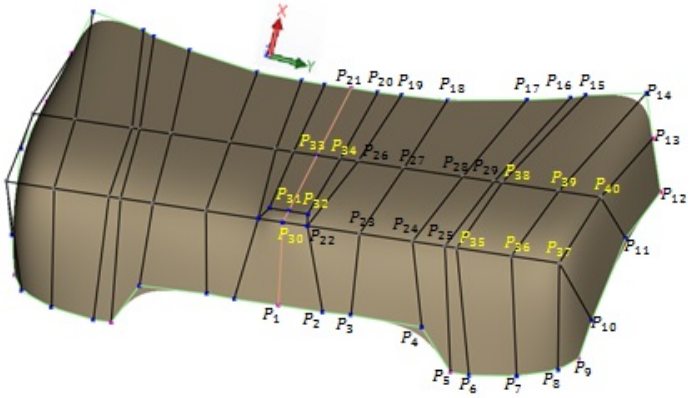

Fig. 16. A car hood model represented using a B-spline surface with 40 control points.

shapes and ensures hood flatness. The other control points, P_{30}-P_{40}, are mobile points, some of which move together in one or more coordinate axes, such as P_{31}-P_{32} and P_{33}-P_{34}, to generate plausible hood shapes. The geometric parameters we define for the car hood are some control point coordinates of P_{30}-P_{40}: $x_0, \ldots, x_{10}$. Table 17 lists all control point positions and the geometric parameters. Table 18 and 19 depict geometric constraints and parameter ranges, respectively.

Yacht hull. A yacht hull model generated using Khan et al.'s design framework [43] is used, as shown in Fig. 17. Eleven geometric parameters are defined on the hull sur-

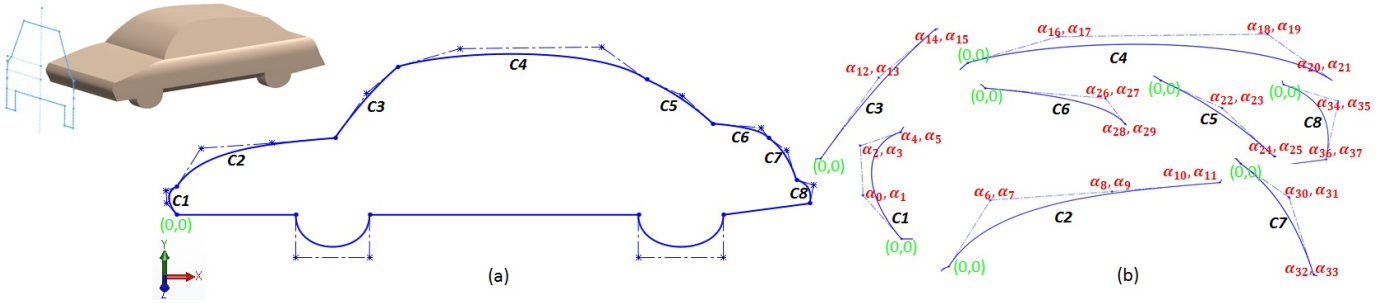

Fig. 14. (a) A car side silhouette is represented using eight curves (*C1, C2, ..., C8*) and its lower portion. A simplified CAD model (in brown) for the car body is generated after sweeping the side silhouette in the Z direction and performing cut sweep using the blue sketch in the front profile. (b) Geometric parameters (in red) are defined as the control points of the curves.

Control Points		
$P_1 = (19.0, -2.2, -127.8)$	$P_2 = (19.0, 20.5, -127.7)$	$P_3 = (19.8, 34.9, -127.8)$
$P_4 = (22.7, 70.2, -124.7)$	$P_5 = (21.4, 84.8, -101.3)$	$P_6 = (21.9, 94.0, -100.8)$
$P_7 = (27.0, 117.0, -100.8)$	$P_8 = (36.7, 135.3, -101.0)$	$P_9 = (49.3, 142.4, -101.0)$
$P_{10} = (59.3, 146.5, -117.6)$	$P_{11} = (98.6, 152.9, -141.4)$	$P_{12} = (152.1, 155.0, -133.8)$
$P_{13} = (168.4, 146.6, -153.5)$	$P_{14} = (182.5, 139.7, -170)$	$P_{15} = (173.3, 111.0, -170.0)$
$P_{16} = (169.1, 104.5, -170.0)$	$P_{17} = (162.2, 84.3, -170.0)$	$P_{18} = (152.1, 46.6, -170.0)$
$P_{19} = (150.6, 23.4, -170.0)$	$P_{20} = (150.2, 11.4, -170.0)$	$P_{21} = (150.4, -2.2, -170.0)$
$P_{22} = (30.0, 10.4, x_9)$	$P_{23} = (30.0, 37.2, x_9)$	$P_{24} = (30.0, 64.4, x_9)$
$P_{25} = (30.0, 80.9, x_9)$	$P_{26} = (90.0, 18.9, x_{10})$	$P_{27} = (90.0, 42.4, x_{10})$
$P_{28} = (90.0, 72.5, x_{10})$	$P_{29} = (90.0, 88.7, x_{10})$	$P_{30} = (x_0, -2.2, -170.0)$
$P_{31} = (x_1, -2.2, x_2)$	$P_{32} = (x_1, 7.7, x_2)$	$P_{33} = (x_3, -2.2, x_4)$
$P_{34} = (x_3, 9.8, x_4)$	$P_{35} = (x_5, 87.0, x_6)$	$P_{36} = (x_5, 115.1, x_6)$
$P_{37} = (x_5, 139.5, x_8)$	$P_{38} = (x_7, 92.7, x_6)$	$P_{39} = (x_7, 121.8, x_6)$
$P_{40} = (x_7, 143.2, x_8)$		

Table 17
Control points for the car hood model

Geometric Constraints	
$\phi_0 : f(\phi_0) = \alpha_3 - \alpha_1 - 20.0 \geq 0$	$\phi_1 : f(\phi_1) = \alpha_1 - \alpha_0 - 20.0 \geq 0$
$\phi_2 : f(\phi_2) = \alpha_7 - \alpha_5 - 20.0 \geq 0$	

Table 18
Geometric constraints for the car hood model

Parameter ranges			
$0.0 \leq \alpha_0 \leq 30.0$	$30.0 \leq \alpha_1 \leq 90.0$	$-200.0 \leq \alpha_2 \leq -170.0$	$50.0 \leq \alpha_3 \leq 150.0$
$-200.0 \leq \alpha_4 \leq -170.0$	$40.0 \leq \alpha_5 \leq 100.0$	$-240.0 \leq \alpha_6 \leq -170.0$	$50.0 \leq \alpha_7 \leq 150.0$
$-220.0 \leq \alpha_8 \leq -170.0$	$-170.0 \leq \alpha_9 \leq -130.0$	$-170.0 \leq \alpha_{10} \leq -150.0$	

Table 19
Geometric parameter ranges for the car hood model

face. The terms α_0, α_1, and α_2 represent the entrance, middle, and run region lengths, respectively. Moreover, α_3, α_4, and α_5 are beams of the sections, and α_6, α_7, and α_8 denote the depths of the sections. Finay, α_9 and α_{10} represent the entrance and bow angles (in degrees), respectively. Tables 20 and 21 show geometric constraints and parameter ranges, respectively.

Wheel rim. The front view of a wheel rim model is generated by first mirroring a Bezier curve with seven control points and then rotating the curves around the center point. Figure 18 shows ten geometric parameters. Here, $\alpha_0, \ldots, \alpha_8$ represent some of the control point coordinates for the curve. Moreover, α_9 denotes the angle (in degrees)

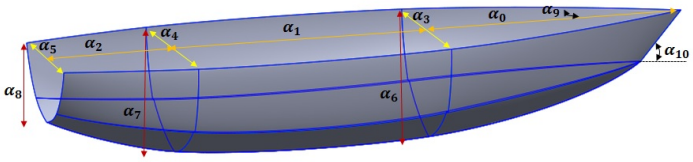

Fig. 17. A yacht hull model with 11 geometric parameters

Geometric Constraints	
$\phi_0 : f(\phi_0) = \alpha_6 - \alpha_7 \geq 0$	$\phi_1 : f(\phi_1) = \alpha_7 - \alpha_8 \geq 0$
$\phi_2 : f(\phi_2) = \alpha_3 - \alpha_4 \geq 0$	$\phi_3 : f(\phi_3) = \alpha_4 - \alpha_5 \geq 0$
$\phi_4 : f(\phi_4) = 5 - \frac{\alpha_0+\alpha_1+\alpha_2}{\alpha_4} \geq 0$	$\phi_5 : f(\phi_5) = \frac{\alpha_0+\alpha_1+\alpha_2}{\alpha_4} - 4.5 \geq 0$

Table 20
Geometric constraints for the yacht hull model

Parameter ranges			
$6.0 \leq \alpha_0 \leq 12.0$	$6.0 \leq \alpha_1 \leq 12.0$	$4.0 \leq \alpha_2 \leq 8.0$	$4.8 \leq \alpha_3 \leq 10.8$
$4.6 \leq \alpha_4 \leq 10.6$	$3.0 \leq \alpha_5 \leq 9.0$	$2.3 \leq \alpha_6 \leq 4.3$	$2.2 \leq \alpha_7 \leq 4.21$
$1.75 \leq \alpha_8 \leq 3.75$	$25.0 \leq \alpha_9 \leq 75.0$	$30.0 \leq \alpha_{10} \leq 120.0$	

Table 21
Geometric parameter ranges for the yacht hull model

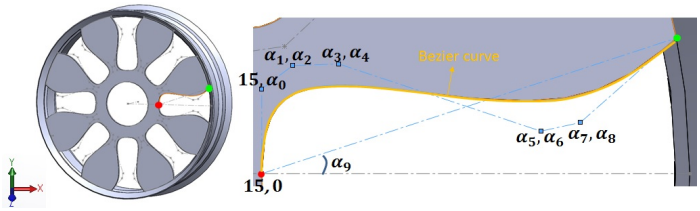

Fig. 18. A wheel rim model represented using ten geometric parameters

Parameter ranges			
$0.0 \leq \alpha_0 \leq 9.0$	$15.0 \leq \alpha_1 \leq 20.0$	$0.0 \leq \alpha_2 \leq 9.0$	$15.0 \leq \alpha_3 \leq 30.0$
$0.0 \leq \alpha_4 \leq 10.0$	$25.0 \leq \alpha_5 \leq 35.0$	$0.0 \leq \alpha_6 \leq 11.0$	$30.0 \leq \alpha_7 \leq 39.0$
$1.0 \leq \alpha_8 \leq 11.0$	$0.0 \leq \alpha_9 \leq 22.0$		

Table 22
Geometric parameter ranges for the wheel rim model

between the horizontal axis and the line connecting the start and end points of the Bezier curve. No geometric constraints are considered for this test model. Table 22 shows the parameter ranges for the rim model.

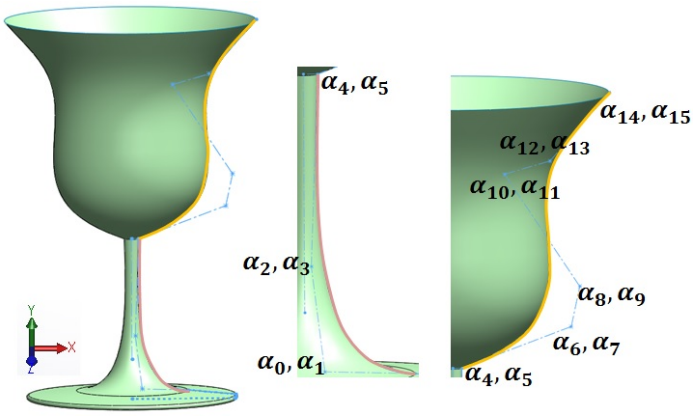

Fig. 19. A wine glass model represented using 16 geometric parameters

Geometric Constraints	
$\phi_0 : f(\phi_0) = \alpha_1 + \alpha_0/11.0 - 25.0/22.0 = 0$	$\phi_1 : f(\phi_1) = \alpha_5 - \alpha_3 \geq 0$
$\phi_2 : f(\phi_2) = \alpha_{15} - \alpha_{13} \geq 0$	$\phi_3 : f(\phi_3) = \alpha_7 - \alpha_5 \geq 0$
$\phi_4 : f(\phi_4) = \alpha_{14} - \alpha_{11} \geq 0$	

Table 23
Geometric constraints for the wine glass model

Parameter ranges			
$0.0 \leq \alpha_0 \leq 1.5$	$1.0 \leq \alpha_1 \leq 1.5$	$0.0 \leq \alpha_2 \leq 1.0$	$1.0 \leq \alpha_3 \leq 15.0$
$0.5 \leq \alpha_4 \leq 1.0$	$15.0 \leq \alpha_5 \leq 25.0$	$0.5 \leq \alpha_6 \leq 15.0$	$25.0 \leq \alpha_7 \leq 35.0$
$0.5 \leq \alpha_8 \leq 15.0$	$30.0 \leq \alpha_9 \leq 40.0$	$0.5 \leq \alpha_{10} \leq 15.0$	$35.0 \leq \alpha_{11} \leq 45.0$
$0.5 \leq \alpha_{12} \leq 15.0$	$40.0 \leq \alpha_{13} \leq 50.0$	$5.0 \leq \alpha_{14} \leq 15.0$	$45.0 \leq \alpha_{15} \leq 55.0$

Table 24
Geometric parameter ranges for the wine glass model

Geometric Constraints	
$\phi_0 : f(\phi_0) = \alpha_1 - \alpha_0 \geq 0$	$\phi_1 : f(\phi_1) = \alpha_{14} - \alpha_{13} \geq 0$
$\phi_2 : f(\phi_2) = \alpha_{15} - \alpha_{14} \geq 0$	

Table 25
Geometric constraints for the bottle model

Parameter ranges			
$20.0 \leq \alpha_0 \leq 40.0$	$20.0 \leq \alpha_1 \leq 80.0$	$10.0 \leq \alpha_2 \leq 160.0$	$0.0 \leq \alpha_3 \leq 100.0$
$10.0 \leq \alpha_4 \leq 160.0$	$40.0 \leq \alpha_5 \leq 140.0$	$10.0 \leq \alpha_6 \leq 160.0$	$80.0 \leq \alpha_7 \leq 180.0$
$10.0 \leq \alpha_8 \leq 160.0$	$120.0 \leq \alpha_9 \leq 220.0$	$10.0 \leq \alpha_{10} \leq 160.0$	$160.0 \leq \alpha_{11} \leq 260.0$
$14.0 \leq \alpha_{12} \leq 18.0$	$200.0 \leq \alpha_{13} \leq 300.0$	$280.0 \leq \alpha_{14} \leq 340.0$	$320.0 \leq \alpha_{15} \leq 380.0$

Table 26
Geometric parameter ranges for the bottle model

Wine glass. A wine glass model is represented using two Bezier curves (see Fig. 19), one for the upper portion and other for the holder. $\alpha_0, \alpha_1, \ldots, \alpha_{15}$ denote the X and Y coordinates of the control points. Tables 23 and 24 show the geometric constraints and the parameter ranges, respectively, for the wine glass model.

Bottle. After setting values for the 16 geometric parameters, a bottle model can be generated (see Fig. 20). Tables 25 and 26 show the geometric constraints and parameter ranges, respectively, for the bottle model.

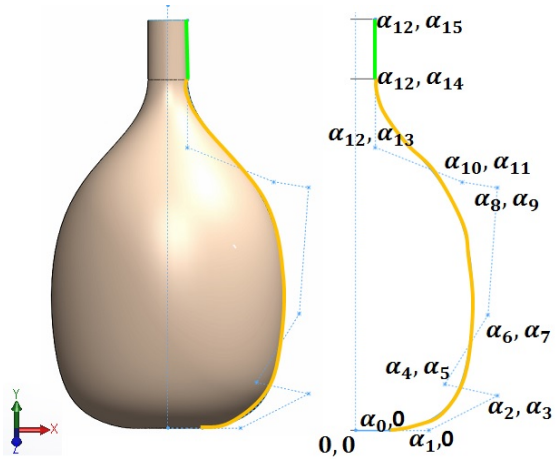

Fig. 20. A bottle model represented using 16 geometric parameters

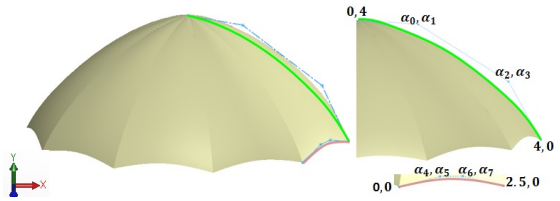

Fig. 21. A park shed model represented using eight geometric parameters

Parameter ranges			
$0.0 \leq \alpha_0 \leq 4.0$	$0.0 \leq \alpha_1 \leq 4.0$	$0.0 \leq \alpha_2 \leq 4.0$	$0.0 \leq \alpha_3 \leq 4.0$
$0.0 \leq \alpha_4 \leq 2.5$	$0.0 \leq \alpha_5 \leq 3.0$	$0.0 \leq \alpha_6 \leq 2.5$	$0.0 \leq \alpha_7 \leq 3.0$

Table 27
Geometric parameter ranges for the park shed model

Park shed. Eight geometric parameters are defined for a park shed model, as shown in Fig. 21. No geometric constraints are considered for this model. Table 27 shows the parameter ranges for the park shed model.

References

[1] D. Eppstein, M. T. Goodrich, E. Kim, R. Tamstorf, Motorcycle graphs: canonical quad mesh partitioning, Eurographics Symposium on Geometry Processing 27 (5) (2008) 1477–1486.

[2] P. Audze, V. Eglais, New approach for planning out of experiments, Problems Dynamics and Strengths 35 (1977) 104107.

[3] 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) 680689.

[4] G. Yu, X. Wang, P. Li, A constraint based evolutionary decision support system for product design, Chinese control and decision conference. IEEE industrial electronics (IE) (2009) 2585–2590.

[5] L. Caladas, Gene_arch: an evolution-based generative design system for sustainable architecture, in: Lecture Notes in Computer Science. 4200:109, 2006.

[6] P. J. Bentley, An introduction to evolutionary design by computers., In: Bentley PJ, editor. Evolutionary design by computers. San Francisco (CA): Morgan Kaufmann (1999) 1–73.

[7] P. Deak, C. Reed, G. Rowe, Cad grammars: extending shape and graph grammars for spatial design modeling, In: Computer

Aided Methods in Optimal Design and Operations (2006) 119–128.

[8] G. S. Hornby, J. B. Pollack, The advantages of generative grammatical encodings for physical design, in: In: Congress on evolutionary computation. Seoul: IEEE, 2001.

[9] Z. Gu, M. X. Tang, J. H. Frazer, Capturing aesthetic intention during interactive evolution, Computer-Aided Design 38 (3) (2006) 224–237.

[10] S. Krish, A practical generative design method, Computer-Aided Design 43 (2011) 88–100.

[11] M. Ovsjanikov, W. Li, L. Guibas, N. J. Mitra, Exploration of continuous variability in collections of 3d shapes, ACM Trans. Graph. 30 (4) (2011) 33:1–33:10.

[12] E. Kalogerakis, S. Chaudhuri, D. Koller, V. Koltun, A probabilistic model for component-based shape synthesis, ACM Trans. Graph. 55 (2012) 1–11.

[13] S. Chaudhuri, E. Kalogerakis, S. Giguere, T. Funkhouser, Attribit: Content creation with semantic attributes, ACM Symposium on User Interface Software and Technology (UIST) (2013) 193–202.

[14] Y. Kleiman, N. Fish, J. Lanir, D. Cohen-Or, Dynamic maps for exploring and browsing shapes, Proceedings of the Eleventh Eurographics/ACMSIGGRAPH Symposium on Geometry Processing (2013) 187–196.

[15] N. Fish, M. Averkiou, O. V. Kaick, O. Sorkine-Hornung, D. Cohen-Or, N. J. Mitra, Meta-representation of shape families, ACM Trans. Graph. 33 (4) (2014) 34:1–34:11.

[16] Q. Huang, F. Wang, L. Guibas, Functional map networks for analyzing and exploring large shape collections, ACM Trans. Graph. 33 (4) (2014) 36:1–36:11.

[17] M. Averkiou, V. G. Kim, Y. Zheng, N. J. Mitra, Shapesynth: Parameterizing model collections for coupled shape exploration and synthesis, Comput. Graph. Forum 33 (2) (2014) 125–134.

[18] M. E. Yumer, P. Asente, R. Mech, L. B. Kara, Procedural modeling using autoencoder networks, Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology (2015) 109–118.

[19] L. Gao, Y. P. Cao, Y. K. Lai, H. Z. Huang, L. Kobbelt, S. M. Hu, Active exploration of large 3d model repositories, IEEE Transactions on Visualization and Computer Graphics 21 (12) (2015) 1390–1402.

[20] A. Schulz, J. Xu, B. Zhu, C. Zheng, E. Grinspun, W. Wojciech, Interactive design space exploration and optimization for cad models, ACM Transactions on Graphics 36 (4).

[21] Y. L. Yang, Y. J. Yang, H. Pottmann, N. J. Mitra, Shape space exploration of constrained meshes, ACM Transactions on Graphics 30 (6) (2011) 124:1–124:12.

[22] X. Zhao, C. C. Tang, Y. L. Yang, H. Pottmann, N. L. Mitra, Intuitive design exploration of constrained meshes, Advances in Architectural Geometry 2012 (2013) 305–318.

[23] M. D. Morris, T. J. Mitchell, Exploratory designs for computer experiments, Journal of Statistical Planning and Inference 45 (6) (1995) 381–402.

[24] R. L. Smith, Efficient monte carlo procedures for generating points uniformly distributed over bounded regions, Operations Research 32 (6) (1984) 1296–1308.

[25] C. G. E. Boender, R. J. Caron, J. F. McDonald, A. H. G. R. Kan, H. E. Romeijn, R. L. Smith, J. Telgen, A. C. F. Vorst, Shake-and-bake algorithms for generating uniform points on the boundary of bounded polyhedra, Operations Research 39 (6) (1991) 945–954.

[26] E. Gunpinar, H. Suzuki, Y. Ohtake, M. Moriguchi, Generation of bi-monotone patches from quadrilateral mesh for reverse engineering, Computer-Aided Design 45 (2) (2013) 440–450.

[27] E. Gunpinar, M. Moriguchi, H. Suzuki, Y. Ohtake, Feature-aware partitions from motorcycle graph, Computer-Aided Design 47 (2014) 85–95.

[28] E. Gunpinar, M. Moriguchi, H. Suzuki, Y. Ohtake, Motorcycle graph enumeration from quadrilateral meshes for reverse engineering, Computer-Aided Design 55 (2014) 64–80.

[29] M. D. Shields, J. Zhang, The generalization of latin hypercube sampling, Reliability Engineering and System Safety 148 (2016) 96–108.

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

[31] R. Haupt, S. Haupt, Practical Genetic Algorithms, John Wiley and Sons Inc, 2004.

[32] B. W. Kort, D. P. Bertsekas, A new penalty function method for constrained minimization, in: Proceedings of the 1972 IEEE Conference on Decision and Control and 11th Symposium on Adaptive Processes, Vol. 11, 1972.

[33] J. Cai, G. Thierauf, Discrete optimization of structures using an improved penalty function method, Decision and Control 21 (4) (1993) 293–306.

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

[35] J. Halton, On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals, Numerische Mathematik 2 (1960) 84–90.

[36] E. P. Gill, W. Murray, M. A. Saunders, E. Wong, User's guide for SNOPT 7.6: Software for large-scale nonlinear programming (CCoM 17-1).

[37] E. P. Gill, W. Murray, M. A. Saunders, SNOPT: An SQP algorithm for large-scale constrained optimization, SIAM Rev. 47 (2005) 99–131.

[38] S. Khan, E. Gunpinar, M. Moriguchi, Customer-centered design sampling for cad products using spatial simulated annealing, Proceedings of CAD'17, Okayama, Japan (2017) 100–103.

[39] B. Chen, Y. Pan, J. Wang, Z. Fu, Z. Zeng, Y. Zhou, Y. Zhang, Even sampling designs generation by efficient spatial simulated annealing, Mathematical and Computer Modelling 58 (3-4) (2013) 670–676.

[40] J. W. V. Groenigen, A. Stein, Constrained optimization of spatial sampling using continuous simulated annealing, Mathematical and Computer Modelling 27 (1998) 1078–1086.

[41] D. Francois, in: High-dimensional data analysis: From optimal metric to feature selection, VDM Verlag, Saarbrcken, Germany, 2008.

[42] P. Store, Hanedan Ewer, `https://www.pasabahcemagazalari.com/`, [Online; accessed 03-August-2016] (2016).

[43] 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.