DMS2015-33: Generative Interface Structure Design for Supporting Existing Objects

Nurcan Gecer Ulu
Carnegie Mellon University

Levent Burak Kara
Carnegie Mellon University

***Abstract*—Increasing availability of high quality 3D printing devices and services now enable ordinary people to create, edit and repair products for their custom needs. However, an effective use of current 3D modeling and design software is still a challenge for most novice users. In this work, we introduce a new computational method to automatically generate an organic interface structure that allows existing objects to be statically supported within a prescribed physical environment. Taking the digital model of the environment and a set of points that the generated structure should touch as an input, our biologically inspired growth algorithm automatically produces a support structure that when physically fabricated helps keep the target object in the desired position and orientation. The proposed growth algorithm uses an attractor based form generation process based on the space colonization algorithm and introduces a novel target attractor concept. Moreover, obstacle avoidance, symmetrical growth, smoothing and sketch modification techniques have been developed to adapt the nature inspired growth algorithm into a design tool that is interactive with the design space. We present the details of our technique and illustrate its use on a collection of examples from different categories.**

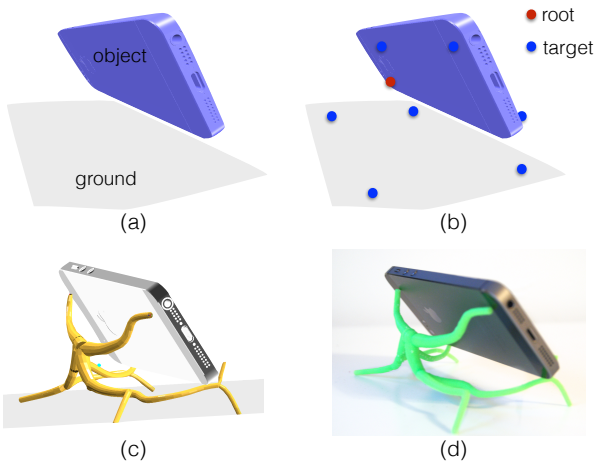

Fig. 1. Example problem to generate a phone stand (a) given object and environment configuration (b) user defined target and root points (c) generated interface structure (d) 3D printed result.

I. Introduction

The customization and personalization of products started to compete with traditional mass production principles with the contribution of maker movement and DIY (Do-It-Yourself) culture. DIY commonly refers to any fabrication, modification or repair event that is outside of one's professional expertise [1]. With the rise of DIY culture, there is a growing interest for design and fabrication tools tailored towards non-expert users.

Recent advances in 3D design and manufacturing technologies now have made content creation accessible to novice users. Besides the basic consumer level 3D printers, online on-demand 3D printing services (e.g. Shapeways, i.Materialise) have enabled ordinary people to access high quality machines. 3D modeling software, such as Autodesk 123D and Tinkercad, allow consumers to create 3D shapes using simplified geometric interaction methods. However, current commercial design software do not take advantage of capabilities of 3D printing. While *almost anything* can be fabricated using 3D printing, these design software limit potential design outputs by mimicking features of traditional manufacturing and assembly methods. In this work, we extend the design possibilities by taking a generative design approach to create organic looking branching shapes that would be challenging to design and fabricate with traditional methods.

We propose a framework that automatically generates interface structures under prescribed constraints. The input to our algorithm is a surface mesh for the object to support and a mesh to represent the ground surface with target and root points to create a shape in between (Fig.1). Then, automatic interface structure generation is achieved by a nature inspired growth mechanism. Users can control the design by changing target-root combinations at the input phase as well as by using sketch modifications after the shape is created. Moreover, the stochastic nature of the growth algorithm lets users design one of a kind pieces by generating different outputs for the same problem on each run. The main contribution of this work is the novel application of a nature inspired growth algorithm for automatic product generation. This is accomplished by the introduction of target attractor and pruning concepts, embedding product design considerations and user interaction.

II. Related Work

Design Tools for Non-Expert Users have recently received significant attention. In [2], a chair design tool is proposed to create balanced chairs from extruded 2D profile sketches. To enable informed exploration, Umetani et al. [3] proposed a suggestive design tool for plank-based furniture. In that work, the user adds planks and edits their positions, orientations or size. A data-driven approach to interactive design of model

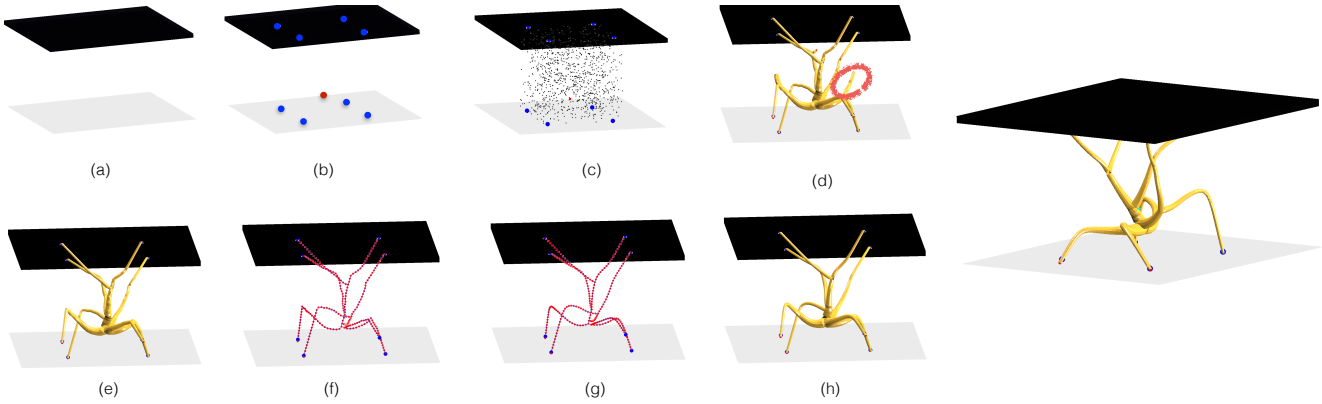

Fig. 2. Interface structure generation process. First, user defines the environment configuration (a) and selects the root and target points shown as red and blue, respectively (b). Attractors are generated randomly in the design space (c) and interface structure is grown automatically (d). Then, unnecessary branches are removed automatically (d-e) and the skeleton of the interface structure is smoothed as desired (f-h). Resulting shape is shown on the right.

airplanes is proposed in [4] where the user creates free-flight gliders with 2D sketches. In this work, we focus on creating a large range of products instead of one specific group such as chairs or gliders. While all three systems are notable interactive tools, components of the resulting designs are limited to 2D laser cut pieces. Our system generates organic 3D geometries that can take advantage of the opportunities in 3D printing.

Much of recent research on design for 3D printing addresses modifications of existing digital models by optimizing physical properties, such as balance and structural strength. For this purpose, inner carving with deformations [5, 6] and thickening of thin sections [7] have been used. Here, we focus on geometric shape *generation* whereas their focus is on shape modification.

Generative Design methods are recognized as significant technologies to rapidly generate different design alternatives. Fabrication of generatively produced designs have been examined illustrating how geometrically complex shapes can be physically created in [8, 9]. In computer graphics, generative design methods have been used to create architectural models such as buildings [10], virtual cities [11] and trees [12, 13]. In this paper, we are inspired by a specific generative design method developed to simulate the tree growth process for automatic shape generation.

Tree Growth methods have been widely used in computer graphics for urban modeling and computer animation. As such, tree growth has been vastly investigated in the literature. L-systems have been used to generate trees in [12]. Runions et al. [13] proposed a space colonization algorithm to mimic open and closed venation process in the leaf formation. Later, the space colonization algorithm has been extended to grow trees in [14] and [15]. Tree generation inside various geometries is investigated using spatial attractor distribution in [16]. In this paper, our automatic interface generation process has been inspired by the space colonization algorithm. We use its spatial attractor distribution feature to enable the interaction with the design space. Moreover, interaction of the tree model with

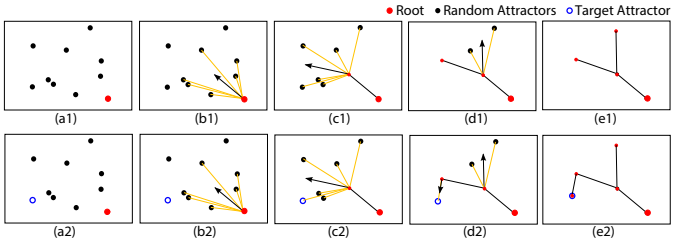

Fig. 3. Space colonization algorithm with (a1-e1) and without (a2-e2) target attractors.

the obstacles in the environment has been studied in [17]. In that, a fully grown tree model is placed around an object and the colliding branches are removed. For our purposes, this approach cannot be used since the grown structure has to be connected to the target points and a branch connected to a target can not be removed. Hence, we utilize obstacle avoidance during the growth process.

III. Automatic Shape Generation

In this work, the aim is to automatically create an interface structure between given objects. This process is illustrated in Fig.2 starting with the user input and the steps of the automatic shape creation performed on the background. First, the user supplies the input geometries as 3D mesh models (a). Then, a set of target points are selected by the user to define where the interface structure should be in contact with the input models (b). Then, a root point or points are provided by the user to start the growth process (b). Attractors are randomly generated inside the design space (c). The structure is generated akin to a tree originating at its root and growing in 3D space to reach the targets (d). Branches that are not connected to the input objects through target points are removed from the structure (e). We also refer to this step as pruning or unnecessary branch removal. Finally, the skeleton of the structure (f) is smoothed (g-h). In the following sections, the details of these steps are described.

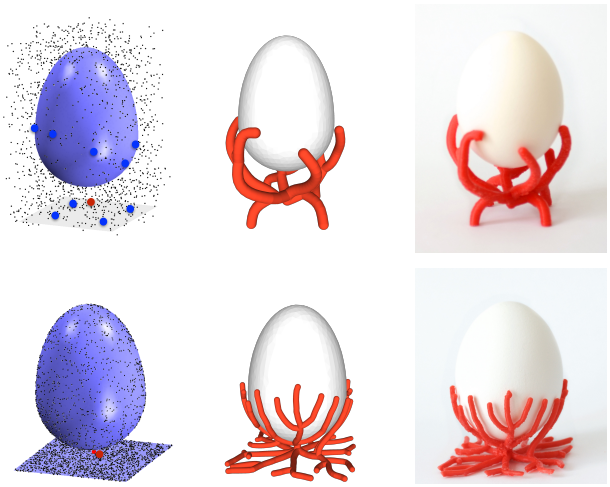

Fig. 4. Egg holder generated using volume (top) and surface (bottom) hormones. Left: problem setup, middle: digital model, right: 3D printed result.

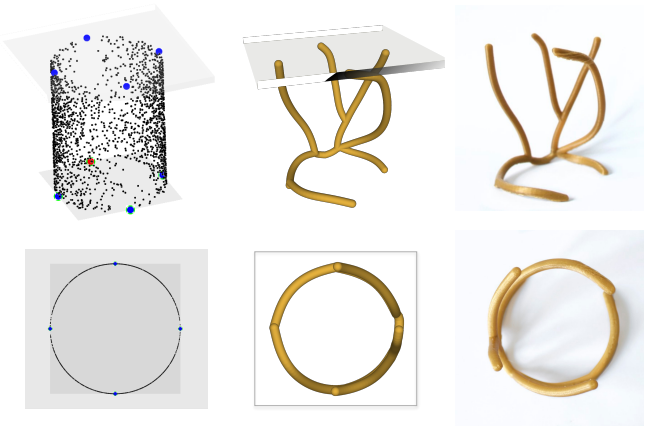

Fig. 5. Example projection growth. Generating the interface structure .Left: problem setup with root & target points illustrated, middle: digital model, right: 3D printed result. Orthogonal (top) and top (bottom) views of the same part are given.

A. Growth Algorithm

The proposed method uses an attractor based growth approach of space colonization algorithm given in [13]. Space colonization algorithm creates a branching tree structure in space as demonstrated in Fig.3.a1-e1. The tree structure grows without the guarantee that it would touch any specific point in the design environment. In this work, we need to create shapes between objects ensuring that the generated interface would be in contact with the target objects to support them. For this reason, we introduce a novel target attractor concept to create branching structures that grow to the required target positions (Fig.3.a2-e2).

The target based growth process starts with the definition of the design space, e.g. rectangle in (a2) and target-root point selections. Then, random attractors are sampled uniformly inside the design space. These random attractors have an *influence distance* that they can pull a branch to themselves as well as a *kill distance* that makes them inactive when they get too close to a branch in the growing structure. At every growth step, depending on the influence and kill distance, each attractor is associated with the tree node that is closest to it (yellow lines) if the node is within the influence distance. Then, normalized vectors from the node to the attractors are created and their average (black arrow) is calculated and used as the growth direction for the node (b2). The new node is added in the growth direction in the distance of branch length. All attractors are checked if they are in the kill distances of nodes. In other words, an attractor is killed if it is close enough to the tree (c2). This process iterates until all attractors are killed. While target attractors also pull the branches towards them, they are a special type of attractor with *zero* kill distance. If an attractor is a target attractor, it does not get *killed* until a tree node reaches it (notice difference in d1 and d2). The position of a new node is calculated as follows

$$\vec{v}' = \begin{cases} \vec{t}, & \text{if reaching target} \\ \vec{v} + L\hat{n}, & \text{otherwise} \end{cases} \tag{1}$$

$$\vec{n} = \frac{\vec{a} - \vec{v}}{||\vec{a} - \vec{v}||} \tag{2}$$

$$\hat{n} = \frac{\vec{n}}{||\vec{n}||} \tag{3}$$

$\vec{v}'$, $\vec{v}$, $\vec{t}$, L, $\hat{n}$, $\vec{a}$ are the position vector of a new node, the position vector of node in the tree set, the position vector of the target attractor, branch length, unit growth direction vector and position vector of attractor in the set, respectively.

B. Random Attractor Placement

Placement of random attractors is a crucial step in our algorithm, especially to create variations for the same problem. Since the placement of the attractors defines the virtual design space in which our structure can grow, how attractors are placed in 3D drastically affects the resulting geometry. This effect is demonstrated for three distinct cases in Fig.4 and Fig.5. Figure 4 compares the use of volume and surface attractors to generate an interface structure between the same objects. In the first one, we use the bounding box volume of the two objects to generate the attractors inside of the volume. On the other hand, in the second one, attractors are sampled on the surface of these objects. From this figure, it can be seen that the resulting interface geometries with very distinct characteristics can be obtained by only changing the distribution of the attractors even for the same problem setting. Here, an important distinction between these two cases is that we do not require target attractors for the surface growth case simply because we are guaranteed to touch the surfaces of both objects in this case. Another distinct case for attractor distribution is illustrated in Fig. 5. Here, the aim is to generate an interface structure that would give a desired 2D profile

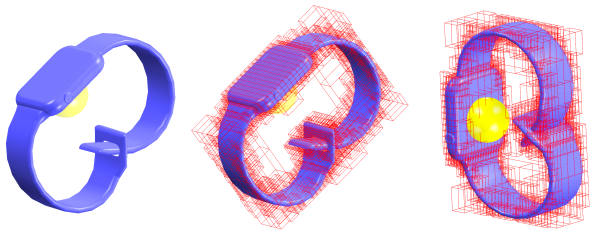

Fig. 6. Obstacle avoidance is used to restrict growth in specific parts in the space. The restricted regions may be the objects represented as octrees or user defined spheres.

when viewed from a certain direction. For this purpose, we sample the attractors on a surface that is created by extruding the desired 2D profile in the viewing direction. Hence, this specific attractor generation case can be classified as a subset of the surface growth explained previously. Moreover, any 3D swept/curved surface can be used to create 3D profiles. However, the main distinction here is that we require target attractors to be defined in this case, to ensure the resulting structure touches and supports the objects in the problem setup. This is mainly because the surface which the attractors generated on is a virtual one rather than the actual surface of the objects.

Apart from the aforementioned cases, we also enable symmetry in the resulting geometries. In product design, symmetry is considered to be a critical feature for everyday objects [18]. In our shape creation algorithm, we can ensure symmetry by simply placing the attractors in the design space symmetrically. Hence, the addition of a symmetry feature does not add any computational cost in our algorithm. However, the only case that may need special attention is the one where the root point is placed on the symmetry plane. In such cases, growth only happens on the symmetry plane because of the equal attraction from both sides. We solve this problem by moving the user defined root point slightly in both directions orthogonal to the symmetry plane by duplicating the root point.

C. Obstacle Avoidance

When designing an object, its interaction with the environment is important. For this reason, structure growth may be restricted in some parts of the design space. First of all, the interface structure should not intersect with the objects that it is intended to support. For this purpose, we utilize mesh representations of the objects for collision detection. In addition, users may define geometric obstacles in the form of spheres to restrict the growth. As an example for the use of spheres for functional purposes, a sphere is placed under the smart watch to limit the growth of the interface structure on the magnetic touch charging area in Fig 6.

During growth, the intersection of the new branch and obstacles are tested at each step. If there is a collision between the obstacle and the branch, a random direction is chosen for growth until collision is eliminated or maximum number of trials is reached. Intersection tests are conducted using a parametric representation of a line segment and an implicit representation of spherical and triangular objects. Details of the intersection test can be found in [19]. To increase the efficiency of collision detection for triangular meshes, we use octree representation [20].

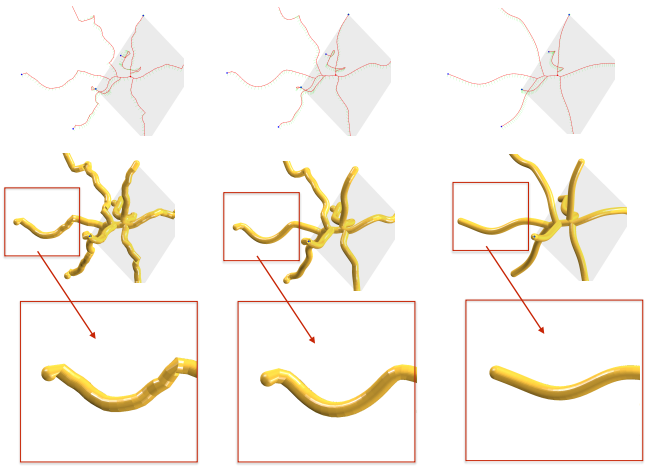

Fig. 7. Effect of Laplace and Biharmonic operators on smoothing is illustrated for skeleton (top) and skin (middle-bottom) of the resulting geometries. Left: the original, middle: after smoothing with Laplace & Biharmonic operators together, right: after smoothing with Laplace operator only. Note that the use of the combined Laplace & Biharmonic operators allows smoothing without significant shrinkage.

D. Smoothing

While jagged transitions between biological branches look realistic for trees, smooth transitions are usually more appealing in product design. For this purpose, we apply curve smoothing to the tree structure as shown in Fig.7. Here, the position of each node on the tree skeleton is updated based on the positions of the neighboring nodes using Laplacian and Biharmonic operators as follows [21, 22].

$$\vec{v}_i = \vec{v}_i + \lambda_1 \triangle\vec{v}_i + \lambda_2 \triangle(\triangle\vec{v}_i) \tag{4}$$

where Laplace and Biharmonic operators can be defined as:

$$\triangle\vec{v}_i = \nabla^2\vec{v}_i = \frac{1}{2}(\vec{v}_{i+1} - \vec{v}_i) + \frac{1}{2}(\vec{v}_{i-1} - \vec{v}_i) \tag{5}$$

$$\triangle(\triangle\vec{v}_i) = \nabla^4\vec{v}_i \tag{6}$$

$\vec{v}_{i-1}$ and $\vec{v}_{i+1}$ denote two neighbors of the node, $\vec{v}_i$.

In order to achieve smooth curves, we linearly combined Laplace and Biharmonic operators. Although it is possible to accomplish smoothing with only the Laplacian term, Biharmonic term is included to suppress the shrinking behavior arising from the Laplace operator when used alone (Fig.7). In this paper, we select the coefficients λ_1, λ_2 as 0.2 and 0.1, respectively.

E. Branch Pruning

As can be observed from Fig.2 and Fig.3, our approach creates many branches that may not serve a structural function on the interface (i.e., branches that do not touch a target point). Hence, branches that are not connected to the target object or ground are automatically detected and removed form the tree graph (Fig.2(d)-(e)).

F. Modifications and Variation In The Design

Although we produce the interface structures automatically, we enable users to control many aspects of the geometry generation. The user control starts by importing 3D models of the objects and the selection of root-target attractor configurations. Then, another significant control comes from the placement of random attractors as explained previously. In addition to these inputs, there are four factors that affect the growth process (1) influence distance, (2) kill distance, (3) branch length and (4) number of random attractors. These factors are very important to generate variations in the space colonizations algorithm for tree generation such as trees with dense or sparse branch structures [14]. On the other hand, results of our proposed growth algorithm are not affected by the changes in those parameters primarily due to the target attractor and branch removal concepts. As long as the parameters are *suitable*, results do not change significantly. There is a wide range of suitable parameters for a given problem. Any suitable parameter set has the following properties:

- Influence distance is greater than kill distance.
- Branch length, which can be considered as step size, is small compared to the environment dimensions but it is large enough to facilitate efficient growth computation.
- The number of random attractors is high enough to create uniform distribution in the design space, we used 2000 attractors for the examples in the paper.
- Influence distance is high enough to enable attraction of a node for the created uniform distribution.

In this work, we choose default values using the given guidelines. For each problem setup, we use the default values by scaling them with the dimensions of the bounding box of the system.

Another set of important controls comes into play after the interface structure is generated automatically. At this point, users can control the radius variation in the branches of the interface structure as well as modify the skeleton of the structure by sketched strokes. Now that the skeleton of the structure is obtained, a 3D skin is created by covering each branch with a truncated cone and taking the union of all cones. The radius at each node is calculated based on its age as

$$r = r_{min} + (r_{max} - r_{min}) * e^{-k\alpha} \tag{7}$$

where r, r_{min}, r_{max}, α, k is the radius, minimum radius, maximum radius, age and decay of radius, respectively. Here, age of each node is determined in such a way that every node starts with age of 0 and the age increases by 1 at each growth step. Decay of radius defines how fast the radius changes from the root to the targets between maximum radius and minimum radius and is set by the user. Also, r_{min}, r_{max} are set by the user.

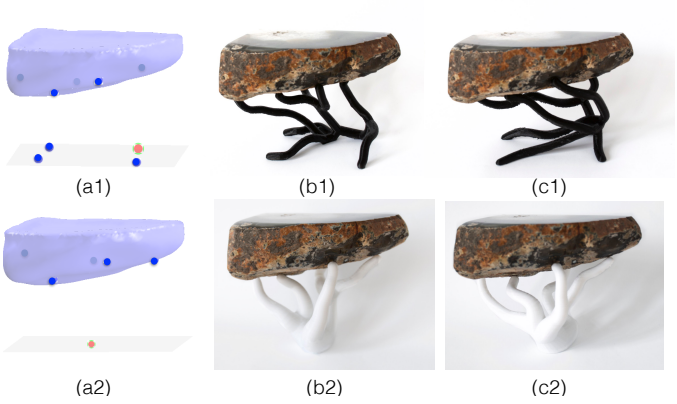

Fig. 8. Effect of target-root selection and stochastic growth demonstrated on two different target-root configurations (a1-c1, a2-c2). For the same problem setup (a), various stochastic growth results (b and c) can be obtained.

Sketch modifications are performed through modifier sketches performed by the user to specify the new shape of the skeleton curve as it would occur from the current viewpoint. To do this, a surface is created by the rays emanating from the user's eyes, passing through the strokes and extends into the page. In theory, there are infinitely many candidate solutions on this surface. The best 3D configuration is thus found by computing the minimum distance projection of the original curve onto the surface. For the details of the sketch modifications please refer to [23].

IV. Results and Discussion

Our approach enables the automatic generation of interface structures for 3D printing. The user may control the geometry generation through target-root configurations, random attractor placement, skin radius selection and sketch modifications. We applied our algorithm to a variety of examples including gadget accessories, decoration and restoration of existing objects and furniture. In order to transform the existing objects into the digital design environment, we utilized 3D scanning using a Kinect device. We downloaded the 3D digital models through the stock 3D model websites like GrabCAD and Google 3D Warehouse for the common objects.

The latest trends in decorating and modern furniture design include hybrid design approaches where natural materials with imperfections are combined with machine-made parts to create innovative and original designs. In Figure 8, an example hybrid design created using our system is shown. Here, we take a natural rock piece and design a support structure that complements its organic geometrical features. One important control that our system provides is the target-root placement. We generated two different target-root configurations (a1, a2) for the same problem to demonstrate the significant variance in the resulting geometries (b1, b2). Moreover, we would like to draw attention to the stochastic nature of our algorithm

Fig. 9. Use of sketch modifications for a functional purpose. Top part of the planter holder is enlarged to be able to insert the planter. Left: Sketch modification steps, Right: 3D printed result with the inserted planter.

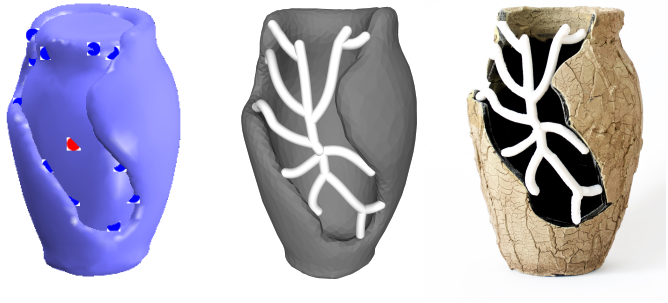

Fig. 10. Form completion: A broken vase is restored using a scanned model. Left: problem setup, middle: digital model, right: 3D printed result.

that comes from the random sampling of attractors inside the design space where different results are obtained for distinct set of random attractors. However, the effect of the stochastic nature on the resulting geometries (b1-c1 and b2-c2) are subtle compared to the effect of target-root selection.

Another direction for craft, arts and design is the restoration of broken objects through 3D printing to obtain new artistic expressions rather than restoring the original object [24]. In that, the motivation is not to restore the initial function of the object, but rather use it to function as a memorial. In Fig.10, we show that our method can be used for similar purposes. Here, a missing part of a broken vase is restored with the generated interface shape. For this, we first scanned the broken vase and placed desired target-root points. Then, the resulting piece to complete the broken part is grown using our algorithm. We also 3D printed and assembled the resulting part to the broken vase. Another alternative interface structure for this example can also be seen in Fig.12.a.

In addition to the aesthetic needs, the need for sketch modifications may arise from functional requirements. Use of sketch modifications for a functional purpose is illustrated in Fig.9. In this example, a hanger is designed to suspend the planter. However, the planter can not be inserted into this automatically generated structure. For this reason, sketch modifications are applied on the skeleton of the structure to enlarge the top part of the hanger so that the planter can be inserted.

In some configurations, users might need to control the growth process more strictly to achieve a geometry with particular desired properties. In such cases, our geometry creation process can be guided by progressively manipulating the problem setup. To do this, instead of defining all target points at once, we start with a subset of targets and progressively add the remaining ones as we grow the structure. Figure 11 demonstrates this on an example to attach the phone to a baseball cap for first person view camera shots. Here, the aim is to guide the growth on the side of the cap instead of any other possible outcome. First, only five of the targets are defined (a) and the growth process is completed (b). Then, a new target point is added (b) and another growth process is accomplished. This process is iterated (c) until the final desired shape is created (d). Since we are using a consumer level 3D printer with a limited build volume, we partitioned the resulting object into smaller pieces to be able to 3D print. For the assembly, we manually added dovetail structures on the assembly surfaces (f).

There are many communities that promote reuse of materials through community engagement, resource conservation and creativity e.g. Pittsburgh Center for Creative Reuse and Lancaster Creative Reuse. Since our design framework is developed to work with existing objects, users can easily utilize our algorithm for creative reuse purposes. A virtual example of material reuse is shown in Fig. 12.e. Here, the usage of a seat and back from a broken chair to design a new support and legs is demonstrated. In the example, while we have virtual models for the elements to be reused, as mentioned earlier any object can be scanned and used to create the interface structures. Although for the previous examples, we focus on creating attachment structures that hold the object in place without fixing or gluing, this example requires the interface structure to be fixed to the supported objects.

Since our framework is tailored towards non-expert users, we fabricated all our examples with a consumer level low-cost 3D printer, PrintrBot Simple 1405, to study the printability of our results. However, more advanced 3D printers can be used to fabricate resulting geometries with higher qualities using various material options.

We recorded the computation time for automatic shape generation for a number of examples. Since our method has a stochastic nature, computation time changes as the random attractor set changes. Thus, the results are reported for three different random attractor sets for each example in Table I. One reason computation time changes for each example is the change in the complexity of the objects that increases the time for collision checks. Another, important factor is how easy or difficult it is to reach the targets inside the design space.

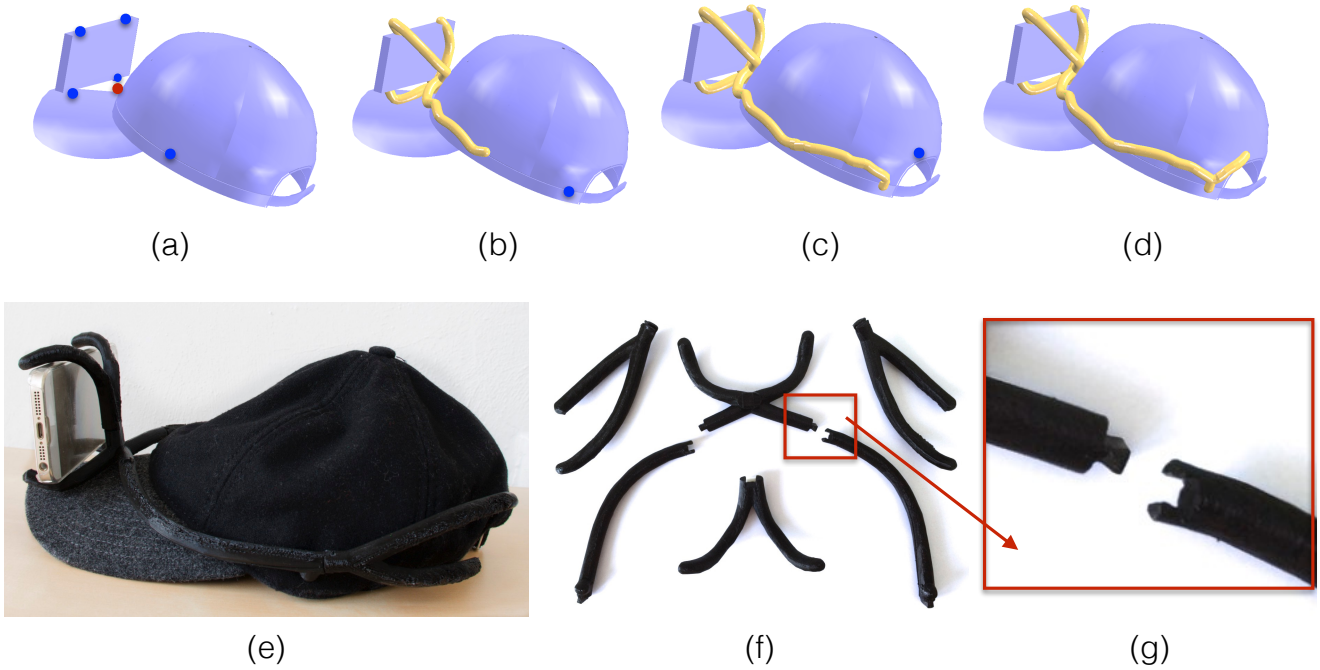

Fig. 11. Guided progressive growth (top) is shown on a baseball cap example to attach a phone for first-person view recording. Use of assembly structures to 3D print larger designs (bottom) have been demonstrated with the zoomed in dovetail joint detail (g).

TABLE I
RUNTIME PERFORMANCE OF OUR GENERATIVE DESIGN ALGORITHM

	Total Run Time [s]				
	Run 1	Run 2	Run 3	Run 4	Run 5
Fig.12.a	11	12	10	10	9
Fig.12.b	< 1	< 1	< 1	< 1	< 1
Fig.12.c	< 1	< 1	< 1	< 1	< 1
Fig.12.d	2	2	2	2	2
Fig.12.e	4	2	4	3	2

User Study:

We conducted a user study to evaluate the usability of our system. 25 users who had no prior knowledge of our software participated in the user study. Each participant was given 30 minutes to finish all the tasks including software introduction, two modeling assignments and completing out a post survey. The same two modeling tasks were assigned to all participants as shown in Fig. 13. Some example designs generated by the users are also shown.

All users were able to complete the tasks in 30 minutes or less. We believe this indicates that users were able to learn the software easily and use it efficiently. The survey results also support this with a strong agreement in questions 1, 3 and 4 (Fig. 14). During the user study, we observed some issues with shape modification. Some participants had a hard time figuring out how sketch modification works. This observation also explains the weaker agreement in question 2 in the survey. However, our observations showed that if the participants experimented more with the sketching part of the software, they were able to understand and efficiently use sketch modifications.

In addition to the Likert scale questions (Fig. 14) on the survey, we asked the participants to write comments if they had any. One important conclusion we drew from some of the comments was that some of the participants tended to imagine a design and tried to generate that exact solution. Since, our system automatically creates shape solutions to a given problem, the software is not intended to be used to produce a specific shape the user has in mind. It was interesting to see that people felt compelled to control every aspect of the shape with the conventional design approaches even when the results were automatically generated for them. We believe increasing the expressive power of such a design system is very important even when the results are automatically created.

V. LIMITATIONS AND FUTURE WORK

Our focus has been on generating tree-like structures on the skeletons we grow that make the resulting designs resemble biological trees. We expect the proposed formulation to be readily applicable with different building blocks instead of our current truncated cones to achieve a richer variation in form. Moreover, since our obstacle avoidance is achieved through random search directions, our algorithm may not converge to a solution within predefined maximum number of trials. While we have observed this issue rarely, increasing

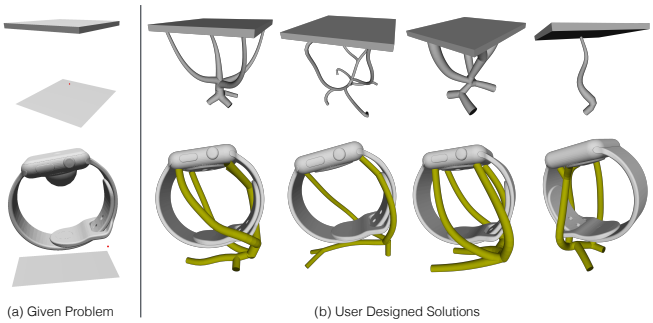

Fig. 13. Users were given two design problems: designing legs for a table top (a-top row), and designing a stand for a smart watch (a-bottom row). Some designs created by the users are shown (b). The users configure the root and target nodes for the given problem and the software produces the final shape. The users may further modify the shapes using sketch input.

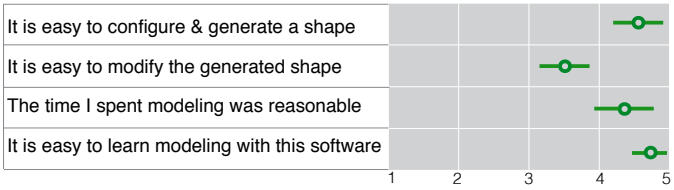

Fig. 14. Survey results collected from all 25 participants of the user study (1: Strongly Disagree, 5: Strongly Agree). Circles and error bars represent the mean value and one standard deviation, respectively.

the maximum trial number for complex problem settings may be required. Finally, in this work, we do not consider structural performance of the resulting shapes. Yet, our algorithm can be extended to ensure structural soundness for a given problem configuration. This may require finite element analysis during the shape generation process.

VI. Conclusion

We present a generative design framework to create interface structures to support existing objects. The proposed method enables novice users to automatically generate geometries and edit them once the shape is created. Our approach introduces a novel application of a nature inspired growth algorithm with embedded product design considerations. Our current studies indicate that the approach works well for a variety of design problems with the presented actual 3D printed results alongside their digital models. Also, the user study supports the practical usage of the proposed system. We consider this work as a step towards future customized design software where users only define functional constraints and the CAD system automatically creates a solution.

Acknowledgment

Authors would like thank Ye Han for his help on the assembly structure creation.

References

[1] C. Mota, "The rise of personal fabrication," in *Proceedings of the 8th ACM Conference on Creativity and Cognition*, ser. C&C '11. New York, NY, USA: ACM, 2011, pp. 279–288. [Online]. Available: http://doi.acm.org/10.1145/2069618.2069665

[2] G. Saul, M. Lau, J. Mitani, and T. Igarashi, "Sketchchair: An all-in-one chair design system for end users," in *Proceedings of the Fifth International Conference on Tangible, Embedded, and Embodied Interaction*, ser. TEI '11. New York, NY, USA: ACM, 2011, pp. 73–80. [Online]. Available: http://doi.acm.org/10.1145/1935701.1935717

[3] N. Umetani, T. Igarashi, and N. J. Mitra, "Guided exploration of physically valid shapes for furniture design," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2012)*, vol. 31, no. 4, 2012.

[4] N. Umetani, Y. Koyama, R. Schmidt, and T. Igarashi, "Pteromys: Interactive design and optimization of free-formed free-flight model airplanes," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 65:1–65:10, Jul. 2014. [Online]. Available: http://doi.acm.org/10.1145/2601097.2601129

[5] R. Prévost, E. Whiting, S. Lefebvre, and O. Sorkine-Hornung, "Make It Stand: Balancing shapes for 3D fabrication," *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, vol. 32, no. 4, pp. 81:1–81:10, 2013.

[6] M. Bächer, E. Whiting, B. Bickel, and O. Sorkine-Hornung, "Spin-It: Optimizing moment of inertia for spinnable objects," *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, vol. 33, no. 4, 2014.

[7] O. Stava, J. Vanek, B. Benes, N. Carr, and R. Měch, "Stress relief: Improving structural strength of 3d printable objects," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 48:1–48:11, Jul. 2012. [Online]. Available: http://doi.acm.org/10.1145/2185520.2185544

[8] Y. Wang and J. P. Duarte, "Automatic generation and fabrication of designs," *Automation in Construction*, vol. 11, no. 3, pp. 291 – 302, 2002, rapid Prototyping. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0926580500001126

[9] L. Sass and R. Oxman, "Materializing design: the implications of rapid prototyping in digital design," *Design Studies*, vol. 27, no. 3, pp. 325 – 355, 2006, digital Design Digital Design. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0142694X05000864

[10] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. Van Gool, "Procedural modeling of buildings," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 614–623, Jul. 2006. [Online]. Available: http://doi.acm.org/10.1145/1141911.1141931

[11] Y. I. H. Parish and P. Müller, "Procedural modeling of cities," in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '01. New York, NY, USA: ACM, 2001, pp. 301–308. [Online]. Available: http://doi.acm.org/10.1145/383259.383292

[12] P. Prusinkiewicz, M. James, and R. Měch, "Synthetic topiary," in *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '94. New York, NY, USA: ACM, 1994, pp. 351–358. [Online]. Available: http://doi.acm.org/10.1145/192161.192254

[13] A. Runions, M. Fuhrer, B. Lane, P. Federl, A.-G. Rolland-Lagan, and P. Prusinkiewicz, "Modeling and visualization of leaf venation patterns," in *ACM SIGGRAPH 2005 Papers*, ser. SIGGRAPH '05. New York, NY, USA: ACM, 2005, pp. 702–711. [Online]. Available: http://doi.acm.org/10.1145/1186822.1073251

[14] A. Runions, B. Lane, and P. Prusinkiewicz, "Modeling trees with a space colonization algorithm," in *Proceedings of the Third Eurographics Conference on Natural Phenomena*, ser. NPH'07. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2007, pp. 63–70. [Online]. Available: http://dx.doi.org/10.2312/NPH/NPH07/063-070

[15] W. Palubicki, K. Horel, S. Longay, A. Runions, B. Lane, R. Měch, and P. Prusinkiewicz, "Self-organizing tree models for image synthesis," in *ACM SIGGRAPH 2009 Papers*, ser. SIGGRAPH '09. New York, NY, USA: ACM, 2009, pp. 58:1–58:10. [Online]. Available: http://doi.acm.org/10.1145/1576246.1531364

[16] S. Longay, A. Runions, F. Boudon, and P. Prusinkiewicz, "Treesketch: Interactive procedural modeling of trees on a tablet," in *Proceedings of the International Symposium on Sketch-Based Interfaces and Modeling*, ser. SBIM '12. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2012, pp. 107–120. [Online]. Available: http://dl.acm.org/citation.cfm?id=2331067.2331083

[17] S. Pirk, O. Stava, J. Kratt, M. A. M. Said, B. Neubert, R. Měch, B. Benes, and O. Deussen, "Plastic trees: interactive

self-adapting botanical tree models," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 50:1–50:10, Jul. 2012. [Online]. Available: http://doi.acm.org/10.1145/2185520.2185546

[18] B. De Mozota, *Design Management: Using Design to Build Brand Value and Corporate Innovation*. Skyhorse Publishing Company, Incorporated, 2003. [Online]. Available: https://books.google.com/books?id=jpy_JBhZ7nUC

[19] P. Shirley and S. Marschner, *Fundamentals of Computer Graphics*, 3rd ed. Natick, MA, USA: A. K. Peters, Ltd., 2009.

[20] J. Revelles, C. Urea, and M. Lastra, "An efficient parametric algorithm for octree traversal," in *Journal of WSCG*, 2000, pp. 212–219.

[21] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. uno Levy, *Polygon Mesh Processing*. AK Peters, 2010.

[22] E. B. Arisoy, G. Orbay, and L. B. Kara, "Free form surface skinning of 3d curve clouds for conceptual shape design," *Journal of Computing and Information Science in Engineering*, vol. 12, p. 031005, 2012.

[23] L. B. Kara and K. Shimada, "Sketch-based 3d-shape creation for industrial styling design," *IEEE Computer Graphics and Applications*, vol. 27, pp. 60–71, 2007.

[24] A. Zoran and L. Buechley, "Hybrid reassemblage: an exploration of craft, digital fabrication and artifact uniqueness," *Leonardo*, vol. 46, no. 1, pp. 4–10, 2013.

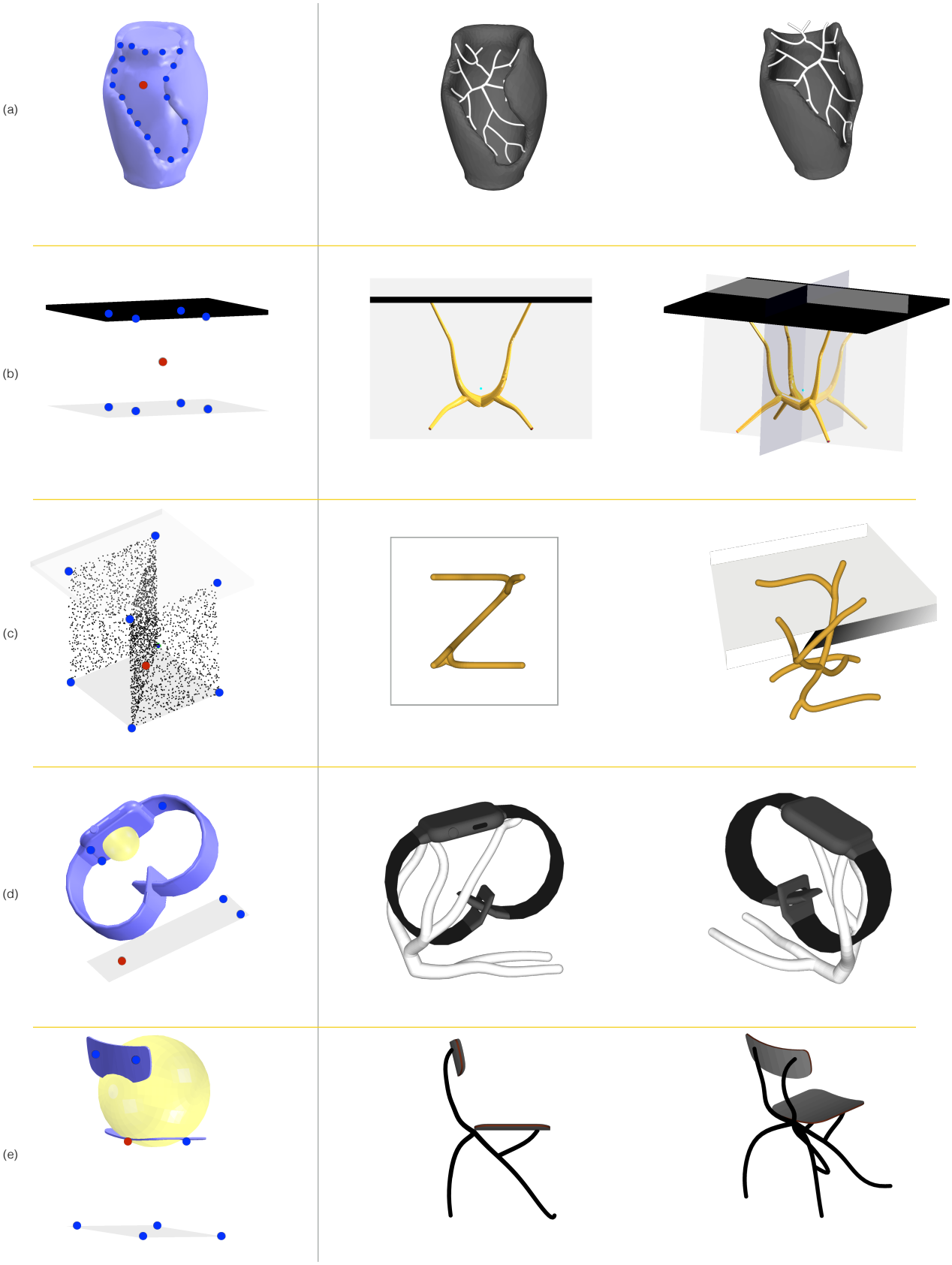

Fig. 12. Designs created with our system. Left: problem setup, right: resulting design from two different views.