


Pen2VR: A Smart Pen Tool Interface for Wire Art Design in VR

Wanwan Li[†] 

George Mason University, Department of Computer Science



Figure 1: Through Pen2VR, a smart 3D pen tool VR interface, wire arts (black curves) are designed by users via VR devices.

Abstract

In this paper, we present Pen2VR: a smart pen tool interface for 3D drawing wire art design in VR. In Pen2VR, with VR headsets put on, users are allowed to create their VR drawings as 3D lines or curves in a virtual space as if they are using a 3D version of the pen tool in Photoshop. During the 3D drawing in VR, users can directly create geometric elements including polylines and Bezier curves by clicking and dragging a VR controller in mid-air. Besides, in order to make Pen2VR smart, we proposed an optimization-based approach to automatically fine-tune the user's 3D initial input into well-beautified 3D drawings. Experimental results show that the qualities of optimized wire art designs have been significantly improved.

CCS Concepts

• *Applied computing* → *Fine arts*; • *Human-centered computing* → *User interface programming*; *User interface toolkits*; *User interface design*; *Wireframes*;

1. Introduction

Wire art [McG02] is a branch of fine art that is creating sculpture-like art crafts using curves and lines. Wire arts have different forms and styles according to different genres. Some of the wire arts are exploring the beauty of simplifications while some are expecting highly detailed minor structures within the design. In general, wire arts are using spatial curves to present the geometrical shapes of the sculpture. Through professional fabrications processes [WYFI19], wire arts are created through entangled metal, wood, or plastic

wires. Wire arts can be classified according to their functionalities in decorations [Rey97]. For example, stand-able wire arts that mimic the daily objects put on the desktop are typically made in a small size. Decorative wire arts can be larger scale and hanged on walls to mimic some picture or painting works. Since such fascinating features of wire arts, wire arts designs have been more popular since the new ages and become important parts of modern arts.

A preferable interactive environment for wire art design is the key. Traditional 2D screen-based drawing interfaces are lacking immersion during the designing process [CGPI16]. In contrast, mid-air 3D sketching interfaces provide the possibilities for the artists to design with 3D virtual curves directly in the air. The content-aware

[†] Email: wanwanli.gmu@gmail.com

art design in virtual space gives art designers more freedom to feel how the created artwork is placed in the environment [Liu12]. At the same time, as virtual reality platforms become mature, accurate hand positioning [HLC*20, KJX20] can be achieved through modern VR devices easily, so that users can draw sketches in VR accurately. These advanced VR technologies are attracting more and more artists to exploring transitioning from traditional 2D interfaces towards 3D VR digital art design interfaces.

There are lots of free-hand 3D VR sketch interfaces that have been well-developed and have been proved to be easy-to-use [DGK*20, GJS18, KJ05]. However, it is inconvenient to use existing VR interfaces for art designers to manually create the fabricable, stand-able, and stabilizable wire artwork which requires lots of minor considerations including the stability analysis, attachment between different parts, and the surface of the contact points to the ground are whether large enough to support the whole body, instead of merely considering the beauty of the art. Therefore, it is a challenging task to design a smart interface to automatically fine-tune the art design to make it satisfy these specific requirements.

To address these problems, we present Pen2VR: a smart pen tool interface for 3D drawing wire art design in VR. We first extended the commonly used pen tool interfaces embedded in Adobe Photoshop into a 3D version using a VR controller. Then, we propose an optimization-based approach to automatically help users fine-tune the user's 3D initial input into well-beautified 3D drawings so that the created wire artworks that are taking into account their regularity, connectivity, and standability. Our proposed algorithm can beautify artists' designs automatically so that no more manual efforts are demanded from artists to fine-tune their work. As shown in Figure 1, some wire artworks are designed through our Pen2VR interface by users. Our demo video can be viewed through this link <https://youtu.be/S1-8jSOq2F8>. We conclude the major contributions of our work as following items:

- Implementing an easy-to-use VR pen tool interface for 3D wire art design by clicking or dragging the VR controllers in mid-air.
- Proposing a novel optimization-based approach for beautifying the rough wire-art design considering the functionality of the wire art design such as regularity, connectivity, and standability.
- Conducting a series of experiments and statistical analysis to validate the effectiveness of our proposed optimization approach.

2. Related Work

Wire Art Design. Lots of recent research works have been done on designing smart wire art design interfaces. Liu et. al. [LCL*17] developed an image-based interface for reconstructing wire art design from 2D sketch into 3D. Through this interface, artists can design the wire art easily by drawing on paper and use the proposed approach to creating 3D wire arts from 2D sketches. Although 2D sketching of the wire art design is straightforward and easy to achieve, the imagination of the design process is seriously restricted compared to 3D design in virtual space. For extending the complexity of the art design, Hsiao et. al. [HHC18] have devised a multi-view wire art designing tool to automatically generate the wire-art that can be projected to different given 2D drawings in different directions. In their work, different constraints are put on to the wire art designing process. That is, the designed work needs

to satisfy the conditions that their particular protection must match with a specific 2D sketch input. Compared to the previous research work, this extends the artist's level of imagination from single view design to multi-view design. Afterward, Yang et. al. [YXFH21] presented an intelligent interface to convert the 3D mesh models directly into 3D wire art designs. This significantly shortens the design period of wire art crafts. However, the output is seriously dependent on the 3D model input.

Mid-Air Drawing. There are many interesting research works on VR art design through mid-air drawings. For example, a commercial product called Tilt Brush [Bru18] allows users to paint and draw in mid-air through HTC VIVE VR controllers. Later, a video-based tutorial system for art design in VR has been developed by Thoravi et al. [TKNDH19] for digital art edutainment. Furthermore, Eroglu et al. [EGS*18] proposed an immersive VR interface for fluid sketching in mid-air. As for solid models design in VR, Rodriguez et al. [Rod19] presented SurfaceBrush, through which they can convert VR drawings into manifold surfaces. Most recently, Arora et al. [AS21] presented an interface to draw curves on the surface in VR. Mid-air drawing interfaces have not only been explored on VR interfaces but also well-studied on AR and mobile devices. Arora et al. [AHKG*18] present a hybrid sketching system that combines 2D drawings and 3D drawings through a HoloLens head-mounted AR headset. Inspired by this, a mobile AR drawing interface has been developed by Kwan et al. [KF19] for users to draw 3D curves using a cellphone. Even though all of the mentioned works have been successfully developed for different kinds of applications on mid-air drawings, none of them beautify the drawings or address the functional aspects of the art design.

Sketch Beautification. Sketch beautification, also called sketch correction or sketch regularization, is a very important work area in the CAD industry. Especially, automation of sketch beautification is extremely important for facilitating accurate industrial designs. Interactive interfaces for sketch beautification have been well-studied by recent works. For example, sketch regularization techniques have been elegantly incorporated in accurate 2D-to-3D sketch reconstruction by Xu et al. [XCS*14] in True2Form. True2Form has explored the regularization terms that can be used to lift 2D sketches into 3D sketches. Our work has been inspired by this work to address the issues that appear during the wire art design. Zheng et al. [ZLDM16] has proposed an interactive sketching interface that can lift 2D sketches into 3D sketches by finding supporting planes for each stroke. This work has reminded us to regularize the coplanar strokes in the wire art design. The most relevant work to ours is the CASSIE proposed by Yu et al. [YAS*21]. Through this interface, users are allowed to use free-hand sketching in mid-air VR to create beautified 3D drawings and 3D surfaces. During the sketch beautification process in CASSIE, curve tangents and curve connectivity are considered to ensure surface smoothness. However, CASSIE only addresses the curves as input and ignores the functionality of the designed artwork such as the standability in the environment. Unlike CASSIE, our proposed Pen2VR can not only take Bezier curves as input but also support polylines as Pen2VR works more like a 3D version of the pen tool embedded in Photoshop which can easily generate accurate vector graphics and has been widely accepted in modern art design interfaces.

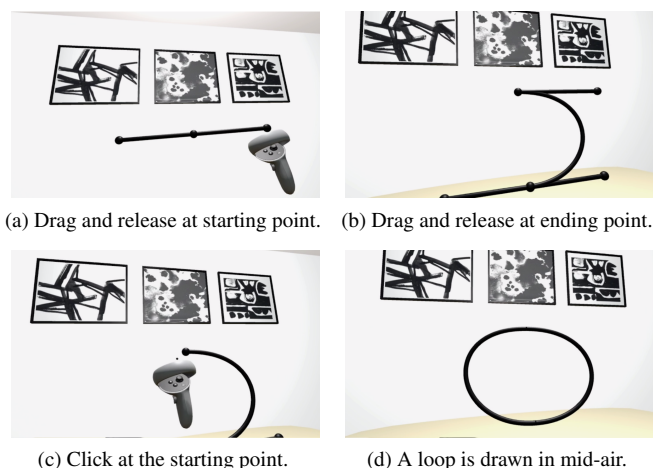


Figure 2: An example of drawing a circle loop in Pen2VR.

3. Technical Approach

3.1. Pen Tool Interface

According to the pen tool interface officially implemented in Adobe Photoshop, input is a sequence of 3D vectors $P = \{\mathbf{p}_i | i = 1, \dots, n\}$ that are representing the points, lines, and curves which are defined as the basic components of vector graphics. Commands for drawing shapes in vector graphics include: move to ($m \mathbf{p}_i$), line to ($l \mathbf{p}_i$), and cubic to ($c \mathbf{p}_i, \mathbf{p}_{i+1}, \mathbf{p}_{i+2}$). Move to ($m \mathbf{p}_i$) specifies the starting points of a continues vector shape (stroke) as \mathbf{p}_i . Line to ($l \mathbf{p}_i$) specifies a line connecting the point \mathbf{p}_i and its previous point \mathbf{p}_{i-1} . Cubic to ($c \mathbf{p}_i, \mathbf{p}_{i+1}, \mathbf{p}_{i+2}$) specifies a cubic Bezier curve of four points ($\mathbf{p}_{i-1}, \mathbf{p}_i, \mathbf{p}_{i+1}, \mathbf{p}_{i+2}$), where \mathbf{p}_i and \mathbf{p}_{i+1} are control points. In our implementation, we called the input vector sequence of all of these points P as anchor points. According to the move to ($m \mathbf{p}_i$) operations, we separate the sequence of anchor points into m subsequences which are called strokes $\{S_i | i = 1, \dots, m\}$ s.t. $P = S_1 \circ S_2 \circ \dots \circ S_m$, where \circ is the concatenation operator. If S_i 's starting point (the first anchor point in S_i) equals to S_i 's ending point (the last anchor point in S_i), we call S_i a loop.

According to these mathematical definitions, we provide a user-friendly interactive VR interface to help users input the strokes and loops efficiently. There are three kinds of VR controller events defined in Pen2VR: click, drag, and release. Typically, the first click is used to add the starting point. Following clicks are used add lines if there is no curve tangent currently defined; otherwise, add curves. Drag and release are used to define the curve tangents. During the dragging period, users can adjust the tangent direction and tangent length. A final click is used to either end up with a stroke by clicking near the stroke's ending point or end up with a loop by clicking near the stroke's starting point. As a hint, when the user's VR controller is moving nearby the starting point, a black sphere will pop up at the starting point. As shown in Figure 2, an example of an interaction sequence for drawing a circle loop is demonstrated.

3.2. Cost Functions

In order to make Pen2VR smart enough to create those wire artworks that are taking into account their regularity, connectivity, and standability, we propose an optimization-based approach to au-

tomatically help users fine-tune their initial input (anchor points $P = \{\mathbf{p}_i | i = 1, \dots, n\}$) into well-beautified 3D drawings (anchor points solution P^*) by minimizing a total cost function $C_{\text{total}}(P)$ which is used to evaluate how inaccurate the input drawing P is. According to the definition of strokes S_i (s.t. $P = S_1 \circ S_2 \circ \dots \circ S_m$) mentioned previously, we consider the cost evaluations on three aspects: inter-tangents costs $C_t(i, j)$, inter-strokes costs $C_s(S_i, S_j)$, and inside-stroke costs $C_k(S_i)$. Then, the total cost $C_{\text{total}}(P)$ is:

$$C_{\text{total}}(P) = \sum_{i,j=1}^{n-1} C_t(i, j)W_t^T + \sum_{i,j=1}^m C_s(S_i, S_j)W_s^T + \sum_{i=1}^m C_k(S_i)W_k^T, \quad (1)$$

where each costs vector C_* on the equation's right side is a vector that contains several independent cost terms, W_* are the blending weights vector for each costs vector C_* . More specifically, we consider the inter-tangents costs C_t as equal length cost C_{EQ} , parallel cost C_{PAR} , and perpendicular cost C_{PER} . Mathematically, we have $C_t(i, j) = [C_{\text{EQ}}(i, j) \ C_{\text{PAR}}(i, j) \ C_{\text{PER}}(i, j)]$. For inter-strokes costs C_s , we consider the close points cost C_{CLO} , near to line cost C_{LN} , and near to curve cost C_{CUR} . Mathematically, we have $C_s(S_i, S_j) = [C_{\text{CLO}}(S_i, S_j) \ C_{\text{LN}}(S_i, S_j) \ C_{\text{CUR}}(S_i, S_j)]$. As far as inside-stroke costs C_k , we have vertical cost C_{VER} , horizontal cost C_{HOR} , ground cost C_{GN} , and coplanar cost C_{PLN} considered. Mathematically, we have $C_k(S_i) = [C_{\text{VER}}(S_i) \ C_{\text{HOR}}(S_i) \ C_{\text{GN}}(S_i) \ C_{\text{PLN}}(S_i)]$. In the following section, we will present the definition for each cost term.

Equal Length Cost. For improving the input drawing's regularity, we hope to ensure that adjacent anchor points (tangents) with similar lengths are optimized into equal lengths. We measure the lengths of each tangent in the anchor points P and compare every two tangents within the anchor points except two kinds of invalid tangents: (1) those tangents that are connecting the ending point of one stroke (or loop) and the starting point of another stroke (or loop); and (2) those tangents that are connecting two control points in a Bezier curve. If a tangent starting at anchor point \mathbf{p}_i shares the similar length with a tangent starting at another anchor point \mathbf{p}_j , then an equal length cost $C_{\text{EQ}}(i, j)$ is turned on:

$$C_{\text{EQ}}(i, j) = \left(1 - \frac{\|\mathbf{p}_{i+1} - \mathbf{p}_i\|}{\|\mathbf{p}_{j+1} - \mathbf{p}_j\|}\right)^2 \quad (2)$$

Parallel Cost. We hope to ensure that those tangents sharing similar directions are parallel to each other. In order to measure the difference between two tangent directions, we introduce the dot product of two normalized tangent vectors, which is the cosine angle of such two tangent directions. If two tangents are sharing similar directions, their dot product returns 1; otherwise 0 (perpendicular). Therefore, if the tangent at anchor point \mathbf{p}_i is likely to be parallel to the tangent at anchor point \mathbf{p}_j , then they should be regularized into parallel through the parallel cost $C_{\text{PAR}}(i, j)$:

$$C_{\text{PAR}}(i, j) = 1 - \left(\frac{\mathbf{p}_{i+1} - \mathbf{p}_i}{\|\mathbf{p}_{i+1} - \mathbf{p}_i\|} \cdot \frac{\mathbf{p}_{j+1} - \mathbf{p}_j}{\|\mathbf{p}_{j+1} - \mathbf{p}_j\|}\right)^2 \quad (3)$$

Perpendicular Cost. We hope to ensure that those tangents are likely to be perpendicular to each other to be perpendicular. Actually, perpendicular relationships commonly appear in furniture-like objects, therefore, it is particularly important for regularizing indoor wire art designs. It is easy to measure the degree to which two tangent directions are perpendicular to each other using the dot product of two normalized tangent vectors too. If two tangents are perpendicular to each other, their dot product returns 0; otherwise 1 (parallel). So, perpendicular cost $C_{PER}(i, j)$ is:

$$C_{PER}(i, j) = \left(\frac{\mathbf{p}_{i+1} - \mathbf{p}_i}{\|\mathbf{p}_{i+1} - \mathbf{p}_i\|} \cdot \frac{\mathbf{p}_{j+1} - \mathbf{p}_j}{\|\mathbf{p}_{j+1} - \mathbf{p}_j\|} \right)^2 \quad (4)$$

Close Points Cost. When considering arbitrary two strokes (or loops) S_i and S_j , there might be an anchor point \mathbf{p} in S_i and an anchor point \mathbf{q} in S_j are very close to each other. In order to ensure the connectivity of the wire art design, those very close points are supposed to be pushed towards each other to form a connection. In this case, we hope to minimize the distance between such two points by introducing the close points cost $C_{CLO}(S_i, S_j)$:

$$C_{CLO}(S_i, S_j) = \sum_{\mathbf{p} \in S_i} \sum_{\mathbf{q} \in S_j} \|\mathbf{p} - \mathbf{q}\|^2 \quad (5)$$

Near to Line Cost. When considering arbitrary two strokes (or loops) S_i and S_j , there might be an anchor point \mathbf{p} in S_i is very near to a line segment $(\mathbf{q}_k, \mathbf{q}_{k+1})$ in S_j . Noted that there must be a line to operation $(\mathbf{q}_k, \mathbf{q}_{k+1})$ in the input. Let normalized tangent direction $\mathbf{q}_k = (\mathbf{q}_{k+1} - \mathbf{q}_k) / \|\mathbf{q}_{k+1} - \mathbf{q}_k\|$. In order to ensure the connectivity of the wire art design, those points near to lines are supposed to be pushed towards the lines to form T-junctions. In this case, we hope to minimize the distance between the point and line by introducing the near to line cost $C_{LN}(S_i, S_j)$:

$$C_{LN}(S_i, S_j) = \sum_{\mathbf{p} \in S_i} \sum_{\mathbf{q}_k \in S_j} \|\mathbf{p} - \mathbf{q}_k - \mathbf{q}_k [(\mathbf{p} - \mathbf{q}_k) \cdot \mathbf{q}_k]\|^2 \quad (6)$$

Near to Curve Cost. Given two arbitrary strokes (or loops) S_i and S_j , if there is any anchor point \mathbf{p} in S_i very near to a cubic Bezier curve segment $(\mathbf{q}_k, \mathbf{q}_{k+1}, \mathbf{q}_{k+2}, \mathbf{q}_{k+3})$ in S_j , then this points is supposed to be pushed towards the curve segment to ensure the connectivity of the wire art design. In this case, we hope to minimize the min distance between the point and curve by introducing the near to curve cost $C_{CUR}(S_i, S_j)$:

$$C_{CUR}(S_i, S_j) = \sum_{\mathbf{p} \in S_i} \sum_{\mathbf{q}_k \in S_j} \min_{t \in [0,1]} \left\| \mathbf{p} - \sum_{d=0}^3 \binom{3}{d} (1-t)^{3-d} t^d \mathbf{q}_{k+d} \right\|^2 \quad (7)$$

Horizontal Cost. Unlike inter-tangent costs which compare two different tangents, also unlike inter-strokes cost which compare two different strokes (or loops), single stroke costs are extracting the geometrical regularities embedded within a single stroke. Hereby, the horizontal cost is used to regularize the stroke S_i where if there is a tangent starting at anchor point \mathbf{p}_j in S_i likely to be horizontal in space, then it should be regularized into perpendicular to the upward direction $(0, 1, 0)$. So, horizontal cost $C_{HOR}(S_i)$ is defined with dot product:

$$C_{HOR}(S_i) = \sum_{\mathbf{p}_j \in S_i} \left(\frac{\mathbf{p}_{j+1} - \mathbf{p}_j}{\|\mathbf{p}_{j+1} - \mathbf{p}_j\|} \cdot (0, 1, 0) \right)^2 \quad (8)$$

Vertical Cost. As being opposite to the horizontal cost, the vertical cost is used to regularize the stroke S_j where if there is a tangent starting at anchor point \mathbf{p}_j in S_i likely to be vertical in space, then it should be regularized into parallel to the upward direction $(0, 1, 0)$. Vertical tangents are perpendicular to the horizontal plane. So, vertical cost $C_{VER}(S_i)$ is defined with dot product:

$$C_{VER}(S_i) = \sum_{\mathbf{p}_j \in S_i} 1 - \left(\frac{\mathbf{p}_{j+1} - \mathbf{p}_j}{\|\mathbf{p}_{j+1} - \mathbf{p}_j\|} \cdot (0, 1, 0) \right)^2 \quad (9)$$

Ground Cost. In order to make the wire art design stand-able on the ground, we need to ensure that the ground points of the design are on the same plane. So, we first calculate the ground elevation as the minimum elevation of the anchor points in the drawing. Elevation of an anchor point $\mathbf{p} \in P$ can be calculated through the dot product between \mathbf{p} and the upward vector $(0, 1, 0)$. Then, during the regularization process, if any anchor point \mathbf{p}_j in stroke S_i is close enough to the ground plane then that point is supposed to be push onto the ground through the ground cost $C_{GN}(S_i)$:

$$C_{GN}(S_i) = \sum_{\mathbf{p}_j \in S_i} \left(\mathbf{p}_j \cdot (0, 1, 0) - \min_{\mathbf{q}_k \in P} [\mathbf{q}_k \cdot (0, 1, 0)] \right)^2 \quad (10)$$

Coplanar Cost. Lots of contours in 3D drawings are representing plane surfaces. Therefore, it is an important consideration to ensure that the coplanar points in a stroke (or loop) of the design are lying on the same plane. We first calculate the center point of stroke S_i as $\mathbf{c}_i = \sum_{\mathbf{q}_k \in S_i} \mathbf{q}_k / |S_i|$. Then, during the regularization process, we ensure the coplanarity of anchor points \mathbf{p}_j in stroke S_i on the same plane through the coplanar cost $C_{PLN}(S_i)$:

$$C_{PLN}(S_i) = \sum_{\mathbf{p}_j \in S_i} \left((\mathbf{p}_j - \mathbf{c}_i) \cdot \frac{\sum_{\mathbf{q}_k \in S_i} (\mathbf{q}_{k+1} - \mathbf{c}_i) \times (\mathbf{q}_k - \mathbf{c}_i)}{\|\sum_{\mathbf{q}_k \in S_i} (\mathbf{q}_{k+1} - \mathbf{c}_i) \times (\mathbf{q}_k - \mathbf{c}_i)\|} \right)^2 \quad (11)$$

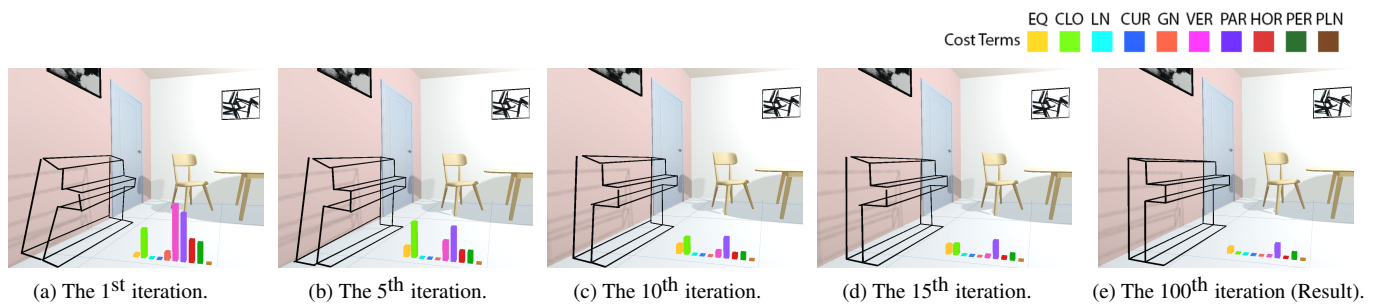


Figure 3: An example of the optimization process. Black polylines are the wire art design from the user. In Figure (a), the initial design is trying to mimic the shape of a piano. However, the shape is not well-drawn. Throughout the iterations of optimization, anchor points in the wire art design are updated into the new ones that are tending to decrease the total cost function. In order to show the changes of those cost terms, we plotted those cost values into a 3D histogram which consists of several 3D columns with different colors. Every single column's height is corresponding to the value of a cost term. Figure (b-d) shows the intermediate results generated through the optimization process. During the process, as the cost values are decreasing, the quality of the drawing is improving. Figure (e) shows the result of the final beautified wire art design. As we can see, the result (e) looks more like a "real" piano compared to the input (a).

3.3. Optimization

Given the mid-air drawings in Pen2VR as vector graphics, the input drawings (anchor points P) can be beautified and regularized by solving an optimization problem through three steps: (1) Extracting the regularities of user's drawings (anchor points P) as initial input; (2) Evaluating the total cost function of the drawings $C_{\text{total}}(P)$ described in Equation 1 according to those regularities; and (3) Finding gradient of the cost functions $\nabla C_{\text{total}}(P)$ and update those anchor points in the drawings according to the gradients; After iterating between Step (2) and (3), until the maximum number of iterations is reached, the wire art design is beautified.

During the optimizations, we use L-BFGS (Limited-memory BFGS) [SAG*] which is an optimization algorithm in the family of quasi-Newton methods that approximates the Broyden-Fletcher-Goldfarb-Shannon algorithm (BFGS) using a limited amount of computer memory. L-BFGS can automatically move the anchor points of the original input and try to find the gradient that can minimize the total cost function. During the L-BFGS optimization process, for each iteration step, there is an approximate Hessian matrix and a search direction solved from the gradient. Iteration step size is solved through a one-dimensional optimization (line search), then the updates are the search direction multiplied by the step size. These optimization steps are applied several times until a local minimal solution is found or the maximum number of iterations is reached.

As shown in Figure 3, a wire artwork designed in Pen2VR is beautified through the L-BFGS optimizer. The user draws a piano-like wire artwork in mid-air, after multiple clicks through a VR controller, the initial work is completed in a virtual environment. During the beautification process, the strokes in the artwork are corrected step by step through the L-BFGS optimization algorithm without any user interactions. As we can see from the subfigures, the optimization algorithm converges pretty fast and get quite satisfying result within the first 50 iterations. In this example, we set the maximum number of iterations as 100 (5-10 sec to run on a PC) and the result is visually acceptable in the end. The cost values for those cost terms are plotted in a 3D histogram with different colors.

4. Experimental Design

4.1. Implementation

We have implemented the proposed Pen2VR interactive user interface using Unity 3D with the 2019 version. We have implemented this VR interactive interface using the Steam VR 2.0 plugin. Our proposed L-BFGS optimization algorithm is implemented in MinGW C++ using the StanMath and Eigen library. The hardware configurations contain Intel Core i5 CPU, 32GB DDR4 RAM, and NVIDIA GeForce GTX 1650 4GB GDDR6 Graphics Card. The VR program is configured on Oculus Quest 2.0 version as shown in the above figure.



4.2. Wire Art Design

In order to test our proposed Pen2VR interface in wire art design hands-on, we invited the professional user who has a solid background in Photoshop CC. This user is very familiar with the pen tool interface in Photoshop. Therefore, when the user is put into a VR platform to design wire art in mid-air, it takes a pretty short amount of time to get familiar with Pen2VR as Pen2VR is a direct extension from a 2D pen tool into a 3D mid-air pen tool. We have assigned 10 drawing tasks for the user to design, including a chair, a piano, a teapot, a vase, a lamp, a cup, a shelf, a stands, a plane, and a car. For each task, the user is not allowed to see any picture during the design. Also, the user is not allowed to correct the design during the drawing process. The user is allowed to redraw only if the drawing's quality is too "poor". The reason that the user is not allowed to correct the drawing is to test the ability of our optimization approach to whether can correct the "bad" drawings significantly. If we allow the user to input high-quality drawings, then the efficacy of our interface will not be validated sufficiently. For each drawing task, it takes about 5-15 mins for the user to finish, which depends on how complex the proposed shape is. After the input is recorded, we apply the optimization process to beautify the user's drawing. Results are presented in next section.

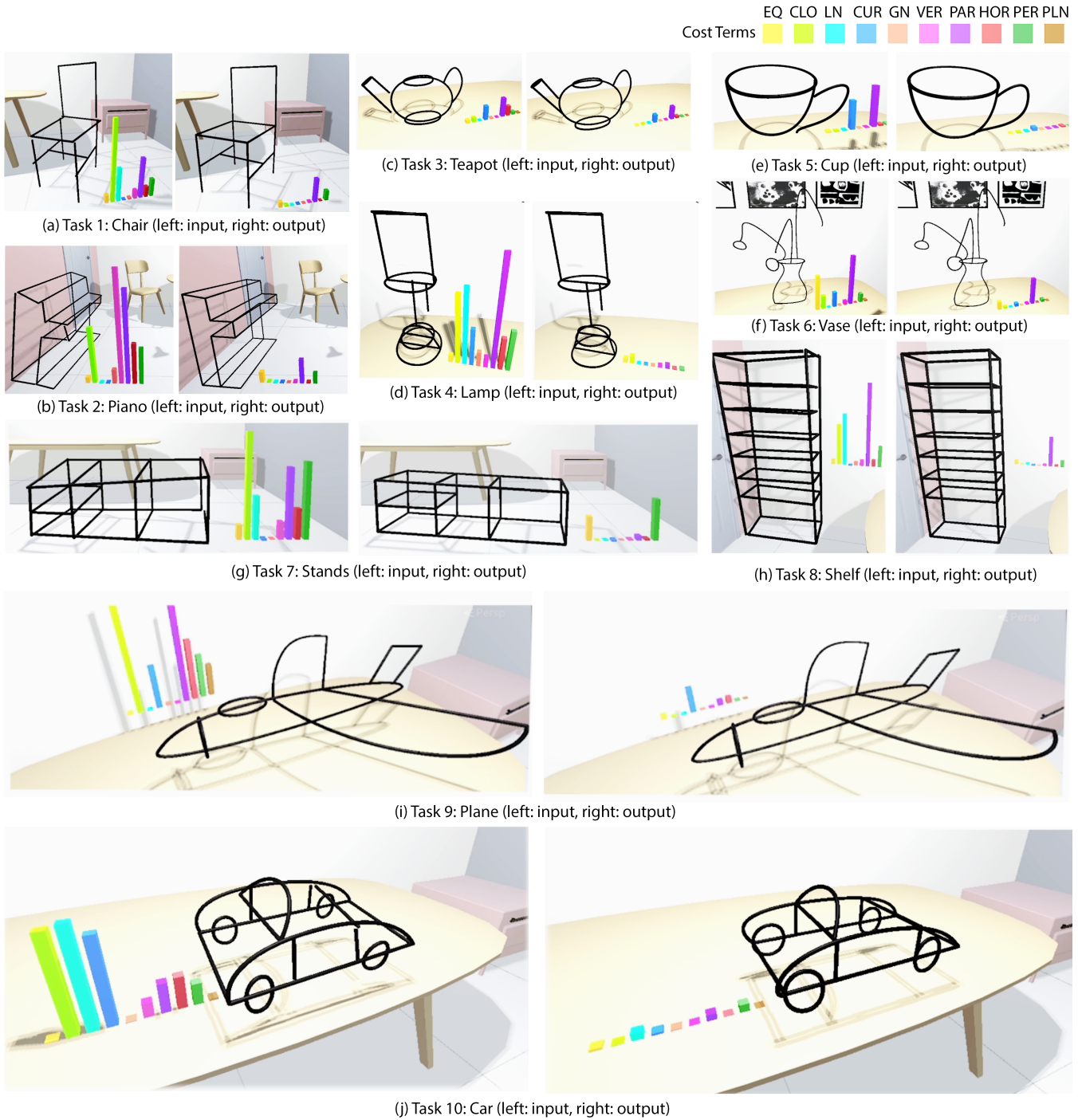


Figure 4: Experimental Results. We have assigned 10 wire art design tasks for the user who has professional skills in Photoshop digital art design. The user's design for those tasks are plotted in the subfigure (a)-(j). For each of user's initial design, we apply the optimization process to beautify the drawing. The maximum number of iterations is set to 100. Initial inputs are plotted in the left subfigures and the beautified output are plotted in the right subfigures. Cost values are plotted with 3D histogram in different colors labeled in the legend.

5. Results and Discussions

As shown in the figure 4, 10 different wire art design tasks are assigned to the user with professional skills in Photoshop. As we can see most of the initial inputs are "poor" and their optimized outputs look "better". However, due to the limitation of 2D rendering, dif-

ferences between input and output in some examples are not that clear. Therefore, we render some examples with three-view drawing in Figure 5 through the Windows 10's 3D viewer app. We select three polylines-based designs including the chair, piano, and shelf. And three curves-based designs including the lamp, car, and plane.

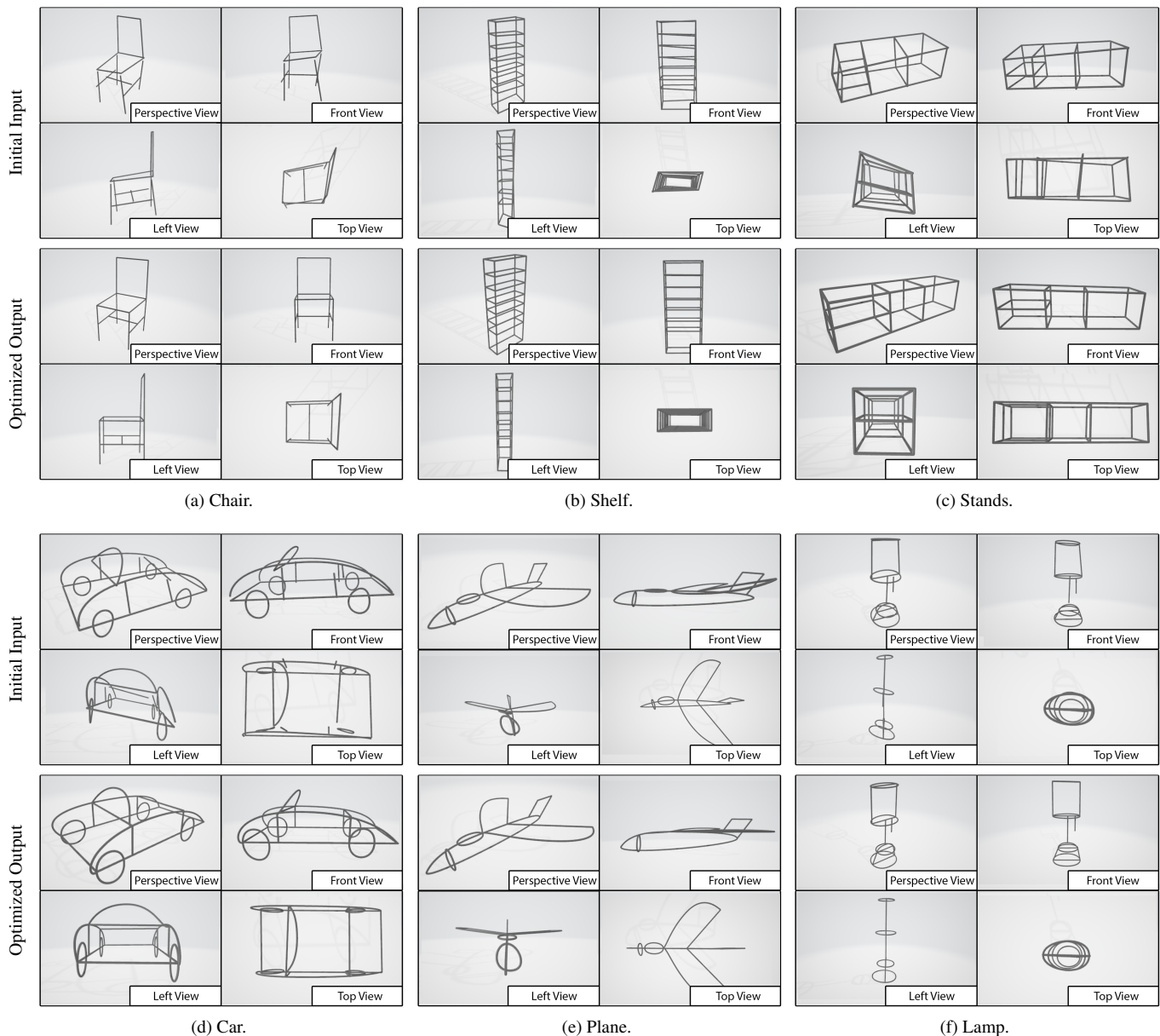


Figure 5: Three-view drawings. This figure shows the three-view drawings of six examples including (a) chair, (b) shelf, (c) stands, (d) car, (e) plane, and (f) lamp. For each example, the first row shows the users' raw input and the second row shows the output from the optimizer.

Among these examples, a perspective view, front view, left view and top view drawings are presented. In order to test the drawings' improvements made by the optimizer that we propose, we conducted two statistical tests: numerical test and perceptual test. In the numerical test, we applied the ANOVA test on the values of each cost term. We compare the cost values before the optimization and after, then check whether there are significant improvements for these 10 examples. In the perceptual test, we ask 30 participants to score the design perceptually. For each example, they give separate scores for the user's initial input and beautified output. Then according to the perceptual scores gathered from the participants' answers, we apply the ANOVA test to tell whether there are significant improvements between initial designs and optimized designs.

Numerical Test. According to the cost functions we defined in Section 3.2, we evaluate how much extent to which an input of wire art design is inaccurate. And through a series of geometrical calculations, we got the values that are called costs. In the numerical test, we want to prove that there are significant improvements in the numerical evaluation (cost values) for the user's drawings after they are optimized. According to these 10 examples we have collected from the user, we compare the average cost values for each cost term at the beginning and at the end of the optimization. They are box-plotted in Figure 6. Each color is corresponding to one cost term. The left subfigure (a) shows the costs values before the optimization, while the right subfigure (b) shows those after optimizations. As we can see there are improvements in overall.

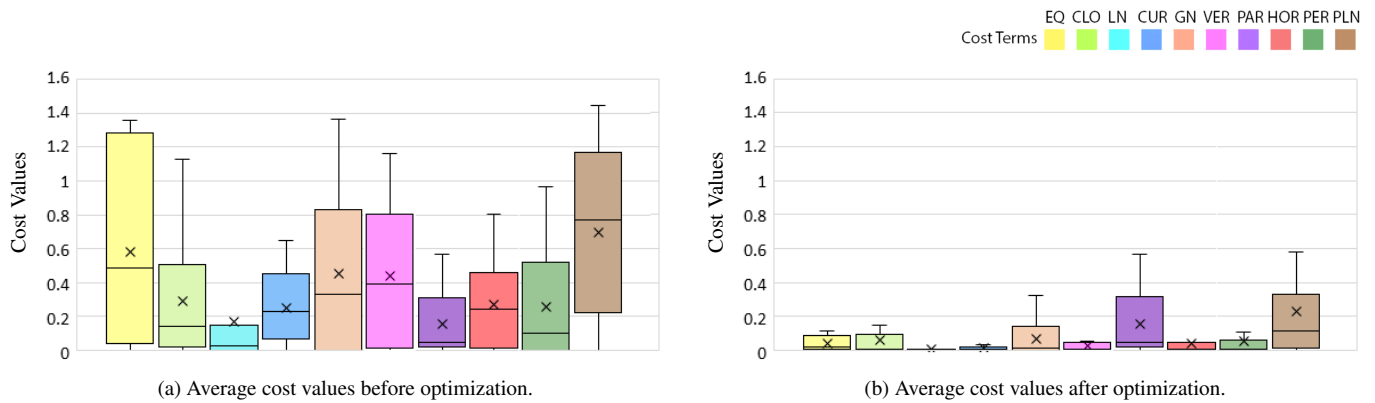


Figure 6: Cost Values. This figure shows the box plots for the cost values of 10 wire art design examples proposed in the experiment. Subfigure (a) plots the average cost values before optimization. Subfigure (b) plots the average cost values after optimization. Means of those cost values plotted in this figure as 'X' marks in the boxes are averaged from the 10 examples according to each type of cost term. Different cost terms (boxes) are plotted in different colors which are labeled according to the legend on the upright corner.

After we have calculated the average cost values for these 10 examples, we got 20 numbers: among which 10 are those means ('X' marks) in the left subfigure (a) and another 10 are means ('X' marks) in the right subfigure (b). Then we have applied a single factor ANOVA test on these 20 numbers to prove there are significant improvements. According to the ANOVA test, we have an average cost value before the optimization is 0.36, and an average cost value after the optimization is 0.068. We set $\alpha = 0.05$ and get $P_{\text{value}} = 0.002 < 0.05$, therefore, we have 95% confidence to prove that the costs values have significantly decreased. Therefore, we prove that our optimization approach has significantly improved the quality of the wire art designs from a statistical point of view.

Perceptual Test. The numerical test is not enough to prove that our optimizer has improved the design from a perceptual view. Therefore, we collected perceptual data from 30 participants with different majors and ages between 20-30. For each participant, we ask them to score those 20 wire art designs shown in Figure 4. Score values are set between 1 to 5, where 1 is a very bad design and 5 is a very good design. Among those 20 questions, there are 10 for original designs and 10 for optimized designs. Those designs appear in the questions in random order. For each question, participants are asked to open the obj file of the 3D model of the wire art design through Windows 3D viewer. After we collected the data, we classified the scores corresponding to which piece of art it belongs to. As shown in Figure 7, we plot error bar charts for perceptual scores. Scores for 10 original wire artworks are plotted in the top subfigure (a), scores for 10 optimized wire artworks are plotted in the bottom subfigure (b). Overall speaking, we see a pattern that most of the optimized wire arts are scored with higher values than those original designs. But some examples do not show such significant improvements such as the vase, teapot, and cup, etc.

In order to statistically show whether there are significant improvements in the perceptual scores after being optimized, we calculated the average score for these 10 examples for before and after optimization respectively, we got 20 numbers in total. Then we have applied a single factor ANOVA test on these 20 numbers to prove there are significant improvements. According to the ANOVA test, we have an average perceptual score before the op-

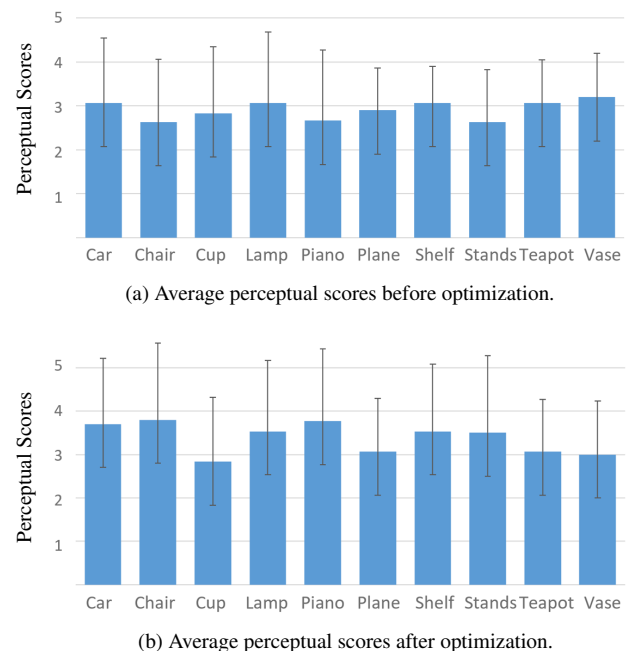


Figure 7: Perceptual Scores. This figure shows the error bar charts for the perceptual scores of 10 wire art designs. Subfigure (a) plots the perceptual scores before optimization. Subfigure (b) plots the perceptual scores after optimization. Different columns represented the perceptual scores for different wire art design examples.

timization is 2.91, and an average perceptual score after the optimization is 3.38. We set $\alpha = 0.05$ and get $P_{\text{value}} = 0.0035 < 0.05$, therefore, we have 95% confidence to prove that the perceptual scores have significantly increased. Therefore, we prove that our optimization approach has significantly improved the quality of the wire art designs from a perceptual aspect. As the conclusion of the perceptual test results matches with the conclusion of numerical test results, therefore, we have proved that the cost terms that we have proposed for optimizations are reasonable evaluations for measuring the extent to which a wire art design looks "good".

6. Conclusions and Future Work

In this paper, we present Pen2VR: a smart pen tool interface for 3D drawing wire art design in VR. As a natural extension from 2D pen tool interfaces in Photoshop, we have implemented a 3D version of the pen tool interface that allows users to create their 3D drawings in mid-air via virtual reality. Besides, we propose several cost terms to evaluate how much extent to which an input of wire art design is inaccurate. Then, according to those cost terms, we implemented an L-BFGS optimizer to automatically fine-tune the user's 3D initial input into well-beautified 3D drawings so that the created wire artworks are stable, stand-able, and fabricable. During the experiment, we invited the user with strong skills in using the Photoshop pen tool and assign 10 wire art design tasks. After then, we applied the optimization process to beautify those inputs. In order to statistically show whether there are significant improvements between the raw inputs and the beautified outputs, we applied ANOVA tests both on cost values and perception scores. As shown in the results, we show a positive conclusion that with 95% confidence the optimization approach has significantly improved the quality of the wire art designs from both numerical and perceptual aspects.

As future works, we will consider more cost terms to extend our optimization approach. For example, it can consider the self-symmetry of the design, so that it ensures the symmetrical parts of the design to be symmetrical. Besides, it can consider the weight center of the design to ensure the design is physically stable when being out on the table or floor and apply physics simulation to test the stability. As for the VR pen tool interface, in the future, we will recruit more artists to test our interface and conduct a larger-scale user study to include both artists who are familiar with the pen tool interface and those who are not. We will collect more valuable feedback from the artists with different skill sets. In all, as our work is successful preliminary research, we believe our technical contributions can inspire more promising follow-up studies that are exploring smart interfaces for interactive art designs in virtual reality.

References

- [AHKG*18] ARORA R., HABIB KAZI R., GROSSMAN T., FITZMAURICE G., SINGH K.: Symbiosissketch: Combining 2d & 3d sketching for designing detailed 3d objects in situ. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (2018), pp. 1–15. 2
- [AS21] ARORA R., SINGH K.: Mid-air drawing of curves on 3d surfaces in virtual reality. *ACM Transactions on Graphics (TOG)* 40, 3 (2021), 1–17. 2
- [Bru18] BRUSH T.: Painting from a new perspective. *Vitattu* 20 (2018), 2018. 2
- [CGP16] CHANG T. P., GERARD J., PUSIC M. V.: Screen-based simulation, virtual reality, and haptic simulators. In *Comprehensive Healthcare Simulation: Pediatrics*. Springer, 2016, pp. 105–114. 1
- [DGK*20] DREY T., GUGENHEIMER J., KARLBAUER J., MILO M., RUKZIO E.: Vrsketchin: Exploring the design space of pen and tablet interaction for 3d sketching in virtual reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020), pp. 1–14. 2
- [EGS*18] EROGLU S., GEBHARDT S., SCHMITZ P., RAUSCH D., KUHLEN T. W.: Fluid sketching—immersive sketching based on fluid flow. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)* (2018), IEEE, pp. 475–482. 2
- [GJS18] GIUNCHI D., JAMES S., STEED A.: 3d sketching for interactive model retrieval in virtual reality. In *Proceedings of the Joint Symposium on Computational Aesthetics and Sketch-Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering* (2018), pp. 1–12. 2
- [HHC18] HSIAO K.-W., HUANG J.-B., CHU H.-K.: Multi-view wire art. *ACM Trans. Graph.* 37, 6 (2018), 242:1–242:11. 2
- [HLC*20] HAN S., LIU B., CABEZAS R., TWIGG C. D., ZHANG P., PETKAU J., YU T.-H., TAI C.-J., AKBAY M., WANG Z., ET AL.: Megatrack: monochrome egocentric articulated hand-tracking for virtual reality. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 87–1. 2
- [KF19] KWAN K. C., FU H.: Mobi3dsketch: 3d sketching in mobile ar. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (2019), pp. 1–11. 2
- [KJ05] KAVAKLI M., JAYARATHNA D.: Virtual hand: an interface for interactive sketching in virtual reality. In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)* (2005), vol. 1, IEEE, pp. 613–618. 2
- [KJX20] KIM W., JUNG J., XIONG S.: Vrmouse: Mouse emulation with the vr controller for 2d selection in vr. In *International Conference on Applied Human Factors and Ergonomics* (2020), Springer, pp. 663–670. 2
- [LCL*17] LIU L., CEYLAN D., LIN C., WANG W., MITRA N. J.: Image-based reconstruction of wire art. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 63. 2
- [Liu12] LIU Q.: The virtual reality technology in art design. In *2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)* (2012), IEEE, pp. 2226–2228. 2
- [McG02] MCGUIRE B. A.: *Wire in Design: Modern Wire Art & Mixed Media*. F+W Media, Inc., 2002. 1
- [Rey97] REYNOLDS C. B.: *Library of Congress and the Interior Decorations: A Practical Guide for Visitors with Descriptions of All the Painting, Sculptures and Statues, the Wall Quotations, Floor Plans, and Sixteen Interior Views from Photographs*. Foster & Reynolds, 1897. 1
- [Rod19] RODRÍGUEZ J.: Surfacebrush: from virtual reality drawings to manifold surfaces. *OPENAIRE* (2019). 2
- [SAG*] SHTOF A., AGATHOS A., GINGOLD Y., SHAMIR A., COHEN-OR D.: Geosemantic snapping for sketch-based modeling: Optimization details. 5
- [TKNDH19] THORAVI KUMARAVEL B., NGUYEN C., DIVERDI S., HARTMANN B.: Tutorivr: A video-based tutorial system for design applications in virtual reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (2019), pp. 1–12. 2
- [WYFI19] WANG Y., YANG X., FUKUSATO T., IGARASHI T.: Computational design and fabrication of 3d wire bending art. In *SIGGRAPH Asia 2019 Posters*. 2019, pp. 1–2. 1
- [XCS*14] XU B., CHANG W., SHEFFER A., BOUSSEAU A., MCCRAE J., SINGH K.: True2form: 3d curve networks from 2d sketches via selective regularization. *ACM Transactions on Graphics* 33, 4 (2014). 2
- [YAS*21] YU E., ARORA R., STANKO T., BÆRENTZEN J. A., SINGH K., BOUSSEAU A.: Cassie: Curve and surface sketching in immersive environments. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (2021), pp. 1–14. 2
- [YXFH21] YANG Z., XU P., FU H., HUANG H.: Wireroom: model-guided explorative design of abstract wire art. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–13. 2
- [ZLDM16] ZHENG Y., LIU H., DORSEY J., MITRA N. J.: Smartcanvas: Context-inferred interpretation of sketches for preparatory design studies. In *Computer Graphics Forum* (2016), vol. 35, Wiley Online Library, pp. 37–48. 2