StripBrush: A Constraint-Relaxed 3D Brush Reduces Physical Effort and Enhances the Quality of Spatial Drawing

Enrique Rosales
University of British Columbia, Canada
Universidad Panamericana, México
albertr@cs.ubc.ca

Jafet Rodriguez
Universidad Panamericana, México
arodrig@up.edu.mx

Chrystiano Araújo
University of British Columbia, Canada
araujoc@cs.ubc.ca

Nicholas Vining
University of British Columbia, Canada
NVIDIA, Canada
nvining@cs.ubc.ca

Dongwook Yoon
University of British Columbia, Canada
yoon@cs.ubc.ca

Alla Sheffer
University of British Columbia, Canada
sheffa@cs.ubc.ca

ABSTRACT

Spatial drawing using ruled-surface brush strokes is a popular mode of content creation in immersive VR, yet little is known about the usability of existing spatial drawing interfaces or potential improvements. We address these questions in a three-phase study. (1) Our exploratory need-finding study (N=8) indicates that popular spatial brushes require users to perform large wrist motions, causing physical strain. We speculate that this is partly due to constraining users to align their 3D controllers with their intended stroke normal orientation. (2) We designed and implemented a new brush interface that significantly reduces the physical effort and wrist motion involved in VR drawing, with the additional benefit of increasing drawing accuracy. We achieve this by relaxing the normal alignment constraints, allowing users to control stroke rulings, and estimating normals from them instead. (3) Our comparative evaluation of StripBrush (N=17) against the traditional brush shows that StripBrush requires significantly less physical effort and allows users to more accurately depict their intended shapes while offering competitive ease-of-use and speed.

Author Keywords

Spatial drawing; VR drawing; 3D sketching; constraint relaxation; 3D brush design; wrist-twisting motion.

INTRODUCTION

Spatial drawing is an increasingly popular mode of content creation in immersive VR. Virtual reality device manufacturers have provided art packages for their hardware [54, 41] targeted at both amateurs and professionals, as a competitive selling point. Despite the popularity and potential of spatial sketching applications, and the recent resurgence of interest in virtual reality software and hardware at a consumer level, numerous barriers prevent true widespread commercial adoption. Three-dimensional interfaces, and virtual reality devices in general, cause new usability challenges without precedent in traditional screen-based GUIs, including "gorilla arm syndrome", and depth perception problems. However, HCI and VR literature lags behind software development in understanding the usability challenges of VR haptic interfaces specific to 3D drawing tasks, and ways of addressing them.

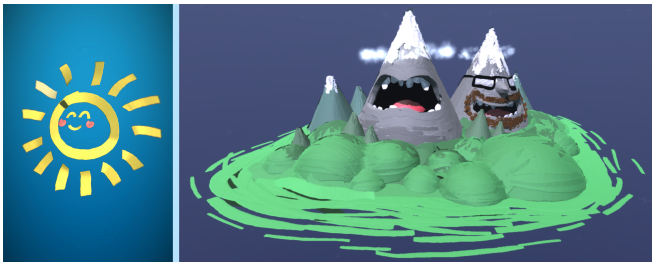


Figure 1: Representative ribbon brush drawings. Left: © Enrique Rosales, Right: © Andrew Bell.

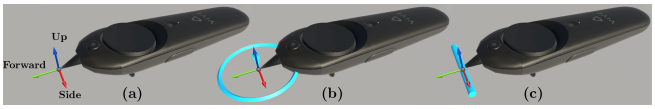


Figure 2: (a) Canonical VR controller coordinate system. (b,c) Ribbon brush interfaces: the normal brush interface uses the 'up' axis as proxy for ribbon normal, and expects users to align a circular disk with their intended ribbon tangent plane; (c) our StripBrush interface directly employs the 'side' axis as the ribbon ruling.

We study these drawing-specific usability challenges, focusing on popular ruled-surface[1], or *ribbon* based, brushes (Figure 1), and consider the mechanisms behind how users communicate their intended ribbon shapes - specifically, the ribbon's spatial orientation or normal.

One of the most popular implementations of 3D brushes - hereafter referred to as the *normal brush* - uses the orientation of

[1] A *ruled surface* is a surface that can be swept out by moving a line, or *ruling*, in space.

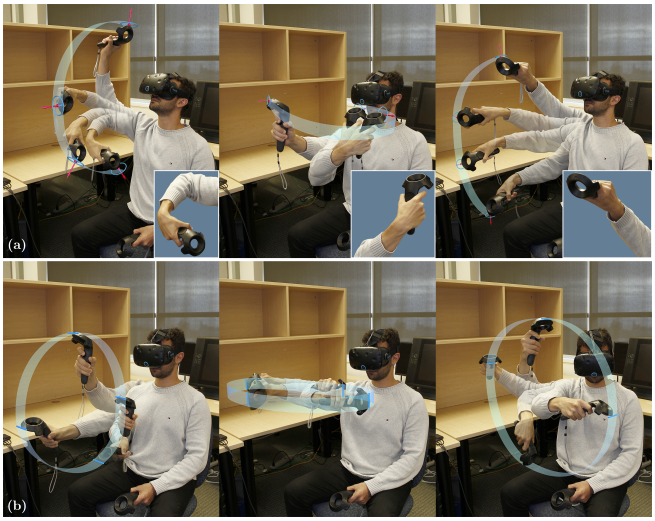


Figure 3: Using the normal brush (a) can require users to twist their hands into awkward positions; using StripBrush can (b) significantly reduce users' physical discomfort.

the user's controller at drawing time to define the orientation of the ribbon surface, as illustrated in Figure 2(b) (specifically, it uses the 'up' direction of the controller's local coordinate frame as a proxy for the user-intended ribbon normal). Our aim is to critically examine the design of this interface, with likely the largest outreach in VR drawing software. We therefore commence this study by asking the following research questions:

1. RQ1. What are the user's needs and frustrations, if any, specific to the design of the normal brush when the users draw different shapes in VR?
2. RQ2. How do we design and implement a new brush interface that can address the identified normal brush problems?
3. RQ3. To what extent can our design solution address the problems posed by the normal brush?

To answer these questions, we performed an explorative needs-finding study (N=8) in which we asked users to surface predetermined shapes that span a wide range of surface curvature patterns. We then analyzed their recorded motions and their feedback. In this study, we identified wrist-twisting motions as an important usability problem specific to spatial ribbon-stroke drawing tasks. The normal brush used by virtual reality ribbon drawing programs constrains users to align their 3D controller, and hence the hand that holds it, with their intended stroke normal direction; we found that this often forced users to twist their hands into awkward positions when drawing curved surfaces best depicted using strokes with high normal variation (Figure 3(a)). We characterized the source of this wrist-twisting problem as due to the constrained nature of the controller-to-surface mapping used by the normal brush.

We suggest constraint relaxation as a novel and effective solution to the over-constraining problem of the normal brush. After analyzing the mathematical properties of ruled surfaces and the possible linkages between controller orientation and resulting ruled surface shapes, we first replaced the over-constrained ruled surface definition employed by the normal interfaces with a more general one (specifically, we relaxed the implicit requirement, used by normal brushes, for rulings to be orthogonal to their sweeping trajectory). We then explored the core design spaces for ideating and designing new ways to control ribbon ruling orientations. We converged to a control metaphor, the *StripBrush*, that has users directly specify rulings using a natural mapping from everyday painting tools (Figure 2(c)) and significantly reduces physical discomfort (Figure 3(b)).

Our comparative evaluation of StripBrush (N=17) against the normal brush shows that StripBrush requires significantly less physical effort and allows users to more accurately depict their intended shapes while offering competitive ease-of-use and speed.

This paper makes three major contributions: (1) identification and characterization of wrist-twisting motions as a core usability challenge of ribbon-based 3D drawing interfaces, (2) the design of StripBrush and its technical implementation, and (3) evaluation results which are indicative of the efficacy of StripBrush and have design implications for further improvement.

RELATED WORK

Our work is inspired by prior research on interfaces for 3D modeling, and 2D or 3D curve drawing, and by ergonomics of VR-based interfaces in general and drawing interfaces in particular.

Shape Modeling Interfaces

Research on interfaces for modeling or creating 3D content can be traced back to the early SketchPad system of Sutherland [51]. The range of state-of-the-art modeling interfaces spans from traditional menu-based systems such as Maya [6] or 3D Studio Max [5], through sculpting based systems such as ZBrush [45], to 2D sketch-based interfaces [42, 28] and VR-based tools such as TiltBrush [54], GravitySketch [18], Oculus Medium [41], and Google Blocks [17]. While menu-based and sculpting-based systems primarily target expert modelers, many 2D-sketch-based modeling frameworks aim for a broader user base. However despite recent advances [39, 59, 24, 7, 36], their scope is still limited.

There are various genres of VR-based modeling tools. VR sculpting tools [33, 41, 50] target expert users interested in creating complex free-form shapes and aim for a more fluid interaction than their traditional counterparts. VR systems that target target low-expertise users often allow users to create simple CSG shapes by applying Boolean operations to a fixed set of initial primitives [11, 12, 17, 53]. Researchers have proposed several VR interfaces for drawing swept surfaces in 3D space [31, 49], an approach that has been popularized by the GravitySketch commercial system [18]. Using such interfaces to effectively create more general geometry requires non-trivial expertise [47].

Another option, popularized by several VR interfaces, is to use 3D sketches as a stepping stone toward generating free-form 3D shapes. Earlier VR sketch-based modeling systems target users with some design or modeling expertise, and expect

these users to draw connected 3D curve networks [15, 26, 34, 57, 48]; these networks can then be surfaced algorithmically by employing generic [14] or more targeted [10, 44] surfacing methods. State-of-the-art methods [47, 23] directly convert dense ribbon-brush VR drawings consisting of disjoint stroke collections into manifold surfaces. In this study, we seek to lower the barriers to VR-based freeform sketching tools, with a keen focus on the design of spatial brushes at the forefront of such tools intended for a wide variety of users, including both novices and experienced users.

Pen and Brush Interfaces

2D stylus-based stroke drawing interfaces are increasingly ubiquitous, migrating from high-end hardware [56] to consumer-level laptops and tablets. Raw user strokes are captured as polylines replicating the path of the tip of the user's stylus over the drawing pad or a pen-sensitive display. Software systems that use this raw input, such as Adobe Illustrator [1], typically apply local fairing [8, 37] to the recorded path data to overcome user hand-tremors and other sources of noise in the recorded path.

Researchers have experimented with a range of techniques to form three-dimensional curves [25, 19, 52, 26, 12, 3, 32] to enable direct 3D drawing. For example, Grossman et al. [19] allow users to manipulate a physical tape that represents the 3D curve; Keefe et al. [30] propose a system consisting of a virtual pencil, attached to a haptic device, and a stereoscopic display. Multiple VR systems [54, 43, 40, 18, 31, 30] allow users to draw 3D strokes that follow the path of a hand-held controller and are rendered in real time, providing users with instant visual feedback. These tools are becoming increasingly popular, as they allow users to create rich 3D content (Figure 1) and are well suited for users to express and communicate purely virtual, static visuals of their intended shapes. Many systems [43, 30] represent and render the strokes as tubular shapes centered around the captured tip paths. Others [31, 30, 54, 40, 18] generate ribbon-like strokes, defined by sweeping a ruling line along the captured controller path.

Our study focuses on these ribbon-based approaches, for the promise they show in not only creating rich 3D visuals but also serving as intermediate data for 3D surface modeling [47]. Uniquely defining the shape of a ruled surface ribbon requires specifying the orientation of the ribbon's rulings at each point along its trajectory. Existing interfaces [31, 30, 54, 40] interpret the orientation of the hand-held controller as a proxy for the surface normal, as described in Section 3 and use this normal to derive a ruling orientation. These systems convey this linkage to the users by rendering a circular disk at the tip of the controller, aligning the disk normal with the "up" orientation of the controller's coordinate system. (Figure 2, (a,b)). As our study (Section 3) shows, using this circle-based interface requires users to exert significant flexion and torque through the wrist and elbow joints to create their desired drawings; additionally, as discussed in Section 3 the planar circle may provide misleading feedback to the user as to the geometry of the output ribbon, as the actual normal may significantly diverge from the user-entered proxy. Very recently (September 2019) GravitySketch unofficially pre-released a ribbon-drawing tool

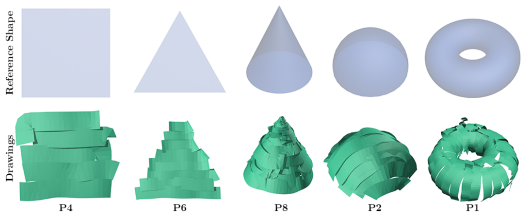


Figure 4: Formative study: (top) test shape set. (bottom) representative participant 3D drawings of these shapes.

that has users directly specify the ribbon rulings by positioning and moving a line segment rendered along the "forward" axis of the controller's coordinate system. We compare our brush design against this alternative in Section 4 and explain design rationales for our approach.

Ergonomics of generic 3D VR Interfaces

Previous work in HCI and VR literature has thoroughly documented the problem of "gorilla arm" [35], in which "mid-air interactions are prone to fatigue and lead to a feeling of heaviness in the upper limbs" [22]. Fatigue modeling [22, 27] provides tools for estimating subjective fatigue caused by gorilla arm, but primarily focuses on estimating mid-air pointing fatigue as a function of shoulder joint torque, and does not evaluate strain caused by torque applied to the wrist. Anecdotal evidence exists that overconstrained VR interfaces can cause physical strain [20, 60]. We conceptualize the wrist-twisting behavior for brush-based VR interfaces as caused by requiring users to provide *over-constrained* input that requires unnatural exertion (Section 3). This observation serves as the basis for suggesting constraint relaxation as a solution (Section 4).

Empirical Studies of VR Drawing

A recent study [4] noted that four of five expert users experienced ergonomic issues such as neck and shoulder pain when using a VR sketching system over an extended period. Several studies analyzed user accuracy when drawing in space, concluding that 3D drawing is far less accurate than its 3D counterpart [30, 46, 4]. Arora et al. [4] indicate that 3D drawings are rarely accurate due to lack of a physical drawing surface, and that visual guidelines are insufficient to improve accuracy. Our comparative study (Section 5) shows how different 3D brush designs impact user-perceived drawing accuracy and their physical strain. Barera Machuca et al. [9] suggest that user's spatial ability influences shape quality. Wiese et al. [58] demonstrate that users VR drawing skills improve rapidly as they gain more experience. Our study (Section 5) reinforces this conclusion for the scenarios tested.

FORMATIVE EVALUATION OF TRADITIONAL SPATIAL BRUSH INTERFACES

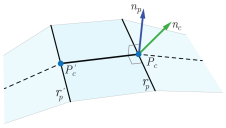
We performed a formative study to examine how users worked with the most widely-adopted implementation of spatial sketching interfaces, the *normal-based* ribbon brush, and to identify areas where users encountered challenges and might benefit from interface improvements. Specifically, we observed how non-expert users drew a set of target shapes in VR using normal brushes when instructed to surface a target

shape as cleanly as possible, and then sought feedback from the users about their experience. Our goal was to answer the following questions:

1. How challenging do users find the task of drawing using normal brushes?
2. What are the common challenges that stem from the design of normal-based VR drawing interfaces?
3. How does the target shape that the user is attempting to draw affect these challenges?

Apparatus: The Normal Brush

Normal-based ribbon drawing interfaces present the user with a disk orthogonal to the controller's 'up' direction (Figure 2,a). At drawing time, they record this direction as n_c, and the position of the controller tip as p_c.



To embed the controller's spatial path in the rendered ribbons, for each new position c_p and given the previous captured position p'_c, they compute the direction of the ruling r_p at p_c as $r_p = n_c \times (p_c - p'_c)$ (see inset). They then form a discrete ruled surface comprised of quadrilaterals along the stroke path by connecting consecutive vertices along the left and right sides of the rulings. This construction generates rulings which are orthogonal to the vectors $\overrightarrow{p'_c p_c}$ but does not guarantee alignment between ribbon and controller normals (see difference between the ribbon normal n_p at p and n_c in the inset); nor does it provide an upper bound on the angle between these input and output normals.

Methodology

The core promise of ribbon brush interfaces is their suitability for drawing free-form shapes, both closed and open, with a wide range of curvature configurations and magnitudes. Therefore, we tasked our participants with drawing a set of shapes (Figure 4, top) that covers all possible curvature configurations (planar, spherical (isotropic non-planar), parabolic, elliptical, and hyperbolic) [13] and include both open and closed surfaces:

- **Square**: Planar; $k_{min} = k_{max} = 0$, where k_{min} and k_{max} are minimal and maximal principal curvature values respectively).
- **Triangle**: Planar; $k_{min} = k_{max} = 0$.
- **Cone**: Parabolic $k_{min} = 0, k_{max} > 0$.
- **Hemisphere**: Spherical $k_{min} = k_{max} > 0$.
- **Torus**: Contains both elliptic ($k_{max} > k_{min} > 0$) and hyperbolic $k_{min} > 0, k_{max} > 0$ regions at outer and inner parts of the torus

Note that two shapes are planar (square and triangle); one is singly-curved, namely it has one non-zero, and one zero, principal curvatures (cone); and two are doubly curved, namely they have two non zero principal curvatures (hemisphere and torus). The torus is a closed shape, while the others are open.

All participants were given a brief introduction to the drawing user interface (20 minutes) and a short explanation of the test (5 minutes). We then asked them to draw the five shapes. They were asked to draw each shape over a predefined scaffold, shown as a semi-transparent surface, replicating the underlying surfaces as cleanly as possible. Rosales et al. [47] indicate that first time users require time to achieve what they consider an acceptable drawing quality, but ultimately converge to a common drawing style, depicting surfaces using partially overlapping ribbon strokes with locally similar tangents. Thus to save user time, we suggested to the participants that they follow a similar style, placing strokes side-by-side and avoiding crisscrossing strokes. The order in which shapes were shown to the participants was balanced.

After drawing each shape, participants were asked to express their level of agreement with the statement that "Drawing this surface is hard" using a 1 to 5 scale (strongly disagree-strongly agree, Likert-style questions). Finally, we conducted follow-up semi-structured interviews (15 min) with a focus on specific usability challenges they encountered, if any.

The task (interview) sessions were video- (audio-) recorded for future analysis. We manually annotated our video recordings, marking every time the users erased a previous stroke, or performed an undo operation. We counted the number of these *error corrections*, per participant and per shape. A detailed record of the answers and the error correction counts ($8 \times 5 \times 2$) is provided in the supplementary material and is summarized in Figure 5 and Table 1.

Sampling and Screening

Prior research [4] indicates that a small percentage of the population lacks perception of depth in VR, and thus faces an inherent, tool-independent, difficulty when using VR content creation tools. Since our goal is to focus on challenges presented by a specific interface, rather than globally, we identified such participants and excluded them from the study using the following test: we asked the participants to draw three vertical ribbons side-by-side and roughly at the same depth. Two participants (one male and one female) were unable to place the ribbons at the same depth by far (depth variation was larger than three times the ribbon width) even after several attempts, and were excluded. Through convenience sampling, we recruited eight participants (6 male, 2 female, all passed the depth-perception screening) consisting of 6 CS graduate students and 2 graduate students in electrical engineering. One had previous experience with VR drawing.

Results and Discussion

Overall, the participant-generated drawings captured the core geometric characteristics of the target shapes and adhered to the drawing instructions (examples in Figure 4). Here we present our core findings:

Drawing Curved Ribbons Was Challenging

Analysis of the survey responses with a within-population paired t-test [29] and qualitative comments indicated that the level of difficulty when surfacing a shape is correlated with the shape's complexity as captured by the curvature variation across it (square/triangle vs cone: mean diff -2.062, std dev:

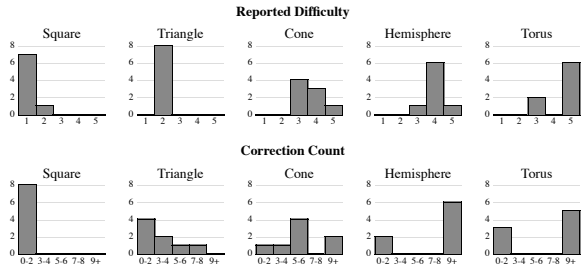


Figure 5: Reported difficulty (top) and correction counts (bottom) per shape.

Shape	Difficulty avg/median	Corrections avg/median
Square	1.125 /1	0.750 / 0.5
Triangle	2.000 / 2	2.625 / 2.0
Cone	3.625 / 3.5	11.125 / 5.5
Hemisphere	4.000 / 4	12.625 / 13.5
Torus	4.500 / 5	21.0 / 9.0

Table 1: Summary of quantitative findings (formative evaluation): Left to right: shape, average and median difficulty score, average and median error correction count. Shapes ordered top to bottom based on average difficulty.

0.929, t = -8.883, $p < 0.001$; cone vs torus/hemisphere: mean diff -0.625; std.dev 1.258; t = -1.987; p = 0.0033). Participants ranked the planar shapes as easiest to draw, and performed the least amount of corrections when drawing those; and rank the doubly-curved shapes as the most difficult and performed the most error corrections when surfacing them (Table 1 and supplementary material). From a drawing perspective, the core difference between these shapes is that while the square and the triangle can be drawn using planar ribbons, the hemisphere and the torus require using ribbons with non-zero curvature. Notably the cone can be drawn using either curved or planar ribbons (the latter choice requires the ribbons to follow the $k_{min} = 0$ direction). Our interviews confirmed that drawing difficulty is indeed linked with the need to draw high curvature ribbons when depicting these shapes. Participants' comments supports our interpretation: "I need to rotate them [the ribbons] and align the ribbons to actually make it curved (P1)"; "When you draw the hole [of the torus], it's kind of, like, a little hard to determine how does the surface should go (P1)", "The other problem with the cone is that the surface is curved and the brush is not, the easiest way is to follow the straight lines [the min principal curvature] as oppose to the curve lines (P3)."

We conclude that the difficulty stems from the need to draw curved ribbons which exhibit high normal variation.

The Wrist-Twisting Problem

Analyzing the annotated video footage as well as qualitative comments gave us a deeper insight as to the roots of the observed curved ribbon drawing difficulty. Reviewing the video, we identified critical incidents where users encountered visible difficulty when trying to form long strokes with high normal variation. Specifically, when changing the surface normal by 90° or more from an initial orientation, they bent and rotated their wrists in unnatural directions (see Figure 3, (a) for an illustration of a typical scenario).

These observations are supported by user interview feedback. Users noted: (when describing drawing the hemisphere) "The problem is that, when I am drawing, if I would do this [aligning the hand naturally] you see my ribbon is in a completely wrong direction. If my ribbon was pointed flat, [aligning the ribbon horizontally] then the sphere would be a lot easier to trace (P3)."; "The problem with the cone is that the surface is curved and the brush is not, the easiest way is to follow the straight lines [the min principal curvature] as oppose to the curve lines P3."; (describing the torus) "It has an inner surface, drawing the inner surface is very hard, it is a surface that is constantly curved, so it is difficult to follow the curvature twisting your wrist (P7).".

We further hypothesize that this wrist twisting behavior occurs due to the constraining nature of the current ribbon orientation control, where users are effectively required to align the back of their hands with the surface tangent plane. For a fixed tangent plane, e.g. for each of the major planes defined with respect to the user's orientation (Figure 6, (a)), one can find a comfortable (straight wrist, low elbow) drawing pose. Drawing curved ribbons however requires continuously changing the hand orientation to correctly depict the intended ribbon normal. This task, as illustrated in Figure 3(a), is likely to require users to gradually bend or twist their wrists into unnatural poses.

We note that there is a unique mapping between the controller's orientation and the user's hand orientation due to fixed button positions on the haptic device: the user cannot simply rotate the current controllers for more comfort, and can only rotate the controller by rotating their hand. Wrist flexion and extension, in which the wrist is bent up and down from a neutral pose, is a reasonably comfortable action. However, it can only adjust the hand orientation by one Euler angle, specifically pitch. The other angle we are interested in, yaw, can only be adjusted by twisting the wrist or bending it sideways. Both motions permit a smaller range of angular movement, and are controlled by handle muscles which are not usually as well-developed. Thus the wrist-twisting behavior identified in our review of video footage cannot be alleviated by users simply choosing a different orientation strategy.

Based on this finding, we developed a new brush-based interface that attempted to minimize wrist exertion, described in the following section.

STRIPBRUSH DESIGN

Our goal is to find a new control scheme for ribbon drawing which minimizes the amount of unnatural arm and wrist twisting that users need to perform when drawing curved ribbons, without making drawing zero-curvature (flat) ribbons much harder. Our first insight, derived from the exploratory study video footage, is that users can effectively draw ruled ribbons without directly specifying a desired ribbon normal. We further note that the ribbons formed using the normal brush are constructed so that the ribbon rulings are orthogonal to their spatial trajectory. While this property may sometimes be ad-

Figure 6: (a) Mapping the three major drawing planes (front facing, side facing and horizontal) to corresponding hand-held controller orientations using normal brush. (b-d) Wrist twist minimizing hand and controller orientations necessary to align a local controller axis (b - forward, c - up, d - side) with each of the major global axes (up, side, front).

vantageous - for instance, it ensures that, for a fixed ruling length, the ribbon width remains constant - we hypothesize that it can be sacrificed to improve usability without reducing drawing accuracy. Consequently, our key idea is to provide the user with an interface based on an actual ruling instead of a tangential disc, and to allow the angle between these rulings and the controller path to change freely.

We therefore require a way for users to directly specify the ruling directions at each point along the path. We use one of the primary axes of the controller coordinate system (Figure 2) as a ruling direction, since the mental mapping between those axes and the user's controller orientation, and consequently the user's hand gestures, is self-evident. The next important question is which of the three possible axes to use. Our experiments, one of which is shown in Figure 3, bottom, indicate that using any of these axes as a ruling, and forming ribbons by sweeping this ruling by moving the controller along a desired path, drastically reduces the amount of wrist twisting compared to the normal-based drawing scheme. Our choice of axis was therefore made based on two additional design goals. First, we wish to make sure that the drawing process is as easy as possible; second, we wish for an interface that users can easily understand. Figure 6, (b-d) illustrates wrist-effort minimizing grip poses for drawing rulings aligned with the major world directions using these three options. The "side" and "up" orientations require minimal wrist twisting, and allow the users to hold the controller in a *power* grip (as defined by [38]) for all major ruling orientations (Figure 6, (c,d)). The power grip distributes the controller's weight along all fingers, making the effort of holding a heavy object easier. In contrast, the "forward" axis requires users to switch to a different grip to form vertical rulings, one in which the controller is held vertically; this grip concentrates the object weight on just a couple of fingers, making prolonged drawing more tiring. Following the second argument, we note that the drawing process we use is conceptually similar to wall or other surface painting (a connection noted by [47] who used the "fence painting" metaphor to describe the process that users employ when drawing shapes). Standard paint brushes and rollers are designed for, and held in, the same way as our controller would be when the "side" axis is used as a ruling (Figures 2, (a) and 6, (d)). We therefore expect the postive transfer would contribute to the learnability of a sideways brush better than the alternatives.

Based on these considerations we converged to the choice of the "side" axis, parallel to the controller's top and orthogonal to its handle. Extending the analogy with a paint brush or roller, we center the ruling at the controller's tip, making it symmetric with respect to the "forward" axis. Moving the controller using our scheme generates new ribbons that look exactly like paint strips created by moving a roller along the controller path. Controlling the orientation of the ribbons no longer requires wrist twisting motions; instead, the orientation is determined by a rolling gesture, driven by the shoulder and forearm. This alleviates pressure on the wrist and results in natural movements. We note that the new pre-release of GravitySketch uses the "forward" axis as their mapping choice. Given an obvious lack of documentation we cannot guess the reasoning behind it; we discarded this option for the reasons outlined above.

Technical Implementation

We implemented StripBrush as a Unity application, using the OpenVR SDK and the SteamVR Unity plugin [55]. The user interacts with StripBrush using two controllers, one in the dominant hand and one in the non-dominant hand. The dominant hand performs drawing actions; the non-dominant hand controls additional user interface options, such as undo and redo functionality, mimicking the TiltBrush interface. We also provide users with "draw" and an "erase" modes.

To compute new ribbon points, we sample the dominant hand controller position when the "draw" trigger is engaged, and after the controller has moved an ε-distance away from the last ribbon endpoint; we set ε to empirically match observed behaviour in other packages. We place ruling endpoints at a distances of half a ribbon width away from the controller tip along its "side" axis. We then connect consecutive endpoints as discussed in Section 3 to obtain a ruled ribbon mesh.

EVALUATION METHODS

We performed a comparative evaluation of StripBrush against a normal brush implementation (the baseline) using a 2 (Tools) by 8 (Shapes) within-subject factorial design. We asked participants to surface eight different shapes using one drawing tool first, and then the same eight shapes using the other drawing tool. We split our participants into two groups: the first group used StripBrush first, then the baseline; the second group used the baseline brush first, then StripBrush. The order in which participants drew the shapes was randomized.

Two of the shapes were planar (square, circle); two contained singly curved surfaces (cone, cylinder); four were doubly curved (ellipsoid, hyperbolic paraboloid (commonly known as

a 'saddle'), sphere and torus.) Three (square, circle, saddle) had open boundaries, the rest were closed; two (cylinder, cone) had sharp features; the rest are smooth. These shapes cover the full spectrum of curvature types.

Our goals were to compare workload (measured using the NASA TLX questionnaire), user perceived accuracy, usability (System Usability Scale, SUS), and user preference for both tools.

Apparatuses and Study Setup

We implemented StripBrush as described in the previous section. To minimize the impact of external factor we re-implemented the normal brush using an identical interface. The only difference between the two interfaces was the brush tip interface (Figure 2): while StripBrush employed the interface in Section 4, the normal brush used the disk, or circle based interface described in Section 3. During the study, to avoid divulging implementation details, we referred to the two interfaces using their respective signifiers, as *LINE* and *CIRCLE* tools respectively. Our study was performed using the HTC Vive VR headset, working inside a drawing area of approximately 3 meters by 3 meters. Participants remained seated during the VR drawing experiments, in a chair placed in the middle of the drawing area.

Participants

We conducted a pilot study with one participant, and conducted the final study reported on below with 17 other participants with different academic backgrounds. Six participants were female, and eleven were male. All participants were between 23 and 39 years old. Seven participants had a computer science background, four had an engineering background, and three had a background in visual arts; the rest had backgrounds in dentistry, dance, and business. Four participants had no previous experience using VR. The rest of the participants had less than six months of experience; four of those participants had some previous experience drawing in VR using TiltBrush [54]. Eight participants used StripBrush first, and nine used the normal brush first.

Procedures

Participants were introduced to the features of our VR brush implementation, including a quick introduction on how the LINE (StripBrush) and the CIRCLE (baseline) brush tools worked. We then let participants use the system to get used to the navigation tools and basic drawing, and performed an initial assessment of their depth perception by asking them to perform a simple tasks of drawing three vertical ribbons at the same depth. We then asked participants to perform three more practice tasks, using the undo, delete, and redo features. Again, all participants were able to perform this exercises correctly.

Once participants felt comfortable using the system, we asked them to perform the main study task inside the VR environment. We showed participants a 2D reference image of a surface inscribed in a bounding box, and presented them in the virtual reality space with a three-dimensional bounding box with the same proportions of the one on the image (Figure 7, right). We then asked the participants to depict the reference

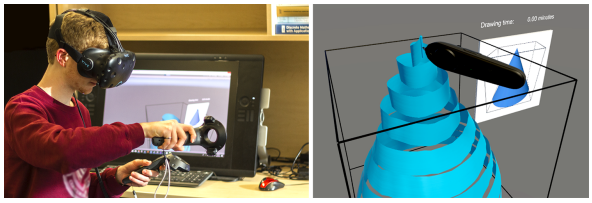


Figure 7: Comparative study setup.

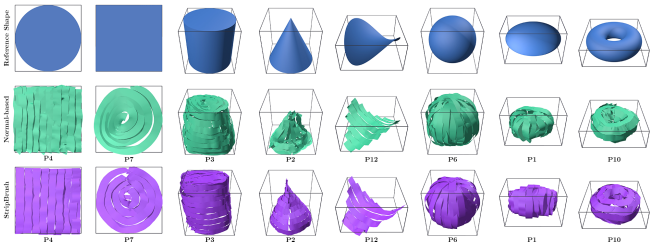


Figure 8: Reference shapes used in the study and representative pairs of participant drawings created using normal brush (green) and StripBrush (purple).

surface as accurately, or as cleanly, as possible. Participants were shown a timer bar set to 3 minutes; we explained that this was a typical time for drawing these surfaces, but that they should take more or less time as needed. We again emphasized that the goal was to produce accurate descriptions of the reference shapes.

After drawing each shape, we asked participants to verbally answer their level of agreement (using a 5 point Likert scale) with the following sentence: "drawing this shape was easy with the [LINE/CIRCLE] tool". We then confirmed their response by reading their agreement level from the Likert scale. The recording was done orally since the participants were wearing the VR headsets at that point.

We repeated this task for each of the eight different shapes using the first tool; after that, we asked participants to take off the VR headset and answer (on a laptop) the weighted NASA TLX workload survey [21], which assessed their perceived workload when using the tool, and a System Usability Scale (SUS) survey [2] which measured users assessment of the tool's usability. We then repeated the eight drawing tasks and collected similar survey responses for the second tool. Participants answered a pre-task demographic survey and a post-task survey on their experience during the tasks. After completing both sets of tasks, we conducted follow-up interviews with the participants focusing on usability, and asked them to elaborate on their answers on the usability survey and compare the two tools they used from a usability perspective. During the task sessions, our software counted the number of correction (undo/redo/delete) operations and runtime, and saved the completed drawings.

RESULTS

We summarize the study results below. Example user drawings created using both tools are shown in Figure 8. All results and study data are provided as supplementary material. In the data analysis below, we recall that all quantitative evaluations we performed are within-group; in other words, the

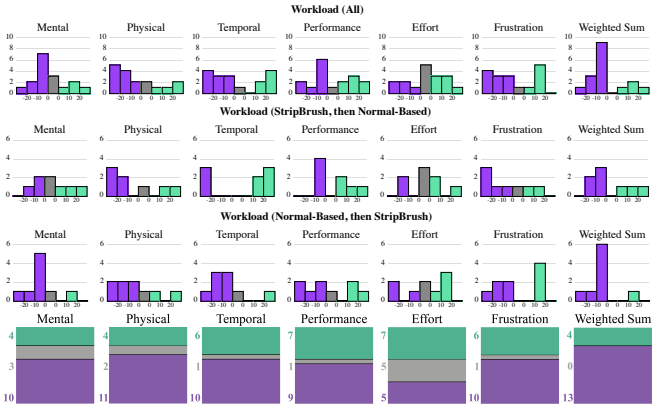


Figure 9: Summary of workload TLX feedback. All values shown are differences between corresponding values for StripBrush and normal brush interfaces. Lower (negative) numbers indicate better performance by StripBrush, higher ones indicate better performance by the alternative. (top) All participants; (row 2) participants who used the StripBrush tool first; (row 3) those who used the normal brush first. Shading (purple vs green) highlights the halfspaces which correspond to better performance of StripBrush and the normal brush respectively. (bottom row) Participant preference summary: (purple) prefer StripBrush, (green) prefer normal brush, (grey) neutral.

information we need to assess is the difference between the quantitative metrics provided for the two tools by the same user (SUS,TLX), and for questions and propertied measured per drawn shape, the difference between the corresponding quantities for the same user and for the same reference shape. We visualize these differences in the figures below, and use a paired t-test [29] when comparing samples from the normal brush and StripBrush populations when appropriate. As is common in these cases, we report the mean of differences and the standard error of mean differences only.

Analysis of our results reveals strong statistical evidence that StripBrush is less physically demanding than the normal brush, allows participants to describe shapes more accurately than the normal brush, and is rated by participants as significantly more usable overall as well. Users found StripBrush to be less frustrating than normal brush, and also made fewer corrections when using it. At the same time, we observe that there is no statistically significant change between StripBrush's and the normal brush's performance on a range of other comparative measures. From this, we conclude that StripBrush offers a clear improvement on normal brush interfaces in terms of physical exertion, accuracy, frustration and usability, with no significant downsides.

StripBrush is Less Physically Demanding Than Baseline

We summarize our workload (NASA-TLX) study findings in Figure 9. Using the NASA-TLX scale, participants deemed StripBrush to be less physically demanding than the traditional normal brush (t=-1.921; p=0.036, diff=-13.529, std=29.035; 95% confidence interval: (-28.458, 1.399)). We therefore find strong evidence that our system is less physically demanding than the normal brush approach. We analyzed the impact of

Figure 10: Summary of usability (SUS) feedback. All values shown are differences between corresponding values for StripBrush and normal brushes. Higher (positive) numbers indicate better performance by StripBrush: (rows 1-2) all participants; (rows 3-4) participants who used the StripBrush tool first; (rows 5-6) those who used the normal-based tool first. Last row shows aggregate preferences.

tool access order on the perceived workload by separating the findings for the group that used StripBrush first (Figure 9, row 2) and for those who used it second (Figure 9, row 3). In general, one may expect workload to reduce as participants gain experience. We note that the preference for StripBrush over normal brush among participants who used StripBrush as their second tool (row 3) is stronger across all workload indicators; however, even among the first group we observe a clear difference in physical workload perception, with five participants scoring StripBrush as being less physically demanding, two scoring the normal brush as such, and one scoring them as equal (for the second group the split is six to two with one neutral score).

We compare the overall participant perceived usability of StripBrush and normal brush interfaces using SUS (Figure 10). Based on our collected data, specifically the combined usability score (last column), participants find StripBrush more usable than the normal brush (t=3.427; p=0.002, diff=13.882 std=16.484, confidence interval: (5.407, 22.358)). We conclude that there is strong statistical evidence that our interface is significantly more usable than normal brush interfaces. As with workload, the preference is stronger in the group that used StripBrush second (Figure 10, rows 5-6) with 8 partici-

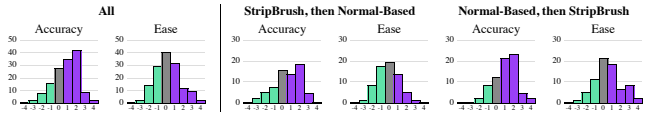


Figure 11: Feedback collected during user drawing sessions on a per-participant, per shape basis. The values shown are differences between corresponding (same participant, same shape) scores for StripBrush and normal brushes. Higher (positive) numbers indicate better performance by StripBrush, compared to the normal brush. (left) Difference in perceived usability and accuracy across **all** participants. (center, right) Breakdown of responses separating participants who used the StripBrush tool first (center) and those who used the normal brush tool first (right).

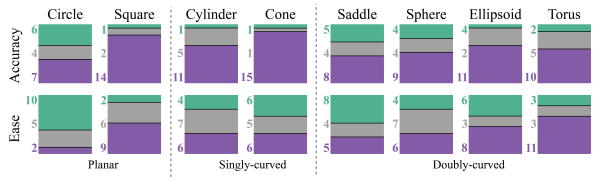
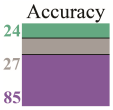


Figure 12: User perceived easiness and accuracy feedback scores per reference shape, expressed as preference values. Green: participants preferred normal brush as being easier/more accurate than StripBrush. Grey: no preference. Purple: StripBrush was preferred for ease/accuracy. We see that participants clearly prefer StripBrush for accuracy, and find it at least as equally easy to use as normal brush.

pants scoring StripBrush as more usable overall (last column), compared to one scoring the normal brush as more usable); in the group that used StripBrush first (rows 3-4) the overall preferences (last column) were evenly split.

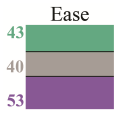
StripBrush Enhances Perceived Accuracy of Drawing

After completing all drawings, we showed participants each reference shape and the drawings they created for this shape using the two tools. We asked them to rate how strongly they agreed with the statement "This drawing accurately represents the surface on the image", using the Likert scale, for each of their drawings. The differences between the scores are summarized in Figure 11. We find that, on average, participants find results created using StripBrush significantly more accurate ($t=6.863$; $p<0.001$; diff=0.831, std=1.412; conf. interval: (0.591, 1.070)) than those created using the normal brush, and conclude that there is strong statistical evidence that users feel they are able to more accurately depict their intended shapes using StripBrush than using the normal brush. The breakdown based on tool use order (Figure 11, right) shows that both groups perceive their StripBrush drawings as more accurate, with the preference being much stronger for the group who used it second.



In addition to score differences, we computed preference data (counting the number of times each participant scored one interface higher than the other for each individual shape drawn). As summarized in the inset 85 responses indicated that StripBrush drawings were more accurate than those drawn with normal brush; 27 responses indicated that StripBrush and normal brush drawings were equally accurate; 24 responses indicated that normal brush drawings were more accurate. A per-shape breakdown of these preferences is shown in Figure 12. These numbers further reinforce the observation that using StripBrush users can more accurately draw their intended shapes than when using the normal brush.

We also asked participants to rate, for each shape, whether they thought it was easy to draw it with Stripbrush, and separately whether they thought it was easy to draw it with normal brush (these responses were collected immediately after they drew each shape). We phrased the question as "Drawing this shape was easy with the (LINE/CIRCLE) tool." Participants were asked to rate their agreement with this statement on the Likert scale. The differences between the scores are summarized in Figure 11. We find that, on average, participants find StripBrush significantly easier to use on a per-shape basis ($t=1.666$; $p=0.049$; diff=0.206, std=1.441; confidence interval: (-0.038, 0.450)) compared to normal brush. The breakdown based on tool use order (Figure 11, right) shows that while the participants who used StripBrush first see the normal brush as slightly easier to use; those who used it second strongly prefer StripBrush.



In addition to score differences, we computed preference scores. Overall, 53 responses indicated that StripBrush was easier to use than normal brush; 40 responses indicated that StripBrush and normal brush were equally easy to use; 43 responses indicated that normal brush was easier to use than Stripbrush (see inset). A per-shape breakdown of these preferences is show in Figure 12.

Participant interviews reinforce our conclusions. Oral feedback included comments such as: “Line is simple, there is too much effort for the circle, I needed to keep my hand steady, and the circle makes more noise” (P5), “I didn’t manage to draw with the circle until [having] more practice. The line is very easy to learn; with the circle it’s very easy to get the ribbon wrongly twisted because I need to rotate my wrist” (P9), “The line tool is much more controllable” (P17), and “The line tool felt more technically precise when you start and stop, you can do very clean paths” (P3). Several users commented on cases where the normal-based tool may be more suitable: “To make a planar curve, the circle is easier, that will be my only preference for that tool” (P10), and “Circle has two degrees of freedom, line tool is much easier, most of the images are easier with the line tool except for the planar circle” (P11)

These observations align with our conclusion that StripBrush is generally less taxing and is most beneficial for drawing curved shapes, the most common use scenario for such tools [16]. They also highlight the improved drawing accuracy it provides.

Performance

When directly comparing user performance using StripBrush vs. the alternative, on a per-shape, per-user basis, participants take the same amount of time on average to describe their target shapes and use on average the same number of

strokes. We conclude that StripBrush and the normal-based approach perform equally well in terms of time and ribbon count. Participants perform fewer correction operations (undo, redo, delete) when using StripBrush (Figure 13). The difference in correction count was statistically significant (diff: -2.235, std=11.368, CI (-4.163, -0.307); t=-2.293, p=0.012). We saw no perceivable difference when breaking the performance down into groups based on tool use order.

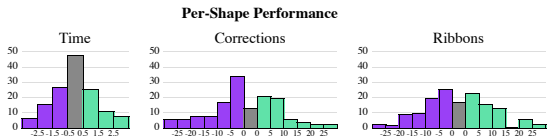


Figure 13: Performance measurements collected during study drawing sessions on a per shape basis shown as differences between corresponding values for StripBrush and normal brush. Lower (negative) numbers indicate better performance by StripBrush. Left to right: runtime (seconds), correction count, ribbon count. There difference in values when breaking the responses based on tool assessment order is negligible. Both methods perform equally well for these metrics.

DISCUSSION

We hypothesized that the design of StripBrush would reduce physical exertion, and this is clearly shown in the results. Users also found our tool easier to use than the normal-based brush, and we were surprised that users perceived the tool as being more *accurate* than the normal brush. We hypothesize that the ease-of-use of StripBrush, in addition to the reduction of physical exertion, is because it presents a familiar metaphor: users are familiar with a paint roller and know how it works, and can intuitively apply this metaphor to VR spaces. We hypothesize that the higher accuracy score is due to two factors. First, participants using a tool that they find easier are more likely to relax when using it, and hence more likely to produce a result that they perceive as accurate. Second, StripBrush grants users a finer degree of control over generated ribbons. Consequently, they are able to more precisely achieve the idea that they ideate in their head, and hence perceive the tool as more accurate.

As the target sufaces used in our evaluation cover a wide variety of curvature characteristics, the results from our controlled evaluation (Section 5) can serve as useful empirical grounds to estimate advantages and limitations of StripBrush in real drawing settings when the user draws more general shapes. As the catalogue of a popular spatial sketch sharing platform [16] indicates, artists and VR drawing hobbyists tend to draw more organic subjects that require the use of curved ribbons to successfully surface. We expect StripBrush to have context-specific advantages on these designs, rendering it more promising than what can be reported in a controlled study.

At the same time, while StripBrush performs best when drawing curved ribbons, the normal brush - or another brush entirely - may be easier to use for planar surfaces. It would therefore be useful to consider hybrid solutions where the system can adaptively transition between the two interaction modes by detecting and further predicting the curvature of the target surface intended by the user, given the drawing context. Further taking the constraint relaxation as a design space, one may explore the impact of relaxing the constraint further than what is designed for StripBrush where a pen tip can be simply represented as a *point* at the extreme end of the circle-line-dot continuum, and the ribbon orientation is derived from contextual cues.

CONCLUSIONS AND FUTURE WORK

Our paper studied the usability challenges in ribbon-based VR brush drawings. Our study concluded that users experience both difficulty and physical discomfort when drawing ribbons with large surface normal variation. We proceeded to propose an alternative brush tip interface, StripBrush, which as validated by our comparative study, reduces the physical effort required to draw 3D shapes, improves drawing accuracy and overall usability, and maintains a range of other performance indicators on par with state-of-the-art normal brush interfaces.

Our work has a number of exciting followup avenues. Designing and implementing a hybrid system like the one imagined in Section 7 is a major and promising undertaking. As a stepping stone toward this goal it would be interesting to come up with concise user guidelines for when they want to switch between tools. Finally, while our tool reduces the physical effort involved in drawing, it does not eliminate it. Further research could explore other means to further reduce this effort.

ACKNOWLEDGMENTS

We are deeply grateful to Jonathan Griffin and Nico Schertler for their help on this project. This work has been supported by NSERC and CONACYT.

APPENDIX

For completeness, we provide a short informal primer on the geometric and differential properties involved in ribbon surfaces and the shapes users create with them. Readers interested in a more rigorous presentation of these properties can refer to [13]. Given a differentiable surface S in R^2 defined as a parametric function of two parameters $S(u,v)$, the tangent plane of the surface is spanned by the partial derivative vectors $\delta S_u = \frac{\delta S(u,v)}{\delta u}$ and $\delta S_v = \frac{\delta S(uv)}{\delta v}$. The normal to the surface n is then defined as $n = \frac{\delta S_u}{\|\delta S_u\|} \times \frac{\delta S_v}{\|\delta S_v\|}$. Informally, the *curvature* of a curve C at a point p is defined as the reciprocal of the radius of the circle that most closely conforms to the curve at the given point (where a straight line is viewed as a circle of infinite radius). The *normal curvature* of a surface at a point p with respect to a tangential direction t is defined as the curvature at p of a curve formed by intersecting the surface with a plane that contains p and whose normal is orthogonal to both t and the surface normal at p. The maximal k_{max} and minimal k_{min} curvatures at each point p are defined as the maximal and minimal values of normal curvature at this point. The curvature directions that correspond to these values are orthogonal, and are referred to as the principal curvature directions. The surface curvature at any given point can be characterized based on the properties of the minimal and maximal curvatures (positive, negative, zero). Notably, the normal curvature of a ruled surface along the ruling direction is always zero.

REFERENCES

[1] Adobe. 2019. Adobe Illustrator. (2019). https://www.adobe.com/products/illustrator.html

[2] William Albert and Thomas Tullis. 2013. *Measuring the user experience: collecting, analyzing, and presenting usability metrics*. Newnes.

[3] Judith Amores and Jaron Lanier. 2017. HoloARt: Painting with Holograms in Mixed Reality. In *Proc. Human Factors in Computing Systems*. 421–424.

[4] Rahul Arora, Rubaiat Habib Kazi, Fraser Anderson, Tovi Grossman, Karan Singh, and George Fitzmaurice. 2017. Experimental Evaluation of Sketching on Surfaces in VR. In *Proc. Human Factors in Computing Systems*. 5643–5654.

[5] Autodesk. 2019a. 3DS Max. (2019). https://www.autodesk.com/products/3ds-max/overview

[6] Autodesk. 2019b. Maya. (2019). http://www.autodesk.com/maya

[7] Seok-Hyung Bae, Ravin Balakrishnan, and Karan Singh. 2008. ILoveSketch: As-natural-as-possible Sketching System for Creating 3D Curve Models. In *Proc. Symposium on User Interface Software and Technology*. 151–160.

[8] Ilya Baran, Jaakko Lehtinen, and Jovan Popovic. 2010. Sketching Clothoid Splines Using Shortest Paths. *Comput. Graph. Forum* 29, 2 (2010), 655–664.

[9] Mayra Donaji Barrera Machuca, Wolfgang Stuerzlinger, and Paul Asente. 2019. The Effect of Spatial Ability on Immersive 3D Drawing. In *Proceedings of the 2019 on Creativity and Cognition (C&C '19)*. 173–186.

[10] Mikhail Bessmeltsev, Caoyu Wang, Alla Sheffer, and Karan Singh. 2012. Design-Driven Quadrangulation of Closed 3D Curves. *ACM Trans.Grap.* 31, 5 (2012).

[11] DesignSpace. 2019. DesignSpace VR. (2019). http://www.designspacevr.org/

[12] Holger Diehl, Franz Müller, and Udo Lindemann. 2004. From raw 3D-Sketches to exact CAD product models Concept for an assistant-system. In *Sketch Based Interfaces and Modeling*.

[13] M.P. do Carmo. 2016. *Differential Geometry of Curves and Surfaces: Revised and Updated Second Edition*. Dover Publications.

[14] Gerald Farin. 2002. *Curves and Surfaces for CAGD: A Practical Guide* (5th ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[15] Michele Fiorentino, Raffaele de Amicis, Giuseppe Monno, and Andre Stork. 2002. Spacedesign: A Mixed Reality Workspace for Aesthetic Industrial Design. In *Proc. Symposium on Mixed and Augmented Reality*.

[16] Google. 2019. Google Tilt Brush repository. (2019). https://poly.google.com/tiltbrush

[17] GoogleBlocks. 2019. Google Blocks. (2019). https://vr.google.com/blocks/

[18] GravitySketch. 2019. Gravity Sketch. (2019). https://www.gravitysketch.com/

[19] Tovi Grossman, Ravin Balakrishnan, Gordon Kurtenbach, George Fitzmaurice, Azam Khan, and Bill Buxton. 2002. Creating Principal 3D Curves with Digital Tape Drawing. In *Proc. Human Factors in Computing Systems*. 121–128.

[20] Tovi Grossman, Ravin Balakrishnan, and Karan Singh. 2003. An Interface for Creating and Manipulating Curves Using a High Degree-of-freedom Curve Input Device. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. 185–192.

[21] Sandra G Hart and Lowell E Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. *Human mental workload* 1, 3 (1988), 139–183.

[22] Juan David Hincapié-Ramos, Xiang Guo, Paymahn Moghadasian, and Pourang Irani. 2014. Consumed Endurance: A Metric to Quantify Arm Fatigue of Mid-air Interactions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. 1063–1072.

[23] Zhiyang Huang, Nathan Carr, and Tao Ju. 2019. Variational Implicit Point Set Surfaces. *ACM Trans. Graph.* 38, 4, Article 124 (July 2019), 124:1–124:13 pages.

[24] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. 1999. Teddy: A Sketching Interface for 3D Freeform Design. In *Proc. SIGGRAPH*. 409–416.

[25] J.H. Israel, E. Wiese, M. Mateescu, C. Zöllner, and R. Stark. 2009. Investigating three-dimensional sketching for early conceptual designâĂŤResults from expert discussions and user studies. *Computers and Graphics* (2009), 462 – 473.

[26] B. Jackson and D. F. Keefe. 2016. Lift-Off: Using Reference Imagery and Freehand Sketching to Create 3D Models in VR. *IEEE Trans. on Visualization and Computer Graphics* (2016), 1442–1451.

[27] Sujin Jang, Wolfgang Stuerzlinger, Satyajit Ambike, and Karthik Ramani. 2017. Modeling Cumulative Arm Fatigue in Mid-Air Interaction Based on Perceived Exertion and Kinetics of Arm Motion. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. 3328–3339.

[28] Joaquim Jorge and Faramarz Samavati. 2011. *Sketch-based Interfaces and Modeling*.

[29] James G Kalbfleisch. 2012. *Probability and statistical inference*. Springer Science & Business Media.

[30] D. Keefe, R. Zeleznik, and D. Laidlaw. 2007. Drawing on Air: Input Techniques for Controlled 3D Line Illustration. *IEEE Transactions on Visualization and Computer Graphics* 13, 5 (2007), 1067–1081.

[31] Daniel F. Keefe, Daniel Acevedo Feliz, Tomer Moscovich, David H. Laidlaw, and Joseph J. LaViola, Jr. 2001. CavePainting: A Fully Immersive 3D Artistic Medium and Interactive Experience. In *Proc. Symposium on Interactive 3D Graphics*. 85–93.

[32] Yongkwan Kim, Sang-Gyun An, Joon Hyub Lee, and Seok-Hyung Bae. 2018. Agile 3D Sketching with Air Scaffolding. In *Proc. Human Factors in Computing Systems*. 238:1–238:12.

[33] Kodon. 2019. TenkLabs Kodon. (2019). https://www.tenklabs.com/kodon

[34] Jung-hoon Kwon, Han-wool Choi, Jeong-in Lee, and Young-Ho Chai. 2005. Free-Hand Stroke Based NURBS Surface for Sketching and Deforming 3D Contents. In *Proc. Pacific-Rim Conference on Advances in Multimedia Information Processing*. 315–326.

[35] Joseph J LaViola Jr, Ernst Kruijff, Ryan P McMahan, Doug Bowman, and Ivan P Poupyrev. 2017. *3D user interfaces: theory and practice*. Addison-Wesley Professional.

[36] Changjian Li, Hao Pan, Yang Liu, Xin Tong, Alla Sheffer, and Wenping Wang. 2017. BendSketch: Modeling Freeform Surfaces Through 2D Sketching. *ACM Trans. Graph.* (2017), 125:1–125:14.

[37] James McCrae and Karan Singh. 2009. Sketch-Based Interfaces and Modeling (SBIM): Sketching Piecewise Clothoid Curves. *Comput. Graph.* 33, 4 (2009), 452–461.

[38] John Napier. 1993. *Hands*. Princeton University Press.

[39] Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. 2007. FiberMesh: Designing Freeform Surfaces with 3D Curves. *ACM Trans. Graph.* 26, 3 (2007).

[40] Oculus. 2019. Quill. (2019). https://quill.fb.com/

[41] OculusMedium. 2019. OculusMedium. (2019). https://oculus.com/medium/

[42] L. Olsen, F.F. Samavati, M.C. Sousa, and J. Jorge. 2009. Sketch-Based Modeling: A Survey. *Computers & Graphics* 33 (2009). Issue 1.

[43] PaintLab. 2019. PaintLab VR. (2019). http://paintlabvr.com/

[44] Hao Pan, Yang Liu, Alla Sheffer, Nicholas Vining, Chang-Jian Li, and Wenping Wang. 2015. Flow Aligned Surfacing of Curve Networks. *ACM Trans. Graph.* (2015), 127:1–127:10.

[45] Pixologic. 2019. ZBrush. (2019). http://pixologic.com/

[46] Dominik Rausch, Ingo Assenmacher, and Torsten Kuhlen. 2010. 3D Sketch Recognition for Interaction in Virtual Environments. In *Workshop in Virtual Reality Interactions and Physical Simulation*. The Eurographics Association.

[47] Enrique Rosales, Jafet Rodriguez, and Alla Sheffer. 2019. SurfaceBrush: From Virtual Reality Drawings to Manifold Surfaces. *ACM Transaction on Graphics* 38, 4 (2019).

[48] E. Sachs, A. Roberts, and D. Stoops. 1991. 3-Draw: a tool for designing 3D shapes. *IEEE Computer Graphics and Applications* 11, 6 (1991), 18–26.

[49] Steven Schkolne, Michael Pruett, and Peter Schröder. 2001. Surface Drawing: Creating Organic 3D Shapes with the Hand and Tangible Tools. In *Proc. Human Factors in Computing Systems*. 261–268.

[50] ShapeLab. 2019. ShapeLab. (2019). https://store.steampowered.com/app/571890/ShapeLab/

[51] Ivan E. Sutherland. 1964. Sketch Pad a Man-machine Graphical Communication System. In *Proceedings of the SHARE Design Automation Workshop (DAC '64)*. 6.329–6.346.

[52] Shun'ichi Tano, T. Kodera, Takashi Nakashima, I. Kawano, K. Nakanishi, G. Hamagishi, M. Inoue, A. Watanabe, T. Okamoto, K. Kawagoe, K. Kaneko, T. Hotta, and M. Tatsuoka. 2003. Godzilla: Seamless 2D and 3D Sketch Environment for Reflective and Creative Design Work. In *INTERACT*.

[53] Shun'ichi Tano, Shinya Yamamoto, Junko Ichino, Tomonori Hashiyama, and Mitsuru Iwata. 2013. Truly Useful 3D Drawing System for Professional Designer by "Life-Sized and Operable" Feature and New Interaction. In *Human-Computer Interaction – INTERACT 2013*. 37–55.

[54] TiltBrush. 2019. Google TiltBrush. (2019). https://tiltbrush.com/

[55] Unity. 2019. SteamVR Plugin. (2019). https://assetstore.unity.com/packages/tools/integration/steamvr-plugin-32647

[56] Wacom. 2019. Wacom. (2019). https://www.wacom.com

[57] Gerold Wesche and Hans-Peter Seidel. 2001. FreeDrawer: A Free-form Sketching System on the Responsive Workbench. In *Proc. Virtual Reality Software and Technology*. 167–174.

[58] E. Wiese, J. H. Israel, A. Meyer, and S. Bongartz. 2010. Investigating the Learnability of Immersive Free-hand Sketching. In *Proc. Sketch-Based Interfaces and Modeling Symposium*. 135–142.

[59] Baoxuan Xu, William Chang, Alla Sheffer, Adrien Bousseau, James McCrae, and Karan Singh. 2014. True2Form: 3D Curve Networks from 2D Sketches via Selective Regularization. *ACM Trans. Graph.* 33, 4 (2014), 131:1–131:13.

[60] Shumin Zhai. 1998. User Performance in Relation to 3D Input Device Design. *SIGGRAPH Comput. Graph.* 32, 4 (Nov. 1998), 50–54.