https://doi.org/10.2352/EI.2022.34.12.ERVR-297

VirtualForce: Simulating Writing and Sketching on a 2D-Surface in Virtual Reality

Ziyang Zhang, Jürgen P. Schulze; University of California, San Diego; La Jolla, California

Abstract

Realistically writing and sketching on virtual surfaces in virtual reality (VR) enhance the potential uses of VR in production and art creation. To achieve a similar writing experience as in the real world, the VR system needs to be pressure sensitive to the user's stylus on the virtual plane to create the effect of different pen strokes and improve the user experience. Typical 6 degree-of-freedom (DoF) VR controllers do not measure force or pressure on surfaces because they are held in mid-air. We propose a new method, VirtualForce, to calculate the force based on the difference between the user's physical hand position and their hand avatar in VR. Our method does not require any specialized hardware. Furthermore, we explore the potential of our method to improve the creation of VR art, and we present several ways in which VirtualForce can greatly enhance the accuracy of drawing and writing on virtual surfaces in virtual reality.

Introduction

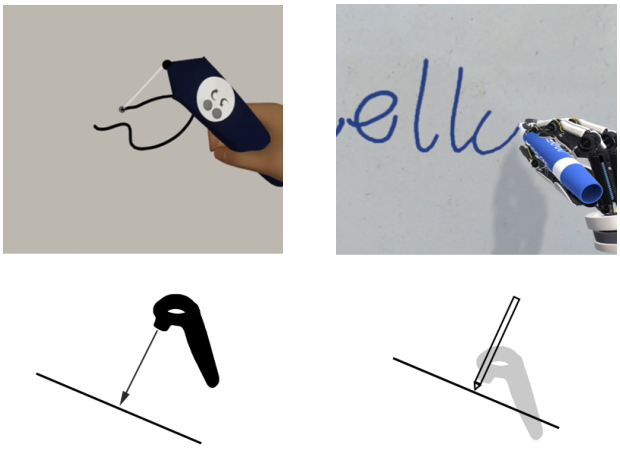
Writing by using virtual reality (VR) techniques has been popularized with the increasing availability of VR devices. Researchers have started to study the interaction techniques that can be deployed in VR applications to improve writing precision and user experience.

However, most popular VR writing/sketching applications such as Tilt Brush[1] are focused on 3D mid-air painting. Several applications have shown that 2D writing is an important user interaction component in VR[2, 3]. Therefore, techniques and applications that focus on writing on a 2D plane are few and need to be explored further.

Current studies in 2D writing or sketching in VR are focusing on using specialized hardware to achieve the user experience [4, 5]. With only common 6 degree-of-freedom (DoF) controllers, those techniques cannot be deployed because they generally require a digital pad with 6-DoF tracking devices on it. Although specialized hardware can enable users to sketch on a real plane and provide real tactile feedback, it greatly compromises the availability and accessibility.

The main challenge to use general VR controllers only to write on 2D planes is the non-existence of the physical planes. Without physical planes, users are unable to get accurate tactile feedback. A more serious problem is the application has no way to acquire how much force does the user puts on the plane. Without the force information, the potential of the writing and sketching application in collaboration and artistic creation will be greatly compromised because of the lack of feedback and inaccurate strokes.

Therefore, we present a method, *VirtualForce*, that can enable VR systems to gain pressure sensitivity. As shown in the figure 2, our method only uses a generally available 6-DoF controller and calculates the force by the *hand-unaligned distance*, d. Furthermore, we conduct several experiments to evaluate the enhancement of writing accuracy and legibility by using our method.



(a) Laser Pointer (CollabHub[6]) (b) Virtual Stylus (The Lab [7])

Figure 1: Different methods for writing/sketching on 2D planes in VR.

Related Works

Unconstrained 3D Painting in VR

The development of VR techniques enables artists to paint in mid-air (i.e., directly paint in 3D spaces). Common 3D VR painting software (e.g., Tilt-Brush[1], and Quill[8]) allows users to create 3D paths and models immersively in a VR space. Those painting procedures are often unconstrained. Consequently, the precision of painting is low. Several previous studies demonstrate methods to help unconstrained painting be accurate, which includes stroke auto-correction [9].

2D Writing/Sketching Methods in VR

2D sketching in VR often enables users to draw strokes on a virtual 2D plane. Compared to 3D painting in VR, 2D sketching in VR is less intuitive because users need to interact with a non-existing virtual plane. Therefore, some intuitive feedback in the real world, such as force feedback, is hard to simulate in VR. Various approaches are attempted to solve this problem, such as using specialized hardware and changing physical collision-based pens into ray casting pens.

Methods with Specialized Hardware

The main problem with 2D sketching is that the 2D plane in VR does not exist in the physical world, so one approach to solv-

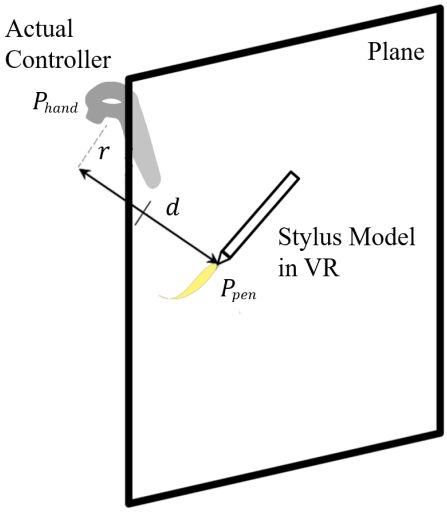


Figure 2: The schematic of our method. The hand-unaligned distance, d, can be used to adjust the size of the brush.

ing this problem is introducing a physical plane into the physical world. Keefe et al. invent a method to sketch in a CAVE which is a four-wall immersive VR system[10]. With more mobility, several studies have introduced digital pads and pens with 6 DoF tracking devices. Those approaches often use the pads and pens as the interacting controllers and build corresponding 3D models in the VR system to help users sketch on 2D planes in VR accurately and intuitively[4, 5]. Because of the existing physical plane, users can naturally gain all the physical feedback, and the system can also gain important information from users, such as how much force the user put onto the plane, which can enable the system to create more realistic strokes.

Those studies demonstrate that with specialized physical hardware such as digital pads and pens, VR systems can support 2D sketching well. However, specialized hardware severely affects the availability of the system. In common VR systems such as Oculus Quest, HTC Vive, and Valve Index, there are only two 6 DoF controllers designed for gaming. Currently, digital pads with 6 DoF trackers are rare and expensive to get.

Methods with General Controllers

Without specialized hardware, one common approach to paint on a 2D plane is to use a virtual laser pointer. As shown in figure 1(a), a laser pointer generates a virtual ray towards the plane and produces stroke at the intersection point. Several commonly available applications such as CollabHub[6] are using this method.

Because the ray casting pen does not require an actual collision between the pen and the plane, the physical feedback to users is not essential. However, this method does not follow people's intuition either because people generally do not use this kind of pen in the real world. Also, because this method has no way to gain force information from users, the ability of this method to produce accurate and realistic strokes is limited.

Another way is to simply simulate a virtual plane and use the controller as the stylus (e.g., the whiteboard in The Lab) as shown in figure 1(b). The advantage of this method is that it matches people's intuition and common practice; however, the precision of this method is low. As shown in the experiment section, because of the lack of physical feedback, it is very hard to control the stroke, especially when lifting the pen after finishing writing one letter.

Figure 3: Strokes created by the VirtualForce. Because of the pressure sensitivity, the brush size and transparency is changing during the writing.

In order to produce realistic and accurate strokes with general controllers, we propose a new method, VirtualForce, based on the virtual stylus. Our method can enable the VR system to gain force information without using any specialized hardware, and the system can create stroke based on users' force input. Our main contributions in this paper are:

- **VirtualForce** A method that calculates the force input from users based on the unaligned distance between the real controller and the avatar in the VR. This system can create realistic stroke based on the force information, which improves the quality of the stroke, user experience and the potential of general VR controllers in art creation.
- **Evaluation** We evaluate our VirtualForce by several experiments. Our results show that VirtualForce can improve the precision of the stroke during writing and sketching. Also, we propose a new experiment method to evaluate the legibility of text written in VR by using a optical character recognition (OCR) system to recognize the written letters.

Method

Force Calculation and Stroke Mapping

To enhance the capability of creating realistic strokes, VR systems need to control the brush size by user's force input. We use the term force and pressure interchangeably since we assume the size of the pen nibs is constant. Assume we have a brush texture with width, w, without force information, such as naive virtual plane or ray casting method, w remains constant during painting. Usually, painting applications allow users to change the brush size manually, so there is also a coefficient, c, that controls the brush size. Therefore, the final brush size, W, will be:

$$W = c \cdot w \tag{1}$$

After we get W, a stroke will be mapped on a canvas, represented as a virtual 2D plane in the VR space.

Considering W should change while the users put different amounts of force on the virtual plane, we need to build a mapping between the brush size and the force. We represent this mapping

relation as $S(F)$, where F is the force input, and S is a pressure-sensitivity function that produces a coefficient that alters the brush size based on the force input. Equation 1 can be modified into:

$$W = c \cdot S(F) \cdot w \tag{2}$$

For painting applications on modern digital tablets, this function is usually called pressure sensitivity or pressure curve. In our setting, since our pen is represented in the VR space with a fixed size of contacting area with the canvas, the pressure can be directly represented by force, and we define this function as **pressure-sensitivity function**. The choice of the sensitivity curve can be various (e.g., linear functions, exponential functions, or manually controlled by users). In our case, to keep simplicity, the sensitivity curve function is given by a piece-wise linear function:

$$S(F) = \begin{cases} \alpha F & F < F_{max} \\ \alpha F_{max} & F \geq F_{max} \end{cases} \tag{3}$$

where $F \geq 0$, α is a coefficient that control the sensitivity to the force input, and F_{max} is a constant that limits the maximum force input.

An important problem is how to measure F. If a VR system only has general 6-DoF controllers without any physical tablets, it is hard to measure the force directly. As shown in the figure 2, the *hand-unaligned distance*, d, can be used to measure the force input.

Assume we have a pen in the VR space whose position is determined by the position of a corresponding controller. Normally, the location of the pen strictly follows the position of the controller. However, when the pen collides with the canvas, we need to prevent the pen from going through the canvas. More formally, in the world coordinate system, assume the position of the pen head is represented as P_{pen}, and the canvas is a rectangle. Also, we have a hand position in VR space, P_{hand}, which represents the location of the controller in the VR space. When the pen does not collide with the canvas, we strictly map the P_{pen} to the P_{hand} with a fixed offset r:

$$P_{pen} = P_{hand} + r \tag{4}$$

where the offset r is used to adjust the avatar location based on different models of the pen.

When there is a collision between the pen and the canvas, we instead set the p_{pen} to the collision position. In this condition, the position of the hand and the position of the stylus model are unaligned. Then, we can compute the hand position unaligned distance, d, by:

$$d = F = ||P_{pen} - P_{hand} - r||_2 \tag{5}$$

Intuitively, if a user puts more force onto the canvas, the user's hand will go deeper into the canvas, and thus the hand position unaligned distance will be larger. Therefore, we can use d to measure the force input F.

After we can calculate the force input F, our final brush size mapping equation is:

$$W = c \cdot S(d) \cdot w \tag{6}$$

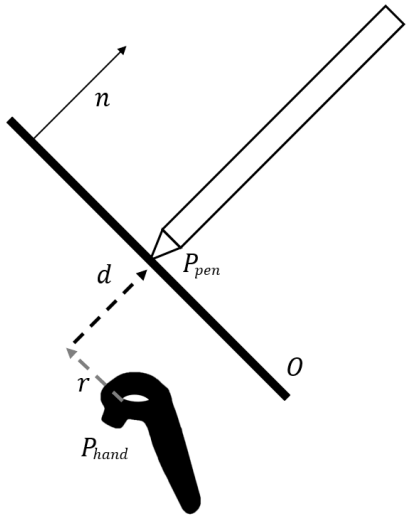


Figure 4: The schematic of our method without using rigid-body collision detection algorithm.

As shown in the figure 3, the writing created by our method enable the brush size to be changed by the pressure. Also, it is worth to notice that the virtual force can not only change the size of the brush, but also be able to modify the transparency of the stroke, which is also very common in digital painting applications.

Stylus-Controller Location Mapping

Our method requires gaining the position of the collision between the stylus and the plane in order to calculate the hand-unaligned distance. We propose a method that achieves our method without using rigid-body collision detection algorithms in order to simplify the deployment procedure in different VR frameworks and platforms.

As shown in the figure 4, the plane or canvas is defined by its normal, n, and a point O. We can determine whether the pen penetrates the plane or not by calculating

$$\delta = n \cdot (P_{hand} + r - O) \tag{7}$$

If $\delta > 0$, there is no penetration nor hand-unaligned distance. If $\delta < 0$, there is penetration, and we project the location of the hand onto the plane and map the P_{pen} to:

$$P_{pen} = P_{hand} + r - \delta \cdot n \tag{8}$$

Then, we can calculate the hand-unaligned distance without using rigid-body collision detection algorithms.

Experiments

In this section, we evaluate how much accuracy and legibility can our method enhance qualitatively and quantitatively. We also explore the possibility of our method in sketch creation.

Accuracy and Legibility

Accuracy

We define accuracy as how well users draw strokes on designated paths (e.g., writing on top of some printed text). To eliminate human factors, we only record the writing path and separately

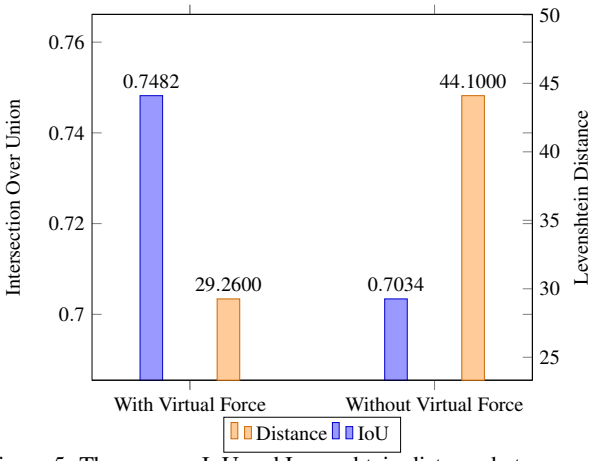


Figure 5: The average IoU and Levenshtein distance between our method and the baseline.

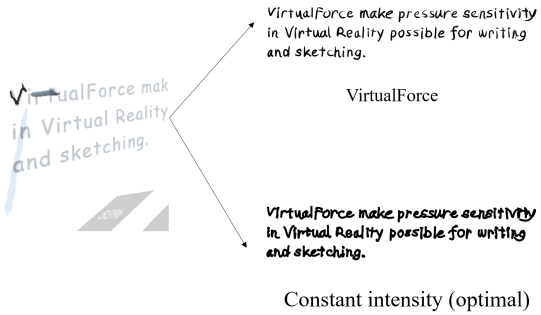


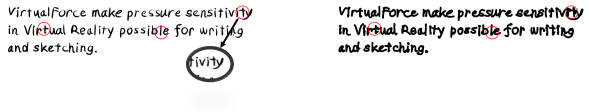
Figure 6: We write on top of printed text, and simultaneously generate handwritten text with and without VirtualForce to eliminate human-factors during experiments.

generate the stroke with and without using VirtualForce, as shown in the figure 6. We test our method on 20 pieces of printed text and calculate the intersection over union (IoU) between the printed text and our written images, with and without VirtualForce. IoU is defined as $\frac{|W \cap T|}{|W \cup T|}$ where W is all the black pixels of our written text, and T is all the black pixels of the printed text. Our experiment shows that using our method, the IoU raises from 76.98% to 79.83%.

Legibility

Writing accuracy is not equivalent to legibility. Some errors in important parts of the text, such as the conjunctions of letters, can severely affect the legibility even if they affect the IoU mildly. We find that without pressure sensitivity, errors are frequently made when lifting the pen after finishing writing one letter (figure 7). Because of the lack of a physical plane, users cannot precisely control the virtual stylus, especially for lifting the stylus nib from the plane. With constant brush size, such accidental writing will leave heavy strokes, which can affect the legibility. With our method, however, it is not a problem because then hand unaligned distance is small, and the intensity of the accidental strokes will be low.

The legibility can also be quantitatively measured. We generate two types of handwritten text with and without using VirtualForce, similar to the accuracy experiment, and put them into an OCR model. We assume that the more legible the text is, the more accurate the recognized text from the OCR is. Such methods which use a classification model to measure the performance of another method are often used in machine learning[11]. Our experiments show that our method can decrease the Levenshtein distance[12] between the OCR recognized handwritten text and ground truths by 33% (figure 5), which shows that VirtualForce can enhance handwriting legibility greatly.



(a) VirtualForce (b) Constant size (optimal)

Figure 7: The errors when lifting pen are significantly eliminated with VirtualForce.

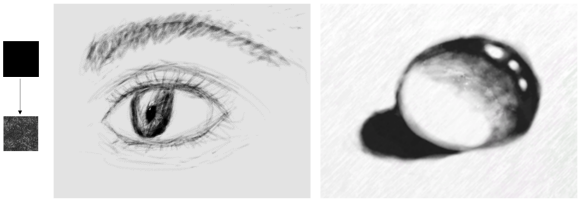


Figure 8: Sketches drawn in VR with a pencil texture. Pressure sensitivity makes user easier to control the strokes.

Sketches Creation

We can also apply more complex textures to the brush to enable high-quality sketching. As shown in the figure 8, we replace the pure black texture with a pencil texture and use our method to create drawing in VR. As the demo image shows, our method can greatly enhance the quality of 2D sketching in VR. Those sketches are hard to create without pressure sensitivity.

Conclusion

We propose a new method, VirtualForce, that can calculate the force input from users and enable the VR system to gain pressure sensitivity similar to graphic tablets. Unlike previous VR 2D sketching methods, our method can acquire the user's intended force without any specialized hardware but only commercial 6-DoF controllers. Our experiments show that our approach can greatly improve the 2D writing accuracy and legibility in VR. Our method can also be used in fine art creation and creating high-quality sketches.

References

[1] D. Skillman and P. Hackett. Tilt brush, 2016.

[2] Kiwon Yeom, Jounghuem Kwon, JooHyun Maeng, and Bum-Jae You. [poster] haptic ring interface enabling air-writing in virtual reality environment. In *2015 IEEE International Symposium on Mixed and Augmented Reality*, pages 124–127. IEEE, 2015.

[3] Marianne Patera, Steve Draper, and Martin Naef. Exploring magic cottage: A virtual reality environment for stimulating children's imaginative writing. *Interactive Learning Environments*, 16(3):245–263, 2008.

[4] Ivan Poupyrev, Numada Tomokazu, and Suzanne Weghorst. Virtual notepad: handwriting in immersive vr. In *Proceedings. IEEE 1998 Virtual Reality Annual International Symposium (Cat. No. 98CB36180)*, pages 126–132. IEEE, 1998.

[5] Tobias Drey, Jan Gugenheimer, Julian Karlbauer, Maximilian Milo, and Enrico Rukzio. 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*, pages 1–14, 2020.

[6] CollabHub. Collabhubvr. https://www.collabhubvr.com/, 2021.

[7] Valve. The lab. https://store.steampowered.com/app/450390/The_Lab/, 2016.

[8] Oculus. Quill. https://www.oculus.com/experiences/rift/1118609381580656/, 2020.

[9] Rahul Arora and Karan Singh. Mid-air drawing of curves on 3d surfaces in virtual reality. *ACM Transactions on Graphics (TOG)*, 40(3):1–17, 2021.

[10] Daniel F Keefe, Daniel Acevedo Feliz, Tomer Moscovich, David H Laidlaw, and Joseph J LaViola Jr. Cavepainting: A fully immersive 3d artistic medium and interactive experience. In *Proceedings of the 2001 symposium on Interactive 3D graphics*, pages 85–93, 2001.

[11] Shai Ben-David, John Blitzer, Koby Crammer, Fernando Pereira, et al. Analysis of representations for domain adaptation. *Advances in neural information processing systems*, 19:137, 2007.

[12] Li Yujian and Liu Bo. A normalized levenshtein distance metric. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1091–1095, 2007.

Author Biography

Mr. Ziyang Zhang is a Master's student at the University of California, San Diego majoring in computer science. He received his B.S. degree from the University of California, Irvine. His research interests include virtual reality and non-photo-realistic rendering.

Dr. Jurgen Schulze is an Associate Adjunct Professor in the computer science department at the University of California, San Diego. His research interests include novel applications for virtual and augmented reality systems, 3D human-computer interaction and medical data visualization. He holds an M.S. degree from the University of Massachusetts and a Ph.D. from the University of Stuttgart, Germany. After his graduation he spent two years as a post-doctoral researcher in the Computer Science Department at Brown University working on real-time volume rendering in virtual reality.