PM4WireArt: A Scriptable Parametric Modeling Interface for Conceptual Wire Art Design Using PM4VR

Wanwan Li
University of Tulsa
Tulsa, Oklahoma, USA
wanwan-li@utulsa.edu

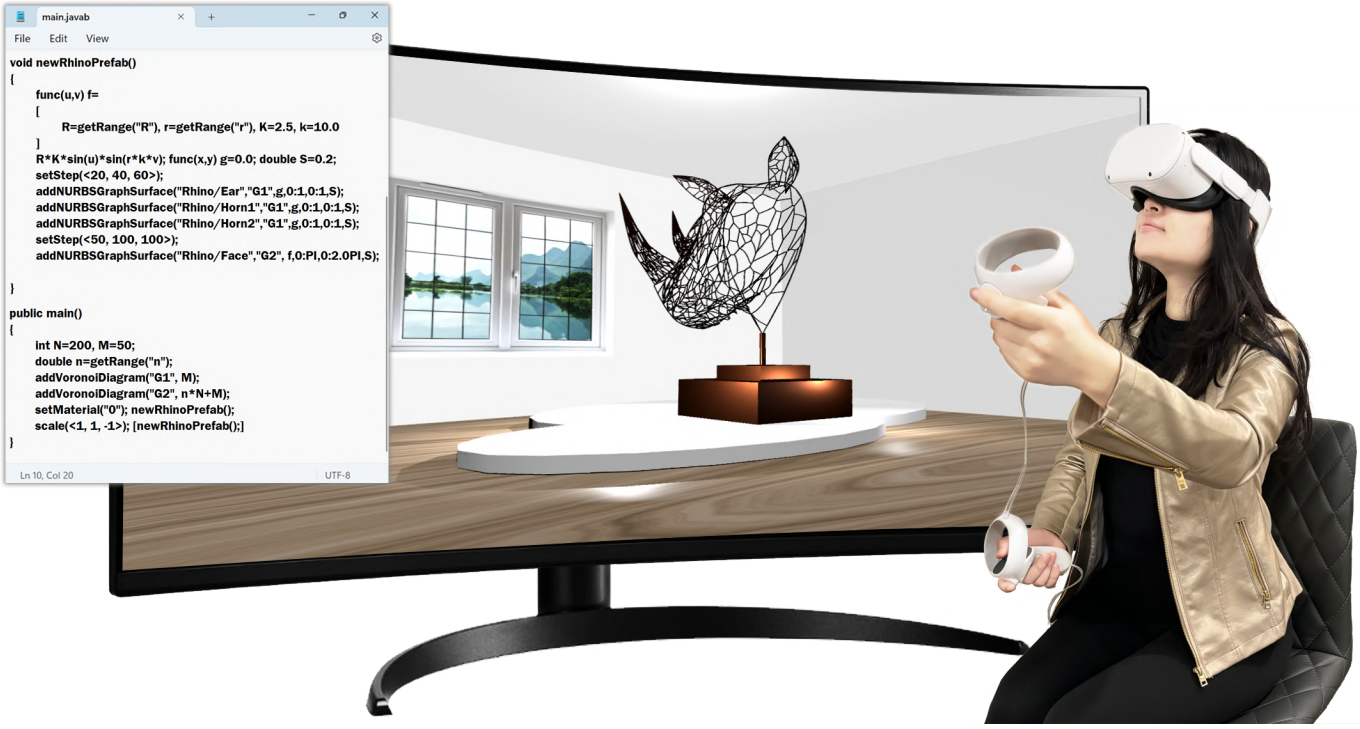


Fig. 1. Teaser. This teaser showcases PM4WireArt, a scriptable parametric modeling tool for conceptual wire art designs using PM4VR. Designers create a Java$^\flat$ script (left figure) to define the parameters, and PM4WireArt generates a 3D wire art model in real-time (middle figure). By connecting PM4WireArt to Oculus Quest 2 through the SteamVR plugin, designers can fine-tune the parameters using VR controllers within an immersive virtual environment of a wire art show lobby (right figure).

***Abstract*—This paper introduces PM4WireArt, a novel scriptable parametric modeling interface designed for conceptual wire art design. Leveraging the capabilities of PM4VR, a virtual reality-based parametric modeling platform, PM4WireArt provides artists with a versatile and intuitive toolset for designing intricate wire sculptures. This paper outlines the key features, design principles, and potential applications of PM4WireArt, highlighting its potential to revolutionize the way artists conceptualize and create wire art in virtual environments.**

***Keywords*—Parametric Modeling, Conceptual Wire Art Design**

I. Introduction

Wire art [1], [2] stands as a distinctive and captivating form of artistic expression, characterized by the skillful shaping and manipulation of metallic wires [3] to forge visually stunning structures. Wire art design has a rich history [4] with artists engaged in the meticulous crafting of intricate sculptures that transcend mere physicality, effectively conveying emotion, movement, and form through the versatile medium of metallic wires. Traditionally, the creation of wire art demanded a delicate fusion of manual dexterity and the artist's intuitive sensibility [5], necessitating labor-intensive efforts for the realization of their envisioned forms [6].

However, the landscape of artistic creation has undergone a transformative evolution with the advent of technology [7]. This evolution opens up a realm of possibilities [8] by presenting an opportune juncture to integrate cutting-edge Computer-Aided Design (CAD) [9] techniques and Virtual Reality (VR) [10], [11] into the traditional practice of wire art [12]. In this paradigm shift, the digital wire art design interface emerges as a revolutionary tool [13]–[15], providing artists with the means of enhanced efficiency and creativity and allowing artists to experiment with intricate and complex geometries that were once confined to the realm of imagination. The integration of VR amplifies the creative process by immersing artists in virtual space [16]–[24] to iterate their designs with a level of immediacy previously unimaginable.

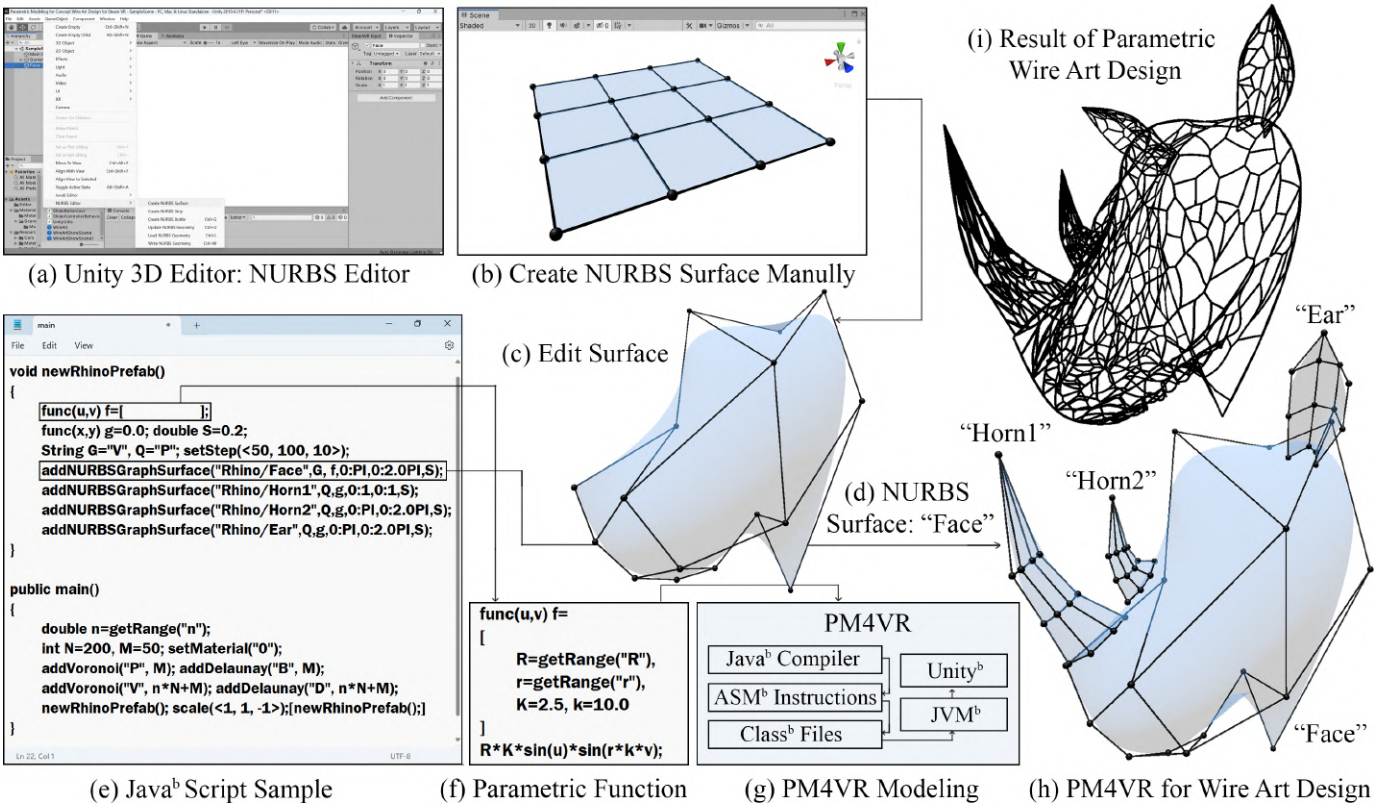


Fig. 2. Overview of our approach.

This paper introduces PM4WireArt, a novel scriptable parametric modeling interface for conceptual wire art design. Fig. 1 provides a preview of the capabilities inherent in PM4WireArt. In this example, a wire art designer embarks on the creative design by crafting a Java♭ script (left figure). The real-time execution of this Java♭ script within the PM4VR [25]–[28] interface can automatically generate realistic 3D conceptual wire art sculpture models (middle figure). After connect PM4WireArt with the Oculus Quest 2 VR headset through the SteamVR plugin, the designer armed with VR controllers was seamlessly transitioned into a virtual environment (right figure) and gains the ability to fine-tune intricate parameters within a wire art show lobby, facilitating the transformation of the initial conceptual design into a immersive showcase.

II. Technical Approach

Overview. Our technical approach is illustrated in Fig. 2. We introduced a Unity3D Editor plugin named NURBS Editor (a). By pressing Control+G, a default NURBS surface is created (b), which can be modified by adjusting control points while pressing Control+U (c). A refined NURBS surface can be saved with Control+W (d). The Java♭ script (e) and its corresponding parametric function (f) serve as the script input for a wire art design. When this Java♭ script is fed into the PM4VR interface (g), it automatically loads the NURBS surfaces of "Face", "Horn", "Ear", etc.(h), generating a parametric wire art design (i) in real-time via the PM4WireArt interface.

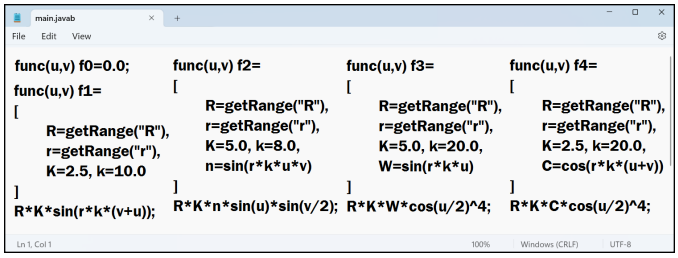


Fig. 3. Java ♭ Scripts for Parametric Functions.

Graph Structure. For PM4WireArt interface, we extended the PM4VR interface by implementing three basic graph structures to generate the wire sculpture geometry including: (a) Random Graph, (b) Delaunay Triangle, and (c) Voronoi Diagram. With this extension, designer can add these three graph structures in Java♭ script using function calls of **addRandomGraph** $(\langle G\rangle, |V|)$, **addDelaunayTriangle**$(\langle G\rangle, |V|)$, and **addVoronoiDiagram**$(\langle G\rangle, |V|)$ respectively, where $\langle G\rangle$ is a string specifying the name of graph data structure $G = (V, E)$.

Graph Surface. After above Java♭ function calls are invoked in PM4WireArt, those graph structures will be generated and inserted into HashTable H with key-value pairs of $(\langle G\rangle, G)$. Give, arbitrary surface's parametric equation $\mathbf{f}(u, v) = \langle x(u, v), y(u, v), z(u, v)\rangle$, where $u \in [u_0, u_1]$, $v \in [v_0, v_1]$. Designer can add graph surface in Java♭ script using function **addGraphSurface**$(\langle G\rangle, \mathbf{f}(u, v), u_0 : u_1, v_0 : v_1, S)$.

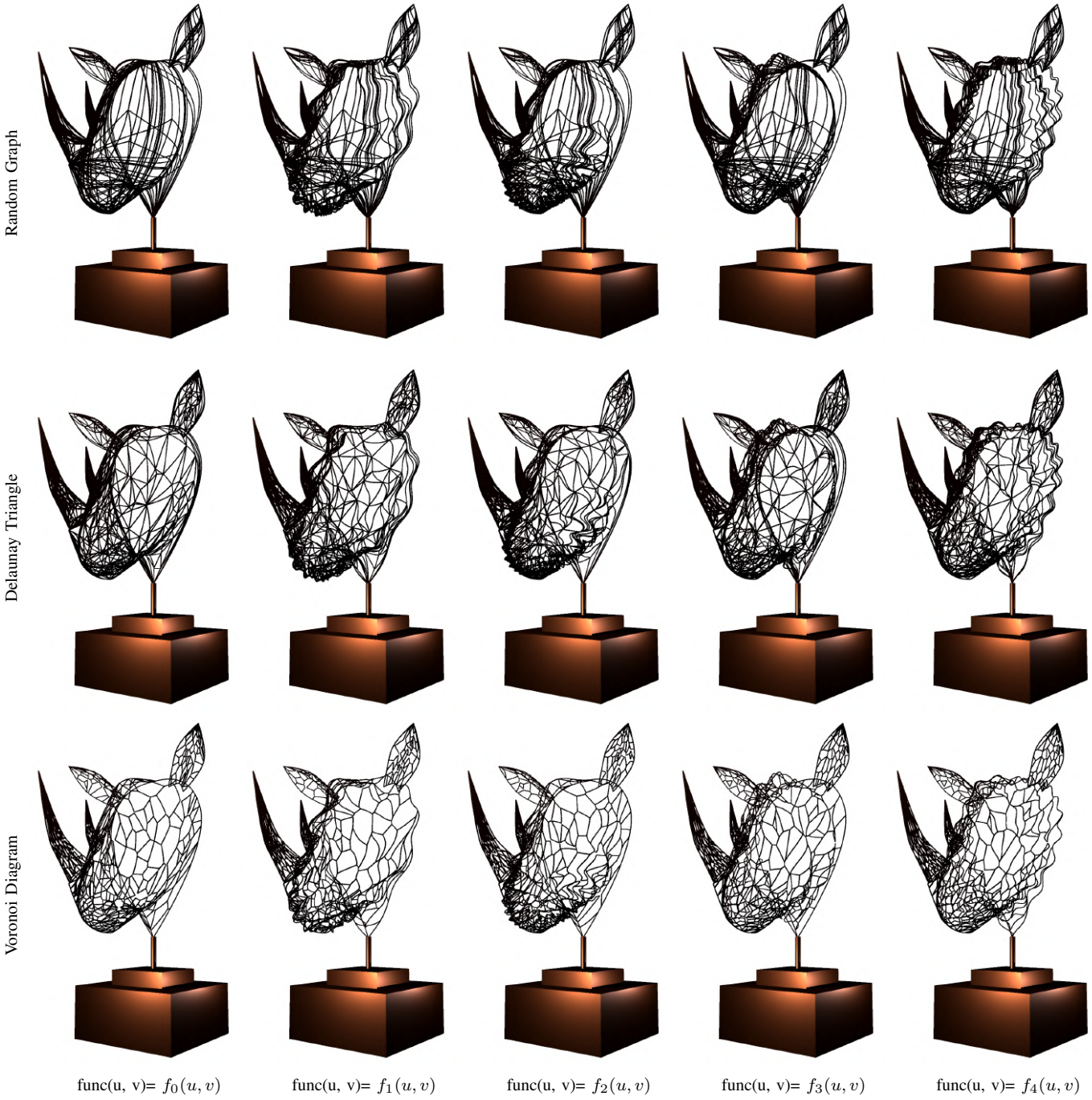


Fig. 4. Different Parametric Settings. This figure shows wire arts designed via PM4WireArt with different parametric settings.

NURBS Graph Surface. The NURBS Graph Surface $\mathbf{g}(u,v)$, where $(u,v) \in E$, is created by nesting the graph structure $G = (V,E)$ on top of a NURBS surface $\mathbf{s}(u,v)$ with a displacement defined by the parametric function $f(u,v)$. Designer can add NURBS graph surface by **addNURBSGraphSurface**("NURBS name", $\langle G \rangle, \mathbf{f}(u,v), u_0 : u_1, v_0 : v_1, S$).

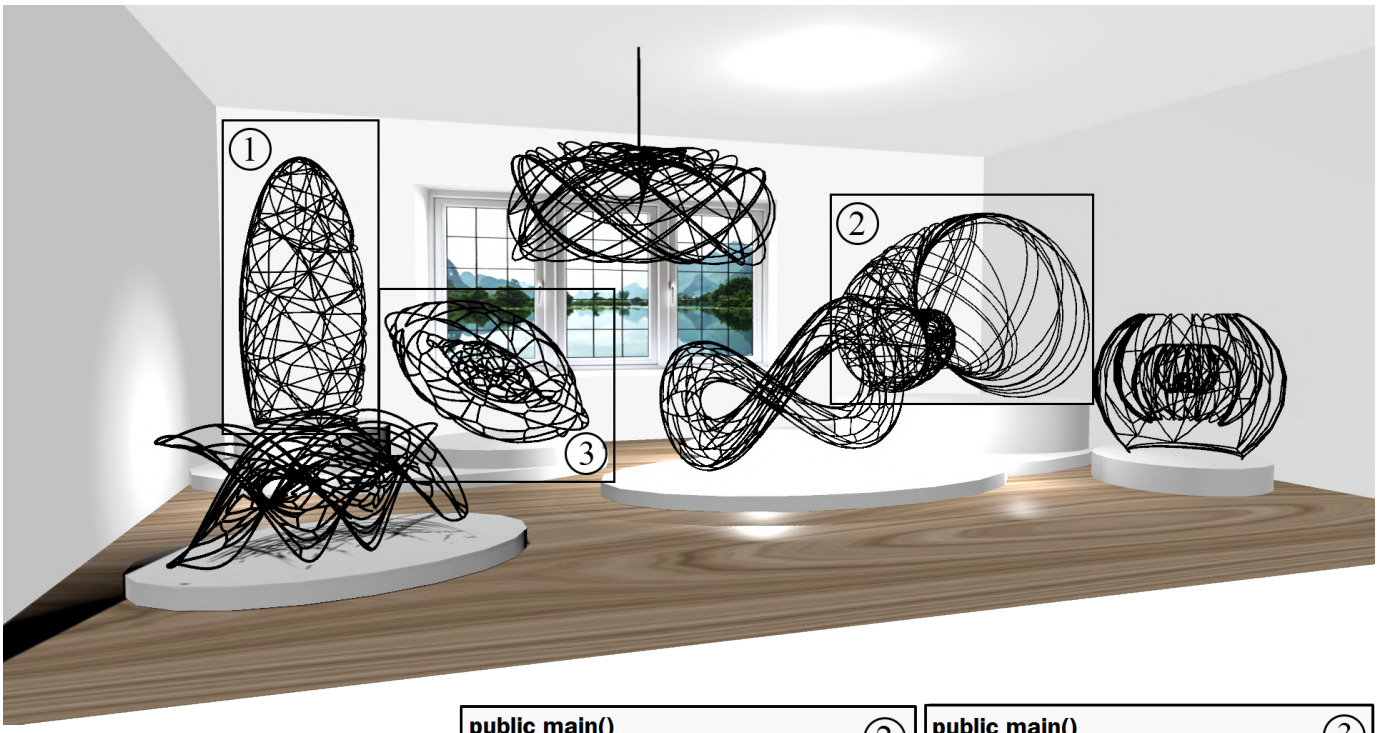
$$\mathbf{g}(u,v) = \mathbf{s}(u,v) + \frac{\mathbf{s}_u(u,v) \times \mathbf{s}_v(u,v)}{||\mathbf{s}_u(u,v) \times \mathbf{s}_v(u,v)||} f(u,v),$$

where given NURBS control points $\mathbf{p}_{i,j}$ and weights $w_{i,j}$ as:

$$\mathbf{s}(u,v) = \frac{\sum_{i=0}^{m} \sum_{j=0}^{n+q} w_{i,j} b_p^i(u) b_q^j(v) \mathbf{p}_{i,j}}{\sum_{i=0}^{m} \sum_{j=0}^{n+q} w_{i,j} b_p^i(u) b_q^j(v)}$$

III. Experiment Results

To assess the effectiveness of PM4WireArt, a group of numerical experiments were conducted on the examination of conceptual wire art designs. The experimentation process leveraged the scriptable parametric modeling interface provided by PM4WireArt which was implemented within the Unity 3D Editor of the 2019 version. The computational tasks were executed on a gaming laptop with an Intel Core i5 CPU, 32 GB of DDR4 RAM, and an NVIDIA GeForce GTX 1650 graphics card with 4 GB of GDDR6 memory. This hardware configuration provides a high-performance computing environment to perform wire art modeling tasks on PM4WireArt.




```
public main()
{
    func(w,v) f=
    [
        H=2.45,R=0.85,Rv=R*cos(v)
    ]
    <Rv*cos(w),H*sin(v),Rv*sin(w)>;
    int N=200, M=50;
    double n=getRange("n");
    setStep(<50, 100, 100>);
    addDelaunayTriangle("D", n*N+M);
    setMaterial("0"); addGraphSurface
    (
        "D",f,0:2.1PI,-0.2PI:0.5PI,0.3
    );
}
```

1

```
public main()
{
    func(u,v) f=[R=1.2*v,r=2.5]
    <
        R*sin(u)*sin(u)*sin(v),
        R*r*sin(u)*cos(u)/2,
        R*sin(u)*sin(u)*cos(v)
    >;
    int N=20, M=10;
    double n=getRange("n");
    setStep(<50, 100, 100>);
    addRandomGraph("G", n*N+M);
    setMaterial("0"); addGraphSurface
    (
        "G",f,0:-2.0PI,0:2.5PI,0.3
    );
}
```

2

```
public main()
{
    func(u,v) f=
    <
        v*sin(u)*cos(v),
        v*sin(u)*sin(v),
        1.5*v*cos(u)
    >;
    int N=200, M=50;
    double n=getRange("n");
    setStep(<50, 100, 100>);
    addVoronoiDiagram("V", n*N+M);
    setMaterial("0"); addGraphSurface
    (
        "V",f,0:2.0PI,0:2.0PI,0.4
    );
}
```

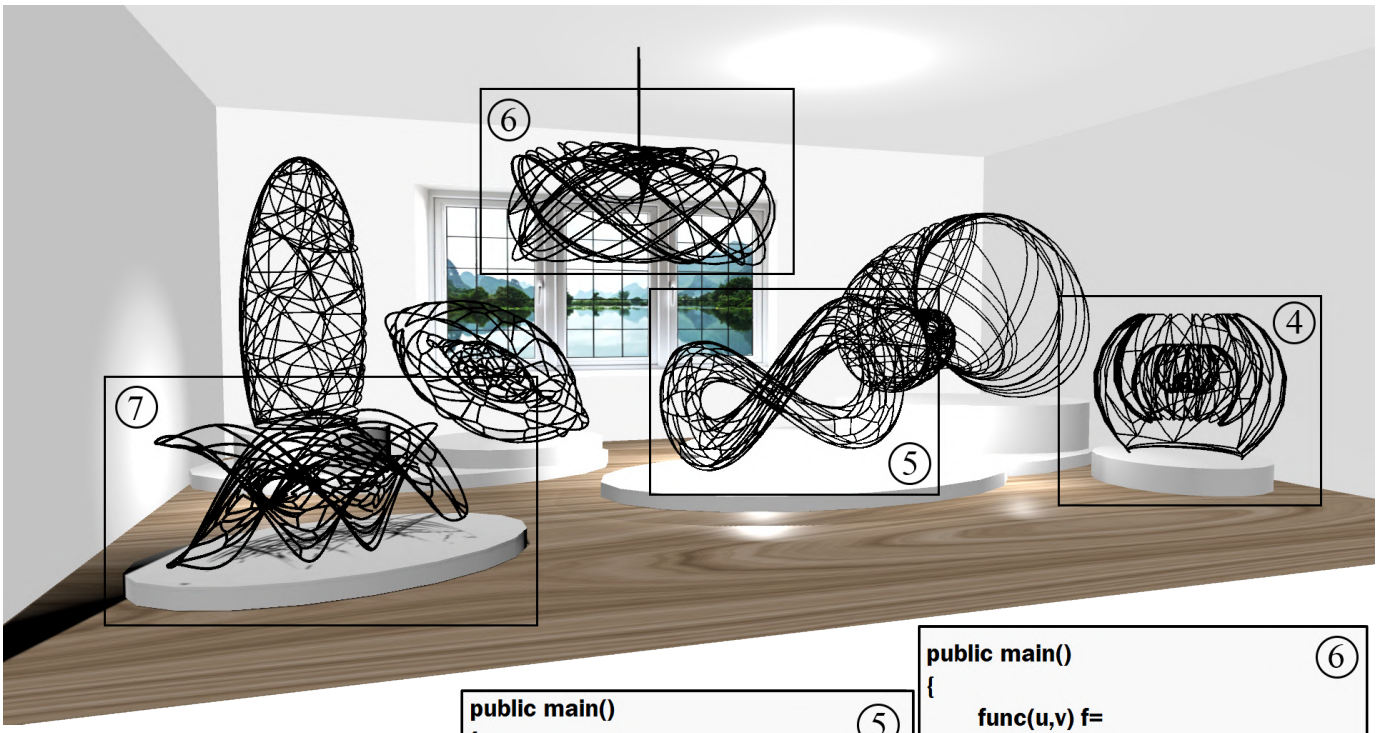
3

Fig. 5. Experiment Results (Part I). This figure shows first three wire art designed with PM4WireArt Java♭ programming.

In Figure 4, a collection of wire art designed with PM4WireArt using NURBS graph surfaces is showcased, featured by varying parametric configurations. Specifically, these conceptual wire art designs are characterized by adding NURBS surfaces of "Rhino Head" as shown in Fig. 2 and setting parameter values of $R = 0.5$ and $r = 0.5$. The illustration comprises three distinct rows, each corresponding to a different graph structure: Random Graph, Delaunay Triangle, and Voronoi Diagram, respectively. Furthermore, the figure is organized into five columns, each aligned with a unique design parametric function as detailed in Fig. 3. The combination of different graph structures and parametric functions yields a matrix of outputs in the realm of wire art design. This visual representation facilitates a comprehensive analysis of the interplay between graph structures and parametric functions.

Fig. 5-6 display the experimental outcomes of seven wire art designs created with PM4WireArt using graph surfaces. Within a virtual lobby setting, these seven wire art designs are labeled numerically from 1 to 7, depicted in the respective subfigures with their associated Java♭ script, offering a detailed insight into the scripting details for each wire art design.

Figure 7 illustrates a user refining the parameter values of a wire art design through the PM4WireArt interface within the virtual environment of a wire art show lobby. The immersive quality of VR enhances the user experience by enabling interaction with design parameters in virtual space, offering a dynamic and intuitive platform for real-time adjustments of the parameters. For a more in-depth exploration of the experiment process, a video showcasing the interactive process is available at the following link: https://youtu.be/TVUidRADdX8.




```
public main()                                                    4
{
    func(u,v) f=
    <
        u*cos(cos(u))*cos(v),
        u*cos(cos(u))*sin(v),
        u*sin(cos(u))
    >;
    int N=200, M=50;
    double n=getRange("n");
    setStep(<50, 100, 100>);
    addDelaunayTriangle("D", n*N+M);
    setMaterial("0"); addGraphSurface
    (
        "D",f,0:5.0PI,0:1.2PI,0.3
    );
}
```

```
public main()                                                    5
{
    func(u,v) f=
    [
        r=0.2,w=1.0,
        c=1.0+r*cos(u),
        Y=r*sin(u), Z=c*sin(v),
        x=c*cos(v), a=x+w*0.5PI,
        y=Y*cos(a)-Z*sin(a)
    ]
    <x,y,Y*sin(a)+Z*cos(a)>;
    int N=200, M=50;
    double n=getRange("n");
    setStep(<50, 100, 100>);
    addVoronoiDiagram("V", n*N+M);
    setMaterial("0"); addGraphSurface
    (
        "V",f,0:-2.0PI,0:2.0PI,0.3
    );
}
```

```
public main()                                                    6
{
    func(u,v) f=
    [
        R=getRange("R"),
        r=getRange("r"),
        n=5.0, p=2.0*n-1.0,
        k=R*v^p/p
    ]
    <
        4.0*r*v^n*cos(n*u)/n,
        v*sin(v)+k*sin(p*u)-k/R,
        v*cos(v)-k*cos(p*u)
    >;
    int N=20, M=10;
    double n=getRange("n");
    setStep(<500, 50, 200>);
    addDelaunayTriangle("D", n*N+M);
    setMaterial("0"); addGraphSurface
    (
        "D",f,0:-2.0PI,0:1.3,0.3
    );
}
```

```
public main()                                                    7
{
    func(u,v) f=
    [
        R=getRange("R")*6, r=getRange("r")*4, c=2.0*ch(v/2), d=c*sin(u), w=0.1,
        x=R*2*c*cos(u), k=10.0, s=0.05, p=sin(w*x+k), q=cos(w*x+k), m=2.0*v*q, h=2.0*v*p
    ]
    <x, r*(d*p+m), r*(d*q-h)>*s; int N=200, M=50; double n=getRange("n"); setStep(<500, 10, 200>);
    addVoronoiDiagram("V", n*N+M); setMaterial("0"); addGraphSurface("V",f,-PI:PI,-PI:PI,0.3);
}
```

Fig. 6. Experiment Results (Part II). This figure shows last four wire art designed with PM4WireArt Java[♭] programming.

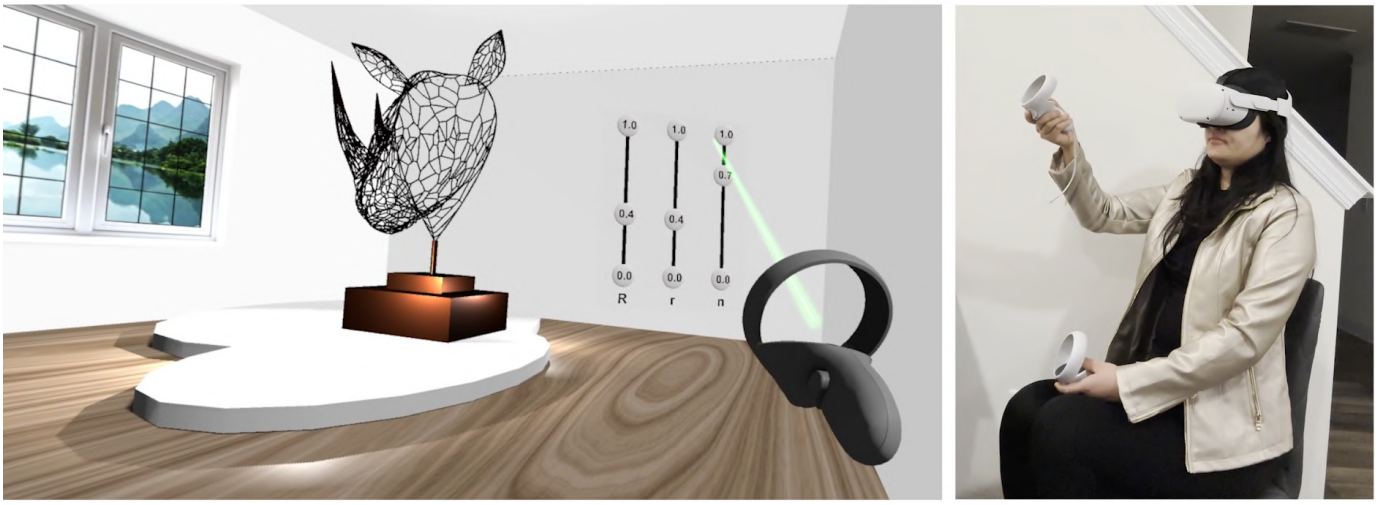


Fig. 7. User Study. This figure shows a user refining the parameter values of a wire art design using PM4WireArt.

IV. Conclusion

This paper introduces PM4WireArt, a novel scriptable parametric modeling interface for conceptual wire art design. The essence of PM4WireArt lies in its commitment to ushering in a paradigm shift within the creative processes of wire artistry by integrating scriptable parametric modeling into the artistic design process. By integrating this design approach with the immersive features of PM4VR, PM4WireArt not only enhances the creative experience but also opens up unexplored avenues for artists to delve into and innovate within the realm of wire art design. PM4WireArt's integration of scripting, real-time visualization, and virtual reality not only streamlines the design process but also provides an interactive dimension to the wire art creation process, offering a dynamic platform where the traditional wire art craft meets the limitless possibilities of contemporary wire art design.

References

[1] T. Green, "Wire sculpture," *Forces*, vol. 2008, no. 1, p. 44, 2008.

[2] L. Slobodkin, *Sculpture: Principles and practice*. Courier Corporation, 1973.

[3] J. C. Rich, *The materials and methods of sculpture*. Courier Corporation, 1988.

[4] H. L. Oubre, "The art of wire sculpture," *Design*, vol. 69, no. 5, pp. 17–18, 1968.

[5] K. ALLEN and U. O. C. A. DENVER, "Sculptor of tradition," 2006.

[6] F. Zhang, I. Campbell, and I. Graham, "Application of additive manufacturing to fine art sculpture," 2011.

[7] B. A. McGuire, *Wire in Design: Modern Wire Art & Mixed Media*. Penguin, 2002.

[8] G. Zhu, "Application of digital technology in future sculpture teaching," in *International Conference on Human-Computer Interaction*. Springer, 2022, pp. 148–159.

[9] C.-K. Yeh, T.-N.-H. Le, Z.-Y. Hou, and T.-Y. Lee, "Generating virtual wire sculptural art from 3d models," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 18, no. 2, pp. 1–23, 2022.

[10] W. Li, "Pen2vr: A smart pen tool interface for wire art design in vr," 2021.

[11] W. Li, "Synthesizing 3d vr sketch using generative adversarial neural network," in *Proceedings of the 2023 7th International Conference on Big Data and Internet of Things*, 2023, pp. 122–128.

[12] Y.-T. Yue, X. Zhang, Y. Yang, G. Ren, Y.-K. Choi, and W. Wang, "Wiredraw: 3d wire sculpturing guided with mixed reality," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 2017, pp. 3693–3704.

[13] L. Liu, D. Ceylan, C. Lin, W. Wang, and N. J. Mitra, "Image-based reconstruction of wire art," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–11, 2017.

[14] K.-W. Hsiao, J.-B. Huang, and H.-K. Chu, "Multi-view wire art." *ACM Trans. Graph.*, vol. 37, no. 6, pp. 242–1, 2018.

[15] Z. Yang, P. Xu, H. Fu, and H. Huang, "Wireroom: model-guided explorative design of abstract wire art," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–13, 2021.

[16] W. Li, "Compare the size: Automatic synthesis of size comparison animation in virtual reality," in *2023 5th International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*. IEEE, 2023, pp. 1–4.

[17] W. Li, "Procedural marine landscape synthesis for swimming exergame in virtual reality," in *2022 IEEE Games, Entertainment, Media Conference (GEM)*. IEEE, 2022, pp. 1–8.

[18] W. Li, "Terrain synthesis for treadmill exergaming in virtual reality," in *2023 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. IEEE, 2023, pp. 263–269.

[19] W. Li, "Elliptical4vr: An interactive exergame authoring tool for personalized elliptical workout experience in vr," in *Proceedings of the 2023 5th International Conference on Image, Video and Signal Processing*, 2023, pp. 111–116.

[20] W. Li, "Planettxt: A text-based planetary system simulation interface for astronomy edutainment," in *Proceedings of the 2023 14th International Conference on E-Education, E-Business, E-Management and E-Learning*, 2023, pp. 47–53.

[21] W. Li, "Insectvr: Simulating crawling insects in virtual reality for biology edutainment," in *Proceedings of the 7th International Conference on Education and Multimedia Technology*, 2023, pp. 8–14.

[22] W. Li, "Synthesizing realistic cracked terrain for virtual arid environment generation," in *2023 8th International Conference on Communication, Image and Signal Processing (CCISP)*. IEEE, 2023, pp. 361–366.

[23] W. Li, "Simulating ice skating experience in virtual reality," in *2022 7th International Conference on Image, Vision and Computing (ICIVC)*. IEEE, 2022, pp. 706–712.

[24] W. Li, "Surfchessvr: Deploying chess game on parametric surface in virtual reality," in *2023 9th International Conference on Virtual Reality (ICVR)*. IEEE, 2023, pp. 171–178.

[25] W. Li, "Pm4vr: A scriptable parametric modeling interface for conceptual architecture design in vr," in *Proceedings of the 18th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry*, 2022, pp. 1–8.

[26] W. Li, "Pm4car: A scriptable parametric modeling interface for concept car design using pm4vr." in *2024 International Conference on Computer Graphics and Image Processing (CGIP)*, 2024, pp. 1–6.

[27] W. Li, "Pm4pottery: A scriptable parametric modeling interface for conceptual pottery design using pm4vr," in *2024 International Conference on Automation, Robotics, and Applications (ICARA)*, 2024, pp. 1–5.

[28] W. Li, "Pm4music: A scriptable parametric modeling interface for music visualizer design using pm4vr," in *2024 International Conference on Digital Signal Processing (ICDSP)*, 2024, pp. 1–5.