PM4Car: A Scriptable Parametric Modeling Interface for Conceptual Car Design Using PM4VR

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

Fig. 1. Teaser. This teaser shows a running example of PM4Car, a scriptable parametric modeling interface designed for conceptual car design using PM4VR. Conceptual car designer write a Java♭ script (left figure) and run it on PM4Car, the corresponding 3D conceptual car model is generated in real-time (middle figure). After connecting PM4Car with Oculus Quest 2 VR headset via SteamVR plugin, designer can tune parameters using VR controllers in virtual environment (right figure).

***Abstract*—This paper introduces PM4Car, a novel scriptable parametric modeling interface tailored for conceptual car design, leveraging the power of PM4VR (Parametric Modeling for Virtual Reality). The integration of parametric modeling techniques with VR environments offers a unique platform for conceptual car designers to explore parametric design concepts. PM4Car aims to enhance the efficiency and creativity of the conceptual car design process by providing a easy-to-use, flexible and intuitive interface for designers to script and manipulate parametric models in an immersive virtual environment.**

***Keywords*—Parametric Modeling, Car Design, Virtual Reality**

I. Introduction

The conceptual car design industry is undergoing a transformative shift with the advent of parametric modeling [1] and virtual reality technologies [2]. Parametric Modeling (PM) [3] is a innovative approach in Computer-Aided Design (CAD) [4] that revolutionizes the way objects are created and manipulated. Unlike traditional modeling techniques, parametric modeling establishes relationships between geometric elements and defines their properties through parameters [5]. These parameters serve as dynamic variables [6], allowing designers to easily modify the shape, size, and other characteristics of a model by adjusting the associated parameters [7]. This not only enhances efficiency and flexibility in the design process but also facilitates the exploration of design iterations [8].

Virtual Reality (VR) [9] is a transformative technology that immerses users in virtual environments, creating a sense of presence and interaction beyond the physical world's confines. By employing VR headsets and controllers [10], users are transported to a simulated reality where they can engage with three-dimensional spaces and objects [11], making VR a powerful tool for applications ranging from edutainment [12]–[14] to training [15]–[19], exercising [20]–[23], and designing [24]–[26]. VR technology has rapidly evolved, providing a new dimension for human-computer interaction and fostering innovative solutions across industries.

With the combination of PM and VR, Parametric Modeling for Virtual Reality (PM4VR) proposed by Li et al. [27] revo-

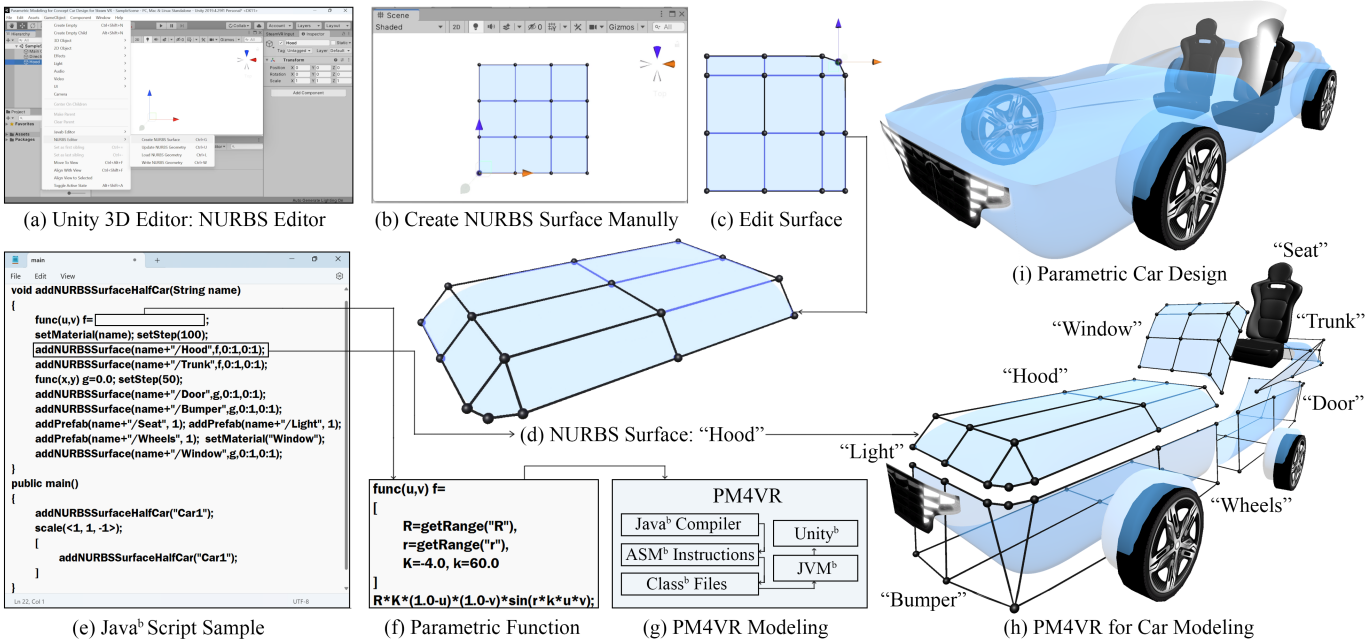


Fig. 2. Overview of our approach.

lutionizes the traditional way of parametric modeling by immersing designers in virtual environments through a scriptable interactive user interface. PM4VR enables designers to create and visualize parametric design iterations in virtual space, fostering innovation and accelerating the design process. However, there is no existing research work that has systematically explored a scriptable parametric modeling interface for conceptual car design. Therefore, given this observation, we propose PM4Car, a novel scripting-based parametric modeling interface integrated with PM4VR, to empower designers in the conceptualization phase of parametric car design.

Fig. 1 showcases our proposed interactive interface of PM4Car, a scriptable parametric modeling interface for conceptual car design using PM4VR. In this demo, a conceptual car designer writes a Java^b script as shown in the left figure. Upon executing this design script within PM4Car, the middle figure displays a real-time generation of a 3D car model. As shown in the right figure, the integration of PM4Car with Oculus Quest 2 VR headset, facilitated by SteamVR plugin, opens up a new dimension of interactive conceptual car design interface that allows designer fine-tuning parameters using VR controllers, fostering an immersive designing experience.

II. Overview

Fig. 2 shows the overview of our approach. As shown in Fig. 2 (a), we develop a novel plugin for Unity3D Editor, called NURBS Editor, that supports four NURBS geometry operations including: (1) Create NURBS Surface (Ctrl+G), (2) Update NURBS Surface (Ctrl+U), (3) Load NURBS Surface (Ctrl+L) and (4) Write NURBS Surface (Ctrl+W). After pressing "Ctrl+G" key, a default NURBS surface (a square quad with 4x4 control points) is generated in the scene as

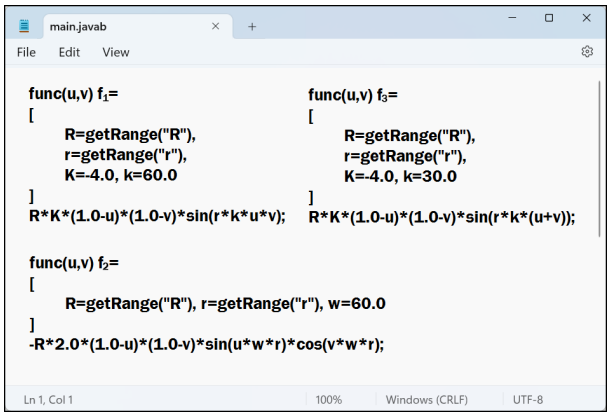
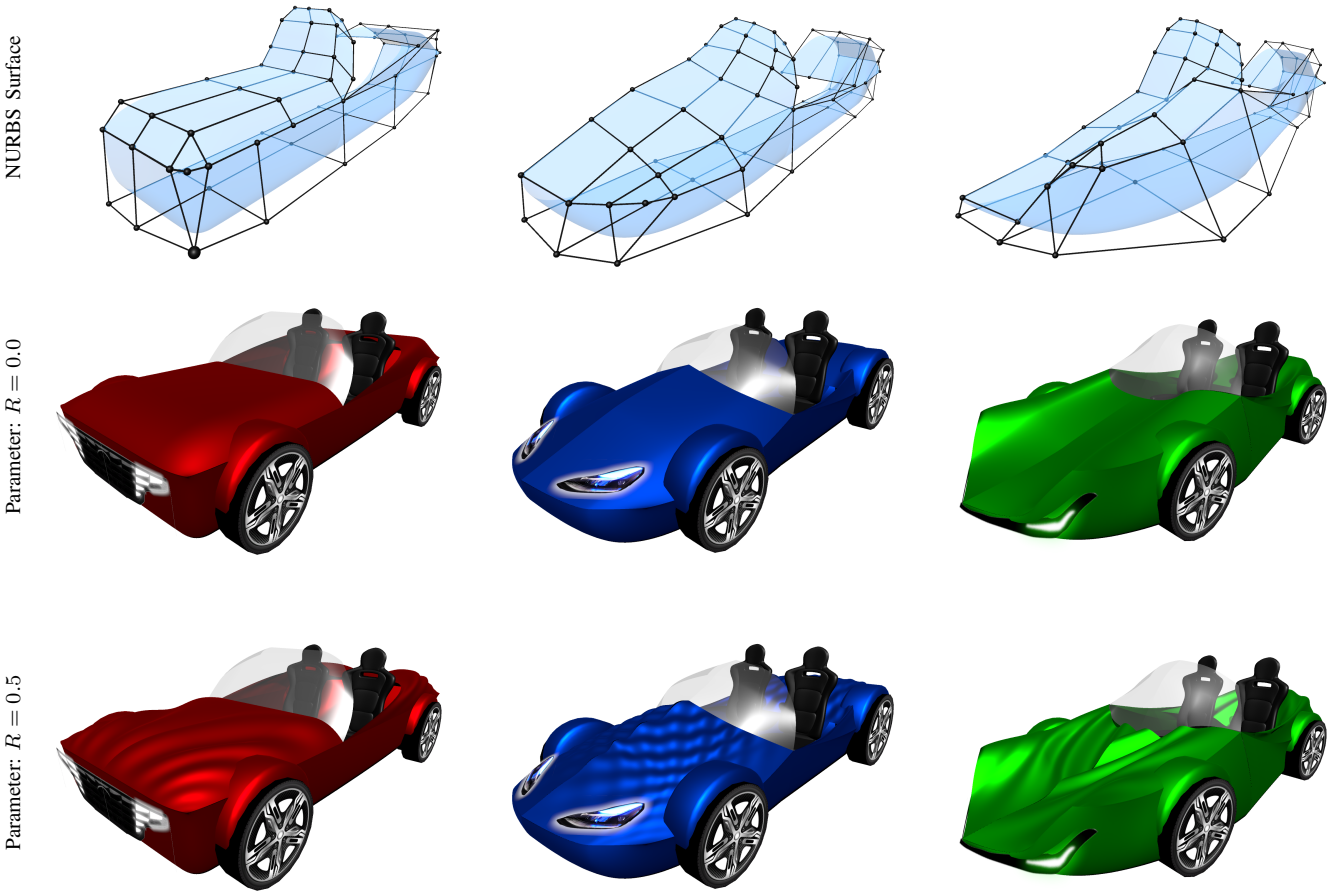


Fig. 3. Parametric Functions (Part I).

shown in Fig. 2 (b). The NURBS Surface can be updated via dragging the control points' positions (or scaling the control points's radius to adjust the weights) followed by pressing "Ctrl+U" key as shown in Fig. 2 (c). After a NURBS Surface is well-designed, for example in Fig. 2 (d), a NURBS Surface named "Hood" is written into a file called "Hood.NURBS" by pressing "Ctrl+W" key; Or, loaded into the scene by pressing "Ctrl+L" key. These operations are implemented by us in the C# scripts of "NURBSGeometry.cs" and "NURBSEditor.cs".

Fig. 2 (e) shows a Java^b script sample for parametric car modeling using PM4Car. For more details about Java^b parametric language programming, please refer to the paper for PM4VR proposed by Li et al. [27]. The parametric function implemented for this example is shown in Fig. 2 (f). After sending this Java^b script into the PM4VR modeling interface as shown in Fig. 2 (g), the conceptual car is synthesized by invoking three parts of the Java^b scripts including (1) Adding



(1) Parametric Function: func$(u, v) = f_1(u, v)$ (2) Parametric Function: func$(u, v) = f_2(u, v)$ (3) Parametric Function: func$(u, v) = f_3(u, v)$

Fig. 4. Different Parametric Configurations. This figure shows parametric conceptual car models designed with PM4Car using different parametric configurations such as NURBS surface designs, parametric functions and parameter values.

NURBS surfaces of "Hood" and "Trunk" which are defined by parametric function "func(u,v) f"; (2) Adding NURBS surfaces of "Door", "Bumper", and "Window" which are defined by parametric function "func(u,v) g = 0"; and (3) Adding prefabs of "Seat", "Wheels" and "Lights" which are hand-made 3D models designed in Unity3D Editor as shown in Fig. 2 (h). In the end, a parametric 3D conceptual car model is automatically generated in real-time as shown in Fig. 2 (i).

III. Technical Approach

To facilitate NURBS (Non-Uniform Rational B-Splines) geometry operations within the Unity3D Editor using PM4Car, we devise a plugin comprising two essential C# scripts: "NURBSEditor.cs." and "NURBSGeometry.cs". The former script, "NURBSEditor.cs," is placed in the "Assets/Editor" directory to ensure its integration into the Unity3D Editor's dropdown menu, complemented by hotkey support for convenient accessibility. This script serves as the interface for users to engage with NURBS functionality efficiently. On the other hand, the "NURBSGeometry.cs" script is designed to be attached as a MonoBehaviour component to a GameObject within the scene. This enables the application of NURBS geometry operations directly to specific game objects, thereby extending the versatility of the plugin for parametric modeling within the Unity3D Editor environment.

NURBS Editor. The plugin incorporates key NURBS geometry operations to streamline the modeling process in Unity3D Editor, featuring: (1) Create NURBS Surface (Ctrl+G), (2) Update NURBS Surface (Ctrl+U), (3) Load NURBS Surface (Ctrl+L), and (4) Write NURBS Surface (Ctrl+W). Initiating "Ctrl+G" generates a default NURBS surface configured as a square quad with 4x4 control points. Users can dynamically refine the NURBS surface by manipulating the positions of control points or adjusting the radius to modify weights, followed by the application of changes through "Ctrl+U." To preserve a well-designed NURBS surface, users can write it to a file using the "Ctrl+W" command or load a pre-existing NURBS surface into the scene with "Ctrl+L." These NURBS operations provide an interactive environment for creating and managing NURBS surfaces within Unity3D Editor.

NURBS Surface. After connecting this manually designed NURBS surfaces with user-specified Java[þ] script via the PM4VR modeling interface, parametric NURBS surface can

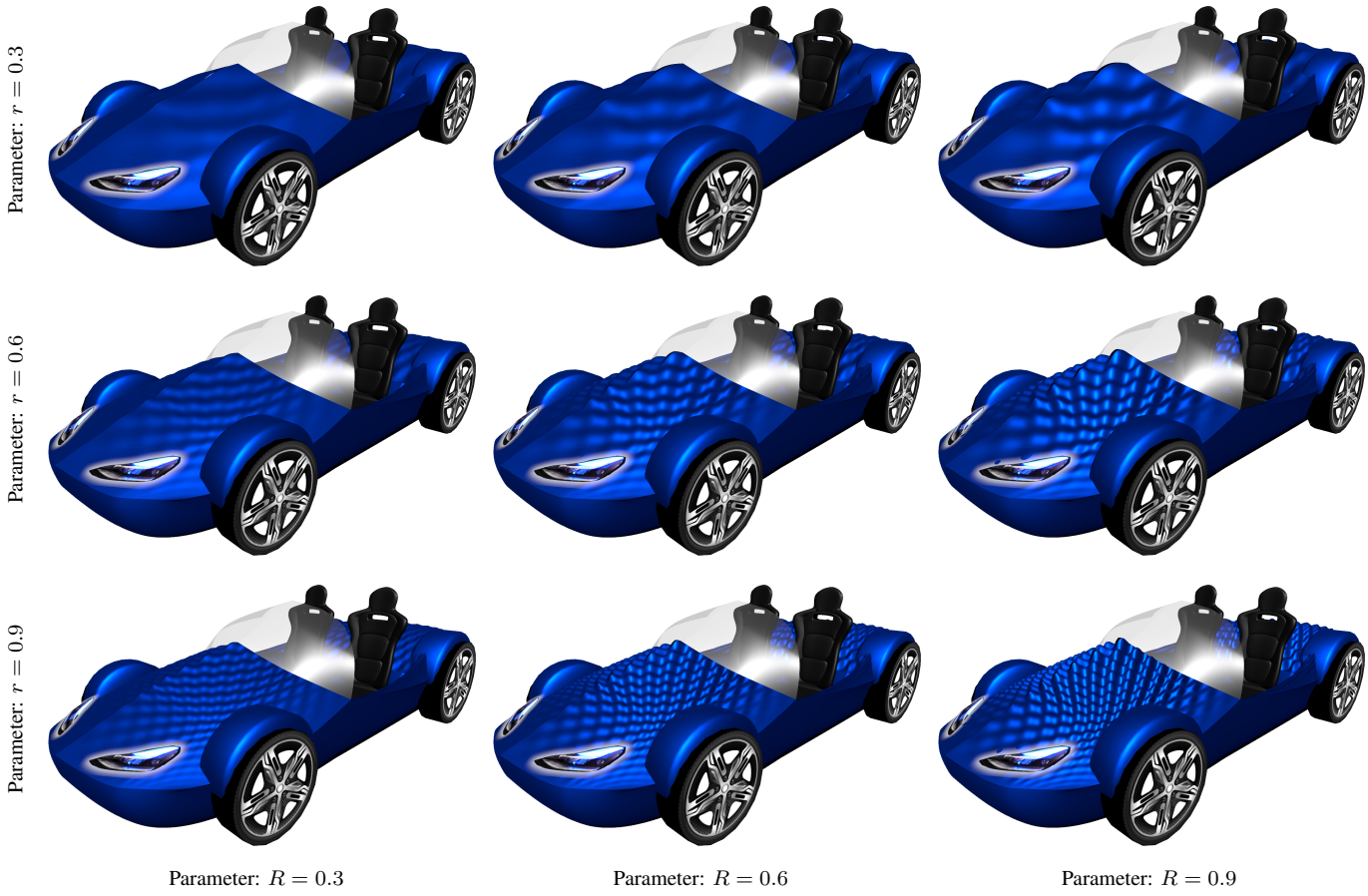


Fig. 5. Different Parameter Values. This figure shows parametric conceptual car models designed with PM4Car using the same NURBS surface design and the parametric function $f_2(u, v)$ under different parameter values of R and r, where $R, r \in [0, 1]$.

be generated automatically using PM4Car interface. Mathematically, for each NURBS surface, the control points' positions are represented as $\{\mathbf{p}_{i,j} | i \in [0, m], j \in [0, n]\}$, and the control weights are the radius of each control point which are represented as $\{w_{i,j} | i \in [0, m], j \in [0, n]\}$, then, the NURBS surface's parametric equation $\mathbf{s}(u, v)$ is represented as:

$$\mathbf{s}(u, v) = \frac{\sum_{i=0}^{m} \sum_{j=0}^{n} w_{i,j} b_p^i(u) b_q^j(v) \mathbf{p}_{i,j}}{\sum_{i=0}^{m} \sum_{j=0}^{n} w_{i,j} b_p^i(u) b_q^j(v)}, \quad (1)$$

where $b_p^i(u)$ is the p^{th} order B-spline basis function of the i^{th} control point. Given knot values $\{0, ..., k_i, k_{i+1}, ..., 1\}$, if $u \in [k_i, k_{i+1}]$, $b_0^i(u) = 1$; Otherwise, $b_0^i(u) = 0$. We have:

$$b_p^i(u) = b_{p-1}^i(u) \frac{u - k_i}{k_{i+p} - k_i} + b_{p-1}^{i+1}(u) \frac{k_{i+1+p} - u}{k_{i+1+p} - k_{i+1}} \quad (2)$$

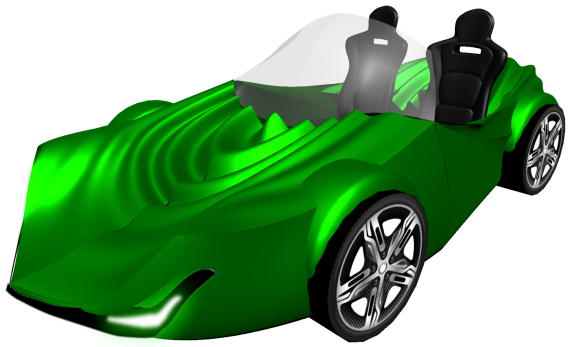
where the knot values are Bezier uniform NURBS knots.

Parametric Function. In the PM4Car interface, in order to parameterize NURBS surface, NURBS surface $\mathbf{s}(u, v)$ is deformed along normal direction at distance defined as parametric function $f(u, v)$. We propose a Java♭ function called **addNURBSSurface**("NURBS name", $\mathbf{f}(u, v)$, $u_0 : u_1$, $v_0 : v_1$). Mathematically, we construct the surface by first loading the NURBS geometry from "¡NURBS name¿.NURBS" file as $\mathbf{s}(u, v)$; Then, deforming NURBS surface along normal direction at (u, v) with distance of parametric function $f(u, v)$. This results in a new parametric surface $\mathbf{g}(u, v)$, mathematically, we have $\mathbf{g}(u, v)$ calculated with the following equation:

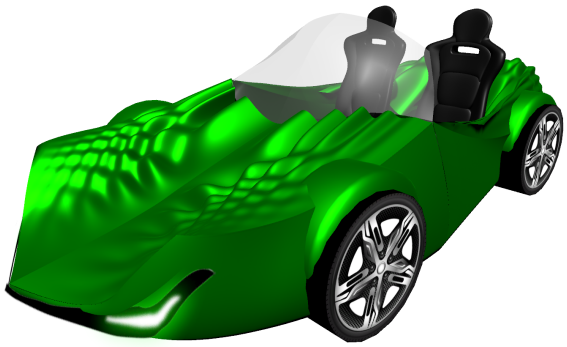
$$\mathbf{g}(u, v) = \mathbf{s}(u, v) + \frac{\mathbf{s}_u(u, v) \times \mathbf{s}_v(u, v)}{||\mathbf{s}_u(u, v) \times \mathbf{s}_v(u, v)||} f(u, v) \quad (3)$$

IV. Experiment Results

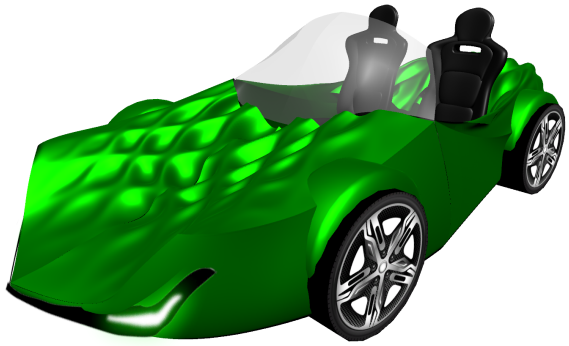
To assess the effectiveness of our proposed technical approach, a series of numerical experiments were undertaken on conceptual car designs using the scriptable parametric modeling interface of PM4Car. The implementation of our proposed approach was executed within Unity 3D with the 2019 version and generated these experiment results with hardware configurations containing Intel Core i5 CPU, 32GB DDR4 RAM, and NVIDIA GeForce GTX 1650 4GB GDDR6 Graphics Card. Figure 4 presents the outcomes of the parametric design process for conceptual car models within PM4Car, incorporating diverse parametric configurations such as NURBS surfaces, parametric functions, and parameter values. The three columns illustrate distinct NURBS surface designs, with the corresponding Java♭ scripts for the parametric functions showcased



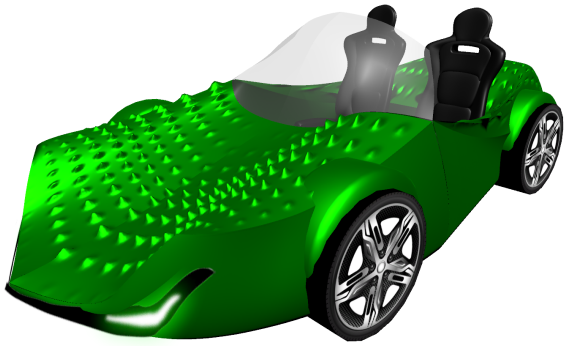
(a) Parametric Function: func$(u, v) = f_a(u, v)$



(b) Parametric Function: func$(u, v) = f_b(u, v)$



(c) Parametric Function: func$(u, v) = f_c(u, v)$



(d) Parametric Function: func$(u, v) = f_d(u, v)$

Fig. 6. Different Parametric Functions. This figure shows parametric conceptual car models designed with PM4Car using the same NURBS surface design and the parameter values of $R = 0.5, r = 0.5$ under different parametric function settings.

in Fig.3. The second and third rows of Fig.4 display the impact of different parameter values, specifically $R = 0.0$ and $R = 0.5$, respectively. Moving forward, Fig.5 exhibits results from PM4Car utilizing the same NURBS surface design and parametric function $f_2(u, v)$ but with varying parameter values of R and r, where $R, r \in 0.3, 0.6, 0.9$. Fig.6 showcases parametric conceptual car models with consistent NURBS surface design and $R = 0.5, r = 0.5$, but employing different parametric functions, detailed in Fig.7. Lastly, Fig.8 captures a designer wearing an Oculus Quest 2 headset, engaged in the immersive experience of fine-tuning conceptual car parameter values within PM4Car's virtual reality environment. For an in-depth exploration, an experiment result video is available at the following link: https://youtu.be/ymAvDBE6tdw.

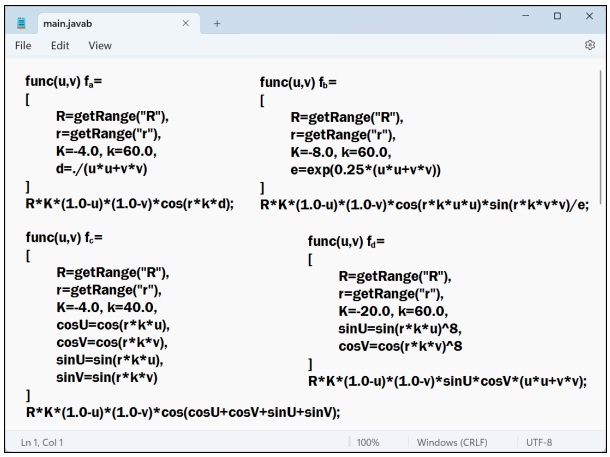


Fig. 7. Parametric Functions (Part II).

V. Conclusion

In this paper, we introduce PM4Car, a scriptable parametric modeling interface tailored for conceptual car design. Leveraging the capabilities of PM4VR, an innovative virtual reality (VR) extension, PM4Car provides a dynamic platform for designers to create and refine conceptual car models through a script-driven approach. The interface incorporates diverse parametric configurations, including NURBS surface designs, scriptable parametric functions, and customizable parameter values. Through PM4Car, designers can engage in an intuitive design experience within a virtual environment.

As shown from the experiment results, PM4Car stands as a cutting-edge scriptable parametric modeling interface for conceptual car design, offering a balance between precision and creativity. Its integration with PM4VR elevates the design experience, providing designers with a versatile and immersive platform for crafting innovative car models. This paper outlines the efficacy of PM4Car in the context of conceptual car design and sets the stage for future advancements in scriptable parametric modeling interfaces.

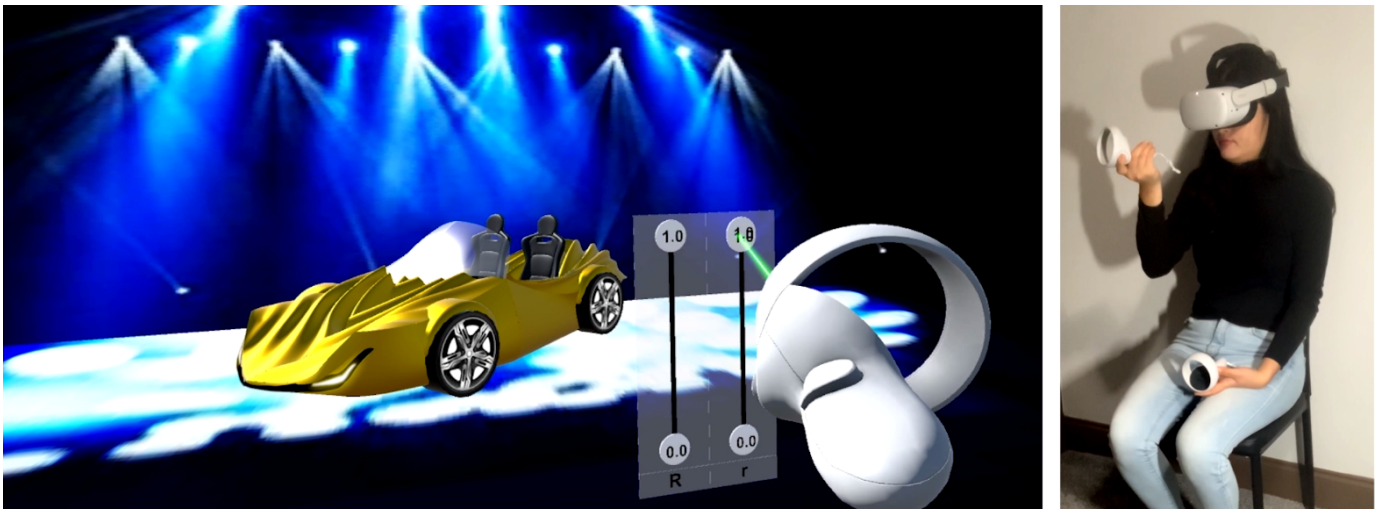


Fig. 8. User Study. This figure shows a user fine-tuning the conceptual car's parameter values using PM4Car in virtual reality.

REFERENCES

[1] N. Wang, J. Wan, and G. Gomez-Levi, "Parametric method for applications in vehicle design," SAE Technical Paper, Tech. Rep., 2005.

[2] P. Zimmermann, "Virtual reality aided design. a survey of the use of vr in automotive industry," in *Product engineering: tools and methods based on virtual reality*. Springer, 2008, pp. 277–296.

[3] K. I Kyivska, S. V Tsiutsiura, M. I Tsiutsiura, O. V Kryvoruchko, A. V. Yerukaiev, and V. V Hots, "A study of the concept of parametric modeling of construction objects," *Tsiutsiura, Svitlana and I. Tsiutsiura, Mikola and V. Kryvoruchko, Olena and Yerukaiev, Andrii V. and V. Hots, Vladyslav, A Study of the Concept of Parametric Modeling of Construction Objects*, pp. 636–646, 2019.

[4] Q.-H. Wang, J.-R. Li, B.-L. Wu, and X.-M. Zhang, "Live parametric design modifications in cad-linked virtual environment," *The International Journal of Advanced Manufacturing Technology*, vol. 50, pp. 859–869, 2010.

[5] M. Stavric and O. Marina, "Parametric modeling for advanced architecture," *International journal of applied mathematics and informatics*, vol. 5, no. 1, pp. 9–16, 2011.

[6] M. Turrin, P. Von Buelow, and R. Stouffs, "Design explorations of performance driven geometry in architectural design using parametric modeling and genetic algorithms," *Advanced Engineering Informatics*, vol. 25, no. 4, pp. 656–675, 2011.

[7] L. Avallone, G. Monacelli, F. Pasetti, F. Giardina, and F. A. SpA, "Parametric design methods for car body design," in *XII ADM International Conference, Rimini, Italy-Sept. 5th-7th*, 2001.

[8] J. Wan, N. Wang, and G. Gomez-Levi, "Parametric modeling method for conceptual vehicle design," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 47403, 2005, pp. 403–410.

[9] C. Anthes, R. J. García-Hernández, M. Wiedemann, and D. Kranzlmüller, "State of the art of virtual reality technology," in *2016 IEEE aerospace conference*. IEEE, 2016, pp. 1–19.

[10] D. K. Chen, J.-B. Chossat, and P. B. Shull, "Haptivec: Presenting haptic feedback vectors in handheld controllers using embedded tactile pin arrays," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1–11.

[11] H. Benko, C. Holz, M. Sinclair, and E. Ofek, "Normaltouch and texturetouch: High-fidelity 3d haptic shape rendering on handheld virtual reality controllers," in *Proceedings of the 29th annual symposium on user interface software and technology*, 2016, pp. 717–728.

[12] 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.

[13] W. Li, "Creative molecular model design for chemistry edutainment," in *Proceedings of the 14th International Conference on Education Technology and Computers*, 2022, pp. 226–232.

[14] W. Li, "Insectvr: Simulating crawling insects in virtual reality for biology edutainment," ser. ICEMT '23. New York, NY, USA: Association for Computing Machinery, 2023, pp. 1–7. [Online]. Available: https://doi.org/10.1145/3625704.3625757

[15] W. Li, H. Huang, T. Solomon, B. Esmaeili, and L.-F. Yu, "Synthesizing personalized construction safety training scenarios for vr training," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 5, pp. 1993–2002, 2022.

[16] G. Avveduto, C. Tanca, C. Lorenzini, F. Tecchia, M. Carrozzino, and M. Bergamasco, "Safety training using virtual reality: A comparative approach," in *Augmented Reality, Virtual Reality, and Computer Graphics: 4th International Conference, AVR 2017, Ugento, Italy, June 12-15, 2017, Proceedings, Part I 4*. Springer, 2017, pp. 148–163.

[17] W. Li, B. Esmaeili, and L.-F. Yu, "Simulating wind tower construction process for virtual construction safety training and active learning," in *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. IEEE, 2022, pp. 369–372.

[18] D. C. Schwebel, T. Combs, D. Rodriguez, J. Severson, and V. Sisiopiku, "Community-based pedestrian safety training in virtual reality: A pragmatic trial," *Accident Analysis & Prevention*, vol. 86, pp. 9–15, 2016.

[19] W. Li, J. Talavera, A. G. Samayoa, J.-M. Lien, and L.-F. Yu, "Automatic synthesis of virtual wheelchair training scenarios," in *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 2020, pp. 539–547.

[20] 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.

[21] W. Li, B. Xie, Y. Zhang, W. Meiss, H. Huang, and L.-F. Yu, "Exertion-aware path generation." *ACM Trans. Graph.*, vol. 39, no. 4, p. 115, 2020.

[22] 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.

[23] 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.

[24] Q. Liu, "The virtual reality technology in art design," in *2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)*. IEEE, 2012, pp. 2226–2228.

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

[26] 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.

[27] 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.