From Scenario Models to Test Cases: Automated System Testing for VR Applications

Gerry Longfils[1], Xavier Devroey[1]

[1]NADI,University of Namur, Rue de Bruxelles,61, B5000, Namur, Belgium

Abstract

Virtual Reality (VR) applications are increasingly used in industrial cases, yet adoption remains limited by the lack of effective, systematic, and reproducible VR-tailored testing. We present UltraInstinctVR, an approach that automates the generation and execution of system-level VR tests. Across ten open-source applications, it outperformed state-of-the-art tools in unique failure detection, and a follow-up developer study showed strong efficiency and dependability. The tool is available at https://github.com/snail-unamur/UltraInstinctVR, with a demonstration video at https://youtu.be/U8DWTmXmeh4.

Keywords

Virtual Reality, Software Testing, Model-Based Testing, Petri Nets

1. Introduction

Virtual Reality (VR) applications let users engage with immersive, computer-generated worlds [1]. VR software engineering practices remain immature, especially for testing [2]: interactions are natural (gesture, voice, gaze) rather than keyboard/mouse-based, and existing tools mostly detect crashes, leaving VR testing largely manual or blind to VR-specific faults [3]. We address this gap with UltraInstinctVR, a Model-Based Testing approach generating and executing VR system tests for *selection*, *locomotion*, and *manipulation* [4]. It reuses existing VR scenario models [5] typically Petri nets, as test oracles: a tester attaches assertions to scenario transitions, from which UltraInstinctVR derives reusable, re-executable test cases for fully automated functional testing.

2. Background

Andrade et al. [6] document the lack of systematic VR testing and propose a taxonomy of VR faults (*e.g.,* performance, networking, stability), identifying those most impactful for users and developers. Model-Based Testing automates test generation but is constrained by model construction cost; VR scenario-based approaches mitigate this by reusing formal models (*e.g.,* Petri nets) as both test generators and oracles, as in Xareus [7] for complex, collaborative interactions. Closer to our work, XRIntTest [8] and VRExplorer [9] generate interaction sequences to improve coverage but detect mostly runtime exceptions rather than functional faults, motivating a model-based approach coupling interaction generation with correctness verification.

3. Published contribution

UltraInstinctVR [10, 11] is a model-based testing approach, implemented as a Unity package built on *Xareus* [7], that automatically generates and executes VR system tests. It leverages VR scenario models using Petri nets as test oracles: a tester defines automated test agents that perform interactions in the scene (*e.g.,* teleporting, grabbing objects), and associates each with a Petri net whose transitions

BENEVOL2026: Belgium-Netherlands Software Evolution Workshop 26–27 November, 2026, Madrid, Spain
gerry.longfils@unamur.be (G. Longfils); xavier.devroey@unamur.be (X. Devroey)
https://www.unamur.be/fr/profil/glongfil (G. Longfils); http://xdevroey.be (X. Devroey)
0009-0005-3286-0720 (G. Longfils); 0000-0002-0831-7606 (X. Devroey)
© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

are verified at runtime via paired *Sensor*/*Effector* components from VR scenario models, decoupling interaction execution from result verification. Evaluated against state-of-the-art VR testing tools across ten open-source projects, UltraInstinctVR detected the most unique failures overall, owing to its oracle-based design [11]; a follow-up developer study using the *User Experience Questionnaire (UEQ)* [12], where ten participants used UltraInstinctVR to identify and fix seeded defects in a VR application built for this purpose, showed positive efficiency/dependability, neutral for remaining dimensions, and 62.85%/47.14% of defects identified/fixed. Tool paper and preprint are available at https://doi.org/10.1145/3832783.3834642 and https://arxiv.org/pdf/2605.07534.

4. Conclusion & Future Work

Our research introduces UltraInstinctVR, an automated framework generating and executing system-level tests for interaction-based VR behaviors. User evaluations show it effectively supports defect identification, though remediation support remains limited; UEQ results highlight strengths in efficiency and dependability, with room for improvement in stimulation, attractiveness, and clarity. In the near term future works, an eDSL will let testers define more complex, multimodal interactions; in the longer term future works, we aim to model XR interaction complexity for effective test generation, aligning with three broader gaps in XR testing: the lack of fault models and a resolved oracle problem (**G1**), the difficulty of specifying multimodal interactions (**G2**), and the shortage of open-source projects/dataset to understand real-world XR faults (**G3**).

References

[1] S. Agrawal, A. Simon, S. Bech, K. Bærentsen, S. Forchhammer, Defining immersion: Literature review and implications for research on immersive audiovisual experiences, AES (2019). doi:10.17743/aesconv.2019.978-1-942220-31-2.

[2] S. A. Andrade, F. L. Nunes, M. E. Delamaro, Towards the systematic testing of virtual reality programs, in: SVR, IEEE, 2019. doi:10.1109/SVR.2019.00044.

[3] D. E. Rzig, N. Iqbal, I. Attisano, X. Qin, F. Hassan, Virtual reality (vr) automated testing in the wild: A case study on unity-based vr applications, in: ISSTA, 2023. doi:10.1145/3597926.3598134.

[4] B. Spittle, M. Frutos-Pascual, C. Creed, I. Williams, A review of interaction techniques for immersive environments, IEEE TVCG (2022). doi:10.1109/TVCG.2022.3174805.

[5] G. Claude, V. Gouranton, R. B. Berthelot, B. Arnaldi, Short paper:# seven, a sensor effector based scenarios model for driving collaborative virtual environment, 2014. Https://hal.science/hal-01086237.

[6] S. A. Andrade, A. J. U. Quevedo, F. L. Nunes, M. E. Delamaro, Understanding vr software testing needs from stakeholders' points of view, in: SVR, IEEE, 2020. doi:10.1109/SVR51698.2020.00024.

[7] L. Gramoli, F. Nouviale, A. Reuzeau, A. Audinot, M. Risy, T. Marchand-Guerniou, M. Mavromatis, B. Arnaldi, V. Gouranton, Xareus: a framework to create interactive applications without coding, in: IEEE VRW, 2025. doi:10.1109/VRW66409.2025.00474.

[8] R. Gu, J. M. Rojas, D. Shin, Xrinttest: An automated framework for user interaction testing in extended reality applications, in: ASE, IEEE, 2025. doi:10.1109/ASE63991.2025.00363.

[9] Z. Zhu, H.-N. Dai, H. Guo, Z. Liao, Z. Zheng, Vrexplorer: A model-based approach for semi-automated testing of virtual reality scenes, in: ASE, IEEE, 2025. doi:10.1109/ASE63991.2025.00047.

[10] G. Longfils, M. Cauz, A. Blouin, X. Devroey, UltrainstinctVR: A model-based tool for automated system testing of vr applications (2026). doi:10.1145/3832783.3834642.

[11] G. Longfils, M. Cauz, A. Blouin, X. Devroey, System test generation for virtual reality applications using scenario models, ArXiv preprint (2026). URL: https://arxiv.org/abs/2605.07534.

[12] U. team, A fast and reliable questionnaire to measure the user experience of interactive products., https://www.ueq-online.org/, 2025.