Non-functional requirements in 3D Mobile Applications and Virtual Reality Environments

Sebastián Dapoto[0000-0001-7593-0198], Federico Cristina[0000-0003-3838-417X], Pablo Thomas [0000-0001-9861-987X] and Patricia Pesado[0000-0003-0000-3482]

III-IDI, Facultad de Informática, UNLP. La Plata. Calle 50 y 120, Buenos Aires. Argentina.
{sdapoto, fcristina, pthomas, ppesado}@lidi.info.unlp.edu.ar

Abstract. Non-functional requirements (NFR) are characteristics that specify the quality criteria and constraints a system must meet. These requirements are essential for the successful development of software, particularly in 3D mobile applications, as they have a direct impact on user experience. Graphics performance is one of the most critical NFR in this type of application, as it affects both user experience and the viability of these applications on devices with varying hardware capabilities. This paper examines the impact of NFR on 3D mobile applications, particularly focusing on the main variables influencing graphics performance, such as 3D modeling, texture usage, lighting, visual effects, and physical effects. Additionally, an experiment is presented on 3D mobile applications with and without virtual reality environments, where the impact of different features on the frames per second rate is evaluated. The results obtained help identify the main bottlenecks in one of the most critical non-functional requirements, contributing to optimizing performance and improving the overall quality of 3D mobile applications and virtual reality environments.

Keywords: Non-functional requirements, 3D mobile applications, virtual reality, graphics performance, Unity.

1 Introduction

Mobile devices have experienced considerable growth and sophistication in recent times, showing significant improvements in terms of speed and efficiency. Multicore architectures and advancements in chip design have enabled greater processing capacity, facilitating the execution of more complex tasks and intensive computations. Additionally, the amount of RAM available in these devices has increased significantly. This allows applications to access more data and perform larger operations, especially in scenarios involving intensive manipulation of graphical resources.

Technological evolution has enabled the execution of increasingly complex applications with higher hardware demands. At the same time, the increase in mobile device screen resolutions has enhanced visual quality, offering more detailed and realistic representations. As a result, 3D mobile applications have gained relevance and popularity. Currently, numerous game engines, libraries, and frameworks enable the development

Proceedings of the 28th Workshop on Requirements Engineering (WER2025),
August 20-22, 2025, Rio de Janeiro-RJ, Brazil.
https://doi.org/10.29327/1588952.28-14

of 3D applications for mobile devices, including support for virtual reality (VR) and augmented reality (AR) [1][2].

Furthermore, the growing use of mobile applications in almost every aspect of modern life makes application quality an increasingly relevant factor. Quality testing in mobile applications, particularly those utilizing 3D environments, presents new challenges, most of which are related to non-functional requirements (NFR) [3] [4].

Among the most important non-functional requirements for a 3D mobile application, we can mention graphical performance, efficient loading of 3D models and textures, energy consumption optimization, intuitive touch interaction, portability, among others. In particular, considering that many potential users of 3D mobile applications may not have the latest-generation devices, it is crucial to conduct an in-depth analysis of the parameters affecting the final graphical performance of a 3D mobile application [5] [6].

This paper presents a study on the impact of NFR on 3D mobile applications, with a particular focus on graphical performance. An experiment is conducted on 3D mobile applications with and without virtual reality environments, evaluating the impact of different characteristics on the frame rate per second (FPS). The results obtained help identify the most relevant aspects to consider in order to optimize performance and improve the final quality of 3D mobile applications and virtual reality environments.

The rest of the paper is organized as follows: Section 2 presents the relevant non-functional requirements in 3D mobile applications, highlighting graphical performance and its impact on user experience. Section 3 describes the methodology used to evaluate graphical performance, detailing the selected criteria and metrics. Section 4 introduces the prototypes developed for the corresponding evaluation. Section 5 presents the experiments conducted and the results obtained. Section 6 discusses the findings in depth, focusing on the interplay between non-functional requirements and outlining the main limitations of the study. Finally, Section 7 provides conclusions and suggests possible future research directions.

2 Non-functional requirements in 3D mobile applications

Non-functional requirements (NFR) are those requirements that do not specifically refer to a system's functionality but impose constraints on the product being developed and the development process itself. In other words, NFR describe aspects of a system's behavior, capturing the properties and constraints under which the system must operate. NFR are just as important as functional requirements (FR) in software development. While FRs describe what a system must do, NFR focus on how it should do it and the characteristics that determine its performance, usability, reliability, efficiency, maintainability, security, among others, ensuring a high level of quality.

On the other hand, 3D applications have gained increasing relevance and popularity. These applications not only provide immersive and engaging experiences for users but also present unique challenges in terms of non-functional requirements.

NFR play a fundamental role in the successful development of 3D mobile applications, as they directly affect the user experience. Table 1 lists some of the most

important NFR in 3D mobile applications. Beyond the importance of complying with all the NFR described in the table, one particularly critical requirement in 3D mobile applications is graphical performance [7].

In 3D mobile applications, graphical performance refers to the ability to render and display graphics in real-time smoothly and without issues. This is essential for providing an immersive and visually appealing experience to users. Poor graphical performance can lead to slow or choppy visualization of 3D elements, negatively affecting the quality of the experience and user satisfaction.

3D mobile applications often require intensive graphical processing due to the complexity of the models and visual effects that must be rendered in real-time. These elements include shadows, textures, dynamic lighting, particle systems, and complex animations.

Graphical performance has a direct impact on user experience. Users expect a seamless and lag-free experience in 3D mobile applications, especially in games, virtual reality or augmented reality applications, and interactive visualizations. Poor graphical performance can cause frustration, reduce immersion, affect the overall perception of quality, and discourage application usage.

Table 1. Most important NFR in 3D mobile applications.

NFR Name	Description
Graphical Performance	It is crucial in 3D mobile applications to ensure a smooth and uninterrupted experience. It includes real-time graphics fluidity, fast rendering capability, optimized resource loading, and quick response to user interactions.
Energy Efficiency	3D mobile applications can be resource-intensive and consume significant power, making it essential to optimize battery consumption and minimize its impact on battery life. This involves efficient energy management and optimization of hardware and software resources.
Stability and Reliability	These are essential to ensure that the application runs correctly without crashes or unexpected failures. This includes proper memory management, bottleneck prevention, error detection and handling, and resilience to unstable network conditions.
Intuitive User Interface	The user interface should be intuitive, easy to use, and adapted to the capabilities and limitations of mobile devices. This involves ergonomic design, intuitive controls, and a clear navigation flow.
Security	Security is a critical requirement in any mobile application, including 3D applications. It is necessary to protect user data, secure transactions, and ensure software integrity to prevent threats such as data theft or unauthorized access.
Portability	3D mobile applications should be compatible with a wide range of mobile devices, operating systems, and versions, ensuring that the application functions consistently across different devices and platforms.

If a developer wants their 3D mobile applications to be used by the largest possible number of users, they must strive to optimize graphical performance as much as possible. This becomes even more relevant considering that many potential users of 3D mobile applications may not have the latest-generation devices. Applications should be designed to run smoothly on as many devices as possible.

Moreover, inefficient graphical performance can negatively impact the overall performance of the mobile device. Excessive resource consumption can cause the application to slow down and lead to delayed responses in other device functions. This can negatively affect the user experience and reduce overall satisfaction with the application, potentially leading users to decrease their usage or even uninstall the application entirely.

For these reasons, this study focuses specifically on graphical performance as a fundamental non-functional requirement, given its direct influence on the quality of the visual experience, user satisfaction, and the overall performance of the mobile device.

3 Evaluation methodology

While profiling tools can be useful for analyzing the graphical performance of a 3D mobile application, they also present several limitations. The main aspect to consider is the accuracy of the data obtained through these tools. Generally, the values lack acceptable precision and often vary depending on the specific tool being used.

Additionally, when using such tools in virtual environments or emulators, the accuracy of the results is further affected. This is mainly because these environments cannot fully replicate the behavior and limitations of the physical hardware of mobile devices. If certain hardware-specific characteristics and behaviors are not adequately captured in the virtual environment, this affects the accuracy of performance measurements.

Furthermore, profiling tools can consume system resources and impact the overall performance of the application during execution. This can lead to inaccurate performance results, as the tool itself introduces additional load on the system and affects the application's behavior.

To achieve the highest possible accuracy in the results, an independent set of tests is proposed to be conducted on real mobile devices, evaluating the main characteristics of a 3D mobile application and analyzing their impact on graphical performance.

The proposed evaluation consists of isolating each of the key characteristics involved in a 3D mobile application, particularly those that have a direct impact on the device's processing load. These characteristics include aspects such as the number of polygons, the use of lighting and shadows, the application of textures and/or transparencies, the rendering of particle systems, and the physics calculations of objects composing a scene in a 3D application.

While the impact of these characteristics on graphical performance may vary depending on the software and hardware on which the application is executed, the goal is to identify common patterns regarding performance degradation in relation to increasing visual requirements.

Table 2 lists the independent tests designed to evaluate graphical performance. In most tests, FPS is recorded throughout the simulation based on the number of objects displayed on the screen. The only exception is the test called Complex Mesh Rendering, where FPS is recorded throughout the simulation based on the rendering distance.

Table 2. Test suite for evaluating graphical performance in 3D mobile applications.

No - Name	Description
1 - Basic Mesh Rendering	Simple, untextured objects in motion are progressively displayed on the screen in a scene without lighting or shadows. The objects must rotate continuously at a constant speed. The number of objects on screen increases over time.
2 - Complex Mesh Rendering	A complex moving object is displayed, containing a high number of polygons. The rendering distance (clipping plane) gradually increases as the test progresses.
3 - Lights & Shadows	A simulation similar to Basic Mesh Rendering is performed, but in this case, the scene includes lighting and objects with shadow casting and reception.
4 - Textures	A simulation similar to Basic Mesh Rendering is performed, but in this case, objects have complex textures, such as transparency, reflection, or other visual effects.
5 - Particle Systems	A scene is created where new instances of a particle system are progressively introduced, such as smoke, fire, sparks, explosions, among others.
6 - Physics	A simulation similar to Basic Mesh Rendering is performed, but in this case, objects are subject to physical rules, such as friction, force, or gravity.

4 Evaluation Prototype Development

To carry out the proposed graphical performance evaluation methodology on real devices, it was necessary to develop two prototypes. The first prototype implements the set of tests for non-VR environments, while the second does so for VR environments. The game engine chosen for developing the prototypes was Unity [8], due to its versatility, extensive documentation, available tutorials, large component repository, and strong community support. Additionally, to develop the VR evaluation prototype in Unity, the Google Cardboard SDK [9][10] was used. This tool, provided by Google, allows the creation of virtual reality applications for mobile devices, primarily on Android and iOS platforms. Google Cardboard is an accessible device that transforms a smartphone into a VR headset.

For both prototypes, the evaluation tests defined in the previous section were implemented as follows:

- Cubes were used for tests involving simple objects (Basic Mesh Rendering, Lights & Shadows, Textures, Physics).

- For the Complex Mesh Rendering test, the model developed by Cristina et al. [11] was selected, corresponding to the Faculty of Informatics building at the National University of La Plata. This model consists of more than 500,000 polygons.
- In the Textures test, a transparency effect was applied to the objects involved.
- In the Particle Systems test, a system simulating fire sparks was used.
- In the Physics test, a gravity effect was applied to the simple objects in the scene. Additionally, to ensure that the objects followed random trajectories, a large central sphere was placed in the scene. The objects collide with this sphere and are propelled in different directions.

The graphical performance evaluation prototypes progressively double the number of instantiated objects. In other words, at the start of each test, there is only one object on the screen, but this number doubles at fixed time intervals. Throughout the test, the frames per second (FPS) rendered by the application are continuously monitored and recorded. This methodology is applied to all tests, except for the Complex Mesh Rendering test.

In this specific test, where a single complex object is used, performance is measured based on rendering distance rather than the number of objects on the screen. The complex object is centered on the screen and continuously rotates. At the beginning of the test, the rendering distance is minimal, so no part of the object is visible. At fixed time intervals, the rendering distance is increased, progressively revealing more of the object and increasing the number of polygons visible on the screen. Throughout this process, FPS is continuously monitored and recorded.

Fig. 1 shows the main menu of the non-VR prototype. The screen displays six buttons, each with a descriptive image representing the corresponding test. Additionally, it allows the user to select the execution quality, offering two options: Minimum (Fastest) and Maximum (Fantastic). The results obtained can also be saved in a log file.

Fig. 2 presents the main menu of the VR prototype, where the screen is split into two sections, projecting a different image for each eye. The screen contains six objects, each corresponding to a different test. To start a test, the user must point at a specific object and select it. Upon completing each test, the results are automatically stored in a log file. The execution quality can be adjusted by selecting the quality object.

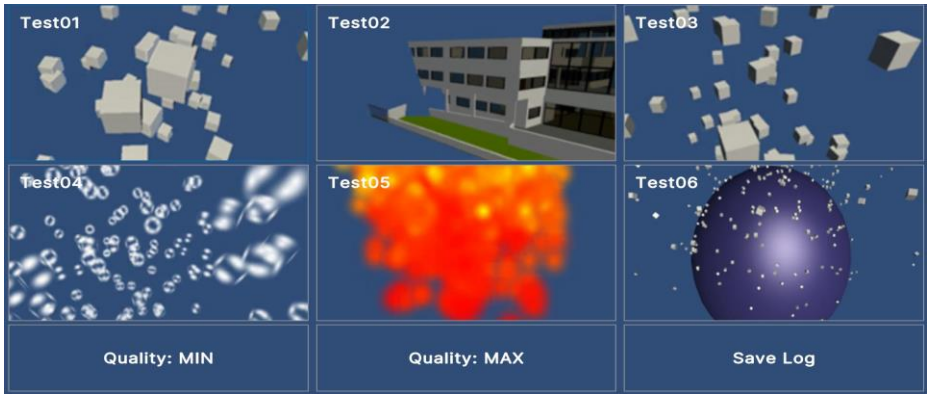

Fig. 1. Non-VR prototype – Main screen.

Fig. 2. VR prototype – Main screen.

5 Experiments and Results

To carry out the experimentation in both prototypes, it was necessary to use devices compatible with VR functionality, meaning they had to include sensors such as an accelerometer and gyroscope, which enable motion and orientation tracking. Table 3 details the technical specifications of the four devices used in the experimentation, which consisted of performing the complete set of tests on both prototypes and across all four devices, with the objective of comparing the results obtained.

Table 3. Test suite for evaluating graphical performance in 3D mobile applications.

Device (Brand and Model)	Operating System	General Specifications
Motorola Moto G6	Android 8.	Octa-core 1.8 GHz Cortex-A53 3 GB RAM.
Motorola Moto G9 Plus	Android 10	Octa-core (2x2.2 GHz Kryo 470 Gold & 6x1.8 GHz Kryo 470 Silver) 4 GB RAM.
Motorola Moto G52	Android 12.	Octa-core (4x2.4 GHz Kryo 265 Gold & 4x1.9 GHz Kryo 265 Silver) 4 GB RAM.
Samsung Galaxy A24	Android 13.	Octa-core (2x2.2 GHz Cortex-A76 & 6x2.0 GHz Cortex-A55) 4 GB RAM.

Fig. 3 and 4 show the average values of the results obtained from the tests conducted using the non-VR prototype, considering the two possible extremes in rendering quality in Unity. It is important to note that the maximum frame rate of the evaluation prototype was limited to 60 FPS, as the Samsung Galaxy A24 and Motorola Moto G52 support up to 90 FPS, and this difference would affect the calculated average values.

Fig. 5 and 6 present the average values of the results obtained from the tests conducted using the VR prototype, again considering the two possible rendering quality extremes. The maximum frame rate was also limited to 60 FPS, for the previously mentioned reasons.

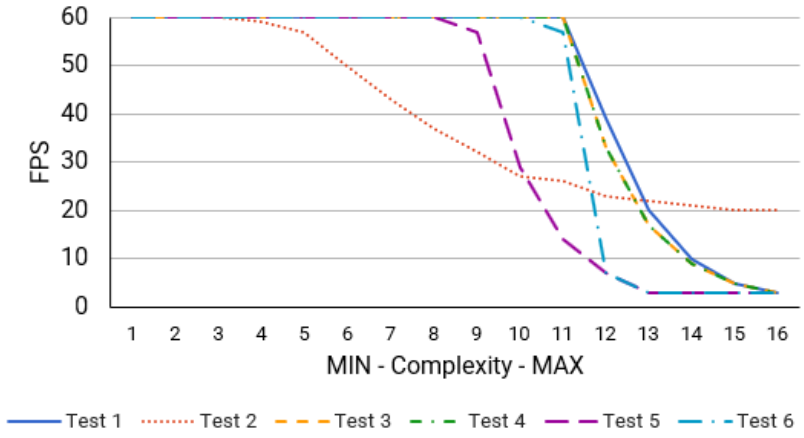

Fig. 3. Evolution of each test in the Unity prototype without virtual reality at the lowest graphics quality (Fastest).

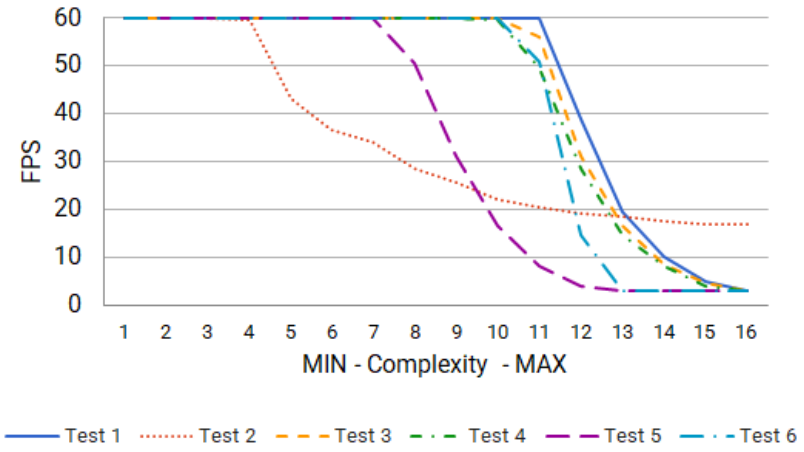

Fig. 4. Evolution of each test in the Unity prototype without virtual reality at the highest graphics quality (Fantastic).

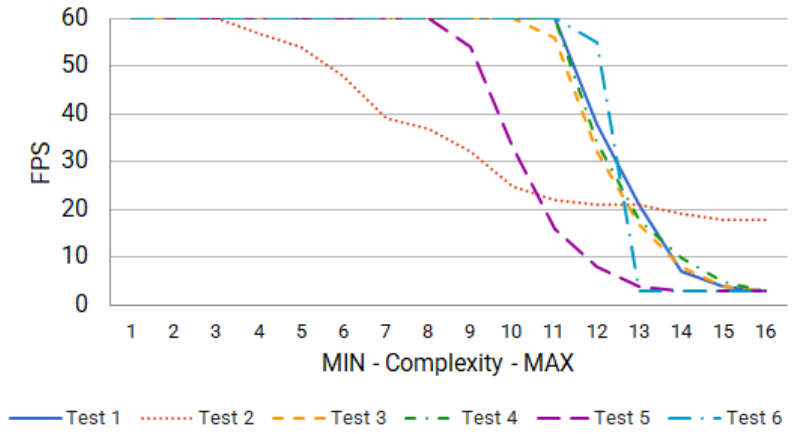

Fig. 5. Evolution of each test in the Unity prototype with virtual reality at the lowest graphics quality (Fastest).

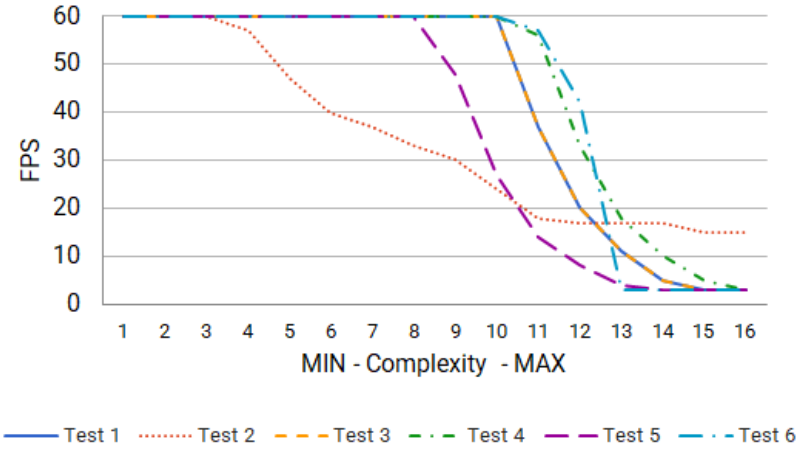

Fig. 6. Evolution of each test in the Unity prototype with virtual reality at the highest graphics quality (Fantastic).

Fig. 7 and 8 illustrate the comparison of the results obtained from both prototypes, non-VR and VR, using the same set of devices and considering the two possible rendering quality extremes.

A ratio equal to 1 indicates that both prototypes had the same graphical performance. If the ratio is greater than 1, it means that performance was better in the non-VR prototype. Conversely, if the ratio is less than 1, performance was better in the VR prototype.

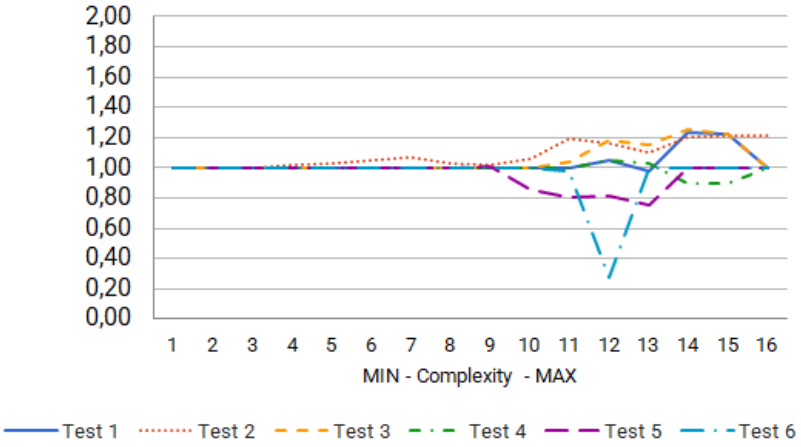

Fig. 7. Comparison of the results obtained in the Non-VR and VR prototypes at the lowest graphics quality (Fastest).

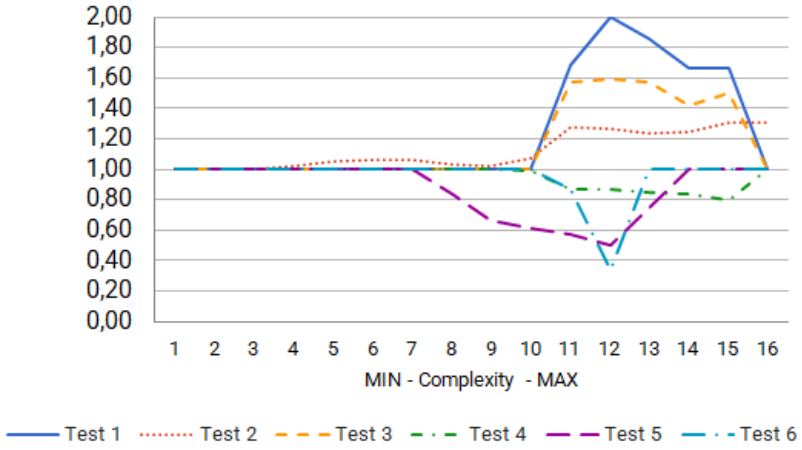

Fig. 8. Comparison of the results obtained in the Non-VR and VR prototypes at the highest graphics quality (Fantastic).

It should be noted that in tests involving an increase in the number of object instances, the final complexity level (32,768 instances) results in the lowest possible frame rate (3 FPS) in all cases. For this reason, in those cases, the ratio always converges to 1. The only exception is Test 2, since, as previously mentioned, it does not involve an increase in the number of instances throughout the test.

When analyzing Fig. 7 and 8, it can be observed that, in general terms, the values obtained in both prototypes are similar. The non-VR prototype exhibits slightly better performance in scenarios with higher graphical complexity. At lower complexity levels, both prototypes maintain similar performance. However, as the number of instances increases, the non-VR prototype begins to slightly outperform the VR prototype, especially when using the highest graphical quality setting. On the other hand, in Tests 5 and 6, labeled Particle Systems and Physics, the VR prototype demonstrates a slightly better performance, reaching performance peaks in certain specific cases.

The greatest differences, however, are observed at higher complexity levels, where the achieved frame rates are very low. In these cases, even a small performance difference can result in more abrupt variations in the ratio between both prototypes. For example, if the FPS values for complexity level 15 are 6 and 3, respectively, the ratio will be 2.0, even though in both cases, the frame rate is already very low.

Initially, the lack of significant differences in the comparative results might seem counterintuitive, since the VR prototype must duplicate the scene to display a different viewpoint for each eye, meaning it should be rendering twice as many objects on screen.

The similarity in the results obtained could be due to the fact that modern devices compatible with VR applications are designed with specific hardware optimizations to enhance performance for this type of application.

Additionally, game engines—in this case, Unity—and APIs such as Google VR employ more efficient rendering techniques, including resolution reduction, shared

buffers, and quality reduction in certain graphical effects, to balance the workload and optimize resources.

Thus, even though the screen is split into two, game engines are optimized to render both views almost simultaneously, rather than duplicating graphics entirely independently. Instead, they adapt the original scene to the viewpoint of each eye.

6 Discussion and limitations

The interplay between various non-functional requirements is crucial, particularly when balancing graphical performance and energy consumption. While high graphical fidelity is key to immersive experiences, it often leads to increased energy consumption. For example, increasing polygon counts or enabling real-time lighting effects places greater strain on CPU/GPU cycles, draining battery life faster. Developers must strike a balance between rendering quality and sustainability, especially when targeting mid-range or older devices.

Furthermore, graphical fidelity influences perceived usability and user satisfaction. Applications with stable, smooth frame rates generally receive higher user ratings, highlighting a correlation between NFR compliance and user retention. When animations stutter or lag, users may abandon the app, regardless of its functional richness. This suggests that user-centric NFR, such as responsiveness and interface intuitiveness, are closely tied to underlying graphical and hardware constraints.

Future implementations should consider NFR trade-offs not in isolation, but as part of a broader user experience centered strategy. For instance, optimizing textures may improve performance, but excessive optimization could reduce immersion and visual appeal. Understanding these trade-offs can lead to better-informed design decisions [12][13][14].

While this analysis offers valuable insights, several limitations must be acknowledged. The study is limited by the absence of user-centric evaluations and real-world case studies. Furthermore, results are constrained to the specific devices and engine used (Unity with Google Cardboard SDK). These constraints may affect the generalizability of our findings.

7 Conclusions and future work

This study highlights the significance of non-functional requirements in the context of 3D mobile and virtual reality applications, as they directly affect the user experience. Among them, graphical performance is particularly crucial, affecting not only user satisfaction and system responsiveness but also the strain on hardware. It is key to creating an immersive and visually appealing experience, influencing both visual quality and overall device performance.

To evaluate graphical performance in 3D mobile applications, two prototypes were developed, implementing a set of tests to assess the different characteristics of a 3D mobile application, with and without virtual reality. These tests aimed to identify common patterns related to performance degradation. Additionally, this approach allowed

for a comparison of graphical performance behavior in VR and non-VR environments, providing insights into the impact of virtual reality on a 3D mobile application.

Regarding the future continuation of this research, several development paths can be considered.

First, an evaluation of energy efficiency could be explored, as it is another highly relevant non-functional requirement in 3D mobile applications and VR environments.

Additionally, an equally interesting research area is the study of augmented reality (AR) applications. A potential future study could involve developing evaluation prototypes for 3D mobile applications with AR, focusing on both graphical performance and energy efficiency. This approach would enable the analysis of camera and sensor usage, interaction with the physical environment, and variations in graphical processing, given the real-time processing requirements in such cases.

Also, for future work, the inclusion of qualitative and user-centric evaluations is suggested. By collecting feedback through surveys, usage tracking, or interviews, researchers can gain a better understanding of how perceived performance aligns with technical metrics such as frame rate or loading time.

Furthermore, expanding the testing framework to include real-world case studies is recommended, particularly in educational, gaming, or industrial VR applications, where context-specific constraints influence NFR prioritization.

Lastly, comparative studies using alternative engines like Unreal [15] or Godot [16], and testing on newer devices, will provide deeper insights into the role of optimizations, rendering techniques, and hardware acceleration in managing trade-offs between performance, portability, and energy consumption.

The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Linowes, J.: “Unity Virtual Reality Projects”. 1st ed. ISBN-13: 978-1783988556 (2015).
2. Siriwardhana, Y., Porambage, P., Liyanage M., Ylianttila, M.: "A Survey on Mobile Augmented Reality With 5G Mobile Edge Computing: Architectures, Applications, and Technical Aspects," in IEEE Communications Surveys & Tutorials, vol. 23, no. 2, pp. 1160-1192. doi: 10.1109/COMST.2021.3061981 (2021).
3. Costa, M.: "Automated verification of compliance of non-functional requirements on mobile applications through metamorphic testing," IEEE 13th International Conference on Software Testing, Validation and Verification (ICST), Porto, Portugal, pp. 421-423, doi: 10.1109/ICST46399.2020.00053 (2020).
4. Corbalán, L. et al: “A Study of Non-functional Requirements in Apps for Mobile Devices.”. Cloud Computing and Big Data. JCC&BD 2019. Communications in Computer and Information Science, vol 1050. Springer, Cham. https://doi.org/10.1007/978-3-030-27713-0_11 (2019).
5. Xiangfei, L., Wang X., Sun R.: "Real-time 3D graphics for mobile devices on reconfigurable hardware," IET International Conference on Smart and Sustainable City 2013, Shanghai, pp. 471-475, doi: 10.1049/cp.2013.1963 (2013).

6. Unity Manual. Graphics performance fundamentals. https://docs.unity3d.com/Manual/OptimizingGraphicsPerformance.html
7. Ma, X., Deng, Z., Dong, M., Zhong L.: "Characterizing the Performance and Power Consumption of 3D Mobile Games," in Computer, vol. 46, no. 4, pp. 76-82, doi: 10.1109/MC.2012.190 (2013).
8. Unity Game Engine. https://unity.com/es.
9. Open source Cardboard SDK - Google for Developers. https://developers.google.com/cardboard.
10. Guía de inicio rápido de Google Cardboard para Unity [Google Cardboard Quick Start Guide for Unity]. https://developers.google.com/cardboard/develop/unity/quickstart.
11. Cristina, F., Dapoto, S., Thomas, P., Pesado, P.: "InfoUNLP3D: An interactive experience for freshman students". "Computer Science & Technology Series - XXII Argentine Congress of Computer Science. Selected Papers". Editorial: EDULP. ISBN: 978-987-4127-28-0, pp. 249-256 (2017).
12. Weichbroth, P.: "Usability Testing of Mobile Applications: A Methodological Framework." https://www.researchgate.net/publication/378402991 (2024).
13. Li, Z., et al.: "VR User Reviews Analysis: Impact of Non-functional Attributes." arXiv:2308.06783. https://arxiv.org/abs/2308.06783 (2023).
14. Jasani, K.: "Performance Optimization in VR Applications: QA's Role.". International Journal of Leading Research Publication (IJLRP). E-ISSN: 2582-8010 (2024).
15. Unreal Game Engine. https://www.unrealengine.com/es-ES.
16. Godot Game Engine. https://godotengine.org.