Analysis of the effect of the Operating System in renderization performance.

Víctor Ortiz Ruiz
Escuela de Ingeniería en Computación
Tecnológico de Costa Rica

Allan Perez Valverde
Escuela de Ingeniería en Computación
Tecnológico de Costa Rica

Ignacio Valdivia Aguero
Escuela de Ingeniería en Computación
Tecnológico de Costa Rica

Abstract

Most PC users have the tendency to believe that the OS they engage with the most, is the best one. Nevertheless, many of these believes are drawn empirically rather than through statistical research to support these theories.

To determine the true capabilities of each operating system and determine whether there is a statistically significant difference, an experiment was therefore proposed to compare the performance of several operating systems rendering images with Physically Based Rendering Technique (PBRT).

Keywords

PBRT, Operating System, OS, Performance, Linux, Windows, Virtual Machine, Ubuntu

I. Introduction

Most PC users have the tendency to believe that the OS they engage with the most, is the best one. Nevertheless, many of these believes are drawn empirically rather than through statistical research to support these theories.

To determine each operating system's true capabilities and whether there is a statistically significant difference, an experiment was proposed to compare the performance of several operating systems by rendering images on a single platform.

Rendering is the process of producing an image from the description of a 3D scene. It is a very broad task and there are many ways to approach it.

Physics-based methods attempt to simulate reality, that is, they use the principles of physics to model the interaction of light and matter. Although one approach based on physics may seem like the most obvious way to approach rendering, it has only been widely adopted in practice in the last 10 years.Physically Based Rendering Technique (PBRT), provides a base system for rendering 3D scenes based on ray-tracing algorithms (1).

This investigation is divided into seven parts, where Section II presents works related to the research on the performance of operating systems. While that in sections III and IV the proposal is presented together with the problem posed for the study in conjunction with the methodology used to carry out the experiment. In the sections V and VI the results obtained and the discussion of said results can be found. Finally, in section VII there will be the final conclusions and possible future work based on the findings.

II. Related Work

Multiple comparative studies have been carried out previously to determine which operating system performs better over another through the use of tools. In (2) the performance of information systems was evaluated based on CPUs and domestic operating systems using the LoadRunner and JMeter tools as performance metrics performance may not reflect the user. These were used programs as a basis for creating improved versions of these, with which they compare the user experience time with each of the proposals; The results show that the JMeter scheme gives better results than LoadRunner in home systems. On the other hand, comparative studies have been carried out by means of containers as a base tool. In (3), it shows how containers have an important use in high-reliability applications, as these can be created with greater ease and lightness compared to a virtual machine, which is why the research presents an analysis of the performance of a Linux container like Windows. Better performance is observed using Linux, due to the lack of Windows for emulation of a permanent machine for the storage device. Following this same line, in (4) they compare the performance of the physical operating system installed and running natively and Docker. Using as a validation method with the FileBench tool, known for being a point reference among open source software to try system capabilities, comparisons are made between physical and virtualized systems.

III. Design of the experiment

Operating systems have differences that could affect the performance perceived by users, to confirm this and prevent the different abstractions that both operating systems influence the performance of both operating systems, It was decided to use an image synthesis tool that would allow having the same configuration regardless of operating system in action. The program selected for this task is the PBRT) (1). To design an experiment that shows precisely if there is a statistically significant

difference between the different operating systems and that in turn takes into account It has a wide range of characteristics such as the synthesis of images of hair, and intricate shapes with different indices of reflection, among others. Taking these properties into account, we selected four images where the most important factor was the duration of the synthesis. Where three scenes were chosen for their bass relative synthesis time, while the other one has a considerably longer compilation time. The experiment was designed as a factorial experiment of parameters 2k, where the two-level factors were the accelerator and integrator variables. The first one is in charge of efficiently finding the intersections of the aces of light, While the second implements the medium by which these aces of light will be transported for the calculation of radiance on surfaces (1). Factorial experiments with 2k parameters allow the use of analysis of variance to define whether groups of data present a statistically significant difference among them. This statistical test requires 3 assumptions: randomness, homoscedasticity, and normality.(5). To comply with the principle of randomness, it should be carried out a random synthesis on a random operating system. However, this design has practical limitations, so it was decided to use a stratified design at the level of the operating system, where you select an operating system randomly, and within this all possible experiments are in random order. This design presents a slight non-compliance with the assumption of randomness, which is why it introduces an increased type I error, in order to reduce this error, several random repetitions to ensure that execution orders in multiple experiments will ensure the principle of independence. The initially chosen value for the experiment it was five repetitions, however, due to time limitations, this number was reduced to two repetitions. This number allows us to obtain a calculation of 324 experiments, with a low probability that the experiments have the same order between executions. The number of experiments is given by the five systems, by the four scenes, for two repetitions, and finally for the 2k factors, where k = 2, for a total of 324 experiments. An important aspect is that machine executions Virtuals were assigned half of the available memory on the machine since the total memory allocation does not allow to obtain the genuine performance of the virtual system, but not so with the logical nuclei, where they were assigned in its completeness so that the comparison was fair between all systems.

IV. Methodology

In the experiment, a methodology is based on the one used in (6). Where the factor analysis of variance (ANOVA), is designed to efficiently evaluate the effects that different factors under investigation have on the response variable. In the analysis, the following factors and their effect were evaluated in the output variable.

• Scene: The type of scene impacts the timing of performance, depending on the complexity of the image. Here's the legend of how to read Scene values:

• Accelerator (Accel): The Accelerator parameter affects the synthesis time by selecting different strategies

Table I: Accelerator legend

Number	Accelerator
1	cloud/smoke.pbrt
2	killeroos/killeroo-simple.pbrt
3	figures/f3-18.pbrt
4	simple/room-sppm.pbrt

for accelerating the renderization of rays of light and mesh textures. Here's the legend of how to read Accelerator values:

Table II: Accelerator legend

Number	Accelerator
0	No accelerator, default settings
1	Accelerator "bvh"
2	Accelerator "kdtree"

• Integrator: The Integrator parameter also affects the synthesis time by selecting different strategies of integration of the rays of light and textures. Here's the legend of how to read Integrator values:

Table III: Integrator values legend

Number	Integrator
0	No Integrator changes, default settings
1	Integrator "path"
2	Integrator "bdpt"

• Operating system (OS): This is the main topic of interest in this research, to be able to analyze the effect of the operating system on the renderization time of different scenarios. Here's the legend of how to read Operating system values:

Table IV: Operating Systems values legend

Number	Operating System (OS)
1	Windows10_OS
2	Ubuntu_LinuxOS
3	Windows10_VM in LinuxOS (Ubuntu)
4	LinuxVM in WindowsOS
5	Ubuntu_LinuxVM in Ubuntu_LinuxOS

The OS selected from the table IV is because they are the most widely used settings among users and servers. Most users use Windows, next Linux. Also to check wether a virtual machine can give the same benefits as using a native OS, using diferent settings (If Windows improves by using a LinuxVM, or Linux improves by using a WindowsVM). We also included LinuxVM on LinuxOS to mimic how most servers work around the world.

In order to obtain which operating system the better performance in the synthesis of photo-realistic images. The Physically Based Rendering program was selected Physically Based Rendering (PBRT), since it allows be compiled on multiple operating systems with the same setting. This allows a fair comparison between the operational systems, because when compiling the program with the same configuration, the insertion of optimizations is avoided according to the operating system and a metric of Genuine performance for operating systems. For the synthesis, four scenes were chosen, these scenes present a wide spectrum of characteristics. For example,

some present dynamics of hair synthesis, while others present intricate and complex shapes. In addition, times of low and high syntheses, cover a large percentage of characteristics that a real synthesis could achieve to have. Within each scene, two factors were selected to modify, the accelerator and the integrator, these factors present 2 levels so they can be modeled as experiments 2k factorials. The accelerator was chosen because changing the synthesis acceleration strategy can favor one operating system over the other, while the integrator allows changing the complexity of solutions for the intersection of the rays that penetrate the surface of the scene. For practical reasons of time, the experiment was carried out in a stratified manner at the operating system level, due to the low practicality of executing a synthesis of an image and then changing the system for another to perform the same proof. This slightly violates the principle of independence, for which was decided to perform two repetitions to mitigate the error introduced by this experimental practice. Three were chosen repetitions, due to the time limitations of the project. Therefore, it was not enough to execute the five repetitions initially planned. The stratified design allows all experiments to be executed of a complete operating system, these experiments are generated randomly according to all possible combinations for the scene. It should be noted that the system operative is also selected randomly. All experiments were run on the same computer. with the characteristics seen in table V, while the running operating systems are displayed in table IV.

Table V: Characteristics of the experimental equipment]

Processor	Intel® Core™ 7th generation
CPU	3.50GHz
Memory	16GB DDR4
Disk space	240 GB SSD

V. Results

To obtain the results, a specific process was followed: In the first instance, before starting the analysis statistical, simple interaction graphs were created of the factors depending on the response variable, to be able to generate a preliminary idea of how they could interact these factors with each other. With this premature conclusion, we proceeded with the statistics analysis of the data collected. To proceed to execute the Anova, its assumptions of normality and homoscedasticity on its residuals must be verified; the randomness criterion was guaranteed in the design of the experiment. Figure 1 shows the histogram of the residuals of the response variable, in which if a visual criterion is applied it can be said that the residuals are normal. Following this same methodology visual, the quartile-quartile graph was made as a verification additional. Figure 2 shows that a significant amount of the data do not meet the normality criterion. For this reason the square root transformation is performed on the data, and not by the assumption that verifies the variance with a Levine's test that is shown in figure shows the results of applying Levine's test, and to the case of the original response variable it has a p-value of 1.557E-09, indicating that the assumption of homoscedasticity does not comply for the original data

Remember that in tables I, II, III and IV are the correspondent values to each scenario used on each experiment, to ease reading the results.

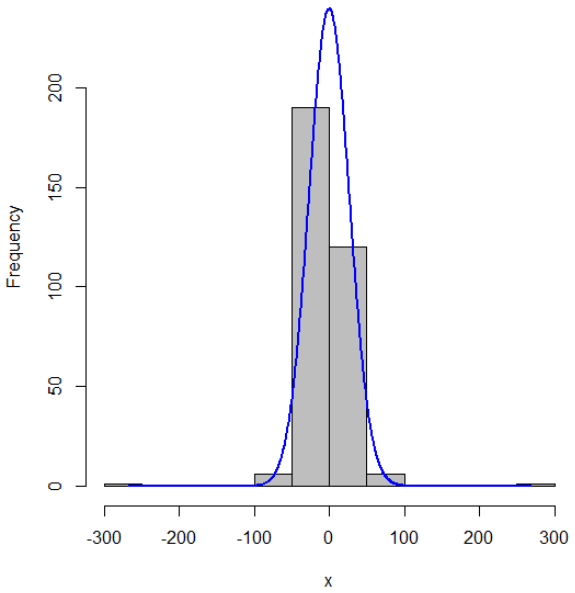

Figure 1: Histogram for the original data

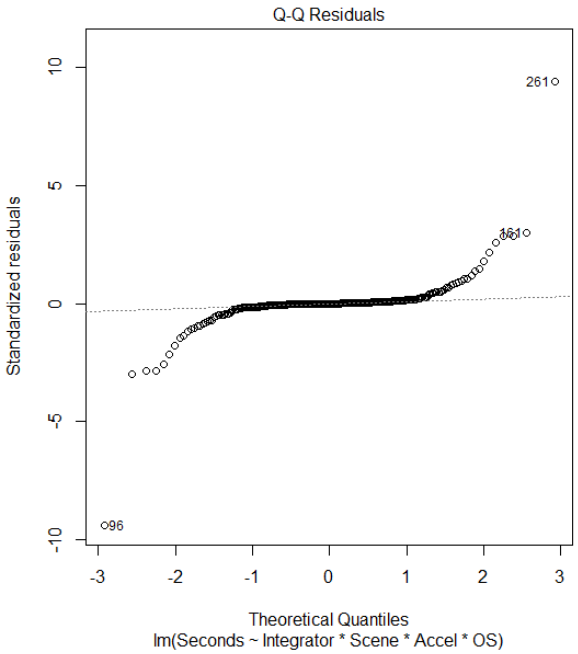

Figure 2: Q-Q test for original Data

Once the square root transformation has been performed Figures 3 and Figure 4 are obtained, where an improvement can be seen visually concerning the untransformed data. In this scenario, with the p-value of 0.007503 shown in table VI for Levine's test can't meet the Anova criteria for the assumption of homoscedasticity.

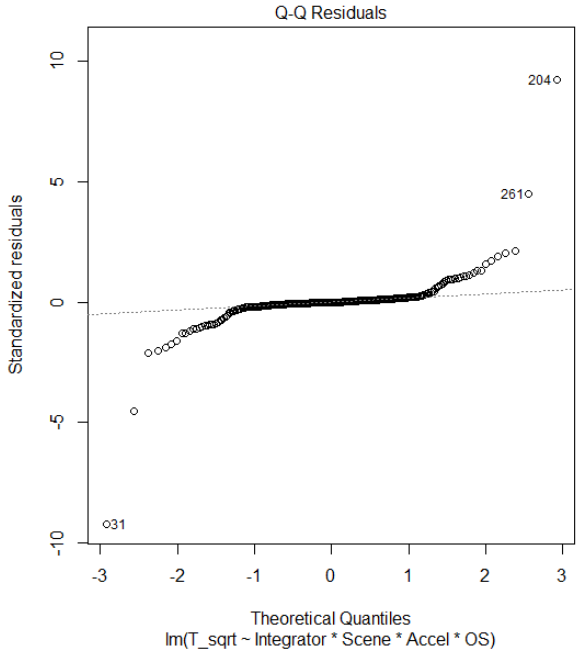

Figure 3: Q-Q test for data with a square root transformation

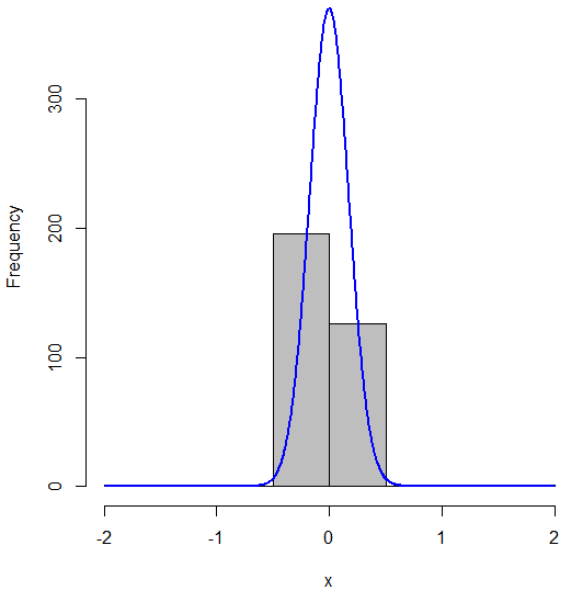

Figure 5: Histogram for the data with a cubic transformation

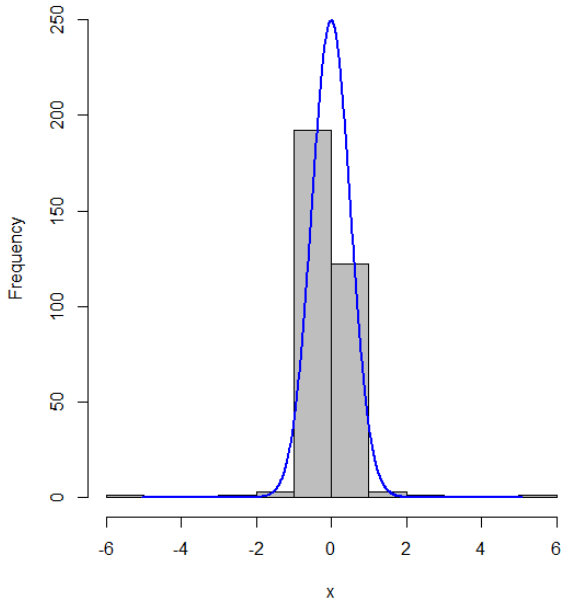

Figure 4: Histogram for the data with a square root transformation

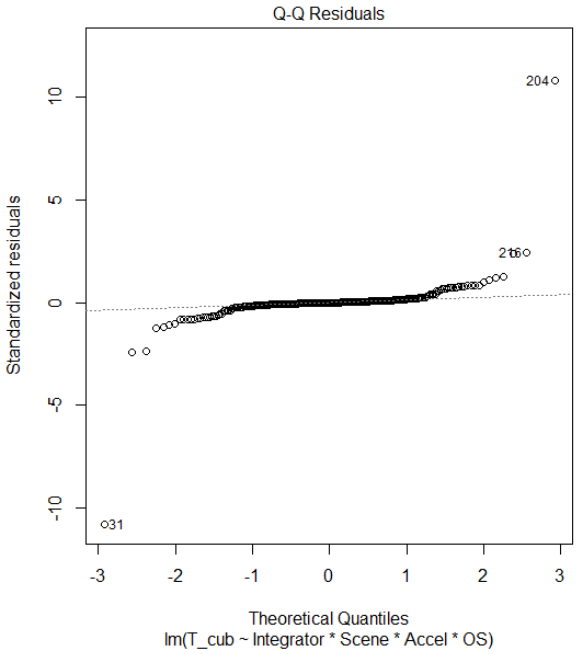

Figure 6: Q-Q test for data with a cubic transformation

With these results new transformations have to be performed, the figures 5 and 6 show the histogram and Q-Q test for the cubic transformation. The figures obtained have another visual improvement concerning the untransformed data and the square root transformation, for verification of the ANOVA assumptions, the Levine's test show a p-value of 0.5956, that indicates that this transformation comply with ANOVA assumption

In the image 7 shows the result of the Anova for the response variable with the cubic transformation. The result obtained shows that each factor by itself has a significant statistical difference, as well as the interactions of the operating system factors with the factor 2k, the operating system with the scenes factor, and the factor 2k with the scenes also show a difference significant. Using the result of image 7 as a criterion, proceeds to analyze the behavior of each of the factors depending on

the operating system and the square root of the response variable. For this, the graphics are created of averages, the dependent variable being the average of the response variable with the transformation applied. The whiskers in these graphs correspond to the standard error. From here we obtain the figures 8, 9 and 10, which corresponds to the graph of averages of scenes depending on the variable transformed response for the different factors.

Table VI: Levine's test results for all transformations

Transformation	p-value
Original data	1.557E-09
Square Root	0.007503
Cubic	0.5956

```
> Anova(model, type="II")
Anova Table (Type II tests)

Response: T_cub
                      Sum Sq  Df    F value    Pr(>F)
Integrator            654.06   2  4787.5930 < 2.2e-16 ***
Scene                2198.00   3 10726.0340 < 2.2e-16 ***
Accel                  24.52   2   179.4557 < 2.2e-16 ***
OS                   1817.57   4  6652.1651 < 2.2e-16 ***
Integrator:Scene      476.07   6  1161.5872 < 2.2e-16 ***
Integrator:Accel        0.61   4     2.2244   0.06924 .
Scene:Accel             2.84   6     6.9182 1.787e-06 ***
Integrator:OS          64.51   8   118.0583 < 2.2e-16 ***
Scene:OS              325.55  12   397.1684 < 2.2e-16 ***
Accel:OS                8.19   8    14.9840 7.670e-16 ***
Integrator:Scene:Accel  0.96  12     1.1772   0.30477
Integrator:Scene:OS    82.02  24    50.0339 < 2.2e-16 ***
Integrator:Accel:OS     1.55  16     1.4216   0.13927
Scene:Accel:OS          3.20  24     1.9514   0.00865 **
Integrator:Scene:Accel:OS  3.28  48     1.0001   0.48412
Residuals               9.84 144
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 7: Enter Caption

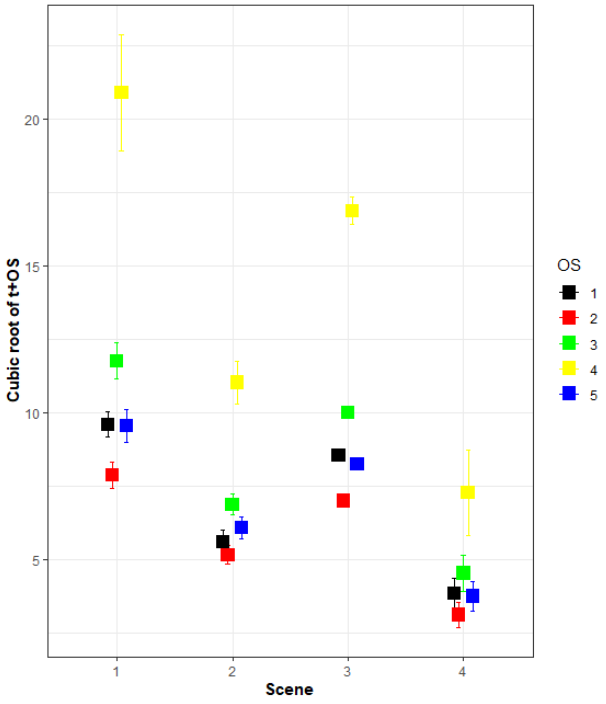

Figure 8: Graph of averages of the scene depending on the variable transformed response for the factor Operating System

Also, we obtained the graph 11 that corresponds to the average of the operating system depending on the variable

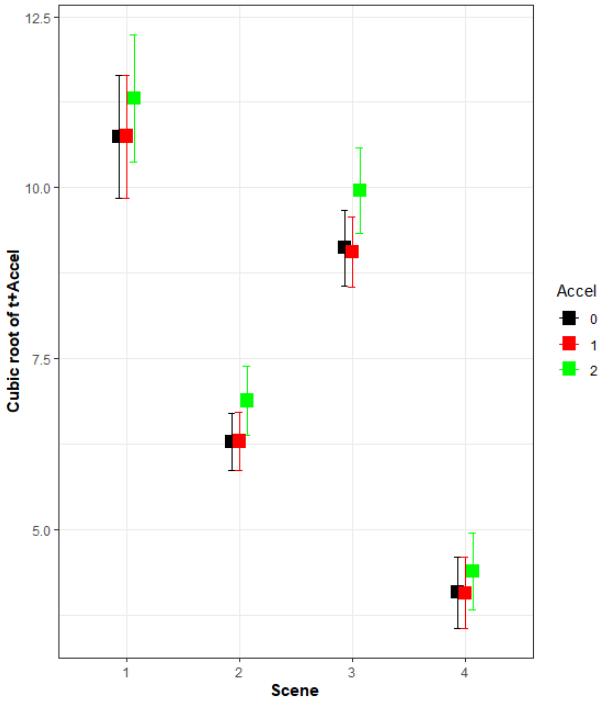

Figure 9: Graph of averages of the scene depending on the variable transformed response for the factor Acceleration

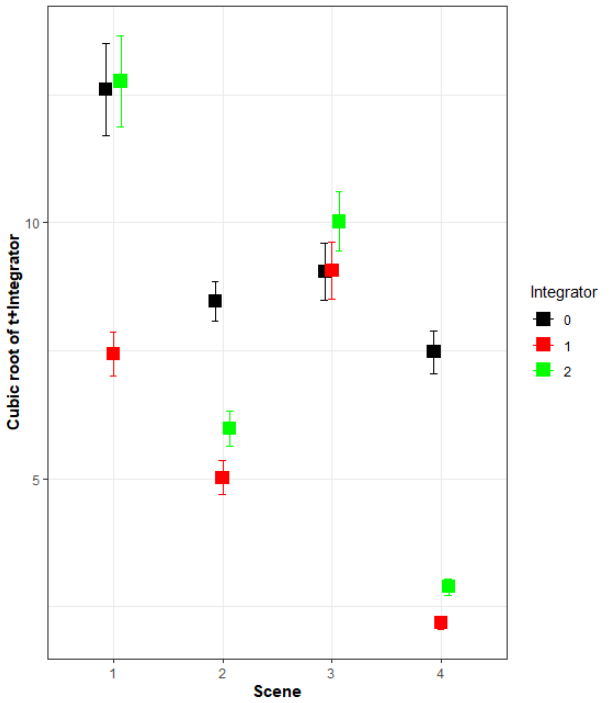

Figure 10: Graph of averages of the scene depending on the variable transformed response for the factor Integrator

transformed response for the factor operating system, and the result table 12 shows the p-values of this interactions.

Continuing with the analysis graph 13 shows the average of the accelerator in the function of the integrator and graph 14 shows the average of the accelerator in the function of the operating system

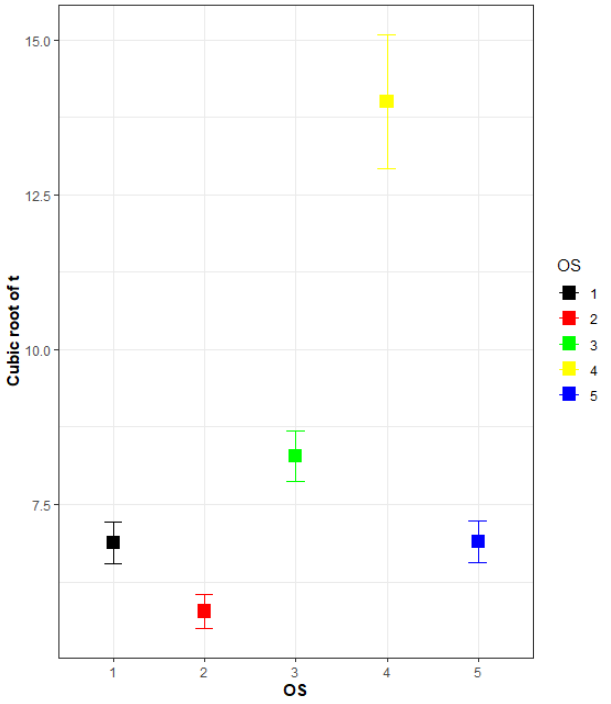

Figure 11: Graph of averages of the operating system depending on the variable transformed response for the factor Operating System

```
Results are averaged over the levels of: Integrator, Scene, Accel
Confidence level used: 0.95

$contrasts
 contrast  estimate     SE  df  t.ratio p.value
 OS1 - OS2   1.1105 0.0436 144   25.494  <.0001
 OS1 - OS3  -1.3978 0.0436 144  -32.090  <.0001
 OS1 - OS4  -7.1246 0.0533 144 -133.547  <.0001
 OS1 - OS5  -0.0177 0.0436 144   -0.407  0.9942
 OS2 - OS3  -2.5083 0.0436 144  -57.584  <.0001
 OS2 - OS4  -8.2351 0.0533 144 -154.363  <.0001
 OS2 - OS5  -1.1282 0.0436 144  -25.901  <.0001
 OS3 - OS4  -5.7268 0.0533 144 -107.346  <.0001
 OS3 - OS5   1.3801 0.0436 144   31.683  <.0001
 OS4 - OS5   7.1069 0.0533 144  133.215  <.0001
```

Figure 12: POST-HOC analysis with pairwise for the factor operating system

VI. Discussions

As shown in the results section, the factors operating system and the scenes, together with the 2k factor, have a statistically significant difference. The objective of the study is to compare the performance of the different systems operatives, for this reason, the interaction of the factor of Scenes with the 2k factor is discarded for the analysis. Using the average graph for the response variable with the transformation applied as a function of the system operating and the scene factor of figure 8, it can be understood that each image varies its rendering time in each operating system. Due to this lack of convergence, It is decided to leave out of the post-hoc analysis the interaction of the operating system with scenes. On the other hand, the factors are analyzed individually for each of their possible combinations. Table 17 shows the pairwise t-test for the scenes factor, where different p-values for said combinations, some significant and others not, prove that the intuition of the Anova was partially wrong. Because it cannot be intuited uniformly

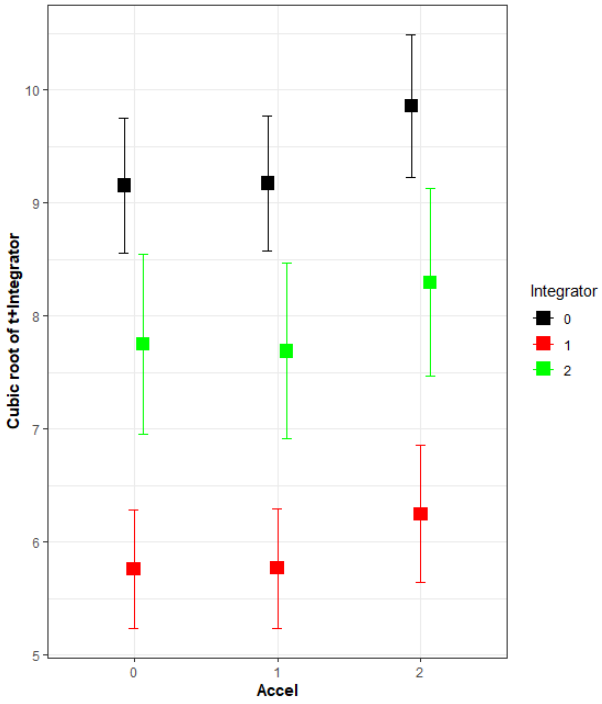

Figure 13: Graph of the average of the accelerator in the function of the integrator

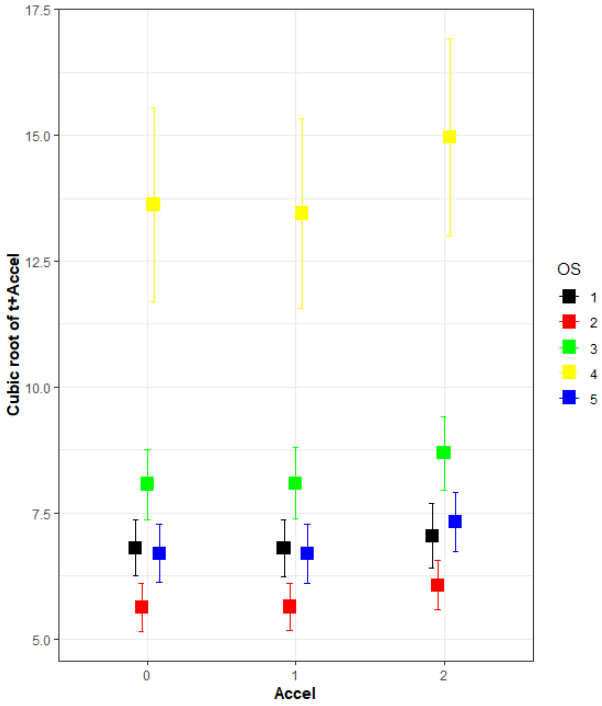

Figure 14: Graph of the average of the accelerator in the function of the operating system

any conclusion regarding this factor, With the previously mentioned reason, it follows that the scenes, whether alone or in their interaction, do not impact significantly the response variable. On the contrary, we proceed to carry out the analysis of the average of the response variable transformed as a function of the 2k factor depending on the operating system. The figure 14 shows the aforementioned average graph, which can observe

```
Results are averaged over the levels of: Scene, Accel, OS
Confidence level used: 0.95

$contrasts
 contrast                  estimate    SE  df t.ratio p.value
 Integrator0 - Integrator1     3.76 0.037 144 101.675  <.0001
 Integrator0 - Integrator2     1.58 0.037 144  42.746  <.0001
 Integrator1 - Integrator2    -2.18 0.037 144 -58.929  <.0001
```

Figure 15: POST-HOC analysis with pairwise for the factor Integrator

```
Results are averaged over the levels of: Integrator, Scene, OS
Confidence level used: 0.95

$contrasts
 contrast        estimate    SE  df t.ratio p.value
 Accel0 - Accel1   0.0287 0.037 144   0.776  0.7182
 Accel0 - Accel2  -0.6535 0.037 144 -17.682  <.0001
 Accel1 - Accel2  -0.6822 0.037 144 -18.458  <.0001
```

Figure 16: POST-HOC analysis with pairwise for the factor Accelerator

how each of the combinations for the factor have similar rendering times for each of the operating systems, that is, this factor behaves similarly regardless of the operating system in which the program used for testing is executed. This statement can be reaffirmed using table 16, where the combinations between the different levels of the factor do not show a significant statistical difference. Therefore, it deduces that both the Accelerator and the Integrator work very similarly regardless of what operating system it is configured.

Finally, we have the average graph in the figure 11. Only the average of the variable is shown here transformed response as a function of operational system performance. Looking at the image you can see that Linux has a better runtime than all operating systems. This is corroborated by the data from Table 12, where the pairwise t-test shows that Linux has a significant difference in its interactions with the other operating systems. These data, together with those shown in the figure referenced above, point out that Linux has the best execution time among the operating systems within the studio. The data analyzed in this study, as mentioned previously, is the response variable with an applied transformation. The study has statistical validity since transforming data involves cutting distances, which makes all tests more strict.

VII. Conclusions

It can be seen that the Linux operating system has a statistically significant difference concerning the other systems operational, presenting the best performance in terms of image synthesis. Future work would be to consider the impact of the elimination of secondary tasks (daemons) that the systems would present operative in image synthesis, and quantify how much the consumption of resources that they would possess or observe the difference with a bare-metal application. A modification to the previous proposal that would allow obtain results on the performance of operating systems, would be the use of containers since they share the kernel with the host operating system, having only the processes necessary for the correct functioning of the container. Another interesting aspect to evaluate is the increase in the quantity of equipment with the same characteristics, to mitigate possible problems in the installation or the configuration of the systems.

```
Results are averaged over the levels of: Integrator, Accel, OS
Confidence level used: 0.95

$contrasts
 contrast        estimate     SE  df t.ratio p.value
 Scene1 - Scene2     4.99 0.0427 144 116.929  <.0001
 Scene1 - Scene3     1.80 0.0427 144  42.252  <.0001
 Scene1 - Scene4     7.44 0.0427 144 174.392  <.0001
 Scene2 - Scene3    -3.19 0.0427 144 -74.677  <.0001
 Scene2 - Scene4     2.45 0.0427 144  57.462  <.0001
 Scene3 - Scene4     5.64 0.0427 144 132.140  <.0001
```

Figure 17: POST-HOC analysis with pairwise for the factor SCENE

References

[1] M. Pharr and G. Humphreys, *Physically based rendering: From theory to implementation*, 9 2004. [Online]. Available: http://ci.nii.ac.jp/ncid/BB03369267

[2] L. Dong, X. Jing, and Y. Chunhui, "Study of Performance Testing of Information System Based on Domestic CPU and OS," *Third International Conference on Trustworthy Systems and their Applications*, 9 2016. [Online]. Available: https://doi.org/10.1109/tsa.2016.27

[3] A. Sergeev, E. Rezedinova, and A. Khakhina, "Docker Container Performance Comparison on Windows and Linux Operating Systems," *International Conference on Communications, Information, Electronic and Energy Systems (CIEES)*, 11 2022. [Online]. Available: https://doi.org/10.1109/ciees55704.2022.9990683

[4] B. Dordevic, A. Stefanovic, and V. Timcenko, "File system performance comparison in the case of Docker container-based virtualization with Volumes and Bind mounts," *2022 30th Telecommunications Forum (TELFOR)*, 11 2022. [Online]. Available: https://doi.org/10.1109/telfor56187.2022.9983751

[5] S. Srinivasan and D. C. Montgomery, "Design and analysis of experiments," *Technometrics*, vol. 34, no. 2, p. 230, 5 1992. [Online]. Available: https://doi.org/10.2307/1269246

[6] E. Rivera-Alvarado and F. J. Torres-Rojas, "APU performance Evaluation for accelerating computationally expensive workloads," *Electronic notes in theoretical computer science*, vol. 349, pp. 103–118, 6 2020. [Online]. Available: https://doi.org/10.1016/j.entcs.2020.02.015

Appendix

All the results and executables files can be found on the Git repository https://github.com/llNach/PBRT_OS_Performance/