Chapter 2 Design and construction of a render farm

Capítulo 2 Diseño y construcción de una granja de render

GONZÁLEZ-DOMINGUEZ, Marcos Crecencio†*, JIMÉNEZ-SIMÓN, Fernando, MEDINA-GALINDO, David Rafael and BLAS-ENRIQUE, Roberto

Tecnológico Nacional de México-Tecnológico de Estudios Superiores de Jocotitlán, Digital Animation and Visual Effects Engineering

ID 1st Author: *Marcos Crecencio, Gonzalez-Dominguez* / **ORC ID**: 0000-0002-0933-6374

ID 1st Co-author: *Fernando Jiménez-Simón* / **ORC ID**: 0009-0000-2320-061X

ID 2nd Co-author: *David Rafael, Medina-Galindo* / **ORC ID**: 0009-0000-7209-4753

ID 3rd Co-author: *Roberto, Blas-Enrique* / **ORC ID**: 0009-0001-6994-7791

DOI: 10.35429/H.2023.13.14.24

M, González, F. Jiménez, D, Medina and R. Blas

*marcos.gonzalez@tesjo.edu.mx

A. Reyes, E. López and B. Hernández (AA. VV.) Computer Technology and Innovation. Handbooks-TI-©ECORFAN-Mexico, Mexico City, 2023

Abstract

In the animation industry there are different techniques that allow the creation of audiovisual content such as 3D, 2D, Stop Motion animation, etc. In the case of 3D animation, this consists of a process where everything is made of digitally encompassing the creation of characters, settings, objects and the animation itself. In the first stage of a 3D production, it is not required to use highly capable computing equipment until the final stage known as post-production. At this stage, all the elements that will make up the animation are brought together so that it can be approved and distributed on digital platforms. When trying to unite all the elements that make up the animation, they go through a process called rendering that allows us to obtain photo-realistic images and which in turn gives us the final product of the finished animation, for this stage it is no longer possible work with home computing equipment since even the most powerful computers on the market present problems or difficulties when rendering this type of projects. Faced with this problem, many studios and companies resort to the use of so-called "Rendering Farms" or "Clusters", which consist of the parallel connection of several computing devices, thus achieving that they work as one, allowing greater power and efficiency when it comes to rendering perform tasks that require a lot of processing. Knowing this, the goal of this work is to create a render farm that allows emerging production houses with low resources to implement this tool, helping them complete animation projects that contain complex compositions or that require computing power to render these works.

Rendering, Cluster, Parallel, Platforms, Efficiency, Compositions

Resumen

Dentro de la industria de la animación existen diferentes técnicas que permiten la creación de contenidos audiovisuales como la animación 3D, 2D, Stop Motion Animation, etc. En el caso de la animación 3D, se trata de un proceso en el que todo se hace de forma digital, abarcando la creación de personajes, configuraciones, objetos y la propia animación. En la primera etapa de una producción 3D, no es necesario utilizar equipos informáticos de alta capacidad hasta la etapa final conocida como postproducción. En esta etapa se reúnen todos los elementos que conformarán la animación para que pueda ser aprobada y distribuida en plataformas digitales. Al tratar de unir todos los elementos que componen la animación pasan por un proceso llamado renderizado que nos permite obtener imágenes fotorrealistas y que a su vez nos da el producto final de la animación terminada, para esta etapa ya no es posible trabajar con equipos informáticos domésticos ya que incluso los ordenadores más potentes del mercado presentan problemas o dificultades a la hora de llevar a cabo este tipo de proyectos. Ante esta problemática, muchos estudios y empresas recurren al uso de las denominadas "Granjas de Renderizado" o "Clusters", que consisten en la conexión en paralelo de varios dispositivos informáticos, haciendo así que funcionen como uno solo, permitiendo una mayor potencia y eficiencia a la hora de realizar tareas de renderizado que requieren mucho procesamiento. Sabiendo esto, el objetivo de este trabajo es crear una granja de render que permita a las casas productoras emergentes de bajos ingresos implementar esta herramienta ayudándoles a completar proyectos de animación que contengan composiciones complejas o que requieran potencia de cómputo para renderizar estas obras.

Renderizado, Clúster, Paralelo, Plataforma, Eficiencia, Composiciones

1. Introduction

When an animated project is about to be completed, the chapters or the final short film must go through the rendering process, where photorealistic compositions demand enormous amounts of computing power and the costs of the components are excessively high, so the aim is to collect the necessary information to be able to put together an economical and functional render farm.

Within the digital animation industry, mainly in emerging production houses with low budgets, it is necessary to implement render tools to help them produce quality audiovisual material in a short time, this with the intention of being able to provide the requested works in a timely manner with the highest detail in terms of image and composition. Therefore, it is necessary to search for information about all the hardware and software that is required to be able to assemble the equipment.

In this case, there are four different ways to set up render farms:

- By GPU (Graphic Processing Unit)
- By CPU (Central Processing Unit)
- Hybrid (GPU & CPU Usage)
- By Servers

Each alternative fulfills the same function as long as the assembly is optimal. Aspects such as the number of components and that each piece of equipment has the same characteristics in terms of hardware and software must be considered (Jaros *et al*., 2019).

One of the alternatives that can be implemented for the elaboration of such a farm is to use a hybrid model which consists of computers that have high-performance GPUs and CPUs, which will help the implementation and operation of the render farm.

The present research is divided into several segments of which we will start with the structure of a render farm, its construction and finally its operation.

A comparison will be made between the individual team and the farm using the same project to check the effectiveness of the farm. According to the results obtained, an analysis will be made to verify if it is really viable to use a rendering farm in the animation industry, mainly with emerging production houses.

2. Structure of a render farm

2.1. Structure and function in general

A render farm is made up of different machines connected in parallel via network cables. The main machine that controls the entire farm is known as "*Master*", it is responsible for managing all the tasks and processes that correspond to each machine.

The machines that are in charge of doing all the computational work are known as "*Slaves, Slaves,* or simply *Nodes*", these are in charge of receiving the tasks granted by the Master.

The connection between machines is made through network cables connected to a *Network Switch*, this device is responsible for distributing the data and information that is sent from the master node to the slaves and vice versa simultaneously.

Internally, you can count on various software that allows you to manage the nodes as well as allow you to configure and execute a 3-D scene that is about to be rendered. The software will depend a lot on the render engine to be used since there are several options that are up to the user.

2.2. Hardware

For optimal operation, each node must meet the same specifications, so they must have the same model of processor, GPU and other components that make them up.

In our case, we used a hybrid model that works with CPU and GPU at the same time. These devices have the same specifications and components.

Listed below are the main components of the nodes implemented for the elaboration of our render farm:

- GPU: NVIDIA GeForce RTX 2070 SUPER
- CPU: Intel Core i7-10700K leaves 3.80 GHz at 8 cores
- 32GB DDR4 RAM
- 500 GB SSD
- Dell 1KD4V OptiPlex XE SFF Motherboard

In the same way, the components to establish the connection between nodes are the following:

- 5-Port Gigabit Network Switch with 1000 Mbps Transfer Rate
- CAT6 Network Cables (These allow a data transfer speed of up to 1000 Mbps)

All nodes, including the Master, have these components so that the farm can work optimally.

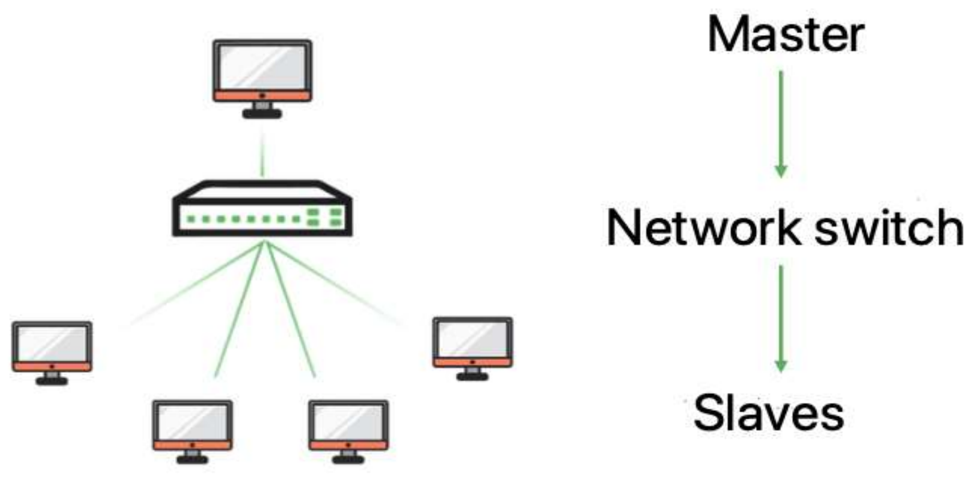
The nodes can be built according to the user's needs, so you can even work with other processor models, GPUs and other components as long as the nodes are the same in construction.

The only essential requirement is the network connection, so you must have a switch and cables that can transfer at least 1000 Mbps. If lower transfer components are used, this will cause a bottleneck that will cause the farm to work at its lowest capacity.

Each computer will need to be connected to the switch with the network cables so that the software can detect their IP addresses.

In summary, the render farm hierarchy is composed of the Master Node, Network Switch, and Slave Nodes as shown in Figure 1.

Figure 1 Schema implemented for the render farm



Source: Authors' Own Creation

2.3. Software

The operating system of each node is Windows 11, this is because the farm is designed to be built with more accessible tools for a not so experienced user.

2.3.1. Blender

"*Blender is a public project hosted on blender.org, licensed under the GNU GPL, owned by its contributors*." (Blender, 2023).

This software allows us to create 3-D scenes that can be rendered with the help of the farm. For this, we will also have secondary software that will allow us to connect the nodes.

2.3.2. LogicReinc.BlendFarm

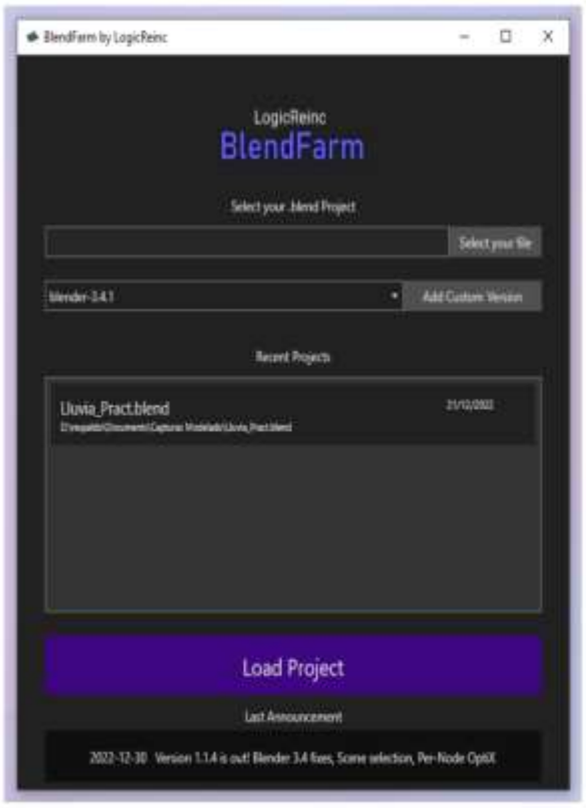
BlendFarm is a cross-platform program that allows you to connect nodes so that they can work together on any project assigned to them. The main feature of this program is that it is open source, so its use is accessible to all users.

The program is a GitHub repository developed by LogicReinc that is intended to help the community that does not have access to render services or simply does not have access to a powerful computer.

2.3.3. BlenFarm Interface

When you run the installer, a window will open with three sections (Figure 2) of which the first will allow you to select your file from which you want to render, the second section will allow you to choose the version of blender (Figure 3) and finally there is the section that will show you the projects you have recently opened.

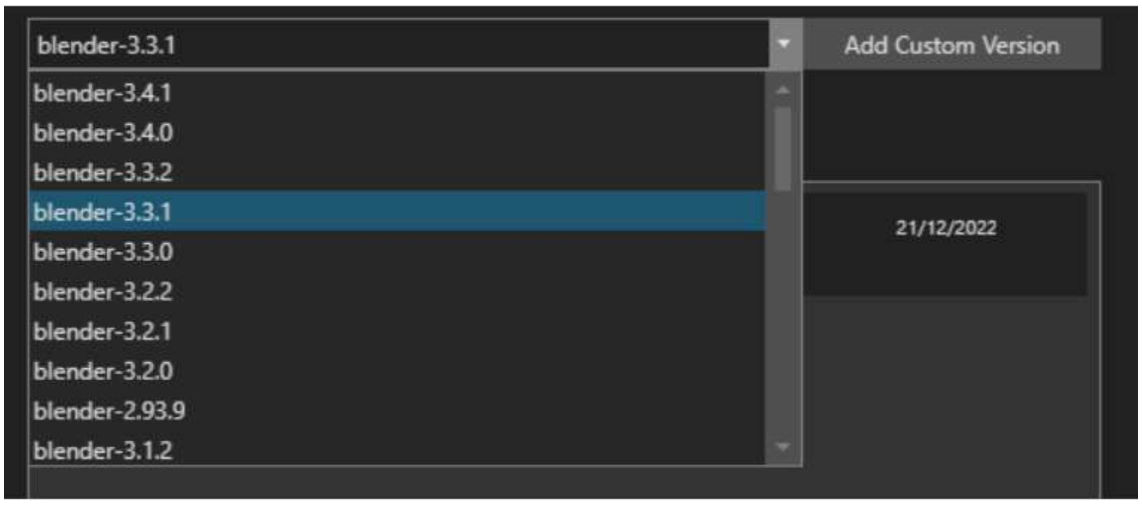
Figure 2 Program Opening Interface



2022-12-30 Version 1.1.4 is out! Blender 3.4 fixes, Scene selection, Per-Node OptiX

Source: Authors' own creation

Figure 3. Program Opening Interface



Source: Authors' Own Creation

In this case, we worked with version 3.3.1 of blender and selected an animation project which can consist of several elements that make the animation more realistic.

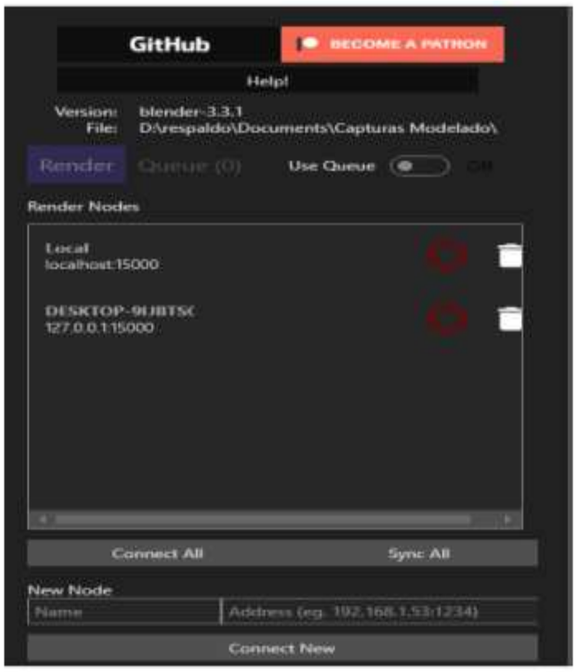
Once you've selected your project, a window will be displayed that will show you the main work interface where you can configure your scene and render the project.

Being on the main interface we will be able to see some panels and options that we will be breaking down little by little, from where we can see two main panels of which the larger one will allow us to visualize the scene we are rendering and the small panel on the right will allow us to configure the nodes and the configuration of the scene.

In Figure 4 you can also see the *Render Nodes panel* and in it we will be able to configure our *Slave nodes* thanks to the IP of each slave computer. To do this, the program must be run on each node with the same configuration used in the *Master*.

Having the program running on each slave, the master will automatically detect them by marking the nodes in green.

Figure 4 Render nodes panel



Version: blender-3.3.1
File: D:\respaldo\Documents\Capturas Modelado\

Local
localhost:15000

DESKTOP-9UBTSC
127.0.0.1:15000

Source: Authors' Own Creation

Once the nodes are configured, we will click on the Master node gear, which in this case has the default name "*local*" and immediately we will start downloading the version of blender that we previously chose.

Once the version is downloaded, we can also notice that the gear gives us more options to configure the node, which are:

- Cores: This option allows you to choose the number of CPU cores to render the scene
- Auto Performance
- Render Type: Displays a list of options with which we can choose which component we want to work on, whether by CPU, GPU or both.
- Performance

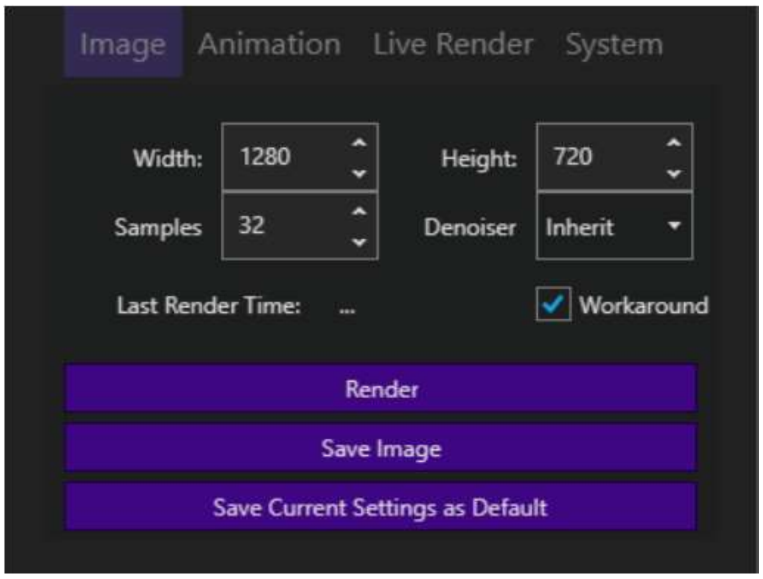
The configuration that was maintained for each node was 16 cores with auto performance enabled and finally we chose the CUDA option that is responsible for using the CPU and GPU to render the project.

2.4. Rendering Options

Image Rendering (Image Tab)

This tab shows us different options that we can configure for the output format of the Render. In Figure 5 we can see four boxes, the first two serve to adjust the output resolution of the image, which in this case has by default 1280 pixels wide (Width) and 720 pixels long (Height).

Figure 5 Image rendering settings



Width: 1280
Height: 720
Samples 32
Denoiser Inherit
Last Render Time: ...
✓ Workaround

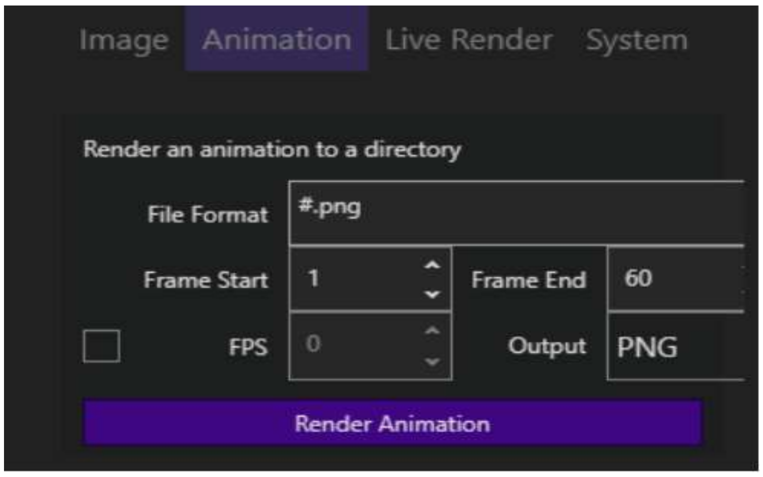
Source: Authors' Own Creation

Rendering an Animation

The *Animation* tab (Figure 6) has five frames. In the first box you can name the output files or frames, for example Image# .png where the # represents the frame number of the animation. The second and third boxes allow us to choose the number of frames to be rendered, as well as the specific range we want to render. The first (Frame Start) is the starting frame and the second (Frame End) is the final frame of the animation. The number of frames must match the number of frames in the original project. In the fourth box we can choose the animation speed (FPS) so it is up to the user to consider if they want to use this box.

Finally, we have the output box where we can choose the format of the image sequence whether it is .png, .jpg, etc. Having the configuration ready you can press the *Render Animation* button for the rendering process to begin.

Figure 6 Animation panel



Render an animation to a directory
File Format #.png
Frame Start 1
Frame End 60
FPS 0
Output PNG

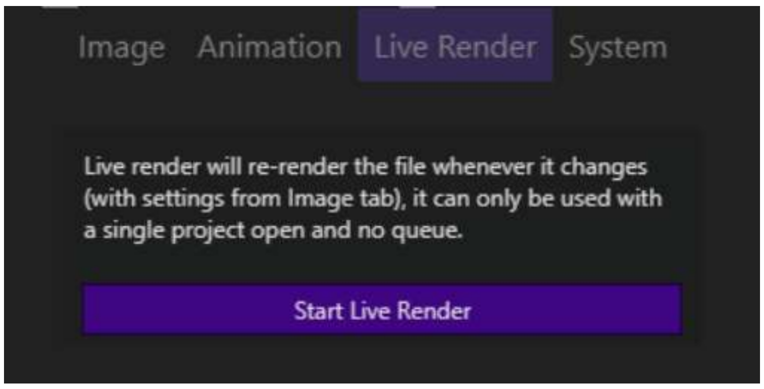
Source: Authors' own creation

The third box (Samples) allows us to control the number of samples that the render engine will take, that is, the limit of calculations of points of light that interact on the object. The fourth box (Denoiser) allows you to configure the noise that the image may have at the end of the render, since the final result can usually give us a very grainy image or with white dots.

Real-time rendering

In Live Render we can visualize the render of our project in real time so that we can manipulate the scene and the result will be seen simultaneously in the main panel. To do this, we must have our blender project open and hit the *Start Live Render button.* (Figure 7).

Figure 7. Panel by Live Render



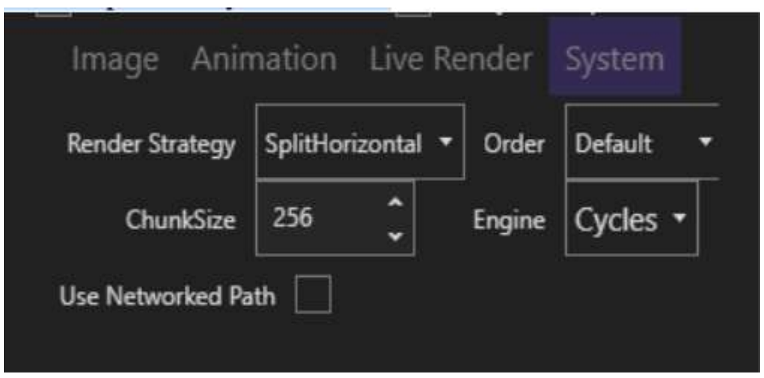
Live render will re-render the file whenever it changes (with settings from Image tab), it can only be used with a single project open and no queue.

Source: Authors' Own Creation

Configuration (System)

Here we can configure the technical preferences of the render farm. (Figure 8) To do this, we will have four boxes, of which the first one (Render Strategy) allows us to change the render display mode and the flow of tiles or frames while rendering. The second box (Order) allows us to configure the order in which tiles, boxes or chunks can be rendered either sparsely or originating from the center of the panel. The first two boxes go hand in hand with the third (ChunkSize) which allows you to adjust the size of the tiles. The last box (Engine) allows you to choose the render engine to use, which in this case Blender has two default render engines, which are Cycles and Evee.

Figure 8 Farm rendering settings



Render Strategy: SplitHorizontal
Order: Default
ChunkSize: 256
Engine: Cycles
Use Networked Path

Source: Authors' Own Creation

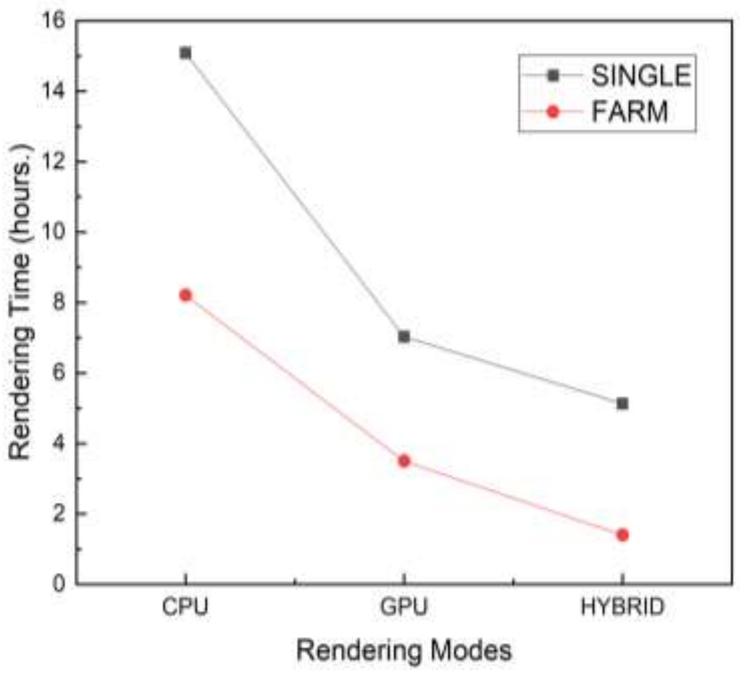
3. Results

The first tests made within the Render farm were done using the CPU, GPU and mixed mode applied to an animation of a ship, which consists of the simulation of fluids and various hard-surface textures due to the type of material that the ship is made of, which makes the animation on its own complicated to render on a conventional computer (Figure 9). This animation was made in Blender which consists of 250 frames, this animation tried to address the most outstanding physical aspects and then compare its efficiency with an individual node; For the comparison with the individual node, a computer was used, which consists of GPU: NVIDIA GeForce RTX 2070 SU- PER, CPU: Intel Core i7-10700K at 3.80 GHz 8 cores with 32 GB of DDR4 RAM and a 500 GB SSD hard drive.

Figure 9 3-D Boat Animation

Source: Authors' Own Creation

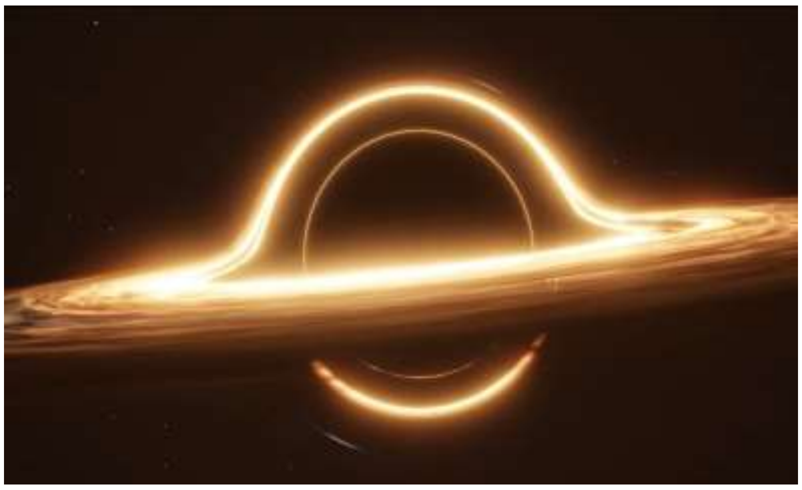
Figure 10 Comparison graph between an individual node and the render farm



Source: Authors' Own Creation

Figure 10 shows the behavior of the 3 main architectures for the elaboration of the render farm compared to an individual node, from which for the three cases we can see an individual decrease in the efficiency of the farm before the individual node, decreasing for each of the three cases the rendering time by almost 50% due to the fact that when using the CPU for the individual mode we have a render time of almost 16 hours, while for the farm we have that the 250 frames were rendered in 8:30 hours, for the case of using the GPUs the time was 8 and 4 hours for the individual node and the farm respectively; while for the case of the hybrid architecture, the time was 5:40 hours and 2:50 hours for the individual node and the farm, respectively, showing that the rendering time decreases to more than 50%, for this case the efficiency of the farm before the individual mode was demonstrated, so that for each of the cases the farm shows a time efficiency.

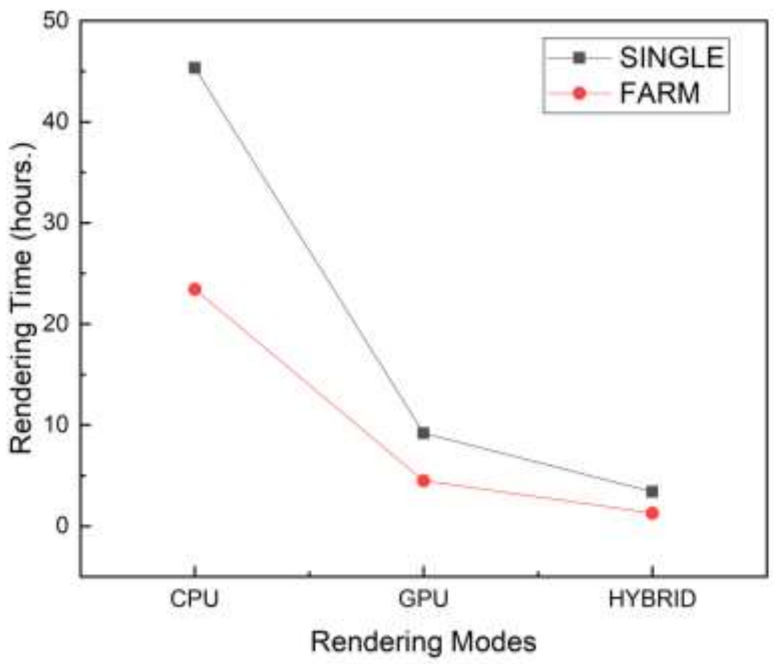
Figure 11 Animation of a black hole



Source: Authors' Own Creation

Figure 11 shows the animation of a black hole which consists of 250 frames, in which we tried to address the most salient physical aspects of where it makes the animation more complicated to render because of all the details that compose it, because we tried to stick to it as realistic as possible. In order to make these effects, the geometry nodes were implemented, which makes our animation heavier, as well as the lighting and shading effects, which are present, in addition to the rotation and vorticity effects.

Figure 12. Comparison graph between an individual node and the render farm



Source: Authors' Own Creation

Figure 12 shows the behavior of the farm before the individual node in rendering times, as can be seen in the graph, the 3 possible forms of architecture were compared when rendering the final project, which is in CPU, GPU and mixed mode; In which for the three cases it was found that the implementation of the render farm reduces about 50% in time for each of the cases, also finding that of the 3 possible forms of rendering, the mixed mode is the one that meets the objective of reducing the rendering times compared to the other two cases. because when the CPU mode is implemented it is 48 hours and 24 hours using the individual node and the farm respectively, for the GPU case the times were 12 and 6 hours respectively, while when using the mixed architecture the render times for such animation were 5 and 2:30 hours, finding that for all 3 cases the efficiency of the farm before the individual node is approximately 50%.

Conclusions

The implementation of a render farm in the culmination of projects that have to do with the world of digital animation, plays an important role because it helps us to make these animations more efficient and enable within the industry, because it contributes to the reduction in the rendering times of very complex animations compared to if you intend to render in individual teams. In this work it was shown that the implementation of a render farm from 4 slaves and a master reduces the rendering time by half compared to an individual node, which allows to reduce costs within animated works and speed up the post-production of such animations, in addition to it was shown that to achieve this the mixed mode architecture is the one that yields the best result because it does not Not only does it reduce the rendering time by half, but it also does it in less time compared to using only CPU or GPU, therefore it is an effective and reliable alternative to emerging production houses with low budgets which will help you develop quality audiovisual content in less time.

References

Lee Huamaní, E., Condori, P., & Roman-Gonzalez, A. (2019). Implementation of a Beowulf Cluster and Analysis of its Performance in Applications with Parallel Programming. International Journal of Advanced Computer Science and Applications(IJACSA), pp. 522–527. 10.14569/IJACSA.2019.0100869

Blender. (12 de Enero de 2023). The freedom to create. Obtenido de Blender: https://www.blender.org/about/

Jaros, M., Strakos, P., Riha, L., & Maly, L. (2019). Interactive rendering with blender cycles for virtual reality using high performance computing clusters. AIP Conf. Proc. 10.1063/1.5114327

Olivares, U., Ferreira, H., Vega, C. A., Cendejas, J. L., & Rosano, G. (2014). Diseño de clústeres computarizados con algoritmos de renderizado para la construcción de modelos tridimensionales a través del uso de herramientas open source. Multiciencias, pp.88-96. https://www.redalyc.org/pdf/904/90430816012.pdf

Peralta, M., & Akwafuo, S. (2022). Freddy Render: A Horizontally Scaled Blender-Based Solution for 3D Graphics Rendering. 2023 Springer Nature. 10.1007/978-981-19-1607-6_73

Seticaya, N. (2019). Implementasi Rendering Farm dengan Teknologi Cluster Computing Menggunakan Back Burner di Laboratorium Multimedia. Jurnal Dinamika Informatika, pp. 41–56. https://jdi.upy.ac.id/index.php/jdi/article/view/6

V. Patil, G., & Santosh L., D. (2017). Distributed rendering system for 3D animations with Blender. IEEE Xplore, pp. 91–98. 10.1109/ICAECCT.2016.7942562

Yao , J., Pan, Z., & Zhang, H. (2009). A Distributed Render Farm System for Animation Production. Entertainment Computing – ICEC 2009. https://doi.org/10.1007/978-3-642-04052-8_31

Yu-Chen, W., Yu-Ting, W., & Yung-Yu, C. (2021). Learning to cluster for rendering with many lights. Association for Computing Machinery, pp 1–10. https://doi.org/10.1145/3478513.3480561

Zhou, Q., & Liu, R. (2016). Strategy optimization of resource scheduling based on cluster rendering. 2023 Springer Nature. 10.1007/s10586-016-0655-9