Mem. S.A.It. Vol. 96, 50

Memorie della

VFX in Blender for scientific visualisation and storytelling

D. De Luca[1] and M.C. Liguori[1]

HPC Department - Cineca, Via Magnanelli, 6/3, 40033 Casalecchio di Reno, Bologna, Italy
e-mail: d.deluca@cineca.it
e-mail: m.liguori@cineca.it

Received: 29-10-2024; Accepted: 04-03-25

Abstract. This paper explores the use of Blender software for creating simulations, visual effects, and other techniques in various research and dissemination projects. The authors present four case studies, categorized by time and budget availability. The paper highlights how Cineca VisitLab adapts its approach based on project constraints, showcasing the versatility of Blender and exploring techniques like point cloud rendering, fluid simulations, and procedural textures.

Key words. Visual Effects, Computer Graphics, Scientific visualisation, Blender, Storytelling

1. Introduction

Over the last 10 years, Cineca VisitLab (Visual Information Technology Laboratory)[1] has moved with increasing conviction towards open solutions, particularly suitable for use on supercomputers and in the educational context. At the centre of the different workflows developed in time is the Blender software, used with great satisfaction both for the creation of content dedicated to scientific visualisation and dissemination, and for projects dedicated to Cultural Heritage. In this paper we will explore, through some recent realisations developed at VisitLab, some possible solutions for the creation of simulations, visual effects and other techniques that may be useful in various areas of research and dissemination.

2. A matter of time and resources

The examples we want to present can be ordered along a pattern of time and budget availability Fig. 1 lists the tools investigated over the years for these tasks.

2.1. Tubke Monumental – When you have enough budget and time

Tubke Monumental project is an ”immersive” gigapixel art installation, led and produced by Kunstkraftwerk (KKW) in Leipzig[2], about

[1] VisitLab website

[2] Kunstkraftwerk website

Fig. 1: Four different projects developed at Cineca VisitLab graphically distributed depending on time and budget availability.

the monumental painting Early Bourgeois Revolution in Germany, by East German painter Werner Tübke[3]. For this project VisitLab Cineca, under the direction of the video artist Franz Fischnaller, collaborated in transforming the scenes from the panoramic painting into 2D and 3D scenes and 3D models unfolding along the walls of the KKW to create the optical illusion of the circular panorama. Among the several solutions deployed we can list Fig. 2: Fracture tools before Calculate Mass function and Rigid Body simulation; Stellarium; Fluid simulation; Dynamic paint wave modifier; Flow mapping and the Adaptation of the movie to the specific camera setup at KKW.

2.1.1. Fracture tools plus rigid body simulation

To develop the scene of the destruction of the tower of Babel, we created a copy of the tower and, using the Blender *Cell Fracture* addon, we fractured it. As a guideline for cutting the mesh, we used the annotation tool to make drawings on the surface of the mesh to precisely guide the cuts. The system created pieces from the guidelines using a recursive Voronoi algorithm. From these pieces, a *Rigid Body* simulation was set up. The pieces were automatically weighed according to volume and the physical characteristics of Granite, which was already present in Blender's internal preset library (*Object* → *Rigid Body* → *Calculate Mass* → Granite (Broken)).
Afterwards, all pieces were joined and connected by constraints in order to use a special Rigid Body option called Breakable, which keeps the tower pieces cohesive and solid until a force above a threshold is applied. In order to achieve organic and realistic destruction, various collision set-ups were therefore animated, with which we hit the tower to start the destruction simulation.

[3] YouTube video

2.1.2. Stellarium

For the sky scene, with the use of the open-source software *"Stellarium"* and some *ad-hoc* coding, the sky was recreated as it was the exact day and time of the battle depicted in the painting. Afterwards, the sky object was separated and rigged to be animated by rotating stars and constellations.

2.1.3. Fluid simulation

The animation and simulation of the *"flood"* was achieved by using the *fluid simulation* features of Blender and is a combination of two simulations: a *circular wave* with a *vortex force* and a river that flows horizontally with some *turbulence force*.

2.1.4. Dynamic paint wave modifier

In the pond scene, some flower petals are floating in a fountain and the trail of waves were created by the petals moving on the water. This effect was achieved with a *Dynamic Paint simulation*, that was used to bake the intersection between the water and the petals in a texture, and then use this texture to drive a displace *wave modifier* with some shader effects on top.

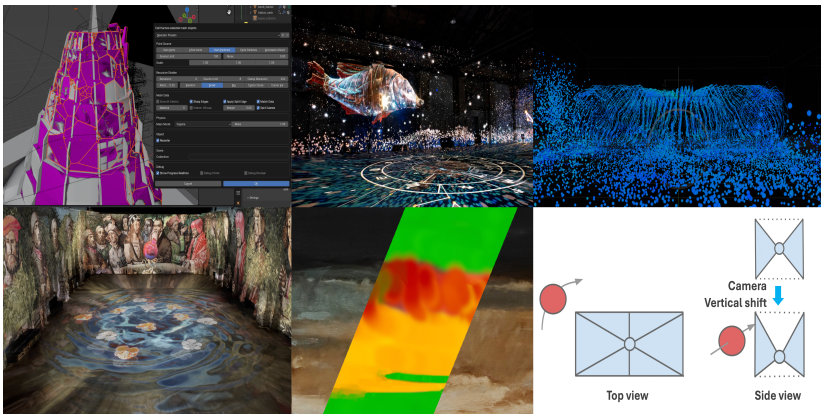

Fig. 2: Fracture tools plus Rigid Body simulation (Fig. 2a); Stellarium (Fig. 2b); Fluid simulation (Fig. 2c); Dynamic paint wave modifier (fig. 2d); Flow mapping (Fig. 2e) and the Adaptation of the movie to the specific camera setup at KKW (Fig. 2f)

2.1.5. Flow mapping

In the ending, there is a psychedelic sky with vortexes, lights and a sandstorm. To make this sky from the painting alive the hypothesis of recreating this scenery with 3d models and traditional animation was discarded, so it was turned to a 2.5D novel approach using the so-called *"flow-maps"*. A flow map is an auxiliary image that can be drawn directly inside Blender where colours encode a motion-field, this motion field is then used inside a custom shader to move the image pixels during the rendering.

2.1.6. Adaptation of the movie to the specific camera setup at KKW

Another topic of interest is the 3d Camera setup. In this show the 3D world is all around the viewer but Eevee, the chosen render engine, does not support a 360-degree camera, so it needed to render the movie with a multiple camera configuration. In order to seamlessly move any 3D object from one wall to another and from any wall to the floor we developed a set of 7 cameras where the frustum of each camera needs to be tangent to its neighbour frustum. In detail, 6 cameras were used for the walls (one on each short wall and two for each of the long ones), plus a single camera for the floor. The short walls cameras have a focal length of 36 mm and the long wall ones have a 18 mm lens, furthermore the four cameras for the long walls use a horizontal off-centered frustum (camera Lens settings → Shift X of ±0.5), resulting in a physical narrower lens then the original 18mm. Every camera, except for the one dedicated to the floor projection, has a vertical off-centered frustum in order to capture the 3D world from the height of a human in the central area of the room. The resulting renders were stitched together with a custom made Python script with an ad-hoc configuration suitable for the KKW immersive environment.

2.2. Leonardo supercomputer – When you have enough budget but little time

For the Grand opening of the Leonardo supercomputer at Cineca[4], at the presence of the President of the Italian Republic Sergio Mattarella, a video trailer was needed in order to briefly introduce the new supercomputer and its many fields of use, such as AI,

[4] Leonardo opening event

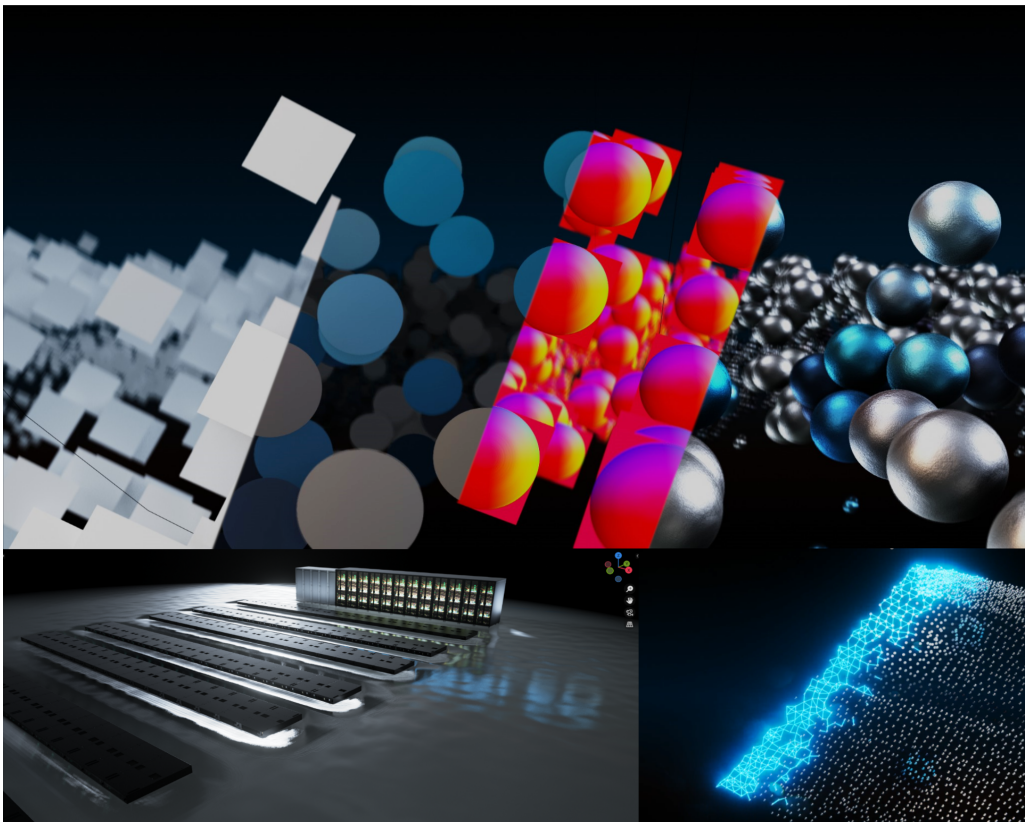

Fig. 3: Some of the effects used in the Leonardo video trailer: Point clouds (Fig. 3a); plexus effect (Fig. 3b); fluid simulation (Fig. 3c).

Astrophysics, Drug design, Geophysics, and several others [5].

The CAD blueprints of the main components were handed down and used to recreate the machine.

Among the several solutions adopted we cite Fig. 3: Point clouds; plexus effect; fluid simulation.

2.2.1. Point clouds

In the initial sequences, in order to visualise the point clouds of the supercomputer as perfect spheres without burdening the rendering made with the light and fast *Eevee* render engine, a system with *Geometry nodes* was created for this visualization. Other solutions, such as the instantiation of highpoly spherical meshes for each vertex of the point cloud, or using the Geometry nodes *Mesh to Points* node, which only works in the *Cycles* render engine, were discarded due to the heavy burden placed on editing and rendering. In the Geometry nodes system, the vertices of the photogrammetry were represented by means of square planes with the Instance on points node, then those planes are rotated towards the scene camera so as to appear as billboards. The material of the planes then inherited features, such as *vertex colors* of the point clouds or the *UVmap* of each plane, from the Geometry nodes. Those features were combined in the shader with a circular stencil mask texture, that acts as alpha,

[5] Leonardo - The European Pre-exascale Supercomputer - Video

to visualize the squares as circles. Then a *bump mapping* texture was added to display shadows and light on the circles, so that they appear as spheres.

2.2.2. Fluid simulation

To create the effect of the supercomputer being born from water, and thus emphasize Leonardo's water-cooled nature, the same *dynamic painting* technique was used as for the pond scene in Tübke Monumental. In this case, the ground is the canvas, while the brush is the Leonardo computer, creating waves as if it were emerging from the water. In this case, a *bake* was performed on a sequence of EXR textures of the floor mesh, and then used to modify the relief effect through combinations of *normal map* and *bump map* textures on the floor shader itself.

2.2.3. Plexus effect

During the appearance of supercomputer meshes from the point cloud, it was thought to use a *3D plexus effect*, which represents the interconnections of vertices that create matter. To obtain a procedural plexus, an ad hoc *Geometry nodes* system was created that, starting from the point cloud, generates *urchin* geometries that are merged by proximity, simplified and, in order to render the plexus, the resulting wireframe was solidified into thin parallelepiped connections. Finally, using an *empty* object, the vertices of the point cloud are filtered to control the plexus effect and make it appear only where it is necessary, and with the same empty, it's possible to manage and animate its position and size.

2.3. Quantum Computing – When you have time but little budget

As Cineca is going to host a quantum computer in the near future, it was decided to realise an introductory video to such a complex topic.

After managing the challenges in defining a storyboard capable of explaining difficult concepts without displeasing neither professionals nor the general public, the first decision was the definition of two environments, aesthetically juxtaposed. One papery white, the space of theories, with the formulas and explanations, and the other velvety black, the world of quantum physics. The latter one giving away also an impression of being far larger than the outside.

To connect the white scene with the black scene via portals, the short film makes extensive use of Blender's scene linking system. In this configuration, the alternating white and black relative files are linked in an additional compositing file called "compo". This file contains the renders and information provided by the white and black scenes, information such as portal masks passed using *Cryptomatte* or gradients and visual noise passed using the *AOV shader function*. These masks and information are used via *ad hoc* node groups to animate the reveal of the portals and the transition between the two worlds (Fig. 4).

In this example we focus on Fig. 5: Interference waves, a Galton board, Baked volumetric Shading.

2.3.1. Interference waves

In the quantum interference scene, a series of *Wave Modifiers* combined on the same surface made it possible to generate the displacement that simulates the combination of waves coming from different interactions of the quantum world. A network of *ad hoc Geometry nodes* was then created to generate dots of different size representing the probability on the curve derived from the interferences: the greater the interference, i.e. the displacement of the curve, the larger the dot, i.e. the probability of finding the desired solution to a problem at that point.

2.3.2. A Galton board

In one scene of the video, to explain the concept of equal probability of getting a 0 or a 1, a Galton board with two outputs was used. To get a working system of falling spheres, *Blender's Rigid Body* feature was used to create a region between two planes where the spheres can fall,

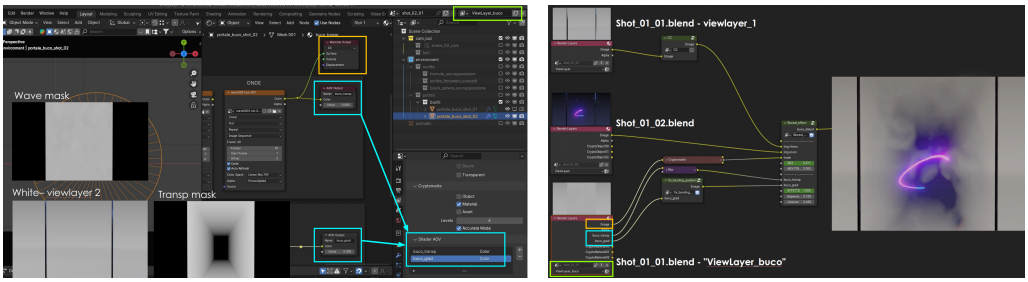

Fig. 4: Portal scene configuration and linking: outputs and masks from a portal shot (Fig. 4a) were used as inputs into the "compo" file (Fig. 4b) in order to mix the black and the white scene with the animated reveal effect of the portal.

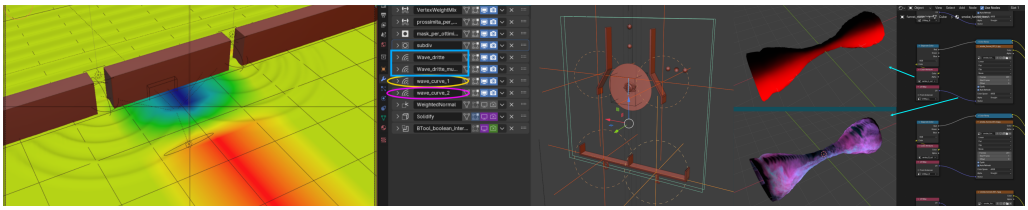

Fig. 5: Some effects adopted in the realisation of the video on Quantum Computing: Interference waves (Fig. 5a), a Galton board (Fig. 5b), Baked volumetric Shading (Fig. 5c)

bounce off an animated peg that acts as a collider, and land at the bottom of the board. *Drag* forces were used to slow down the fall, and the friction of the collisions with the peg and the borders of the board, was used to better control the path of the spheres.

2.3.3. Baked volumetric Shading

A significant part of the video is devoted to quantum particles and their properties. To represent these very peculiar particles, it was decided to imbue them with light and volumetric effects. In order to use the Eevee render engine with these effects, since it was not possible to render them *as-is*, it was necessary to bake the volumes onto sequences of textures, which were then re-applied to the surfaces of the particles, to their internal parts and to the meshes that represent the quantum entanglement links.

2.4. Sebastien – When you have little time and little budget

The last example we present in this contribution is a video teaser realised for the EU project Sebastien, about smart management in livestock farming[6].

In this case the solutions adopted to speed up the production are simpler scenography, without camera movement; everything happens in static scenes. Also, the animation of the farm animals is basic and cartoony, obtained with simple rigs that only involve body, legs and tail when needed[7]. Another big-time saver was a procedural paper system (Fig. 7) that could create an infinite number of backgrounds. It was created using shaders and empty objects to control the paper characteristics. The paper system can change color and grain of the paper background. Small and large wrinkles could be adjusted and the paper wear around the wrinkles could be controlled.

[6] Sebastien project website
[7] YouTube video

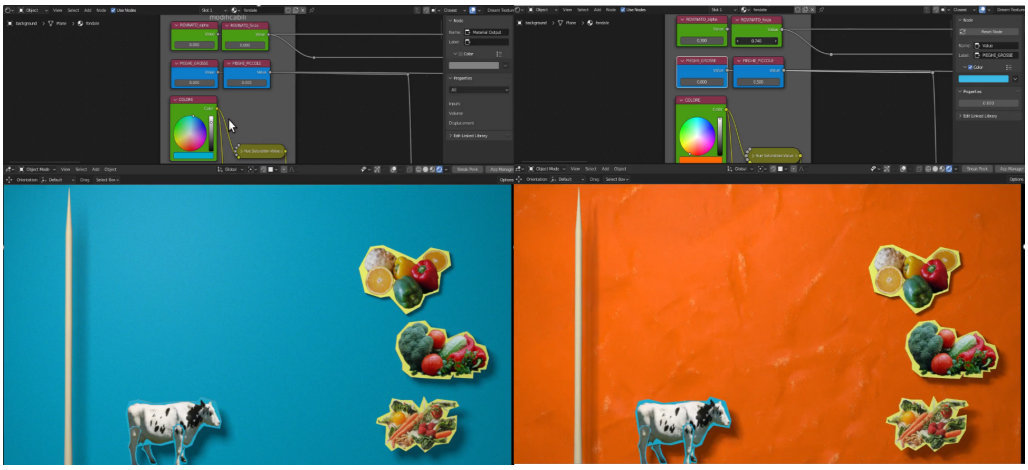

Fig. 6: A procedural paper system developed for the Sebastien project video.

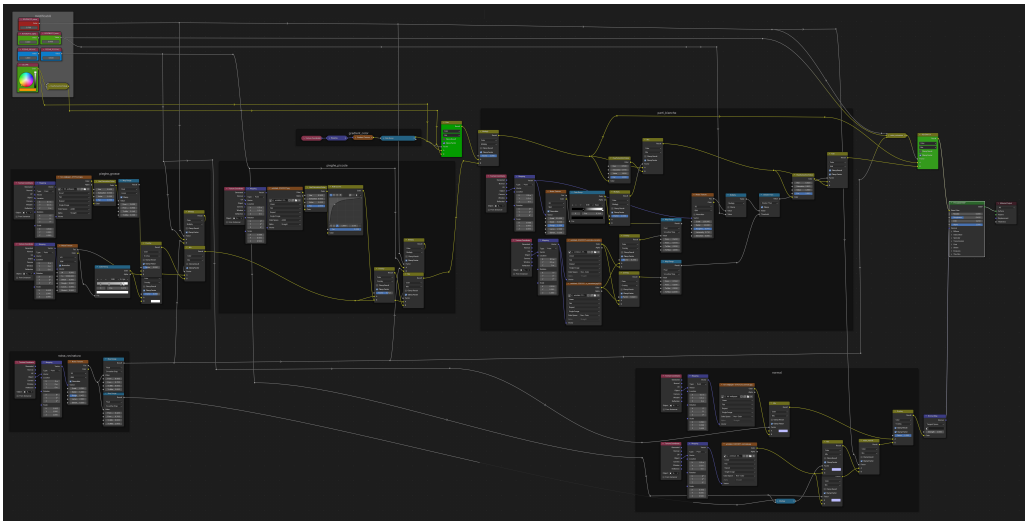

Fig. 7: Shader nodes for the customizable paper material.

(Fig. 6).

3. Conclusions

The interindipendent constraints Time/Resources/Quality can be addressed in multiple ways. The management of the point clouds in the opening scene of Leonardo supercomputer video, for example, was a solution for optimising rendering time. The little more time needed in order to create the scene was justified by the possibility of having that scene, with such quality, at all.