



GRADO EN INGENIERÍA MULTIMEDIA



VNIVERSITAT  
DE VALÈNCIA

TRABAJO FIN DE GRADO

---

COMPLEMENTO PARA BLENDER CON  
HERRAMIENTAS PARA LA CREACIÓN DE EFECTOS  
ESPECIALES Y ANIMACIONES

---

AUTOR: RAFAEL POLOPE CONTRERAS

TUTOR: RAFAEL JAVIER MARTÍNEZ DURA

SEPTIEMBRE 2021





UNIVERSITAT  
DE VALÈNCIA



Escola Tècnica Superior  
d'Enginyeria ETSE-UV

## TRABAJO FIN DE GRADO

---

# COMPLEMENTO PARA BLENDER CON HERRAMIENTAS PARA LA CREACIÓN DE EFECTOS ESPECIALES Y ANIMACIONES

---

**AUTOR: RAFAEL POLOPE CONTRERAS**

**TUTOR: RAFAEL JAVIER MARTÍNEZ DURA**

---

## TRIBUNAL

PRESIDENTE/A:

VOCAL 1:

VOCAL 2:

FECHA DE DEFENSA:

CALIFICACIÓN:



**Declaración de autoría:**

Yo, Rafael Polope Contreras, declaro la autoría del Trabajo Fin de Grado titulado “Complemento para Blender con herramientas para la creación de efectos especiales y animaciones” y que el citado trabajo no infringe las leyes en vigor sobre propiedad intelectual. El material no original que figura en este trabajo ha sido atribuido a sus legítimos autores.

Valencia, 2 de septiembre de 2021

Fdo: Rafael Polope Contreras



---

### **Resumen:**

Este trabajo consiste en el desarrollo de un complemento para el software de desarrollo de productos en 2D y 3D, Blender, para facilitar la creación de animaciones y elaboración de efectos especiales a través de un conjunto de herramientas con las que se podrá crear, de manera directa, ciertos tipos de efectos que se pueden integrar en las escenas. Estos efectos o funcionalidades son cuatro hasta el momento: creación de un rayo, efecto de transición de wireframe (hilo de alambre), creación de un rayo láser y creación de un escudo de fuerza.

Es un complemento que se podrá instalar Blender y usar su paquete de herramientas accediendo a estas a través de un panel principal que se integra en la interfaz. Este panel contiene diferentes subpaneles correspondientes a los diferentes efectos en los que se encuentran agrupados los parámetros para poder producirlos junto a los botones para ejecutar la herramienta de ese subpanel y crear el efecto correspondiente. El complemento está escrito en el lenguaje Python. Además, los objetos creados para generar los efectos tendrán un subpanel dentro del panel de objetos de la interfaz de Blender con el que se podrán modificar las propiedades de los objetos generados para los efectos, que además, permitirá insertar keyframes en las propiedades de manera que se puedan animar.

---



---

### **Agradecimientos:**

En primer lugar quiero agradecer a las personas que me han acompañado durante todo este período y que me han ayudado a llegar hasta este punto, en especial a mis padres, que no hay palabras para decir lo que suponen en mi vida y cómo de importantes han sido para poder conseguir llegar hasta aquí, y a mis familiares y amigos por sus ánimos y preocupación por mí.

En segundo lugar también a mis compañeros por su apoyo y ayuda en los momentos que lo he necesitado y a los profesores que me han brindado el apoyo, el respaldo y su paciencia, y que además, me han enseñado nuevas metas y retos a los enfrentarme, así como nuevas aspiraciones.

Por último quiero agradecer a el resto de personas que he conocido durante estos cuatro años y que han hecho de este período algo más que cuatro años universitarios. Personas que quizá sólo ya visto una vez pero que me han ido enriqueciendo, que me han hecho madurar, aprender a ver de manera distinta lo que me rodea y a crecer como persona.

Gracias a todos por estos cuatro años, de corazón.

---



# Índice general

<b>1. Introducción</b>	<b>15</b>
1.1. Introducción . . . . .	15
1.2. Motivación . . . . .	16
1.3. Objetivos . . . . .	17
1.4. Metodología de desarrollo . . . . .	18
1.5. Organización de la memoria . . . . .	19
<b>2. Estado del arte</b>	<b>21</b>
2.1. Tecnologías . . . . .	21
2.2. Análisis de herramientas similares . . . . .	33
2.2.1. Páginas web: librerías de vfx online . . . . .	33
2.2.2. Herramientas de Blender (Addons) . . . . .	36
2.2.3. Herramientas de 3DS MAX (Plugins) . . . . .	42
2.2.4. Herramientas de Maya (Plugins) . . . . .	46
2.2.5. Herramientas de Cinema 4D (Plugins) . . . . .	50
2.3. Conclusión . . . . .	53
<b>3. Requisitos, especificaciones, coste, riesgos, viabilidad</b>	<b>55</b>
3.1. Requisitos . . . . .	55
3.1.1. Requisitos funcionales . . . . .	55
3.1.2. Requisitos no funcionales . . . . .	56
3.2. Especificaciones del sistema . . . . .	57
3.2.1. Especificación hardware . . . . .	57
3.2.2. Especificación software . . . . .	57
3.3. Planificación temporal . . . . .	58
3.3.1. Desglose de tareas . . . . .	58
3.3.2. Estimaciones temporales . . . . .	59
3.4. Estimación de costes . . . . .	64
3.4.1. Costes directos . . . . .	64

3.4.2. Costes indirectos . . . . .	67
3.5. Viabilidad . . . . .	68
3.5.1. Viabilidad económica . . . . .	68
3.5.2. Viabilidad legal . . . . .	68
3.5.3. Viabilidad técnica . . . . .	69
3.6. Riesgos . . . . .	70
3.6.1. Análisis de riesgos . . . . .	70
<b>4. Desarrollo del proyecto</b>	<b>71</b>
4.1. Análisis . . . . .	71
4.1.1. Análisis de casos de uso . . . . .	72
4.1.2. Diagramas de actividad . . . . .	88
4.2. Diseño . . . . .	93
4.2.1. Diagrama de clases . . . . .	93
4.2.2. Diagramas de secuencia . . . . .	95
4.2.3. Diseño de la interfaz . . . . .	99
4.3. Implementación . . . . .	100
4.3.1. Implementación del add-on . . . . .	100
4.3.2. Implementación efecto de rayo . . . . .	106
4.3.3. Implementación efecto de transición de wireframe . . . . .	110
<b>5. Pruebas</b>	<b>123</b>
5.1. Pruebas . . . . .	123
5.1.1. Pruebas funcionales . . . . .	123
5.1.2. Pruebas de rendimiento . . . . .	145
5.1.3. Pruebas de usabilidad . . . . .	152
<b>6. Conclusiones</b>	<b>157</b>
6.1. Conclusiones . . . . .	157
6.2. Trabajo futuro . . . . .	158
<b>7. Anexo</b>	<b>159</b>
7.1. Planificación temporal y diagrama de Gantt . . . . .	159
7.2. Manual de instrucciones de VFX Tool . . . . .	161
7.2.1. Efecto del rayo . . . . .	161
7.2.2. Efecto de transición wireframe: . . . . .	164
7.2.3. Efecto de rayo láser: . . . . .	165
7.2.4. Efecto de escudo de fuerza: . . . . .	166

Bibliografía

**167**



# Capítulo 1

## Introducción

### 1.1. Introducción

Actualmente el mundo de los gráficos por ordenador se ha convertido en uno de los que más rápido y más ha evolucionado en los últimos años y, a día de hoy, este proceso de evolución aún continúa. En la actualidad, la gran mayoría de los productos comerciales se anuncian haciendo uso de técnicas audiovisuales, por lo que el sector audiovisual, que engloba todos los campos como la animación 2D y 3D, los efectos visuales o VFX, los efectos por capas de posprocesado o after effects, etc. se encuentran en un momento de auge, impulsado por su uso y su importancia creciente en el mercado, pues un producto que “entra bien por los ojos” siempre es más atrayente y ahí es donde juega un papel crucial el mundo del software gráfico, incluso algunas profesiones ajenas hace unos años a la informática gráfica como la medicina hacen uso de softwares gráficos para recreación de escenarios virtuales. La ciencia y las simulaciones son otro ejemplo de campo donde se han empezado a aplicar los gráficos recientemente, y en general la visualización de datos ha sido un motor para el impulso de estas tecnologías. También ha contribuido a aumentar enormemente el consumo de productos multimedia el crecimiento del sector del entretenimiento.

Cada vez son más las personas que consumen contenido digital dirigido al entretenimiento como series, películas y videojuegos, en los cuales se hace uso de la animación, los VFX y demás técnicas audiovisuales. Esto aumento del consumo ha sido propiciado, en gran parte, propiciado por los fenómenos surgientes relacionados con los creadores de contenido en redes sociales y la pandemia de la COVID-19 que ha impulsado el consumo de estos productos digitales y multimedia. Esto supone un aumento de las empresas que buscan producir este tipo de producto y contenido multimedia y la competitividad en el sector, por lo que, a la vez que aumenta el uso de esta tecnología aumenta el número de desarrolladores que hacen uso de las herramientas que les facilitan el trabajo para producir contenido multimedia.

Como ya he dicho anteriormente, la informática gráfica ha evolucionado mucho y muy rápido en los últimos veinte años, atendiendo a las demandas del consumo de productos que requerían de uso de gráficos cada vez de mejor calidad. Los VFX, o efectos visuales, son uno de esos recursos multimedia que han aumentado su demanda y, por tanto, el número de personas y empresas interesadas en ellos. Es por este auge y el crecimiento del software de código abierto que surgió la idea de esta propuesta.

## 1.2. Motivación

Los miembros de las comunidades de cierto software desarrollan herramientas para usarla en proyectos, bien personales o bien laborales y estos usuarios de las comunidades a menudo comparten sus trabajos y sus herramientas con lo que se produce feedback entre las comunidades y los usuarios, haciendo más fácil el desarrollo de trabajos con un software, cuya comunidad facilita el trabajo, y que, por tanto, hace que este software se vuelva más popular y más útil, pues fomenta el aumento de especialistas en él, permitiendo a las empresas encontrar personal más cualificado. Como se puede entender de esto último, las comunidades aportan enriquecimientos a los desarrolladores, al software del cual se genera la comunidad y esto beneficia a las empresas que encuentran especialistas en creación de productos multimedia.

Como ya se ha dicho en el apartado de introducción, las herramientas para software de código abierto son un producto demandado en la actualidad pues permite realizar productos multimedia de manera más eficiente sin tener que repetir trabajo, que quizá ya ha sido realizado por alguien que también hace uso de las mismas tecnologías y que pertenece a la misma comunidad de desarrollo. El motivo de realizar esta herramienta es justamente ese: crear una herramienta útil para el desarrollo de animaciones y VFX para que pueda ser utilizada por profesionales freelance y/o empresas que quieran hacer uso de los efectos que se pueden producir con esta herramienta ahorrando el tiempo y dinero que supondría crearlos ellos mismos desde la base.

La aplicación escogida para crear esta herramienta ha sido Blender, pues es un software de código abierto que, como dije antes, a menudo estas aplicaciones o softwares de código abierto cuentan con una comunidad muy grande detrás que favorecen al propio software y a los desarrolladores que trabajan con él. Esto ha sido un motivante para hacer la herramienta para este software pues ya hay muchas empresas que hacen uso de este software, por los resultados que produce y por la cantidad de expertos que hay en él, propiciado por su comunidad como ya dije, y me propuse como objetivo hacerla para Blender porque me ayudaría a mejorar mis conocimientos y habilidades sobre este software que es demandado en la actualidad, que es de código abierto, con lo cual puede hacer uso de esta herramienta cualquiera que esté interesado, bien para uso profesional o principiante.

El lenguaje que he utilizado ha sido Python aprovechando que Blender tiene un módulo de Python integrado que permite escribir dentro de la propia aplicación el código para la creación de herramientas y dispone de una API propia.

La creciente demanda de productos multimedia, de desarrolladores de estos productos y de herramientas para desarrollarlos, así como la mejora de los resultados que se consiguen con aplicaciones de software libre como Blender y el crecimiento de sus comunidades ha sido el principal motivante para escoger este trabajo de fin de grado, que se enfoca al mercado de productos multimedia y aprovecha las facilidades del software libre para el desarrollo de un producto, como he dicho, demandado en la actualidad como son las herramientas para softwares gráficos.

### 1.3. Objetivos

El objetivo principal de este trabajo es crear una herramienta que permita a los animadores y artistas del mundo de la animación y los VFX hacer uso de ella para crear efectos sin tener que tener unos conocimientos técnicos sobre las propiedades en los que se basan, preocupándose únicamente de los resultados sin tener que verse envueltos en los entresijos y complejidades técnicas que pueda suponer el proceso de creación y saltándose la curva de aprendizaje necesaria para llegar a tener la habilidad suficiente para ser capaces de producir los efectos deseados; sin tener los conocimientos apropiados o suficientes, ni sobre la propia aplicación con la que están trabajando, Blender en este caso, ni sobre los elementos que necesiten usar para poder crear los efectos deseados.

Los objetivos subyacentes del proyecto en referencia a la programación de la herramienta son los siguientes: conseguir que los efectos se integren perfectamente en una escena a través de los parámetros que se establecen en la interfaz para controlar sus resultados finales, conseguir el mayor abanico posible de usos de los efectos para que el usuario que los utilice en el mayor número de escenas posibles, es decir, que sean versátiles; que sean intuitivos de usar y que no compliquen más el proceso de creación de lo que lo favorecen y por último que sean lo más baratos posibles en términos de consumo de memoria, dado que algunos de ellos deben crear texturas y de tiempo de ejecución, pues quizás el número de operaciones a realizar por la herramienta de creación del rayo, por ejemplo, para originar una mesh y realizar las modificaciones en esta para dar lugar al efecto pueden requerir una exigencias de rendimiento que quizás no todos los equipos son capaces de correrlas ya que unas de las desventajas que tiene Python es que de por sí es más lento que lenguajes como C++, y si a eso le añadimos la complejidad en términos de coste temporal del algoritmo nos da este problema como resultado. Por esto último la búsqueda de la optimización de los algoritmos es un objetivo principal.

## 1.4. Metodología de desarrollo

Una metodología de desarrollo de software es un marco de trabajo que se usa para estructurar, planificar y controlar el proceso de desarrollo de un sistema. Existen diferentes metodologías que se pueden seguir para llevar a cabo un proyecto de desarrollo de software y aunque para este proyecto en concreto no se ha seguido una metodología estrictamente se podría considerar que se ha seguido una metodología mixta, pues se han seguido fases de desarrollo como en un modelo en cascada, donde para poder comenzar la siguiente etapa de desarrollo se debía terminar la anterior y se han llevado a cabo prototipos de los efectos a generar, con el fin de comprobar paso a paso las funcionalidades de cada uno de ellos antes de incorporarlos al proyecto completo, lo que es una práctica propia de la metodología de prototipos. También cabe destacar que para el proyecto se ha usado un sistema de control de versiones: el sistema subversion (SVN), donde usaba un repositorio local para guardar las versiones del proyecto, además de guardar copias periódicamente cada cinco días en la nube. Se ha optado por subversion como sistema de control de versiones ya debido a que es un proyecto desarrollado únicamente por mí y, por tanto, no se necesitaba de repositorios locales intermedios como en GitHub lo que facilitaba el uso del control de versiones.

## 1.5. Organización de la memoria

En este apartado se explica cual es la estructura del proyecto y de qué trata cada uno de los capítulos de la memoria.

En la Introducción se explica a grandes rasgos en qué consiste el proyecto, cuál es su motivación y cuáles son los objetivos a los que se pretende llegar.

Seguidamente, el Estado del arte hace una revisión de las tecnologías similares a las empleadas, alternativas posibles y soluciones similares o relacionadas con el la que se pretende dar en esta propuesta.

En el apartado de Especificación se definirá cuáles son los recursos utilizados para llevar a cabo el proyecto, se definirán los requisitos que se han de cumplir y se hará una estimación de los costes, tanto temporales como económicos, se hará un análisis de la viabilidad y de los riesgos que conlleva el proyecto.

En el Desarrollo del proyecto se especificará cómo se ha llevado a cabo el sistema, mostrando detalles de la implementación y del análisis y diseño previos.

En Pruebas y resultados se mostrará el funcionamiento del complemento y se mostrarán los resultados obtenidos usando diferentes parámetros.

Por último en el capítulo de Conclusiones y trabajo futuro se comentarán posibles mejoras y añadidos que se podrían desarrollar y en qué líneas se puede seguir desarrollando este trabajo.



# Capítulo 2

## Estado del arte

### 2.1. Tecnologías

Con el auge de los gráficos 3D se han desarrollado multitud de programas y tecnologías para llevar a cabo trabajos de este ámbito. A continuación se explicarán los más populares y utilizados de la actualidad y qué beneficios e inconvenientes tiene cada uno.

Posteriormente se harán comparaciones con el software utilizado para este trabajo y porqué se ha utilizado Blender en lugar de las otras opciones detalladas en este apartado.

#### ■ AUTODESK MAYA

Maya es un programa dedicado al desarrollo de gráficos 3D por ordenador, efectos especiales, animación y de dibujo. Posee diversas herramientas para animación, modelado, renderización, simulación de ropa y cabello, fluidos, etc. Se caracteriza por su potencia y las posibilidades de expansión y personalización de su interfaz y herramientas pues se puede personalizar cada herramienta a su gusto. MEL (Maya Embedded Language) es el código que forma el núcleo de Maya gracias al cual se pueden crear scripts y personalizar el paquete. Es un software flexible pues posee soporte para los lenguajes C++ y Python ademas del MEL. [1]

La gran ventaja de Maya radica en su arquitectura, basada en una red de nodos llamada gráfico de dependencias, el cual se ve afectado por prácticamente todo lo que se haga en el programa haciéndolo más complejo eliminando secciones de este. Maya está bien considerado dentro de las industrias cinematográficas y de videojuegos por ser un software destacable por su animación y manipulación de personajes. Gracias a su herramienta MASH, permite al usuario generar efectos visuales rápidamente.

Tiene de desventaja más notable que el tiempo de aprendizaje de este software es más largo que el de otros softwares como el 3DMax y que es un software de pago. Aunque es más sencillo de usar que Houdini y más intuitivo que Blender. [2]

## ■ CINEMA 4D

Cinema 4D es un software profesional de creación de gráficos y animación 3D. Se utiliza al igual que Maya para animación, simulación y renderizado 3D. Posee un conjunto de herramientas rápido, potente y flexible que hace que el trabajo sea eficiente en cuestiones de diseño gráfico, gráficos en movimiento, VFX, AR/MR/VR, desarrollo de juegos, etc.[3]

Sus principales virtudes son una muy alta velocidad de renderización, una interfaz altamente personalizable y flexible, y una curva de aprendizaje muy vertical que permite aprender mucho en poco tiempo.[4]

Una de las características más destacables de Cinema 4D es la modularidad, gracias a la cual permite añadir módulos especializados independientes en función de las necesidades del proyecto a realizar.

De este software destaca MoGraph, un conjunto de herramientas de animación y modelado de procedimientos que brinda a los diseñadores de movimiento la capacidad de crear animaciones complejas y abstractas de manera rápida y sencilla. Con MoGraph se puede clonar objetos, agregar efectos, crear movimiento de manera simple y la tecnología de Cineware, que integra escenas 3D de Cinema 4D en After Effect. Este sistema fue reconocido por la Academia de Artes y Ciencias Cinematográficas con un Premio al Logro Técnico en 2019. [5]

Como aspectos negativos se pueden destacar que es un software de pago y que es utilizado por muchos usuarios, por lo que ya son demasiado notorias las producciones hechas con este programa. Como apunte final, cabe destacar que existe rivalidad entre Maya y Cinema 4D ya que ambos software son reconocidos dentro de la industria de los gráficos y la animación 3D y son dos de los más populares y utilizados a día de hoy.[6]

## ■ UNREAL ENGINE

Unreal Engine es un motor de juego creado por la compañía Epic Games, mostrado inicialmente en el shooter en primera persona Unreal en 1998. Aunque se desarrolló principalmente para los shooters en primera persona, se ha utilizado con éxito en una variedad de otros géneros, incluyendo videojuegos de sigilo, lucha, MMORPG y otros RPG. Con su código escrito en C++, el Unreal Engine presenta un alto grado de portabilidad y es una herramienta utilizada actualmente por muchos desarrolladores de juegos.

La versión más estable es Unreal Engine 4, el cual fue lanzado en 2014 bajo un modelo de suscripción. Desde 2015, puede descargarse gratuitamente, con su código fuente disponible en GitHub.[7]

El sistema de efectos visuales Niagara de Unreal Engine 4 se utiliza para crear y previsualizar efectos de partículas en tiempo real. Con Niagara, el artista técnico

tiene la capacidad de crear funcionalidades adicionales por sí mismo, sin la ayuda de un programador.[\[8\]](#)

En el sistema Niagara VFX, hay cuatro componentes principales:

- **Sistemas** Son objetos contenedores para múltiples emisores, todos combinados en un solo efecto.
- **Emisores** Los emisores son contenedores para módulos. Se pueden crear simulaciones utilizando la pila de módulos y luego renderizar esa simulación de varias formas en un mismo emisor.
- **Módulos** Los módulos Niagara son el nivel básico de Niagara VFX. Los módulos hablan de datos comunes, encapsulan comportamientos, se apilan con otros módulos y escriben funciones. Se crean utilizando un lenguaje de shaders de alto nivel (HLSL).
- **Parámetros** Los parámetros son una abstracción de datos en una simulación. Hay de cuatro tipos: Primitivo, Enum, Estructura e interfaces de datos

[\[9\]](#)

Unreal tiene algunas ventajas distintas sobre muchas de las soluciones de renderizado tradicionales, ya que es capaz de renderizar cuadros de alta calidad de 4k extremadamente rápido debido a su naturaleza como motor de juego en tiempo real. Además, incluye Sequencer, una herramienta de animación y edición cinematográfica en tiempo real totalmente no lineal diseñada para la colaboración. permite definir y modificar la iluminación, el bloqueo de la cámara, los personajes y el vestuario por frame. Incluso puede grabar animaciones de captura de movimiento vinculadas a personajes de la escena y desde fuentes externas trackeadas directamente a Sequencer para reproducción futura.

Como desventaja principal, a pesar de sus buenas prestaciones para render no es bueno para realizar animaciones por sí solo, por ello es habitual combinarlo con otros softwares de animación como Maya y después usar los resultados; los proyectos pesan bastante, por lo que exige unos requisitos mínimos de capacidad del disco si queremos trabajar en varios proyectos con Unreal, la comunidad de Unreal es pequeña ya que no hay mucha gente que use o conozca este software, y no se pueden realizar juegos estilo 2D de gran calidad.

## ■ NUKE

Nuke es un software de composición digital. Se utiliza para la posproducción en el cine y la televisión.

NUKE está disponible para Microsoft Windows, Mac OS X y Linux. [\[10\]](#)

Contiene más de 200 tipos de nodos para abordar todo tipo de creación visual. Las herramientas de composición de Deep Image le permiten crear y trabajar con imágenes que contienen múltiples muestras de opacidad, color y profundidad relativa a la cámara por píxel, por lo que no es necesario volver a renderizar los elementos cuando cambia el contenido. También posee un conjunto de herramientas de aprendizaje

automático que permite a los artistas crear y aplicar sus propios efectos específicos de alta calidad. Como parte de esto, los artistas pueden entrenar redes neuronales para completar tareas automáticamente. [11]

Existe una familia de software de Nuke, Nuke Studio, Hiero y Hiero Player conformando el ecosistema Nuke que permiten a los supervisores de efectos visuales, artistas de efectos visuales, editores de efectos visuales, productores y coordinadores optimizar sus flujos de trabajo de revisión, al compartir líneas de tiempo sin problemas para una mayor visibilidad y control de los proyectos. [11]

Nuke tiene un workflow con mucha mejor integración con programas 3D, como Maya, que otros programas de posproducción, como After Effects, del que hablaremos después. Además, es un poco más estable y resiste trabajar en más resolución que After Effects, con archivos de 32k y más. [12]

Los principales problemas de este programa de edición son que solo permite trabajar con canales RGBA; los núcleos complicados pueden provocar tiempos de espera de controlador en la GPU si tardan demasiado en ejecutarse, así que cuanto menor sea la especificación de su GPU, es más probable que esto suceda; se puede producir choques y bucles infinitos dado que posee un nodo llamado BlinkScript que permite escribir y ejecutar código arbitrario, y es posible bloquear o bloquear Nuke, si antes no se ha asegurado que este código no produce bucles y es un código seguro de ejecutar.[13]

A pesar de esto es el software más utilizado en la industria profesional para posproducción y se ha utilizado en superproducciones como Jurassic World o Los vengadores.

## ■ SUBSTANCE PAINTER

Substance painter es un software de pintura 3D que permite texturizar y renderizar meshes 3D.

Es un software especializado que permite crear la apariencia de los objetos, dándoles texturas. Es un software enfocado a la parte gráfica pero no es utilizado para efectos especiales ni animación solamente enfocado a la parte estética. Es un software parecido en cuanto a uso a photoshop, salvando las diferencias en cuanto a que photoshop se utiliza para imágenes y este para meshes 3D. Es un software de pago.

## ■ PHOTOSHOP

Adobe Photoshop es un software de pago de edición de fotografías usado principalmente para la manipulación de gráficos y fotografías. Se usa en la industria de la animación y los gráficos como herramienta para la edición de imágenes en posproducción. Soporta varios modelos de colores como el RGB, CYMK, CIELAB, colores

sólidos y semitonos.

Utiliza un sistema de capas que permite componer una imagen añadiendo capas y haciendo retoques sobre estas para crear una composición donde se superponen y se suman las adiciones u otras operaciones que se realicen sobre las demás capas.

Phtoshop puede tratar imágenes de una gran variedad de formatos: PSD, PSB, Post-Script, EPS, DCS, BMP, GIF, JPEG, TIFF, PICT, PNG, PDF, ICO, IFF, PCX, RAW, TGA, Filmstrip, JPEG2000, FlashPix y Scitex CT. [14]

Es el programa líder en diseño gráfico y ha evolucionado mucho a lo largo de los años. A día de hoy es un programa que incorpora las nuevas tecnologías, como la inteligencia artificial, para hacerlo más completo y añadirle nuevas funciones como: agregar filtros a las fotografías, crear luces y efectos especiales, agregar texturas, realizar ilustraciones en 3D y realizar vídeos en GIF.

Como aspectos más negativos de este software cabe destacar que es caro comparado con otros programas de edición de imágenes y requiere uno aprendizaje previo antes de poder hacer uso de él ya que es difícil de aprender.

#### ■ ZBRUSH

Zbrush es un programa de pago utilizado en la industria de los gráficos como editor de modelos 3D, permitiendo modificar modelos con una técnica de esculpido de los mismos como si se dibujara sobre ellos, lo que ha hecho que se popularizara dentro de la industria, y ha sido usado en producciones como "Underworld", "Avatarz "El señor de los anillos."entre muchas otras.

ZBursh permite crear modelos 3D muy detallados esculpiendo modelos importados o creados en el propio programa a partir de primitivas. Uno de los problemas a los que se podía enfrentar este software es el de quizá no se pudiera exportar todo el nivel de detalle que permite esta herramienta para en los formatos actuales de modelos 3D, por lo que permite exportar junto con los modelos unos paquetes con mapas de normales y/o de desplazamiento, incluidos los desplazamientos vectoriales para poder provechar todo el detalles que este ofrece.

Además de las funciones ya nombradas se pueden destacar como aspectos positivos de este software, además, que el programa ha existido desde hace mucho tiempo, y tiene una gran base de usuarios, con una gran cantidad de recursos que apoyarse y que aunque es relativamente caro, actualizaciones siempre han sido libres. Por otra parte como desventajas se puede decir que es un programa caro y que su curva de aprendizaje es poco vertical, por lo que puede ralentizar un tanto el trabajo de los artistas si no tienen experiencia en su uso. [15]

#### ■ AFTER EFFECTS

Affter Effects es un software para el desarrollo de vfx, gráficos en movimiento y composición de imágenes popular por su uso en posproducción. Este programa fue galardonado con el premio de la Academia por el logro científico y técnico en 2019. [16]

A diferencias de otros softwares para posproducción, como Photoshop, este programa se utiliza en archivos de vídeo y es un software basado en línea de tiempo, lo que quiere decir que controlamos qué es lo que queremos que aparezca en el resultado final en el momento que indique la barra que muestra la duración del vídeo cuando estamos modificando nuestro archivo fuente.

After Effects permite crear títulos, efectos especiales, animar , etc. la diferencia principal con Nuke, por ejemplo, que también es un software usado para posproducción es que con After Effects se trabaja de manera lineal con un sistema de capas, similar a Photoshop, mientras que Nuke opera con un sistema basado en nodos, como ya se habló anteriormente cuando se definió Nuke. Esto hace que para aplicar un efecto en After Effects se deba hacer aplicando ese efecto a cada elemento al que se lo quieras aplicar, mientras que Nuke controla ese efecto con un nodo de manera centralizada, por lo que para aplicar ese efecto a un elemento solo hay que conectarlo al nodo y para alterar ese efecto solo hay que manipular dicho nodo. Esto tiene sus ventajas y desventajas, pues en de la manera en que se ha en After Effects puedes modificar el efecto de manera particular para cada elemento al que se lo quieras aplicar, mientras que con Nuke esto supondría crear nuevos nodos y complicar más el árbol de nodos. [17]

After Effects es un programa muy utilizado en todo el mundo por sus múltiples ventajas. Alguna de ellas ya las hemos nombrado como su capacidad de generar animaciones 2D y 3D y de una manera rápida o los efectos visuales, gráficos en movimiento y composiciones que se pueden hacer con él, pero además, también permite hacer tracking de cámara , codificación cromática, interfaces de usuario, tiene capacidad de mejorar sus prestaciones gracias a funciones adicionales utilizando complementos externos de red Giant, entre otros; varias opciones de edición de audio, gran flexibilidad, permite comandos en Python para usuarios avanzados, puede crear cualquier efecto VFX usando sistemas de partículas y físicas y además tiene una gran compatibilidad con los demás softwares de la familia Adobe como Photoshop, Premiere, Media encoder, Illustrator y con Cinema 4D y Mocha, por lo que tiene una integración enorme dentro de cualquier estudio de la industria audiovisual. En cuanto a los inconvenientes se puede decir que requiere de sistemas de alta gama para que funcione sin problemas, tiene una curva de aprendizaje lenta, por lo que es difícil de aprender para la gente que no esté familiarizada y es un programa de pago, que aunque es muy completo, es bastante costoso. [18]

## ■ AUTODESK COMBUSTION

Es un software de para la aplicación de gráficos en movimiento, composición y efectos visuales. Forma parte de una familia de sistemas de postproducción a la que pertenecen también los programas de Autodesk como Flame, Smoke e Inferno. Tie-

ne un funcionamiento similar al de After Effects en que es un sistema basado en capas. Con este programa se trabaja la animación con módulos. Con este software de posproducción se puede aplicar corrección de color, manejo de máscaras, keyers y tracking de cámara. [19]

Es un software de que permite edición y animación basada en vectores, por lo que sus resultados son independientes de la resolución en la que se vean.

Como características claves de este software se pueden nombrar la composición, por supuesto, los efectos de partículas, pintura, edición, captura, tracking, herramientas para películas, el renderizado y el workflow. Permite crear espacios de trabajo personalizados permitiendo realizar variedad funciones dentro del espacio como composiciones, proyectos de pintura y efectos visuales al mismo tiempo.

El sistema de partículas 2D de Combustion ha sido un estándar de la industria durante muchos años, con una biblioteca de emisores de partículas predefinidos, los usuarios pueden disfrutar de un control total sobre la forma, densidad, flujo y cada parte de las propiedades y formas de las partículas. [20]

## ■ FUSION

Fusion es una herramienta de composición digital para especialistas en animaciones gráficas, efectos visuales y elementos tridimensionales. Cuenta con una interfaz intuitiva que permite lograr efectos sofisticados gracias a diferentes herramientas para el procesamiento de imágenes. Fusion ofrece múltiples prestaciones para crear animaciones, títulos y vfx. Es un software que ha sido utilizado, como otros nombrados anteriormente, para largometrajes de Hollywood.

Este software posee una estructura de nodos para crear sus composiciones. Cada nodo representa un efecto, filtro u otra operación. Durante el procesamiento de imágenes pueden conectarse o combinarse con otras herramientas y objetos para crear intrincados efectos visuales. [21]

Fusion dispone de una gran variedad de funciones que permiten llevar a cabo todo tipo de proyectos, ya sea que impliquen realizar superposiciones, seguir la trayectoria de un objeto, retocar imágenes, animar títulos o generar efectos con partículas. El programa incluye un área de trabajo tridimensional y herramientas para llevar a cabo composiciones con contenidos estereoscópicos y elementos de realidad virtual (Permite usar gafas especiales para ver los resultados en tiempo real). Además, permite combinar efectos con animaciones gráficas, crear modelos en 3D y, por supuesto, hacer renders. Otras de las funciones que tiene Fusión son las de creación de máscaras mediante técnicas rotoscópicas y funciones vectoriales para aplicar efectos y correcciones a partes específicas de la imagen. [22]

La aplicación es compatible con los lenguajes de programación Lua y Python. De este modo, es posible ahorrar tiempo generando secuencias de comandos para crear

herramientas personalizadas, intercambiar datos entre Fusion y otros programas, automatizar tareas repetitivas o incluso añadir funciones nuevas. Uno de los inconvenientes de este software al igual que con After Effects es que es muy caro y difícil de aprender al principio y de dominar.

## ■ HOUDINI

Houdini es una aplicación de software de animación 3D desarrollada por SideFX. Houdini es un programa que es utilizado principalmente para la creación de efectos visuales en películas y juegos y ha sido utilizado por las principales empresas de efectos visuales como Walt Disney Animation Studios , Pixar , DreamWorks Animation, etc. Está escrito en C++ y Python. [23]

El flujo de trabajo de Houdini está basado en nodos, donde cada acción se almacena en un nodo. Estos nodos se conectan a redes que definen una "receta" que se puede modificar para refinar el resultado y luego se repite para crear resultados similares pero personalizados. La capacidad de los nodos de guardarse y pasar información, en forma de atributos, a lo largo de la cadena es lo que le da a Houdini una naturaleza procesal. Esta naturaleza procesal permite crear simulaciones dinámicas y de partículas sofisticadas. Además, los nodos en Houdini se pueden encapsular y compartir entre proyectos de Houdini. Este software también permite incorporar efectos de calidad cinematográfica en tiempo real utilizando partículas, dinámicas de cuerpos rígidos fracturados, fluidos entre otros recursos.

En definitiva, podemos decir que la naturaleza procesal de Houdini es lo que hace característico a este software ya que, al tener un workflow basado en nodos, como ya hemos mencionado, permite a los artistas crear objetos detallados en un número relativamente corto de pasos, a diferencia de otros programas. Aunque es conocido por sus efectos especiales, comprende todas las herramientas que se espera encontrar en un importante programa 3D. Houdini ofrece modelado y animación geométrica estándar a través de fotogramas clave. El programa viene con el motor de renderizado Mantra, pero también es compatible con motores de renderizado de terceros como Renderman. Los personajes se pueden agrupar en un solo nodo de activos para que los utilice un equipo de animación determinado. [24]

Este programa está enfocado para artistas visuales con formación técnica, pues tiene una curva de aprendizaje y requiere de unos conocimientos mínimos de matemáticas y programación para sacar su máximo partido.

Es uno de los softwares más importantes en la industria día de hoy para realizar vfx y por ello su licencia es de las más caras el mercado, sólo apta para estudios serios de animación, aunque hay una edición de aprendizaje libre, y una versión independiente para los estudios que ganan menos de 100.000\$ al año que se les hace un descuento. [25]

## ■ V-RAY

V-Ray es un motor de renderizado usado como extensión para algunas aplicaciones de gráficos computacionales, como 3ds max, maya, modo, sketchUp, Nuke, entre otros.[26]

Es un motor de renderización que usa técnicas avanzadas, como por ejemplo algoritmos de Iluminación Global tales como Path Tracing, Mapeo de Fotones, Mapas de Irradiación y Fuerza bruta, siendo esta última la opción principal establecida en sus versiones recientes ( 3.0) por su precisión y reducción del tiempo de render. Los renders generados con estas técnicas se ven más reales, como los efectos de iluminación que son emulados de manera más realista. Por esto, es utilizado tanto en la industria de los videojuegos, como cinematográfica y para arquitectura, usado para renders ultrarealistas de modelos. [27]

Es un software, como hemos dicho, destinado a complementar otros programas para mejorar la calidad de los renders con lo que se puede dar mejor calidad a los VFX creados con las herramientas que incorporen V-Ray a sus resultados.

## ■ UNITY

Unity es uno de los softwares de la industria de los gráficos y gráficos en tiempo en tiempo real más populares de la actualidad. Este motor de juego multiplataforma está disponible como plataforma de desarrollo para Microsoft Windows, Mac O y Linux. Cuenta con dos versiones: Unity Professional y Unity Personal.

Es un software de uso libre, sin licencia, que nació con la idea de ''democratizar<sup>el</sup> desarrollo de videojuegos y hacerlo accesible a un número mayor de personas.

Unity puede usarse junto con programas de modelado como Blender, ZBrush, 3ds Max, Maya, Cinema 4D, etc. para modificar los objetos de la escena y actualizando dichos objetos en Unity son necesidad de reimportar nada.

El motor gráfico de Unity utiliza OpenGL en Windows, mac y Linux; Direct3D sólo en Windows y OpenGL ES en Android y iOS. Su lenguaje para creación de shaders es el ShaderLab y pueden escribirse con él tres tipos de shaders: de superficie, de vértice y de fragmento. Unity permite modificar el comportamiento de componentes mediante scripts que hacen posible la interactividad con eventos en la escena. Estos scripts pueden estar escritos en tres lenguajes: UnityScript (un lenguaje personalizado inspirado en la sintaxis ECMAScript), C# o Boo (que tiene una sintaxis inspirada en Python). [28]

Actualmente Unity cuenta con una herramienta llamada Visual Effect Graph que consiste en un entorno de trabajo con un workflow basado en nodos para crear VFX al estilo de Nuke o Houdini. Esta herramienta está disponible a partir de la versión de 2018.3 con la que se nos permite hacer uso de este grafo para la realización de efectos visuales. Su aparición permite una alternativa a los sistemas de partículas de Unity a la hora de crear efectos especiales, los cuales pueden llegar a afectar

bastante al rendimiento de la aplicación. La diferencia básica entre un sistema y otro radica en el uso del hardware. Mientras que los sistemas de partículas utilizan la CPU, el grafo de VFX hace uso de la GPU. Además, este grafo de efectos hace uso del High Definition Render Pipeline (HDRP), que no siempre es la mejor opción para ordenadores que carezcan de los requisitos apropiados o dispositivos móviles. VFX nos provee de un sistema de creación basado en nodos y bloques, parecido al Shader Graph de Unity. [29]

Por último, en cuanto a resumen de las ventajas de Unity podemos decir que es un software muy rápido para crear juegos, tiene un proceso de importación fácil y rápido, soporte para la eliminación de errores que permite mostrarlos e tiempo de ejecución, vastos mercados y comunidades de juegos Unity 3D que ofrecen amplios componentes de sonido, física, renderizado, controles, etc. El AssetStore recuerda mucho a cualquier tienda de aplicaciones para teléfonos, multiplataforma... En cuanto a las desventajas se puede destacar el consumo de memoria de los proyectos de Unity, que puede ser problemático para dispositivos móviles; como no se proporcionan códigos fuente, los problemas de rendimiento son difíciles de encontrar, abordar y solucionar, no hay soporte para HTML5 WebGL por ahora, cambiar los objetivos de compilación requiere volver a importar todo y tomar tiempo cuando se trabaja en un juego multiplataforma [30].

## ■ MAMOSET TOOLBAG

Marmoset Toolbag es una aplicación de renderizado en tiempo real con editor de materiales, animaciones y renderización. Por ello, cuenta con herramientas para renderización 3D, animaciones y configuración de luces, sombras e iluminación en tiempo real. Este software suele utilizarse principalmente antes de la producción o en posproducción y dispone, además, de herramientas para crear imágenes y vídeos realistas. Está pensado para que veas todos los cambios sobre materiales, efectos visuales e iluminación, ambiente, oclusión y deep of field al momento y así conseguir acabados profesionales. También dispone de shaders realistas y un editor de escenas con los que podremos conseguir efectos avanzados, oclusión ambiental, iluminación en tiempo real, importar animaciones en malla a través de archivos FBX y controlar su animación con fotogramas clave, reflejos, etc. [31]

Las características principales de este software son las siguientes:

- Shader de piel con Skin Shader 2.0
- Teselación con DirectX3D 11
- Soporte de shader con Anisotropía
- Iluminación IBL con soporte de sombras, especular, piel y anisotropía
- Herramienta para creación de entornos sky tool.
- Sistema de luces dinámicas con múltiples efectos, incluyendo el volumétrico
- Efectos de postproducción “Next Gen” DOF, 3D y lens flares
- Anitialiasing mediante Supersampling
- Super Screenshots con Canal Alpha para su posterior composición

- Captura de video
- Conjunto de presets

Como inconveniente cabe destacar que es un software que maneja cosas como la especulación y la construcción de materiales de manera muy diferente a los motores de juego completos en el mercado, la forma en que se ve su modelo en Marmoset no es necesariamente la forma en que se verá cuando finalmente a la plataforma donde se vayan a utilizar los activos. Marmoset se considera un renderizador independiente para creación de imágenes de nivel portfolio, por lo que no es considerado un software de producción en sí mismo. Es un software de pago pero bastante asequible.[\[32\]](#)

## ■ AUTODESK 3DS MAX

Autodesk 3ds Max (anteriormente 3D Studio Max) es un programa de creación de gráficos y animación 3D desarrollado por Autodesk y posee una arquitectura basada en plugins.

El software les permite simular las propiedades físicas de líquidos como agua, aceite y lava. Además, 3ds Max tiene controladores de animación que los diseñadores pueden crear, modificar y compartir. [\[33\]](#)

El software también es muy útil para el diseño de edificios, infraestructura y construcción, así como para el desarrollo de productos y la planificación de la fabricación, por ellos es muy utilizado también en las industrias inmobiliaria y arquitectónica lo utilizan para generar imágenes fotorrealistas.

3ds Max se utiliza a menudo para el modelado y la animación de personajes. El software puede manejar varias etapas del proceso de animación, incluyendo previsualización, diseño, cámaras, modelado, texturizado, rigging, animación, efectos visuales, iluminación y renderizado. También proporciona a sus usuarios herramientas robustas que les ayudan a gestionar y editar sus animaciones que podrán aplicar a juegos de ordenador, películas y emisiones, ilustraciones médicas o presentaciones forenses.

3ds Max se dirige sobre todo a los diseñadores de arquitectura y a los diseñadores de videojuegos. En cuanto a rigging y animación, 3ds Max dispone de todo lo necesario para el trabajo profesional. Por lo tanto, es utilizado a menudo por animadores profesionales que trabajan en películas de gran presupuesto, películas indie, o incluso anuncios comerciales más pequeños que necesitan un poco de movimiento en 3D.

3ds Max es a menudo comparado con Maya entre los de la industria creativa.

Mientras que Maya es generalmente más poderoso en la mayoría de las áreas, 3ds Max es más fácil de usar especialmente en lo que se refiere a modelado.

A pesar de las facilidades y versatilidad de este software se le pueden apuntar un par de contras y es que a pesar de que es algo más fácil de usar que Maya, sigue

teniendo una curva de aprendizaje bastante horizontal y tiene una licencia bastante costosa. [34]

## ■ BLENDER

Finalmente definiremos y pasaremos a explicar porqué se ha usado este programa y no otra de las opciones ya mencionadas anteriormente.

Blender es un programa multiplataforma, dedicado especialmente al modelado, iluminación, renderizado, la animación y creación de gráficos en 3D. También sirve para la composición digital utilizando la técnica procesal de nodos, edición de vídeo, escultura y pintura digital.

El programa fue inicialmente distribuido de forma gratuita pero sin el código fuente, posteriormente pasó a ser software libre. Actualmente es compatible con todas las versiones de Windows, macOS, GNU/Linux (incluyendo Android), Solaris, FreeBSD e IRIX. [35]

Además de las funcionalidades que ya posee es posible añadir nuevas creando scripts propios ya que Blender está diseñado como un software “extensible, que como indica la propia palabra, quiere decir que puede extender sus funcionalidades.

Una de las características que ha hecho destacar a este software es que, además de permitir crear modelos a partir de primitivas, como todos los programas de modelado, también tiene la capacidad de combinar Nurbs o B-Splines mediante operaciones booleanas. Esto ha permitido realizar modelados morfológicamente más exactos mediante scripts. [35]

Blender ofrece características avanzadas como despliegue UV, texturas y, por supuesto, animación 3D con rigging, mezcla y formas. Este software es obviamente capaz de compilar para un renderizado excelente, gracias a la GPU además de la CPU. También es posible realizar todo tipo de simulaciones físicas de cuerpos flexibles, fluidos o incluso partículas para conseguir efectos 3D cada vez más sorprendentes. [36]

A pesar de tener tantas características, Blender es un programa bastante ligero, ya que sólo pesa de 200 a 300 MB dependiendo de la versión de cada momento.

En cuanto a la experiencia de usuario, Blender cuenta con un gran menú de atajos de teclado que hace que el trabajo con este software sea rápido y ágil y cuenta con una interfaz totalmente personalizable y flexible pudiendo organizar una misma ventana diferentes entornos de trabajo en los que realizar trabajos distintos como scripting, modelado, etc.

Además cuenta con una inmensa comunidad y tutoriales y cursos gratuitos que permiten aprender rápidamente y perfeccionar el manejo del programa.

Históricamente, Blender se basaba en el motor Blender Internal Hybrid, que desde la versión 2.61 ha sido sustituido por el motor Cycles. Blender siguió evolucionando para ofrecer en la versión 2.8 un renderizado en tiempo real gracias al motor Cycles con EEVEE.

El programa ahora incluye Freestyle para generar líneas 2D en un dibujo 3D, por ejemplo para la gestión de contornos, y es compatible con otros motores externos.[\[37\]](#)

Blender se ha utilizado para crear animaciones de películas de éxito como “Spider Man 2” o “Friday or another day”. Es esto último lo que ha permitido a Blender construir una reputación aún más fuerte, pues sabemos que todos los efectos especiales han sido creados en Blender y utilizando la plataforma de software libre GNU/Linux.

## 2.2. Análisis de herramientas similares

El cine, las series, animación, videojuegos, etc. han dado paso a un mercado de productos audiovisuales muy competitivo y ha generado muchos productores de estos recursos multimedia que han dado lugar a una gran variedad de herramientas, complementos y paquetes de utilidades para crear VFX los cuales usar en producciones audiovisuales. A continuación, se va a hablar sobre la mayoría de ellos.

### 2.2.1. Páginas web: librerías de vfx online

Estás librerías son plataformas donde hay recursos VFX en vídeo, los cuales tienen un fondo de transparencia de manera que cuando te descargas el recurso y lo integras en tu escena sólo veas el efecto que querías integrar.

El principal inconveniente de este tipo de librería de VFX es que, al tener efectos prefabricados, estos pueda no terminar de encajar con la escena al no tratarse de elementos nativos de la propia escena y problemas de iluminación, por ejemplo, se tendrían que compensar con horas de postproducción. Mientras que en una escena normal los VFX son un añadido a algo ya existente, estos deben implementarse adecuándose a la escena, sin embargo aquí se ha de tener en cuenta como encaja el efecto y quizás en algunos casos adecuar la escena en consecuencia. La principal ventaja de este tipo de efectos es que permite integrar efectos con una gran calidad dado que están hechos con un software aparte del que se usa para componer el resto de la escena y evita las limitaciones que este pueda suponer para producir dichos efectos, además de evitar la necesidad de personal cualificado que los produzca ni software específico y especialistas en dicho software para poder producirlos.

Las principales diferencias entre los recursos que puede proporcionar esta plataforma online y mi propuesta es que con mi herramienta se pueden originar VFX desde la propia escena sin necesidad de ser especialista en VFX porque solo hay que introducir los parámetros que se desean, en función del resultado buscado, de manera intuitiva. Además

mi herramienta permite usar efectos teniendo en cuenta elementos de la propia escena, es decir, el efecto forma parte de la escena como cualquier otro elemento con lo cual se integra de manera total y no hay problemas de encajes con esta.

Los más reconocidos actualmente son los siguientes:

#### ■ ACTIONVFX

actionvfx.com es una página web con un gran stock de recursos de vídeo en los que aparecen ya desarrollados VFX en una gran variedad de estilos. Desde efectos de energía, explosiones, fuego, etc. a otros efectos con partículas, de roturas de materiales, de sonido, de gente y multitudes, etc. Todos ellos están sobre un fondo con transparencia de manera que cuando integres el archivo de vídeo a tu escena, únicamente verás el efecto que querías integrar. Se ha utilizado en algunas superproducciones como "Jumanji". Tiene tanto recursos de pago como algunos gratuitos. [38]

#### ■ FXELEMENTS

fxelements.com es un sitio web similar a actionvfx.com ya que también es una plataforma con un stock de VFX en vídeo con un fondo transparente. Tienen los efectos clasificados por categorías y agrupados en packs para su distribución comercial. Cada pack se puede enviar almacenado en discos duros externos.

#### ■ PRODUCTIONCRATE

productioncrate.com es una plataforma con productos desarrollados por la industria de Hollywood. Usa VFX en HD y Motion Graphics utilizando códecs sin pérdida (que conservan el color y los detalles en bruto verdaderos). [39]

#### ■ RODYPOLIS

La empresa RodyPolis, a través de rodyropolis.com, también distribuye paquetes de VFX para películas de acción. En RodyPolis se podrán encontrar archivos de efectos para artistas con fugas de luz, efectos de tinta, audio basado en acción e incluso material de archivo de lapso de tiempo. RodyPolis también ha creado cosas como efectos de fuego basados en ventanas de tamaño real. Además, RodyPolis ha desarrollado un nuevo paquete de efectos para explosiones y fuego, como resultado de la demanda y en respuesta a errores que los artistas denunciaban en otros paquetes de efectos. El paquete que están desarrollando para subsanar estos problemas es ActionVFXTM, que a su vez contiene subpaquetes con los distintos efectos de fuego y explosiones.

Algunas características de estos nuevos paquetes de ActionVFXTM son:

- **Los efectos permanecen en el marco . Sin cortes en los bordes:** los efectos que se cortan son inútiles para la mayoría de las tomas amplias, por lo que estos paquetes ActionVFX TM los mantienen a todos en el marco.

- **Entregado en resolución 4K y 2K a 60 fps o más:** la resolución y la velocidad de fotogramas más altas permiten a los usuarios hacer más con los efectos. Cuando se trata de VFX, las opciones siempre son bienvenidas.
- **Incendios de ventana / puerta / pared :** algunos Fire FX de este paquete tienen formas de ventana y puerta.
- **Filmado en LOG y expuesto correctamente :** los efectos que se recortan debido a una exposición incorrecta ofrecen poca flexibilidad en la publicación. RodyPolis está tomando medidas adicionales para garantizar que todas las imágenes se filmen y expongan correctamente.

[40]

## 2.2.2. Herramientas de Blender (Addons)

Existen una gran variedad de complementos para Blender para usar herramientas integradas al igual que la propuesta de este trabajo para realizar efectos visuales, desde efectos de partículas, a simulaciones de líquidos, fuego y humo, etc. Estas herramientas permiten a través del control de determinados parámetros producir efectos visuales integrados en tu escena de Blender ahorrando tiempo en su producción y dando resultados de calidad. A continuación se van a explicar algunos de los más usados y con mejor reputación. [41]

### ■ PARTICLES LINK

Esta herramienta permite crear efectos. Conecta las partículas entre sí, creando una malla. También se puede utilizar para generar curvas y trazador de partículas.

El enlace de partículas funciona de dos modos de gráfico que puede ser una conexión de red o de varias líneas. Lo bueno de este complemento es su capacidad para conectar partículas en vivo y dar retroalimentación mientras trabajas en algo. Además, es posible utilizar el sistema de partículas múltiples. Es una herramienta de pago. [41]



Figura 2.1: particleslink add-on.

## ■ SHOT MATCHER

Este complemento está dirigido a cuadrar los elementos de la escena con el fondo. En ocasiones aunque la iluminación sea la correcta y corresponda con la del fondo, no se consigue que la integración de los elementos sea total, por ello, este complemento te permite comprar fusionar el renderizado con la imagen de fondo. Es un complemento útil para agregar imágenes de fondo al render o efectos visuales de gradación de color. También es de pago. [41]



Figura 2.2: shot matcher add-on.

## ■ WATERIAL

Este complemento permite crear superficies de agua animadas con menor demanda de memoria y unas demandas de rendimiento menores que los efectos creados por simulaciones.

Este complemento tiene crea un material el cual es englosado en un Node-Group, el cual, además, se puede modificar permitiendo realizar cambios al gusto. Es muy útil y barato en términos de requisitos de memoria y rendimiento para el diseño de escenas de costa. Es un complemento de pago como los anteriores.

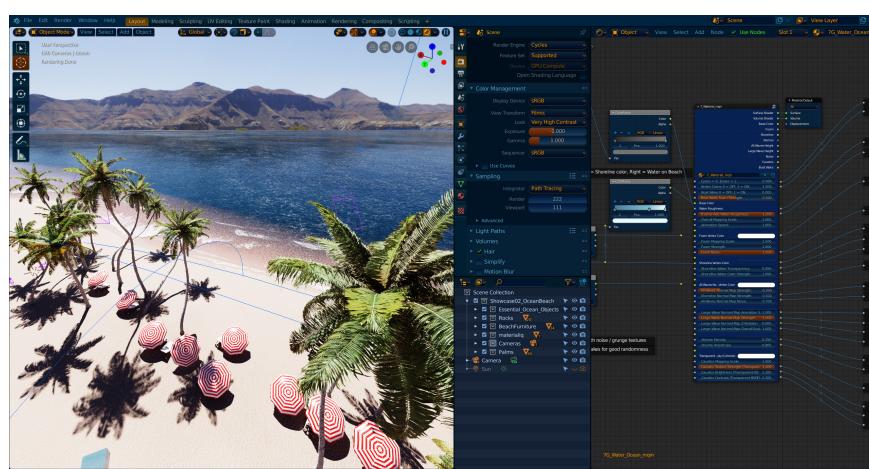


Figura 2.3: waterial add-on.

### ■ BLAZE

Es un complemento que permite crear shaders cinematográficos de fuego y humo. Permite ahorrar tiempo en lidiar con los materiales para lograr un buen fuego. Se utiliza para recrear incendios, explosiones, explosiones nucleares, de ciencia ficción, fogatas, etc. [41]



Figura 2.4: blaze add-on.

### ■ ADVANCED OCEAN MODIFIER

Esta herramienta sirve para crear superficies de agua interactiva de forma rápida, pudiendo hacer así que los artistas creen océanos animados. Incluye tres Presets para incorporar características de clima; permite crear clima tranquilo, tormentoso y una tercera opción que consigue un punto intermedio entre ambos. Permite modificar todo esto usando la configuración.

En general, este complemento permite crear olas oceánicas animadas con efectos de espuma. También puede hacer que los objetos 3D floten en la superficie del océano y creen espuma. Como inconveniente tiene que no alcanza un gran nivel de realismo como sí se puede conseguir con Houdini, Bifrost o Phoenix. Es una herramienta de pago. [41]



Figura 2.5: advanced ocean modifier add-on.

## ■ NEBULA GENERATOR

Esta herramienta es uno de los complementos de Blender para simulaciones y efectos visuales que ha existido durante un largo período de tiempo. pero ahora, aprovechando el sistema volumétrico mejorado de EEVEE, se volvió mejor. Es, en parte, fruto de la comunidad, cuyos resultados y logros fueron recogidos en un paquete para crear esta herramienta.

Tiene una gran cantidad de parámetros y configuraciones que puede cambiar para ajustar el aspecto de la nebulosa como el color, las texturas del gas o el polvo galáctico, el número, brillo y distribución de las estrellas que se crea automáticamente mediante un sistema de partículas.

Uno de los principales problemas que presenta este complemento es la representación de animaciones, que no es problema del complemento sino del de la naturaleza de cómo funciona el sistema volumétrico en sí a la hora de realizar animaciones.

Hay dos generadores que vienen con este complemento, uno para crear nebulosas en 3D y otro para 2D, para efectos de nebulosas panorámicas de fondo en cielos.

Tiene tiempos de renderizado rápido incluso a altas resoluciones. Además, al animar los parámetros, se puede animar aparezcan como si se estuvieran moviendo. también se puede utilizar para generar texturas en otros elementos. Es un complemento de pago y general bastante buen recurso a pesar del handicap de la animación. [41]



Figura 2.6: nebula generator add-on.

### ■ KHAOS

Es una herramienta que permite a los artistas crear explosiones que suceden en un corto período de tiempo.

Incluye explosiones de fuego, de humo y desechos de partículas como vidrio, rocas, ramas, tierra, metralla, piezas de madera, etc. Todo esto permite agregar mucho realismo a la explosión. Además, también incluye una base de datos de más de 35 modelos y escaneos de escombros de explosiones. Los escombros de explosión se pueden agregar dentro de la simulación de partículas personalizadas. Se puede renderizar los campos de escombros por separado para combinarlos con el compositor de efectos.

Es un de los mejores complementos de Blender para explosiones y simulación de efectos, por lo que como todos los demás es de pago para su uso. [41]



Figura 2.7: khaos add-on.

### ■ MOLECULAR SCRIPT

Este complemento de colisión de partículas hace que las partículas choquen para crear ciertos tipos de simulaciones y efectos. Es uno de los complementos de Blender para simulación y efectos visuales que son totalmente gratis y es el que viene por defecto en el panel de sistema de partículas al seleccionar un objeto. [41]

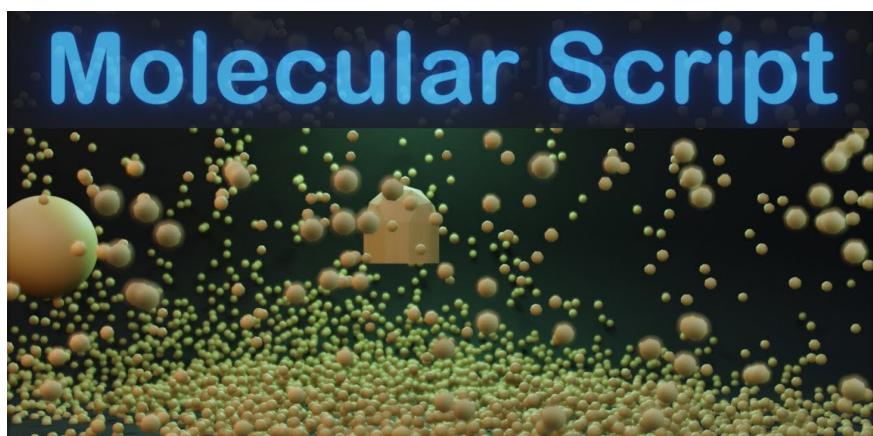


Figura 2.8: molecular script add-on.

### ■ KHAOS

Este complemento permite “reconfigurar” una malla de un sistema de partículas en tiempo real. Se puede usar en combinación con el complemento anterior, molecular Script, ya que están desarrollado por el mismo desarrollador.

Permite hacer efectos tales como efectos de rebote usando un cuerpo blando, como telas, y objetos sólidos como bolas; efectos de derretimiento o fusión de cuerpos sólidos, efectos de lágrimas etc. También es una herramienta gratuita. [41]

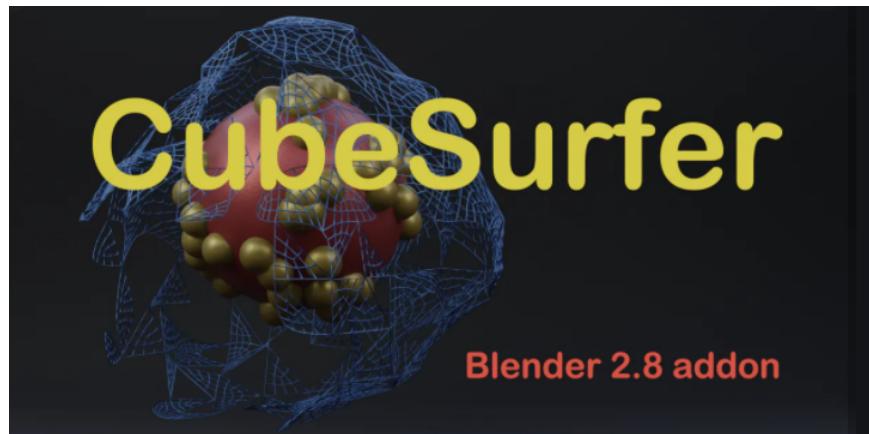


Figura 2.9: cube surfer add-on.

### ■ FLIP FLUIDS

Flip fluids es un complemento que ayuda a configurar, ejecutar y renderizar efectos de simulación de fluidos. Tiene un motor de fluidos personalizado que se basa en la técnica de simulación FLIP que también puede encontrarse en otras herramientas profesionales como Houdini, Phoenix FD, Bifrost y mantaflow. Es uno de los complementos que hace de Blender competitivo en simulaciones de fluidos para la industria. Se pueden simular y generar millones de partículas de espuma, burbujas y spray para dar sensación de realismo a grandes masas de agua. Cuenta además con un control de viscosidad para simular líquidos más ligeros o espesos que se combinan y se enrollan, o cualquier punto intermedio entre estos fenómenos. Es de pago. [41]

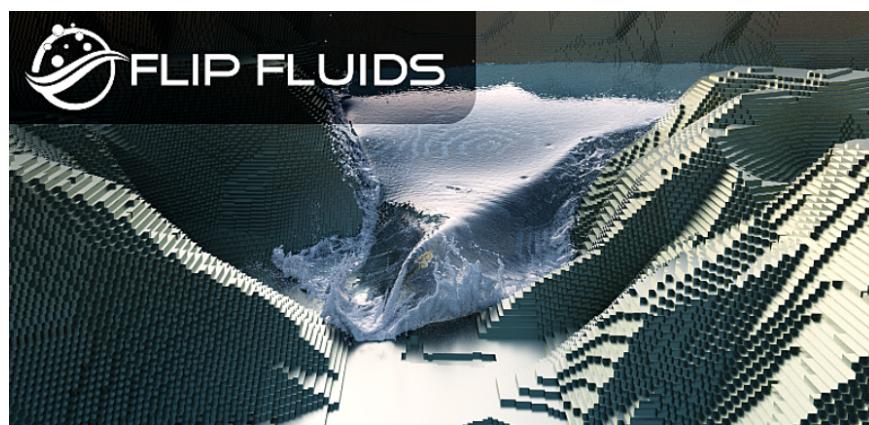


Figura 2.10: flip fluids add-on.

- **NEONER**

Es un complemento que facilita la tarea de crear letreros que simulan las luces de neón. Sirve para crear carteles, lámparas, indicadores luces, etc. Es uno de los principales complementos para usar cuando se desee recrear escena nocturnas o futuristas. Esta optimizado para el motor de render EEVEE, pero también se puede utilizar con Cycles. Es de pago. [41]

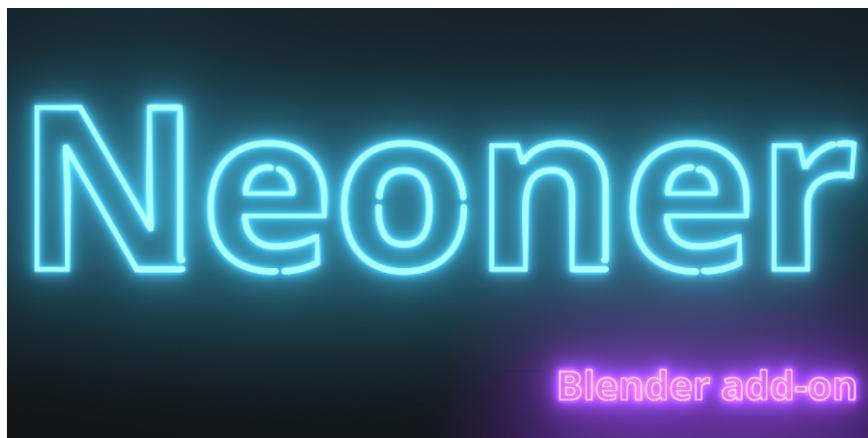


Figura 2.11: neoner add-on.

### 2.2.3. Herramientas de 3DS MAX (Plugins)

- **FUMEFX**

Una de las herramientas más utilizadas en el Cine, TV y animaciones 3D en general que te permite crear explosiones, manipular fuego y humo con un control increíble, sus físicas son precisas y generan comportamientos reales de sistemas de partículas gaseosos.

Es muy versátil y viene en versiones para 3ds Max y Maya, su costo puede resultar elevado para artistas freelance sin embargo es una herramienta que no puede faltar en tu arsenal, simple, poderosa y con un potencial enorme. [42]



Figura 2.12: fumefx plugin.

## ■ THINKING PARTICLES

Es una herramienta de CEBAS Technologies, y es una herramienta que se encarga de producir y manejar cantidades enormes de partículas, físicas aplicadas a las mismas y opciones adicionales. La característica de este Plugin es que no es un sistema de partículas basados en eventos si no en reglas por lo que el comportamiento es mas preciso y real.

Se utiliza en la mayoría de las películas que contienen VFX hoy día en Hollywood y el cine en general, películas como The Avengers, Transformers y muchas más son un claro ejemplo de lo que puede lograrse con este sistema. Es un plugin de pago con un costo apto para estudios de animación y películas. [43]

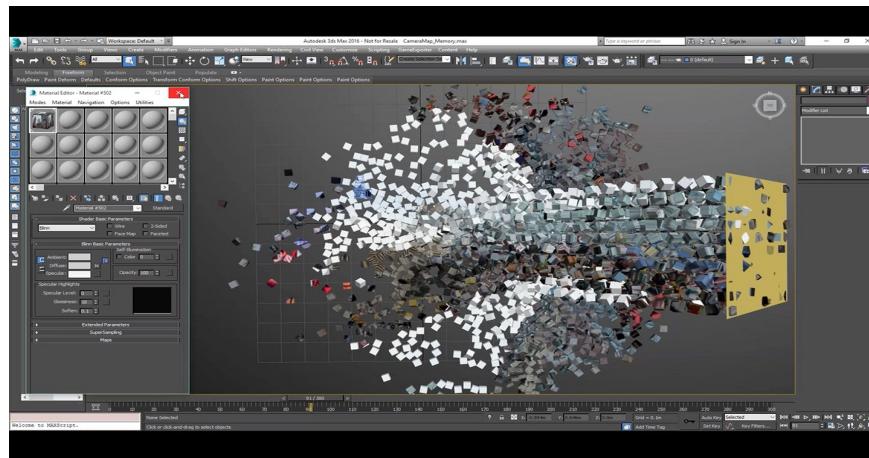


Figura 2.13: thinking particles plugin.

## ■ RAY FIRE

Es un Plugin que te permite romper y destruir objetos mediante animaciones o simulaciones de comportamientos reales. Puedes similar efectos de destrucción como impactos de bala, terremotos, desgaste, ruptura de elementos como madera, roca, vidrios entre otros. Una herramienta muy útil dentro de los VFX y la animación para cortometrajes. Es un plugin con acceso de pago. [43]



Figura 2.14: ray fire plugin.

## ■ KRAKATOA

Krakatoa es el kit de herramientas de gestión, manipulación y procesamiento de partículas de Thinkbox Software. Fue diseñado para procesar y renderizar millones de partículas para representar fenómenos naturales como polvo, humo, espuma, plasma e incluso objetos sólidos.

Está diseñado para funcionar en CPU y optimizado para procesadores de 64 bits. Se puede usar en los equipos que usen como sistemas operativos Windows o Linux.

Hay algunas características que hicieron que Krakatoa sea especial, que son:

- Krakatoa ofrece modos de renderizado de partículas y vértices utilizando los mismos datos de origen.
- Admite shading de partículas volumétrico y aditivo, texturizado, auto-sombreado de alta calidad y proyección de sombras desde y hacia objetos mate.
- Los canales de dispersión, emisión, absorción y densidad por partículas, así como varios modelos de dispersión de luz, permiten niveles profundos de control sobre la imagen final.
- Los efectos de cámara Motion Blur y Depth of Field son naturalmente compatibles.
- Krakatoa se integra con los sistemas de partículas nativos de 3ds Max y Maya y proporciona capacidades con otras aplicaciones 3D y de simulación.

[44]

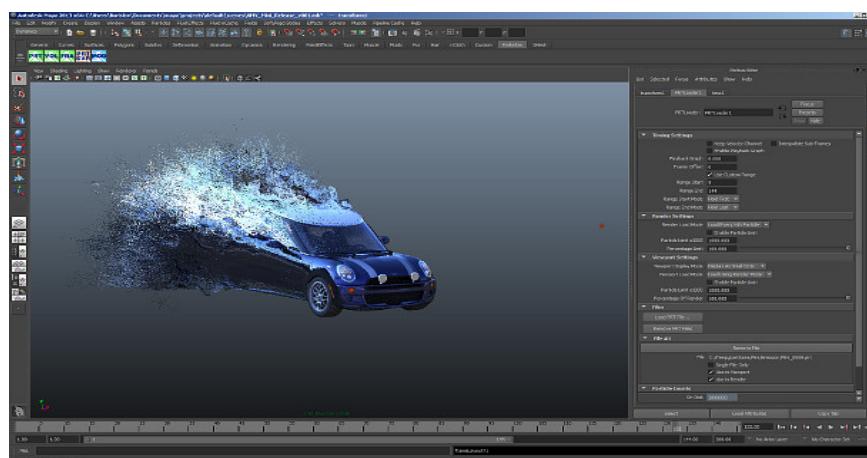


Figura 2.15: krakatoa plugin.

### ■ PHOENIX FD

Gracias a este plugin se puede simular fuego, humo, líquidos, olas, niebla y todo tipo de dinámicas de fluidos. Tiene unos ajustes preestablecidos, configuración rápida y controles intuitivos que permiten crear fluidos basados en física con controles flexibles para renderizar, reprogramar y refinar simulaciones. Además, está integrado a la perfección en 3ds Max y optimizado para renderizar con V-Ray. Este plugin permite crear simulaciones realistas directamente en 3ds Max. Con esta herramienta se puede simular efectos complejos muy realistas con controles sobre los detalles ultrafinos y aumentar la resolución sin cambiar la forma o el comportamiento de la simulación. [44]



Figura 2.16: phoenix fd plugin.

### ■ FOREST PACK

Plugin ideal para la reproducción de vegetación en grandes superficies de terreno que ayuda a optimizar el viewport de trabajo y el procesamiento de elevadas cantidades de polígonos. Ideal para crear bosques, selvas o campos de vegetación extensa. Es un software de pago. [42]



Figura 2.17: forest pack pluggin.

## 2.2.4. Herramientas de Maya (Plugins)

Cabe desatacar que muchos plugins están disponibles tanto para Maya como para 3DS Max ya que ambos pertenecen a la misma desarrolladora y aunque Maya está programado en Python y MEL, que es su lenguaje propio, ambos están programados en C++, por lo que los plugins compatibles con Maya pueden estar disponibles para 3DSMax, ya que son software muy similares que desarrollan prácticamente el mismo tipo de actividades y solamente diferenciándose en el flujo de trabajo. Maya está un poco más orientada a animación y 3DS Max ofrece un mejor interoperabilidad con otros productos de Autodesk, por esto, muchos usuarios prefieren 3ds Max para los materiales y la renderización, dada su facilidad para configurar materiales.

Algunos de los plugins comunes de ambos softwares son Ozone, Phoenix FD, V-Ray, Surface Pinter, Krakatoa entre muchos otros.

### ■ PARTICLE FLOCKER

Este plugin permite simular comportamientos naturales como lo de las bandadas de pájaro, bancos de peces, manadas, etc. haciendo uso del comportamiento de flocking para hacer que las partículas se muevan de manera conjunta, con una velocidad similar haciendo que las partículas se mantengan a una distancia mínima. Se puede hacer que las partículas se dirijan a una ubicación, huyan de una localización o que deambulen aleatoriamente. también tiene la capacidad de definir un volumen de donde las partículas no pueden salir para prever de manera aproximada cual será el espacio que ocupe el flock. [44]

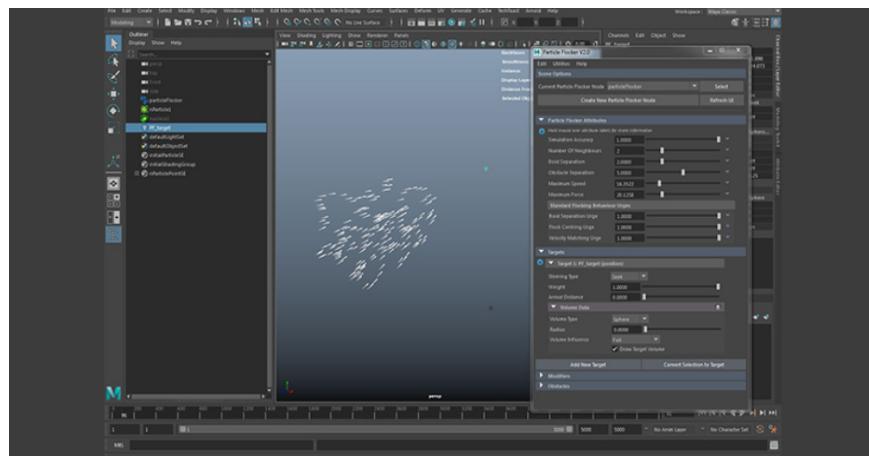


Figura 2.18: particle flocker plugin.

## ■ OZONE

Ozone es un complemento utilizado por muchos estudios profesionales para crear atmósferas realistas. Viene con cuatro modelos de atmósfera como Mapeo Estándar , Volumétrico , Espectral y Ambiental y es totalmente compatible con los renderizadores de producción de clase mundial, Mental Ray y V-ray. Ozone tiene una interfaz muy simple y fácil de usar que contiene controles y configuraciones con respecto al sol, cielos, capas de nubes, arco iris, neblina, etc. Todas estas configuraciones son animables para que el usuario pueda crear fácilmente una atmósfera hermosa y creíble con sol en movimiento, nubes y otros. elementos atmosféricos. [45]

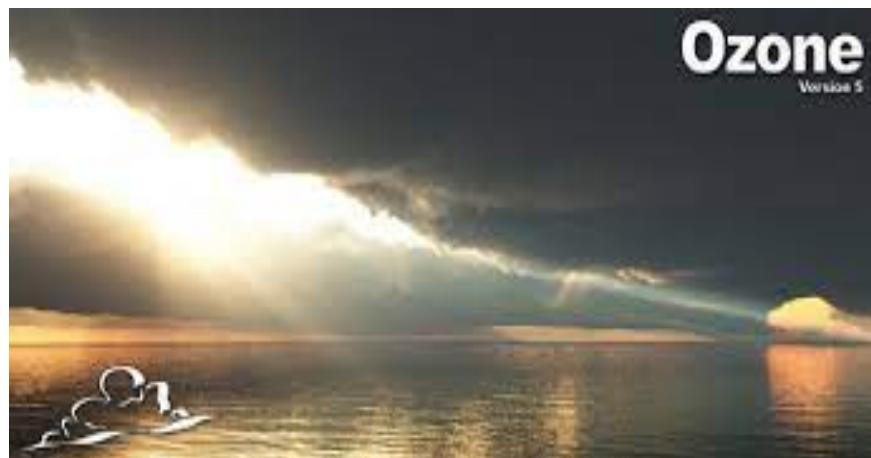


Figura 2.19: ozone plugin.

## ■ REALFLOW

Realflow es una herramienta de simulación de fluidos para VFX que puede ser usada para otras cosas que no sean líquidos como materiales granulares, viscosos, viscoelásticos, rígidos, elásticos, cuerpos blandos y mallas. Recibió en 2008 el Premio al Logro Técnico de la Academia de las Artes y Ciencias Cinematográficas por su contribución al cine y ha sido utilizado en largometrajes como ''300, los ''X-Men'' El Señor de los anillos.entre otros. También en videojuegos para cinemáticas y en comerciales. [46]

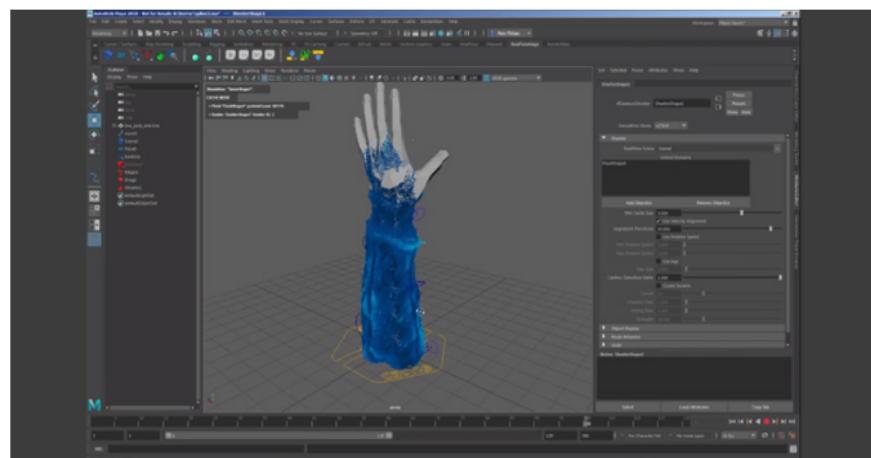


Figura 2.20: realflow plugin.

## ■ ZIVA VFX

Esta herramienta se especializa en simulaciones de músculos avanzada, lo que le permite tener físicas dinámicas de tejidos blandos. Esta herramienta te permite ahorrar tiempo de esculpido de músculos para que parezcan realistas.

Fue creada para crear movimientos realistas para personajes y animales yendo de dentro hacia afuera, creando huesos, músculos y piel. Los animadores aún tienen que animar pero el movimiento de los músculos debajo de la piel es automático. [46]



Figura 2.21: ziva vfx plugin.

## ■ GOLAEM CROWD

Golaem Crowd es un complemento de producción probada para simulaciones de multitudes utilizado en muchas producciones como Hércules, The Walking Dead, Game of Thrones, etc. Ofrece un conjunto de herramientas robusto para crear multitudes creíbles que se pueden utilizar para películas, comerciales y cinematográficas de juegos.

Tiene varios modos de colocación con detección automática de obstáculos. Es compatible con la simulación física para que pueda crear fácilmente multitudes que interactúan con fuerzas y explosiones. [45]

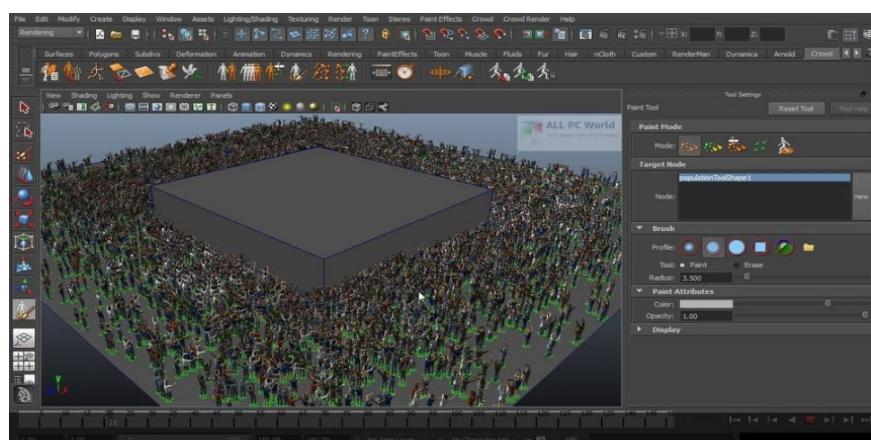


Figura 2.22: golaem plugin.

### ■ BIFROST

Bifrost es un sistema de simulación para efectos visuales de alta calidad. Puede generar líquido a partir de emisores y hacer que caiga bajo la gravedad, interactuar con colisiones para dirigir el flujo, crear salpicaduras y usar campos para crear chorros y otros efectos. Es considerado como uno de los mejores complementos de Maya para VFX en la industria. [45]

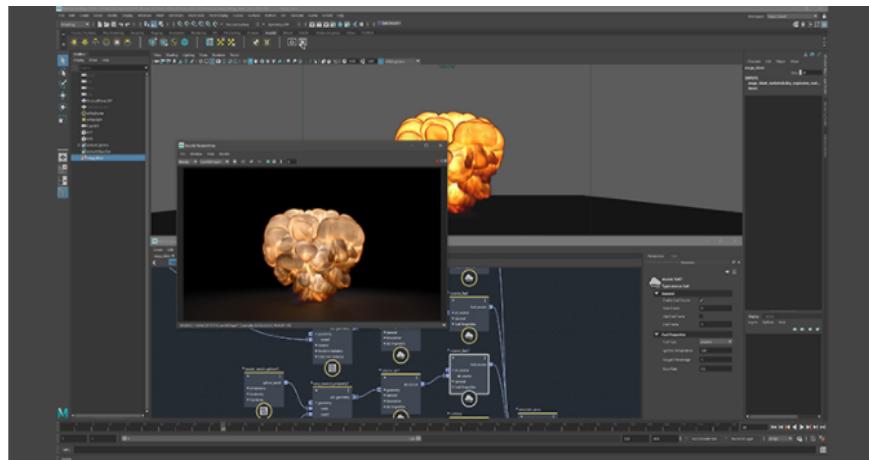


Figura 2.23: bifrost plugin.

### ■ FRACTURE FX

Fracture FX es un complemento para la industria de VFX usado en tomas de destrucción para trabajar en películas y programas de televisión.

Posee una arquitectura escalable y basada en eventos que brinda a los artistas control sobre las simulaciones de destrucción, lo cual es necesario porque cada efecto de demolición es diferente y cada toma necesitará configuraciones personalizadas, lo que da como resultado un mayor control sobre el proceso y mejores tomas con menos horas por artista y por toma. [46]

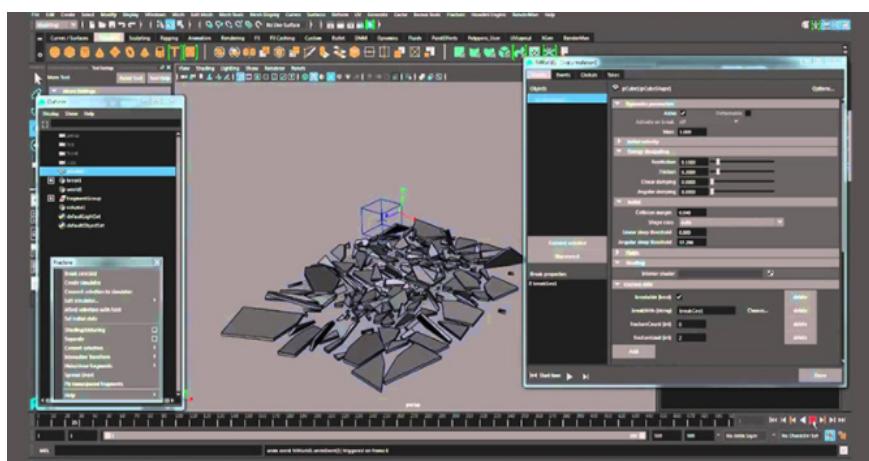


Figura 2.24: fracture fx plugin.

### 2.2.5. Herramientas de Cinema 4D (Plugins)

#### ▪ TURBULENCE FD

Turbulence FD es un plugin para Cinema 4D que sirve para simular efectos de humo, nubes y partículas. La tubería de simulación de TurbulenceFD implementa un solucionador basado en voxels basado en las ecuaciones incompresibles de Navier Stokes. Para configurar una simulación de fluidos, el artista utiliza cualquier tipo de objeto geométrico o sistema de partículas para pintar las fuentes de humo, calor, combustible, etc. en el espacio. El flujo luego transporta estas emisiones de una manera físicamente plausible que crea la apariencia realista de fuego, explosiones, vapor, nubes, polvo y mucho más. Usa un sistema mixto de uso de CPU y GPU donde si la memoria de la GPU se acaba se pasa a usar la CPU. La GPU le da una gran aceleración a la simulación que le permite alcanzar velocidades cercanas a las de tiempo real para bajas resoluciones y escalar sin problemas a altas resoluciones en los cientos de millones de voxels.

Para generar fuego se usa un shader de fuego que simula colores de fuego realistas basados el modelo de radiación de cuerpo negro. Este modelo está controlado por solo dos valores de temperatura. Genera los colores que tendría el fuego real a estas temperaturas. Además para el fuego se usa una técnica de iluminación global llamada dispersión múltiple que no agrega ruido y, por lo tanto, funciona bien con la animación.

Turbulence FD también permite hacer que el flujo del fluido actúe con objetos sólidos. Los objetos de colisión pueden agitar el fluido, agitarlo hacia un lado o actuar como un obstáculo. otra característica de este plugin es que permite agregar ruido de procedimiento al campo de velocidad del fluido es una forma de obtener flujos rizados que se ven más turbulentos e interesantes. [47]



Figura 2.25: turbulence fd plugin.

### ■ FORESTER

Forester es un complemento para Maxon Cinema 4D que proporciona herramientas para la creación de elementos naturales como árboles, plantas, rocas, así como su distribución geológica sobre terreno poligonal. Forester se compone de 4 módulos, a saber, Forester Trees, Multiflora, MultiCloner y Forester Rock. Forester es uno de los mejores y más exitosos complementos de Cinema 4D porque ofrece una solución completamente procedimental para la creación de entornos, todo dentro del espacio de trabajo familiar de Cinema 4D. [48]



Figura 2.26: forester plugin.

### ■ INFINITE SPACE

Infinite Space de C4Depot es un entorno de espacio exterior 3D y un generador de estrellas para animadores de Cinema 4D. Infinite Space utiliza la tecnología de entorno infinito de C4Depot para que nunca te quedes sin tu entorno. Este complemento se integra con Real Earth gratuito de C4Depot para un entorno espacial totalmente realista. Requiere Cinema 4D R13 Broadcast o Studio. Las últimas versiones incluyen assets como entorno espacial Infinite Stars con estrellas 3D, nebulosas volumétrica, nebulosas 2D, galaxias 3D, fondo de estrella 2D personalizable, etc. Es un software de pago.[49]



Figura 2.27: infinite space plugin.

### ■ X-PARTICLES

X-Particles es un sistema avanzado de partículas y efectos visuales con todas las funciones para Cinema 4D de Maxon. Su exclusivo sistema de reglas de preguntas y acciones permite un control completo sobre las simulaciones de partículas. X-Particles es un sistema unificado que incluye varios módulos: ParticleFX para combinar emisores y modificadores y crear sistemas solares, hologramas, visualizaciones médicas y todo tipo de efectos; ExplosiaFX, para simulaciones realistas de humo, fuego y explosivos; un simulador de fluidos a gran y pequeña escala; DynamicsFx ofrece la solución perfecta para doblar y romper la realidad para una producción de efectos visuales; y por último ClothFX que permite la simulación de tejidos. [50]



Figura 2.28: X-Particles plugin.

### ■ LIGHT KIT PRO

Light Kit Pro 3 es un complemento de iluminación de estudio para Cinema 4D, compatible con los renderizadores Estándar, Físico, Arnold, Octane y Redshift. Es el complemento Greyscalegorilla original, reconstruido y reinventado para los artistas 3D de hoy que permite crear iluminación de estudio profesional para productos y objetos con total control.[49]



Figura 2.29: light kit pro plugin.

## 2.3. Conclusión

En conclusión, como ya hemos visto, Blender tiene la capacidad de creación y producción de softwares de pago siendo software libre. Esto es, en parte, gracias a que cuenta con una gran comunidad de usuarios y desarrolladores y mucho material de aprendizaje como los cursos y tutoriales gratuitos, cuyos fondos van destinados al desarrollo del programa, por lo que la atracción de usuarios que supone ser software libre, sumado a los fondos que el consumo del material de aprendizaje de Blender por parte de estos usuarios proporciona a la compañía desarrolladora, hacen la fórmula para que Blender haya crecido tanto. Y es que al haber desarrollado este proceso mejora, se ha conseguido que el desarrollo de la aplicación se vuelva directamente proporcional al tamaño de la comunidad de usuarios de este programa, llegando a eclipsar otros softwares de pago como 3ds Max.

Entrando en comparativas con otros softwares similares se puede decir que Blender a diferencia de Maya, por ejemplo, no tiene nada que envidiar en cuanto prestaciones y no tiene un licencia tan cara como la de Maya, que si bien es cierto que cuenta con una licencia gratuita para estudiantes, una vez finalicen mis estudios me vería obligado a comprar la licencia si quiero seguir trabajando con él. Es un software enfocado a estudios más que a particulares. Z-Bursh es un software especializado en el modelado, pero sin embargo, Blender tampoco se queda corto en este campo y además permite modelar desde scripts, con lo que es ideal para desarrollar este trabajo, ya que no prima la precisión del modelado ni la escultura, sino la capacidad de automatizar el proceso y que este se pueda ejecutar sin necesidad de que el usuario tenga conocimientos de modelado.

Con 3ds Max y Cinema4D pasa algo similar que con Maya, son los programas de pago dedicados al modelado, animación, render, etc. más parecidos a Blender que se encuentran en el mercado como competencia de pago. Y la resolución de porqué he elegido Blender por encima de estos es la misma. Cuentan con unas prestaciones similares siendo software libre facilitándome la tarea de especializarme más en Blender sin tener que pagar licencias y aunque los softwares de pago como Maya, Max y Cinema4D cuentan con un servicio de atención al cliente que hace que las empresas valoren más estos softwares por encima de Blender, la comunidad de Blender sigue siendo suficiente para poder solventar problemas con este programa.

Si bien es cierto que softwares como Houdini o Nuke dan mejores resultados para VFX, los costes de sus licencias son demasiado elevados ya que son softwares enfocados a estudios profesionales de animación y usan un paradigma de composición basado en nodos con el que no estoy suficientemente familiarizado para poder desarrollar los efectos con calidad.

Otros de los principales candidatos para realizar este proyecto eran los softwares de gráficos en tiempo real: Unity y Unreal. Pero dado que algunos de los efectos requieren de modelado vía script y ninguno de estos softwares tiene herramientas para modelar se descartaron como opciones posibles.

Otros softwares como Photoshop y After Effects permiten hacer efectos visuales pero sobre una escena ya editada y renderizada, "a posteriori". En nuestro caso, queremos que los efectos tengan lugar en el momento en que suceden las acciones en la escena para que

estos formen parte de la misma y del relato visual. Por esto, aunque los softwares que edición de imágenes y vídeo también son opciones posibles para realizar vfx, al igual que los paquetes de vfx de las plataformas online, se ha optado por la opción de Blender porque el objetivo de este trabajo no es solo realizar los efectos, sino también llevar a cabo el proceso técnico necesario para que los puedan realizar personas inexpertas y sin los conocimientos necesarios para poder realizarlos, mientras que los softwares mencionados como Photoshop y After Effects requieren de conocimientos más profundos y dominio de dichos programas, además de que el trabajo de realizar los efectos especiales en sí sería un trabajo más artístico que técnico.

A lo largo de esta sección hemos visto muchos softwares de modelado, de gráficos en tiempo real, de creación de texturas, de vfx y todos ellos son softwares sofisticados y populares en la industria de los softwares gráficos y la creación de contenido multimedia, pero dado que la gran ventaja de Blender es su comunidad y su capacidad de auto desarrollo gracias a los usuarios y el feedback que recibe de estos, es especialmente importante para mí hacerlo en este software porque integra todos los procesos que se llevan a cabo en la industria de la animación: modelado, escultura, pintura, simulación, etc. junto con los cursos necesarios para capacitarte en estas áreas de manera gratuita y aportando un ‘‘granito de arena’’<sup>a</sup> la comunidad de Blender y al propio Blender, lo cual no solo beneficia a la compañía desarrolladora y a la comunidad sino a mí mismo, ya que a la vez que crece la reputación de este software crece también la cotización de los expertos en el mismo.

# Capítulo 3

## Requisitos, especificaciones, coste, riesgos, viabilidad

### 3.1. Requisitos

En esta sección se definirán los requisitos, tanto funcionales como no funcionales del proyecto. Los requisitos definirán cuáles son los objetivos en cuanto al funcionamiento del complemento y las características en cuanto al uso por parte de los usuarios.

#### 3.1.1. Requisitos funcionales

Los requisitos funcionales concretan cual debe ser el comportamiento del complemento de forma específica.

1. El usuario ha de poder desplegar y plegar tanto los subpaneles de los efectos como el panel principal.
2. Cada efecto será configurable a través de parámetros que darán un resultado u otro en función de los valores introducidos en dichos parámetros.
3. Los parámetros que no sean aplicables al efecto o no estén activos no permitirán al usuario introducir un valor.
4. El complemento generará cada efecto desde el mismo panel donde se configuran sus parámetros.
5. Cada efecto podrá ser reproducido tantas veces como el usuario desee.
6. Las propiedades con que se generó un efecto, serán modificables desde la interfaz de Blender.
7. Las modificaciones en las propiedades del efecto deben verse reflejados al momento para la comprobación de su resultado.
8. Las modificaciones realizadas a una instancia de un efecto deberán verse reflejadas en esa instancia únicamente.
9. Los objetos generados al crear los efectos se añadirán a na colección propia en Blender.

### 3.1.2. Requisitos no funcionales

Los requisitos funcionales concretan cual debe ser el comportamiento del complemento de forma específica.

1. El complemento debe poderse instalar en Blender para versiones de la 2.9 en adelante.
2. Los efectos deben poder generarse sin un gran consumo de memoria ni de tiempo de ejecución para que puedan ser ejecutados por el mayor número de equipos posible.
3. Los efectos deben ser los más adaptables posibles para que se puedan usar para el mayor número de propósitos posible.
4. Los efectos deben ser versátiles para poder usarlos en todo tipo de escenas.
5. La implementación del complemento será con el lenguaje Python.
6. Los nombres de los parámetros para la implementación del efecto serán claros dando a entender qué son y lo que hacen claramente.
7. El usuario podrá introducir los parámetros de manera sencilla e intuitiva para generar el efecto que desee.
8. El código será público y podrá ser visto por los usuarios que lo utilicen.
9. El add-on debe permitir ser ampliable sin que esto suponga una re-estructuración del código inicial del add-on.
10. Los paneles de cada efecto aparecerán cerrados por defecto cuando se instale el add-on.
11. El add-on será gratuito.

## 3.2. Especificaciones del sistema

### 3.2.1. Especificación hardware

para este proyecto el hardware utilizado ha sido mi equipo personal; un ordenador portátil con las siguientes características:

- Intel Core i5-8250U
- CPU de 1,8 GHz 4 núcleos y 8 procesadores lógicos
- Ratón y resolución de pantalla de 1920 x 1080
- 8 GB de memoria RAM
- tarjeta gráfica NVIDIA GeForce GTX 1050 con 6 GB de RAM.

Aunque las pruebas están hechas con este equipo, en principio, el complemento debería poder funcionar en la mayoría de equipos que soporten los requisitos mínimos para correr Blender de manera fluida, que son:

- 32 bits de doble núcleo CPU 2GHz con soporte SSE2.
- 2 GB de RAM.
- 24 bits 1280 × 768 pantalla.
- Ratón o trackpad.
- Tarjeta gráfica compatible con OpenGL con 256 MB de RAM.

### 3.2.2. Especificación software

A continuación, explicaré de manera breve cuales han sido las tecnologías y herramientas que he utilizado para la propuesta de este trabajo, las cuales han sido Blender y Python únicamente, pues al ser un add-on, el único lenguaje que Blender admite es Python.

- **Blender**

Ya hemos comentado qué es Blender y cuales han sido las causas que han motivado la elección de este software. Blender es la herramienta usada para este trabajo pues es el software al que va dirigido la herramienta que se ha desarrollado, y como no podía ser de otra manera, es con el que se tenía que desarrollar.

- **Python**

El lenguaje de programación Python es el utilizado, pues Blender consta de un módulo de Python y una API que permite a los desarrolladores realizar los ya mencionados addons, para extender las funcionalidades de este software de código abierto. Cabe mencionar que otros softwares como Maya permiten programar sus

herramientas en otros lenguajes como MEL o C++. No obstante, Python es un lenguaje flexible que gracias a ser un lenguaje débilmente tipado, hace posible una programación flexible que facilita el tratamiento de estructuras y tipos de datos propios de Blender y que con otros lenguajes de programación sería más tedioso y complejo.

### 3.3. Planificación temporal

En esta sección se estudiará cuáles son los tiempos estimados para la realización de cada parte del proyecto, evaluada por expertos para hacer un aproximación lo más correcta posible, se mostrará cuál es la desviación y la probabilidad de terminar el proyecto en los tiempos evaluados por los expertos y se mostrará un diagrama de Gantt donde se visualice la duración de las tareas. A continuación, se muestra la tabla de la planificación del proyecto. El diagrama de Gantt del proyecto aparecerá completo en el anexo.

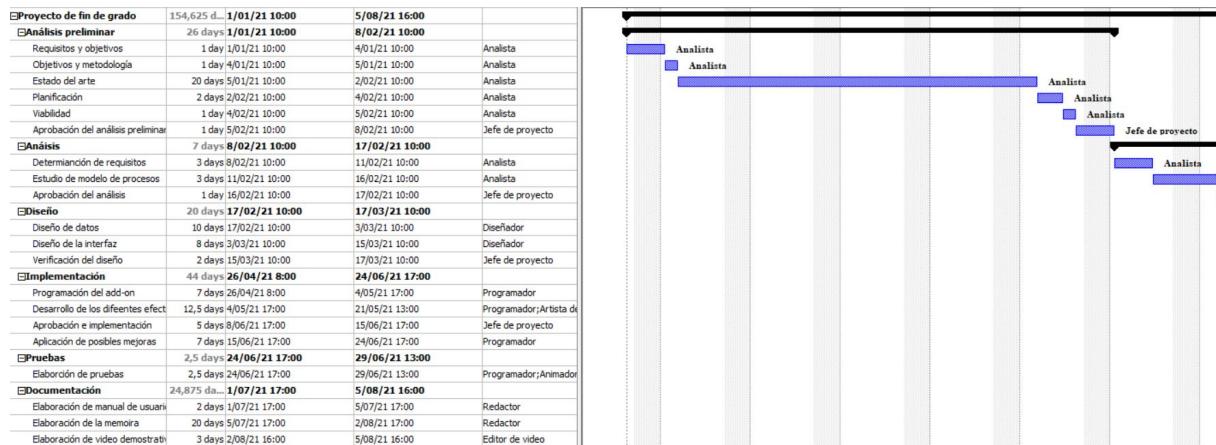


Figura 3.1: Planificación temporal y diagrama de gantt

#### 3.3.1. Desglose de tareas

- Análisis preliminar.** Esta tarea consiste en hacer una recopilación de los requisitos y objetivos a cumplir, así como la planificación y estudio de la viabilidad del proyecto del estado del arte referente a este proyecto.
- Análisis.** En la tarea de análisis se analizarán los requisitos que se exigirán al add-on, se estudiarán los modelos que se llevarán a cabo para que el add-on actúe siguiendo sus funciones y se analizarán los comportamientos que este tendrá durante su funcionamiento.
- Diseño.** Esta tarea se centra en como se va a construir el add-on a partir de diferentes diagramas como los diagramas de casos de uso, secuencia, clases, etc, llevando a cabo un diseño de los datos y de la interfaz.
- Implementación.** La tarea que consiste en la programación del add-on y el desarrollo de los efectos que lo componen, así como sus posibles mejoras y extensiones.

5. **Pruebas y resultados.** Esta tarea trata sobre como ha funcionado el add-on y sobre sus resultados, realizando diferentes pruebas que muestran cuáles son sus capacidades testeadas por usuarios.
  
6. **Documentación.** La documentación es la tarea que recoge las actividades de elaboración de la memoria, del manual de usuario y de video demostrativo, que documenta el proceso de elaboración del add-on y su funcionamiento.

### 3.3.2. Estimaciones temporales

Para el cálculo temporal de cada tarea se realizará un cálculo estimado basado en la siguiente fórmula:

$$t_e = \frac{t_o + 4t_m + t_p}{6} \quad (3.1)$$

Donde  $t_e$  es el tiempo estimado,  $t_o$  el tiempo optimista,  $t_p$  es el tiempo pesimista y  $t_m$  es el tiempo probable.

Una vez determinados los tiempos estimados del total del proyecto por los expertos, se hará un cálculo ponderado del mismo en base a unos pesos que tendrá cada estimación según la fórmula:

$$t_e = exp_1 * 0,5 + exp_2 * 0,35 + exp_3 * 0,15 \quad (3.2)$$

Donde  $exp_1$  es el tiempo estimado calculado con la fórmula 3.1 del experto 1,  $exp_2$  el tiempo estimado del experto 2 y  $exp_3$  el tiempo estimado del experto 3.

Los expertos y sus pesos en el cálculo final del tiempo estimado son los siguientes<sup>1</sup>

- Dr. Rafael Javier Martínez Dura: Tutor de este TFG y profesor titular en la Universidad de Valencia, siendo experto en gráficos en tiempo real, simulación y modelado su peso en la estimación total será del 50
  
- Dr. Ángel Rodríguez Cerro: Profesor asociado en la Universidad de Valencia, siendo experto en simulación y gráficos por computador, tendrá un peso del 35
  
- Darío Rodríguez Hernández: Estudiante titulado en el grado de ingeniería multimedia por la Universidad de Valencia y autor de otro add-on para Blender como trabajo de fin de grado, tendrá un peso total en la estimación del 15

---

<sup>1</sup>Todos los tiempos serán medidos en días

Tarea	Descripción	A	B	M	
		T. Optimista	T. Pesimista	T. Probable	T. Estimado
<b>1.</b>	<b>Análisis preliminar</b>				
1.1	Requisitos y objetivos	1	3	2	2,50 ▾
1.2	Objetivos y metodología	1	2	1	1,67
1.3	Estado del arte	18	25	20	23,00
1.4	Planificación	1	5	2	3,83
1.5	Viabilidad	1	3	1	2,33
1.6	Aprobación análisis preliminar	1	5	1	3,67
<b>2.</b>	<b>Análisis</b>				
2.1	Determinación de requisitos	1	2	3	2,00
2.2	Estudio del modelo de procesos	1	3	1	2,33
2.3	Aprobación del análisis	1	5	1	3,67
<b>3.</b>	<b>Diseño</b>				
3.1	Diseño de datos	5	15	10	12,50
3.2	Diseño de la interfaz	5	20	15	16,67
3.3	Verificación del diseño	1	5	2	3,83
<b>4.</b>	<b>Implementación</b>				
4.1	Programación del add-on	5	15	10	12,50
4.2	Desarrollo de los diferentes efectos	10	25	20	21,67
4.3	Aprobación e implementación	3	7	5	6,00
4.4	Aplicación de posibles mejoras	1	10	5	7,67
<b>5.</b>	<b>Pruebas</b>				
5.1	Elaboración de pruebas	1	4	3	3,33
<b>6.</b>	<b>Documentación</b>				
6.1	Elaboración del manual de usuario	3	6	5	5,33
6.2	Elaboración de la memoria	8	20	10	16,33
6.3	Elaboración del vídeo demostrativo	1	4	3	3,33
<b>TOTAL</b>		<b>69</b>	<b>184</b>	<b>120</b>	<b>154,17</b>

Figura 3.2: Estimación temporal del primer experto, Rafael Javier Martínez Dura.

Tarea	Descripción	A	B	M	
		T. Optimista	T. Pesimista	T. Probable	T. Estimado
1	<b>Análisis preliminar</b>				
1.1	Requisitos y objetivos	1	2	1	1,67 ▾
1.2	Objetivos y metodología	1	2	1	1,67
1.3	Estado del arte	10	20	20	18,33
1.4	Planificación	1	2	2	1,83
1.5	Viabilidad	1	3	1	2,33
1.6	Aprobación análisis preliminar	1	2	1	1,67
2.	<b>Análisis</b>				
2.1	Determinación de requisitos	1	5	3	4,00
2.2	Estudio del modelo de procesos	2	5	3	4,17
2.3	Aprobación del análisis	1	5	1	3,67
3.	<b>Diseño</b>				
3.1	Diseño de datos	5	15	10	12,50
3.2	Diseño de la interfaz	2	5	4	4,33
3.3	Verificación del diseño	1	5	2	3,83
4.	<b>Implementación</b>				
4.1	Programación del add-on	5	15	7	12,00
4.2	Desarrollo de los diferentes efectos	10	30	25	25,83
4.3	Aprobación e implementación	3	7	5	6,00
4.3	Aprobación e implementación	3	7	5	6,00
4.4	Aplicación de posibles mejoras	1	10	7	8,00
5.	<b>Pruebas</b>				
5.1	Elaboración de pruebas	3	7	5	6,00
6.	<b>Documentación</b>				
6.1	Elaboración del manual de usuario	1	3	2	2,50
6.2	Elaboración de la memoria	15	35	20	29,17
6.3	Elaboración del vídeo demostrativo	1	5	3	4,00
<b>TOTAL</b>		<b>66</b>	<b>183</b>	<b>123</b>	<b>153,50</b>

Figura 3.3: Estimación temporal del segundo experto, Ángel Rodríguez Cerro.

Tarea	Descripción	A	B	M	
		T. Optimista	T. Pesimista	T. Probable	T. Estimado
1	<b>Análisis preliminar</b>				
1.1	Requisitos y objetivos	1	2	2	1,83 ▾
1.2	Objetivos y metodología	1	2	1	1,67
1.3	Estado del arte	20	30	25	27,50
1.4	Planificación	1	5	3	4,00
1.5	Viabilidad	1	5	3	4,00
1.6	Aprobación análisis preliminar	1	3	1	2,33
2.	<b>Análisis</b>				
2.1	Determinación de requisitos	5	7	3	6,00
2.2	Estudio del modelo de procesos	3	7	5	6,00
2.3	Aprobación del análisis	1	5	1	3,67
3.	<b>Diseño</b>				
3.1	Diseño de datos	7	15	10	12,83
3.2	Diseño de la interfaz	3	7	4	5,83
3.3	Verificación del diseño	2	5	3	4,17
4.	<b>Implementación</b>				
4.1	Programación del add-on	7	15	8	12,50
4.2	Desarrollo de los diferentes efectos	25	50	35	43,33
4.3	Aprobación e implementación	3	7	5	6,00
4.4	Aplicación de posibles mejoras	2	15	7	11,50
5.	<b>Pruebas</b>				
5.1	Elaboración de pruebas	5	15	10	12,50
6.	<b>Documentación</b>				
6.1	Elaboración del manual de usuario	1	3	2	2,50
6.2	Elaboración de la memoria	10	30	25	25,83
6.3	Elaboración del vídeo demostrativo	1	2	2	1,83
<b>TOTAL</b>		100	230	155	195,83

Figura 3.4: Estimación temporal del tercer experto, Darío Rodríguez Hernández.

Una vez calculados los tiempos estimados por los expertos hay que aplicar los pesos para calcular las ponderaciones. El tiempo estimado ponderado queda tal que:

	Te	Peso	Te * Peso
Angel	153,5	0,35	53,73
Rafa	154,7	0,5	77,35
Darío	154,17	0,15	23,13
Total	462,37	1	154,20

Figura 3.5: Tiempos estimados ponderados.

Además de los tiempos ponderados también se ha decidido calcular otros datos de interés como son la desviación típica  $\sigma^2$  y la varianza  $\sigma$  para determinar la desviación de los valores respecto del promedio.

$$\sigma^2 = \left( \frac{t_p - t_o}{6} \right)^2 \quad (3.3)$$

$$\sigma = \sqrt{\sigma^2} \quad (3.4)$$

	$\sigma$	$\sigma^2$
Angel	19,50	380,25
Rafa	19,17	367,36
Darío	21,67	469,44
Total	39,17	1534,03

Figura 3.6: Varianzas y desviaciones típicas respecto a los valores promedios.

Gracias a estos valores podemos hacer un cálculo de cual podría ser el tiempo de realización del proyecto en base a las estimaciones de los expertos con determinados niveles de confianza. Calcularemos el tiempo que se tardaría en realizar el proyecto según la estimación de los expertos con un 90 % de confianza y con un 99 % de confianza usando la distribución normal estándar para cálculo de probabilidades de variables aleatorias continuas. La fórmula para el cálculo de probabilidad es la siguiente.

$$Z = \frac{X - \mu}{\sigma} \quad (3.5)$$

Con un 90 % de confianza, mirando en la tabla, el valor de Z es 1,2815:

$$1,2815 = \frac{X - 160,45}{39,17}; X = 210,65 \quad (3.6)$$

Con un 90 % de confianza, mirando en la tabla, el valor de Z es 2,3263:

$$2,3263 = \frac{X - 160,45}{39,17}; X = 251,57 \quad (3.7)$$

Según los cálculos, con un 90 % de confianza, el proyecto debería terminarse en 210,65 días y con un porcentaje del 99 % el proyecto debería terminarse en 251,57 días como máximo.

## 3.4. Estimación de costes

En este apartado se calcularán los costes económicos del proyecto teniendo en cuenta costes directos y indirectos amortizaciones.

### 3.4.1. Costes directos

#### 3.4.1.1. Costes de hardware

En este apartado se calculará el precio del equipo necesario para desarrollar este proyecto, así como su amortización. El material necesario es un ordenador portátil con las características definidas en el apartado de especificación hardware y un ratón, ya que sin él no se puede navegar con libertad por el espacio 3D de Blender.

El coste del ordenador es de 1500 euros, que, según la Agencia Tributaria, tiene una amortización máxima de 8 años y el del ratón 15 euros con tiempo máximo de amortización de 10 años, según los datos de la Agencia Tributaria. [51] Ambos componentes del equipo se ha utilizado el 100 % del proyecto, el coste amortizado se calcula tal que:

$$\text{Coste amortizado} = \text{Días de uso} * \% \text{ de uso} * \text{Precio de amortización por día}$$

Donde el precio amortizado por día se calcula:

$$\text{Precio de amortización por día} = \text{precio} / \text{vida útil}$$

El tiempo de vida útil de un equipo informático se estima es aproximadamente en 5 años y el de un ratón en 6 años, de manera que el cálculo de precio de amortización queda de la siguiente manera:

Material	Vida útil (días)	Precio de amortización/día
Ordenador	1825	0,82
Ratón	2190	0,01

Por tanto las amortizaciones del equipo queda así:

Material	Días de uso	%Uso	Precio de amortización/día	AMORTIZACIÓN
Ordenador	90	100%	0,82	73,8
Ratón	90	100%	0,01	0,9

El ordenador tiene una amortización de 73,8€ por día y el ratón de 0,9€ por día. Lo que supone un total de 74,7€ amortizados por día.

Coste actual	0 €
Coste añadido	1.515 €
Coste total	1.515 €

### 3.4.1.2. Costes de software

Son los costes asociados a los programas necesarios para el desarrollo del proyecto. En este caso, todos los recursos software utilizados han sido gratuitos, ya que Blender es un software gratuito, y es el único programa de desarrollo necesario para este proyecto, el programa con el que se ha gestionado el proyecto es el ProjectLibre, el cual es una alternativa gratuita al Project Profesional de Microsoft y para el uso de hojas de cálculo y presentaciones se ha utilizado el paquete OpenOffice, el cual es de código abierto y gratuito también por lo tanto los costes en software son 0€.

Coste actual	1.515 €
Coste añadido	0 €
Coste total	1.515 €

### 3.4.1.3. Costes de personal

En cuanto al personal, se ha necesitado de los siguientes roles para el desarrollo del proyecto, cuyos salarios han sido extraídos de los datos del Boletín Oficial del Estado en su resolución de 10 de mayo de 2021, de la Dirección General de Trabajo, por la que se registra y publica el Acta del acuerdo relativo a la revisión salarial y actualización de dietas, del II Convenio colectivo de la Industria de la Producción Audiovisual (Técnicos).

1. **Analista:** Es el encargado de planificar la arquitectura del proyecto, los presupuestos y extraer funciones y requisitos del sistema. Siguiendo los datos de la Dirección General de Empleo, el sueldo anual medio de un analista de sistemas es de 23.505,72 euros anuales.
2. **Diseñador:** Es el responsable de encargado de decidir cómo se distribuirán los paneles, qué elementos son los apropiados para que sea lo más intuitivo posible, la colocación de los mismos y en definitiva todos los elementos visuales del add-on. Siguiendo los datos de la Dirección General de Empleo, el sueldo anual medio de un diseñador es de 22.993,74 euros anuales.
3. **Programador:** El encargado de producir el código de la herramienta y generar los efectos mediante scripts. Siguiendo los datos de la Dirección General de Empleo, el sueldo anual medio de un diseñador es de 14.349,67 euros anuales.
4. **Artista de VFX:** Persona con conocimientos en Blender y VFX para generar los efectos originalmente de manera manual ideándolos para poder programar la herramienta que los genere automáticamente. Siguiendo los datos de la Dirección General de Empleo, el sueldo anual medio de un artista es de 19.535,64 euros.
5. **Animador 3D:** Persona con conocimientos en animación para hacer las pruebas necesarias y animaciones de comprobación de calidad de los efectos. Siguiendo los datos de la Dirección General de Empleo, el sueldo anual medio de un artista es de 20.514,15 euros.
6. **Editor de vídeo:** Persona con conocimientos en edición de vídeo y softwares de edición para la realización del montaje final del vídeo demo. Siguiendo los datos de la Dirección General de Empleo, el sueldo anual medio de un artista es de 15.500,00 euros.

- 7. Redactor:** Persona encargada de generar la documentación del proyecto como son la memoria y el manual. Siguiendo los datos de la Dirección General de Empleo, el sueldo anual medio de un artista es de 19.535,64 euros.
- 8. Jefe de proyecto:** Persona encargada de planificar, ejecutar y monitorizar las acciones del proyecto. Es la persona por quien pasan todas las decisiones que van orientadas a la consecución de los objetivos de cada fase y del proyecto en conjunto. Siguiendo los datos de la Dirección General de Empleo, el sueldo anual medio de un jefe superior es de 37.418,88 euros anuales.

Analista	248 horas
Determinación de requisito	24 horas
Estudio de modelo de proce	24 horas
Requisitos y objetivos	8 horas
Objetivos y metodología	8 horas
Estado del arte	160 horas
Planificación	16 horas
Vialidadao	8 horas
Programador	332 horas
Programación del add-on	56 horas
Desarrollo de los difeentes	200 horas
Aplicación de posibles mejo	56 horas
Elaboración de pruebas	20 horas
Diseñador	144 horas
Diseño de la interfaz	64 horas
Diseño de datos	80 horas
Artista de VFX	0 horas
Jefe de proyecto	72 horas
Aprobación del análisis	8 horas
Verificación del diseño	16 horas
Aprobación e implementaci	40 horas
Aprobación del análisis pre	8 horas
Animador 3D	20 horas
Elaboración de pruebas	20 horas
Editor de video	24 horas
Elaboración de video demo	24 horas
Redactor	176 horas
Elaboración de manual de u	16 horas
Elaboración de la memoria	160 horas

Figura 3.7: Tabla de horas por trabajador

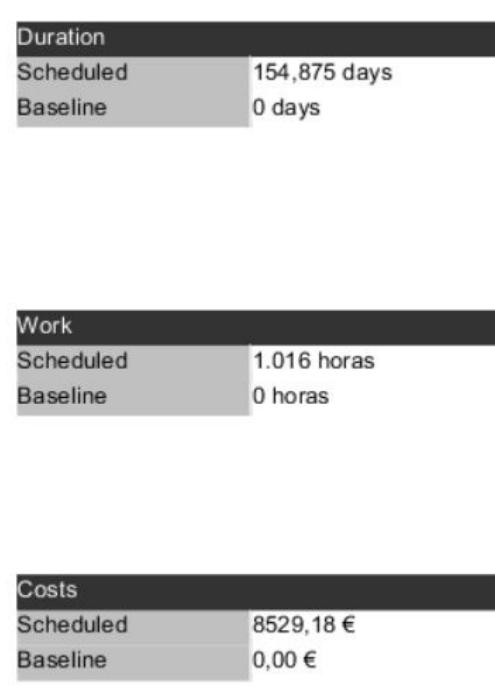


Figura 3.8: Informe del proyecto

Los costes de los trabajadores son los siguientes: el analista tiene un costo de 2790,00€, el programador de 1812,50€, el diseñador de 1133,94€, artista de VFX de 669,03€, el jefe de proyecto de 922,66€, el animador 140,51€, el editor de vídeo 123,29€ y el redactor 937,26€.

En el informe de proyecto se define un coste total del proyecto de 8.529,18€, 1.016 horas de trabajo y una duración de 154,875 días. El costo del proyecto está calculado en bruto, por lo que a este costo habría que sumarle la cantidad de la Seguridad Social, que es del 30 % respecto al total, de manera que el costo del proyecto sería:  $8.529,18€ * 1,3 = 11.087,93€$ .

Los costes totales del proyecto quedan hasta ahora quedan así:

Coste actual	1.515 €
Coste añadido	11.088 €
Coste total	12.603 €

Figura 3.9: Costes personales del proyecto

Finalmente los costes directos los podemos representar como:

	Costes hardware	Costes software	Costes personales	Total
Costes directos	1.515 €	0 €	12.603 €	14.118 €

Figura 3.10: Costes directos del proyecto

### 3.4.2. Costes indirectos

En cuanto a los costes indirectos, se necesitaría al menos de un lugar de trabajo, luz, agua e internet. La tabla de gastos quedaría de la siguiente manera:

Concepto	Coste
Alquiler oficina	150 €
Agua	40 €
Luz	50 €
Internet	50 €
	290 €

Figura 3.11: Costes indirectos del proyecto

Por tanto, los costes totales finales del proyecto quedan tal que:

Costes directos	14.118 €
Costes indirectos	290 €
Costes totales	14.408 €

Figura 3.12: Costes totales del proyecto

En resumen y para concluir, el proyecto ha tenido una duración de 83,875 días, 655 horas y un coste total de 11.052€. La duración del proyecto ha sido más larga de lo esperado, debido a la poca experiencia en el uso de Blender y de su API, lo que ha interrumpido el proceso de trabajo en busca de soluciones para una gran parte de problemas que surgían y por la gran cantidad de estos que han ido surgiendo. También ha afectado la voluntad de perfeccionar cada vez más los efectos en busca de mejores resultados, mejor interactividad con el usuario y tratar de facilitar la mayor parte del trabajo a este, por lo que el trabajo a realizar y los problemas asociados a estos iba siendo mayor.

## 3.5. Viabilidad

El análisis de viabilidad va dirigido a determinar si el proyecto es posible de realizar en términos económicos, de rentabilidad y legales entre otros. Si un proyecto no resulta viable no es recomendable comenzarlo, por lo que esta parte es de gran importancia. En esta sección trataremos la viabilidad económica, la viabilidad legal y la viabilidad técnica.

### 3.5.1. Viabilidad económica

En cuanto a la viabilidad económica, tras haber estudiado los costes, no habría excesivo problema en cuanto a costes de hardware, ya que es el mínimo equipo necesario para la realización del proyecto. En cuanto a software no habría ninguno ya que todo el software que se ha utilizado es todo software libre con 0€ de costo. El mayor problema vendría de el coste de personal, el cual supone la mayor parte del costo del proyecto. Además, si la duración del proyecto se alargase más de lo esperado este coste sería todavía mayor, por lo que sería totalmente inviable. No obstante, para este proyecto, en la práctica, los costes de personal también han sido nulos ya que he sido yo el que ha llevado a cabo todos los roles necesarios para la realización del este proyecto.

En cuanto a los costes indirectos, excepto el coste de alquiler, el cuál se puede obviar también dado que el local donde se realiza el trabajo es el propio domicilio o sitio de trabajo habitual, son costes que ya existían, pues la luz, el agua y el internet son costes cotidianos. Por tanto los costes indirectos no suponen un impedimento para el proyecto.

Teniendo en cuenta que tanto los costes directos como los indirectos no suponen un problema, podemos concluir que el proyecto es viable económicamente.

### 3.5.2. Viabilidad legal

Como todos los proyectos, cualquier proyecto informático debe cumplir con la legislación pertinente a la protección de la propiedad intelectual. Este proyecto está basado en el uso de efectos ideados por artistas expertos en Blender que han sido divulgados a través de Internet, por lo tanto, no soy autor del procedimiento seguido para llevar a cabo ninguno de estos efectos. Según aparece en el artículo 270 del Código Penal, en lo contendiente a los delitos contra la propiedad intelectual[52]:

*“Se castiga a quien, con ánimo de lucro y en perjuicio de tercero, reproduzca, plagie, distribuya o comunique públicamente, en todo o en parte, una obra o presentación literaria, artística, científica, o su transformación, interpretación o ejecución artística fijada en cualquier tipo de soporte o comunicada a través de cualquier medio, sin la autorización de los titulares de los correspondientes derechos de propiedad intelectual o de sus cesionarios”.*

No obstante, esta herramienta no tiene ánimo de lucro, pues es una herramienta pública, que no supone ningún perjuicio a sus autores, puesta a disposición de la comunidad de Blender, al igual que lo es el procedimiento para generar los efectos que los mismos

autores han compartido. A todo esto cabría añadir que este proyecto es de fin de grado, no es de investigación, por lo que no ha de ser un trabajo original propio.

Habiendo hecho un repaso por la legislación relacionada con el proyecto, el proyecto es viable legalmente.

### 3.5.3. Viabilidad técnica

En este apartado se analizará la viabilidad del proyecto en cuanto a capacidad técnica y habilidades para poder llevarlo a cabo.

Este proyecto está totalmente desarrollado en Blender, usando el lenguaje Python. Ambos, el lenguaje y la aplicación, son conocidos por mí a un nivel intermedio-básico. Sin embargo, en cuanto al lenguaje, se podría decir que Python es menos complejo que otros como Java o C/C++, con los cuales también tengo experiencia, en concreto C++, el cuál he utilizado y profundizado en él durante mis estudios en varias asignaturas. Por tanto, Python, debido a sus similitudes con otros lenguajes con los que tenía más experiencia, junto con la experiencia que ya poseía y añadido a la gran documentación que hay y contenido explicativo, debido a su popularidad, dan como resultado que el manejo de este lenguaje no supondrá un problema para el desarrollo del proyecto.

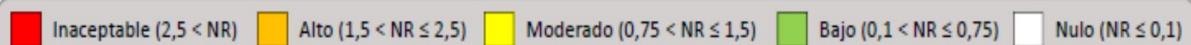
En cuanto a Blender, es un programa muy versátil, con muchas funcionalidades y con una curva de aprendizaje relativamente horizontal, es decir, que no es tan intuitivo al principio como otros programas de modelado y 3D, como se ya analizó en el estado del arte. Esto ha hecho que, aunque ya poseía algo de experiencia en Blender, no sea suficiente como para especializarme en VFX, modelado, shading, etc. Habilidades necesarias para desarrollar este proyecto. Aún con estos inconvenientes, y como también se dijo en la introducción y el estado del arte, la comunidad de Blender es la mayor fortaleza que posee este software, la cual proporciona una gran cantidad de soluciones a problemas de todo tipo y recursos sobre como desarrollar todo tipo de efectos en Blender que han sido muy útiles para poder desarrollar este proyecto. Por ello, gracias a la comunidad de Blender, la API de Blender para scripting en Python y la ayuda de la propia aplicación, este proyecto ha sido perfectamente viable.

## 3.6. Riesgos

En esta sección se llevará a cabo un análisis de los riesgos que pueden suceder durante el desarrollo del proyecto. El nivel de riesgo se calcula a partir de la probabilidad y el impacto<sup>2</sup>.

### 3.6.1. Análisis de riesgos

Riesgo	Probabilidad	Impacto	Nivel de Riesgo	Calificación
Aparece un add-on similar que ofrece mejores resultados.	30%	10	3	Alto
Que los efectos sean de baja calidad.	30%	2	0,6	Bajo
Que el add-on deje de funcionar en versiones posteriores de Blender.	75%	2	1,5	Medio
Tareas más complejas de lo previsto.	90%	2	1,8	Medio
Mala planificación inicial y re-estructuración del add-on.	60%	1	0,6	Bajo
Que los paneles y nombres de parámetros no resulten intuitivos.	20%	1	0,2	Bajo
Incompatibilidad entre efectos que se aplican a la vez.	60%	2	1,2	Bajo
Ritmo de trabajo inferior al previsto	40%	2	0,8	Bajo



<sup>2</sup>El impacto se mide en semanas de retraso del proyecto

# Capítulo 4

## Desarrollo del proyecto

En este apartado se tratará cómo se ha desarrollado el complemento y como se extraído las funcionalidades y características en función de los requisitos.

Este capítulo se divide en:

- **Análisis:** En esta sección se hará un análisis de cuales son las funcionalidades que deberá implementar el complemento y cual será la composición y arquitectura adecuadas para poder realizaras de la manera óptima. En esta sección se describirán los casos de uso, así como los diagramas de actividad.
- **Diseño:** Una vez terminada la parte de análisis se llevará a cabo el diseño del complemento donde se modelará el comportamiento del add-on a través de diagramas de estados y de secuencia.
- **Implementación:** Por último describiremos como se han traducido los modelos anteriores a código y mostraré capturas de las partes más relevantes del código de los diferentes efecto y del add-on.

### 4.1. Análisis

En esta sección se mostrarán los diagramas de casos de uso, sus especificaciones y los diagramas de actividad de los efectos, así como el diagrama de actividad que representa el comportamiento del add-on en general, donde se podrá comprender cuáles van a ser los detalles de las funciones que realiza el add-on, cómo se relacionan el usuario con el complemento y cómo se comporta.

Cabe mencionar que, al ser un add-on que no realiza una única función, sino que cada efecto tiene una implementación y unos usos diferentes los diagramas de casos de uso se harán independientes tratándolos como add-ons diferentes aunque forman parte del mismo.

### 4.1.1. Análisis de casos de uso

#### 4.1.1.1. Diagrama de casos de uso del efecto de rayo

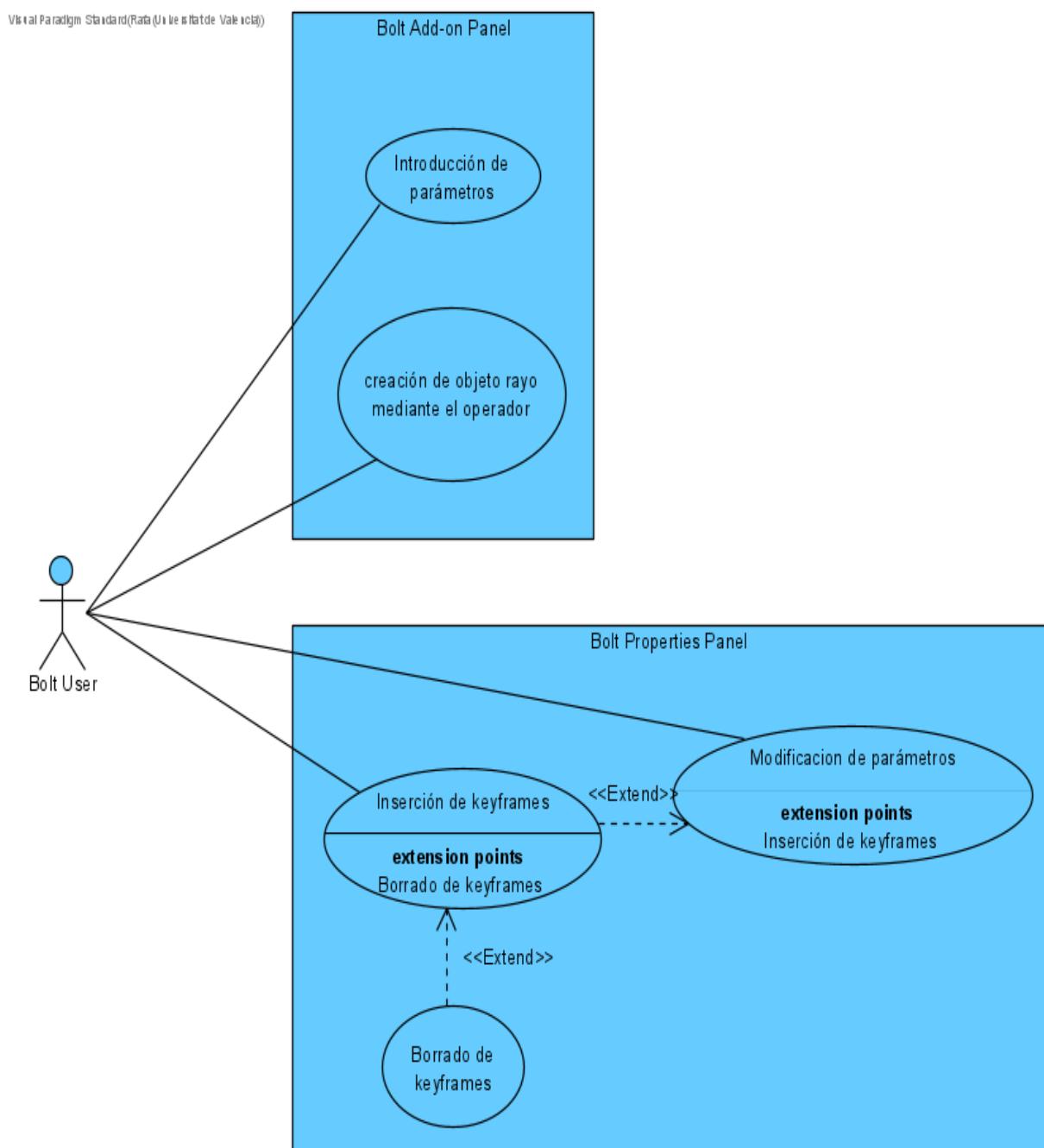


Figura 4.1: Diagrama de casos de uso del efecto del rayo

#### 4.1.1.2. Especificación de casos de uso del efecto de rayo

Identificador	R-CU-01
Nombre	Introducción de parámetros
Actor	Usuario que quiera crear un rayo con el add-on.
Descripción	En este caso de uso se introducen los valores iniciales de los parámetros en los campos correspondientes para crear el rayo deseado.
Precondición	Haber instalado correctamente el add-on.
Postcondición	Todos los valores contienen un valor dentro de los límites permitidos por los campos.
Caso básico	Se instala el add-on y, tras desplegar el panel del rayo, se introducen los valores.

Identificador	R-CU-02
Nombre	Creación de objeto rayo mediante el operador.
Actor	Usuario que quiera crear un rayo con el add-on.
Descripción	En este caso de uso se genera un objeto rayo con los valores de los parámetros introducidos mediante el operador del panel del rayo.
Precondición	Todos los parámetros han de tener algún valor introducido.
Postcondición	Generación de un objeto rayo.
Caso básico	Tras introducir los valores de los parámetros se clica en el operador del panel del rayo para generararlo.

Identificador	R-CU-03
Nombre	Modificación de parámetros del rayo.
Actor	Usuario que quiera modificar algunos de los parámetros iniciales con los que se creó el rayo.
Descripción	En este caso de uso se modifican los valores de algunos de los parámetros con los que se creó el rayo para modificar el rayo de acuerdo a estos.
Precondición	Ha de haber un objeto rayo creado en la escena.
Postcondición	Los valores introducidos cambian el aspecto y/o la forma del rayo inicialmente creado.
Caso básico	Tras crear un rayo, se clica sobre este y se abre el menú de propiedades del objeto, donde se abrirá la pestaña de propiedades del rayo y se modificarán los parámetros.

Identificador	R-CU-04
Nombre	Inserción de keyframes en propiedades del rayo.
Actor	Usuario que quiera insertar keyframes en las propiedades del rayo.
Descripción	En este caso de uso se clica sobre el operador de insertar keyframes del panel de propiedades del rayo, dentro del panel propiedades del objeto, en la interfaz de Blender para para animar las propiedades que se modifiquen insertando keyframes en el frame que indique el marcador de la línea de tiempo.
Precondición	Ha de haber un objeto rayo creado en la escena.
Postcondición	Las propiedades que tengan los valores iniciales con los que el rayo fue creado y aquellas que hayan sido modificadas tendrán keyframes en los frames indicados por el marcador de la línea de tiempo cuando se clicó el operador .
Escenario básico	Tras crear un rayo, se clica sobre este y se abre el menú de propiedades del objeto, donde se abrirá la pestaña de propiedades del rayo y se clicará en el operador de inserción de keyframes para insertar los keyframes en el frame actual de la línea de tiempo.
Escenario alternativo	Se trata de insertar un keyframe en un propiedad en un frame que donde ya posee un keyframe. El último keyframe sobrescribe al primero.

Identificador	R-CU-05
Nombre	Borrado de keyframes en propiedades del rayo.
Actor	Usuario que quiera borrar los keyframes en las propiedades del rayo.
Descripción	En este caso de uso se clica sobre el operador de borrar keyframes del panel de propiedades del rayo, dentro del panel propiedades del objeto, en la interfaz de Blender para para eliminar los keyframes de las propiedades hubieran sido animadas en el frame que indique el marcador de la línea de tiempo.
Precondición	Ha de haber un rayo con una propiedad animada, al manos.
Postcondición	Las propiedades que tuvieran keyframes en el frame marcado por el marcador de la línea de tiempo cuando se clicó el operador dejarán de tenerlo.
Escenario básico	Tras haber insertado un keyframe en alguna propiedad del rayo se clicará en el operador de borrado de keyframes para eliminar los keyframes en el frame actual de la línea de tiempo de todas aquellas propiedades del rayo que tuvieran alguno.
Escenario alternativo	Se trata de borrar los keyframes de las propiedades en un frame donde no hay ninguno. El operador no tendrá efecto.

#### 4.1.1.3. Diagrama de casos de uso del efecto de transición de wireframe

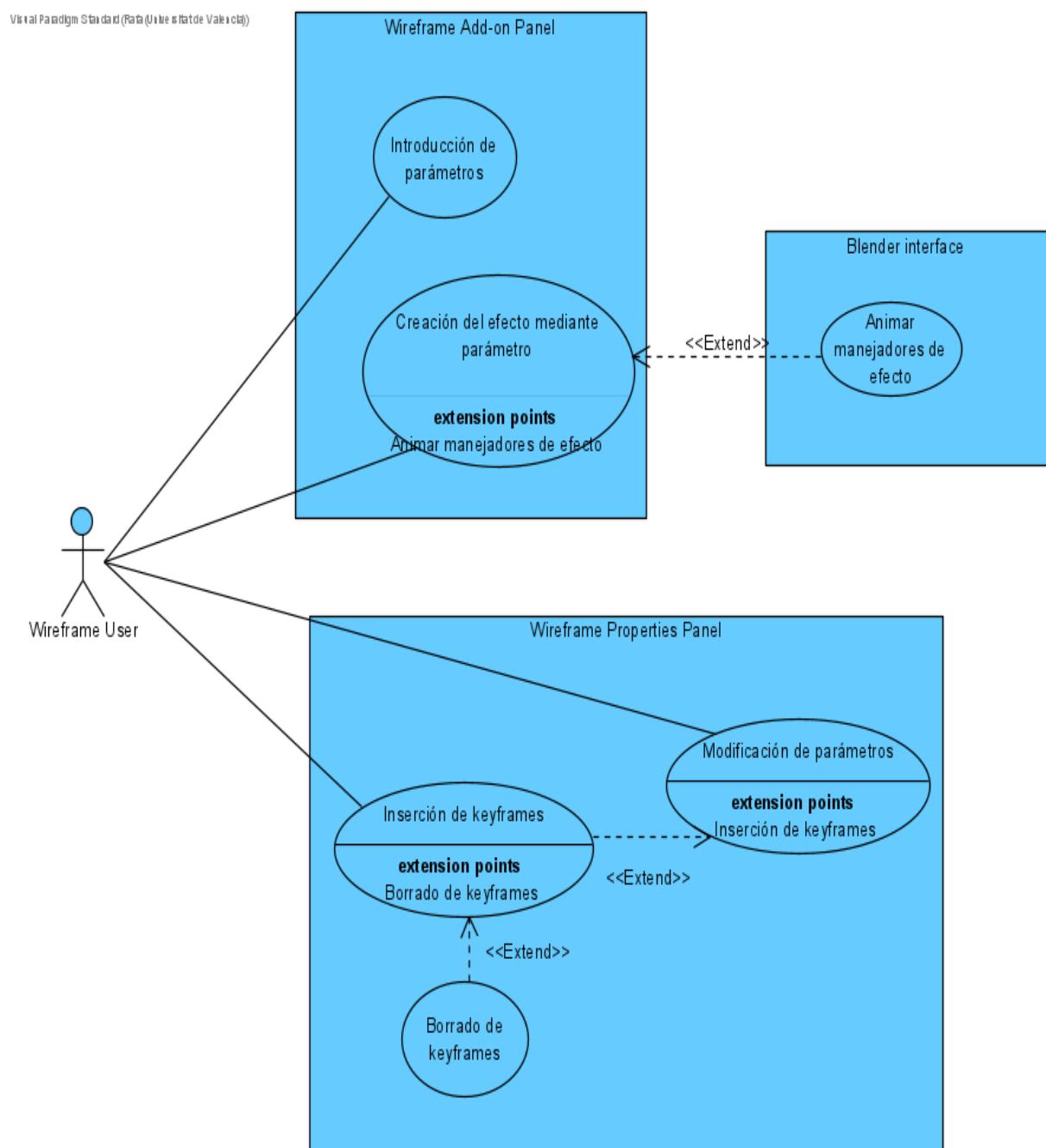


Figura 4.2: Diagrama de casos de uso del efecto de transición wireframe

#### 4.1.1.4. Especificación de casos de uso del efecto de transición de wireframe

Identificador	W-CU-01
Nombre	Introducción de parámetros.
Actor	Usuario que quiera crear el efecto de transición wireframe con el add-on.
Descripción	En este caso de uso se introducen los valores iniciales de los parámetros en los campos correspondientes para crear el efecto deseado.
Precondición	Haber instalado correctamente el add-on.
Postcondición	Todos los valores contienen un valor dentro de los límites permitidos por los campos.
Caso básico	Se instala el add-on y, tras desplegar el panel del efecto de wireframe, se introducen los valores

Identificador	W-CU-02
Nombre	Creación de efecto de transición wireframe mediante el operador
Actor	Usuario que quiera crear el efecto de transición wireframe con el add-on.
Descripción	En este caso de uso, al objeto que se tenga seleccionado, se le aplicará el efecto de transición de wireframe con los valores de los parámetros introducidos mediante el operador del panel del efecto.
Precondición	Todos los parámetros han de tener algún valor introducido.
Postcondición	Aplicación del efecto al objeto seleccionado.
Caso básico	Tras introducir los valores de los parámetros se clica en el operador del panel del efecto para generarlo.

Identificador	W-CU-03
Nombre	Animación de los manejadores del efecto de transición de wireframe.
Actor	Usuario que haya aplicado el efecto transición de wireframe y quiera animarlo mediante los manejadores.
Descripción	En este caso de uso, se animarán los manejadores del efecto efecto de transición de wireframe de tal manera que el objeto al que se le aplica desaparezca con una transición a hilo de alambre cuando los manejadores se acerquen a él.
Precondición	Debe haber un objeto con, al menos, con el efecto aplicado.
Postcondición	Animación del efecto sobre el efecto al que se le aplicó.
Caso básico	Tras aplicar el efecto a un objeto se animan los manejadores de manera que el efecto suceda como el usuario decida.

Identificador	W-CU-04
Nombre	Modificación de parámetros del efecto de transición wireframe.
Actor	Usuario que quiera modificar algunos de los parámetros iniciales con los que se creó el efecto.
Descripción	En este caso de uso se modifican los valores de algunos de los parámetros con los que se originó el efecto para modificarlo de acuerdo a estos.
Precondición	Ha de haber un objeto con el efecto aplicado en la escena.
Postcondición	Los valores introducidos cambian el aspecto del efecto inicialmente aplicado.
Caso básico	Tras aplicar el efecto, se clica sobre el objeto receptor y se abre el menú de propiedades del objeto, donde se abrirá la pestaña de propiedades del efecto y se modificarán los parámetros.

Identificador	W-CU-05
Nombre	Inserción de keyframes en propiedades del efecto de transición de wireframe.
Actor	Usuario que quiera insertar keyframes en las propiedades del efecto
Descripción	En este caso de uso se clica sobre el operador de insertar keyframes del panel de propiedades del efecto, dentro del panel propiedades del objeto, en la interfaz de Blender, para animar las propiedades que se modifiquen insertando keyframes en el frame que indique el marcador de la línea de tiempo.
Precondición	Ha de haber un objeto con el efecto aplicado en la escena.
Postcondición	Las propiedades que tengan los valores iniciales con los que el efecto fue creado y aquellas que hayan sido modificadas tendrán keyframes en los frames indicados por el marcador de la línea de tiempo cuando se clicó el operador .
Escenario básico	Tras aplicar el efecto, se clica sobre el objeto receptor y se abre el menú de propiedades del objeto, donde se abrirá la pestaña de propiedades del efecto y se clicará en el operador de inserción de keyframes para insertar los keyframes en el frame actual de la línea de tiempo.
Escenario alternativo	Se trata de insertar un keyframe en un propiedad en un frame que donde ya posee un keyframe. El último keyframe sobrescribe al primero.

Identificador	W-CU-06
Nombre	Borrado de keyframes en propiedades del efecto de transición de wireframe.
Actor	Usuario que quiera borrar los keyframes en las propiedades del efecto.
Descripción	En este caso de uso se clica sobre el operador de borrar keyframes del panel de propiedades del efecto, dentro del panel propiedades del objeto, en la interfaz de Blender para para eliminar los keyframes de las propiedades que hubieran sido animadas en el frame que indique el marcador de la línea de tiempo.
Precondición	Ha de haber un objeto con el efecto aplicado con una propiedad animada, al menos.
Postcondición	Las propiedades que tuvieran keyframes en el frame marcado por el marcador de la línea de tiempo cuando se clicó el operador dejarán de tenerlo.
Escenario básico	Tras haber insertado un keyframe en alguna propiedad del efecto se clicará en el operador de borrado de keyframes para eliminar los keyframes en el frame actual de la línea de tiempo de todas aquellas propiedades del efecto que tuvieran alguno.
Escenario alternativo	Se trata de borrar los keyframes de las propiedades en un frame donde no hay ninguno. El operador no tendrá efecto.

#### 4.1.1.5. Diagrama de casos de uso del rayo láser

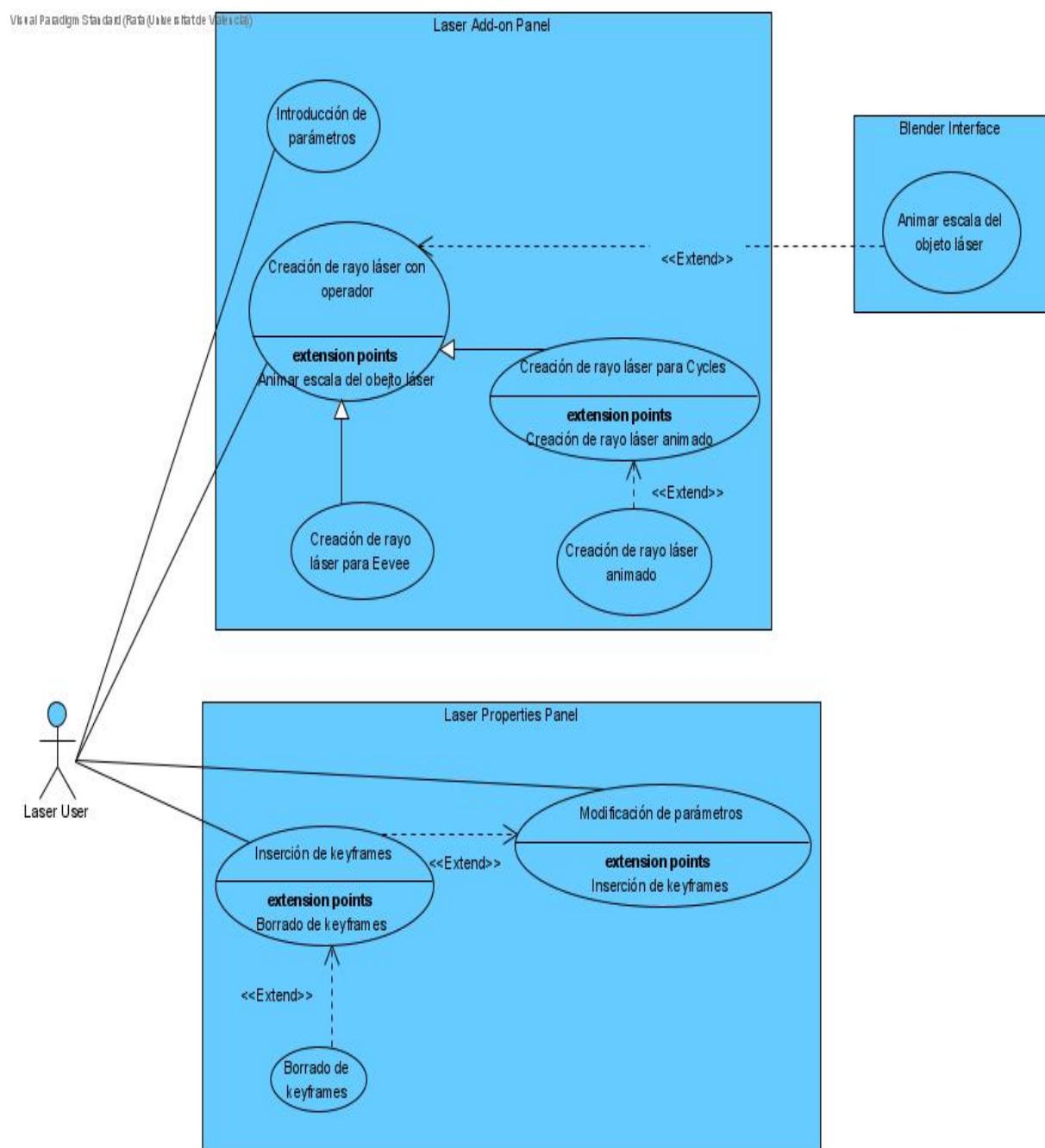


Figura 4.3: Diagrama de casos de uso del efecto de rayo láser

#### 4.1.1.6. Especificación de casos de uso del rayo láser

Identificador	L-CU-01
Nombre	Introducción de parámetros.
Actor	Usuario que quiera crear el efecto de láser con el add-on.
Descripción	En este caso de uso se introducen los valores iniciales de los parámetros en los campos correspondientes para crear el láser deseado.
Precondición	Haber instalado correctamente el add-on.
Postcondición	Todos los valores contienen un valor dentro de los límites permitidos por los campos.
Caso básico	Se instala el add-on y, tras desplegar el panel del efecto de láser, se introducen los valores

Identificador	L-CU-02
Nombre	Creación de efecto de rayo láser para Cycles
Actor	Usuario que quiera crear el efecto de rayo láser con el add-on usando el renderizador Cycles de Blender.
Descripción	En este caso de uso, se creará el objeto rayo láser con los valores de los parámetros introducidos mediante el operador del panel del efecto con un material visualizable y animable con el renderizador Cycles de Blender.
Precondición	El parámetro de renderer debe ser Cycles y todos los parámetros correspondientes han de tener algún valor introducido
Postcondición	Creación del objeto rayo láser para Cycles.
Caso básico	Tras introducir los valores de los parámetros se clica en el operador del panel del efecto para generar el láser con el material dedicado a Cycles.

Identificador	L-CU-03
Nombre	Animación de la escala del láser.
Actor	Usuario que haya creado un láser y quiera animarlo.
Descripción	En este caso de uso, se animarán la escala del rayo creado para hacer que aparezca progresivamente y que de la sensación de que está siendo disparado.
Precondición	Debe haber un objeto rayo láser creado.
Postcondición	Animación sobre la escala del láser.
Caso básico	Tras crear un objeto láser se anima la escala como el usuario desee para conseguir el efecto de disparo del láser.

Identificador	L-CU-04
Nombre	Creación de efecto de rayo láser para Eevee
Actor	Usuario que quiera crear el efecto de rayo láser con el add-on usando el renderizador Eevee de Blender.
Descripción	En este caso de uso, se creará el objeto rayo láser con los valores de los parámetros introducidos mediante el operador del panel del efecto con un material visualizable con el renderizador Eevee de Blender.
Precondición	El parámetro de renderer debe ser Eevee y todos los parámetros correspondientes han de tener algún valor introducido.
Postcondición	Creación del objeto rayo láser para Eevee.
Caso básico	Tras introducir los valores de los parámetros se clica en el operador del panel del efecto para generar el láser con el material dedicado a Eevee.

Identificador	L-CU-05
Nombre	Modificación de parámetros del rayo láser.
Actor	Usuario que quiera modificar algunos de los parámetros iniciales con los que se creó el láser.
Descripción	En este caso de uso se modifican los valores de algunos de los parámetros con los que se creó el rayo para modificarlo de acuerdo a estos.
Precondición	Ha de haber un objeto láser en la escena.
Postcondición	Los valores introducidos cambian el aspecto del láser creado inicialmente.
Caso básico	Tras crear el láser, se clica sobre este y se abre el menú de propiedades del objeto, donde se abrirá la pestaña de propiedades del láser y se modificarán los parámetros.

Identificador	L-CU-06
Nombre	Inserción de keyframes en propiedades del efecto de rayo láser.
Actor	Usuario que quiera insertar keyframes en las propiedades del láser
Descripción	En este caso de uso se clica sobre el operador de insertar keyframes del panel de propiedades del láser, dentro del panel propiedades del objeto, en la interfaz de Blender, para animar las propiedades que se modifiquen insertando keyframes en el frame que indique el marcador de la línea de tiempo.
Precondición	Ha de haber un objeto láser en la escena.
Postcondición	Las propiedades que tengan los valores iniciales con los que el efecto fue creado y aquellas que hayan sido modificadas tendrán keyframes en los frames indicados por el marcador de la línea de tiempo.
Escenario básico	Tras crear el láser, se clica sobre este y se abre el menú de propiedades del objeto, donde se abrirá la pestaña de propiedades del efecto y se clicará en el operador de inserción de keyframes para insertar los keyframes en el frame actual de la línea de tiempo.
Escenario alternativo	Se trata de insertar un keyframe en un propiedad en un frame que donde ya posee un keyframe. El último keyframe sobrescribe al primero.

Identificador	L-CU-07
Nombre	Borrado de keyframes en propiedades del efecto de rayo láser.
Actor	Usuario que quiera borrar los keyframes en las propiedades del láser.
Descripción	En este caso de uso se clica sobre el operador de borrar keyframes del panel de propiedades del láser, para eliminar los keyframes de las propiedades que hubieran sido animadas en el frame que indique el marcador de la línea de tiempo.
Precondición	Ha de haber un objeto láser con una propiedad animada, al manos.
Postcondición	Las propiedades que tuvieran keyframes en el frame marcado por el marcador de la línea de tiempo cuando se clicó el operador dejarán de tenerlo.
Escenario básico	Tras haber insertado un keyframe en alguna propiedad del láser se clicará en el operador de borrado de keyframes para eliminar los keyframes en el frame actual de la línea de tiempo de todas aquellas propiedades del efecto que tuvieran alguno.
Escenario alternativo	Se trata de borrar los keyframes de las propiedades en un frame donde no hay ninguno. El operador no tendrá efecto.

#### 4.1.1.7. Diagrama de casos de uso del escudo de fuerza

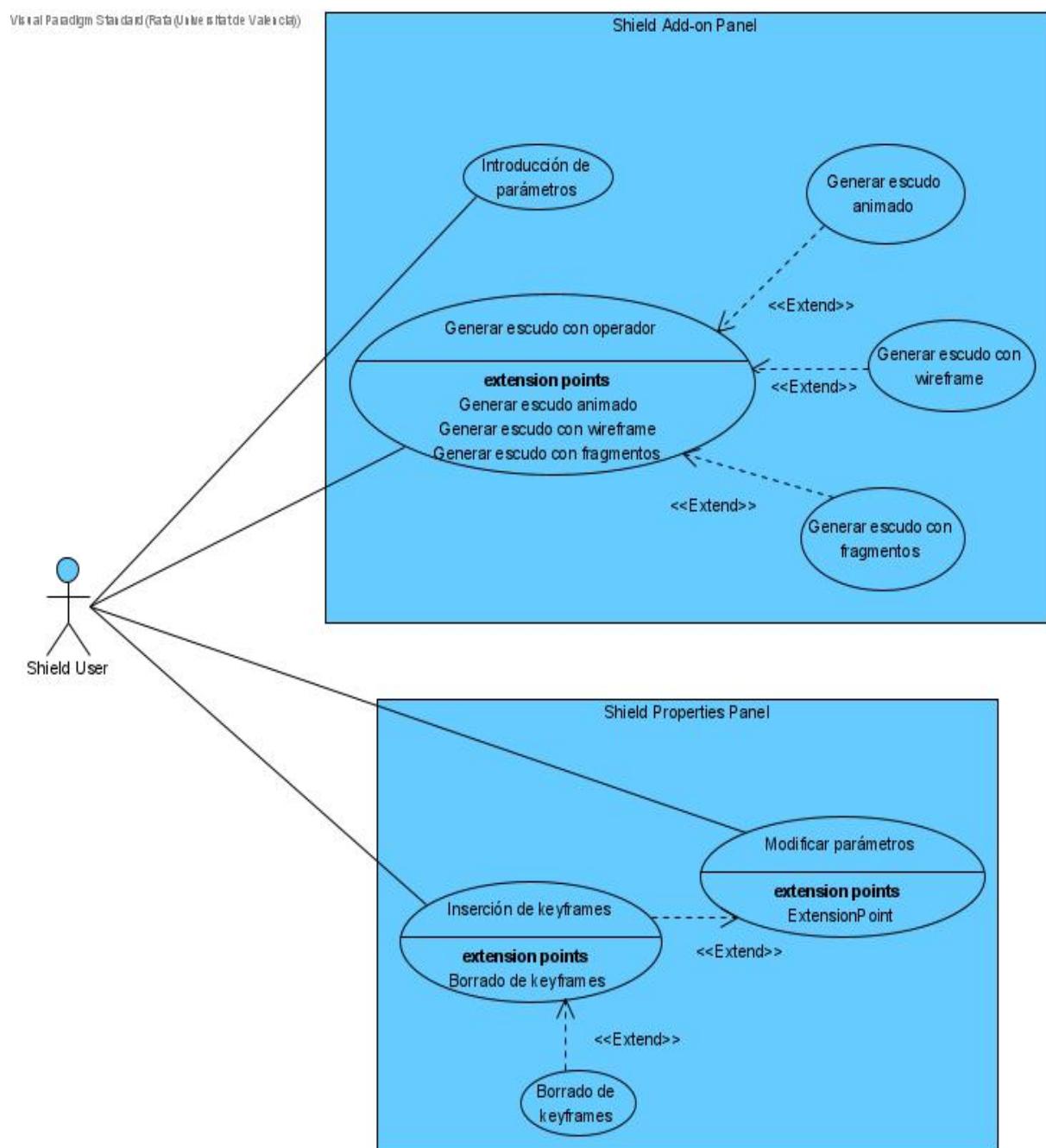


Figura 4.4: Diagrama de casos de uso del efecto de escudo de fuerza

#### 4.1.1.8. Especificación de casos de uso del escudo de fuerza

Identificador	E-CU-01
Nombre	Introducción de parámetros.
Actor	Usuario que quiera crear el efecto de escudo con el add-on.
Descripción	En este caso de uso se introducen los valores iniciales de los parámetros en los campos correspondientes para crear el escudo de fuerza deseado. Se podrá seleccionar si se quiere que el escudo tenga wireframe y fragmentos que salen de él y, en caso de que así sea, introducir sus parámetros de configuración. También se podrá seleccionar si se quiere que el material del escudo esté animado o no y en caso afirmativo introducir los parámetros correspondientes a la animación.
Precondición	Haber instalado correctamente el add-on.
Postcondición	Todos los valores contienen un valor dentro de los límites permitidos por los campos.
Caso básico	Se instala el add-on y, tras desplegar el panel del efecto de escudo de fuerza, se introducen los valores

Identificador	E-CU-02
Nombre	Creación de efecto de escudo de fuerza
Actor	Usuario que quiera crear el efecto de escudo de fuerza.
Descripción	En este caso de uso, se creará el objeto escudo de fuerza básico con los valores de los parámetros introducidos mediante el operador del panel del efecto que se generará con su escala animada por defecto para dar la sensación de que emerge. Esta animación se podrá ajustar desde el panel de propiedades.
Precondición	Todos los parámetros han de tener algún valor introducido.
Postcondición	Creación del objeto escudo de fuerza básico.
Caso básico	Tras introducir los valores de los parámetros se clica en el operador del panel del efecto para generar el escudo.

Identificador	E-CU-03
Nombre	Creación del efecto escudo de fuerza con wireframe.
Actor	Usuario que quiera crear un escudo con wireframe.
Descripción	En este caso de uso, se extenderá la funcionalidad del caso de uso anterior añadiendo un segundo escudo de wireframe que acompaña al primero.
Precondición	El uso de wireframe debe estar activo y todos los parámetros han de tener algún valor introducido.
Postcondición	Creación del objeto escudo de fuerza con wireframe.
Caso básico	Tras introducir los valores de los parámetros y haber marcado la opción de wireframe, introduciendo sus parámetros correspondientes, se clica en el operador del panel del efecto para generar el escudo.

Identificador	E-CU-04
Nombre	Creación del efecto escudo de fuerza con fragmentos.
Actor	Usuario que quiera crear un escudo con fragmentos salientes del escudo.
Descripción	En este caso de uso, se extenderá la funcionalidad del caso de uso dos del escudo, donde se crea el escudo principal, añadiendo un segundo escudo de fragmentos que acompaña al principal. Este escudo tendrá el material como el principal permitiendo cambiar determinados parámetros. Al igual que los efectos de escudo y de escudo wireframe también se genera con animación de escalado por defecto.
Precondición	El uso de fragmentos debe estar activado y todos los parámetros han de tener algún valor introducido.
Postcondición	Creación del objeto escudo de fuerza con fragmentos.
Caso básico	Tras introducir los valores de los parámetros y haber marcado la opción de fragmentos, introduciendo sus parámetros correspondientes, se clica en el operador del panel del efecto para generar el escudo.

Identificador	E-CU-05
Nombre	Creación del efecto escudo de fuerza con animación del material.
Actor	Usuario que quiera crear un escudo con el material animado.
Descripción	En este caso de uso, se extenderá la funcionalidad del caso de uso dos del escudo, donde se crea el escudo principal, haciendo que el material del escudo pueda estar animado cuando se cree. Esta característica podrá ser modificada en el panel de propiedades del objeto.
Precondición	Debe.
Postcondición	Creación del objeto escudo de fuerza con fragmentos.
Caso básico	Tras introducir los valores de los parámetros y haber marcado la opción de animación del material, introduciendo sus parámetros correspondientes, se clica en el operador del panel del efecto para generar el escudo con el material animado.

Identificador	E-CU-06
Nombre	Modificación de parámetros del escudo de fuerza.
Actor	Usuario que quiera modificar algunos de los parámetros iniciales con los que se creó alguno de los escudos.
Descripción	En este caso de uso se modifican los valores de algunos de los parámetros con los que se creó el escudo seleccionado: wireframe, fragmentos o principal, para modificarlo de acuerdo a estos.
Precondición	Ha de haber un objeto escudo en la escena.
Postcondición	Los valores introducidos cambian el aspecto del escudo creado inicialmente.
Caso básico	Tras crear el o los escudos, se clica sobre el que se desee modificar las propiedades y se abre el menú de propiedades del objeto, donde se abrirá la pestaña de propiedades del escudo y se modificarán los parámetros.

Identificador	E-CU-07
Nombre	Inserción de keyframes en propiedades del efecto de escudo.
Actor	Usuario que quiera insertar keyframes en las propiedades del escudo
Descripción	En este caso de uso se clica sobre el operador de insertar keyframes del panel de propiedades del escudo, dentro del panel propiedades del objeto, para animar las propiedades que se modifiquen insertando keyframes en el frame que indique el marcador de la línea de tiempo.
Precondición	Ha de haber un objeto escudo en la escena.
Postcondición	Las propiedades que tengan los valores iniciales con los que el efecto fue creado y aquellas que hayan sido modificadas tendrán keyframes en los frames indicados por el marcador de la línea de tiempo.
Escenario básico	Tras crear el o los escudos, se clica sobre el escudo sobre el que se quiera insertar keyframes y, en el menú de propiedades del objeto, se clicará en el operador de inserción de keyframes para insertar los keyframes en el frame actual de la línea de tiempo.
Escenario alternativo	Se trata de insertar un keyframe en un propiedad en un frame que donde ya posee un keyframe. El último keyframe sobrescribe al primero.

Identificador	E-CU-08
Nombre	Borrado de keyframes en propiedades del efecto de escudo.
Actor	Usuario que quiera borrar los keyframes en las propiedades del escudo.
Descripción	En este caso de uso se clica sobre el operador de borrar keyframes del panel de propiedades del escudo, para eliminar los keyframes de las propiedades que hubieran sido animadas en el frame que indique el marcador de la línea de tiempo.
Precondición	Ha de haber un objeto escudo con una propiedad animada, al manos.
Postcondición	Las propiedades que tuvieran keyframes en el frame marcado por el marcador de la línea de tiempo cuando se clicó el operador dejarán de tenerlo.
Escenario básico	Tras haber insertado un keyframe en alguna propiedad del escudo se clicará en el operador de borrado de keyframes para eliminar los keyframes en el frame actual de la línea de tiempo de todas aquellas propiedades del efecto que tuvieran alguno.
Escenario alternativo	Se trata de borrar los keyframes de las propiedades en un frame donde no hay ninguno. El operador no tendrá efecto.

## 4.1.2. Diagramas de actividad

### 4.1.2.1. Diagrama de actividad del efecto del rayo

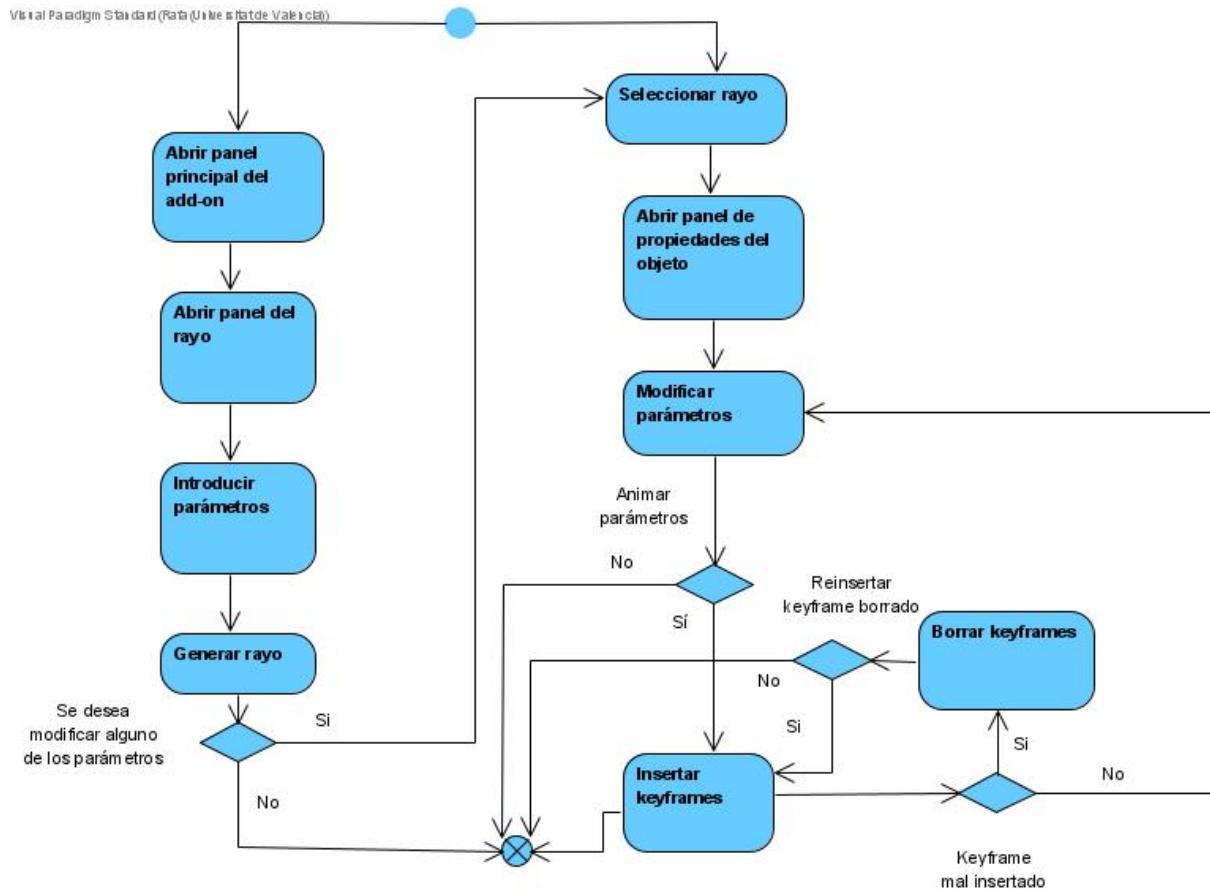


Figura 4.5: Diagrama de actividad del efecto del rayo

#### 4.1.2.2. Diagrama de actividad del efecto de wireframe

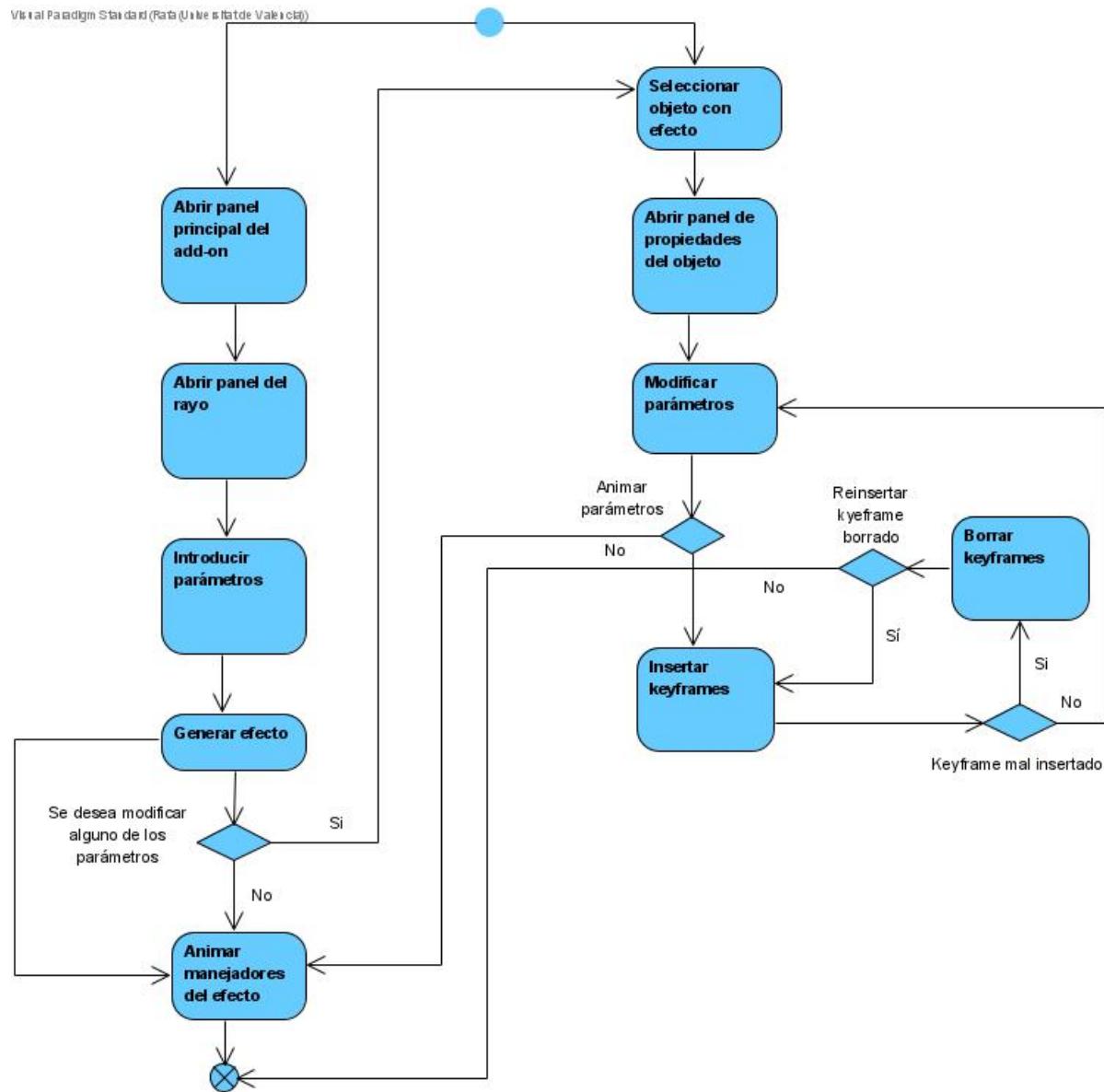


Figura 4.6: Diagrama de actividad del efecto de transición de wireframe

#### 4.1.2.3. Diagrama de actividad del efecto del rayo láser

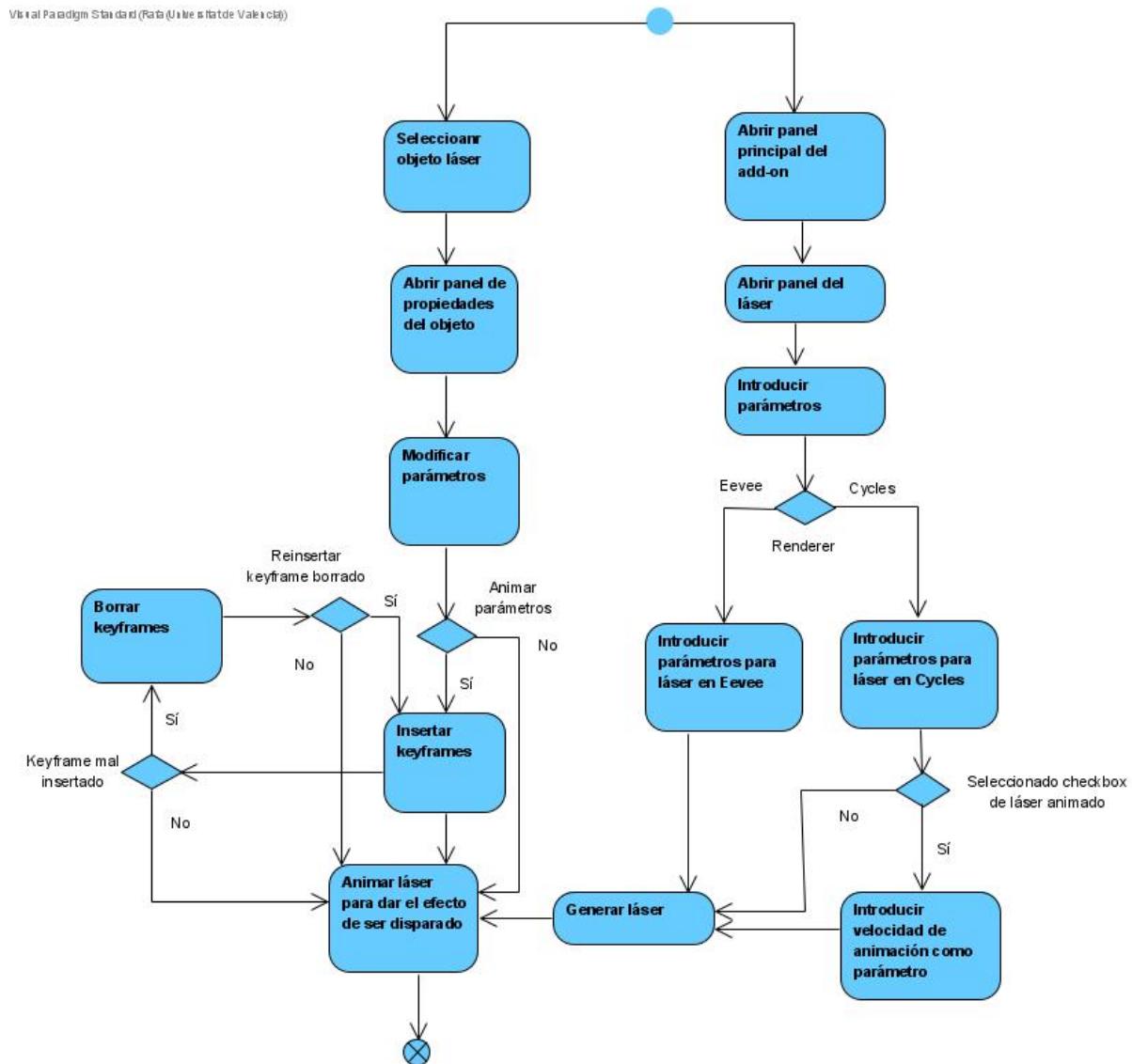


Figura 4.7: Diagrama de actividad del efecto del rayo láser

#### 4.1.2.4. Diagrama de actividad del efecto de escudo de fuerza

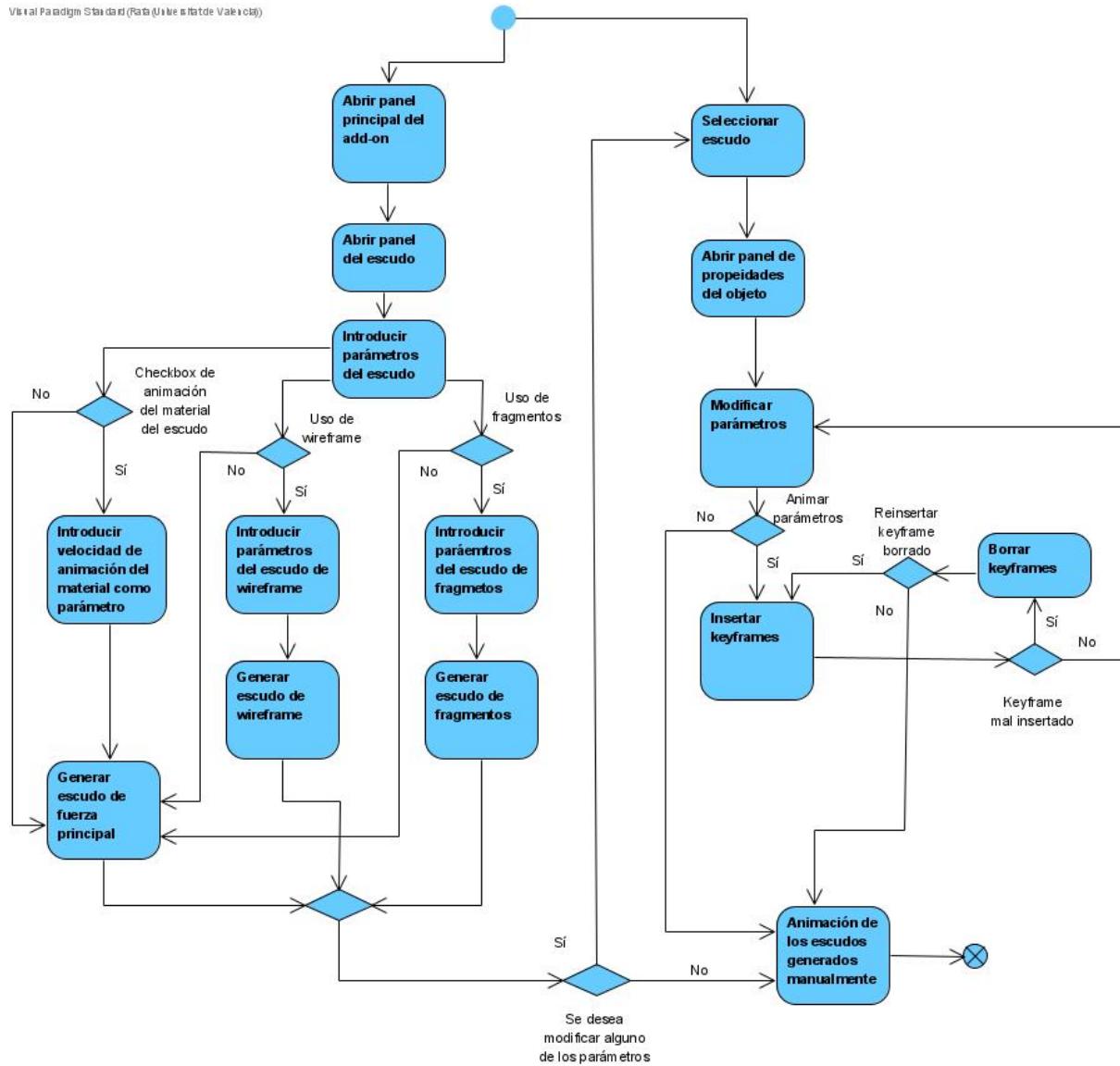


Figura 4.8: Diagrama de actividad del efecto de escudo de fuerza

#### 4.1.2.5. Diagrama de actividad general del add-on

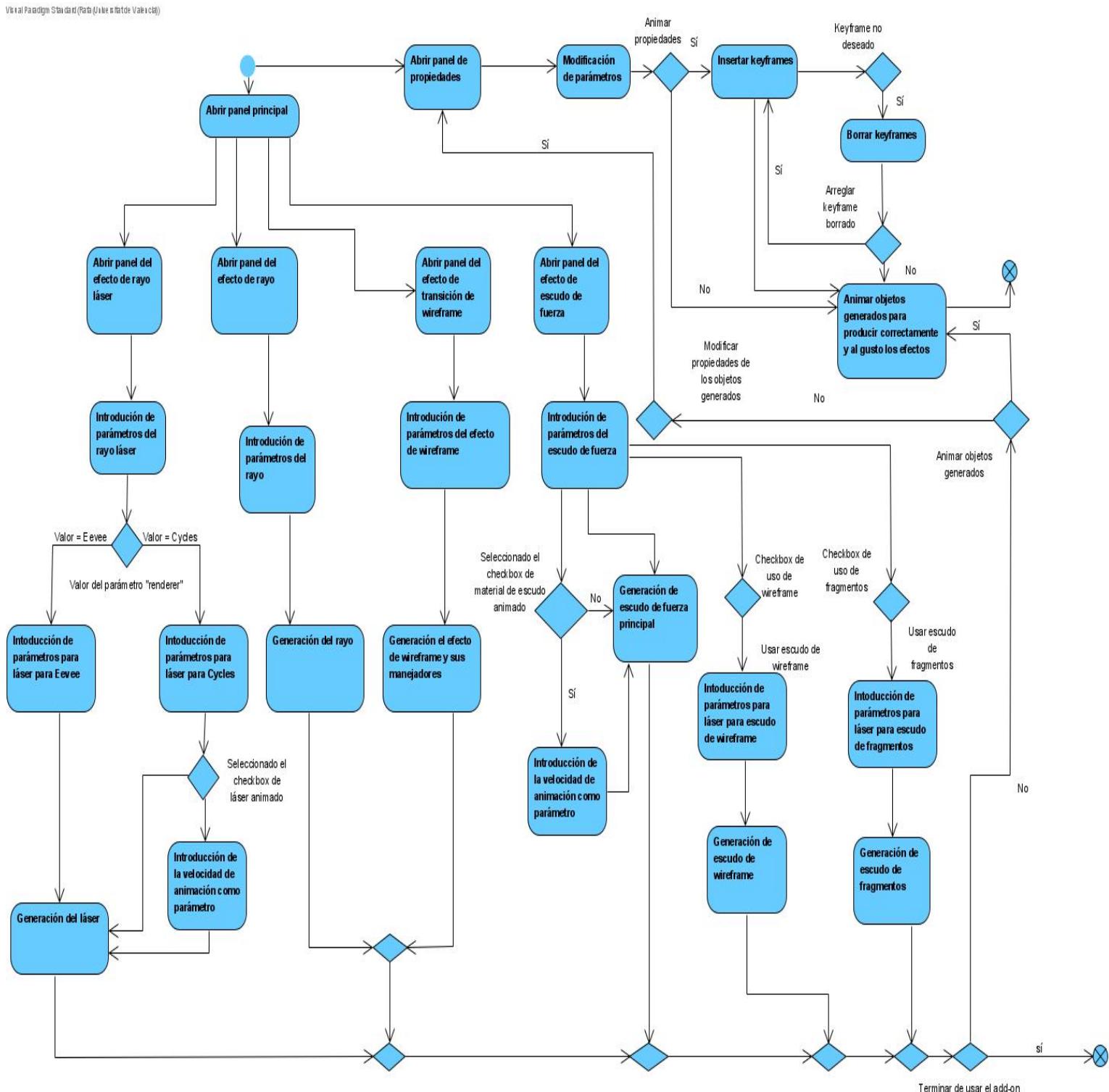


Figura 4.9: Diagrama de actividad general del add-on

## 4.2. Diseño

Una vez realizada la parte de análisis, en este capítulo explicaré cómo se han diseñado las partes del sistema y sus relaciones, así como su comportamiento mediante diagramas de clases y secuencia.

#### 4.2.1. Diagrama de clases

Este diagrama permite visualizar de manera global cuáles son y cómo se relacionan las diferentes entidades de este proyecto.

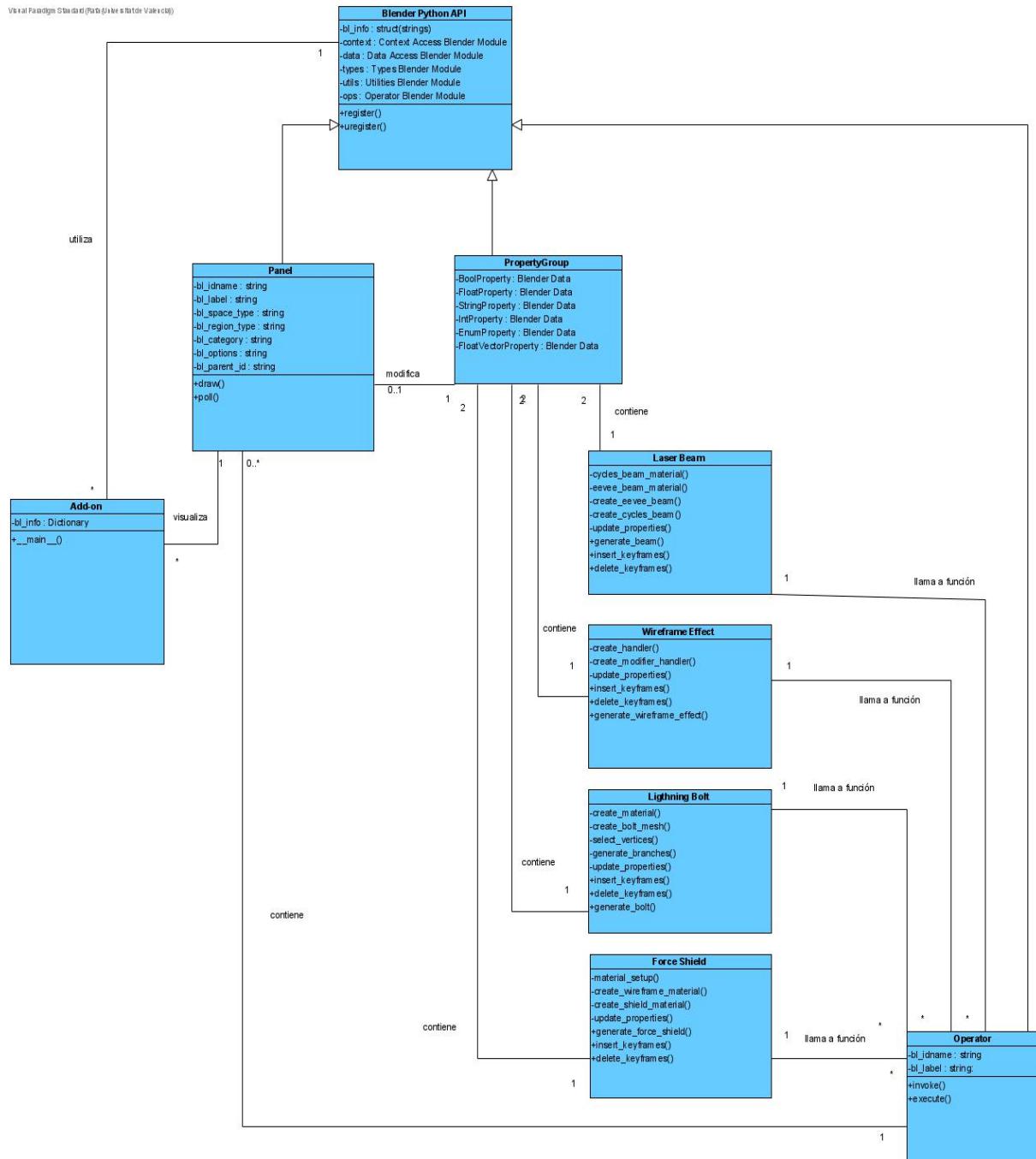


Figura 4.10: Diagrama de clases

A continuación explicaré las clases que aparecen en el diagrama:

- **Blender Python API:** Esta clase es una abstracción de la API de Blender para Python que contiene los atributos, clases y métodos de la API que son utilizados.
- **Add-on:** Aunque no es una clase como tal, el add-on ha sido representado como una clase pues cumple una función específica de registro de todas las demás clases y las propiedades en la clase Scene y Object de Blender para que el add-on funcione. Aunque no es necesario crear una clase para estas tareas, el código principal del add-on se puede considerar como una entidad representable como clase.
- **Panel:** Esta clase representa a todas las clases que heredan del tipo bpy.types.Panel, de la API de Blender. Esta clase representa a los paneles de los efectos así como a los subpaneles de propiedades creados dentro del panel de propiedades de objetos de Blender.
- **PropertyGroup:** Esta clase representa a las clases contenedoras de propiedades que heredan del tipo bpy.types.PropertyGroup de la API de Blender, las cuales se usan para guardar la configuración con la que se crean los efectos y, una vez generados estos, guardan los valores de las propiedades de los objetos con los que fueron creados. Existen dos instancias de esta clase por efecto: la que guarda la configuración para crearlo y la que, una vez creado el objeto que genera el efecto, guarda los valores de sus propiedades.
- **Operator:** Esta clase representa a las clases de los operadores de los efectos y de inserción y borrado de keyframes que heredan del tipo bpy.types.Operator de la API de Blender, las cuales se usan para llamar a las funciones de generación de los efectos y de operaciones de keyframes.
- **Laser Beam:** Esta clase representa al conjunto de funciones que se utilizan para generar el objeto láser. Al igual que la clase de Add-on, no es una clase como tal, no obstante, y por estructuración y facilitación de lectura y escritura del código, se separó el código correspondientes a los efectos en ficheros diferentes, por tanto, para llamar a las funciones de este efecto se deberá incluir el fichero en el fichero del add-on, lo que se asemeja , en cierta manera, a la instancia de un objeto de una clase para poder usar sus métodos. Además, en la clase Laser Beam se ha definido un método llamado update\_properties que representa a los métodos definidos de update para cada propiedad del láser por simplificación del diagrama.
- **Wireframe Effect:** Clase similar a Laser Beam. Contiene los métodos para generación del efecto de transición de wireframe y las funciones de update de los parámetros.
- **Ligthning Bolt:** Contiene los métodos para generación del efecto de rayo y las funciones de update de los parámetros.
- **Force Shield:** Contiene los métodos para generación del escudo de fuerza y las funciones de update de los parámetros.

## 4.2.2. Diagramas de secuencia

### 4.2.2.1. Diagramas de secuencia del efecto de rayo

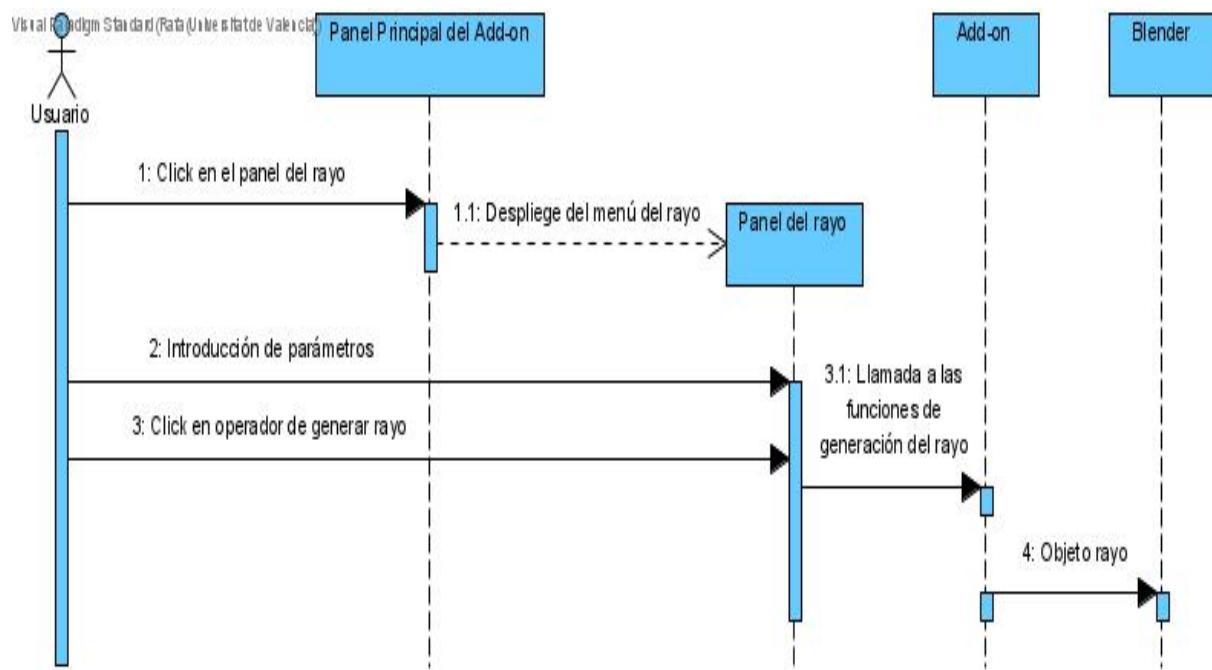


Figura 4.11: Diagrama de secuencia para el caso de uso de generar rayo

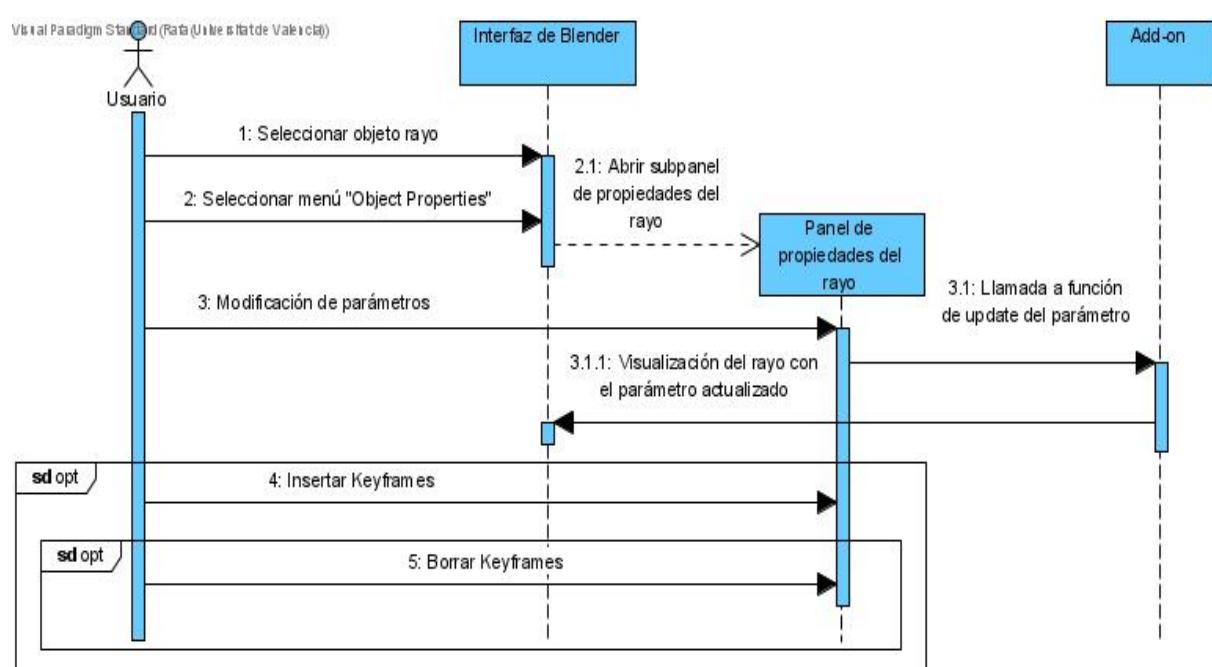


Figura 4.12: Diagrama de secuencia para el caso de uso de modificación de parámetros

#### 4.2.2.2. Diagramas de secuencia del efecto de transición wireframe

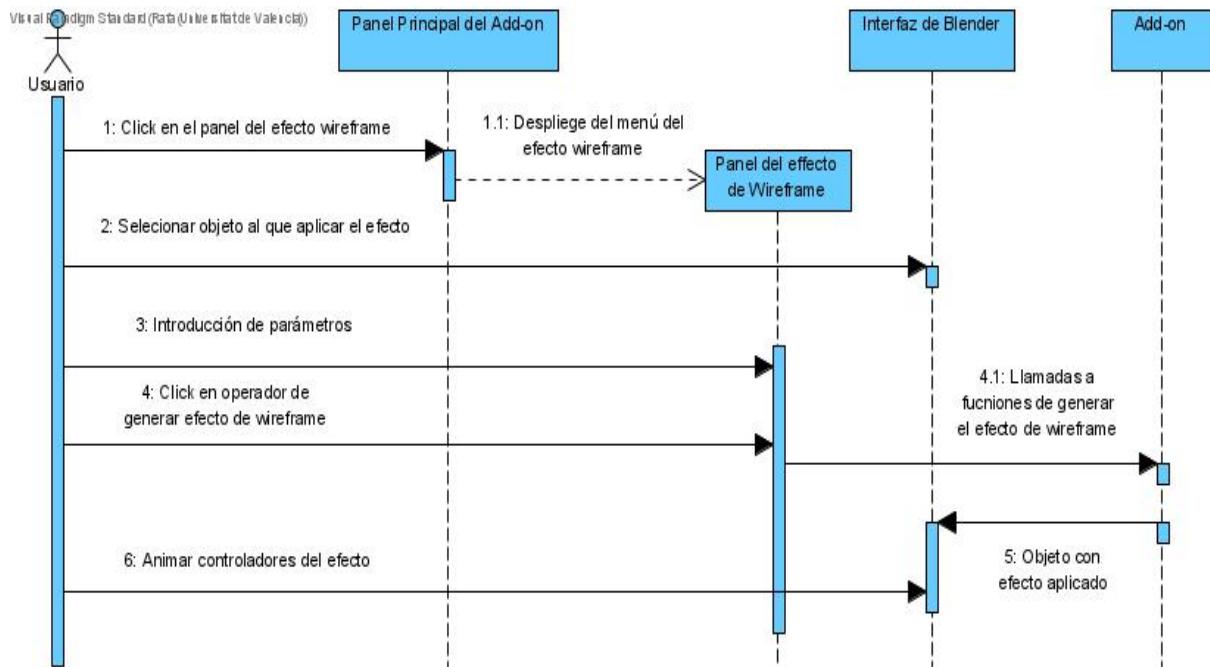


Figura 4.13: Diagrama de secuencia para el caso de uso de generar efecto de wireframe

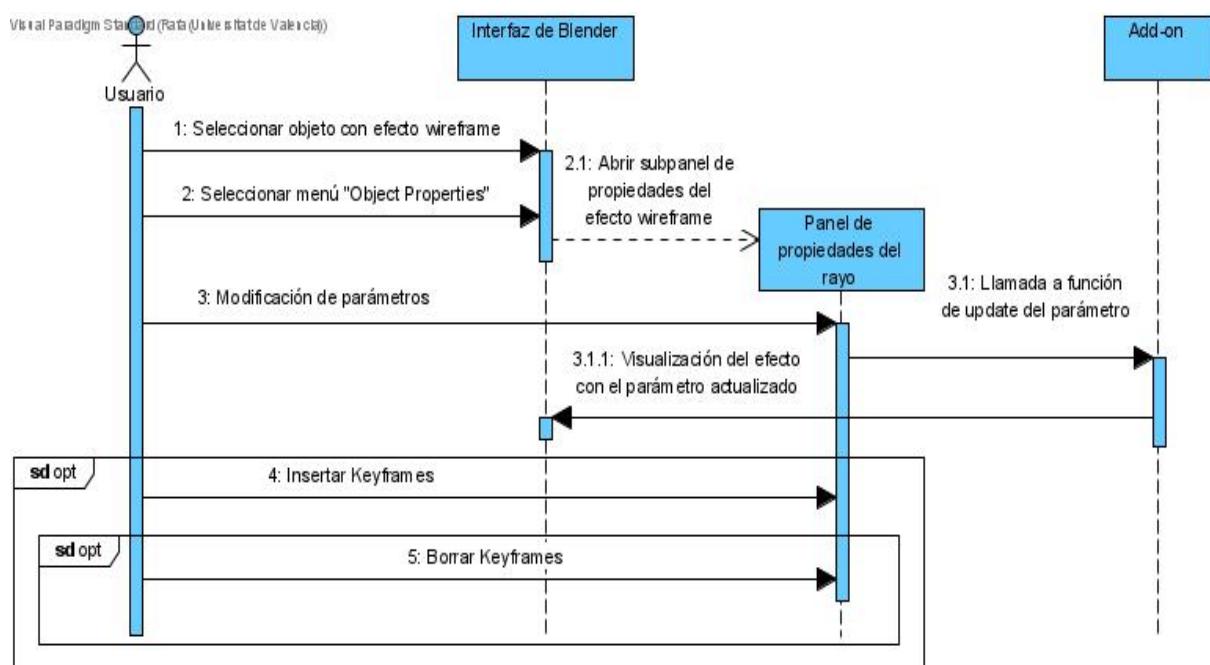


Figura 4.14: Diagrama de secuencia para el caso de uso de modificación de parámetros

#### 4.2.2.3. Diagramas de secuencia del efecto de rayo láser

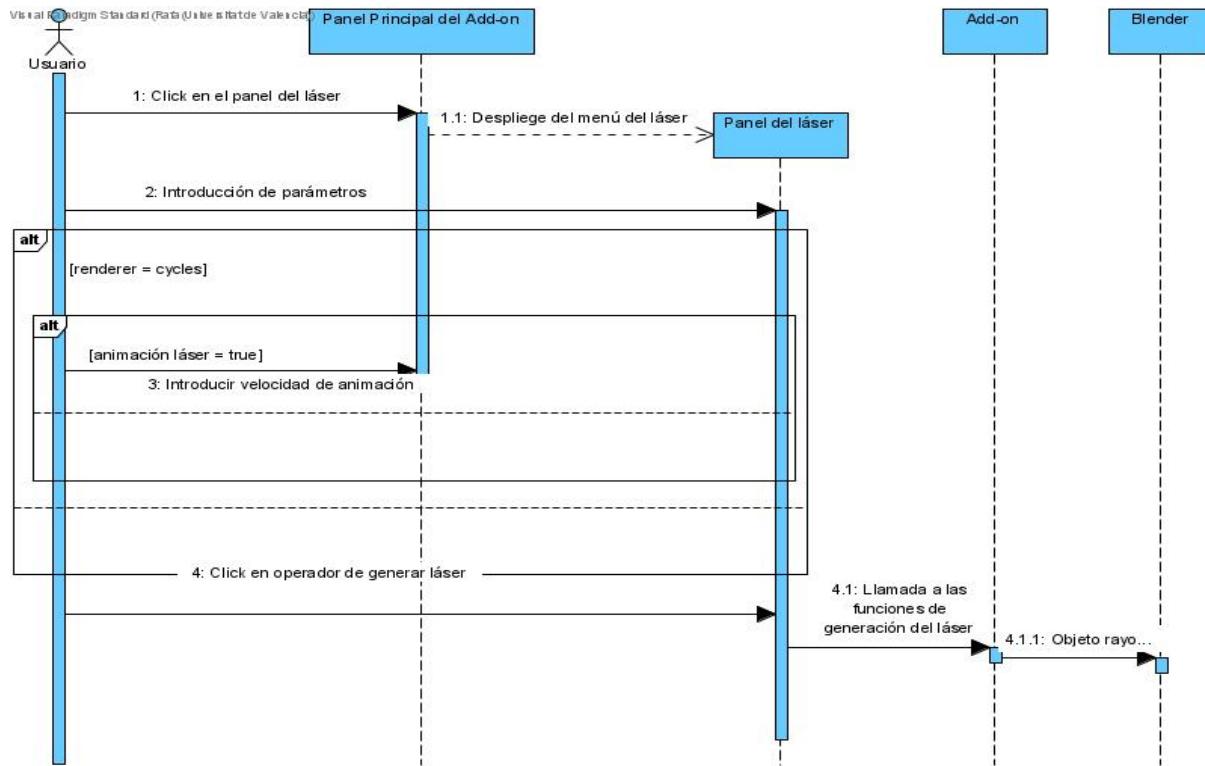


Figura 4.15: Diagrama de secuencia para el caso de uso de generar efecto de rayo láser

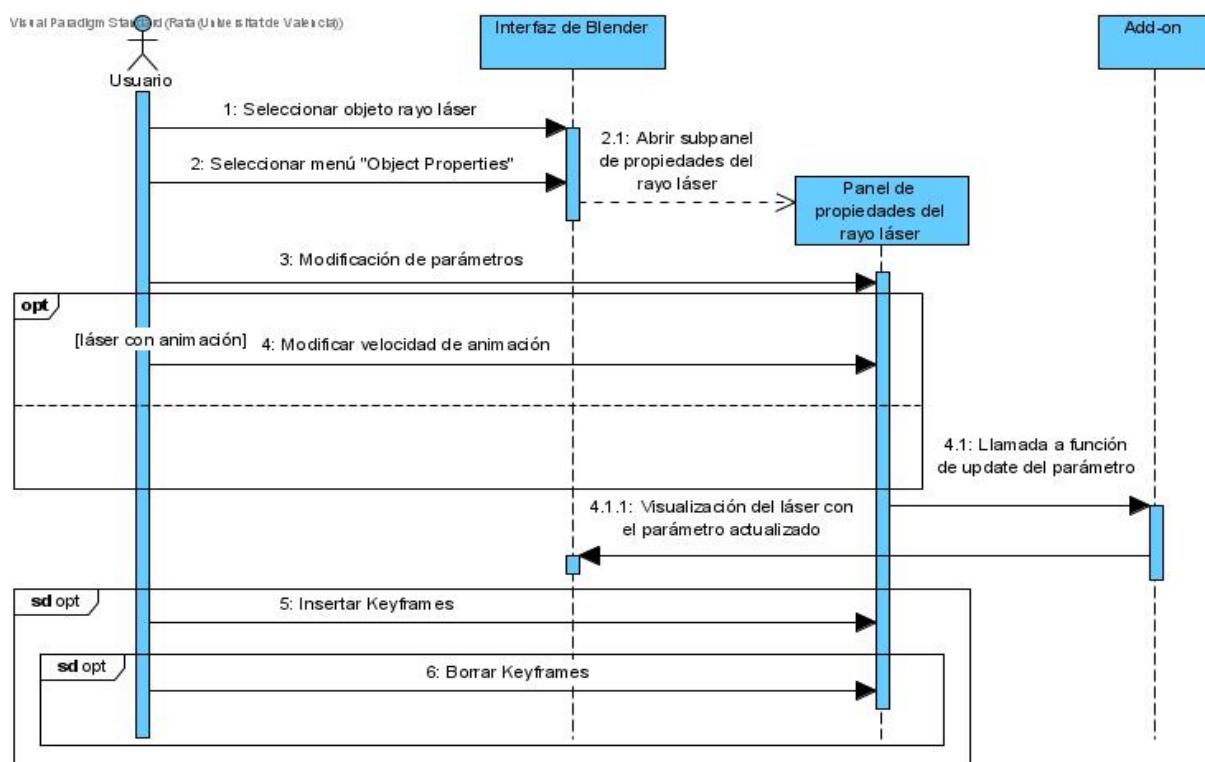


Figura 4.16: Diagrama de secuencia para el caso de uso de modificación de parámetros

#### 4.2.2.4. Diagramas de secuencia del efecto de escudo de fuerza

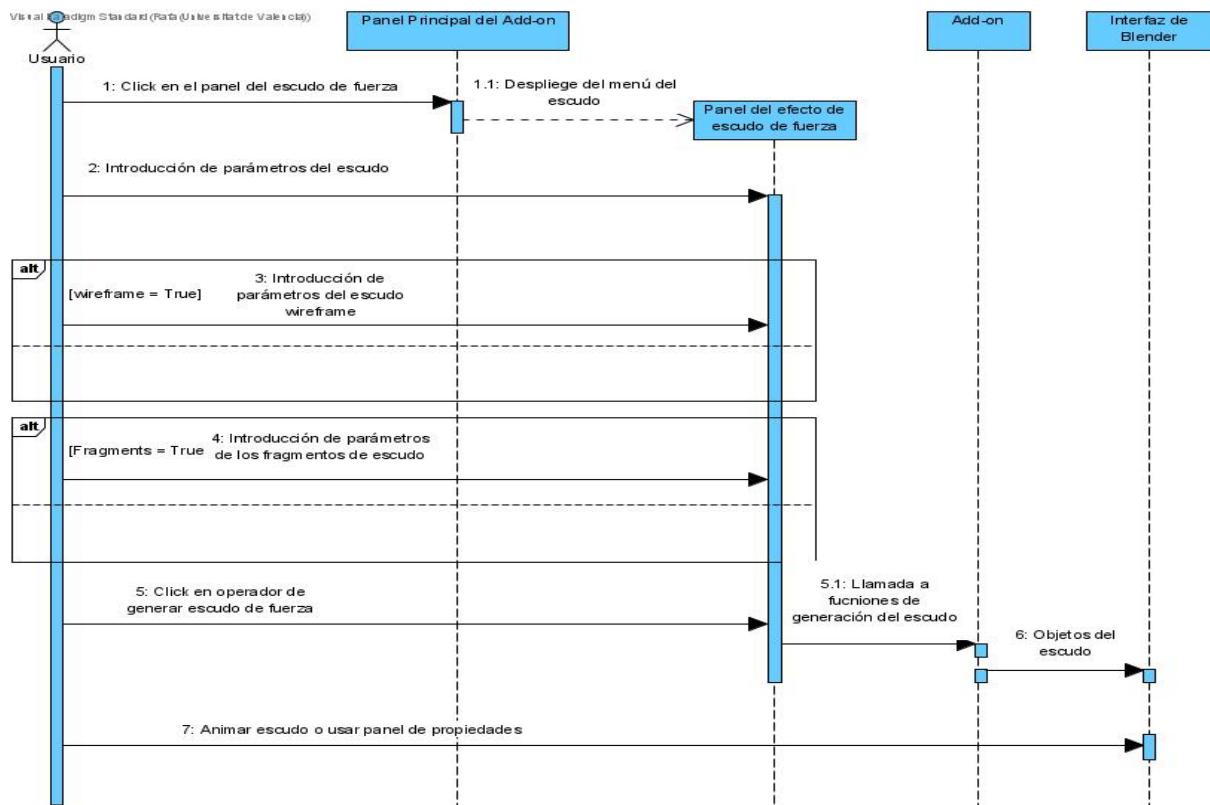


Figura 4.17: Diagrama de secuencia para el caso de uso de generar de escudo de fuerza

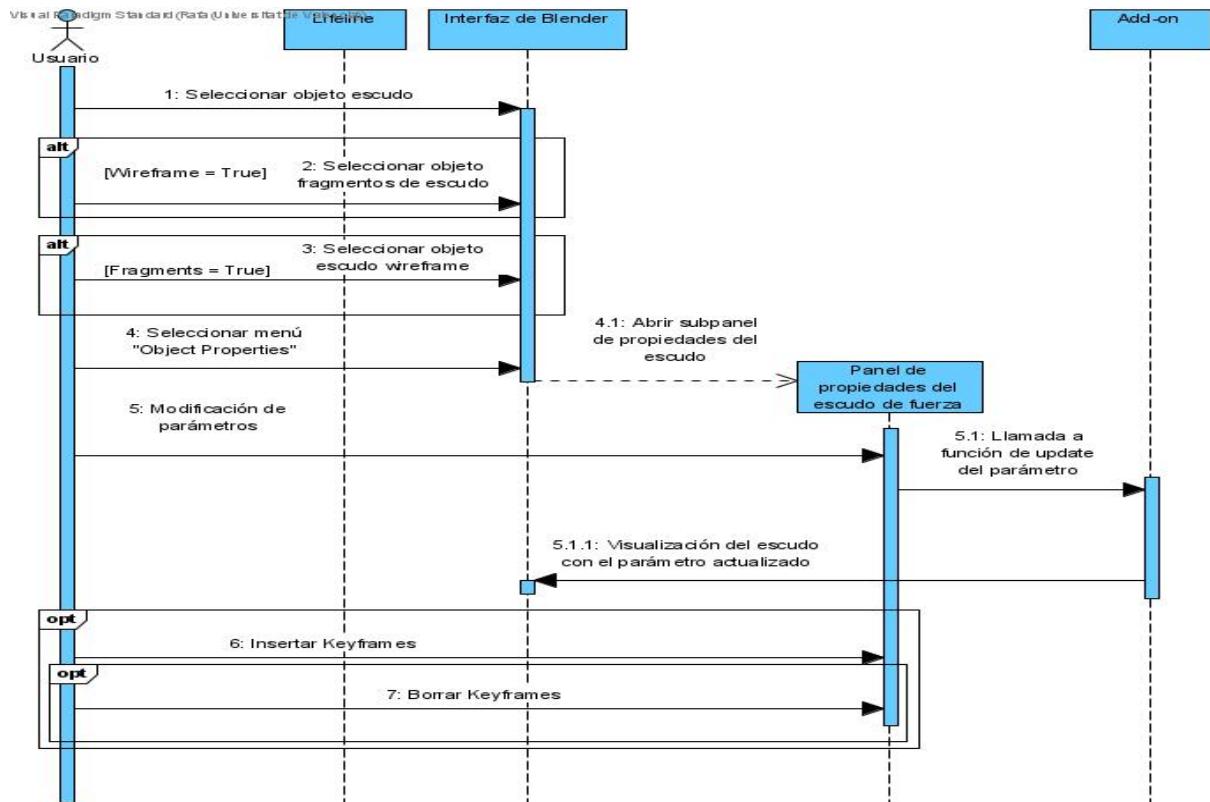


Figura 4.18: Diagrama de secuencia para el caso de uso de modificación de parámetros

#### 4.2.3. Diseño de la interfaz

Para la interfaz se ha optado por colocar el panel principal del add-on en el menú lateral del espacio 3d de Blender, pues es un menú que sólo se podrá utilizar cuando se trabaje sobre este espacio. Los paneles de los efectos serán desplegables y estarán plegados por defecto. En los paneles de los efectos aparecerá una lista de parámetros para introducir la configuración con la que se pretende generar el efecto y un operador al final que, al clicarlo, generará el efecto correspondiente. Los parámetros se podrán modificar en los paneles mediante la introducción directa del valor o clicando y deslizando el cursor sobre el parámetro para incrementar o decrementar su valor hasta un valor mínimo o máximo que puede tener definido el parámetro. Los color de los paneles, la fuente de letra y diseño de los elementos vienen definidos por la propia clase de la que heredan los paneles: bpy.types.Panel, de la API de Blender. Por último, los subpaneles de propiedades de los objetos creados con los efectos serán similares a los paneles de configuración de los mismos, pues son instancias del mismo tipo de clase y son un subconjunto de los paneles de configuración en cuanto a las propiedades que contiene. La única diferencia es que en lugar de un operador de generación de efectos tienen los dos operadores de relativos a los keyframes los cuales llevan un ícono indicativo de si es de inserción o borrado, los cuales vienen con el propio Blender.

## 4.3. Implementación

Una vez explicados el análisis y el diseño pasará a explicar de manera más detallada cómo se han implementado los efectos y el add-on, para lo cual, comentaré funcionalidades, características destacables que han hecho que se implementen de esa manera, integración en el addon, como interactúan con el usuario y otros aspectos relevantes de mención.

El desarrollo de toda la implementación está hecho en Python, pues como ya se dijo en el apartado de especificación del software, es el lenguaje que utiliza Blender para el desarrollo de sus addon y para el cual tiene desarrollada su API.

Ninguno de los efectos que se han integrado en esta herramienta son efectos originales e ideados por mí, sino que han sido adaptados de efectos hechos por otros artistas y divulgados a través de Internet, de manera que puedan ser generados automáticamente mediante script, introduciendo parámetros para que estos efectos puedan ser producidos por cualquier usuario sin los conocimientos necesarios para desarrollarlos por sí mismo. En la explicación de la implementación de cada efecto se hará referencia al autor y propietario intelectual de dicho efecto.

### 4.3.1. Implementación del add-on

Supone la parte principal de la herramienta y la base sobre la que se integran las funcionalidades de los efectos que contiene. Esta constituido por el archivo `__init__.py` y contiene todas las clases de los paneles y operadores que compondrán la interfaz de la herramienta. Consta en total de veintiocho clases, las cuales se clasifican en clases de paneles, subpaneles, operadores y contenedores de propiedades de los diferentes efectos.

El panel principal es el contenedor de los demás paneles, los cuales permitirán introducir los parámetros para la creación del efecto. Este panel padre de los demás se integrará en la interfaz de usuario, en el menú desplegable de la esquina superior izquierda de la ventana de vista 3D, y cuya única función es ser el contenedor de los demás paneles.

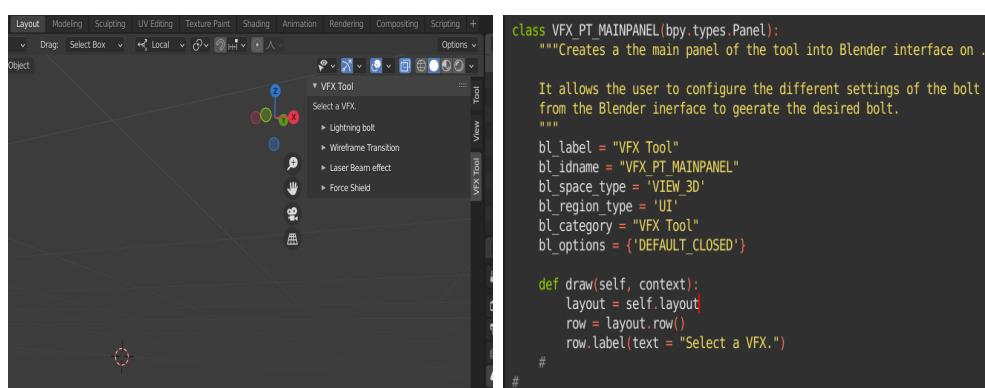


Figura 4.19: Panel principal del add-on en la interfaz.  
Figura 4.20: Código del panel principal.

#### 4.3.1.1. Paneles

Las clases de los paneles de los efectos son ocho, pues aunque para este trabajo sólo se han desarrollado cuatro efectos, cada efecto consta de un panel de configuración, en el cual se introducirán los parámetros de creación del efecto. Estos paneles cuentan con una serie de parámetros para generar los efectos con las características que desee el usuario, algunos de los parámetros de los paneles serán booleanos, que se representarán como checkboxes, y servirán para determinar si se permiten usar ciertos parámetros o no, ocultando o desactivando estos parámetros vinculados a su valor booleano.

```
class BOLT_PT_PANEL(bpy.types.Panel):
    """Creates a subpanel into main panel of the tool.

    It allows the user to configure the different settings of the bolt
    from the Blender interface to generate the desired bolt.
    """

    bl_label = "Lightning bolt"
    bl_idname = "BOLT_PT_PANEL"
    bl_space_type = 'VIEW_3D'
    bl_region_type = 'UI'
    bl_category = "Lightning bolt"
    bl_parent_id = "VFX_PT_MAINPANEL"
    bl_options = {'DEFAULT_CLOSED'}
```

```
def draw(self, context):
    bolt_settings = context.scene.bolt_settings
    layout = self.layout
    layout.scale_x = 0.5
    col = layout.column(align=True)
    col.scale_x = 0.5
    col.prop(bolt_settings, "bolt_length")
    col.prop(bolt_settings, "bolt_opacity")
    col.prop(bolt_settings, "bolt_strength")
    col.prop(bolt_settings, "bolt_thinness")
    col.prop(bolt_settings, "bolt_sharpness_x")
    col.prop(bolt_settings, "bolt_sharpness_y")
    col.prop(bolt_settings, "bolt_branches_dens")
    col.prop(bolt_settings, "bolt_branches_depth")
    col.prop(bolt_settings, "bolt_branches_length")
    col.prop(bolt_settings, "bolt_max_subbranches")
    col.prop(bolt_settings, "bolt_color")
    col.prop(bolt_settings, "bolt_begin")
    col.prop(bolt_settings, "bolt_end")
    row = layout.row()
    row.enabled = context.mode == 'OBJECT'
    row.operator('vfx.bolt_operator')
    #
```

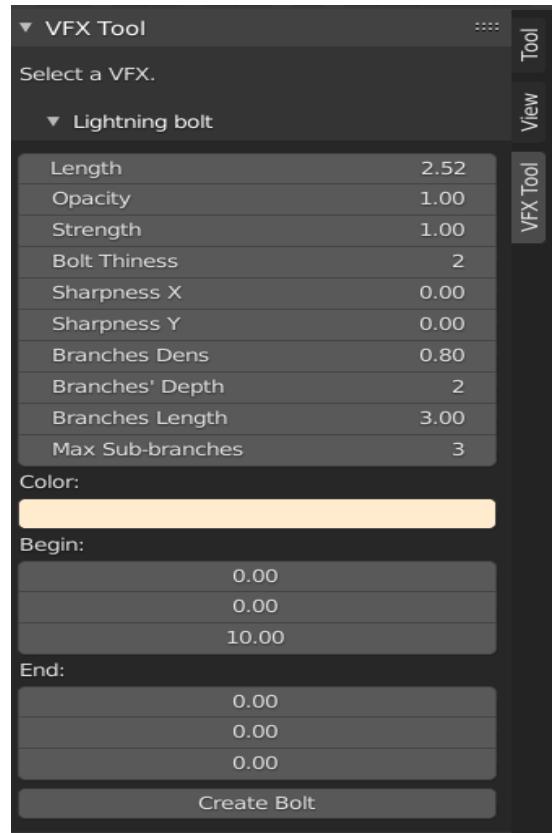


Figura 4.21: código del panel del efecto del rayo.

Figura 4.22: panel del efecto del rayo.

#### 4.3.1.2. Subpaneles de propiedades

Dentro del panel de propiedades de objetos, en la interfaz de Blender, habrá un subpanel del objeto del efecto generado que será visible al ser seleccionado. A través de este supanel se podrán modificar ciertos parámetros que no estén relacionados con la estructura fundamental del objeto creado, pero sí con sus características visuales, generalmente del material. Este subpanel, además, constará de dos operadores de inserción de keyframes y borrado de los mismos en los fotogramas marcados por el marcador de la línea de tiempo.

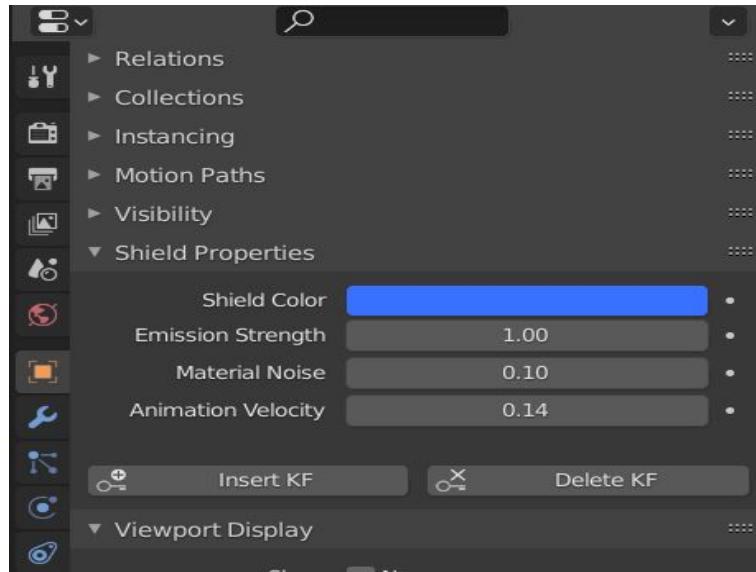


Figura 4.23: Subpanel de propiedades del efecto.

Las propiedades de los objetos, al cambiar su valor, llaman a unas funciones de actualización que están asignadas a los parámetros dentro de las clases contenedoras de propiedades, de las cuales hablaré más tarde.

```
bolt_strength: bpy.props.FloatProperty(
    name = "Strength",
    description = "Indicates how strong is the emission",
    default = 1,
    update = bolt_strength_changed
)
```

Figura 4.24: Función de update asignada a la propiedad.

```
def bolt_strength_changed(self, context):
    """Bolt length update function.

    It changes the strength of the bolt's material emission.
    """

    bolt_properties = context.object.bolt_properties
    context.object.data.materials[0].node_tree.nodes['Group']
    #
```

Figura 4.25: Código de la función de update.

#### 4.3.1.3. Operadores

Como se ha dicho antes, dentro de los subpaneles que se integran en el panel de propiedades de objetos en Blender, hay dos operadores de inserción y borrado de keyframes, los cuales insertan y borran respectivamente keyframes en el frame indicado por el marcador de la timeline o línea de tiempo de la interfaz de Blender. Estos operadores no actúan sobre parámetros fundamentales de la composición estructural de los objetos creados, como serían el número de subramas del rayo, el número de niveles de subramas, el grosor del rayo, etc. Porque, por cómo están hechos, realizar esos cambios en tiempo real sobre los objetos sería imposible en muchos casos. Es por esto que las propiedades modificables en el panel de propiedades de los objetos de los efectos están relacionados, en gran parte, con sus materiales y modificadores.

```

class BOLT_OT_INSERT_KEYFRAME(bpy.types.Operator):
    """Operator that allows to insert a keyframe on properties that have been changed on the bolt
    at the current frame. It also inserts a keyframe if bolt has its initial settings.

    Displayed on bolt properties panel.
    """

    bl_label = 'Insert KF'
    bl_idname = 'bolt.insert_keyframe_operator'
    bl_description = 'Inserts a keyframe on properties that have been changed at current frame'
    bl_options = {'REGISTER', 'UNDO'}

    def execute(self, context):
        bolt_insert_keyframe(context)
        return {'FINISHED'}
    #

    def invoke(self, context, event):
        self.execute(context)
        return {'FINISHED'}
    #
#

```

Figura 4.26: Código del operador de inserción de keyframes.

```

class BOLT_OT_DELETE_KEYFRAME(bpy.types.Operator):
    """Operator that allows to delete the keyframes on properties that have been keyframed on the bolt
    at the current frame.

    Displayed on bolt properties panel.
    """

    bl_label = 'Delete KF'
    bl_idname = 'bolt.delete_keyframe_operator'
    bl_description = 'Deletes keyframes on properties that has keyframes at current frame'
    bl_options = {'REGISTER', 'UNDO'}

    def execute(self, context):
        bolt_delete_keyframe(context)
        return {'FINISHED'}
    #

    def invoke(self, context, event):
        self.execute(context)
        return {'FINISHED'}
    #
#

```

Figura 4.27: Código del operador de borrado de keyframes.

Esta imagen resume la implementación de todos los operadores de keyframes, los cuales solo se diferencian en que llaman a un método u otro, dependiendo del efecto al que correspondan.

Los operadores de los efectos también son similares. Son clases que contienen dos métodos: invoke y execute. El método invoke se llama al detectar el clic sobre el operador y llama al método execute, quien se encarga de llamar a la función de generar el efecto que corresponda. Además, en este método, la función que genera el efecto devuelve el objeto generado para implementar dicho efecto y actualiza sus propiedades iniciales de acuerdo a los parámetros introducidos en el panel de creación del mismo, en el panel principal, para que haya coherencia entre los valores que aparecen en los parámetros de las propiedades y los del panel de configuración. El ejemplo del panel del rayo sirve para mostrar como es un operador. Solo cambia para los demás la función que se llama de generación del efecto y las propiedades que se actualizan.

```

class VFX_OT_BOLT(bpy.types.Operator):
    """Operator that generates the lightning bolt with the settings in bolt settings panel, on the main VFX panel.

    Displayed on bolt settings panel.
    """

    bl_label = "Create Bolt"
    bl_idname = 'vfx.bolt_operator'
    bl_parent_id = "BOLT_PT_PANEL"

    def execute(self, context):
        bolt = generate_bolt(context)

        bolt.bolt_properties.bolt_color = bpy.context.scene.bolt_settings.bolt_color
        bolt.bolt_properties.bolt_strength = bpy.context.scene.bolt_settings.bolt_strength
        bolt.bolt_properties.bolt_length = bpy.context.scene.bolt_settings.bolt_length
        bolt.bolt_properties.bolt_opacity = bpy.context.scene.bolt_settings.bolt_opacity
        bolt.bolt_properties.bolt_sharpness_x = bpy.context.scene.bolt_settings.bolt_sharpness_x
        bolt.bolt_properties.bolt_sharpness_y = bpy.context.scene.bolt_settings.bolt_sharpness_y

        return {'FINISHED'}
    #

    def invoke(self, context, event):
        self.execute(context)
        return {'FINISHED'}
    #
#

```

Figura 4.28: código del operador del rayo.

#### 4.3.1.4. Clases contenedoras de propiedades

Las clases contenedoras de propiedades son, como su propio nombre indica, clases que almacenan variables, cuyos valores son los que definen el valor de una determinada pro-

piedad de un objeto que se utilice en el efecto. Estas variables son de un tipo definido por el módulo bpy.props, y dentro de este pueden ser FloatProperty, FloatVectorProperty, BoolProperty, StringProperty, IntProperty, etc. para almacenar un tipo de dato u otro. Además, también permiten establecer valores mínimos y máximos, valor por defecto, descripción de la propiedad, nombre, etc.

Para esta herramienta se han definido dos tipos de clases contenedoras de propiedades: La clase que almacena el valor de la configuración inicial del efecto (con la que este se crea) la cual es introducida en el panel principal de la interfaz, y la clase contenedora de las propiedades de los efectos ya creados, para que puedan ser modificadas las propiedades que se quiera por separado para cada instancia de cada efecto. La necesidad de que esto sea así nace del problema de identificar qué objeto es el que está seleccionado mediante eventos de clic de ratón, saber con qué parámetros fue creado y modificarlos sin que se vean afectados las demás instancias. Además, también existía el problema de saber si un objeto era de un efecto o no. Por ello, se aplicó esta solución que, añadiendo al tipo Object de Blender un atributo que fuese de esta clase, permitiría, para cada instancia, tener un registro de los parámetros con que fue instanciada. Las propiedades de la clase de propiedades suelen ser un subconjunto de las propiedades de configuración, pues como también se mencionó anteriormente, no todas las propiedades con las que se genera el efecto pueden ser modificadas desde el subpanel de propiedades de los objetos generados a posteriori. Las propiedades del objeto y los valores de configuración del objeto se relacionan cuando se genera el efecto, al clicar en el operador, igualando los valores de las propiedades a los valores de las propiedades de la clase de configuración.

```
class BoltPanelSettings(bpy.types.PropertyGroup):
    ''' Custom Settings to create the bolt with an initial configuration

    Attributes:
        bolt_length,
        bolt_opacity,
        bolt_strength,
        bolt_sharpness_x,
        bolt_sharpness_y,
        bolt_branches_dens,
        bolt_branches_depth,
        bolt_max_subbranches,
        bolt_thinness,
        bolt_begin,
        bolt_end,
        bolt_color
    ...

    bolt_length: bpy.props.FloatProperty(
        name = "Length",
        description="indicates the length of the bolt from begining",
        default = 0
    )

    bolt_opacity: bpy.props.FloatProperty(
        name = "Opacity",
        description="Indicates the visibility of the bolt.",
        soft_min = 0,
        soft_max = 1,
        default = 1
    )

    bolt_strength: bpy.props.FloatProperty(
        name = "Strength",
        description = "Indicates how strong is the emission of the the bolt's material.",
        default = 1
    )

    bolt_sharpness_x: bpy.props.FloatProperty(
        name = "Sharpness X",
        description = "Indicates how sharp the irregularities of the bolt surface are on the X axis",
        soft_min = 1,
        default = 1.5
    )
```

Figura 4.29: código de la clase contenedora de la configuración del láser.

```
class BoltObjectProperties(bpy.types.PropertyGroup):
    ''' Custom properties to modify the bolt attributes from Blender interface
    and see how it affects to the bolt.

    Attributes:
        is_bolt,
        bolt_opacity,
        bolt_strength,
        bolt_sharpness_x,
        bolt_sharpness_y,
        bolt_color
    ...

    is_bolt: bpy.props.BoolProperty(
        name="Is bolt",
        description="Indicates if the object is a bolt",
        default=False
    )

    bolt_length: bpy.props.FloatProperty(
        name = "Length",
        description="indicates the length of the bolt from begining",
        soft_min = 0,
        default = 0,
        update = bolt_length_changed
    )

    bolt_opacity: bpy.props.FloatProperty(
        name = "Opacity",
        description="Indicates the visibility of the bolt.",
        soft_min = 0,
        soft_max = 1,
        default = 0,
        update = bolt_opacity_changed
    )

    bolt_strength: bpy.props.FloatProperty(
        name = "Strength",
        description = "Indicates how strong is the emission of the the bolt's material.",
        default = 1,
        update = bolt_strength_changed
    )

    bolt_sharpness_x: bpy.props.FloatProperty(
        name = "Sharpness X",
        description = "Indicates how sharp the irregularities of the bolt surface are on the X axis.")
```

Figura 4.30: código de la clase contenedora de las propiedades del objeto láser.

Las diferencias entre ambas clases contenedoras son que la clase contenedora de las propiedades es un subconjunto de la clase contenedora de la configuración, en algunos casos, como en el efecto de transición de wireframe, tienen el mismo número de atributos las dos, no obstante, la clase contenedora de las propiedades del objeto siempre va a tener un atributo que la clase contenedora de la configuración no va a tener y es un atributo booleano el cual recoge el valor de si el objeto es de un efecto o no, del tipo: `is_bolt`, `is_beam`, etc. La otra diferencia es que la clase que guarda la configuración se guarda en la clase Scene de Blender y la clase contendora de las propiedades de los objetos se guarda en la clase Object, de manera que cada instancia de un objeto que es de tipo Object va a tener en sus atributos una instancia de la clase contenedoras de propiedades, cuyo valor booleano de si es de un efecto o no va a ser True o False si el objeto ha sido generado por alguno de los efectos del add-on o no, respectivamente.

Para terminar de explicar la estructura del addon comentaré las funciones `register` y  `unregister`, las cuales se encargan de registrar y dar de baja, cuando se cierre Blender, las clases que queremos que formen parte del addon. También se crean nuevos atributos para la clase Scene y Object de Blender, las cuales son del tipo: clase contenedora de la configuración y clase contenedora de las propiedades de un objeto de un efecto, respectivamente. La función `register` es la única función llamada cuando se ejecuta el addon, de ahí se visualiza la UI donde se encuentran los demás elementos que permiten llamar a la funciones del código.

```
def register():

    """Register classes and do other necessary tasks when registering the Addon."""
    for cls in classes:
        bpy.utils.register_class(cls)

    bpy.types.Scene.bolt_settings = bpy.props.PointerProperty(type = BoltPanelSettings)
    bpy.types.Object.bolt_properties = bpy.props.PointerProperty(type = BoltObjectProperties)
    bpy.types.Scene.wireframe_settings = bpy.props.PointerProperty(type = WireframePanelSettings)
    bpy.types.Object.wireframe_properties = bpy.props.PointerProperty(type = WireframeObjectProperties)
    bpy.types.Scene.beam_settings = bpy.props.PointerProperty(type = BeamPanelSettings)
    bpy.types.Object.beam_properties = bpy.props.PointerProperty(type = BeamObjectProperties)
    bpy.types.Scene.shield_settings = bpy.props.PointerProperty(type = ShieldPanelSettings)
    bpy.types.Object.shield_properties = bpy.props.PointerProperty(type = ShieldObjectProperties)

    print("\n"                                \n)
    print("VFX Library is a free and open source Add-on following the GNU General Public License.\n")
    print("If you are an addon developer and you can afford to create GNU-GPL addons, do so " +
          "(if you want). Blender has given you a lot for free, give a little bit back " +
          "to the community. :)\n")
    print("\nRafael P.C." \n)
    print("                                \n")
# end register

def unregister():

    """Unregister classes and do other necessary tasks when unregistering
    the Addon.
    """
    for cls in classes:
        bpy.utils.unregister_class(cls)

    del bpy.types.Scene.bolt_settings
    del bpy.types.Object.bolt_properties
    del bpy.types.Scene.wireframe_settings
    del bpy.types.Object.wireframe_properties
    del bpy.types.Scene.beam_settings
    del bpy.types.Object.beam_properties
    del bpy.types.Scene.shield_settings
    del bpy.types.Object.shield_properties
# end unregister

if __name__ == "__main__":
    register()
```

Figura 4.31: código de las funciones `register` y `unregister`.

#### 4.3.2. Implementación efecto de rayo



Figura 4.32: Efecto de Rayo.

El efecto de generación de rayos es el primer efecto en la lista de posibles de esta herramienta. Es un efecto ideado por Joey Carlino, y propietario intelectual del procedimiento para generar este efecto. Este efecto tiene una función principal : generate\_bolt, la cual llama a la función de generar la mesh del rayo, el material y generar sus ramas. Para conseguir este efecto y generar la mesh del rayo se ha de generar un plano y contraerlo en un solo punto. Este punto será el origen del objeto. El punto generado se extrudirá y se harán siete divisiones. Con siete divisiones he considerado que tenía suficiente geometría para que tuviera un buen nivel de detalle con prácticamente cualquier tamaño del rayo, además de que es el número realizado por el autor originalmente.

Posteriormente se selecciona el vértice del origen para crear un grupo de vértices, sólo con este vértice. Este grupo se utiliza para que al aplicar los modificadores de desplazamiento el punto donde acaba el rayo se mantenga intacto. A continuación, se aplican diferentes modificadores de subdivisión y de desplazamiento de la geometría en x e y con una magnitud introducida por el usuario a través de los parámetros de la interfaz, el desplazamiento en z es el desplazamiento vertical en Blender y no se le ha dado, pues hacía parecer al rayo demasiado antinatural. Se añade un último modificador: el "Skin modifier" para crear una cobertura y dar algo de grosor y cuerpo a la geometría y se aplica el sombreado suave para suavizar la forma. Después, se procede al algoritmo de generación de ramas del rayo. Para la generación de las ramas intervienen las variables branches\_depth, que expresa los niveles de subramas; max\_subbranches, que expresa el número máximo de subramas por nivel; el parámetro branches\_dens, que expresa la densidad de ramas e indica la probabilidad de que se alcance el número máximo de subramas por nivel; en otras palabras, expresa cuantas de todas las ramas posibles serán generadas. El proceso de generación de ramas es el siguiente: se selecciona un número de vértices dependiendo de la densidad de ramas introducido, dichos vértices se extruden, haciendo uso del módulo Bmesh de Blender y devuelve un array de vértices resultantes de la extrusión. Dichos vértices devueltos serán los vértices seleccionados para un bucle que se repetirá un número aleatorio de veces en entre 0 y el max\_subbranches, de donde se extrudirán los vértices seleccionados para generar las ramas de ese nivel. Este proceso se repetirá tantas veces como el parámetro branches\_depth marque, es decir, todo este proceso será un bucle que se repetirá branches\_depth veces.

```

verts_idx = [v.index for v in bolt.data.vertices]
verts_extr = verts_idx
for i in range(BRANCHES_DEPTH):
    verts_sel = vertices_select(verts_extr, BRANCHES_DENS, bolt)
    verts_extr = generate_branches(bolt, verts_sel, end)
    if i > 0:
        n = random.randint(0, MAX_SUBBRANCHES)
        for j in range(0,n):
            verts_extr += generate_branches(bolt, verts_sel, end)
    for v in verts_sel:
        bolt.data.vertices[v].select = False

```

Figura 4.33: código del bucle de generación de ramas.

Posteriormente a generar la mesh del rayo, en sus bolt\_properties (el atributo de la clase Object que contiene las propiedades de esa instancia del rayo), se le da el valor True al parámetro is\_bolt, que indica que ese objeto es parte del efecto del rayo y por tanto debe mostrar sus propiedades en el panel de Objetos de la interfaz de Blender. Por último se genera el material del rayo que se lo que permite que se anime dando el efecto de que el rayo cae. El material está constituido por un nodo de coordenadas de textura, un grupo de nodos, el cuál realiza la tarea de hacer que el rayo caiga, y el nodo de salida. El grupo de nodos recibe las coordenadas de objeto del rayo, la longitud, el color, la fuerza de emisión del material y la opacidad; con estos valores hace un comparación del valor de la componente Z (altura) de las coordenadas de objeto recibidas en el grupo con el valor de entrada de la longitud (recordemos que el origen está en la base del rayo, donde cae, por lo que la coordenada más alejada está en la parte superior). Si la comparación devuelve que el valor de la altura es mayor que el de longitud introducido, el rayo visualiza su color hasta esa longitud introducida, de manera que modificando el valor de la longitud desde el panel de propiedades del objeto se puede modificar la longitud del rayo que se ve, haciendo así el efecto de la caída del rayo. La opacidad apaga el rayo o lo enciende y la fuerza (el parámetro strength) indica cómo de fuerte es la emisión del material. Todos estos parámetros se pueden animar mediante los operadores de keyframes del panel de propiedades.

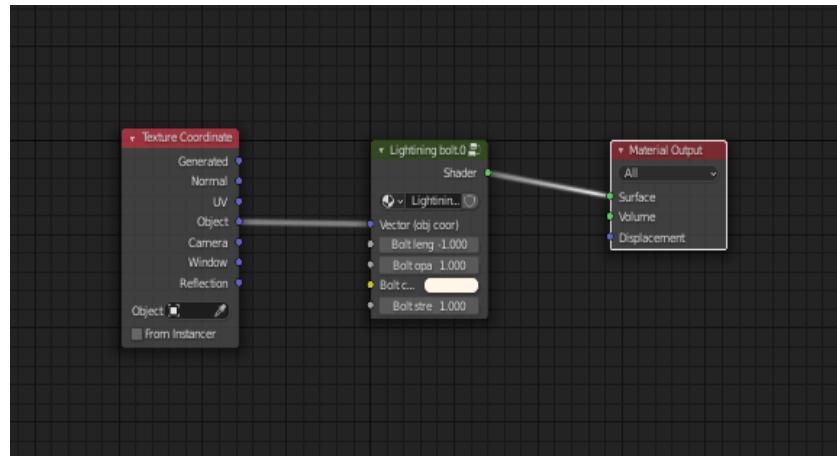


Figura 4.34: Grafo del material del rayo.

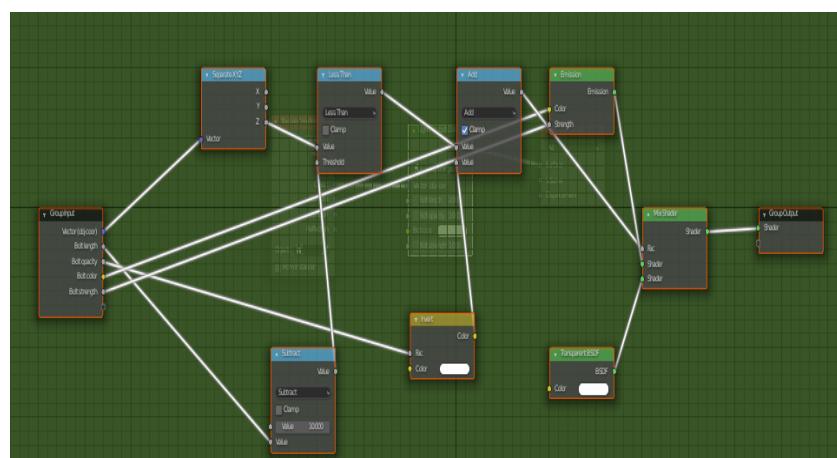


Figura 4.35: Composición del grupo de nodos del material.

Para la inserción y borrado de keyframes se implementan dos funciones, las cuales permiten animar las propiedades del objeto. Estas funciones son similares para todos los objetos por lo que sólo mostraré las de este efecto. Son las funciones: bolt\_insert\_keyframes y bolt\_delete\_keyframes. Estas funciones en función de la propiedad que se pretenda animar insertarán los keyframes de una manera u otra.

```

def bolt_insert_keyframe(context):
    ''' Function that inserts a keyframe on the properties that have been modified
        at current frame
    '''
    bolt_props = ["Bolt length", "Bolt strength", "Bolt color", "Bolt opacity", "Bolt sharpness y",
    for prop in bolt_props:
        if(prop != "Bolt sharpness x" and prop != "Bolt sharpness y"):
            frm = context.scene.frame_current
            mat = bpy.context.object.data.materials[0]
            path = 'nodes["Group"].inputs["'+prop+'"].default_value'
            mat.node_tree.keyframe_insert(path, frame=frm)
        elif (prop == "Bolt sharpness x"):
            context.object.modifiers['DisplaceX'].keyframe_insert("strength", frame=frm)
        elif (prop == "Bolt sharpness y"):
            context.object.modifiers['DisplaceY'].keyframe_insert("strength", frame=frm)
        #
    #
#end insert_keyframe

def bolt_delete_keyframe(context):
    ''' Function that deletes the keyframes on the properties that have been keyframed.
    If there are not keyframes it has not any effect.
    '''

    bolt_props = ["Bolt length", "Bolt strength", "Bolt color", "Bolt opacity", "Bolt sharpness y",
    for prop in bolt_props:
        if(prop != "Bolt sharpness x" and prop != "Bolt sharpness y"):
            frm = context.scene.frame_current
            mat = bpy.context.object.data.materials[0]
            path = 'nodes["Group"].inputs["'+prop+'"].default_value'
            mat.node_tree.keyframe_delete(path, frame=frm)
        elif (prop == "Bolt sharpness x"):
            context.object.modifiers['DisplaceX'].keyframe_delete("strength", frame=frm)
        elif (prop == "Bolt sharpness y"):
            context.object.modifiers['DisplaceY'].keyframe_delete("strength", frame=frm)

    #
#end delete keyframe

```

Figura 4.36: Funciones de inserción y borrado de keyframes.

#### 4.3.3. Implementación efecto de transición de wireframe

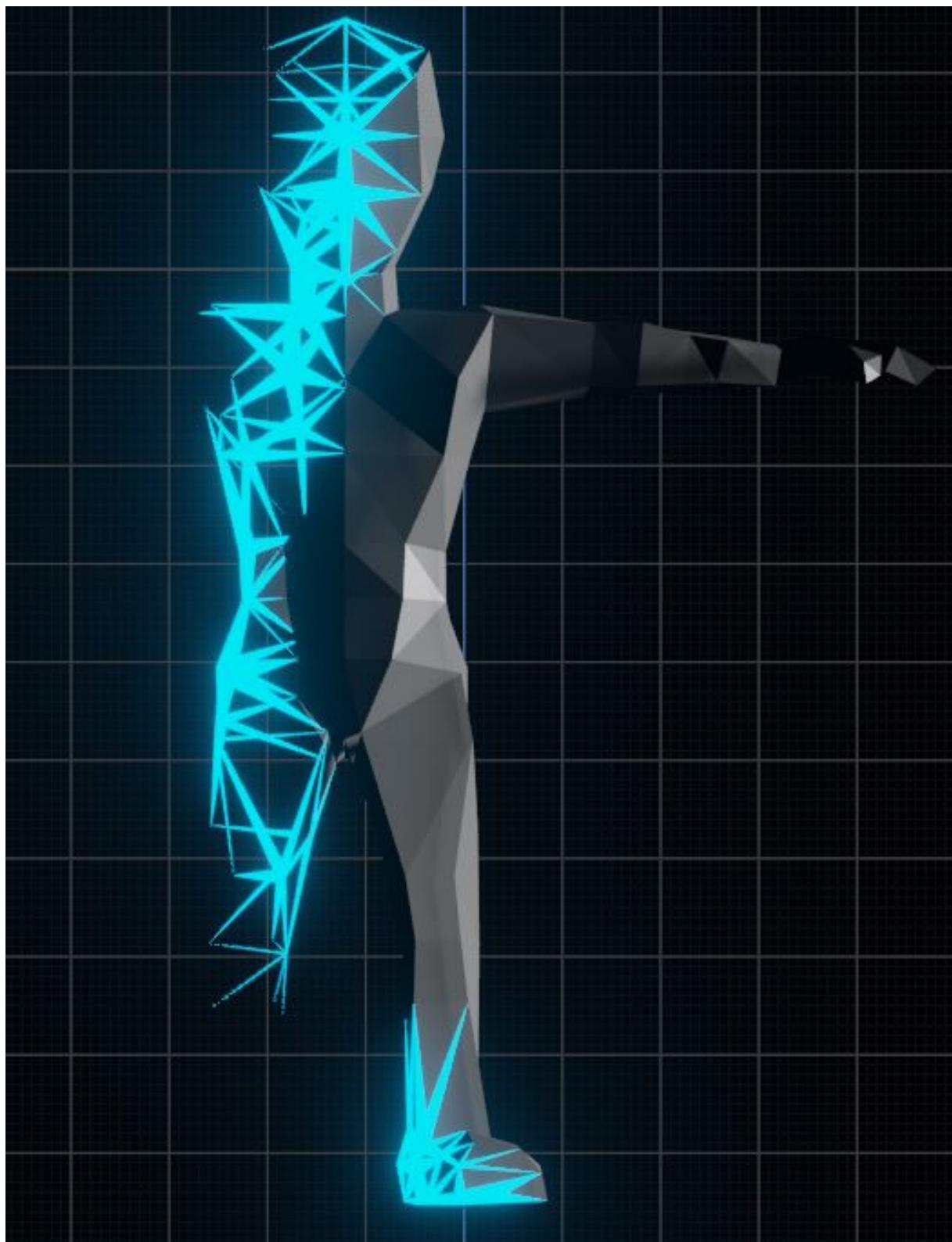


Figura 4.37: Efecto de Transición de Wireframe.

Este efecto es el segundo en la lista de posibles para los desarrollados hasta ahora y permite hacer que un objeto desaparezca haciendo una transición a hilo de alambre con un material emisor. Es un efecto ideado por Steve Lund, o como es conocido en redes sociales CG Geek, quien es propietario intelectual del procedimiento para generar este efecto.

Para la creación de este efecto ha sido necesario aplicar diferentes modificadores al objeto seleccionado al cual se le quiere aplicar el efecto. En primer lugar, se han de seleccionar, en modo edición, todos los vértices del objeto y a continuación se crea un grupo de vértices. Posteriormente se crea un manejador del efecto, en este caso se ha optado por una icoesfera, a la cual se le ha eliminado la mitad para evitar tener geometría innecesaria. Una vez se tiene el grupo de vértices creado y el manejador se aplica al objeto un modificador de "VertexWeightProximity" se le asigna el grupo de vértices creado y el manejador: esto hará que varíe el peso de los vértices del objeto según la distancia al manejador, lo cual tendrá un papel clave, como explicaré a continuación. Se ha de configurar este modificador de manera que los vértices incrementen su peso cuando el manejador se acerque. Una vez tenemos el primer modificador aplicado podemos aplicar el segundo modificador: el modificador "Wireframe", el cual genera un engrosamiento del borde de los triángulos que forman la mesh del objeto creando un hilo de alambre del objeto con un grosor especificado en el modificador. Esto, junto con el modificador de peso por proximidad, hace que los vértices, cuando se acerca el manejador, aparezcan y, al estar aplicándose el modificador de Wireframe, hace que aparezca el wireframe generado.

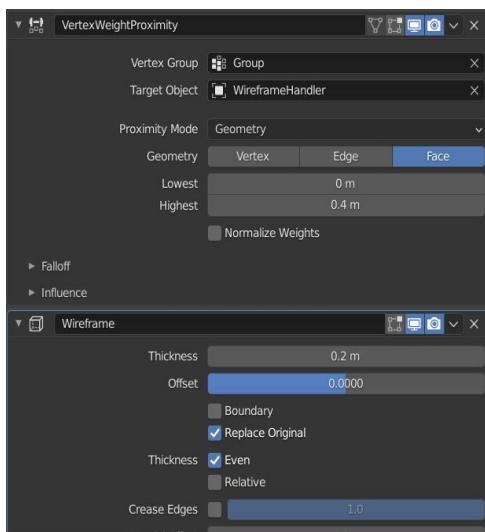


Figura 4.38: Modificadores del objeto wireframe.

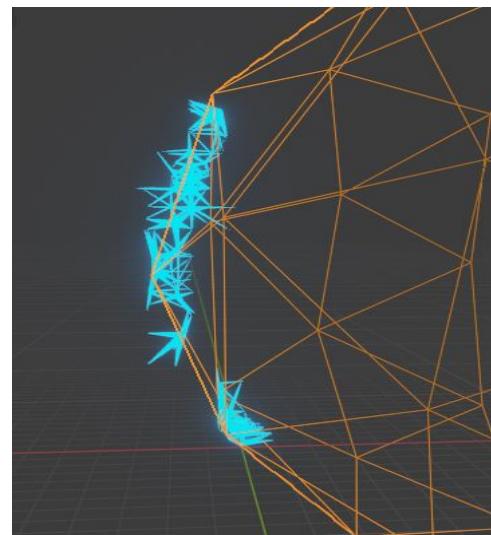


Figura 4.39: Aparición del wireframe con el manejador.

Una vez tenemos, el proceso de aparición de wireframe hay que hacer que se combine con el objeto original para crear el efecto de transición. Para ello se duplica el objeto actual y se elimina el modificador de Wireframe, sustituyéndolo por otro modificador: el modificador "Mask", el cual aplica una máscara que hace que el objeto sea visible o no. Este modificador también se va a ver afectado por el modificador VertexWeightProximity, por tanto seguirá la misma lógica que el método de aparición del wireframe; ahora, cuando se acerque el manejador, ha de desaparecer el objeto, es decir, se ha de aplicar la máscara. Esto se hará creando otro manejador para el objeto duplicado. Se creará un manejador del objeto visible, el cual sustituirá al manejador que aparece en el modificador de VertexWeightProximity del objeto duplicado, resultado precisamente de haber sido duplicado

del anterior.

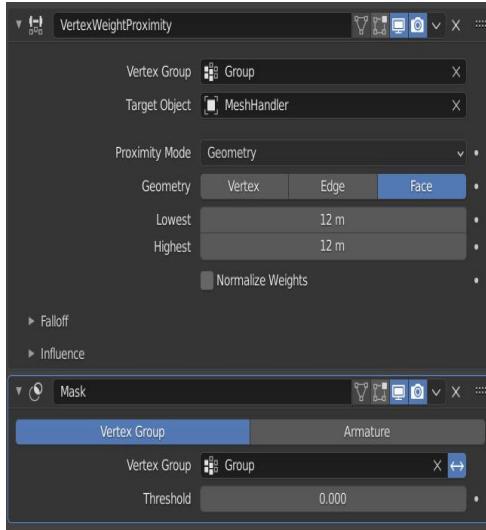


Figura 4.40: Modificadores del objeto visible.

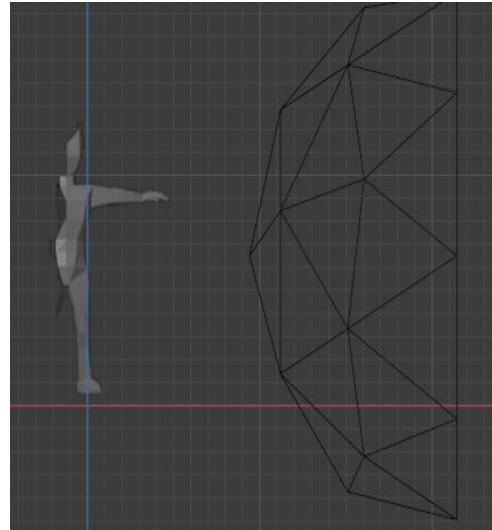


Figura 4.41: Aplicación de la máscara con el manejador.

Un dato importante a destacar es que la zona que a la que afecta el modificador de peso por proximidad está definida con unos parámetros del modificador: el “lowest” “highest”, que indican cuál es la distancia de menor aplicación y de mayor aplicación del modificador, en otras palabras, si el lowest es un valor pequeño y menor que el highest quiere decir que la zona donde menos peso van a tener los vértices y, por tanto, donde menos se vana a ver, será la que se encuentre a esa distancia “lowest” del manejador, y la que más se verá será la que esté a la distancia “highest” del manejador. Se decidió usar dos manejadores por temas de logística ya que con un solo manejador no daba los resultados deseados aún ajustando estos parámetros lo máximo posible.

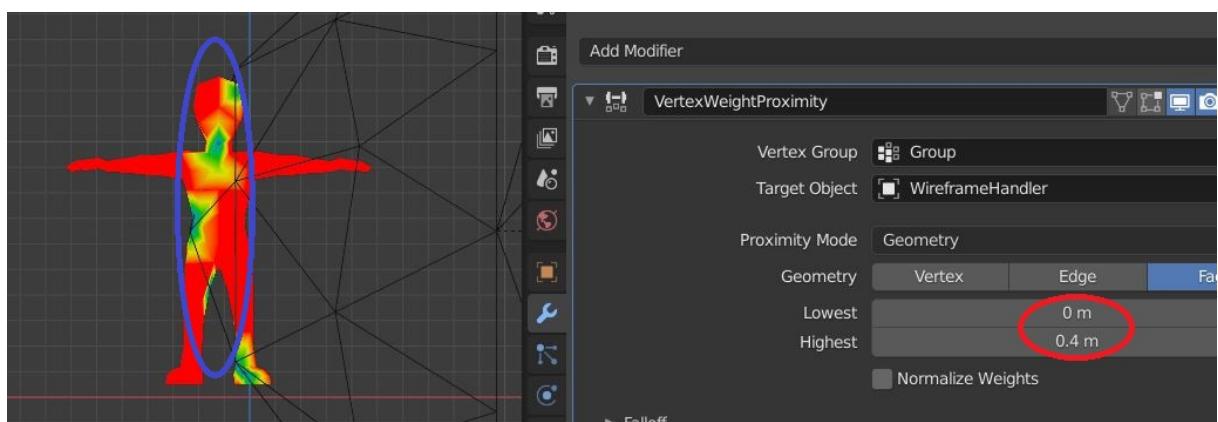


Figura 4.42: Pesos de los vértices por proximidad al manejador

La diferencia entre el valor lowest y highest para el modificador del objeto wireframe es introducido por el usuario a través del parámetro `wire_width`. Los valores en el modificador del objeto de la máscara son los mismos, pues así no hay transición y el paso de la zona enmascarada a la que se ve se hace de manera totalmente inmediata, sin zonas semitransparentes ni nada por el estilo. Ambos objetos, el original con el modificador de

wireframe y el duplicado con el de la máscara, están en la misma posición y por tanto superpuestos. Los manejadores se emparentan para que cuando se mueva el modificador de wireframe lo haga también el de la máscara y así se realice la transición, haciendo aparecer el wireframe y desaparecer el objeto con la máscara al mismo tiempo. Este movimiento de los manejadores es el que el usuario podrá animar directamente desde la interfaz de Blender como cualquier otra animación de traslación para animar el efecto. La distancia de separación entre manejadores depende del tamaño del objeto. Para calcularlo se extraen las dimensiones del objeto a través de la caja que lo envuelve, a la que se puede acceder con la API de Blender, y esta separación es la que tendrá el manejador de wireframe por delante del manejador de la máscara. Esto es porque la distancia lowest del modificador de proximidad del objeto wireframe es 0, por lo tanto, el wireframe empezará a aparecer justo donde el manejador contacte con el objeto hasta donde determine el parámetro highest (que ha introducido el usuario a través del parámetro wire\_width), por tanto, sólo se deberá ver el objeto visible, una vez el modificador de wireframe haya recorrido todo el objeto, es decir, lo haya recorrido a lo ancho, por lo que deberá empezar a hacer efecto cuando la distancia entre el objeto enmascarado y el manejador de la máscara sea de la anchura del objeto. Aquí parte del código donde se han implementado estas operaciones.

```
# Add modifiers and custom settings
mesh_obj.modifiers.remove(mesh_obj.modifiers.get("Wireframe"))
mesh_obj.modifiers.new(name = "Mask", type='MASK')
maskModif = mesh_obj.modifiers["Mask"]
maskModif.vertex_group = "Group"
maskModif.invert_vertex_group = True
VWPModif = mesh_obj.modifiers["VertexWeightProximity"]
VWPModif.min_dist = max - WIREFRAME_WIDTH
VWPModif.max_dist = max
VWPModif.target = mesh_handler
```

Figura 4.43: Código para modificador de máscara.

```
# Add the modifiers to our character and custom the settings
wireframe_obj.modifiers.new(name = "VertexWeightProximity", type='VWPModif')
VWPModif = wireframe_obj.modifiers["VertexWeightProximity"]
VWPModif.vertex_group = "Group"
VWPModif.target = wire_handler
VWPModif.proximity_mode = 'GEOMETRY'
VWPModif.proximity_geometry = {'FACE'}
VWPModif.min_dist = 0
VWPModif.max_dist = WIREFRAME_WIDTH
wireframe_obj.modifiers.new(name = "Wireframe", type='WIREFRAME')
wireModif = wireframe_obj.modifiers["Wireframe"]
wireModif.thickness = WIREFRAME_THICKNESS
wireModif.vertex_group = "Group"
wireModif.invert_vertex_group = True
```

Figura 4.44: Código para modificador de wireframe.

La variable max que aparece en algunas imágenes es el valor de la dimensión máxima, o en otras palabras, del lado mayor de la caja que envuelve al objeto, de ahí el nombre de max. Esto es así para que la transición se pueda realizar en cualquiera de las direcciones.

```
mesh_handler.parent = wire_handler
handlers_offset = Vector((0,0,max))
mesh_handler.location = (handlers_offset)
wire_handler.rotation_euler = (0,PI/2,0)

wire_handler.location = wireframe_obj.location.copy()
wire_handler.location[2] += wireframe_obj.bound_box.data.dimensions[2]/2
```

Figura 4.45: Posicionamiento de los manejadores

El material que se aplica al objeto wireframe es simplemente un material de emisión, con un color introducido por el usuario a través del panel. El grosor del wireframe también lo introduce el usuario por el panel. En cuanto a las propiedades que se pueden modificar de este efecto a través del panel de propiedades del objeto, se pueden modificar todos los que aparecen en el panel de configuración.

#### 4.3.3.1. Implementación del efecto rayo láser

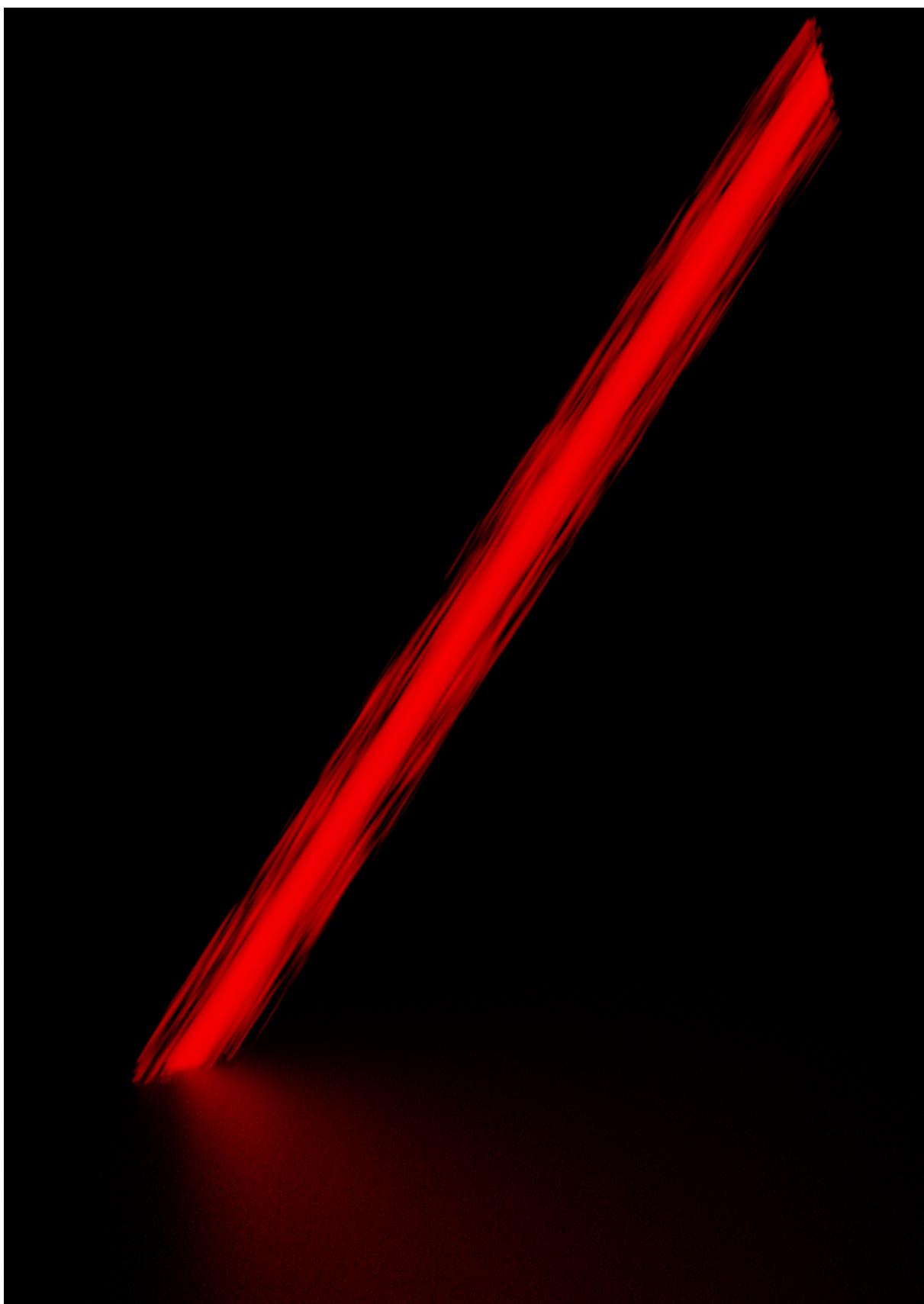


Figura 4.46: Efecto de Rayo Láser

Este efecto es el tercero de la lista de los posibles efectos desarrollados hasta ahora y es un efecto ideado por Joel Adams, quien posee un canal en YouTube llamado “Iridesiumz” es el autor intelectual del procedimiento de generación de este efecto. El efecto de rayo láser te da dos opciones de generación de láseres: una para el rendizador de Cycles y otra para el de Eevee.

Para generar el láser para Cycles se han de introducir color, fuerza de emisión del material, radio del láser, posición de origen y fin del láser, ruido del material y como parámetro opcional, una velocidad de animación si queremos que el láser parezca que está siendo disparado continuamente animando el material. Una vez introducidos estos parámetros (siempre tendrán valor, ya que aunque el usuario no introduzca uno, tienen configurados unos valores por defecto, como pasa con los parámetros del resto de efectos) se genera un cubo, el cual va a ser el cuerpo del láser. Se calcula la longitud del vector origen-fin y se escala la mitad de esa magnitud en el eje horizontal, pues se escala en ambos sentidos. Posteriormente se rota, calculando la diferencia del ángulo entre el vector origen-fin y el vector (1,0,0), dado que se toma el eje horizontal para hacer el escalado, el vector que indica la dirección en la que apunta el láser hasta este momento es el horizontal, es decir, el (1,0,0). Una vez calculada la diferencia de rotación con el método `rotation_difference` de la clase Vector de la librería mathutils, el cual devuelve un quaternion, se aplica esta rotación al cubo escalado para que quede correctamente orientado. Una vez tenemos la magnitud y orientación del láser correctos pasamos a posicionarlo para que su origen y fin se correspondan con los introducidos en el panel. Esto se hace calculando la distancia media desde el origen al final para cada coordenada, una vez calculadas se igualan a la localización del láser.

```
bpy.ops.mesh.primitive_cube_add()
beam = bpy.context.object
beam.beam_properties.is_beam = True
beam.beam_properties.beam_anim = beam_settings.beam_anim

width = (BEAM_END[0] + BEAM_BEG[0]) * 0.5
height = (BEAM_END[1] + BEAM_BEG[1]) * 0.5
depth = (BEAM_BEG[2] + BEAM_END[2]) * 0.5

material = cycles_beam_material(beam)
beam.active_material = material

beam.scale = Vector((BEAM_LENGTH/2, BEAM_RADIUS/2, BEAM_RADIUS/2))

direction = BEAM_END - BEAM_BEG
q = Vector((1,0,0)).rotation_difference(direction)

beam.rotation_mode = 'QUATERNION'
beam.rotation_quaternion = q
beam.location = Vector((width, height, depth))
```

Figura 4.47: Transformaciones del objeto láser

La parte que da aspecto de láser a este objeto es su material, el cual es construido mediante una composición de diferentes nodos. El material es un material volumétrico, el cual utiliza las coordenadas del objeto, al igual que el efecto de rayo. Se genera en el

interior del rectángulo, resultado de las transformaciones del cubo inicial, gracias a un gradiente esférico que tiene su valor máximo en el origen del objeto, por tanto esto hará que se genere un degradado de blanco a negro con forma cilíndrica en el interior del rectángulo, lo que generará la forma propia del láser. Después se añade un conjunto de nodos para generar el ruido, el cual hará que el láser tenga más disperso o menos en función de un parámetro entre 0 y 1 y que el usuario podrá introducir por el panel: este es el factor `laser_noise`. Este conjunto de nodos contiene un nodo de mapeando, el cual mapea las coordenadas del objeto a las coordenadas de ruido. Este nodo se puede animar modificando sus inputs de localización, desplazando en cualquier dirección las coordenadas del objeto e insertando un keyframe. Añadiendo un modificador a su curva de animación que genera una animación continua, se puede hacer que se desplace continuamente la textura animando el láser.

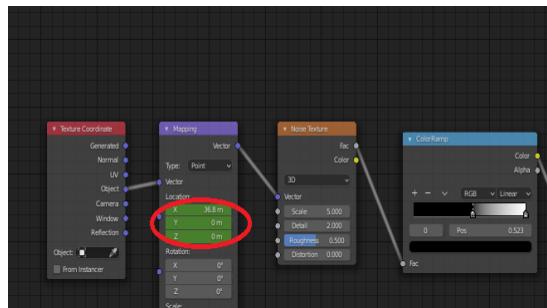


Figura 4.48: Nodo de mapeado con animación

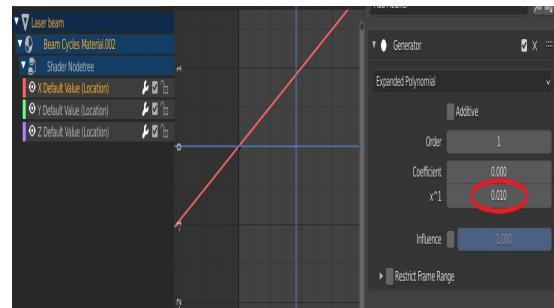


Figura 4.49: Modificador de animación continua del material

Por último se combinan los nodos que generan el gradiente cilíndrico y el ruido con otro nodo que regula la anchura del láser y se combina finalmente con el nodo que le da el color y la emisión, multiplicando este factor de emisión por el valor del parámetro `laser_strength`.

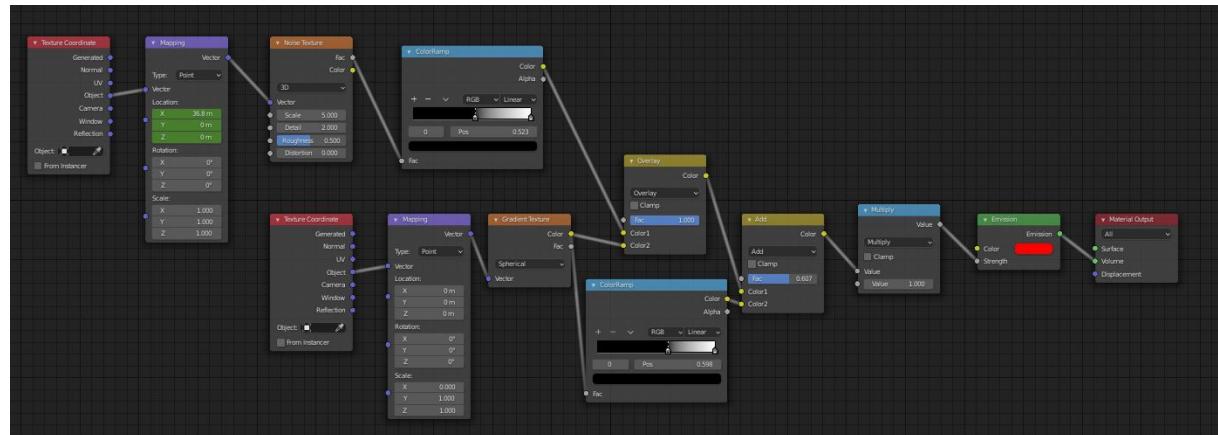


Figura 4.50: Material del láser

Para hacer el láser para Eevee el proceso es mucho más simple. Simplemente se crea un círculo que se extrude, tal como el láser para Cycles, se rota de la misma manera y se posiciona de la misma manera. Al ser en Eevee, que es un renderizador, para tiempo real, el material omite toda la parte del ruido y la animación y se centra en la emisión,

creando simplemente el material emisor del color determinado por el usuario y activando la opción “Bloom.” en el panel de renderización. A pesar, de la simplicidad de este segundo método el propósito, como se dijo en los requisitos, es que los efectos sean lo más usables posible y hacer una opción únicamente para Cycles me parecía poco coherente con este requisito.

#### 4.3.3.2. Efecto escudo de fuerza

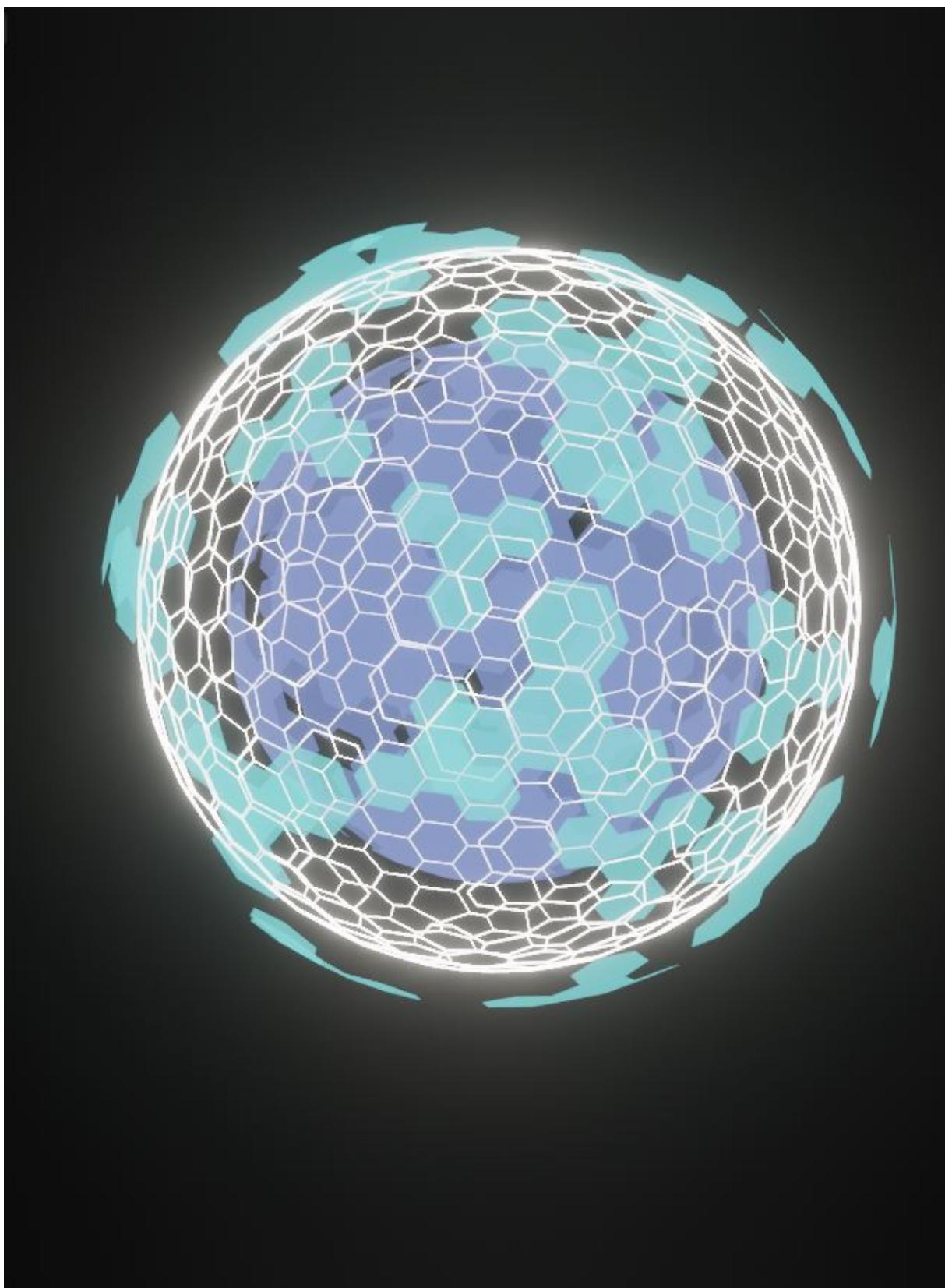


Figura 4.51: Efecto Escudo de Fuerza

Este es el último efecto de la lista de actuales desarrollados para el add-on el cual genera un escudo de fuerza animado por defecto y un efecto original de Derek Elliott, quien es el autor intelectual del procedimiento seguido para generar este efecto. Este efecto en concreto es más particular que los otros ya que no genera un solo objeto sino tres. Un escudo con el modificador de Wireframe, el cual expliqué en la descripción Efecto de transición de wireframe, otro escudo principal, el cual es el único no omitible de este efecto, y otro escudo opcional formado por partes del principal. Como he mencionado, sólo uno de ellos no es opcional, los otros se podrán generar o no dependiendo de si se marca la casilla de su uso en el panel de configuración.

Para generar el escudo principal creamos una icoesfera y le aplicamos el modificador "Bevel", el cual es un modificador biselador de los bordes de la malla con un cierto control del mismo gracias al parámetro "Amount". Una vez hecho esto, y si está activada la opción del escudo de wireframe, duplicamos el objeto con el modificador y le aplicamos el modificador Wirefrmae al objeto duplicado. Este será el escudo de wireframe, el cual tendrá un material de emisión simple.

```
# Create the wireframe estructure of the shield
FSwire = None
if(USE_WIREFRAME):
    FSwire = bpy.ops.mesh.primitive_ico_sphere_add(subdivisions=3, radius=4)
    FSwire = context.object
    bpy.context.object.name = "Shield wire"
    FSwire.modifiers.new(name = "Bevel", type='BEVEL')
    bevel = FSwire.modifiers['Bevel'].affect = 'VERTICES'
    bevel = FSwire.modifiers['Bevel'].width = BEVEL_WIDTH
    FSwire.modifiers.new(name = "Wireframe", type='WIREFRAME')
    wireframe = FSwire.modifiers['Wireframe'].thickness = WIREFRAME_THICKNESS
    FSwire.modifiers.new(name = "Build", type='BUILD')
    build = FSwire.modifiers['Build']
    build.frame_start = WIREFRAME_BUILD_START
    build.frame_duration = WIREFRAME_BUILD_DURATION

# Creating a new material for the Wireframe
material = create_wireframe_material()
FSwire.active_material = material

FSwire.shield_properties.is_wire_shield = True
#
```

Figura 4.52: Código de generación del escudo de wireframe.

Una vez hecho esto se le aplica un material al escudo principal, el cual podrá estar animado si se activa esta opción en el panel de configuración, al igual que con la generación de los otros dos escudos.



Figura 4.53: Checkboxes de uso de escudos.  
Figura 4.54: Opción de animación del material del escudo.

El material del escudo principal, si no lleva animación, será un material emisor al uso, como en el caso del rayo láser para Eevee. Si sí lleva animación, se generará un mapeado de las coordenadas del objeto a una textura de ruido y se animará insertando un keyframe sobre uno de los inputs de "location" del nodo de mapeado, y con un modificador sobre su curva de animación para hacer que la animación sea continua, al igual que se hizo con el material del rayo láser para Cycles. En este caso, tanto el ruido como la animación son factibles en Eevee y Cycles, por lo que no hay que elegir el renderizador que se va a usar para animar o no el escudo, a diferencia del láser. Los parámetros para este material animado serán la cantidad de ruido y la velocidad de animación. El color será una propiedad, que se podrá variar además, tanto si está animado como si no, así como la fuerza de emisión del material.

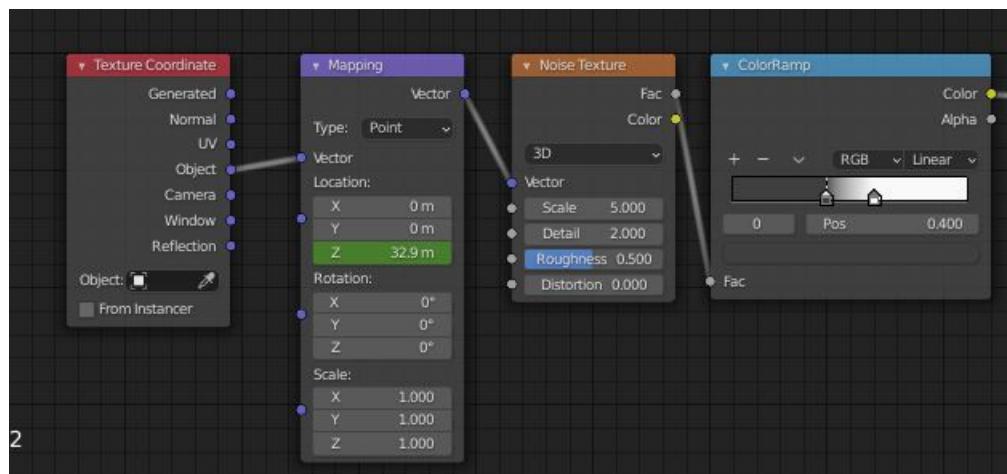


Figura 4.55: Nodos de generación del ruido del material del escudo.

Una vez tenemos generados los dos escudos anteriores, el de wireframe y el principal,

se hará una selección de caras de la mesh del escudo principal y se separarán de este, creando el escudo de fragmentos. Una vez separado se independizará su material y su animación de las del escudo principal, pudiendo así modificar el color y la emisión de los fragmentos de manera independiente y diferenciando su material del del escudo principal.

```
FS2 = None
if(USE_FRAGMENTOS):
    for obj in context.selected_objects:
        obj.select_set(False)

    FS.data.polygons[101].select = True
    bpy.ops.object.editmode_toggle()
    bpy.context.tool_settings.mesh_select_mode = (False, False, True)
    bpy.ops.mesh.select_similar(type='PERIMETER', threshold = 0.1)
    #Separate the selected faces
    bpy.ops.mesh.separate(type='SELECTED')
    bpy.ops.object.editmode_toggle()
    FS2 = context.selected_objects[0]
    FS2.name = 'Shield frags'
    bpy.ops.object.make_single_user(object=False, obdata=False, material=True, animation=False)
    FS2.active_material.node_tree.nodes['Emission'].inputs[0].default_value = FRAGMENTS_EMISSION
    FS2.active_material.node_tree.nodes['Emission'].inputs[1].default_value = FRAGMENTS_EMISSION
    FS2.shield_properties.is_frag_shield = True
#
#
```

Figura 4.56: Código de generación del escudo de fragmentos.

Una vez creados los escudos se generan sus animaciones. Estas animaciones son animaciones de su escala, la cual se keyframes a 0 al principio y a 1 al final, en un intervalo de 50 frames, este intervalo se da por defecto, pudiendo el usuario desplazar los keyframes por la línea de tiempo manualmente y separalos a su antojo. Las animaciones de los tres escudos aparecen con un retardo por defecto de 20 frames, pero en los parámetros del panel de propiedades de los objetos, aparece el parámetro animation \_delay, el cual permite modificar este retardo o anularlo por completo haciendo que los tres escudos emergan a la vez. Cada escudo tiene sus paneles de propiedades independientes con sus propiedades correspondientes, los cuales aparecerán cuando se seleccionen.

```
#Keyframe insertion and keyframe interpolation mode change
for i in range(num_curves):
    delay = i * ANIM_DELAY
    shield = shields[i]
    if(shield is not None):
        shield.scale = (0,0,0)
        shield.keyframe_insert(data_path="scale", frame=1 + delay)
        shield.scale = (1,1,1)
        shield.keyframe_insert(data_path="scale", frame=50 + delay)
        fcurves = shield.animation_data.action.fcurves
        for fcurve in fcurves:
            for kf in fcurve.keyframe_points:
                kf.interpolation = TYPE_INTERPOLATION
#
#
```

Figura 4.57: Código de animación del escudo.



# Capítulo 5

## Pruebas

### 5.1. Pruebas

Para comprobar el funcionamiento y desempeño de la herramienta se han realizado tres tipos de pruebas: funcionales, de rendimiento y de usabilidad.

En el primer tipo de pruebas se realiza un seguimiento de los resultados obtenidos con los casos de uso para comprobar que los resultados obtenidos son los correctos. En el segundo se comprueba el coste temporal de los casos de uso para analizar su rendimiento y en el tercero se comprueba cómo de usable es por usuarios que utilizarán la herramienta.

#### 5.1.1. Pruebas funcionales

En este apartado describiré la funcionalidad de cada uno de los efectos individualmente.

##### 5.1.1.1. Pruebas funcionales para el efecto de rayo

Para el rayo las pruebas funcionales han consistido básicamente en probar los diferentes parámetros y comprobar sus resultados. Las pruebas han sido realizadas principalmente en el equipo con el que se ha desarrollado el add-on, habiendo sido probados por otros usuarios con otros equipos de capacidades similares, inferiores y superiores. A continuación, se muestran algunos de los resultados obtenidos con los diferentes parámetros en el equipo en el que se ha desarrollado el add-on, el mío, cuyas características ya se definieron en el apartado de especificaciones hardware:

**5.1.1.1.1. Parámetro Strength** El parámetro strength es el parámetro que determina la luminosidad del rayo. Puesto que cuando un relámpago cae puede aumentar muchísimo la luz del cielo y del entorno donde cae, es uno de los parámetros más importantes para que en una animación o un vídeo al que se incorpore este efecto quede realista, debiendo poder hacer que se ilumine toda la escena si así fuera necesario. El material del rayo está creado con un nodo shader de emisión, por lo que se probaron valores altos del factor strength de ese nodo para comprobar que a valores altos de este se conseguían los

resultados esperados. El parámetro strength, tanto del panel de configuración del objeto como del panel de propiedades del mismo actúan sobre el nodo del shader de emisión en su factor strength. He aquí algunas muestras de valores altos, medios y bajos del valor.



Figura 5.1: Strength = 50



Figura 5.2: Strength = 100



Figura 5.3: Strength = 500



Figura 5.4: Strength = 1000

Cuadro 5.1: Imágenes de los resultados según el valor del parámetro strength

Cabe destacar que aunque con el valor 1000 el resultado ya es aceptable se pueden llegar a valores altísimos, pudiendo llegar a sobrepasar valores de  $10^{15}$ . Esto confirma que la escena se va a poder iluminar con la luminosidad del material del rayo.

**5.1.1.1.2. Parámetro Thiness** El parámetro thiness es el parámetro que determina la delgadez del rayo, es decir, cómo es de fino. El rayo se hará más fino en la zona donde impacta y se ensanchará más según se aproxima al comienzo del rayo, siendo el comienzo la parte más ancha si el valor es mayor que 0. El valor 0 hace que el rayo sea muy grueso y por igual por todo. El grosor del rayo también será relativo a la altura del rayo. Cuanto más alto sea el rayo menor será el impacto de un valor de thiness pequeño, o dicho de otra manera, menor se verá afectado el rayo si tiene un grosor alto cuanto más alto sea, ya que el grosor tiene un máximo, que es el que el modificador "Skin" permite, que le da el "cuerpo" al rayo, y no se puede modificar, es fijo, y el efecto de grosor será menor ya que la relación grosor/altura será menor. Si se quiere tener un rayo grueso y de gran altura se deberá generar un rayo con una altura menor y escalarlo. Ya que el origen del rayo está donde el rayo impacta, al escalarlo se escalará hacia arriba, pues se escala en la dirección opuesta a la indicada por el vector que va desde el borde del objeto hacia el centro del mismo. Aquí se muestran ejemplo de diferentes valores para el thiness de un rayo con una altura de diez metros y una altura de 100 metros.



Figura 5.5: Thiness = 0, Altura = 10



Figura 5.6: Thiness = 1, Altura = 10



Figura 5.7: Thiness = 2, Altura = 10

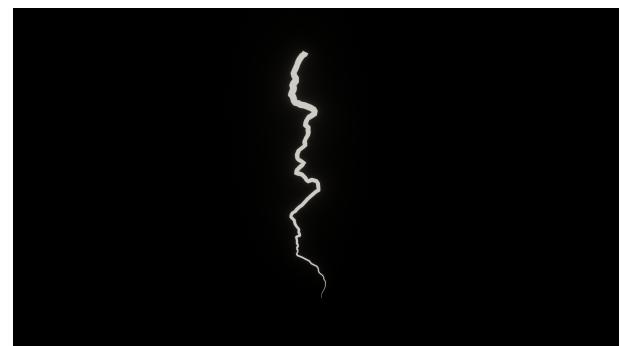


Figura 5.8: Thiness = 4, Altura = 10

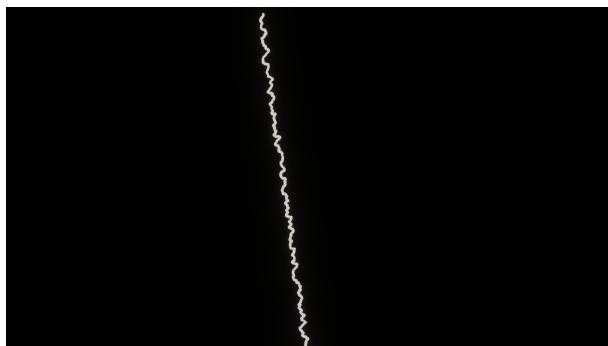


Figura 5.9: Thiness = 0, Altura = 100



Figura 5.10: Thiness = 1, Altura = 100



Figura 5.11: Thiness = 2, Altura = 100

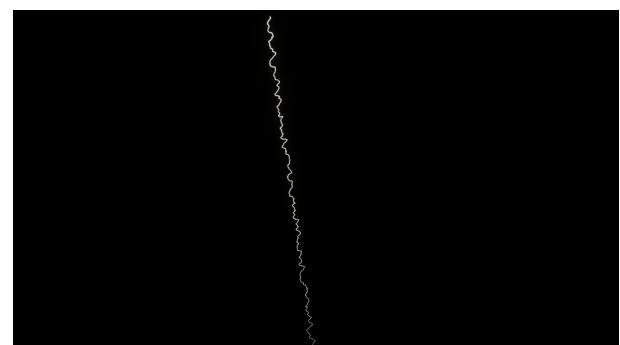


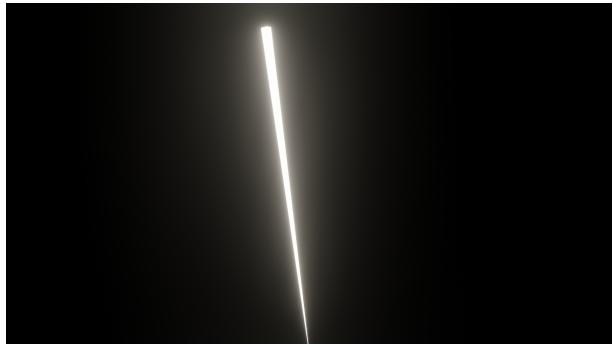
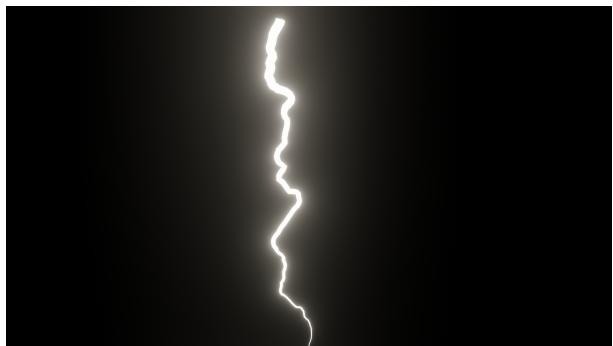
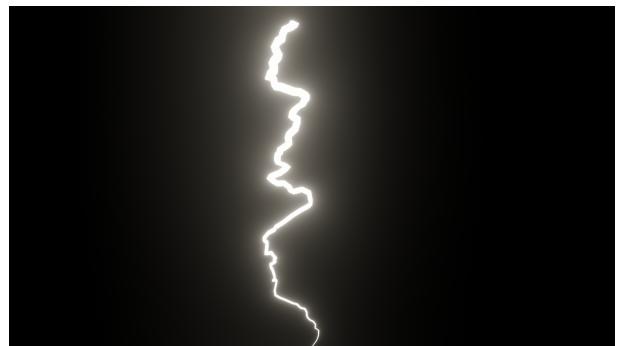
Figura 5.12: Thiness = 4, Altura = 100

Cuadro 5.2: Resultados según el valor del parámetro thiness y la altrua

Como se puede comprobar, el resultado es más notorio en el rayo de menor altura.

**5.1.1.3. Parámetros Sharpness X/Y/Z** Estos parámetros son los que le dan la forma en zigzag al rayo. Actúan sobre los modificadores de desplazamiento de la geometría del rayo y permiten cambiar su valor desde el panel de propiedades del objeto una vez creado el rayo. De nuevo, al igual que pasaba con el parámetro thickness, dependiendo de la altura del rayo este será más notorio o menos. No tiene una limitación como el parámetro thickness en el sentido de la cantidad de desplazamiento, pues pueden ser cantidades muy grandes, pero cuanto mayores sean las cantidades más antinatural se verá la forma del rayo. A este problema la solución es la misma que con el thickness: crear un rayo a menor escala con un valor de sharpness en Z, X e Y que de el resultado deseado en cuanto a la forma y escalarlo para que el rayo tenga la altura deseada. El parámetro sharpness X controla las irregularidades en el eje x y los parámetros sharpness z e y en sus respectivos ejes. El parámetro de sharpness z es especial ya que no se recomienda aumentar su valor por encima de 1 pues da resultados antinaturales, aún así se ha dejado para que el usuario tenga más posibilidades y quede a su elección su uso. A continuación, se muestran ejemplos de diferentes valores para los parámetros sharpness (se han aplicado los mismos valores sobre los parámetros sharpness x e y, únicamente, el parámetro sharpness z ha conservado el valor 1 por defecto excepto en la última imagen).

Las pruebas se han realizado con un rayo de altura 10, por lo que con valores pequeños de sharpness se puede conseguir una forma de rayo realista, no obstante, si queremos reproducir esta forma con un rayo de altura mayor (50 metros o más) quizás se deba mover los vértices de la geometría manualmente para dar la forma curva que se quiera, ya que la forma dada por los modificadores de desplazamiento, dan la forma de zigzag propia de los rayos, pero aparte de esta forma de zigzag, los rayos pueden seguir una trayectoria curva. Cuando el rayo es de poca altura los parámetros sharpness permiten recrear tanto la forma en zigzag como la trayectoria curva, pero a medida que el rayo aumenta de tamaño esto deja de ser así, por lo que se deberá modificar manualmente o generarla en pequeño y escalarla como se mencionó anteriormente. Se ha optado por hacer que sea el usuario quien tenga que modificar los vértices de la geometría debido a que los rayos son fenómenos anárquicos e irregulares cuyo comportamiento a la hora de caer es aleatorio, dependiendo de las condiciones de temperatura, humedad y carga electrostática del medio, y por tanto, no hay ninguna ecuación o patrón que permita definir la trayectoria de un rayo al caer. Debido al inmenso abanico de posibilidades que existe de posibles formas de los rayos se ha dejado a elección del usuario. Además, hacer que se recalcule la posición de cada vértice cuando el rayo es mayor supondría un coste adicional al algoritmo de creación del rayo que, como veremos en el apartado de pruebas de rendimiento, supondría una sobrecarga excesiva de coste temporal, pues a medida que aumenta la altura del rayo lo hace proporcionalmente el número de vértices con lo que empeoraría significativamente el rendimiento, ya que el algoritmo de por sí puede llegar a ser muy costoso.

Figura 5.13: Sharpness  $x/y = 0$ Figura 5.14: Sharpness  $x/y = 1$ Figura 5.15: Sharpness  $x/y = 2$ Figura 5.16: Sharpness  $x/y = 4$ Figura 5.17: Sharpness  $x/y = 10$ Figura 5.18: Sharpness  $z = 2.5$ , Sharpness  $x/y = 1$ 

Cuadro 5.3: Imágenes de los resultados según el valor del parámetro strength

**5.1.1.1.4. Branches Depth, Branches Dens, Max Subbranches** A continuación se explicarán los 3 parámetros fundamentales para la generación de subramas del rayo principal. Este proceso es un proceso iterativo que se repetirá Branches Depth veces como máximo y que dependerá de los parámetros Branches Dens y Max Subbranches.

El parámetro Branches Depth expresa el número de niveles de subrayos que queremos que haya, en otras palabras, expresa el nivel de iteratividad. La explicación de su papel es la siguiente: por cada nivel habrá un número de subrayos de los cuales surgirán nuevos subrayos. Estos nuevos subrayos surgirán en un número NO superior a Max Subbranches, pudiendo ir entre 1 y Max Subbranches. El parámetro Branches Depth determina, en cada iteración, cuantos de los vértices generados generarán nuevas ramas, en otras palabras y dando sentido al nombre del parámetro, expresa la densidad de vértices que serán

generadores de ramas (recordemos que por cada vértice se generarán max subbranches como máximo). Si el parámetro Branches Dens es pequeño puede que dejen de haber niveles de subrayos antes de llegar al número de niveles de Branches Depth, ya que puede por tanto, Branches Depth es un límite máximo de niveles de subramas. El valor de Max Subbranches también determinará el número de niveles que habrá pues, si es pequeño (menor de 0.5), surgirán pocos subrayos por nivel y esto hará que la disminución progresiva del número de rayos que surgen por nivel acabe antes con los niveles por rayo pero sólo si el parámetro Branches Dens es pequeño. Estos han sido algunos resultados:

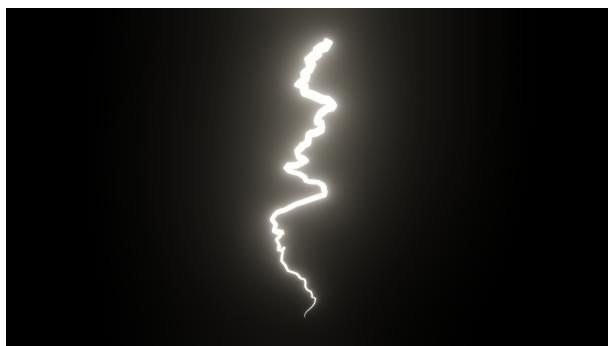


Figura 5.19: Branches Depth = 0, Max Subbranches (no se usa con depth = 0), Branches Dens ((no se usa con depth = 0))



Figura 5.20: Branches Depth = 3, Max Subbranches = 1, Branches Dens = 1



Figura 5.21: Branches Depth = 2, Max Subbranches = 2, Branches Dens = 0.70



Figura 5.22: Branches Depth = 7, Max Subbranches = 2, Branches Dens = 0.48



Figura 5.23: Branches Depth = 7, Max Subbranches = 2, Branches Dens = 0.5



Figura 5.24: Branches Depth = 2, Max Subbranches = 3, Branches Dens = 1

Cuadro 5.4: Imágenes de los resultados según el valor de los parámetros

Como se especifica en la figura 5.19, cuando el parámetro Branches Depth está a 0 no se ejecuta el código de generación de ramas ya que esto significa que no hay subramas. En la figura 5.20 se aprecia que si el nivel máximo de subramas es 1, aunque Branches Depth sea alto<sup>1</sup> las subramas que se generen serán prolongaciones de las que ya habían, por tanto, sólo habrán tantas ramas como vértices se decidan extrudir con el parámetro Branches Dens, que en este caso son todos, excepto los que corresponden al inicio y al fin del rayo, ya que Branches Dens es 1. En la figura 5.21 se aprecia que hay bifurcaciones en las subramas debido a que el parámetro Max Subbranches es 2, no obstante, debido a que Branches Depth vale 2 y la densidad es del 70 %, no se tienen demasiadas ramas, pero podemos observar que, tanto en el ejemplo anterior como en este, se llegan a ejecutar todos los niveles de generación de subramas debido a que la densidad es superior al 50 %, sin embargo, en el ejemplo de la figura 5.22, a pesar de tener un nivel de Branches Depth altísimo y de Max Subbranches medio, no se llegan a completar todas las iteraciones marcadas por Branches Depth pues el número de vértices que se dejan de seleccionar en cada iteración es mayor que el número de vértices nuevos que se generan, aunque la densidad es muy ligeramente inferior al 50 %, del 48 %. En la siguiente figura 5.23 sí se ve que aumentando la densidad hasta el 50 % y con los mismos parámetros que antes se llega a completar el número de iteraciones de máximo. La última imagen 5.24 muestra una combinación con valores altos pero no excesivos, que se podrían considerar "normales".



Figura 5.25: Branches Depth = 2, Max Subbranches = 5, Branches Dens = 1

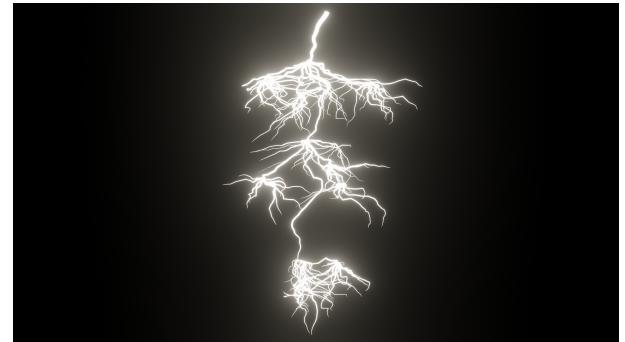


Figura 5.26: Branches Depth = 2, Max Subbranches = 10, Branches Dens = 0.5

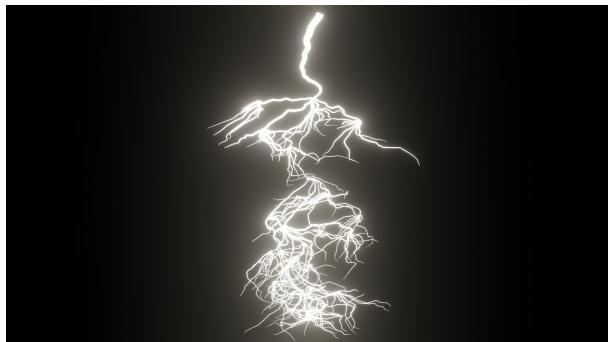


Figura 5.27: Branches Depth = 2, Max Subbranches = 7, Branches Dens = 0.75



Figura 5.28: Branches Depth = 2, Max Subbranches = 6, Branches Dens = 0.75

Cuadro 5.5: Imágenes de los resultados con valores altos de subramas y densidad

<sup>1</sup>Se consideran niveles altos aquellos que incrementen significativamente o puedan incrementar el tiempo de ejecución del algoritmo. Branches Depth: a partir de 2, Max Subbranches a partir de 2, Branches Dens a partir de 0.5

**5.1.1.1.5. Parámetros Color, Begin, End** Estos parámetros se podrían decir que son los menos relevantes para la generación del rayo. El color es cambiabile desde el panel de propiedades del objeto, en el panel de propiedades de los objetos de la interfaz de Blender, y los parámetros begin y end indican donde comienza y donde impacta el rayo relativamente. A continuación, se muestran algunos ejemplos de uso:



Figura 5.29: Begin (4,5,6); End (0,10,-5); Color rojo



Figura 5.30: Begin (0,10,-5); End (4,5,6); Color azul



Figura 5.31: Begin (0,10,-5); End (4,5,-15); Color verde



Figura 5.32: Begin (4,5,-15); End (0,10,-5); Color amarillo

Cuadro 5.6: Imágenes de los resultados con distintos begins, ends y colors

Para el correcto funcionamiento del rayo a la hora de caer se deberá tener en cuenta que el resultado sólo es correcto si el rayo es vertical, tanto si cae de arriba a abajo como si es al revés, pero de izquierda a derecha o viceversa puede no funcionar correctamente si el rayo tiene subrayos. Cuando el rayo es horizontal y tiene subrayos se puede dar el caso de que las puntas de los subrayos, en sus extremos, empiecen a hacerse visibles antes que el rayo principal a la altura a la que ese subrayo nace, por ello, si bien es cierto que esto no tiene porqué pasar a priori, a más número de ramas mayores probabilidades hay de que en alguno de ellos suceda este problema. Por este motivo las pruebas de comprobación como se comporta el rayo según donde empiece y donde acabe han sido necesarias.

### 5.1.1.2. Pruebas funcionales para el efecto de transición wireframe

Para el efecto de transición wireframe, al igual que el rayo, las pruebas funcionales han consistido básicamente en probar los diferentes parámetros y comprobar sus resultados. A continuación, se muestran algunos de los resultados obtenidos con los diferentes parámetros en el equipo en el que se ha desarrollado el add-on, el mío, cuyas características ya se definieron en el apartado de especificaciones hardware:

**5.1.1.2.1. Parámetro Strength** Este parámetro es el que se encarga, al igual que en el caso del rayo, de dar mayor o menor luminosidad al material del wireframe, en el objeto al que se le ha aplicado el efecto. A continuación, se muestran unas imágenes de los resultados obtenidos con los valores más normales y habituales para este parámetros:

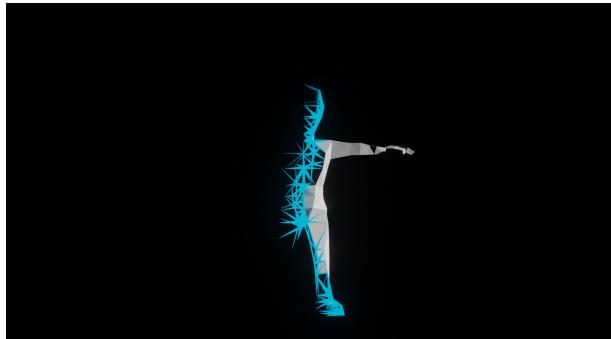


Figura 5.33: Strength = 1

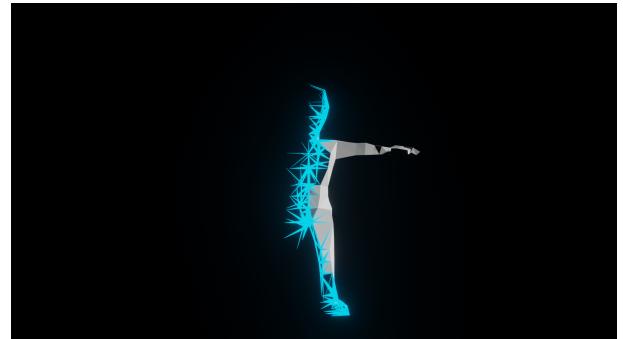


Figura 5.34: Strength = 2

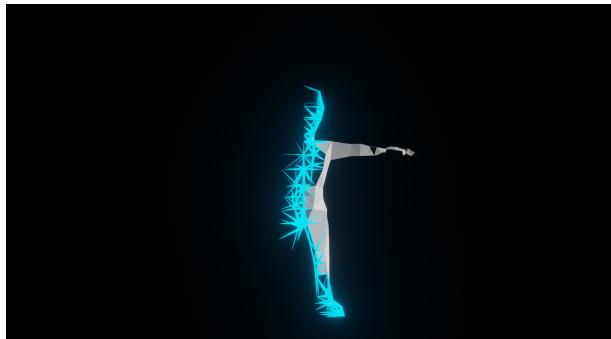


Figura 5.35: Strength = 3

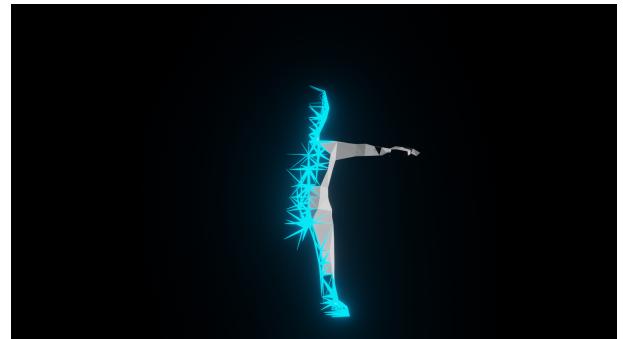


Figura 5.36: Strength = 4

Cuadro 5.7: Imágenes de los resultados para diferentes valores de strength

**5.1.1.2.2. Parámetro Wire Width** Este parámetro es el que determina como de ancha es el área donde es visible el wireframe. A continuación, se muestran unas imágenes de los resultados obtenidos con los diferentes valores para este parámetros que dan los resultados posibles para este objeto:

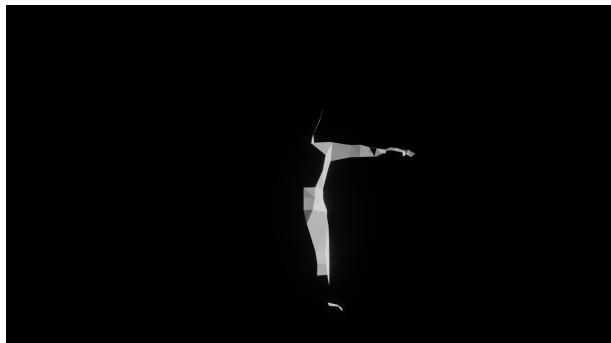


Figura 5.37: Wire Width = 0

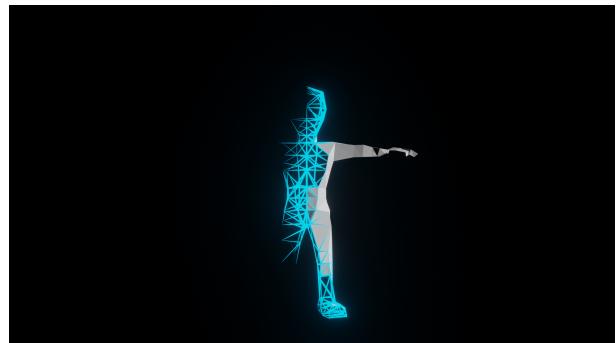


Figura 5.38: Wire Width = 1

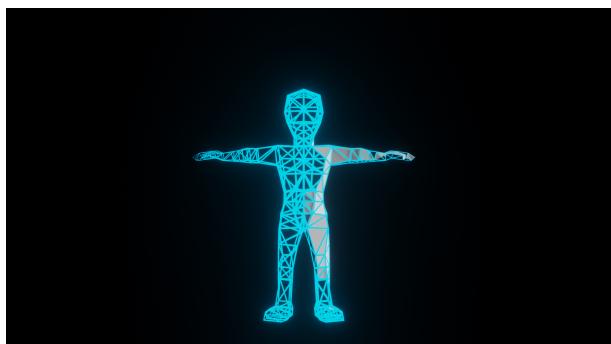


Figura 5.39: Wire Width = 6

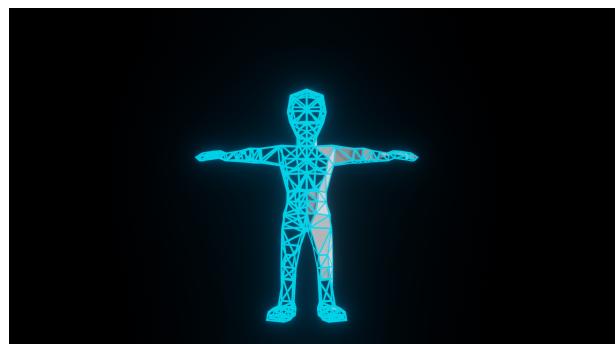


Figura 5.40: Wire Width = 12

Cuadro 5.8: Imágenes de los resultados para diferentes valores de wire width

Los valores de este parámetro dependen de las dimensiones del objeto al que se aplique el efecto. Para este caso, el objeto al ser uno pequeño los valores son bajos, pero en caso de que el objeto sea mayor los valores tendrían que ser mayores para conseguir el mismo resultado. Como se ve en la figura 5.37 cuando el valor es 0 no hay ninguna zona de wireframe visible, sin embargo cuando el valor es 6 como se ve en la figura 5.39 todo el objeto con el efecto aplicado se hace visible, esto es porque el valor de la anchura en la que se ve todo el objeto con el efecto aplicado debe ser igual al tamaño máximo de entre la anchura, altura y profundidad, para ajustar los umbrales lowest y highest del modificador de peso de los vértices por proximidad, como ya se explicó en el apartado de implementación de este efecto. En la figura 5.40 se observa como aunque se supere el valor donde ya se ve el cuerpo entero del wireframe, si se supera este valor no se ve cambios significativos, tan solo aumenta la intensidad de la luz del material.

**5.1.1.2.3. Parámetro Wire Thickness y Wire Color** Estos parámetros hacen referencia al grosor del wireframe del efecto y su color. A continuación, se muestran unas imágenes de los resultados obtenidos con los diferentes valores para este parámetros que dan los resultados posibles para este objeto:

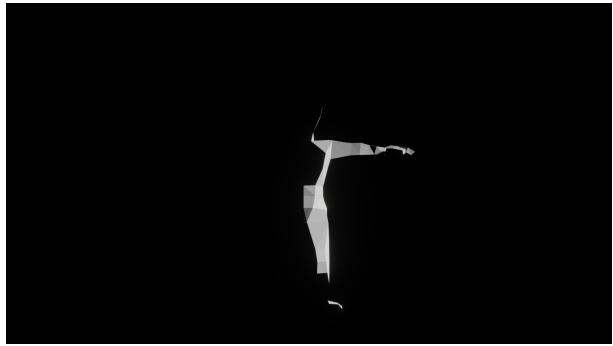


Figura 5.41: Thickness = 0

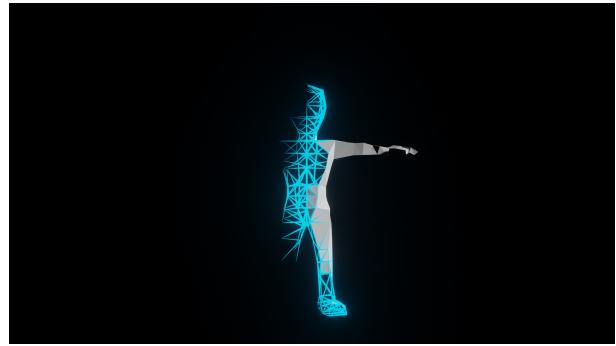


Figura 5.42: Thickness = 0.1; Color azul



Figura 5.43: Thickness = 0.5; Color rojo



Figura 5.44: Thickness = 1; Color violeta

Cuadro 5.9: Imágenes de los resultados para diferentes valores de thickness y color

Como se puede ver en la figura 5.41, este valor de thickness equivale a un valor del parámetro de wire width de 0. En la figura 5.43 se observa que a valores cercanos a 0.5 la anchura es tal que, incluso para un objeto de pocos polígonos ya da resultados muy poco nítidos y a partir de valores aún mayores como se observa en la figura 5.44 ya empiezan a producirse artefactos. Los valores que pueden producir resultados correctos dependerán en cierta medida del número de polígonos, pues si la densidad de polígonos es alta, se requerirá un valor bajo de thickness para se aprecie de manera correcta.

### 5.1.1.3. Pruebas funcionales para el efecto de rayo láser

Para el efecto de rayo láser, al igual que el rayo, las pruebas funcionales han consistido básicamente en probar los diferentes parámetros y comprobar sus resultados. A continuación, se muestran algunos de los resultados obtenidos con los diferentes parámetros en el equipo en el que se ha desarrollado el add-on, el mío, cuyas características ya se definieron en el apartado de especificaciones hardware:

**5.1.1.3.1. Parámetros Max Radius, Radius, Noise** Este parámetro es el que determina las dimensiones del cubo o el cilindro si se trata del láser para Cycles o Eevee respectivamente en cuanto a anchura y altura (la longitud vendrá determinada por el vector indicado por los parámetros end y begin). Este radio será el máximo que podrá tener el rectángulo en el caso del láser para Cycles, dado que el material podrá hacer que el láser contenido en el rectángulo (el material, que es el que da el efecto del láser, es un material volumétrico) sea mayor o menor cuanto mayor o menor sea el valor del parámetro Radius, que permite que el láser tenga un radio entre 0 y Max Radius, su valor va entre 0 y 1, por lo que es un porcentaje de cuanto del máximo radio queremos que tenga el láser. El parámetro Noise determina como de nítido y liso queremos que sea el láser. El parámetro Noise puede influir en como se visualiza el láser y puede influir en que se aprecie más o menos el parámetro Radius, pues a mayor ruido menos se apreciará donde acaba el rayo y empieza la dispersión del ruido, por ello lo he analizado junto a los otros dos parámetros. Para el parámetro de velocidad de animación las pruebas que se han hecho han consistido principalmente en probar diferentes valores de velocidad para comprobar sus resultados. A continuación, algunos ejemplos de uso de estos parámetros:



Figura 5.45: Radius = 0; Noise = 1



Figura 5.46: Radius = 1; Noise = 0



Figura 5.47: Radius = 0.5; Noise = 0.5

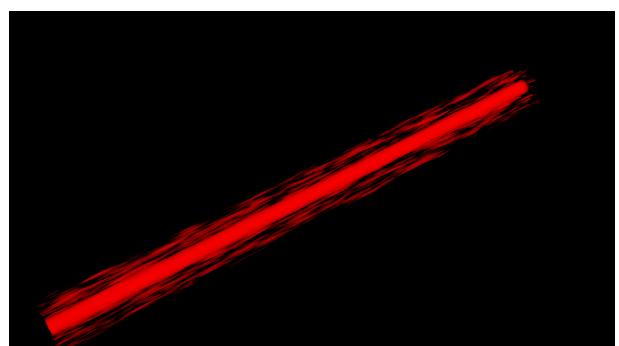


Figura 5.48: Radius = 0.35; Noise = 0.35

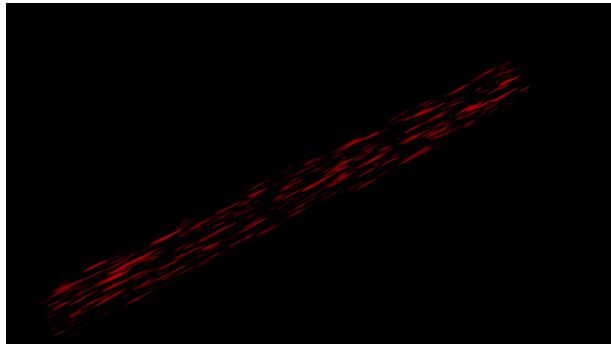


Figura 5.49: Radius = 0; Noise = 0.3

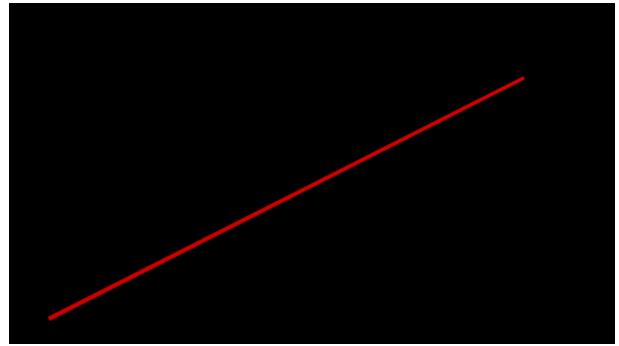


Figura 5.50: Radius = 0.1; Noise = 0

Cuadro 5.10: Imágenes de los resultados para diferentes valores de Radius y Noise

Cuanto más ruido menos se aprecia la figura del láser en si porque queda opacado por el ruido, además, se hace más visible la forma cuadrada del rectángulo donde se genera el láser, por lo que no es recomendable valores altos de ruido. También se pude apreciar como a pesar de que el radio del láser sea 0, cuando se añade ruido, se sigue viendo pues es independiente del láser. También se puede comprobar que el ruido toma valores entre 0 y 1 pudiendo estar desvanecido o llenando el rectángulo donde está contenido [5.45](#).

**5.1.1.3.2. Parámetros strength y color** Este parámetro determina la potencia de emisión que tendrá el láser, además, puede dar un color blanco al cuerpo del láser cuando el valor del strength es alto, dando mayor efecto de láser. Estos son algunos ejemplos:

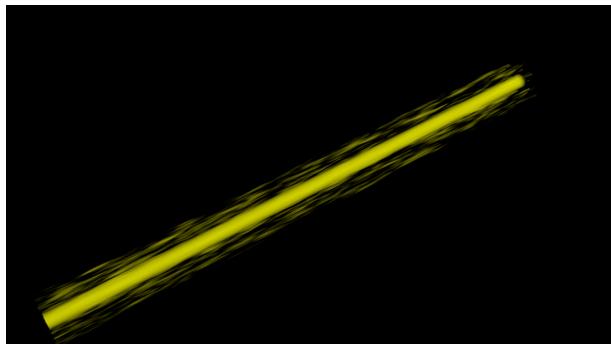


Figura 5.51: Strength = 10, Color amarillo

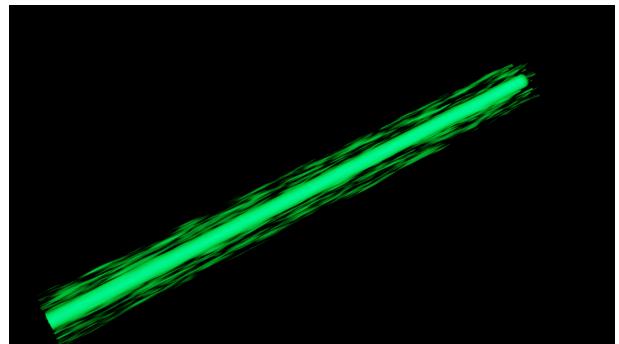


Figura 5.52: Strength = 50, Color verde

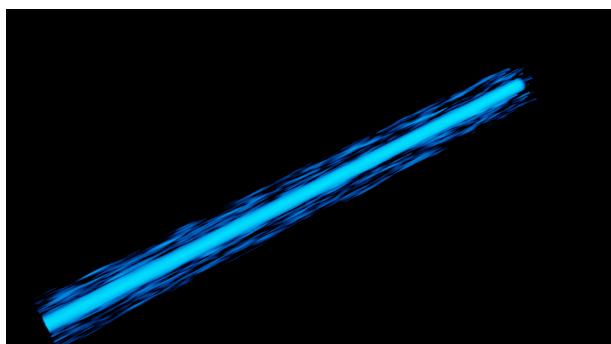


Figura 5.53: Strength = 100, Color azul

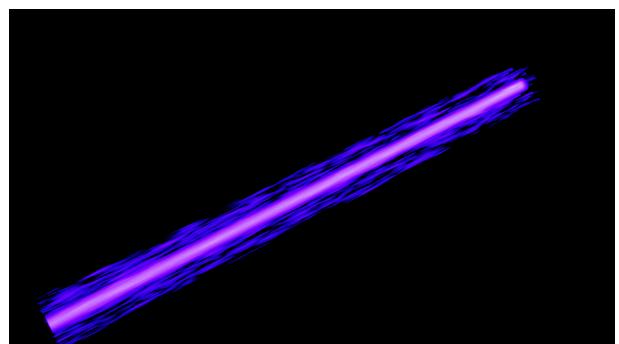


Figura 5.54: Strength = 500, Color violeta

Cuadro 5.11: Imágenes de los resultados para varios valores de strength y color en Cycles

**5.1.1.3.3. Parámetro renderer** Este parámetro permite cambiar entre un tipo de realización de rayo adaptándose a los renderizadores Cycles o Eevee de Blender. A pesar de que el láser de Eevee es un versión muy simple de láser permite añadir otra opción en caso de que se quiera usar en una escena que se vaya a renderizar en Eevee, pues uno de los requisitos a los que había que responder era que el add.on fuese utilizable en el mayor número de escenas posibles y que fuese versátil, por esto, se ha optado por realizar esta versión. Las pruebas que se podían hacer a este láser de Eevee eran tan solo de color y de fuerza de emisión (strength) por lo que aquí va dos ejemplos de los resultados que da este láser con valores altos y bajos de strength y diferente color:

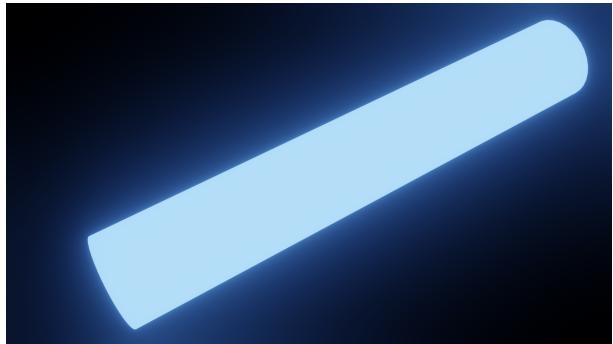


Figura 5.55: Strength = 5, Color azul

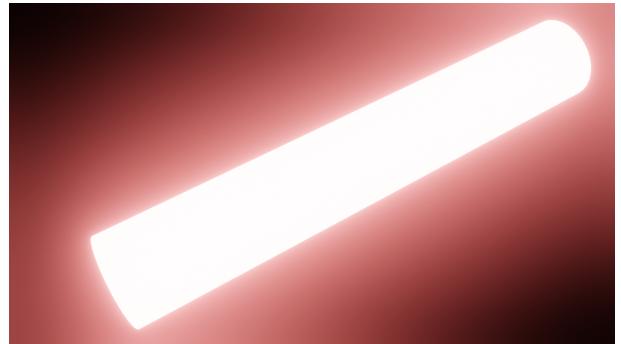


Figura 5.56: Strength = 50, Color rojo

Cuadro 5.12: Imágenes de los resultados para diferentes valores de strength y color en Eevee

#### 5.1.1.4. Pruebas funcionales para el efecto de escudo de fuerza

Para el efecto del escudo de fuerza, al igual que los anteriores efectos, las pruebas funcionales han consistido básicamente en probar los diferentes parámetros y comprobar sus resultados. A continuación, se muestran algunos de los resultados obtenidos con los diferentes parámetros en el equipo en el que se ha desarrollado el add-on, el mío, cuyas características ya se definieron en el apartado de especificaciones hardware. Este escudo está compuesto de 3 objetos, dos de ellos opcionales, uno principal que va a ser generado en todos los casos y otros dos opcionales: el escudo de wireframe y el escudo de fragmentos. Dado que la mayoría de estos parámetros son similares se harán agrupaciones de los parámetros similares de los diferentes escudos.

**5.1.1.4.1. Parámetros de los materiales del escudo** Estos parámetros engloban a todos los parámetros relacionados con las características de los materiales de los escudos que tienen que ver con su parte más visual: el color y la luminosidad. Entrarían en este grupo los parámetros Shield Emission Strength, Wireframe Emission Strength, Fragments Emission Strength: estos son los encargados de dar la fuerza de emisión a los respectivos escudos, dado que los 3 escudos son independientes sus colores también lo son. También entrarían los parámetros Shield Color, Wireframe Color, Fragments Color que se encargan de dar el color a los escudos. Estos parámetros son iguales a los ya definidos para los efectos anteriores, por lo que se obviarán las imágenes por no aportar información diferente a la anterior. También engloba en este grupo de parámetros el parámetro wireframe thickness del escudo de wireframe, el cual es igual al parámetro thickness del efecto de transición de wireframe, por lo que las pruebas realizadas son las mismas que para el otro y no se podrán las imágenes tampoco.

**5.1.1.4.2. Parámetro Shield Noise** este parámetro es similar al parámetro Noise del efecto de rayo láser, no obstante su aspecto es diferente por lo que mostraré un par de imágenes cuales son los resultados obtenidos si se tiene ruido y si no se tiene:



Figura 5.57: Noise 0

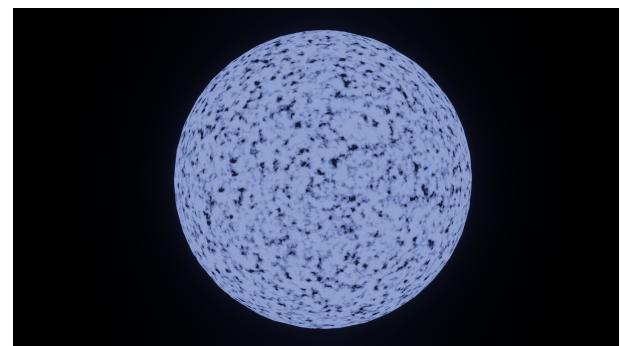


Figura 5.58: Noise 1

Cuadro 5.13: Imágenes de los resultados del escudo principal con y sin ruido

**5.1.1.4.3. Parámetros de control de la animación** Además de los parámetros que se encargan de dar el aspecto visual existen otros parámetros encargados de controlar la

animación por defecto generada para los escudos, estos son: Wireframe Animation Delay, Fragments Animaton Delay, Wire build start frame, wire build duration frames. Los dos primeros parámetros se encargan de establecer una diferencia en frames respecto a la animación del escudo principal, de manera que la animación de estos escudos se produzca antes, a la vez o después que la del escudo principal. Las animaciones de los 3 escudos constan de 2 keyframes de escalado, por lo que la diferencia en keyframes es la misma entre los keyframes de las dos animaciones. A continuación se mostrarán imágenes que permitan apreciar como la diferencia entre keyframes de las animaciones del escudo de wireframe (escogido aleatoriamente ya que en el caso del escudo de fragmentos es igual) son la misma para todos los keyframes.

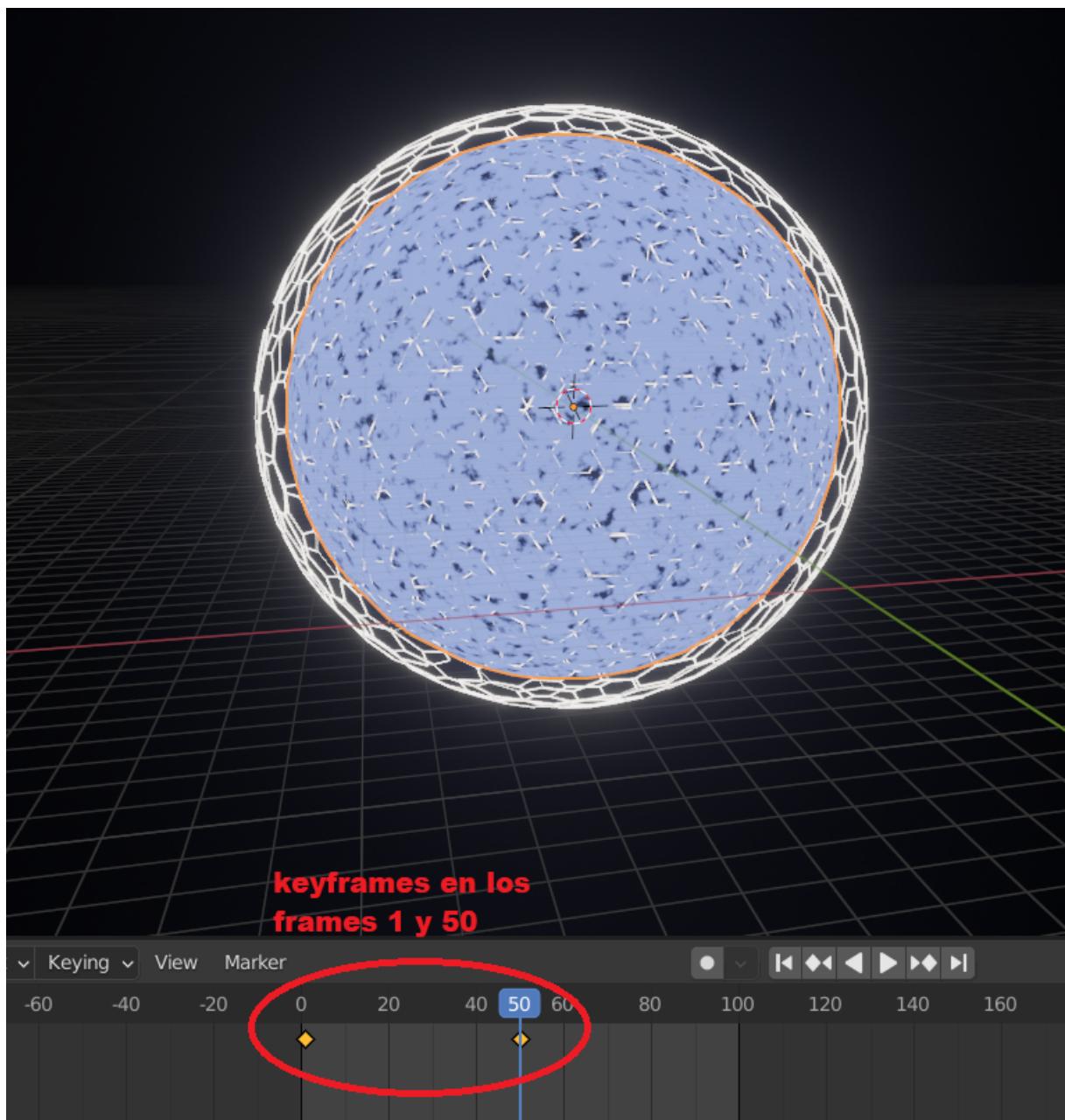


Figura 5.59: Keyframes de la animación del escudo principal

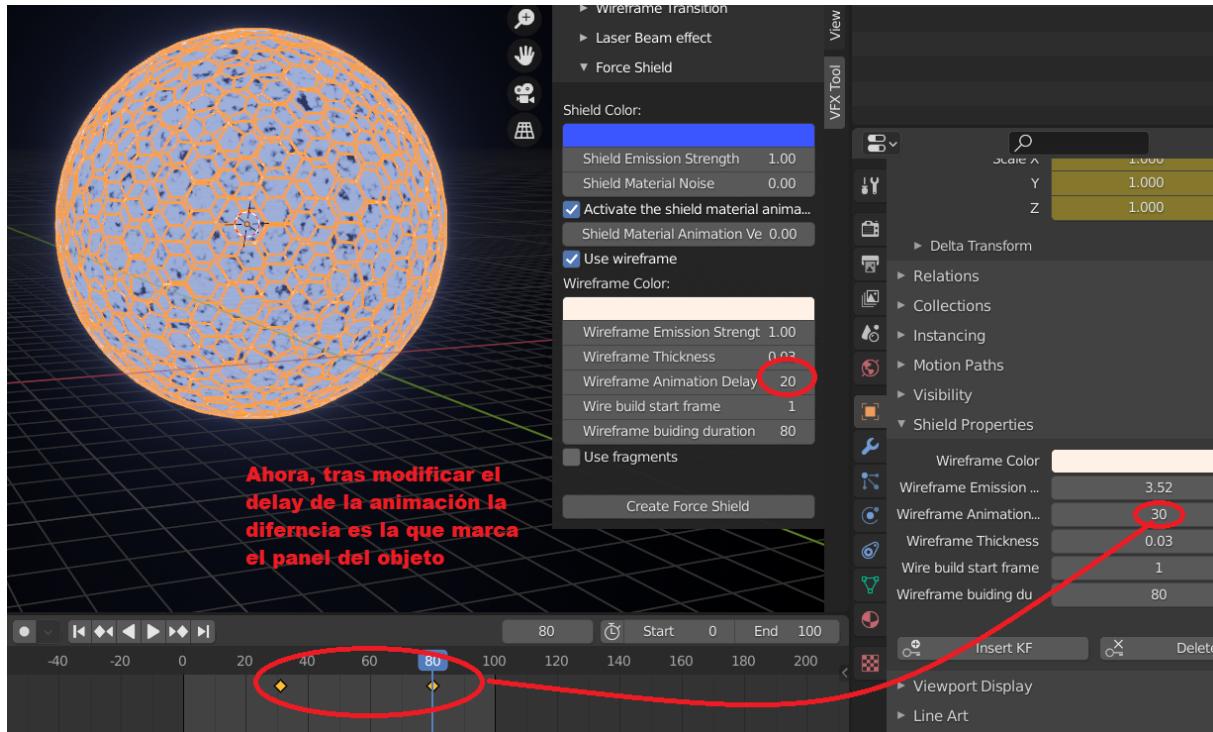


Figura 5.61: Nueva diferencia de keyframes modificada en el panel del objeto

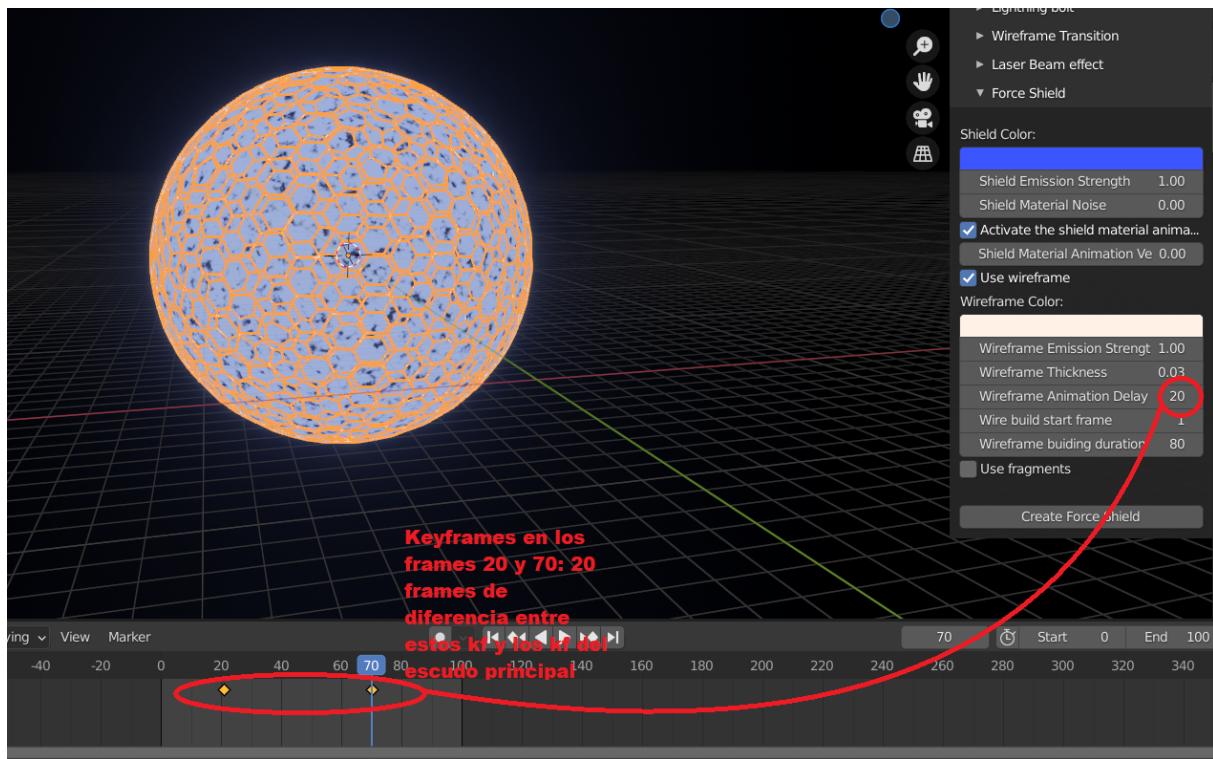


Figura 5.60: Keyframes de la animación del escudo wireframe con la diferencia que marca el delay

En la figura 5.59 se puede observar como la animación tiene una duración de 50 frames con sus keyframes en los frames 1 y 50, la siguiente imagen, 5.60, muestra que la animación del escudo de wireframe comienza en el frame 20 y acaba en el 70, es decir, tiene la misma duración pero tiene un desfase de 20 frames. Por último, en la figura 5.61 se observa que

ahora el desfase de la animación del escudo de wireframe es de 30 frames debido a que se ha modificado desde le panel de propiedades del objeto. Se han hecho pruebas de qué pasaba si se eliminaba la animación y se creaba una nueva con un desfase mayor para saber si desde el panel se podían seguir modificando el desfase y el resultado es que así es, no obstante, la diferencia que marca la propiedad ya no tiene porqué corresponderse con el desfase real, aunque sigue funcionando.

Por último, los dos últimos parámetros de control de animaciones, el Wire build start frame y el Wire build duration frame son los frames encargados de controlar los frames en los que el wireframe del escudo de wireframe aparecerá, ya que, además del efecto de surgir de la nada del escudo debido a la animación de escalado, el wireframe también se "construye" durante los frames que indiquen estos dos parámetros.

### 5.1.1.5. Pruebas funcionales de inserción y borrado de keyframes

Se debía comprobar si se insertaban keyframes en propiedades que no se habían modificado, pues, por ejemplo en el efecto del rayo, al presionar el botón de insertar keyframes habiendo cambiado el parámetro length y el parámetro strength y después, algunos frames más adelante, insertar de nuevo keyframes habiendo cambiado el parámetro length y el strength otra vez, si se volvía a presionar el operador de insertar keyframes por tercera vez, en un frame intermedio entre los que acaba de insertar, y habiendo cambiado el parámetro length pero no en el strength, había que comprobar si se insertaba keyframe en el strength a pesar no haberlo modificado, ya que si así era, rompía la interpolación de la animación hecha por los 2 primeros keyframes introducidos antes. También se debían realizar pruebas de si al borrar keyframes las propiedades en qué estado se , si como antes de haber insertado el keyframe o con el valor que tenían.

**5.1.1.5.1. Pruebas de animación del rayo** A continuación se mostrarán ejemplos de uso de los operador de keyframes en el efecto del rayo. Primero aparecerán las propiedades iniciales con keyframes, después se modificarán ciertos parámetros y se volverán a insertar, una vez hecha esta segunda inserción, se modificarán, en un frame intermedio entre los dos frames anteriores donde se han hecho las inserciones alguna de las propiedades con keyframes y se hará la tercera inserción para comprobar si se insertan keyframes en las propiedades no modificadas y se rompe la animación producida por las dos inserciones anteriores, siguiendo con el ejemplo descrito en el párrafo anterior. Por último, se insertará un cuarto keyframe al final..

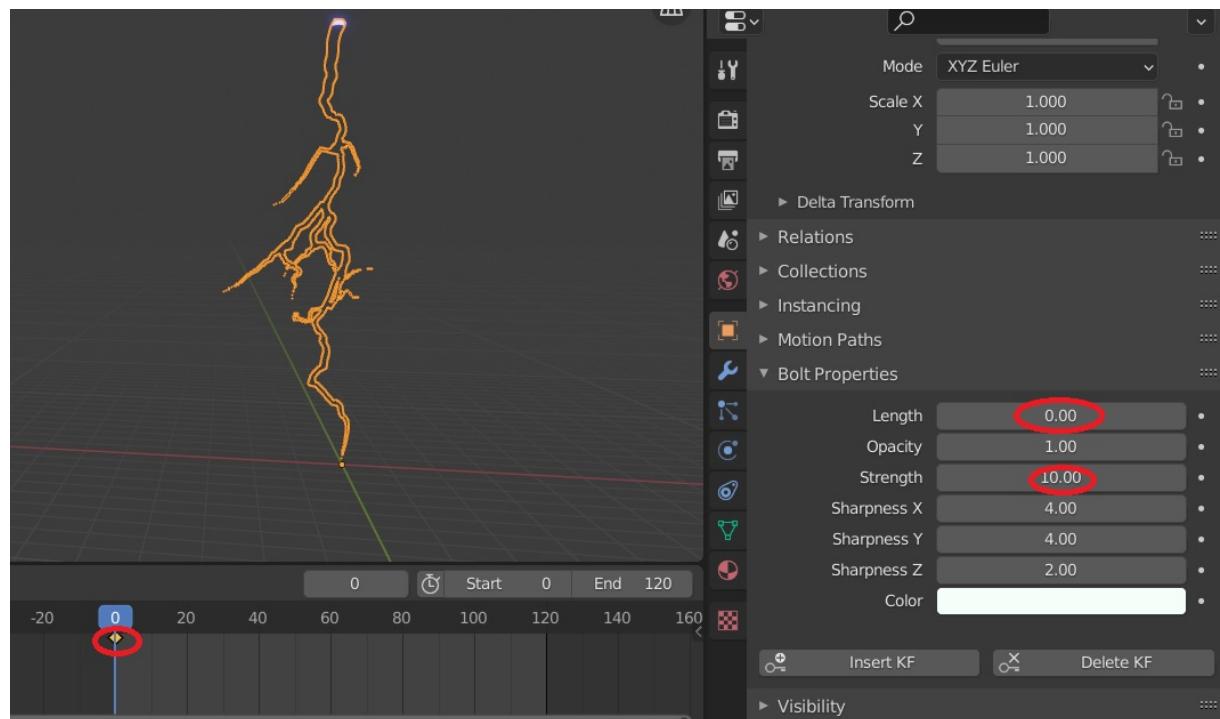


Figura 5.62: Propiedades con keyframes (Primera inserción)

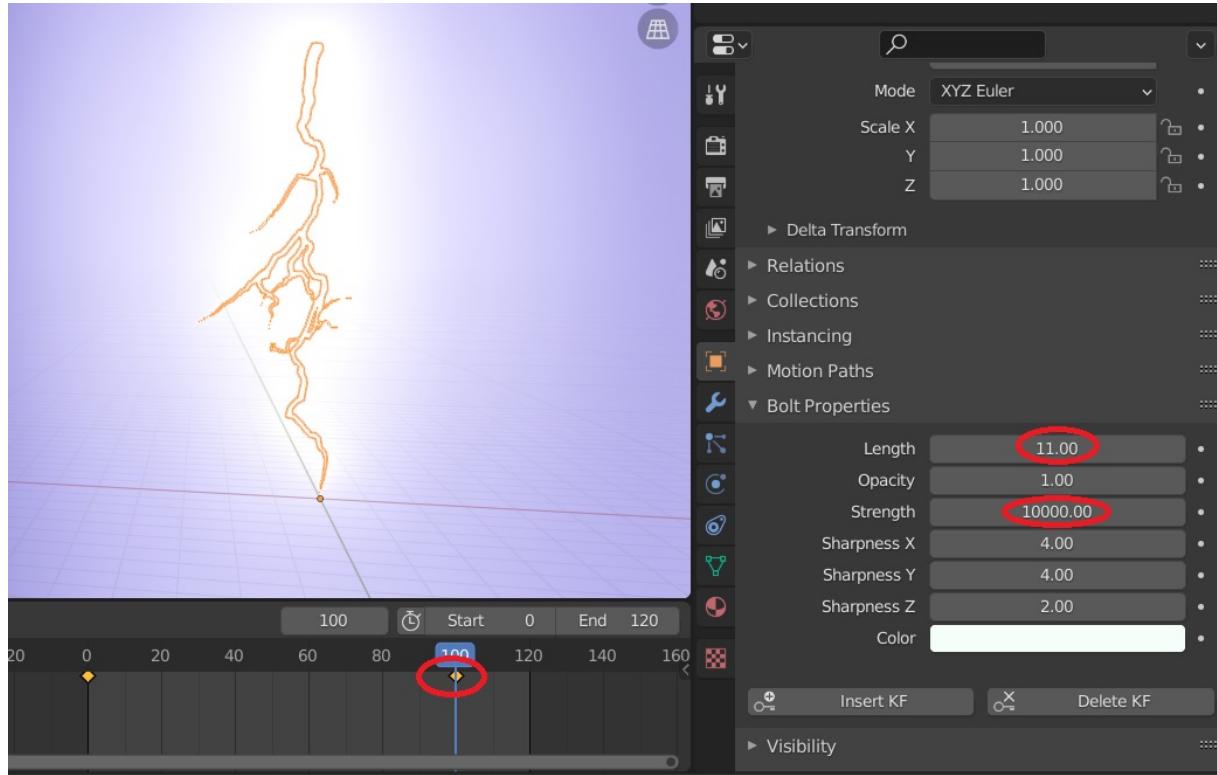


Figura 5.63: Propiedades con keyframes (Segunda inserción)

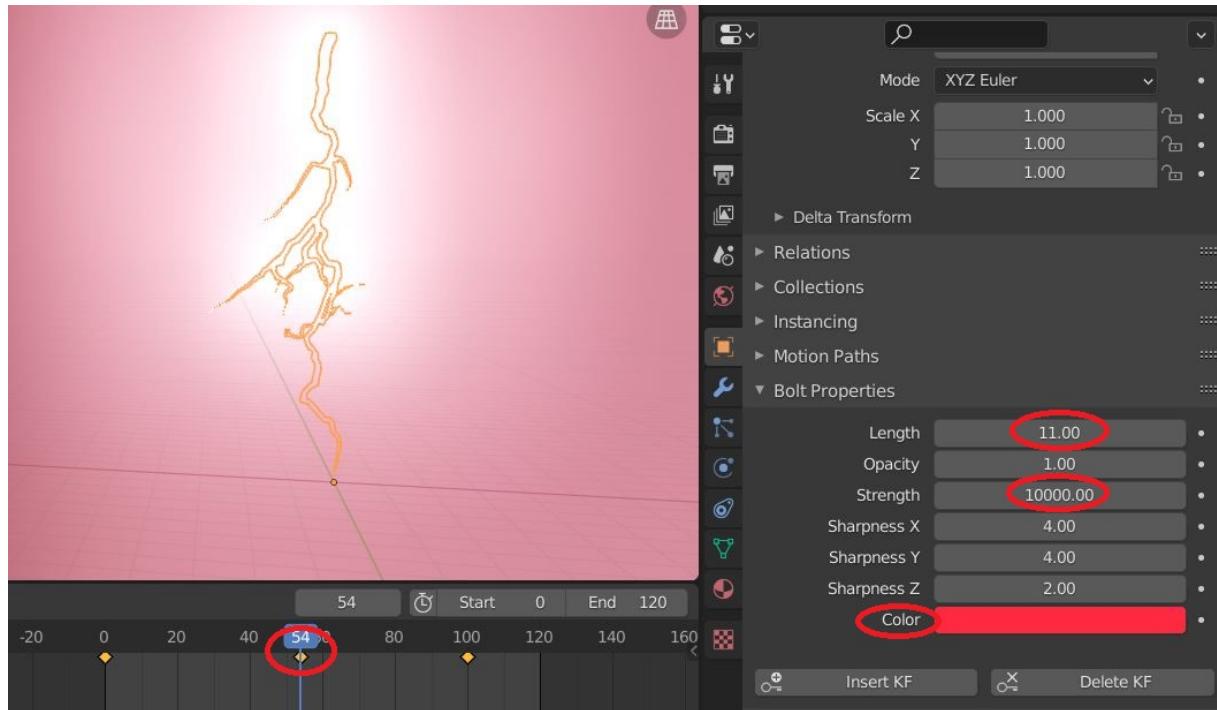


Figura 5.64: Propiedades con keyframes (Tercera inserción)

En la figura 5.64 se observa como, el único parámetro que se ha cambiado respecto a la segunda inserción de keyframes es el color, los parámetros length y strength tienen los mismos valores que en la segunda inserción, y no han sido modificados, por lo tanto, si en estos parámetros también se insertaran keyframes en la tercera inserción, la animación

de length y strength terminaría en el frame donde se ha hecho la tercera inserción, ya que del frame de la tercera al frame de la segunda estos parámetros no variarían. No obstante, esto como se puede comprobar en las siguientes imágenes no es así, pues parece que Blender sólo anima los parámetros que han sido modificados, cuando anteriormente se hayan animado otros parámetros que sí hayan sido modificados, en otras palabras, sólo si no se han animado parámetros modificados se podrán insertar keyframes de parámetros que no hayan variado sus valores con solo clicar el operador, una vez que se insertan keyframes en parámetros que han variado sus valores sólo se podrán animar parámetros sin variar sus valores reinsertando los mismo valores en cada inserción.

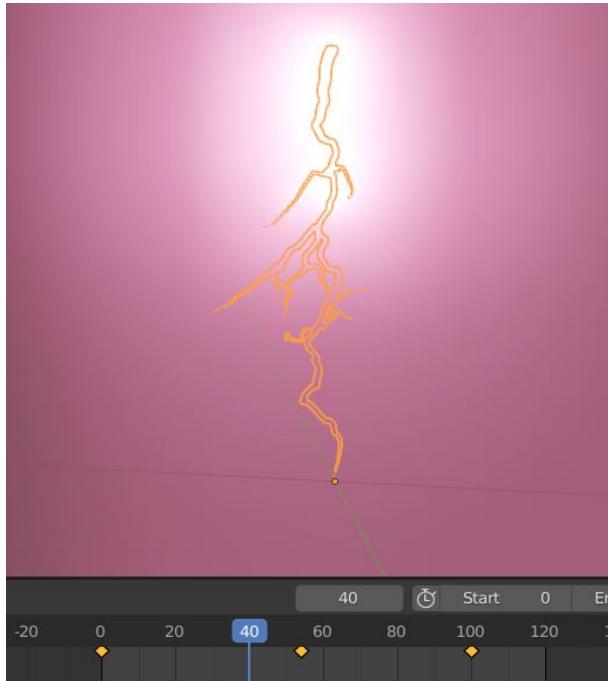


Figura 5.65: Rayo entre la primera y la segunda inserción de keyframes

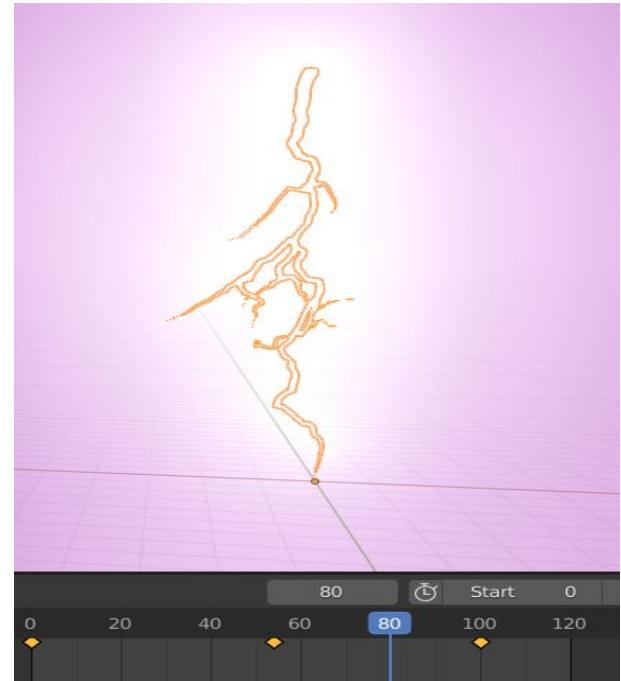


Figura 5.66: Rayo entre la segunda y la tercera inserción de keyframes

Cuadro 5.14: Imágenes de los resultados del escudo principal con y sin ruido

Como se aprecia en la figura 5.65 aunque en el frame 50 se insertaron keyframes donde el valor del parámetro length era de 11, es decir, la longitud completa del rayo, a falta de 10 frames para llegar al frame de la inserción de los keyframes no se ha completado el 80 % de la longitud del rayo, siendo la interpolación de la animación lineal, significa que aunque el valor de length cuando se hizo la inserción era 11, este valor provenía de la inserción anterior en el frame 100, donde sí se varió el valor de length y por tanto, la inserción de keyframes sí afectó a este parámetro, donde también se modificó el parámetro strength a 10000 y que también se vio afectado por la inserción de keyframes, por ello, aunque en el frame 50 el parámetro strength conservaba su valor de 10000 de la inserción en el frame 100 en el frame 40 tampoco se ha llegado al 80 % de la luminosidad final ya que en el frame 50, al solo haber cambiado el color, sólo este parámetro se ha visto afectado en la inserción. En la figura 5.66 se aprecia como la luminosidad ha seguido aumentando hasta llegar al valor de 10000 en el parámetro strength en el frame 100 que es donde está afectado por el keyframe de la segunda inserción, de igual manera que length. El color también está volviendo a cambiar ya que cuando se hizo la inserción en el frame 100 el color no había sido modificado, por lo que la inserción también le afectó, en el frame 50, la

tercera inserción, se cambió al color rojo, por lo que al haber sido modificado la inserción afectó a este parámetro, siendo el único al que le afectó por haber sido el único que se modificó, y está volviendo al color que tenía el rayo en el frame 100 cuando se insertó keyframe, que era el mismo que en el principio: azul.

Por último, para comprobar el funcionamiento de los operadores de keyframes veremos qué pasa si se borran con el operador de borrado de keyframes los keyframes del frame 50, el intermedio.

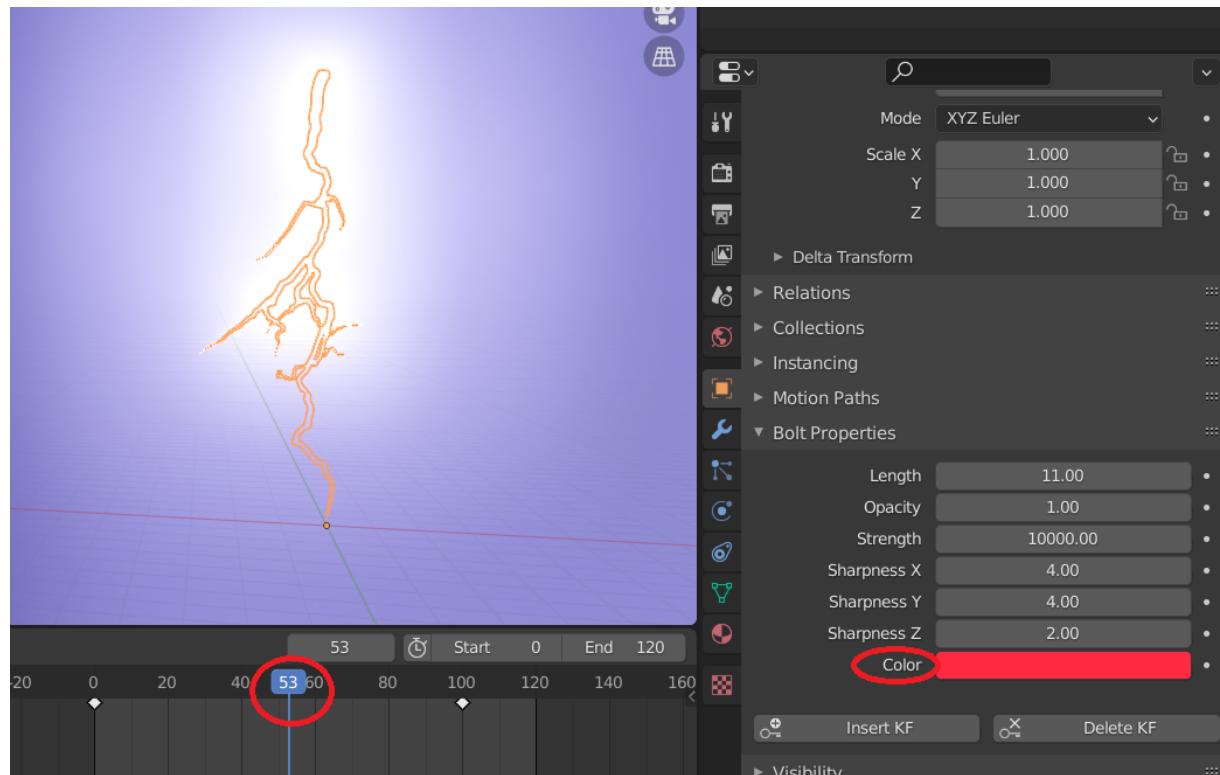


Figura 5.67: Borrado de keyframes en el frame intermedio

Como se ve en la figura 5.67 a pesar de tener el color rojo en el panel de propiedades al haberse borrado el keyframe de color en el frame 50 el color vuelve a ser el anterior al que tenía cuando se insertó el keyframe.

## 5.1.2. Pruebas de rendimiento

En las pruebas de rendimiento comentaré la eficiencia en términos de costes temporales y de memoria. Los primeros efectos, Ligthning Bolt y Wireframe Transition son los que más pueden exigir al equipo donde se usen, el efecto del rayo debido al coste temporal del algoritmo de generación del rayo que incluso puede llegar a colapsar Blender si se abusa de los valores de los parámetros de generación de ramas: Branches Depth, Branches Dens y Max Subbranches; el efecto de Wireframe transition no tiene ese problema, en cambio, tiene el inconveniente de que puede bajar mucho el frame rate si se usa para el efecto un objeto de muchos polígonos ya que se debe duplicar la geometría para el efecto y el modificador peso de los vértices en función de la proximidad obliga a realizar muchos cálculos por cada vértice de distancia respecto al manejador, como este modificador está presente en los dos objetos para el efecto se hace muy costoso si el número de polígonos es alto y, por ende, el de vértices. Los efectos de Laser Beam y Force Shield no tienen un coste de memoria ni temporal relevante, ambos se pueden generar y funcionan sin más problemas, por ello, no se profundizará en el rendimiento de estos pues no tienen más que destacar que lo destacable de cualquier objeto de Blender.

### 5.1.2.1. Pruebas de rendimiento del efecto rayo

El coste temporal del algoritmo del rayo vendrá definido por el número de vértices seleccionados para ser extrudidos y el número de veces que será extrudido cada vértice. Dado que por cada nivel de ramificación el rayo aumenta en MAX\_SUBBRANCHES cada vértice extrudido, el crecimiento del número total de vértices vendrá determinado por la siguiente fórmula:

$$d*[BRANCHES\_DEPTH, MAX\_SUBBRANCHES^{BRANCHES\_DEPTH} - 1]*d*(N_0 - 2) \quad (5.1)$$

Donde  $N_0$  es el número de vértices inicial y d la densidad de vértices a seleccionar, la siguiente ecuación determina que el número total de vértices puede variar entre BRANCHES\_DEPTH o  $MAX\_SUBBRANCHES^{BRANCHES\_DEPTH} - 1$  dependiendo si el número de subramas por nivel ( $MAX\_SUBBRANCHES$ ) es 1 o mayor que 1. En la ecuación, lo que está entre corchetes determina el rango del número de vértices que pueden llegar a ser extrudidos por vértice inicial del rayo, BRANCHES\_DEPTH representa el límite inferior y  $MAX\_SUBBRANCHES^{BRANCHES\_DEPTH} - 1$  el límite superior. Además, el coste del algoritmo es de orden  $O(N*M*W)$  pues el código de generación de las subramas es un bucle anidado que se ejecuta tantas veces como BRANCHES\_DEPTH indique (el factor N), y por cada iteración se realizan otras M iteraciones por cada rama que se tenga que generar por nivel, además, en cada generación de ramas se deben recorrer y extrudir los vértices seleccionados (el factor W), no obstante como el factor W, si el número de MAX\_SUBBRANCHES es mayor que 1, sigue una fórmula exponencial, el coste del algoritmo será exponencial. A continuación, se mostrarán unas gráficas que muestran como aumentan los vértices seleccionados, extrudidos y totales en función de los parámetros BRANCHES\_DENS y MAX\_SUBBRANCHES. Para cada prueba realizada se han usado 4 valores de densidad: 0.25, 0.5, 0.75 y 1 para valores de subramas de 1, 2 y 3:

Con MAX\_SUBBRANCHES = 1

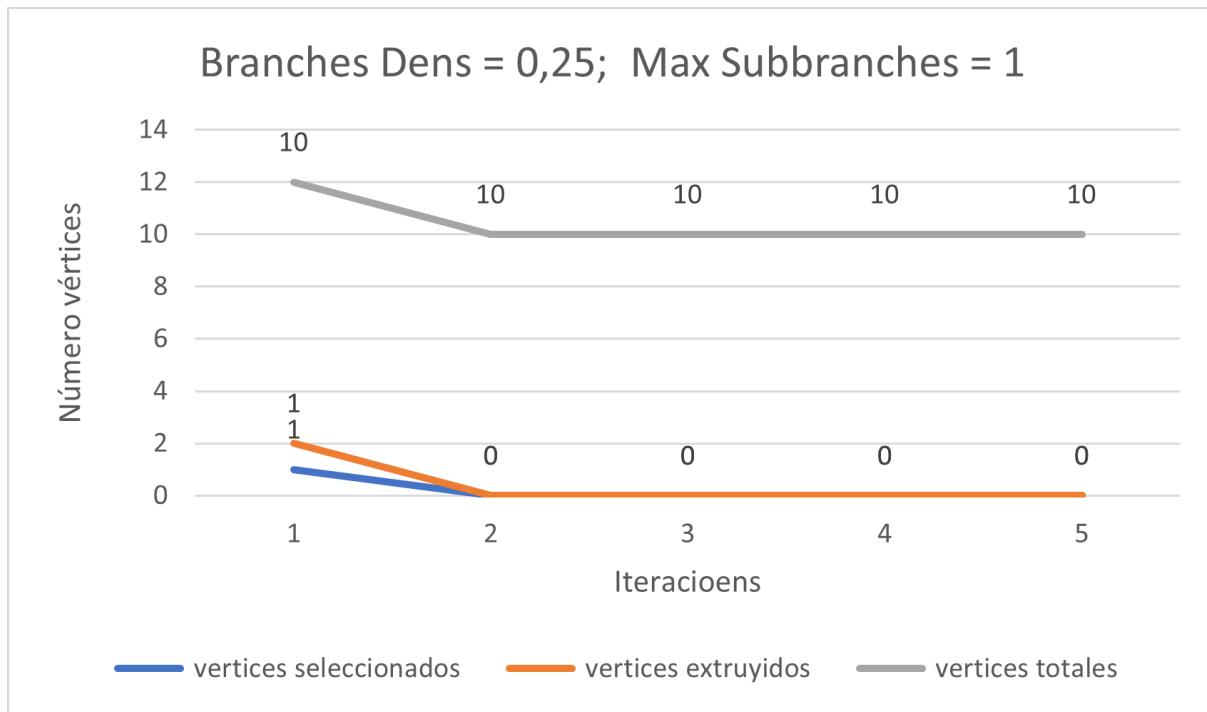


Figura 5.68: Valores de los vértices con densidad del 25 % y max subbranches = 1

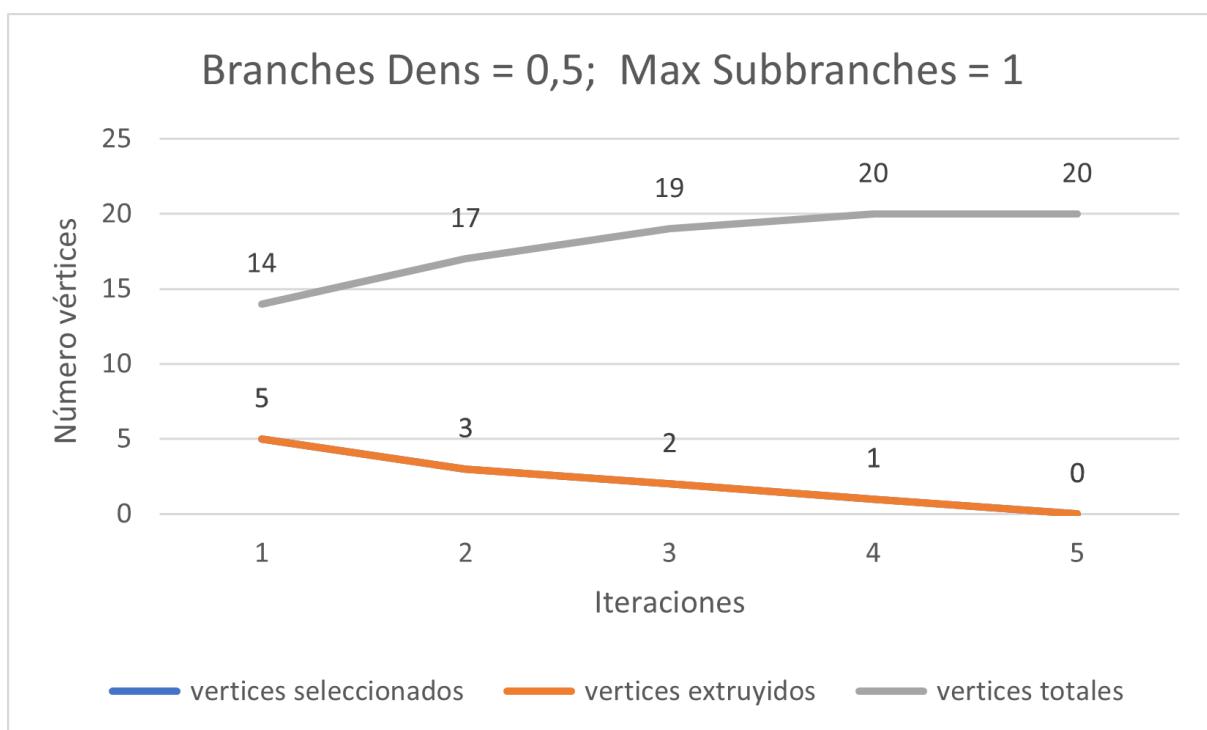


Figura 5.69: Valores de los vértices con densidad del 50 % y max subbranches = 1

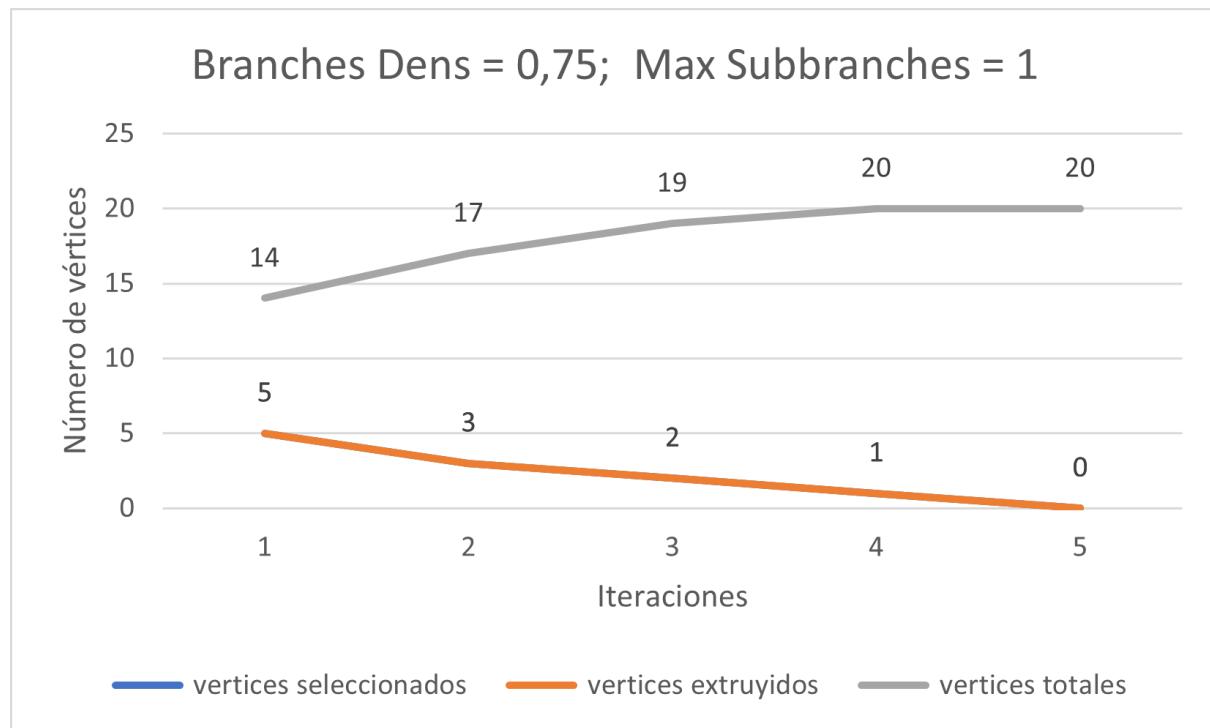


Figura 5.70: Valores de los vértices con densidad del 75 % y max subbranches = 1

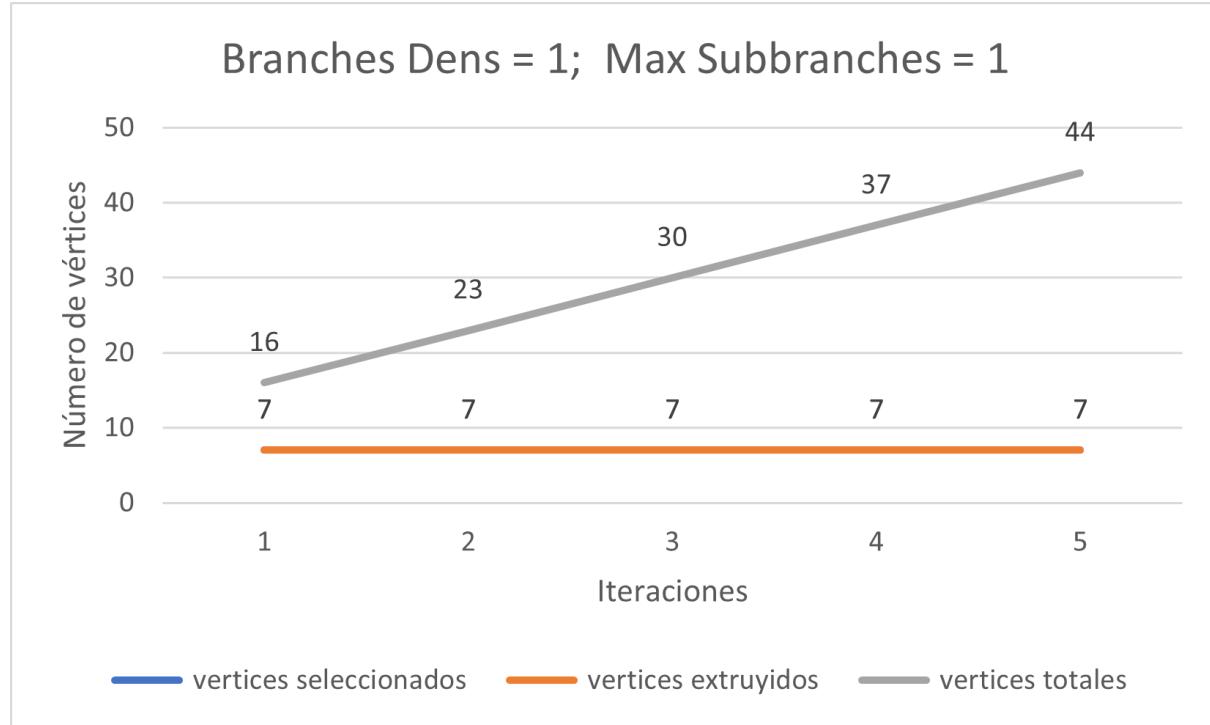


Figura 5.71: Valores de los vértices con densidad del 100 % y max subbranches = 1

En estos gráficos, como se puede observar, las líneas que marcan los vértices seleccionados para la extrusión son los mismos que los que son extrudidos ya que cada vértice seleccionado sólo se extrude una vez por iteración, tal y como marca el parámetro MAX\_SUBBRANCHES. Esto sólo pasa para este caso particular, cuando el número máximo de subramas es mayor que 1 el número de vértices extrudidos por iteración es mayor que el número de vértices seleccionados, debido a que por cada seleccionado se extruyen

`MAX_SUBBRANCHES` nuevos vértices, tal y como veremos en las imágenes siguientes. También es destacable el hecho de que cuando la densidad está por debajo de 0.5 el número de vértices seleccionados llega mucho antes a 0 de que terminen las iteraciones determinadas por `BRANCHES_DEPTH`, lo cual cambia pasado el umbral del 0.5.

Con `MAX_SUBBRANCHES` = 2

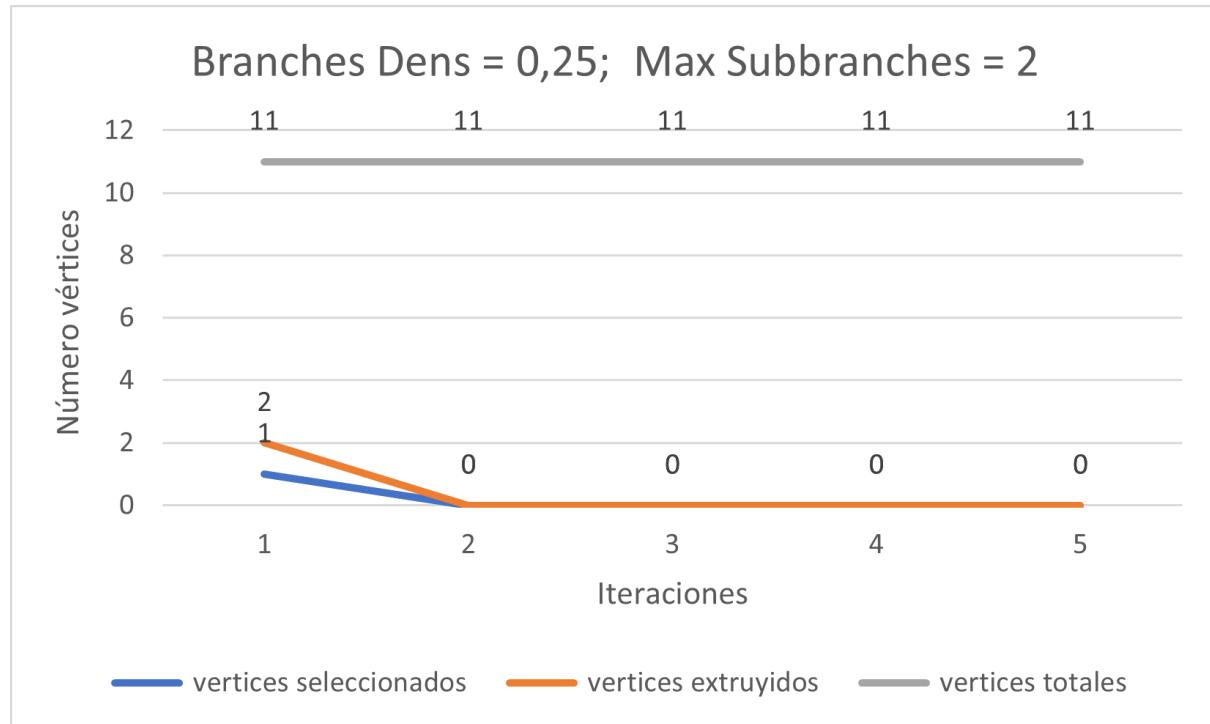


Figura 5.72: Valores de los vértices con densidad del 25 % y max subbranches = 2

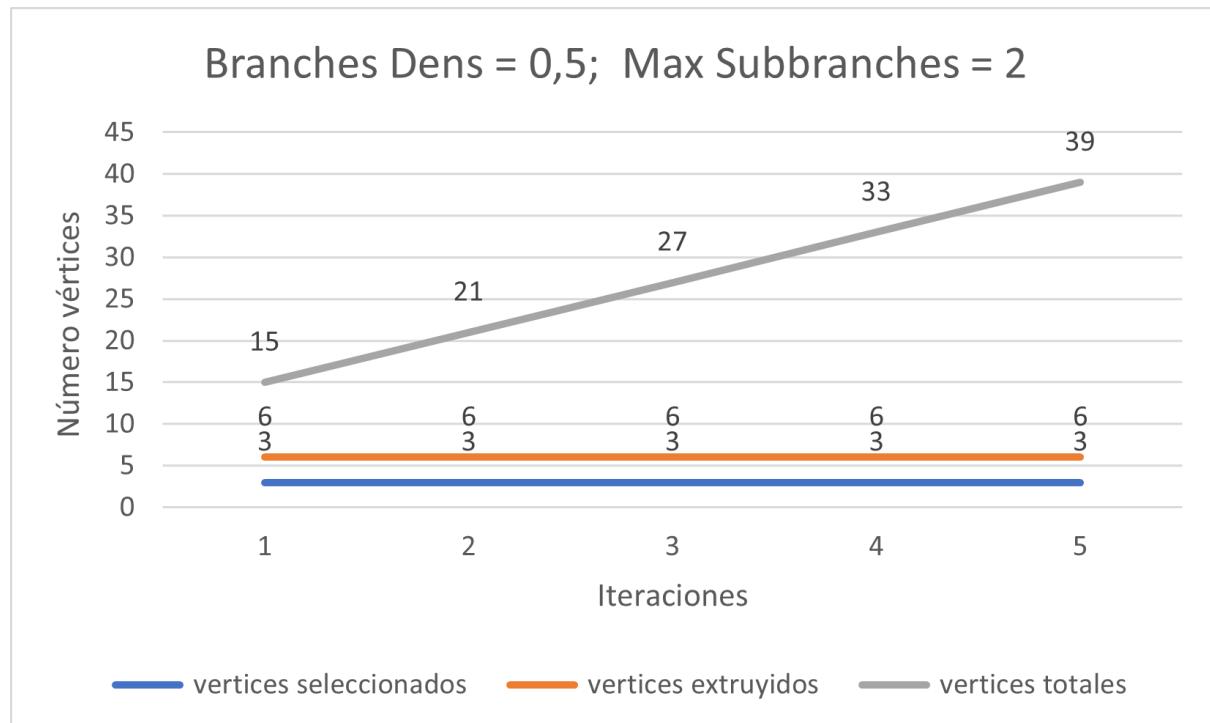


Figura 5.73: Valores de los vértices con densidad del 50 % y max subbranches = 2

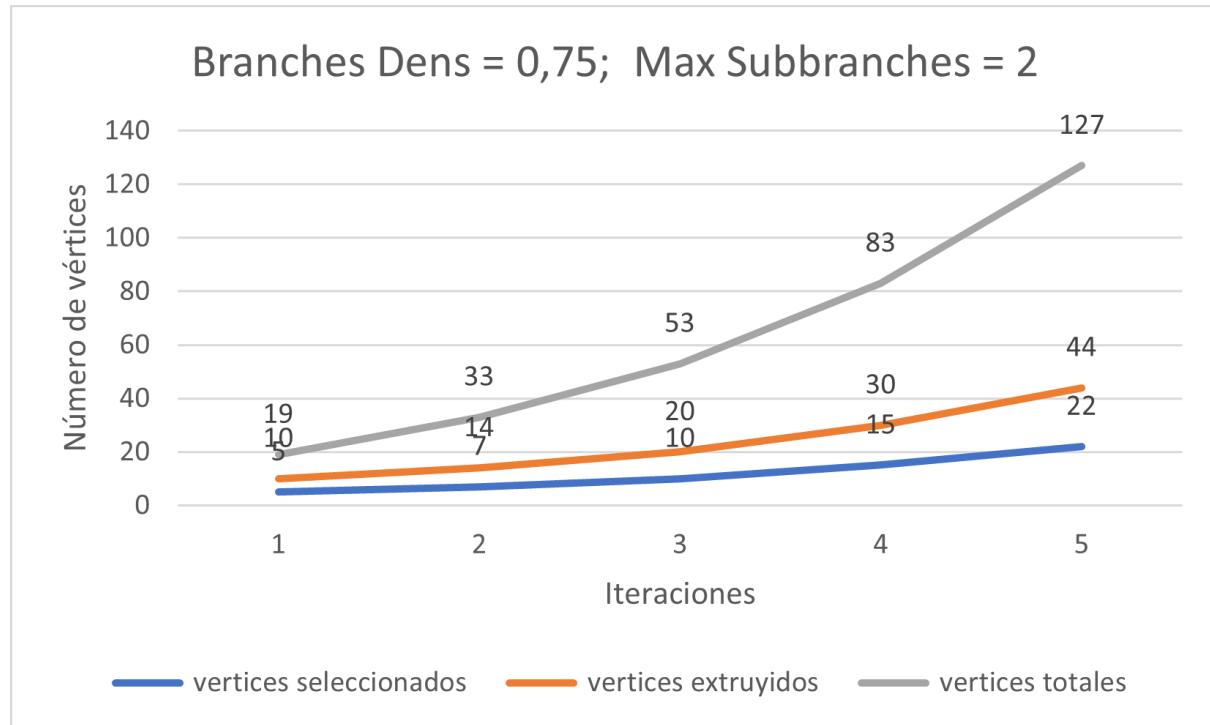


Figura 5.74: Valores de los vértices con densidad del 75 % y max subbranches = 2

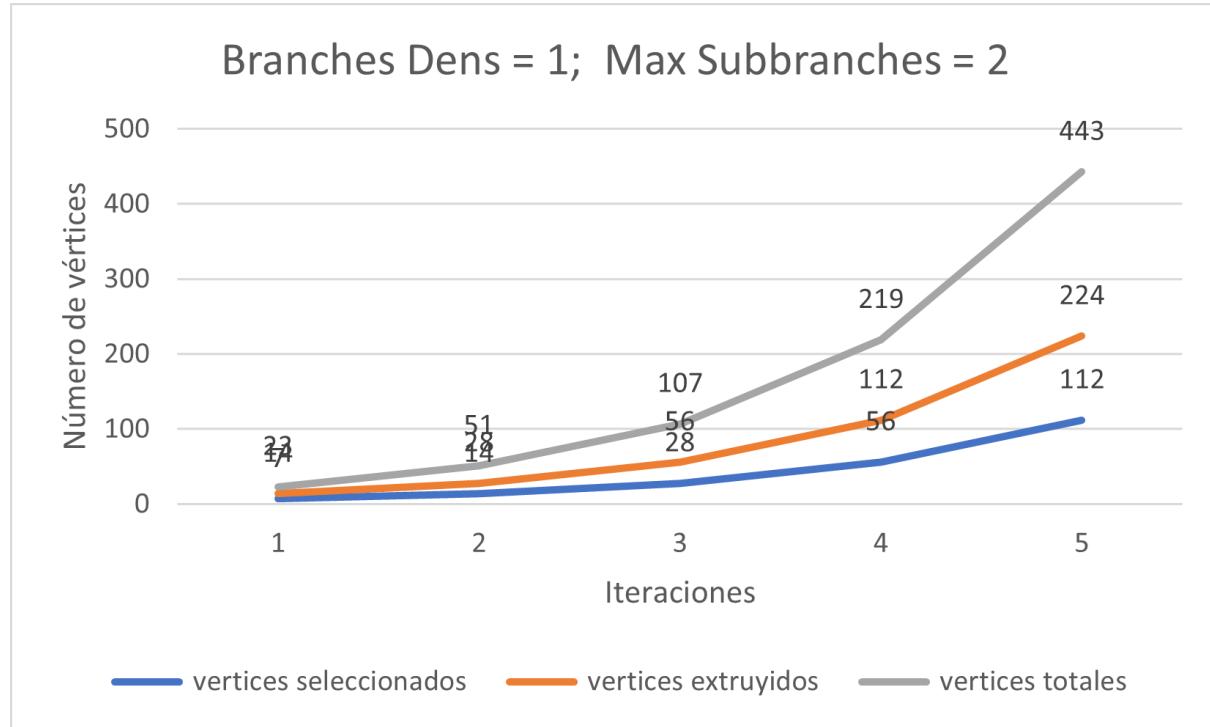


Figura 5.75: Valores de los vértices con densidad del 100 % y max subbranches = 2

En las imágenes 5.70 y 5.71, el crecimiento es lineal ya que en cada iteración el número de vértices extrudidos es constante debido a que siempre se extruyen el tantos vértices como se seleccionan, no obstante, con MAX\_SUBBRANCHES = 2 ya empieza a notarse un crecimiento exponencial en el número de vértices totales tras cada iteración.

Con MAX\_SUBBRANCHES = 3

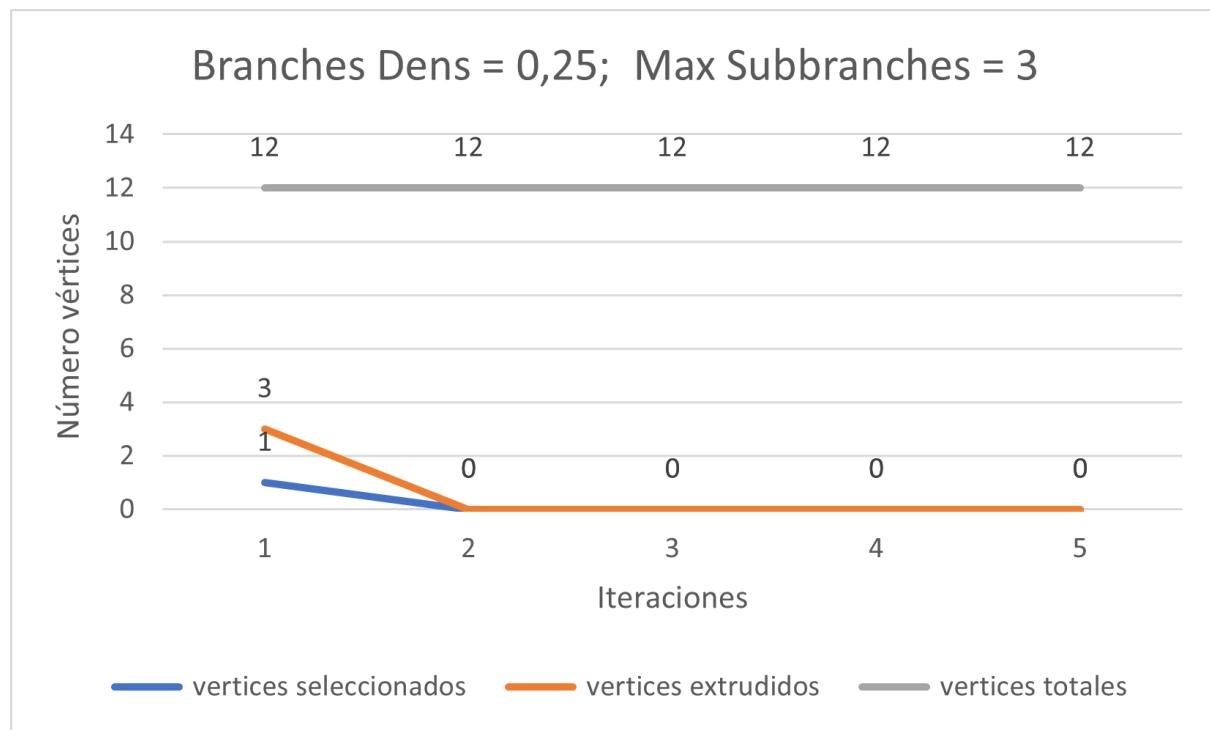


Figura 5.76: Valores de los vértices con densidad del 25 % y max subbranches = 3

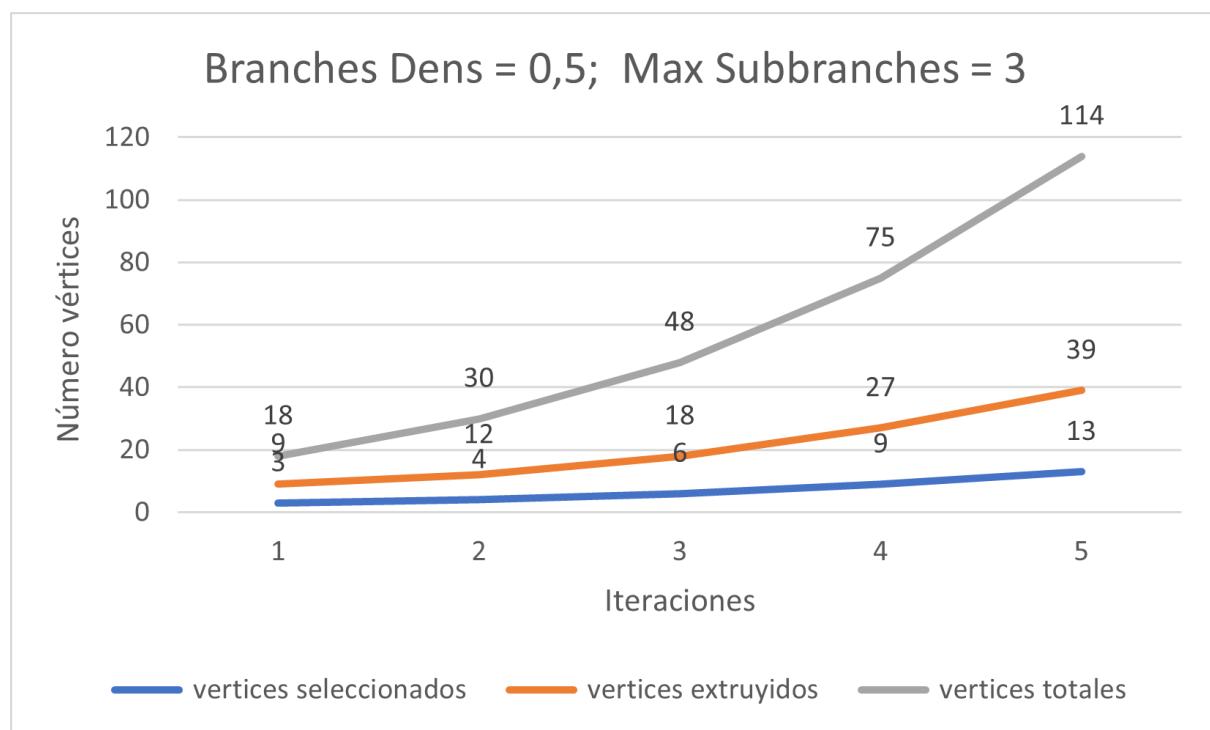


Figura 5.77: Valores de los vértices con densidad del 50 % y max subbranches = 3

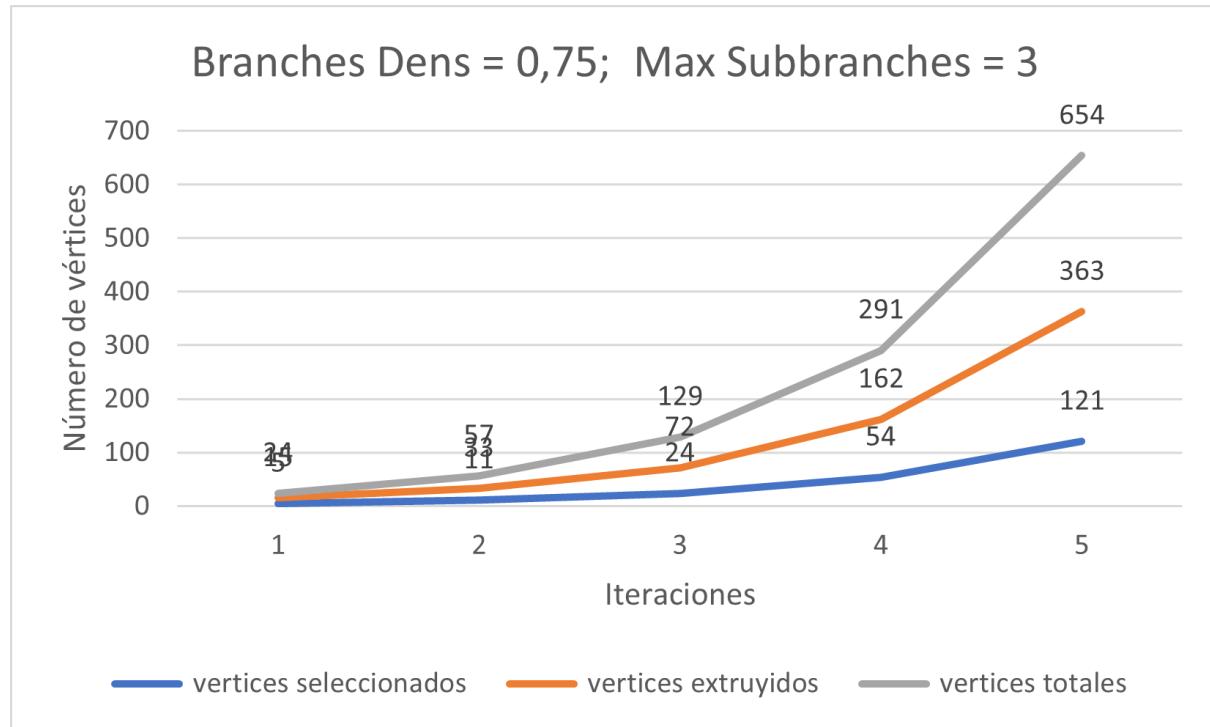


Figura 5.78: Valores de los vértices con densidad del 75 % y max subbranches = 3

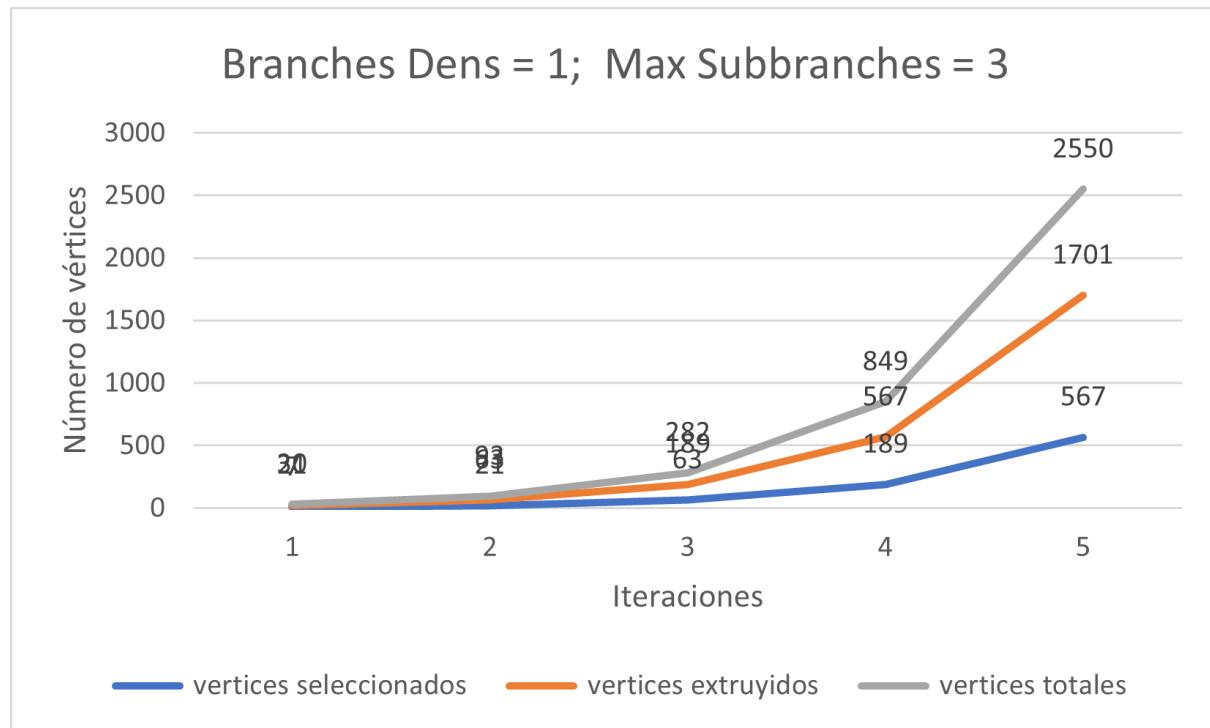


Figura 5.79: Valores de los vértices con densidad del 100 % y max subbranches = 3

Por último, en la prueba con MAX\_SUBBRANCHES = 3 ya se aprecia claramente el coste exponencial del algoritmo. Para la generación del rayo a partir de la densidad de 0.5 ya se requería de una cantidad de cálculo bastante notable que al llegar a los dos últimos ejemplos era enorme, llegando a necesitar varios minutos para generar el rayo.

### 5.1.2.2. Pruebas de rendimiento del efecto de transición wireframe

Como se dijo al comienzo del apartado de las pruebas de rendimiento, el principal problema de este efecto es que el frame rate al iniciar una animación puede bajar bastante si la cantidad de vértices del objeto al que se aplica es alta. A continuación, he hecho un estudio de la caída del frame rate en función del número de vértices del objeto al que es aplicado el efecto.

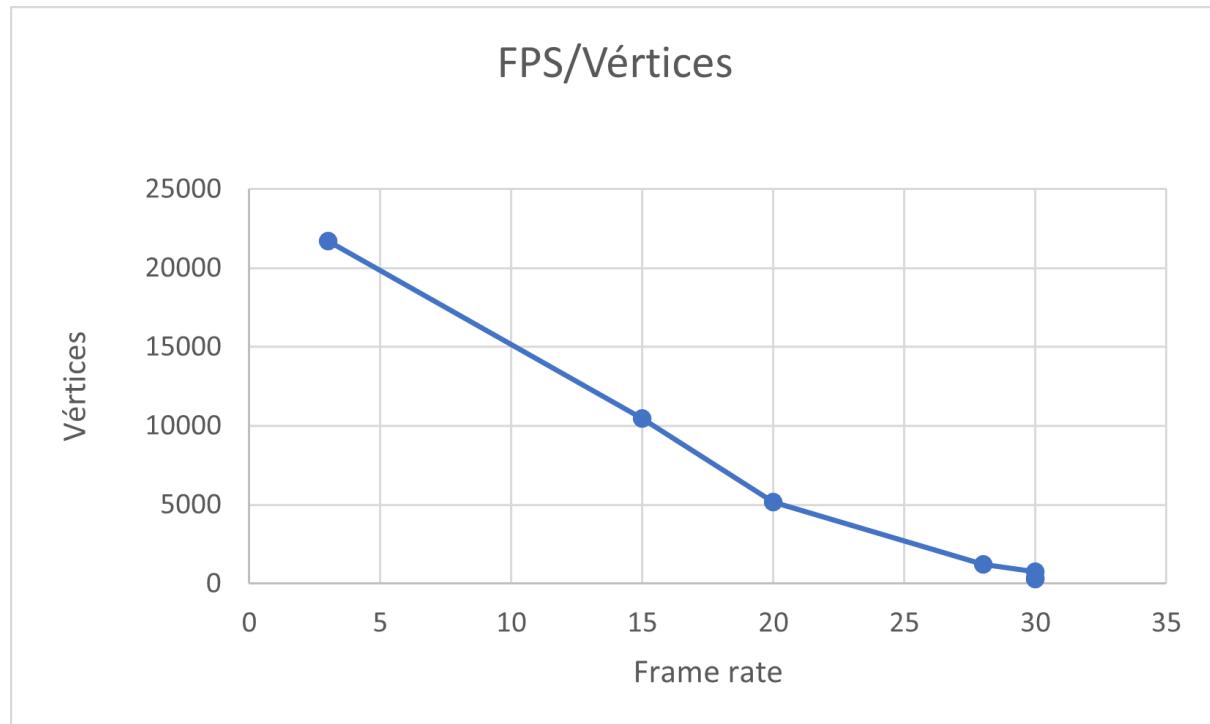


Figura 5.80: Gráfica que muestra la caída del frame rate según aumenta el número de vértices

### 5.1.2.3. Pruebas de rendimiento de los efectos de láser y escudo de fuerza

Estos dos efectos son los más eficientes de los 4 implementados hasta ahora, ambos efectos generan efectos, los cuales no necesitan una gran capacidad de cómputo para ser generados pues no están implementados con algoritmos que impliquen bucles ni operan con el número de vértices de ningún objeto, simplemente son creados con algoritmos no costosos y no realizan ninguna interacción después. Además de eso, la única memoria que ocupan es la de las texturas de los objetos creados con lo cual la memoria que consumen es mínima y tampoco es objeto de análisis. Por estos motivos no se han necesitado realizar pruebas de rendimiento a estos efectos.

## 5.1.3. Pruebas de usabilidad

Para las pruebas de usabilidad se ha pedido a 10 usuarios que internaran generar los diferentes efectos y modifiquen sus parámetros. También que insertaran keyframes y los borran y trataran de hacer una animación sencilla con ellos. Una vez realizada las tareas encomendadas se les pasó una encuesta que debían llenar la cual constaba de 22 preguntas:

1. ¿En qué rango de edad estás?
2. ¿Qué sexo eres?
3. ¿Has trabajado o tienes alguna experiencia en el ámbito de la animación y VFX?
4. Si la respuesta ha sido afirmativa, ¿qué experiencia tienes en ese ámbito?
5. Si has contestado afirmativamente a la pregunta anterior, ¿qué tipo de perfil eres?
6. ¿Tienes experiencia en el uso de tecnologías para animación o VFX?
7. Si la respuesta a la anterior pregunta ha sido afirmativa indica cuáles.
8. Si has utilizado Blender, ¿Qué experiencia tiene usándolo?
9. ¿Qué versión de Blender has usado?
10. ¿Consideras intuitiva la interfaz?
11. ¿Consideras que los nombres de los parámetros son intuitivos y expresan con claridad lo que hacen?
12. ¿Te ha resultado complicado usar el add-on?
13. Si la respuesta anterior ha sido afirmativa indica los motivos.
14. ¿Cuál ha sido el efecto que te ha parecido que da o puede dar mejores resultados? ¿Por qué?
15. ¿Crees que esta herramienta sería útil para producciones de animación y/o VFX?
16. ¿Has tenido algún problema generando algún efecto?
17. Si la respuesta anterior ha sido afirmativa indica cuáles.
18. ¿Consideras que los resultados obtenidos han sido esperables según los parámetros introducidos?
19. Si la respuesta anterior ha sido negativa indica cuáles han sido y los motivos.
20. ¿Cómo evaluarías el proceso de generación y edición de los efectos generados?
21. ¿Cómo evaluarías en general la experiencia de uso del add-on?
22. Por favor, si tienes algún comentario que te gustaría hacer, o alguna idea para que el add-on sea mejor y te gustaría compartirlo, hazlo libremente ;)

Estas han sido las preguntas realizadas en la encuesta y las respuestas obtenidas han sido las siguientes:

1. 10 % entre 13-17 años, 60 % entre 18-25 años, 10 % entre 25-35 años, 20 % más de 35 años.
2. El 50 % de los encuestados han sido mujeres y el otro 50 % hombres.

3. El 30 % de los encuestados ha dicho: Sí, estoy familiarizado y tengo experiencia; el 20 % ha dicho: No; y el 50 % ha dicho: tengo nociones y tengo un poco de experiencia en ese ámbito.
4. El 10 % tenía experiencia laboral únicamente, el 80 % únicamente estudiantil y el 10 % tenía ambas.
5. Un 10 % tenía un perfil artístico y un 90 % un perfil técnico.
6. Un 80 % ha dicho tener experiencia en el uso de tecnologías y un 20 % no.
7. Las tecnologías que han sido utilizadas por los usuarios han sido Blender, Maya, 3D Studio Max, Houdini, Nuke y Unity.
8. El 30 % de los encuestados ha dicho: No tengo apenas experiencia; el 20 % ha dicho: Estoy familiarizado pero no tengo un gran manejo; y el 40 % ha dicho: Tengo algo de experiencia y he realizado varios trabajos; y el 10 % ha dicho: Conozco bien Blender y soy capaz de realizar proyectos de calidad con este software.
9. Todas las versiones de Blender han sido de la 2.8 en adelante, habiendo 2 usuarios que han usado la versión 2.92 y la 2.93.
10. Un 90 % ha respondido que sí y un 10 % que no.
11. Un 70 % ha respondido que sí y un 30 % que no.
12. Un 80 % ha respondido que sí y un 20 % que no.
13. Algunas de los motivos por los que le ha sido a algunos usuarios usar el add-on han sido que no encontraban la ubicación del add-on en la interfaz, no sabían como modificar parámetros de los objetos de los efectos o que no eran capaces de generar alguno de los efectos por no leer o no entender el manual. También ha influido la versión que usaban de Blender, pues los que tenían una versión más desactualizada no les ha sido posible realizar algún efecto.
14. El efecto de rayo ha sido el que más ha gustado a los usuarios con un 70 %, después el efecto del escudo con un 20 % y el efecto del láser con un 10 %. El efecto de transición de wireframe ha sido el más problemático para algunos usuarios y el que menos ha gustado.
15. El 90 % de los encuestados ha dicho que sí le parece una herramienta útil para producciones, mientras que el 10 % cree que a algunos efectos les falta calidad.
16. El efecto que más problemas ha dado, como adelantaba ha sido el de transición de wireframe, un 70 % ha tenido problemas con este efecto, aunque la mayoría no había leído el manual. Un 10 % ha tenido problemas con el escudo.
17. Algunos usuarios no se daban cuenta que para generar el efecto era necesario tener seleccionado un objeto al que aplicárselo. Tampoco llegaron a entender que para hacer que el efecto se produjese había que mover los manejadores manualmente. En cuanto al efecto del escudo, algunos encuestados se vieron sorprendidos al ver que cuando generaban el escudo no aparecía nada y es que el escudo aparece con una escala de 0, es decir, reducido a nada, porque se crea con una animación por defecto de escalado que comienza con un valor de 0.

18. El 100 % de los usuarios ha contestado que los resultados cono acordes a los parámetros introducidos.
19. Ningún encuestado ha llegado a contestar esta pregunta.
20. Las valoraciones de la interfaz y la generación de efectos se evaluaban con una puntuación d el 1 al 10. La media a esta pregunta ha sido de 8,67, es decir, la nota media de los usuarios al proceso de generación y edición de los efectos ha sido muy positiva.
21. Las valoraciones generales del uso del add-on también se evaluaban con una puntuación d el 1 al 10. La media a esta pregunta ha sido de 8,33 también ha sido muy positiva.
22. Algunos de los comentarios que han dejado los encuestados han sido que al hacer ctrl-z les ha llegado a colapsar y cerrar Blender, debido a que hay algunas operaciones que no es posible deshacer y que se realizaron cuando se genera el efecto y que al intentar deshacer el efecto, intentar deshacer esta operación hace que colapse Blender. Otro comentario decía que el rayo con un valor alto de densidad hacía que Blender no respondiera, y es que debido al coste del algoritmo, si se abusa del valor de los parámetros esto puede suceder si los valores son muy altos en densidad y subramas.

En conclusión, y a la vista de las valoraciones de los usuarios, el add-on parece ser una herramienta que les ha sido fácil de usar a la mayoría de usuarios y que puede ser usada en producciones de VFX.



# Capítulo 6

## Conclusiones

### 6.1. Conclusiones

Para concluir creo que la propuesta de este proyecto es una propuesta que supone una solución pragmática para la agilización de producciones de animaciones que requieran de cualquier efecto que se pueda desarrollar mediante script, además de los ya implementados, y que además, permita realizarlos a profesionales que no posean suficiencia experiencia en animación, VFX o que simplemente no tengan todavía los conocimientos necesarios en Blender lo que lo hace una herramienta de buen uso para estudiantes del mundo de la animación y VFX, una faceta de este add-on que se ve favorecida por ser un add-on gratuito de código abierto. También cabe decir que la realización de este proyecto ha supuesto un reto personal por la poca experiencia que poseía en Blender, al menos en los campos necesarios para la realización de varios efectos como son la composición de materiales mediante nodos, el manejo de modificadores, edición de meshes, etc. que ha supuesto que a cada paso tuviera que frenar para informarme sobre como realizar determinadas tareas o investigar el funcionamiento de ciertas herramientas de Blender. También ha supuesto un reto en cuanto a la exigencia, pues es el primer proyecto que llevo a cabo en todas sus fases, desde el estudio preliminar, análisis, diseño, implementación, pruebas, etc, siendo este un proyecto con muchas cosas nuevas, que en algunos momentos ni siquiera tenía claro si las iba a poder manejar por mi inexperiencia y por el tiempo que me supondría ello; por ser un proyecto al que le he podido dedicar menos tiempo del que en un principio pensé que le iba a poder dedicar, al menos durante los meses del curso en sí, y por ser un proyecto largo sin un final claro establecido ya que una herramienta de las características que tiene esta, una "librería" de efectos, puede tener tantos efectos como uno esté dispuesto a implementar y estos, a su vez, pueden ser tan complejos o completos como uno esté dispuesto a llegar.

La calidad de la herramienta considero que es buena, los resultados se pueden utilizar dando resultados bastante satisfactorios en una producción y creo que como herramienta para principiantes en Blender y el mundo de los VFX también es muy útil. Los costes económicos no son elevados, sobre todo si, como en este caso, es el desarrollador el que realiza todas las funciones necesarias para la realización del proyecto. En cuanto al coste temporal, ya se ha visto en el apartado de planificación temporal que el proyecto es largo, debido a la cantidad de problemas que pueden surgir, la cantidad de efectos que hay, que aunque parece moderada, para el tiempo dedicable a este proyecto ha sido bastante

grande, y a la inexperiencia en este tipo de herramientas y VFX en general. Por otra parte, creo que los resultados han sido satisfactorios y ha sido un proyecto muy instructivo como primer proyecto de este ámbito. Los efectos han tenido buenas valoraciones en las pruebas realizadas por los usuarios y el uso de la herramienta ha cumplido con la gran mayoría de objetivos que se plantearon al principio. Por ello, para concluir, creo que el add-on, en general, ha cumplido su objetivo que era el de desarrollar un herramienta que facilitara la producción de contenido multiemdia, en este caso, a través de los VFX, respondiendo a la demanda de este tipo de contenido, la cual continúa en auge, y que sirve como otra herramienta para los desarrollados en Blender, los cuales también han aumentado en número siguiendo el auge de la demanda de la multimedia y de desarrolladores de este software que se ve favorecido por el crecimiento del software de código abierto.

## 6.2. Trabajo futuro

En cuanto al trabajo futuro de este proyecto la dirección es clara: continuar programando efectos y añadiéndolos a la herramienta, si esta llegase a tener un número demasiado elevado de efectos hacer varios add-ons que agrupe varios efectos similares y agrupar estos add-ons en kits que congreguen librerías de efectos, creando una librería de efectos personalizados, hechos por desarrolladores expertos y que pueda ser usada como herramienta para estudios de animación que concentre efectos propios de manera que puedan usarlos rápidamente y reproducirlos tantas veces como quieran sin necesidad de estar elaborándolos manualmente cada vez. También se mejorarán los efectos ya implementados para que sean más versátiles y completos, con un mejor rendimiento y más fáciles de usar. Además, se tratará de hacer que estos efectos utilicen más medios ofrecidos por Blender de los que usan actualmente, ya que la inexperiencia ha limitado bastante el abanico de posibilidades que se podían usar para desarrollar mas efectos. Por último, y a la vista del estudio del estado del arte realizado, no se han encontrado herramientas como esta en el sentido de ser librerías que generen efectos, ni siquiera en otros softwares, por lo que también existe la posibilidad de hacer de VFX Tool una especie de marca y hacer versiones para otros softwares de la industria como 3D Max o Maya.

# Capítulo 7

## Anexo

### 7.1. Planificación temporal y diagrama de Gantt

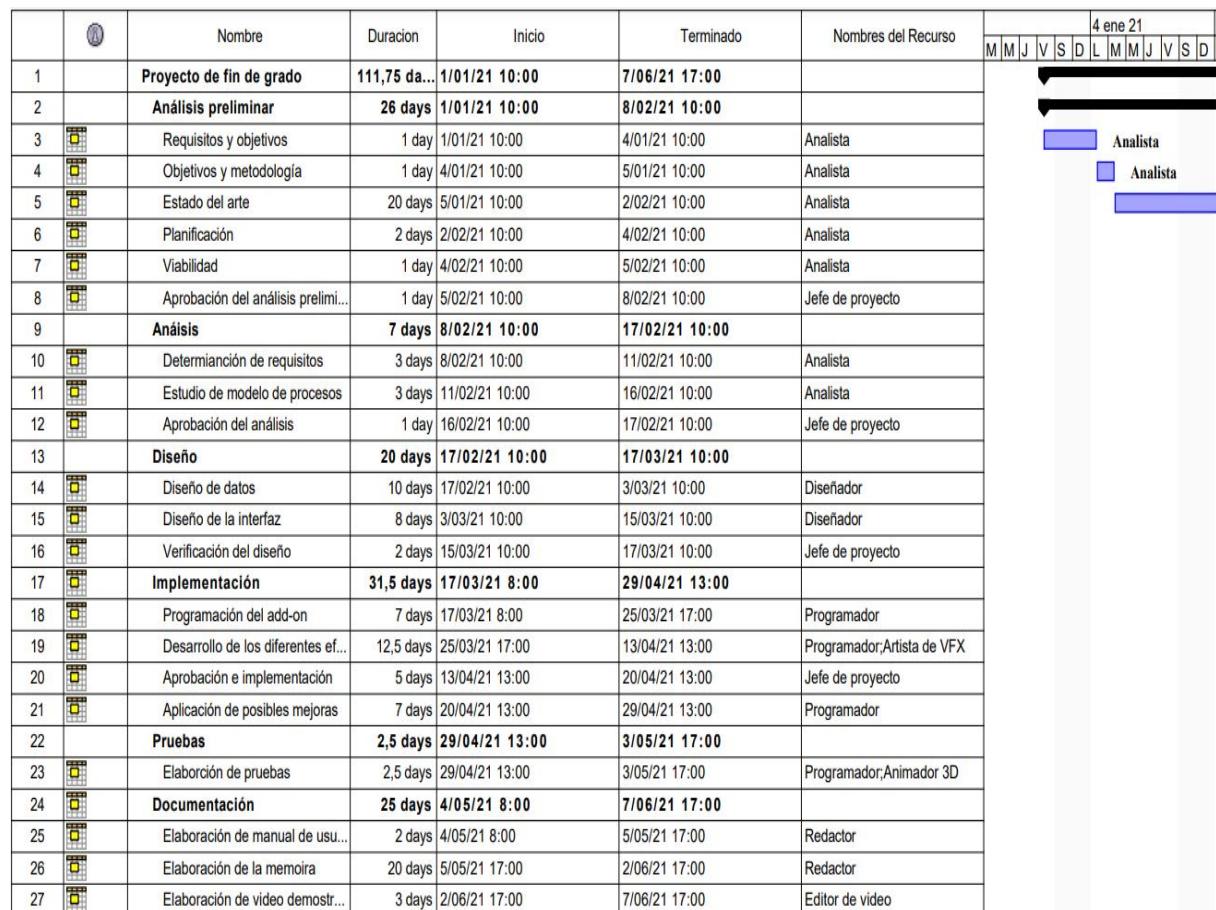


Figura 7.1: Planificación temporal y diagrama de Gantt (1)

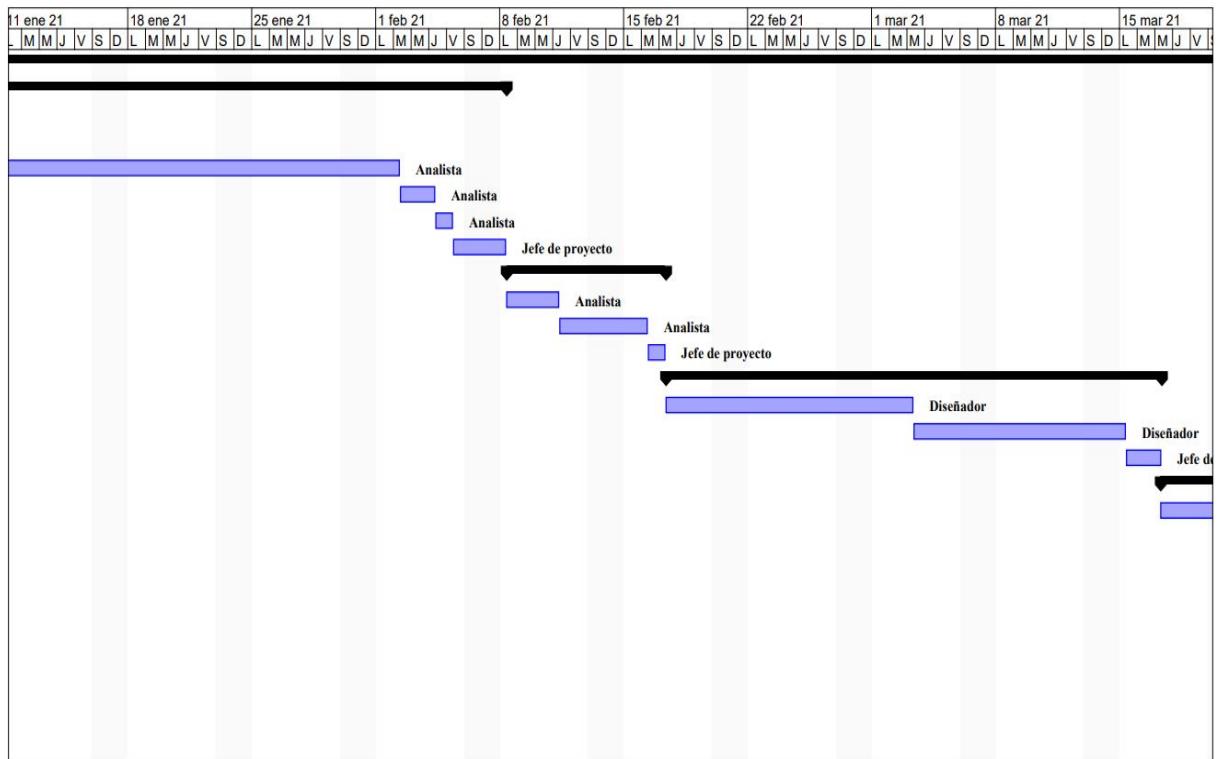


Figura 7.2: Diagrama de Gantt (2)

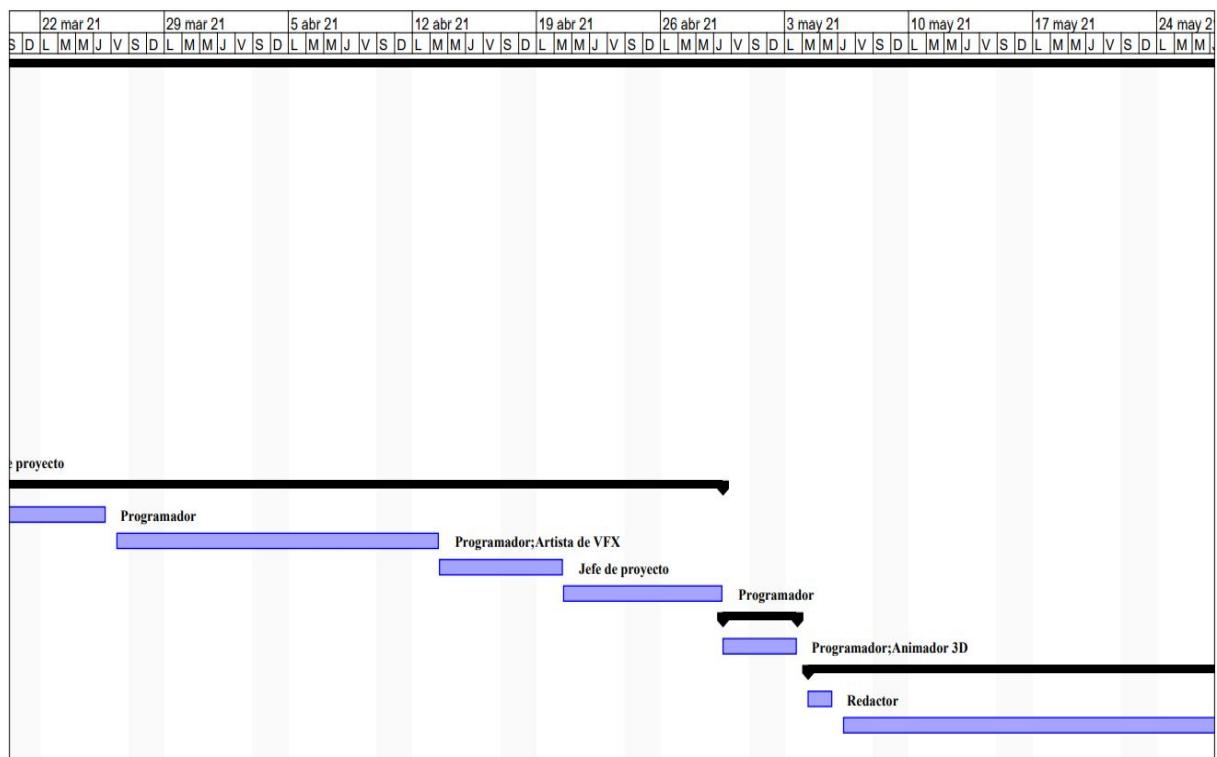
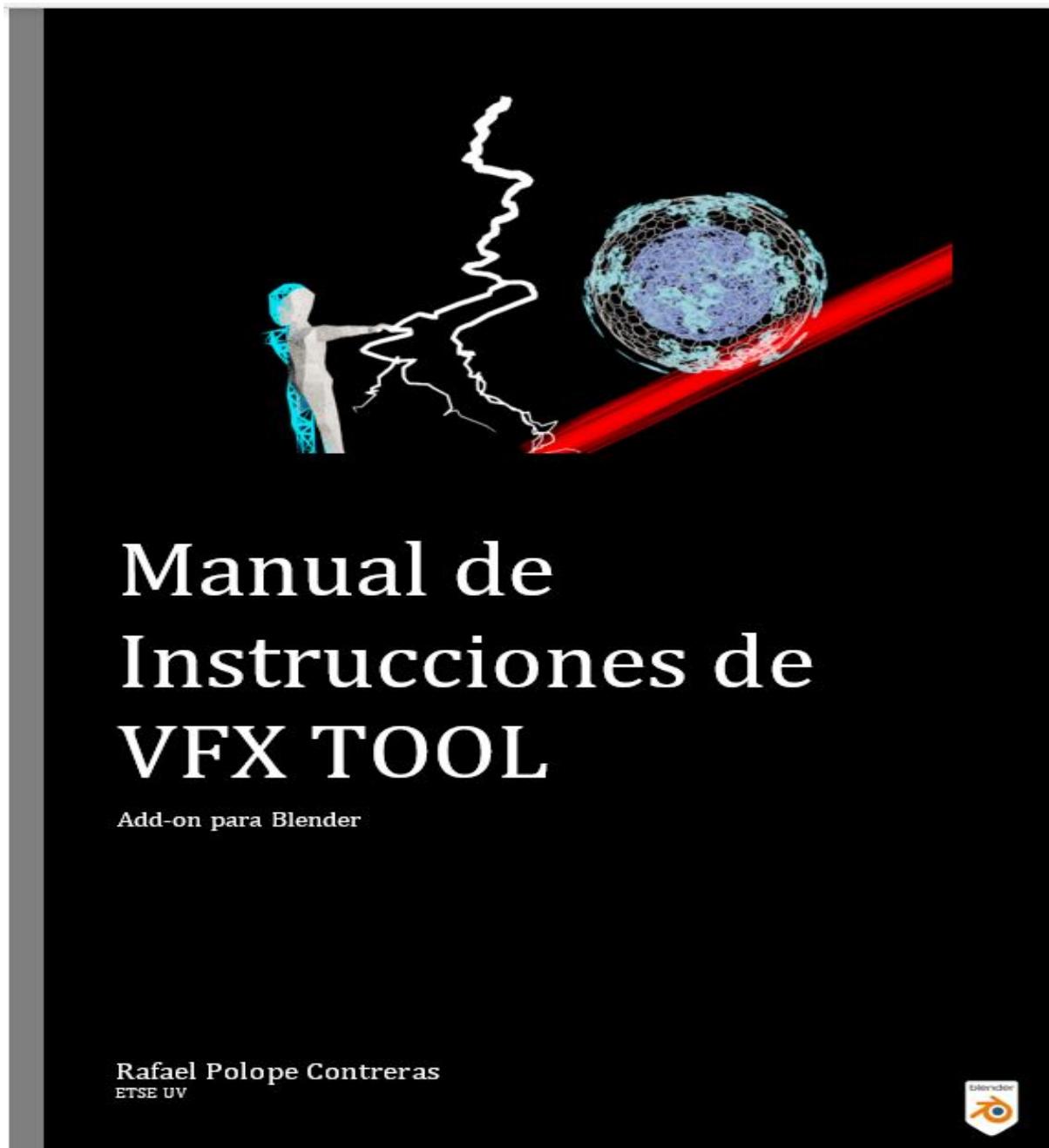


Figura 7.3: Diagrama de Gantt (3)

## 7.2. Manual de instrucciones de VFX Tool



### 7.2.1. Efecto del rayo

Para generar este efecto se ha de abrir el panel de este efecto (el primero) en el panel principal del add-on. Una vez abierto el panel se deberán llenar los parámetros del panel con los valores adecuados para generar el rayo como desee. A continuación, se explican el significado los parámetros:

### 7.2.1.1. Parámetros

- **Length:** Indica la longitud que tendrá el rayo en el momento de su creación.
- **Strength:** Indica la energía de la luz que emitirá el rayo.
- **Opacity:** Indica como de transparente será el rayo. Sirve para hacer que el rayo aparezca de manera instantánea sin que se modifique el parámetro strength. Es un valor entre 0 y 1.
- **Branches Length:** Indica como de largos queremos que sean los subrayos que surgen.
- **Max Subbranches:** Este parámetro indica cuál queremos que sea el número máximo de subrayos que queremos que surja de cada subrayo, incluido el rayo principal.
- **Branches Dens:** Este parámetro expresa cuantos subrayos, del número de subrayos máximo, queremos que surjan del rayo o cada subrayo. Es un valor entre 0 y 1: 0 es equivalente a tener un valor de subrayos máximo de 0 y 1 hará que surjan tantos subrayos como Max Subbranches indique.
- **Branches Depth:** Este parámetro es uno de los más fundamentales y complicados de este efecto. Expresa el número de niveles de subrayos que queremos que haya, en otras palabras, expresa el nivel de recursividad. La explicación de su papel es la siguiente: por cada nivel habrá un número de subrayos de los cuales surgirán nuevos subrayos. Estos nuevos subrayos surgirán en un número NO superior a Max Subbranches. El número de subrayos que surjan se hará más pequeño conforme se suba de nivel. Si el parámetro Branches Dens es pequeño puede que dejen de haber niveles de subrayos antes de llegar al número de niveles de Branches Depth, por tanto, Branches Depth es un cota superior o límite máximo de niveles de subramas. El valor de Max Subbranches también determinará el número de niveles que habrá pues, si es pequeño, surgirán pocos subrayos por nivel y esto hará que la disminución progresiva del número de rayos que surgen por nivel acabe antes con los niveles por rayo.
- **Bolt Thiness:** Indica cómo de delgado será el rayo. Cuanto mayor sea este valor más fino será el rayo. Este parámetro puede dar resultados no deseados ya que si el valor de Bolt Thiness es muy alto puede que las ramas se hagan demasiado finas las ramas, incluso antes de sus extremos y no se lleguen a ver por completo. Se debe tener en cuenta que este parámetro actúa sobre los vértices extremos del rayo y sus subramas, por lo que cuantos más tengamos menor será el valor necesario para obtener la delgadez que queremos (Valores altos con muchos vértices dan resultados no deseados generalmente). También cabe decir que este parámetro es relativo a la altura del rayo pues tiene una anchura máxima que es dada por el modificador que crea la envoltura del rayo y que no se puede sobrepasar, de manera que cuanto más largo sea el rayo menos se apreciará el grosor ya que la relación grosor/altura será menor.
- **Sharpness X:** Sirve para definir cómo de pronunciadas queremos que sean las irregularidades del rayo en la dirección X.
- **Sharpness Y:** Sirve para definir cómo de pronunciadas queremos que sean las irregularidades del rayo en la dirección Y.

- **Sharpness Z:** Sirve para definir cómo de pronunciadas queremos que sean las irregularidades del rayo en la dirección Z.
- **Color:** Indica el color del rayo.
- **Begin:** Punto de comienzo del rayo.
- **End:** Punto de finalización del rayo.

Una vez definidos los parámetros de creación del rayo, si queremos modificar alguno de los parámetros con los que creamos inicialmente el efecto o animar el rayo, sólo tendremos que seleccionar el rayo que hemos creado y clicar sobre el panel de objetos (el panel con el icono del cuadrado naranja) en la parte derecha de la interfaz de Blender. Ahí, tendremos un subpanel con el nombre de “Bolt properties”, el cual contendrá algunos de los parámetros y los valores que introdujimos a la hora de crear el rayo. En el caso del rayo son: length, opacity, strength, color, sharpness X y sharpness Y. Modificando estos parámetros podremos ver que efecto tienen sobre el rayo en ese momento.

#### 7.2.1.2. Animación del rayo

Si queremos animar las propiedades del rayo mencionadas (length, opacity, strength, etc.) sólo tendremos que clicar sobre el botón de insertar keyframes o borrar keyframes que aparece en el mismo panel que las propiedades. Cuando se clique sobre el botón de insertar keyframes se insertarán keyframes sobre las propiedades que hayan sido modificadas, por lo que si queremos insertar un keyframe en una propiedad con el mismo valor que tenía tendremos que introducir el mismo valor que ya tenía esa propiedad para poder insertar el keyframe. Para borrar un keyframe sólo tendremos que situarnos en el frame donde está insertado el keyframe y este se borrará al clicar el botón de borrado. De otra manera, si no se introduce manualmente el mismo valor que tenía una propiedad, esta no podrá tener dos keyframes seguidos con el mismo valor. Los keyframes se insertan y se borran en el frame que indica la línea de tiempo.

**Importante:** Las pruebas se han realizado con un rayo de altura 10, por lo que con valores pequeños de sharpness se puede conseguir una forma de rayo realista, no obstante, si queremos reproducir esta forma con un rayo de altura mayor (50 metros o más) quizás se deba mover los vértices de la geometría manualmente para dar la forma curva que se quiera, ya que la forma dada por los modificadores de desplazamiento, dan la forma de zigzag propia de los rayos, pero aparte de esta forma de zigzag, los rayos pueden seguir una trayectoria curva. Cuando el rayo es de poca altura los parámetros sharpness permiten recrear tanto la forma en zigzag como la trayectoria curva, pero a medida que el rayo aumenta de tamaño esto deja de ser así, por lo que se deberá modificar manualmente o generarla en pequeño y escalarlo.

El rayo funciona de manera óptima si es vertical y la dirección del rayo es hacia abajo, si no es así, en caso de que la longitud de los subrayos o su cantidad sea grande, puede tener ligeros fallos (pueden aparecer antes las puntas de los subrayos que el cuerpo principal del rayo).

### 7.2.2. Efecto de transición wireframe:

Para generar este efecto se ha de abrir el panel de este efecto (el segundo) en el panel principal del add-on y seleccionar el objeto al que queramos aplicar el efecto. Una vez abierto el panel se deberán llenar los parámetros del panel con los valores adecuados para generar el efecto como deseé. A continuación, se explican el significado los parámetros:

#### 7.2.2.1. Parámetros

- **Width:** Indica la anchura de la zona de wireframe que se va a ver durante la transición.
- **Strength:** Indica la energía de la luz que emitirá el wireframe.
- **Color:** Indica el color del wireframe.
- **Wire thickness:** Grosor del wireframe.

#### 7.2.2.2. Animación del rayo

Para realizar el efecto de transición de wireframe tendremos que mover unos manejadores (dos semi-icosferas), los cuales controlarán el efecto acercándolos o alejándolos del objeto que tenga aplicado el efecto. Al acercar los manejadores, la transición de wireframe sucederá haciéndolo desaparecer dejando una zona con el wireframe visible. Si queremos animar el efecto de transición de wireframe, podremos insertar keyframes en las propiedades del objeto que posea el efecto aplicado para que estas varíen sus valores entre los frames en los que insertemos los keyframes, al igual que con el efecto del rayo. Para hacer que suceda la transición, será necesario mover los manejadores del efecto y animarlos insertando keyframes de posición. También es recomendable emparentar el manejador del efecto (la primera semi-icosfera) al objeto al que se aplicó la transición si se desea aplicar este efecto sobre un objeto animado, pues así, el manejador acompañará al objeto animado al que se aplique y la transición se realizará sólo cuando el usuario quiera, no cuando se mueva el personaje y, por no estar emparentado, se aleje del controlador, haciendo que este actúe sobre el objeto.

**Importante:** Este efecto crea un **DUPLICADO DEL OBJETO** al que se aplicó el efecto para poder llevarlo a cabo. Si queremos aplicar este efecto sobre un objeto riggeado y con animación, deberemos realizar primero la animación y después aplicar el efecto.

Este efecto crea un duplicado del objeto al que se le aplica, si este objeto posee mucha geometría, puede **BAJAR** considerablemente **EL FRAME RATE** debido a que debe realizar operaciones de comparación de proximidad de cada vértice con el manejador por duplicado, por lo que no es recomendable para objetos con mucha geometría.

Este efecto se aplica **SÓLO A UN OBJETO** cada vez, eso quiere decir que si queremos aplicar el efecto a un objeto compuesto por varias mallas (meshes) tendremos que aplicar el efecto a cada malla por separado y sincronizarlo manualmente para que sea uniforme en todo el objeto.

### 7.2.3. Efecto de rayo láser:

Para generar este efecto se ha de abrir el panel de este efecto (el tercero) en el panel principal del add-on. Una vez abierto el panel se deberán llenar los parámetros del panel con los valores adecuados para generar el láser como deseé. A continuación, se explican el significado los parámetros:

#### 7.2.3.1. Parámetros

- **Beam Maximum Radius:** Indica el radio máximo que podrá alcanzar el láser.
- **Beam Radius:** Indica el radio del láser, el cual estará entre 0 y el radio máximo definido en Beam Maximum Radius. Es un valor entre 0 y 1.
- **Beam Strength:** Indica la energía de la luz que emitirá el láser.
- **Beam Color:** Indica el color del láser.
- **Renderer:** Este parámetro sirve para indicar cual será el motor de renderizado que se usará para renderizar la animación (Cycles o Eevee), pues dependiendo de si se usa uno u otro, se generará un láser u otro, ya que el láser para Cycles es un láser creado a partir de un material volumétrico, que en Eevee no se puede visualizar bien. Gracias a este parámetro podremos tener un láser visualizable con ambos motores de render.
- **Beam Noise:** Este parámetro sólo será visible si se ha elegido Cycles para generar el láser. Indica la nitidez del láser. Es un valor entre 0 y 1. Si este valor es bajo el láser será recto y uniforme, si es alto el láser será más borroso y disperso.
- **Animation:** Este checkbox indica si queremos que el láser se anime dando el efecto de que esta siendo disparado y que el láser está corriendo.
- **Aniamtion Vel:** Indica la velocidad de animación del láser. Este parámetro sólo estará habilitado si el checkbox de Animation está marcado.
- **Begin:** Indica la posición de comienzo del láser.
- **End:** Indica la posición de fin del láser.

#### 7.2.3.2. Animación del rayo láser

Para hacer que el láser aparezca desde el comienzo indicado en el parámetro Begin, hasta el final indicado en End tendremos que hacerlo insertando keyframes de escalado. El láser tiene su origen de coordenadas en Begin (al comienzo del láser), por lo que si escalamos el rayo se alargará en la dirección que indique el vector Begin-End. Para hacer que el rayo láser aparezca como si fuese disparado basta con insertar un keyframe con un valor de escalado de 0 y otro más adelante con el valor 1. Así, podremos recrear el disparo del láser. Si queremos animar las propiedades del láser, podremos insertar keyframes en las propiedades del mismo para que estas varíen sus valores entre los frames en los que insertemos los keyframes, al igual que con el efecto del rayo y el de transición de wireframe.

### 7.2.4. Efecto de escudo de fuerza:

Para generar este efecto se ha de abrir el panel de este efecto (el cuarto) en el panel principal del add-on. Una vez abierto el panel se deberán llenar los parámetros del panel con los valores adecuados para generar el láser como deseé. A continuación, se explican el significado los parámetros:

#### 7.2.4.1. Parámetros

- **Shield Strength:** Indica la energía de la luz que emitirá el escudo principal.
- **Shield Color:** Indica el color del escudo principal.
- **Shield Noise:** Este parámetro indica la cantidad de ruido del material del escudo, es decir, las manchas en el color de este. Es un valor entre 0 y 1. Si este valor es bajo el escudo tendrá un color uniforme, si es alto el escudo tendrá más manchas.
- **Animation:** Este checkbox indica si queremos que el material del escudo principal se anime dando el efecto de que está rotando alrededor del eje Z de Blender.
- **Shield Material Aniamtion Vel:** Indica la velocidad de animación del material del escudo. Este parámetro sólo estará habilitado si el checkbox de Animation está marcado.
- **Use Wirefame:** Este parámetro indica si queremos que se genere el escudo de wireframe que acompaña al principal.
- **Wireframe Color:** Indica el color del escudo de wireframe. Este parámetro sólo será visible si el checkbox Use Wireframe está marcado.
- **Wireframe Strength:** Indica la fuerza de emisión del escudo de wireframe. Este parámetro sólo será visible si el checkbox Use Wireframe está marcado.
- **Wireframe Thickness:** Indica la anchura del wireframe del escudo de wireframe. Este parámetro sólo será visible si el checkbox Use Wireframe está marcado.
- **Wireframe Animation Delay:** Indica el retardo de la animación de wireframe respecto a la del escudo principal. Este parámetro sólo será visible si el checkbox Use Wireframe está marcado.
- **Wireframe Animation Delay:** Indica el retardo de la animación de wireframe respecto a la del escudo principal. Este parámetro sólo será visible si el checkbox Use Wireframe está marcado.
- **Use Fragments:** Este parámetro indica si queremos que se genere el escudo de fragmentos que acompaña al principal.
- **Fragments Color:** Indica el color del escudo de fragmentos. Este parámetro sólo será visible si el checkbox Use Fragments está marcado.
- **Fragments Strength:** Indica la fuerza de emisión del escudo de fragmentos. Este parámetro sólo será visible si el checkbox Use Fragments está marcado.

- **Fragments Animation Delay:** Indica el retardo de la animación de fragmentos respecto a la del escudo principal. Este parámetro sólo será visible si el checkbox Use Fragments está marcado.

#### 7.2.4.2. Animación del escudo de fuerza

El escudo de fuerza ya se genera con una animación por defecto, la cual se puede desplazar a lo largo de la timeline como se deseé. El parámetro Animation Delay indica el retardo de las animaciones de los escudos de wireframe y fragmentos respecto a la animación del escudo principal. La animación es una animación de escalado, partiendo el escudo con un valor de escala de 0 en el frame inicial y una escala de 1 en el frame final de la animación, pero esto se puede modificar como usted deseé, incluso borrando los keyframes y escalándolo manualmente usted mismo, del mismo modo que puede hacer uso del parámetro Animation Delay para separar las animaciones del escudo de fuerza y wireframe o desplazar usted mismo los keyframes de cada escudo en la línea de tiempo si así lo prefiere\*. Si queremos animar las propiedades del escudo, podremos insertar keyframes en las propiedades del mismo para que estas varíen sus valores entre los frames en los que insertemos los keyframes, al igual que con efectos anteriores. Este efecto tiene una particularidad y es que, al generar tres escudos, y, por tanto, 3 objetos, en lugar de uno solo como los anteriores, para modificar las propiedades de los diferentes escudos habrá que clicar sobre ellos por separado para cambiarlas.

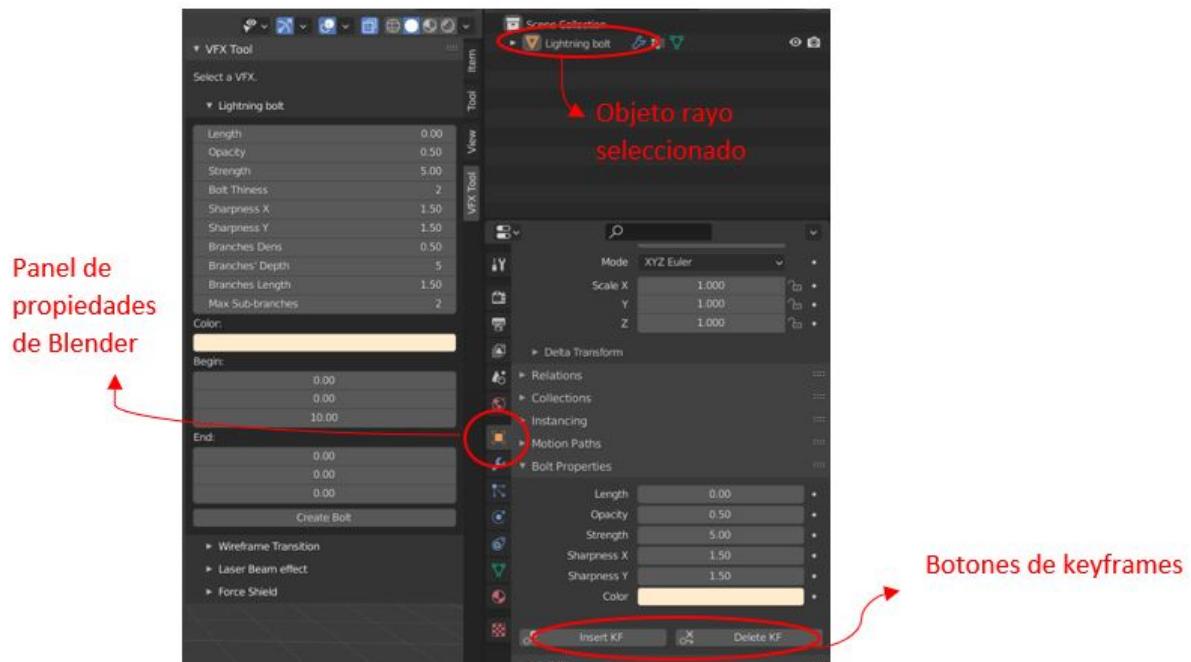


Figura 7.4: Posición de los paneles en la interfaz y objeto seleccionado



# Bibliografía

- [1] Wikipedia. Autodesk maya, 24 de abril de 2021.
- [2] Wikipedia. Autodesk maya, 12 de noviembre de 2017.
- [3] Wikipedia. Cinema4d, 30 de Enero de 2021.
- [4] BIMCommunity. Tools to create successful projects, 2021.
- [5] Maxon. Cinema 4d, software de animación, modelado, simulación y renderizado 3d por computadora, 2021.
- [6] Hugo Rodríguez. Cinema4d vs maya, 9 de febrero de 2020.
- [7] Wikipedia. Unreal engine, 28 de mayo de 2021.
- [8] Epic Games. Unreal engine, 2021.
- [9] Epic Games. Niagara visual effects, 2021.
- [10] Wikipedia. Nuke, 17 de Noviembre de 2020.
- [11] Foundry. Industry standard compositing, editorial and review, 2021.
- [12] Ejezeta. After effects vs nuke para componer, 8 de Noviembre de 2017.
- [13] Neil Bennett. The foundry nuke 5, 24 de abril de 2008.
- [14] Wikipedia. Photoshop, 24 de mayo de 2021.
- [15] Wikipedia. Zbrush, 24 de mayo de 2021.
- [16] Wikipedia. Affter effects, 10 de abril de 2019.
- [17] Adobe. Affter effects, 2021.
- [18] TrustRadius. Adobe after effects reviews, 26 de marzo de 2021.
- [19] Wikipedia. Autodesk combustion, 4 de febrero de 2021.
- [20] Shakuro. Autodesk combustion, 5 de junio de 2020.
- [21] Wikipedia. Blackmagic fusion, 8 de marzo de 2021.
- [22] Blackmagic Design Pty. Ltd. Fusion 17, 2021.
- [23] Wikipedia. Houdini, 18 de mayo de 2021.

- [24] SideFX. Houdini, 2021.
- [25] SparthanGeek. Houdini: Review del mejor software para modelado 3d del año, 27 de diciembre de 2018.
- [26] Wikipedia. V-ray, 14 de abril de 2021.
- [27] Itziar Garcés Loperena. Vray (que es) mas que un motor de renderizado, 13 de junio de 2018.
- [28] Wikipedia. Unity, 26 de mayo de 2021.
- [29] Unity Technologies. Visual effect graph, 2021.
- [30] Logic Simplified. Unity 3d game development: Advantages and disadvantages, 2020.
- [31] ArtistaPirata. Marmoset toolbag para render en tiempo real, 29 de diciembre de 2020.
- [32] Imagen y Sonido CEV, Escuela Superior de Comunicación. Especialízate en modelado con marmoset, el software profesional con el que aprenderás a renderizar en tiempo real, 26 de septiembre de 2019.
- [33] Wikipedia. Autodesk 3ds max, 4 de marzo de 2021.
- [34] H2A Comunicacion. 3d max vs maya: los pros y los contras de cada software de animación, 2020.
- [35] Wikipedia. Blender, 26 de junio de 2021.
- [36] Blender. Visual effects, 2021.
- [37] Blender3D. Blender: modelado y animación 3d gratis, 2020.
- [38] RodyPolis LLC. Actionvfx, 2021.
- [39] FX Elements LLC. Fx elements, 2021.
- [40] LesterBanks. Rodypolis tiene como objetivo redefinir las explosiones y las imágenes de archivo de fuego, con su ayuda, 2017.
- [41] Inspiration Tuts. Blender addons for simulation and visual effects, 9 de febrero de 2020.
- [42] Gov3dstudio. 15 plugins de 3ds max que debes tener en tu pc.
- [43] Factoria 5. Los mejores plugins para dominar 3ds max, 20 de abril de 2020.
- [44] InspirationTuts. 12 great 3ds max vfx plugins, 10 de diciembre de 2019.
- [45] Makeitcg. Los 25 mejores complementos para autodesk maya, 2015.
- [46] Inspiration Tuts. Maya plugins for vfx, 9 de febrero de 2020.
- [47] Jawset Visual Computing. Turbulence fd, 2021.
- [48] 3DQuakers. Forester for cinema 4d, 2021.

- [49] c4Depot. Infinite space, 2018.
- [50] INSYDIUM LTD. X-particles, 2021.
- [51] agenciatributaria LTD. Tabla de coeficientes de amortizacion lineal, 2015.
- [52] Agencia Estatal Boletín Oficial del Estado (AEBOE). Artículo 270 del boletín oficial del estado, sección 1.<sup>a</sup> de los delitos relativos a la propiedad intelectual. *BOE*, 2015.
- [53] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The L<sup>A</sup>T<sub>E</sub>X Companion*. Addison-Wesley, Reading, Massachusetts, 1993.
- [54] Albert Einstein. Zur Elektrodynamik bewegter Körper. (German) [On the electrodynamics of moving bodies]. *Annalen der Physik*, 322(10):891–921, 1905.
- [55] Donald Knuth. Knuth: Computers and typesetting.