



ugr | Universidad
de **Granada**

TRABAJO FIN DE GRADO
GRADO EN INGENIERÍA INFORMÁTICA

Biblioteca de Representación Gráfica 3D con Blender para la Visualización Científica en Nanoelectrónica

Autora

Laura Muñoz Rodríguez

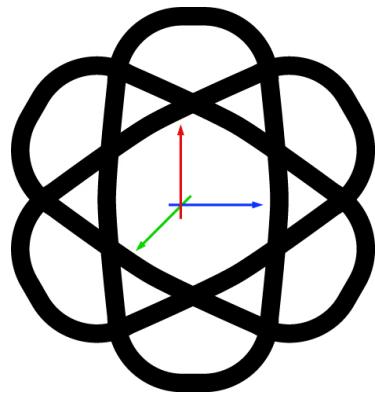
Tutores

Francisco M. Gómez Campos
Sergio Alonso Burgos



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE
TELECOMUNICACIÓN

Granada, 11 de Julio de 2016



Biblioteca de Representación Gráfica 3D con Blender para la Visualización Científica en Nanoelectrónica

Autora

Laura Muñoz Rodríguez

Tutores

Francisco M. Gómez Campos
Sergio Alonso Burgos

Biblioteca de Representación Gráfica 3D con Blender para la Visualización Científica en Nanoelectrónica

Laura Muñoz Rodríguez

Palabras clave: Blender, Nanoelectrónica, Python, Punto cuántico, Memristor, 3D, Visualización científica

Resumen

En este trabajo hemos desarrollado unas bibliotecas de representación de datos científicos para dos líneas de investigación que se están llevando a cabo en el departamento de electrónica y tecnología de los computadores de la Universidad de Granada. Uno en colaboración con el Instituto de Microelectrónica de Barcelona y otro en colaboración con la Universidad de Leeds en Reino Unido.

Se han utilizado herramientas de software libre y hemos generado una serie de scripts adaptados a las necesidades del investigador. A lo largo del proyecto se han mantenido entrevistas con los usuarios para conocer qué requisitos debía tener el producto final y el resultado de nuestro trabajo son unos vídeos científicos generados automáticamente.

Library for 3D Representation of Scientific Data for Research in Nanoelectronics with Blender

Laura Muñoz Rodríguez

Keywords: Blender, Nanoelectronics, Python, Quantum Dot, RRAM, 3D, Scientific visualization

Abstract

In this project we have developed two libraries for 3D Representation of Scientific Data for two research lines carried out in the Electronics Department of the University of Granada. One of them is done in collaboration with CNM Barcelona (National Centre for Microelectronics) and the other one with the University of Leeds (United Kingdom). We used free software tools and have generated a series of scripts tailored to the needs of the researcher. Throughout the project we have made interviews with both clients to know what requirements should have the final product and the result of our work is a scientific video generated automatically.

Yo, **Laura Muñoz Rodríguez**, alumna de la titulación Grado en Ingeniería Informática de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 45107621, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.



Fdo: Laura Muñoz Rodríguez

Granada a 11 de Julio de 2016

D. **Francisco M. Gómez Campos**, Profesor del Área de Electrónica del Departamento Electrónica y Tecnología de los Computadores de la Universidad de Granada.

D. **Sergio Alonso Burgos**, Profesor del Área de Lenguajes y Sistemas Informáticos del Departamento Lenguajes y Sistemas Informáticos de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado *Biblioteca de Representación Gráfica 3D con Blender para la Visualización Científica en Nanoelectrónica*, ha sido realizado bajo su supervisión por **Laura Muñoz Rodríguez**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 11 de Julio de 2016

Los directores:



Francisco M. Gómez Campos



Sergio Alonso Burgos

Agradecimientos

A mis padres, Jesús y Mila, porque gracias a ellos he podido llegar hasta aquí.

A Virginia, mi hermana, porque es mi punto de apoyo y a pesar de la distancia siempre está ahí cuando la necesito.

A Francisco y Sergio, por haber aceptado ser mis tutores, por la continua disposición a lo largo de este reto, por todos los conocimientos transmitidos y por su eterna paciencia.

Y todo este agradecimiento a pesar de, como me dijo Francisco, "haberme metido en un infierno".

Índice general

Índice	I
1. Introducción	1
1.1. Problemas	1
1.1.1. RRAM	2
1.1.2. Puntos Cuánticos	3
1.2. Herramientas que vamos a usar	4
1.2.1. Blender	4
1.2.2. Python	5
2. Objetivos	7
2.1. Objetivos principales y secundarios	7
2.2. Materias y herramientas de desarrollo	8
3. Planificación	9
3.1. Metodología de desarrollo	9
3.2. Fases	10
3.2.1. Fase 0. Planteamiento del problema	11
3.2.2. Fase 1. Especificaciones de requisitos	11
3.2.3. Fase 2. Análisis	11
3.2.4. Fase 3. Diseño e Implementación	11
3.2.5. Fase 4. Pruebas	11
3.2.6. Fase 5. Documentación	12
3.3. Temporización	12
3.3.1. Temporización RRAM	12
3.3.2. Temporización puntos cuánticos	14
4. Análisis	17
4.1. Análisis de requisitos	17
4.1.1. Requisitos funcionales	17
4.1.2. Requisitos no funcionales	18
4.2. Descripción de actores	18
4.3. Descripción de casos de uso ¹⁵	19

4.3.1. CU-1. Abrir aplicación	19
4.3.2. CU-2. Lanzar aplicación	20
4.3.3. CU-3. Borrar modelo 3D	21
4.4. Diagrama de casos de uso	22
5. Diseño e Implementación	23
5.1. Bocetos de las interfaces	24
5.2. Interfaces de usuario	26
5.3. Implementación	27
6. Pruebas	31
7. Conclusiones y lecciones aprendidas	33
8. Trabajos futuros	37
A. Installation Manual	43
A.1. Blender installation	43
A.1.1. Install in Windows	44
A.2. Blender configuration	47
B. RRAM User Manual	48
B.1. First contact	48
B.2. Fill the form	49
C. QD User Manual	51
C.1. First contact	51
C.2. Fill the form	52

Índice de figuras

1.1.	Imagen generada con Matlab	3
1.2.	Imagen generada con Grace	4
1.3.	Blender	4
1.4.	Python	5
3.1.	Desarrollo iterativo - http://www.vass.es	9
3.2.	Fases del proyecto - http://www.vass.es	10
3.3.	Iteración 1 RRAM Estimada	12
3.4.	Iteración 1 RRAM Real	12
3.5.	Iteración 2 RRAM Estimada	13
3.6.	Iteración 2 RRAM Real	13
3.7.	Iteración 1 QD Estimada	14
3.8.	Iteración 1 QD Real	14
3.9.	Iteración 2 QD Estimada	15
3.10.	Iteración 2 QD Real	15
4.1.	Diagrama de Casos de Uso	22
5.1.	Interfaces de usuario, editor de scripts	23
5.2.	Boceto RRAM 1	24
5.3.	Boceto RRAM 2	24
5.4.	Boceto puntos cuánticos 1	25
5.5.	Boceto puntos cuánticos 2	25
5.6.	Interfaz RRAM	26
5.7.	Interfaz puntos cuánticos	26
5.8.	IDE de Blender	27
5.9.	Gráfica interpolación temperaturas	28
5.10.	Obtención del color por interpolación	28
5.11.	Punto cuántico con un nivel de opacidad para los átomos del 100%	29
5.12.	Punto cuántico con un nivel de opacidad para los átomos del 50%	29
7.1.	Variación de color según temperatura	35
7.2.	Isosuperficie del módulo al cuadrado de la función de onda (densidad de probabilidad)	36

A.1. Download Blender	43
A.2. Blender Setup	44
A.3. End-User License Agreement	44
A.4. Custom Setup	45
A.5. Ready to install Blender	45
A.6. Installing Blender	46
A.7. Completed the Blender setup	46
A.8. Blender User Preferences	47
A.9. Auto Run Python Scripts	47
B.1. Interface RRAM	48
B.2. Form RRAM	49
B.3. Folders	50
C.1. Interface QD	51
C.2. Form QD	52

Índice de cuadros

4.1. CU-1. Abrir aplicación	19
4.2. CU-2. Lanzar aplicación	20
4.3. CU-3. Borrar modelo 3D	21

Capítulo 1

Introducción

En los últimos años se promueve la representación de datos científicos usando herramientas audiovisuales. Así las revistas científicas promueven el uso de material complementario en la publicación de artículos que va más allá de las tradicionales gráficas o tablas.

En este proyecto se ha usado el software abierto Blender para realizar una biblioteca de representación de datos científicos en tres dimensiones aprovechando su enorme potencialidad. En el trabajo se han realizado una serie de scripts¹ versátiles y reutilizables en lenguaje Python que servirán al investigador para obtener visualizaciones en movimiento de gran calidad de los resultados de su investigación para poder ser usados en su divulgación.

Como ejemplos se trabajarán representaciones de datos obtenidos de simulaciones de memorias RRAM² y de la fotofísica de puntos cuánticos³ de semiconductores.

1.1. Problemas

Los estudios científicos suelen llevarse a cabo de forma experimental y de forma teórica. En la sección teórica se suele realizar un primer análisis matemático del problema en el que se establecen unas hipótesis de partida y en la gran mayoría de las ocasiones se hace necesario llevar a cabo cálculos matemáticos de gran complejidad. En esa etapa surje el concepto de la simulación. Según R. E. Shannon⁴, "la simulación es el proceso de diseñar un modelo de un sistema real y llevar a término experiencias con él, con la finalidad de comprender el comportamiento del sistema o evaluar nuevas estrategias - dentro de los límites impuestos por un cierto criterio o un conjunto de ellos - para el funcionamiento del sistema."

Una simulación computacional sigue la definición de Shannon empleando para este propósito un ordenador, en el que se introducen los modelos matemáticos que describen la física de un sistema. Habitualmente las simulaciones proporcionan una cantidad ingente de datos que es necesario analizar. Este análisis puede ser de muchos tipos aunque está claro que obtener representaciones gráficas de los fenómenos simulados facilita poder tener una visión

global de los resultados obtenidos que tradicionalmente ha sido una gráfica y que en los últimos tiempos se está imponiendo el formato de vídeo o material interactivo.

Existen herramientas de visualización de datos científicos como Excel, Gnuplot, Mathematica, Matlab, que son capaces de proporcionar gráficas en 2D, en 3D, así como gráficas interactivas y animaciones. Sin embargo, la estética de los resultados a veces hace que no sean fácilmente comprensibles y muchas veces solamente los pueden interpretar personas que trabajan en el campo de investigación concreto.

Los programas de modelado y animación 3D concretamente no están orientados a la visualización científica de datos. Sin embargo están preparados para producir cualquier imagen controlando tanto la posición de cámara como las texturas, la iluminación, etc, de forma que se obtiene una representación de mayor valor estético y que por tanto puede llegar a ser comprendido por mucha más gente. El hecho de que puedas representar los datos de la forma más clara posible, facilita la comprensión tanto para el espectador experto como para el novel.

No obstante el investigador no está familiarizado con modelar y animar en 3D, por lo que el objetivo fundamental de este proyecto es crear una serie de scripts que permitan al investigador realizar una representación de los datos de muy alta calidad sin necesidad de aprender modelado y animación 3D.

1.1.1. RRAM

Las RRAM, del inglés *Resistive Random-Access Memory*, son un tipo de memorias de acceso aleatorio no volátiles, que funcionan mediante el cambio de la resistencia a lo largo de un material de estado sólido dieléctrico a menudo referido como memristor².

Existe un equipo de investigadores en el departamento de Electrónica y Tecnología de los Computadores de la Universidad de Granada, que investiga teóricamente el comportamiento de las RRAM en colaboración con el Instituto de Microelectrónica de Barcelona. El centro catalán realiza medidas experimentales de estos dispositivos y el equipo de investigación granadino (en lo sucesivo el cliente de las RRAM) modela y simula los sistemas estudiados con el fin de relacionar los modelos teóricos con los experimentos de los que se disponen.

La dificultad que entraña es que es un dispositivo nuevo del que todavía hay muchos mecanismos físicos por determinar y se desea encontrar la forma más atractiva de mostrar los datos tanto para observar detalles concretos obtenidos con las simulaciones, como para poder exponer los resultados obtenidos ante la comunidad científica. Concretamente es interesante hacer representaciones de las distribuciones de temperatura y los caminos de percolación⁵ que van apareciendo conforme se aplican tensiones más altas.

Hasta la fecha el método de representación que ha utilizado el cliente es Matlab⁶. Los resultados mostrados con esta herramienta son claros pero el cliente nos pide potenciar la

estética de los mismos, para poder llegar a un público más amplio y así promocionar su investigación.

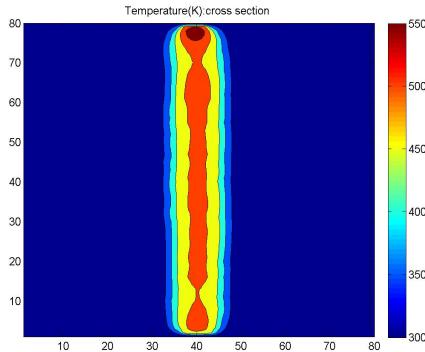


Figura 1.1: Imagen generada con Matlab

Vamos a desarrollar un procedimiento dinámico y en 3D usando Blender⁷ para hacer más atractivo y pedagógico el resultado.

1.1.2. Puntos Cuánticos

Los puntos cuánticos son cristales nanométricos hechos de materiales semiconductores. Un nanómetro (nm) es la mil millonésima parte de un metro, lo que significa que estas partículas extra pequeñas son más pequeñas que 1/10000 de un cabello humano.

El departamento de Electrónica de Tecnología de los Computadores de la Universidad de Granada colabora con el Institute of Microwave and Photonics de la Universidad de Leeds (en lo sucesivo el cliente de los puntos cuánticos) en el estudio teórico de la fotofísica de puntos cuánticos de semiconductor y en el estudio del transporte de electrones en sistemas periódicos de puntos cuánticos. El equipo inglés resuelve la ecuación de Schrödinger⁸ para puntos cuánticos encontrando las funciones de onda y los niveles de energía de los estados permitidos en los mismos.

Anteriormente a este proyecto no podían representarse los datos conjuntamente con los átomos de la estructura, siendo difícil encontrar una relación entre la simetría de la función de onda y las magnitudes físicas relacionadas con el solapamiento de funciones de onda de los distintos estados.

El programa que el cliente estaba usando con anterioridad para visualizar la función de onda era Grace y para los átomos RasMol⁹ y Jmol¹⁰. La representación conjunta de todo no era posible con el software que el cliente manejaba.

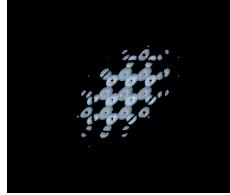


Figura 1.2: Imagen generada con Grace

En este caso Blender nos da la posibilidad de representarlo todo, tanto por separado como conjuntamente. Además podemos movernos alrededor de la estructura e incluso introducirnos dentro de la misma cambiando la transparencia de los átomos. El script que hemos desarrollado pretende realizar esta tarea.

1.2. Herramientas que vamos a usar

Para realizar este proyecto se ha decidido utilizar Python y el software Blender por sus características ventajosas. A continuación voy a detallar las herramientas que han sido utilizadas en el proyecto:

1.2.1. Blender

Blender es un software dedicado al modelado en 3D de objetos que además posibilita dar texturas y materiales a dichos objetos. También nos permite crear animaciones y tiene su propio motor de juegos que nos permite desde crear visitas virtuales hasta videojuegos. A todas estas funcionalidades se le añade la posibilidad de programar en Python¹¹ para extender sus capacidades.



Figura 1.3: Blender

1.2.2. Python

Python es un lenguaje de programación diseñado en el 2006 por Guido van Rossum, orientado a objetos, interpretado y multiplataforma, que últimamente está despertando gran interés en el ámbito de la investigación. Una de las ventajas que nos ofrece para nuestro proyecto es que está integrado en Blender y gracias a sus bibliotecas nos permite utilizar todo su potencial.



Figura 1.4: Python

Capítulo 2

Objetivos

El objetivo de este proyecto es facilitar la representación de datos científicos para su mejor entendimiento y facilitar las tareas de divulgación de los resultados. Se le proporcionará al usuario una biblioteca Blender donde podrá encontrar una interfaz de usuario donde llenar los datos requeridos para así generar la representación en 3D. Será multiplataforma y totalmente Software Libre.

Este objetivo se divide en los siguientes objetivos principales (OBJ) y secundarios (OBJS, los que a su vez pueden ser obligatorios (O) y no obligatorios (N).

2.1. Objetivos principales y secundarios

Principales:

- **OBJ-1:** Crear la gráfica para el problema de la RRAM

Dependencias: OBJS-1-O

- **OBJ-2:** Crear la gráfica para el problema de los puntos cuánticos

Dependencias: OBJS-1-O

Secundarios:

- **OBJS-1-O:** Familiarización con el entorno, la biblioteca Python y Blender

Es importante remarcar que esta fase cobra una especial importancia ya que el programa Blender tiene una curva de aprendizaje empinada e incluso es muy diferente el manejo desde el punto de vista de script a hacerlo a nivel de usuario. Además el disponer de escasa documentación de Blender y que la que hay cambia respecto a la versión que uses es lo que ha hecho que sea más complicado el proceso de familiarización.

- **OBJS-2-N:** Preparación de rutinas de ayuda
Creación de un código más modular y reutilizable.
- **OBJS-3-O:** Interfaz para el problema de la RRAM
Crear una interfaz para facilitar al cliente la introducción de los parámetros.
- **OBJS-4-O:** Interfaz para el problema de los puntos cuánticos
Crear una interfaz para facilitar al cliente la introducción de los parámetros.
- **OBJS-5-N:** Creación de animaciones apoyadas por los scripts
Crear una animación manualmente usando los scripts de forma auxiliar para modelar y texturizar.
- **OBJS-6-O:** Creación de animaciones automáticas
- **OBJS-7-N:** Creación de un sistema de render distribuído

2.2. Materias y herramientas de desarrollo

Para realizar este proyecto he puesto en práctica conocimientos aprendidos en las siguientes materias:

- **Fundamentos de Ingeniería del Software** para el análisis y la ingeniería de requisitos
- **Fundamentos Físicos Tecnológicos** para comprender los conceptos físicos básicos
- **Diseño de Interfaces de Usuario** para realizar los bocetos y las interfaces de usuario

Además he tenido que utilizar mis conocimientos, o mejorarlo, en los siguientes campos:

- **Python**, ya que es el lenguaje de programación utilizado. No tenía conocimientos previos por lo que he tenido que formarme por mi cuenta.
- **Blender**, puesto que es la herramienta principal del proyecto. He puesto en práctica muchos de los conocimientos adquiridos en el Curso de Animación y Modelado 3D impartido por la Escuela de Posgrado, además de tener que buscar otra información para complementar mis conocimientos.
- **LaTeX¹²** para la realización de la documentación del proyecto y los manuales de usuario.

Capítulo 3

Planificación

3.1. Metodología de desarrollo

Para realizar este proyecto se ha seguido una metodología ágil¹³, basada en el desarrollo iterativo e incremental¹⁴ debido a que en las primeras reuniones quedó patente que el cliente no tenía claro el resultado que esperaba, sino solo ciertas ideas muy generales de algunas visualizaciones de datos que necesitaba.

En un desarrollo de este tipo el proyecto se planifica en diversos bloques temporales llamados iteraciones. En cada una de estas iteraciones se repite un proceso de trabajo similar para proporcionar un resultado completo sobre el producto final. Se trata de iterar e involucrar al cliente en el proceso de desarrollo, no solo mejorando el producto, sino incluyendo nuevas funcionalidades conforme se necesitan.

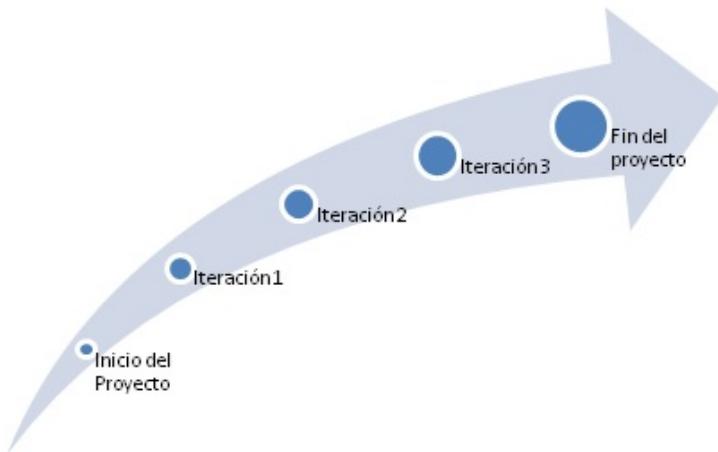


Figura 3.1: Desarrollo iterativo - <http://www.vass.es>

Se han ido concertando una serie de reuniones con el cliente para mostrarle lo desarrollado en cada iteración y para concretar nuevas mejoras a desarrollar en la siguiente. Al principio del proceso las reuniones eran cortas y semanales, según se iba encauzando el proyecto se fueron espaciando y durante todo el proceso se han ido grabando audios para poder retomar conceptos hablados con el cliente.

3.2. Fases

En cada una de las iteraciones de desarrollo de los prototipos, en general, planteamos las siguientes fases:

- Fase 0. Planteamiento del problema
- Fase 1. Especificaciones de requisitos
- Fase 2. Análisis y diseño
- Fase 3. Implementación
- Fase 4. Pruebas
- Fase 5. Documentación



Figura 3.2: Fases del proyecto - <http://www.vass.es>

3.2.1. Fase 0. Planteamiento del problema

- Primera reunión con el cliente donde se plantea el problema a resolver
- Descripción de los objetivos
- Planteamiento de las tecnologías a utilizar

3.2.2. Fase 1. Especificaciones de requisitos

- Extracción de los requisitos funcionales a partir de la reunión anterior
- Familiarización con las tecnologías elegidas

3.2.3. Fase 2. Análisis

- Análisis de requisitos
- Casos de uso

Apartado: Capítulo 4. Análisis

3.2.4. Fase 3. Diseño e Implementación

- Creación de la interfaz para el problema de la RRAM
- Creación de la interfaz para el problema de los Puntos Cuánticos
- Implementar una solución para el problema de la RRAM
- Implementar una solución para el problema de los Puntos Cuánticos

Apartado: Capítulo 5. Implementación

3.2.5. Fase 4. Pruebas

- Pruebas unitarias
- Pruebas de integración
- Pruebas de aceptación

Apartado: Capítulo 6. Pruebas

3.2.6. Fase 5. Documentación

- Manual de instalación
- Manual de usuario para cada uno de los problemas
- Documentación del proyecto

3.3. Temporización

Para llevar a cabo el proyecto hemos hecho una temporización estimada dividida en dos partes, una por cada problema. Hemos empezado por el problema de las RRAM estableciendo una duración desde el 5 de octubre de 2015 hasta el 14 de febrero de 2016, a continuación nos hemos centrado en el problema de los puntos cuánticos desde el 15 de febrero de 2016 hasta el 19 de Junio de 2016. Cada uno de los problemas consta de seis fases realizándose dos iteraciones.

De la temporización se han realizado los diagramas de Gantt, correspondientes tanto a los tiempos estimados como a los reales.

3.3.1. Temporización RRAM

ITERACIÓN 1

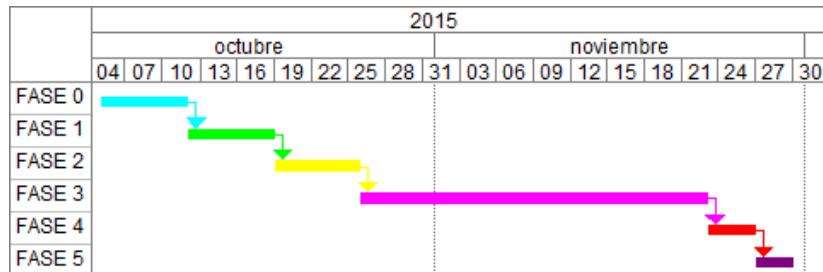


Figura 3.3: Iteración 1 RRAM Estimada

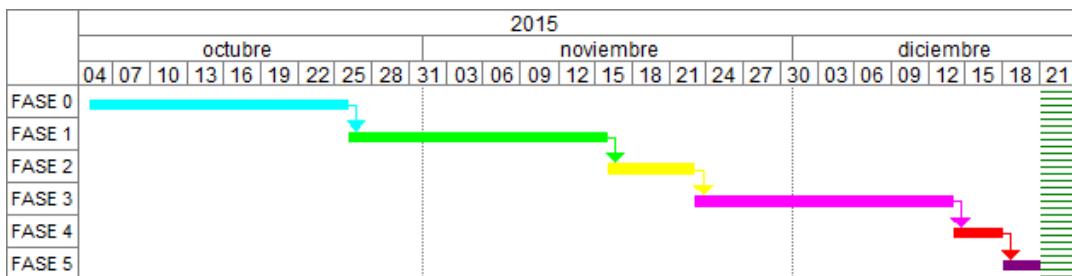


Figura 3.4: Iteración 1 RRAM Real

Se puede observar que el tiempo real empleado tanto en la fase 0 como en la fase 1 ha sido superior al estimado, debido a:

- La falta de definición del problema por parte del cliente
- La dificultad para encontrar documentación de Blender

Este retraso sufrido en las dos primeras fases ha sido parcialmente recuperado gracias a que la implementación nos ha llevado menos tiempo del estimado.

ITERACIÓN 2

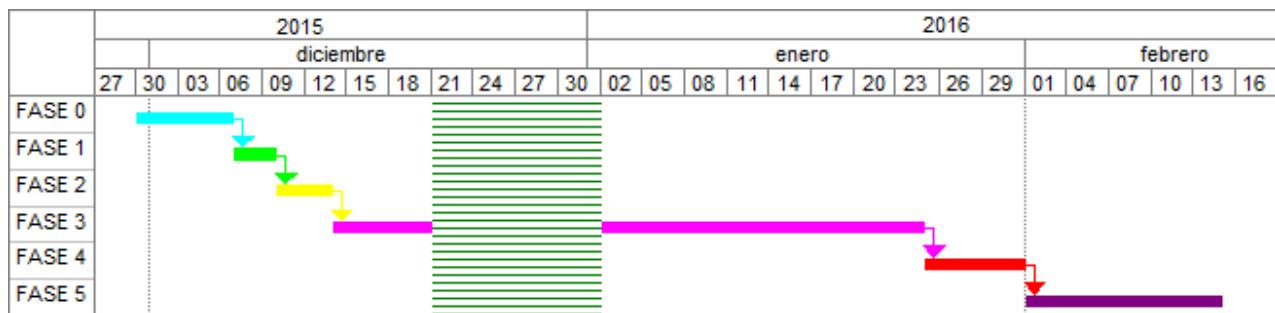


Figura 3.5: Iteración 2 RRAM Estimada

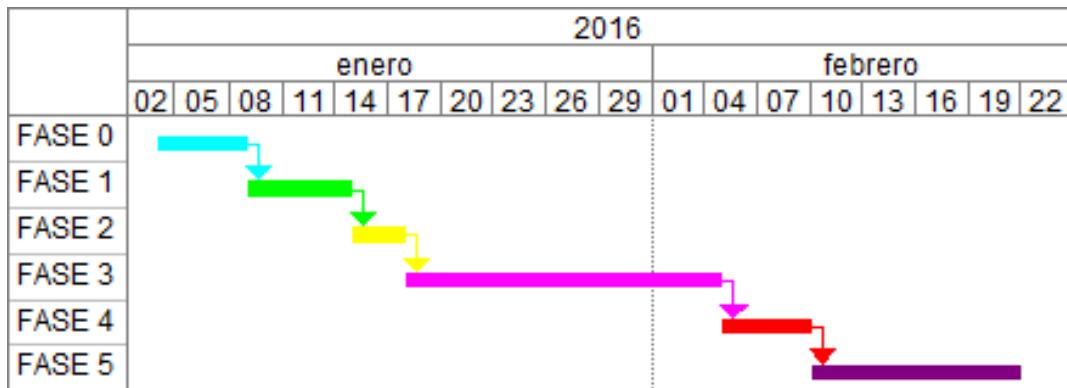


Figura 3.6: Iteración 2 RRAM Real

En la segunda iteración podemos ver que el tiempo real ha sido menor que el estimado, esto se debe a que en la primera iteración ha quedado bastante bien definido el problema y no ha hecho falta emplear tanto tiempo.

3.3.2. Temporización puntos cuánticos

ITERACIÓN 1

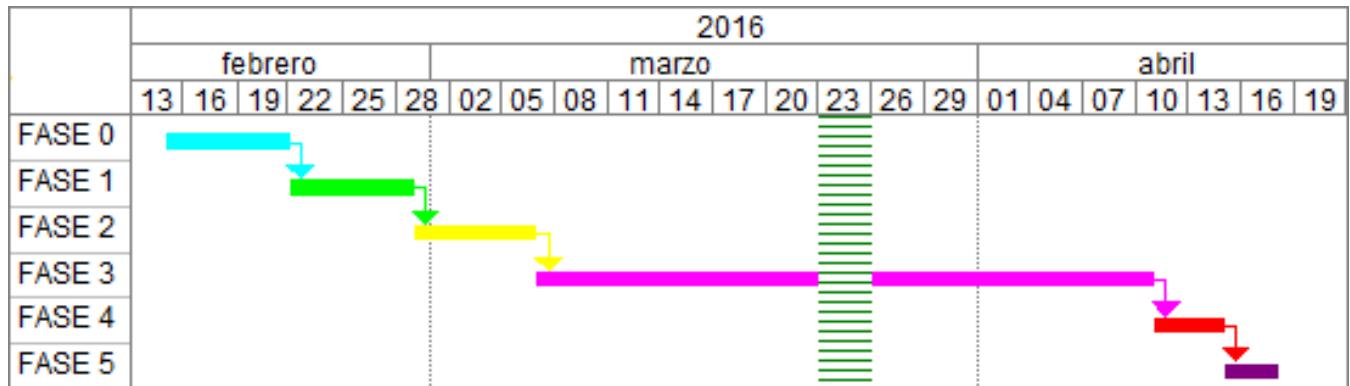


Figura 3.7: Iteración 1 QD Estimada

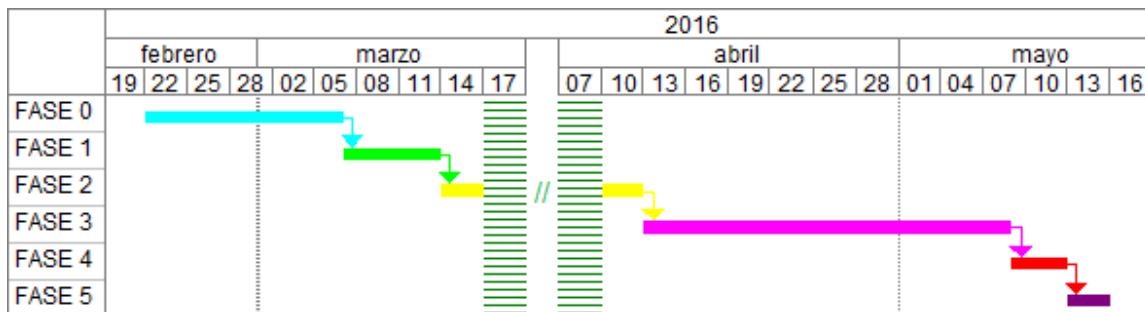


Figura 3.8: Iteración 1 QD Real

En este nuevo problema, aunque distinto, usamos las mismas tecnologías, por lo que la fase de adaptación no ha requerido tanto tiempo. Lo que si sigue ocupando más tiempo del esperado es la Fase 0, ya que el cliente, aunque sabe lo que quiere, no sabe cómo lo quiere.

En medio de la Fase 2 por asuntos personales hicimos un parón, en parte coincidiendo con la Semana Santa.

ITERACIÓN 2

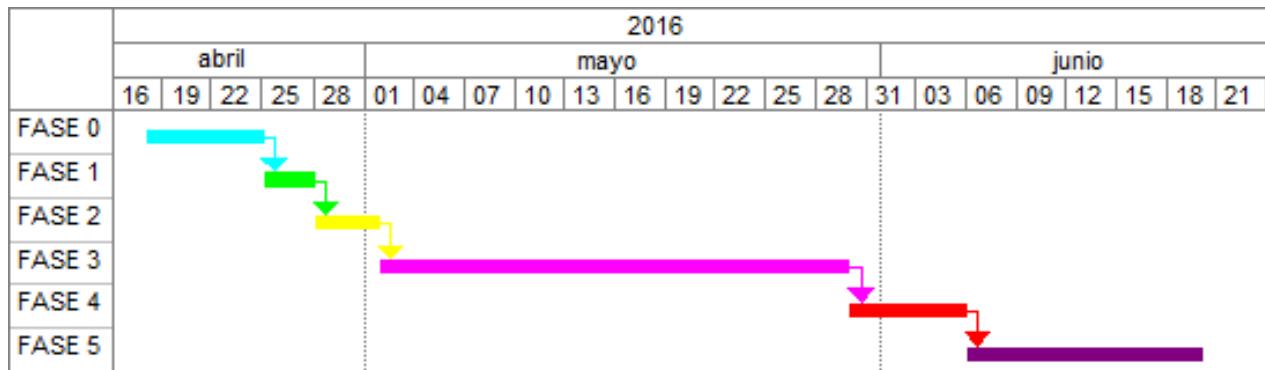


Figura 3.9: Iteración 2 QD Estimada

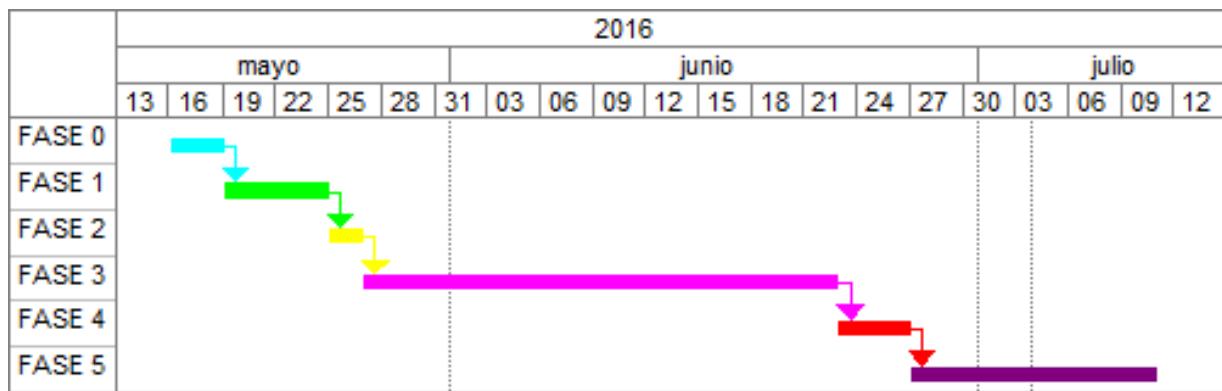


Figura 3.10: Iteración 2 QD Real

En la temporización estimada dejamos dos semanas (del 20 de junio al 3 de Julio de 2016) al final del proyecto para posibles imprevistos y los atrasos acumulados durante todo el proyecto nos han llevado a consumir dichas semanas. Además de que a la temporización real se le ha añadido una semana, acabando el 10 de Julio de 2016.

Capítulo 4

Análisis

4.1. Análisis de requisitos

El primer paso a dar en el análisis de un desarrollo de software es el de identificar los requisitos funcionales y no funcionales. Obtendremos estos requisitos gracias a las reuniones con el cliente y su cumplimiento nos garantizará que será un software funcional que cumple con sus necesidades.

4.1.1. Requisitos funcionales

Generales

- **RF-1:** El programa deberá permitir introducir datos al usuario
- **RF-2:** El programa representará el entorno 3D para las distintas simulaciones
- **RF-3:** El programa generará todos los frames de la animación
- **RF-4:** El programa creará un vídeo con la animación

RRAM

- **RF-5:** El programa generará una representación de la estructura a la que se le realizará una sección para ver en su interior
- **RF-6:** El programa representará las temperaturas en el sistema mediante una paleta de colores
- **RF-7:** Se podrá decidir el intervalo de archivos de datos que se utilizarán para realizar la simulación.
- **RF-8:** Se podrá decidir el intervalo de archivos de temperaturas que se utilizarán para realizar la simulación.

- **RF-9:** Se podrá decidir la distancia de cada punto dentro de la malla.
- **RF-10:** Se podrá decidir el tamaño de la malla.
- **RF-11:** Se podrá decidir si habrá un corte en la representación o no.
 - **RF-11.1:** Se podrá decidir dónde se realizará el corte tanto en el eje X como en el eje Y.
- **RF-12:** Se podrá decidir de dónde coger los archivos
- **RF-12:** Se podrá decidir dónde se guardarán los frames y el vídeo generados

Puntos cuánticos

- **RF-7:** El usuario será capaz de modificar el tamaño de los átomos que forman el punto cuántico

4.1.2. Requisitos no funcionales

Generales

- **RNF-1:** La aplicación será compatible con los principales sistemas operativos (Windows, Linux, Mac OS)
- **RNF-2:** El idioma de la interfaz será inglés
- **RNF-2:** El idioma de los manuales será inglés
- **RNF-3:** Utilizar licencias libres para publicar el proyecto como Software libre

4.2. Descripción de actores

AC-1: Usuario

- **Descripción:** Persona que utilizará la aplicación
- **Características:** Es el usuario estándar de la aplicación.
- **Relaciones:** Ninguna
- **Atributos:** Ninguno
- **Comentarios:** El usuario es físico con un perfil profesional de modelar y simular sistemas físicos, pero con conocimientos de tecnología o modelado y animación 3D.

4.3. Descripción de casos de uso¹⁵

4.3.1. CU-1. Abrir aplicación

- **Actores:** Usuario
- **Tipo:** Primario, Esencial
- **Precondición:** La opción *Auto Run Python Scripts* debe estar activada y *Relative Paths* desactivada en las propiedades de Blender.
- **Postcondición:** En Blender se muestra la interfaz creada que ha sido lanzada por un script.
- **Propósito:** Iniciar la aplicación.
- **Resumen:** El usuario abrirá el archivo .blend.
- **Autor:** Laura Muñoz Rodríguez
- **Versión:** 1.0

Curso normal		
	Actor	Sistema
1	Usuario: Abre el archivo .blend	
		2 El sistema lanza el script y muestra el formulario de entrada de datos

Cuadro 4.1: CU-1. Abrir aplicación

4.3.2. CU-2. Lanzar aplicación

- **Actores:** Usuario
- **Tipo:** Primario, Esencial
- **Precondición:** Haber abierto la aplicación
- **Postcondición:** Se muestra el objeto 3D en pantalla y tanto el vídeo como los frames se crean en la carpeta elegida por el usuario para el render.
- **Propósito:** Crear el modelo 3D y el vídeo.
- **Resumen:** El usuario rellenará una serie de campos y presionará START para que se realice el modelo 3D y el vídeo.
- **Autor:** Laura Muñoz Rodríguez
- **Versión:** 1.0

Curso normal		
	Actor	Sistema
1	Usuario: rellena el formulario y presiona el botón START	
		2 El sistema hace los cálculos oportunos y devuelve el modelo 3D. Además creará los frames y el vídeo en la carpeta que el usuario haya elegido.

Cuadro 4.2: CU-2. Lanzar aplicación

4.3.3. CU-3. Borrar modelo 3D

- **Actores:** Usuario
- **Tipo:** Secundario, Esencial
- **Precondición:** Haber abierto la aplicación
- **Postcondición:** El modelo 3D y el clip de vídeo, en caso de haber sido creados, serán eliminados dentro de Blender.
- **Propósito:** Borrar modelo 3D y clip de vídeo.
- **Resumen:** El usuario presionará el botón CLEAR 3D y tanto el modelo 3D como el clip de vídeo serán eliminados, en caso de existir.
- **Autor:** Laura Muñoz Rodríguez
- **Versión:** 1.0

Curso normal		
	Actor	Sistema
1	Usuario: pulsa el botón CLEAR 3D	
		2 El sistema borra el modelo 3D y el secuenciador de vídeo

Cuadro 4.3: CU-3. Borrar modelo 3D

4.4. Diagrama de casos de uso

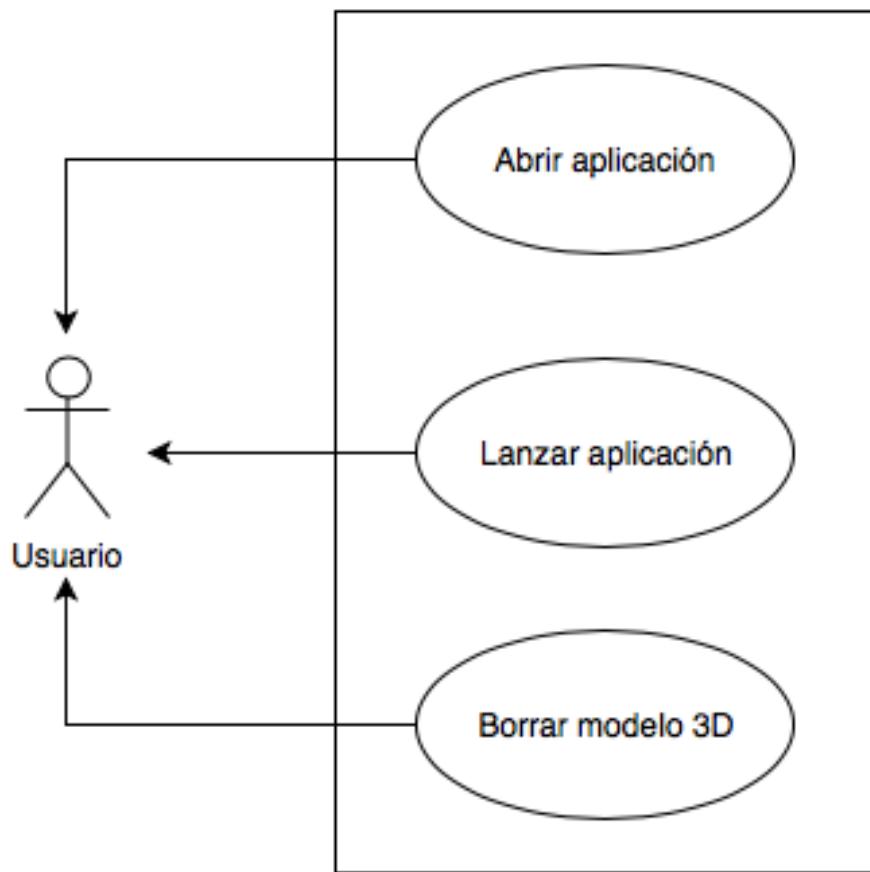


Figura 4.1: Diagrama de Casos de Uso

Capítulo 5

Diseño e Implementación

El diseño de los programas no es excesivamente complicado. De hecho no se ha tenido que seguir una estructura de clases compleja. Sin embargo, hemos prestado mayor atención al diseño de interfaces de usuario. Nos hemos familiarizado con Blender, hemos visto cómo crear un panel y cómo disponer los diferentes parámetros para así facilitar al cliente el uso del programa.

Para cada una de las gráficas tenemos, al menos, 3 scripts: uno donde almacenamos los datos, otro donde está todo lo relacionado con la interfaz de usuario¹⁶ y por último uno o varios scripts donde se crean los gráficos, las imágenes y las animaciones.

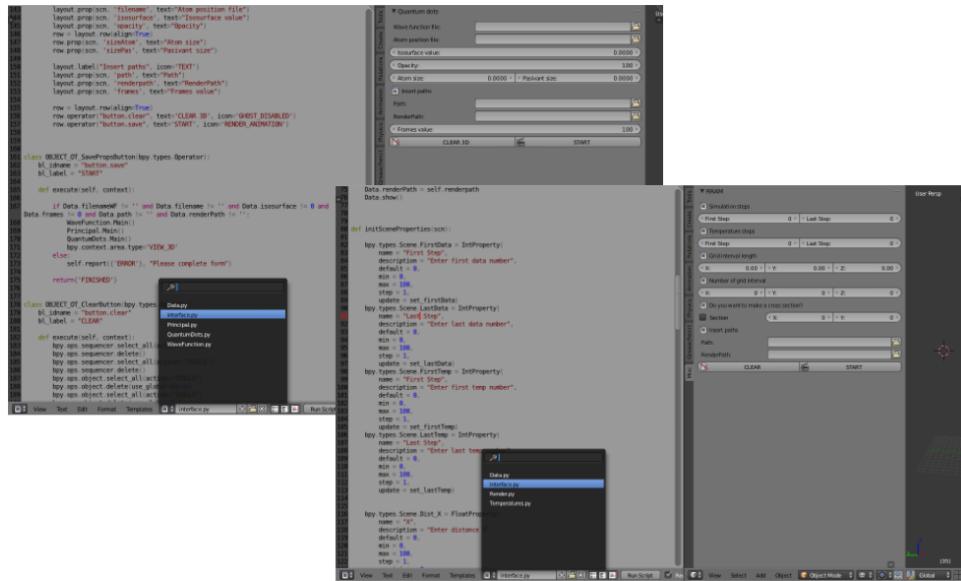


Figura 5.1: Interfaces de usuario, editor de scripts

5.1. Bocetos de las interfaces

A lo largo del proceso se han ido modificando los bocetos de las interfaces para que se fueran adaptando a las necesidades del usuario. Al igual que la aplicación ha ido experimentando mejoras, las interfaces también.

RRAM

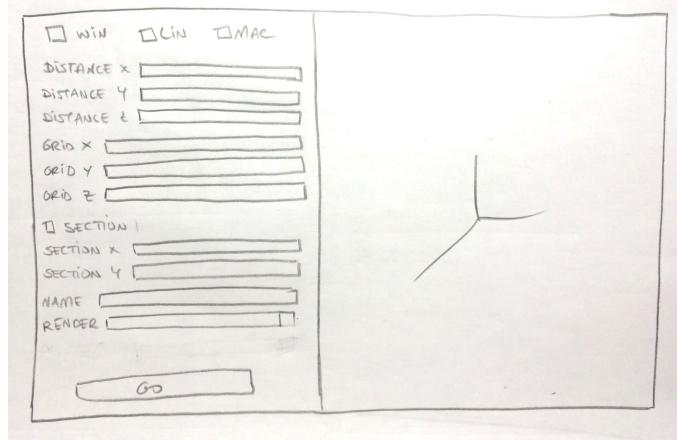


Figura 5.2: Boceto RRAM 1

En los primeros prototipos teníamos que seleccionar el sistema operativo que íbamos a utilizar pero tras una serie de mejoras logramos que no hiciera falta y directamente el programa se encarga de ello sin necesidad de que el usuario intervenga.

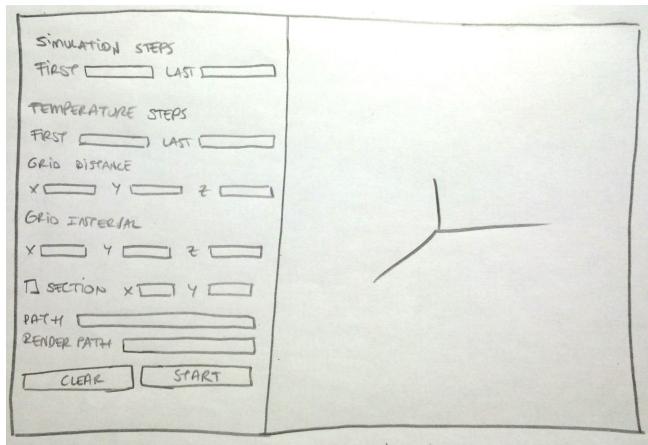


Figura 5.3: Boceto RRAM 2

Algunos parámetros se fueron incorporando a la interfaz para que así el usuario tuviera mayor control sobre la animación, además de un selector de rutas para indicar al programa dónde salvar los fotogramas y el vídeo. También se añadió la opción de que el usuario decidiera qué archivos quería incluir en la simulación, tanto de datos como de temperaturas.

Puntos cuánticos

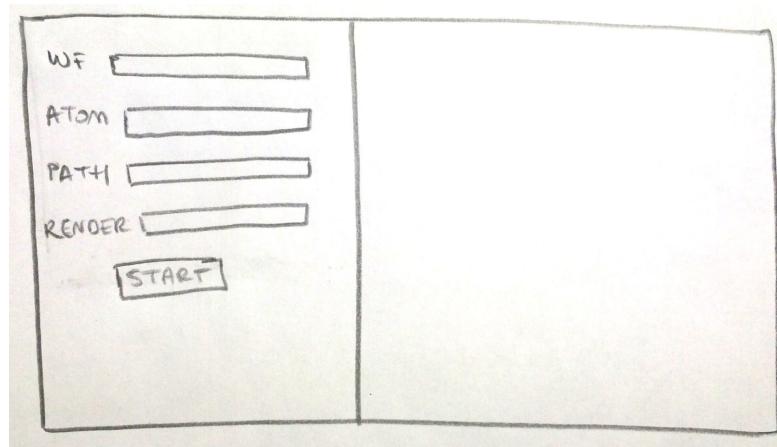


Figura 5.4: Boceto puntos cuánticos 1

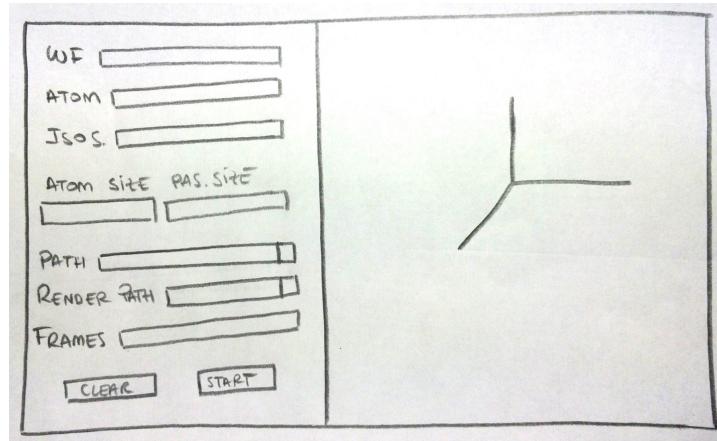


Figura 5.5: Boceto puntos cuánticos 2

Al igual que con el problema anterior se añadieron campos a la interfaz para personalizar cuanto más posible la animación.

5.2. Interfaces de usuario

Uno de los mayores problemas que nos hemos encontrado a la hora de hacer este proyecto, en lo que a programación se refiere, ha sido en todo lo referente a la interfaz de usuario, debido a la falta de documentación. Teníamos como objetivo facilitarle al cliente la representación en 3D y era fundamental crear una interfaz de usuario amigable y fácil de usar.

RRAM

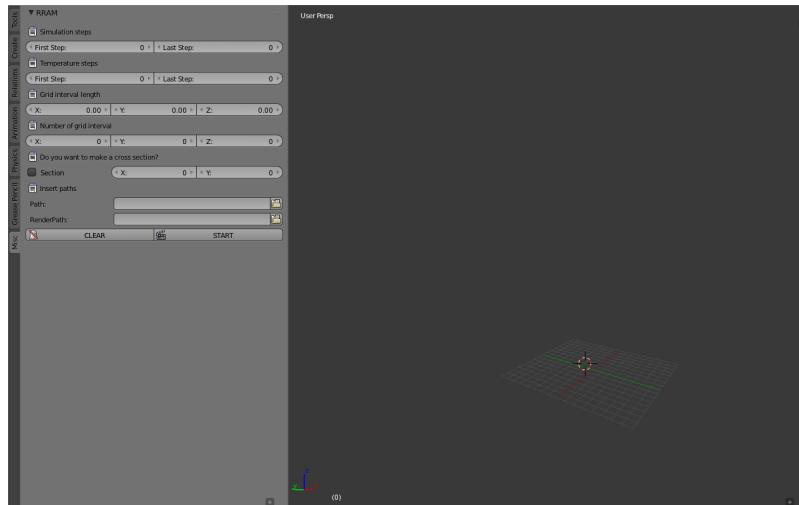


Figura 5.6: Interfaz RRAM

Puntos cuánticos

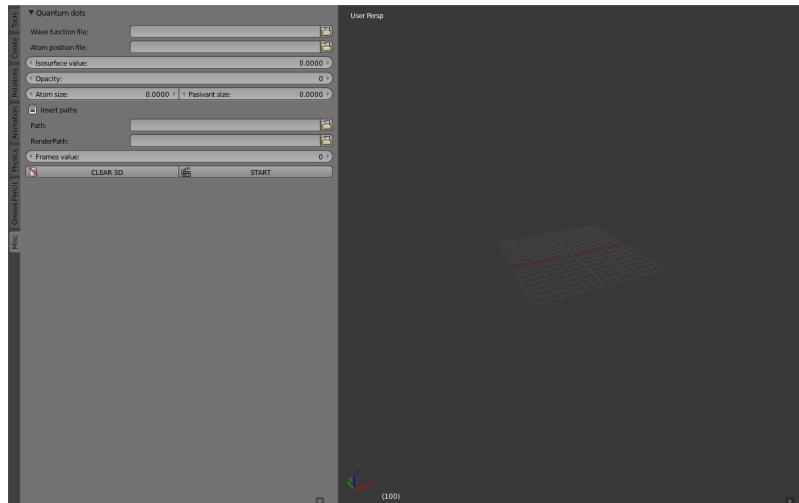


Figura 5.7: Interfaz puntos cuánticos

5.3. Implementación

Para la implementación se ha utilizado el mismo IDE¹⁷ que incorpora Blender, donde podemos ejecutar los scripts y además hacer consultas en su consola incorporada acerca de los comandos a utilizar.

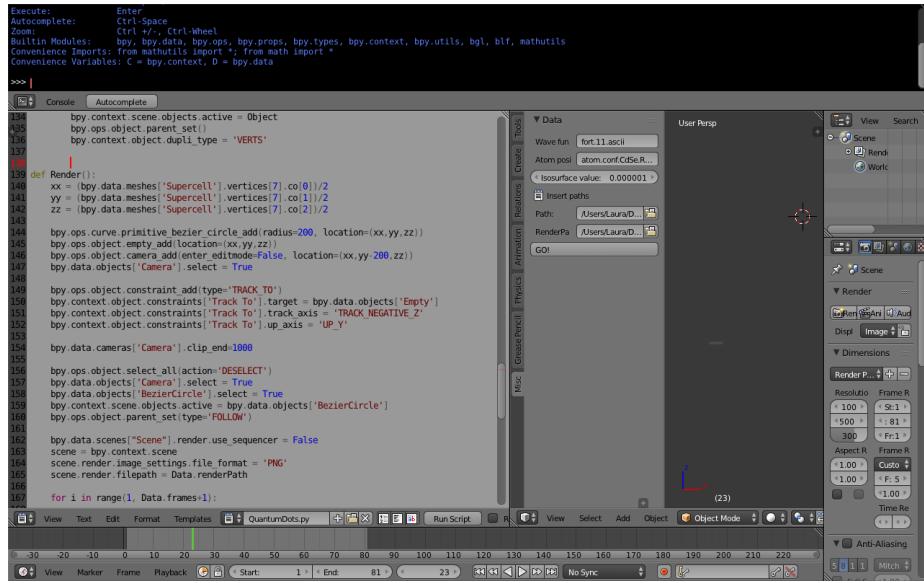


Figura 5.8: IDE de Blender

Tanto nombres de variables, funciones o comentarios están en inglés. Además todos los elementos salvo las clases seguirán la regla lowerCamelCase¹⁸, es decir, los nombres se escribirán juntos y sin ningún tipo de guión de separación. Para las clases se ha utilizado UpperCamelCase, donde la única diferencia es que la primera letra en este caso será en mayúscula.

En el caso de las constantes no se utiliza esta regla ya que irán completamente en mayúsculas y utilizaremos el guión bajo en caso de ser necesario para separar palabras.

En el problema de las RRAM, para poder asignar un color a cada una de las temperaturas, dividimos el intervalo de temperatura (300K - 450K) en 4 tramos. Asignamos a los extremos de los mismos un color ditribuido en una gama de colores, como se muestran en la Figura 5.10.

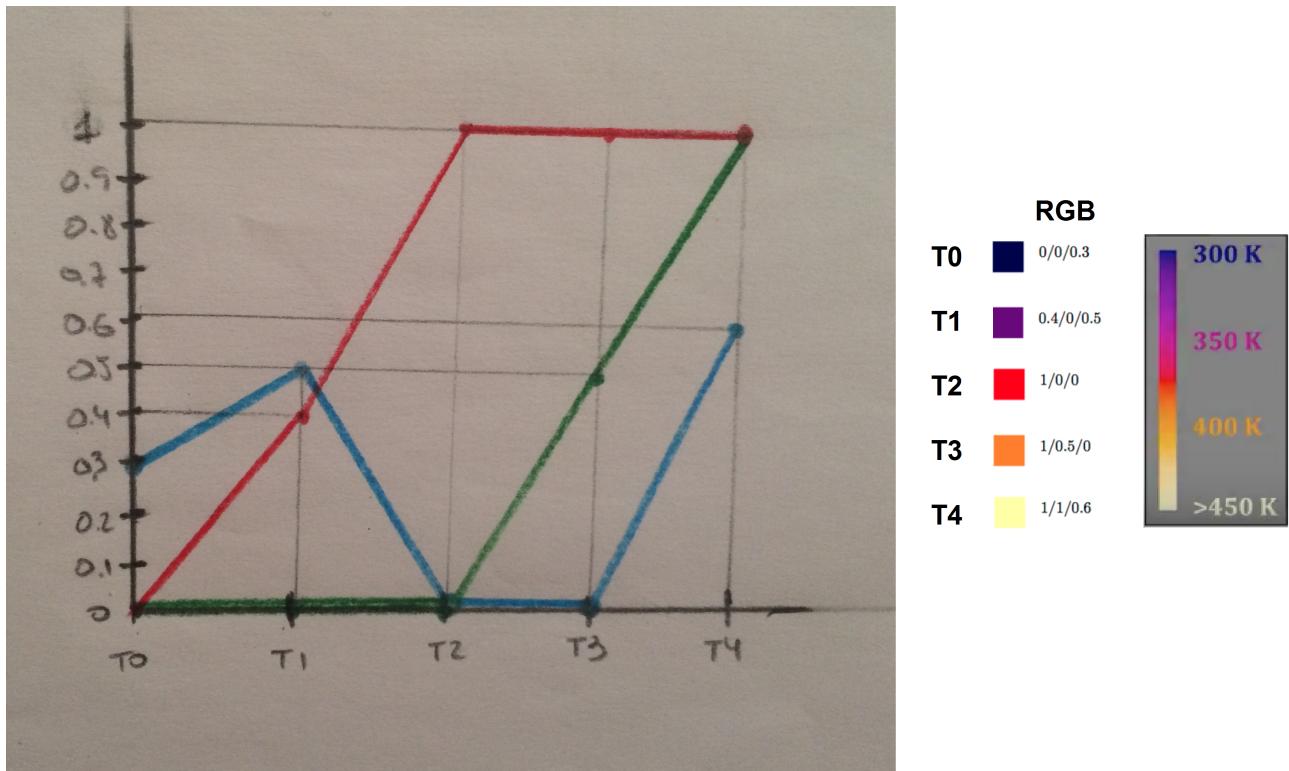


Figura 5.9: Gráfica interpolación temperaturas

Para el cálculo del color correspondiente a cualquier temperatura incluída entre los extremos de un determinado tramo, se realiza mediante un proceso de interpolación que nos calcula las componentes RGB¹⁹ correspondientes.

```

10
11 def Colors(RN, RBef, GN, GBef, BN, BBef, TN, TBef, i):
12     bpy.data.materials.new('color'+str(i))
13     bpy.data.materials['color'+str(i)].diffuse_color = (((RN -
14     RBef)/(TN - TBef)) * (float(i)-TBef)) + RBef,(((GN - GBef)/(TN -
     TBef)) * (float(i)-TBef)) + GBef,(((BN - BBef)/(TN - TBef)) *
     (float(i)-TBef)) + BBef)

```

Figura 5.10: Obtención del color por interpolación

Por otra parte, en el problema de los puntos cuánticos, hemos usado un material tipo halo para representar una isosuperficie del módulo al cuadrado de la función de onda (densidad de probabilidad). De esta forma nos permite ver el punto cuántico con sus tipos de átomos diferenciados por colores y tamaños, como podemos apreciar en las Figuras 5.11 y 5.12.

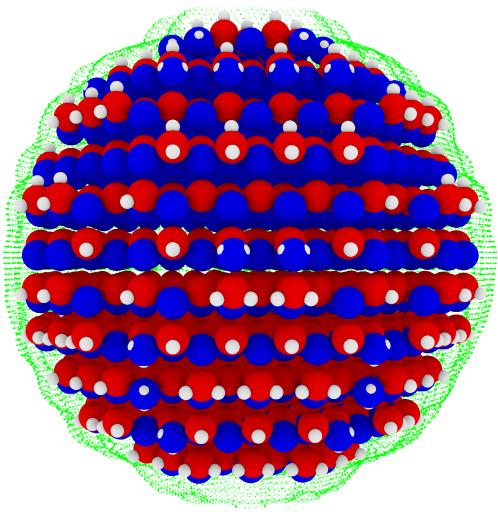


Figura 5.11: Punto cuántico con un nivel de opacidad para los átomos del 100 %

En la interfaz le damos la opción al usuario de poder regular la transparencia de los átomos para tener una mejor visualización del interior del punto cuántico.

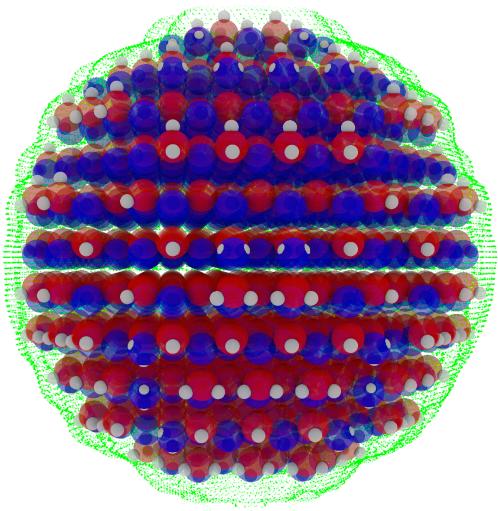


Figura 5.12: Punto cuántico con un nivel de opacidad para los átomos del 50 %

Capítulo 6

Pruebas

A lo largo de todo el proceso se han ido haciendo distintas pruebas de software²⁰. Dichas pruebas han sido realizadas de forma informal y durante todo el proceso de implementación.

Pruebas unitarias

Se han ido realizando pruebas unitarias para comprobar el correcto funcionamiento de los distintos módulos de código por separado. Como, por ejemplo, a la hora de tomar los datos de los archivos. Para asegurarnos que se realizaba de forma correcta lo que hicimos fue recoger los datos de un archivo y volcarlos en otro, luego comprobábamos que ambos coincidían y así sabíamos que todo había ido correctamente.

Pruebas de integración

Una vez sabíamos que los módulos de código funcionaban por separado, hemos realizado las pruebas de integración para comprobar que también tenían un correcto funcionamiento los unos con los otros y con Blender. Una de las pruebas de integración fue comprobar que al cargar Blender se nos ejecutaba el script adecuado. En las primeras pruebas vimos que no lo hacía bien y ahí nos dimos cuenta que era necesario marcar una opción en las preferencias de Blender para que permitiera ejecutar los scripts de forma automática.

Pruebas de aceptación

Por último se han llevado a cabo las pruebas de aceptación. Estas pruebas las realiza una persona, o varias, externas al desarrollo del proyecto. En este caso se han seleccionado varios usuarios para que probaran ambas aplicaciones, entre ellos los futuros clientes. De esta forma podemos comprobar si se cumplen tanto los requisitos funcionales como los no funcionales, planteados por el cliente en las reuniones. Como ejemplo concreto se le entregó al cliente de los puntos cuánticos el manual de usuario y el archivo Blender para que lo probara y como retroalimentación nos pidió añadir la opción de opacidad para ver el halo de la función de onda en el interior del punto cuántico.

Capítulo 7

Conclusiones y lecciones aprendidas

Durante el desarrollo del proyecto hemos experimentado una serie de problemas que nos han dificultado la ejecución del mismo.

De estos problemas y para evitar que se repitan, obtenemos las siguientes conclusiones:

- **Falta de definición de las especificaciones del cliente**

Hay que tenerla en cuenta cuando te muevas en el ámbito de la visualización de datos científicos por las características propias del campo, debido a que la investigación científica se suele hallar en el límite del conocimiento, y por tanto la experimentación, el ensayo y el error son metodologías habituales que te llevan a tener que valorar cuando se empieza un proyecto de esta naturaleza, debiendo estar mentalizados a que el desconocimiento por parte del cliente de cómo quería exactamente la representación supuso tener que realizar más reuniones con el mismo de las esperadas. Puesto que son aplicaciones para investigadores sabíamos que sería complicado el proyecto, ya que los requisitos en un principio expuestos tras una prueba pueden no ser satisfactorios y por lo tanto cambiados por otros, debiendo ajustar el proyecto a los nuevos requisitos.

Lección aprendida: cuando se empieza un proyecto tenemos que estar mentalizados de que puede ocurrir la indefinición de las especificaciones por parte del cliente, lo que nos hará requerir más tiempo, dificultad al coordinar horarios y provoca la realización de muchos cambios desde una posible idea inicial.

- **Dificultad en la obtención de información aplicada a la programación**

El lenguaje de programación usado ha sido Python, del que existe mucha información. Sin embargo hay escasez de manuales de uso de Blender en lo relativo a la programación y una mala documentación de Blender, siendo conscientes a priori de esto, nos propusimos el reto de llevarlo adelante.

Lección aprendida: para futuros trabajos previamente habría que mirar si existe documentación, clara o no, acerca de las herramientas que vayamos a utilizar.

- **Respuesta a las modificaciones en los scripts**

Un imprevisto que surgió y que nos dio muchos quebraderos de cabeza fue que, cuando al ejecutar el programa se producía un error, al volver al script para subsanarlo y pensando que ya estaba solucionado, cuando volvíamos a ejecutar se observaba que permanecía el mismo fallo. Esto supuso un transtorno hasta que nos percatamos de que la aplicación no respondía a los cambios realizados si no la cerrabas y la volvías a abrir. Ahora sí, al ejecutarlo, respondía a las correcciones efectuadas.

Lección aprendida: hay que tener en cuenta posibles imprevistos pues te pueden hacer perder el tiempo porque no surten efecto los cambios realizados.

- **Optimizar el modelo 3D para que sea lo más ligero posible**

Cuando trabajamos con Blender y tenemos que hacer renders nos arriesgamos a que consuma mucho tiempo, por lo que tendremos que procurar que nuestro modelo sea lo más ligero posible. Durante este proyecto nos surgió un problema de eficiencia ya que creábamos un objeto cubo diferente para cada uno de los vértices de nuestra malla, por lo que el render tardaba horas, incluso en alguna ocasión días. Cuando nos dimos cuenta buscamos una solución que mejorara el rendimiento del renderizado y decidimos usar un único cubo que iríamos duplicando por cada uno de los vértices de la malla, es decir, sólo crearíamos un cubo y no uno por cada vértice. Este simple cambio nos ahorró horas de renderizado por lo que conseguimos mayor eficiencia.

Lección aprendida: al diseñar el proyecto debemos prever que habrá que optimizar los modelos.

- **Blender como herramienta de representación de fenómenos físicos**

Con otros programas no tenemos tantas posibilidades de representación como con Blender. Éstos nos ofrecen la opción de representar o bien el punto cuántico, o bien la función de onda. Gracias a poder programar en Blender somos capaces de incorporar los datos que se han calculado en los simuladores científicos dentro de una imagen. Ahí está la verdadera potencia del programa, ya que podemos representar todo aquello que queramos tal como queramos y así crear imágenes donde están contenidos los resultados que se han calculado en la investigación realizada.

Una vez finalizado el proyecto y puesto a prueba, podemos concluir que se ha desarrollado de forma satisfactoria, consiguiendo para cada uno de los problemas planteados:

Los objetivos principales

- Una **interfaz de usuario** que favorece la introducción de datos por parte del usuario para personalizar la animación
- Una **biblioteca de representación gráfica** para resolver los problemas físicos

- Los **fotogramas** que componen el vídeo de la animación 3D
- El **vídeo** que facilita el entendimiento del problema

Y además el objetivo secundario de la creación de un **código más modular y reutilizable**, lo que hace un código más legible y nos da la posibilidad de usar dichos módulos en otros programas.

A continuación plasmamos unas imágenes resultantes de la ejecución de cada uno de los problemas:

RRAM

La variación de temperatura que experimenta el filamento viene representada por el cambio de color según se demuestra en las siguientes figuras.

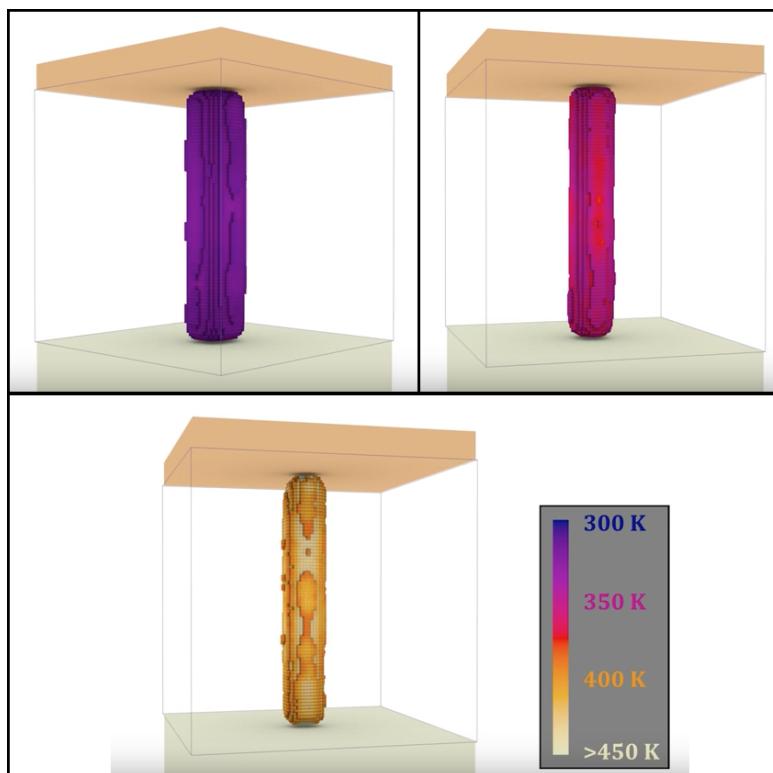


Figura 7.1: Variación de color según temperatura

Puntos cuánticos

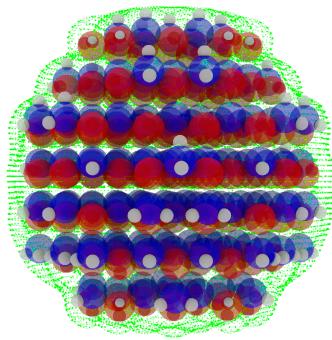


Figura 7.2: Isosuperficie del módulo al cuadrado de la función de onda (densidad de probabilidad)

A continuación se muestran los enlaces donde están subidos los vídeos resultantes de cada uno de los problemas.

Problema de las RRAM: https://youtu.be/t5yAR_VfMws

Problema de los puntos cuánticos: <https://youtu.be/xopr3ajz12E>

Además tanto la memoria como los archivos Blender generados durante este proyecto han sido subidos a Github en el siguiente repositorio:

<https://github.com/Laurasmus/3D-Libraries/>

Capítulo 8

Trabajos futuros

El presente trabajo está abierto a posibles mejoras y cambios. Hay muchas opciones de trabajos futuros ya que son proyectos de investigación, adecuándose al ritmo de trabajo de los científicos y de su demanda.

Éstos trabajos los podemos considerar desde dos perspectivas diferentes:

1. Añadir casos totalmente nuevos a la biblioteca de representación gráfica 3D.

Dada la amplia cantidad de fenómenos físicos que son motivo de estudio hoy en día, es seguro que surgirá la necesidad de hacer representaciones 3D de dichos fenómenos.

2. Desde el punto de vista de los fenómenos que han sido estudiados en este proyecto.

Algunas mejoras que se podrían implementar en un futuro en el problema de las RRAM son:

- Representar sistemas donde las vacantes de oxígeno se pudieran desplazar en el tiempo, dado que este fenómeno físico se ha observado que puede ocurrir, y no simplemente que vayan apareciendo conforme se van generando dichas vacantes
- Representar el campo eléctrico, que ya se calcula en el simulador en módulo. Sería interesante mostrar este campo por la dificultad que representa visualizar un campo vectorial junto con las partículas que lo producen
- Mostrar el potencial en el sistema, que tendrá relación con el potencial aplicado entre los electrodos y, por tanto, también con la corriente inducida

Y en el problema de los puntos cuánticos se podría implementar:

- Que la transparencia vaya cambiando a lo largo del vídeo
- Tener la opción de superponer dos funciones de onda, con distintos colores del material halo, para calcular transiciones de un estado a otro viendo cómo se solapan las funciones de onda

- Adaptar el programa al estudio de láminas bidimensionales de puntos cuánticos de semiconductor

Además completar los objetivos opcionales que no se han cumplido en este proyecto, como son:

- Creación de animaciones apoyadas por los scripts
- Creación de un sistema render distribuído

Bibliografía

- [1] SCRIPT, <https://es.wikipedia.org/wiki/Script>
Última revisión: 18 de Diciembre de 2015
- [2] RRAM, https://es.wikipedia.org/wiki/Resistive_random-access_memory
Última revisión: 18 de Diciembre de 2015
- [3] PUNTO CUÁNTICO, https://es.wikipedia.org/wiki/Punto_cuantico
Última revisión: 18 de Diciembre de 2015
- [4] SIMULACIÓN, <https://es.wikipedia.org/wiki/Simulacion>
Última revisión: 18 de Diciembre de 2015
- [5] PERCOLACIÓN, https://en.wikipedia.org/wiki/Percolation_theory
Última revisión: 19 de Diciembre de 2015
- [6] MATLAB, <http://es.mathworks.com/products/matlab>
Última revisión: 19 de Diciembre de 2015
- [7] BLENDER, <https://www.blender.org>
Última revisión: 19 de Diciembre de 2015
- [8] SCHRÖDINGER, https://es.wikipedia.org/wiki/Ecuación_de_Schrödinger
Última revisión: 20 de Diciembre de 2015
- [9] RASMOL, <http://www.openrasmol.org>
Última revisión: 20 de Diciembre de 2015
- [10] JMOL, <http://jmol.sourceforge.net>
Última revisión: 20 de Diciembre de 2015
- [11] PYTHON, <https://www.python.org>
Última revisión: 20 de Diciembre de 2015
- [12] LATEX, <https://www.latex-project.org>
Última revisión: 12 de Febrero de 2016
- [13] METODOLOGÍA AGIL, https://es.wikipedia.org/wiki/Desarrollo_agil_de_software
Última revisión: 12 de Febrero de 2016

- [14] DESARROLLO ITERATIVO E INCREMENTAL, <https://proyectosagiles.org/desarrollo-iterativo-incremental>
Última revisión: 12 de Febrero de 2016
- [15] CASOS DE USO, https://es.wikipedia.org/wiki/Caso_de_uso
Última revisión: 12 de Febrero de 2016
- [16] INTERFAZ DE USUARIO, https://es.wikipedia.org/wiki/Interfaz_de_usuario
Última revisión: 13 de Mayo de 2016
- [17] IDE, https://es.wikipedia.org/wiki/Entorno_de_desarrollo_integrado
Última revisión: 13 de Mayo de 2016
- [18] CAMEL CASE, <https://es.wikipedia.org/wiki/CamelCase>
Última revisión: 14 de Mayo de 2016
- [19] RGB, <https://es.wikipedia.org/wiki/RGB>
Última revisión: 14 de Mayo de 2016
- [20] PRUEBAS DE SOFTWARE, https://es.wikipedia.org/wiki/Pruebas_de_software
Última revisión: 14 de Mayo de 2016

APÉNDICES

Apéndice A

Installation Manual

A.1. Blender installation

To launch our program we have to install Blender, a 3D modeling program. You can download it from the 'download section' of its website: <https://www.blender.org/download/>. It is recommended to download the version **2.77a** to ensure the proper functioning of the program.



Figura A.1: Download Blender

We choose our operating system (Windows, Mac OS X or GNU / Linux), we choose a download mirror among those appearing to the right belonging to the architecture of your computer and it automatically starts downloading. Once the download is complete we proceed to install.

A.1.1. Install in Windows

Run the .exe file that you just have downloaded and press Next in the next window.

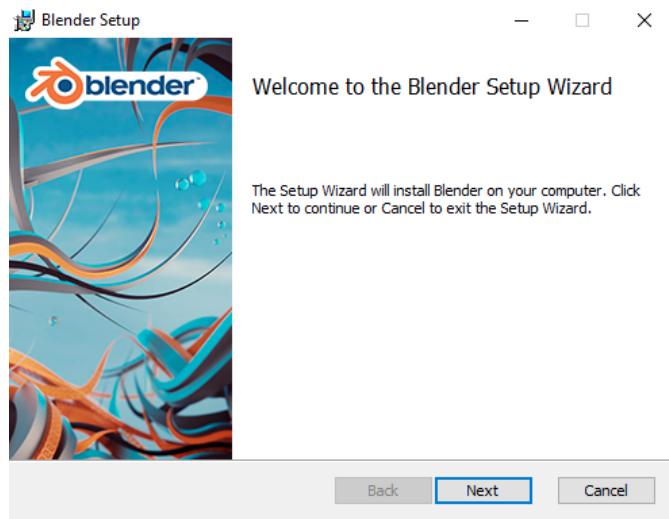


Figura A.2: Blender Setup

On the next screen we accept the terms and we press Next.

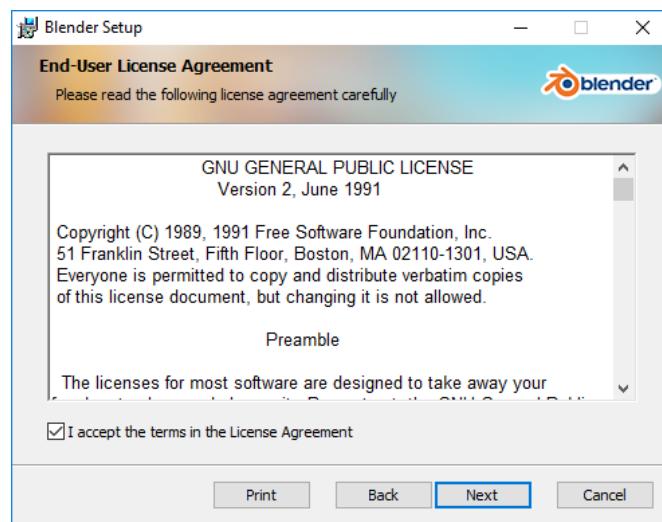


Figura A.3: End-User License Agreement

If you want to change the location you can do it by clicking on 'Browse...'. To use the default location we continue by clicking on Next again.

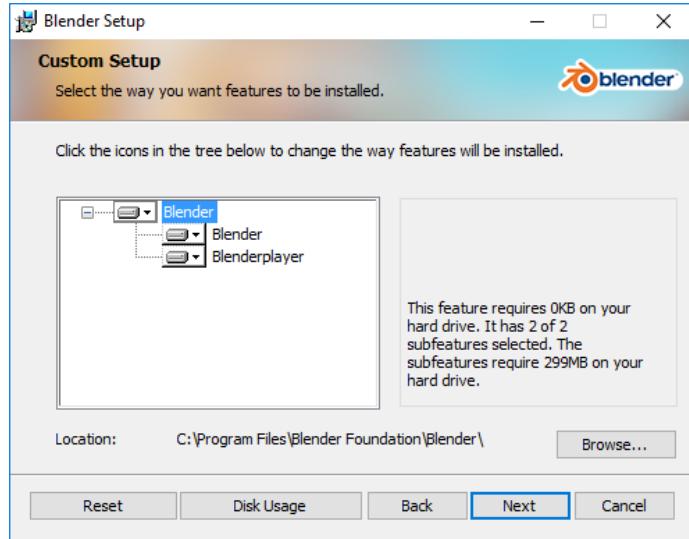


Figura A.4: Custom Setup

To finish the installation process click on Install.

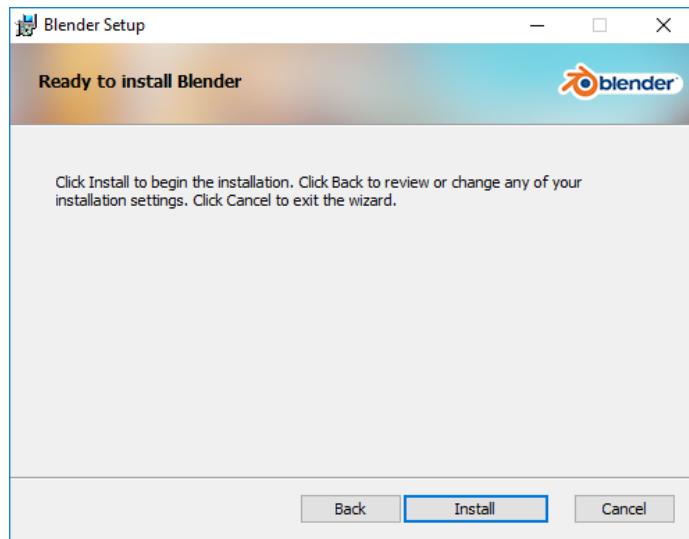


Figura A.5: Ready to install Blender

For a few seconds we will see how the installation is being completed.

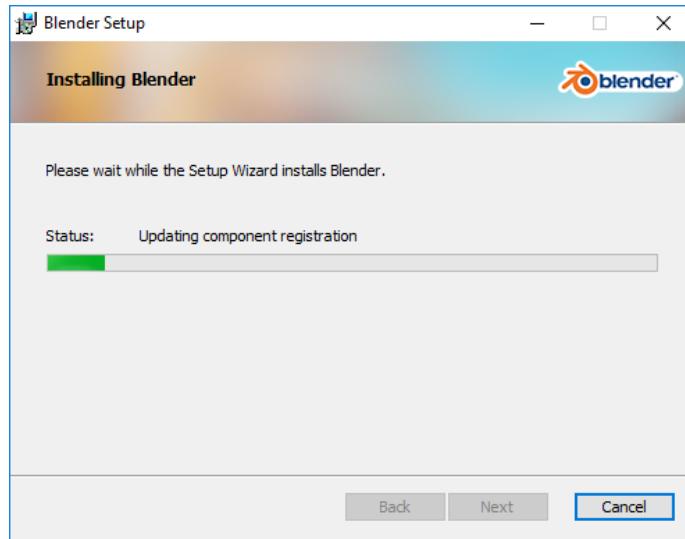


Figura A.6: Installing Blender

When the next window appears we already have Blender installed on our machine.

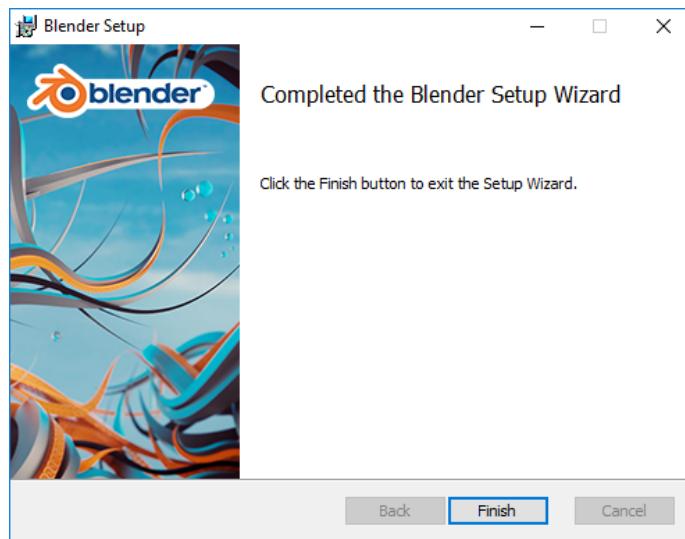


Figura A.7: Completed the Blender setup

Press the Finish button and we can proceed to open Blender.

A.2. Blender configuration

To run our script we must do a number of changes in our Blender. Once opened Blender we go to *File > User Preferences...*

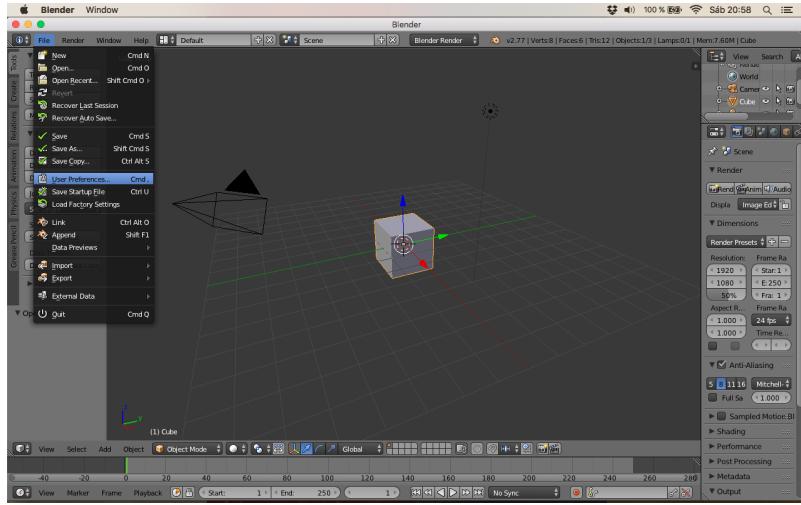


Figura A.8: Blender User Preferences

In the new window we will go to the *File* tab, check the option *Auto Run Python Scripts* and uncheck the *Relative Paths* option.

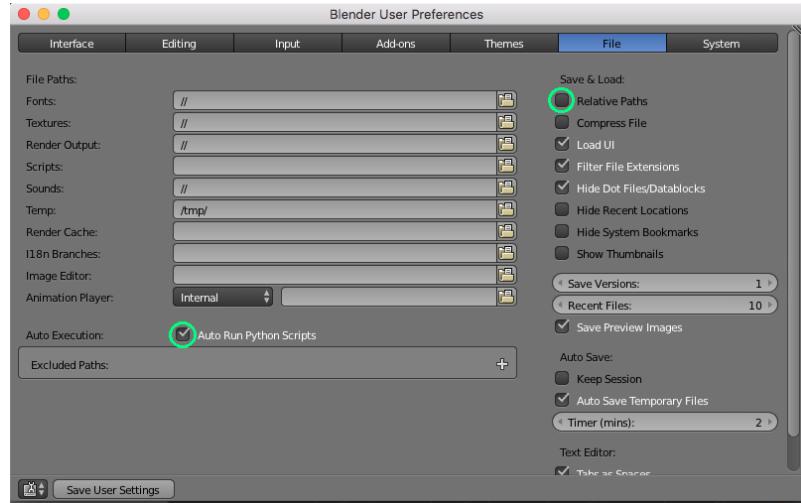


Figura A.9: Auto Run Python Scripts

Once these changes are done, we click on the *Save User Settings* button at the bottom left of the screen preferences. We can close the preference screen.

Apéndice B

RRAM User Manual

The following will show how to use the library for representation of the RRAM.

B.1. First contact

To launch the application we have to open the file RRAM.blend, or by double-clicking from the file, or from the application in *File > Open...* and look for the file on your machine. Opening the file we find the following interface.

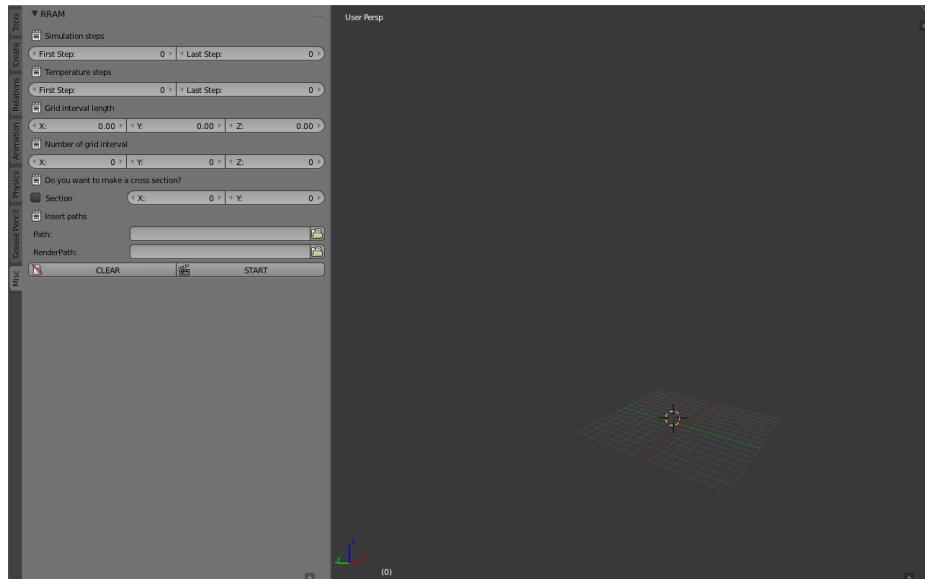


Figura B.1: Interface RRAM

B.2. Fill the form

Then we describe each of its elements:

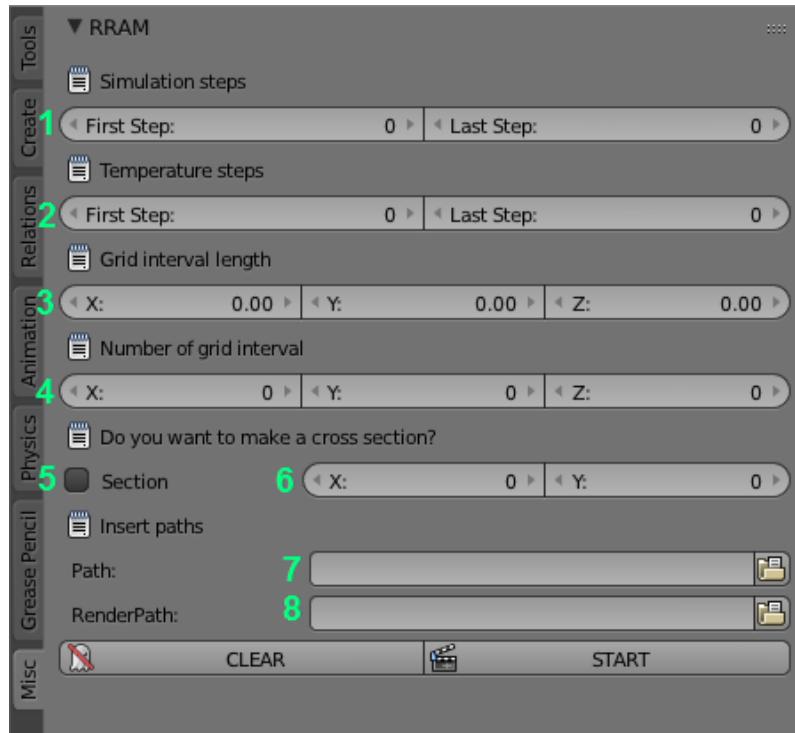


Figura B.2: Form RRAM

Then we describe each of its elements:

1. **First / Last Simulation Step:** First and last file you want to use in the simulation
2. **First / Last Temperature Step:** first and last temperature file
3. **Distance X/Y/Z:** gap between two grid points
4. **Grid X/Y/Z:** dimensions of the grid
5. **Section:** if you want a section in the representation check this box
6. **Section X/Y:** section position in the X axis and Y axis
7. **Path:** folder where we have our data
8. **RenderPath:** folder where the images are stored and the video at the end of rendering

We need the next structure of folders and files.

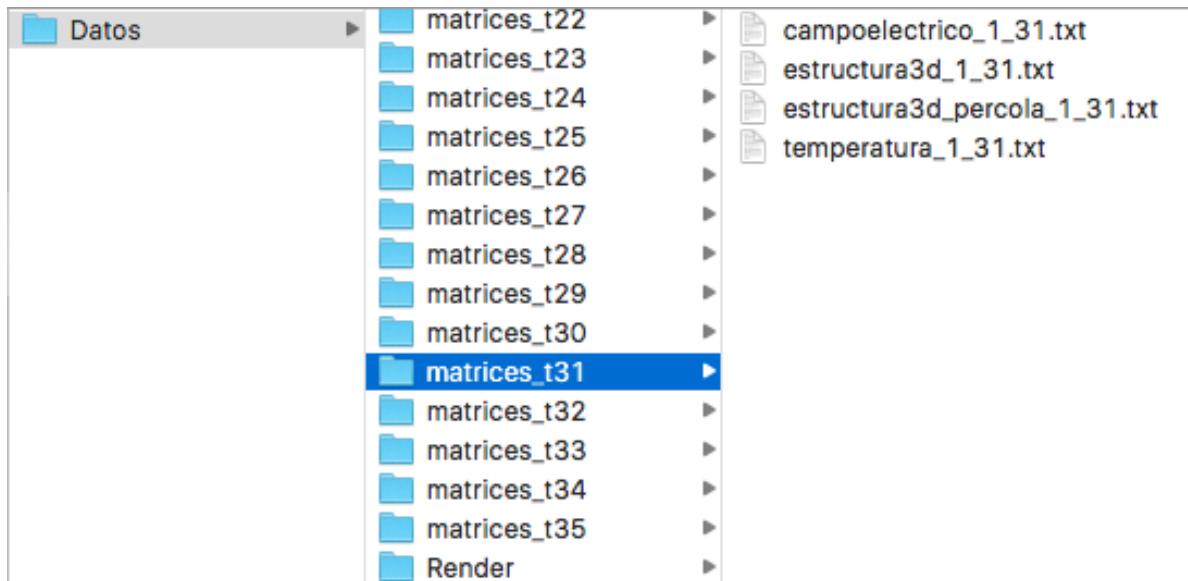


Figura B.3: Folders

Apéndice C

QD User Manual

The following will show how to use the library for representation of the Quantum Dots.

C.1. First contact

To launch the application we have to open the file RRAM.blend, or by double-clicking from the file, or from the application in *File >Open...* and look for the file on your machine. Opening the file we find the following interface.

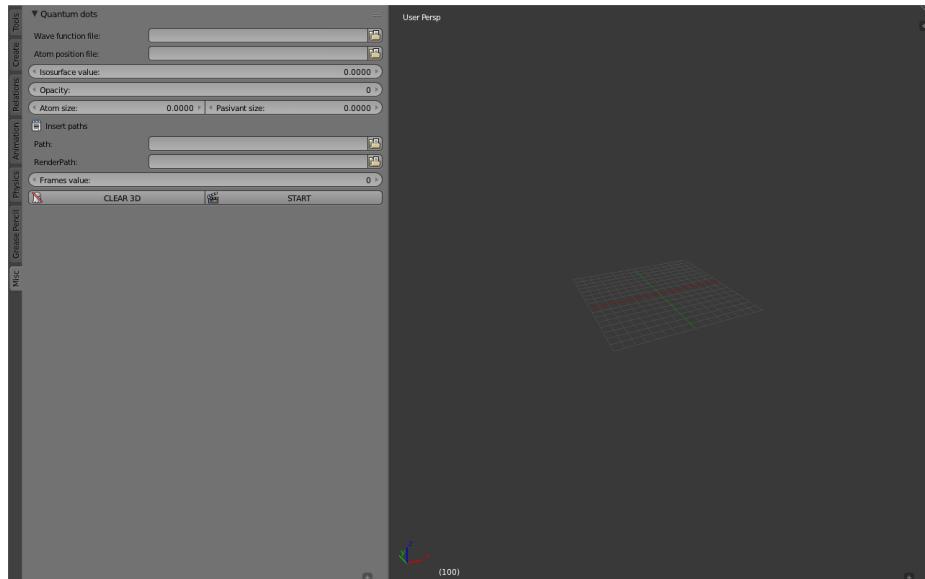


Figura C.1: Interface QD

C.2. Fill the form

To the left of the screen we can see a form, within the Misc tab.

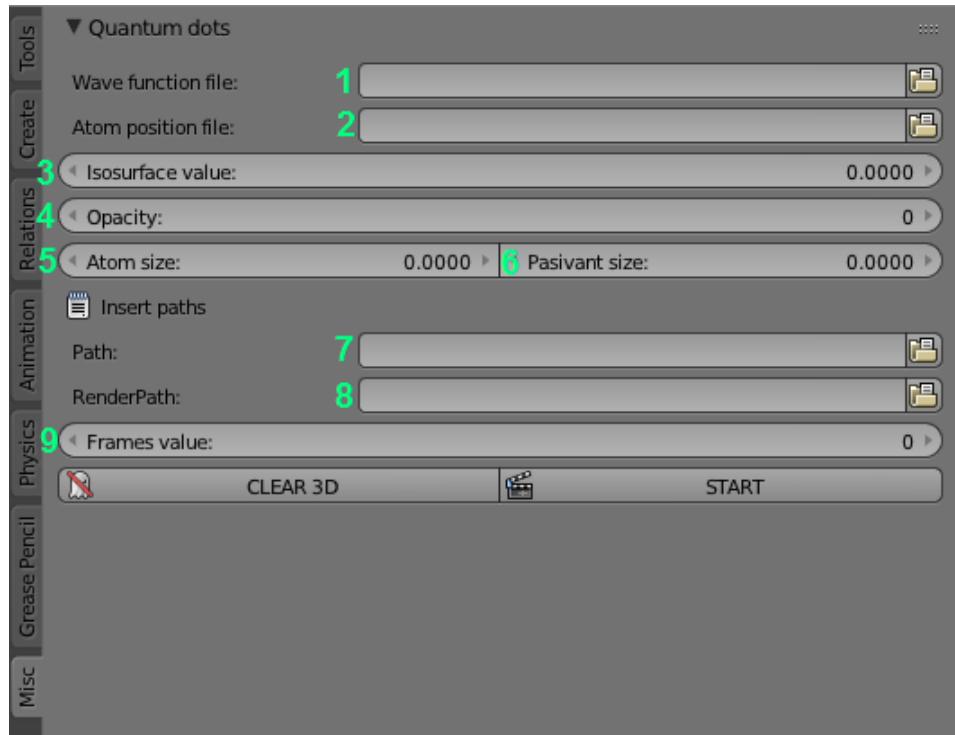


Figura C.2: Form QD

Then we describe each of its elements:

1. **Wave function filename:** file with the wave function data
2. **Atom position filename:** data file with the positions of the atoms
3. **Isosurface value**
4. **Atom size**
5. **Pasivant size**
6. **Path:** folder where we have our data
7. **RenderPath:** folder where the images are stored and the video at the end of rendering
8. **Frames value:** numbers of frames that we want in our video