Escuela
Politécnica
Superior

LLMSearch: Buscador multimedia basado en lenguaje natural

Grado en Ingeniería Informática

Trabajo Fin de Grado

Autor:
Izan Gandía Ruiz
Tutor:
Iván Gadea Saéz

Universitat d'Alacant
Universidad de Alicante

Mayo 2025

LLMSearch: Buscador multimedia basado en lenguaje natural

Qué pongo aquí???

Autor
Izan Gandía Ruiz

Tutor
Iván Gadea Saéz
Departamento del tutor???

Grado en Ingeniería Informática

Escuela
Politécnica
Superior

Universitat d'Alacant
Universidad de Alicante

ALICANTE, Mayo 2025

Preámbulo

Las razones que me han llevado a la realización de este proyecto han sido tanto el aprendizaje en profundidad de las inteligencias artificiales multimodales como la propia frustración que he tenido durante muchos años a la hora de buscar un fichero específico, como una imagen, sobre una enorme variedad de documentos en mi teléfono u ordenador, aunque este problema se presenta también en situaciones más modernas como podría ser encontrar el "sticker" en apps de mensajería como WhatsApp o Telegram.

Agradecimientos

Me encantaría aprovechar este apartado para darle las gracias a las decenas de personas que me han acompañado a lo largo de la carrera y de las cuales he podido aprender cientos de cosas tanto académicas como lecciones de vida, sin todos ellos no habría tenido la motivación suficiente para seguir en muchos momentos de mi vida y creo que son, en gran parte, la razón de que pueda estar aquí escribiendo esto. Del mismo modo, agradecer a mi familia por confiar siempre en mí y por darme todas las facilidades que han podido para permitirme avanzar, en especial a mi hermano Abel Gandía pues él fue quien me animó y me ayudó a entrar en el maravilloso mundo de la programación cuando yo aun no sabía nada. Me gustaría agradecer también a mis profesores, pues algunos han sido de gran inspiración y me han ayudado a ver la belleza en campos donde nunca me habría imaginado que estaría, y a mi Tutor Iván Gadea Sáez por supervisar este proyecto y ayudarme con todas las preocupaciones que he tenido. Gracias.

A quienes me inspiraron a soñar y a programar, recordándome que, si puedo imaginarlo, puedo crearlo. [1]

[1] Alejandro Taboada, creador del canal "Programación ATS"

Índice general

Índice de figuras

Índice de tablas

Índice de Códigos

1. Introducción (Con ejemplos de contenido)

Antes de comenzar la lectura de este documento debo agradecer el trabajo realizado por Pedro Pernías Peco en su plantilla de "tfg" que se puede ver en `https://github.com/lcg51/tfg`. Gracias a esa plantilla me he lanzado a crear mi versión. Algunos contenidos aquí mostrados han sido extraídos de la plantilla de Pedro.

Esta plantilla se ha diseñado de 0 y por ello no utiliza la misma estructura que la plantilla de Pedro. Pero la estructura de contenido para un TFG/TFM es la misma y a continuación se muestran las diferentes partes que debe tener un TFG/TFM redactado por Pedro.

1.1. ¡Importante!, leer primero

Este texto está escrito pensando en orientar a los alumnos que usarán LaTeX para escribir su Trabajo Final de Grado (TFG) y Trabajo Final de Máster (TFM).

Contiene información útil para aquellos que no tengan experiencia previa en LaTeX así como algunos datos acerca de cómo escribir mejor su TFG. A continuación, se ofrece una copia de la información que hay en el libro de estilo para la realización de los TFG de la EPS de la Universidad de Alicante.

En los capítulos siguientes encontrarás ejemplos de muchas de las cosas que se pueden realizar con LaTeX. Con un poco de paciencia, estudia cómo se hacen estas cosas y luego aplícalas en tus documentos.

1.2. Estructura de un TFG

En caso de que el TFG/TFM tenga como finalidad la elaboración de un proyecto o un informe científico o técnico, deberá ajustarse a lo dispuesto en las normas UNE 157001:2002 y UNE 50135:1996 respectivamente.

Si el TFG/TFM tiene por finalidad la elaboración de un trabajo monográfico, el documento presentado deberá constar de las siguientes partes, teniendo como base la norma UNE 50136:1997.

Preámbulo: se describirán brevemente la motivación que ha originado la realización del TFG/TFM, así como una breve descripción de los objetivos generales que se quieren alcanzar con el trabajo presentado.

Agradecimientos: se podrán añadir las hojas necesarias para realizar los agradecimientos, a veces obligatorios, a las entidades y organismos colaboradores.

Dedicatoria: se podrá añadir una única hoja con dedicatorias, su alineación será derecha.

Citas: (frases célebres) se podrá añadir una única hoja con citas, su alineación será derecha.

Índices: cada índice debe comenzar en una nueva página, se incluirán los índices que se estimen necesarios (conforme UNE 50111:1989), en este orden:

Índice de contenidos: (obligatorio siempre) se incluirá un índice de las secciones de las que se componga el documento, la numeración de las divisiones y subdivisiones utilizarán cifras arábigas (según UNE 50132:1994) y harán mención a la página del documento donde se ubiquen.

Índice de figuras: si el documento incluye figuras se podrá incluir también un índice con su relación, indicando la página donde se ubiquen.

Índice de tablas: en caso de existir en el texto, ídem que el anterior.

Índice de abreviaturas, siglas, símbolos, etc.: en caso de ser necesarios se podrán incluir cada uno de ellos.

Cuerpo del documento: en el contenido del documento se da flexibilidad para su organización y se puede estructurar en las secciones que se considere. En todo caso obligatoriamente se deberá, al menos, incluir los siguientes contenidos:

Introducción: donde se hará énfasis a la importancia de la temática, su vigencia y actualidad; se planteará el problema a investigar, así como el propósito o finalidad de la investigación.

Marco teórico o Estado del arte: se hará mención a los elementos conceptuales que sirven de base para la investigación, estudios previos relacionados con el problema planteado, etc.

Objetivos: se establecerán el objetivo general y los específicos.

Metodología: se indicarán el tipo o tipos de investigación, las técnicas y los procedimientos que serán utilizados para llevarla a cabo; se identificarán la población y el tamaño de la muestra así como las técnicas e instrumentos de recolección de datos.

Resultados: incluirá los resultados de la investigación o trabajo, así como el análisis y la discusión de los mismos.

Conclusiones: obligatoriamente se incluirá una sección de conclusiones donde se realizará un resumen de los objetivos conseguidos así como de los resultados obtenidos si proceden.

Bibliografía y referencias: se incluirá también la relación de obras y materiales consultados y empleados en la elaboración de la memoria del TFG/TFM. La bibliografía y las referencias serán indexadas en orden alfabético (sistema nombre y fecha) o se numerará correlativamente según aparezca (sistema numérico). Se empleará la familia 1 como tipo de letra. Podrá utilizarse cualquier sistema bibliográfico normalizado predominante en la rama de conocimiento, estableciéndose como prioritarios el sistema ISO 690, sistema American Psychological Association (APA) o Harvard (no necesariamente en ese orden de preferencia). En esta plantilla Latex se propone usar el estilo APA indicándolo en la línea correspondiente como

```
\bibliographystyle{apacite}
```

Anexos: se podrán incluir los anexos que se consideren oportunos.

1.3. Apartados dentro de los capítulos

En LaTeX existen diferentes niveles de títulos para realizar secciones, subsecciones, etc. En esta web puedes ver más información al respecto `https://en.wikibooks.org/wiki/LaTeX/Document_Structure`

Para ello se utilizan los siguientes comandos;

```
\section{Esto es una sección}
Y este el contenido de la sección.
\subsection{Esto es una subsección}
Y este el contenido de la subsección.
\subsubsection{Esto es una subsubsección}
Y este el contenido de la subsubsección.
\paragraph{Esto es un paragraph}
Y este el contenido del paragraph. Que siempre se inicia en la misma línea que el título del mismo.
```

Y se genera lo siguiente:

1.4. Esto es una sección

Y este el contenido de la sección.

1.4.1. Esto es una subsección

Y este el contenido de la subsección.

1.4.1.1. Esto es una subsubsección

Y este el contenido de la subsubsección.

1.4.1.1.1. Esto es un paragraph Y este el contenido del paragraph. Que siempre se inicia en la misma línea que el título del mismo.

1.5. Citar bibliografía

Para citar la bibliografía tal como se define en el sistema APA (en esta web se indica como debe aparecer en el texto la cita: `http://guides.libraries.psu.edu/apaquickguide/intext`) se debe realizar con alguno de los comandos mostrados a continuación:

```
Esto es una cita estándar: \citet{Shaw1996}, que también puedes mostrar con paréntesis así: \citep{Shaw
    ↪ 1996}. También se puede realizar una cita indicando a qué parte te refieres \citep[ver][Cap. 2]{Shaw
    ↪ 1996} o \citep[Cap. 2]{Shaw1996} o \citep[ver][]{Shaw1996}.

También puedes mostrar todos los autores cuando hay más de 2 autores añadiendo un asterisco después del
    ↪ comando como: \citet*{Akyildiz2005}, sin el asterisco quedaría así: \citet{Akyildiz2005}.
```

```
O puedes citar dos o más fuentes al mismo tiempo: \citep{Barkan1995,Leighton2012}
```

Y LaTeX genera lo siguiente:

Esto es una cita estándar: ?, que también puedes mostrar con paréntesis así: (?). También se puede realizar una cita indicando a qué parte te refieres (ver ?, Cap. 2) o (?, Cap. 2) o (ver ?).

También puedes mostrar todos los autores cuando hay más de 2 autores añadiendo un asterisco después del comando como: ?, sin el asterisco quedaría así: ?.

O puedes citar dos o más fuentes al mismo tiempo: (??)

1.6. Notas a pie de página

Para introducir notas a pie de página se debe escribir lo siguiente:

```
La plantilla necesita el motor XeLaTeX \footnote{Para más información sobre XeLaTeX visita \url{https
    ↪ ://es.sharelatex.com/learn/XeLaTeX}} (el más recomendable actualmente), por lo que si el
    ↪ programa que utilizas compila la plantilla con el motor pdfLaTeX \footnote{También puedes
    ↪ buscar más información en internet} (el más habitual pero menos potente) debes cambiarlo por
    ↪ XeLaTeX en las opciones del programa. Si no sabes como hacerlo busca en el manual del programa
    ↪  o en google.
```

LaTeX genera lo siguiente (observa las notas a pie de página):

La plantilla necesita el motor XeLaTeX[1] (el más recomendable actualmente), por lo que si el programa que utilizas compila la plantilla con el motor pdfLaTeX[2] (el más habitual pero menos potente) debes cambiarlo por XeLaTeX en las opciones del programa. Si no sabes como hacerlo busca en el manual del programa o en google.

1.7. Estilos de texto

A continuación se muestran ejemplos de distintos estilos de texto:

- \textit{Cursiva} → *Cursiva*
- \emph{Cursiva 2} → *Cursiva 2*
- \textbf{Negrita} → **Negrita**
- \texttt{Monoespacio} → `Monoespacio`
- \textsc{Mayúsculas capitales} → MAYÚSCULAS CAPITALES
- \uppercase{Todo mayúsculas} → TODO MAYÚSCULAS

[1] Para más información sobre XeLaTeX visita `https://es.sharelatex.com/learn/XeLaTeX`
[2] También puedes buscar más información en internet

1.8. Acrónimos

Ahora vamos a ver cómo se ponen los acrónimos.

La norma dice que la primera vez que aparece un acrónimo debe ponerse su fórmula completa, es decir lo que significa, al lado del acrónimo. Después de ello, podemos usar sólo el acrónimo salvo cuando consideremos que debemos volver a usar la fórmula completa por alguna razón de legibilidad.

¿Cómo llevar la cuenta de cuándo es la primera vez que ponemos el acrónimo? si hacemos cambios en el doc es fácil que perdamos esa información así que lo mejor es que sea el propio LaTeX el que lleve esa cuenta. Para ello tenemos que hacer dos cosas:

Primero: creamos la entrada del acrónimo en el fichero acronimos.tex. Revisa los comentarios de su cabecera para saber cómo crear esa entrada. Básicamente lo que hacemos allí es poner la "fórmula corta" y la "fórmula larga" del acrónimo es decir, el propio acrónimo y su significado

Segundo: escribimos en el texto el acrónimo SIEMPRE diciendo que es un acrónimo y el tipo de fórmula que queremos usar. Por ejemplo, si siempre que queremos hacer referencia al IEEE escribimos

```
\gls{ieee}
```

se consigue que la primera vez que aparezca el acrónimo ponga las fórmulas larga y corta y en las siguientes ocasiones sólo aparecerá la corta.

Aquí va un ejemplo:

Si escribimos:

```
El \gls{ieee} es una institución muy importante en el mundo de la
ingeniería. El \gls{ieee} lleva marcando normas y protocolos desde
hace mucho tiempo. Pero el \gls{ieee} no está solo en esta tarea.
Además del \gls{ieee} hay muchas otras instituciones para ello.
```

Obtendremos:

El Institute of Electrical and Electronics Engineers (IEEE) es una institución muy importante en el mundo de la ingeniería. El IEEE lleva marcando normas y protocolos desde hace mucho tiempo. Pero el IEEE no está solo en esta tarea. Además del IEEE hay muchas otras instituciones para ello.

1.9. Tareas por hacer

En esta plantilla se ha incluido un paquete para incluir notas/comentarios en el texto para recordar partes que hay que revisar o terminar de desarrollar. El uso es sencillo, el manual para conocer todos los comandos se encuentra en `http://osl.ugr.es/CTAN/macros/latex/contrib/todonotes/todonotes.pdf`, a continuación se muestran algunos ejemplos:

Para incluir un comentario sobre el texto:

```
Recomiendo utilizar programas LaTeX que permitan trabajar con sistema de archivos para poder editar el
↪ conjunto de capítulos en la misma ventana. Este tipo de función lo tienen programas como
↪ TexStudio, es multiplataforma. \todo{Incluir más ejemplos de programas}
```

LaTeX genera lo siguiente:

Recomiendo utilizar programas LaTeX que permitan trabajar con sistema de archivos para poder editar el conjunto de capítulos en la misma ventana. Este tipo de función lo tienen programas como TexStudio, es multiplataforma.

Incluir más ejemplos de programas

Para incluir un comentario sobre el texto pero dentro del texto:

```
Recomiendo utilizar programas LaTeX que permitan trabajar con sistema de archivos para poder editar el
↪ conjunto de capítulos en la misma ventana. Este tipo de función lo tienen programas como
↪ TexStudio, es multiplataforma. \todo[inline]{Incluir más ejemplos de programas}
```

LaTeX genera lo siguiente:

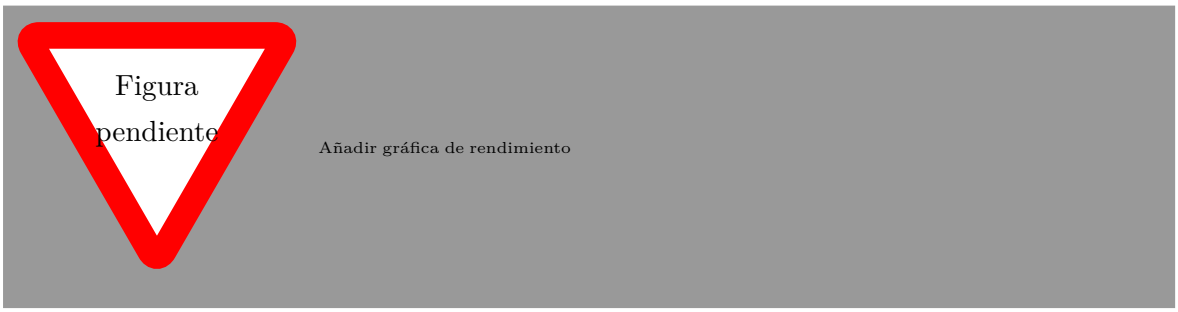
Recomiendo utilizar programas LaTeX que permitan trabajar con sistema de archivos para poder editar el conjunto de capítulos en la misma ventana. Este tipo de función lo tienen programas como TexStudio, es multiplataforma.

Incluir más ejemplos de programas

También se puede dejar indicado donde falta una imagen o figura, para incluirla más adelante del siguiente modo:

```
\missingfigure{Añadir gráfica de rendimiento}
```

LaTeX genera lo siguiente:



2. Estado del Arte

2.1. Orquestador de tareas

Herramientas para la automatización y gestión de flujos de trabajo.

2.1.1. Prefect

Plataforma de orquestación de flujos de trabajo en Python que permite diseñar, ejecutar y monitorizar pipelines de datos y machine learning de forma fiable y escalable.

2.1.1.1. Ventajas

- **Facilidad de uso:** Prefect ofrece una sintaxis intuitiva y una configuración sencilla, lo que facilita la definición y gestión de flujos de trabajo complejos.
- **Flexibilidad:** Permite la orquestación de tareas en entornos locales, en la nube o híbridos, adaptándose a diversas necesidades.
- **Monitoreo y gestión:** Incluye herramientas integradas para el monitoreo, registro y manejo de errores en tiempo real.

2.1.1.2. Desventajas

- **Madurez:** Aunque ha ganado popularidad, Prefect es relativamente nuevo en comparación con otras herramientas más consolidadas.
- **Comunidad:** Su comunidad es más pequeña, lo que puede limitar la disponibilidad de recursos y soporte.

2.1.2. Kafka

Sistema de mensajería distribuido de alto rendimiento.

2.1.2.1. Ventajas

- **Alto rendimiento:** Kafka es conocido por su capacidad para manejar grandes volúmenes de datos con baja latencia.
- **Escalabilidad:** Diseñado para escalar horizontalmente, puede manejar cargas de trabajo crecientes de manera eficiente.
- **Ecosistema robusto:** Cuenta con una amplia gama de herramientas y conectores que facilitan su integración con otros sistemas.

2.1.2.2. Desventajas

- **Complejidad:** La configuración y gestión de Kafka pueden ser complejas, especialmente para usuarios sin experiencia previa.
- **Requisitos de recursos:** Para un rendimiento óptimo, Kafka suele requerir una infraestructura robusta, lo que puede ser excesivo para proyectos más pequeños.

2.1.3. Airflow

Plataforma para crear, programar y monitorear flujos de trabajo.

2.1.3.1. Ventajas

- **Popularidad y comunidad:** Amplia adopción y una comunidad activa que proporciona numerosos recursos y soporte.
- **Flexibilidad:** Permite la programación y monitoreo de flujos de trabajo complejos.

2.1.3.2. Desventajas

- **Curva de aprendizaje:** Puede ser complejo de configurar y requiere conocimientos avanzados para su implementación efectiva.

2.2. Script de detección de cambios en un path y Servidor

Herramientas para monitorear cambios en el sistema de archivos.

2.2.1. Python

Lenguaje de programación versátil con varias bibliotecas para detección de cambios.

- **Watchdogs:** Biblioteca multiplataforma diseñada específicamente para detectar eventos en el sistema de archivos.
- **pyinotify:** Biblioteca que proporciona monitoreo de eventos del sistema de archivos en sistemas Linux aprovechando el subsistema inotify. Es eficiente para entornos Linux pero no es multiplataforma.
- **inotify-simple:** Un wrapper sencillo alrededor de la API inotify de Linux, que ofrece simplicidad y facilidad de uso para tareas básicas de monitoreo de archivos en sistemas Linux.
- **inotifyx:** Similar a pyinotify, esta biblioteca proporciona acceso al sistema inotify de Linux, permitiendo monitorear eventos del sistema de archivos. Está diseñada para tener una API estable pero también está limitada a plataformas Linux.
- **Polling Methods:** Para plataformas donde inotify no está disponible, o para requisitos más simples, implementar un mecanismo de sondeo puede ser una alternativa viable. Esto implica verificar periódicamente el sistema de archivos para detectar cambios.

2.2.2. Node

Entorno de ejecución para JavaScript con opciones para monitoreo de archivos.

- **chokidar:** Biblioteca eficiente para vigilar cambios en el sistema de archivos.

2.2.3. Java

Lenguaje de programación con APIs nativas para monitoreo.

- **WatchService:** API integrada en Java para monitorear cambios en directorios.

2.2.4. C++/C/C#

Familia de lenguajes con herramientas para vigilancia de archivos.

- **FileSystemWatcher:** Clase para monitorear cambios en el sistema de archivos.

2.2.5. Go

Lenguaje de programación con bibliotecas específicas.

- **fsnotify:** Biblioteca Go para monitoreo del sistema de archivos.

2.2.6. Rust

Lenguaje de programación moderno con enfoque en seguridad.

- **notify:** Biblioteca Rust para vigilar cambios en archivos.

2.3. Base de datos

Opciones para almacenamiento de datos.

2.3.1. Relacional

Bases de datos con estructura definida y relaciones entre tablas.

2.3.1.1. SQLite

Base de datos relacional ligera contenida en un único archivo.

2.3.1.1.1. Ventajas

- **Ligereza y simplicidad:** SQLite es una biblioteca de base de datos integrada que no requiere una configuración de servidor independiente, lo que facilita su implementación y uso.
- **Portabilidad:** Al almacenar toda la base de datos en un único archivo, es fácil de transferir y gestionar, especialmente útil para aplicaciones móviles o integradas.
- **Rendimiento en entornos de bajo recurso:** Funciona eficientemente incluso en sistemas con recursos limitados.

2.3.1.1.2. Desventajas

- **Concurrencia limitada:** SQLite permite múltiples lecturas simultáneas, pero las escrituras se gestionan de una en una, lo que puede ser un cuello de botella en aplicaciones con alta concurrencia de escritura.
- **Escalabilidad:** No está diseñada para manejar grandes volúmenes de datos o aplicaciones que requieren escalabilidad horizontal.

2.3.1.2. MariaDB

Sistema de gestión de bases de datos fork de MySQL.

2.3.1.2.1. Ventajas

- **Rendimiento y escalabilidad:** MariaDB ofrece un alto rendimiento y puede manejar una gran cantidad de transacciones, siendo adecuada para aplicaciones empresariales.
- **Compatibilidad con MySQL:** Como un fork de MySQL, mantiene una alta compatibilidad, facilitando la migración desde MySQL.
- **Soporte para almacenamiento en columnas:** Incluye el motor ColumnStore, optimizado para cargas de trabajo analíticas.

2.3.1.2.2. Desventajas

- **Complejidad en la configuración:** Requiere una configuración y gestión más complejas en comparación con SQLite.
- **Requisitos de recursos:** Necesita más recursos del sistema, lo que puede ser excesivo para aplicaciones pequeñas o integradas.

2.3.2. No relacional

Bases de datos con esquemas flexibles no basados en tablas relacionales.

2.3.2.1. MongoDB

Base de datos NoSQL orientada a documentos.

2.3.2.1.1. Ventajas

- **Flexibilidad del esquema:** Al ser una base de datos NoSQL orientada a documentos, permite almacenar datos en un formato flexible similar a JSON, adaptándose fácilmente a cambios en la estructura de los datos.
- **Escalabilidad horizontal:** Diseñada para escalar horizontalmente mediante sharding, lo que facilita el manejo de grandes volúmenes de datos y altas tasas de tráfico.
- **Alto rendimiento en operaciones de lectura/escritura:** Optimizada para manejar operaciones simultáneas de lectura y escritura de manera eficiente.

2.3.2.1.2. Desventajas

- **Consumo de recursos:** Requiere una cantidad significativa de recursos, especialmente en implementaciones a gran escala.
- **Falta de soporte para transacciones complejas:** Aunque MongoDB ha mejorado en este aspecto, las transacciones en múltiples documentos pueden no ser tan robustas como en bases de datos relacionales.

2.4. Docker

Plataforma de contenedores para desarrollo, envío y ejecución de aplicaciones.

2.4.1. Ventajas

- **Portabilidad:** Los contenedores Docker aseguran que el software se ejecute de manera consistente en cualquier entorno.
- **Aislamiento:** Cada componente del sistema puede ejecutarse en su propio contenedor, evitando conflictos de dependencias.
- **Facilidad de despliegue:** Simplifica la distribución y actualización de aplicaciones.

2.4.2. Desventajas

- **Consumo de recursos:** Aunque ligero, el uso de contenedores añade una capa adicional que consume recursos del sistema.
- **Complejidad adicional:** Requiere conocimientos sobre Docker y la gestión de contenedores.

2.5. Interfaz

Frameworks para el desarrollo de interfaces web.

2.5.1. Angular

Framework completo para desarrollo de aplicaciones web.

2.5.1.1. Ventajas

- **Framework completo:** Angular ofrece una solución integral con herramientas integradas para el desarrollo de aplicaciones web robustas.
- **Arquitectura estructurada:** Facilita la escalabilidad y el mantenimiento de aplicaciones complejas.

2.5.1.2. Desventajas

- **Curva de aprendizaje pronunciada:** Requiere tiempo para dominar conceptos como TypeScript y la inyección de dependencias.
- **Complejidad innecesaria para proyectos simples:** Puede ser excesivo para aplicaciones con funcionalidades limitadas.

2.5.2. React

Biblioteca de JavaScript para construcción de interfaces de usuario.

2.5.2.1. Ventajas

- **Biblioteca flexible:** React se centra en la construcción de interfaces de usuario, permitiendo la integración con diversas bibliotecas según las necesidades del proyecto.
- **Amplia comunidad y recursos:** Su popularidad asegura una gran cantidad de recursos y soporte.

2.5.2.2. Desventajas

- **Necesidad de configuraciones adicionales:** Para funcionalidades más allá de la UI, es necesario integrar bibliotecas adicionales, lo que puede aumentar la complejidad.

2.5.3. Vue

Framework progresivo para construir interfaces de usuario.

2.5.3.1. Ventajas

- **Simplicidad y facilidad de uso:** Vue es conocido por su curva de aprendizaje suave, lo que permite una adopción rápida.
- **Flexibilidad:** Adecuado tanto para proyectos pequeños como para aplicaciones más complejas.

2.5.3.2. Desventajas

- **Menor adopción en grandes empresas:** Aunque está ganando popularidad, Vue aún no es tan común en entornos corporativos grandes.

2.5.4. Astro

Framework moderno para sitios web con enfoque en rendimiento.

2.5.4.1. Ventajas

- **Optimización para contenido estático:** Astro está diseñado para generar sitios web estáticos rápidos, lo que puede ser beneficioso para aplicaciones con contenido predominantemente estático.
- **Integración con otros frameworks:** Permite utilizar componentes de React, Vue y otros dentro de sus proyectos.

2.5.4.2. Desventajas

- **Menor madurez:** Al ser relativamente nuevo, Astro puede carecer de la amplitud de recursos y comunidad que tienen otros frameworks más establecidos.

3. Objetivos

El objetivo de este proyecto es diseñar y desarrollar un buscador multimedia que permita a los usuarios realizar búsquedas avanzadas utilizando lenguaje natural de manera que, el usuario pueda buscar documentos de texto, imágenes, vídeos o archivos de audio usando el lenguaje natural para explorar el contenido de los ficheros, no solo los metadatos de los archivos. La idea es crear una herramienta que permita a los usuarios encontrar contenido multimedia de manera eficiente y precisa utilizando descripciones detalladas en lenguaje natural. Por ejemplo, se podría buscar una fotografía entre miles con una descripción como "busca una foto en la que salía un elefante levantando la trompa y que la hice en Tailandia hace unos 5 o 6 años" o encontrar un archivo PDF con una búsqueda del tipo "encuentra los datos para la declaración de la renta de 2016". De esta forma, se obtendría un buscador que además de encontrar el archivo específico que se busca podría extraer los datos relevantes del archivo para responder a preguntas específicas hechas en la consulta.

4. Metodología

4.1. Organización

La metodología que se va a seguir es Scrum Adaptado (con un toque de Kanban). Scrum es una metodología ágil que facilita la gestión de proyectos complejos a través de equipos autoorganizados que trabajan en ciclos de desarrollo (sprints) de forma iterativa e incrementa y que promueve la entrega continua de valor. La adaptación en este caso consiste en que yo asumiré el rol de Product Owner, Desarrollador y Scrum Master, mientras mi tutor actúa como cliente (requisitos). Se harán sprints de aproximadamente 2 semanas desde el inicio del cuatrimestre donde se verán avances, se resolverán dudas y se definirán los objetivos del siguiente sprint.

4.2. Apartado técnico

Explicar que cosas he elegido y por qué + my hardware

5. Análisis, Especificación y diseño

5.1. Requisitos del sistema

En esta sección se detallan los requisitos del sistema, divididos en requisitos funcionales, no funcionales y de configuración. Los requisitos se han estructurado en formato tabular para facilitar su comprensión y seguimiento durante el desarrollo del proyecto.

5.1.1. Requisitos funcionales

Los requisitos funcionales describen el comportamiento que debe tener el sistema, las funcionalidades que debe ofrecer y las operaciones que debe realizar.

ID	Nombre	Descripción
RF-01	Detección de ficheros	El sistema debe detectar nuevos ficheros en el directorio observado.
RF-02	Diferenciación de tipos	El sistema debe diferenciar el tipo de archivo a analizar (texto, imagen, vídeo, audio, otros).
RF-03	Ejecución de modelos	El sistema debe ejecutar el modelo correspondiente que extraerá la información del fichero a la base de datos.
RF-04	Almacenamiento	El sistema debe almacenar todos los datos posibles sobre el fichero analizado en una base de datos.
RF-05	Interfaz web	El sistema debe tener una interfaz web super-simple donde el usuario podrá escribir su consulta en lenguaje natural y darle a un botón para realizar la búsqueda.
RF-06	Resultados de búsqueda	El sistema responderá con un conjunto de resultados potencialmente interesantes a partir de la consulta de búsqueda, ordenados de más a menos ”interesante”.
RF-07	Entrada por línea de comandos	El sistema debe tener una entrada por línea de comandos (ej: `LLMSearch --query "mapa del mundo en el que hay marcados los mejores parques naturales"`).
RF-08	Presentación de resultados	El resultado será la ruta del fichero junto a una pequeña descripción del mismo (enlaces clicables al fichero y a la carpeta que lo contiene).
RF-09	Inspección de archivos comprimidos	Los ficheros comprimidos deberían poder inspeccionarse por dentro.
RF-10	Tipos de ficheros a procesar	El sistema debe procesar los siguientes tipos de ficheros: - Documentos de texto - Imágenes - Vídeos - Ficheros de sonido - Otros (bases de datos, ejecutables, etc.)

Tabla 5.1: Requisitos funcionales del sistema

5.1.2. Requisitos no funcionales

Los requisitos no funcionales especifican criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos.

ID	Nombre	Descripción
RNF-01	Configuración web	La web debe tener una pequeña parte de configuración discreta pero accesible en todo momento.
RNF-02	Arquitectura modular	La arquitectura se debe dividir en un "buscador" y un "explorador" y deben ser completamente separadas para poder ser reutilizadas.
RNF-03	Ejecución sin GPU	El sistema debe poder ejecutarse en un ordenador sin GPU (opcional).
RNF-04	Parámetro de consulta	La entrada por línea de comandos aceptará un parámetro `--query` junto al término de búsqueda.
RNF-05	Resultados en CLI	La entrada por línea de comandos devolverá los resultados de la misma manera que el buscador web con la diferencia de que solo devolverá información adicional si se le añade el parámetro `--verbose`.
RNF-06	Estado del sistema	La entrada por línea de comandos tendrá un parámetro `--status` que devolverá el estado del sistema: número de archivos procesados sobre el número total de archivos en observación, cantidad de ficheros de cada tipo, errores encontrados...
RNF-07	Configuración por CLI	Se añadirán los parámetros necesarios para poder configurar el sistema desde línea de comandos.

Tabla 5.2: Requisitos no funcionales del sistema

5.1.3. Requisitos de configuración

Los requisitos de configuración especifican las opciones que el usuario debe poder ajustar en el sistema.

ID	Nombre	Descripción
RC-01	Directorio de observación	Directorio donde se están observando nuevos ficheros.
RC-02	Regulación de carga	Regular la carga (limitar la CPU al X%).
RC-03	Tipo de modelo LLM	Tipo de modelo LLM a utilizar (Local *(LLM Studio)* ó en la nube).
RC-04	Búsqueda por imagen	Posibilidad de poner una foto de una persona y que la busque en los ficheros.

Tabla 5.3: Requisitos de configuración del sistema

5.2. Diagrama de arquitectura del sistema

A continuación se presenta un diagrama de la arquitectura del sistema que muestra la división entre el "buscador" y el "explorador" según el requisito no funcional RNF-02.

5.3. Casos de uso

Los casos de uso describen las interacciones típicas entre los usuarios y el sistema, mostrando cómo se utilizarían las funcionalidades principales.

6. Análisis, Especificación y diseño

6.1. Requisitos del sistema

En esta sección se detallan los requisitos del sistema, divididos en requisitos funcionales, no funcionales y de configuración. Los requisitos se han estructurado en formato tabular para facilitar su comprensión y seguimiento durante el desarrollo del proyecto.

6.1.1. Requisitos funcionales

Los requisitos funcionales describen el comportamiento que debe tener el sistema, las funcionalidades que debe ofrecer y las operaciones que debe realizar.

ID	Nombre	Descripción
RF-01	Detección de ficheros	El sistema debe detectar nuevos ficheros en el directorio observado.
RF-02	Diferenciación de tipos	El sistema debe diferenciar el tipo de archivo a analizar (texto, imagen, vídeo, audio, otros).
RF-03	Ejecución de modelos	El sistema debe ejecutar el modelo correspondiente que extraerá la información del fichero a la base de datos.
RF-04	Almacenamiento	El sistema debe almacenar todos los datos posibles sobre el fichero analizado en una base de datos.
RF-05	Interfaz web	El sistema debe tener una interfaz web super-simple donde el usuario podrá escribir su consulta en lenguaje natural y darle a un botón para realizar la búsqueda.
RF-06	Resultados de búsqueda	El sistema responderá con un conjunto de resultados potencialmente interesantes a partir de la consulta de búsqueda, ordenados de más a menos ”interesante”.
RF-07	Entrada por línea de comandos	El sistema debe tener una entrada por línea de comandos (ej: `LLMSearch --query "mapa del mundo en el que hay marcados los mejores parques naturales"`).
RF-08	Presentación de resultados	El resultado será la ruta del fichero junto a una pequeña descripción del mismo (enlaces clicables al fichero y a la carpeta que lo contiene).
RF-09	Inspección de archivos comprimidos	Los ficheros comprimidos deberían poder inspeccionarse por dentro.
RF-10	Tipos de ficheros a procesar	El sistema debe procesar los siguientes tipos de ficheros: - Documentos de texto - Imágenes - Vídeos - Ficheros de sonido - Otros (bases de datos, ejecutables, etc.)

Tabla 6.1: Requisitos funcionales del sistema

6.1.2. Requisitos no funcionales

Los requisitos no funcionales especifican criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos.

ID	Nombre	Descripción
RNF-01	Configuración web	La web debe tener una pequeña parte de configuración discreta pero accesible en todo momento.
RNF-02	Arquitectura modular	La arquitectura se debe dividir en un ”buscador” y un ”explorador” y deben ser completamente separadas para poder ser reutilizadas.
RNF-03	Ejecución sin GPU	El sistema debe poder ejecutarse en un ordenador sin GPU (opcional).
RNF-04	Parámetro de consulta	La entrada por línea de comandos aceptará un parámetro `--query` junto al término de búsqueda.
RNF-05	Resultados en CLI	La entrada por línea de comandos devolverá los resultados de la misma manera que el buscador web con la diferencia de que solo devolverá información adicional si se le añade el parámetro `--verbose`.
RNF-06	Estado del sistema	La entrada por línea de comandos tendrá un parámetro `--status` que devolverá el estado del sistema: número de archivos procesados sobre el número total de archivos en observación, cantidad de ficheros de cada tipo, errores encontrados...
RNF-07	Configuración por CLI	Se añadirán los parámetros necesarios para poder configurar el sistema desde línea de comandos.

Tabla 6.2: Requisitos no funcionales del sistema

6.1.3. Requisitos de configuración

Los requisitos de configuración especifican las opciones que el usuario debe poder ajustar en el sistema.

ID	Nombre	Descripción
RC-01	Directorio de observación	Directorio donde se están observando nuevos ficheros.
RC-02	Regulación de carga	Regular la carga (limitar la CPU al X%).
RC-03	Tipo de modelo LLM	Tipo de modelo LLM a utilizar (Local *(LLM Studio)* ó en la nube).
RC-04	Búsqueda por imagen	Posibilidad de poner una foto de una persona y que la busque en los ficheros.

Tabla 6.3: Requisitos de configuración del sistema

6.2. Diagrama de arquitectura del sistema

A continuación se presenta un diagrama de la arquitectura del sistema que muestra la división entre el ”buscador” y el ”explorador” según el requisito no funcional RNF-02.

6.3. Casos de uso

Los casos de uso describen las interacciones típicas entre los usuarios y el sistema, mostrando cómo se utilizarían las funcionalidades principales.

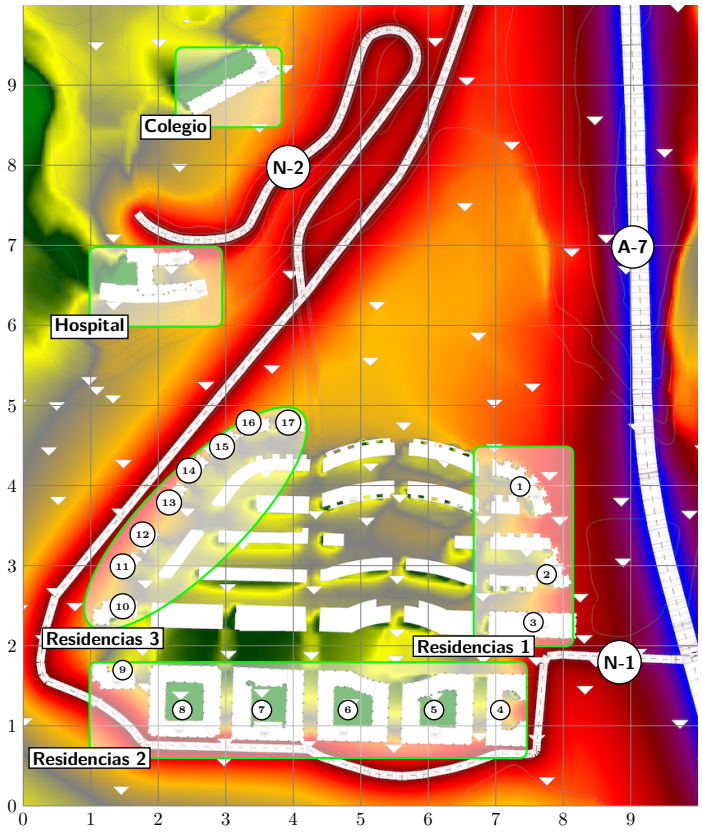
7. Resultados (Con ejemplos de gráficos)

7.1. Diagramas

Gracias al paquete *Tikz* se pueden incluir multitud de medios gráficos, diagramas, capas sobre imágenes, etc. Existen múltiples formas de realizarlo, para ello es recomendable consultar la guía de iniciación disponible aquí: `http://cremeronline.com/LaTeX/minimaltikz.pdf` y también el manual completo disponible aquí: `http://osl.ugr.es/CTAN/graphics/pgf/base/doc/pgfmanual.pdf`.

A continuación se muestran algunos ejemplos. Revisa el archivo .tex para ver cómo se utilizan.

Imagen a la que se le ha añadido cuadros y texto desde latex:



En muchas ocasiones es necesario realizar un diagrama de bloques, más abajo se muestra

un ejemplo de ello. En la red hay multitud de ejemplos que pueden ser fácilmente modificables para un fin concreto, como por ejemplo en esta web: `http://www.texample.net/tikz/examples/tag/block-diagrams/`.

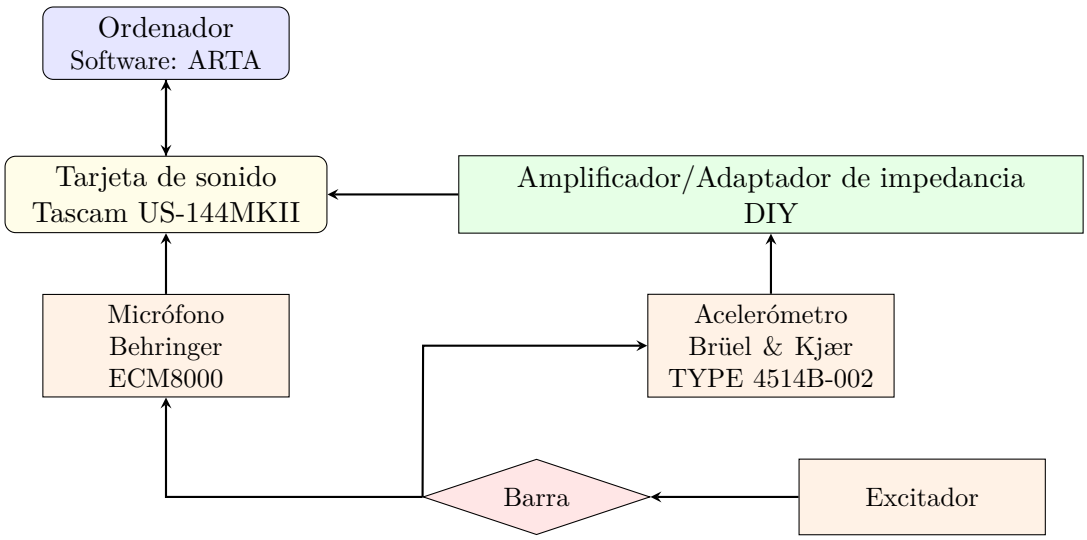


Figura 7.1: Diagrama realizado en latex con Tikz.

7.2. Gráficas

Existen múltiples formas de generar gráficas para latex. Hay disponibles herramientas como GeoGebra que dispone de la utilidad para exportar los gráficos en formato Tkiz. También funciones para Matlab que genera las gráficas que muestra habitualmente pero en código para Tkiz.

7.2.1. Línea

La forma más simple, aunque no sencilla cuando abarca muchos datos es la siguiente:

```
\begin{figure}[ht]
\centering
    \begin{tikzpicture}
        \begin{axis}
            [ymin=0,ymax=5, % Límites del eje y
            xmin=0,xmax=6, % Límites del eje x
            ylabel= eje Y,  % Nombre del eje y
            xlabel= eje X] % Nombre del eje x
            \addplot+[smooth] coordinates % Une los puntos curva suavizada
            {(0,0) (1,2) (2,3 (4,3))}; % Puntos de la gráfica
        \end{axis}
    \end{tikzpicture}
\caption{Gráfica sencilla.}
\end{figure}
```

El resultado es el siguiente:

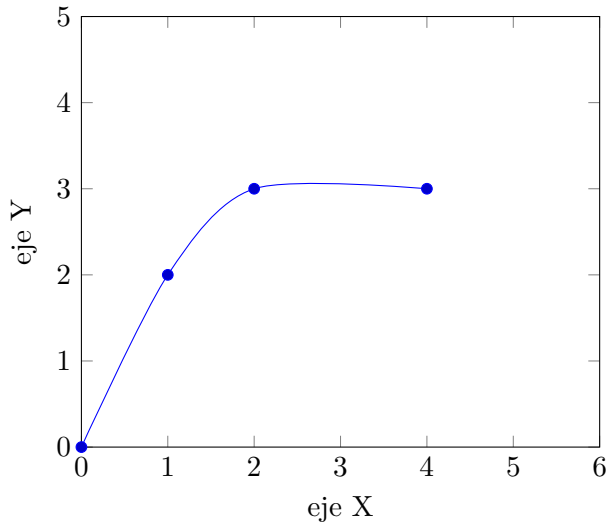


Figura 7.2: Gráfica sencilla.

Otro ejemplo, en este caso las lineas están calculadas directamente en LaTex y después cada una tiene una anotación (el código se encuentra en el archivo archivos/ejemplos/perjudicialesopticacentro.tex):

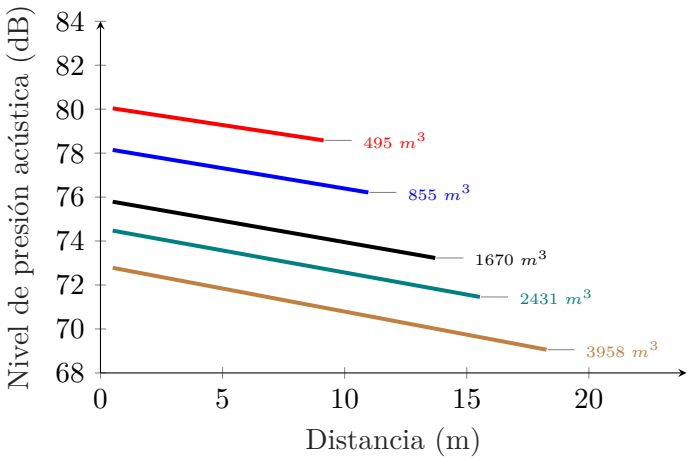


Figura 7.3: OP/S003

7.2.2. Barras

Otro ejemplo es la gráfica de barras:

```
\begin{figure}[ht]
\centering
\begin{tikzpicture}
    \begin{axis}[
        ybar=12pt,
```

```
        ymin=0,ymax=150,
        xtick=data,
        enlarge x limits={abs=2cm},
        symbolic x coords={rubio, moreno},
        bar width = 20pt,
        ylabel= número,
        xlabel= color de pelo,
            ytick align=outside,
            ytick pos=left,
            major x tick style = transparent,
            legend style={at={(0.04,0.96)},anchor=north west, font=\footnotesize, legend cell align=left,},
            ]
        \addplot[ybar,fill=blue, area legend] coordinates {
            (rubio,20)
            (moreno,100)};
        \addplot[ybar,fill=purple, area legend] coordinates {
            (rubio,110)
            (moreno,105)};
    \legend{Chicos, Chicas}
    \end{axis}
\end{tikzpicture}
\caption{Gráfica barras.}
\end{figure}
```

El resultado es el siguiente:

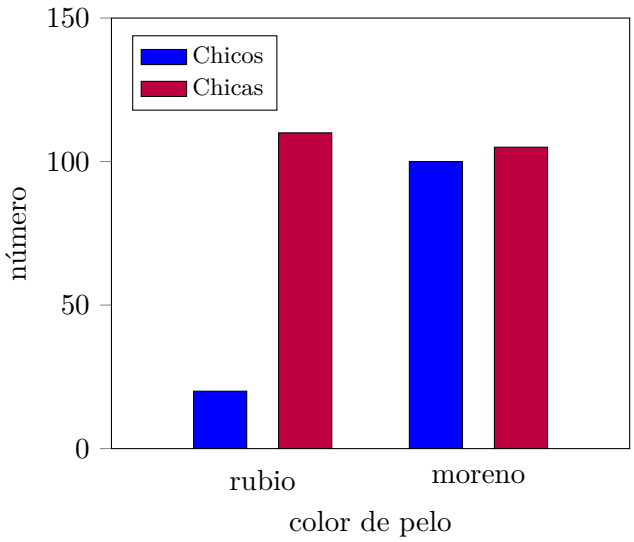


Figura 7.4: Gráfica barras.

7.2.3. Polar

Un ejemplo de gráfica polar semicircular (ver archivo archivos/ejemplos/polarnorm.tex):

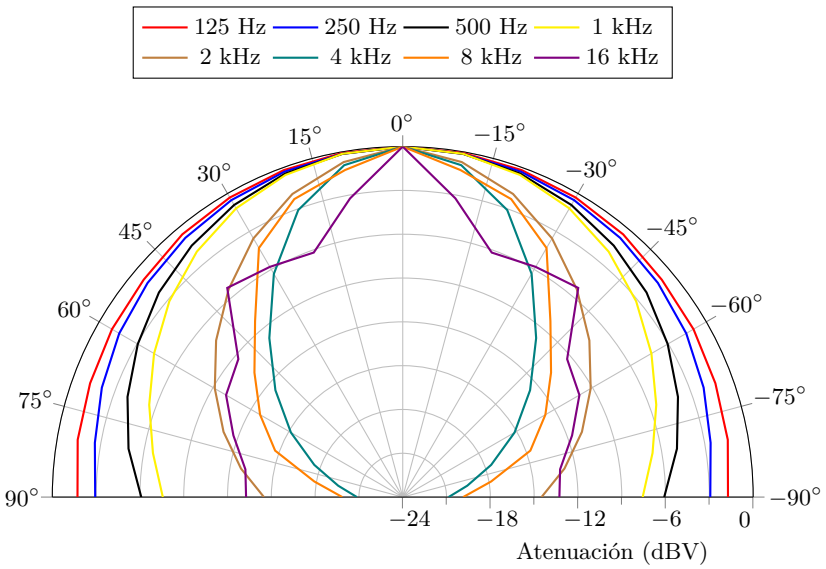


Figura 7.5: Directividad normalizada del altavoz (0 dBV en el eje).

7.3. Importados de MATLAB

Gracias a la herramienta *matlab2tikz* (`https://es.mathworks.com/matlabcentral/fileexchange/22022-matlab2tikz-matlab2tikz`) se pueden exportar las gráficas de cualquier tipo de Matlab a latex. Después de incluir los archivos de *matlab2tikz* se debe escribir una llamada después de crear la figura tal que:

Código 7.1: Ejemplo de llamada a matlab2tikz

```
fig = plot(x,y);
matlab2tikz('figurehandle',fig,'NombreArchivo.tex','height','5cm','width','13.5cm','strict',true,'↩
    ↪ showHiddenStrings',true,'showInfo',false)
```

Y para utilizar el archivo generado por la función en este documento:

```
\begin{figure}[ht]
    \centering
    {\scalefont{0.8}\input{archivos/ejemplos/ParedFina} }
    \caption{Ejemplo de gráfica obtenida con matlab2tikz.}
\end{figure}
```

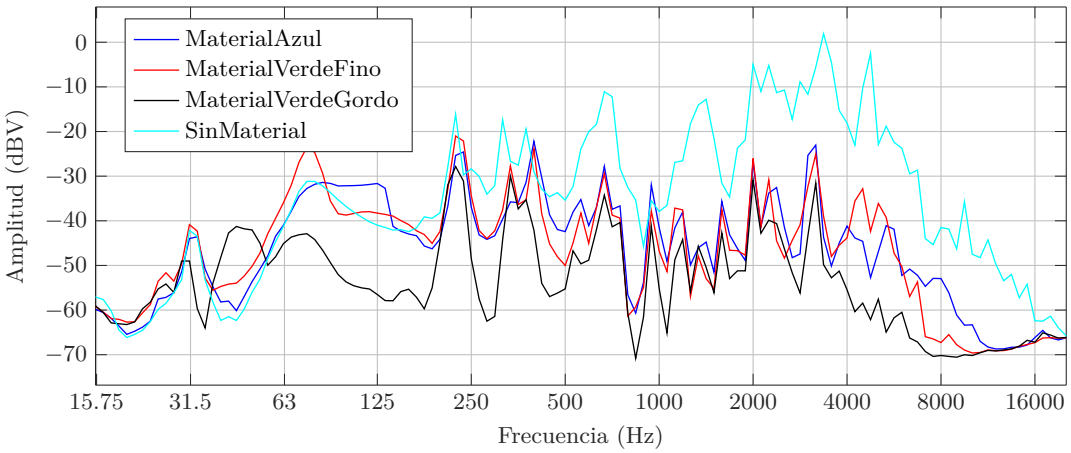


Figura 7.6: Ejemplo de gráfica obtenida con matlab2tikz.

Ejemplo de una gráfica 3D generada en Matlab y exportada por *matlab2tikz*:

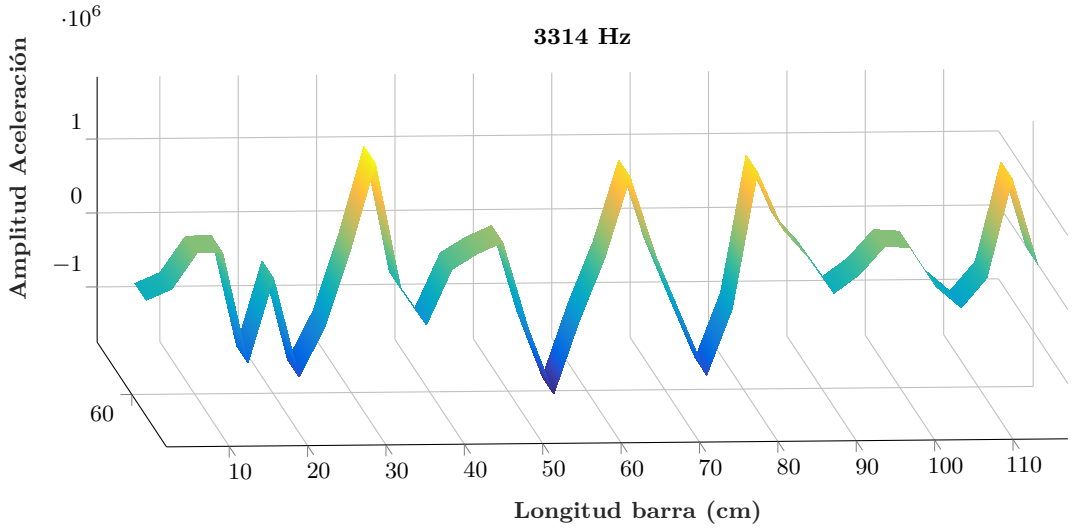


Figura 7.7: Amplitud de la aceleración en el modo número 8.

7.4. Ejemplo avanzado

El potencial del paquete *Tikz* es muy alto, se pueden realizar muchísimas cosas. En la red se facilitan muchos ejemplos para poder ver el funcionamiento y aprender. Existen hilos donde la gente publica sus mejores diseños de *Tikz* como en `https://tex.stackexchange.com/questions/158668/nice-scientific-pictures-show-off` o páginas donde facilitan muchas plantillas como `http://www.texample.net/tikz/examples/all/`.

Un ejemplo de lo que se puede llegar a conseguir es el siguiente:

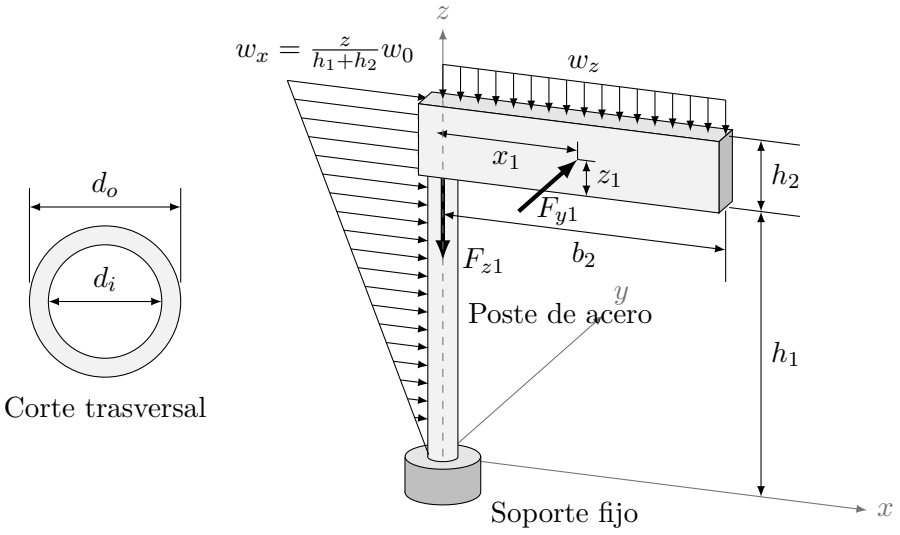


Figura 7.8: Señal realizada con Tikz, sin imágenes.

8. Conclusiones (Con ejemplos de matemáticas)

8.1. Matemáticas

En LaTeX se pueden mostrar ecuaciones de varias formas, cada una de ellas para un fin concreto.

Antes de ver algunas de estas formas hay que conocer cómo se escriben fórmulas matemáticas en LaTeX. Una fuente de información completa es la siguiente: `https://en.wikibooks.org/wiki/LaTeX/Mathematics`. También existen herramientas online que permiten realizar ecuaciones mediante interfaz gráfica como `http://www.hostmath.com/`, `https://www.mathcha.io/editor` o `https://www.latex4technics.com/`

Para mostrar una ecuación numerada se debe utilizar:

```
\begin{equation}
    \nabla\times{\mathbf H}=\left[\frac{1}{r}\frac{\partial}{\partial r}(rH_\theta)-\frac{1}{r}\frac{↩
        ↪ \partial H_r}{\partial\theta}\right]{\hat{\mathbf z}}
    \label{ecuacion}
\end{equation}
```

$$\nabla \times \mathbf{H} = \left[\frac{1}{r}\frac{\partial}{\partial r}(rH_\theta) - \frac{1}{r}\frac{\partial H_r}{\partial \theta}\right]\hat{\mathbf{z}} \tag{8.1}$$

Si es necesario agrupar varias ecuaciones en un mismo índice se puede escribir del siguiente modo:

```
\begin{subequations}
    \begin{eqnarray}
        {\mathbf E}&=&E_z(r,\theta)\hat{\mathbf z}\label{ecu1} \\ % Salto de línea
        {\mathbf H}&=&H_r(r,\theta))\hat{ \mathbf r}+H_\theta(r,\theta)\hat{\bm \theta}\label{ecu2}
    \end{eqnarray}
\end{subequations}
% Se incluye '&' entre la igualdad para centrar las ecuaciones desde el '='.
```

$$\begin{aligned} \mathbf{E} &= E_z(r,\theta)\hat{\mathbf{z}} && \text{(8.2a)} \\ \mathbf{H} &= H_r(r,\theta))\hat{\mathbf{r}} + H_\theta(r,\theta)\hat{\boldsymbol{\theta}} && \text{(8.2b)} \end{aligned}$$

Otras dos formas que son las habituales en muchos lugares para incluir ecuaciones son:

```
Ejemplo de fórmula en línea con el texto $\int_{a}^{b} f(x)dx = F(b) - F(a)$, esta ecuación quedará dentro ↩
    ↪ del texto.

Esta otra, al utilizar dos '\$', se generará en una línea nueva $$\int_{a}^{b} f(x)dx = F(b) - F(a)$$
```

Ejemplo de fórmula en línea con el texto $\int_a^b f(x)dx = F(b) - F(a)$, esta ecuación quedará dentro del texto.

Esta otra, al utilizar dos '$', se generará en una línea nueva

$$\int_a^b f(x)dx = F(b) - F(a)$$

También se puede añadir información adicional a una ecuación con la función *condiciones* creada para esta plantilla:

```
\begin{equation}
    \underset{z=z_0}{\mathrm{Res}}(f(z))=\frac{1}{(m-1)!}\lim_{z \rightarrow z_0}\left[\frac{\text{d↩
        ↪ }^{m-1}}{\text{d}z^{m-1}} \left[\left(z-z_0\right)^m f(z) \right] \right]
\end{equation}

\begin{condiciones}[donde:]
% Excepto 'Descripción y valor' el resto no es necesario el símbolo $para texto matemático.
% Item & Relación & Descripción o valor
    m  & \rightarrow  & Es la multiplicidad del polo $z_0$ \\
    z_0 & \rightarrow  & Es la parte que se iguala a 0 con el polo. \\
    f(z)& \rightarrow  & Es la función contenida en la integral.
\end{condiciones}
```

$$\underset{z=z_0}{\operatorname{Res}}(f(z)) = \frac{1}{(m-1)!}\lim_{z\to z_0}\left[\frac{\mathrm{d}^{m-1}}{\mathrm{d}z^{m-1}}\left[(z-z_0)^m f(z)\right]\right] \tag{8.3}$$

donde: m → Es la multiplicidad del polo z_0
z_0 → Es la parte que se iguala a 0 con el polo.
$f(z)$ → Es la función contenida en la integral.

Si lo que deseas es una ecuación alineada a la izquierda o derecha puedes hacerlo con lo siguiente (el '&' simple es utilizado para alinear las ecuaciones desde ese punto, los iguales):

```
% Alineado a la izquierda al incluir al final el doble '&&'
\begin{flalign}
    y_{h_1}=&\begin{bmatrix}6\cos(\sqrt{6} x) \\ -\sqrt{6}\sin(\sqrt{6}x)\end{bmatrix}e^x &&\\
    y_{h_2}=&\begin{bmatrix}6\sin(\sqrt{6} x) \\ \sqrt{6}cos(\sqrt{6}x)\end{bmatrix}e^x &&
\end{flalign}

% Alineado a la derecha al incluir al inicio el doble '&&'
\begin{flalign}
&& y_{h_1}=&\begin{bmatrix}6\cos(\sqrt{6} x) \\ -\sqrt{6}\sin(\sqrt{6}x)\end{bmatrix}e^x\\
&& y_{h_2}=&\begin{bmatrix}6\sin(\sqrt{6} x) \\ \sqrt{6}cos(\sqrt{6}x)\end{bmatrix}e^x
\end{flalign}
```

$$y_{h_1} = \begin{bmatrix} 6\cos(\sqrt{6}x) \\ -\sqrt{6}\sin(\sqrt{6}x) \end{bmatrix} e^x \tag{8.4}$$

$$y_{h_2} = \begin{bmatrix} 6\sin(\sqrt{6}x) \\ \sqrt{6}cos(\sqrt{6}x) \end{bmatrix} e^x \tag{8.5}$$

$$y_{h_1} = \begin{bmatrix} 6\cos(\sqrt{6}x) \\ -\sqrt{6}\sin(\sqrt{6}x) \end{bmatrix} e^x \tag{8.6}$$

$$y_{h_2} = \begin{bmatrix} 6\sin(\sqrt{6}x) \\ \sqrt{6}cos(\sqrt{6}x) \end{bmatrix} e^x \tag{8.7}$$

Tanto con la función utilizada en (8.1,8.3), como en (8.2a,8.2b) y en las anteriores, si se les incluye un '*' después de 'equation', 'subequation' o 'flalign', se elimina la numeración de las ecuaciones pero manteniendo el resto de características.

A. Anexo I

Aquí vendría el anexo I

B. Páginas horizontales

Aquí se muestra cómo incluir páginas en horizontal.
Esta página está en vertical

Esta página está en horizontal

Esta página también está en horizontal

Esta página está de nuevo en vertical

C. Importar PDF

A continuación se muestra una página importada de un PDF externo. Observar los comentarios en el código de este anexo para más información. También puedes leer el manual con todas las opciones en `http://osl.ugr.es/CTAN/macros/latex/contrib/pdfpages/pdfpages.pdf`.

Alicante 15 DE MARZO DE 2007

Expediente número

Referencia del peticionario **AYUNTAMIENTO DE ALICANTE**

Departamento de Medio Ambiente
C/San Nicolás, nº 2, 4º
03001 ALICANTE
Contacto: Juan Luís Beresaluze

DOCUMENTO DE SÍNTESIS

ELABORACIÓN DEL MAPA ACÚSTICO MUNICIPAL DE LA CIUDAD DE ALICANTE

Fecha de realización del estudio: MAYO 2005 – MARZO 2007