Escuela
Politécnica
Superior

Demo técnica para PC

Grado en Ingeniería Multimedia

Trabajo Fin de Grado

Autor:
Luis González Aracil
Tutor/es:
Francisco José Gallego Durán

Universitat d'Alacant
Universidad de Alicante

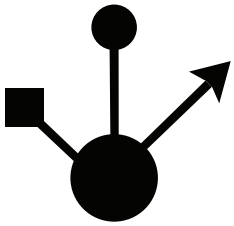
Marzo 2019

Demo técnica para PC

La demoscene

Autor
Luis González Aracil

Tutor/es
Francisco José Gallego Durán
Ciencia de la Computación e Inteligencia Artificial



Grado en Ingeniería Multimedia

Escuela
Politécnica
Superior

Universitat d'Alacant
Universidad de Alicante

ALICANTE, Marzo 2019

Resumen

Me dejo el resumen para el final

Abstract

Some day I will write something in here, hopefuly...

Motivación y objetivo general

Hablando con un compañero del trabajo sobre el lenguaje ensamblador, yo estaba intentando argumentarle su utilidad, a lo que él me contestó *"Hoy en día, saber ensamblador es como saber latín"*. Su afirmación zanjó el tema, pues a partir de ese momento no tuve ganas de seguir discutiendo, pero me hizo reflexionar. ¿Es inútil el ensamblador? ¿Sirve para algo el bajo nivel?

Para mí, la pregunta *"¿Para qué sirve saber ensamblador?"* es perfectamente equiparable a la pregunta "*¿Para qué le sirve a un arquitecto conocer las herramientas y materiales con los que va a construir una casa?"*.

Si una edificación cayera por una mala elección del material de los cimientos por parte del arquitecto, no habría duda en a quién culpar. Nadie abogaría que la culpa no es del arquitecto porque no es su responsabilidad conocer las bases. Sin embargo, hoy en día hay una enorme tendencia en el mundo del desarrollo software por menospreciar o infravalorar los cimientos de la programación, considerándolo algo arcaico y de carácter puramente didáctico, pero no práctico.

Yo me opongo radicalmente a esta visión, no sólo porque estoy convencido de la importancia de conocer el bajo nivel, si no que también encuentro cierta belleza en él. Cómo instrucciones en apariencia tan simples pueden construir sistemas tan complejos. A ello, se suma una gran curiosidad por saber cómo las cosas están hechas, desde el principio.

Una de las cosas que encuentro más apasionantes de la computación es la capacidad de los ordenadores, máquinas inertes y carentes de inteligencia real -por el momento- para reproducir nuestra realidad a partir de modelos matemáticos.

Los gráficos por computador son, por lo general, complejos. Sin embargo, hoy en día es posible crear con un ordenador imágenes que parecen fotografías y son capaces de engañar al ojo humano.

El objetivo principal de este trabajo es ir a las raíces, y revisar algunas de las técnicas que se usaban en los orígenes de los gráficos por computador para, a partir de operaciones con bajo coste computacional, generar escenas complejas.

A mis padres, por estar ahí, siempre.
A mi hermana, por ser mi incordio y mi alegría.
A mi familia y amigos, por apoyarme y alegrarme los días.

A mi tutor,
por la visión que me ha dado sobre el mundo de la programación
y que tan valiosa es.

A Lola, por haber marcado la dirección cuando estábamos perdidos

If you give people
the choice of writing
good code or fast code,
there's something wrong.
Good code should be fast

Bjarne Stroustrup

When the whole world is silent,
even one voice becomes powerful.

Malala Yousafzai

Índice general

Índice de figuras

Índice de tablas

Índice de Códigos

1. Introducción

Sin saber muy bien cómo, hemos llegado a un punto en el que conocer las bases del funcionamiento de un computador nos parece algo obsoleto, e incluso arcaico. Tecnologías de hace 20 años se tachan de reliquias en un mundo que aún no cuenta un siglo de antigüedad.

El irrefrenable avance de la tecnología y velocidad de evolución es innegable, pero a menudo, cuando se avanza muy rápido, también se pierde muy rápido.

La abstracción en el mundo de la computación ha sido un factor clave, de hecho es el factor que ha permitido que un set reducido de instrucciones como el que tienen los ordenadores sea capaz de imitar la realidad. Abstraer el software y llevarlo hacia modelos más cercanos al ser humano ha permitido pensar más en términos de nuestro día a día y menos en términos de mover memoria y realizar sumas y restas. Y esto es bueno, si para realizar cualquier mínima tarea tuviéramos que preocuparnos de hasta el más mínimo detalle de implementación, la curva de aprendizaje sería demasiado inclinada, y la eficiencia de la producción del software caería en picado.

Sin embargo, a más nos alejamos del hardware y más capas de abstracción añadimos, las instrucciones que escribimos se alejan más y más del reducido set de instrucciones que nuestro ordenador puede ejecutar. Como dijo una vez David J. Wheeler, *"Todo problema en computación puede resolverse con otra capa de indirección, excepto el problema de tener demasiada indirección"*[1].

Esta frase, además de tener un punto cómico, plantea un problema más serio del que muchas veces nos damos cuenta. Hoy en día hay aplicaciones construidas dentro de webs. Para ejecutar código en el cliente de una web se usa *JavaScript*, un lenguaje interpretado. Esto significa que para ejecutar una instrucción de código máquina proveniente de *JavaScript* es necesario, primero, interpretar la línea de código, compilarla y traducirla al lenguaje de la máquina virtual para *JavaScript* que integra el navegador y que esta máquina virtual interactúe con el sistema operativo para ejecutar la orden necesaria. Esto obviando una gran cantidad de pasos intermedios e ignorando las propias capas de abstracción de la memoria, el funcionamiento del procesador, los posibles fallos de la caché del procesador... Y si a todo este proceso, ya de por sí complejo y con muchas capas de por medio, añadimos una aplicación compleja que añade nuevas capas de abstracción sobre el propio código que se ejecuta en *JavaScript*... ¿No parece demasiado?

Y sin embargo hoy en día prácticas como esta son perfectamente aceptadas, e incluso a veces, son punteras en la industria. Se justifica el hecho de tener una gran capacidad de

[1] `http://www.stroustrup.com/bs_faq.html`

cómputo para poder añadir más y más carga computacional. Se habla de las ventajas en la velocidad de producción, o en la sencillez de manejo del alto nivel en comparación del bajo nivel. Y es cierto que en muchos casos se gana eficiencia o productividad, ¿pero y lo que estamos perdiendo a cambio?

Podemos estar reduciendo la velocidad de nuestro programa cientos de veces, y aún así muchas veces no importa, porque la diferencia entre que algo tarde en ejecutarse 0,001s a que tarde 0,1s se nota, pero tampoco importa tanto. Sin embargo, pensamientos como este son peligrosos, y son los que han llevado a que programas aparentemente sencillos y ligeros incluyan tiempos de espera al iniciarlos, tal y como argumenta Mike Acton [2].

Y lo que es más, cabe preguntarse, ¿de verdad tanta abstracción simplifica el problema?

La realidad es que hay software donde la gran cantidad de capas que lo forman no solo reducen su tiempo de ejecución, si no que aumenta su complejidad de forma innecesaria, hecho que al que se llama popularmente *overengineering*.

1.1. La demoscene

Void

[2] `https://www.youtube.com/watch?v=rX0ItVEVjHc&t=4620s`

2. 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.

2.1. ¡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.

2.2. 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.

2.3. 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:

2.4. Esto es una sección

Y este el contenido de la sección.

2.4.1. Esto es una subsección

Y este el contenido de la subsección.

2.4.1.1. Esto es una subsubsección

Y este el contenido de la subsubsección.

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

2.5. 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: Shaw y Garlan (1996), que también puedes mostrar con paréntesis así: (Shaw y Garlan, 1996). También se puede realizar una cita indicando a qué parte te refieres (ver Shaw y Garlan, 1996, Cap. 2) o (Shaw y Garlan, 1996, Cap. 2) o (ver Shaw y Garlan, 1996).

También puedes mostrar todos los autores cuando hay más de 2 autores añadiendo un asterisco después del comando como: Akyildiz, Pompili, y Melodia (2005), sin el asterisco quedaría así: Akyildiz y cols. (2005).

O puedes citar dos o más fuentes al mismo tiempo: (Barkan y cols., 1995; Leighton, 2012)

2.6. 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.

2.7. 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

2.8. 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.

2.9. 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:

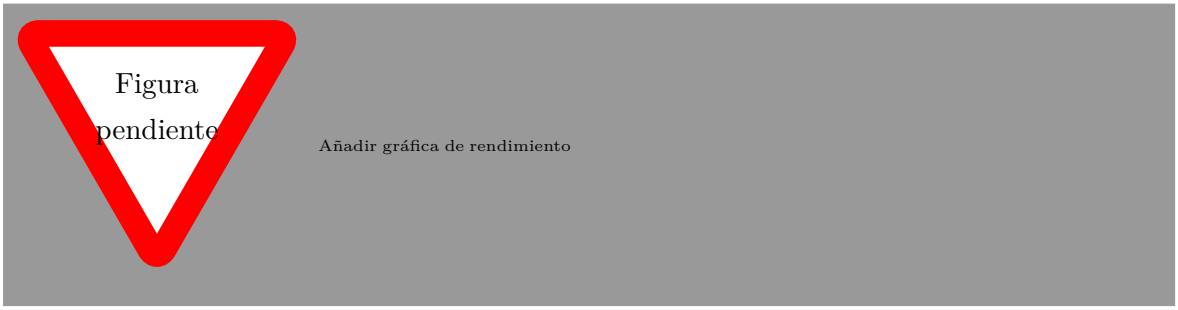
Recomiendo utilizar programas LaTeX que permitan trabajar con sistema de archivos para poder editar el conjunto de capítulos en la misma ventana. Este tipo de función lo tienen programas como TexStudio, es multiplataforma.

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:



3. Marco Teórico (Con ejemplos de listas)

3.1. Listas

Hacer una lista es simple en LaTeX. Para ello has de crear un entorno (así se llama) itemize con

```
\begin{itemize}
...
\end{itemize}
```

Y dentro de esa estructura, añadir cada elemento de la lista precedido de

```
\item primer ítem de lista
\item segundo ítem de lista
...
\item ultimo ítem de lista
```

Es importante que revises este texto tal como aparece en la plantilla y relaciones el aspecto que tiene el PDF final con cómo está escrito el documento LaTeX.

Aquí va una lista con subtérminos:

```
    \begin{itemize}
    \item Ingeniería Informática.
    \item Ingeniería Sonido e Imagen en Telecomunicación.
    \item Ingeniería Multimedia.
        \subitem Mención: Creación y ocio digital.
        \subitem Mención: Gestión de Contenidos.
    \end{itemize}
```

El resultado es el siguiente:

- Ingeniería Informática.
- Ingeniería Sonido e Imagen en Telecomunicación.
- Ingeniería Multimedia.

    Mención: Creación y ocio digital.

    Mención: Gestión de Contenidos.

Aquí va una lista con subtérminos pero numerada:

```
\begin{enumerate}
    \item Ingeniería Informática.
    \item Ingeniería Sonido e Imagen en Telecomunicación.
```

```
    \item Ingeniería Multimedia.
    \begin{enumerate}
        \item Mención: Creación y ocio digital.
        \item Mención: Gestión de Contenidos.
    \end{enumerate}
\end{enumerate}
```

El resultado es el siguiente:

1. Ingeniería Informática.
2. Ingeniería Sonido e Imagen en Telecomunicación.
3. Ingeniería Multimedia.
    a) Mención: Creación y ocio digital.
    b) Mención: Gestión de Contenidos.

3.2. Listas de definición

Puedes realizar una lista de conceptos con su definición del siguiente modo:

```
\begin{description} % Inicio de la lista
    \item[MAPP XT:] Programa desarrollado por \textit{Meyer Sound} para el diseño y ajuste de sistemas ↩
        ↪ formados por altavoces de su marca.
    \begin{description} % Realiza una lista dentro de la lista
        \item[Ventajas:]~
        El programa permite realizar múltiples ajustes tal como se podría realizar en la realidad con un ↩
            ↪ procesador real.

        Permite analizar la fase recibida en cualquier punto y compararla con otras mediciones.

        Dispone de varios tipos de filtros, inversiones de fase, etc.
        \item[Inconvenientes:]~
        No existe una lista global de los altavoces ubicados en el plano, por lo tanto solo se pueden editar ↩
            ↪ seleccionándolos sobre el plano.

        Sólo permite diseñar en 2 dimensiones, principalmente sobre la vista lateral ya que los array de ↩
            ↪ altavoces no permite voltearlos.
    \end{description}
\end{description}
```

Y LaTeX genera lo siguiente:

MAPP XT: Programa desarrollado por *Meyer Sound* para el diseño y ajuste de sistemas formados por altavoces de su marca.

Ventajas: El programa permite realizar múltiples ajustes tal como se podría realizar en la realidad con un procesador real.

Permite analizar la fase recibida en cualquier punto y compararla con otras mediciones.

Dispone de varios tipos de filtros, inversiones de fase, etc.

Inconvenientes: No existe una lista global de los altavoces ubicados en el plano, por lo tanto solo se pueden editar seleccionándolos sobre el plano.

Sólo permite diseñar en 2 dimensiones, principalmente sobre la vista lateral ya que los array de altavoces no permite voltearlos.

4. Objetivos (Con ejemplos de tablas)

4.1. Tablas

Ahora veremos otra estructura más: las tablas.
Aquí va una tabla[1] para que se vea cómo insertar una tabla simple dentro del documento.

```
\begin{table}[h]
    \centering
    \begin{tabular}{lllll}
        &columna A&columna B&columna C\\
        \hline
        fila 1&fila 1, columna A & fila 1, columna B & fila 1, columna C\\
        fila 2&fila 2, columna A & fila 2, columna B & fila 2, columna C\\
        fila 3&fila 3, columna A & fila 3, columna B & fila 3, columna C\\ \hline
    \end{tabular}
    \caption{Ejemplo de tabla.}
    \label{tabladeejemplo}
\end{table}
```

	columna A	columna B	columna C
fila 1	fila 1, columna A	fila 1, columna B	fila 1, columna C
fila 2	fila 2, columna A	fila 2, columna B	fila 2, columna C
fila 3	fila 3, columna A	fila 3, columna B	fila 3, columna C

Tabla 4.1: Ejemplo de tabla.

LaTeX usa un sistema de parámetros para "decorar" las tablas. Puedes consultar estos parámetros en la tabla 4.2 de la página 14. La tabla se ubicará donde, a juicio de LaTeX, menos moleste por lo que puede no aparecer necesariamente donde se ha insertado en el texto original.

Existe la posibilidad de forzar que las tablas, figuras u otros objetos aparezcan en la zona del texto que se desea aunque en ocasiones puede dejar grandes espacios en blanco. El comando a utilizar es:

```
\FloatBarrier
```

Que introducido justo después de una tabla, figura, etc (despues del comando \end{...}) fuerza la aparición en el texto, empujando el contenido.

[1] En http://www.tablesgenerator.com/ se puede encontrar un generador On-Line de tablas para LaTeX

Parámetro	Significado
`h`	Situa el elemento flotante *preferentemente* (es decir, si es posible) en la situación exacta donde se incluye este
`t`	Sitúa el elemento en la parte de arriba de la página
`b`	Sitúa el elemento en la parte de abajo de la página
`p`	Sitúa el elemento en una página aparte dedicada sólo a elementos flotantes; en el caso del formato `article`, ésta se sitúa al final del documento, mientras que para al book es colocada al final de cada capítulo

Tabla 4.2: Parámetros optativos de los entornos flotantes

También es posible elegir el ancho de cada columna y la orientación del texto en cada una. Por ejemplo:

```
\begin{table}[ht]
    \centering
    \begin{tabular}{|C{2cm}|C{2cm}|C{2cm}|C{2cm}|} % 4 columnas de 2cm - texto centrado y con bordes
        \hline
        \multicolumn{4}{|c|}{\textbf{\begin{tabular}[c]{@{}c@{}}FUENTE: TRÁFICO RODADO\\ ↩
            ↪ HORARIO: TARDE\end{tabular}}} \\ \hline
        \textbf{dB(A)} & \textbf{Población expuesta tarde} & \textbf{\%} & \textbf{\scriptsize{↩
            ↪ CENTENAS}} \\ \hline
        \textbf{\textgreater70} & 0 & 0,000 & 0 \\ \hline
        \textbf{65 - 70} & 348,9 & 9,792 & 3 \\ \hline
        \textbf{60 - 65} & 1594,7 & 44,757 & 16 \\ \hline
        \textbf{55 - 60} & 322,1 & 9,040 & 3 \\ \hline
        \textbf{50 - 55} & 0 & 0,000 & 0 \\ \hline
        \textbf{\textgreater50} & 1297,3 & 36,410 & 13 \\ \hline
        \textbf{TOTAL} & 3563 & 100 & 35 \\ \hline
    \end{tabular}
    \label{my-label}
\end{table}
```

LaTeX genera esto:

FUENTE: TRÁFICO RODADO HORARIO: TARDE			
dB(A)	**Población expuesta tarde**	**%**	**CENTENAS**
>70	0	0,000	0
65 - 70	348,9	9,792	3
60 - 65	1594,7	44,757	16
55 - 60	322,1	9,040	3
50 - 55	0	0,000	0
>50	1297,3	36,410	13
TOTAL	3563	100	35

Donde C{2cm} indica que la columna tiene el texto centrado y un ancho de 2 cm. Tambien

se puede utilizar L{} o R{} para poner el texto a la izquierda o derecha y definir un ancho concreto.

Páginas como `https://www.tablesgenerator.com/` ayudan a realizar tablas fácilmente, es lo más recomendado, ahorra mucho tiempo de trabajo y luego si falta algún detalle se puede retocar en el documento.

El formato estándar de las columnas es c, l o r, así lo genera la web mencionada antes, pero una vez generada puedes cambiar ese formato por el definido anteriormente para ajustar el ancho de las columnas, o mantenerlo así si el resultado ya es el deseado.

Para conocer más sobre las tablas puedes leer manuales como este: `https://latexlive.files.wordpress.com/2009/04/tablas.pdf` que contiene muchos ejemplos y explicaciones.

4.2. Otros diseños de tablas

Modelo	15LEX1600Nd	15P1000Fe V2
fs (Hz)	41	45
Re (ohm)	5.5	5.2
Le (μH)	1600	1500
Bl (N/A)	25.7	27.4
M_{MS} (g)	175	157
C_{MS} ($\mu m/N$)	84	78
R_{MS} (kg/s)	6.8	7.6
d (cm)	33.5	33
Vas (dm^3)	91	80.7
Q_{TS}	0.36	0.30
Q_{MS}	6.6	5.9
Q_{ES}	0.38	0.31
Sens (dB @ 2.83V/1m)	96	98
η	1.7%	2.4%
Sd (cm^2)	880	855

Tabla 4.3: Parámetros de los altavoces elegidos de la marca Beyma®.

		140PU				50PU			
		Phase II		Phase I		Phase II		Phase I	
# BJet		≥ 4	2 or 3	≥ 4	2 or 3	≥ 4	2 or 3	≥ 4	2 or 3
# Bkg		123	76	12	7	84	35	7	3
Asimov	NM1	13	6	9	3	15	9	11	4
	NM2	6	2	4	1	7	3	5	1
	NM3	3	1	2	0	4	1	2	0
	STC	6	3	4	1	7	5	5	2

Tabla 4.4: Ejemplo 2

5. Metodología (Con ejemplos de figuras)

5.1. Inserción de figuras

Las figuras son un caso un poco especial ya que LaTeX busca el mejor lugar para ponerlas, no siendo necesariamente el lugar donde está la referencia. Por ello es importante añadirle un "caption" y un "label" para poder hacer referencia a ellas en el párrafo correspondiente. Nosotros ponemos la referencia a la figura 5.1 que está en la página 17, justo aquí debajo, pero LaTeX puede que la ubique en otro lugar. (observa el código LaTeX de este párrafo para observar como se realizan las referencias. Estos detalles también se aplican a tablas y otros objetos).

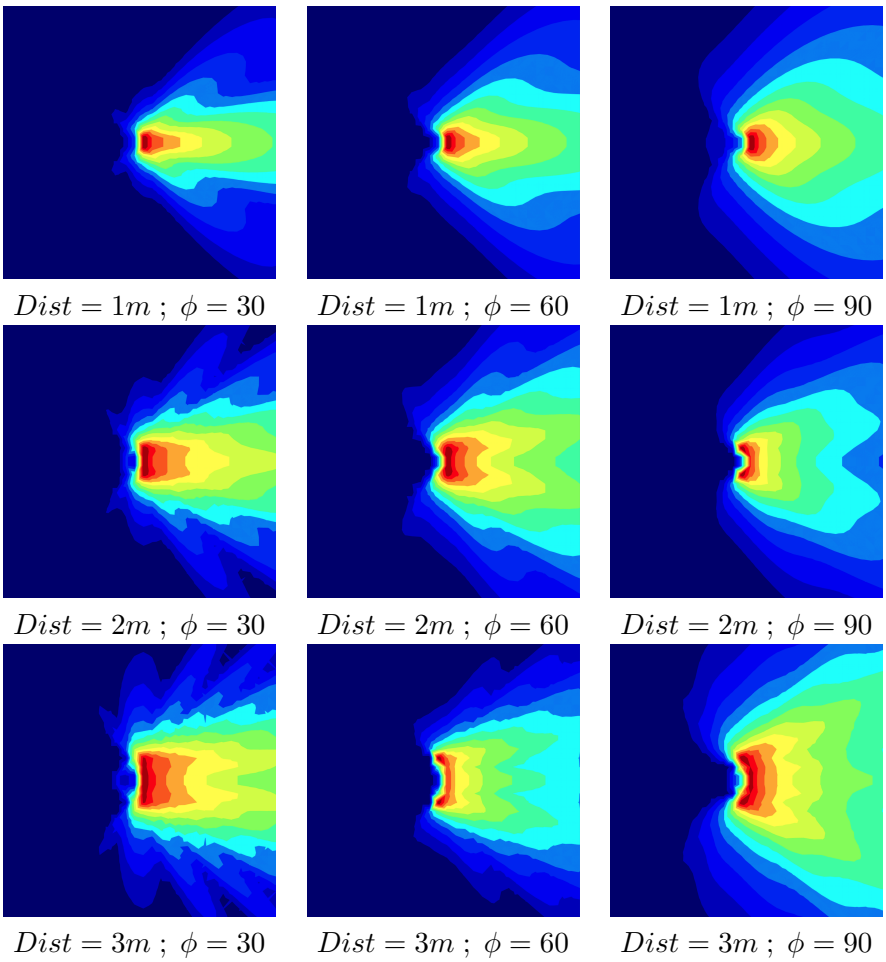
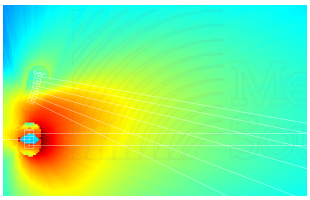


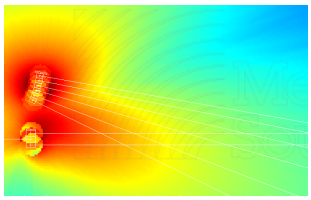
Tabla 5.1: Esta es una tabla con múltiples imágenes. Útil cuando se deben mostrar varias juntas.

Existe también la posibilidad de realizarlo sin tablas, con subfiguras:

```
\begin{figure}[h]
    \centering
    \begin{subfigure}[b]{0.4\textwidth} % Espacio horizontal ocupado por la subfigura
        \centering
        \includegraphics[width=4cm]{archivos/subs-sin} % Tamaño de la imagen
        \caption{Sin procesado.}
        \label{fig:gull}
    \end{subfigure}
    ~ % Añadir el espacio deseado, si se deja la linea en blanco la siguiente subfigura ira en una nueva linea
    \begin{subfigure}[b]{0.4\textwidth} % Espacio horizontal ocupado por la subfigura
        \centering
        \includegraphics[width=4cm]{archivos/subs-con} % Tamaño de la imagen
        \caption{Con procesado.}
        \label{fig:tiger}
    \end{subfigure}
    \caption{Ejemplo de subfiguras}\label{sistemass}
\end{figure}
```

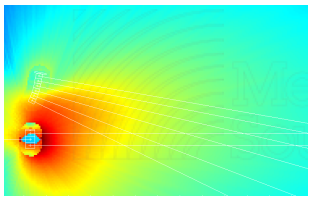


(a) Sin procesado.

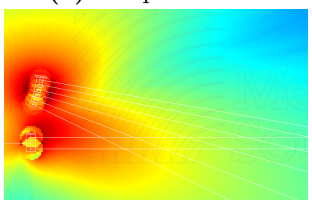


(b) Con procesado.

Figura 5.1: Ejemplo de subfiguras



(a) Sin procesado.



(b) Con procesado.

Figura 5.2: Ejemplo de subfiguras vertical

Si eliminas la línea '\caption' de las subfiguras, tendrás las imágenes sin la información individual, aunque sí con la principal. Y obviamente, si eliminas el de la figura no se mostrará ninguna información.

6. Desarrollo (Con ejemplos de código)

6.1. Inserción de código

A veces tendrás que insertar algún pedazo de código fuente para explicar algo relacionado con él. No sustituyas explicaciones con códigos enormes. Si pones algo de código en tu TFG que sea para demostrar algo o explicar alguna solución.

LaTeX te ayuda a escribir código de manera que su presentación tenga las marcas y tabulaciones propias de este tipo de texto. Para ello, debes poner el código que escribas DENTRO de un entorno que se llama "listings". La plantilla ya tiene una serie de instrucciones para incluir el paquete "listings" y añadirle algunos modificadores por lo que no tienes que incluirlo tú. Simplemente, mete tu código en el entorno "lstlisting" y ya está. Puedes indicar el lenguaje en el que está escrito el código y así LaTeX lo mostrará mejor.

En el archivo *estiloscodigoprogramacion.tex* están definidos algunos lenguajes para mostrarlos con un diseño concreto, se pueden modificar para cambiar el coloreado del código, qué términos se ponen en negrita, etc. Si se quiere profundizar más en la función "listings" se puede consultar su manual en `http://osl.ugr.es/CTAN/macros/latex/contrib/listings/listings.pdf`, aunque hay mucha información en foros y blog's que es más fácil de comprender.

Veamos un ejemplo en la figura 6.1:

```
\begin{lstlisting}[style=C, caption={ejemplo código C},label=C_code]
    #include <stdio.h>
    int main(int argc, char* argv[]) {
    puts("Hola mundo!");
    }
\ end{lstlisting}
```

El resultado será:

Código 6.1: ejemplo código C

```
1 #include <stdio.h>
2 // Comentario
3 int main(int argc, char* argv[]) {
4   puts("Hola mundo!");
5 }
```

Si lo quieres en color, está definido el estilo C-color en el archivo *estiloscodigoprogramacion.tex*, con algunos parámetros para mejorar la visualización:

```
\begin{lstlisting}[style=C-color, caption={ejemplo código C en color},label=C_code-color]
#include <stdio.h>
// Comentario
int main(int argc, char* argv[]) {
```

```
  puts(”Hola mundo!”);
}
\ end{lstlisting}
```

Código 6.2: ejemplo código C en color

```
1   #include <stdio.h>
2   // Comentario
3   int main(int argc, char* argv[]) {
4   puts(”Hola mundo!”);
5   }
```

Por supuesto, puedes mejorar esta presentación utilizando más modificadores. En la sección 6.2 se indican algunos detalles.

Otro ejemplo, ahora para mostrar código PHP, sería escribir en tu fichero LaTeX lo siguiente:

```
 \begin{lstlisting}[style=PHP, caption={ejemplo código PHP},label=PHP_code]
 /*
Ejemplo de código en PHP para escribir tu primer programa en este lenguaje
Copia este código en tu ordenador y ejecútalo
*/
<html>
  <head>
    <title>Prueba de PHP</title>
  </head>
  <body>
    <?php echo '<p>Hola Mundo</p>'; ?> //esto lo escribe TODO el mundo
  </body>
</html>
 \ end{lstlisting}
```

y el resultado es el siguiente:

Código 6.3: ejemplo código PHP

```
100 /*
101 Ejemplo de código en PHP para escribir tu primer programa en este lenguaje. Copia este código en tu ↩
        ↪ ordenador y ejecútalo
102 */
103  <html>
104   <head>
105     <title>Prueba de PHP</title>
106   </head>
107   <body>
108     <?php echo '<p>Hola Mundo</p>'; ?> //esto lo escribe TODO el mundo
109   </body>
110 </html>
```

O también en color:

Código 6.4: ejemplo código PHP

```
1 /*
2 Ejemplo de código en PHP para escribir tu primer programa en este lenguaje. Copia este código en tu ↩
        ↪ ordenador y ejecútalo
3 */
4  <html>
5   <head>
6     <title>Prueba de PHP</title>
```

```
  </head>
  <body>
    <?php echo '<p>Hola Mundo</p>'; ?> //esto lo escribe TODO el mundo
  </body>
</html>
```

Observa cómo LaTeX ha puesto los comentarios en gris y ajustado el código para que se muestre más claro.

A continuación se muestran otros ejemplos:

Código 6.5: ejemplo código Matlab en color

```
%% Code sections are highlighted.
% System command are supported...
!touch testFile.txt
A = [1, 2, 3;... %... as is line continuation.
     4, 5, 6];
fid = fopen('testFile.text', 'w');
for k=1:10
  fprintf(fid, '%6.2f \n', k)
end
x=1; %% this is just a comment, not the start of a section
% Context-sensitive keywords get highlighted correctly...
p = properties(person); %(here, properties is a function)
x = linspace(0,1,101);
y = x(end:-1:1);
% ... even in nonsensical code.
]end()()(((end while { end )end ))))end (end
%{
    block comments are supported
%} even
runaway block comments are
```

Código 6.6: ejemplo código Matlab en blanco y negro

```
%% Code sections are highlighted.
% System command are supported...
!touch testFile.txt
A = [1, 2, 3;... %... as is line continuation.
     4, 5, 6];
fid = fopen('testFile.text', 'w');
for k=1:10
  fprintf(fid, '%6.2f \n', k)
end
x=1; %% this is just a comment, not the start of a section
% Context-sensitive keywords get highlighted correctly...
p = properties(person); %(here, properties is a function)
x = linspace(0,1,101);
y = x(end:-1:1);
% ... even in nonsensical code.
]end()()(((end while { end )end ))))end (end
%{
    block comments are supported
%} even
runaway block comments are
```

Código 6.7: ejemplo código Python en color

```
class Example (object):
    def ___init___ (self, account, password):
        """e.g. account = 'bob@example.com/test'
                password = 'bigbob'
        """

        reg = telepathy.client.ManagerRegistry()
        reg.LoadManagers()

        # get the gabble Connection Manager
        self.cm = cm = reg.GetManager('gabble')

        # get the parameters required to make a Jabber connection
        # begin ex.basics.dbus.language-bindings.python.methods.call
        cm[CONNECTION_MANAGER].RequestConnection('jabber',
            {
                'account': account,
                'password': password,
            },
            reply_handler = self.request_connection_cb,
            error_handler = self.error_cb)
        # end ex.basics.dbus.language-bindings.python.methods.call
```

Código 6.8: ejemplo código Python en blanco y negro

```
class Example (object):
    def ___init___ (self, account, password):
        """e.g. account = 'bob@example.com/test'
                password = 'bigbob'
        """

        reg = telepathy.client.ManagerRegistry()
        reg.LoadManagers()

        # get the gabble Connection Manager
        self.cm = cm = reg.GetManager('gabble')

        # get the parameters required to make a Jabber connection
        # begin ex.basics.dbus.language-bindings.python.methods.call
        cm[CONNECTION_MANAGER].RequestConnection('jabber',
            {
                'account': account,
                'password': password,
            },
            reply_handler = self.request_connection_cb,
            error_handler = self.error_cb)
        # end ex.basics.dbus.language-bindings.python.methods.call
```

6.2. Usos y personalización

El texto que acompaña al código puedes incluirlo o no, también puedes decidir si el texto va numerado o no. A continuación se muestra como:

```
% Con esta línea el código no tendrá título
\begin{lstlisting}[style=Python]
    micodigo
\ end{lstlisting}
```

```
1    micodigo
```

```
    % Con esta línea el código tendrá el título abajo
    \begin{lstlisting}[style=Python, caption={Ejemplo de título abajo},captionpos=b]
        micodigo
    \ end{lstlisting}
```

```
1    micodigo
```

Código 6.9: Ejemplo de título abajo

```
    % Con esta línea el código tendrá título no numerado
    \begin{lstlisting}[style=Python, title={Ejemplo de título no numerado}]
        micodigo
    \ end{lstlisting}
```

Ejemplo de título no numerado

```
1    micodigo
```

```
    % Con esta línea el código no tendrá las líneas numeradas
\begin{lstlisting}[style=Python,numbers=none, title={Ejemplo de código sin número de líneas}]
    micodigo
    sin
    número
    de
    líneas
\ end{lstlisting}
```

Ejemplo de código sin número de líneas

```
micodigo
sin
número
de
líneas
```

6.3. Importar archivos fuente

Existe la posibilidad de importar un archivo de código en lugar de copiar su contenido y pegarlo en LaTeX.

Para realizarlo debes escribir:

```
\lstinputlisting[style=C++-color,caption={Archivo C++ importado}]{archivos/ejemplos/holamundo.cpp}
```

Y se importará con el formato establecido entre los '[ ]':

Código 6.10: Archivo C++ importado

```cpp
#include <stdio.h>
int main()
{
   // printf() displays the string inside quotation
   printf("Hello, World!");
   return 0;
}
```

A continuación se muestran otros ejemplos

```
\lstinputlisting[style=Python-color,caption={Archivo Py importado},label=importado_py]{archivos/ejemplos/holamundo.py}
```

Código 6.11: Archivo Py importado

```python
#------------------------------------ ↩
    ↪ ------------------------------------
# helloworld.py
#------------------------------------ ↩
    ↪ ------------------------------------

import stdio

# Write 'Hello, World' to standard output.
stdio.writeln('Hello, World')

#------------------------------------ ↩
    ↪ ------------------------------------

# python helloworld.py
# Hello, World
```

```
\lstinputlisting[style=Matlab-color,caption={Archivo Matlab importado},label=importado_m]{archivos/ejemplos/holamundo.m}
```

Código 6.12: Archivo Matlab importado

```matlab
function y = hello_world %#codegen

y = 'Hello World!';

end
% Copyright 2010 The MathWorks, Inc.
```

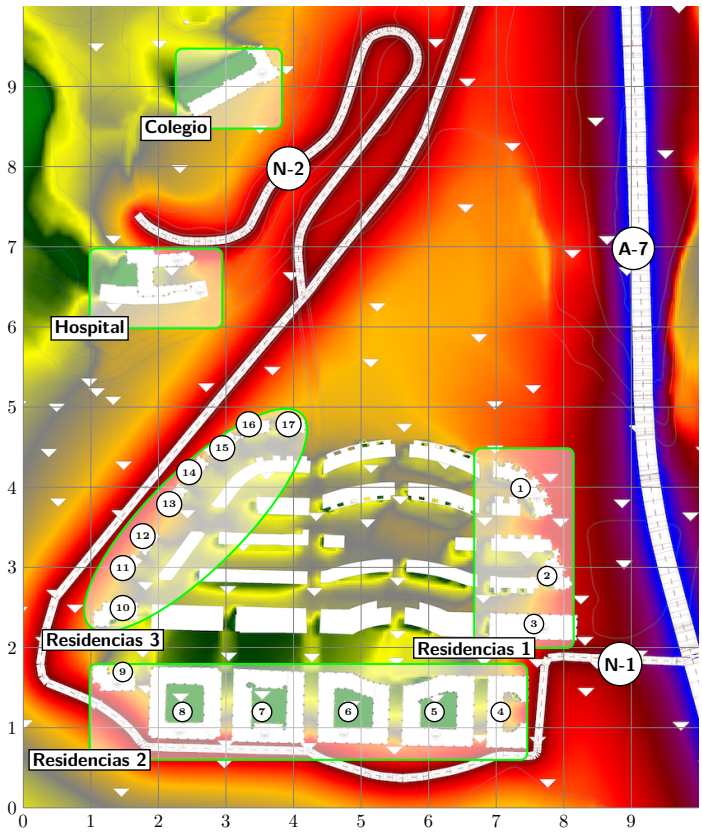
7. Resultados (Con ejemplos de gráficos)

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/`.

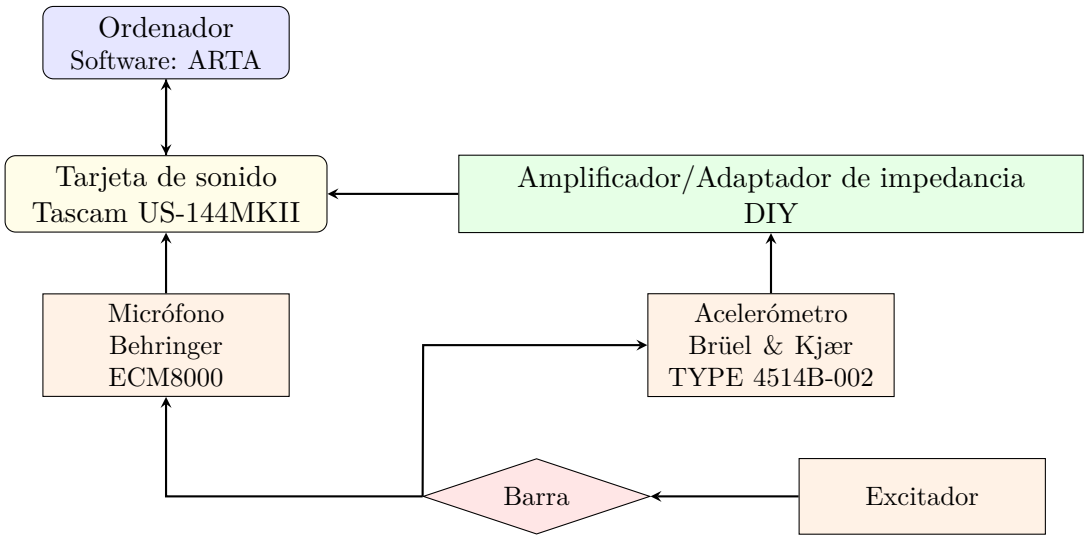


Figura 7.1: Diagrama realizado en latex con Tikz.

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:

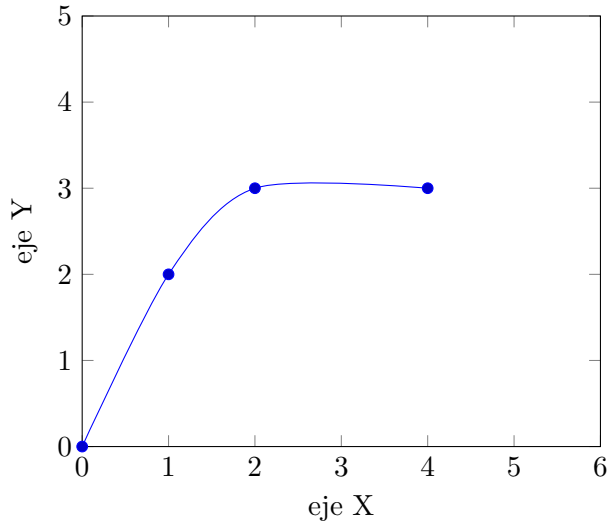


Figura 7.2: Gráfica sencilla.

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):

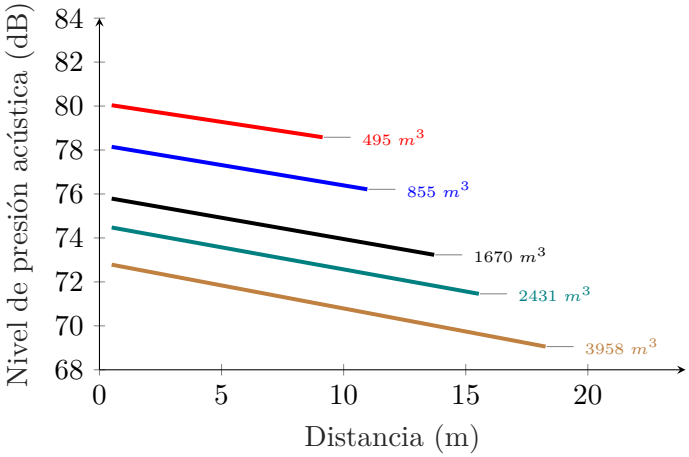


Figura 7.3: OP/S003

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:

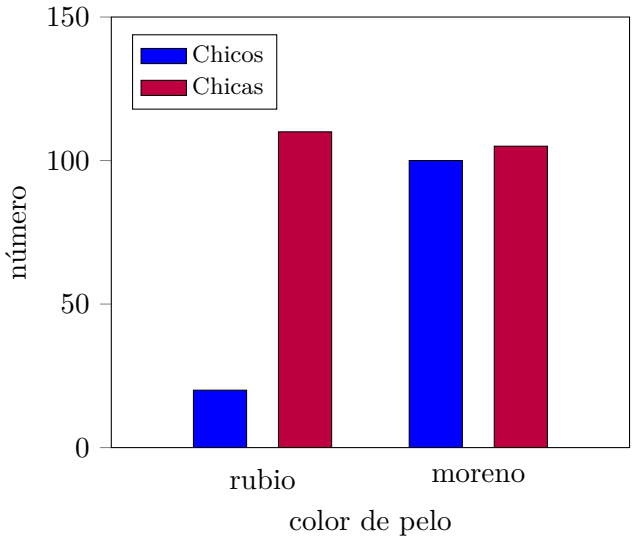


Figura 7.4: Gráfica barras.

7.2.3. Polar

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

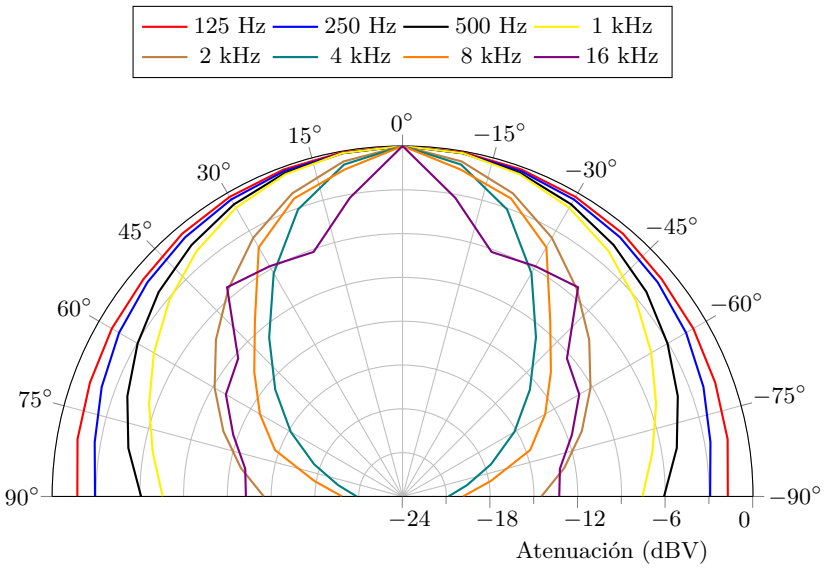


Figura 7.5: Directividad normalizada del altavoz (0 dBV en el eje).

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}
```

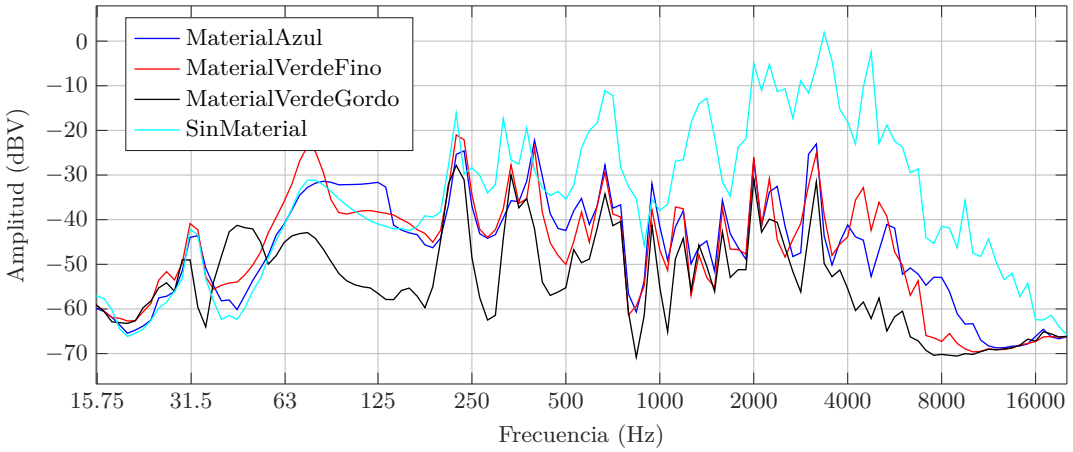


Figura 7.6: Ejemplo de gráfica obtenida con matlab2tikz.

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

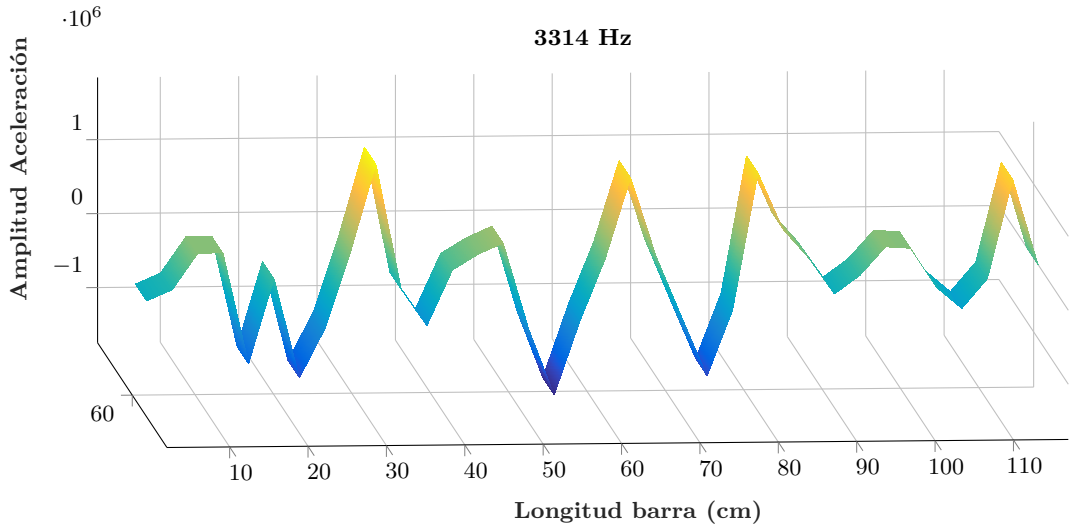


Figura 7.7: Amplitud de la aceleración en el modo número 8.

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:

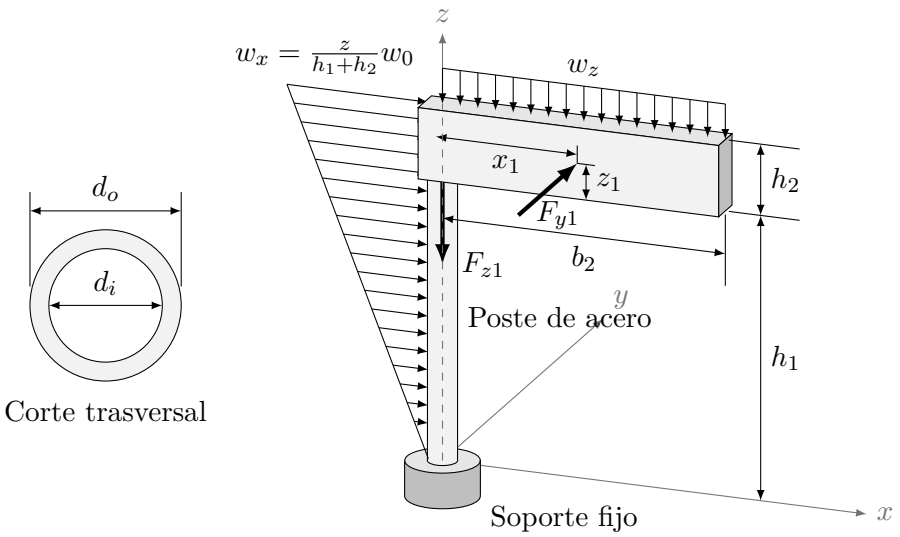


Figura 7.8: Señal realizada con Tikz, sin imágenes.

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.

Bibliografía

AENOR. (1997). *norma une 50136:1997.* Descargado de `http://docubib.uc3m.es/CURSOS/Documentos_cientificos/Normas%20y%20directrices/UNE_50136=ISO%207144.pdf`

Akyildiz, I. F., Pompili, D., y Melodia, T. (2005). Underwater acoustic sensor networks: research challenges. *Ad hoc networks*, *3*(3), 257–279.

Alexander, G. E., y Crutcher, M. D. (1990). Functional architecture of basal ganglia circuits: neural substrates of parallel processing. *Trends in neurosciences*, *13*(7), 266–271.

Anderson, J. R. (2013). *The architecture of cognition.* Psychology Press.

Auld, B. A. (1973). *Acoustic fields and waves in solids.* .

Baddeley, A. D. (1966). Short-term memory for word sequences as a function of acoustic, semantic and formal similarity. *The Quarterly Journal of Experimental Psychology*, *18*(4), 362–365.

Barkan, A., Merlino, R. L., y D'angelo, N. (1995). Laboratory observation of the dust-acoustic wave mode. *Physics of Plasmas*, *2*(10), 3563–3565.

Bass, L. (2007). *Software architecture in practice.* Pearson Education India.

Biot, M. A. (1962). Mechanics of deformation and acoustic propagation in porous media. *Journal of applied physics*, *33*(4), 1482–1498.

Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., y Weiss, W. (1998). *An architecture for differentiated services* (Inf. Téc.).

BOE. (2012, marzo). *Resolución de 7 de marzo de 2012, de la universidad de alicante, por la que se publica el plan de estudios de graduado en ingeniería multimedia.* BOE, 22 marzo de 2012. Descargado de `http://www.boe.es/boe/dias/2012/03/22/pdfs/BOE-A-2012-4008.pdf`

Boll, S. (1979). Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on acoustics, speech, and signal processing*, *27*(2), 113–120.

Burdic, W. S. (1991). *Underwater acoustic system analysis.* Prentice Hall.

Buschmann, F., Henney, K., y Schimdt, D. (2007). *Pattern-oriented software architecture: on patterns and pattern language* (Vol. 5). John wiley & sons.

Cerjan, C., Kosloff, D., Kosloff, R., y Reshef, M. (1985). A nonreflecting boundary condition for discrete acoustic and elastic wave equations. *Geophysics*, *50*(4), 705–708.

Colton, D., y Kress, R. (2012). *Inverse acoustic and electromagnetic scattering theory* (Vol. 93). Springer Science & Business Media.

Conrad, R. (1964). Acoustic confusions in immediate memory. *British journal of Psychology*, *55*(1), 75–84.

Cox, T. J., D'antonio, P., y Schroeder, M. (2005). Acoustic absorbers and diffusers, theory, design and application. *The Journal of the Acoustical Society of America*, *117*(3), 988–988.

Eisenstein, D. J., Zehavi, I., Hogg, D. W., Scoccimarro, R., Blanton, M. R., Nichol, R. C., … others (2005). Detection of the baryon acoustic peak in the large-scale correlation function of sdss luminous red galaxies. *The Astrophysical Journal*, *633*(2), 560.

Fahlman, S. E., y Lebiere, C. (1990). The cascade-correlation learning architecture. En *Advances in neural information processing systems* (pp. 524–532).

Fant, G. (1971). *Acoustic theory of speech production: with calculations based on x-ray studies of russian articulations* (Vol. 2). Walter de Gruyter.

Fodor, J. A., y Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition*, *28*(1), 3–71.

Foschini, G. J. (1996). Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas. *Bell labs technical journal*, *1*(2), 41–59.

Fowler, M. (2002). *Patterns of enterprise application architecture*. Addison-Wesley Longman Publishing Co., Inc.

Griffin, D. R. (1958). Listening in the dark: the acoustic orientation of bats and men.

Heinz, M., Carsten, y Hoffmann, J. (2014, March). *The listings package, march 2014.* http://texdoc.net/texmf-dist/doc/latex/listings/listings.pdf. Descargado 12/12/2014, de `http://texdoc.net/texmf-dist/doc/latex/listings/listings.pdf`

Heinzelman, W. B., Chandrakasan, A. P., y Balakrishnan, H. (2002). An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on wireless communications*, *1*(4), 660–670.

Hennessy, J. L., y Patterson, D. A. (2011). *Computer architecture: a quantitative approach*. Elsevier.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., … others (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, *29*(6), 82–97.

Hubel, D. H., y Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, *160*(1), 106–154.

Hubel, D. H., y Wiesel, T. N. (1968). Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, *195*(1), 215–243.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., … Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. En *Proceedings of the 22nd acm international conference on multimedia* (pp. 675–678).

Kemp, D. T. (1978). Stimulated acoustic emissions from within the human auditory system. *The Journal of the Acoustical Society of America*, *64*(5), 1386–1391.

Kushwaha, M. S., Halevi, P., Dobrzynski, L., y Djafari-Rouhani, B. (1993). Acoustic band structure of periodic elastic composites. *Physical review letters*, *71*(13), 2022.

Leighton, T. (2012). *The acoustic bubble*. Academic press.

McCarthy, B. (2012). *Sound systems: design and optimization: modern techniques and tools for sound system design and alignment*. CRC Press.

Mitola, J. (2000). Cognitive radio—an integrated agent architecture for software defined radio.

Muthupillai, R., Lomas, D., Rossman, P., Greenleaf, J. F., Manduca, A., y Ehman, R. L. (1995). Magnetic resonance elastography by direct visualization of propagating acoustic strain waves. *science*, *269*(5232), 1854–1857.

Rao, N., Shukla, P., y Yu, M. Y. (1990). Dust-acoustic waves in dusty plasmas. *Planetary and space science*, *38*(4), 543–546.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., y Riedl, J. (1994). Grouplens: an open architecture for collaborative filtering of netnews. En *Proceedings of the 1994 acm conference on computer supported cooperative work* (pp. 175–186).

Rosenbaum, J. (1988). *Bulk acoustic wave theory and devices*. Artech House on Demand.

Shaw, M., y Garlan, D. (1996). *Software architecture: perspectives on an emerging discipline* (Vol. 1). Prentice Hall Englewood Cliffs.

Simon, H. A. (1991). The architecture of complexity. En *Facets of systems science* (pp. 457–476). Springer.

Ulrich, K. (1995). The role of product architecture in the manufacturing firm. *Research policy*, *24*(3), 419–440.

Villa, D. (2008, 4). *Latex: Listados de código cómodos y resultones con listings.* http://crysol.org/es/node/909. Descargado 12/12/2014, de `http://crysol.org/es/node/909`

Wolniansky, P. W., Foschini, G. J., Golden, G., y Valenzuela, R. A. (1998). V-blast: An architecture for realizing very high data rates over the rich-scattering wireless channel. En *Signals, systems, and electronics, 1998. issse 98. 1998 ursi international symposium on* (pp. 295–300).

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