



## **GRADO EN INGENIERÍA DE ROBÓTICA SOFTWARE**

Escuela Técnica Superior de Ingeniería de Telecomunicación

Curso académico 2021-2022

### **Trabajo fin de grado**

Sistema de Monitorización de animales de laboratorio  
bajo plataforma de bajo coste

**Tutor:** Julio Vega Pérez

**Autor:** Isabel Cebollada Gracia



Este trabajo se distribuye bajo los términos de la licencia internacional CC BY-NC-SA International License (Creative Commons AttributionNonCommercial-ShareAlike 4.0). Usted es libre de *(a) compartir*: copiar y redistribuir el material en cualquier medio o formato; y *(b) adaptar*: remezclar, transformar y crear a partir del material. El licenciador no puede revocar estas libertades mientras cumpla con los términos de la licencia:

- *Atribución.* Usted debe dar crédito de manera adecuada, brindar un enlace a la licencia, e indicar si se han realizado cambios. Puede hacerlo en cualquier forma razonable, pero no de forma tal que sugiera que usted o su uso tienen el apoyo de la licenciatante.
- *No comercial.* Usted no puede hacer uso del material con propósitos comerciales.
- *Compartir igual.* Si remezcla, transforma o crea a partir del material, debe distribuir su contribución bajo la misma licencia del original.

# Agradecimientos

---

Unas bonitas palabras...

Quizás un segundo párrafo esté bien. No te olvides de nadie.

Un tercero tampoco viene mal para contar alguna anécdota...

¿Alguien más? Aunque sean *actores* secundarios.

Un quinto párrafo como colofón.

*A alguien especial;  
si no, tampoco pasa nada*

Madrid, xx de xxxxxx de 20xx

*Tu nombre*

# Resumen

---

Desde hace siglos, la experimentación con animales se ha llevado a cabo para saber qué les sucede a los humanos y al mundo que les rodea. A día de hoy, aunque de una manera distinta debido tanto al avance de la humanidad como de la tecnología, esta experimentación sigue desarrollándose en diferentes ámbitos; para el estudio de los comportamientos de animales o el uso de éstos como modelo de investigación en diferentes campos, desde el testado de productos hasta la sanidad humana.

Los estudios con animales en la medicina han tenido gran importancia en el desarrollo de vacunas modernas como la tuberculosis o la meningitis ya que los animales, más concretamente los ratones, sufren enfermedades similares a los humanos. De esta manera, los ratones se han convertido en uno de los principales animales para estudiar su comportamiento y el efecto que éste puede tener en la detección de enfermedades neurológicas, como el autismo, el parkinson o el alzheimer.

La observación de estos animales en todo momento es una necesidad para analizar y estudiar su comportamiento, así como tener la información de las condiciones del entorno a las que se encuentran, haciendo necesario que haya trabajadores analizando las condiciones y grabaciones que tienen que hacerse a los ratones diariamente. Para evitar esto, el fin de este trabajo es crear un sistema dotando con los sensores necesarios al entorno de los roedores para obtener la información de forma automatizada y traducirlo en una interfaz comprensible para cualquier usuario, así como el control por vídeo a través de algoritmos de visión artificial para detectar los movimientos de los animales bajo una plataforma económicamente accesible para todo el mundo.

Durante todo el desarrollo del trabajo han surgido problemas en conexiones, seguridad, instalación, en el uso de algunos sensores y el uso de herramientas no usadas hasta el momento. Tras el estudio y la comprensión de los errores encontrados, así como con la ayuda de internet y de las experiencias de otros usuarios, tanto con problemas similares como con posibles soluciones, todos los errores han podido solucionarse.

# Acrónimos

---

**AERO** *Autonomous Exploration Rover*

**AI** *Artificial Intelligence*

**ANN** *Artificial Neural Network*

**API** *Application Programming Interface*

**EKF** *Extended Kalman Filter*

**FOA** *Focus of Attention*

**GA** *Genetic Algorithm*

**GPIO** *General Purpose Input/Output*

**GPS** *Global Positioning System*

**HCI** *Human-Computer Interaction*

**HRI** *Human-Robot Interaction*

# Índice general

---

<b>1. Introducción</b>	<b>1</b>
1.1. La robótica . . . . .	1
1.1.1. Inteligencia Artificial . . . . .	3
1.1.2. Visión Artificial . . . . .	3
1.1.3. Machine Learning . . . . .	5
1.1.4. Deep Learning . . . . .	6
1.1.5. Sistema multisensorial . . . . .	8
1.1.6. Comparación con otros sistemas del mercado . . . . .	9
<b>2. Objetivos</b>	<b>12</b>
2.1. Descripción del problema . . . . .	12
2.2. Requisitos . . . . .	13
2.3. Metodología . . . . .	13
<b>3. Plataforma de desarrollo</b>	<b>15</b>
3.1. Infraestructura hardware . . . . .	15
3.2. Lenguaje python . . . . .	15
3.3. TensorFlow Lite . . . . .	16
3.4. OpenCV . . . . .	17
3.5. Flask . . . . .	18
3.6. Node-Red . . . . .	19
<b>4. Diseño</b>	<b>22</b>
4.1. Snippets . . . . .	22
4.2. Verbatim . . . . .	22
4.3. Ecuaciones . . . . .	23
4.4. Tablas o cuadros . . . . .	23
<b>5. Conclusiones</b>	<b>25</b>
5.1. Conclusiones . . . . .	25

5.2. Corrector ortográfico . . . . .	26
--------------------------------------	----

# Índice de figuras

---

1.1.	Robots previamente mencionados. . . . .	2
1.6.	Patrón de movimientos e imagen grabada del grupo de cerdos. . . . .	10
1.7.	Esquema del desarrollo del sistema mediante sensores. . . . .	10
3.1.	Detección de ratones en una imagen usando TensorFlow Lite. . . . .	17
3.2.	Ejemplo de detección de bordes con OpenCV. . . . .	18
3.3.	Muestra de los dos servidores del trabajo usando Flask. . . . .	19
3.4.	Flujo del proyecto en Node-Red. . . . .	20
3.5.	Muestra del uso de Node-Red en el trabajo. . . . .	21

# Listado de códigos

---

4.1. Función para buscar elementos 3D en la imagen . . . . .	22
4.2. Cómo usar un Slider . . . . .	23

# Listado de ecuaciones

---

4.1. Ejemplo de ecuación con fracciones . . . . .	23
4.2. Ejemplo de ecuación con array y letras y símbolos especiales . . . . .	23

# Índice de cuadros

---

4.1. Parámetros intrínsecos de la cámara . . . . .	24
--	----

---

# Capítulo 1

## Introducción

---

*Quizás algún fragmento de libro inspirador...*

Autor, Título

El desarrollo tecnológico ha provocado el avance en la robótica, un campo muy amplio en la actualidad que está facilitando la vida a los humanos en distintos ámbitos, entre ellos, unos de los más importantes: la visión artificial, cuyo objetivo es que los ordenadores sean capaces de percibir y entender el entorno, pudiendo así actuar por ellos mismos en función de la situación que se encuentren.

Para que la visión funcione correctamente, es necesario tener algoritmos de Deep Learning, que es la tecnología que permite a un ordenador aprender como un ser humano a base de entrenamiento. Esto tiene un gran impacto sobre el ser humano, ya que trabajos en los que se requiere la vigilancia o atención completa de una persona, podrían ser sustituidos por sistemas con visión artificial y Deep Learning.

### 1.1. La robótica

La robótica, aunque no bajo ese nombre, se lleva utilizando desde hace mucho tiempo, siempre pensándola con el concepto de "herramientas automatizadas", desde Aristóteles hasta mediados del siglo pasado. Hasta entonces esta disciplina siempre se había entendido como máquinas que realizan un trabajo repetitivo por el ser humano, facilitándole a éste la vida. Esta concepción se refleja con Henry Ford y la producción en serie.

Sin embargo, la palabra *robota* que significa "trabajo forzado", origen de la palabra robot, aparece por primera vez en 1921 en una obra de teatro de un autor checo llamado Karel Čapek. Empezando a concebir esta palabra a principios de siglo pasado, empezó a nacer el desarrollo de la idea de robótica que tenemos hoy en día, en el que la concepción de robótica ya no es "herramienta automatizadas", sino una industria

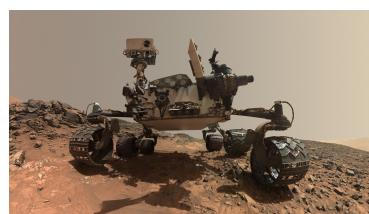
interdisciplinaria que surge de la intersección de la ciencia, la ingeniería y la tecnología, uniendo el conocimiento científico, computacional e informático, con diversas ramas de la ingeniería, ya que no solo implica el estudio de robots, sino de su diseño, programación y aplicación. Es un conjunto de múltiples disciplinas como la informática, la AI, la informática, la visión, la programación o el álgebra entre otras muchas.

Con el paso de estos últimos años la robótica ha adquirido un gran peso y un desarrollo convierte a las películas futuristas que surgían hace unos años en el presente y que, aunque queda todavía mucho camino por continuar en esta amplia disciplina, es, actualmente una realidad entre los humanos.

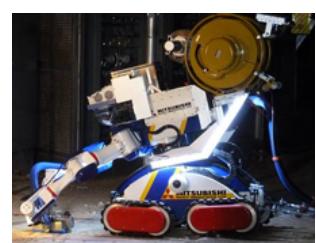
Esta disciplina tiene como fin ayudar al ser humano en todos los ámbitos, tanto en la vida cotidiana como en la vida profesional, evitando los trabajos peligrosos y aburridos, e incluso haciendo trabajos que el ser humano no sería capaz de hacer por si mismo en distintos sectores, como pueden ser la medicina, facilitando el trabajo a los científicos con el robot Da Vinci 1.1, uno de sus robots más famosos en el que recibiendo las órdenes que el cirujano comanda a través de la consola, es capaz de hacer cirugías de alta precisión, eliminando los tembleques que un humano pueda tener y aportando una mayor visión al cirujano; la robótica espacial, que permite que no sea necesaria la movilidad del humano además de portar sistemas sensoriales para las condiciones del entorno en otro planeta como el Curiosity 1.1, cuya misión era evaluar la habitabilidad en Marte; frente a la radiación, pues es un trabajo que pondría en peligro a cualquier humano exponiéndole frente a la radioactividad del entorno, con el ejemplo de los robots que se usaron en Fukushima 1.1 después que un tsunami y un terremoto provocasen el desastre, pudiendo acceder a la zona y poder limpiarla o en la industria del automóvil con los coches autónomos, donde es el propio coche capaz de realizar todas las funciones de conducción desde un punto A a un punto B sin necesidad de la intervención humana.



(a) Robot Da Vinci



(b) Robot Curiosity



(c) Uno de los robots de Fukushima

Figura 1.1: Robots previamente mencionados.

### 1.1.1. Inteligencia Artificial

La robótica es un conjunto de diferentes disciplinas que permiten abordar todos los campos que esta comprende. Uno de los campos en que a día de hoy la robótica está desarrollando y es muy importante es la AI, que consiste en replicar el funcionamiento mental a través del desarrollo de algoritmos para realizar determinadas tareas, e ir aprendiendo a medida que las realizan en base a la experiencia. Hay varios estudios en la actualidad con redes neuronales o algoritmos genéticos.

Hay dos tipos de inteligencia artificial:

- AI fuerte, que está compuesta por la AI general, que es una teoría en la que una máquina tuviera una inteligencia humana, siendo capaz de resolver cualquier problema por sí misma; y la superinteligencia artificial, que superaría a la capacidad del cerebro humano. La AI fuerte es todavía teórica y no hay ejemplos reales de su uso.
- AI débil, está entrenada para realizar tareas específicas, que operan dentro de un rango que previamente ha sido definido. Los sistemas dotados con esta inteligencia parecen inteligentes, pero están limitados y que no pueden realizar nada fuera de ese contexto, como es *Siri de Apple*.

La AI abarca diferentes campos como la comprensión, el reconocimiento y aprendizaje, *a groso modo*, que a su vez, para su correcto funcionamiento, tienen extensos campos de estudio, entre los que destaca la visión artificial.

### 1.1.2. Visión Artificial

La visión artificial es uno de los campos más importantes de la AI que permite que los sistemas obtengan la información importante de imágenes digitales, vídeos u otras entradas visuales, y tomen acciones o hagan recomendaciones basadas en dicha información. Así pues, si la AI permite que los ordenadores piensen, la visión permite su observación, análisis y comprensión. Sin visión, una máquina no es capaz de reconocer ningún objeto, animal o persona en su entorno.

La visión artificial se trata de asemejar lo máximo posible a la visión humana, con la diferencia de que esta segunda cuenta con las experiencias aprendidas para diferenciar los elementos que le rodean, su movimiento, su distancia o su tamaño. Así pues la visión artificial trata de asemejar el funcionamiento del ojo humano y su posterior procesamiento con el cerebro con cámaras, datos y sensores para poder obtener la

máxima información del entorno así como algoritmos para poder procesar la imagen adecuadamente, en el menor tiempo posible ya que es necesaria una reacción rápida. Tiene diferentes usos, entre ellos los más fundamentales son:

- *Clasificación de imágenes 1.2*, que consiste en asignar imágenes en una serie de categorías predefinidas a través de un algoritmo. Es necesario contar con una gran cantidad de datos para poder tener suficiente entrenamiento para fallar lo mínimo posible.
- *Detección de objetos 1.2*, que consiste en analizar partes de la imagen para localizar objetos mediante cuadros limitadores. Esto se consigue mediante el entrenamiento de una gran cantidad de imágenes en las que se etiquetan diferentes tipos de objetos y etiquetándolos por su nombre, obteniendo así los datos que se usarán para la detección en una imagen.
- *Segmentación de instancias 1.2*, que consiste en el enmascaramiento exacto de los píxeles que representan a los objetos en una imagen. Es un paso más en la detección de objetos, siendo capaz de separar el objeto concreto del resto de la imagen.

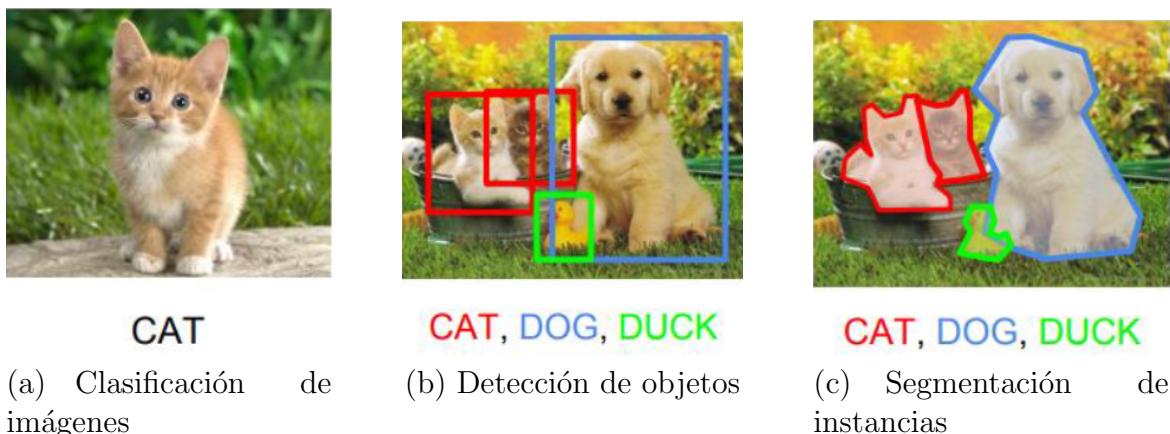


Figura 1.2: Algunos usos en la visión artificial.<sup>1</sup>

Un sistema dotado de visión funciona siguiendo tres niveles de operación:

En primer lugar, la obtención de imágenes o vídeos mediante una cámara. Estas imágenes son transferidas al sistema para poder procesarlas posteriormente. En segundo lugar, el procesamiento de las imágenes para poder representar correctamente los datos

<sup>1</sup>Imágenes obtenidas de shorturl.at/tyACR

de interés. Este proceso consiste en un trabajo automático del sistema en el que se elimina el ruido, reescalar, ajustar el contraste y otros ajustes para adaptar las imágenes.

Finalmente, la comprensión de imágenes, que es el paso clave para que el sistema pueda llamarse inteligente, donde a través de un modelo de aprendizaje, es capaz de realizar el objetivo de interés, como puede ser detectar un determinado objeto utilizando datos aprendidos en el pasado.

Un ejemplo que está muy presente y en desarrollo actualmente es el caso de los coches autónomos 3.1, que necesitan de una visión artificial impecable y sin fallos debido a que sus consecuencias pondrían en riesgo a los humanos. Estos coches utilizan distintos colores para detectar distintos elementos que puede encontrarse, como pueden ser líneas divisorias de la calzada, líneas delimitadoras de la calzada, semáforos, señales, objetos estáticos u objetos que se acercan. Así la computadora del coche procesa la información pudiendo detectar todo lo que afecta o potencialmente pueda afectar en la conducción y actúa en consecuencia.



Figura 1.3: Vision de las cámaras y sensores de un Tesla autónomo.<sup>2</sup>

Como se puede apreciar, es imprescindible disponer de una amplia cantidad de datos y de información basada en el aprendizaje pasado para que el algoritmo sea preciso y tenga el menor número de errores, para poder definirlo como robusto y fiable. Para lograr esto se pueden usar dos tecnologías; deep learning y una red neuronal convolucional o CNN.

### 1.1.3. Machine Learning

Es necesario dotar a los sistemas con una inteligencia artificial para poder conseguir que sean capaces de aprender automáticamente sin necesidad de ayuda externa. Dentro

---

<sup>2</sup>Imágenes obtenidas de shorturl.at/huBFL

de la AI se encuentra una categoría denominada *Machine Learning* o aprendizaje automático cuyo fin es que los algoritmos descubran patrones en conjuntos de datos que les hacen aprender y mejorar respecto a la experiencia anterior pudiendo así tener un aprendizaje autónomo para poder realizar una tarea sin ayuda externa.

En el sistema de aprendizaje de un algoritmo de *Machine Learning* hay tres partes principales: En primer lugar un proceso de decisión, en el que una vez recibidos los datos de entrada, que pueden no estar etiquetados o bien estarlo para así indicarle al modelo lo que debe identificar, el algoritmo genera una estimación en los datos partiendo de un patrón.

La segunda parte es una función de error que sirve para evaluar la precisión del modelo sobre la decisión tomada, y la tercera y última parte es un proceso de optimización de modelos donde las ponderaciones se ajustan en el caso de que el modelo se pueda ajustar mejor a el conjunto de datos de entrenamiento, repitiendo este proceso hasta cumplir un umbral de precisión.

Hay diferentes tipos de *Machine Learning*:

- *Aprendizaje supervisado*, en el que los conjuntos de datos para entrenar los datos que servirán como experiencia y base al algoritmo a la hora de hacer una clasificación con un nuevo dato están etiquetados, usando así un conjunto de datos entrenados iniciales a la hora de recibir un nuevo dato.
- *Aprendizaje no supervisado*, en el que en este caso los datos están sin etiquetar, es decir, no cuenta con un conjunto entrenado previamente.
- *Aprendizaje semisupervisado*, que es el término medio entre los dos anteriores. En este caso, en el entrenamiento se usa un conjunto de datos más pequeño que en el aprendizaje supervisado y usa un grupo más grande de datos sin etiquetar.

#### 1.1.4. Deep Learning

Dentro del *Machine Learning* se encuentra el *Deep Learning* o aprendizaje profundo, que se diferencia del primero en el tipo de datos con el que trabaja así como en la manera en que aprende cada algoritmo, pues el *Deep Learning* elimina algunos de los datos de preprocessado, eliminando gran parte de la intervención humana y aprendiendo autónomamente con cada nueva información que recibe y permitiendo conjuntos de datos más grandes.

El *Deep Learning* es una forma especializada de *Machine Learning*, capaz de detectar las características de un conjunto de datos para clasificarlos en distintas clases y no necesitando a un humano para definir previamente estas características como sucedía con ML. A través de distintos procesos, el algoritmo se ajusta a la mayor exactitud, mejorando así la precisión para el próximo dato. Es la técnica que más se asemeja al aprendizaje humano, y lo hace usando redes neuronales, que tratan de asemejar el funcionamiento del cerebro con la combinación de distintos datos de entrada, pesos de ponderación y *bias*. El *Deep Learning* también puede ser de tipo supervisado o no supervisado, dependiendo de si los datos están o no etiquetados, como sucede en el *Machine Learning*.

Las redes neuronales 1.4 consisten en distintas capas de nodos interconectados, donde cada nodo se basa en su capa anterior para refinar y optimizar la precisión. Está compuesto por la capas visibles, que son la primera, donde el modelo recibe los nuevos datos de entrada, y la última, donde se obtiene el resultado de predicción o clasificación, y las capaz ocultas o *hidden layers* que son las que realizan todo el proceso de optimización hasta llegar al resultado final.

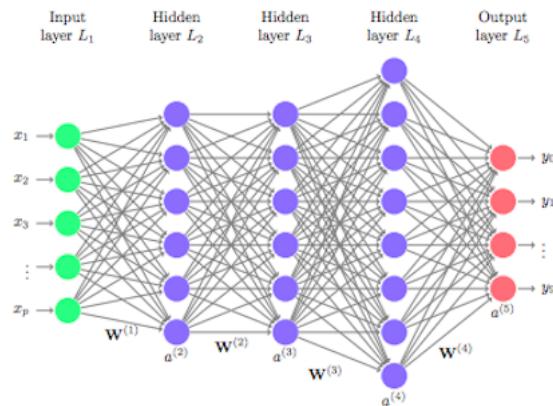


Figura 1.4: Simple ejemplo de una red neuronal en el algoritmo de Deep Learning.<sup>3</sup>

Este aprendizaje es una técnica muy importante a día de hoy que es necesaria para obtener resultados óptimos como en el campo de la visión artificial.

Algunas de las aplicaciones más conocidas o que acompañan a los humanos en su día a día son los coches autónomos, como se ha comentado anteriormente, que necesitan reconocer de manera inmediata qué decisiones tomar frente a la situación que se les

<sup>3</sup>Imágenes obtenidas de shorturl.at/kmpT5

presente, los asistentes virtuales, que son capaces de establecer una conversación, y que toman cada contacto con el usuario como un nuevo dato que utilizan para aprender continuamente, reconocimiento visual 1.5 de caras, personas o elementos en imágenes o vídeos pudiendo detectar el objeto de interés, generación automática de texto o la traducción de este partiendo de imágenes, recomendaciones de películas o serie en base a los gustos de cada persona en plataformas como *Netflix*, *Spotify* o *Amazon*, hasta en el campo de la biomedicina, siendo capaz de predecir los cambios en los procesos celulares debido a la variación genética o de detectar si las radiografías indican enfermedades, o con el Big Data, que son todos los datos básicos que se recogen constantemente de los humanos a través de internet donde el *Deep Learning* es capaz de extraer análisis y aprender de todas estas cantidades de datos.

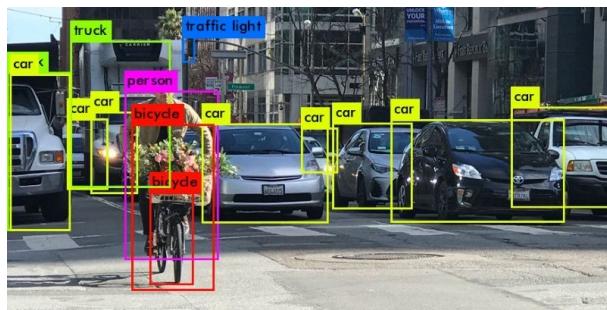


Figura 1.5: Reconocimiento visual en visión con técnica de Deep Learning.<sup>4</sup>

### 1.1.5. Sistema multisensorial

Debido a la importancia de las anteriores secciones y al auge que están teniendo en la actualidad, desde la robótica al *Deep Learning*, la decisión de este trabajo ha sido realizar un sistema multisensorial monitorizado.

La base idea de este trabajo ha sido pensada para un laboratorio de animales, pues la robótica está presente en este ámbito así como en otros muchos aunque sin embargo, en este sector no trabajan ingenieros, por lo que puede haber trabajo que se realice en el laboratorio que podría mejorarse hasta incluso automatizarse. Para obtener más información, se ha contactado con el laboratorio de animales de la Universidad de Alcalá de Henares<sup>5</sup> para conocer la situación en la que trabajan. Se ha notado que hay bastantes cosas que se podrían mejorar y se ha enfocado el objetivo en el estudio de los ratones del laboratorio, que requieren una observación constante así como del entorno

<sup>4</sup>Imágenes obtenidas de shorturl.at/kKRS6

<sup>5</sup><https://www.uah.es/es/>

en el que se encuentran debido a que deben ser unas condiciones determinadas. Después de conocer esta información la idea del trabajo ha tomado más forma e idea, decidiendo así componer un sistema multisensorial, que consiste en dotar a un computador de diferentes sensores que le permitan obtener datos del entorno que le rodea para poder actuar en función a ellos para ofrecer a la cubeta en la que se encuentran los ratones las mejores condiciones y así poder obtener parámetros importantes del entorno de los ratones como pueden ser la temperatura, humedad o calidad del aire. Así mismo, debido al desconocimiento de los trabajadores del laboratorio del mundo de la robótica y la ingeniería en general, se ofrece una interfaz gráfica donde los valores numéricos obtenidos por los sensores, así como cualquier interacción necesaria se traducen en *widgets* comprensibles para cualquier usuario de una manera intuitiva y sencilla.

Por último, con una cámara, que es uno de los sensores incluidos en este sistema, se obtienen vídeos de los ratones, que, aplicando a la visión un algoritmo de *Deep Learning* para realizar reconocimiento de objetos, en este caso, ratones, sea el sistema capaz de evaluar el tiempo que pasan los ratones en las diferentes zonas de la cubeta así como en cada una de las dos cubetas interconectadas por un tubo, evitando así el trabajo que tiene una persona a día de hoy de grabar vídeos de veinticuatro horas y visualizarlos para poder analizar las características del entorno y el comportamiento de los ratones sin automatización alguna.

Una característica extra a este sistema es que se ha realizado con una plataforma de bajo coste, lo que significa que tanto el sistema computacional como los sensores utilizados son económicamente accesibles para cualquier usuario, por lo que está al alcance de cualquier usuario que lo quiera incorporar.

### 1.1.6. Comparación con otros sistemas del mercado

Algunos de los sistemas que hay actualmente en el mercado son:

- *Sistema de monitorización en tiempo real la salud animal*<sup>6</sup>. La Universidad Complutense de Madrid<sup>7</sup> realizó un sistema para detectar de forma temprana las enfermedades o infecciones que tienen síntomas febriles en animales de granja, más concretamente en cerdos. Este sistema se desarrolló con la instalación de microchips y otro tipo de sensores en los animales para poder controlar las condiciones como su temperatura o el consumo de agua de los bebederos, de

---

<sup>6</sup>[https://www.ucm.es/data/cont/docs/3-2017-05-22-2017\\_05\\_not5.pdf](https://www.ucm.es/data/cont/docs/3-2017-05-22-2017_05_not5.pdf)

<sup>7</sup><https://www.ucm.es>

forma que un ordenador muestra esta información de manera gráfica.

También se incluye una monitorización grupal mediante grabaciones diarias para que el sistema detecte cualquier tipo de actividad no común.

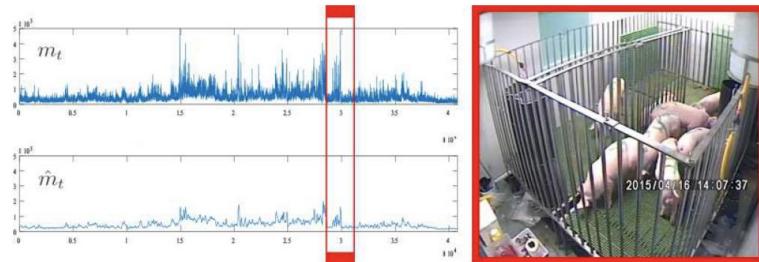


Figura 1.6: Patrón de movimientos e imagen grabada del grupo de cerdos.

- *Monitorización de rebaños de bovinos a través de redes de sensores inalámbricos*<sup>8</sup>. La Facultad de Zootecnia e Ingeniería de la Universidad de São Paulo<sup>9</sup> desarrolló un sistema de sensores con el fin de obtener datos fisiológicos de los animales, en este caso vacas, y obtener las condiciones en las que los animales tienen menos perturbaciones en su comportamiento natural, pues las condiciones climatológicas en cada región pueden afectar de diferentes maneras con estrés, pérdidas productivas o incluso la muerte. Este sistema se creó con la instalación de sensores en cada una de las vacas creando una red de sensores que se comunican con una estación para obtener todos los datos del conjunto de animales.

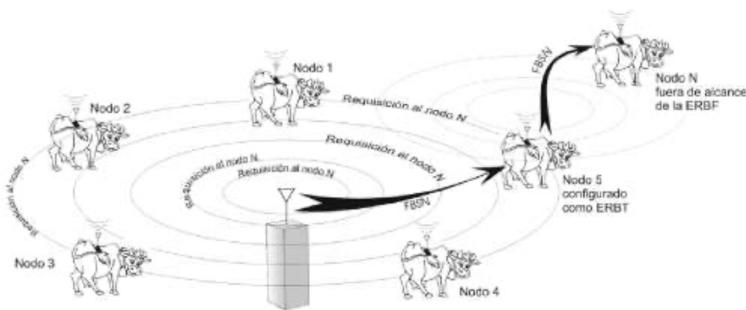


Figura 1.7: Esquema del desarrollo del sistema mediante sensores.

El comportamiento que tienen los animales puede ofrecer mucha información acerca de su salud si se mantiene una observación constante, pero también tienen una gran

---

<sup>8</sup>[https://scielo.isciii.es/scielo.php?script=sci\\_arttext&pid=S0004-05922009000200010](https://scielo.isciii.es/scielo.php?script=sci_arttext&pid=S0004-05922009000200010)  
<sup>9</sup><http://www.usp.br>

importancia las condiciones del entorno en la que se encuentran, ya que pueden ser decisivas para detectar el motivo de su comportamiento. Por ello, es importante tener un seguimiento constante y automatizado de estas características permitiendo al humano encargarse únicamente en controlar los datos que el sistema registra.

En el capítulo posterior se presentarán los objetivos planteados para obtener el proyecto así como el plan de trabajo con los requisitos definidos y cómo se han llevado a cabo.

En próximos capítulos se profundizará más en el proyecto, desde las herramientas usadas para llevarlo a cabo hasta el desarrollo específico que se ha seguido.

---

# **Capítulo 2**

## **Objetivos**

---

*Quizás algún fragmento de libro inspirador...*

Autor, Título

Después de dar un contexto al trabajo en el anterior capítulo, en este se presenta el plan de trabajo, definiendo los objetivos tanto generales como específicos que se han marcado para desarrollarlo y los requisitos que debe respetar el proyecto. Posteriormente se explica la metodología utilizada para cumplir con los objetivos.

### **2.1. Descripción del problema**

El objetivo general de este proyecto es crear un sistema accesible económicamente para cualquier usuario que a través del uso de diferentes sensores sea capaz de medir las condiciones del entorno en la que se encuentran los ratones de laboratorio y mostrarlo en una interfaz gráfica en tiempo real y que sea fácilmente manejable por cualquier usuario. Asimismo, que el sistema también sea capaz de controlar el tiempo en el que los animales pasan haciendo las diferentes tareas.

Actualmente el control de estas características es mediante un humano, no estando automatizado por ningún sistema. Es por ello que el objetivo de este trabajo tiene una necesidad real de la que actualmente carece el laboratorio.

Para cumplir este objetivo general, es necesario marcar una serie de objetivos específicos para el correcto desarrollo del trabajo:

- Recoger la lectura de los sensores en un mismo fichero. Cada sensor necesita las librerías pertinentes para obtener una lectura correcta, por lo que se creará una clase por sensor de manera que sea más fácil unificar todas estas lecturas. Se utilizará la librería *Thread* que permitirá ejecutar concurrentemente

la lectura de todos los sensores simultáneamente con el uso de la función `threads.append(Thread(target=func))`.

- Crear un servidor web para los dos sensores que recogen imágenes. Para la cámara térmica, la única función del servidor será mostrar la imagen que graba la cámara. Sin embargo, para la cámara normal, además de mostrar las imágenes, deberá incorporar un botón que permita iniciar y parar la grabación cuando el usuario quiera, guardando el vídeo automáticamente en el sistema, así como un recuadro indicando la hora y la fecha. Para llevar esto a cabo se utilizará la librería *OpenCV*, que lo permitirá utilizando la función `cv2.rectangle()` y `cv2.putText()` que permiten crear un rectángulo y poner un texto en las coordenadas indicadas como parámetro respectivamente.

Estos servidores de flask estarán protegidos por un factor de doble autenticación para aportar seguridad.

- Detectar los diferentes ratones mediante un algoritmo de reconocimiento de objetos a través de *TensorFlowLite* para poder determinar el tiempo que pasan los animales haciendo las diferentes actividades.

## 2.2. Requisitos

El trabajo cumplirá la siguiente serie de requisitos:

- El sistema sobre el que se realizará el trabajo será la RaspberryPi 4b+, haciendo de este un sistema económico.
- El lenguaje de programación será Python, debido a la familiaridad del autor con él además de la amplia variedad de librerías que permite usar.
- La interfaz de usuario será creada con Node-Red, y deberá ser fácil e intuitiva.

## 2.3. Metodología

Para la satisfacción de los objetivos y el cumplimiento de los requisitos mencionados anteriormente, se han usado diferentes herramientas para el correcto control y desarrollo del proyecto.

Para el seguimiento del trabajo se han llevado a cabo reuniones semanales con el tutor del trabajo en la plataforma Microsoft Teams<sup>1</sup>, donde se compartían los problemas

---

<sup>1</sup><https://www.microsoft.com/es-ar/microsoft-teams/log-in>

surgidos durante la semana junto a posibles soluciones que evaluar, además del control y evaluación de los objetivos marcados la semana anterior así como el establecimiento de los nuevos para la próxima semana. También se ha utilizado el correo electrónico para problemas que surgían a lo largo de la semana.

Además, se ha contactado con los biotecnólogos del laboratorio de la Universidad de Alcalá de Henares tanto por videoconferencia como por correo electrónico para conocer la situación y la prioridad de los problemas que les gustaría solucionar.

La evolución de todo el trabajo se ha registrado en GitHub<sup>2</sup> donde se ha ido escribiendo el progreso que se ha llevado para realizar el trabajo, junto a los problemas que han ido surgiendo acompañados de imágenes, así como las soluciones que han servido para solventar dichos problemas. Se han aportado fotos y vídeos mostrando el funcionamiento de las distintas partes del sistema, así como el funcionamiento entero de este.

---

<sup>2</sup><https://github.com/jmvega/tfg-icebollada>

---

# Capítulo 3

# Plataforma de desarrollo

---

*Quizás algún fragmento de libro inspirador...*

Autor, Título

En este capítulo se definen tanto la infraestructura utilizada como los métodos, tanto a nivel software como hardware, que se han utilizado para desarrollar este trabajo.

## 3.1. Infraestructura hardware

Para este trabajo se ha utilizado la Raspberry Pi 4b+ con su respectivo sistema operativo Raspbian. Junto a ella, se han utilizado una serie de sensores como la PiCam, que es la cámara de Raspberry, una cámara térmica, sensores de temperatura, humedad, calidad del aire o medición del nivel de agua. Para conectar de forma cómoda todos estos sensores se ha utilizado una protoboard junto con los cables dupont necesarios para la correcta instalación de dichos sensores.

Para conseguir que todo funcione adecuadamente, se ha utilizado la infraestructura software descrita a continuación.

## 3.2. Lenguaje python

Python<sup>1</sup> es un lenguaje de alto nivel de programación, interpretado y orientado a objetos cuya filosofía hace hincapié en la legibilidad de su código y por ello su sintaxis es sencilla. Utiliza módulos y librerías como TensorFlow para aplicaciones en Machine Learning, Pandas para análisis de datos, Flask para aplicaciones web o SQLAlchemy para comunicación entre bases de datos y programas, entre otros. Esto

---

<sup>1</sup><https://www.python.org>

facilita la reusabilidad de código y la programación modulada. No requiere del proceso de compilación, lo que lo convierte en un lenguaje muy rápido en el ciclo de edición, prueba y depuración.

Actualmente es el lenguaje de programación más usado en el mundo según la calificación de la empresa TIOBE<sup>2</sup> debido a las características anteriores, formando una gran comunidad. Además, también es el más popular en el ámbito del Machine Learning. Algunas de las aplicaciones que usan Python son Google, Netflix, Dropbox o Spotify.

La decisión de usar Python para el desarrollo de este TFG ha sido, además de que el autor está familiarizado con el lenguaje, por las numerosas librerías útiles para este proyecto adaptadas a Raspberry que Python posee.

En este trabajo, Python se utiliza para la lectura de los sensores de forma concurrente, para la creación de los dos servidores web que tiene tanto la PiCamera como la cámara térmica con Flask y para el reconocimiento de los ratones a través de la librería TensorFlow Lite.

### 3.3. TensorFlow Lite

TensorFlow Lite<sup>3</sup> es una variación de TensorFlow adaptada a dispositivos como teléfonos móviles con sistemas operativos como Android o IOS, microcontroladores o dispositivos embebidos como Raspberry. Es una plataforma de código abierto y multiplataforma que entrena un modelo para la posterior clasificación o reconocimiento de imágenes, entre otros.

Las características que hacen que pueda utilizarse en este tipo de dispositivos son ligereza, ya que estos dispositivos tienen límite tanto en el almacenamiento como sobretodo en la capacidad de cómputo, baja latencia dado a que las inferencias se realizan en el dispositivo y no en un servidor externo, seguridad ya que el modelo viene implementado en el propio dispositivo o consumo de energía óptimo dado a que no es necesario que el dispositivo esté conectado a la red, que consume mucha energía.

El uso de TensorFlow Lite ha sido, en primer lugar, para la creación y entrenamiento de un modelo capaz de detectar ratones en una imagen, basándose en un dataset de 60 imágenes. En segundo lugar, para la detección de ratones en cualquier imagen o vídeo, siendo capaz de detectar el número de ratones o el lugar en el que se encuentra cada

---

<sup>2</sup><https://www.tiobe.com/tiobe-index/>

<sup>3</sup><https://www.tensorflow.org/lite/>

uno. Para conseguir esta detección a tiempo real del vídeo grabado por la cámara es requerido el uso de otra librería llamada OpenCV.



Figura 3.1: Detección de ratones en una imagen usando TensorFlow Lite.

### 3.4. OpenCV

OpenCV<sup>4</sup> (Open Source Computer Vision Library) es una librería de software de visión computacional y machine learning de código abierto. Es ampliamente usada en todo el mundo debido al soporte multiplataforma que ofrece para Windows, Linux, MacOS y Android, además de ofrecer interfaz en diferentes lenguajes como Python, Java, C++ y MATLAB.

Compuesto por más de 2500 algoritmos, OpenCV está especializado en la visión computacional y en algoritmos de aprendizaje, permitiendo poder hacer cosas como el reconocimiento de rostros, la identificación o clasificación de objetos, la extracción de modelos 3D, el rastreo de los movimientos que hay en cámara, mejorar la calidad de las imágenes, el filtrado de color, la detección de bordes, o las eliminación de los ojos rojos de las fotos.

El uso de OpenCV en este trabajo ha permitido la manipulación de los vídeos grabados por las cámaras en tiempo real, de manera que se ha podido integrar en él la fecha y hora del momento, así como poder guardar los vídeos cuando el usuario lo solicite. Debido a su amplio uso y su multitud de librerías, es posible mostrar los vídeos de

---

<sup>4</sup><https://opencv.org>

ambas cámaras en el servidor web sin problema gracias al fácil uso que tiene con Flask.

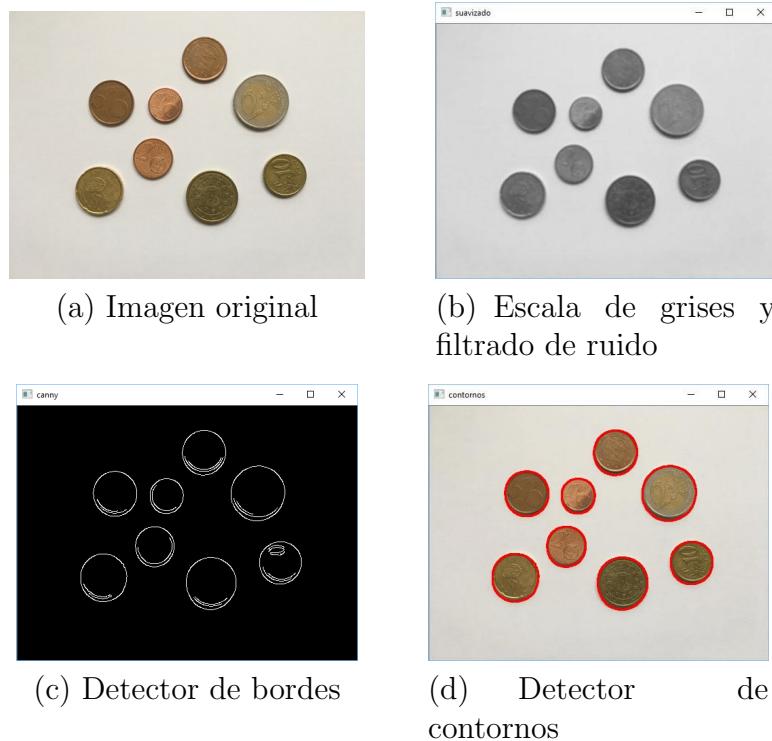


Figura 3.2: Ejemplo de detección de bordes con OpenCV.

### 3.5. Flask

Flask<sup>5</sup> es un *microframework* web escrito en Python. Esto significa que ofrece al usuario herramientas y librerías para crear una aplicación web, y al ser minimalista, no requiere de otras dependencias para realizar las cosas básicas. Ofrece una serie de extensiones que permiten dotar de más características a la aplicación, como autenticación o manejo a través de comandos entre otros.

Se compone de dos partes; los *templates*, que están escritos en HTML e indican la organización y diseño que tendrá cada página al ser visualizada y la parte de código en python que indica las rutas de cada página y las funciones de cada ruta, es el fichero que se ejecuta para crear el servidor, y por defecto éste se crea en el puerto 5000, aunque se puede modificar.

Para este proyecto se ha utilizado Flask para crear los dos servidores web de las

---

<sup>5</sup><https://flask.palletsprojects.com/en/2.1.x/>

cámaras. Se ha preferido Flask frente a Django<sup>6</sup>, que es el más conocido para el uso con Python, debido a que es más sencillo de utilizar y de aprender. En estos servidores, después de haber realizado el *login* o el registro con el plugin flask-login, se accede a la visualización de cada cámara. Además, en el caso de la cámara normal, usando Flask se ha integrado un botón que permite empezar o parar la grabación del vídeo.



Figura 3.3: Muestra de los dos servidores del trabajo usando Flask.

### 3.6. Node-Red

Node-Red<sup>7</sup> es una herramienta de programación creada por IBM y escrita en JavaScript que permite conectar dispositivos hardware y servicios en línea. Muestra de manera visual las relaciones de los distintos componentes mediante nodos en un panel, mostrando gráficamente el flujo de la información, además de que es un entorno de ejecución ligero que está basado en Node.js, haciéndolo ideal para ejecutarse en hardware de bajo coste como la Raspberry.

Está compuesto por dos partes principales. Una de ellas es la edición de flujo desde navegador, donde muestra los nodos disponibles en el margen izquierdo así como la disposición y conexión del flujo que siguen los nodos seleccionados en la pantalla central. La otra parte, es la llamada *dashboard*, que muestra la interfaz de usuario en base al flujo de los nodos de la parte previamente comentada.

Node-Red tiene una amplia variedad de nodos, permitiendo conexiones MQTT, UDP,

<sup>6</sup><https://www.djangoproject.com>

<sup>7</sup>[urlhttps://nodered.org](https://nodered.org)

TCP, la interacción directa con los pines de Raspberry, la conexión con twitter o el correo electrónico, la creación de ficheros tipo csv, xml o json o la ejecución de ficheros ya existentes en la computadora. También permite la creación de funciones directamente en el flujo de nodos, teniendo en cuenta que tienen que ser escritas en JavaScript. Pueden crearse nuevos nodos para agregar nuevas capacidades gracias a los más de 225000 módulos que hay en el repositorio de paquetes.

Los flujos de datos se pueden importar o exportar con ficheros JSON, permitiendo compartirlos con cualquier persona. La página de Node-Red tiene una librería de flujos que han creado otros usuarios, teniendo así acceso a ellos para utilizarlos como base en otros proyectos.

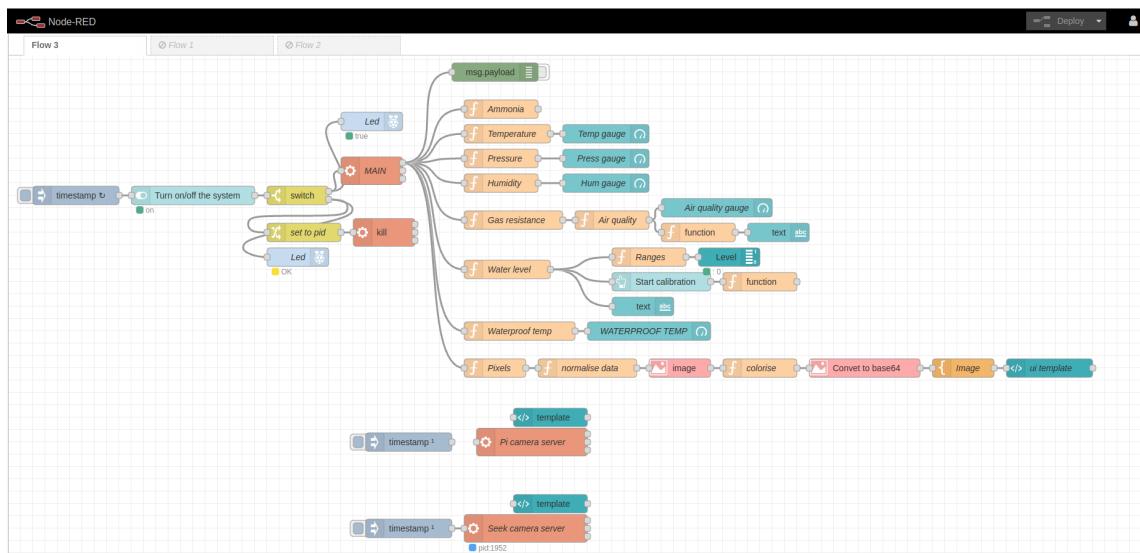


Figura 3.4: Flujo del proyecto en Node-Red.

Para el desarrollo de este TFG, Node-Red ha tenido un papel muy importante. Además de tener una gran comunidad que permite la resolución rápida de cualquier duda, ha permitido la creación de la interfaz de usuario sin problema, donde se muestran las mediciones de cada sensor, los dos servidores web de las cámaras y los *widgets* necesarios que permiten al usuario interactuar con la interfaz de una manera sencilla para obtener toda la información.

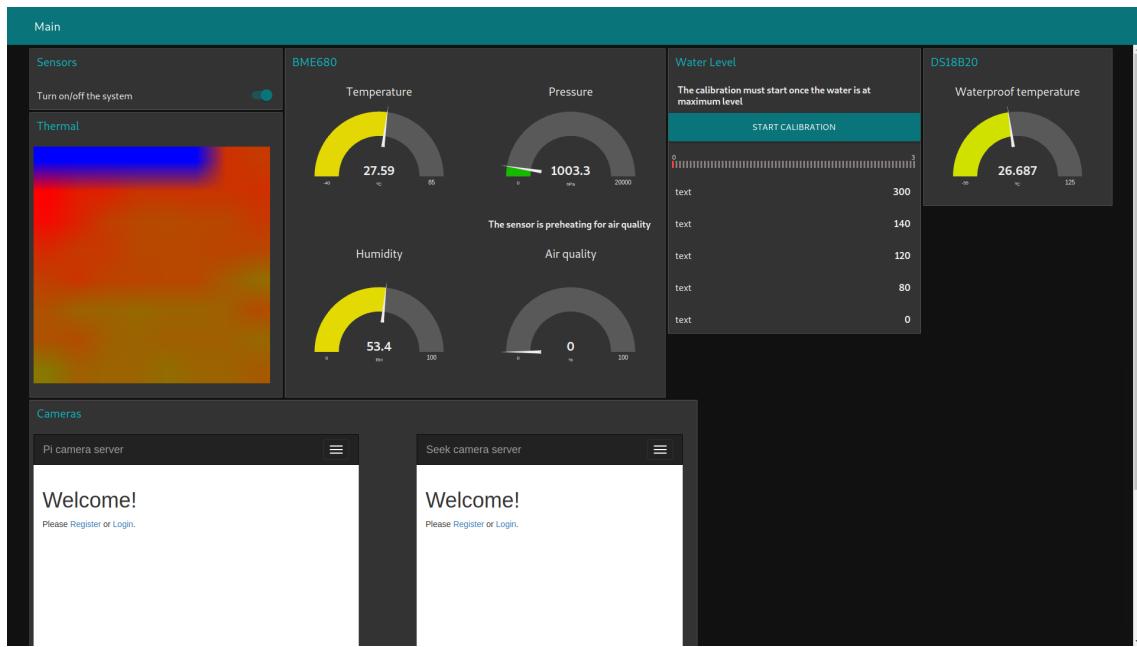


Figura 3.5: Muestra del uso de Node-Red en el trabajo.

---

# Capítulo 4

## Diseño

---

*Quizás algún fragmento de libro inspirador...*

Autor, Título

Escribe aquí un párrafo explicando brevemente lo que vas a contar en este capítulo. En este capítulo (y quizás alguno más) es donde, por fin, describes detalladamente qué has hecho y qué experimentos has llevado a cabo para validar tus desarrollos.

### 4.1. Snippets

Puede resultar interesante, para clarificar la descripción, mostrar fragmentos de código (o *snippets*) ilustrativos. En el Código 4.1 vemos un ejemplo escrito en C++.

---

```
void Memory::hypothesizeParallelograms () {
    for(it1 = this->controller->segmentMemory.begin(); it1++) {
        squareFound = false; it2 = it1; it2++;
        while ((it2 != this->controller->segmentMemory.end()) && (!squareFound))
        {
            if (geometry::haveACommonVertex((*it1), (*it2), &square)) {
                dist1 = geometry::distanceBetweenPoints3D ((*it1).start, (*it1).end);
                dist2 = geometry::distanceBetweenPoints3D ((*it2).start, (*it2).end);
            }
        // [...]
    }
}
```

---

Código 4.1: Función para buscar elementos 3D en la imagen

En el Código 4.2 vemos un ejemplo escrito en Python.

### 4.2. Verbatim

Para mencionar identificadores usados en el código —como nombres de funciones o variables— en el texto, usa el entorno literal o verbatim

---

```

def mostrarValores():
    print (w1.get(), w2.get())

master = Tk()
w1 = Scale(master, from_=0, to=42)
w1.pack()
w2 = Scale(master, from_=0, to=200, orient=HORIZONTAL)
w2.pack()
Button(master, text='Show', command=mostrarValores).pack()

mainloop()

```

---

Código 4.2: Cómo usar un Slider

`hypothesizeParallelograms()`. También se puede usar este entorno para varias líneas, como se ve a continuación:

```

void Memory::hypothesizeParallelograms () {
    // add your code here
}

```

### 4.3. Ecuaciones

Si necesitas insertar alguna ecuación, puedes hacerlo. Al igual que las figuras, no te olvides de referenciarlas. A continuación se exponen algunas ecuaciones de ejemplo: Ecuación 4.1 y Ecuación 4.2.

$$H = 1 - \frac{\sum_{i=0}^N \frac{(\frac{d_{js} + d_{je}}{2})}{N}}{M} \quad (4.1)$$

Ecuación 4.1: Ejemplo de ecuación con fracciones

$$v(\text{entrada}) = \begin{cases} 0 & \text{if } \epsilon_t < 0,1 \\ K_p \cdot (T_t - T) & \text{if } 0,1 \leq \epsilon_t < M_t \\ K_p \cdot M_t & \text{if } M_t < \epsilon_t \end{cases} \quad (4.2)$$

Ecuación 4.2: Ejemplo de ecuación con array y letras y símbolos especiales

### 4.4. Tablas o cuadros

Si necesitas insertar una tabla, hazlo dignamente usando las propias tablas de L<sup>A</sup>T<sub>E</sub>X, no usando pantallazos e insertándolas como figuras... En el Cuadro 4.1 vemos

un ejemplo.

Parámetros	Valores
Tipo de sensor	Sony IMX219PQ[7] CMOS 8-Mpx
Tamaño del sensor	3.674 x 2.760 mm (1/4"format)
Número de pixels	3280 x 2464 (active pixels)
Tamaño de pixel	1.12 x 1.12 um
Lente	f=3.04 mm, f/2.0
Ángulo de visión	62.2 x 48.8 degrees
Lente SLR equivalente	29 mm

Cuadro 4.1: Parámetros intrínsecos de la cámara

---

# **Capítulo 5**

# **Conclusiones**

---

*Quizás algún fragmento de libro inspirador...*

Autor, Título

Escribe aquí un párrafo explicando brevemente lo que vas a contar en este capítulo, que básicamente será una recapitulación de los problemas que has abordado, las soluciones que has prouesto, así como los experimentos llevados a cabo para validarlos. Y con esto, cierras la memoria.

## **5.1. Conclusiones**

Enumera los objetivos y cómo los has cumplido.

Enumera también los requisitos implícitos en la consecución de esos objetivos, y cómo se han satisfecho.

No olvides dedicar un par de párrafos para hacer un balance global de qué has conseguido, y por qué es un avance respecto a lo que tenías inicialmente. Haz mención expresa de alguna limitación o peculiaridad de tu sistema y por qué es así. Y también, qué has aprendido desarrollando este trabajo.

Por último, añade otro par de párrafos de líneas futuras; esto es, cómo se puede continuar tu trabajo para abarcar una solución más amplia, o qué otras ramas de la investigación podrían seguirse partiendo de este trabajo, o cómo se podría mejorar para conseguir una aplicación real de este desarrollo (si es que no se ha llegado a conseguir).

## 5.2. Corrector ortográfico

Una vez tengas todo, no olvides pasar el corrector ortográfico de L<sup>A</sup>T<sub>E</sub>Xa todos tus ficheros *.tex*. En Windows, el propio editor TeXworks incluye el corrector. En Linux, usa aspell ejecutando el siguiente comando en tu terminal:

```
aspell --lang=es --mode=tex check capitulo1.tex
```