



GRADO EN INGENIERÍA DE ROBÓTICA SOFTWARE

Escuela de Ingeniería de Fuenlabrada

Curso académico 20xx-20xx

Trabajo Fin de Grado

Escribe el título del trabajo aquí
con la segunda línea aquí

Tutor: Julio Vega Pérez
Autor: Julia López Augusto



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

Primero de todo me gustaría agradecer a mi tutor Julio toda la confianza y el apoyo depositado para poder hacer posible este proyecto. Sin ti nada de esto hubiera sido posible.

También agradecer a todos los profesores que he tenido a lo largo de la carrera lo importante que habéis sido para mí tanto en mi crecimiento académico, profesional como personal. Gracias a vosotros me habéis hecho ingeniera.

Otras personas que han sido, son y serán fundamentales en mi vida son mis padres; a los cuáles estaré eternamente agradecida por no soltarme nunca de la mano y darme todas las herramientas que han hecho posible ser quien soy hoy en día. Os quiero muchísimo.

Una persona que apareció en mi vida hace ya unos 7 años fue Fran, quien trajo luz en medio de tanta oscuridad y ha sido el mejor acompañante, amigo y amante de cada aventura que se presenta en la vida. Te amo con todo mi corazón y gracias de verdad por todo, ya lo sabes.

Los abuelos deberían ser eternos pero desgraciadamente en mi caso marcharon mucho antes de lo debido. Sé que les encantaría ver a su nieta graduada como ingeniera y me encantaría poder darles un abrazo muy fuerte y poder celebrarlo juntos pero bueno, sé que allá donde estéis estaréis muy orgullosos mí tanto como yo os echo de menos a vosotros. Gracias por los años que pude conocerlos y disfrutarlos, quedarán para siempre en mi retina.

Los amigos son la familia que uno puede elegir y hoy en día puedo estar muy orgullosa y agradecida de la gente que tengo a mi lado. Gracias a Jimena, Elisa, Sofía, Maryu, Santi, Gonzalo y Álvaro por ser tan maravillosos y por estar ahí estos 4 años de alegrías, lloros, mucho trabajo y dolores de cabeza; sois increíbles.

*A mi misma;
por nunca tirar de la toalla*

Chozas de Canales, 4 de Junio de 2024

Julia López Augusto

Resumen

Escribe aquí el resumen del trabajo. Un primer párrafo para dar contexto sobre la temática que rodea al trabajo.

Un segundo párrafo concretando el contexto del problema abordado.

En el tercer párrafo, comenta cómo has resuelto la problemática descrita en el anterior párrafo.

Por último, en este cuarto párrafo, describe cómo han ido los experimentos.

Acrónimos

SRI *Stanford Research Institute*

KUKA *Keller und Knappich Augsburg*

ABB *Asea Brown Boveri*

BSA *Backtracking Spiral Algorithm*

FLL *First Lego League*

AGV *Automatic Guided Vehicles*

AMR *Autonomous Mobile Robots*

MITMA *Ministerio de Transportes, Movilidad y Agenda Urbana*

CEDEX *Centro de Estudios y Experimentación de Obras Públicas*

AEC *Asociación Española de la Carretera*

ACEX *Asociación de Empresas de Conservación y Explotación de Infraestructuras*

SIG *Sistema de Información Geográfica*

IA *Inteligencia Artificial*

GPS *Global Positioning System*

DANA *Depresión Aislada en Niveles Altos*

CAD *Computer-Aided Design*

PDCA *Plan Do Check Act*

CSI *Camera Serial Interface*

UART *Universal Asynchronous Receiver-Transmitter*

PWM *Pulse Width Modulation*

LTS *Long Time Support*

ROS *Robot Operating System*

CLI *Command Line Interface*

YOLO *You Only Look Once*

Índice general

1. Introducción	1
1.1. La robótica	1
1.1.1. Robots industriales	4
1.1.2. Robots de servicio	6
1.2. Robots de campo	9
1.3. Robots de bajo coste	10
1.4. Conservación de carreteras en España	11
1.5. Deterioros del pavimento	13
2. Estado del arte	18
3. Objetivos	24
3.1. Descripción del problema	24
3.2. Requisitos	25
3.3. Competencias	26
3.3.1. Competencias empleadas	26
3.3.2. Competencias adquiridas	27
3.4. Metodología	28
3.5. Plan de trabajo	28
4. Plataforma de desarrollo	31
4.1. Hardware	31
4.1.1. Raspberry Pi 4	31
4.1.2. Raspberry Pi cámara	32
4.1.3. GPS NEO 6M	33
4.1.4. Sevomotor Estándar Parallax	34
4.1.5. Ruedas ActivityBot	34
4.1.6. Google Coral USB	35
4.1.7. Power Bank	36

4.1.8. Rueda Loca	36
4.1.9. Ordenador principal	37
4.2. Software	38
4.2.1. FreeCAD	38
4.2.2. Python	38
4.2.3. OpenCV	40
4.2.4. Ubuntu	40
4.2.5. ROS 2	41
4.2.6. Gazebo	43
4.2.7. Herramientas de monitorización	44
4.2.8. Google Colaboratory	44
4.2.9. YOLOv8	45
4.2.10. TensorFlow Lite	46
4.2.11. Interfaz Web	46
5. Diseño	48
5.1. Snippets	48
5.2. Verbatim	48
5.3. Ecuaciones	49
5.4. Tablas o cuadros	49
5.5. Segunda sección	50
5.5.1. Números	51
5.5.2. Listas	51
6. Conclusiones	53
6.1. Conclusiones	53
6.2. Corrector ortográfico	54
Bibliografía	55

Índice de figuras

1.1.	Ingenios de la antigüedad con fines religiosos	2
1.2.	Ingenios de la antigüedad con fines no religiosos	2
1.3.	Electro y Sparko	3
1.4.	Leyes de la robótica	4
1.5.	Robot Shakey	5
1.6.	Carreta de Stanford	5
1.7.	Robots industriales	6
1.8.	Robots de limpieza	7
1.9.	Robótica enfocada al entretenimiento	7
1.10.	Robots de salud	8
1.11.	Robots de logística	9
1.12.	Robots de campo	10
1.13.	Mano Antropomórfica Híbrida Rígida y Suave	11
1.14.	Firme de una carretera	14
1.15.	Clasificación de deterioros	15
1.16.	Agrietamiento	15
1.17.	Degradación del material de la capa de rodadura	16
1.18.	Deformación de la capa de rodadura sin degradación de material	16
1.19.	Otro tipo de daños	16
1.20.	Operarios reparando un bache ²⁵	17
2.1.	Robot usado para la reconstrucción 3D de baches	19
2.2.	Proyecto HERON	20
2.3.	Visualización de la señal en un mapa en eyesNroad	21
2.4.	Plataforma robótica diseñada para el sellado de grietas	23
4.1.	Raspberry Pi 4 ³³	32
4.2.	Raspberry Pi Cámara V2 ³⁴	33
4.3.	Módulo GPS NEO 6M ³⁵	33

4.4. Tipos de Servomotores	34
4.5. Rueda ActivityBot ³⁸	35
4.6. Google Coral USB ³⁹	35
4.7. Xiaomi Powerbank ⁴⁰	36
4.8. Rueda loca ⁴¹	36
4.9. ASUS VivoBook 14 ⁴²	37
4.10. Logo de Freecad ⁴³	38
4.11. Logo de Python ⁴⁴	39
4.12. Logo de OpenCV ⁴⁵	40
4.13. Logo de Ubuntu ⁴⁶	41
4.14. Distribuciones de ROS 2 usadas	42
4.15. Logo de Gazebo ⁵²	43
4.16. Logo de Rviz ⁵³	44
4.17. Logo de Google Colab ⁵⁶	45
4.18. Logo de YOLOv8 ⁵⁷	45
4.19. Logo de TensorFlow Lite ⁵⁸	46
4.20. Logo de Open Street Maps ⁶⁰	47
5.1. Robot aspirador Roomba de iRobot.	50

Listado de códigos

5.1. Función para buscar elementos 3D en la imagen	48
5.2. Cómo usar un Slider	49

Listado de ecuaciones

5.1. Ejemplo de ecuación con fracciones	49
5.2. Ejemplo de ecuación con array y letras y símbolos especiales	49

Índice de cuadros

4.1. Especificaciones técnicas del ordenador usado	37
4.2. Diferencias entre Ubuntu 20.04 y Ubuntu 22.04	41
5.1. Parámetros intrínsecos de la cámara	50

Capítulo 1

Introducción

La motivación nos impulsa a comenzar y el hábito nos permite continuar

Jim Ryun

La robótica ha sufrido una transformación enorme a lo largo de su historia aunque siempre teniendo en mente el mismo objetivo: cumplir con el deseo humano. Debido a esa transformación y ese deseo, se ha podido consolidar este campo en la actualidad que abarca cada sector que se pueda imaginar. Otra vertiente que ha destacado en la robótica estos últimos años ha sido la creación de robots de bajo coste para que puedan llegar a un mayor número de personas y se puedan beneficiar de esta ciencia.

En el presente capítulo se va a abordar el contexto de la robótica, explicando brevemente su historia para poder entender realmente qué es la robótica y lo que es un robot. También se van a encuadrar los tipos de robots que existen y sus múltiples aplicaciones. Todo esto nos ayudará a poder entender mejor dónde se encuadra el presente trabajo, proporcionando los conocimientos necesarios, tanto teóricos como prácticos, que se describirán a lo largo del documento.

1.1. La robótica

La robótica es el campo de la ingeniería que se enfoca en el diseño, la construcción y la programación de robots para tareas específicas. Por ende, un robot se podría definir como un sistema informático formado por sensores y actuadores imprecisos, ya que tienen ruido. Los robots realizan tareas sucias, aburridas y peligrosas y tienen que ser sensibles al entorno. Sin embargo, no existe una definición unívoca al respecto y depende del campo y de la época de la que queramos hablar. Para poder entenderlo, se va a hacer un breve resumen sobre la historia de la robótica.

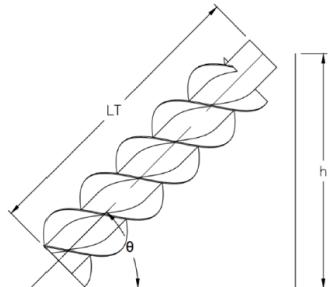
Desde la antigüedad se han desarrollado ingenios o autómatas de los cuales muchos de ellos tenían fines religiosos muestra la Figura 1.1.

Estatuas de Memnon ¹Guerreros de Terracota ²

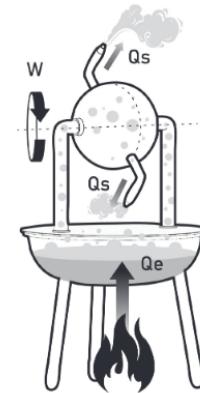
Figura 1.1: Ingenios de la antigüedad con fines religiosos

Otras invenciones que destacaban por otras aplicaciones fueron el tornillo de Arquímedes de Siracusa, la eolípila de Herón de Alejandría, el ornitóptero de Leonardo Da Vinci y el hombre de palo de Juanelo Turriano, entre otras. Se pueden apreciar algunas de dichas invenciones en la Figura 1.2.

En [?] se trata el principio de generación hidroeléctrica, tomando en cuenta el modelo tornillo de Arquímedes. También en el artículo [?] se traza una línea histórica de las máquinas térmicas, teniendo en cuenta a la eolípila.



Tornillo de Arquímedes



Eolípila

Figura 1.2: Ingenios de la antigüedad con fines no religiosos

Durante el siglo XX la ciencia dejó de ser una actividad desarrollada en aislamiento y se desarrolló en laboratorios con más gente. El movimiento del positivismo lógico

¹https://es.wikipedia.org/wiki/Colosos_de_Memn%C3%B3n

²https://es.wikipedia.org/wiki/Guerreros_de_terracota

fomentó las investigaciones ya que este movimiento trataba de dar importancia a la ciencia y dejar de lado la filosofía; también, el contexto de las guerras mundiales y de las bombas nucleares influyeron significativamente en este aspecto. Ya se empieza a acuñar la palabra robot y surge Electro y Sparko de Westinghouse Electric Corporation (Figura 1.3), tratado en muchas ocasiones como uno de los primeros robots, como se describe en [Bidaud,]. Un descubrimiento muy destacado fue y sigue siendo hoy en día son Las Leyes de la Robótica de Isaac Asimov (Figura 1.4), así lo transmite [Barceló, 2004].



Figura 1.3: Electro y Sparko

[Nilsson et al., 1984] nos cuenta que en 1969 se construye por el *Stanford Research Institute* (SRI) Internacional un prototipo experimental llamado Shakey que era una unidad independiente de 1.5 metros, equipado con 2 motores, cámara de televisión y una radio conectada a un ordenador, capaz de navegar en entornos cerrados y estructurados de una forma autónoma. Sus objetivos eran aprender del medio y ser capaz de planificar trayectorias de movimiento y las tareas que le asignaron fueron mover y detectar bloques. Sin embargo, cada movimiento podría tardar más de una hora en computarse y aún así, podrían producirse fallos. A Shakey se le conoce como el primer robot móvil (Figura 1.5).

Asimov's Laws of Robotics (1940)

First Law:

A robot may not injure a human being, or, through inaction, allow a human being to come to harm.

Second Law:

A robot must obey the orders given it by human beings, except where such orders would conflict with the First Law.

Third Law:

A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.

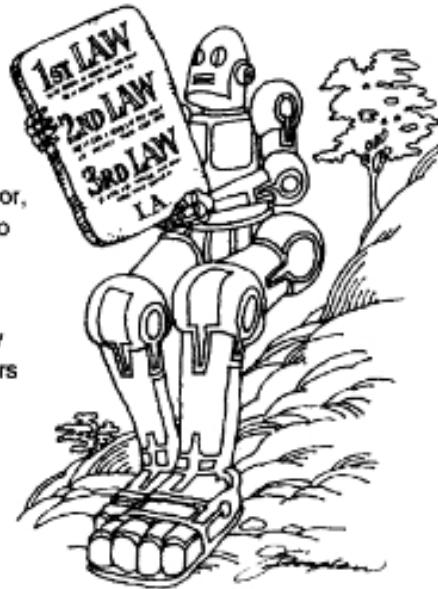


Figura 1.4: Leyes de la robótica

Otro robot muy conocido y descrito en [Earnest, 2012] es la carreta de Stanford que era capaz de ver y moverse en cualquier ambiente. Con la cámara que disponía, era capaz de calcular y trazar distancias. Sin embargo, tardaba 5 horas en recorrer 30 metros (Figura 1.6).

Ya a partir de la década de los 80, se decidió que el acceso a los robots fuese para todo el mundo. Lo que provocó asombro, inquietud y miedo ya que el desconocimiento de aquella situación generó rechazo; cosa que ocurre hoy en día. El mundo de la literatura y cine tampoco ayudaba en ese aspecto ya que se presentaba al robot como algo perjudicial para la humanidad. Afortunadamente, esta situación va menguando con el tiempo y se está consiguiendo ver a los robots como un asistente del ser humano que lo que quiere es mejorar su calidad de vida. Para conocer en qué ámbitos un robot puede ayudar al ser humano, es necesario conocer que se clasifican en robots industriales y de servicio.

1.1.1. Robots industriales

Los robots industriales son brazos robotizados y manipuladores que tienen más de tres grados de libertad, trabajan en entornos controlados y usan efectores como: pinzas,

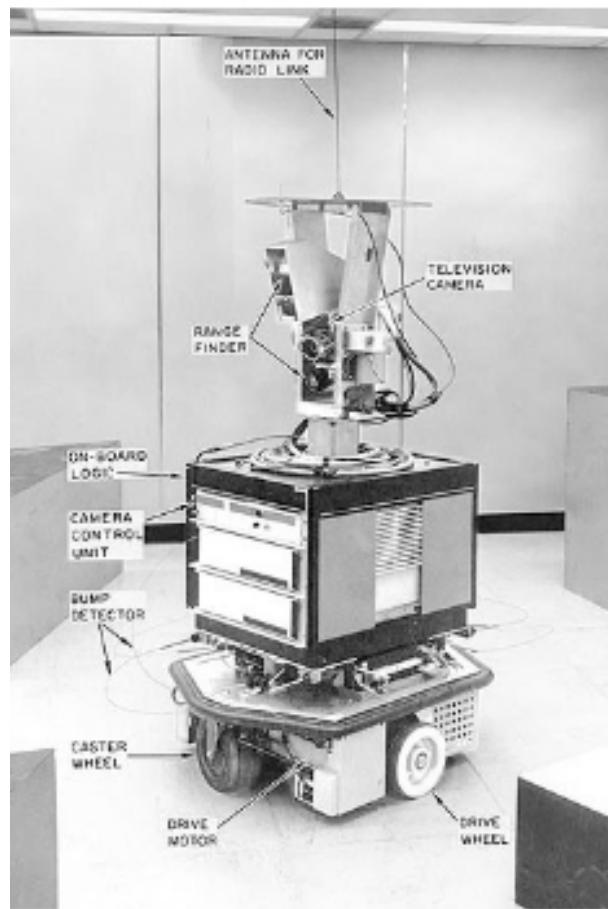


Figura 1.5: Robot Shakey

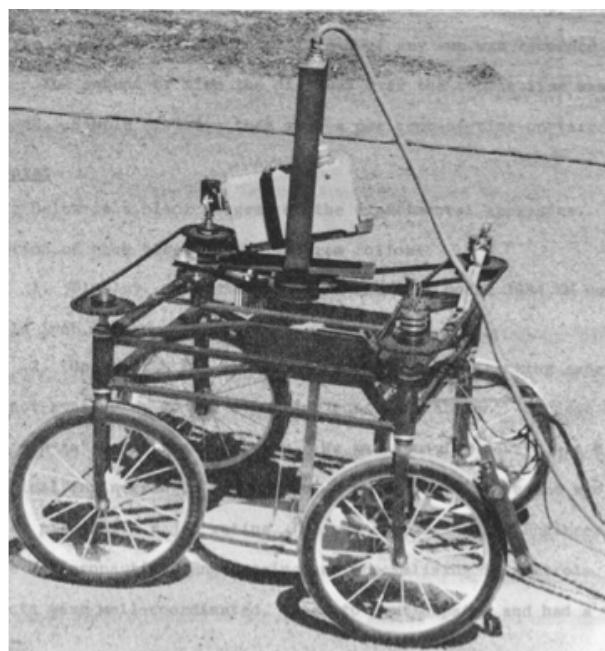
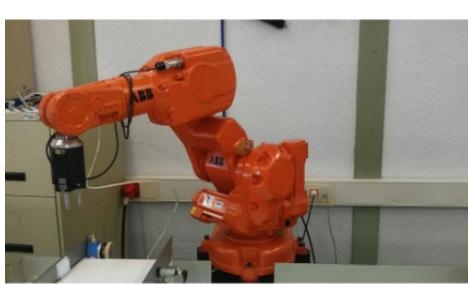
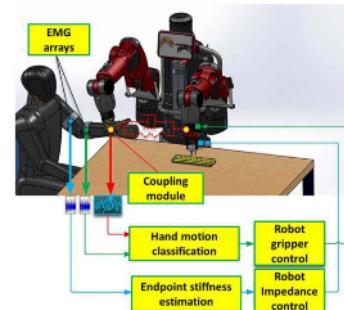


Figura 1.6: Carreta de Stanford

ventosas. etc. Usan control por posición y planificación de trayectorias para poder controlar dichos brazos y poder realizar operaciones como *pick and place*, ensamble de piezas, entre otros. Una variante de los robots industriales son los cobots: capaces de interactuar y colaborar con humanos como se describe en [El Zaatri et al., 2019]. Dentro del mercado de los robots industriales se puede ver que tiene proveedores asentados como KUKA o ABB (Figura 1.7).



Brazo robot IRB 140 ³.



Cobot

Figura 1.7: Robots industriales

1.1.2. Robots de servicio

Los robots de servicios son todos aquellos que no son industriales; por lo tanto, tienen menos de 3 grados de libertad, no trabajan en un entorno controlado, son más difíciles de programar, toman aplicaciones heterogéneas y todavía se encuentran en un mercado inmaduro. Existen muchos tipos de robots de servicio, de los cuales a continuación se podrán enumerar algunos campos con algunos de sus respectivos ejemplos.

Robots de limpieza

En [Plaza, 2023] se define a los robots de limpieza como aquellos robots que se encargan de eliminar la suciedad. Dependiendo de sus características, pueden ser capaces de aspirar y fregar el suelo, o de limpiar los cristales de las ventanas. Estos últimos, son comunes en edificios que tienen grandes ventanales y un difícil acceso a ellos. Las aspiradoras se encuentran en un mercado mundial asentado cuyos inicios eran aspiradoras que usaban sensores de contacto, encoders y una navegación pseudoaleatoria (Figura 1.8 izquierda); y han evolucionado hasta el punto de usar mapas para poder navegar y poder localizarse. También, usan algoritmos sofisticados como navegación de cobertura por barridos sistemáticos BSA y se componen de sensores

³<https://www.youtube.com/watch?v=BBrLA0r89KY>

más sofisticados como láseres y cámaras que son capaces de detectar obstáculos como calcetines y ser capaz de esquivarlos (Figura 1.8 derecha).



Modelo económico



Gama alta

Figura 1.8: Robots de limpieza

Robots de entretenimiento

Son aquellos robots que tienen aplicaciones heterogéneas en un mercado educativo asentado pero también existen muchos prototipos sin un uso comercial claro. En la educación, se ha ido introduciendo la robótica de manera muy atractiva y didáctica a los estudiantes hasta el punto de conseguir tener una asignatura destinada a la robótica y poder participar en competiciones como: *First Lego League* (FLL), Robocup Junior y Robocampeones, entre otros (Figura 1.9 izquierda). En dicha asignatura se adquieren conocimientos generales de programación, impresión 3D, lógica, introducción a la electrónica y los microcontroladores.

Los prototipos nombrados anteriormente suelen ser demostradores tecnológicos que se crean para exhibiciones, se encuentran a la vanguardia de la tecnología y sirven para atraer un mayor número de clientes como puede ser: Spot de Boston Dynamics, Pepper de Softbank o Sophia de Hanson Robotics. En la Figura 1.9 (derecha) se pueden ver un ejemplo de estos demostradores.

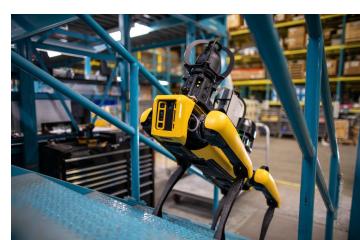
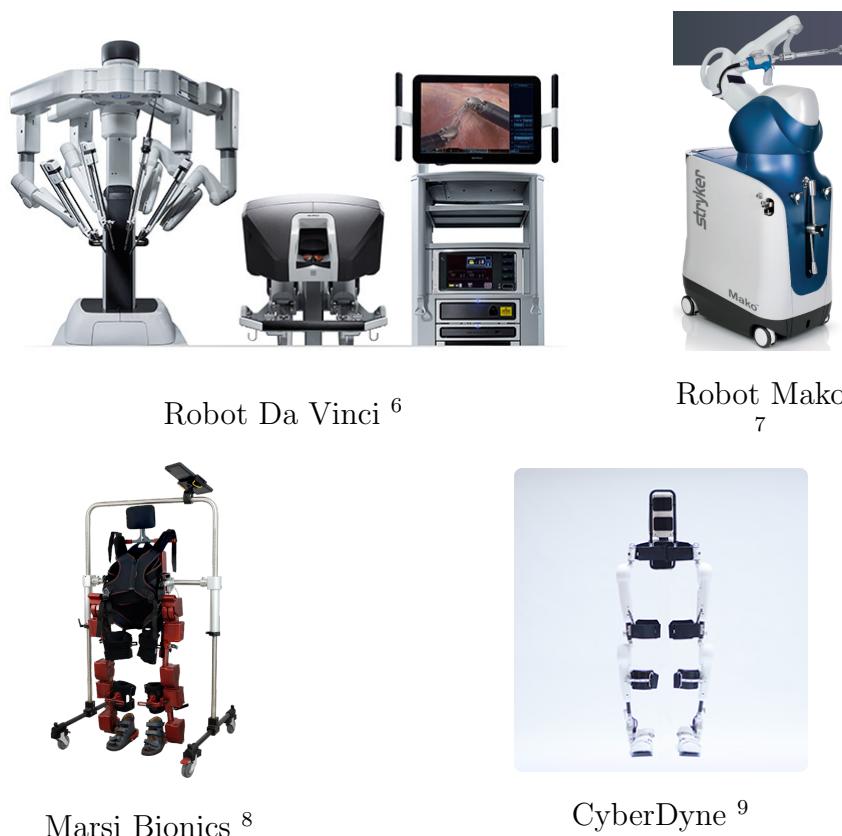
FLL Toledo ⁴Spot de Boston Dynamics ⁵

Figura 1.9: Robótica enfocada al entretenimiento

Robots de salud

Los robots de salud sirven para mejorar la calidad de la atención médica y apoyar a los profesionales de la salud en diversas tareas. De esas tareas se pueden enumerar las siguientes: telepresencia, asistentes personales, cirugía, desinfección, esterilización, transporte interno y rehabilitación. De este gran abanico de tareas se puede destacar algunas aplicaciones que se puede ver en la Figura 1.10.



Robot Da Vinci⁶

Robot Mako⁷

Marsi Bionics⁸

CyberDyne⁹

Figura 1.10: Robots de salud

Robots de logística

Los robots de logística son sistemas automatizados diseñados para mejorar la eficiencia, precisión y velocidad en la gestión de la cadena de suministro y operaciones logísticas en cadenas de montaje y almacenes. Generalmente los sistemas automatizados

⁴<https://www.uclm.es/noticias/febrero2019/toledo/finalfirstlegoleague>

⁵<https://bostondynamics.com/products/spot/>

⁶<https://www.abexsl.es/es/sistema-robotico-da-vinci/que-es>

⁷<https://www.stryker.com/content/dam/stryker/joint-replacement/systems/mako-system-overview/resources>

⁸<https://www.marsibionics.com/>

⁹<https://www.cyberdyne.com/>

son flotas de robots a las que se les aplica distintas arquitecturas software como puede ser AGV o AMR para poder realizar la tarea asignada. También existen prototipos de robots de reparto que son capaces de hacer entrega de última milla. La Figura 1.11 muestra ejemplos de robots de reparto.

Skypod de Exotec¹⁰Amazon Prime Air¹¹

Figura 1.11: Robots de logística

1.2. Robots de campo

En [Thorpe and Durrant-Whyte, 2003] se define a los robots de campo como la automatización de muchas plataformas terrestres, marítimas y aéreas en aplicaciones como la minería, la manipulación de carga, la agricultura, la exploración y explotación submarina, las carreteras, la exploración planetaria, la vigilancia costera y el rescate, entre otros. La robótica de campo se caracteriza por la aplicación de los principios robóticos más avanzados en cuanto a sensado, control y razonamiento en entornos no estructurados y difíciles. El atractivo de la robótica de campo es que es una ciencia desafiante, involucra los últimos principios de ingeniería y diseño de sistemas, y ofrece la verdadera posibilidad de que los principios robóticos hagan una contribución económica y social sustancial en muchas áreas de aplicación diferentes. En general, los robots de campo son plataformas móviles que trabajan al aire libre, a menudo produciendo interacciones fuertes con sus entornos, sin supervisión humana.

En los últimos años se ha podido notar un gran progreso en el desarrollo y la implementación de sistemas robóticos de campo. En la Figura 1.12 se pueden ver ejemplos al respecto.

¹⁰<https://exotecbydexter.com/skypod/>

¹¹<https://www.aboutamazon.es/noticias/innovacion/prime-air>

¹²<https://advanced.farm/technology/strawberry-harvester/>

¹³<https://www.komatsu.com/en/technology/smart-mining/loading-and-haulage/autonomous-haulage-system/>

¹⁴<https://www.ryzerobotics.com/es/tello>

¹⁵<https://es.wikipedia.org/wiki/Perseverance>



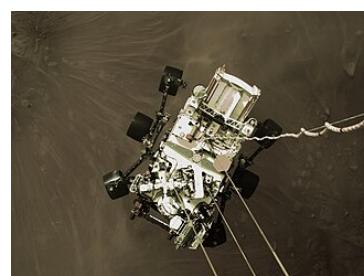
TX Robotic Strawberry Harvester¹²



FrontRunner
Autonomous Haulage
System (AHS)¹³



Drone Tello¹⁴



Perseverance¹⁵

Figura 1.12: Robots de campo

Por contra, debido a las condiciones que se tienen que someter estos robots, su coste es elevado e imposibilita que su uso se pueda extender a aquellos lugares que necesiten de su servicio y que tienen bajos recursos. Es por esto que existe una necesidad que para ciertas aplicaciones, que no tienen condiciones tan extremas, se creen robots asequibles que puedan lidiar con ciertas tareas.

1.3. Robots de bajo coste

Los robots de bajo coste son aquellos robots diseñados y fabricados con el objetivo de ser económicos, accesibles y fáciles de producir. Estos robots suelen emplear componentes menos costosos y métodos de fabricación simplificados para reducir el precio final. Algunas características clave de los robots de bajo coste incluyen:

- *Componentes asequibles.* Utilizan materiales y componentes electrónicos más baratos, como pueden ser: motores de bajo coste como Parallax, microcontroladores como Arduino y ordenadores monoplaca como Raspberry pi.
- *Simplicidad en el diseño.* Tienen diseños más sencillos que han sido creados usando técnicas de diseño e impresión 3D, facilitando su actualización y mantenimiento.
- *Accesibilidad.* Están diseñados para ser utilizados por cualquier tipo de persona, sin tener una formación avanzada de la materia.

- *Educación y prototipos.* Ampliamente usados en educación para facilitar el aprendizaje y también en la creación de prototipos rápidos y asequibles.
- *Versatilidad.* Se adaptan a numerosas aplicaciones, desde las más sencillas hasta a proyectos más complejos.

En resumen, los robots de bajo coste permiten la democratización de la tecnología robótica, facilitando su acceso a un público más amplio y fomentando la innovación y el aprendizaje en diferentes campos. En la Figura 1.13 se puede apreciar una aplicación real reciente de la robótica de bajo coste de la universidad *Carnegie Mellon* descrita en el artículo [Shaw and Pathak, 2024].

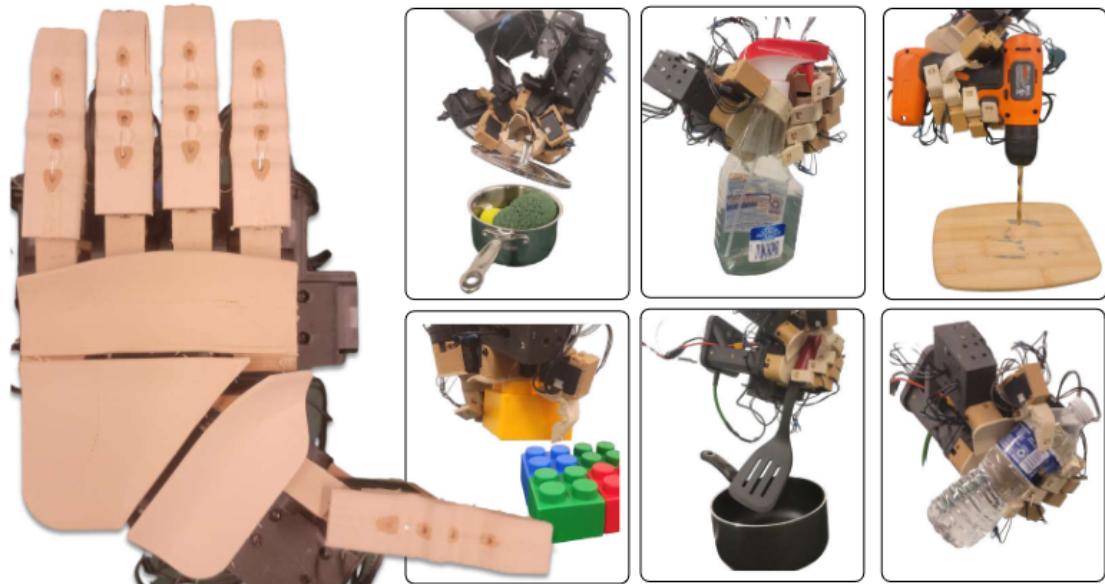


Figura 1.13: Mano Antropomórfica Híbrida Rígida y Suave

Sabiendo que existen los robots de campo y la robótica de bajo coste, es necesario explorar y conocer las instituciones, los tipos de irregularidades en la carretera y los procesos actuales que existen en España para poder tener un correcto mantenimiento de ellos.

1.4. Conservación de carreteras en España

La conservación de las carreteras es un aspecto fundamental para garantizar la seguridad vial, la eficiencia del transporte y la durabilidad de las infraestructuras viarias. En España, donde la red de carreteras juega un papel crucial en la conectividad

y el desarrollo económico, el mantenimiento adecuado de estas infraestructuras es vital para asegurar un tránsito seguro y fluido, así como para prevenir accidentes y reducir los costos derivados del deterioro del pavimento.

España cuenta con una extensa red de carreteras que incluye autopistas, autovías, carreteras nacionales, autonómicas, y provinciales. Cada una de estas vías requiere un enfoque específico en cuanto a su mantenimiento y conservación, dado que las condiciones de uso y los desafíos asociados varían considerablemente.

La conservación de las carreteras en el país está regulada y gestionada por diversas instituciones y organismos. La Dirección General de Carreteras¹⁶, bajo la gestión del *Ministerio de Transportes, Movilidad y Agenda Urbana* (MITMA), es la entidad responsable de la Red de Carreteras del Estado, que incluye las vías de mayor envergadura y tráfico. Por su parte, las comunidades autónomas y las diputaciones provinciales gestionan las carreteras que pertenecen a sus respectivas jurisdicciones, como es el caso de las carreteras autonómicas de Castilla-La Mancha¹⁷ y las carreteras provinciales de Toledo¹⁸.

Además, varias organizaciones juegan un papel destacado en la promoción, estudio y mejora continua de las infraestructuras viarias. El *Centro de Estudios y Experimentación de Obras Públicas* (CEDEX)¹⁹, por ejemplo, es un organismo de referencia en la investigación y experimentación en el ámbito de las obras públicas y la movilidad. Fundado en 1957, el CEDEX trabaja para mejorar y conservar las infraestructuras, impulsar una movilidad segura y sostenible, y proteger el medioambiente.

Otra entidad relevante es la *Asociación Española de la Carretera* (AEC)²⁰, una entidad sin ánimo de lucro fundada en 1949, que trabaja en la defensa y promoción de las carreteras. La AEC se enfoca en aspectos como la seguridad vial, la sostenibilidad y la calidad de las infraestructuras, adaptando sus actividades a las necesidades y desafíos contemporáneos, como la digitalización y la descarbonización del transporte.

En el ámbito de la conservación propiamente dicha, la *Asociación de Empresas de Conservación y Explotación de Infraestructuras* (ACEX)²¹, creada en 1995, agrupa a empresas dedicadas a la conservación de carreteras y se centra en promover la eficiencia

¹⁶<https://www.transportes.gob.es/carreteras/organizacion-y-funciones/secretaria-general-de-infraestructuras/direccion-general-de-carreteras>

¹⁷<https://www.castillalamancha.es/gobierno/fomento/estructura/dgfcartra/actuaciones/cat%C3%A1logo-y-mapa-de-carreteras-de-castilla-la-mancha>

¹⁸<http://eiel.diputoledo.es/visor/index.php>

¹⁹<https://www.cedex.es/presentacion>

²⁰<https://www.aecarretera.com/quienes-somos>

²¹<https://www.acex.eu/la-asociacion/>

y sostenibilidad en el mantenimiento de estas infraestructuras, así como en mejorar la seguridad vial y laboral. También es reseñable destacar sus premios anuales²² que permiten avanzar a pasos agigantados en materia de conservación y seguridad vial.

Finalmente, cabe destacar la labor de la Plataforma de Trabajadores de Conservación de Carreteras²³, un grupo independiente que agrupa a profesionales del sector preocupados por la seguridad laboral, la reducción de la siniestralidad en las actividades de conservación viaria y por mejorar sus condiciones laborales.

Tras conocer a todos los actores que conforman parte del mantenimiento correcto de las carreteras de España, es necesario conocer los tipos de deterioros del pavimento que se tienen que enfrentar.

1.5. Deterioros del pavimento

En el artículo [Llopis Castelló and Pérez Zuriaga, 2020] se presentan los distintos tipos de deterioros o daños en diferentes tipos de pavimentos, así como su rehabilitación y mantenimiento.

Para entender las distintas categorías de deterioros, es necesario conocer qué es el firme de una carretera y de qué capas está compuesto. El firme es un conjunto de varias capas de materiales seleccionados y, en la mayoría de los casos, tratados, que forman la superestructura de la plataforma. Su propósito es soportar las cargas del tráfico y garantizar que la circulación se realice de manera segura y cómoda. (Figura 1.14)

El firme se compone de tres capas: rodadura, capa intermedia y capa base. Las dos primeras forman el pavimento, que soporta las cargas del tráfico y proporciona características como resistencia al deslizamiento y una superficie regular. La capa base, en cambio, tiene una función estructural al absorber las presiones transmitidas y proteger la subbase de cargas excesivas.

Los firmes se dividen en cuatro tipos: flexibles, semiflexibles, semirrígidos y rígidos. Los firmes flexibles y semiflexibles tienen una capa bituminosa sobre capas granulares. Si el espesor de la capa bituminosa es menor de 15 cm, el firme es flexible; si es mayor, es semiflexible. Los firmes semirrígidos cuentan con un pavimento bituminoso sobre capas tratadas con conglomerantes hidráulicos, con un espesor mínimo de 20 cm. Finalmente, los firmes rígidos tienen un pavimento de hormigón sobre una capa de zahorras.

²²<https://www.acex.eu/premios-acex/>

²³https:////conservacion.es/index.php?option=com_content&view=article&id=1&Itemid=101

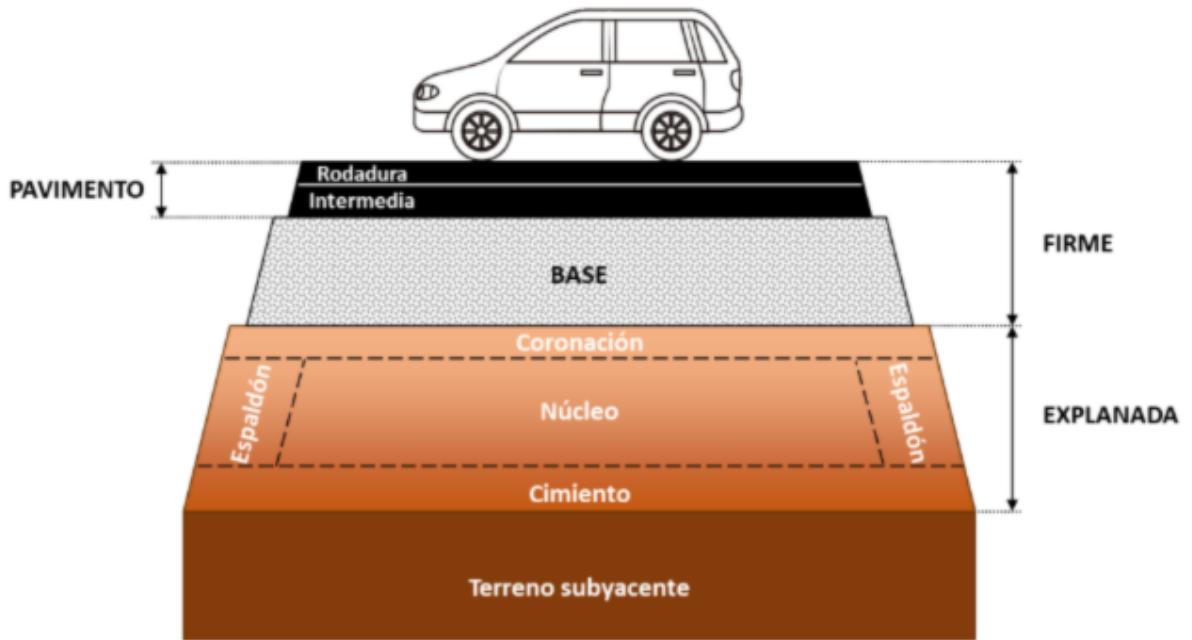


Figura 1.14: Firme de una carretera

Tipos de deterioros

Los distintos tipos de deterioros o daños más comunes que se pueden presentar en pavimentos flexibles, semiflexibles y semirrígidos urbanos los podemos clasificar en cuatro grandes categorías: *agrietamiento* (Figura 1.16), *degradación del material de la capa de rodadura* (Figura 1.17), *degradación de la capa de rodadura sin degradación de material* (Figura 1.18) y *otro tipo de daños* (Figura 1.19). Todo está esquematizado en la Figura 1.15.

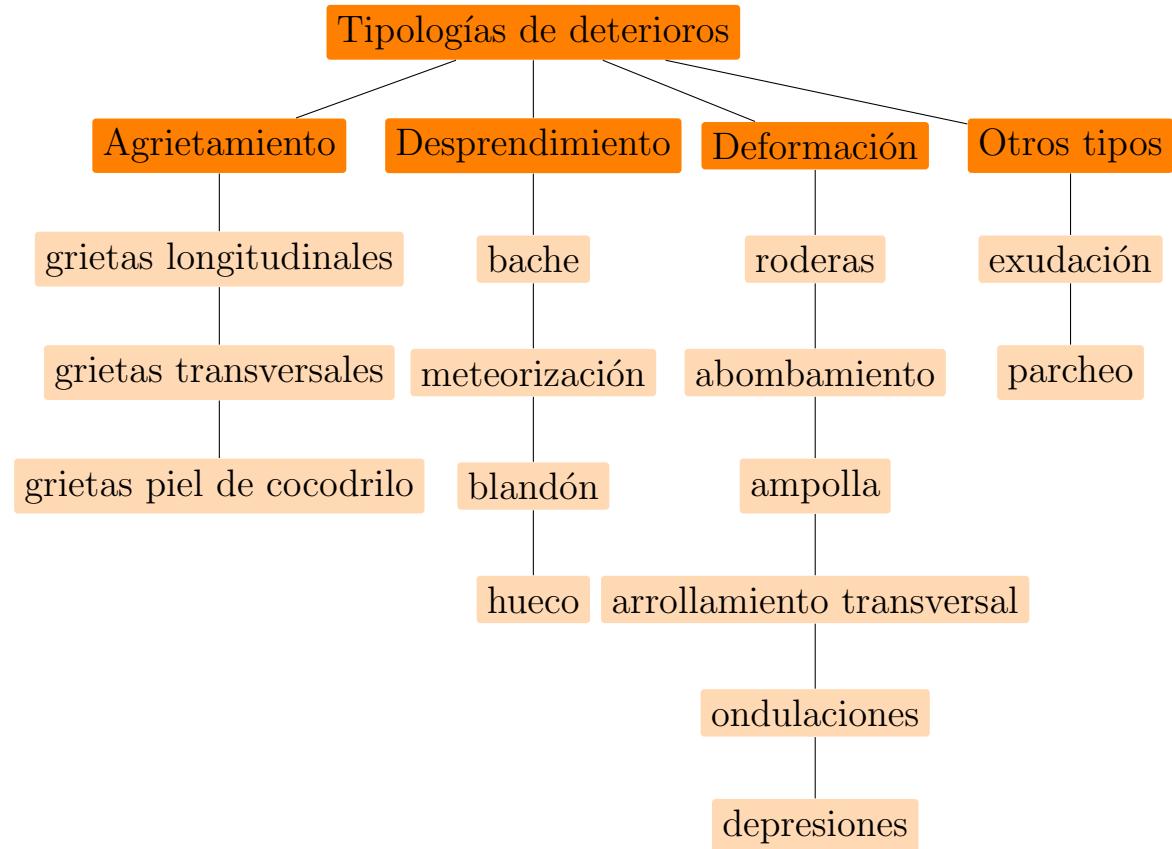


Figura 1.15: Clasificación de deterioros



Grietas longitudinales

Grietas transversales

Grietas piel de cocodrilo

Figura 1.16: Agrietamiento

Tras haber mostrado toda la gran cantidad de tipos de deterioros que existen, es necesario centrarnos en ser capaces del mantenimiento de uno de ellos ya que el problema es muy grande y sería inabordable para este proyecto. De todos los tipos mostrados, se van a tratar los *baches*.

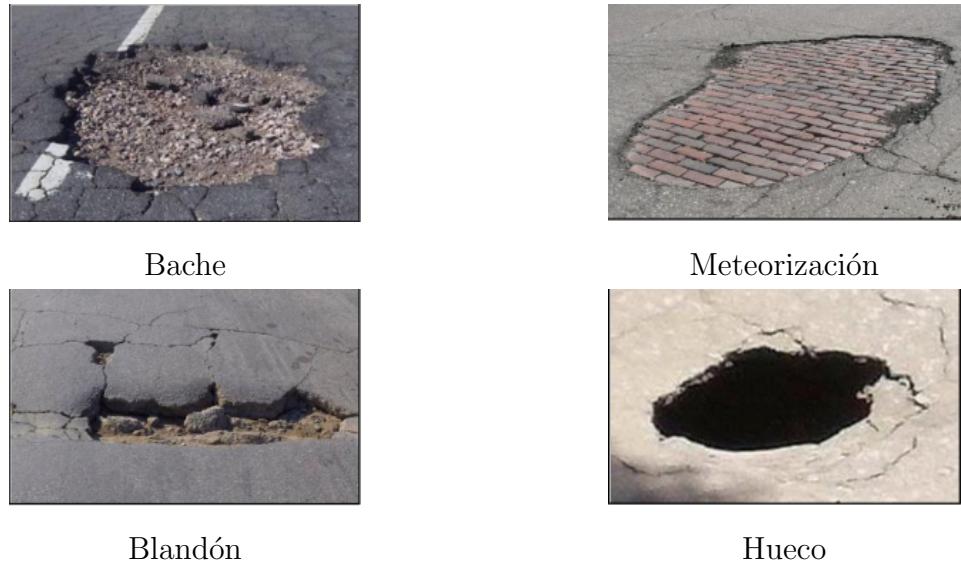


Figura 1.17: Degradación del material de la capa de rodadura



Figura 1.18: Deformación de la capa de rodadura sin degradación de material



Figura 1.19: Otro tipo de daños

Mantenimiento de baches

La reparación de baches en los pavimentos de asfalto es un proceso importante para mantener las carreteras en buen estado. Un bache es una depresión en la superficie de la carretera que puede ser causada por una variedad de factores, como el tráfico pesado, el clima extremo y el envejecimiento del pavimento. Si se dejan sin reparar, los baches pueden causar daños a los vehículos y crear un peligro para los conductores.

El proceso de reparación de baches comienza con la eliminación del material dañado. Esto se hace mediante el corte del área afectada con una sierra de asfalto o una fresadora. Una vez que se ha eliminado el material dañado, se limpia la superficie hasta dejar el hueco con forma de cuadrilátero regular y se aplica una capa de asfalto fresco. El asfalto se compacta y se alisa para que se ajuste perfectamente al pavimento existente. El resultado final es una superficie de carretera lisa y sin baches²⁴. En la Figura 1.20 se puede ver a dos operarios arreglando un bache con asfalto en frío.



Figura 1.20: Operarios reparando un bache ²⁵

Este proyecto se centra en el desarrollo de un robot de campo de tamaño compacto y bajo costo, diseñado para ser fácil de usar y controlar. El robot ha sido fabricado completamente mediante impresión 3D y está equipado con herramientas ampliamente utilizadas en el campo de la robótica. Esta combinación permite que cualquier persona, incluso sin un conocimiento profundo en robótica, pueda replicar el robot y ponerlo en funcionamiento. En el próximo capítulo, se presentarán diversos prototipos y futuras aplicaciones que guardan una cierta relación con el tipo de robot desarrollado.

²⁴<https://www.youtube.com/watch?v=YPx1bIywgzo>

²⁵<https://fixer.es/blog/como-reparar-bache-con-asfalto-en-frío/>

Capítulo 2

Estado del arte

Todo progreso depende de la irracionalidad del hombre razonable.

George Bernard Shaw

En el presente capítulo se van a describir algunos de los prototipos y posibles aplicaciones más destacables sobre la detección y mantenimiento del pavimento de las carreteras usando inteligencia artificial y técnicas robóticas. En este estado del arte, se revisan cuatro soluciones tecnológicas innovadoras desarrolladas recientemente: el sistema robotizado basado en *Raspberry Pi* para la reconstrucción 3D de baches (Infrarob), el Proyecto HERON para el mantenimiento robótico de carreteras, la plataforma EyesNroad para la detección de deterioros viales mediante *Inteligencia Artificial* (IA), y el proyecto OMICRON para la automatización del sellado de grietas.

Sistema robotizado basado en Raspberry Pi para la reconstrucción 3D de baches

Este sistema explicado en [Bruno et al., 2023], es creado como parte del proyecto Infrarob²⁶ y desarrollado en el marco del proyecto europeo Horizon 2020, utiliza una *Raspberry Pi 4B* acoplada a un robot autónomo para capturar imágenes usando una cámara y coordenadas GPS, usando el módulo de baches en carreteras. Las imágenes, tomadas desde diferentes ángulos, son procesadas mediante técnicas fotogramétricas para generar modelos 3D de los baches, permitiendo calcular con precisión el volumen que debe ser rellenado. El sistema también integra un *Sistema de Información Geográfica* (SIG) para mejorar la gestión del mantenimiento de pavimentos. (Figura 2.1)

Las ventajas de este proyecto son las siguientes:

²⁶<https://infrarobproject.com/>



Figura 2.1: Robot usado para la reconstrucción 3D de baches

- *Bajo costo.* Utiliza componentes de bajo costo como: *Raspberry pi*, *Raspberry pi camera* y el módulo *GPS NEO-6M* que también es *low-cost*.
- *Precisión en la detección.* Capacidad para detectar baches de hasta 75 cm de diámetro y crear modelos 3D precisos.
- *Integración con SIG.* Facilita la planificación de reparaciones.

Las desventajas de este proyecto son las siguientes:

- *Limitaciones climáticas.* No puede operar en condiciones adversas ni durante la noche.
- *Uso de software de pago.* Utiliza *ContextCapture*, que es de pago, para el procesamiento fotogramétrico.
- *Movimiento circular para detección del volumen.* Una vez detectado el bache, es

necesario moverse alrededor de él para que el algoritmo de fotogrametría sea capaz de encontrar su volumen, lo que ralentiza el cálculo.

El proyecto Herón

En [Katsamenis et al., 2022] se habla del Proyecto HERON²⁷, también parte del programa Horizon 2020, propone una solución integral para el mantenimiento de infraestructuras viales mediante el uso de vehículos modulares robóticos autónomos y drones. El sistema está diseñado para optimizar la eficiencia y seguridad de las operaciones de mantenimiento, utilizando sensores y escáneres para crear mapas 3D, inteligencia artificial para coordinar flujos de trabajo, y módulos de análisis de imágenes para detectar defectos. Todavía está en etapa de desarrollo. (Figura 2.2)

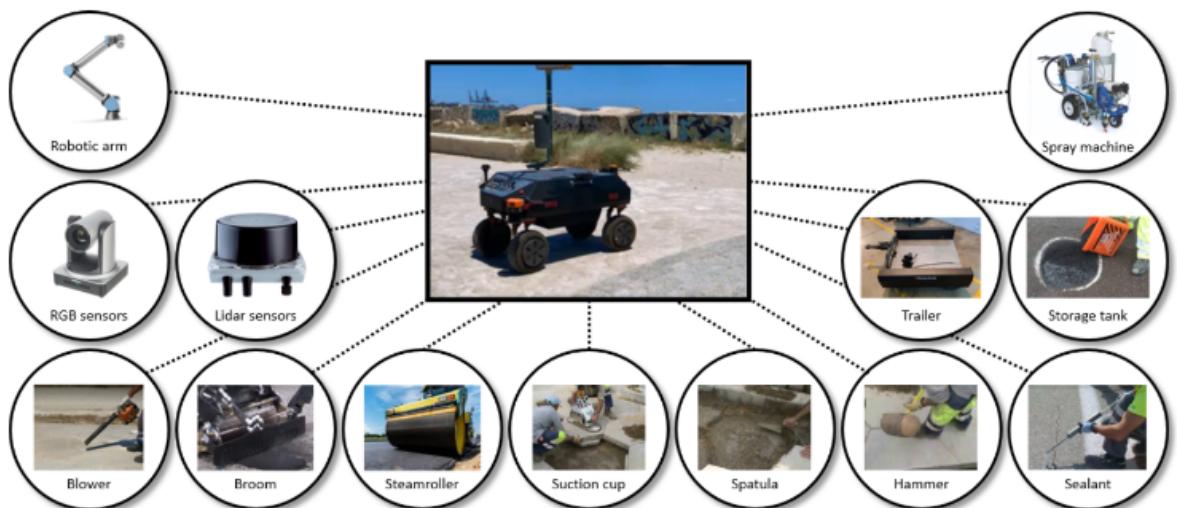


Figura 2.2: Proyecto HERON

Las ventajas de este proyecto son las siguientes:

- *Automatización completa.* Minimiza la intervención humana y reduce riesgos laborales.
- *Eficiencia.* El objetivo es mejorar la capacidad y eficiencia de las redes viales.
- *Intercambio de datos en tiempo real.* Otro de los objetivos es mejorar la toma de decisiones
- *Diseño modular.* Para poder maximizar sus capacidades, facilitar el transporte y reducir los costos de mantenimiento y accidentes.

²⁷<https://www.heron-h2020.eu/>

Las desventajas de este proyecto son las siguientes:

- *Costo inicial elevado.* La implementación de tecnologías avanzadas es costosa.
- *Dependencia tecnológica.* Requiere una infraestructura tecnológica robusta para su funcionamiento.

EyesNroad

EyesNroad es una plataforma que emplea IA y *Machine Learning* para identificar baches, señalización y otros deterioros en carreteras a partir de videos capturados por cámaras georreferenciadas instaladas en vehículos. La plataforma genera inventarios detallados y reportes del estado de las carreteras, facilitando el control y mantenimiento de las mismas. Esta aplicación ha sido desarrollada para participar en los XX premios de conservación de carreteras de ACEX (año 2024)²⁸ que ha resultado ganadora en la en la categoría general. Puedes ver su aspecto en la Figura 2.3.

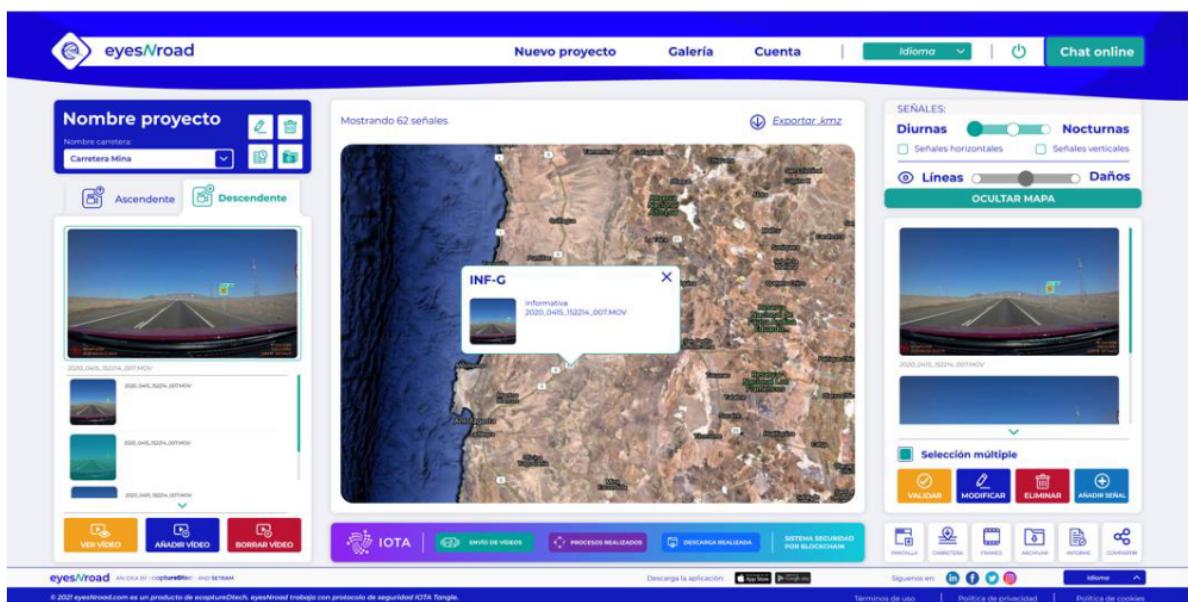


Figura 2.3: Visualización de la señal en un mapa en eyesNroad

Las ventajas de este proyecto son las siguientes:

- *Facilidad de implementación.* Puede utilizarse con cámaras comunes y es compatible con dispositivos móviles.
- *Amplia detección.* No solo identifica baches, sino también señalización y otros elementos viales.

²⁸<https://www.acex.eu/candidatura-3-4/>

- *Actualización continua.* La base de datos se amplía constantemente para cubrir más regiones.

Las desventajas de este proyecto son las siguientes:

- *Limitación geográfica.* Actualmente, su funcionamiento completo está limitado a Portugal.
- *Es de pago.* Para cualquier persona que desee usar esta herramienta tiene que asumir los costes.
- *No calcula volumen.* La plataforma no ofrece estimaciones del volumen de los baches, lo que podría ser útil para reparaciones.

El proyecto OMICRÓN

Dentro del proyecto OMICRON se ha decidido crear un sistema robótico que busca automatizar y robotizar el proceso de sellado de grietas en pavimentos, con el objetivo de eliminar la exposición de los operarios a condiciones peligrosas. El sistema integra un brazo robótico y herramientas de inteligencia artificial para detectar grietas y realizar el sellado de manera autónoma, mejorando así la seguridad y eficiencia en las operaciones de mantenimiento. Esta aplicación ha sido desarrollada para participar en los XX premios de conservación de carreteras de ACEX (año 2024)²⁹ que ha resultado ganadora en la categoría asociados. Puedes ver su aspecto en la Figura 2.4.

Las ventajas de este proyecto son las siguientes:

- *Seguridad laboral.* Elimina la necesidad de que los operarios sufren accidentes de tráfico y tengan contacto con materiales peligrosos.
- *Reducción de tiempo.* Debido a la automatización del proceso, se puede reducir el tiempo de intervención.
- *Innovación en materiales.* Al invertir en crear un mástico nuevo, se puede operar a temperaturas más bajas.

Las desventajas de este proyecto son las siguientes:

- *Es un prototipo.* Todavía está en fase temprana de implementación lo que le impide a los clientes tener una disponibilidad inmediata.

²⁹<https://www.acex.eu/candidatura-12-5/>



Figura 2.4: Plataforma robótica diseñada para el sellado de grietas

- *Alto costo.* Necesita componentes caros para poder llevar a cabo sus objetivos: un camión, un brazo robótico, entre otros.
- *Complejidad técnica.* Ya que se trata de una automatización completa, necesita un alto nivel de precisión y coordinación, supone un gran desafío para su implementación.

Los prototipos y desarrollos tecnológicos presentados demuestran el potencial de la robótica y la inteligencia artificial en el mantenimiento y detección de deterioros en el pavimento. Desde la creación de modelos 3D de baches hasta la automatización del sellado de grietas, estos avances no solo mejoran la seguridad y eficiencia de las operaciones, sino que también optimizan la gestión y planificación del mantenimiento vial. No obstante, cada solución presenta desafíos específicos, como la dependencia de condiciones climáticas, la limitación geográfica y los costos asociados, que deben ser considerados al evaluar su implementación en diferentes contextos. En el siguiente capítulo se va a proceder a definir una serie requisitos, metodología y un plan de trabajo que se ha seguido para dar solución al problema que se expone a lo largo de esta memoria.

Capítulo 3

Objetivos

Establecer metas es el primer paso para convertir lo invisible en visible.

Tony Robbins

Tras haber establecido el marco contextual del presente proyecto, se procederá a presentar la descripción del problema, los requisitos, las competencias tanto adquiridas como empleadas, la metodología y el plan de trabajo seguido.

3.1. Descripción del problema

La idea del diseño del robot para este trabajo de fin de grado nace tras los acontecimientos vividos por la *Depresión Aislada en Niveles Altos* (DANA) que afectó el pasado septiembre de 2023 en la zona de Toledo y Madrid. Estos hechos dejaron inundaciones, pueblos anegados, carreteras cortadas, muchos vecinos perdieron sus casas y desgraciadamente se cobró la vida de 3 personas.

Esta situación se produjo, en parte, por la falta de mantenimiento de los pavimentos y carreteras y que posteriormente a lo sucedido, también fue una de las infraestructuras que más tardaron en arreglarse. La principal causa de que eso ocurriese es los bajos recursos y la falta de mano de obra que justamente hay en la zona que sucedieron los hechos. Los operarios que conforman esta mano de obra llevan tiempo exigiendo mejoras de seguridad en su entorno laboral como defiende la Plataforma de Trabajadores de Conservación de Carreteras³⁰ mencionada en el Capítulo 1.

La solución propuesta busca ayudar a mejorar esta situación, proporcionando un robot de bajo coste y accesible para cualquier persona y que sirva para poder mejorar el mantenimiento de las carreteras y reducir el riesgo de exposición de los operarios. Por lo tanto, este proyecto pretende como objetivo principal crear un robot que usando

³⁰https://conservacion.es/index.php?option=com_content&view=article&id=1&Itemid=101

materiales de bajo coste sea capaz de navegar por las carreteras, detectar los baches que vaya encontrando y calcular el área necesario de material que necesitará el bache para ser tapado. De igual manera, todo quedará registrado en una interfaz web en la que cada bache quedará marcado sobre un mapa con su correspondiente descripción para que los operarios puedan operar cuando estimen oportuno.

Con el fin de solventar el problema definido previamente, se han establecido los siguientes subobjetivos:

1. Investigar los robots o soluciones actuales que cumplan con las características y objetivos establecidos.
2. Seleccionar los componentes hardware de bajo coste necesarios para construir el esqueleto del robot.
3. Analizar las diferentes opciones de diseño que más encajen con la forma del robot.
4. Diseñar las piezas en CAD usando herramientas de *software* libre.
5. Usar material típico de impresión 3D para imprimir las partes del robot, como puede ser ABS o PLA.
6. Desarrollar un modelo del robot para que pueda ser usado en simulación usando herramientas robóticas.
7. Desarrollar software necesario usando herramientas robóticas para poder controlar el robot físico.
8. Poder realizar algunos experimentos en entorno reales o adaptados.

3.2. Requisitos

Tras nombrar los objetivos y subjetivos necesarios para cumplir en este proyecto, se deberá cumplir también con los siguientes requisitos:

1. El coste total de la fabricación del robot no debe superar los 220€.
2. Todas las piezas diseñadas deben poderse imprimir en cualquiera impresora convencional.
3. Se usará Ubuntu como sistema operativo para tanto el ordenador como el robot.

4. A fin de facilitar la implementación de este proyecto para cualquier tipo de usuario, no será necesario disponer de ninguna tarjeta gráfica para entrenar los modelos.
5. Los modelos entrenados se deben ajustar a las limitaciones hardware del robot.
6. Se busca que sea un proyecto a largo plazo y por eso se debe realizar la integración con la plataforma ROS2.

3.3. Competencias

Las competencias son aquellas habilidades, conocimientos y aptitudes que una persona emplea y adquiere para la realización, en este caso, del presente proyecto aunque es aplicable para cualquier tarea.

3.3.1. Competencias empleadas

Las competencias empleadas para la realización del presente proyecto fin de grado, que han sido tomadas de las distintas asignaturas del grado, son las siguientes:

1. Evolución y futuro de la robótica: *CE1*. Capacidad para analizar la evolución de la Ingeniería Robótica y ser capaz de identificar sus aplicaciones, oportunidades de emprendimiento y su impacto en el futuro. Esta competencia ha sido empleada para poder desarrollar los capítulos 1 y 2 de este proyecto.
2. Laboratorio de sistemas: *CE9*. Capacidad de conocer y manejar los sistemas y las herramientas de las que dispone para su gestión y programación. Esta competencia ha sido empleada para poder configurar y ejecutar código del robot en entornos no gráficos.
3. Sensores y actuadores: *CE12*. Capacidad de diseñar robots y sistemas inteligentes atendiendo a los elementos de sensorización y actuación más adecuados dependiendo de la aplicación, los requerimientos del sistema y las condiciones del entorno. Esta competencia ha sido empleada para poder encontrar los componentes hardware necesarios para poder llevar a cabo el esqueleto del robot.
4. Arquitectura *software* para robots: *CE15*. Capacidad de diseñar y programar aplicaciones robóticas y sistemas inteligentes en red usando *middlewares*, mecanismos de comunicación y estándares propios del ámbito de la Ingeniería Robótica. Esta competencia ha sido empleada en el momento de decidir el tipo de arquitectura *software* necesaria a implementar en el robot.

5. Visión artificial: *CE25*. Capacidad de conocer y aplicar métodos de extracción de información a partir de la información percibida por cámaras y sensores 3D al desarrollo de aplicaciones en robots y sistemas inteligentes. Esta competencia ha sido empleada para poder extraer información de la cámara y ser capaz de tratarla.
6. Mecatrónica: *CE32*. Capacidad de diseñar y construir robots móviles. Esta competencia ha sido empleada en el diseño e impresión en 3D de mi robot.
7. Aprendizaje automático: *CE27*. Capacidad de construir sistemas capaces de resolver problemas a partir de información no estructurada proporcionada por ejemplos o por la experiencia. Esta competencia ha sido empleada para poder ser capaz de crear modelos de aprendizaje automático y aplicarlos al robot.

3.3.2. Competencias adquiridas

Las competencias adquiridas a lo largo del desarrollo del presente proyecto fin de grado que aparecen descritas en la guía docente de la propia asignatura, son las siguientes:

1. *CB2*. Que los estudiantes sepan aplicar sus conocimientos a su trabajo o vocación de una forma profesional y posean las competencias que suelen demostrarse por medio de la elaboración y defensa de argumentos y la resolución de problemas dentro de su área de estudio. Esta competencia se adquiere gracias a la aplicación de las competencias empleadas justificadas anteriormente y que se pueden ver plasmadas en todo el proyecto.
2. *CB4*. Que los estudiantes puedan transmitir información, ideas, problemas y soluciones a un público tanto especializado como no especializado. Esta competencia se adquiere al describir de forma precisa y comprensible todo el proceso complejo implicado en este proyecto dentro del presente documento.
3. *CB5*. Que los estudiantes hayan desarrollado aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores con un alto grado de autonomía. Esta competencia se logra al adquirir el conocimiento suficiente para desarrollar este trabajo de forma totalmente autónoma, contrastando información con distintas fuentes, haciendo pruebas con distintos tipos de *software*, entre otras.
4. *CE28*. Desarrollo de las capacidades adecuadas para realizar un ejercicio original individual (o excepcionalmente colectivo), presentarlo y defenderlo ante un

tribunal universitario, consistente en un proyecto en el ámbito de las tecnologías específicas del campo de la Robótica de naturaleza profesional en el que se sinteticen e integren las competencias adquiridas en las enseñanzas. Esta última competencia se cumple con el desarrollo de este proyecto: que abarca desde la elección del tema, el conocer el estado del arte, la implementación tanto *hardware* como *software*, el desarrollo de la presente memoria hasta su defensa ante un tribunal.

3.4. Metodología

Para llevar a cabo este proyecto final de grado se ha optado por seguir una metodología que se iniciaba en una profunda investigación sobre el estado del arte para comprobar la viabilidad del desarrollo del presente proyecto. Posteriormente tras haber elegido el *hardware* necesario para el robot, se decidió usar una metodología experimental que ayudó a decidir el diseño final que tendría el robot. Una vez cumplido esto, se imprimió y se ensambló las distintas piezas.

Para el desarrollo *software* en el robot real se probó y configuró cada sensor y actuador del robot en distintos sistemas operativos y versiones hasta encontrar la mejor que cumpliese todos los objetivos. Una vez conseguido eso, se decidió seguir un ciclo de desarrollo conocido como *Plan Do Check Act* (PDCA) y así poder ir haciendo pequeños avances consistentes hasta llegar a una versión completamente funcional. Para el desarrollo del robot en simulación también se decidió usar la metodología PDCA. Esta metodología sigue los siguientes pasos cíclicos:

1. *Planear*. Ante el problema presentado, se decide qué hacer.
2. *Hacer*. Tras conocer qué hacer, se pone en marcha lo planeado previamente.
3. *Comprobar*. Tras conocer los resultados obtenidos, se comprueban.
4. *Actuar*. Tras comprobar los resultados, se ajusta lo que tiene que ser corregido.

3.5. Plan de trabajo

El desarrollo del presente proyecto fin de grado ha estado dividido en las siguientes etapas:

1. *Investigación del estado del arte*. En esta fase inicial, se realizaron búsquedas en plataformas online como Google Scholar y otras relacionadas con el

mantenimiento de carreteras con el fin de encontrar soluciones al problema descrito.

2. *Planteamiento hardware del robot.* Una vez conocida la viabilidad del proyecto, se decidió estudiar qué componentes *low-cost* eran necesarios para dar forma al esqueleto del robot.
3. *Diseño de prototipos del robot.* Tras conocer cuál será el esqueleto del robot, usando cartón y pegamento se hicieron una serie de prototipos hasta encontrar la solución final. Posteriormente se usó la herramienta FreeCAD para modelar las distintas piezas.
4. *Impresión 3D y ensamblaje de las piezas.* Una vez el diseño estaba hecho, se decidió imprimirlo usando filamento de tipo PLA azul. Finalmente, se produjo el ensamblaje de todas las piezas.
5. *Desarrollo del robot en simulación.* Tras tener el robot completamente montado, se decidió desarrollar el modelo del robot pero esta vez para que se pudiera trabajar con él en simulación, usando Gazebo.
6. *Configuración hardware del robot.* Tras tener al robot listo en simulación, se decidió configurar cada componente hardware del robot en distintos sistemas operativos hasta encontrar el que mejor encajase con la arquitectura del mismo.
7. *Desarrollo software del robot en físico.* El robot finalmente está configurado y listo para operar en un entorno real. Asimismo se desarrolló una serie de nodos en ROS2 que mostraban un correcto funcionamiento de cada componente hardware siguiendo el propósito buscado.
8. *Experimentos en un entorno real.* En esta etapa final, se realizaron distintos experimentos en el entorno real para demostrar que se cumplía el objetivo principal.

Asimismo, a lo largo de todo el proceso se ha ido elaborando la presente memoria. La dinámica seguida con el tutor ha sido de reuniones semanales o cada dos semanas, dependiendo de la disponibilidad por las dos partes. Además en dichas reuniones se comentaban todos los avances realizados y se proponían aspectos a mejorar, sugerencias y se definían nuevos objetivos a conseguir.

Todo el contenido del proyecto está alojado en un repositorio público de GitHub³¹. Además, todo el trabajo diario está documentado en el apartado Wiki³² de dicho repositorio; dividido en *diario* y en *evolución del proyecto*. El apartado de *diario* trata de contar de forma coloquial qué se ha ido realizando cada día o cada pocos días. Por otro lado, en *evolución del proyecto* se puede encontrar pasos de instalación seguidos, explicación detallada de códigos o de conceptos teóricos, entre otros detalles.

³¹<https://github.com/RoboticsURJC/tfg-jlopez>

³²<https://github.com/RoboticsURJC/tfg-jlopez/wiki>

Capítulo 4

Plataforma de desarrollo

*Las herramientas adecuadas en las manos adecuadas
pueden cambiar el mundo*

Steve Jobs

Tras haber establecido los objetivos que se pretenden alcanzar en este proyecto, en este capítulo se van a tratar las distintas plataformas de desarrollo tanto *hardware* como *software* que han contribuido para lograr dichos objetivos.

4.1. Hardware

En este apartado se van a describir el conjunto de componentes hardware que han sido necesarios adquirir para llevar a cabo este proyecto siempre primando por el menor coste de cada componente.

4.1.1. Raspberry Pi 4

La Raspberry Pi es una computadora de bajo coste y con un tamaño compacto que es ideal para proyectos de electrónica, programación y educación. Esta, en su cuarta versión, dispone de un procesador ARM Cortex-A72 de cuatro núcleos a 1,50GHz fabricado en 28nm y con 3 configuraciones de memoria. Además, entre otras características, ahora solo necesita una alimentación de USB-C, tiene 2 conectores micro HDMI, tiene conexión Wifi, Bluetooth 5.0, 2 USB 2.0 y 2 USB 3.0. Esta versión de Raspberry permite la compatibilidad con la mayoría de accesorios gracias al conector GPIO de 40 pines y el conector *Camera Serial Interface* (CSI). Gracias a esos 40 pines, el usuario puede interactuar con una amplia variedad de dispositivos externos como sensores, LEDs y motores, lo que lo hace excelente para proyectos de automatización y robótica.

Sin embargo, la Raspberry Pi usa procesadores ARM, que, aunque eficientes energéticamente, no están optimizados para realizar tareas intensivas de IA como entrenar modelos de redes neuronales o ejecutar inferencias en tiempo real a gran escala. En la figura 4.1 se puede ver una imagen de la placa con referencia a las distintas partes de la misma. Esta pieza fue prestada por la universidad pero tiene un coste de unos 65€.

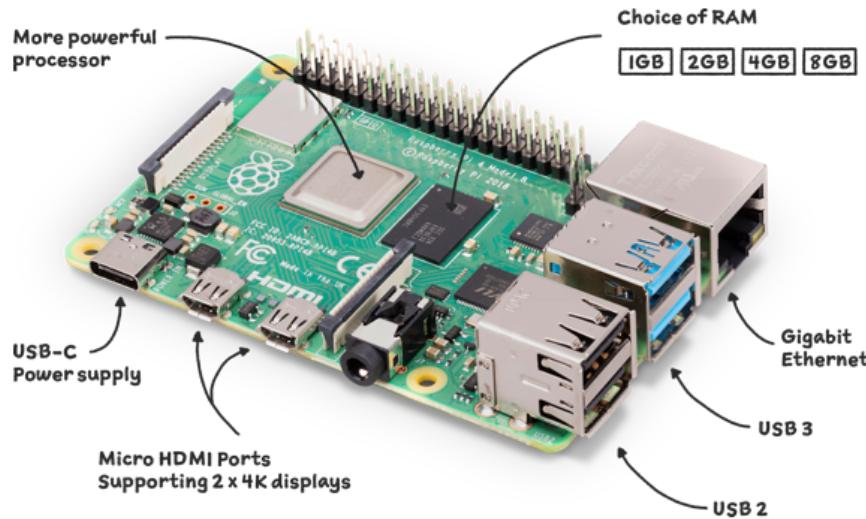


Figura 4.1: Raspberry Pi 4³³

4.1.2. Raspberry Pi cámara

Esta placa que tiene un tamaño de 23.86 x 25 x 9mm (Figura 4.2), utiliza el sensor de imagen IMX219PQ de Sony que ofrece imágenes de vídeo de alta velocidad y alta sensibilidad. Dispone también de funciones de control automático como el control de exposición, el balance de blancos y la detección de luminancia. Esta cámara usa un cable plano que se conecta directamente al puerto CSI.

Es muy importante destacar su bajo coste, lo que le convierte en muy buena opción para hacer aplicaciones *low-cost*. Sin embargo, el cable de la cámara es bastante sensible a tensiones y torsiones y es debido a ello que aunque la cámara fue prestada por la universidad, fue necesario adquirir una unidad más a través de Amazon con un coste aproximado de 18€.

³³<https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>

³⁴<https://www.raspberrypi.com/products/camera-module-v2/>



Figura 4.2: Raspberry Pi Cámara V2³⁴

4.1.3. GPS NEO 6M

Para poder identificar dónde se encuentra cada bache es necesario hacer uso de un posicionamiento global mediante satélites. En este caso se decidió comprar el módulo NEO 6M (Figura 4.3) cuyo asequible precio en Amazon es de 9€. El módulo utiliza el chipset u-blox NEO-6M, que proporciona un seguimiento preciso de la posición utilizando satélites GPS, permitiendo determinar la latitud, longitud, altitud, y velocidad. Además, incluye una antena cerámica externa de alta ganancia (puede ser interna en algunas versiones), que mejora la recepción de la señal GPS, incluso en áreas con una señal débil.

Tiene una precisión de aproximadamente 2.5 metros en condiciones abiertas. Soporta hasta 22 satélites simultáneamente, lo que le permite ofrecer posicionamiento confiable y estable. Asimismo, utiliza comunicación por *Universal Asynchronous Receiver-Transmitter* (UART) comúnmente a 9600 bps, lo que facilita su integración con muchas placas del mercado.



Figura 4.3: Módulo GPS NEO 6M³⁵

³⁵<https://www.amazon.es/dp/B088LR3488?ref>

4.1.4. Sevomotor Estándar Parallax

Este tipo de servo tiene un rango de rotación de 0 a 180 grados y es compatible su control usando *Pulse Width Modulation* (PWM) con un pulso alto de 0.75–2.25 ms en intervalos de 20 ms (Figura 4.4 izquierda). Estos tipos de servos son muy comunes para utilizar en aplicaciones de animatrónica y robótica. Sin embargo, su coste ha subido en los últimos años y el precio por unidad ronda los 17€. La universidad ha sido quien me ha prestado estos servomotores pero para esta aplicación no es necesario usarlos y se pueden sustituir por otros de menor precio que cumplan con el requisito de tener rotaciones de 0 a 180 grados (Figura 4.4 derecha), cuyo precio de 2 servomotores es de casi 8€.

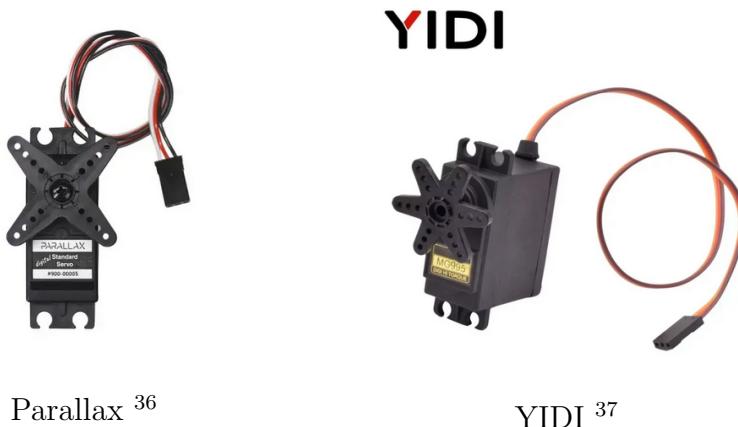
Parallax³⁶YIDI³⁷

Figura 4.4: Tipos de Servomotores

4.1.5. Ruedas ActivityBot

Para implementar las ruedas de este robot, se decidió tomar las ruedas del kit robótico ActivityBot que usan este tipo de ruedas. Se trata de una rueda de plástico con neumático tipo junta tórica (Figura 4.5). El perfil estrecho convierte a esta rueda en ideal para aplicaciones que requieren una dirección precisa y el diámetro de la rueda es de 66mm. La universidad ha sido quien me ha prestado su uso pero también se pueden adquirir individualmente con un coste de 4,54€ la unidad.

³⁶<https://www.parallax.com/product/parallax-standard-servo/>

³⁷https://es.aliexpress.com/item/1005004551539283.html?spm=a2g0o.productlist.main.15.bee4rd8Srd8SS0&algo_pvid

³⁸https://es.rs-online.com/web/p/accesorios-para-kits-de-desarrollo/8430897?srsltid=AfmB0rv3a6_tQNdqAoKx_21Mn1m2MAum68oApyvr5mq8ExPTuh_CVNy



Figura 4.5: Rueda ActivityBot³⁸

4.1.6. Google Coral USB

El Google Coral USB Accelerator (Figura 4.6) es un dispositivo que permite acelerar tareas de IA en dispositivos que no cuentan con hardware especializado para ello. Está diseñado para ejecutarse con modelos de aprendizaje automático utilizando la unidad de procesamiento de tensores de Google, lo que mejora significativamente la velocidad de inferencia en aplicaciones de IA. Está optimizado para ejecutar modelos preentrenados en TensorFlow Lite, la versión ligera de TensorFlow diseñado para dispositivos con recursos limitados. Sólo es compatible con las versiones desde Python 3.6 hasta la versión 3.9.

Gracias a todas esas características y para mejorar la detección de baches en tiempo real y solventar las limitaciones de la Raspberry Pi, se ha decidido incluir este componente en el proyecto. Su precio es elevado pero se puede conseguir desde 65€.



Figura 4.6: Google Coral USB³⁹

³⁹<https://coral.ai/products/accelerator/>

4.1.7. Power Bank

Para poder alimentar a todos los componentes se ha decidido incluir en el proyecto una *power bank* de bastante capacidad para que permita la autonomía del robot durante mucho tiempo. Para ello se ha elegido la Xiaomi Redmi Power Bank de 20000 mAh cuyas dimensiones son: 73,6 x 27,3 x 154 mm. Cuando la adquirí costaba en torno a unos 20€ pero ahora se puede encontrar hasta por el doble de precio.



Figura 4.7: Xiaomi Powerbank⁴⁰

4.1.8. Rueda Loca

Para poder conseguir un movimiento correcto y poder encontrar el punto de apoyo para el robot, es necesario incluir al robot una rueda loca. Tras investigar para encontrar la que mejor encaja, se ha decidido incluir una rueda loca como la que aparece en la Figura 4.8 que tiene las siguientes dimensiones: 5,3 x 2,9 x 2 cm. El precio de una de ellas sale por 1,13€.



Figura 4.8: Rueda loca⁴¹

⁴⁰<https://buy.mi.com/es/item/3202200053>

4.1.9. Ordenador principal

Para poder desarrollar programas, hacer pruebas en simulación, permitir conectarse por SSH a la Raspberry Pi y poder comandar acciones al robot; ha sido necesario tener un ordenador principal que permita realizar todas las tareas. El ordenador que aparece en la Figura 4.9 es el que se ha empleado en este proyecto y cumple las características descritas en el Cuadro 4.1.



Figura 4.9: ASUS VivoBook 14⁴²

Características	Descripción
Pantalla	14 pulgadas, Full HD (1920x1080), tecnología LED, antirreflejo
Procesador (CPU)	Intel Core i7 de 10 ^a generación
Memoria RAM	8GB
Almacenamiento	512GB
Tarjeta gráfica (GPU)	Intel UHD Graphics de la serie Comet Lake-U GT2
Sistema Operativo	Windows 10 y Ubuntu 22.04
Puertos 1	1x USB 3.2 Gen 1 Tipo-A, 1x USB 3.2 Gen 1 Tipo-C, 2x USB 2.0
Puertos 2	1x HDMI, lector de tarjetas microSD, entrada combo de audio
Conectividad	Wi-Fi 5 (802.11ac), Bluetooth 4.1 / 5.0
Batería	37 Whr
Peso	1.4 kg
Dimensiones	32.5 x 21.6 x 1.99 cm

Cuadro 4.1: Especificaciones técnicas del ordenador usado

⁴¹https://www.amazon.es/dp/B0BZZCJJT8?ref=cm_sw_r_mwn_dp_N64JV6SGENWN4YPSD849&ref_=cm_sw_r_mwn_dp_N64JV6SGENWN4YPSD849&social_share=cm_sw_r_mwn_dp_N64JV6SGENWN4YPSD849&language=es-ES

⁴²<https://www.asus.com/es/laptops/for-home/vivobook/vivobook-14-k413/>

4.2. Software

En este apartado se van a describir el conjunto de programas y librerías que han sido necesarios usar cumplir con los objetivos del Capítulo 3.

4.2.1. FreeCAD

Esta herramienta de código abierto (Figura 4.10) es un *software* de diseño asistido por computadora utilizado principalmente para la creación de modelos 3D en diferentes áreas como la ingeniería, arquitectura, y diseño de productos. Está diseñado para ser altamente modular formado por *workbenches*, lo que permite a los usuarios adaptar y extender su funcionalidad según sus necesidades.

FreeCAD se basa en el concepto de modelado paramétrico, lo que permite modificar el diseño fácilmente. Los usuarios pueden retroceder en la historia de un modelo y cambiar parámetros que actualizan automáticamente el diseño. También FreeCAD es multiplataforma, cuenta con una consola de Python integrada y soporta una amplia gama de formatos de archivo como STL y SVG entre otros. Esta ha sido la herramienta elegida para hacer el diseño 3D de la pieza y la generación del formato STL para su posterior impresión.



Figura 4.10: Logo de Freecad ⁴³

4.2.2. Python

Python (Figura 4.11) es un lenguaje de programación de alto nivel, interpretado y de propósito general, ampliamente reconocido por su simplicidad y legibilidad. Fue creado por Guido van Rossum y lanzado por primera vez en 1991, y ha crecido rápidamente en popularidad debido a su versatilidad y facilidad de uso tanto para principiantes como para programadores experimentados lo que le convierte en un lenguaje multiplataforma y multiparadigma.

⁴³<https://www.freecad.org/>

Python es un lenguaje interpretado, lo que significa que el código se ejecuta línea por línea sin necesidad de ser compilado. Esto facilita el desarrollo rápido y la depuración. Tiene una sintaxis sencilla, legible y es ampliamente utilizado en áreas como: desarrollo web (Django y Flask), ciencia de datos y aprendizaje automático (NumPy, Pandas, TensorFlow, y PyTorch), automatización y scripting, entre otros.

Debido a esa simplicidad y versatilidad en Raspberry Pi, se considera Python como uno de los lenguajes de programación oficiales recomendados y es por ello que se ha decidido usar este lenguaje para este proyecto.



Figura 4.11: Logo de Python ⁴⁴

Software matemático

NumPy (Numerical Python) es una biblioteca fundamental para el cálculo numérico en Python. Está diseñada para facilitar el manejo eficiente de vectores, grandes matrices y arrays multidimensionales, junto con una amplia colección de funciones matemáticas para realizar operaciones sobre estos arrays. Es tal su poder que se ha dedicado usar en este proyecto para poder calcular el área del bache.

Software localización

PyNMEA2 es una biblioteca de Python que se utiliza para analizar y generar mensajes en formato NMEA (National Marine Electronics Association), un estándar utilizado en dispositivos GPS y otros sistemas de navegación marina. Es especialmente útil cuando trabajas con módulos GPS en proyectos de Raspberry Pi u otros dispositivos embebidos, ya que te permite interpretar la información que estos dispositivos envían, como la localización, velocidad y tiempo.

PyNMEA2 te permite interpretar las sentencias NMEA, que son los datos en bruto que los módulos GPS envían, usualmente en forma de texto. Algunas sentencias

⁴⁴<https://es.python.org/>

comunes son:

\$GPGGA: proporciona datos como la latitud, longitud, y altitud.

\$GPRMC: contiene información esencial de ubicación, velocidad y tiempo.

\$GPGLL: latitud y longitud.

Es este proyecto, se van a emplear las sentencias de \$GPGGA para estimar la ubicación de cada bache.

4.2.3. OpenCV

OpenCV (Figura 4.12) es una biblioteca de software de código abierto diseñada para la visión artificial y el procesamiento de imágenes. Fue desarrollada inicialmente por Intel en 1999 y ahora es mantenida por una gran comunidad de desarrolladores. OpenCV es ampliamente utilizada en aplicaciones que requieren análisis de imágenes, detección de objetos, reconocimiento de rostros, visión computacional, y mucho más. Gracias a su gran aplicación se ha decidido integrar en el TFG para la detección del bache y el cálculo del área.



Figura 4.12: Logo de OpenCV ⁴⁵

4.2.4. Ubuntu

Ubuntu (Figura 4.13) es una distribución GNU/Linux basada en Debian GNU/Linux, que incluye principalmente software libre y de código abierto y es mantenida por Canonical Ltd. Puede utilizarse en ordenadores y servidores. Está orientado al usuario promedio, con un fuerte enfoque en la facilidad de uso y en mejorar la experiencia del usuario. Debido a ser software libre es uno de los sistemas operativos más populares y es conocido por su facilidad de uso, estabilidad, amplia comunidad de soporte y por sus actualizaciones periódicas.

⁴⁵<https://opencv.org/>

⁴⁶<https://ubuntu.com/>



Figura 4.13: Logo de Ubuntu ⁴⁶

Entre todas las versiones existentes para la realización del proyecto se ha usado Ubuntu 22.04 y Ubuntu 20.04. El ordenador usado (descrito en el Apartado 4.1.9) tiene instalado Ubuntu 22.04 *Long Time Support* (LTS) pero el robot inicialmente usaba Ubuntu 22.04 y finalmente está usando Ubuntu 20.04 LTS Server, una versión de Ubuntu sin interfaz gráfica. La decisión de usar Ubuntu 20.04 es debido a que el dispositivo Google Coral USB descrito en el Apartado 4.1.6 es compatible con versiones de Python entre 3.6 y 3.9 siendo en Ubuntu 22.04 la instalación de Python por defecto de 3.10 y el intento de cambiar dicha versión generó numerosos problemas de dependencias sobre todo con la versión de ROS2 usada. En muchos foros tampoco aconsejaban modificar las versiones de Python que vienen instaladas por defecto⁴⁷ y que la Rasberry únicamente tuviese esta aplicación se recomienda migrar a Ubuntu 20.04⁴⁸ que usa Python 3.8. A continuación, se va a mostrar un Cuadro 4.2 con algunas de las diferencias entre Ubuntu 22.04 y Ubuntu 20.04.

Características	Ubuntu 20.04	Ubuntu 22.04
Fecha de lanzamiento	Abril 2020	Abril 2022
Soporte	Hasta Abril 2025	Hasta Abril 2027
Kernel	Linux 5.4	Linux 5.15
Entorno de escritorio	GNOME 3.36	GNOME 42
Python	Python 3.8	Python 3.10
Soporte gráfico NVIDIA	Soporte estándar	Soporte mejorado con Wayland

Cuadro 4.2: Diferencias entre Ubuntu 20.04 y Ubuntu 22.04

4.2.5. ROS 2

Por sus siglas en inglés, *Robot Operating System* (ROS) se trata de un *middleware* para programar robots. Un *middleware* es una capa de software entre el sistema

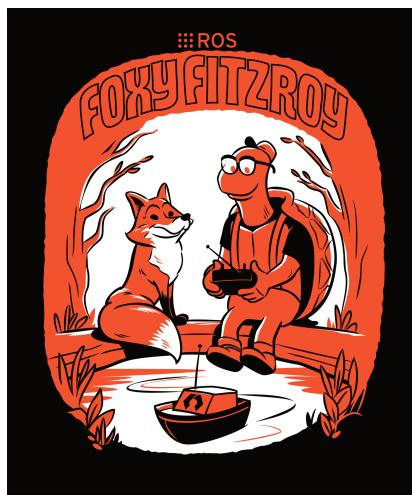
⁴⁷<https://github.com/google-coral/pycoral/issues/81>

⁴⁸<https://robotics.stackexchange.com/questions/104413/can-an-ros2-node-run-in-a-venv-and-use-a-different-python-than-that-used-by-the>

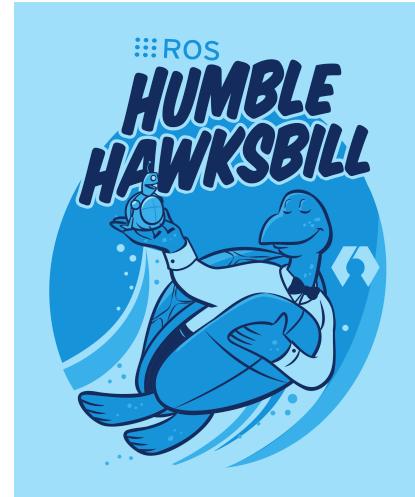
operativo y las aplicaciones de usuario que permiten llevar a cabo aplicaciones *software* independiente del dominio que ese encuentre. Es decir, ROS aporta un conjunto de herramientas, bibliotecas y convenciones. También, ofrecen desarrollo, integración, ejecución y herramienta de monitorización. El número 2 indica que es la segunda generación de este middleware.

ROS 2 tiene tres diferentes dimensiones: la comunidad de ROS 2 que contribuye al desarrollo de aplicaciones es enorme, el grafo de computación de una aplicación en ROS 2 está formado por nodos conectados con distintos paradigmas de comunicación y el espacio de trabajo, también conocido como *workspace*, es aquel que se trata del *software* instalado que se divide a su vez en *underlay* (la instalación básica de ROS 2) y *overlay* (son el resto de programas que se desarrollan).

Según se ha explicado en el Apartado 4.2.4, debido a que he necesitado para este proyecto usar Ubuntu 20.04 y Ubuntu 22.04; he tenido que usar las dos distribuciones más estable de ROS para cada versión de Ubuntu: ROS 2 Foxy y ROS 2 Humble respectivamente (Figura 4.14).



Ros2 Foxy ⁴⁹



Logo Ros2 Humble ⁵⁰

Figura 4.14: Distribuciones de ROS 2 usadas

ROS 2 Control

ROS 2 Control⁵¹ es un framework dentro de ROS 2 diseñado para facilitar el control de robots en tiempo real. Proporciona una infraestructura modular y escalable para manejar controladores que gestionan los actuadores (motores, servomotores, etc.) de

⁴⁹<https://docs.ros.org/en/foxy/Installation.html>

⁵⁰<https://docs.ros.org/en/humble/index.html>

⁵¹https://control.ros.org/rolling/doc/getting_started/getting_started.html

un robot, así como la lectura de sensores. Se utiliza comúnmente en robots móviles, brazos robóticos, drones y otras plataformas robóticas.

Los componentes principales de ROS 2 Control son:

- *Controller Manager*. Gestiona controladores e interfaces de hardware en el framework ROS 2 Control.
- *Resource Manager*. Abstacta y gestiona el hardware, permitiendo la reutilización flexible de componentes.
- *Controllers*. Utilizan la teoría de control para interactuar con el hardware.
- *User interface*. Pueden interactuar con el sistema mediante servicios y una interfaz de línea de comandos.
- *Hardware Components*. Realizan comunicación con hardware físico como puede ser: sistemas, sensores y actuadores.

En este proyecto se ha decidido usar ROS 2 Control para crear un modelo del robot en simulación y poder trabajar con él usando la filosofía de ROS 2 Control.

4.2.6. Gazebo

Gazebo (Figura 4.15) es un simulador de robots que permite modelar entornos físicos tridimensionales y probar robots en ellos sin necesidad de tener el robot en físico y evitar así posibles caídas o golpes. Es ampliamente utilizado en la investigación y desarrollo de robótica, especialmente en combinación con ROS y ROS 2. Además, ofrece una simulación realista en un entorno 3D y por ello se decidió usar este simulador para el proyecto.



Figura 4.15: Logo de Gazebo ⁵²

⁵²<https://gazebosim.org/home>

4.2.7. Herramientas de monitorización

Rviz

RViz (Figura 4.16) es una herramienta de visualización en 3D utilizada en ROS y ROS 2 para representar información del sistema robótico. Permite a los usuarios ver datos en tiempo real como la posición de un robot, sensores, cámaras, mapas y otros elementos. Se ha utilizado para monitorear el comportamiento del robot en simulación.

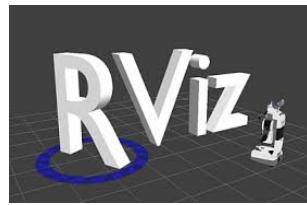


Figura 4.16: Logo de Rviz ⁵³

RQT Image View

RQt Image View ⁵⁴ es una herramienta gráfica dentro del ecosistema de ROS y ROS 2 que permite visualizar imágenes en tiempo real de un flujo de datos de imágenes publicado por una cámara en un sistema robótico. Se ha usado esta herramienta para monitorizar la cámara del robot físico.

Ros2cli

ROS 2 *Command Line Interface* (CLI)⁵⁵ es una herramienta que permite interactuar con el sistema ROS 2 mediante comandos desde la terminal. Ofrece una forma rápida y sencilla de acceder a las funciones y características de ROS 2 sin necesidad de escribir o ejecutar código complejo. Ha sido la herramienta principal en este proyecto para ejecutar los distintos nodos, comprobar si estaban bien lanzados, si se producía buena comunicación entre ellos, entre otras aplicaciones.

4.2.8. Google Colaboratory

Google Colaboratory (Figura 4.17) un servicio en la nube proporcionado por Google que permite escribir y ejecutar código en Python a través de un entorno de Jupyter

⁵³<http://wiki.ros.org/rviz>

⁵⁴http://wiki.ros.org/rqt_image_view

⁵⁵<https://docs.ros.org/en/humble/Tutorials/Beginner-CLI-Tools.html>

Notebook. No requiere configuración para su uso y proporciona acceso gratuito a recursos de computación, incluidos GPUs y TPUs. Es especialmente usado para el aprendizaje automático, la ciencia de datos y la educación. Gracias a esta herramienta se puede cumplir del apartado 3.2 de Requisitos, el número 4.



Figura 4.17: Logo de Google Colab ⁵⁶

4.2.9. YOLOv8

YOLOv8 (Figura 4.18) desarrollado por la empresa Ultralytics, es una versión avanzada del popular modelo de detección de objetos *You Only Look Once* (YOLO), diseñado para identificar y localizar objetos en imágenes y videos en tiempo real. Esta versión es conocida por su eficiencia y precisión en la detección de objetos. Es compatible con TensorFlow y PyTorch. También ofrece optimización para diferentes plataformas incluidos como EdgeTPU para dispositivos con recursos limitados como puede ser Raspberry Pi. Debido a todo lo anterior, esta herramienta ha sido elegida para la el entrenamiento del modelo de detección de baches.



Figura 4.18: Logo de YOLOv8 ⁵⁷

⁵⁶<https://colab.research.google.com/>

⁵⁷<https://docs.ultralytics.com/es>

4.2.10. TensorFlow Lite

TensorFlow es una plataforma de código abierto desarrollada por Google para el aprendizaje automático y el desarrollo de redes neuronales. Su diseño permite a los desarrolladores e investigadores crear, entrenar y desplegar modelos de aprendizaje automático de manera eficiente en una variedad de dispositivos, desde ordenadores de alto rendimiento hasta dispositivos con recursos limitados como puede ser Raspberry Pi. Para poder ejecutar modelos en dispositivos una Raspberry se usa TensorFlow Lite (Figura 4.19) ya que está preparada para optimizar los modelos. Es por ello que cualquier modelo que se quiera usar se tiene que convertir a este formato.



Figura 4.19: Logo de TensorFlow Lite ⁵⁸

4.2.11. Interfaz Web

Para poder hacer más amigable la interacción humano-robot, se ha decidido plasmar los datos obtenidos a través de una interfaz web y para ello se ha decidido usar las siguientes herramientas:

ROS2bridge Server

ROS2bridge Server⁵⁹ es parte de ros2bridge_suite y ofrece una capa de transporte WebSocket para la comunicación bidireccional entre páginas web y ROS 2. Convierte mensajes JSON en llamadas a ROS 2 y viceversa, permitiendo que las páginas web interactúen con ROS 2.

OpenStreetMaps

OpenStreetMaps (Figura 4.20) un proyecto internacional colaborativo desde 2004 que proporciona datos geográficos gratuitos y abiertos, permitiendo la creación de

⁵⁸<https://ai.google.dev/edge/lite/>

⁵⁹https://wiki.ros.org/rosbridge_server

mapas detallados y editables de cualquier parte del mundo por usuarios voluntarios que den crédito a OpenStreetMaps. OpenStreetMaps es utilizado en una amplia gama de aplicaciones, incluyendo navegación, análisis de datos geográficos, y proyectos de código abierto relacionados con mapas y geolocalización.



Figura 4.20: Logo de Open Street Maps⁶⁰

Tras conocer todas las plataformas de desarrollo para realizar el presente trabajo fin de grado, es el momento de contar el desarrollo completo llevado a cabo para la construcción hardware del robot que se explicará detalladamente en el siguiente capítulo.

⁶⁰<https://www.openstreetmap.org>

Capítulo 5

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.

5.1. Snippets

Puede resultar interesante, para clarificar la descripción, mostrar fragmentos de código (o *snippets*) ilustrativos. En el Código 5.1 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 5.1: Función para buscar elementos 3D en la imagen

En el Código 5.2 vemos un ejemplo escrito en Python.

5.2. 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 5.2: 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
}

```

5.3. 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 5.1 y Ecuación 5.2.

$$H = 1 - \frac{\sum_{i=0}^N \left(\frac{d_{js} + d_{je}}{2} \right)}{M} \quad (5.1)$$

Ecuación 5.1: Ejemplo de ecuación con fracciones

$$v(\text{entrada}) = \begin{cases} 0 & \text{if } \epsilon_t < 0,1 \\ K_p \cdot (T_t - T) & \text{if } 0,1 \leq \epsilon_t < M_t \\ K_p \cdot M_t & \text{if } M_t < \epsilon_t \end{cases} \quad (5.2)$$

Ecuación 5.2: Ejemplo de ecuación con array y letras y símbolos especiales

5.4. Tablas o cuadros

Si necesitas insertar una tabla, hazlo dignamente usando las propias tablas de L^AT_EX, no usando pantallazos e insertándolas como figuras... En el Cuadro 5.1 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 5.1: Parámetros intrínsecos de la cámara

En los textos puedes poner palabras en *cursiva*, para aquellas expresiones en sentido *figurado*, palabras como *robota*, que está fuera del diccionario castellano, o bien para resaltar palabras de una colección: *(a)* es la primera letra del abecedario, *(b)* es la segunda, etc.

Al poner las dos líneas del anterior párrafo, este aparecerá separado del anterior. Si no las pongo, los párrafos aparecerán pegados. Sigue el criterio que consideres más oportuno.

5.5. Segunda sección

No olvides incluir imágenes y referenciarlas, como la Figura 5.1.



Figura 5.1: Robot aspirador Roomba de iRobot.

Ni tampoco olvides de poner las URLs como notas al pie. Por ejemplo, si hablo de la Robocup¹.

¹<http://www.robocup.org>

5.5.1. Números

En lugar de tener secciones interminables, como la Sección 1.1, divídelas en subsecciones.

Para hablar de números, mételos en el entorno *math* de L^AT_EX, por ejemplo, $1,5Kg$. También puedes usar el símbolo del Euro como aquí: 1.500€ .

5.5.2. Listas

Cuando describas una colección, usa `itemize` para ítems o `enumerate` para enumerados. Por ejemplo:

- *Entorno de simulación.* Hemos usado dos entornos de simulación: uno en 3D y otro en 2D.
 - *Entornos reales.* Dentro del campus, hemos realizado experimentos en Biblioteca y en el edificio de Gestión.
1. Primer elemento de la colección.
 2. Segundo elemento de la colección.

Referencias bibliográficas Cita, sobre todo en este capítulo, referencias bibliográficas que respalden tu argumento. Para citarlas basta con poner la instrucción `\cite` con el identificador de la cita. Por ejemplo: libros como [Vega et al., 2012], artículos como [Vega and Cañas, 2019], URLs como [Vega, 2019], tesis como [Vega, 2018a], congresos como [Vega, 2018b], u otros trabajos fin de grado como [Vega, 2008].

Las referencias, con todo su contenido, están recogidas en el fichero `bibliografia.bib`. El contenido de estas referencias está en formato BibTex. Este formato se puede obtener en muchas ocasiones directamente, desde plataformas como `Google Scholar` u otros repositorios de recursos científicos.

Existen numerosos estilos para reflejar una referencia bibliográfica. El estilo establecido por defecto en este documento es APA, que es uno de los estilos más comunes, pero lo puedes modificar en el archivo `memoria.tex`; concretamente, cambiando el campo `apalike` a otro en la instrucción `\bibliographystyle{apalike}`.

Y, para terminar este capítulo, resume brevemente qué vas a contar en los siguientes.

Capítulo 6

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.

6.1. 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).

6.2. Corrector ortográfico

Una vez tengas todo, no olvides pasar el corrector ortográfico de L^AT_EXa 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
```

Bibliografía

[Barceló, 2004] Barceló, M. (2004). De nuevo los robots. *Paradojas*, (64):1–2.

[Bidaud,] Bidaud, P. Les robots et les hommes.

[Bruno et al., 2023] Bruno, S., Loprencipe, G., Di Mascio, P., Cantisani, G., Fiore, N., Polidori, C., D'Andrea, A., and Moretti, L. (2023). A robotized raspberry-based system for pothole 3d reconstruction and mapping. *Sensors*, 23(13).

[Earnest, 2012] Earnest, L. (2012). The stanford cart. Consultado el 19 de julio de 2024.

[El Zaatri et al., 2019] El Zaatri, S., Marei, M., Li, W., and Usman, Z. (2019). Cobot programming for collaborative industrial tasks: An overview. *Robotics and Autonomous Systems*, 116:162–180.

[Katsamenis et al., 2022] Katsamenis, I., Bimpas, M., Protopapadakis, E., Zafeiropoulos, C., Kalogerias, D., Doulamis, A., Doulamis, N., Martín-Portugués Montoliu, C., Handanos, Y., Schmidt, F., Ott, L., Cantero, M., and Lopez, R. (2022). Robotic maintenance of road infrastructures: The heron project. In *Proceedings of the 15th International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '22, page 628–635, New York, NY, USA. Association for Computing Machinery.

[Llopis Castelló and Pérez Zuriaga, 2020] Llopis Castelló, D. and Pérez Zuriaga, A. M. (2020). Deterioros en pavimentos urbanos.

[Nilsson et al., 1984] Nilsson, N. J. et al. (1984). *Shakey the robot*, volume 323. Sri International Menlo Park, California.

[Plaza, 2023] Plaza, J. M. C. (2022-2023). Robótica de servicio: Robots de limpieza. Consultado el 19 de julio de 2024.

- [Shaw and Pathak, 2024] Shaw, K. and Pathak, D. (2024). LEAP hand v2: Dexterous, low-cost anthropomorphic hybrid rigid soft hand for robot learning. In *2nd Workshop on Dexterous Manipulation: Design, Perception and Control (RSS)*.
- [Thorpe and Durrant-Whyte, 2003] Thorpe, C. and Durrant-Whyte, H. (2003). Field robots. In *Robotics Research: The Tenth International Symposium*, pages 329–340. Springer.
- [Vega, 2008] Vega, J. (2008). Navegación y autolocalización de un robot guía de visitantes. Master thesis on computer science, Rey Juan Carlos University.
- [Vega, 2015] Vega, J. (2015). De la tiza al robot. Technical report.
- [Vega, 2018a] Vega, J. (2018a). *Educational framework using robots with vision for constructivist teaching Robotics to pre-university students*. Doctoral thesis on computer science and artificial intelligence, University of Alicante.
- [Vega, 2018b] Vega, J. (2018b). JdeRobot-Kids framework for teaching robotics and vision algorithms. In *II jornada de investigación doctoral*. University of Alicante.
- [Vega, 2019] Vega, J. (2019). El profesor Julio Vega, finalista del concurso 'Ciencia en Acción 2019'. URJC, on-line newspaper interview.
- [Vega and Cañas, 2019] Vega, J. and Cañas, J. (2019). PyBoKids: An innovative python-based educational framework using real and simulated Arduino robots. *Electronics*, 8:899–915.
- [Vega et al., 2012] Vega, J., Perdices, E., and Cañas, J. (2012). *Attentive visual memory for robot localization*, pages 408–438. IGI Global, USA. Text not available. This book is protected by copyright.