Facultad
de
Ciencias

Diseño de un vehículo inteligente mediante aprendizaje por observación
(Design of an intelligent vehicle through learning from observation)

Trabajo de Fin de Grado
para acceder al

GRADO EN INGENIERÍA INFORMÁTICA

Autor: Zoraida Mediavilla Sansegundo

Director: Cristina Tîrnăucă

Co-Director: José Luis Montaña

Septiembre - 2016

Índice

1. Introducción **3**

2. Antecedentes **5**
2.1. Google Car 5
2.2. ALVINN 6

3. Fase de diseño del prototipo **8**

4. Construcción del modelo **10**
4.1. Ventajas de la construcción del prototipo frente a la simulación . . 10
4.2. Rasberrry Pi 10
4.3. Arduino 13
4.4. Conceptos físicos necesarios 14
4.4.1. Circuitos 15
4.4.2. Elementos básicos 15
4.5. Componentes utilizados 17
4.5.1. Sensor ultrasónico 17
4.5.2. Motores 18
4.5.3. Controlador 19
4.5.4. Cámara 20
4.5.5. Alimentación 20
4.5.6. Chasis y ensamblado 21
4.5.7. Extras 21
4.6. Componentes descartados 22
4.6.1. Kinect 22

5. Software del prototipo **23**
5.1. Sistema Operativo 23
5.2. Lenguajes de programación 23
5.3. Controlador del robot 24
5.4. OpenCV 25
5.5. Almacenado de datos 25
5.6. Problemas 26

6. Inteligencia Artificial **27**
6.1. Aprendizaje por observación 27
6.2. Redes neuronales 27
6.3. Knime 30
6.4. PCA 30

6.5. Recolección de datos 31
6.6. Implementación neuronal realizada con Knime 31

7. Conclusiones **37**

Índice de figuras

1. Diagrama de los pesos de la red neuronal [9] 7
2. Chasis similar al utilizado (http://goo.gl/Twfj7N) 9
3. Conexiones Raspberry Pi (Adaptada de https://www.raspberrypi.org/) 11
4. Mapa de pines de las salidas GPIO (http://goo.gl/FvjUks) 13
5. Arduino Uno (http://arduino.cl/arduino-uno/) 14
6. Elementos de interconexión 16
7. Funcionamiento del sensor 17
8. Circuito del sensor 18
9. Tipos de motores utilizados en robótica 19
10. Circuito del controlador y los motores 20
11. Foto del prototipo 21
12. Circuito del led 22
13. Webiopi 24
14. Estructura de las neuronas [7] 27
15. Esquema de la red neuronal 29
16. Implementación 1 32
17. Implementación del bloque Red Neuronal 32
18. Implementación 2 33
19. Implementación del bloque Red Neuronal con todos los datos como entrenamiento 35

Índice de cuadros

1. Características de todos los modelos de Raspberry Pi 11
2. Resultados de la predicción del estado de los motores 33
3. Resultados de la predicción del movimiento 33
4. Resultados de la predicción del estado de los motores con retroalimentación del anterior 34
5. Resultados de la predicción del movimiento con retroalimentación del anterior 34
6. Resultados de la predicción del estado de los motores con todos los datos como entrenamiento 35
7. Resultados de la predicción del movimiento con todos los datos como entrenamiento 35

Resumen

En una sociedad en la que se exige cada vez tareas más complejas a los dispositivos electrónicos, llegando incluso a la conducción autónoma de vehículos, se necesitan nuevas formas de llevar estas tareas a cabo. Por ello, y con la intención de conseguir iguales o mejores resultados pero sin tanta preparación del posible entorno, este trabajo final de grado trata de comprobar si un vehículo autónomo puede aprender partiendo de acciones que ya ha realizado anteriormente.

Con el aprendizaje por observación se ahorraría la ardua tarea de preveer todo tipo de entornos y situaciones, para pasar a recoger los datos provenientes de la conducción de un piloto experto. Entre estos datos pueden encontrarse, como es en este caso concreto, imágenes tomadas con una cámara y distancias captadas con algún tipo de sensor como puede ser el ultrasónico. Obteniendo los suficientes datos de entrada para alimentar una red neuronal creada con un programa de minería de datos como puede ser el Knime, se espera conseguir unos buenos resultados de predicción de movimiento del robot.

En este documento se detalla desde la creación del robot en cuestión, construido utilizando una placa de bajo coste Raspberry Pi, hasta la explicación de las predicciones generadas mediante la red neuronal, e incluso algunos errores o problemas encontrados durante el proceso.

Palabras clave: Raspberry Pi, robot, sensor, red neuronal, Knime, vehículo autónomo, cámara, aprendizaje por observación.

Abstract

In a society where are demanded complex tasks by electronics devices, even to drive vehicles, new ways to perform these improvements are needed. For this reason, and with the intention to achieve equal or better results without much preparation of the possible environment, this final project is about to verify whether an autonomous vehicle can learn based on actions already undertaken previously.

With learning from observation would save the arduous task of mapping all kind of environments and situations, to saving data by the conduction of an expert pilot. These data can be found, as in this particularly case, images taken with a camera and distances captured with some type of sensor as can be the ultrasonic. Getting enough input data to feed a neural network, created with a data mining program such as Knime, it is expected to achieve good results of robot motion prediction.

In this document is detailed since the creation of the robot, made using a low-cost board Raspberry Pi, to the explanation of the predictions generated by the neural network, and even some errors or problems encountered along the way.

Key words: Raspberry Pi, robot, sensor, neural network, Knime, autonomous car, camera, learning from observation.

1. Introducción

Recientemente se ha visto incrementada la inclusión de la palabra *smart* o inteligente en todo tipo de dispositivos para indicar que ahora están dotados de "inteligencia". Muchas de las empresas desarrollan sus propios productos, a veces imitando a otro existente en el mercado y otras veces proponiendo algo novedoso. Relojes, ciudades, casas, coches e incluso electrodomésticos, todos incluyen ya este término, y no sólo las empresas están fabricando sus productos, sino que cada vez es más frecuente que gente en sus hogares fabrique sus propias máquinas, el llamado "movimiento *maker*", que sigue el lema de "hazlo tú mismo" o DIY (*Do it yourself* en inglés).

Esta tecnología se encuentra en parte desarrollada gracias a los grandes avances en hardware que se han producido desde hace unos años. Cada vez hay mayor número de dispositivos de bajo coste, de pequeño tamaño y muy potentes, que han facilitado su proliferación. Éste es el caso de la salida a mercado de electrónica como Arduino, Raspberry Pi, UDOO, Banana Pi, BeagleBone, ODroid, etcétera, que permiten montar un proyecto con las características deseadas con escaso coste económico. A pesar de que una gran parte de los usuarios utilizan algunas de placas para dotar a sus televisiones de mayores prestaciones (*smart tv*) o como servidor de páginas webs, este uso infravalora sus posibilidades, dado que son dispositivos que se pueden emplear para fines mucho más interesantes como por ejemplo, para la creación de un vehículo autónomo. Aunque no sea una tarea excesivamente compleja, es lo suficiente como para aprovechar las ventajas que estas placas ofrecen.

Estos avances no se producen sólo como consecuencia de las nuevas mejoras en hardware sino que llevan una gran cantidad de software que hace todo esto posible. Anteriormente, un dispositivo inteligente simplemente estaba dotado de la capacidad de conectarse fácilmente a Internet y de tener algunas aplicaciones sencillas, por lo general un navegador intuitivo (véase el caso de las PDAs ahora en desuso). Pero cada vez resuelven problemas más complicados y que requieren de procesadores con mayor potencia que trabajen a gran velocidad en cálculos complejos. Esto ha dado lugar a la utilización de la inteligencia artificial y otros recursos computacionales para solventar las cada vez más arduas tareas que se les piden.

Existen robots para todo tipo de tareas. Un ejemplo de lo más reciente es el robot Zenbo de Asus [3]. Se trata de un asistente para el hogar que se desplaza sobre dos ruedas y tiene una pantalla donde se muestra una cara muy amigable

que puede expresar incluso emociones, aunque también puede utilizarse para videollamadas o ver películas. Algunas de sus tareas son bastante simples o no tan importantes pero, principalmente, sirve para ayudar a gente de avanzada edad o entretener y vigilar a los niños. Y es que la tecnología no sólo sirve para el ocio; otro ejemplo es el de un robot policía en un centro comercial de Estados Unidos [2]; pero quizás es en el sector automovilístico donde se ha producido un mayor auge ya que pueden no sólo ofrecer mayor comodidad, sino también salvar vidas.

Grandes marcas de coches se encuentran desde hace unos años desarrollando sus propios vehículos dotados con la capacidad de conducir ellos mismos, interactuando con todo tipo de objetos (semáforos, otros coches, señales, ...) o personas (peatones). Compañías como Google, Toyota o Tesla están ahora probando sus nuevos modelos y cada vez es más frecuente la aparición de noticias sobre ellos. Concretamente es en este sector, el automovilístico, donde se va a centrar este proyecto, ya que fabricar un prototipo y programarlo, como se va a mostrar más adelante, es perfectamente factible.

2. Antecedentes

Un vehículo autónomo es un medio de locomoción capaz de decidir sus propios movimientos evitando obstáculos y obedeciendo ciertas reglas como pueden ser las de circulación. Se apoyan en diversas tecnologías para conseguir buenos resultados. Tanto es así que incluso se han llegado a desarrollar compañías especializadas que se dedican a fabricar hardware para dicho fin, como la empresa que hace el utensilio que es usado, por ejemplo por Google, para crear el mapa 3D del entorno del coche.

Las empresas no son tan reservadas en cuanto a la difusión de las características hardware como con el software. El dispositivo más común y completo es el LIDAR (acrónimo de *Light Detection and Ranging*) que es un instrumento que se instala en la mayoría de los modelos en la parte superior del coche y que es el encargado de medir distancias a los objetos, utilizando un sensor láser, para crear un mapa 3D del entorno. También suelen incorporar radares para calcular la velocidad de los objetos, GPS para tener la localización exacta, videocámaras para identificar las señales de tráfico, peatones u otros vehículos, acelerómetros y algún mecanismo para registrar el comportamiento de las ruedas. Además, algunos modelos han añadido la capacidad de comunicarse con otros vehículos que se encuentren relativamente cercanos a su posición.

En cuanto al tipo de inteligencia artificial que utilizan, las compañías son mucho más cautelosas con intención de no divulgar demasiada información a sus competidores. Entre algunas de las técnicas utilizadas se encuentran los algoritmos de localización, los algoritmos de planificación de ruta, el aprendizaje automático, la lógica probabilística y la visión artificial que se utiliza tanto en los mapas detallados en 3D como en el reconocimiento de señales, semáforos y otros elementos.

De entre toda la variedad que existe actualmente, este informe se centrará en lo siguiente por ser de gran relevancia.

2.1. Google Car

Google es una empresa puntera que aunque anteriormente destacaba en servicios de Internet y software, actualmente se encuentra en auge también en dispositivos electrónicos y diversa tecnología, entre la que se encuentran los vehículos autónomos. Desde 2009 están trabajando en su propio coche que incorpora la conducción autónoma como principal característica. Este proyecto está desarro-

llado en colaboración con el laboratorio de inteligencia artificial de la Universidad de Stanford, que en 2005 ganó el *DARPA Grand Challenge* con su robot "Stanley" que recorrió cerca de 229 Km sin la intervención humana por el desierto de Mojave. Hasta la fecha ha recorrido aproximadamente 2,41 millones de Km por diferentes carreteras de Estados Unidos.

Actualmente cuentan con dos modelos: un Lexus SUV modificado y un prototipo de creación propia. Ambos modelos incorporan, a parte de otras tecnologías mencionadas anteriormente, un LIDAR que es el que se encarga de crear la escena en 3D, *renderizando* los objetos en forma de cubos. También incluye un GPS y, aprovechando los mapas y otros servicios que ya posee la compañía es capaz de conducir el vehículo por el camino que le indiques.

El Google Car tiene la ventaja de que te llevará a tu destino correctamente por el camino más corto o con menos tráfico, en función de lo que se haya planificado previamente. No obstante, tiene el inconveniente de que no es posible sacarlo de su entorno programado, al contrario de otro tipo de vehículos utilizados en las fuerzas militares como el BigDog [1]. Otra desventaja es la capacidad de respuesta ante inclemencias del tiempo, es decir, están preparados para circular en lluvia y nieve pero si se presenta una tormenta han sido entrenados para esperar a que amaine.

2.2. ALVINN

ALVINN corresponde a las siglas de *Autonomous Land Vehicle In a Neural Network* y es un proyecto que se desarrolló en la Carnegie Mellon University en 1988. Es una base de referencia fundamental para este proyecto, dado que aprende partiendo de observar la conducción de un piloto y utiliza redes neuronales para dicho aprendizaje.

Para la toma de datos emplea una cámara y un telémetro láser que calcula la distancia hasta los objetos. Utiliza el canal azul de la imagen, debido a que es el que mejor contraste produce de la carretera, y las distancias proporcionadas por el telémetro como entrada de la red neuronal.

La red neuronal construida tiene una capa de entrada, una capa oculta y genera una salida de 46 nodos. La capa de entrada está constituida por la imagen de vídeo que es una matriz de 30x32 y el array generado por la cámara telemétrica que es de 8x32. En cuanto a la salida que genera la red neuronal, los 45 primeros nodos corresponden a lo que se espera sea la vía. Puede encontrarse

otro nodo al inicio tanto de la entrada como de la salida de la red, que correspondería a la retroalimentación de la intensidad de la carretera de la anterior entrada.

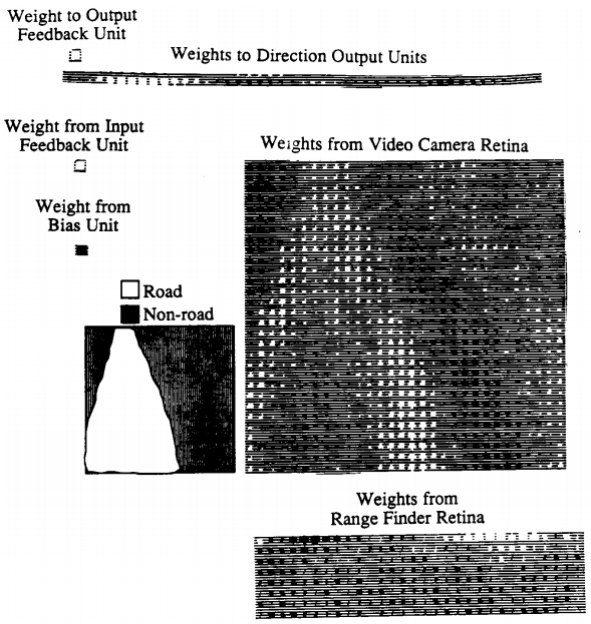


Figura 1: Diagrama de los pesos de la red neuronal [9]

Analizando la Figura 1 se puede ver que los píxeles blancos corresponden a la carretera, mientras que los negros pertenecen al resto del entorno. Dependiendo de donde estén orientados los píxeles blancos en la salida, se puede saber hacia donde tendrá que girar en el caso en que sea necesario. La carretera, en este caso concreto, se encuentra situada a la izquierda por lo que si no hay ningún obstáculo delante deberá girar en ese sentido para no salirse de ella.

Lo primero que hicieron para la parte del entrenamiento fue crear las imágenes con un simulador, el procedimiento que se utiliza para hacer que no se diferencien de las reales es reducir la calidad de la foto. En una imagen de baja calidad y en escala de grises es prácticamente imposible distinguir el resto de objetos y sólo se ve visible la carretera que es lo que se necesita para la conducción del vehículo. Al principio de las pruebas, la unidad de retroalimentación se asignaba aleatoriamente para evitar que se asociara con un tipo de movimiento concreto.

3. Fase de diseño del prototipo

El objetivo principal de este trabajo fin de grado es la creación de un tipo de vehículo autónomo que sea capaz de aprender a conducir imitando a un conductor humano. Por eso, el prototipo necesitaba unos sensores, una cámara para poder recabar toda la información necesaria del entorno y su comportamiento en cada instante. Para hacer esto, se necesitaba que tuviera un controlador remoto o algún tipo de mecanismo que permitiera su manejo.

Se optó por diseñar un vehículo de ruedas porque es más sencillo tanto en componentes como en programación para un principiante. Mover unas ruedas sólo implica el giro de unos motores; sin embargo, para desplazar otro tipo de robots que utilizan brazos y patas es indispensable la utilización de por lo menos dos pares de motores por cada una. Al igual que una pierna humana tiene la unión de la cadera y la rodilla, un robot necesita de un mecanismo similar que le permita hacer desplazamientos en los diferentes ejes (grados de libertad), lo que le permitiría ir por todo tipo de terrenos. Además, es imprescindible algún tipo de sensor que mida la inclinación del plano para acomodar la moción a éste.

Al ser un terreno liso y no terroso por el que se va a transitar no existe la obligación de emplear tracción de oruga como las destinadas por ejemplo, en los tanques militares; con simples ruedas, todo es menos complejo. Existen dos formas de establecer los giros del vehículo: una es la que emplean los coches teledirigidos que se basa en instalar ruedas movibles y la otra es la que se ha utilizado en este prototipo. En la primera, ambas ruedas giran a la misma velocidad; lo que influye en el movimiento es la posición del eje de la rueda, haciendo que se coloquen en la dirección a la que se quiere girar. Al contrario que ésta, si se mantienen las ruedas fijas, como ocurre con la segunda opción, para girar es necesario aplicar diferentes velocidades a cada rueda dependiendo de hacia donde se quiere ir. Por ejemplo, para virar a la derecha se debe suministrar mayor potencia al motor de la izquierda y el motor de la derecha puede tanto girar más despacio como no girar o incluso rotar hacia atrás para completar el giro correctamente.

En el caso del chasis (Figura 2) el único requisito ha sido que fuera lo suficientemente resistente y grande para albergar los componentes ensamblados encima. Existe un amplio abanico de materiales que sirven adecuadamente para esta labor pero el escogido es el plástico por ser el más económico y estéticamente visible. El kit adquirido incluía multitud de aberturas para encajar todo tipo de dispositivos. Se ha utilizado blu-tack por ser de fácil retirada, e incluso una

tercera rueda movible que facilita el desplazamiento.

Figura 2: Chasis similar al utilizado (http://goo.gl/Twfj7N)

En cuanto al hardware imprescindible para la adquisición de los datos de entrada, el instrumento más completo y que más se asemejaba al utilizado por el coche de Google, era el Kinect pero finalmente no pudo ser utilizado. Al descartar este componente, se requería de una cámara para tomar datos y un sensor que midiera la distancia a los objetos. Para esta labor existen sensores láser y sensores ultrasónicos. Con ambos dispositivos trabajando en conjunto, se llega a simular en menor medida lo que se hubiera obtenido con el Kinect, ya que en el caso de las distancias se obtiene únicamente un solo valor.

Lo ideal del registro de movimientos era almacenar el ángulo que gira el robot pero para ello es necesario de algún dispositivo que lo calcule. Aunque en estos momentos se encuentra en fase de prueba, un módulo de acelerómetro con giroscopio parece ser adecuado para producir los datos buscados. Mientras tanto, los valores registrados son el estado del motor (apagado o encendido) y el tipo de movimiento que se produce (parado, adelante o atrás).

4. Construcción del modelo

4.1. Ventajas de la construcción del prototipo frente a la simulación

Es bien cierto que con simulación se podría haber llevado a cabo este trabajo final de grado, como se ha visto en la Sección Antecedentes, en cambio parece más firme un proyecto que se basa en datos reales y no inventados o generados arbitrariamente. Además, también ofrece la posibilidad de probar todo tipo de superficies para la toma de datos.

Una característica muy importante de la construcción del prototipo frente a la utilización de un simulador es la comprobación en componentes físicos, lo que conlleva considerables ventajas en la implantación debido a que se ha desarrollado en un primer prototipo y la fase de pruebas ya está realizada. Entre las desventajas se encuentra que a cada problema que se presenta en el código obliga a tomar otra vez todos los datos y que ante cualquier error se pierde cantidad de tiempo del desarrollo del proyecto.

4.2. Rasberrry Pi

Como se ha mencionado con anterioridad, cada vez los dispositivos electrónicos son más económicos y éste es el caso concreto de la Raspberry Pi. La Raspberry Pi es un ordenador de bajo coste que puede usarse tanto de centro multimedia como de servidor, o también para implementar dispositivos tales como tablets, máquinas recreativas o drones. Además, es empleada en domótica o para la robótica en general. Su tamaño es el de una tarjeta de crédito y sólo necesita una memoria con el sistema operativo para funcionar. Conectándole un monitor, un teclado y un ratón es un pc completamente funcional. Este proyecto británico surgió como *crowdfunding* para llevar la educación digital a todas las escuelas de Gran Bretaña, de ahí su precio.

En la Figura 3 puede verse la disposición de los conectores en el modelo que ha salido recientemente al mercado, el 3B.

No obstante hay diversos modelos más, que se diferencian en tamaño y características, e incluso poseen diferentes conectores. El Cuadro 1 refleja la evolución desde las primeras que se empezaron a comercializar hasta las que se encuentran ahora mismo a la venta.

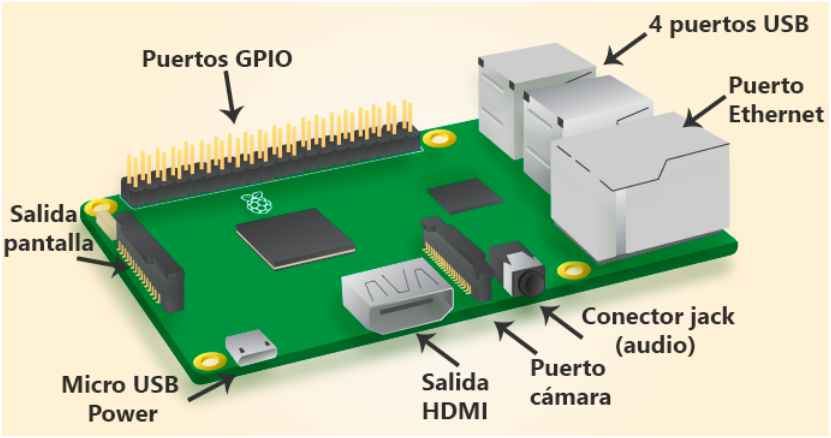


Figura 3: Conexiones Raspberry Pi (Adaptada de https://www.raspberrypi.org/)

Como puede verse en el Cuadro 1, ha evolucionado mucho y cada vez dispone de mayor capacidad con menor consumo de energía. A pesar de que no queda reflejado, el modelo 3B también cuenta con bluetooth y wifi integrado, y los primeros modelos incluían una salida de video RCA que más tarde se suprimió, probablemente dado al poco uso que la gente le daba.

	1A	1B	1A+	1B+	2B	Zero	3B
CPU	ARM Familia 11 700 MHz				ARM Cortex-A7 900 MHz	ARM Familia 11 1 GHz	ARMv8 4 cores 1.2 GHz (64 bits)
RAM	256 MB	512 MB	256 MB	512 MB	1 GB	512 MB	1 GB
Ethernet	No	Si	No	Si		No	Si
GPIO	26		40				
USB	1	2	1	4		1 (micro)	4
Tarjeta	SD		Micro SD				
Consumo (W)	1,5	3,5	1	3	4	0,8	4
Tamaño	8,56 cm × 5,65 cm		6,5 cm × 5,65 cm	8,56 cm × 5,65 cm		6,5 cm × 3 cm	8,56 cm × 5,65 cm

Cuadro 1: Características de todos los modelos de Raspberry Pi

Las versiones 1A salieron a la venta por unos 25 €, mientras que las 1B valían unos 35 € aunque actualmente puede ser difícil encontrar estos modelos al estar descatalogados. En su lugar se vende la Raspberry Pi 3 por unos 40 € y el modelo

Zero, el más barato hasta la fecha, por sólo 10 € pero parece estar fuera de stock en muchas de las tiendas. Los modelos B+ y 2B si que se pueden conseguir algo más baratos que el 3B, sobre los 25 €.

El modelo que se ha usado en este proyecto es el modelo B+, salió en julio de 2014 y costó alrededor de 35 €. Sus características son intermedias entre los primero modelos y los últimos que han salido. Cuenta con 512 MB de RAM, 4 puertos USB, conexión ethernet, HDMI, etcétera, y lo más importante, dispone de 40 pines GPIO para poder llevar a cabo el proyecto sin quedarnos escasos. También incluye otras diferencias respecto al modelo anterior que son el consumo de potencia reducido más o menos a la mitad y el cambio de ranura de SD a micro SD.

GPIO

Hasta ahora sólo se han mencionado las conexiones y las características que suelen tener ordenadores o tablets, obviando unos puertos que sí venían reflejados en el Cuadro 1, los puertos GPIO. Estas siglas significan *General Purpose Input/Output*, que en español corresponden a Entrada/Salida de Próposito General. Son pines que sirven de entrada y salida, ya sea de datos o señales. Por ejemplo, imaginemos que se quiere comprobar la humedad de unas plantas para saber cuándo hay que regarlas (ya sea de forma manual o automática). Conectando un sensor de humedad a estas salidas se podría tanto alimentar al sensor con la energía que necesite, como recoger el valor obtenido en el momento de la medición.

No todos los pines sirven para lo mismo. En la Figura 4 se puede ver la disposición de los 40 de los que disponen las últimas versiones. Entre ellos se encuentran salidas de 5 y 3,3 Voltios, tomas de tierra, GPIO normales, UART, SPI y I^2C.

Su programación es bastante simple. Una vez se conectan los componentes que se van a utilizar, sabiendo el número de pin GPIO al que están conectados, se indica si es de entrada o de salida como inicialización. Los de salida tienen dos valores posibles, alto y bajo, que corresponden a 0 o 1 en cuanto a tensión que se va a proporcionar como en cualquier circuito digital, y los de entrada lógicamente sólo pueden utilizarse para leer datos.

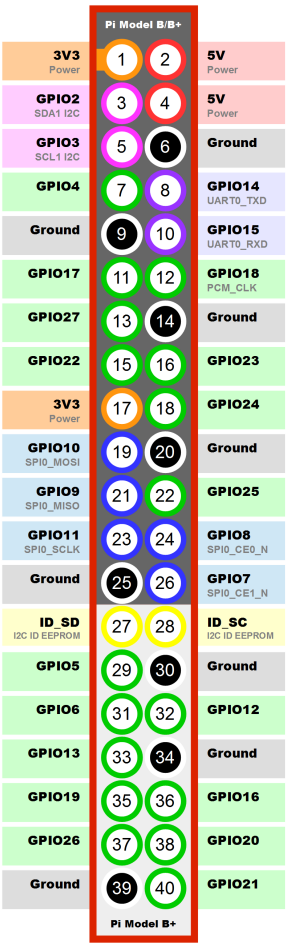


Figura 4: Mapa de pines de las salidas GPIO (http://goo.gl/FvjUks)

4.3. Arduino

Pese a que no se ha hecho uso de este componente, se podría haber utilizado como alternativa a la Raspberry Pi y merece la pena su mención en esta memoria.

Se trata de otro proyecto también desarrollado para estudiantes pero éste tiene su origen en Italia. Su intención era crear una placa de menor costo que las que se estaban comprando en ese momento y que cada uno pudiera montarse su propio prototipo; por ello los esquemas y planos son de libre distribución. Esto ha dado lugar a diversas versiones de la misma, siendo las más comunes y utilizadas las de esta marca.

La forma de utilización de Arduino es algo diferente a la de Raspberry. Al con-

trario que ésta, no constituye un pc completo ya que es un microcontrolador, por lo que es necesario uno para programar e introducir el código. Como puede verse en la Figura 5, no dispone de puertos USB por lo que todas las funcionalidades que se quieran añadir a la placa original deben hacerse mediante otros dispositivos llamados escudos o por su nombre en inglés, *shields*. Por ejemplo, si se hubiera querido incluir conectividad wifi al coche, en vez de usar un adaptador USB destinado a esta función, se hubiera necesitado adquirir un escudo que lo proporcione. Otra cosa que también la diferencia es que tiene un lenguaje de programación propio basado en C++.

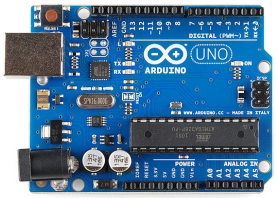


Figura 5: Arduino Uno (http://arduino.cl/arduino-uno/)

Se han detallado algunas de las desventajas pero, asimismo, cabe destacar que tiene también muchas ventajas. Su precio es algo inferior a Raspberry Pi y proporciona más salidas digitales, e incluso cuenta con salidas analógicas de las que carece la placa elegida. Otra de las ventajas es que los pines funcionan con una tensión de 5 Voltios por lo que nos ahorraríamos hacer el divisor de tensión que se explicará más adelante. La principal razón de utilizar la Raspberry Pi es por su capacidad de computación para generar la red neuronal. Mucha gente prefiere por ello utilizar ambas opciones juntas, combinando las ventajas que cada plataforma ofrece.

4.4. Conceptos físicos necesarios

Antes de comenzar a explicar los componentes del prototipo de robot, repasaremos previamente unas básicas nociones de física. Estas explicaciones serán muy concretas, adaptadas a la Raspberry Pi.

4.4.1. Circuitos

Un circuito es una red de elementos eléctricos o electrónicos conectados entre sí formando mínimo un camino cerrado. El más sencillo que se puede encontrar es el constituido por una fuente que produce electricidad, que puede ser una pila o en este caso un pin GPIO de la Raspberry, y un elemento como puede ser un motor. El sentido del circuito es siempre desde el polo positivo de la fuente de alimentación hacia el negativo.

Ninguno de los circuitos que se han realizado en este proyecto son demasiado complejos. Normalmente son fuente, resistencia y elemento, sea sensor, led o controlador.

4.4.2. Elementos básicos

Resistencias

Son componentes eléctricos que presentan una oposición al paso de corriente en el circuito. Su valor está determinado por unas franjas de colores y su unidad de medida es el Ohmio (Ω).

Led

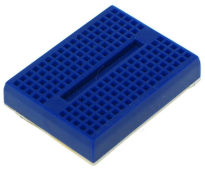
Es un componente que emite luz al aplicarle corriente. Tiene dos patillas de diferente tamaño, la más larga corresponde al polo positivo.

Elementos de conexión

Para unir los diversos componentes se han utilizado los siguientes elementos:

- Cables: como puede verse en la Figura 6a son cables que no requieren soldadura y existen tres tipos: macho-macho, macho-hembra, hembra-hembra.
- Placa de prototipos: son unas tablas con agujeros donde se insertan todos los elementos. En la Figura 6b se puede visualizar una captura de la misma. Cada orificio de una línea perpendicular a la marca central se encuentra conectado en serie con todos los demás de esa misma recta.

(a) Cables (https://goo.gl/UgRTBe)



(b) Placa de prototipos (http://goo.gl/UL2lzE)

Figura 6: Elementos de interconexión

Divisor de tensión

Anteriormente se mencionaba que había pines de 5 y de 3,3 Voltios pero estos pines proporcionan una tensión continua de salida por lo que si se quieren controlar los dispositivos se necesita utilizar los pines GPIO preparados para ese fin. Estos pines funcionan a 3,3 V y, por lo general, los sensores trabajan con 5 V por lo que necesitamos disminuir esa tensión para no quemar la placa. La forma más sencilla es hacer un divisor de tensión o voltaje, que consiste en colocar la conexión al pin de salida entre dos resistencias en serie para hacer que la tensión sea menor. Se rige por la siguiente función:

$$V_{salida} = V_{entrada} \cdot \frac{R_2}{R_1 + R_2} \tag{1}$$

Teniendo el $V_{entrada}$ y sabiendo que la salida no debe ser superior de 3,3 V, sólo necesitamos variar los valores de las resistencias para ajustarlos a los resultados adecuados. En este caso concreto, la primera es de 470 Ω y la segunda de 517 Ω, obteniéndose 2,62 V que es suficiente para obtener el valor necesario.

PWM

La modulación por ancho de pulsos, mucho más conocida como PWM (*Pulse-Width Modulation*) por sus siglas en inglés, es un mecanismo mediante el que se puede modificar una señal de tipo periódica.

En una señal digital binaria sólo se tienen dos valores: alto y bajo. La proporción de tiempo en el que está en cada valor es la misma, produciendo una intensidad media en la salida. Si modificamos esos tiempos podemos obtener mayor o menor intensidad pudiendo regular la emisión de luz de un led o la velocidad de los motores.

4.5. Componentes utilizados

Ahora se procederá a la explicación de todos los elementos que se han empleado.

4.5.1. Sensor ultrasónico

Es el sensor encargado de comprobar la distancia que hay al objeto más cercano justo situado delante del robot. En este caso también es necesario saber algo de física. El modelo que se ha utilizado es el HC-SR04, bastante común y muy económico.

Para calcular la distancia se emite un pulso sónico desde el emisor y se espera a que regrese, en la Figura 7 puede verse con mayor claridad esto. Con el tiempo que se tarda en recibir la onda de rebote y sabiendo la velocidad de propagación del sonido, se calcula la distancia al objeto que tiene más próximo.

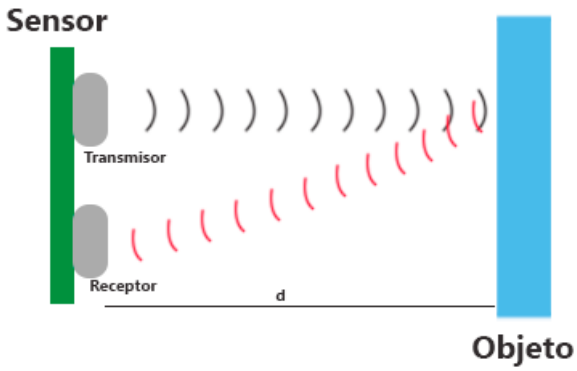


Figura 7: Funcionamiento del sensor

El sensor consta de 4 pines. El primero va conectado a la salida de 5 V para obtener la energía necesaria para su funcionamiento y el último corresponde a la toma de tierra para cerrar el circuito. Los dos pines centrales corresponden a la transmisión y la recepción, es decir, mandamos un pulso por el emisor y contamos el tiempo que pasa hasta que se recibe el pulso en el receptor. Con ese tiempo, sabiendo que la propagación del sonido es 3,43 m/s, si se sustituye en la fórmula $d = \frac{t \cdot V}{2}$ se obtiene la distancia al objeto.

En la Figura 8 se puede ver un esquema del circuito con los pines exactos en los que está conectado el sensor ultrásonico del coche. El cable amarillo corresponde al emisor o *trigger* y el azul al receptor, que se encuentra ensamblado

entre dos resistencias utilizando un divisor de tensión.

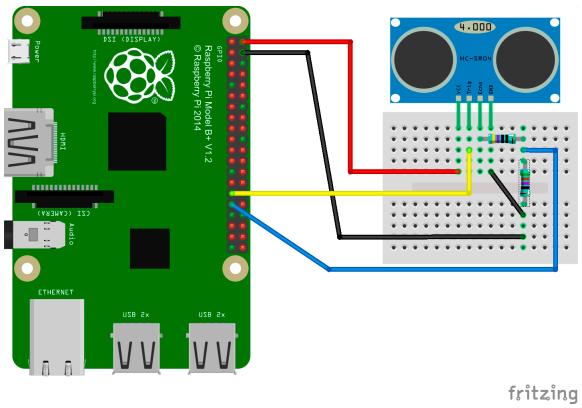


Figura 8: Circuito del sensor

4.5.2. Motores

Un motor eléctrico es un dispositivo capaz de transformar la energía eléctrica en energía mecánica. Existen dos tipos de motores dependiendo del tipo de corriente que los alimenta: corriente continua y corriente alterna. Se analizará más específicamente el primer tipo ya que es el grupo al que pertenecen los que se utilizan en electrónica. Dentro de éstos, hay varias clases pero los tres más importantes son los siguientes:

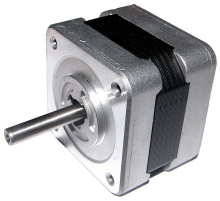
- Motores dc (9a): Tienen un movimiento continuo mientras se aplica corriente y para cuando deja de recibirla.
- Motores paso a paso (9b): El fabricante estipula un tamaño de paso y cada vez que se le aplica corriente se mueve en esa cantidad.
- Servomotores (9c): Se establece un ángulo de giro y ése es el movimiento que se produce.

Los motores giran en el sentido de la corriente por lo que si se invierten las polaridades de la pila o fuente, rodará en sentido contrario. Si la pila o fuente de energía no tiene la potencia suficiente, éstos no girarán o lo harán muy lentamente.

El motor que se ha elegido para este proyecto es el dc (como el de la Figura 9a) por ser el más común, económico y adecuado para montar un vehículo. No

es demasiado potente pero para esto es más que suficiente (de hecho se ha limitado la velocidad a la mitad).

(a) Motor dc (http://goo.gl/FyGnRP)



(b) Motor paso a paso (http://goo.gl/44J1vb)

(c) Servomotor (http://goo.gl/dfjSFs)

Figura 9: Tipos de motores utilizados en robótica

4.5.3. Controlador

Como se mencionó con anterioridad, la Raspberry Pi tiene pines de como máximo 5 Voltios y los motores elegidos, que son de los menos potentes existentes, trabajan entre 6 y 9 Voltios cada uno por lo que es necesaria otra fuente de energía y algún elemento para controlar su flujo. Esta acción se realiza mediante un componente electrónico llamado controlador, que libera energía de una fuente poniendo los pines por los que queremos que pase la corriente en alto como hacemos con los GPIO conectados a los elementos directamente.

Los motores funcionan en el sentido en el que les apliques la corriente (ya explicado en el apartado de los motores) por lo que hay que tener en cuenta cada pin del controlador al que está conectado el motor. Hay dos formas posibles de control: una es dando toda la potencia posible y la otra es regulando la velocidad a la que van a moverse conectando también otros pines llamados PWM. Con estas salidas se puede regular la cantidad de energía que sale por el cable y así disminuir la velocidad del robot, ya que por defecto puede que vaya demasiado rápido.

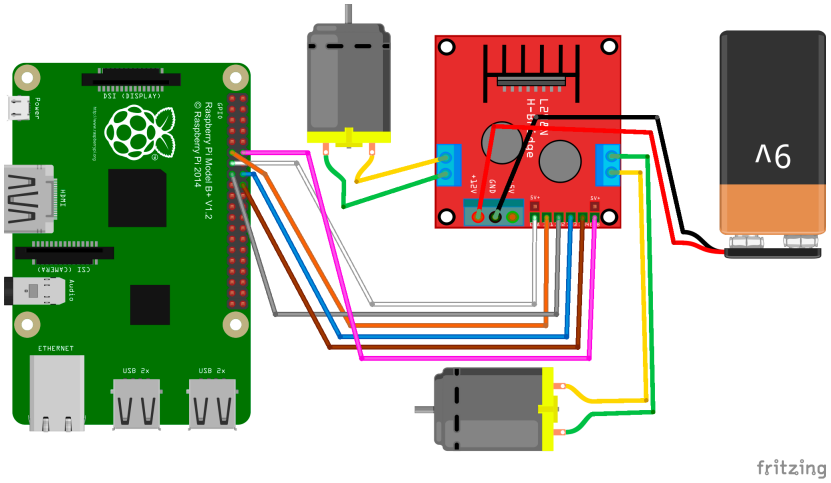


Figura 10: Circuito del controlador y los motores

Los cables blanco, naranja y gris corresponden al motor A, que se encuentra situado a la izquierda en la Figura 10. El primero corresponde a la activación del PWM para regular la velocidad y los otros dos son las salidas del motor. En el caso del motor B, el cable morado corresponde al PWM y los otros dos son las salidas del motor como en el caso anterior.

4.5.4. Cámara

Es un dispositivo electrónico que se utiliza para capturar fotografías. En este caso concreto se utiliza una webcam antigua conectada por USB, aunque existe una especialmente creada para la Raspberry que funciona más rápido y posee mejores características que ésta.

Originariamente tenía un trípode fijo hecho con una varilla de madera pero actualmente cuenta con un soporte de plástico que permite regularla fácilmente.

4.5.5. Alimentación

Para dar corriente a la Raspberry se ha utilizado una batería externa de las que se venden para los móviles. Tiene una capacidad de 2600 mAh que es más que suficiente para dar energía por un par de horas. También necesita una pila (mencionado en el apartado del controlador) para dar la potencia necesaria a los motores. Ésta posee una capacidad de 9 V que no es demasiado dado lo que consumen pero cumple perfectamente la función. Para mayor comodidad y ahorro económico, es una pila recargable porque prácticamente es de sólo un uso

debido a que tiene que alimentar también al controlador.

Un problema que surgía con esto era que no se podía estimar la distancia recorrida por el robot debido a que cuanto más gastada está la pila más despacio se mueven los motores por lo que en la misma cantidad de tiempo girando, unas veces podría desplazarse más distancia o menos.

4.5.6. Chasis y ensamblado

Todos los componentes han sido ensamblados al chasis con blu-tack para facilitar su montaje y desmontaje en caso de fallo. En la Figura 11 se puede ver el prototipo completamente montado.

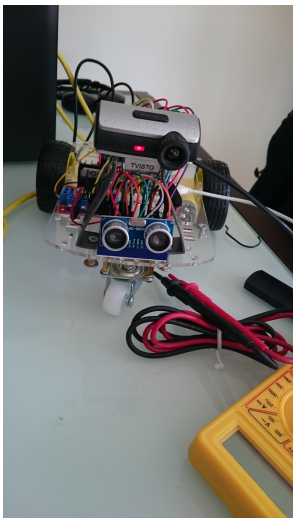


Figura 11: Foto del prototipo

4.5.7. Extras

Tiene un led para indicar el momento en el que se están captando los datos para evitar errores al parar el servicio que controla los movimientos del robot. Sigue el esquema del circuito mostrado en la Figura 12.

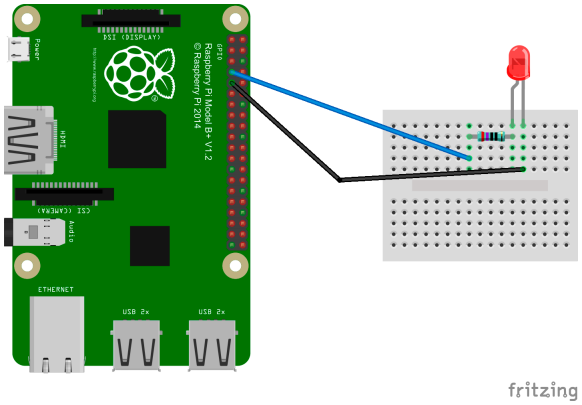


Figura 12: Circuito del led

4.6. Componentes descartados

4.6.1. Kinect

La idea original era utilizar el Kinect que Microsoft tiene como complemento de su consola Xbox para la toma de la imagen y de la profundidad, puesto que incorpora un sensor láser 3D que hace un barrido por todo el espacio del tamaño de la imagen, capturando las distancias. Esta singularidad proporcionaba mucha más similitud de este proyecto al de ALVINN.

Esta idea se descartó debido a las incompatibilidades que presentan las librerías de software libre desarrolladas para linux al ser un dispositivo propiedad de Microsoft. No se consiguió construir las imágenes correctamente y se mostraban en fragmentos desordenados.

En el caso de haberlo utilizado se habrían presentado problemas con la energía necesaria para hacer funcionar este dispositivo, dado que necesita una toma de 12 Voltios, además de la potencia recibida por el USB. La Raspberry, por ejemplo, se quedaba bastante corta de potencia a la hora de dar energía para usar el láser 3D y se producían errores de lectura en algunas ocasiones.

Otro problema por el cual también se desechó la idea de usarlo, fue el tamaño de este dispositivo. Es demasiado grande para el chasis y la dimensión pensada para el prototipo. Asimismo, al tener que poner una fuente de corriente, se incrementaba todavía más el volumen del coche.

5. Software del prototipo

Una vez conocidos todos los componentes físicos necesarios, su funcionamiento y forma de conexión, se va a hablar sobre el software y la forma de programarlo.

5.1. Sistema Operativo

Dentro del robot hay una distribución llamada Raspbian, que no es más que una versión de Debian adaptada al procesador ARM y al resto de sus componentes. En la página oficial de la Raspberry Pi podemos ver que hay multitud de sistemas operativos algunos preparados para convertirla en un centro multimedia como Openelec u OSMC, otras distribuciones de linux como por ejemplo algunas versiones de Ubuntu y, también, una versión de Windows preparada para el Internet de las cosas. No sólo existen los que hay en la web oficial, hay sistemas por cada uso que le quieras dar a la placa, llegando a haber por ejemplo, uno para crear un reproductor de música llamado Pi MusicBox.

El elegido fue Raspbian por ser el más completo de todos. También entonces por ser el más estable, ya que no había tanto donde elegir ni tanto soporte de la comunidad.

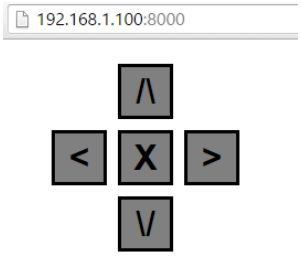
5.2. Lenguajes de programación

Al utilizar un sistema operativo completo, realmente se puede programar en cualquier lenguaje, siendo el más común Python pero no el único que se utiliza, ya que hay quien prefiere programar en C, C++ o Java. También puede usarse Scratch, debido a que la misión de la Raspberry era llevar la educación digital a los más pequeños en sus escuelas, esta forma de programar viene incluida en el sistema operativo. Es una forma de implementación muy intuitiva y visual (similar a la que se usa en la programación de los productos de Lego) en la que se utilizan una especie de cajas llamadas bloques. Toda la información se puede encontrar en su página web e incluso probar como funciona, puesto que se puede hacer directamente online.

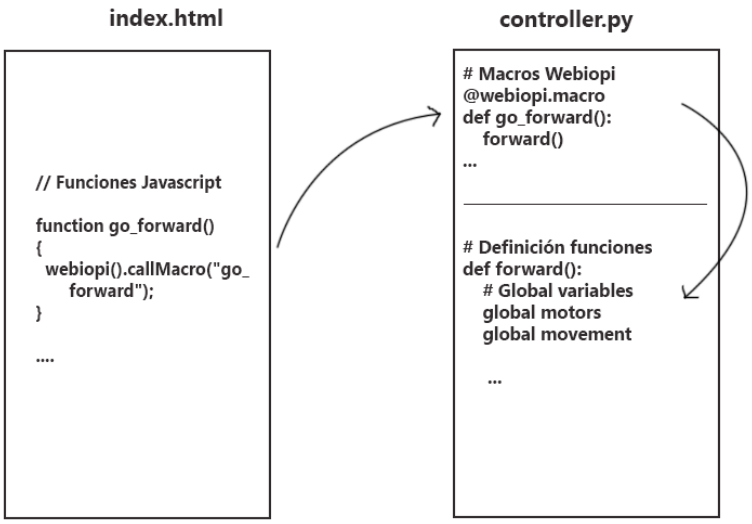
El lenguaje escogido fue Python por ser el más utilizado en todo tipo de proyectos creados con la Raspberry Pi. Asimismo, por ser un lenguaje con gran cantidad de librerías que facilitan la manipulación de grandes arrays como son los que genera una cámara.

5.3. Controlador del robot

Para esta labor, lo primero que se pensó hacer fue capturar los botones pulsados en el teclado en un programa en Python al igual que el movimiento del personaje de un videojuego. Esto exigía entrar por *ssh* a la Raspberry y arrancar el programa. Al requerir un ordenador bien para conducir o sólo para arrancar, no iba a ser excesivamente cómodo y se descartó la idea. Más tarde, se descubrió una solución más sencilla, tanto de implementar como de usar, que era controlarlo mediante un *framework* preparado para utilizar los pines GPIO llamado Webiopi. Para esta medida no era necesario pantalla, teclado o ratón, permitiendo controlar al robot desde cualquier dispositivo que disponga de conexión wifi y un navegador. Este complemento se inicia junto con el sistema operativo, quedando a la espera de las órdenes para el control del robot.



(a) Front-end del servidor Webiopi



(b) Esquema funcionamiento Webiopi

Figura 13: Webiopi

Este servicio proporciona una forma de utilizar funciones de Python mediante macros que pueden ser llamadas desde un archivo HTML. Para hacer esto es necesario tener dos ficheros, uno *.html* y otro *.py* con todo el código necesario para la ejecución. En el HTML se encuentran los botones que llaman a las macros proporcionadas por las librerías de Webiopi, y en el archivo *.py* se tienen detalladas las macros que ejecutan funciones normales definidas en Python. En la Figura 13b tenemos un esquema de cómo se realizan las llamadas a las macros y funciones. La Figura 13a corresponde a la vista del usuario.

Webiopi tiene su propia librería GPIO implementada y algunos nombres o funciones propiamente dichas, varían por lo que es necesario comprobar la documentación para ver su utilización. También incorpora un servicio de REST Api para el manejo pero todavía no ha sido probado.

Actualmente este *framework* está alojado en un servidor *nginx* pero comenzó utilizándose en uno apache. Se decidió migrar de uno al otro por optimizar recursos debido a que *nginx* es más ligero, siendo más adecuado para pequeños proyectos personales.

5.4. OpenCV

Es una librería implementada con herramientas para la visión artificial. Aunque sólo se ha utilizado para capturar las imágenes mediante la cámara web y, también, para pasarlas a escala de grises, ofrece la posibilidad de buscar patrones en fotografías entre otras cosas.

Existen proyectos en los que coches de similares características a este reconocen señales de tráfico y semáforos. Otro tipo de tareas que también se pueden realizar con este paquete es el reconocimiento facial.

5.5. Almacenado de datos

Para guardar los datos mientras el robot se mueve se ha implementado un *thread daemon* que va tomando datos cada cuatro segundos aproximadamente (tres para capturarlos y otro segundo de espera para que se estabilice todo). Existe una variable que corresponde al estado de los motores y otra que indica el movimiento que se está realizando. Estos dos atributos se bloquean por medio de una señal para que queden consistentes con la captura de las imágenes y el valor calculado por el sensor de distancia. Una vez se tienen todos los datos se

guardan en una sola línea en un fichero de texto creado para este fin.

Para hacer la manipulación de la imagen más ligera se convierte el array a base64, que es un formato utilizado en bases de datos para ocupar menos memoria e incluso puede mostrarse directamente sin decodificar con algunos lenguajes orientados a desarrollo web.

A continuación de la foto en base64 se encuentra un decimal con la distancia en centímetros que ha captado el sensor ultrasónico. Después se ha guardado en un número de dos cifras el estado de los motores. En éste se pueden ver los siguientes valores: 00, 01, 10 y 11, donde el número de la derecha corresponde a ese motor y el otro al de la izquierda. El último dígito corresponde al tipo de movimiento que está ejecutando el robot: 0 si está parado, 1 si está desplazándose hacia delante y 2 si, por el contrario, lo hace hacia detrás.

5.6. Problemas

A lo largo de este tiempo han surgido varios problemas con el montaje del robot. La mayoría de ellos estaban provocados por lo que se creía que era una mala configuración del wifi, que hacía que la Raspberry se bloqueara y se hiciera imposible apagarla de forma segura, no quedando más remedio que desenchufarla. Actualmente parece haberse solucionado con un reemplazo de fuente de alimentación que parece no proporcionar la energía necesaria de forma continua.

Aunque al principio estaba todo conectado a la red local de casa, actualmente se conecta a la zona wifi del móvil. De esta forma, se reducen los problemas de retardo que se producían, haciendo que frenara más tarde de lo normal entre otras cosas.

6. Inteligencia Artificial

6.1. Aprendizaje por observación

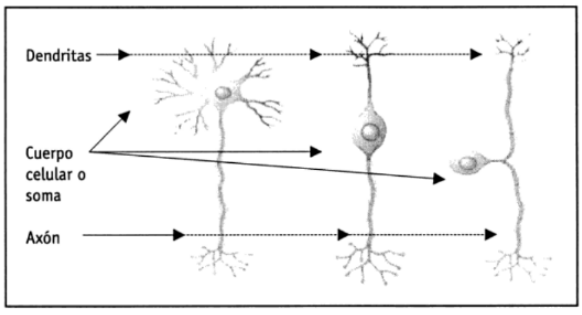
En psicología, el aprendizaje por observación es la adquisición de conocimientos mediante la visualización de comportamientos de otros llamados modelos. Este tipo de aprendizaje se da por ejemplo, en niños. Un niño se fija en lo que hacen el resto de personas a su alrededor, ya sean otros niños, sus padres o demás familia, e imita esas acciones.

En computación se toman muchas cosas prestadas de la naturaleza y éste es un ejemplo. Aunque no es exactamente igual, el caso de una máquina tiene ciertas similitudes con el aprendizaje por observación que se comentaba en el párrafo anterior. Se necesita una entrada de datos, al igual que el niño recibe estímulos sonoros y visuales, para poder determinar y reproducir el comportamiento.

En este caso concreto tenemos imágenes, la distancia al objeto más próximo que se encuentra delante, las acciones que están realizando en ese mismo momento los motores y el tipo de movimiento realizado. Estos datos se guardan por líneas y conforman el *dataset* que se toma de entrada para la red neuronal.

6.2. Redes neuronales

Otro ejemplo de elementos que tomamos de la naturaleza es éste, más concretamente de la biología. El cerebro está formado por neuronas interconectadas entre sí que reciben estímulos del entorno y envían impulsos para ejecutar acciones en función de éstos.



FUENTE: Adaptado de IIBE (2007).

Figura 14: Estructura de las neuronas [7]

Las neuronas están compuestas de tres partes: dendritas, cuerpo celular o soma y axón. Las dendritas conectan una o varias neuronas y son las encargadas de recibir la información. En la parte del cuerpo celular o soma es donde se decide lo que se va a realizar, siendo el axón el encargado de enviar esa información a las otras neuronas. Todo esto puede verse mejor en la Figura 14.

El caso del ordenador es bastante similar al caso del cerebro pero tiene algunas diferencias. Aunque la velocidad del cálculo en una máquina es mucho más rápida, el cerebro tiene mayor y mejor éxito en ciertas tareas que a un computador le cuesta mucho realizar.

Al igual que se explicaba anteriormente sobre las neuronas, las redes neuronales artificiales también están formadas por partes bastante parecidas. Como puede verse en la Figura 15, están formadas por la entrada de datos que es similar a las dentritas, otra parte que correspondería al cuerpo o soma de la neurona que coincidiría con la capa o capas ocultas de la red y finalmente la salida que se asemeja a los axones. En este caso concreto, la entrada serían los píxeles de la imagen y la distancia recogida por el sensor ultrasónico. La salida estaría formada por las acciones que deben ejecutar los motores o el tipo de movimiento, dependiendo de lo que estemos prediciendo.

En el caso de las redes neuronales artificiales necesitan entrenar con el fin de conseguir los mejores resultados de salida. Para ello, hay que destinar una parte de los datos obtenidos al entrenamiento y la parte restante queda para la “comprobación” del aprendizaje. En las implementaciones, que se explicarán más adelante, se han formado dos grupos uno con el 65 % de los datos y otro con los restantes.

Imágenes

Una imagen es un array de la resolución admitida por la cámara, en este caso 320x232 píxeles, donde se guarda el valor de cada píxel por cada color RGB. Es decir, en este caso concreto es una matriz de 232x320x3 con los valores de cada píxel.

Al contrario, si la imagen es en blanco y negro se reduce únicamente a un canal, por lo que quedaría un array más simple sólo del tamaño de la resolución.

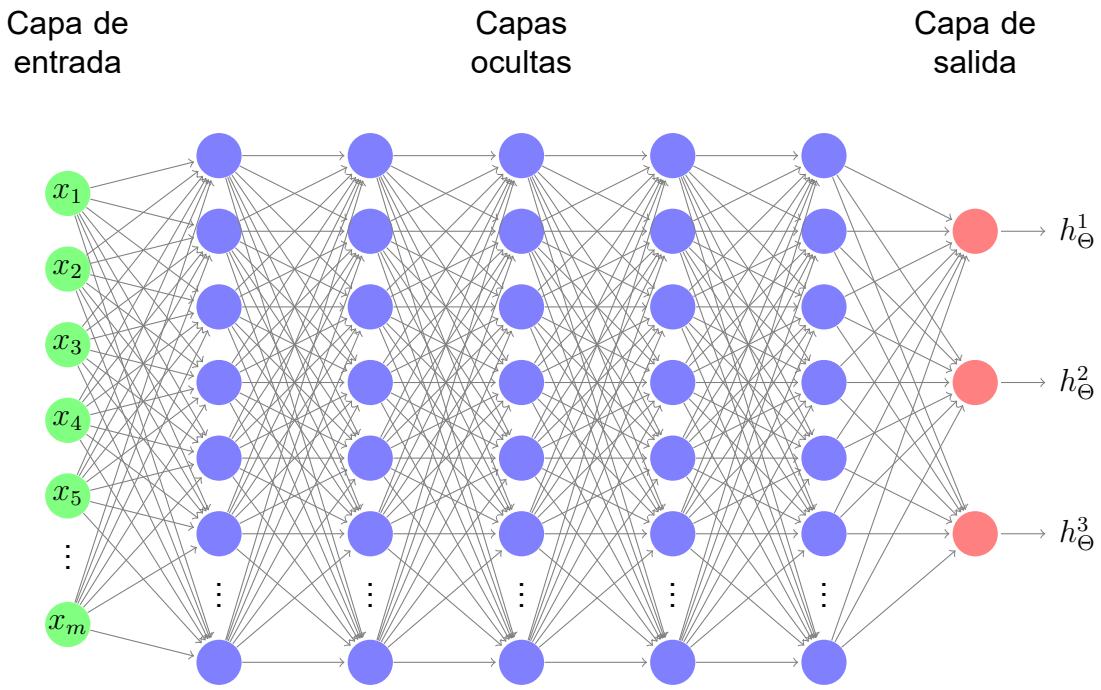


Figura 15: Esquema de la red neuronal

Reducción de las imágenes

Como se mencionaba en la composición de las imágenes, esta matriz es bastante grande ya que tiene 222.720 valores, estructurados en un array de 232x320x3. Si la imagen se convierte a escala de grises se reduce la matriz a 74.240 campos, que es con lo que se ha trabajado para que fuera mucho más manejable y, porque tampoco eran necesarios los colores.

La idea era crear una entrada para la red neuronal de cada valor pero el Knime no tiene la suficiente memoria para trabajar con ello y no llegaba ni a importar todos los datos del fichero. Al surgir este error, se decidió seguir con el entorno de R porque es algo más potente que Knime. Tras probar con Rstudio y diversos paquetes para la creación de redes neuronales, se manifestaba el mismo problema de memoria, por lo que se procedió a abordar el problema de otra manera con el Knime nuevamente.

La solución fue crear bloques de 4x4 píxeles y hacer la media de su valor. Esto, aunque podía realizarse con la Raspberry Pi, por mayor comodidad se ha probado en un ordenador, ya que tenía que ser utilizado para Knime.

Finalmente hubo que incrementar el tamaño de los bloques a 8x8 porque al hacer PCA continuaba produciéndose un error.

6.3. Knime

Es un software que proporciona herramientas para el análisis de datos. En el grado, es el programa utilizado en la asignatura de Aprendizaje Automático y Minería de Datos.

Es un programa muy visual y potente, que permite programar mediante módulos llamados nodos. Cada uno de ellos ofrece una funcionalidad diferente e incluso se pueden crear o descargar de repositorios que incluye el propio programa. Algunos de sus nodos sirven para tratar los datos, hacer operaciones con ellos, realizar manipulaciones en los datos leídos o su visualización. También se pueden aplicar funciones tanto estadísticas como de minería de datos como pueden ser la regresión lineal, árboles de decisión o Naive Bayes, entre otras.

6.4. PCA

En el apartado anterior se ha mencionado este concepto pero es en esta sección donde se va a explicar lo qué es y para qué se utiliza.

PCA corresponde a las siglas de *Principal Component Analysis*. Es un procedimiento estadístico que sirve para reducir la dimensión de un conjunto de datos. Como se explicaba en el apartado de la composición de las imágenes, con esta entrada se juntan demasiados valores que hacen difícil el manejo de los datos aunque estos se hayan reducido. Con esta técnica conseguimos extraer la información más importante del conjunto de valores de entrada.

El objetivo del PCA es reducir la dimensión $\mathbb{R}^n$ a $\mathbb{R}^k$. Para ello hace falta calcular la matriz de co-varianza que sigue la fórmula que se encuentra a continuación:

$$\Sigma = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)})(x^{(i)})^T \tag{2}$$

donde $\{x^{(1)}, ..., x^{(m)}\}$ es el conjunto de datos de entrenamiento. Cada $x^{(i)}$ es un vector de $\mathbb{R}^n$.

Después hay que obtener sus vectores propios. Sea $U = (u_1, u_2, \ldots, u_n)$ la matriz cuadrada que contiene en las columnas los n vectores propios de la matriz de co-varianza Σ. De estos n vectores, sólo se utilizarán los primeros k para

calcular la proyección $z^{(i)}$ en $\mathbb{R}^k$ de un punto original $x^{(i)} \in \mathbb{R}^n$ con la fórmula $z^{(i)} = (U_k)^T * x^{(i)}$, donde $U_k = (u_1, \ldots, u_k)$. Luego, el valor aproximado $x^{(i)}_{aprox}$ se puede obtener calculando $U_k * z^{(i)}$.

Dependiendo de la cantidad de información que se quiera conservar se elegirán unos k u otros. La siguiente fórmula corresponde a un 1 % de error, que es lo que se ha utilizado para no perder demasiada información:

$$\frac{\frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)} - x^{(i)}_{aprox}\right\|^2}{\frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)}\right\|^2} \leq 0,01 \tag{3}$$

6.5. Recolección de datos

El archivo que se utiliza para crear la red neuronal contiene 1989 filas, que se ha completado conduciendo el robot intentando obtener una amplia variedad de movimientos. Los valores se han capturado al manejar el coche en campo abierto, es decir, no existía ningún camino o carretera en el suelo que guiara su trayectoria. En ocasiones se ha seguido un línea recta esquivando objetos. Otras veces dando marcha atrás al encontrar una pared y en algunas ocasiones repitiendo hacia adelante y hacia atrás con diferentes aproximaciones para guardar tanto diferentes ángulos de vista como distancias. En cuanto a los giros, en ciertos momentos se ha rotado alrededor de un punto y otros simulando un circuito ovalado o en forma de un polígono irregular.

Cada línea tarda en tomarse aproximadamente cuatro segundos y los datos de los que disponemos son 1989 como se ha mencionado anteriormente, por lo que en esta fase se ha empleado un total de 7956 segundos que son algo más de dos horas. El 65 % de las filas de este fichero se utilizan para el entrenamiento de la red, mientras que el resto sirven para el cálculo de la precisión que ha adquirido la red después del aprendizaje.

6.6. Implementación neuronal realizada con Knime

Se han creado dos implementaciones utilizando Knime para contrastar los diferentes resultados en las predicciones tanto del estado de los motores como del tipo de movimiento. La primera implementación (Figura 16) está constituida por tres bloques además de la lectura y el cálculo de los resultados finales. Estas partes son: Procesado de los datos, PCA y Red Neuronal. En el primer bloque es donde se escogen los datos que van a alimentar la red neuronal. En el segundo

se hace PCA a esos datos para extraer la información más importante. Finalmente, en la última parte es donde se entrena la red y se predicen los resultados.

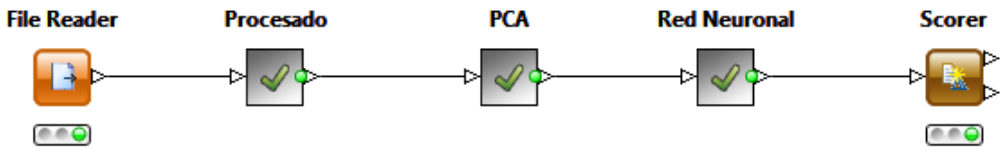


Figura 16: Implementación 1

En la Figura 17 se pueden visualizar las funciones contenidas en el módulo de la red neuronal. En ésta se ha utilizado la herramienta *RProp MLP*, el módulo *learner* para el aprendizaje y el predictor para la parte de "comprobación". Este bloque utiliza la propagación hacia atrás con una capa oculta de 10 nodos para calcular los pesos de la red neuronal, ajustando así lo máximo posible estos valores haciendo varias pasadas a los datos de entrenamiento.

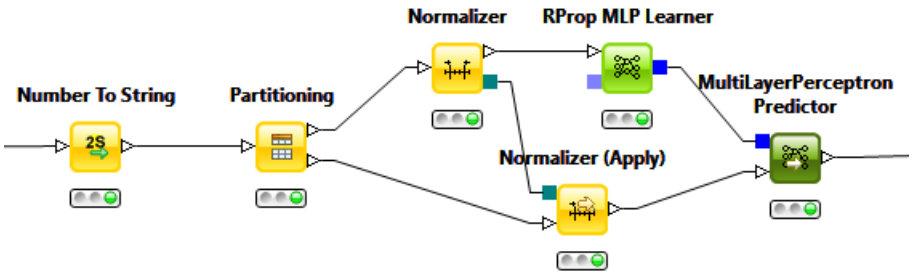


Figura 17: Implementación del bloque Red Neuronal

Los datos arrojados por la red neuronal se dividen en cuatro grupos: Positivos (P), Falsos Positivos (FP), Negativos (N) y Falsos Negativos (FN). Si tenemos en cuenta por ejemplo los resultados de predicción del estado de motores para el 00:

- P: Número de veces que la red predijo que era 00 y acertó.
- FP: Número de veces que la red etiquetó como 00 pero era 01, 10 o 11.
- N: Número de veces que la red predijo que no era 00 y acertó.
- FN: Número de veces que la red clasificó como distinto (01, 10 o 11) pero era 00.

	Positivos	Falsos Positivos	Negativos	Falsos Negativos	Precisión
00	58	1	161	0	?
01	25	35	138	22	?
10	16	13	151	40	?
11	41	31	130	18	?
Total	140	80	580	80	0,636

Cuadro 2: Resultados de la predicción del estado de los motores

Como se puede ver en el Cuadro 2, el estado de motores que ha sido predecido con mayor número de positivos y negativos fue el estado de motores 00. También se visualiza que es el que menos número de falsos positivos y falsos negativos obtiene. La precisión se calcula siguiendo la fórmula $p = \frac{2P}{2P+FP+FN}$, con la que se consigue un número entre 0 y 1. Cuanto más cercano a 1 es, mejor precisión se logra. En este caso concreto es de 0,636.

	Positivos	Falsos Positivos	Negativos	Falsos Negativos	Precisión
0	62	18	130	7	?
1	29	26	119	43	?
2	48	34	107	28	?
Total	139	78	356	78	0,641

Cuadro 3: Resultados de la predicción del movimiento

En el Cuadro 3 se pueden ver los resultados obtenidos en la predicción del movimiento que debe realizar el robot. En este caso también se ha predecido con mayor número de positivos la acción correspondiente a parar. Además es la acción que menos falsos negativos ha obtenido con un total de 7. La precisión de estas predicciones es de 0,641 que es ligeramente superior a la conseguida en la predicción del estado de los motores.

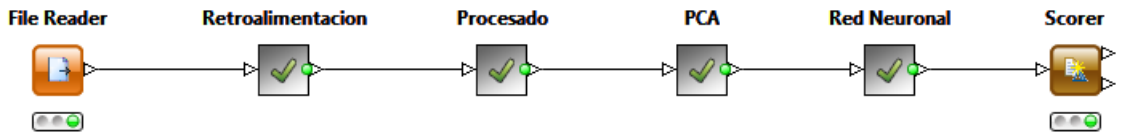


Figura 18: Implementación 2

Otra implementación que se ha realizado lleva además un bloque que añade una columna más a los datos en la que se refleja el anterior estado de motores

o movimiento. Este módulo se encuentra entre la lectura de datos del fichero y el procesado, como puede verse en la Figura 18, ya que los datos filtrados en el procesado se escogen de manera aleatoria.

	Positivos	Falsos Positivos	Negativos	Falsos Negativos	Precisión
00	52	9	155	4	?
01	17	30	133	40	?
10	19	43	127	31	?
11	32	18	145	25	?
Total	120	100	560	100	0,545

Cuadro 4: Resultados de la predicción del estado de los motores con retroalimentación del anterior

	Positivos	Falsos Positivos	Negativos	Falsos Negativos	Precisión
0	54	13	137	13	?
1	36	34	111	36	?
2	44	36	103	34	?
Total	134	83	351	83	0,618

Cuadro 5: Resultados de la predicción del movimiento con retroalimentación del anterior

Los Cuadros 4 y 5 corresponden a los resultados obtenidos con esta nueva implementación. Como puede verse, los datos conseguidos son algo inferiores a los anteriores. Esto quiere decir que el movimiento anterior no parece tener relación con el siguiente.

Cabe mencionar que en la parte de procesado de datos se escoge aleatoriamente la misma cantidad de datos de cada tipo de movimiento o de estado de motores por lo que cada vez que se ejecuta se obtienen diferentes valores. Algunas veces son superiores y otras veces inferiores a los que se presentan aquí.

Los resultados anteriores se han calculado designando una parte de los datos al entrenamiento y otra a la predicción. Si por el contrario destinamos el total a ambas cosas, la precisión aumenta significativamente. Para esta parte se ha prescindido del bloque de procesado y, como se puede ver en la Figura 19, se ha modificado la entrada de la red neuronal para adecuarla a la nueva situación. Se ha eliminado el bloque que particionaba los datos y se ha reducido a sólo un

módulo de normalización ya que todos los datos pasan por él.

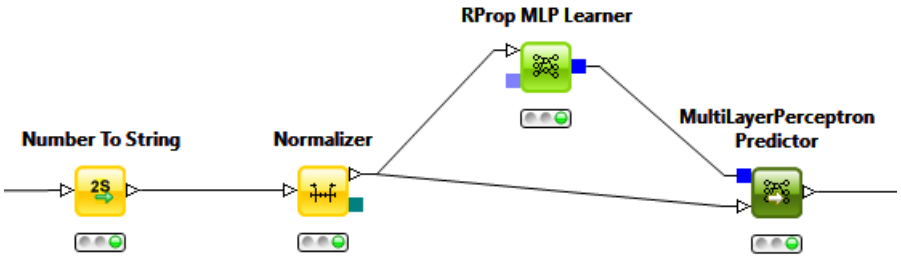


Figura 19: Implementación del bloque Red Neuronal con todos los datos como entrenamiento

En los Cuadros 6 y 7 se puede visualizar como la precisión se ve incrementada hasta llegar a 0,949 y 0,912 para la predicción del estado de los motores y la del movimiento, cuando anteriormente no llegaba en ninguno de los casos al 0,7. Esto está causado por la utilización de los mismos datos con los que se ha entrenado la red para la predicción. También influye el hecho de que son un mayor número de datos los que alimentan a la red neuronal.

	Positivos	Falsos Positivos	Negativos	Falsos Negativos	Precisión
00	886	1	1102	0	?
01	112	13	1819	45	?
10	131	26	1787	45	?
11	759	61	1158	11	?
Total	1888	101	5866	101	0,949

Cuadro 6: Resultados de la predicción del estado de los motores con todos los datos como entrenamiento

	Positivos	Falsos Positivos	Negativos	Falsos Negativos	Precisión
0	886	0	1103	0	?
1	866	145	947	31	?
2	61	31	1752	145	?
Total	1813	176	3802	176	0,912

Cuadro 7: Resultados de la predicción del movimiento con todos los datos como entrenamiento

Algo curioso que se puede apreciar es que no se producen falsos positivos ni falsos negativos (o solamente uno para los falsos positivos del estado de motores) para el estado parado. Esto puede estar debido a la cantidad de datos de ese tipo que se tienen almacenados en el fichero de entrada.

Con este comportamiento se reproducen los patrones de movimiento del conductor modelo en un cierto camino. Esto podría ser útil en conducción autónoma de, por ejemplo, una silla de ruedas desde una habitación a otra de una casa después de haber hecho el recorrido un par de veces.

7. Conclusiones

Una vez finalizado este trabajo de fin de grado se puede decir que se han alcanzado los objetivos que se plantearon en el inicio de forma satisfactoria. Llegando incluso a realizar algunas tareas que no estaban planteadas al comienzo.

En cuanto a la parte de construcción del prototipo el acabado ha sido bastante bueno y los resultados obtenidos lejos de ser perfectos, cumplen correctamente los requisitos de inteligencia artificial. Aunque los datos no son exactamente los que se plantearon, se ha podido solventar este inconveniente de forma razonable.

En la parte de la red neuronal se han obtenido mucho mejores resultados de los esperados en un principio. Inicialmente fue algo complejo saber como iban a ser tratados los datos al ser de tan gran magnitud el fichero pero después de la primera aproximación con la reducción de píxeles se volvieron más manejables.

Algunas de las mejoras o líneas de trabajo futuras para contrastar diferentes resultados podría ser la creación de un circuito o pista donde tomar datos para comprobar si el entorno influye en el aprendizaje. También sería bueno cambiar el sensor ultrasónico (o incluso añadirlo y tener ambos valores reflejados) por uno láser para ver si es mejor o peor en ciertos entornos ya que parece ser que no es muy eficiente en zonas con brillos.

La mayor mejora que se puede implementar es un seguimiento de la trayectoria que ejecuta el robot utilizando un acelerómetro y giroscopio, que permitan reflejar ángulos de giro o posiciones. No obstante también estaría bien registrar las distancias recorridas con cada movimiento por lo menos del punto anterior a ése.

Referencias

[1] Noticia bigdog. http://www.bostondynamics.com/robot_bigdog.html. Accedido por última vez el 08/09/2016.

[2] Noticia robocop. https://goo.gl/bpNonS. Accedido por última vez el 08/09/2016.

[3] Noticia zenbo de asus. http://goo.gl/oZTrfY. Accedido por última vez el 08/09/2016.

[4] Web oficial de raspberry pi. https://www.raspberrypi.org/. Accedido por última vez el 08/09/2016.

[5] Web oficial servicio webiopi. http://webiopi.trouch.com/. Accedido por última vez el 08/09/2016.

[6] Hebert, M., Thorpe, C., and Stentz, A. (2012). Intelligent Unmanned Ground Vehicles: Autonomous Navigation Research at Carnegie Mellon. The Springer International Series in Engineering and Computer Science. Springer US.

[7] López, R. and Fernández, J. (2008). Las Redes Neuronales Artificiales. Metodología y Análisis de Datos en Ciencias Sociales. Netbiblo.

[8] Masaki, I. (2012). Vision-based Vehicle Guidance. Springer Series in Perception Engineering. Springer New York.

[9] Pomerleau, D. A. (1989). Alvinn: An autonomous land vehicle in a neural network. Technical report, DTIC Document.

[10] Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., et al. (2006). Stanley: The robot that won the darpa grand challenge. Journal of field Robotics, 23(9):661–692.

[11] Waldrop, M. M. Autonomous vehicles: No drivers required. Nature (London), 518:20–23.

[12] Wooden, D., Malchano, M., Blankespoor, K., Howardy, A., Rizzi, A. A., and Raibert, M. (2010). Autonomous navigation for bigdog. In Robotics and Automation (ICRA), 2010 IEEE International Conference on, pages 4736–4741. IEEE.