Trabajo de Fin de Grado

Grado en Ingeniería Informática

Aplicaciones computacionales en la agricultura

Computational applications in agriculture

Carlos Gregorio Martín Pérez

La Laguna, 20 de agosto de 2017

D. **Eduardo Magdeleno Castello**, con N.I.F. 43.824.397-J profesor Titular de Universidad adscrito al Departamento de Ingeniería industrial de la Universidad de La Laguna, como tutor

D. **Manuel Jesús Rodríguez Valido**, con N.I.F. 52.840.833-N profesor Titular de Universidad adscrito al Departamento de Ingeniería industrial de la Universidad de La Laguna, como cotutor

C E R T I F I C A N

Que la presente memoria titulada:

"Aplicaciones computacionales en la agricultura"

ha sido realizada bajo su dirección por D. **Carlos Gregorio Martín Pérez**,

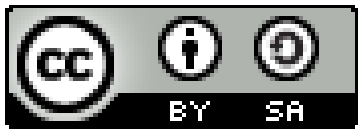
con N.I.F. 78.647.744-A.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 20 de agosto de 2017.

Agradecimientos

A mis tutores Manuel Jesús Rodríguez y Eduardo Magdaleno, que me han guiado y apoyado en todo momento. A los profesores Jesús Torres, Vicente Blanco y Fernando Pérez Navas, por prestarme su tiempo y resolver algunas dudas. A Néstor Albelo, compañero de carrera, por ponerme al día con la versión anterior del proyecto. Y a las personas que trabajan en la finca.

Licencia



Resumen

El proyecto que se ha realizado como TFG, se ha basado en una implementación previamente iniciada por antiguos alumnos de diversas facultades y que consistía en la creación de un sistema de carga totalmente eléctrico, conocido como PACA. Este sistema se desarrolló junto con una empresa privada, Finca Las Lucanas, y ha ido mejorando a lo largo del tiempo gracias al paso de los diferentes alumnos y profesores que han trabajado en él.

La importancia del proyecto PACA pretende facilitar las labores agrícolas de los empleados de la planta donde se encuentra el carro haciendo uso de las aplicaciones computacionales, a la hora de transportar carga de manera sencilla en lugares por donde otra maquinaria, debido a sus dimensiones, no pueda o le sea complicado llegar.

En la parte que corresponde al alumno, el principal objetivo del trabajo realizado ha sido, el de mejorar el carro añadiéndole nuevas funcionalidades a nivel tanto de hardware como de software.

Algunas de las principales tareas realizadas han sido la implementación de un sistema computacional que ejecuta un algoritmo de seguimiento, mejora del control remoto haciendo uso de un Smartphone y la APP desarrollada.

Palabras clave: seguimiento, conducción guiada, computación, agricultura, carro de carga.

Abstract

This project has been designed as FWD, and it is based on a previously version initiated by students of different faculties and the idea was the creation of a charging system known as PACA. This system was developed with the help of a private company called Finca Las Lucanas, and has been improving over time thanks to the students and teachers who have worked on it.

The project PACA has its importance in the able to facilitate agricultural work of employees where it is, helping them to transport heavy loads easily in places where other machinery, due to its size, could not or would be difficult to reach.

The main target of the work developed by the student, was to improve the system adding new features at hardware and software level.

Some of the main tasks performed were the implementation of a complementary computational system that runs a tracking algorithm, and improving the application for remote control for Android system.

Keywords: tracking, guided driven, computing, agriculture, loading carriage

Índice general

Índice de figuras

Índice de tablas

Capítulo 1
Introducción

En las últimas décadas, de forma paralela al desarrollo tecnológico, se ha producido un incremento en la utilización de ordenadores en labores agrícolas. Este incremento se debe a la reducción de costes del hardware y software, al aumento de la población que demanda una mayor cantidad de alimentos y a la ineficiencia de las labores manuales en determinadas tareas [**1**]. Los ordenadores permiten la gestión de explotaciones agrícolas a gran escala. Para ello utilizan distintas tecnologías como los robots agrícolas en invernaderos, los tractores guiados por GPS o la visión artificial para el control de calidad. En esta línea de trabajo, y como objetivo de este proyecto, proponemos realizar una implementación software para seguimiento de personas de un prototipo de vehículo agrícola. Concretamente, usaremos como soporte un prototipo eléctrico denominado PACA que se encuentra en Finca Las Lucanas.

1.1 Contexto

Como comentamos, PACA se encuentra en Finca Las Lucanas, una empresa situada en Valle Guerra que se dedica principalmente, a la producción de plantas verdes, flores, plantas aromáticas y hortalizas.

PACA es un sistema de carga totalmente eléctrico, puede ser controlado tanto a distancia como de forma manual. Principalmente lo compone, un chasis que soporta todo el sistema, un cuadro de control, donde se encuentran el joystick de control manual y el selector de estado, dos motores eléctricos situados en la parte delantera junto a las baterías de alimentación, y un cajetín donde se encuentra la circuitería principal de alimentación y procesado de datos. PACA posee un sistema de tracción formado por dos motores eléctricos capaz de transportar 200Kg y con una

autonomía de 8 horas, una jornada laboral. En la figura 1.1 se muestra una imagen de PACA con indicaciones de los componentes anteriormente citados.

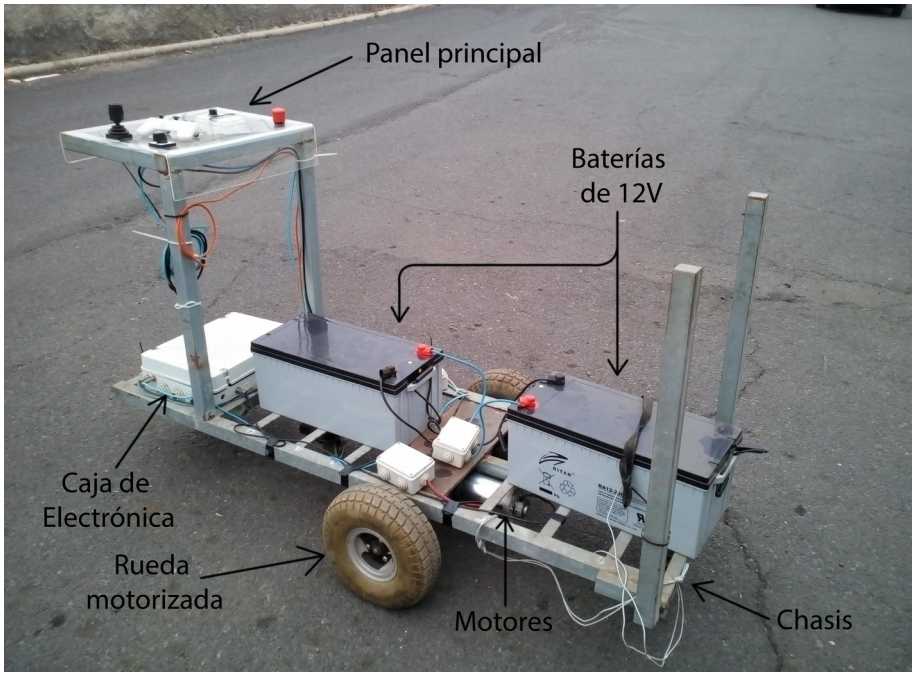


Figura 1.1: Componentes principales de PACA

El propósito principal de PACA, es el de poder facilitar el transporte de mercancías a lo largo de la finca. Cuenta con un sistema de control remoto, por tanto también podría servir para acercar ciertos elementos a otros operarios simplemente depositándolos en el carro y controlándolo remotamente hasta su destino. Esto permite que el personal no sufra tanto estrés físico al tener que cargar con materiales pesados por las inmediaciones de la finca.

1.2 Objetivos

Una de las aplicaciones computacionales en agricultura y objetivo principal del desarrollo de este trabajo de final de grado es desarrollar un software para la conducción autónoma de PACA. Para ello es necesario añadir un sistema computacional adicional al que posee PACA actualmente, que se encargue de la ejecución de éste algoritmo y se comunique con el sistema principal ya presente en el dispositivo.

Por otro lado, como objetivo secundario, se ha marcado el estudio de profiling e implementación hardware del algoritmo para reducir los tiempos de procesamiento y de esta manera optimizar el algoritmo. Así como la propuesta de mejoras para el refinado de las técnicas de seguimiento desarrolladas.

1.3 Organización del proyecto

Este documento tiene una estructura similar a la que se ha seguido en el desarrollo del proyecto (figura 1.2). Además del capítulo introductorio, se ha documentado el proceso de descripción del hardware y software que serán necesarios para el desarrollo del algoritmo. A continuación se ha desarrollado la primera versión del mismo. Por otro lado, se ha realizado una mejora de la APP móvil existente. Finalmente se ha llevado a cabo el desarrollo de la segunda versión del algoritmo con las pruebas situacionales.

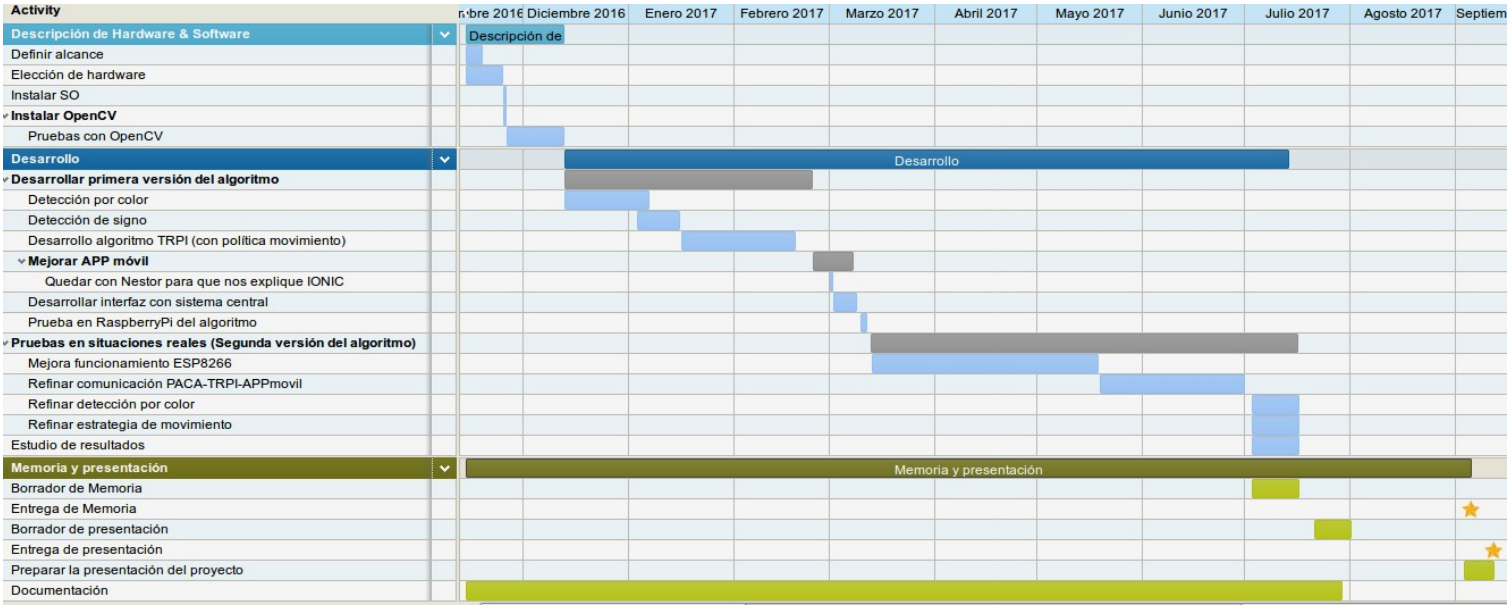


Figura 1.2: Diagrama de Gantt

Capítulo 2
Descripción de Hardware y Software

Hoy en día cualquier sistema que nos encontramos tiene un diseño basado en un hardware y un software para implementar su funcionalidad. En este caso, al margen de que posee otros sistemas hardware tales como mecánicos para producir movimiento, PACA posee un sistema maestro basado en el microcontrolador *mbed* cuyo *firmware* gestiona el funcionamiento del vehículo. También consta de electrónica de pequeña señal, para comunicaciones y lectura de sensores, y de potencia para suministrar energía a los motores y así poder mover al carro (figura 2.1).

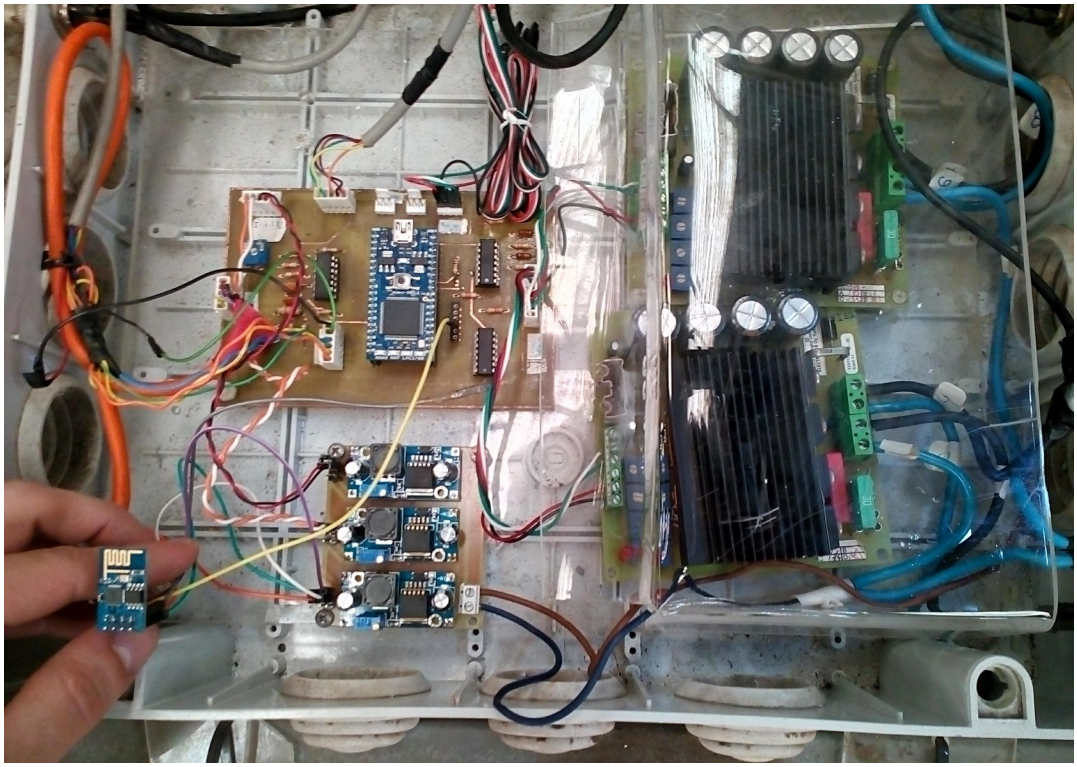


Figura 2.1: Caja de electrónica

Con el fin de conseguir el objetivo principal propuesto, al hardware existente se ha añadido un procesador Raspberry Pi (figura 2.2) más una cámara para dotar a PACA de visión (figura 2.3). El nuevo hardware ha sido instalado en el panel principal del carro.

Además de la interfaz humana que poseía, se ha mejorado la aplicación móvil ofreciendo a todo el sistema una mejor interacción y control humano haciéndola más intuitiva, visual y fácil de usar.

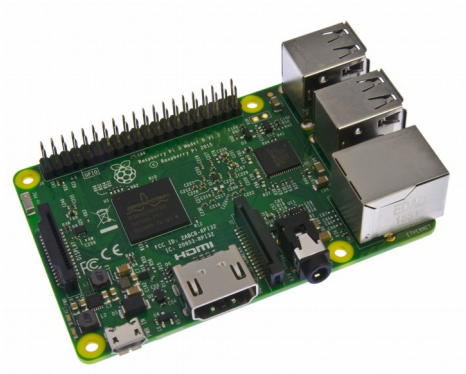


Figura 2.2: Raspberry Pi v3

Figura 2.3: Sony playstation eye

2.1 Raspberry Pi

La Raspberry Pi 3 es la tercera generación de ordenadores de bajo coste y alto rendimiento [**2**]. Funciona con sistemas Linux en este caso Raspbian [**3**], el sistema operativo oficial.

2.1.1 Características

- Quad Core 1.2GHz BCM2837 CPU de 64 bits y 1GB de RAM.
- BCM43438 wireless LAN y Bluetooth Low Energy (BLE).
- GPIO extendido de 40 pines.
- 4 puertos USB 2.0.
- Salida estéreo de 4 polos y puerto de vídeo compuesto.
- HDMI.
- Puerto de cámara CSI para conectar una cámara Raspberry Pi.
- DSI para conectar una pantalla táctil Raspberry Pi.
- Puerto Micro SD para cargar el sistema operativo y almacenar datos.
- Fuente de alimentación conmutada Micro USB mejorada hasta 2.5A.

2.1.2 Sistema operativo

Al funcionar con un sistema operativo basado en Linux, se pueden explotar las funcionalidades que dispone, como el uso de distintos lenguajes de programación, diferentes compiladores y programas adaptados a su arquitectura. También se aprovechará la gestión de procesos y servicios que ofrece un sistema operativo como Raspbian.

2.2 Firmware PACA

PACA aloja su firmware en el microcontrolador *mbed LPC1768* [**4**], que recibe y envía órdenes al resto de la electrónica. Actualmente acepta tres modos de funcionamiento, como se puede ver en la figura 2.4:

- Modo manual: Se puede controlar a PACA desde el panel principal mediante el joystick manual.
- Modo auto: Mediante la APP móvil, con un joystick virtual, se controla de forma remota vía wifi.
- Modo PS3: Ofrece conectividad bluetooth a un mando PS3 para que PACA pueda ser dirigida.

Hasta ahora, PACA requería de un operario a cargo de su control. Lo que se pretende con el objetivo principal del desarrollo de este TFG es prescindir de ese control directo y procurar que el operario también pueda estar haciendo su trabajo (como por ejemplo cargar el carro mientras éste le sigue).

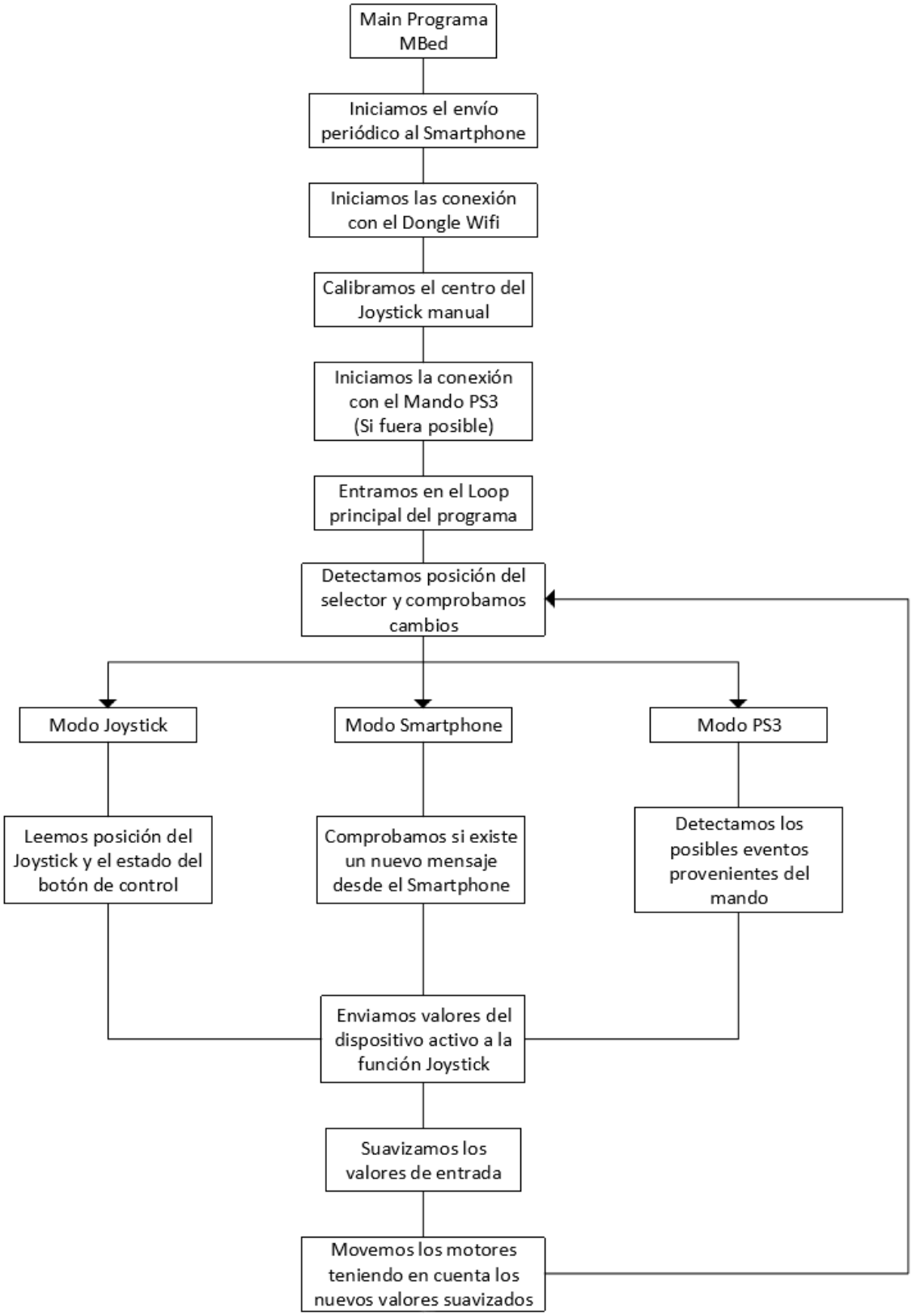


Figura 2.4: Diagrama funcionamiento PACA

2.3 Firmware ESP8266

ESP8266 (figura 2.5) es un *chip on board* que puede ser programado con comandos *AT* [**5**], un lenguaje de programación poco expresivo, por lo que se ha instalado un firmware distribuido por *NodeMCU* [**6**] que permite que éste interprete código *Lua* [**7**], un lenguaje de programación más fácil de usar y con mayor expresividad.

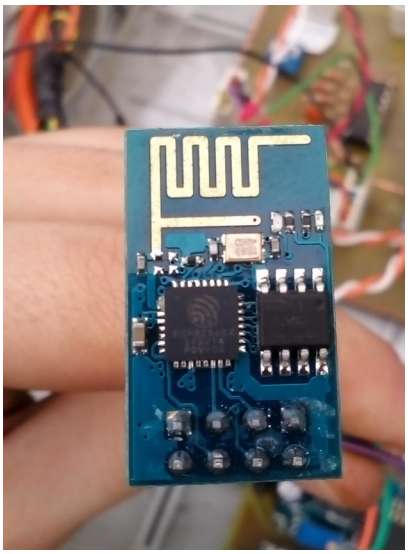


Figura 2.5: Microcontrolador ESP8266. Gestiona el wifi

El funcionamiento del *microcontrolador* es crear una red wifi de nombre *PacaRed*, contraseña *12345678* y con dirección IP *192.168.4.1*. Abre un puerto de escucha (8888) para que un dispositivo pueda conectarse. El *dongle* envía todo lo que recibe vía wifi a PACA a través de su conexión cableada. Y al contrario, cuando el ESP8266 detecta datos vía serial, los transmite al dispositivo conectado. A diferencia de la versión anterior, ahora el *dongle* es capaz de administrar varios dispositivos conectados a la vez.

Capítulo 3
Algoritmo de seguimiento TRPI

Se ha implementado un algoritmo que se ha llamado *Tracking RaspBerry Pi* (TRPI) en C++. Hace uso de las librerías *OpenCV* **[8 y 9]** para el tratamiento de las imágenes. El algoritmo debería simular el joystick virtual de la APP móvil y obtener de las imágenes unas coordenadas que, como en el caso del joystick, hacen referencia a la velocidad y dirección del desplazamiento para enviarlas a PACA vía wifi (figura 3.1).

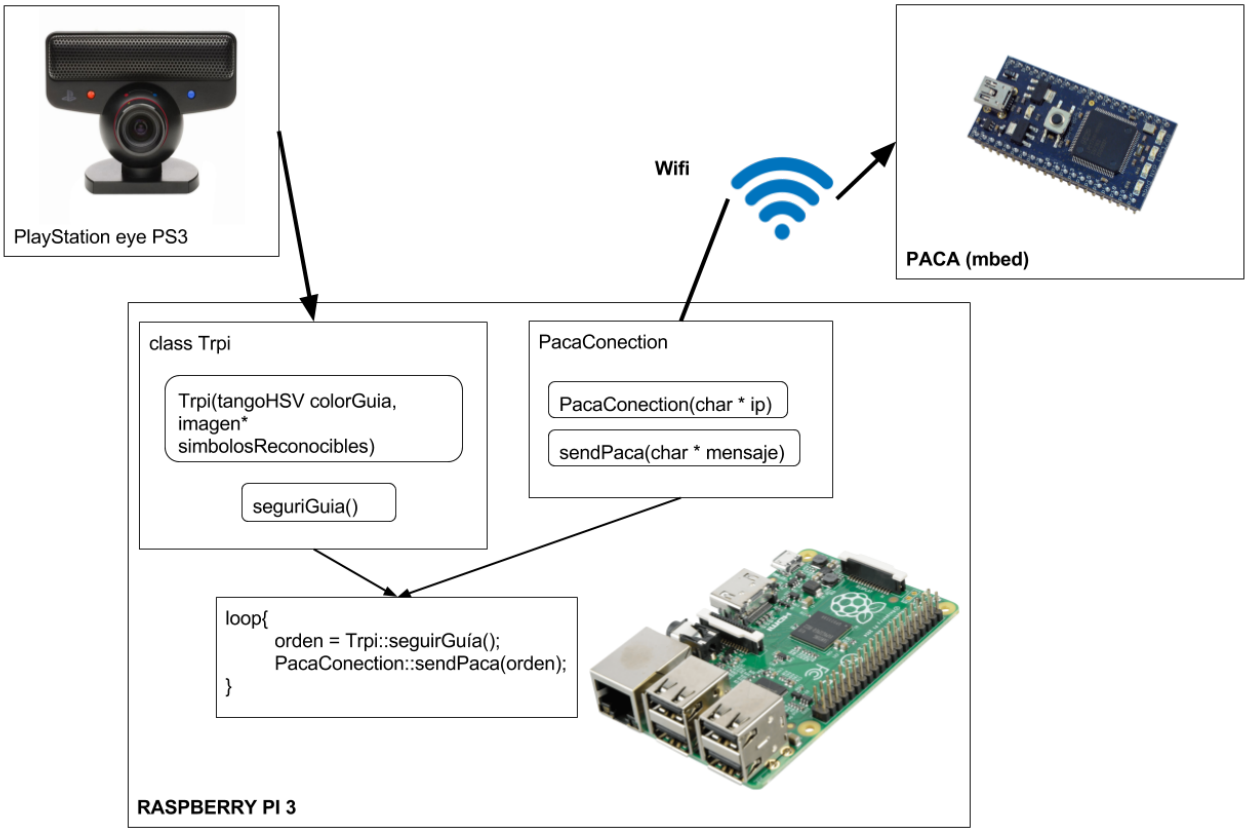


Figura 3.1: Diagrama previo de entidades TRPI

En sus primeras versiones funcionó en un portátil haciendo uso de algunas ventanas que mostraban los resultados de las trasformaciones (lo que PACA interpretaría como el objeto, *imagen threshold*), las funciones de suavizado y mejora de la precisión del seguimiento de las imágenes y para indicar el rango HSV del color del guía (figura 3.2).

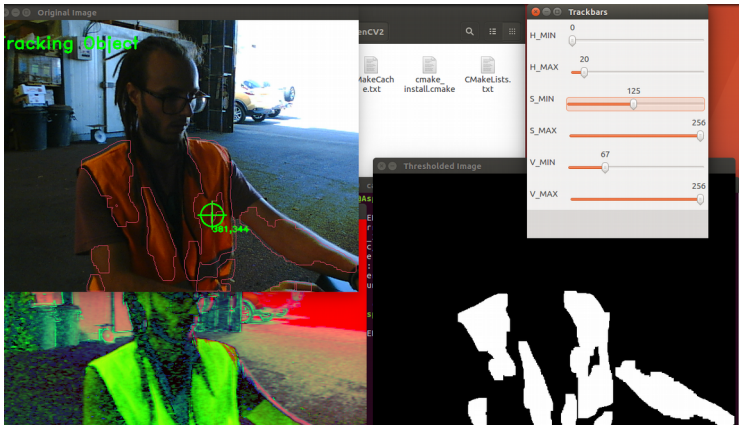


Figura 3.2: Pruebas de color TRPI. Imagen original, esquema HSV, threshold y barras para el rango de color respectivamente.

Para alcanzar el objetivo marcado se ha tenido que eliminar todo componente gráfico con la intención de crear un servicio que se inicia al encender la Raspberry sin necesidad de ser iniciado de forma manual tal y como se detallará más adelante en este documento.

Por otro lado, para reducir la complejidad a la hora de probar en Raspberry Pi o el portátil, se ha usado la herramienta *CMake* **[10]**. Se trata de una herramienta *open-source* para la compilación y depuración multi-plataforma.

3.1 OpenCV

OpenCV **[8 y 9]** es una biblioteca libre de visión artificial, que permite el tratamiento de las imágenes capturadas. Cuenta con funciones tales como operaciones morfológicas, transformaciones de esquemas de color, etc. Se utiliza desde sistemas de seguridad con detección de movimiento, hasta aplicaciones de control de procesos donde se requiere reconocimiento de objetos. Se distribuye bajo licencia *BSD* que permite que sea usada libremente para propósitos comerciales y de investigación.

3.1.1 Detección por color

Se puede detectar un objeto de varias formas, entre ellas reconociendo el color, la forma o el movimiento del mismo. En éste proyecto se pretende que PACA siga a un operario, así que las imágenes tomadas tendrán multitud de formas y habrá muchos objetos moviéndose. Por tanto, la técnica de detección por color ha sido, en un primer momento, la técnica seleccionada.

El modelo de color HSV representa valores de matiz, saturación y brillo como se observa en la figura 3.4. Permite una distinción bastante precisa de un color indicando el rango HSV (máximos y mínimos) en el que se presenta.

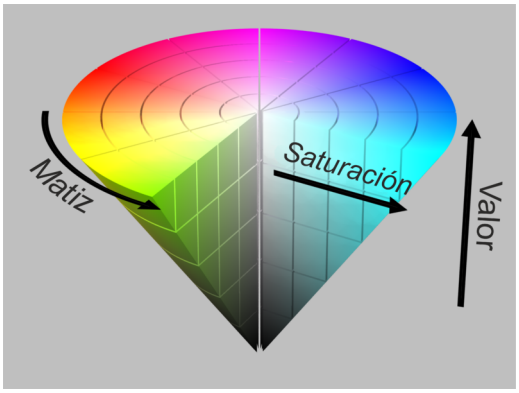


Figura 3.4: Cono de colores del espacio HSV

Figura 3.3: Chaleco reflectante operario guía

El desarrollo del algoritmo se ha escudado en éste método de detección por color con las operaciones morfológicas dilatar (*dilate*) y erosionar (*erode*) que provee *OpenCV* con la intención de eliminar los detalles de la imagen y destacar los objetos en primer plano en la imagen resultante.

Los valores finales que se han usado se detallan a continuación, y corresponden a las condiciones ambientales de la finca y el color naranja de un chaleco reflectante de alta visibilidad (tabla 3.3).

Atributos esquema de color HSV	Valor mínimo	Valor máximo
H (Matiz)	0	20
S (Saturación)	125	256
V (Brillo)	125	256

Tabla 3.1: Valores HSV del guía

3.1.2 Detección de signos

Se ha previsto el uso de algunos signos reconocibles (figura 3.5) a lo largo de las inmediaciones de la finca tales como giros de 90 y 45 grados además de una señal de *stop* que sirvan de soporte al seguimiento del guía y eviten que PACA pueda salir del camino.

Figura 3.5: Signos reconocibles por TRPI

Las señales son reconocidas dentro de la imagen transformándola a escala de grises y buscando rectángulos dentro de la imagen. Una vez identificadas las cuatro esquinas del marco, se extrae y se compara con los signos haciendo una diferencia matricial de las imágenes para reconocer cuál de ellos es [**11**].

3.2 Política de movimiento

Debido a la implementación de las técnicas mencionadas anteriormente en este capítulo (seguimiento del guía por detección de color y reconocimiento de signos), se ha fijado una política de movimiento que indica cómo ha de moverse PACA en cada caso (figura 3.6).

Se ha querido dar preferencia a la señal de *stop*, ya que en caso de producirse un comportamiento indebido, el operario puede parar el carro poniendo la señal en su campo de visión. En cualquier otro caso, el operario tiene preferencia.

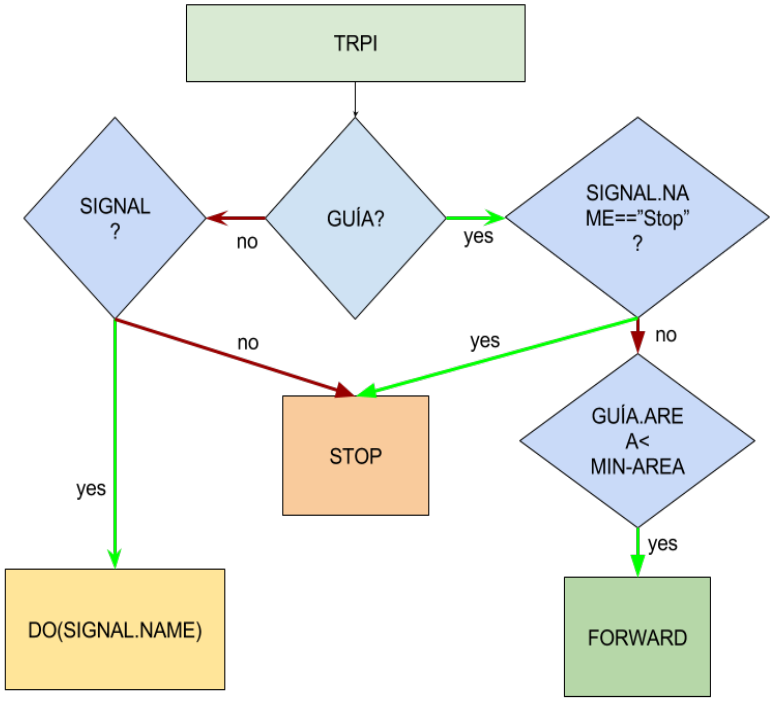


Figura 3.6: Diagrama política de movimiento

3.3 Codificación de las coordenadas

PACA originalmente funcionaba en el modo automático recibiendo una serie de mensajes (figura 3.7) vía wifi que contenían las coordenadas del joystick virtual de la APP (x,y) y además un bit (0 o 1) que permite saber si el sensor está siendo pulsado de forma premeditada o se ha movido por error.

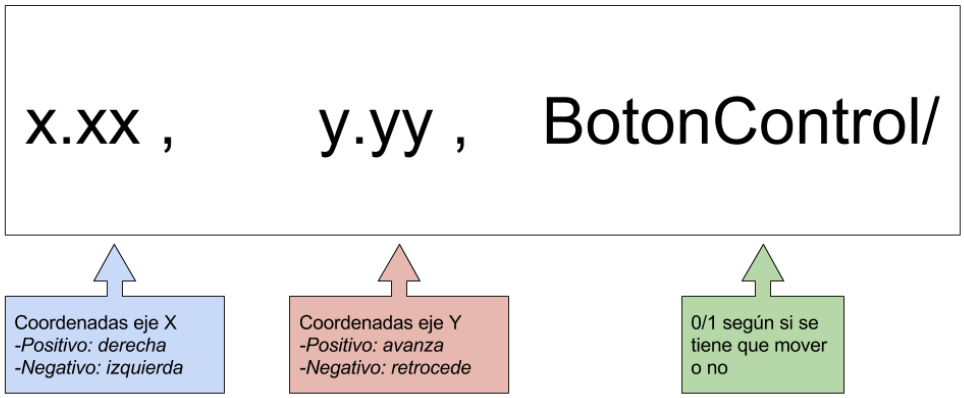


Figura 3.7: Codificación anterior mensajes

Para garantizar la compatibilidad de TRPI con PACA, en un principio se ha procedido a realizar la misma funcionalidad que realizaba el joystick virtual desde el móvil, pero ahora generando las coordenadas desde el algoritmo TRPI, que como se detallará en éste capítulo, las obtiene del movimiento del guía.

3.3.1 Eje X

Representa la dirección, el movimiento hacia los lados. Se obtiene haciendo una comparación de varias imágenes donde está presente el guía. Haciendo uso del método de detección por color, se traza un contorno sobre el objeto y se calcula su punto medio. Si se produce un desplazamiento con respecto a imágenes posteriores, quiere decir que el guía se mueve hacia los laterales.

3.3.2 Eje Y

El *eje y* representa, en un joystick, el movimiento de avanzar o retroceder, es decir, el sentido del movimiento. Estudiando las imágenes que captura el algoritmo, si un operario se acerca y aleja de su campo de visión, se puede observar un cambio en el área del objeto. Por tanto, haciendo seguimiento de su área, se puede obtener el sentido en el que PACA tiene que moverse (o parar si el área no varía).

3.4 Afinado del movimiento.

Se observa que, en las imágenes capturadas por TRPI, las condiciones ambientales de luz y las características de la propia cámara web producen una serie de movimientos involuntarios en PACA.

Depurando el funcionamiento del algoritmo, se observa que, efectivamente, mientras en un instante captura el chaleco completamente, el instante posterior apenas captura la mitad de éste, produciendo un cambio en las coordenadas del centro del objeto y en su área.

3.4.1 Suavizado de imagen

En un primer momento se ha optado por crear un método *suavizaCoordenadas* que, para cada coordenada a enviar, se suaviza realizando una media de las últimas 5 coordenadas que se enviaron. Se observó que un *buffer* de tamaño 5 para 24 *frames* que se extraen por segundo son pocos, con que se ha aumentado hasta 15 y se han obtenido mejores resultados, aunque no concluyentes.

Posteriormente, se decide emplear una función *accumulateWeighted* de *OpenCV* que nos realiza una imagen promedio. La función recibe la imagen actual que se acumula con un valor *alpha* (que indica el peso que recibe dicho frame con respecto al acumulado). En esta ocasión la mejora de la estabilidad de las coordenadas es notable, aunque se siguen produciendo movimientos involuntarios.

3.4.2 Detección de movimientos

Otra de las técnicas más empleadas a la hora de buscar un objeto en una imagen es a través de su movimiento. Se debe hacer un promedio, para obtener una imagen *fondo,* que al aplicar la diferencia con la imagen actual, nos resulte una imagen final sólo con el objeto en movimiento [**12**].

En principio esta técnica se ha incluido al reconocimiento por color para afinar más la posición y movimiento del objeto. Dando lugar un mayor suavizado de la imagen y mayor fiabilidad, ya que reduce los ruidos ambientales existentes producidos por la presencia de objetos de color similar al del guía.

Por otro lado, debido a que PACA estará en continuo movimiento y el fondo en continuo cambio, tampoco se producirá una mejora razonable en la detección del objeto. Ésta técnica se suele usar para cámaras estáticas, que no están moviéndose continuamente.

3.4.3 Movimiento en función a un plano

Se colocan tres puntos en el chaleco formando un triángulo. El algoritmo estudia la posición de estos puntos deduciendo un vector normal como se puede ver en la figura 3.8. El vector normal ofrece información sobre la dirección del desplazamiento del guía.

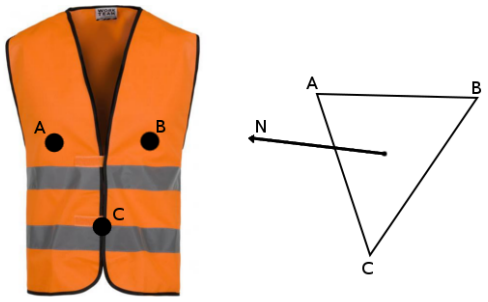


Figura 3.8: Servicio TRPI en pruebas

Por otro lado, si se estudia la distancia de los puntos entre sí se puede obtener la distancia real entre el guía y PACA para saber si éste se aleja o se acerca deduciendo el sentido de su movimiento.

Éste método no se ha podido probar debido a la falta de tiempo, pero podría ofrecer mejores resultados si se usa en conjunto a los métodos anteriores.

3.5 Algoritmo como servicio

Como ya se ha comentado anteriormente, para reducir la interacción humana con PACA y el nuevo algoritmo de seguimiento TRPI, es necesario hacer que éste se lance de forma automática creando algún fichero *log* para posteriormente poder estudiar su comportamiento.

Debido a que el algoritmo se alojará en el sistema operativo Raspbian de Raspberry Pi, se debe estudiar su gestor de servicios. En éste caso, Raspbian ya no usa el antiguo *SystemV* **[13]** sino que implementa la nueva

versión del gestor de procesos, *Systemd* [**14**]. A pesar de algunos cambios conceptuales como el uso de *targets* en vez de *niveles de arranque*, ofrece una gestión de procesos y servicios del sistema muy similar a *SystemV*.

```
pi@TRPI-Paca:~ $ sudo systemctl status trpi.service
● trpi.service - Tracking Raspberry Pi TRPI algoritmo de conducción autónoma.
   Loaded: loaded (/etc/systemd/system/trpi.service; enabled)
   Active: failed (Result: timeout) since Tue 2017-06-20 22:38:30 UTC; 4min 51s ago
  Process: 646 ExecStart=/home/pi/TRPI-master/TRPI (code=killed, signal=TERM)

Jun 20 22:37:03 TRPI-Paca TRPI[646]: - Creating TRPI raspberry
Jun 20 22:37:03 TRPI-Paca TRPI[646]: Loading symbols ...... Done.
Jun 20 22:37:03 TRPI-Paca TRPI[646]: Done.
Jun 20 22:38:14 TRPI-Paca TRPI[646]: Recibiendo de paca2 bytes: 3
Jun 20 22:38:14 TRPI-Paca TRPI[646]: Enviando a paca Orden enviada
Jun 20 22:38:30 TRPI-Paca TRPI[646]: Recibiendo de paca287 bytes: 3
Jun 20 22:38:30 TRPI-Paca TRPI[646]: Enviando a paca Orden enviada
Jun 20 22:38:30 TRPI-Paca systemd[1]: trpi.service start operation timed out. Terminating.
Jun 20 22:38:30 TRPI-Paca systemd[1]: Failed to start Tracking Raspberry Pi TRPI algoritmo de conducción autónoma..
Jun 20 22:38:30 TRPI-Paca systemd[1]: Unit trpi.service entered failed state.
```

Figura 3.9: Servicio TRPI en pruebas

Para la creación del servicio TRPI, se ha ubicado el ejecutable del algoritmo en la ruta */home/pi/TRPI-master/* con permisos de ejecución para todos los usuarios (775). Además, se ha creado un fichero *trpi.service* en la ruta */etc/systemd/system/* para que *Systemd* se encargue de crear el servicio y lanzarlo de forma automática con el inicio del sistema (figuras 3.9 y 3.10).

```
systemd-analyze                 systemd-notify
systemd-ask-password            systemd-nspawn
systemd-cat                     systemd-path
systemd-cgls                    systemd-run
systemd-cgtop                   systemd-stdio-bridge
systemd-delta                   systemd-tmpfiles
systemd-detect-virt             systemd-tty-ask-password-agent
systemd-escape
pi@TRPI-Paca:~/Desktop $ sudo systemctl status tr
triggerhappy.service  trpi.service
pi@TRPI-Paca:~/Desktop $ sudo systemctl status trpi.service
● trpi.service - Tracking Raspberry Pi TRPI algoritmo de conducción autónoma.
   Loaded: loaded (/etc/systemd/system/trpi.service; enabled)
   Active: active (running) since Wed 2017-06-21 01:49:00 UTC; 9min ago
 Main PID: 641 (TRPI)
   CGroup: /system.slice/trpi.service
           └─641 /home/pi/TRPI-master/TRPI

Jun 21 01:55:36 TRPI-Paca TRPI[641]: Recibiendo de paca123 bytes: 3
Jun 21 01:55:36 TRPI-Paca TRPI[641]: Enviando a paca Orden enviada
Jun 21 01:55:39 TRPI-Paca TRPI[641]: Recibiendo de paca13 bytes: 3
Jun 21 01:55:39 TRPI-Paca TRPI[641]: Enviando a paca Orden enviada
Jun 21 01:55:41 TRPI-Paca TRPI[641]: Recibiendo de paca15 bytes: 3
Jun 21 01:55:41 TRPI-Paca TRPI[641]: Enviando a paca Orden enviada
Jun 21 01:55:44 TRPI-Paca TRPI[641]: Recibiendo de paca9 bytes: 3
Jun 21 01:55:44 TRPI-Paca TRPI[641]: Enviando a paca Orden enviada
Jun 21 01:55:49 TRPI-Paca TRPI[641]: Recibiendo de paca15 bytes: 3
Jun 21 01:55:49 TRPI-Paca TRPI[641]: Enviando a paca Orden enviada
```

```
1  [Unit]
2  Description= Tracking Raspberry Pi TRPI algoritmo de
3  After=sockets.target network.target
4
5  [Service]
6  Type=simple
7  ExecStart=/home/pi/TRPI-master/TRPI
8  Restart=on-abort
9
10 [Install]
11 WantedBy=multi-user.target
12
```

Figura 3.10: Servicio TRPI ejecutándose y su correspondiente fichero trpi.service

Capítulo 4
Control de funcionamiento mediante APP móvil

La APP móvil existente, que se puede observar en la figura 4.1, se ha mejorado tanto en usabilidad como en funcionamiento con el objetivo de hacerla más fácil de usar y además como elemento de control de PACA en el estado automático.

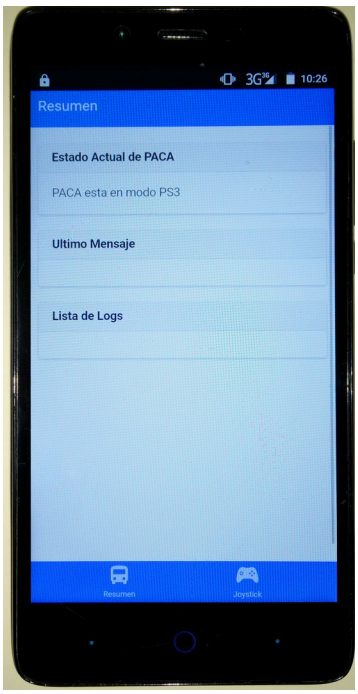


Figura 4.1: Versión anterior APP móvil

4.1 Interfaz de usuario

Se ha creado una pantalla principal, donde el usuario puede hacerse una idea del funcionamiento general de la APP. Se ha añadido un botón de información y un botón que nos devuelve al inicio en la barra superior. Algunos de las mejoras se pueden observar en las figuras siguientes 4.2 y 4.3

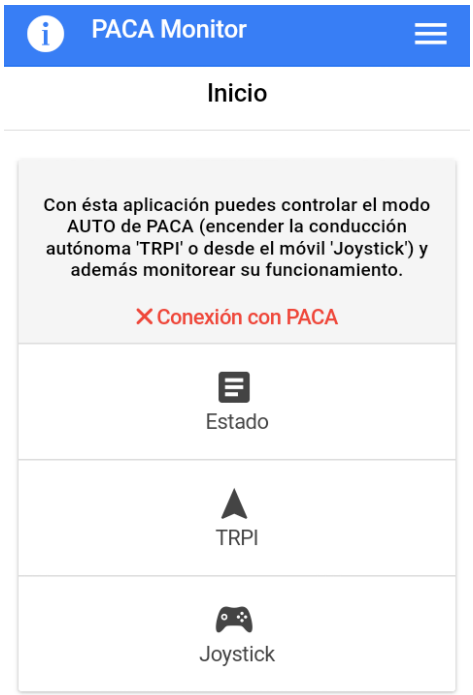


Figura 4.2: Pantalla principal PACA Monitor

Figura 4.3: Botón de información PACA Monitor

4.2 Interfaz con sistema maestro

Como elemento de arbitraje, el móvil ahora envía una serie de mensajes de control que indican a PACA a qué dispositivo hacer caso (APP móvil o TRPI). Para ello se ha añadido los campos “CONTROL” e “ID” a los mensajes que anteriormente recibía PACA (figura 4.4). Se ha tenido que modificar el *firmware* del sistema maestro para que también pueda recibir este tipo de mensajes y gestionarlos de forma correcta.

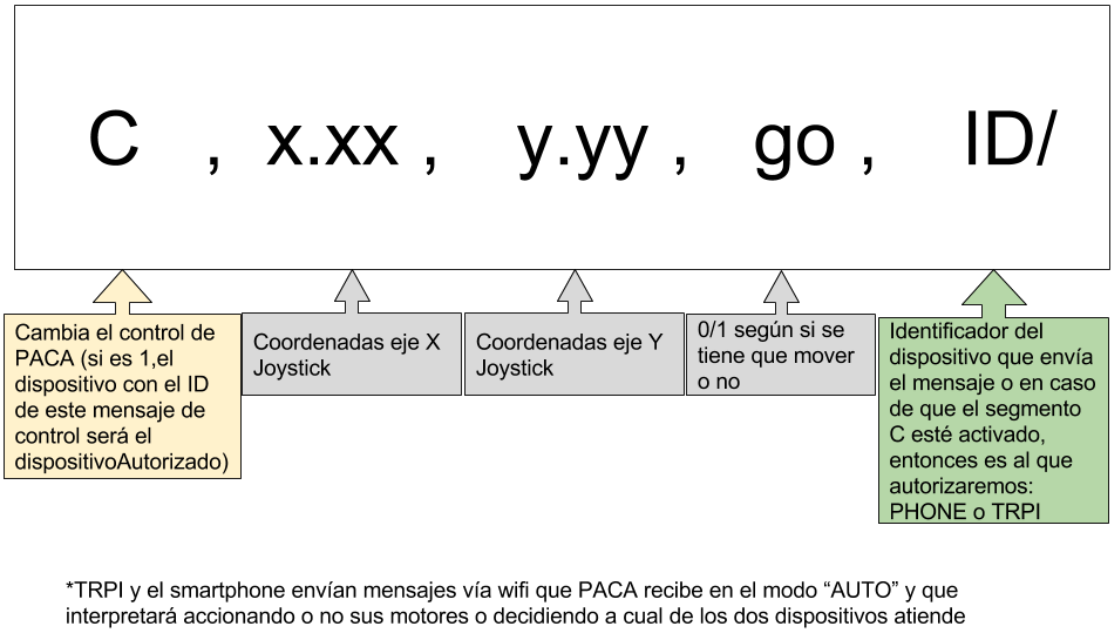


Figura 4.4: Codificación actual mensajes

Este fue el proceso más costoso en el tiempo de todo el desarrollo del proyecto debido a que se estaban poniendo tres sistemas aislados a funcionar en conjunto y se tuvo que monitorizar los mensajes de los tres dispositivos que intervienen (TRPI, APP móvil y PACA). Además, debido a la complejidad del funcionamiento del *ESP8266*, tuvo que ser formateado y reprogramado varias veces.

Finalmente, se ha logrado que la APP sirva de dispositivo intermedio y gestione la autorización (para el joystick del propio móvil o TRPI) en el nuevo firmware de PACA (figura 4.5).

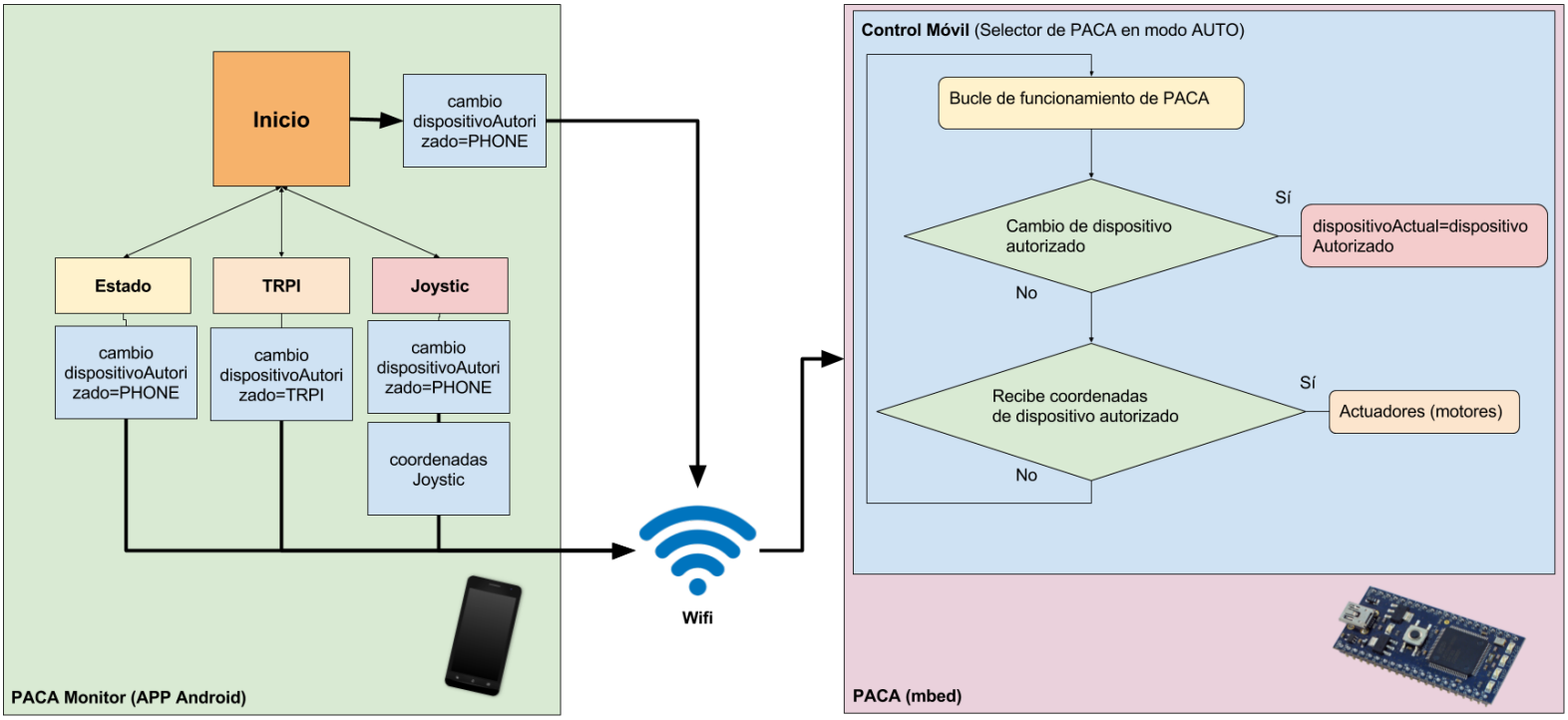


Figura 4.5: Diagrama funcionamiento del nuevo firmware PACA y APP móvil

Capítulo 5
Conclusiones y líneas futuras

Al finalizar el trabajo realizado, se ha llegado a la conclusión de que, en mayor o menor medida, todos los objetivos principales propuestos han sido realizados con éxito. Como resultado, a falta de realizar algunas pruebas de rendimiento exhaustivo, se ha obtenido un producto que podría usarse como dispositivo de carga en un entorno real.

En cuanto a experiencias y habilidades adquiridas, se deben resaltar los conocimientos a nivel hardware, donde se tuvo más dificultad a la hora de abordar el desarrollo del nuevo sistema.

Como principales objetivos alcanzados caben destacar:

- *Implementación de un sistema secundario de conducción autónoma (TRPI) para PACA.*
- *Mejora de la APP móvil.*
- *Mejora y adición del nuevo sistema de dispositivos al firmware de PACA.*
- *Instalación y configuración del módulo Wifi.*

A lo largo del transcurso del trabajo se han ido planteando posibles mejoras o implementaciones a realizar en un futuro para mejorar y refinar el comportamiento del algoritmo TRPI, como por ejemplo:

- *Añadir un reconocimiento de plano y posición 3D del guía, como se ha detallado mejor en el capítulo 3.4.3.*
- *Añadir un símbolo reconocible al chaleco. Evita que el guía sea confundido con las formas y colores presentes en el entorno en el que*

trabaja PACA.

- *Añadir un sistema GPS que, junto a TRPI y el uso de algoritmos heurísticos, dotarían de autonomía completa al dispositivo.*
- *Finalmente, estudiar el profiling del algoritmo para ser implementado en hardware, reduciendo así los tiempos de ejecución de las partes más complejas del algoritmo.*

Por otra parte, también se ha planteado, de cara a un producto más acabado a nivel mecánico y electrónico, sustituir las ruedas actuales por unas de oruga para mejorar su desplazamiento por cualquier tipo de terreno. Además, modularizar el sistema de electrónica principal, facilitando posibles cambios en el diseño del carro y ofrece cierto nivel de abstracción. Por último, implementar en el Smartphone, el cambio de estados del carro, para que funcione remotamente y no tener que acercarse al panel principal del carro.

Capítulo 6
Summary and Conclusions

At the end of work, it has concluded that, all the goals established in the beginning have been successfully completed, resulting in a product that could be used in a real environment after realize a few more exhaustive tests.

In terms of experiences and skills acquired, should be highlighted the knowledge acquired at hardware level. It was more difficult solve problems at hardware level than at software developing the new system.

Main highlighted objectives are achieved:

- *Added an autonomous drive secondary system (TRPI) to PACA.*
- *Improved Smartphone APP.*
- *Improved and added to PACA's firmware the new system.*
- *To install and to set up Wifi module.*

Throughout the course of the project, we have been considering possible improvements or implementations to perform in the future to improve TRPI behaviour, such as:

- *Add a plane and 3D position recognition of the guide. As it showed in the chapter 3.4.3.*
- *Add a symbol to the guide in order to avoid confusing it with the environment.*
- *Add a GPS system, that works with TRPI and heuristic algorithms offering completely autonomous drive system.*
- *Finally, studying and profiling TRPI due to make a hardware implementation and reduce computational load.*

Moreover, it has also been raised, the replacement of current wheels adding ones caterpillar wheels, which would improve PACA's movement over all ground types. Another idea was modulate the main electronic system, thus facilitating futures changes in the design, while this part stays unchanged. Finally was proposed the idea of develop a system to change PACA's state using the Smartphone.

Capítulo 7
Presupuesto

En este capítulo se desglosarán todas las partes que componen el sistema tanto a nivel *hardware* como *software*, indicando un precio aproximado del componente, que servirá como guía a la hora de replicar el carro.

El **coste total** aproximado es **5.170€**

7.1 Componentes mecánicos

Concepto	Unidades	Precio Unidad
Diseño y fabricación de un chasis	1	1.500 €
Neumáticos y llantas	2	6 €
Ruedas locas	2	6 €
Batería	2	300 €
Selector de estados	1	10 €
Seta de seguridad	1	40 €
Interruptor	1	12 €
Total	--------	1.874,00 €

Tabla 7.1: Componentes mecánicos

7.2 Electrónica de potencia

Concepto	Unidades	Precio Unidad
Controlador electrónico de Potencia (EM-115 DC Motor Control Unit 12-36V 25A 4Quad)	2	160 €
Motores (PM90LWS)	2	400 €
Placa de conversión de voltaje (3-5-12V)	1	40
Total	--------	600 €

Tabla 7.2: Electrónica de potencia

7.3 Componentes de interfaz

Concepto	Unidades	Precio Unidad
Joystick Manual (CH Product HF Series Hall effect joystick)	1	100 €
Mando de control PS3	1	25 €
Dongle Bluetooth	1	13 €
Módulo Wifi ESP8266	1	10 €
Microcontrolador MBed	1	45 €
Raspberry Pi v3	1	50€
Playstation eye PS3	1	15€
Total	--------	258 €

Tabla 7.3: Componentes de interfaz

7.4 Desarrollo software

Concepto	Unidades	Precio Unidad
Diseño de aplicación móvil Android	1	850 €
Diseño de software de control MBed	1	800 €
Diseño de software de seguimiento TRPI	1	650€
Total	--------	2.300 €

Tabla 7.4: Desarrollo software

7.5 Componentes varios

Concepto	Unidades	Precio Unidad
Metacrilato	1	48 €
Caja estanca de protección electrónica	1	60 €
Cableado	1	30 €
Total	--------	138 €

Tabla 7.5: Componentes varios

Haciendo una comparación con los presupuestos anteriores se puede percibir un incremento en el costo del desarrollo del software debido en parte a que aún el carro se encuentra en fase de desarrollo, pero también refleja, como en muchos proyectos la curva costo hardware/software de la industria informática.

Capítulo 8
Apéndice 1: Códigos implementados

8.1 Algoritmo TRPI

interfazPaca.hpp

```
/*************************************************************************
*
* interfazPaca.hpp
*
*************************************************************************
*
* AUTOR Carlos Gregorio Martín Pérez
*
* FECHA 20-06-2017
*
* DESCRIPCION Clases y métodos para establecer una interfaz de comunicación
* entre TRPI y PACA. Sólo se han incluido las partes importantes
*
*************************************************************************/
class PacaConection{
	private:
		struct sockaddr_in paca; //struct con la información del server
		int pacaSocket;
		int timeAnt; //tiempo de envío del último mensaje
		int timeAct; //tiempo de envío del mensaje actual, se intenta no saturar al mbed
	public:
		PacaConection(const char* ipaddress){ //constructor
			//timeAnt = clock();
			paca.sin_family = AF_INET;
			paca.sin_port = htons(8888);
			paca.sin_addr.s_addr = inet_addr(ipaddress);
			pacaSocket = socket(AF_INET,SOCK_STREAM,0);
		  connect(pacaSocket, (struct sockaddr *)&paca, sizeof(paca));
		}
		int sendPaca(char* mensaje){ //envía el mensaje a PACA
			printf("Enviando a paca");
			if(write(pacaSocket,mensaje,strlen(mensaje))){
```

```
                        return 0;
                }
                else
                        return -1;
        }
        int recivePaca(){
                printf("Recibiendo de paca");
                char buffer[1024];
                int bytes = recv(pacaSocket, buffer, 1024, 0);
                printf("%d bytes: %d\n", bytes, atoi(buffer));
                return atoi(buffer);
        }
};
```

trpi.hpp

```
/*********************************************************************
*
* trpi.hpp
*
*********************************************************************
*
* AUTOR Carlos Gregorio Martín Pérez
*
* FECHA 14-07-2017
*
* DESCRIPCION Clases y métodos TRPI para la detección y tracking de un objeto
* usando las librerías OpenCV (incluye ultimas pruebas y posibles mejoras
* sin acabar). Sólo se han incluido las partes más importantes
*
*********************************************************************/
//HSV color scheme range
struct HSVrange{
        //Hue Matiz
        int H_MIN;
        int H_MAX;
        //Saturation
        int S_MIN;
        int S_MAX;
        //Value, brightness
        int V_MIN;
        int V_MAX;
};
//Class for save symbols info
class Symbol {
        public:
                Mat img;
                string name;
};
//Class for track a guide and symbols
class Trpi{
        private:
                //matrix storage for HSV image
                Mat HSV;
                //Matrix to store actual frame
                Mat camera;
```

```
		//matrix storage for binary threshold image
		Mat thresholded;
		//video capture object
		VideoCapture cap;
		//Localización del objeto
		int x;
		int y;
		//it saves the value of the last area in order to know the side of the guide
		double areaant;
		//max number of objects to be detected in frame
		int max_num_objects;
		//minimum and maximum object area
		int min_object_area;
		int max_object_area;

		//buffer for save recents coord.
		double coord_acumuladas[15][2];

		HSVrange objectColor; //guide color
		Symbol symbols[7];

	public:
		Trpi(HSVrange guideColor, int frame_width=640, int frame_height=480){
			objectColor.H_MIN = guideColor.H_MIN;
			objectColor.H_MAX = guideColor.H_MAX;
			objectColor.S_MIN = guideColor.S_MIN;
			objectColor.S_MAX = guideColor.S_MAX;
			objectColor.V_MIN = guideColor.V_MIN;
			objectColor.V_MAX = guideColor.V_MAX;
			//location of the guide
			x=0, y=0;
			areaant=0.0;

			//max number of objects to be detected in frame
			max_num_objects=50;
			//minimum and maximum object area
			min_object_area = 10000;
			//max area for the guide, if guide area is greater, PACA stops
			max_object_area = frame_height*frame_width/2;

			//open camera 0 default, 1.. extra usb
			cap.open(1);
			//adjust the resolution of the camera and enable the automatic exposure in
case it doesnt bring it enabled by default
			cap.set(CV_CAP_PROP_AUTO_EXPOSURE, 1);
			cap.set(CV_CAP_PROP_FRAME_WIDTH,frame_width);
			cap.set(CV_CAP_PROP_FRAME_HEIGHT,frame_height);

			if (readRefImages(symbols) == -1) {
				printf("Error reading reference symbols\n");
			}
		}
		double * suavizaCoordenadas (double * coordenadas){
			if (sizeof (coordenadas)/sizeof (*coordenadas)==1){ //si hay dos coordenadas
				//vamos a tomar otros valores anteriores para suavizar el movimiento
de PACA
				double acumuladosentido =  0.0;
```

```
                        double acumuladodireccion =  0.0;
                        for (int i=0; i++;i<15){
                                acumuladodireccion += coord_acumuladas[i][0];
                                acumuladosentido += coord_acumuladas[i][1];
                        }
                        acumuladodireccion += coordenadas[0];
                        acumuladosentido += coordenadas[1];
                        acumuladodireccion = acumuladodireccion/15;
                        acumuladosentido = acumuladosentido/15;
                        //actualizamos las coordenadas acumulados, desplazamos los valores e
insertamos el que nos han pasado
                        for (int i=14;i--;i>0){
                                        coord_acumuladas[i][0] = coord_acumuladas[(i-1)][0];
                                        coord_acumuladas[i][1] = coord_acumuladas[(i-1)][1];

                        }
                        //añadimos en ultima posición las coordenadas actuales a suavizar
                        coord_acumuladas[0][0] = coordenadas[0];
                        coord_acumuladas[0][1] = coordenadas[1];

                        //ahora ponemos en las coordenadas que nos han pasado, los valores de
media
                        coordenadas[0] = acumuladodireccion;
                        coordenadas[1] = acumuladosentido;
                        return coordenadas;
                }
        }

        void morphOps(Mat &thresh){
                //create structuring element that will be used to "dilate" and "erode"
image.
                Mat erodeElement = getStructuringElement( MORPH_RECT,Size(10,10));
                //dilate with larger element so make sure object is nicely visible
                Mat dilateElement = getStructuringElement( MORPH_RECT,Size(35,35));

                erode(thresh,thresh,erodeElement);
                erode(thresh,thresh,erodeElement);
                erode(thresh,thresh,erodeElement);

                dilate(thresh,thresh,dilateElement);
                dilate(thresh,thresh,dilateElement);
                dilate(thresh,thresh,dilateElement);
        }

        string trackFilteredObject(int &x, int &y, Mat thresholded, Mat &cameraFeed){

                string output= string();
                Mat temp;
                thresholded.copyTo(temp);
                //these two vectors needed for output of findContours
                vector< vector<Point> > contours;
                vector<Vec4i> hierarchy;
                //find contours of filtered image using openCV findContours function
                findContours(temp,contours,hierarchy,CV_RETR_CCOMP,CV_CHAIN_APPROX_SIMPLE );
                //use moments method to find our filtered object
                double refArea = 0.0;
                bool objectFound = false;
```

```
                    int xtmp, ytmp = 0.0; /*VARIABLE TEMPORAL PARA SABER A DONDE SE MUEVE EL
OBJETO*/
                    if (hierarchy.size() > 0){ //si se ha encontrado algun objeto y además han
pasado almenos 3milisegundos
                            int numObjects = hierarchy.size();
                            //if number of objects greater than MAX_NUM_OBJECTS we have a noisy
filter
                            if(numObjects<max_num_objects){
                                    for (int index = 0; index >= 0; index = hierarchy[index][0]) {
                                            Moments moment = moments((cv::Mat)contours[index]);
                                            double area = moment.m00;

                                            //if the area is less than 20 px by 20px then it is
probably just noise
                                            //if the area is the same as the 3/2 of the image
size, probably just a bad filter
                                            //we only want the object with the largest area so we
safe a reference area each
                                            //iteration and compare it to the area in the next
iteration.
                                            if(area>min_object_area  &&  area<max_object_area  &&
area>refArea){ /*OBJECT FOUND*/
                                                    xtmp = moment.m10/area;
                                                    ytmp = moment.m01/area;
                                                    objectFound = true;
                                                    refArea = area;
                                                    double coordenadas[2];
                                                    coordenadas[0]      =      fmod((double)(ytmp-
y)/100.00,1.00);
                                                    coordenadas[1]        =        round((areaant-
area))/10000.00;
                                                    double * coordenadas_suavizadas;
                                                    coordenadas_suavizadas                          =
suavizaCoordenadas(coordenadas);
                                                    Truncate(coordenadas[0]);//dejamos   solo   dos
decimales
                                                    Truncate(coordenadas[1]);//dejamos   solo   dos
decimales

       output+="0,"+doubleTostring(coordenadas_suavizadas[0])
+","+doubleTostring(coordenadas_suavizadas[1])+",1,TRPI/";

                                                    y=ytmp;
                                                    x=xtmp;
                                                    areaant = area;
                                            }
                                    }

                            }else{
                                    output+="0,0.00,0.00,0,TRPI/"; //Demasiados objetos en escena,
hay mucho ruido.
                            }
                    }
                    return output;
            }

            string checkSymbols(Symbol * symbols,Mat &camera){  //Le pasamos los símbolos y el
```

```
frame donde ha de detectarlos
                    Mat new_image;
                    Mat greyImg;
                    Mat canny_output;
                    vector<vector<Point> > contours;
                    vector<Point> approxRect;
                    vector<Vec4i> hierarchy;
                    cvtColor(camera, greyImg, CV_RGB2GRAY);
                    GaussianBlur(greyImg, greyImg, Size(9, 9), 2, 2);

                    /// Detect edges using canny
                    Canny(greyImg, canny_output, 0, 0 * 3, 3); //50→porcentaje mínimo de
threshold
                    findContours(canny_output, contours, hierarchy,
CV_RETR_TREE,CV_CHAIN_APPROX_SIMPLE, Point(0, 0));
                    for (size_t i = 0; i < contours.size(); i++) {
                           approxPolyDP(contours[i], approxRect,arcLength(Mat(contours[i]),
true) * 0.05, true);
                           if (approxRect.size() == 4) {
                           float area = contourArea(contours[i]);
                                  if (area > 1000) {
                                         std::vector<cv::Point2f> corners;

                                         vector<Point>::iterator vertex;
                                         vertex = approxRect.begin();
                                         //vertex++;
                                         circle(camera, *vertex, 2, Scalar(0, 0, 255), -1, 8,
0);
                                         corners.push_back(*vertex);
                                         vertex++;
                                         circle(camera, *vertex, 2, Scalar(0, 0, 255), -1, 8,
0);
                                         corners.push_back(*vertex);
                                         vertex++;
                                         circle(camera, *vertex, 2, Scalar(0, 0, 255), -1, 8,
0);
                                         corners.push_back(*vertex);
                                         vertex++;
                                         circle(camera, *vertex, 2, Scalar(0, 0, 255), -1, 8,
0);
                                         corners.push_back(*vertex);
                                         Moments mu;
                                         mu = moments(contours[i], false);
                                         Point2f center(mu.m10 / mu.m00, mu.m01 / mu.m00);
                                         sortCorners(corners, center);
                                         // Define the destination image
                                         Mat correctedImg = ::Mat::zeros(195, 271, CV_8UC3);
                                         // Corners of the destination image
                                         std::vector<cv::Point2f> quad_pts;
                                         quad_pts.push_back(Point2f(0, 0));
                                         quad_pts.push_back(Point2f(correctedImg.cols, 0));
                                         quad_pts.push_back(Point2f(correctedImg.cols,
correctedImg.rows));
                                         quad_pts.push_back(Point2f(0, correctedImg.rows));
                                         // Get transformation matrix
                                         Mat transmtx = getPerspectiveTransform(corners,
quad_pts);
```

```
                                        // Apply perspective transformation
                                        warpPerspective(camera,     correctedImg,     transmtx,
correctedImg.size());

                                        Mat correctedImgBin;
                                        cvtColor(correctedImg, correctedImgBin, CV_RGB2GRAY);
                                        //equalizeHist(correctedImgBin, correctedImgBin);
                                        correctedImgBin.copyTo(new_image);
                                        threshold(correctedImgBin,  correctedImgBin,  140,  255,
0);
                                        double minVal,maxVal,medVal;
                                        minMaxLoc(new_image, &minVal, &maxVal);
                                        medVal=(maxVal-minVal)/2;
                                        threshold(new_image, new_image, medVal, 255, 0);
                                        Mat diffImg;
                                        int match, minDiff, diff;
                                        minDiff = 12000;
                                        match = -1;
                                        for (int i = 0; i < 6; i++) {
                                                diffImg = symbols[i].img-new_image;

                                                bitwise_xor(new_image, symbols[i].img, diffImg,
        noArray());
                                                diff = countNonZero(diffImg);
                                                if (diff < minDiff) {
                                                        minDiff = diff;
                                                        match = i;
                                                }
                                        }
                                        if (match != -1) {
                                                return symbols[match].name;
                                        }
                                }
                        }
                }
                return std::string();
        }

        string seguirGuia(){
                while(1){//Se ejecuta hasta que encuentre un objeto o el tiempo de espera se agote
                        cap >> camera; //lee el siguiente frame
                        //Guarda en HSV la imagen original transformada al esquema de colores HSV
                        cvtColor(camera,HSV,COLOR_BGR2HSV);
                        //Filtra el rango e olor en la imagen hsv y el resultado en threshold que
indicará a presencia del guía

inRange(HSV,Scalar(objectColor.H_MIN,objectColor.S_MIN,objectColor.V_MIN),Scalar(objectColor.H_MAX,
objectColor.S_MAX,objectColor.V_MAX),thresholded);
                        morphOps(thresholded);
                        //pasamos  threshold  a  la  función  para  obtener  su  posición  dentro  de  la
imagen
                        //y mostrar por pantalla la valocidad de movimiento y la dirección del guía
                        string outputSymbols = checkSymbols(symbols,camera);
                        std::cout<<"Símbolo etectado → "<<outputSymbols<<endl;
                        string outputObject = trackFilteredObject(x,y,thresholded,camera);
                        if  (outputSymbols=="GoTrack"  &&  outputSymbols!="0,0.0,0.0,0,TRPI/"){  //GUÍA
si esá presente el cartel GO a no ser que haya una señal STOP
```

```
                        return outputObject;
                    }else   if(outputSymbols!=std::string()){ //SI NO => SIGNO
                        return outputSymbols;
                    }
                }
                return "";
        }
};
```

main.cpp

```
/**************************************************************************
*
* main.cpp
*
**************************************************************************
*
* Carlos Gregorio Martín Pérez
*
* 06-07-2017
*
* Código principal del funcionamiento del algoritmo, aquí se
* emplean los ficheros fuentes que se ven con anterioridad
*
**************************************************************************/
#include "interfazPaca.hpp"
#include "trpi.hpp"

const char* IPADRESS = "192.168.4.1"; //Indica la ip del receptor del mensaje (puerto 8888)
const int H_MIN = 0;
const int H_MAX = 20;
const int S_MIN = 125;        //indica el rango de colores HSV
const int S_MAX = 256;        //que TRPI interpretará como guía
const int V_MIN = 97;
const int V_MAX = 256;

int main(int argc, char** argv) {
        cout<<"Initializing"<<endl;
        char* orden; //almacena las ordenes a enviar a PACA
        //definimos los valores de color HSV para reconocer al guía (Pelota ping pong naranja)
        clock_t timeAnt = clock()/10000;
        clock_t timeAct = clock()/10000;
        struct HSVrange guia;
        guia.H_MIN = H_MIN;
        guia.H_MAX = H_MAX;
        guia.S_MIN = S_MIN;
        guia.S_MAX = S_MAX;
        guia.V_MIN = V_MIN;
        guia.V_MAX = V_MAX;

        cout<<"Creating objects..."<<endl;
        cout<<"- Creating PacaConection"<<endl; PacaConection paca(IPADRESS);
        cout<<"- Creating TRPI raspberry"<<endl; Trpi raspberry(guia);
        cout<<"Done."<<endl;
```

```
        while (1){ //bucle principal
                        //Comprobamos si PACA quiere que TRPI funcione o no
                        int controlPaca = paca.recivePaca();

                        if(controlPaca==3){ //Si PACA está en MODO AUTO encendemos el seguimiento
                                string strorden = raspberry.seguirGuia(); //capturamos la orden de
TRPI a enviar

                                if (strorden != ""){
                                        orden =new char[strorden.size() + 1];
        //la guardamos en un char* para enviarla
                                        strcpy(orden, strorden.c_str());

                                        if(paca.sendPaca(orden) == 0)
        //enviamos la posición del joystick
                                                cout<<" Orden enviada\t"<<orden<<endl;
                                        else
                                                cout<<" x ERROR: sendPaca"<<endl;
                                }
                                delete orden;
                        }
                        sleep(1);
        }
}
```

8.2 Firmware PACA

main.cpp

```
/*************************************************************************
*
* main.cpp
*
*************************************************************************
*
* Carlos Gregorio Martín Pérez
*
* 04-06-2017
*
* Código principal de funcionamiento de PACA. Este fichero junto
* a Wifi.cpp son los que se han modificado para la realización de este TFG
* Se ha extraído solo los segmentos modificados (función loop())
*
*************************************************************************/
void loop(){
    TestShell();                                                    //Inicializamos la conexion
con el mando PS3
    while(1){
        selector(d1,d2);                                             //Detectamos el modo del
selector
        checkCambioModo(modo);                                      //Comprobamos los cambios
de modo para resetear las variables xAct e yAct
        switch (modo) {                                             //Comparamos el modo para
saber que hacer
```

```
            case JOYSTICK:
                //Pasamos el boton del Joystick negado porque esta a baja
                //joystick(ejeXJoystick, ejeYJoystick, 0, 0, !botonJoystick);
                joystick(ejeXJoystick, ejeYJoystick, Xcentro, Ycentro, !botonJoystick);
                led2 = true;
                led3 = true;
                break;
            case AUTO:
                //automatico(); //filo guiado
                led2 = false;
                led3 = true;
                //Si recibimos un mensaje nuevo con nuevos valores
                if(mensajeCompletado()){
                    //float *ejes = getEjesWifi(); //Nesto
                    float *ejes = decodificarMensaje(); //Carlos
                    if (ejes!=NULL){ //si es NULL, es de control o proviene de un dispositivo sin
autoridad
                        ejesAuto[0] = ejes[0];  ejesAuto[1] = ejes[1];  ejesAuto[2] = ejes[2];
                    }
                }
                //movemos PACA con las coordenadas anteriores o las nuevas
                         joystick(ejesAuto[0],  ejesAuto[1],  CENTRO_PS3_Y_AUTO,  CENTRO_PS3_Y_AUTO,
ejesAuto[2]);
                break;
            case OFF:
                break;
            case PS3:
                led2 = true;
                led3 = false;
                //Detectamos los eventos del mando PS3
                RunPS3();
                //Pasamos los valores recogidos del mando PS3 a la funcion Joystick
                joystick(getEjeXPS3(), getEjeYPS3(), CENTRO_PS3_Y_AUTO, CENTRO_PS3_Y_AUTO,
getBotonCtrlPS3());
                break;
            default:
                break;
        }//Final switch
        /*Para depurar los ejes
        printf("ejeX: %.2f       ejeY:%.2f", ejeXJoystick,ejeYJoystick);
        printf(!botonJoystick ? "true" : "false");*/
    }//Final while
}//Final metodo loop()
```

Wifi.cpp

```
/*************************************************************************
*
* Wifi.cpp
*
*************************************************************************
*
* Carlos Gregorio Martín Pérez
*
* 04-06-2017
```

```
*
* Parseamos el mensaje que hemos recibido, comprobamos si se
* cambia el control de dispositivo y extraemos los ejes como
* en la funcion getEjeswifi(). Devuelve null en caso de que el
* mensaje recibido sea descartado (Porque no proviene del
* dispositivo con el control).
* Se han exraído solo los segmentos que han sido añadidos (función decodificarMensaje())
*
**************************************************************************/
float* decodificarMensaje(){
    float *pointerEjes=NULL;
     string segmento[6]; //almacena los segmentos del mensaje CONTROL,X.XX,Y.YY,GO,ID como strings
cada uno (5 segmentos)
    int i=0;
     //primero comprobamos que el mensaje tenga un tamaño máximo, si lo sobrepasa se trata de un
mensaje erroneo con que se descarta (errores del dongle)
     //if (mensajeRecibido.length()>0){ //un mensaje esperado no tiene más de 20 caracteres, si se
recibe un mensaje de este tipo entonces hay que descartarlo
        //Mientras encontremos comas, extraemos los valores
          while ((pos = mensajeRecibido.find(delimiter)) != string::npos && i<=5) { //mientras se
sigan leyendo delimitadores o aun no se hayan extraído 5 campos
            segmento[i] = mensajeRecibido.substr(0, pos);
            mensajeRecibido.erase(0, pos + delimiter.length());
            i++;
        }
        //Guardamos el ultimo valor (No existen comas al final del mensaje)
        segmento[i] = mensajeRecibido;
        if (i<=5){ //comprobamos que no se salga del rango para evitar segmentation fault
            //Decodificamos el mensaje por si hay que cambiar de dispositivo
            string msg_control=segmento[0];            //guardamos el primer campo "Control"
            string msg_id=segmento[4];                 //y el ultimo "ID" del mensaje recibido
            printf("\n--Control %s", msg_control.c_str());
            printf(" ID %s\t", msg_id.c_str());
             if (msg_control=="1"){              //si control está activado es que va a cambiar el
dispositivo que tiene el control
              dispositivoAutorizado = msg_id;         //actualizamos al nuevo dispositivo
              printf("**Control a %s", dispositivoAutorizado.c_str());
             }else if (dispositivoAutorizado==msg_id){//si no es mensaje de control y proviene del
dispositivo que tiene el control actualment
                //pasamos los valores de segmento a ejesWifi
                printf("##Dispositivo %s enviando ejes", dispositivoAutorizado.c_str());
                pointerEjes = ejesWifi; //reservamos el espacio necesario para los ejes
                ejesWifi[0] = atof(segmento[1].c_str()); //ejeX
                ejesWifi[1] = atof(segmento[2].c_str()); //ejeY
                ejesWifi[2] = atof(segmento[3].c_str()); //control del boton
            }
            //fin de la decodificacion
        }//si el mensaje tiene mas de 5 campos, ni lo decodificamos, se ignora
    mensajeCompleto = false;                        //Colocamos el mensaje completo a falso
    mensajeRecibido = "";                           //Limpiamos el mensaje anterior
    enviarCodigoWifi(LISTO_PARA_RECIBIR_MENSAJE);      //Indicamos  al  movil  que  podemos  seguir
recibiendo mensajes
    return pointerEjes;
}
```

8.3 Firmware ESP8266

init.lua

```
--------------------------------------------------------------------------
--
-- init.lua
--
--------------------------------------------------------------------------
--
-- Carlos Gregorio Martín Pérez
--
-- 4-07-2017
--
-- Configuración base del dongle wifi. Se configura IP, SSID, clave etc.
-- Tras 5 egundos encendido, se lanza el código principal
--
--------------------------------------------------------------------------
wifi.setmode(wifi.SOFTAP)

cfg={}
cfg.ssid="PacaRed"
cfg.pwd="12345678"

cfg.ip="192.168.4.1"
cfg.netmask="255.255.255.0"
cfg.gateway="192.168.4.1"

wifi.ap.setip(cfg)
wifi.ap.config(cfg)

tmr.alarm(0,5000,0,function() -- run after a delay
      --main.lua contiene el funcionamiento. Si se quere cambiar el programa
      --hay que encender rápidamente ESP8266 y en ESPlorer darle a "Format"
      dofile("main.lua")
end)
```

main.lua

```
--------------------------------------------------------------------------
--
-- main.lua
--
--------------------------------------------------------------------------
--
-- Carlos Gregorio Martín Pérez
--
-- 04-06-2017
--
-- Funcionamiento princila del ESP8266. Hace de puente entre una linea serial
-- conectada a PACA y los distintos dispositivos conectados vía wifi
--
--------------------------------------------------------------------------
port=8888
```

```
dispositivos={}
index = 0

srv=net.createServer(net.TCP, 28800)

srv:listen(port,function(conn)    --abrimos la scucha en 192.168.4.1:8888
    --Almacenamos toda las nuevas cnexiones en una lista enlazada
    conn:on("receive",function(conn,payload)
        uart.write(0, payload)
    end)
    dispositivos[index]=conn
    index = index+1
end)

--Todo lo recibido x uart se envía a los dispositivos (conexiones) existentes
--OJO ESTO BLOQUEA LA INTERACCIÓN DE ESPLORER CON EL CHIP.
--HAY QUE FORMATEAR EL DISPOSITIVO PARA PODER CAMBIAR CUALQUIER COSA
uart.on("data", 0, function(data)
    for i=0 , index do
        if dispositivos[i] ~= nil then  --mientras exista elemento
            dispositivos[i]:send(data)  --enviamos lo que haya en UART
        end
    end
end, 0)
```

Capítulo 9
Apéndice 2: Planos

9.1 Planos Finca Las Lucanas

Se trata de un esquema general sobre cómo está estructurada la finca. En el se representan las distintas secciones que la componen, y además se representa una serie de caminos, por donde el carro debe ser capaz de desplazarse.

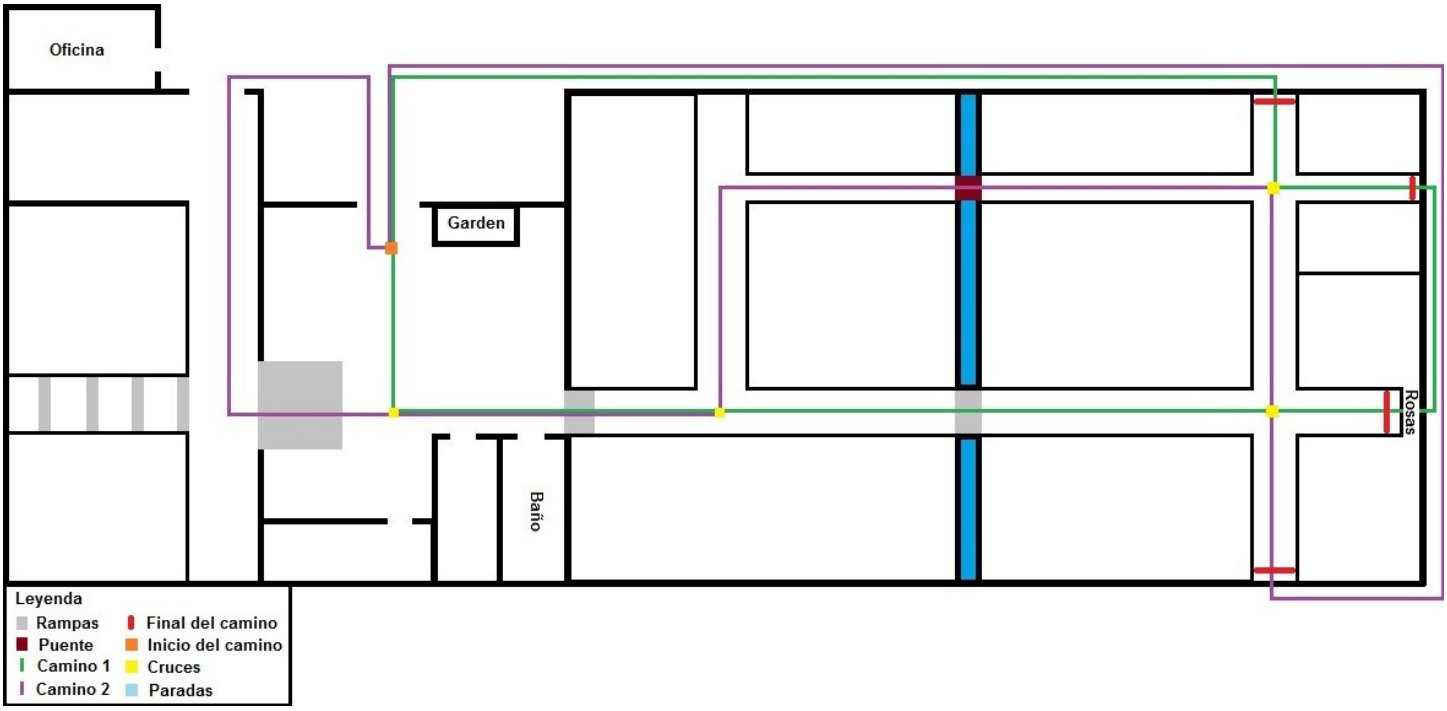


Figura 9.1 – Plano general de Finca Las Lucanas

Bibliografía

[**1**] Nates, J. (2013, April 12). La tecnología punta de la agricultura. Retrieved from http://www.abc.es/natural-biodiversidad/20130412/abci-agricultura-tecnologia-punta-201304121101.html

[**2**] Raspberry Pi Foundation. (n.d.). Retrieved from https://www.raspberrypi.org/about/

[**3**] Welcome to Raspbian. (n.d.). Retrieved from http://www.raspbian.org/

[**4**] Mbed LPC1768. (n.d.). Retrieved from https://developer.mbed.org/platforms/mbed-LPC1768/

[**5**] ESP8266.net — The Internet of Things with ESP8266. (n.d.). Retrieved from http://esp8266.net/

[**6**] NodeMcu - An open source firmware based on ESP8266 wifi-soc. (n.d.). Retrieved from http://nodemcu.com/index_en.html

[**7**] The Programming Language Lua. (n.d.). Retrieved from http://www.lua.org/

[**8**] OpenCV library. (n.d.). Retrieved from http://opencv.org/

[**9**] OpenCV and Pi Camera Board! (2013, June 15). Retrieved from https://thinkrpi.wordpress.com/2013/05/22/opencv-and-camera-board-csi/

[**10**] CMake Tutorial. (n.d.). Retrieved from https://cmake.org/cmake-tutorial/

[**11**] Matos, S. (2014). Electronics and Robotics. Retrieved from http://roboticssamy.blogspot.com.es/2014/02/signs-reading-with-opencv-code.html

[**12**] Torres, J. (2013, February 18). Detección de movimiento con OpenCV – Jesús Torres – Medium. Retrieved from https://jmtorres.webs.ull.es/me/2013/02/deteccion-de-movimiento-con-opencv/

[**13**] UNIX System V. (2017, July 14). Retrieved from https://en.wikipedia.org/wiki/UNIX_System_V

[**14**] Systemd System and Service Manager. (n.d.). Retrieved from https://freedesktop.org/wiki/Software/systemd/