



**Universidad**  
Zaragoza



**Escuela de  
Ingeniería y Arquitectura**  
**Universidad Zaragoza**

## Proyecto Final de Carrera

### Ingeniería de Telecomunicaciones

# Sistema interactivo para manejo de electrodomésticos en entornos domésticos

Marcos Sebastián Magallón

Diciembre de 2013

Director: Eduardo Montijano  
Ponente: Carlos Sagüés

Departamento de Informática e Ingeniería de Sistemas  
Escuela de Ingeniería y Arquitectura  
Universidad de Zaragoza



---

# RESUMEN

En los últimos años, ha habido una gran evolución en la forma de interactuar con el entorno doméstico, pudiendo controlar muchos aspectos de una casa de una forma remota: control de la iluminación, del aire acondicionado, de persianas, de alarmas, etc... Todo ello con la idea de facilitar la vida del usuario, adaptándose a sus necesidades. Por ello, se sigue investigando en nuevos sistemas en el ambiente doméstico que mejoren la comodidad del usuario.

Una ubicación donde se están produciendo grandes avances son las cocinas, donde los electrodomésticos cumplen un papel muy importante en la vida diaria de las personas. Este proyecto se centra en la interacción del usuario con la placa de una cocina, y tiene como objetivo principal su manejo de una forma interactiva y remota, informando de los recipientes que se encuentran en la placa y los eventos que se producen.

El sistema está basado en el uso de una cámara Microsoft Kinect, que está constituida por una cámara de visión convencional y un sensor de rango basado en infrarrojos que permite un conocimiento tridimensional del entorno. Utilizando ambos tipos de imágenes de forma simultánea, en este PFC se proponen procesos eficientes, robustos y completos capaces de capturar lo que ocurre en la cocina e informen al usuario para interactuar con ella.

Estos datos obtenidos mediante visión por computador son comunicados a una aplicación en un dispositivo móvil Android. De esta forma, el usuario es informado de los recipientes que hay sobre la placa y los eventos que se producen en ella, además de información sobre sus áreas, tipo y altura, pudiendo controlar visualmente el estado de la cocina y comunicando la potencia a la que desea que se caliente cada uno de los recipientes.





---

# Agradecimientos

Quiero dedicar esta página a todas las personas que han estado ahí no sólo durante la realización de mi proyecto fin de carrera, sino también a los que me han acompañado a lo largo de todos estos años de carrera que han supuesto una gran etapa en mi vida.

En primer lugar, me gustaría agradecerle a D. Carlos Sagües la oportunidad de realizar este proyecto. Por saber relacionarme con la gente necesaria, por proporcionarme los materiales necesarios, por sus útiles consejos y su apoyo, gracias. También a mi director del proyecto, D. Eduardo Montijano, por todo su tiempo, su dedicación, sus consejos, por su inestimable ayuda siempre que la he necesitado y por haberme guiado de la mejor forma posible, por todo ello, gracias.

Quisiera también dar las gracias a dos compañeros de mi laboratorio, Alejandro Yús y Sergio Ayuso, cuyas ayudas y consejos me impulsaron a seguir avanzando en el proyecto, siempre dispuestos y tratándome amablemente desde el primer momento.

No me puedo olvidar de mis compañeros y amigos de todos estos años, especialmente a mi familia del Movi, por preocuparse por mí y sacarme una sonrisa siempre que era posible, y a los Packets, por estar siempre dispuestos a ayudarme y animarme en cualquier momento.

Para terminar, y por encima de todo, quisiera agradecerles a mi familia, especialmente a mis padres, por la gran dedicación y esfuerzo que han puesto en mi educación, y por permitirme estudiar esta carrera; a ellos les debo lo que soy. Y a Belén, por ser un apoyo incombustible y fundamental en mi vida.

A todos ellos, **gracias.**

---

---

# Índice general

<b>Índice de figuras</b>	<b>XI</b>
<b>Índice de tablas</b>	<b>XI</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Antecedentes . . . . .	1
1.2. Objetivos y alcance . . . . .	3
1.3. Entorno de trabajo . . . . .	3
1.4. Organización de la memoria . . . . .	4
<b>2. Detección de recipientes</b>	<b>7</b>
2.1. Ubicación de la cámara . . . . .	7
2.2. Segmentación de los recipientes . . . . .	8
2.2.1. Calibración de la imágenes RGB y de profundidad . . . . .	8
2.2.2. Detección automática de las esquinas de la placa . . . . .	10
2.2.3. Obtención del plano superficie de la placa (plano de referencia) . . . . .	12
2.2.4. Resta de imagen de profundidad y plano de referencia . . . . .	12
2.3. Detección de contornos . . . . .	13
2.3.1. Contorno de los recipientes . . . . .	14
<b>3. Ajuste de modelo y seguimiento de recipientes</b>	<b>15</b>
3.1. Ajuste de modelo para los recipientes circulares . . . . .	15
3.1.1. Altura del recipiente . . . . .	15
3.1.2. Base del recipiente . . . . .	16
3.2. Seguimiento de los recipientes . . . . .	17
3.3. Interfaz para usuario mediante la imagen RGB . . . . .	19
<b>4. Aplicación móvil</b>	<b>21</b>
4.1. Aplicación COCINA . . . . .	21
4.1.1. Actividad principal . . . . .	22
4.1.2. Actividad de los recipientes . . . . .	22
4.2. Comunicación PC-dispositivo . . . . .	24
4.2.1. Protocolo de envío y recepción de datos . . . . .	24
4.2.2. Protocolo de actualización de potencias . . . . .	25

4.2.3. Protocolo de transmisión y sincronización de la imagen . . .	26
<b>5. Conclusiones y futuras líneas de trabajo</b>	<b>29</b>
5.1. Conclusiones y valoración del trabajo realizado . . . . .	29
5.2. Futuras líneas de trabajo . . . . .	30
<b>Bibliografía</b>	<b>32</b>
<b>A. La cámara Microsoft Kinect</b>	<b>33</b>
A.1. Especificaciones técnicas . . . . .	33
A.2. Funcionamiento del sensor de profundidad . . . . .	39
A.3. Conectividad de la cámara con el ordenador . . . . .	45
A.4. Imágenes RGB y de profundidad . . . . .	45
A.4.1. Obtención de las imágenes RGB y de profundidad . . . . .	46
<b>B. Cálculo analítico del plano de referencia</b>	<b>49</b>
<b>C. Cálculo de homografías</b>	<b>51</b>
C.1. Concepto de homografía . . . . .	51
C.2. Procedimiento . . . . .	52
<b>D. Android</b>	<b>55</b>
D.1. ¿Qué es Android? . . . . .	55
D.1.1. Historia . . . . .	56
D.1.2. Arquitectura Android . . . . .	57
D.1.3. Características . . . . .	58
<b>E. ROS</b>	<b>61</b>
<b>F. Transmisión de datos y protocolos de comunicación</b>	<b>65</b>
F.1. Método de transmisión de datos: Sockets . . . . .	65
F.2. Protocolos de comunicación . . . . .	66
F.3. Comunicación: Servidor . . . . .	68
F.3.1. Abrir socket . . . . .	68
F.3.2. Asociar el socket con un puerto . . . . .	68
F.3.3. Recibir mensaje en el servidor . . . . .	69
F.3.4. Respuesta al cliente . . . . .	70
F.4. Comunicación: Cliente . . . . .	70
F.4.1. Creación socket . . . . .	70
F.4.2. Creación paquete . . . . .	71
F.4.3. Enviar paquete . . . . .	71
F.4.4. Recibir paquete . . . . .	71

---

<b>G. Métodos de detección</b>	<b>73</b>
G.1. Método de Roberts . . . . .	73
G.2. Método de Sobel . . . . .	74
G.3. Método de Prewitt . . . . .	74
G.4. Método de Canny . . . . .	75



# Índice de figuras

<b>Capítulo 1: Introducción</b>	<b>1</b>
1.1. Ejemplo de una cocina inteligente . . . . .	1
1.2. Microsoft Kinect (a) y un ejemplo de imagen tridimensional captada por ella (b) . . . . .	2
<b>Capítulo 2: Detección de recipientes</b>	<b>7</b>
2.1. Posición en la que se coloca la cámara en un extremo del eje menor	7
2.2. Ejemplo de segmentación de recipientes en una imagen . . . . .	8
2.3. Separación entre la cámara IR y la RGB en Kinect . . . . .	9
2.4. Puntos que se corresponden en imagen de profundidad e imagen RGB . . . . .	9
2.5. (a) Imagen RGB (b) Imagen de profundidad con la misma perspectiva que la imagen RGB gracias a la homografía . . . . .	10
2.6. (a) Visión de la placa antes de aplicar la matriz homográfica (b) Visión de la placa transformada con la matriz homográfica . . . . .	10
2.7. (a) Placa vista desde la cámara (b) Marcas después del proceso de filtrado. . . . .	11
2.8. Placa hallada en distintas situaciones . . . . .	12
2.9. (a) Imagen tomada de la placa (b) Imagen donde se toma solo el plano de la placa. . . . .	12
2.10. Placa ubicada y eliminación de márgenes . . . . .	13
2.11. (a) Detección de un recipiente (b) No hay recipiente sobre la placa	13
2.12. (a) Forma del recipiente después del filtro de Gauss (b) Forma del recipiente con el contorno hallado . . . . .	14
<b>Capítulo 3: Ajuste de modelo y seguimiento de recipientes</b>	<b>15</b>
3.1. Esquema explicativo de la geometría del entorno del recipiente . . . . .	16
3.2. Área (zona verde) donde se detecta el punto más alto . . . . .	16
3.3. (a) Perfil de una sartén (b) Detección ajustada de la base de la sartén . . . . .	17
3.4. Seguimiento de una sartén sobre la placa . . . . .	18

3.5.	(a) Imagen de dos recipientes detectados (b) Se sigue identificado el segundo recipiente como el mismo . . . . .	19
3.6.	Imagen RGB modificada para interfaz . . . . .	20
<b>Capítulo 4: Aplicación móvil</b>		<b>21</b>
4.1.	Esquema explicativo de las actividades en la aplicación . . . . .	21
4.2.	Actividad principal de la aplicación Android . . . . .	22
4.3.	(a) Actividad recipiente verde (b) Actividad recipiente rojo . . . . .	23
4.4.	Menú de Android donde se muestra el icono de la aplicación disponible . . . . .	23
4.5.	Esquema general del protocolo de comunicación entre servidor y cliente . . . . .	24
4.6.	Esquema de la transmisión de potencias . . . . .	26
4.7.	Esquema de la transmisión de imágenes . . . . .	27
<b>Capítulo 5: Conclusiones y futuras líneas de trabajo</b>		<b>29</b>
<b>Apéndice A: La cámara Microsoft Kinect</b>		<b>33</b>
A.1.	Esquema de los elementos de la Kinect . . . . .	34
A.2.	(a) Emisor láser que proyecta el patrón de puntos. (b) Sensor CMOS que capta el patrón emitido para calcular la profundidad . . . . .	34
A.3.	Campo de visión horizontal y rango de proximidad de la Kinect . . . . .	35
A.4.	Campo de visión vertical y variación del mismo provocada por la inclinación del motor . . . . .	36
A.5.	Foto de la cámara extraída de la Kinect . . . . .	37
A.6.	a) Sensores y láser empotrados en una placa metálica. b) Placas paralelas a la metálica con la electrónica y microchips que controlan el dispositivo. . . . .	38
A.7.	Esquema de funcionamiento del PS1080 . . . . .	38
A.8.	Captura del patrón de puntos sobre una pared . . . . .	40
A.9.	Distorsión del patrón al introducir un objeto en escena. a) Pared sin nada. b) Se ha situado un libro delante de la pared . . . . .	41
A.10.	Esquema de la triangulación empleada para obtener el valor de profundidad desde la disparidad captada por el sensor . . . . .	42
A.11.	Mapa de disparidad (izquierda) calculado por la Kinect e imagen RGB convencional . . . . .	44
A.12.	Correspondencia entre disparidad $d$ y distancia real en metros (a). Tamaño de paso en milímetros entre dos valores consecutivos de disparidad en función de la distancia en metros (b). Gráficas obtenidas de [10] . . . . .	44
A.13.	Adaptador de USB y corriente para la Kinect . . . . .	45



A.14. Patrón de puntos láser proyectado. . . . .	46
A.15. Imágenes obtenida: (a) Imagen RGB (b) Imagen profundidad . . .	46
A.16. Consola para el manejo de ROS con los nodos que ofrece la cámara para su posible subscripción . . . . .	47
 <b>Apéndice B: Cálculo analítico del plano de referencia</b>	 <b>49</b>
 <b>Apéndice C: Cálculo de homografías</b>	 <b>51</b>
 <b>Apéndice D: Android</b>	 <b>55</b>
D.1. Diferentes dispositivos donde android ya está disponible . . . . .	56
D.2. Esquema de la arquitectura Android . . . . .	57
 <b>Apéndice E: ROS</b>	 <b>61</b>
 <b>Apéndice F: Transmisión de datos y protocolos de comunicación</b>	 <b>65</b>
F.1. Situación de los sockets en capas . . . . .	65
F.2. Esquema del socket actuando como multiplexor/demultiplexor . .	68
 <b>Apéndice G: Métodos de detección</b>	 <b>73</b>



# Capítulo 1

## Introducción

### 1.1. Antecedentes

Todo aquello que hace años parecía ciencia ficción, se va acercando poco a poco a la realidad. La tecnología avanza a pasos agigantados, y cada vez está más integrada en nuestro día a día. Por lo tanto, la incorporación de estos avances en el entorno doméstico supone un paso más en la evolución de la misma. Para el inicio de esta integración, nos remontamos tres décadas atrás.

La primera aproximación de lo que se podría llamar casa inteligente, data del año 1984, y nació en EE.UU. Desde entonces, se han ido desarrollando sistemas de control en los edificios con la idea de lograr una mayor eficiencia y conseguir una mejor gestión del consumo de energía. Todos estos avances han ido encaminados a ofrecer una calidad de vida para el usuario, a través de una reducción del trabajo en labores domésticas y un aumento de la seguridad y del bienestar. Con esta mentalidad han surgido sistemas interactivos para el ambiente doméstico que, año tras año, van mejorando la facilidad de uso de cualquier elemento presente en un hogar.



**Figura 1.1:** Ejemplo de una cocina inteligente

Un ejemplo de esta evolución son las cocinas domóticas (Figura 1.1), donde todo está perfectamente integrado y donde, a través de una pantalla, se pueden realizar seguimientos de varios electrodomésticos (lavadora, lavavajillas, frigorífico). En pocos años, todos estos sistemas interactivos y de control han pasado a ser algo habitual en nuestra vida. Poco a poco nuestros hogares se van adaptando

a esta nueva tecnología y evolucionan, de tal manera que podemos comunicarnos con nuestros electrodomésticos, y mantener un control sobre ellos. Este proyecto se centra concretamente en el control de una placa de cocina, permitiendo al usuario conocer lo que ocurre sobre la misma en todo momento aunque se encuentre en otra ubicación de la casa. La principal razón es la mayor comodidad para el usuario: poder tener control sobre la cocina mientras aprovecha el tiempo en otras tareas, o simplemente poder controlarla mientras disfruta de un rato de tiempo libre en el sofá. Pero a nivel económico, también reporta ventajas: poder tener un dispositivo para manejar los fuegos fuera de la propia placa supone un ahorro notable en el aislamiento térmico que supone incorporar un mando en la misma, además de ahorrar en el hardware. En el desarrollo del mismo, intervienen dos factores: la detección de recipientes y la interacción con el usuario.

Para llevar a cabo la detección, utilizamos visión por computador, un campo de la informática que permite extraer información de alto nivel a partir de las imágenes que son captadas por una cámara o un sensor. Entre los objetivos habituales de la visión por computador, se encuentran: la detección, segmentación, localización o seguimiento de ciertos objetos en las imágenes, registro de una escena de forma tridimensional, etc... Esto resulta útil para numerosas aplicaciones, como puede ser la restauración de imágenes, el reconocimiento de ciertos cuerpos u objetos o para analizar características de una escena.

En el proyecto se utiliza una cámara Microsoft Kinect(Figura 1.2a), que salió al mercado en otoño de 2010. En un principio, ésta tenía fines de entretenimiento. Sin embargo, debido a su potencial, en la actualidad este sensor tiene otros fines, como diversos usos médicos o reconstruir mapas en relieve para el análisis de diversos entornos. Una de las principales utilidades de las cámaras RGB-D como Kinect es la interacción máquina-usuario, permitiendo por ejemplo, controlar ordenadores y otros aparatos con las manos, o informar a las personas de eventos que ocurren a su alrededor. Todo ello de una forma barata y más sencilla de lo que hasta antes de su aparición se había podido hacer.



**Figura 1.2:** Microsoft Kinect (a) y un ejemplo de imagen tridimensional captada por ella (b)

Para la interacción con el usuario, se crea una aplicación móvil. En los últimos años, el sistema operativo Android ha tenido un crecimiento exponencial en el uso de forma global, siendo hoy por hoy el sistema para smartphones más utilizado.

A la hora de realizar una aplicación dentro de este entorno, los usuarios ya van a tener un medio físico donde poder instalarla. Todos saben manejarse en este entorno, descargando aplicaciones, usándolas, y realizando otras tareas gracias a las posibilidades que este sistema operativo ofrece. Además, se trata de un sistema operativo libre. De esta manera, cualquier desarrollador que quiera llevar a cabo una aplicación, puede hacerlo, además de poder subir la aplicación al mercado.

La realización de este proyecto resulta de interés tanto para BSH Electrodomésticos España como para la Universidad de Zaragoza. A grupo BSH Electrodomésticos España le permite investigar otras formas en el control de electrodomésticos, concretamente en una cocina, ofreciendo al usuario una forma de cocinar más fácil y cómoda, con una tecnología que está en auge. A la Universidad de Zaragoza le permite abrir nuevas líneas de investigación sobre el uso de sensores RGB-D en el marco de un intercambio de conocimiento con la empresa privada.

### 1.2. Objetivos y alcance

El objetivo principal del proyecto es conseguir una correcta caracterización de los recipientes y los movimientos que se producen en la placa mediante la visión artificial. Se quiere construir un sistema capaz de detectar los recipientes que están situados en la placa, y hacer un seguimiento y una correcta caracterización de los mismos con la mayor precisión posible.

El otro gran objetivo consiste en enviar al usuario esta información a un dispositivo móvil remoto, de tal manera que pueda saber lo que pasa en la cocina sin estar presente en ella, pudiendo tener el control sobre la misma. Para ello, se propone crear una aplicación con la que controlar la cocina, viendo imágenes de la misma cuando el usuario lo desee, y controlando la potencia de los recipientes desde esta aplicación.

Los objetivos concretos son:

- Implementación y puesta a punto de algoritmos de visión para detección de recipientes.
- Implementación de algoritmos para el seguimiento e identificación de los mismos
- Desarrollo de una interfaz sobre un dispositivo móvil para la interacción remota con los recipientes.

### 1.3. Entorno de trabajo

La visión por computador se realiza con una cámara Microsoft Kinect. Esta cámara captura dos tipos de imágenes simultáneas: por un lado, una imagen RGB como las cámaras convencionales, y por otro lado, tiene un sensor de infrarrojos que permite obtener una imagen de profundidad(Figura 1.2b). Esto supone una gran ventaja, ya que permite crear algoritmos mucho más robustos, eficientes y

complejos que si se utilizara sólo la imagen RGB o la imagen de profundidad por separado. En otros proyectos, se utiliza sólo imagen de profundidad [1] o imagen RGB [2]. Con la imagen de profundidad sólo tenemos datos de distancia, no de color, ni una imagen cercana a la real. Y viceversa, con la imagen convencional no tenemos información de distancia, sólo tenemos la información del color. Por lo tanto, en nuestro caso, al utilizar dos imágenes, tenemos más información.

Para todo el trabajo con las imágenes, se utiliza la librería OpenCV (Open Computer Vision) [7], que contiene más de 500 funciones, que abarcan una gran gama de áreas en el proceso de visión, como reconocimiento de objetos, calibración de cámaras, visión estérea y visión robótica. Para el empleo de todas estas funciones, ha resultado de gran ayuda el libro *Learning OpenCV* [26].

Para la toma de imágenes se ha decidido utilizar *ROS* [3] con el sistema operativo *Ubuntu* [5], ya que nos permite tomar los dos tipos de imágenes de una forma simultánea y tratarlas en tiempo real. *ROS* (Robot Operating System) proporciona librerías y herramientas para que los desarrolladores puedan crear aplicaciones robóticas. En este caso, vamos a utilizar sólo la parte robótica de visión, que es la que nos interesa para conseguir las imágenes de la cámara Kinect. Además, este tipo de librerías funcionan en *Ubuntu*, sistema operativo que se distribuye como software libre y gratuito. Los drivers empleados para el manejo de la cámara son los de OPENNI [6].

Para realizar la interfaz y control de la placa, se ha optado por realizar una aplicación de Android. Una de las razones de que se haya escogido realizar una aplicación, es que a día de hoy la mayoría de personas tiene un dispositivo móvil de android, ya sea un teléfono móvil o una tablet. Otra de las razones por las que se escoge Android es que es software libre. En el apéndice D se profundiza más en su historia, características o arquitectura.

Para desarrollar la aplicación se utiliza como soporte el programa informático *ECLIPSE* [28], compuesto por un conjunto de herramientas de programación de código abierto multiplataforma para desarrollar aplicaciones Java. Para desarrollar una aplicación de Android, se hace habitualmente con el lenguaje de programación Java y el conjunto de herramientas de desarrollo (SDK, Software Development Kit). Comprende un depurador de código, biblioteca, un simulador de teléfono, documentación, ejemplos de código y tutoriales. A la hora de empezar a hacer la aplicación, la página *Android Developers* [29] nos ofrece una amplia variedad de tutoriales para el desarrollo por cuenta propia del usuario.

## 1.4. Organización de la memoria

El resto de la memoria se organiza de la siguiente manera:

- En el capítulo 2, se trata el tema de la segmentación y reconocimiento de los recipientes, los métodos y algoritmos que se han implementado para analizar su ubicación y dimensiones. Todo ello usando la cámara Microsoft Kinect

- El capítulo 3 se centra en ajustar la detección de contornos a un modelo más real, ajustando lo máximo posible los datos recibidos a la realidad, y posteriormente se realiza un seguimiento de los recipientes. También se genera una imagen RGB interfaz que sirve para ser enviada al dispositivo móvil, y que proporciona diversos datos al usuario.
- En el capítulo 4 se explica la aplicación móvil construida para el control de la placa. Por un lado, la interfaz construida en la aplicación, mostrando las distintas actividades que la componen y las posibilidades que ofrece. Por otro, nos adentramos en la comunicación bidireccional que se da entre el PC, que actúa de servidor, y el dispositivo móvil, que actúa de cliente. Se profundiza en el método de envío de datos, el protocolo que se utiliza y se explica con detalle cómo se realiza la comunicación entre ambos.
- Para finalizar, en el capítulo 5 se comentan las conclusiones finales tanto a nivel de proyecto como a nivel personal, además de las futuras líneas de trabajo que podrían surgir a partir de este proyecto fin de carrera.





## Capítulo 2

# Detección de recipientes

Este capítulo se centra en uno de los pilares del proyecto: la detección de los recipientes. En él se explica la base teórica para la obtención de información y ubicación de los recipientes, hallando qué hay encima de la placa y qué características tiene.

Para empezar a trabajar partimos de las imágenes de RGB y profundidad (en el apéndice A.4 se explica su obtención). A raíz de ellas, vemos la segmentación de recipientes (sección 2.2), que explica como aislar esos recipientes en la imagen. Posteriormente, realizamos la detección de contornos (sección 2.3), que nos permite saber dónde se encuentran los recipientes.

### 2.1. Ubicación de la cámara



**Figura 2.1:** Posición en la que se coloca la cámara en un extremo del eje menor

Al comienzo del proyecto, hay que tener en cuenta el posicionamiento de la cámara. Tiene que permitir una correcta ejecución del sistema, por lo que tiene que ser funcional. Por otro lado, tiene que ser un posicionamiento viable, tanto en el lugar donde se está desarrollando el proyecto como pensando en su

posterior comercialización. Lo ideal sería colocar la cámara a una altura óptima de 0.8 metros sobre el cruce del eje mayor y el eje menor de la placa. Esto permitiría ver los recipientes de la forma más correcta posible, ver su fondo y su contenido de forma adecuada. Sin embargo, desde un punto de vista real esto no es viable, ya que los humos y vapores procedentes de la cocina ensuciarían la cámara e impediría una visualización correcta de los recipientes. Por ello se opta por desplazar la cámara hasta el extremo de uno de los ejes menores de la placa.

En el laboratorio, la cámara se ha colocado en un trípode, tal y como se ve en la Figura 2.1, colocado en un extremo del eje menor. Pensando de una manera más comercial, la mejor colocación de la cámara sería en la campana extractora, en el extremo más cercano a la pared.

## 2.2. Segmentación de los recipientes

El objetivo de la segmentación de recipientes [17] consiste en distinguir si un píxel pertenece a nuestro objeto de interés, descartando de alguna manera aquellos que no pertenezcan (Figura 2.2).



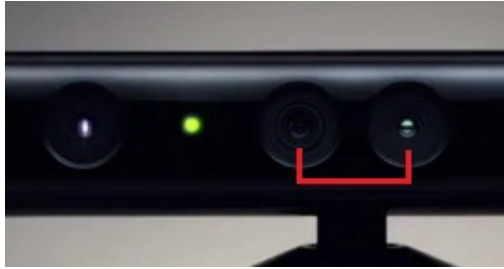
**Figura 2.2:** Ejemplo de segmentación de recipientes en una imagen

El proceso se divide en varias partes. La primera de ellas es la ubicación de la placa en la imagen. Esto se hace de una forma automática mediante un filtrado de color. Esta es una fase de calibración, que se realiza solo una vez cuando se inicialice el programa, capta las cuatro esquinas de la placa y la ubica dentro de la imagen que vayamos a utilizar. Una vez obtenidos los cuatro puntos de la placa, entonces podemos pasar a calcular el plano que forma en la imagen de profundidad. Por último, cuando tengamos ese plano de referencia, podemos detectar los recipientes para trabajar sobre ellos.

### 2.2.1. Calibración de la imágenes RGB y de profundidad

Como se ha comentado antes, la cámara Microsoft Kinect nos permite capturar dos tipos de imágenes: la imagen RGB y la imagen de profundidad. Aunque estos dos tipos de imágenes se toman de manera simultánea, no se corresponden exactamente punto por punto, ya que físicamente la cámara IR y RGB están separadas unos centímetros (Figura 2.3). Por ello, se trata la imagen de profundidad

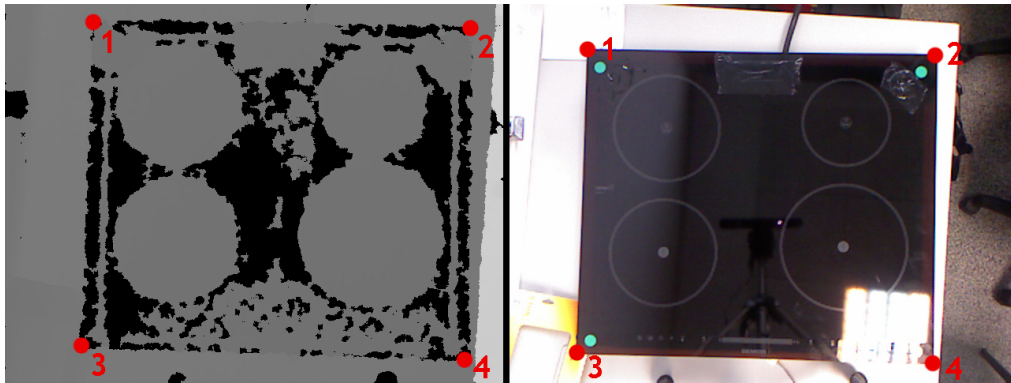
con una homografía para buscar la correspondencia, consiguiendo la misma visión que tendríamos desde la imagen convencional pero transformada a una imagen de profundidad.



**Figura 2.3:** Separación entre la cámara IR y la RGB en Kinect

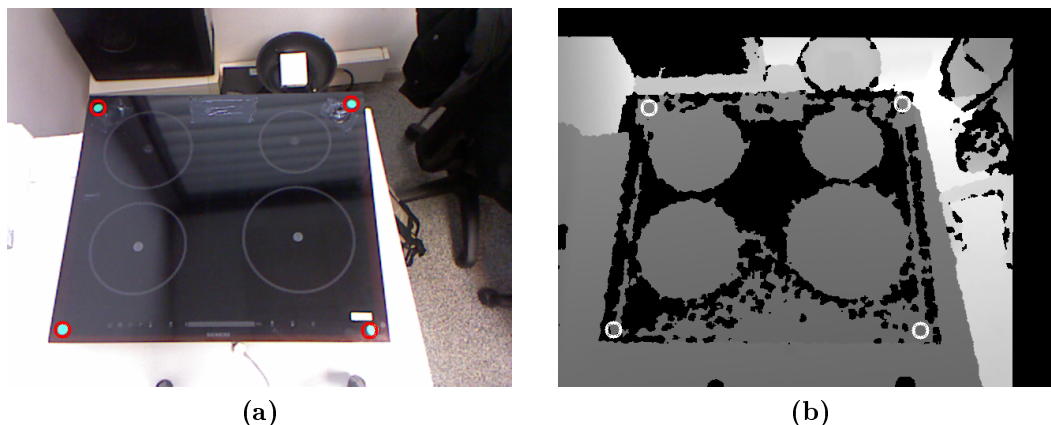
La homografía es una transformación proyectiva que determina una correspondencia entre dos figuras geométricas planas, de tal forma que a cada uno de los puntos y rectas de una de ellas, le corresponden un punto y una recta de la otra, respectivamente. Dicho de otra forma, si tres puntos de un plano están contenidos en una línea, lo seguirán estando en el plano transformado. Por lo tanto, una homografía conserva la naturaleza de los elementos transformados. Viene definida por la ecuación  $\mathbf{x}'_i = H\mathbf{x}_i$ , donde  $\mathbf{x}'_i$  e  $\mathbf{x}_i$  son las coordenadas en píxeles en las dos imágenes correspondientes a la proyección de un mismo punto 3D de la escena. La idea de homografía se profundiza en el apéndice C.

La calibración se realiza de forma manual, tomando cuatro coordenadas de la imagen de profundidad, que se corresponden visualmente con otras cuatro coordenadas de la imagen RGB, tal como se muestra en la Figura 2.4, donde se han tomado como referencia las esquinas de la placa. Esta calibración se realiza una sola vez al principio del proyecto, permitiendo obtener una correspondencia entre las dos imágenes aunque la visión de la placa sea oblicua.



**Figura 2.4:** Puntos que se corresponden en imagen de profundidad e imagen RGB

La propia librería de OpenCV posee las funciones necesarias para hacer la homografía de una forma más rápida, eficaz y robusta. El resultado ya mencionado, es la imagen de profundidad vista con la misma perspectiva que la imagen convencional (Figura 2.5).



**Figura 2.5:** (a) Imagen RGB (b) Imagen de profundidad con la misma perspectiva que la imagen RGB gracias a la homografía

También se podría utilizar la homografía para conseguir una visualización más cercana a la real de los recipientes y sus bases, debido a que esto no es posible por la ubicación de la cámara. Al no colocar la cámara en el cruce del eje mayor y el eje menor, ni los recipientes ni la placa se observan de una forma correcta: los recipientes se ven elípticos y no circulares, y la placa se visualiza con una forma trapezoidal (Figura 2.6a), y no rectangular. Si conseguimos hacer una homografía de la imagen, aunque no tengamos la cámara colocada en el cruce de los ejes mayor y menor de la placa, podemos conseguir una versión simulada de esa visión que deseamos tener (Figura 2.6b).



**Figura 2.6:** (a) Visión de la placa antes de aplicar la matriz homográfica (b) Visión de la placa transformada con la matriz homográfica

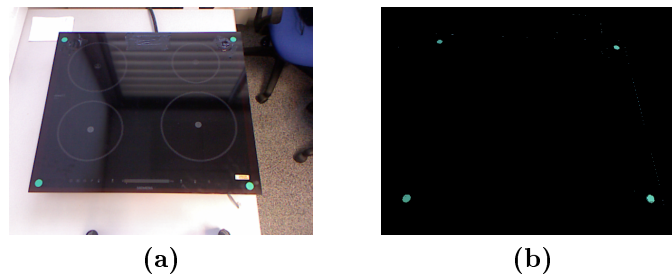
### 2.2.2. Detección automática de las esquinas de la placa

En este apartado lo que se busca es situar la placa dentro de la imagen, conociendo cuáles son sus cuatro esquinas en la imagen RGB. Para ello, se puede hacer una calibración manual (haciendo click en las esquinas de la propia imagen).

Sin embargo, en busca de una automatización del proceso en vistas a una posible comercialización del producto, y favoreciendo una mayor comodidad y rapidez para el usuario, también se ha desarrollado una calibración automática.

Para ello, marcamos las esquinas con círculos de un color característico, de tal forma que puedan ser reconocidos por la cámara RGB mediante un filtrado de color en la imagen. El método de filtrado utilizado acota los valores en cada uno de los canales (rojo, verde o azul), para quedarse sólo con el color deseado. Para analizar los valores a los que tenemos que acotar cada canal, realizamos la captura de varias imágenes con distinta luminosidad. La mayor problemática se basa en establecer estas cotas para cada color, ya que es importante tener una elevada tasa de aciertos. En este caso, el análisis de forma empírica de los valores a tomar ha dado como resultado:

- $50 < R < 110$  ;  $170 < G < 235$  ;  $140 < B < 200$

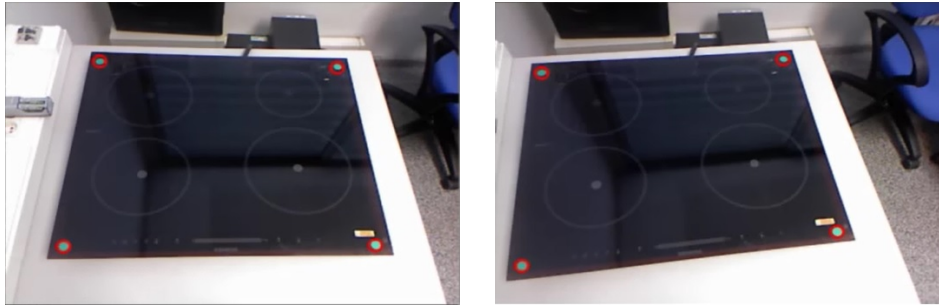


**Figura 2.7:** (a) Placa vista desde la cámara (b) Marcas después del proceso de filtrado.

Una vez introducido el filtro de color, la imagen resultante son sólo los cuatro pequeños círculos que suponen las esquinas, como se puede apreciar en la Figura 2.7. También podemos observar pequeños píxeles fruto de la reflexión en los bordes de la placa, que son descartados filtrando los blobs por tamaño. Una vez tenemos la imagen filtrada, calculamos las coordenadas de las cuatro esquinas utilizando la detección de contornos (sección 2.3) que ofrece OpenCV.

Como resultado de todo el proceso, tenemos perfectamente situada la placa en el entorno de la imagen. El algoritmo es capaz de ubicarla siempre que esté situada dentro del campo de visión de la cámara, como podemos observar en la Figura 2.8.

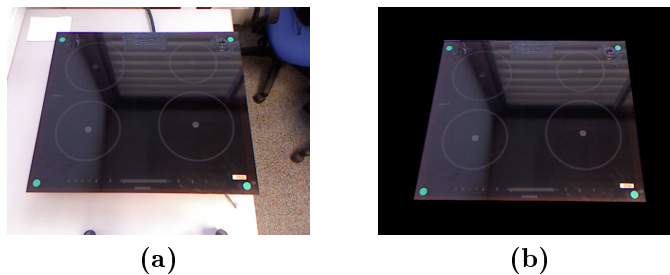
Esta idea de marcar las esquinas para poder realizar la calibración, es una idea pensada con los medios de los que se dispone en el laboratorio. Sin embargo, de cara a la comercialización del producto, la idea es incorporar en las esquinas de las placas unos LEDs que se encendieran sólo el tiempo suficiente para ser captadas y realizar la captura de estos puntos. Una vez han sido calculados estos puntos, ya no se necesita volver a hallar su posición, por lo que estos LEDs ya no volverían a encenderse en todo el proceso hasta que se volviera a reiniciar.



**Figura 2.8:** Placa hallada en distintas situaciones

### 2.2.3. Obtención del plano superficie de la placa (plano de referencia)

Para calcular el plano en tres dimensiones del entorno de la placa se necesitan las coordenadas de al menos tres puntos pertenecientes a ese plano. Por lo tanto, utilizamos tres de los puntos hallados en el apartado anterior. Aunque existen varios métodos para crear un plano[14], se opta por hacerlo de forma analítica por su mayor sencillez. Las ecuaciones para el cálculo se detallan en el apéndice B.



**Figura 2.9:** (a) Imagen tomada de la placa (b) Imagen donde se toma solo el plano de la placa.

A partir de los puntos, calculamos las distancias respecto a la cámara para crear una matriz de 640x480 parejo a la imagen de profundidad (que es también una matriz de 640x480 de un canal) que sea nuestro plano de referencia, para poder hallar después la diferencia entre ambas imágenes. Es por ello que tenemos que mirar cuales son las distancias en esos puntos en la imagen de profundidad. Se cogen tres puntos de los cuatro mencionados para calcular la ecuación general del plano.

### 2.2.4. Resta de imagen de profundidad y plano de referencia

Para realizar la resta de imágenes, se hacen los siguientes pasos:

1. Se toma la imagen de profundidad actual.
2. Se hace un recorte por los márgenes de la imagen (Figura 2.10), para eliminar elementos que pueden proporcionar datos falsos en las medidas de

### 2.3. DETECCIÓN DE CONTORNOS

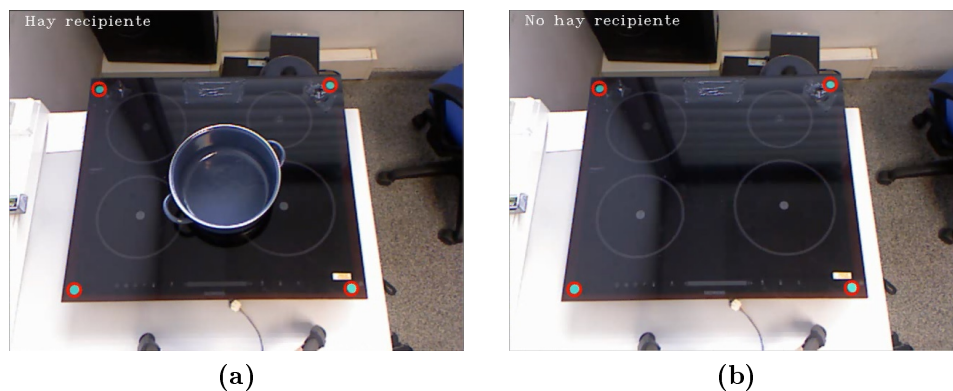
---

profundidad. Estos casos se dan cuando hay objetos que hacen que la luz se refleje.



**Figura 2.10:** Placa ubicada y eliminación de márgenes

3. La resta entre la imagen de profundidad y el plano de referencia se hace por comparación pixel a pixel, recorriendo las matrices por filas y columnas. Esta comparación hace que si un pixel de la imagen de profundidad vale más que el píxel del plano de referencia, entonces su valor es 0. Sino, el valor se mantiene. Como las medidas no son exactas debido al margen de error que posee la propia Microsoft Kinect, y para evitar medidas que puedan falsear el proceso, se sube unos centímetros el plano de referencia. De esta forma, el sistema puede identificar si hay algún recipiente en la encimera (Figura 2.11a), o si no hay recipiente (Figura 2.11b).



**Figura 2.11:** (a) Detección de un recipiente (b) No hay recipiente sobre la placa

### 2.3. Detección de contornos

La detección de contornos es necesaria para conseguir la ubicación precisa de los recipientes. Consiste en encontrar el contorno de los distintos recipientes que puedan situarse sobre la placa, y a raíz de ello, poder estimar de una forma más concreta su situación y características.

En este caso, debemos utilizar la detección de contornos tanto para detectar los recipientes que pueda haber sobre la placa, como para encontrar los cuatro puntos de las marcas que hemos utilizado para situar la cocina y poder formar el plano de referencia. OpenCV nos proporciona una serie de funciones para poder realizar la detección de contornos.

### 2.3.1. Contorno de los recipientes

Para hallar los contornos de los recipientes, se utiliza la imagen de profundidad, que es una matriz de distancias. Se sube el plano de referencia lo suficiente para que no haya medidas falsas, de tal forma que dejemos solamente la forma del recipiente. A partir de ahí, aplicamos el *Threshold* y el filtro gaussiano. El *Threshold*[18] es un método de segmentación, que consiste en separar las regiones de una imagen correspondiente a los objetos que queremos analizar. Esta separación se basa en la variación de la intensidad entre los píxeles de objetos y los píxeles del fondo. Posteriormente, aplicamos un filtro gaussiano, que provoca que el ruido se suavice, y que ayuda a que la detección sea más robusta. Este filtro se suele aplicar previo a la detección de bordes en tratamientos de imagen. Tiene una simetría rotacional, produciendo el mismo efecto en todas las direcciones. Además, está controlado por un umbral: cuanto mayor es este umbral, mayor suavizado. El resultado se muestra en la figura (Figura 2.12a).



**Figura 2.12:** (a) Forma del recipiente después del filtro de Gauss (b) Forma del recipiente con el contorno hallado

Ahora, esta imagen pasa por la función *findContours*, lo que permite encontrar los puntos que formarían contornos y guardar la información en un vector. El método que se utiliza es el contenido en [24]. Una vez encontrados estos contornos, se pasan a la función *minEnclosingCircle*, que nos da como resultado las formas circulares encontradas mediante dos vectores: uno que almacena los radios, y otro que almacena los centros. De esta forma, tenemos los recipientes completamente ubicados en la placa (Figura 2.12b).



## Capítulo 3

# Ajuste de modelo y seguimiento de recipientes

En el capítulo anterior, conseguimos hacer la detección, que nos sirve para saber qué recipientes hay encima de la placa. Sin embargo, queremos ser más precisos, y lograr caracterizar el recipiente por su tipo, altura o base. Con esta información, podemos realizar posteriormente un seguimiento de los mismos más completo, y modificar la imagen RGB para mandarla al dispositivo móvil.

### 3.1. Ajuste de modelo para los recipientes circulares

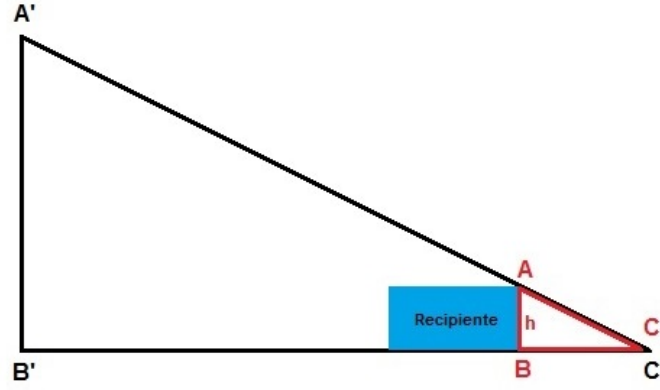
En la sección 2.3, hemos detectado el contorno de los recipientes, pero hay que intentar ajustarlos lo máximo posible a la realidad. Esto se traduce en conocer información adicional acerca de la altura y la base del recipiente. Para hallarlos, es necesario tener en cuenta el espacio geométrico en que nos movemos, conocer donde se encuentra la cámara, el recipiente, las distancias y los ángulos, para poder realizar la estimación con el mayor detalle.

#### 3.1.1. Altura del recipiente

La altura se halla por medio de la semejanza de triángulos. Esto lo visualizamos en la Figura 3.1, donde la altura del recipiente sería  $h$ , que se corresponde a la distancia  $AB$ . Pero esta distancia no la podemos tomar directamente.

La cámara está ubicada en el punto  $A'$ . Sabemos la altura a la que está situada, por lo tanto, sabemos cuál es la distancia  $A'B'$ . Posteriormente, se averigua la distancia  $A'A$  gracia a la matriz de profundidad, donde quedan registradas las distancias. Para saberlo, tomamos el punto más alto dentro del área donde hemos detectado el recipiente (Figura 3.2). De esta forma tenemos la distancia  $A'A$ , que nos sirve en el siguiente paso para hallar la longitud  $AC$  que es la que realmente necesitamos.

A partir de la coordenada del punto más alto, miramos cual es la distancia a ese punto en el plano de referencia. De esta forma, tenemos la distancia  $A'C'$ . Y como antes hemos hallado la distancia  $A'A$ , podemos saber la longitud del segmento  $AC$ .



**Figura 3.1:** Esquema explicativo de la geometría del entorno del recipiente



**Figura 3.2:** Área (zona verde) donde se detecta el punto más alto

Estas medidas están relacionadas de la siguiente manera:

$$h = AB \quad (3.1)$$

$$\frac{AB}{A'B'} = \frac{AC}{A'C'} \quad (3.2)$$

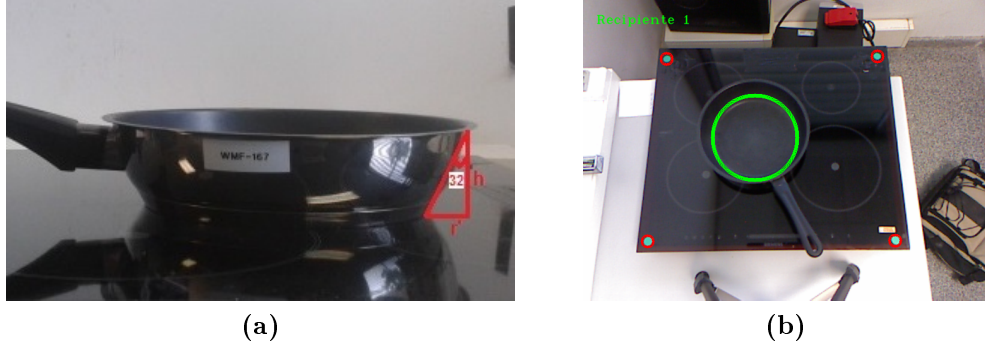
### 3.1.2. Base del recipiente

Hacemos una caracterización de los recipientes, utilizando la altura calculada en el apartado anterior. Si el recipiente pasa de cierto límite, se considera una cazuela, recipiente cuya base tiene el mismo radio que la medida que realizamos en la parte de arriba. Si se da esta situación, ya tenemos hallado el radio y el área de la base. En cambio, cuando el recipiente no sobrepasa cierto límite de altura, se considera una sartén, y por lo tanto el radio del recipiente disminuye conforme disminuimos en altura.

En el caso de ser una sartén, basándonos en la Figura 3.3a, hallamos el valor de  $r'$ , que es el valor que tenemos que restar al radio que nos haya indicado la

### 3.2. SEGUIMIENTO DE LOS RECIPIENTES

---



**Figura 3.3:** (a) Perfil de una sartén (b) Detección ajustada de la base de la sartén

detección de recipiente. Estimamos que el ángulo del triángulo que forman la altura y el lateral de la sartén tiene  $32^\circ$ , formándose así un triángulo rectángulo con el que podemos estimar  $r'$ :

$$\text{Tag}32^\circ = \frac{r'}{h} \quad (3.3)$$

Realizando la diferencia entre el radio medido en la máxima altura del recipiente y la medida  $r'$ , se obtiene como resultado el radio estimado(Figura 3.3b).

### 3.2. Seguimiento de los recipientes

Uno de los objetivos en la aplicación final es que los recipientes estén siempre bajo control, y que se pueda estar pendiente de todos los eventos que se producen en la placa. Encima de ella siempre va a haber movimientos, añadiendo o quitando recipientes, o desplazándolos hacia otra ubicación, de tal manera que, aunque alguno de ellos cambie de sitio, siga siendo identificado como el mismo recipiente, para después poder mostrarle la imagen al usuario de lo que ocurre, y aplicarle la potencia que sea necesaria a cada uno de ellos.

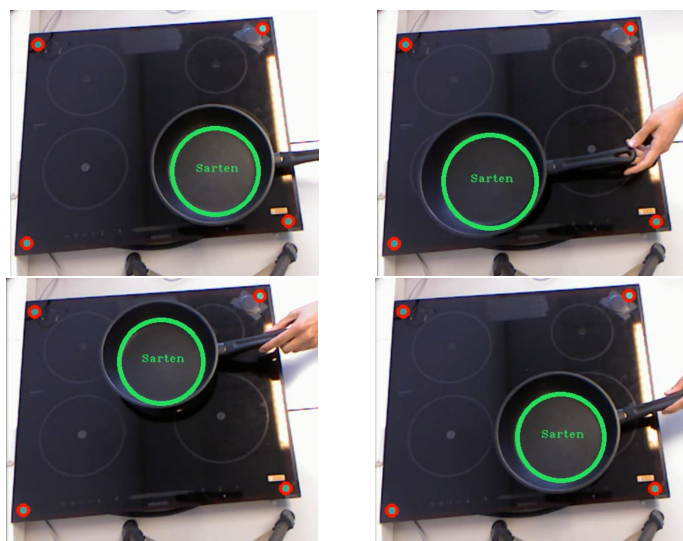
Para realizar este seguimiento, es importante que tengamos en cuenta tanto la imagen que recibimos en esa iteración, como la imagen de la iteración anterior. Para ser concretos, debemos tener en cuenta el radio y el centro de los contornos, que sirven para situar los recipientes e identificarlos como ya existentes, o como nuevos, de una iteración a otra. También nos ayudamos de la altura estimada, que nos permite realizar un seguimiento más robusto.

En cada iteración realizamos una comparación con los datos de la iteración anterior. Esta comparación es la que permite realizar el seguimiento de los recipientes, reconociendo si se trata de recipientes ya existentes, que pueden estar en el mismo sitio o en otro sitio diferente, o de nuevos recipientes. Puede haber movimiento a la hora de cocinar. El proceso de detección es lo suficientemente rápido como para considerar que el tiempo entre dos imágenes es pequeño. Esto implica que la variación en la posición de los recipientes es pequeña, siendo lo más probable que los recipientes guarden proximidad con su antigua posición.

También, es probable que los recipientes no se muevan todos a la vez, sino que se mueva uno, y después otro.

Se pueden dar diversos casos. El caso trivial se da cuando la placa está vacía. El programa entonces no encuentra ningún recipiente, por lo que no tiene ningún radio ni centro que encontrar.

Otro caso se da cuando se coloca un recipiente sobre la placa, encontrándolo e identificándolo con un radio y un centro. Es importante tener en cuenta que si el usuario decide mover ese recipiente, cambiándolo de sitio, debe ser identificado como el mismo. En la siguiente iteración, se busca el centro y el radio del recipiente. Si el recipiente se ha movido, el programa analiza si el nuevo centro está dentro del radio de la iteración anterior. Si esto se cumple, lo identifica como el mismo recipiente, almacenando el nuevo radio y el nuevo centro para poder realizar la comparación con la siguiente imagen que va a llegar. En una situación real, el recipiente no va a ser movido de una forma brusca, ya que hay contenido en su interior. A raíz de esta conclusión, es posible llevar a cabo el método anteriormente mencionado. Un busca de una mayor robustez del proceso, también se tiene en cuenta la altura del recipiente cuyo centro se encontraba dentro de ese radio. Si tiene la misma altura, dentro de una pequeña cota de error, se identifica como el mismo recipiente. El resultado de su seguimiento se observa en la Figura 3.4.



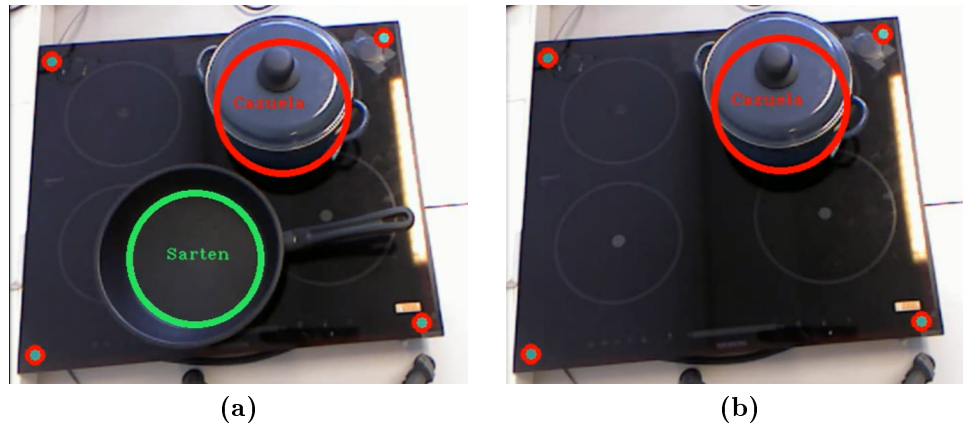
**Figura 3.4:** Seguimiento de una sartén sobre la placa

Cuando hay un recipiente y se introduce alguno más (Figura 3.5a), se da el caso de varios recipientes sobre la placa. Cada uno se identifica con su radio y su centro, igual que en el caso de un solo recipiente. La problemática radica en que siempre se mantenga la misma identificación para los mismos recipientes, de tal manera que cuando se retire uno de la placa, los otros se sigan identificando como el mismo. Esto se consigue asociando un grupo de variables que contengan la información de la ubicación en un vector, una para cada recipiente. De esta forma, si un recipiente se quita, esa componente del vector queda libre para asociarse con cualquier otro recipiente que se decida introducir. Como se muestra en la

### 3.3. INTERFAZ PARA USUARIO MEDIANTE LA IMAGEN RGB

---

Figura 3.5b, aunque se quite un recipiente, el otro se siguen identificando por el mismo color.



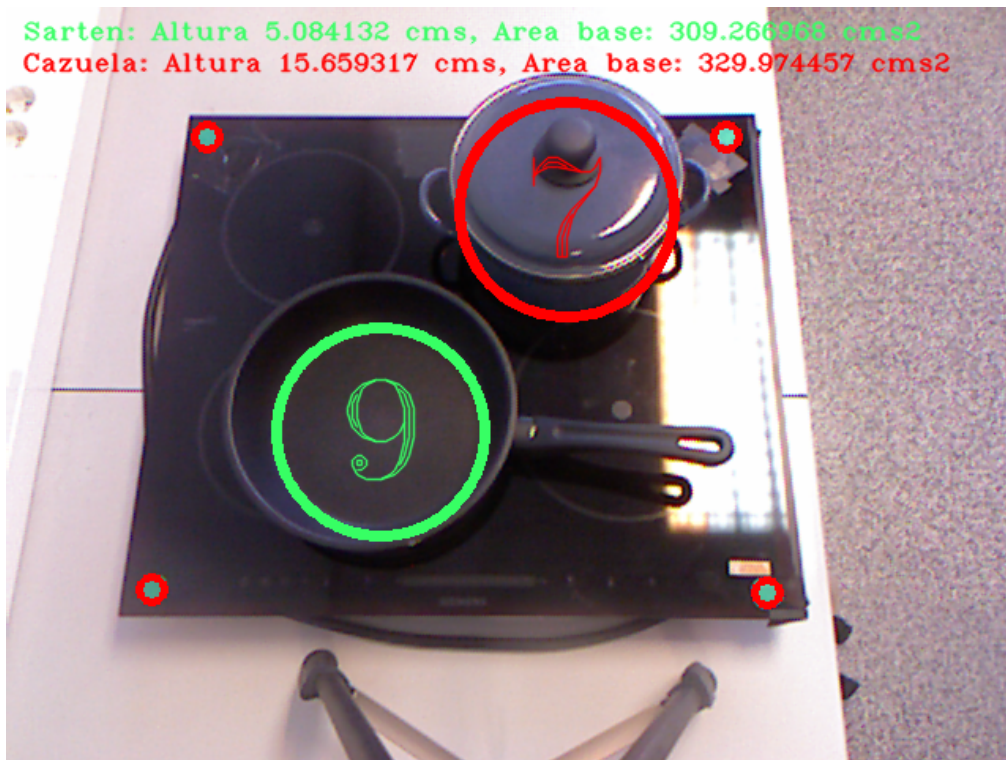
**Figura 3.5:** (a) Imagen de dos recipientes detectados (b) Se sigue identificado el segundo recipiente como el mismo

### 3.3. Interfaz para usuario mediante la imagen RGB

La creación de una imagen que sirva de interfaz para el usuario es necesaria para su control. En anteriores trabajos, donde sólo se podía utilizar la imagen de profundidad, no daba pie a crear esta interfaz, ya que visualmente no es intuitiva, ofrece menos información y visualmente no es del todo agradable. Uno de los objetivos de este proyecto es ofrecer información al usuario utilizando la imagen RGB para crear una interfaz, de tal forma que el usuario pueda ver todo lo que pasa en la cocina. Por ello, la imagen RGB que recibimos se modifica en cada iteración para ofrecer cierta información al usuario de forma gráfica: color del recipiente con el que es identificado, el tipo de recipiente, el área de la base y la altura del mismo. También podemos visualizar la potencia de cada uno de los recipientes, a medida que vayamos subiéndola o bajándola desde nuestro dispositivo móvil.

Para conseguir este objetivo, OpenCV nos proporciona distintas herramientas con las que modificar esta imagen, dibujando formas, colores o pudiendo introducir texto. Gracias a ellas se dibujan circunferencias de diferentes colores sobre cada uno de los recipientes detectados, utilizando el radio estimado. Cada recipiente tendrá un color distinto que lo identifique. El sistema es capaz de manejar un máximo de tres recipientes simultáneamente, que tendrán los colores verde, rojo y azul. El algoritmo utiliza el seguimiento para mantener el color de cada recipiente en diferentes iteraciones, y así identificarlos correctamente a la hora de controlarlos. Posteriormente, veremos que en la aplicación se nos permite elegir qué color de recipiente queremos manejar.

La introducción de texto en la imagen nos sirve para mostrar información a



**Figura 3.6:** Imagen RGB modificada para interfaz

tiempo real de lo que ocurre en la placa. La información aportada mediante el texto es:

- De qué recipiente se trata y tipo
- Altura
- Base
- Potencia

El color del texto se utiliza para identificar a qué recipiente se refiere la información mostrada. Los datos de tipo de recipiente, base y altura, son colocados en la parte superior de la pantalla. En cambio, la potencia es colocada en el centro del recipiente.

La imagen resultante se puede visualizar en la Figura 3.6.

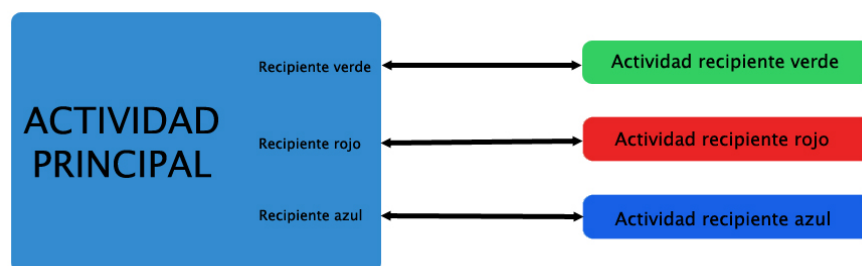
## Capítulo 4

# Aplicación móvil

En este capítulo, se describe la aplicación android que se lleva a cabo para realizar el control de la placa de cocina. Esta aplicación es la interfaz que tiene el usuario para interactuar con la cocina, por lo tanto, es importante que sea intuitiva, cómoda y fácil de usar.

### 4.1. Aplicación COCINA

Una aplicación de Android se compone de un conjunto de actividades. Las actividades son el elemento principal que muestra información en la pantalla del dispositivo para que pueda ser vista por el usuario. Vienen a ser el equivalente a las ventanas de un sistema operativo de escritorio. Para conseguir darle el aspecto que deseamos a la aplicación, hay que organizar lo que vamos a ver en la pantalla dentro de la propia actividad. De todo ello se encarga el *layout*, que es el contenedor de la vista de la actividad. Los layout nos permiten posicionar cada objeto gráfico en el lugar que queramos de la pantalla, diseñando el aspecto gráfico que va a tener nuestra aplicación.



**Figura 4.1:** Esquema explicativo de las actividades en la aplicación

La aplicación para el control de la placa está dividida en una actividad principal, donde podemos visualizar el estado de la cocina y actualizar imagen y potencias, y de tres actividades que derivan de ella, dependiendo del recipiente que se elija. En estas actividades podemos controlar la potencia del recipiente elegido, e incluso controlar el tiempo de lo que estamos cocinando. Se muestra un esquema en la Figura 4.1.

#### 4.1.1. Actividad principal

La actividad principal es sencilla y práctica. Como podemos ver en la Figura 4.2, a la parte derecha de la pantalla tenemos la imagen RGB del estado de la cocina, con la información de los recipientes.

En la parte izquierda están los botones que nos permiten saltar a la actividad relacionada con cada recipiente. La aplicación asigna a cada botón el recipiente correspondiente de la imagen. También tiene un botón de actualización de la imagen: cada vez que se presiona, se visualiza una nueva, y se obtienen las nuevas potencias (por si alguna de ellas hubiera cambiado).

De esta forma, el usuario sabe qué botón pulsar para controlar el recipiente que él quiera: verde, rojo o azul. En la siguiente sección analizamos estas actividades de cada recipiente. También nos encontramos con el botón de actualizar: cada vez que lo presionemos, podemos visualizar una nueva imagen, y obtener las potencias (por si alguna de ellas hubiera cambiado).

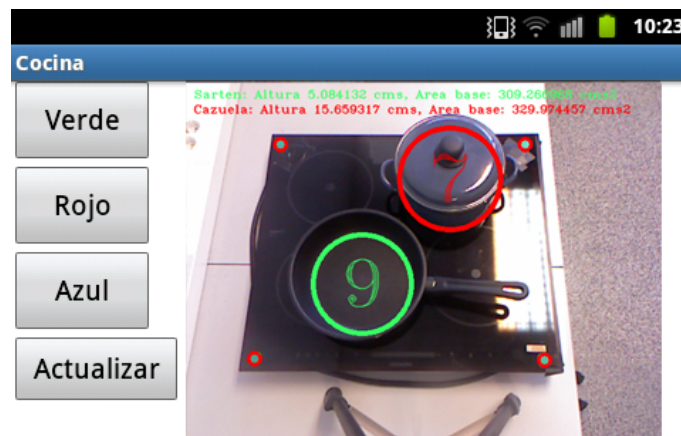


Figura 4.2: Actividad principal de la aplicación Android

#### 4.1.2. Actividad de los recipientes

Cuando el usuario se introduce en la actividad correspondiente al recipiente deseado, puede controlar la potencia del mismo y visualizar el tiempo que lleva el recipiente en la cocina. Pasamos a hacer una visión general del interfaz que proporciona cada una de estas actividades, mostrada en la Figura 4.3.

Como se puede apreciar, en la parte superior de la misma se muestra la actividad en donde nos encontramos. Después, tenemos la parte donde controlar la potencia, con dos botones: uno con el signo +, para subir un número de potencia, y otro con el signo - para bajarla. El rango en el que se mueven las potencias es de 0 a 9. La potencia resultante se actualiza debajo de los dos botones, de tal manera que el usuario puede conocer la potencia asignada en cada momento. Para una mejor percepción por parte del usuario acerca del recipiente que está controlando, el término *POTENCIA* está escrito en el color del recipiente correspondiente.

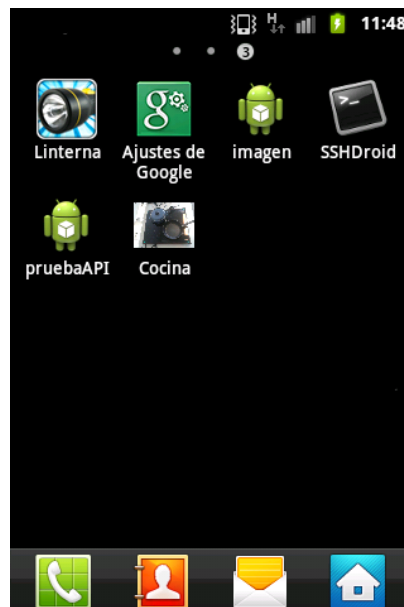




**Figura 4.3:** (a) Actividad recipiente verde (b) Actividad recipiente rojo

Junto a esta necesidad básica por parte de la aplicación de controlar la potencia, se ha implementado un temporizador como una idea surgida en el transcurso del proyecto. El usuario puede iniciarlo y pararlo en el momento que desee. De esta forma, se puede controlar el tiempo que un recipiente ha estado a una determinada potencia en la propia aplicación y para cada recipiente, ahorrando al usuario la molestia de tener que controlar el tiempo llevándolo por su cuenta, y por lo tanto, siendo más precisos en el tiempo de cocción.

Una vez programada y cargada la aplicación en nuestro dispositivo móvil, nos aparece el icono y el nombre de la aplicación junto con otras que podemos tener instaladas ya. En este caso, la aplicación se llamará *Cocina*, y está representada por una imagen de una placa con recipientes encima de la misma, tal como se muestra en la Figura 4.4.



**Figura 4.4:** Menú de Android donde se muestra el icono de la aplicación disponible

## 4.2. Comunicación PC-dispositivo

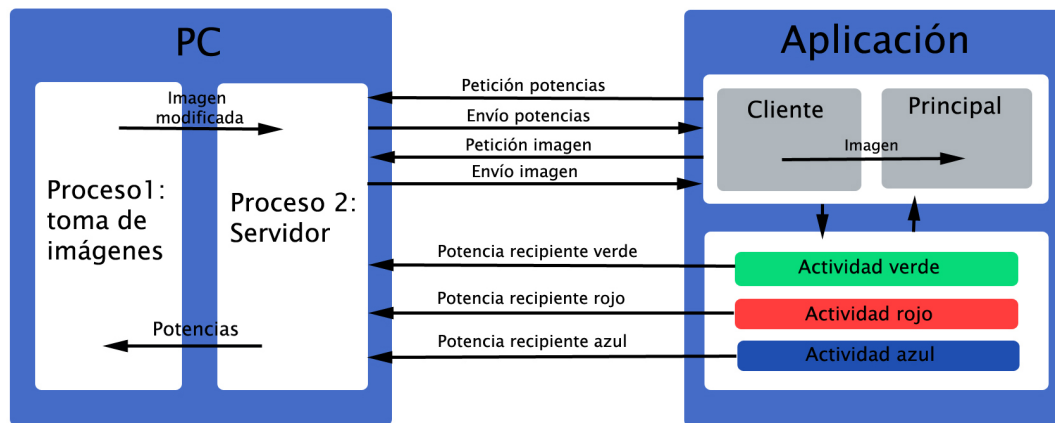
Esta sección explica los protocolos de sincronización y comunicación entre la aplicación y el sistema de detección de recipientes de la cocina. La conexión inalámbrica que se hace entre el ordenador y el dispositivo es vía WI-FI.

Para que dos programas se puedan comunicar entre sí, es necesario que un programa sea capaz de localizar al otro, y que además, ambos sean capaces de intercambiarse datos de forma que se entiendan. Para conseguir esta comunicación se utiliza un método que permite comunicar el lenguaje C++, que se utiliza en el ordenador, con una aplicación del sistema operativo Android, que se utiliza en el dispositivo móvil. Concretamente, se crea una comunicación servidor-cliente, donde el servidor es el ordenador (que tiene que enviar la imagen, y enviar y recibir los datos de potencia), y el cliente (que tiene que recibir la imagen y enviar y recibir los datos de potencia). Para ello se utilizan los sockets, una serie de funciones que se pueden implementar tanto en C++ como en Android.

En el apéndice F se encuentra toda la información referente a los sockets: qué son y cómo funcionan, así como una descripción de los protocolos que se pueden usar (TCP y UDP) . También se explica detalladamente y de manera más profunda cómo se realiza la comunicación servidor - cliente, necesaria en este proyecto y que es el eje central de la comunicación entre la aplicación móvil y el ordenador.

### 4.2.1. Protocolo de envío y recepción de datos

Para tener una visión global del protocolo de envío de datos, nos fijamos en la Figura 4.5. Este esquema resume de una forma general el intercambio de datos.



**Figura 4.5:** Esquema general del protocolo de comunicación entre servidor y cliente

En un primer momento, el servidor es puesto en marcha para permitir conexiones. A continuación, se arranca la aplicación que hace de cliente. En el momento de inicio de la misma, envía una petición para que el servidor le envíe las potencias, e inmediatamente después, el cliente vuelve a enviar otra petición, pero en

este caso indicándole que lo que le tiene que enviar es la imagen. De esta forma, tenemos inicializado los datos del programa al meternos en la aplicación.

El botón *actualizar* permite realizar esta operación cuando el usuario crea necesario. Cuando se presione este botón, el cliente vuelve a enviar una petición al servidor para que le envíe las potencias, y otra petición posterior a esta que le indique que lo que le tiene que enviar es la imagen. Así, el usuario puede actualizar la imagen y las potencias siempre que quiera.

El servidor tiene dos procesos: uno para el procesado de imagen y otro para las comunicaciones. La razón es que el servidor no bloquee la toma de imágenes, ya que tiene que esperar a que llegue la petición del cliente, y la captura de frames debe de ser continua. El proceso encargado de las comunicaciones, cuando reciba desde el cliente la petición que le indique que debe enviar la imagen, llama a la matriz que contiene la imagen RGB modificada, para así coger su contenido y poder mandarlo. Lo mismo pasa con las potencias, ya que las que le llegan al proceso servidor, deben ser pasadas al procesado de imagen para que la pueda ir mostrando por pantalla al mismo tiempo que se van modificando. Esto se explica de forma más detallada en el apéndice F.

Para el manejo de las potencias, hay una actividad respectiva a cada uno de los recipientes. Cada vez que subamos o bajemos la potencia del fuego, la envía al servidor. Como veremos a continuación, se manda la información tanto de la potencia como del recipiente al que pertenece.

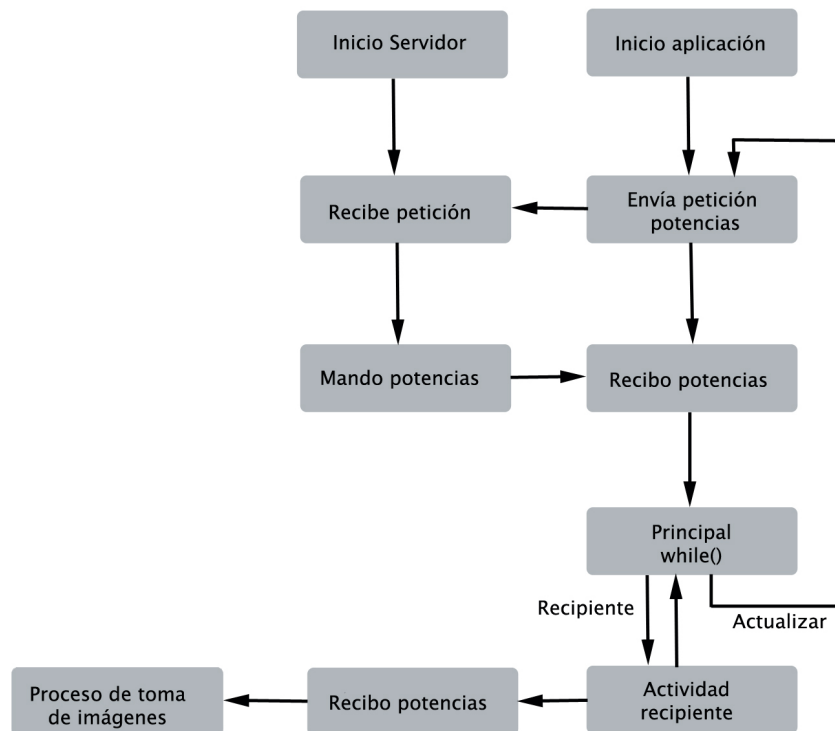
### 4.2.2. Protocolo de actualización de potencias

En esta sección vamos a centrarnos en la comunicación de potencias, y cómo se entienden tanto servidor y cliente para que esta transmisión funcione correctamente, tal como se muestra en la Figura 4.6.

Cuando el cliente contacta con el servidor en la actividad principal, le está enviando una petición que le indica que éste debe enviarle las potencias correspondientes a los recipientes. El servidor, al recibir esta petición, entiende que debe enviar la potencia, y por lo tanto manda al cliente un array de tres bytes, cada uno correspondiente a la potencia de uno de los recipientes: la primera posición es el verde, la segunda posición se trata del rojo, y la tercera posición es el recipiente azul. El cliente recibe las potencias y le otorga el valor correspondiente a las tres variables que contiene su programa principal. Así, cada vez que el usuario entre en la aplicación o presione el botón de actualizar, el cliente avisa con una petición que quiere recibir las potencias, y está preparado para recibirlas y actualizarlas.

La actividad propia de cada recipiente permite modificar las potencias desde la aplicación. En ella tenemos los botones para subir y bajar la potencia. Cuando presionamos uno de estos botones, se envía al servidor el nuevo valor de la potencia, pero concatenado con un *flag*, que le permite saber de qué recipiente estamos hablando. Los *flags* correspondientes son los siguientes:

- Verde: flag = 10; Rojo: flag = 11; Azul: flag = 12



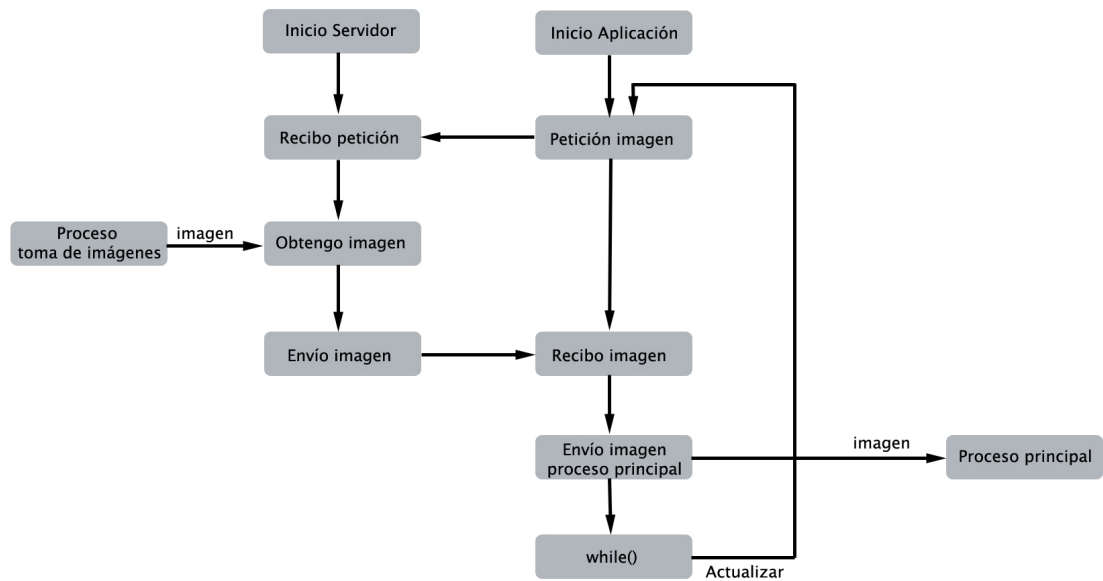
**Figura 4.6:** Esquema de la transmisión de potencias

Así, por ejemplo, si se sube la potencia del recipiente verde al 7, la aplicación envía el valor 107, formado por el flag 10 y la potencia 7; o si queremos subir la potencia del recipiente azul al 9, la aplicación envía en este caso el valor 129, formado por el flag 12 y la potencia 9. Así, el servidor es capaz de reconocer tanto recipiente al que se refiere como potencia. Una vez recibidas y almacenadas, son pasadas al procesamiento de imagen para que puedan ser mostradas por pantalla encima de cada recipiente.

#### 4.2.3. Protocolo de transmisión y sincronización de la imagen

En esta sección tratamos el tema de envío de imágenes, no sólo centrándonos en la comunicación que se da entre servidor y cliente, sino también viendo la comunicación entre los procesos dentro de estos (Figura 4.7).

Al comenzar la aplicación, ésta envía una petición cuyo valor le indica al servidor que le mande la imagen. Así, el proceso que actúa como servidor coge el valor de la matriz que en ese momento contenga la imagen RGB modificada, que ha sido creada en el proceso que se encarga del procesamiento de imágenes. Una vez que el servidor ha cogido esa matriz, la envía por una trama UDP. Para hacer la comunicación de forma eficiente, se aplica a la imagen un redimensionamiento de tamaño, permitiendo que la imagen tenga una calidad suficiente y que pueda



**Figura 4.7:** Esquema de la transmisión de imágenes

ser transmitida dentro de una trama. El mayor tamaño de una trama UDP que puede ser enviado es de 64 KB, incluyendo el tamaño de la cabecera UDP (8 Bytes) y el tamaño de la cabecera IP (20 Bytes). Una vez que hemos variado su tamaño, el servidor procede al envío de la imagen, que es recibida en el proceso que actúa de cliente en la aplicación. Este proceso está actuando "por detrás" de la actividad principal, permitiendo su ejecución a pesar de las funciones bloqueantes de recepción.

Este proceso, además de cuando se inicie la aplicación, se puede hacer siempre que el usuario lo desee por medio del botón *actualizar*. Así, cada vez que este botón sea pulsado, el usuario puede ver el estado de la cocina y de los recipientes que en ella se encuentran.



## Capítulo 5

# Conclusiones y futuras líneas de trabajo

### 5.1. Conclusiones y valoración del trabajo realizado

A continuación se expone la valoración sobre los objetivos propuestos y los resultados obtenidos. En la primera parte de este proyecto, hemos partido de la cámara Microsoft Kinect. Por ello, lo primero que se ha hecho es estudiar las posibilidades que nos ofrece. El segundo paso ha consistido en la realización de algoritmos de visión para la segmentación, detección y seguimiento de los recipientes. Todo ello se ha realizado utilizando una combinación de imágenes convencionales con las de profundidad.

Realizado el proceso que implica los algoritmos de visión para los recipientes, se pasó a trabajar para la creación de una aplicación en Android que permitiera el control de la placa. La realización de una aplicación en este sistema operativo me ha permitido trabajar en un lenguaje con el que no había tenido contacto, acercándome a una tecnología que está en alza en la actualidad, por lo que me siento satisfecho de lo aprendido. Creo que ha sido una muy buena oportunidad para aprender y espero en un futuro ampliar mis conocimientos en este campo.

La comunicación entre el móvil y el ordenador se ha realizado aprovechando una red doméstica, como puede ser la que forma un router en la mayoría de hogares. A partir de ahí, ha habido que buscar una forma de implementar un sistema de comunicaciones para la transmisión, por lo que se ha decidido el uso de socket, que se pueden implementar perfectamente tanto en C++ como en Android. El resultado ha sido la comunicación de datos e imágenes publicadas en el dispositivo móvil, permitiendo así controlar la cocina.

A raíz de todo ello, podemos afirmar que el resultado que se ha obtenido es satisfactorio. Se ha conseguido implementar un sistema capaz de realizar la detección y seguimiento de los recipientes de forma rápida y eficiente, ofrecer información sobre los mismos, y conseguir una comunicación entre PC y aplicación de potencias y de imagen que permita el control de la misma. Al ser un proyecto de investigación, ofrece la gran posibilidad de añadir nuevas ideas, algoritmos o detalles que puedan ir surgiendo a medida que se va avanzando.

Como conclusión y a raíz de lo expuesto, el proyecto cumple con los objetivos que en un principio nos habíamos fijado, además de introducir algunos detalles que han podido surgir durante la propia realización del mismo. Ha sido realizado

en su totalidad en las instalaciones del Centro Politécnico Superior, donde se me ha dado la posibilidad de llevarlo a cabo otorgándome el material necesario para ello.

Mi valoración del proyecto es altamente positiva y satisfactoria. Ha sido el reto más grande al que me he enfrentado durante la carrera, porque me ha servido no sólo para aplicar los conocimientos aprendidos durante estos años, sino para adquirir otros muchos, e iniciarme en otros entornos y lenguajes de programación, realizando un proyecto con tecnología vanguardista. Ha sido una experiencia gratificante que me ha proporcionado un gran desenlace al final de la vida universitaria e inicio de la carrera profesional.

### 5.2. Futuras líneas de trabajo

La detección y seguimiento de los recipientes son aspectos en los que se podría profundizar más, como el reconocimiento de nuevas formas de recipiente. En la realización de este proyecto, el reconocimiento que se ha hecho de los recipientes ha sido de forma circular. Sin embargo, pueden existir recipientes con diferentes formas (ovalados, cuadrados) donde la base de los mismos será diferente. El seguimiento sería más dificultoso y habría que hacerlo mediante su área, estableciendo una estrategia que permita reconocerlos y seguir sus movimientos. Otra idea sería obtener información del color de los objetos. Como hemos visto en este proyecto, se realiza un filtrado de color para realizar la automatización en la búsqueda de la placa. De la misma forma que esto es posible, se podría saber cuándo un alimento colocado en el fuego está cerca de quemarse y bajar la potencia de una forma automática, aumentando la seguridad y disminuyendo los riesgos.

En lo que se refiere a la aplicación, hay que tener en cuenta que se trata de un prototipo, lo que permite en un futuro seguir desarrollándola. En el apartado de interfaz, se podría conseguir que al presionar la pantalla en el recipiente deseado, se abriera la actividad que permite el control de ese recipiente, convirtiéndola en una aplicación más intuitiva. La mayor dificultad radicaría en cómo hacer corresponder los puntos dados (centro y radio) en la imagen redimensionada en la actividad principal. Siguiendo por el mismo camino, también se podría hacer que en la propia actividad del recipiente se tuviera una imagen haciendo zoom al mismo, de tal manera que el usuario podría ver de una forma más clara el recipiente que está tratando. Por último, una buena línea de trabajo sería conseguir visualizar la cocina no sólo por imágenes puntuales, sino de forma continua, como si se tratara de un video.

En otro aspecto, desde BSH se ha desarrollado un mecanismo de inducción móvil para el calentamiento de los recipientes sobre la placa, buscando una mejora en el aspecto energético. De esta forma, pasándole el centro y el radio donde está situado el recipiente, se encargaría de calcular el mejor algoritmo para su calentamiento, acudiendo a la superficie del mismo para calentarlo.



# Bibliografía

- [1] A. PÉREZ *Sistema basado en visión artificial y sensor de rango para control y reconocimiento en encimeras de inducción* Proyecto fin de carrera. Directores: Carlos Sagües. Departamento de Informática e Ingeniería de Sistemas. Universidad de Zaragoza, 2012.
- [2] H. MARTÍNEZ *Visión artificial para ayuda al control inteligente de encimeras de inducción* Proyecto fin de carrera. Directores: Carlos Sagües. Codirector: D. Paesa. Departamento de Informática e Ingeniería de Sistemas. Universidad de Zaragoza, 2009.
- [3] ROBOT OPERATING SYSTEM. <http://wiki.ros.org/>
- [4] C. CRICK, G. JAY, S. OSENTOSKI, O. C. JENKINS, *Ros and Rosbridge*. March 2012.
- [5] UBUNTU, [www.ubuntu.com](http://www.ubuntu.com)
- [6] OPENNI, <http://www.openni.org/>.
- [7] OPENCV [opencv.org](http://opencv.org)
- [8] R. JAIN, R. KASTURI y B. G. SCHUNCK, *Machine Vision*, MacGraw-Hill, 1995.
- [9] M. STRBAC, M. MARKOVIC, AND D. B. POPOVIC., *Kinect in Neurorehabilitation: Computer Vision System for Real Time Hand and Object Detection and Distance Estimation*
- [10] J. SMISEK, M. JANCOSSEK y T. PAJDLA, *3D with Kinect*, CMP, Dept of Cybernetics, FEE, Czech Technical University in Prague.
- [11] B. JÄHNE, *Digital Image Processing. Concepts, Algorithms and Scientific Applications*, 4th edition, Springer, 1997.
- [12] R. G. GONZÁLEZ y R. E. WOODS. *Digital Image Processing*, Addison-Wesley Publishing Company, 1992.
- [13] M. SONKA, V. HLAVAC y R. BOYLE, *Image Processing, Analysis and Machine Vision*. International Thomson Computer Press, 1993.

- [14] T. J. J. TANG, W. L. D. LUI, W. HO LI , *Plane-based detection of staircases using inverse depth*
- [15] C. DAL MUTTO, P. ZANUTTIGH, G. M. CORTELAZZO., *Fusion of Geometry and Color Information for Scene Segmentation*
- [16] Y. ZHAO, Z. LIUT, LU YANG AND H. CHENG, *Combing ROB and Depth Map Features for Human Activity Recognition*
- [17] Z. HUA, Y. LI, J. LI, *Image Segmentation Algorithm Based on Improved Visual Attention Model and Region Growing*
- [18] N. OTSU, *A Threshold Selection Method from Gray-Level Histograms*
- [19] I. SOBEL y G. FELDMAN, *A 3x3 Isotropic Gradient Operator for Image Processing*, presented at a talk at the Stanford Artificial Project in 1968, unpublished but often cited, orig. in Pattern Classification and Scene Analysis, R. Duda, P. Hart, John Wiley and Sons, '73, pp 271-272.
- [20] J. M. S. PREWITT, *Object enhancement and extraction*, Picture Analysis and Psychopictorics. Academic Press, New York, 1970.
- [21] L. G. ROBERTS, *Machine Perception of Three-Dimensional Solids. Optical and Electro-Optical Information Processing*, J. T. Tippett et al., Ed. Cambridge, MA: MIT Press, 1965.
- [22] J. CANNY, *A Computational Approach to Edge Detection*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 8, pp. 679-714, 1986.
- [23] A. SOLIS, A. NAYAK, M. STOJMENOVIC, N. ZAGUIA, *Robust Line Extraction Based on Repeated Segment Directions on Image Contours*
- [24] SUZUKI, S. y ABE, K., *Topological Structural Analysis of Digitized Binary Images by Border Following* CVGIP 30 1, pp 32-46 (1985).
- [25] P. J. ROUSSEEUW, *Robust Regression and Outlier Detection*, John Wiley & Sons, 1987.
- [26] G. BRADSKI, A. KAEHLER, *Learning OpenCV*
- [27] RONG ZHANG, *Automatic Computation of a Homography by RANSAC Algorithm* ECE661 Computer Vision Homework 4.
- [28] ECLIPSE, [eclipse.org](http://eclipse.org)
- [29] DEVELOPER ANDROID, [developer.android.com](http://developer.android.com)
- [30] F. M. MÁRQUEZ GARCÍA, *UNIX. Programación avanzada*, Ed. ra-ma.
- [31] J. TOMÁS GIRONES, *El gran libro de android*, Ed. Marcombo