

# Gestos Inteligentes: Reconocimiento de gestos aplicando la tecnología RFID

Tutor Nistal, Sergio

Curso 2014-2015

Directores: Raúl Parada y Kamruddin Nur

GRADO EN INGENIERÍA DE SISTEMAS AUDIOVISUALES



Universitat  
Pompeu Fabra  
Barcelona

Escola  
Superior Politècnica

## Trabajo Final de Grado



# GESTOS INTELIGENTES: RECONOCIMIENTO DE GESTOS APLICANDO LA TECNOLOGÍA RFID

Sergio Tutor Nistal

---

TRABAJO FINAL DE GRADO

INGENIERÍA DE SISTEMAS AUDIOVISUALES

ESCUELA SUPERIOR POLITÉCNICA UPF

2015

DIRECTORES DEL TRABAJO

Raúl Parada y Kamruddin Nur





A Aurora y a mis abuelos.



## **Agradecimientos**

Me gustaría mostrar un agradecimiento especial a los tutores de este proyecto, Raúl Parada y Kamruddin Nur, que han hecho posible que este proyecto haya sido realizado, y me han guiado a lo largo de estos meses, aconsejándome y proporcionándome las herramientas necesarias. También quisiera agradecer la oportunidad al grupo *UbiCa Lab* y a todos sus miembros.

A mi familia quisiera darles las gracias por haberme dado la oportunidad de estudiar y formarme, así como mostrarles mi más profundo agradecimiento por su esfuerzo e infinitas horas a mi lado. Y a mi chica, Inés, muchas gracias por haberme apoyado en los días más grises.



## Resumen

El objetivo de este proyecto final de carrera es estudiar y tratar de lograr una captura de movimiento ideal usando la tecnología *Radio Frequency Identification (RFID)*, que consiste en la recuperación remota de datos de etiquetas pasivas, usando antenas polarizadas dualmente.

Hay varias formas de abordar este objetivo, pero en este trabajo se presentan dos sistemas diferentes de captura de gestos sin necesidad de colocar una etiqueta en el usuario. Uno utilizará una estantería inteligente y *Support Vector Machine*, un conjunto de algoritmos de aprendizaje supervisado, y el otro consistirá de una superficie inteligente y un método de cuantificación vectorial llamado K-means.

Como parte de la investigación, compararemos y estudiaremos los puntos fuertes y débiles de ambos métodos, teniendo en cuenta los escenarios en los que están implementados. Finalmente, estos gestos serán asociados a diferentes acciones para controlar el reproductor multimedia VLC en tiempo real.

## **Abstract**

The aim of this final thesis is to research and try to achieve an ideal motion capture using the RFID (Radio Frequency Identification) technology, which consists on the wireless capture of data from passive tags using dual polarized antennas.

There are many ways to approach this objective, but in this project we present two different systems to capture gestures without having to place a tag on the user. One will use a smart shelf and *Support Vector Machine*, a set of supervised learning algorithms, and the other one will consist on a smart surface and a vector quantization method called K-means clustering.

As part of the research, we will compare and discuss the strengths and weaknesses of both methods, considering the different scenarios used to implement them. Finally, these gestures captured will be linked to different actions to control the VLC media player software in real time.

# ÍNDICE

<i>Agradecimientos</i> .....	<b>VII</b>
<i>Resumen</i> .....	<b>IX</b>
<b>LISTA DE FIGURAS</b> .....	<b>XIII</b>
<b>LISTA DE TABLAS</b> .....	<b>XIV</b>
<b>1. INTRODUCCIÓN</b> .....	<b>1</b>
1.1. Perspectiva.....	1
1.2. Descripción y objetivo del proyecto .....	1
1.3. Motivación.....	2
<b>2. ESTADO DEL ARTE</b> .....	<b>5</b>
<b>3. METODOLOGÍA</b> .....	<b>9</b>
3.1. Hardware del sistema <i>RFID</i> .....	9
3.1.1. <i>Reader</i> .....	9
3.1.2. Antena .....	10
3.1.3. Tags .....	11
3.1.4. Cables.....	12
3.2. Software del sistema <i>RFID</i> .....	13
3.2.1. AdvanNet .....	13
3.2.2. R .....	14
3.2.3. VLC media player .....	15
<b>4. EXPERIMENTOS Y RESULTADOS</b> .....	<b>17</b>
4.1. Estantería inteligente .....	17
4.1.1. Idea del sistema .....	17
4.1.2. Implementación.....	19
4.1.3. Problemática.....	22
4.2. Superficie inteligente.....	25
4.2.1. Idea del sistema .....	25
4.2.2. Implementación.....	28

4.2.3.	Evaluación.....	32
4.2.4.	Múltiples etiquetas en una antena .....	35
<b>5.</b>	<b>CONCLUSIÓN.....</b>	<b>39</b>
<b>6.</b>	<b>TRABAJO FUTURO.....</b>	<b>41</b>
<b>7.</b>	<b>REFERENCIAS.....</b>	<b>43</b>
7.1.	Referencias Bibliográficas.....	43
7.2.	Referencias lista de figuras .....	45
7.3.	Referencias lista de tablas.....	46
<b>8.</b>	<b>ANEXOS.....</b>	<b>47</b>

## LISTA DE FIGURAS

Figura 3.1. Lector AdvanReader-100 .....	9
Figura 3.2. Advantenna-p11 .....	10
Figura 3.3 Estructura de etiqueta pasiva .....	11
Figura 3.4 Cable micro coaxial.....	13
Figura 4.1. Disposición de los tags (cajitas) en la estantería inteligente.....	18
Figura 4.2. Parte posterior de la Smart Shelf .....	18
Figura 4.3. Ejemplo de archivo .csv .....	19
Figura 4.4. Ejemplo de matriz donde guardamos cada dato de la etiqueta "T" respecto a cada antena "A" .....	20
Figura 4.5. Ejemplo de gráfica de fase con mano .....	20
Figura 4.6. Ejemplo de clasificación de SVM en dos categorías, usando funciones Kernel .....	21
Figura 4.7. Resultados del predictor y coeficiente de precisión .....	22
Figura 4.8. Disposición de antenas y etiquetas.....	23
Figura 4.10. Gráficas de RSSI.....	24
Figura 4.11. Cara delantera de la placa con las etiquetas pegadas.....	26
Figura 4.12. Cara trasera de la placa con las antenas.....	27
Figura 4.13. Gráfica y movimiento correspondiente a Play .....	30
Figura 4.14. Gráfica y movimiento correspondiente a Stop .....	30
Figura 4.15. Gráfica y movimiento correspondiente a Siguiente canción.....	31
Figura 4.16. Gráfica y movimiento correspondiente a Canción anterior .....	31
Figura 4.17. Gráfica y movimiento correspondiente a Subir volumen .....	31
Figura 4.18. Gráfica y movimiento correspondiente a Bajar volumen .....	32
Figura 4.21. Disposición de las dos etiquetas en el sistema probado.....	36

<b>Figura 4.22. Gráfica RSSI dos etiquetas y un tag.....</b>	<b>36</b>
<b>Figura 9.1. Diagrama de Gantt del proyecto.....</b>	<b>52</b>

## **LISTA DE TABLAS**

<b>Figura 4.9. Tabla de ajuste de potencia de lectura .....</b>	<b>23</b>
<b>Figura 4.19. Tabla con recuento de tiempo y compendio de gestos realizados del primer vídeo .....</b>	<b>34</b>
<b>Figura 4.20. Tabla con recuento de tiempo y compendio de gestos realizados del segundo vídeo .....</b>	<b>35</b>

# 1. INTRODUCCIÓN

## 1.1. Perspectiva

La tecnología *RFID* junto al Internet de las cosas es un mercado en crecimiento y abre las puertas a nuevas oportunidades y soluciones para controlar, localizar y seguir cualquier bien material o personas [1].

Esta tecnología está basada en la comunicación remota entre dispositivos. Básicamente los sistemas *RFID* se componen de tres partes, primero el lector, que se encarga de emitir señales de radiofrecuencia mediante antenas, el segundo componente, que reciben las etiquetas, el tercer componente, y devuelven otra señal a modo de respuesta. Una vez procesada esta señal, se puede extraer la identificación de la etiqueta y más información que nos pueda resultar útil [2].

Este proyecto quiere explorar las posibilidades del reconocimiento de gestos utilizando diferentes escenarios inteligentes basados en la tecnología *RFID* explicada anteriormente.

## 1.2. Descripción y objetivo del proyecto

Hasta ahora, ha habido numerosos proyectos en los que se ha localizado un objeto concreto o una parte del cuerpo colocando un *tag* y moviéndolo para que las antenas detectaran variaciones de parámetros, o haciendo uso de sensores de movimiento y proximidad [3] [4], pero el objetivo de nuestra investigación será tratar de saber qué movimiento ha hecho el usuario sin necesidad de que éste lleve una o varias etiquetas encima para monitorear sus gestos. Por ejemplo, podemos imaginar una tienda de CDs con caratulas de algunos discos colgadas en las paredes a modo de promoción. El cliente acerca la mano y empieza a sonar alguna canción de éste. De esta forma se ha detectado la mano del usuario pero no ha habido necesidad de que el cliente lleve un *tag*

encima. En casa, por ejemplo, también podemos pensar en programar ciertas características, como la luz o la temperatura, y modificarlas mediante gestos.

Para llegar a este objetivo, será necesario analizar y discutir posibles soluciones para el reconocimiento de gestos, y una vez sacadas conclusiones, tratar de maximizar los resultados y la efectividad del sistema.

El modelo que utilicemos tiene que seguir unas pautas que enumeramos a continuación:

- 1) El lector procesa las señales enviadas por las antenas y *AdvanNet* nos proporciona archivos *.csv* con información útil
- 2) Se extraen los parámetros y valores de alto y bajo nivel que necesitamos (potencia *RSSI*, identificación de *tag* y antena...)
- 3) Procesamos estos datos
- 4) Se aplica el método elegido de aprendizaje (supervisado o no supervisado)
- 5) El sistema tiene que ser capaz de detectar donde hay una variación de valores y por tanto donde se ha producido un movimiento
- 6) Se añade alguna función práctica a la detección del movimiento

Como queremos reconocer el gesto en tiempo real, la parte más importante a tener en cuenta serán los valores del sistema en reposo (sin mano), cuando empezamos a hacer nuestro movimiento y sobretodo, cuando acabemos y el sistema vuelva a la normalidad.

Finalmente, una vez procesado y recogido el movimiento, enlazaremos nuestro sistema con el reproductor de música VLC para así ofrecer una respuesta auditiva a los gestos que se hagan.

### **1.3. Motivación**

La decisión de llevar a cabo este proyecto viene en gran parte debido a la oportunidad de explorar e investigar una tecnología que hasta ahora me era prácticamente desconocida pero se ha revelado como una fascinante forma de localizar y extraer información de forma poco intrusiva para el usuario.

Mi primer y único contacto hasta ahora con *RFID* fue en la asignatura optativa “ El Internet de las Cosas”, impartida por Rafael Pous, llamándome bastante la atención el aspecto cotidiano de las aplicaciones que nos fueron mostradas. Es decir, el gran uso y provecho que se le puede sacar a esta tecnología para ayudarnos en muchas tareas y mejorar otros servicios.

Aparte del uso que le hemos dado en este proyecto, el reconocimiento de gestos tiene un abanico de aplicaciones muy amplio y útil. Puede suponer una mejora en varios campos científicos y cotidianos, como por ejemplo la seguridad -previniendo robos, mejorando los nuevos métodos de pago como el NFC-, la automatización de los hogares – controlando tu casa con una simple combinación de gestos-, la publicidad –capturando y estudiando cómo actúan las personas en las tiendas y en diferentes situaciones-, logística –evitando fallos de posicionamiento de productos y fácil detección de inventario-, los videojuegos y muchas más.

Si se sigue investigando y apostando por el uso de *RFID* y tecnologías similares como forma de extracción y análisis de la información que la gente -y los objetos que nos rodean- proporcionamos día tras día, no sólo avanzaremos en diversos campos como los mentados antes, sino que podremos optimizar nuestro tiempo y recursos, facilitando nuestra vida cotidiana y la forma en que trabajamos.

También gracias a este proyecto se me ha dado la oportunidad de aprender un nuevo lenguaje de programación, R, el cual desconocía totalmente, y quien sabe, puede que me sea de ayuda en un futuro.



## 2. ESTADO DEL ARTE

Aquí se presentan brevemente varios artículos y proyectos que han sido de mucha ayuda para profundizar en el funcionamiento de la tecnología *RFID* y su uso en el reconocimiento de gestos.

### ***"Human-object Interaction Reasoning using RFID-enabled Smart Shelf"*** [5]

Este artículo presenta una técnica de detección de interacciones entre humanos y objetos en tiempo real usando algoritmos de aprendizaje automático. La primera parte de la técnica propuesta trata sobre analizar los parámetros *RFID* de bajo nivel -la potencia (*RSSI*) y la fase (*RFP*)-, y algunas características de alto nivel que son útiles para el procesamiento posterior de datos como por ejemplo qué antena ha leído qué *tag*, cuantas veces un *tag* ha sido leído en un ciclo, el tiempo y el código de identificación.

Después de esto, mediante técnicas de aprendizaje automático y aplicando formulas probabilísticas, el sistema es entrenado para detectar y clasificar la actividad humana basándose en los parámetros anteriormente mencionados.

La parte de *machine learning*, y en gran medida la parte de captación y análisis de datos, resulta muy interesante ya que hay muchas similitudes con parte del trabajo llevado a cabo en mi proyecto.

### ***"LSI-REC: a Link State Indicator based gesture recognition scheme in a RFID system"*** [6]

En este artículo se estudia una técnica de reconocimiento usando una formación de *tags* y capturando los parámetros de las inmediaciones de su sistema *RFID*. El uso de la potencia *RSSI* se discute en el artículo pero los autores prefieren utilizar el *Link State Indicator* (ratio de número de *tags* que son leídos por unidad de tiempo) para indicar el cambio de la señal de estos dispositivos ya que es más preciso al detectar brazos, piernas y las posibles obstrucciones que se pueden cometer al realizar gestos.

Finalmente se crea una matriz de gestos a partir de las características extraídas para así representar los gestos y se aplica un método estadístico de discriminación lineal para diferenciar los gestos.

El artículo es útil a nivel teórico para comprender como funciona la captación de gestos pero no tanto a la hora de la práctica ya que se distancia mucho de este proyecto al no usar los mismos parámetros *RFID*.

### ***"Passive RFID Gesture Recognition: Final Report"*** [7]

Este trabajo de la Universidad de Rutgers es uno de los pilares básicos sobre los que cimiento mi proyecto. Los autores investigan las posibilidades de uso de la tecnología *RFID* y los *tags* para y crean una interfaz bastante efectiva para reconocer gestos.

La arquitectura del sistema empleado difiere un poco sobre que se ha presentado aquí ya que los autores utilizan un lector por cada antena, pero al final acaban ordenando los datos leídos según el tiempo de captación.

Como en este proyecto, se capta y analiza la variación de parámetros de alto y bajo nivel en tiempo real -el más importante es *Radio Signal Strength Indication (RSSI)*- y se compara con unos gestos anteriormente definidos. Finalmente estos gestos son asociados a varias acciones para controlar un reproductor de música, siendo la fuente de inspiración de la parte final de este proyecto.

### ***"Gesture recognition using RFID technology"*** [8]

En este artículo se presenta un sistema diseñado para detectar gestos, formado por varios *tags* que son rastreados en una interfaz intuitiva.

El trabajo que proponen es interesante ya que, como en este proyecto, tratan de ser lo menos intrusivos posible. El uso de etiquetas pasivas como en nuestro sistema y la creación de una superficie inteligente también se asemejan a nuestro trabajo, y además

nos proporcionan información muy útil sobre las antenas y su haz. Por contrario, la parte de predicción es diferente, ya que ellos utilizan un árbol de hipótesis bastante complejo con el que consiguen detectar muchos y diferentes movimientos en su área de lectura, así como varios gestos al mismo tiempo.

***“A comprehensive operating room information system using the Kinect sensors and RFID” [9]***

En este artículo se presenta una sala de operaciones acondicionada con tecnología *RFID* y sensores Kinect para ayudar a los cirujanos en su tarea, que no solo reside en operar sino que también tienen que llevar un control de la operación y de los utensilios que se van a usar.

Gracias a este trabajo podemos conocer más sobre los sensores Kinect, que consisten en dos sensores de profundidad que rastrean al usuario en 3 dimensiones. Aparte, también se añade una cámara de color y cuatro micrófonos. El resultado de utilizar todos estos dispositivos es una detección de movimiento, cara y voz del usuario en tiempo real, cosa que facilita al cirujano en este caso el controlar otros programas mediante gestos y voz. El sistema *RFID* implementado se utiliza para localizar a las personas en sus diferentes roles (cirujanos, pacientes...) y objetos del inventario.

Como podemos ver, para este proyecto se usan los sensores Kinect para reconocimiento de gestos a nivel más reducido, ya que la cámara limita en cierta medida el rango de utilidad de los sensores, y en cambio, el sistema *RFID* se utiliza para rastrear en todo el hospital, ya que se trata de un sistema que no necesita tener visión directa con el usuario y está más enfocado a la rápida identificación de objetos y personas.



## 3. METODOLOGÍA

### 3.1. Hardware del sistema *RFID*

En esta sección se describen los componentes físicos de nuestro sistema *RFID* y se explica brevemente para que se utilizan y como han sido usados en este proyecto.

#### 3.1.1. *Reader*

La función del lector es enviar pulsos de energía por ondas de radio mediante antenas (próximo apartado) que identifican los *tags* que haya en su área de transmisión. Estos pulsos son detectados por las etiquetas, que devuelven una señal a modo de respuesta.

El lector usado en este proyecto es el *AdvanReader-100* de la empresa *Keonn Technologies*. Este *reader* tiene 4 puertos que le dan capacidad para trabajar hasta con 1024 antenas, gracias a su compatibilidad con multiplexores de 8, 12 y 16 puertos.

Su apartado técnico, incluyendo rango de frecuencias, interfaz de datos, protocolos, etc. se puede consultar en el punto 10 de la bibliografía [2] [8] [10].



Figura 3.1. Lector *AdvanReader-100*. Fuente: <http://keonn.com/rfid-components/readers/advanreader-100.html>

### 3.1.2. Antena

Como hemos visto antes, las antenas son las encargadas de la comunicación entre lector y *tag*. Éstas emiten señales de radiofrecuencia y también son las que reciben las de las etiquetas como respuesta.

El diseño elegido para el sistema de antenas es crítico para el buen funcionamiento de la transmisión de energía. Un buen sistema tendrá en cuenta el área tridimensional de energía que cada antena crea, el rango de lectura de las etiquetas y la orientación del campo electromagnético (polarización).

La antena que se usa en este proyecto es la llamada *Advantenna-p11* de la empresa *Keonn Technologies*. Esta antena es ideal para su uso en estanterías inteligentes, pantallas inteligentes, paneles inteligentes u otras superficies gracias a su delgadez y su patrón de radiación, que teóricamente, se caracteriza por dirigirse en todas las direcciones de un hemisferio.

A la hora de la práctica, sin embargo, he podido comprobar que el haz de la señal es realmente diagonal. Ver apartado "Estantería inteligente" del capítulo 5 "Experimentos" para entender esta afirmación.

Para más información de aspecto técnico, consultar el apartado 11 de la bibliografía [2] [11] [12].

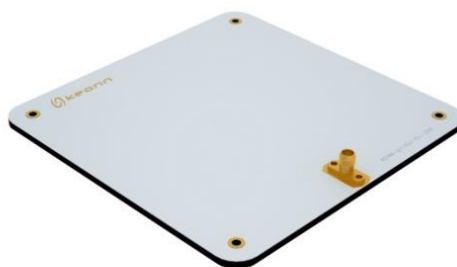


Figura 3.2. Advantenna-p11. Fuente: <http://keonn.com/rfid-components/antennas/advantenna-p11.html>

### 3.1.3. Tags

Los *tags* o etiquetas están formados por un microchip de silicio con un circuito integrado y una antena que se ocupan de la transmisión de señales y memoria interna para el almacenaje de datos.

Hay tres tipos de etiquetas, que se clasifican según el tipo de alimentación que tienen.

#### 1. *Tags* pasivos:

Las etiquetas pasivas son las utilizadas en este proyecto. No reciben alimentación propia, sino que utilizan la que reciben del campo electromagnético que los lectores *RFID* emiten. La señal enviada por el lector abastece de energía el chip del *tag* y le permite enviar una señal de respuesta, que contiene un número de identificación ID y si se le añade una memoria, puede contener información adicional.

La estructura es sencilla, ya que solo constan de un microchip integrado y la antena unidos mediante sustrato.

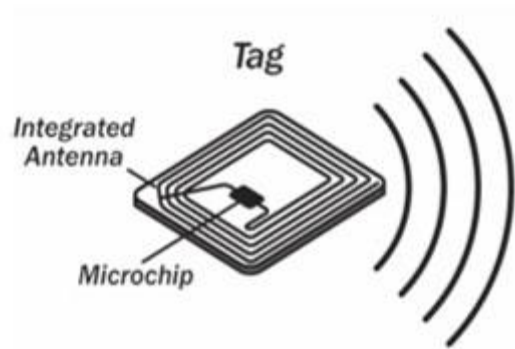


Figura 3.3 Estructura de etiqueta pasiva. Fuente: <http://cdn.barcodesinc.com/cats/rfid-readers/tag.jpg>

A pesar de su corto alcance -hablamos de pocos metros- este tipo de etiqueta es el más utilizado debido a su bajo precio, buena autonomía y reducido tamaño, cosa que permite su implementación en gran cantidad de dispositivos [2] [13] [14].

## 2. *Tags* activos

Estas etiquetas, en cambio, tienen implementado un sistema de alimentación propio que se comunica constantemente con los lectores mediante una señal. Esto aumenta significativamente su alcance de lectura y ancho de banda, siendo menos propensos a interferencias. Además, admiten la integración de otros sensores (luz, temperatura, etc).

Las desventajas de este tipo de *tag* son su elevado precio y tamaño, cosa que no las hace aptas para cualquier tipo de dispositivo [2] [13] [14].

## 3. *Tags* semiactivos/semipasivos

Conocidos con ambos nombres, estos *tags* también tienen su propia fuente de alimentación pero su objetivo es alimentar el microchip y no activar la antena, como era el caso en las etiquetas activas.

A pesar de ser más rápidos en cuestión de tiempo de respuesta y mayor durabilidad que los activos, apenas se usan [2] [13] [14].

### **3.1.4. Cables**

Los cables usados para este proyecto son dos:

- Cable coaxial SMA: Utilizada para la conexión entre lector y antenas.



Figura 3.4 Cable coaxial. Fuente: [www.amazon.com](http://www.amazon.com)

- Ethernet: Se utiliza para conectar el lector y el ordenador a la red.

## 3.2. Software del sistema RFID

En esta sección se describen los programas que forman parte de nuestro sistema y se explica brevemente para que se han utilizado.

### 3.2.1. AdvanNet

Para la realización de las pruebas se ha utilizado la versión de *AdvanNet 2.1.4*. Esta versión ofrece una ventaja en comparación a otras del mismo *software*, ya que proporciona toda la información de cada antena respecto a cada *tag* en el momento de la lectura, mientras que otras versiones sólo nos proporcionan la información de la mejor antena que lee cada *tag*, resultando en unos archivos *.csv* poco concisos, aunque con menos volumen de datos y por tanto más fáciles de procesar.

Para que el programa se adapte a nuestras necesidades empezamos por modificar dos archivos en el apartado de configuración, `adrd-m4-100.xml` y `connectors.xml`.

En el primero de estos podemos establecer la IP del lector, el número de canales (dos en este caso para una lectura más precisa, y poder filtrar la fase), el tiempo y potencia de lectura, y seleccionar las antenas que queremos que emitan señal, modificando las coordenadas que hacen referencia al puerto de la antena, el multiplexor de primer nivel y el multiplexor de segundo nivel, si es que fuese necesario.

En el segundo archivo se determinan el número de ciclos de lectura y se filtra el código electrónico de producto, el *EPC*, para que *AdvanReader* sólo nos muestre información de nuestros *tags*, que son los que nos interesan.

### **3.2.2. R**

El código para analizar y tratar los datos ha sido desarrollado bajo el lenguaje R, debido a que es un lenguaje de programación que nos permite extraer y manipular la información con mucha facilidad. Por ello, este lenguaje está muy enfocado a la estadística, el análisis y la minería de datos.

Además, tiene una gran funcionalidad a la hora del aprendizaje automático, ya que mediante sencillas funciones ya establecidas por el propio R, podemos establecer conjuntos de datos para entrenar y probar nuestro sistema.

R es *Open Source*, y es una ventaja para nosotros ya que tiene una gran cantidad de información y recursos online al alcance de la mano. El programa gratuito que hemos utilizado para crear el código y las gráficas, *RStudio*, nos permite además descargar e instalar una extensa variedad de librerías y paquetes externos [15].

### **3.2.3. VLC media player**

Los gestos captados por el sistema serán utilizados para controlar el reproductor multimedia VLC.

Este programa es de código abierto y reproduce casi cualquier archivo multimedia, aunque en nuestro caso sólo utilizaremos archivos de música [16].



## 4. EXPERIMENTOS Y RESULTADOS

En este capítulo se detallan los pasos que llevamos a cabo para experimentar y tratar de validar los diferentes sistemas propuestos. Estos métodos son peculiares respecto a otros trabajos sobre *RFID* ya que en nuestro caso el usuario no porta ningún *tag*, y mucho menos una antena.

Tampoco utilizamos sensores de movimiento, sino que estudiamos la variación de la potencia *RSSI* al acercar la mano a un *tag* estático en frente de una antena. Para ello, tomamos los archivos .csv que utilizamos para el entrenamiento del sistema, extraemos los parámetros que nos interesan y los comparamos con los datos leídos mientras el sistema se está ejecutando.

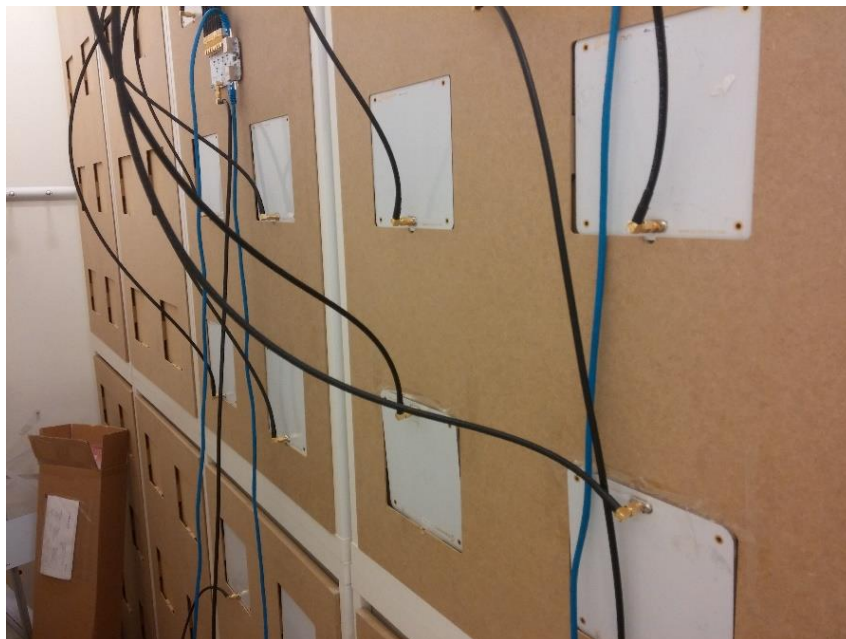
### 4.1. Estantería inteligente

#### 4.1.1. Idea del sistema

La idea original del proyecto fue la de utilizar la estantería inteligente del laboratorio *UbiCA Lab*. Como se puede ver en las imágenes de a continuación, la estantería está formada por varias celdas que contienen dos antenas. El número de antenas leyendo debe ajustarse en *AdvanNet*, así como el multiplexor al que están conectadas, si fuese el caso. El uso de los multiplexores nos permite abastecer el máximo número de antenas posibles. Nosotros usamos 8 antenas y 8 etiquetas pasivas colocadas en la tapa de unas cajas de perfume, dispuestas como se observa en las imágenes de a continuación.



*Figura 4.1. Disposición de los tags (cajitas) en la estantería inteligente*



*Figura 4.2. Parte posterior de la Smart Shelf*

## 4.1.2. Implementación

Situamos cada etiqueta en frente de cada antena, habiendo una separación aproximada de 30 centímetros, y procedemos a tomar las medidas para el *machine learning*. Los datos que nos interesan son la potencia *RSSI*, la identificación del *tag* y la antena que ha leído.

HEX_EPC	ANTENNA_PORT	MUX_PORT	TIME_STAMP	RSSI	FREQ	RF_PHASE	READ_COUNT
abcf000000000003	2		0 2015/06/09 13:40:24.310	-38	865549	160	1
abcf000000000003	2		0 2015/06/09 13:40:13.354	-38	865549	160	1
abcf000000000001	4		0 2015/06/09 13:40:26.129	-36	865549	47	1
abcf000000000003	2		0 2015/06/09 13:40:26.623	-38	865549	165	1
abcf000000000001	4		0 2015/06/09 13:40:36.605	-36	865549	45	1
abcf000000000004	1		0 2015/06/09 13:40:21.938	-36	866061	42	1
abcf000000000001	4		0 2015/06/09 13:40:18.977	-36	865549	47	1
abcf000000000004	1		0 2015/06/09 13:40:41.237	-36	866061	39	1
abcf000000000004	1		0 2015/06/09 13:40:10.269	-36	866061	39	1
abcf000000000001	4		0 2015/06/09 13:40:23.617	-36	865549	45	1
abcf000000000003	2		0 2015/06/09 13:40:31.732	-38	865549	160	1
abcf000000000003	2		0 2015/06/09 13:40:17.565	-38	865549	163	1
abcf000000000003	2		0 2015/06/09 13:40:40.499	-38	865549	163	1
abcf000000000003	2		0 2015/06/09 13:40:44.888	-38	865549	160	1
abcf000000000003	2		0 2015/06/09 13:40:22.198	-38	865549	163	1
abcf000000000001	4		0 2015/06/09 13:40:12.910	-36	865549	47	1
abcf000000000002	3		0 2015/06/09 13:40:25.883	-35	866061	39	1
abcf000000000004	1		0 2015/06/09 13:40:28.283	-36	866061	42	1
abcf000000000003	2		0 2015/06/09 13:40:19.869	-38	865549	160	1
abcf000000000002	3		0 2015/06/09 13:40:40.745	-35	866061	39	1
abcf000000000001	4		0 2015/06/09 13:40:17.080	-36	865549	45	1
abcf000000000002	3		0 2015/06/09 13:40:36.358	-35	866061	39	1
abcf000000000004	1		0 2015/06/09 13:40:19.225	-36	866061	39	1
abcf000000000001	4		0 2015/06/09 13:40:11.024	-35	865549	45	1
abcf000000000004	1		0 2015/06/09 13:40:26.378	-36	866061	39	1
abcf000000000002	3		0 2015/06/09 13:40:31.978	-35	866061	33	1
abcf000000000002	3		0 2015/06/09 13:40:12.583	-35	866061	39	1
abcf000000000003	2		0 2015/06/09 13:40:15.461	-38	865549	160	1
abcf000000000004	1		0 2015/06/09 13:40:17.323	-36	866061	39	1

Figura 4.3. Ejemplo de archivo .csv

Para ello, creamos una matriz donde almacenar los datos que nos proporcionaba cada *tag* respecto a cada antena y poder hacer el aprendizaje del sistema posteriormente. Cada posible estado se define con un *label* (0-8) que dirá en qué posición está la mano, o si no lo está.

Cabe decir que en la figura 4.4 se puede observar como en un principio decidimos guardar los valores de fase, pero finalmente optamos por no hacerlo, ya que a estas distancias y con los problemas que más adelante veremos, la fase era demasiado caótica y no nos daba ninguna información de relevancia (figura 4.5).

A1T3PHASE	A1T4RSSI	A1T4PHASE	A2T1RSSI	A2T1PHASE	A2T2RSSI	A2T2PHASE	A2T3RSSI	A2T3PHASE
0	0	0	-67	22	-69	53	-68	22
0	0	0	-66	33	-69	59	-69	0
0	0	0	-67	14	-70	95	-68	14
0	0	0	-66	42	-70	70	-68	16
0	0	0	-64	19	-70	73	-68	5
0	0	0	-63	115	-69	118	-68	11
0	0	0	-62	120	-69	126	-69	16
0	0	0	-63	118	-68	56	-68	8
0	0	0	-63	106	-68	92	-67	30
0	0	0	-63	112	-68	112	-68	2
0	0	0	-62	135	-68	50	-69	2
0	0	0	-62	95	-69	104	-68	2
0	0	0	-62	61	-68	56	-68	2
0	0	0	-64	59	-68	120	-68	16
0	0	0	-65	47	-66	78	-67	19
0	0	0	-66	36	-67	64	-68	14
0	0	0	-65	59	-67	123	-67	2
0	0	0	-66	50	-68	123	-66	14
0	0	0	-66	64	-68	64	-66	8
0	0	0	-66	25	-67	123	-67	19
0	0	0	-66	14	-67	112	-67	11

Figura 4.4. Ejemplo de matriz donde guardamos cada dato de la etiqueta "T" respecto a cada antena "A"

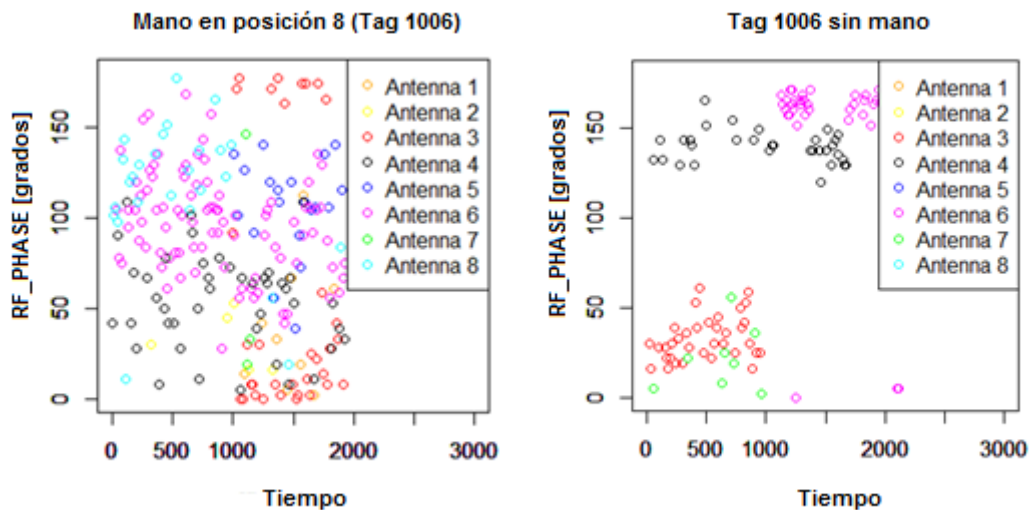


Figura 4.5. Ejemplo de gráfica de fase con mano y sin mano

Una vez tenemos la matriz completa, procedemos a aplicar un algoritmo de máquina de soporte vectorial (*Support Vector Machine*). Este método de aprendizaje supervisado nos permite clasificar un subconjunto de valores en diferentes categorías, y predecir si un valor concreto pertenece a una categoría o a otra.

En el código usamos la función *ksvm*, en la que previamente seleccionamos un 80% de los datos extraídos para hacer el *learning* y otro 20% para hacer pruebas de precisión. Configuramos la función para que la clasificación que llevamos a cabo sea multi-clase, es decir, que los datos que introducimos sean detectados como uno de los 9 posibles estados. Para ello, esta función utiliza a su vez otra función llamada *Kernel*, que nos permite clasificar nuestros datos de forma no lineal.

Esto nos supone grandes ventajas respecto a otros algoritmos de aprendizaje ya que nos permite clasificar en varias categorías como hemos visto antes y agrupar datos que puede que no estén necesariamente juntos [17]. Este hecho junto a la fácil implementación de este algoritmo en *R* (ya hay una función definida como hemos visto) nos hacen decidimos por el uso de este algoritmo.

El aspecto negativo del uso de esta técnica de clasificación es paradójicamente la función *Kernel*, ya que si usamos unos valores poco precisos para el aprendizaje podemos tener problemas de clasificación incorrecta, al ser una función con una gran capacidad de generalización [18]. Aun así, las ventajas superan con creces a las desventajas.

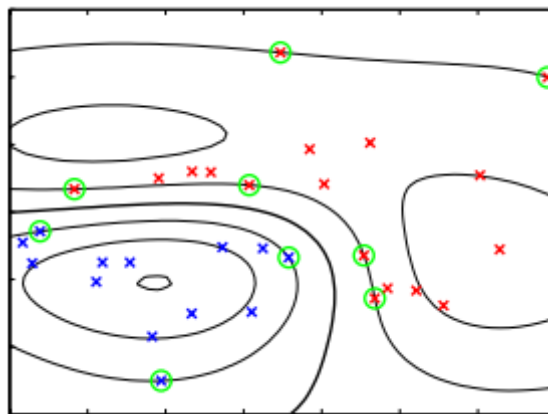


Figura 4.6. Ejemplo de clasificación de SVM en dos categorías, usando funciones Kernel. Fuente: Christopher Bishop (2006). *Pattern Recognition and Machine Learning*.

```

>
> #predictor
> ypred = predict(svp,xtest,type="response")
> table(ytest,ypred)
      ypred
ytest 0  1  2  3  4  5  6  7  8
  0 16  0  0  0  0  0  1  1  1
  1  0 17  0  0  0  0  0  0  0
  2  0  0  6  0  0  0  3  0  0
  3  0  0  0 12  0  1  5  0  0
  4  0  0  0  0 22  0  0  0  0
  5  0  0  0  0  0  9  0  0  0
  6  0  0  0  0  0  0 15  0  0
  7  0  0  1  0  0  1  0 13  0
  8  0  0  2  0  0  0  0  0 14
> sum(ypred==ytest)/length(ytest) #accuracy
[1] 0.8857143
>

```

Figura 4.7. Resultados del predictor y coeficiente de precisión

Como observamos, la tasa de precisión del sistema *off-line* ronda el 80-90%.

### 4.1.3. Problemática

A pesar de los prometedores resultados fuera de línea, cuando probamos el sistema en tiempo real, surgen varios problemas:

- En ocasiones, los .csv quedaban prácticamente vacíos sin una razón aparente, confundiendo al sistema, que creía que la comunicación lector-etiqueta se había interrumpido, por tanto la mano estaba en esa posición. Esto se pudo mejorar variando el tiempo de lectura pero aleatoriamente el problema volvía a surgir, siendo imposible una predicción correcta.

- El haz de las antenas no es semicircular como debería [12], sino que a la hora de la práctica las antenas leen en diagonal.

Para ver esto último más claro, analizamos unas mediciones para comprobar el funcionamiento de la estantería inteligente y la interacción con una mano humana. Primero colocamos dos *tags* y cuatro antenas -habiendo una distancia de aproximadamente 30 centímetros entre antena y *tag*- tal que así:

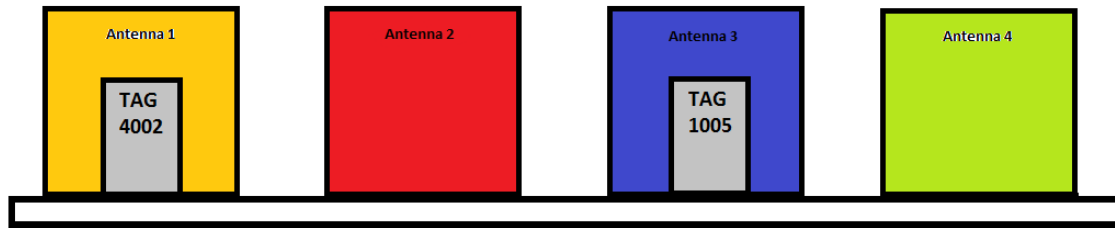


Figura 4.8. Disposición de antenas y etiquetas

Primero ajustamos la potencia de lectura tan baja como sea posible, para intentar evitar que las antenas lean etiquetas que no deben.

Potencia lectura [dB]	Resultado
31.5	Funciona, se leen todos los <i>tags</i> pero dos antenas leen el mismo <i>tag</i>
30	Funciona, se leen todos los <i>tags</i> pero dos antenas leen el mismo <i>tag</i>
29	Funciona, se leen todos los <i>tags</i> pero dos antenas leen el mismo <i>tag</i>
28	Funciona, se leen todos los <i>tags</i> pero dos antenas leen el mismo <i>tag</i>
27	Funciona, se leen todos los <i>tags</i> pero dos antenas leen el mismo <i>tag</i>
<b>26</b>	<b>Funciona, se leen todos pero otras antenas leen el <i>tag</i> (diagonal)</b>
25	No funciona bien
24	No lee nada

Figura 4.9. Tabla de ajuste de potencia de lectura

Parece que 26 dB es la potencia adecuada para este caso, pero vemos como las antenas no tienen un haz semicircular.

Hacemos unas mediciones con el lector, y creamos unas gráficas para ver qué antena ha leído qué etiqueta:

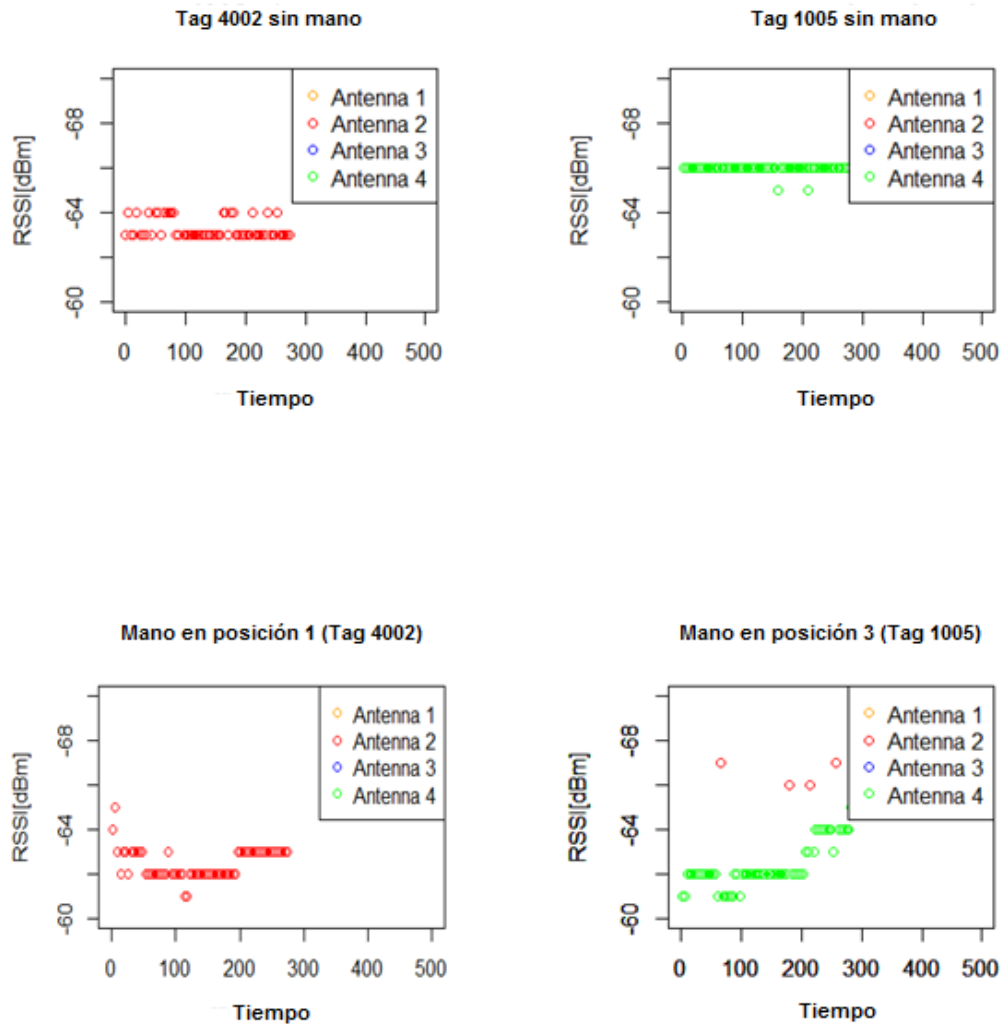


Figura 4.10. Gráficas de RSSI

Los datos son poco esclarecedores ya que en algunos casos el RSSI si varía, como en el ejemplo de la posición 3 –en el que incluso mejoran los datos de lectura con mano, algo

realmente ilógico-, pero en otros prácticamente no, como en la posición 1, siendo una de las causas del rechazo al uso de la estantería inteligente.

El problema deriva del haz de las antenas, ya que si nos fijamos en las gráficas podemos ver muy claramente que las etiquetas han sido leídas por una de las antenas situadas a los lados y no la de en frente, haciendo muy difícil que los valores varíen ya que la mano estaría situada en paralelo a la antena que está en frente de la etiqueta. Incluso en el caso de la última gráfica ambas antenas en diagonal han leído la etiqueta.

Aun así, tratamos de crear un escenario como en el de la figura 4.8 en el que dos antenas lean una etiqueta, y las otras dos lean la siguiente etiqueta. Si variamos la posición de los *tags* y bajamos el *read power* se puede llegar a obtener un resultado óptimo en el que en cada celda de la estantería sólo se lea la etiqueta que esté dentro, pero es una solución poco práctica a este problema ya que la idea es que el usuario no tenga que meter la mano hasta el fondo de la estantería.

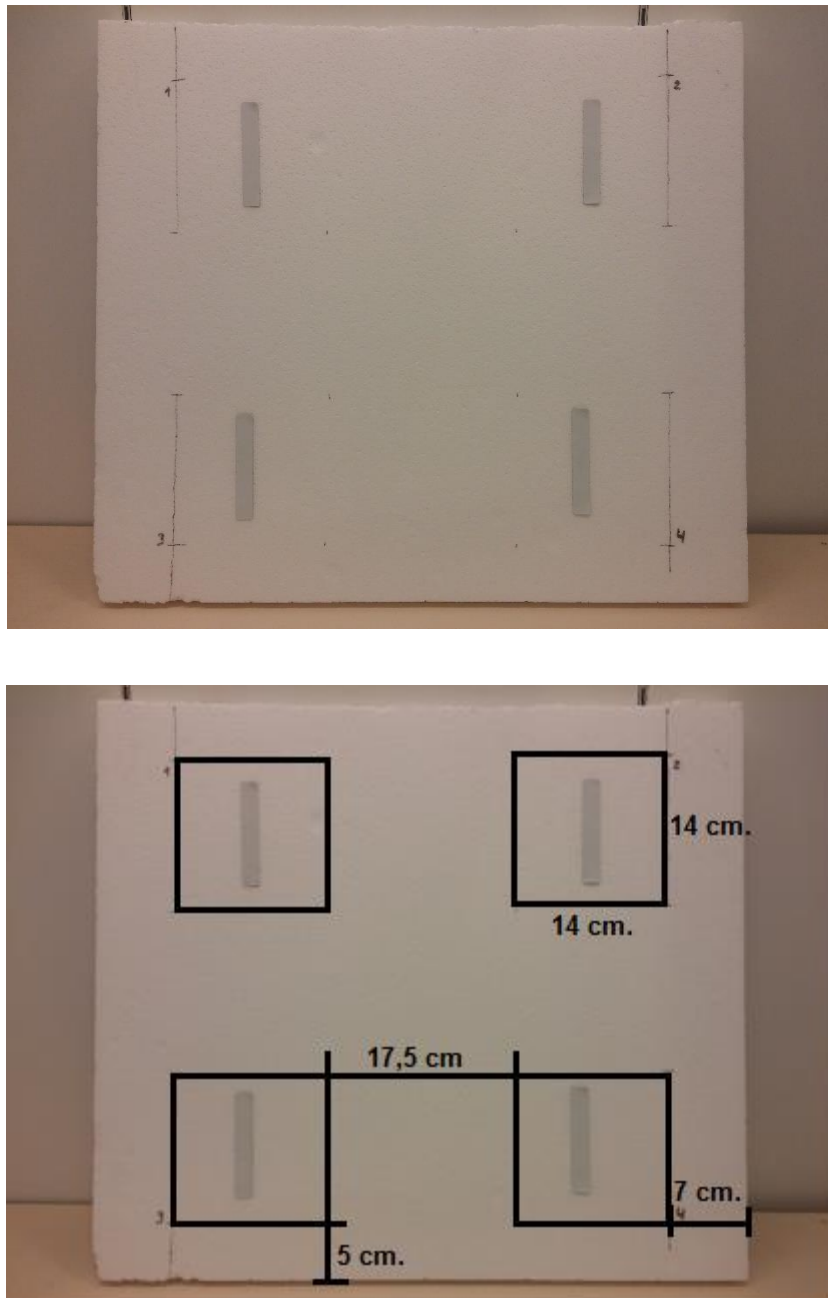
Además, la etiqueta debe estar alejada de la otra celda, haciendo que si tenemos varias celdas consecutivas el *tag* sólo pueda ser colocado en un punto muy específico central para que no sea detectado por ambas celdas adyacentes.

## **4.2. Superficie inteligente**

### **4.2.1. Idea del sistema**

A pesar de haber visto que el proyecto de la estantería no era viable debido a la distancia entre *tag* y antena y los problemas de lectura entre estos, sí que nos damos cuenta de que si acercamos lo suficiente ambos dispositivos y ajustamos la potencia y tiempo de lectura podemos crear una especie de área en las que solo una antena (o dos, dependiendo de lo que queramos) leyese su respectiva etiqueta.

Para ello, utilizamos una placa de *porexpan* a la que adherimos cuatro antenas en posiciones simétricas, dejando la distancia de una antena entre cada una de ellas. En la cara opuesta de la placa, pegamos cuatro etiquetas correspondiendo con la posición de cada antena.



*Figura 4.11. Cara delantera de la placa con las etiquetas pegadas y distribución*

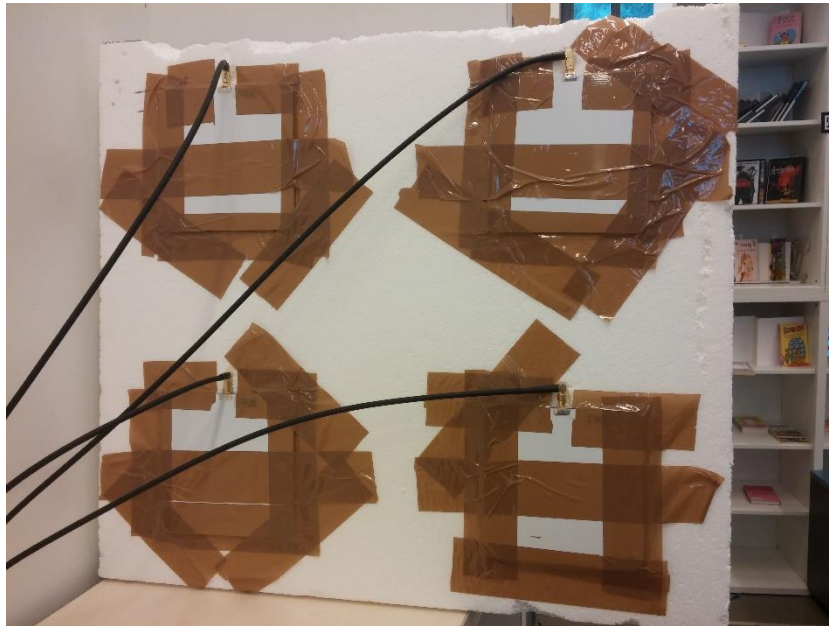


Figura 4.12. Cara trasera de la placa con las antenas

Ajustamos la potencia de lectura ya que al haber apenas 4 centímetros de distancia entre ambos dispositivos el valor debe ser bajo si queremos evitar que cada antena no lea únicamente su *tag* correspondiente. Una potencia de lectura de 5 dB y 160 ms de tiempo de lectura son suficientes para detectar todas las etiquetas, así que utilizaremos esta configuración. Añadir tiempo de lectura no nos aportaría realmente nada ya que no variarían los resultados, y el efecto negativo de subir la potencia ya lo hemos discutido antes.

Los 5 dB de potencia nos permiten separar la mano de la superficie entre 3 y 4 centímetros máximo para que observemos una variación en la *RSSI*. A partir de estos 3-4 cm. no se produce ninguna anomalía en los valores y por tanto la mano no es detectada. Aumentar el valor de potencia de lectura nos permitiría tener más rango de detección de mano pero también podría suponer que las antenas leyeran etiquetas que no les corresponden.

Para este caso y tras analizar los resultados y problemas de la estantería inteligente, nos olvidamos del *machine learning* e intentamos que nuestro sistema pueda funcionar de forma no supervisada en cualquier situación sin necesidad de perder tiempo tratando de buscar unos valores ideales para el nuevo aprendizaje, a parte de los cambios que se

puedan producir dependiendo de las condiciones (de humedad, temperatura...) en las que se encuentre. El método elegido es el “*k*-means”.

Los pasos que sigue éste método de agrupamiento son:

- 1) Seleccionar los centroides
- 2) Asignar cada valor al centroide más cercano
- 3) Recalcular los centroides a la media de todos los datos en un *cluster*
- 4) Asignar los datos al centroide más cercano
- 5) Repetir pasos 3 y 4 hasta que los datos no sean reasignados o se llegue al límite de iteraciones [19] [20] [21].

En nuestro sistema creamos cuatro *k*-means que actúan en paralelo, uno por cada conexión tag - antena, y así detectar en el momento en que los valores de *RSSI* hayan sufrido una variación como para –mediante condiciones- que podamos determinar que ahí está la mano del usuario.

#### **4.2.2. Implementación**

Una vez tenemos los cuatro *k*-means actuando definidos, asignamos un valor a cada posible estado (uno por antena-tag). Si el sistema entra en un estado concreto, asignamos su valor a un vector que guarda el historial de movimiento, y comparando con el valor anterior podremos predecir el movimiento dado.

Hay algo a tener muy en cuenta y que es de vital importancia para asegurar un correcto funcionamiento. No se trata de lo que sucede cuando ponemos la mano o hacemos un gesto, sino más bien todo lo contrario, como se comporta el sistema cuando alejamos la mano o completamos un gesto.

Como hemos descrito anteriormente, nuestro algoritmo *k*-means crea dos *clusters*, uno para “con mano” y otro “sin mano”. Para evitar confusiones cuando hay una mínima variación de potencia sin mano (ej. -35 dB y -36 dB, valores muy normales cuando el sistema está en reposo), ponemos como condición que haya al menos 3 valores

diferentes para que el *k*-means empiece a funcionar. En condiciones normales, sin mano, los valores de *RSSI* se mantienen estables, como mucho cambian 1 dB.

Procedemos a guardar los *clusters* en vectores separados y nos aseguramos de que al menos uno de los dos haya variado lo suficiente como para determinar que tenemos la mano ahí poniendo como condición una diferencia de al menos 3 dBs. Al quitar la mano, sin embargo, el *cluster* con los valores “normales” seguirá actualizándose mientras que el otro mantendrá sus valores, haciendo que la condición de diferencia siempre se cumpla. Para evitar esto, creamos un umbral en el que seleccionamos el valor máximo de los dos *clusters* (ya que al poner la mano éstos empeoran y bajan bastante).

Después, calculamos la diferencia entre este máximo y el último valor de potencia *RSSI* que hayamos cargado de los archivos .csv, que, al igual que la diferencia entre *clusters*, tiene que ser mayor de 3 dBs para que la condición sea verdadera. Cabe puntualizar que al tener 4 algoritmos *k*-means al mismo tiempo, también se crean y actualizan todos estos valores para cada etiqueta independientemente.

Si se cumplen ambas condiciones mencionadas antes, añadimos un valor numérico (0-4) que nos dice que etiqueta ha sufrido el cambio de potencia a un vector que guardará todo el movimiento que realizamos. Este vector nos permite saber en qué posición estábamos antes y donde estamos ahora y por tanto definir el gesto que acabamos de realizar. Una vez definido el gesto que acabamos de hacer, mandamos una *URL* con una orden al reproductor multimedia VLC.

Definimos pues, las posiciones de cada etiqueta, siendo 1 la esquina superior izquierda, 2 la derecha, y abajo 3 a la izquierda y 4 a la derecha. Pondremos 0 al vector de movimiento mientras no se produzca ninguno.

Una vez tenemos el sistema configurado, procedemos a abrir el reproductor VLC en un puerto específico para poderle mandar solicitudes, ejecutamos AdvanNet y posteriormente lo mismo con el código en *RStudio*.

A continuación presentamos las seis acciones que hemos asociado a diferentes gestos, junto a las gráficas en las que se puede ver como la potencia *RSSI* varía a lo largo del tiempo.

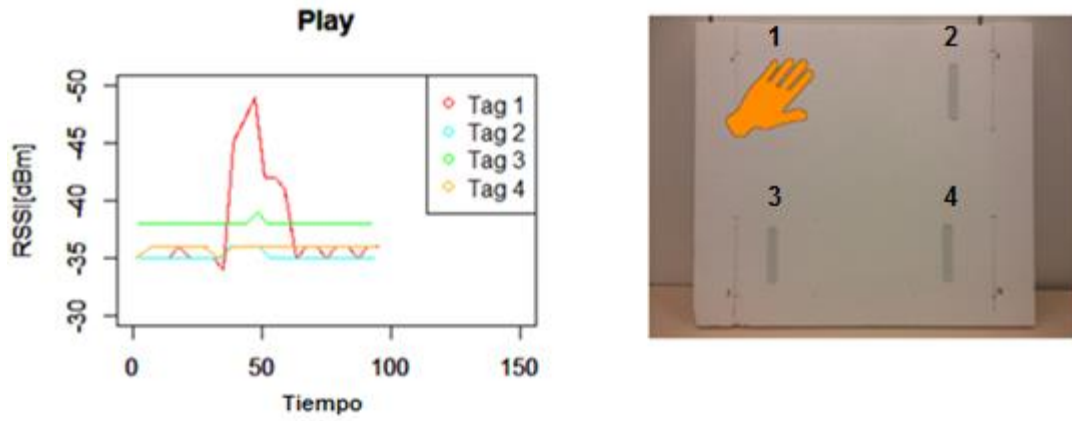


Figura 4.13. Gráfica y movimiento correspondiente a Play

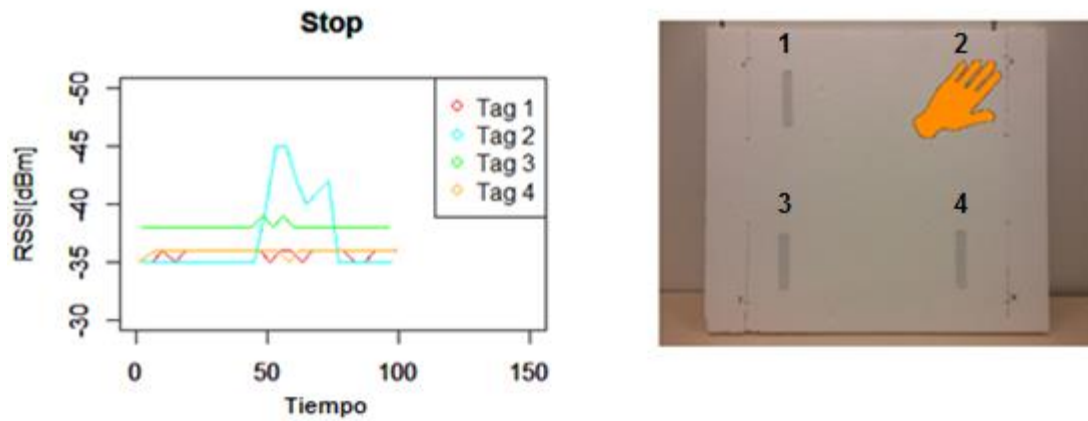


Figura 4.14. Gráfica y movimiento correspondiente a Stop

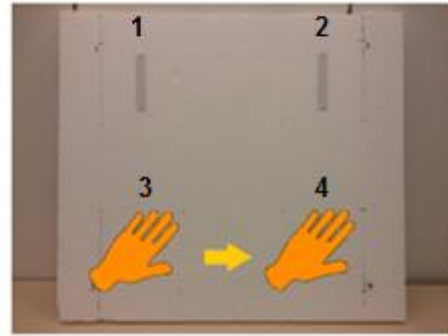
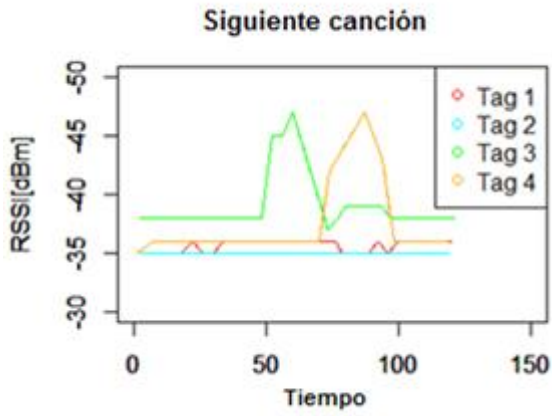


Figura 4.15. Gráfica y movimiento correspondiente a Siguiete canción

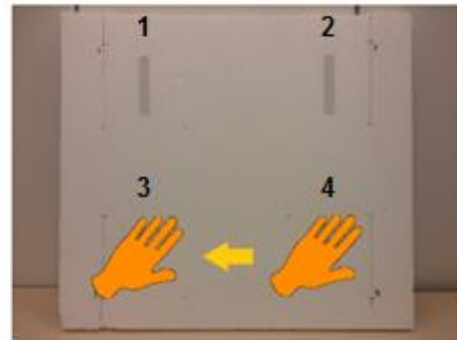
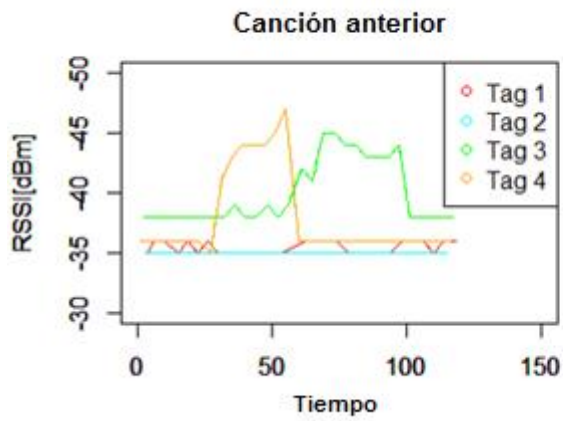


Figura 4.16. Gráfica y movimiento correspondiente a Canción anterior

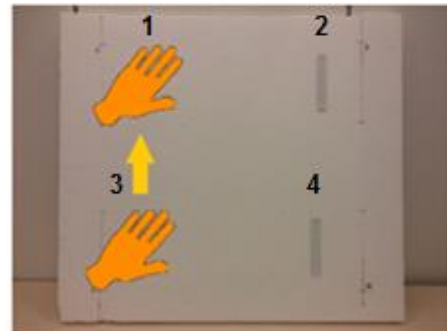
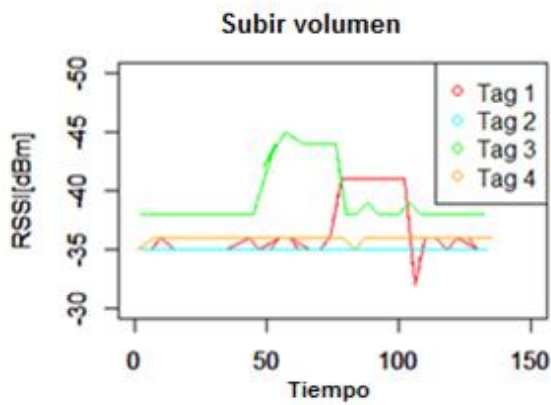


Figura 4.17. Gráfica y movimiento correspondiente a Subir volumen

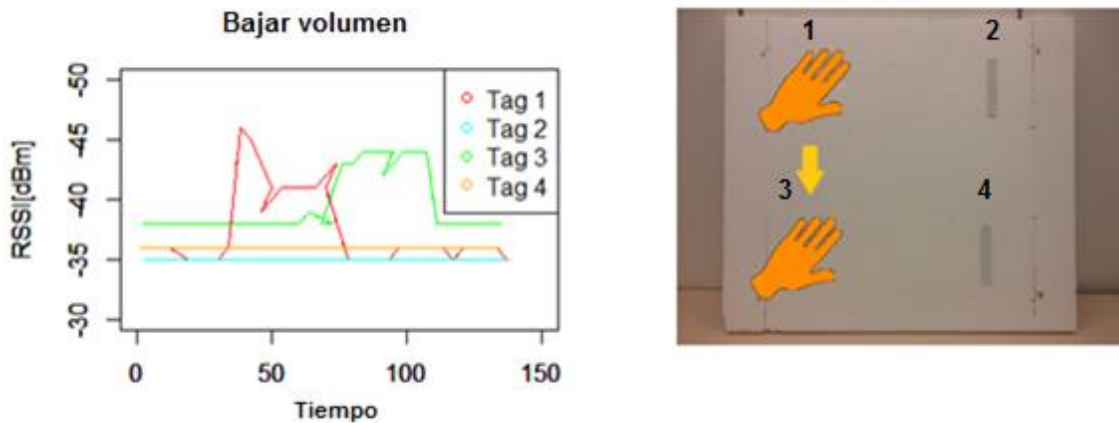


Figura 4.18. Gráfica y movimiento correspondiente a Bajar volumen

VLC tiene una característica que nos supone un problema, ya que utilizan el mismo botón para hacer Play y Pause, y por tanto, si mandamos dos órdenes seguidas de Pausa, la segunda sería reconocida como un Play. Para nuestro sistema nos hemos decantado por usar Stop debido a este hecho, pero con más antenas y más posibles gestos se podría solucionar.

### 4.2.3. Evaluación

Hemos visto como la potencia *RSSI* varía correctamente cuando acercamos la mano y hemos explicado cómo hemos aplicado el algoritmo de *k*-means, así que sólo nos queda comprobar si realmente funciona y si los gestos que realizamos son capturados por el sistema.

La evaluación se ha hecho a partir de dos vídeos adjuntos a esta memoria. A continuación presentamos los resultados obtenidos. Cabe puntualizar que estos son los resultados de un día concreto a una hora determinada, los resultados en otro momento, situación ambiental, etc. pueden variar a mejor o a peor.

Las celdas de acción en blanco representan un movimiento que no produce ninguna acción en sí, pero que combinadas con el siguiente movimiento son detectadas como un gesto y provocan el envío de una orden al reproductor.

<b>Movimiento</b>	<b>Tiempo de respuesta</b>	<b>¿Detectado?</b>	<b>Acción</b>
Mano a 1	00:04 a 00:11 – 7 segs	Sí	Play
Mano a 2	00:14 a 00:18 – 4 segs	Sí	Stop
Mano a 1	00:20 a 00:26 – 6 segs	Sí	Play
Mano a 4	00:32 a 00:37 – 5 segs	Sí	
Mano a 3	00:38 a 00:47 – 9 segs	Sí	Canción anterior
Mano a 2	00:52 a 01:01 – 9 segs	Sí	Stop
Mano a 1	01:03 a 01:07 – 4 segs	Sí	Play
Mano a 1	01:14 a 01:20 – 6 segs	Sí	
Mano a 3	01:22 a 01:28 – 6 segs	Sí	Bajar volumen
Mano a 1	01:31 a 01:36 – 5 segs	Sí	Subir volumen
Mano a 2	01:42 a 01:49 – 7 segs	Sí	Stop
Mano a 1	01:51 a 01:57 – 6 segs	Sí	Play
Mano a 4	02:01 a 02:08 – 7 segs	Sí	
Mano a 3	02:09 a 02:13 – 4 segs	Sí	Canción anterior
Mano a 1	02:17 a 02:23 – 6 segs	Sí	
Mano a 2	02:28 a 02:40 – 12 segs	Sí	Stop
Mano a 1	02:43 a 02:52 – 9 segs	Sí	Play
Mano a 3	02:55 a 03:00 – 5 segs	Sí	
Mano a 4	03:04 a 03:08 – 4 segs	Sí	Siguiente canción
Mano a 3	03:14 a 03:29 – 15 segs	Sí	Canción anterior
Mano a 4	03:31 a 03:35 – 4 segs	Sí	Siguiente canción
Mano a 1	03:37 a 03:47 – 10 segs	Sí	
Mano a 3	03:48 a 03:56 – 8 segs	Sí	Bajar volumen
Mano a 2	03:58 a 04:12 – 14 segs	Sí	Stop
Mano a 1	04:18 a 04:23 – 5 segs	Sí	Play
Mano a 3	04:28 a 04:35 – 7 segs	Sí	Bajar volumen

Mano a 1	04:38 a 04:46 – 8 segs	Sí	Subir volumen
Mano a 4	04:51 a 04:56 – 5 segs	Sí	
Mano a 3	04:58 a 05:06 – 8 segs	Sí	Canción anterior
Mano a 2	05:11 a 05:16 – 5 segs	Sí	Stop
Mano a 3	05:20 a 05:27 – 7 segs	Sí	
Mano a 1	05:28 a 05:27 – 9 segs	Sí	Subir vol./Play
Mano a 3	05:47 a 05:54 – 7 segs	Sí	
Mano a 4	05:57 a 06:02 – 5 segs	Sí	Siguiente canción
Mano a 2	06:07 a 06:14 – 7 segs	Sí	Stop
Mano a 1	06:17 a 06:23 – 6 segs	Sí	Play

Figura 4.19. Tabla con recuento de tiempo y compendio de gestos realizados del primer video

Movimiento	Tiempo de respuesta	¿Detectado?	Acción
Mano a 1	00:02 a 00:10 – 8 segs	Sí	Play
Mano a 2	00:12 a 00:20 – 8 segs	Sí	Stop
Mano a 1	00:22 a 00:29 – 7 segs	Sí	Play
Mano a 2	00:33 a 00:37 – 4 segs	Sí	Stop
Mano a 1	00:40 a 00:47 – 7 segs	Sí	Play
Mano a 4	01:00 a 01:08 – 7 segs	Sí	
Mano a 3	01:11 a 01:19 – 8 segs	Sí	Canción anterior
CORTE VÍDEO	--	--	--
Mano a 1	03:06 a 03:11 – 5 segs	Sí	Play
Mano a 2	03:12 a 03:17 – 5 segs	Sí	Stop
Mano a 3	03:18 a 03:25 – 7 segs	Sí	
Mano a 4	03:27 a 03:34 – 7 segs	Sí	
Mano a 1	03:37 a 03:42 – 5 segs	Sí	Play
Mano a 4	03:44 a 03:51 – 7 segs	Sí	
Mano a 3	03:52 a 03:59 – 7 segs	Sí	Canción anterior
Mano a 1	04:05 a 04:11 – 6 segs	Sí	

Mano a 3	04:15 a 04:22 – 7 segs	Sí	Bajar volumen
Mano a 1	04:25 a 04:30 – 5 segs	Sí	Subir volumen
Mano a 2	04:33 a 04:38 – 5segs	Sí	Stop
Mano a 1	04:41 a 04:47 – 6 segs	Sí	Play
Mano a 4	04:52 a 04:57 – 5 segs	Sí	
Mano a 3	05:00 a 05:14 – 14 segs	Sí	Canción anterior
Mano a 1	05:23 a 05:32 – 9 segs	Sí	Subir volumen
Mano a 2	05:36 a 05:39 – 3 segs	Sí	Stop
Mano a 1	05:43 a 05:49 – 6 segs	Sí	Play
Mano a 2	05:52 a 05:58 – 6 segs	Sí	Stop

*Figura 4.20. Tabla con recuento de tiempo y compendio de gestos realizados del segundo video*

Como podemos observar, la tasa de detección es del 100%. El sistema no ha producido falsos positivos ni errores de esa índole. Si bien es cierto que la precisión, por decirlo de alguna forma, es perfecta, el tiempo de detección es bastante mejorable. En los 6 minutos de prueba del primer vídeo, la media de tiempo de detección ha sido de 6.9 segundos entre que ponemos la mano y el sistema tiene alguna respuesta, ya sea de reconocimiento de mano o de reacción a un gesto (Play, stop). En el segundo este valor se reduce ligeramente, siendo de 6.3 segundos.

Si bien este valor se reduce a las dos acciones mencionadas antes, para las otras cuatro debemos sumar el tiempo de detección de la mano en la primera etiqueta y en la segunda, teniendo como resultado obtendremos un gesto completo y su acción correspondiente (Siguiendo canción, subir volumen...).

#### **4.2.4. Múltiples etiquetas en una antena**

Para finalizar el trabajo con la superficie inteligente quisimos probar alguna variante del sistema propuesto. Si bien disponer de una antena por cada *tag* es muy efectivo como hemos visto anteriormente, es poco práctico por la gran cantidad de espacio necesaria,

así que probamos si realmente se podría distinguir una mano poniendo varios *tags* en una antena.

Para ello tan solo conectamos una antena en la superficie inteligente y colocamos dos etiquetas en el área de la antena de forma que haya una mano de separación entre ellas, para no interferir con la otra cuando pongamos una mano.

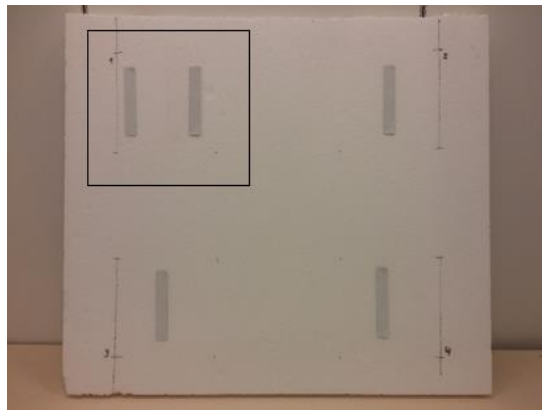


Figura 4.21. Disposición de las dos etiquetas en el sistema probado

Aquí presentamos los resultados de una de las pruebas, siguiendo el orden: mano en *tag* 1, mano en *tag* 2 y una mano en cada *tag*.

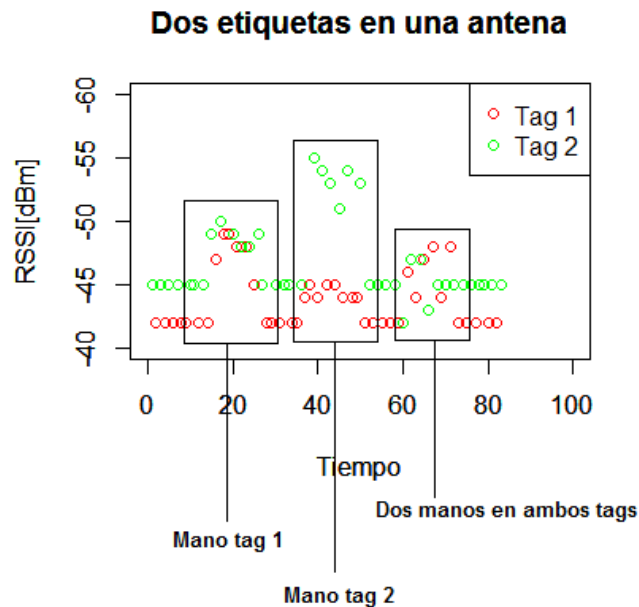


Figura 4.22. Gráfica RSSI dos etiquetas y un tag

Como podemos observar, los valores sí varían pero no como esperamos. En el caso de la mano en la etiqueta 2 los resultados son realmente buenos, ya que solo sus valores de *RSSI* empeoran y no del *tag 1*, pero en el caso anterior, el de la etiqueta 1, ambos valores cambian y el sistema produciría un resultado incorrecto. En el caso de ambas manos los valores sí varían.

Esto significa que para llegar a un resultado fiable se debería estudiar más a fondo la disposición exacta de las etiquetas y la mano para no influir en la etiqueta que no corresponde. Incluso podemos decir que el uso de las antenas de 14x14cm no es la mejor opción, ya que en para este experimento las etiquetas estaban colocadas prácticamente al borde de las antenas (a 2cm del borde, para ser exactos), y seguramente habría que buscar una antena de tamaño rectangular y separarlas más.

En cualquier caso, no parece la mejor opción para nuestro sistema.



## 5. CONCLUSIÓN

Este proyecto se ha centrado en aplicar la tecnología *RFID* para controlar un programa informático mediante el uso de gestos sin necesidad de que el usuario porte una etiqueta encima.

En la primera parte hemos tratado de alcanzar este objetivo utilizando la *Smart Shelf*, pero a base de experimentar nos hemos encontrado con ciertos problemas debido a la forma del haz de las antenas que nos han hecho imposible continuar con este sistema propuesto. Además, el uso de una matriz con tanta información guardada no era viable, ya que en el caso de un error de lectura el sistema se confundiría con gran facilidad.

En la segunda parte, creamos una superficie inteligente acercando las etiquetas y las antenas, haciendo que cada antena sólo lea el *tag* más cercano, evitando así posibles confusiones del sistema. Mediante el uso de un algoritmo de aprendizaje no supervisado, *k-means*, y una serie de condiciones en relación a la potencia, hacemos que nuestro sistema sea capaz de detectar movimiento alrededor de las etiquetas y enviar órdenes a un reproductor de música.

Como se suele decir, nada se sabe bien sino por medio de la experiencia. Si bien el sistema de la estantería inteligente no acabó como esperábamos, nos dio las pautas a seguir para llevar a cabo un método como el de la superficie inteligente, que hemos visto que es funcional, y si bien se puede mejorar el tiempo de respuesta, no tenemos falsas alarmas ni problemas de lectura.



## 6. TRABAJO FUTURO

Como hemos comentado con anterioridad, en un trabajo futuro sería conveniente tratar de mejorar el tiempo de reacción del sistema. Hay que buscar alternativas a la carga de archivos que se hace en este proyecto, ya que debemos dormir el sistema 2 segundos si no queremos que se multipliquen los datos. Una buena solución sería acceder a los datos de AdvanReader mediante REST, ya que de esta forma no necesitamos los archivos .csv sino que directamente podemos trabajar con los datos que nos interesen de forma prácticamente instantánea.

A nivel de interacción, me gustaría añadir más antenas y más etiquetas para tener más posibles combinaciones y por tanto más gestos, ya que estamos limitados a pocas combinaciones con tan solo 4 antenas y etiquetas. También me gustaría seguir indagando en las posibilidades de la interacción entre varios *tags* y una sola antena, ya que se arrojaron algunos resultados prometedores pero poco claros.

Por último, siendo más ambicioso, se podría conectar el código con otros lenguajes de programación como *Arduino*, que nos facilitaría el uso de sensores de movimiento o dispositivos tipo *led*, y explorar nuevas posibilidades de reconocimiento de gestos, así como comprobar cómo se comporta el sistema en nuevos escenarios y nuevas formas de interactuar con él.



## 7. REFERENCIAS

### 7.1. Referencias Bibliográficas

- [1] John Buckley (2006). *FROM RFID TO THE INTERNET OF THINGS: Pervasive networked systems*.
- [2] Tapia D., Cueli J., Corchado J. (2007). *Identificación por Radiofrecuencia: Fundamentos y Aplicaciones*.
- [3] Raúl Parada, Joan Melià-Seguí, Anna Carreras, Marc Morenza-Cinos and Rafael Pous (2014). *Measuring User-Object Interactions in IoT Spaces*.
- [4] Raúl Parada, Joan Melià-Seguí, Anna Carreras, Marc Morenza-Cinos and Rafael Pous (2015). *Using RFID to detect interactions in ambient assisted living environments*.
- [5] Joan Melià-Seguí and Rafael Pous (2014). *"Human-object Interaction Reasoning using RFID-enabled Smart Shelf"*
- [6] Sheng Ye, Hong Zeng, Jin Fan and Xizhe Wang (2014). *LSI-REC: a Link State Indicator based gesture recognition scheme in a RFID system*.
- [7] Christopher Chen, Justin Cruz, Sai Kotikalapudi and Kyung-Tack (Terry) Oh (2014). *Passive RFID Gesture Recognition: Final Report*.
- [8] Parvin Asadzadeh, Lars Kulik, Egemen Tanin (2011). *Gesture recognition using RFID technology*.
- [9] Mahyar Taghizadeh Nouei, Ali Vahidian Kamyad, Ahmad Reza Soroush, Somayeh Ghazalbash (2014). *A comprehensive operating room information system using the Kinect sensors and RFID*.
- [10] Último acceso: 14/06/2015

<http://keonn.com/rfid-components/readers/advanreader-100.html>

[11] Último acceso: 14/06/2015

<http://www.rfidpoint.com/fundamentos/antenas-rfid/>

[12] Último acceso: 14/06/2015

<http://keonn.com/rfid-components/antennas/advantenna-p11.html>

[13] Yan L., Zhang Y., Yang L., Ning H (2008). *From RFID to the Next-Generation Pervasive Networked Systems. The internet of things.*

[14] Último acceso: 14/06/2015

<http://www.actum.es/preguntas-frecuentes/tipos-de-tags>

[15] Último acceso: 14/06/2015

<http://www.genbetadev.com/formacion/r-un-lenguaje-y-entorno-de-programacion-para-analisis-estadistico>

[16] Último acceso: 14/06/2015

<http://www.videolan.org/vlc/>

[17] Alexandros Karatzoglou, David Meyer, Kurt Hornik (2006). *Support Vector Machines in R.*

[18] Último acceso: 14/06/2015

[http://www.webdelprofesor.ula.ve/economia/gcolmen/programa/economia/maquinas\\_vectores\\_soporte.pdf](http://www.webdelprofesor.ula.ve/economia/gcolmen/programa/economia/maquinas_vectores_soporte.pdf)

[19] Último acceso: 14/06/2015

<https://stat.ethz.ch/R-manual/R-devel/library/stats/html/kmeans.html>

[20] Último acceso: 14/06/2015

<http://www.r-statistics.com/2013/08/k-means-clustering-from-r-in-action/>

[21] Anil K. Jain (2009). *Data clustering: 50 years beyond K-means*.

## 7.2. Referencias lista de figuras

Figura 3.1. Último acceso: 14/06/2015

<http://keonn.com/rfid-components/readers/advanreader-100.html>

Figura 3.2. Último acceso: 14/06/2015

<http://keonn.com/rfid-components/antennas/advantenna-p11.html>

Figura 3.3 Último acceso: 14/06/2015

<http://cdn.barcodesinc.com/cats/rfid-readers/tag.jpg>

Figura 3.4 Último acceso: 14/06/2015

[www.amazon.com](http://www.amazon.com)

Figura 4.1. Fuente propia

Figura 4.2. Fuente propia

Figura 4.3. Fuente propia

Figura 4.4. Fuente propia

Figura 4.5. Fuente propia

Figura 4.6. Christopher Bishop (2006). *Pattern Recognition and Machine Learning*.

Figura 4.7. Fuente propia

Figura 4.8. Fuente propia

Figura 4.10. Fuente propia

Figura 4.11. Fuente propia

Figura 4.12. Fuente propia

Figura 4.13. Fuente propia

Figura 4.14. Fuente propia

Figura 4.15. Fuente propia

Figura 4.16. Fuente propia

Figura 4.17. Fuente propia

Figura 4.18. Fuente propia

Figura 4.21. Fuente propia

Figura 4.22. Fuente propia

### **7.3. Referencias lista de tablas**

Figura 4.9. Fuente propia

Figura 4.19. Fuente propia

Figura 4.20. Fuente propia

Figura 9.1. Fuente propia, creado con GanttProject.

## 8. ANEXOS

### Anexo 1.

Aquí presentamos el código comentado utilizado para el reconocimiento de gestos de la superficie inteligente.

```
library(NbClust)

require(RCurl)

setwd("C:/Users/Sergio/Desktop/AdvanNet-2.1.4/bin/csvs") # dirección archivos .csv

t<-Sys.time() # guardamos el tiempo inicial

ti<-Sys.time() # cumplimos la condición del while

rssi1<-vector() # creamos un vector vacío para guardar la potencia RSSI

rssi2<-vector()

rssi3<-vector()

rssi4<-vector()

mov <- vector() # creamos un vector vacío para guardar el movimiento

j <- length(mov)

while (ti<t+3600){ # hacemos que el sistema funcione durante una hora

  ti<-Sys.time()

  temp = list.files(pattern="*.csv")

  fileName = assign(temp[length(temp)], read.csv(temp[length(temp)]))
```

```
# cargamos el último archivo .csv de la carpeta
```

```
rss1<-append(rssi1,  
fileName$RSSI[which(fileName$HEX_EPC=="abcf000000000001")])
```

```
#guardamos el valor de potencia RSSI de cada antena-tag
```

```
rss2<-append(rssi2,  
fileName$RSSI[which(fileName$HEX_EPC=="abcf000000000002")])
```

```
rss3<-append(rssi3,  
fileName$RSSI[which(fileName$HEX_EPC=="abcf000000000003")])
```

```
rss4<-append(rssi4,  
fileName$RSSI[which(fileName$HEX_EPC=="abcf000000000004")])
```

```
Sys.sleep(2) # como el sistema es más rápido que AdvanNet, hacemos que espere 2  
segundos para cada interacción
```

```
if((length(unique(rssi1))>2) | (length(unique(rssi2))>2) | (length(unique(rssi3))>2) |  
(length(unique(rssi4))>2)){
```

```
# no se cumple la condición hasta que haya al menos dos valores diferentes de RSSI de  
alguno de nuestros tags
```

```
if(length(unique(rssi1))>2){ #si se trata de esta etiqueta
```

```
kc1 <- kmeans(rssi1, centers= 2, iter.max=1, nstart=2) # aplicamos k-means con 2  
centroides
```

```
cluster11<-rssi1[which(kc1$cluster==1)] # guardamos cada cluster en una variable
```

```
cluster12<-rssi1[which(kc1$cluster==2)]
```

```
distance1<-abs(min(unlist(cluster11))-min(unlist(cluster12))) # calculamos la diferencia
entre clusters
```

```
threshold1 <-max(c(cluster12,cluster11)) # Nuestro threshold será el valor más alto de
los clusters
```

```
distance11 <- abs(threshold1 - tail(rssi1, n=1)) # Calculamos la diferencia entre el
threshold y el último valor de RSSI
```

```
if (distance1>3 & distance11>3){
```

```
  print("mano1")
```

```
  mov <- append(mov, "1") # añadimos un 1 al vector de movimiento
```

```
  getURL("http://127.0.0.1:9090/requests/status.xml?command=pl_play") # mandamos
la orden al reproductor VLC
```

```
  if(mov[j-1]==3){
```

```
    getURL("http://127.0.0.1:9090/requests/status.xml?command=volume&val=200")
```

```
  }
```

```
}
```

```
}
```

```
if(length(unique(rssi2))>2){
```

```
  kc2 <- kmeans(rssi2, centers= 2, iter.max=1, nstart=2)
```

```
  cluster21<-rssi2[which(kc2$cluster==1)]
```

```
  cluster22<-rssi2[which(kc2$cluster==2)]
```

```
  distance2<-abs(min(unlist(cluster21))-min(unlist(cluster22)))
```

```
  threshold2 <- max(c(cluster21,cluster22))
```

```
  distance21 <- abs(threshold2 - tail(rssi2, n=1))
```

```
  if (distance2>3 & distance21>3){
```

```
    print("mano2")
```

```
    mov <- append(mov, "2")
```

```

if(mov[j-1]==2){
  getURL("http://127.0.0.1:9090/requests/status.xml?command=pl_stop")
}
}
}

if(length(unique(rssi3))>2){
  kc3 <- kmeans(rssi3, centers= 2, iter.max=1, nstart=2)
  cluster31<-rssi3[which(kc3$cluster==1)]
  cluster32<-rssi3[which(kc3$cluster==2)]
  distance3<-abs(min(unlist(cluster31))-min(unlist(cluster32)))
  threshold3 <- max(c(cluster32,cluster31))
  distance31 <- abs(threshold3 - tail(rssi3, n=1))
  if (distance1>3 & distance31>3){
    print("mano3")
    mov <- append(mov, "3")
    if(mov[j-1]==1){
      getURL("http://127.0.0.1:9090/requests/status.xml?command=volume&val=100")
      mov<-append(mov, "0")
    }
    if(mov[j-1]==4){
      getURL("http://127.0.0.1:9090/requests/status.xml?command=pl_previous")
      mov<-append(mov, "0")
    }
  }
}

```



# ANEXO 2.

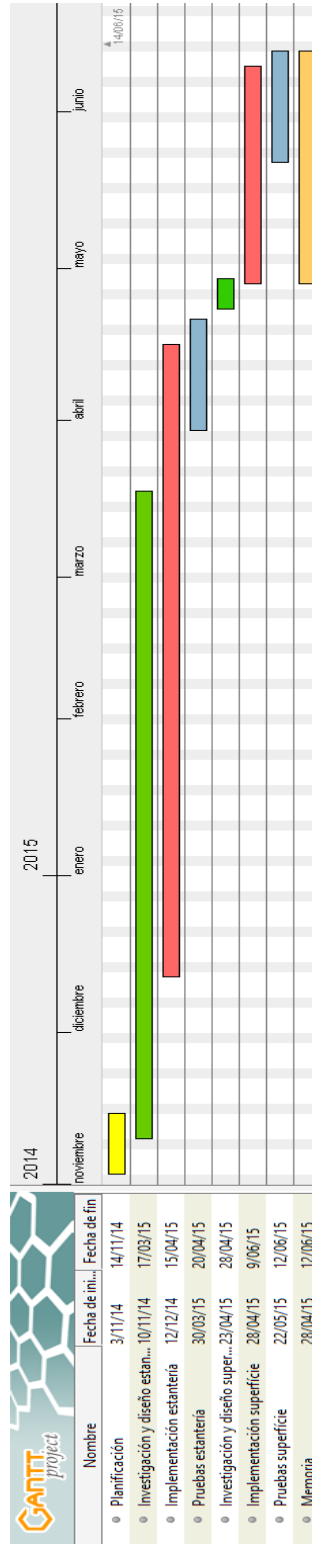


Figura 9.1. Diagrama de Gantt del proyecto