UNIVERSIDAD DE VALLADOLID

E.T.S.I TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA DE TECNOLOGÍAS ESPECÍFICAS DE TELECOMUNICACIÓN, ESPECIALIDAD TELEMÁTICA

DESARROLLO DE UNA APLICACIÓN PARA LA ADQUISICIÓN Y PROCESAMIENTO DE PARÁMETROS PSICOLÓGICOS EXTRAÍDOS DE SENSORES INALÁMBRICOS MEDIANTE UN DISPOSITIVO ANDROID

Autor:

Víctor del Barrio Sánchez

Tutor:

David González Ortega

VALLADOLID, 6 DE SEPTIEMBRE DE 2016

TÍTULO:	**Desarrollo de una aplicación para la adquisición y procesamiento de parámetros psicológicos extraídos de sensores inalámbricos mediante un dispositivo Android.**
AUTOR:	**Víctor del Barrio Sánchez.**
TUTOR:	**David González Ortega**
DEPARTAMENTO:	**Departamento de Teoría de la Señal y Comunicaciones e Ingeniería Telemática**

TRIBUNAL

PRESIDENTE:	**José Fernando Díez Higuera**
VOCAL:	**María Ángeles Pérez Juárez**
SECRETARIO:	**David González Ortega**
SUPLENTE:	**Francisco Javier Díaz Pernas**
SUPLENTE:	**Mario Martínez Zarzuela**

FECHA:	**6 de Septiembre de 2016**
CALIFICACIÓN:	

Resumen

El objetivo principal de este trabajo fin de grado consiste en desarrollar una aplicación para Android para recopilar y analizar los datos procedentes de sensores inalámbricos con la finalidad de detectar si un individuo se encuentra fatigado bajo una situación de conducción real o en un simulador. Los sensores mediante los cuales se recogen los datos pertenecen al kit de desarrollo *Platinum* de *Shimmer*.

La aplicación implementa una serie de filtros para el tratamiento de las señales ECG, EMG y GSR así como los algoritmos necesarios extraídos de artículos de investigación del campo de la biomedicina y herramientas de apoyo para la representación gráfica de los resultados y análisis de los resultados de los ficheros almacenados. Del mismo modo se ha procedido al desarrollo paralelo de los algoritmos en lenguaje de programación *MatLab* para un procesado *offline* más preciso de los ficheros almacenados con la aplicación y otro para la señal procedente de dos acelerómetros a fin de obtener la tasa respiratoria.

Por otra parte se ha realizado la adquisición de datos de las señales ECG y EMG de diferentes sujetos conociendo previamente su estado físico y atención al volante distinguiendo entre fatigado, usando el móvil y atendiendo a la carretera con el objetivo de disponer de un banco de datos para el cálculo de momentos de alto orden espectral y poder determinar el estado del conductor mediante los clasificadores LDA y QDA.

Palabras clave: Android, ECG, EMG, GSR, LDA, procesamiento de señales biomédicas, QDA, sensores fisiológicos.

Abstract

The aim of this Final Project is to develop an Android application to collect and analyze data from wireless sensors in order to detect whether an individual is fatigued under real driving situation or using a simulator. The sensors used to collect all the data belong to Platinum development kit Shimmer.

The application implements a filter set for the treatment of ECG, EMG and GSR signals as well as the necessary algorithms extracted from research articles in the field of biomedicine and support tools for graphical representation and analysis of the results stored in files. Besides this, the development of algorithms in *Matlab* programming language is used to more accurate offline processing files stored by the application and other for two accelerometers to obtain the respiration rate.

On the other hand, data acquisition of ECG and EMG signals of different subject has been made knowing their physical condition and attention while driving distinguishing between fatigued, using a mobile phone and paying attention to the road with the aim of having a database for calculating moments of high spectral order and of determining the state of the driver by the LDA and QDA classifiers.

Keywords: Android, biomedical signal processing, ECG, EMG, GSR, LDA, QDA, physiological sensors.

Agradecimientos

A mi tutor David por haberme dado la oportunidad de desarrollar este proyecto, por sus consejos y apoyo en los malos momentos.

A Rubén Ruiz González, compañero de universidad y actual estudiante de doctorado, por haberme ayudado y orientado.

A mis padres por haberme brindado la oportunidad de estudiar la carrera que quería y apoyarme siempre de manera incondicional.

A mi novia Elena, por haberme apoyado desde el comienzo del grado hasta el final y su gran ayuda y ánimos en el desarrollo de este trabajo.

ÍNDICE:

LISTADO DE FIGURAS:

LISTADO DE TABLAS:

I. INTRODUCCIÓN: PRESENTACIÓN Y JUSTIFICACIÓN DEL TEMA ELEGIDO.

"Los avances tecnológicos no pueden suceder sin científicos o ingenieros. El desafío de la sociedad es equiparar a las suficientes personas con las habilidades correctas y formas de pensar, que lleguen a trabajar en los problemas más importantes. ".

(Eric Schmidt)[1]

El propósito de esta investigación de Trabajo Fin de Grado es el estudio, análisis y desarrollo de una aplicación para un dispositivo móvil, con uno de los sistemas operativos más importantes y expandidos en la actualidad, como es Android[2], que permita con ello una adquisición de los parámetros mesurables de la diferente gama de sensores inalámbricos de la marca comercial *Shimmer Sensing.* Además con este estudio de caso, se contribuye a entender la aplicación de las tecnologías emergentes en los últimos años como son los *Smartphones* y *tablets* en el ámbito cotidiano y de uso diario y normal. Desde el punto de vista de la medicina, se contribuye a entender de una manera sencilla, las diferentes señales de interés relativas al electrocardiograma (ECG), electromiograma (EMG) y la respuesta galvánica de la piel (GSR).

La idea de realizar este trabajo surgió durante el curso 2014/15 en el cual el profesor David González Ortega de la Universidad de Valladolid ofertó y le solicité información sobre este trabajo de investigación y desarrollo sobre el procesamiento de señales extraídas de sensores inalámbricos. De todos los trabajos propuestos, era el que más capto mi atención debido a que la medicina ha sido un campo de interés durante toda mi etapa estudiantil y con este trabajo surgió la oportunidad de poder realizar diferentes investigaciones en los dos campos que más me interesan.

[1]Jorge, Miguel (2012, mayo, 25). *Eric Schmidt: "los gobiernos representan la mayor amenaza para Internet".* Extraído el 31 de mayo de 2015 desde: http://hipertextual.com/2012/05/eric-schmidt-internet-gobiernos

[2]De acuerdo a lo establecido en el *International Data Corporation (IDC)*: "Android dominated the market with a 78.0% share". Extraído el 31 de mayo de 2015 desde: http://www.idc.com/prodserv/smartphone-os-market-share.jsp

Mencionado lo anterior me animé a continuar el trabajo de fin de carrera realizado por otro alumno de la escuela que explicaré con más detalle en el estado de la cuestión.

Además, debo de agradecer el apoyo que tuve al principio del profesor David González Ortega de la Universidad de Valladolid por darme la oportunidad de concederme la realización de dicho trabajo y las pautas que me aportó para comenzar con la investigación y dónde empezar a buscar sobre el tema.

La importancia de esta investigación recae en la adquisición de las señales obtenidas concretamente de los sensores relativos al electrocardiograma, al electromiograma, un acelerómetro situado a la altura del diafragma y el receptor galvánico de la piel, puesto que todos ellos nos proporcionan señales a partir de las cuales podemos obtener, mediante una serie de algoritmos matemáticos y procesamiento de señales, parámetros de gran importancia, como pueden ser el conjunto QRS del corazón, el potencial eléctrico de los músculos, tiempo entre inspiraciones o cuál es la resistencia ofrecida por nuestra piel a la conductividad eléctrica. Su obtención nos va a permitir cálculos como la frecuencia cardiaca, la distancia entre los diferentes latidos y la forma de la señal, pudiendo realizar un diagnóstico de posibles afecciones y patologías sin necesidad de acudir a un centro hospitalario. Esta detección va a favorecer el diagnóstico temprano así como un seguimiento y tratamiento adecuado por parte de equipos médicos especializados en cardiología, de forma que se puedan prevenir fallecimientos derivados de los temas mencionados.

Por otro lado, cabe la posibilidad de realizar un estudio en tiempo real del grado de fatiga en los sujetos, realizando la transformada de Fourier discreta sobre un vector de datos con la distancia entre picos del electrocardiograma, denominado R-R y a continuación, calcular la densidad espectral de potencia del mismo. Aplicando el cociente entre la potencia relativa a la baja frecuencia y la alta potencia, se puede observar el predominio del sistema nervioso simpático, el cual induce a la activación, en relación al sistema nervioso parasimpático que induce a la relajación. En función del valor obtenido, se puede realizar una aproximación sobre si el paciente está entrando en fase de sueño o si por el contrario no corre peligro de ello.

Esta utilidad es muy válida para su aplicación a la monitorización de los conductores, pudiendo prevenir accidentes de tráfico cuyas causas hayan sido despistes debido a la fatiga o que el conductor se haya dormido al volante.

I.1. Estado de la cuestión

Para la elaboración de este trabajo comencé a investigar diferentes artículos sobre el procesamiento de señales médicas para la detección de la fatiga a través de las señales obtenidas por sensores ECG.

A pesar de la existencia de una cantidad considerable de información sobre el objeto de la investigación, muchos de estos datos son repetidos y otros dispersos sin relación directa con la investigación y sin relación con la plataforma de sensores utilizada y su desarrollo sobre el sistema operativo Android.

La selección bibliográfica sobre el tema de la detección de somnolencia en las personas o las patologías relacionadas con el corazón, ha dado como resultado una gran cantidad de documentos, de los cuales he recogido los que me han parecido más interesantes y completos para la aplicación de esta investigación. Todos ellos aparecerán detallados en la sección correspondiente al final del trabajo.

Para comenzar a investigar, consideré que, antes de concentrarme en el tema central del trabajo, tenía que indagar sobre el funcionamiento del corazón, las señales producidas debido a la actividad eléctrica del corazón, analizando así su forma y sus parámetros principales con el objetivo de tener una visión global de la misma y unas férreas bases de donde partir. Siguiendo el mismo patrón, estudiamos las señales derivadas del potencial eléctrico de la activación de los músculos y la respuesta galvánica de la piel.

Por este motivo, orienté la búsqueda en primer lugar en este aspecto, y centrándome en encontrar datos sobre su análisis para su incorporación en la aplicación, detallando en una de las opciones de menú de la aplicación el funcionamiento, de tal manera que cualquier persona que utilice la aplicación, sea capaz de entender grosso modo la tarea que realiza.

Una vez realizada esta búsqueda, nos encontramos con una gran cantidad de documentos sobre la actividad del corazón y la interpretación de un electrocardiograma así como la detección de anomalías y grado de fatiga en los sujetos. Sin embargo, había escasez de datos sobre su procesamiento en la plataforma del sistema operativo Android y el lenguaje de programación Java.

Con todos los datos recogidos de los diferentes libros y artículos, se pueden saber, a grandes rasgos y de manera teórica, los parámetros clave y la forma de procesar las señales, con diagramas de bloques bastante claros, finalizando con una demostración de cuál sería el resultado. Sin embargo, no detallan o bien cuál han sido los sensores que han empleado para la realización de la investigación presentada en el artículo o bien cuál ha sido el lenguaje de programación o software utilizado. En los pocos artículos que lo detallan, están relacionados con el software de *National Instruments* llamado *LabView* o con otro software de gran potencia como es *MatLab*.

Sobre el funcionamiento del corazón he encontrado lo siguiente:

-(López Ramírez, 2006).

Este libro es un método rápido necesario para la comprensión del electrocardiograma (ECG) enfocado, en primer lugar, a estudiantes de medicina o médicos especialistas en medicina general. Con la lectura del libro se consigue que, en unos pocos días, se pueda ser capaz de leer e interpretar un ECG, es decir, tener una visión global del funcionamiento del corazón. Sobre todo me ha servido el apartado introductorio, en cual se detalla con exactitud la actividad eléctrica del corazón y qué relación tiene cada una de las ondas con la misma. (pp.3-5 de (López Ramírez, 2006)). Además, la parte relacionada con el ritmo cardiaco me ha sido de utilidad para el análisis y posible detección de los distintos ritmos junto a sus anomalías en relación con el ritmo cardiaco sinusal, considerado el normal del órgano más importante de nuestro organismo. (pp.13-36 de (López Ramírez, 2006)). Los diferentes intervalos entre las ondas mencionadas anteriormente se detallan en el capítulo 7. (Pp.73-82 de (López Ramírez, 2006)).

-(García Ramiro, 2013). *Los seis segundos del ECG. El generador de ritmos cardiacos estáticos y dinámicos.* Universidad de Jaén.[3]

El articulo explica de forma clara cuáles son los diferentes ritmos del corazón, con sus características propias y particulares para cada uno de ellos, bien sea por la frecuencia del ritmo sinusal, distancia entre intervalos de ondas etc... Además demuestra gráficamente cómo son cada uno de ellos.

Me ha ayudado a comprender cuáles son los diferentes ritmos con sus parámetros propios para determinar en la aplicación si el ritmo es normal o alguna anomalía está presente.

-*Ritmo cardiaco recomendado y avalado por Medicina21 y por PortalesDeMedicos.com* (2015, 30 de Mayo).[4]

De la misma manera que los anteriores me ha ayudado a comprender cuáles son los distintos ritmos del corazón, la forma de leer correctamente el electrocardiograma y las características principales de las ondas derivadas de la actividad eléctrica.

Sobre la detección de la fatiga mediante el procesado de la señal eléctrica del corazón he encontrado los artículos:

-Bonjyotsna,A y Roy,S (2014,Mayo). Correlation of Drowsiness with Electrocardiogram: A Review. 3(5). (Pp.9538-9544).

El artículo detalla la metodología para la detección de la somnolencia en una persona mediante una señal perteneciente a un electrocardiograma. La señal ECG es una de las señales biomédicas más importantes que puede ser empleada para modelar el comportamiento cardiaco de una persona. Actualmente se ha convertido en esencial para la detección del cansancio en un individuo que está realizando tareas importantes como la conducción de un automóvil. La variabilidad de la frecuencia cardiaca (HRV) se deriva de los intervalos RR, relacionados directamente con el sistema nervioso autónomo, constituido por el sistema nervioso simpático y parasimpático. La variabilidad de la señal se somete a un análisis espectral, observando la banda de baja frecuencia (LF) y alta frecuencia (HF), que van a ser los dos parámetros que vamos a emplear para determinar el estado de somnolencia del sujeto.

[3](García Ramiro, 2013). Extraído el 31 de mayo de 2015 desde http://www4.ujaen.es/~pgramiro/
[4]*Ritmo cardiaco recomendado y avalado por Medicina21 y por PortalesDeMedicos.com* (2015, 30 de Mayo). Extraído el 31 de mayo de 2015 desde: http://www.my-ekg.com/como-leer-ekg/ritmo-cardiaco.html.

Además, expone que la tasa del ritmo cardiaco medible con el sensor ECG y la conductividad de la piel mesurable con el receptor galvánico (GSR) están más relacionadas con el nivel de fatiga de un conductor. El electrooculograma también sería de utilidad para, medir el intervalo temporal entre el cierre y la apertura de los ojos, pero no se dispone de un sensor con esas características tan precisas.

Me ha ayudado a entender el diagrama de bloques propuesto para la detección de los complejos QRS y, posteriormente, medir el intervalo entre los picos RR. Como los picos RR no se encuentran a una distancia equidistante en tiempo, la señal de la que hemos obtenido el vector de datos, no ha sido muestreada uniformemente por lo que será necesario realizar una re-muestreo y una interpolación del mismo. Previamente, la señal se tiene que pasar por un filtro paso-alto y un filtro paso-bajo con el objetivo de eliminar las bandas de frecuencia no deseadas. Una vez interpolado el vector de datos, se procede a realizar su análisis en frecuencia y cálculo del ratio LF/HF para detectar la somnolencia.

El artículo (Stein, et al., 1997) detalla cómo varía el ratio entre hombres y mujeres atendiendo a su edad y rango horario, bien si es día o de noche, apreciando un visible cambio entre ambas.

El artículo (Sun, at al., 2011) describe el desarrollo de técnicas de medición incorporadas en vehículos para la toma de señales fisiológicas (frecuencia cardiaca y respiratoria y electroculagrama para medir el *eye blinking*) de los conductores. Estas señales se utilizarán para detectar la aparición de síntomas de fatiga en el conductor mediante el empleo de electrodos aprovechando la actividad eléctrica de nuestro organismo.

Establece la frecuencia cardiaca (HR) y su variabilidad (HRV) como unos buenos indicadores de somnolencia, los que tienen variaciones más bruscas y experimentan un descenso. Por otra parte, la disminución de la frecuencia respiratoria y un aumento del parpadeo para mantener los ojos abiertos se detallan como otros aspectos a tener en cuenta.

- Xun Yu (2009, 5 de Mayo). Real-time Nonintrusive Detection of Driver Drowsiness

El artículo explica que la somnolencia es una de las principales causas de los accidentes de tráfico graves, por lo que su estudio es tremendamente importante debido al impacto social que tiene. La monitorización continua del estado físico y psicológico del conductor es muy importante para reducirlos. La investigación propone una técnica no intrusiva en tiempo real mediante el uso de biosensores con el objetivo de detectar cuáles son los picos del complejo QRS, calcular el intervalo de tiempo entre ellos durante un periodo de tiempo y contemplar las variaciones producidas en la frecuencia cardiaca así como la detección a partir del cociente entre la baja y la alta de la densidad espectral de potencia.

-Sundaraj, Kenneth, Murugappan, Murugappan y Sahayadhas, Arun (2012,7 de Diciembre). Detecting Driver Drowsiness Based on Sensors: A Review. (pp. 16937-16949).

El artículo realiza en primer lugar un análisis de las causas y el número de víctimas en accidentes de tráfico haciendo hincapié en el porcentaje de estos que se han producido porque el conductor se ha quedado dormido al volante. De acuerdo con las estadísticas la somnolencia es una de las principales causas de accidentes de tráfico.

Propone el estudio y análisis de las señales procedentes del sensor ECG, con el objetivo de encontrar un relación directa entre la fatiga de cada automovilista con la señal obtenida. La variabilidad en el ritmo del corazón hace posible que se puedan distinguir diferentes etapas del sueño. Analizando los RR y su análisis en frecuencia es posible determinarlo.

-Medicore (Sin fecha). *Heart Rate Variability Analysis System: Clinical Information.*

El manual ayuda a comprender las variaciones presentes en el ritmo sinusal del corazón así como su posterior análisis en el dominio de la frecuencia incluyendo el significado desde el punto de vista médico que tiene cada banda de frecuencia y el ratio entre la alta y la baja frecuencia. La frecuencia muy baja (VLF) se puede incluir junto con la baja frecuencia (LF) debido a que ambas están relacionadas con la actividad simpática de nuestro sistema nervioso.

Sobre la resistencia galvánica de la piel relacionada con la fatiga he encontrado los siguientes artículos:

-Kurniawan, Hindra, Maslow, Alexadre y Pechenizkiy,Mykola.(Sin fecha). Stress Detection from Speech and Galvanic Skin Response Signals.

El artículo menciona la posibilidad de medir los cambios fisiológicos como la presión sanguínea y la resistencia galvánica de la piel (GSR) pudiendo ser monitorizadas e investigadas. En estudios realizados sobre la conductancia de la piel se pueden medir los cambios producidos y ver la variación de los valores dentro de un rango. Dichos valores son enviados en tiempo real, en nuestro caso por el sensor correspondiente de *Shimmer Sensing* a través de Bluetooth. Los sensores están situados en la mitad de los dedos corazón y anular de la mano derecha. Las pruebas consistieron en una serie de test elementales y otros aritméticos a los que fueron incrementando progresivamente su dificultad de manera que el esfuerzo mental fuese mayor. La segunda prueba consistió en una sesión de relajación de 10 minutos de duración. La señal GSR, experimenta un gran cambio cuando el sujeto está sometido a un mayor estrés y tiene que actuar frente a los diferentes estímulos de manera rápida, siendo los intervalos con los valores más altos aquellos en los que se está realizando el test con mayor intensidad. Es importante tener en cuenta una normalización de los valores para hacer un análisis más objetivo de los resultados obtenidos en los diferentes sujetos que han realizado la prueba. Cuando el sujeto afronta la segunda prueba, consistente en un sesión de relajación se produce un descenso considerable de la señal GSR.

- Milan Stork (Sin fecha). Some methods systems and sensor which are possible for driver’s drowsiness estimation.

Describe la utilidad de la señal procedente de un sensor GSR, su posibilidad de estudio durante la conducción situando los electrodos en el volante y la descripción de la señal, forma y fases que puede tener .

- Zocchi,C, Giusti,A y Rovetta,A (Sin fecha). Biosensor for microsleeps detection during drive simulations.

El artículo detalla cuál es el rango normal de la resistencia galvánica de la piel situada en un rango entre los 15 y 30 kilo Ohmios. En diferentes pruebas y análisis realizados con diferentes grados de actividad, la señal tiene valores bajos menores de 5kΩ, intermedia con valores entre 5 y 10 kΩ, altos con valores entre de 10 a 20 kΩ y muy alta con los valores a más de 20 kΩ.

- Muruganezhumali, Venkatsubramaniam (2015,15 de febrero). Designing for advance identification of inevitable drowsiness using galvanic skin response.

El artículo detalla que cuando un individuo se está quedando dormido o entrando en fase de somnolencia durante la conducción, la resistencia galvánica disminuye.

Sobre la detección de la fatiga mediante el procesado de la señal procedente de acelerómetros y la variación de la frecuencia respiratoria:

- Neil Douglas, David White, Cheryl Pickett, John Weil, Clifford Zwillich (1982). Respiration during sleep in normal man.

El artículo describe cómo durante la fase previa al sueño y sus primeras fases se produce una disminución de la frecuencia respiratoria

- Brandy Warwick, Nicholas Symons, Xiao Chen, Kaiqi Xiong (Sin fecha). Detecting Driver Drowsiness using Wireless Wearables.

Detalla cómo la frecuencia respiratoria experimenta un descenso brusco cuando el sujeto entra en una fase de somnolencia además de tener una respiración más profunda aumentado el tiempo de duración de la inspiración y espiración. Propone aplicar a la señal un ventana de *Hanning* para disminuir el ruido, reducir el *aliasing* en caso de que se produzca por una frecuencia de muestreo excesivamente baja y analizar el espectro mediante la densidad espectral de potencia observando cuál ha sido su variación. Este método solo serviría para detectar cuándo se produce el cambio de estado en el individuo debido a que una vez cambiado de fase la densidad espectral de potencia (PSD) volvería a ser casi una delta por las mínimas variaciones que se producirían.

-Sara Lapi, Federico Lavorini, Giovanni Borgioli, Marco Calzolai, Leonardo Masotti, Massiomo Pistolesi, Giovanni A.Fontana (2013,12 de noviembre). Respiratory rate assessments using a dual-accelerometer device.

El artículo en primer lugar destaca la importancia de la medición de la *respiratoy rate* (RR) debido a la cantidad de información importante que podemos extraer a través de ella desde afecciones pulmonares hasta problemas de miocardio.

En segundo lugar explica cómo a partir del uso de dos acelerómetros podemos calcularla así como el algoritmo matemático detallado paso a paso.

Sobre la detección de la fatiga mediante el procesado de la señal eléctrica procedente de la activación de los músculos:

- Venkatesh Balasubramanian, B.E., Ph.D.-, K. Adalarasu, B.E., M.Tech (2006, Septiembre). EMG-based analysis of change in muscle activity during simulated driving

La primera parte del artículo nos hace una breve introducción a las diferentes maneras en las que se puede analizar la fatiga mediante el electromiograma, el electrocardiograma, el encefalograma y el electrooculagrama y a la adquisición de medidas de otros parámetros fisiológicos como la presión sanguínea, la cantidad de oxígeno en sangre y las respiraciones realizadas en un minuto.

El artículo establece un vínculo directo entre el estrés y una excesiva carga de trabajo de los individuos con la fatiga tanto a nivel muscular como a nivel psicológico. Establece la relación entre el espectro de la señal recogida por un sensor EMG y la somnolencia, produciéndose cambios notables en las frecuencias medias. A medida que el sujeto se encuentra más fatigado, se produce un desplazamiento hacia bajas frecuencias, es decir la señal en frecuencia se traslada hacia la izquierda.

El artículo hace hincapié en la gran ayuda que nos proporciona la transformada de Fourier para analizar el espectro de las señales en frecuencia.

- M.B.I.Raez, M.S.Hussain, F.Mohd-Yasin(2006, 23 de Marzo). Techniques of EMG signal analysis: detection, processing, classification and applications (pp.11-35).

El artículo explica el funcionamiento de la activación de los músculos mediante los impulsos eléctricos y el sistema nervioso periférico, cómo las señales de electromiografía se pueden utilizar para aplicaciones biomédicas y cuáles son los métodos avanzados para la detección, descomposición, procesamiento, reconstrucción y clasificación de la señal y todos los factores que influyen en ella.

- Khadmaoui, Amine (2014). *Desarrollo de una aplicación para el control de sensores fisiológicos mediante un dispositivo Android.* Proyecto fin de carrera, Ingeniero de telecomunicación, Universidad de Valladolid, España.

Destaco también en este punto, el proyecto fin de carrera titulado "*Desarrollo de una aplicación para el control de sensores fisiológicos mediante un dispositivo Android.*" de Amine Khadmaoui, del que he partido para mi TFG, aunque tenga otra

visión. Por un lado, me ha servido como motivación, ya que el conocer la existencia de un trabajo con esta temática, me dio pie a elaborar mi propio trabajo de investigación, pero en este caso, sobre el procesamiento de señales biomédicas sobre la plataforma. Por otro, también me ha sido útil, para conocer, aunque sea de una manera superficial parte del código desarrollado en la aplicación sobre el cuál haya podido desarrollar mis investigaciones acerca del procesado de señales extraídas de los sensores de *Shimmer.*

I.2. Objetivos

El objetivo principal de este trabajo es estudiar las señales biomédicas procedentes de los sensores ECG y GSR. Para ello, se planteó la siguiente hipótesis: Mediante el procesamiento de señales procedentes de los sensores de Shimmer se puede medir el grado de somnolencia de un individuo. Para demostrarla planteo los siguientes objetivos específicos:

- Estudiar las señales procedentes del ritmo sinusal cardiaco, su forma y sus características principales, estableciendo unos parámetros de estudio como son la onda P y el complejo QRS.

- Estudiar la señal procedente del sensor GSR, con la finalidad de obtener la resistencia galvánica de nuestra piel y el análisis de la intensidad de la actividad que está desempeñando el sujeto en relación el resultado obtenido.

- Estudiar la señal procedente del sensor EMG, con el objetivo de obtener los impulsos eléctricos de la activación de los músculos, con la finalidad de observar si se produce un desplazamiento del espectro hacia las bajas frecuencias como consecuencia de la fatiga muscular.

- Estudiar la señal procedente de dos acelerómetros para medir la frecuencia respiratoria del individuo.

- Realizar un estudio de los procesados adecuados para ambas señales, siguiendo los diagramas de bloques necesarios para obtener los parámetros mencionados en los dos puntos anteriores y sacar el ratio entre la baja y la alta en el caso del ECG y las variaciones producidas para el caso de la GSR.

- Analizar la función del sistema nervioso autónomo en relación con las variaciones del ritmo cardiaco y la resistencia galvánica de la piel con el fin de detectar el grado de fatiga.

- Analizar la función del sistema nervioso autónomo en relación con las variaciones producidas a frecuencia intermedia en el espectro de la señal eléctrica procedente del sensor EMG con la finalidad de analizar el grado de fatiga.

- Analizar el bi-espectro de las señales ECG y EMG calculando los momentos y utilizando los métodos de clasificación LDA y QDA para determinar con la máxima fiabilidad el estado en el que se encuentra el conductor.

Estos objetivos marcarán el guion que se debe seguir para realizar nuestro trabajo de fin de grado.

I.3. Metodología y fuentes consultadas

Para la elaboración de este trabajo he necesitado consultar libros, algunos de ellos recogidos en el estado de la cuestión, así como artículos de revistas científicas, fuentes en internet y fotografías.

Las bibliotecas y centros de documentación consultados a fin de encontrar datos sobre el estudio de las señales biomédicas, el funcionamiento del corazón, la resistencia galvánica de la piel, los tipos de señales que podemos recoger mediante la gama de sensores que hemos utilizado en el trabajo así como su correcta interpretación, sus métodos de procesado y la relación entre las mismas y el estado físico y mental del conductor o paciente, han sido: Biblioteca Universitaria de la Facultad de Ciencias de Valladolid, Biblioteca Online de la Universidad de Valladolid y Biblioteca de Medicina y Odontología de la Universidad de Salamanca.

Los documentos audiovisuales son también una fuente importante para ver cómo se recoge y se representa en tiempo real una señal ECG a través del sensor y el funcionamiento del corazón.

En cuanto a las fuentes *online,* nos encontramos con una gran cantidad de páginas web especializadas en la publicación de artículos científicos. En la página web oficial de *Shimmer* (http://www.shimmersensing.com) aportan una suma importante de datos y noticias de interés, las especificaciones técnicas de los sensores

de la gama, así como vías de investigación que tienen ellos abiertas. Debido a que es un sitio oficial y su suficiente prestigio posee máximo rigor y carácter científico-académico. Por otro lado, ofrecen la posibilidad de contactar directamente con ellas a través de direcciones de correo electrónico y redes sociales como *Twitter o Facebook*

Otras páginas web de interés que cabe destacar son: la del repositorio de revistas científico-técnicas *Science Direct* (http://www.sciencedirect.com), la cual ofrece información sobre la una gran variedad de publicaciones, artículos y revistas de los diferentes campos sociales, científicos y humanísticos y la página web (http://www.ieee.org) que, del mismo modo que la anterior ofrece un amplio repertorio de artículos científicos de calidad.

Con todos estos datos recopilados, ordenados y su análisis y estudio comparativos comencé la realización del trabajo siguiendo los puntos expuestos a continuación:

1) Familiarización con el entorno de desarrollo *Android SDK Manager* así como las bibliotecas proporcionadas por *Shimmer* (*Shimmer Android Driver Library* V 2.1).

2) Manipulación y comprobación del funcionamiento de los sensores, diferenciando cada uno de ellos con sus respectivas funcionalidades y la forma en la que guardan los datos recibidos.

3) Familiarización con el funcionamiento de la aplicación y búsqueda de nuevas utilidades para la misma adaptándola a las nuevas necesidades.

4) Programación del procesado de señal adaptado a cada uno de los módulos que vamos a utilizar así como el código correspondiente para la obtención de los datos para su análisis descrito en los artículos de investigación.

5) Programación empleando una API de representación gráfica para mostrar los ratios de la variabilidad de la frecuencia cardiaca obtenidos.

6) Programación del procesado *offline* de la señal ECG.

7) Realización de un botón para que únicamente transmitan los sensores seleccionados en la aplicación de manera sincronizada, conjunta y simultánea.

8) Realización de una opción en el menú para que transmitan todos los sensores en las mismas condiciones que el punto anterior.

9) Familiarización con el entorno de desarrollo matemático *MatLab* para el procesado *offline* de los datos almacenados en los ficheros por la aplicación.

10) Desarrollo de los scripts para la obtención de las características principales de cada una de las señales de acuerdo con lo consultado en los artículos de investigación.

11) Clasificación del estado físico del conductor en función de los parámetros obtenidos, que permita su diferenciación basándonos en métodos de clasificación.

12) Finalmente, la redacción de la memoria donde se detallen las tareas y los logros conseguidos durante este trabajo fin de grado.

I.4. Medios

Durante el desarrollo de este trabajo se utilizará el ordenador portátil Toshiba Satellite A-660 13-T con las siguientes características:

1) Procesador: Intel Core i5-430M (2,26 Ghz / 3 Mb. Caché)

2) Memoria: 4 Gb. DDR3 a 1066 Mhz. (Ampliable a 8 Gb.

3) Disco Duro: 640 Gb. a 5400 rpm.

4) Sistema Operativo: Windows 7 Home Premium (64 bits)

5) Red: Fast Ethernet (10/100), Wifi-N y Bluetooth.

6) Tarjeta Gráfica: nVidia Geforce GT 330M (1 Gb. DDR3).

7) Pantalla: 16" HD (1366x768) Trubrite con tecnología LED

8) Dimensiones: 380 x 254 x 36 mm

9) Peso: 2,62 Kg

Figura 1 Ordenador portátil utilizado para el desarrollo del TFG[5]

Para la instalación de la aplicación y la realización de pruebas hemos utilizado el dispositivo móvil LG L Bello con las siguientes características:[6]

1) **Procesador:** 1.3 GHz Quad Core

2) **Memoria Interna**: 8 Gb (La memoria disponible para el usuario es menor debido a que tiene albergado el Sistema Operativo y las aplicaciones preinstaladas).

3) **Sistema Operativo**: Android 4.4.2 KitKat.

4) **Memoria RAM:** 1 Gb.

5) **Red**: Wifi (802.11 b/g/n) y Bluetooth 4.0.

6) **Pantalla**: 5" IPS FWVGA

7) **Dimensiones**: 138.1 x 70.6 x 10.7

8) **Peso**: 137 g.

[5] "Ordenador Portátil Toshiba Satellite A660-13T" extraído el 12 de agosto de 2016 desde: http://www.audiotronics.es/images/products/500_113665_3.jpg?size=500
[6] "LG L Bello" extraído el 12 de agosto de 2016 desde: http://www.movilzona.es/lg/l-bello/

Figura 2 *Smartphone* utilizado en el desarrollo del TFG y realización de pruebas[7]

I.5. Estructura del trabajo

La estructura de este trabajo de investigación se divide en ocho capítulos. El primer capítulo contiene la introducción en la que expongo y justifico la elección del tema elegido como objeto de investigación; el estado de la cuestión en el que elaboro una recopilación y análisis de una gran cantidad de publicaciones que existen para el procesamiento de señales médicas para la detección de la fatiga a través de las señales obtenidas por sensores ECG y GSR; los objetivos perseguidos en esta investigación; la metodología que he seguido a la hora de investigar y recoger la información, así como las fuentes consultadas y la estructura del trabajo que cierra este capítulo.

El segundo capítulo se divide en dos apartados. En el primero presento la seguridad en la conducción y su relación con la fatiga del conductor y lo relacionado con su estado físico y mental. En el segundo comienzo a realizar un estudio sobre el estado en el que se encuentran los conductores y el efecto que tendrá este en el nivel de seguridad en la conducción así como los riesgos a los que está expuesto.

El tercer capítulo consta de cuatro apartados. Cada uno presenta el estudio sobre cada uno de los diferentes sensores de la gama *Shimmer* disponibles en grupo de investigación GTI (Grupo de Telemática e Imagen) de la Universidad de Valladolid, así como un estudio general acerca de su *hardware* y de los elementos de los que está compuesto, junto a sus especificaciones técnicas. Por otra parte, se explicarán otro tipo de aplicaciones donde están siendo empleados y su finalidad.

[7] "LG L Bello" extraído el 12 de agosto de 2016 desde: http://www.lg.com/es/telefonos-moviles/lg-LBello-D331-dorado.

El cuarto capítulo habla del sistema operativo para *smartphones y tablets* Android, que es el más expandido en la actualidad, su arquitectura, las últimas versiones y la descripción del entorno de desarrollo que se va a utilizar.

El quinto capítulo está centrado en un resumen del estudio de las señales que se obtienen de cada uno de los sensores así como un estudio detallado de cada una de ellas relacionadas con el funcionamiento del corazón y las regiones del cuerpo en las que se colocarán los sensores o las diversas zonas.

En el sexto capítulo se explican las pruebas realizadas empleando los sensores, el resultado de las mismas tanto en la aplicación *Android* como realizando el mismo procesado con *MatLab* y haciendo un análisis bi-espectral de alto orden espectral con el objetivo de clasificar el estado físico del conductor a partir de una base de datos de señales conociendo previamente dicho estado

En el séptimo capítulo detallo un presupuesto económico del coste que tendría este trabajo fin de grado.

En el octavo capítulo expongo las conclusiones obtenidas de la investigación y por último el trabajo finaliza con la relación de la bibliografía y *webgrafía* manejada.

II. SEGURIDAD GENERAL EN LA CONDUCCIÓN

En este apartado se abordará de manera general el tema de la seguridad general en la conducción, las causas más frecuentes por las que ocurren los accidentes de tráfico y propuestas para la mejora del estado físico del conductor durante el manejo del automóvil.

II.1 Seguridad y fatiga en la conducción

La definición otorgada por la RAE del término *accidente* es la siguiente: "Suceso eventual o acción de que resulta daño involuntario para las personas o las cosas."[8]

La Dirección General de Tráfico (DGT) clasifica la evolución del accidente teniendo en cuenta el área:

- Área de Percepción: Comprende el espacio entre el punto de percepción posible y el punto de conflicto.
- Área de maniobra: Comprende el espacio entre el punto de decisión y el punto de conflicto.
- El Área de conflicto: Comprende el espacio entre el punto clave y la posición final.[9]

Gran parte de los accidentes de tráfico originados en España causan solamente daños materiales, aunque las estadísticas sobre el número de accidentes mortales tienen gran transcendencia para la salud de la población.

Según el informe más reciente de la Dirección General de Tráfico el número de accidentes de tráfico con víctimas ha aumentado un 2% respecto al año 2013 y del total entre un 20-30% ocurrieron por algún tipo de distracción. [10] Si nos fijamos solamente en los accidentes causados por la distracción, detectamos la baja atención y concentración al volante, provocado principalmente por el cansancio. Según las estadísticas la fatiga, es la cuarta causa de mortalidad en las carreteras españolas.

[8] Real Academia Española (2014). *Diccionario de la lengua española* (22ª ed.). Extraído el 2 de agosto de 2016 desde: http://dle.rae.es/srv/fetch?id=0KUeoUu
[9] Dirección General de Tráfico (2011) "Fases de un accidente" pp.8-10 extraído el 2 de agosto de 2016 desde: www.dgt.es/Galerias/la-dgt/empleo-publico/.../TEMA_62__-_Parte_General.doc
[10] "Las principales cifras de la siniestralidad vial: España 2014" extraído el 5 de abril de 2016 desde: http://www.dgt.es/Galerias/seguridad-vial/estadisticas-e-indicadores/publicaciones/principales-cifras-siniestralidad/2015-2228_principales_cifras_de_la_Siniestralidad_Vial_2014_ACCESIBLE.pdf

La fatiga altera la capacidad auditiva y visual. Esto afecta tanto a los reflejos como a la manera de conducir, que será más automática y menos activa. Los síntomas de la fatiga y el cansancio pueden ser pesadez en los ojos, parpadeo constante, brazos y pies adormecidos, entre otros. Hay tres factores que influyen en el aumento de la fatiga: el vehículo (ruidos, vibraciones…), medio (clima, hora del día…) y conductor (edad, inexperiencia…).

En un reportaje realizado por la Dirección General de Tráfico en verano de 2015, dan siete consejos para evitar la fatiga durante la conducción, sobre todo para viajes largos (a partir de las seis horas de viaje):[11]

- Dormir: Se aconseja dormir al menos siete horas antes de un viaje. Tampoco es conveniente realizar un trayecto largo justo después de una jornada laboral.

- Temperatura: Se deben evitar las altas y bajas temperaturas, siendo una temperatura ideal los 24º C aproximadamente.

- Síntomas: Hay que tener consciencia de cuáles son los síntomas de la fatiga con el fin de evitar accidentes. Estos pueden ser parpadeos, malestar físico, cansancio, errores en la conducción…

- Paradas: Conviene parar a descansar cada dos horas o 200 kilómetros.

- Hidratación. Hay que mantenerse bien hidratado (agua o refrescos), ya que la ausencia de líquidos ocasiona una disminución de la atención, dolor de cabeza y cansancio muscular.

- Bebidas con cafeína: Según un estudio de INTRAS (Instituto de Tránsito y Seguridad Vial. Universitat de València) y FESVIAL (Fundación española para la seguridad vial) [12] asegura que descanso y consumo de café reduce un 36,38% las pisadas de las líneas de la calzada, un 7,7% el tiempo de reacción y una reducción del nivel de somnolencia del 25,92%.

- Nunca se debe consumir alcohol si se va a conducir y se debe consultar a un médico si está tomando algún medicamento que pueda afectar a la conducción.

[11] "7 Consejos para evitar la fatiga" extraído el 5 de abril de 2016 desde: http://revista.dgt.es/es/reportajes/2015/07JULIO/0714consejos-para-evitar-la-fatiga.shtml#.VwQMyqSg_IU

[12] "El café en la conducción de vehículos: análisis de sus efectos en conductores fatigados o somnolientos" extraído el 5 de abril de 2016 desde: http://www.cgpsst.net/wp-content/uploads/2015/05/Estudio-Caf%C3%A9-y-conducci%C3%B3n.pdf

Los factores a tener en cuenta para que se den las condiciones idóneas a la hora de manejar el vehículo son la vía, el vehículo y el factor humano (de este último hablaré en el siguiente apartado) como se aprecia en la siguiente figura.

Figura 3 Triángulo de la Seguridad Vial

En cuanto a la vía, debe haber una buena infraestructura, mantenimiento y reparación de las carreteras así como una buena señalización para una mayor seguridad. Respecto al vehículo, tendrá que estar en unas correctas condiciones centrándonos, sobre todo, en la puesta a punto de todos los elementos de seguridad del vehículo, como pueden ser los neumáticos o las luces entre otros. Además, se tiene que tener en cuenta la antigüedad del vehículo, ya que según la nueva campaña de vigilancia intensiva de la DGT (julio 2016)[13], ha aumentado en estos primeros seis meses el número de fallecidos que viajaban en coches que tenían más de 14 años de antigüedad (comparada con la cifra del 2014 que era de 12 años). Así mismo, el riesgo de fallecer o resultar herido grave es mayor con vehículos de este tipo. La Inspección Técnica de los Vehículos (ITV) es primordial para la seguridad vial, ya que un 4% de las personas fallecidas en lo

[13] "Las condiciones del vehículo centran la nueva campaña de vigilancia intensiva de la DGT" extraído el 3 de agosto de 2016 desde: http://www.dgt.es/es/prensa/notas-de-prensa/2016/20160711-condiciones-vehiculo-centran-nueva-campania-vigilancia-intensiva-dgt.shtml

que llevamos de año (2016) se debe a accidentes relacionados con que el vehículo implicado tenía la ITV caducada.

II.2 Estado físico del conductor

El conductor es la persona que maneja el mecanismo de dirección o va a los mandos.

Como mencioné en el anterior apartado, los factores a tener en cuenta para que se den las condiciones idóneas a la hora de manejar el vehículo son la vía, el vehículo y el factor humano.

En relación con el factor humano, la Dirección General de Tráfico y algunas investigaciones revelan que el 90% de los accidentes mortales en carretera se deben al mismo[14].

La tarea básica del conductor se puede resumir en cuatro aspectos: *percepción*, en la que este observa a través de los sentidos toda la información sobre la carretera, el automóvil y de sí mismo; *previsión*, en la cual procesa la información, la interpreta y puede prever lo que va a suceder; *decisión*, establece cómo va a actuar*; y acción*, realiza lo que ha decidido. En este aspecto hay que tener en cuenta las condiciones y capacidades físicas del conductor.

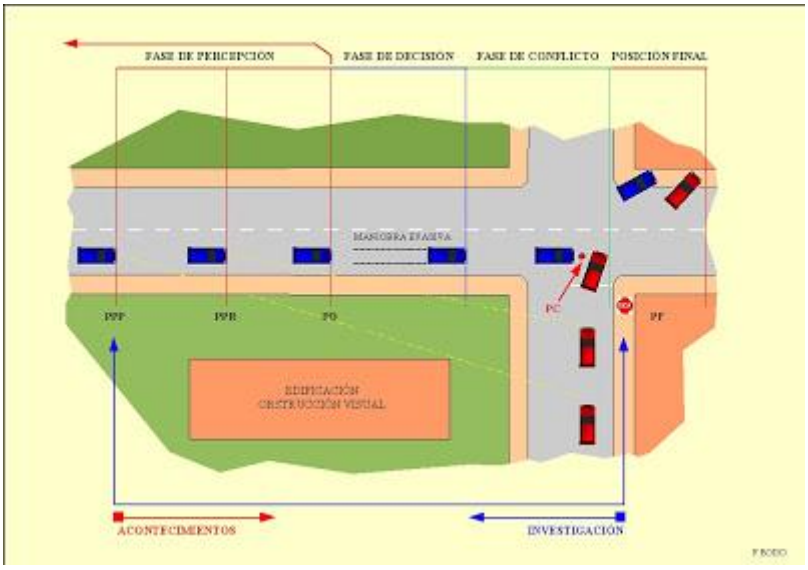


Figura 4. Fases del accidente[15]

[14] "Factor humano" extraído el 3 de agosto de 2016 desde: http://www.dgt.es/revista/archivo/pdf/num150-2001P.18-20.pdf

Para que estas cuatro tareas básicas se realicen en unas condiciones idóneas, hay que tener en cuenta las causas que influyen en las aptitudes del conductor:

- El estado físico y psíquico del conductor: Los procesos físicos y psicológicos implicados en la conducción están relacionados entre sí. El conductor debe tener una buena capacidad perceptiva y atencional para observar, identificar y distinguir todo lo que ocurre a su alrededor, por lo que ha de poseer una buena capacidad visual y auditiva, así como un buen mantenimiento del sistema locomotor para tener buenos reflejos. Además también forman parte del proceso psicológico la capacidad intelectual, la toma de decisiones, la capacidad de respuesta y la personalidad.

- Educación vial: La educación vial no debe ir dirigida exclusivamente a los conductores, sino a todos los ciudadanos, ya que de una manera u otra todos forman parte del escenario donde se desarrolla el tráfico, ya que, según la Dirección General de Tráfico, la educación vial "persigue la formación del comportamiento del ciudadano en tanto que es usuario de las vías públicas, ya sea como peatón, conductor o viajero." [16]

Esta educación no debe tomarse solamente como un simple aprendizaje de normas y señales de circulación, sino también, como una enseñanza para adquirir hábitos de comportamiento y actitudes frente al tráfico que hagan que exista una convivencia ordenada, solidaria y de respeto con la sociedad en la que vivimos.

- Formación de conductores: La educación vial del conductor es una de las bases principales a la hora de prevenir accidentes y mejorar la seguridad vial. Durante la formación se deben adquirir tanto habilidades y aptitudes para el correcto uso del automóvil, como conocimiento sobre las normas y señales que regulan la circulación de las vías públicas.

- Vigilancia y control: La presencia, actuación y control de los agentes encargados de la vigilancia del tráfico es fundamental a la hora de prevenir accidentes de tráfico. Estos imponen sanciones de tráfico con el fin de que no se produzcan nuevas infracciones.

[15] Boiso, Paco (2007) "Fases del accidente (gráfico 2)" extraído el 3 de agosto de 2016 desde: http://pantha-rei.blogspot.com.es/2007/02/fases-del-accidente-grfico-2.html

[16] "Cuestiones de Seguridad Vial, conducción eficiente, medio ambiente y contaminación" pp. 86-89 extraído el 3 de agosto de 2016 desde: http://www.dgt.es/Galerias/seguridad-vial/formacion-vial/cursos-para-profesores-y-directores-de-autoescuelas/doc/directores_2013/Seguridad-vial-Ed.-2013.pdf

III. SENSORES DE LA GAMA SHIMMER[17]

III.1 Descripción y utilidad

Los sensores fisiológicos de la gama *Shimmer* son una plataforma para la obtención de señales que sirven para medir distintos parámetros del cuerpo humano, es decir se utilizan en los grupos de investigación biomédica. Las señales pueden ser almacenadas en una memoria SD que cabe la posibilidad de incorporar en la ranura correspondiente de los sensores o bien ser transmitidas mediante una conexión *Bluetooth* a un ordenador o una aplicación instalada en un dispositivo móvil en tiempo real.

Figura 5 Sensores *Shimmer Platinum* empleados[18]

Las ventajas que presentan este tipo de sensores vienen dadas por su minúsculo tamaño, poco peso y bajo consumo energético. Sin embargo, su elevado coste lo hace poco viable para un uso cotidiano; pero factible desde un punto de vista médico, debido a su portabilidad y al precio de la tecnología empleada en los hospitales y/o centros de salud hoy en día.

[17]Las siglas SHIMMER se corresponden con las palabras inglesas "Sensing Health with Intelligence, Modularity, Mobility and Experimental Reusability" que significa "Sondeando la Salud con Inteligencia, Modularidad, Movilidad y Reutilización Experimental".

[18]"Sensors" extraída el 12 de agosto de 2016 desde: http://www.mdpi.com/sensors/sensors-15-26621/article_deploy/html/images/sensors-15-26621-g001-1024.png

Creado como un sensor portátil, *Shimmer* cuenta con un número de sensores que nos van a permitir la captura de un amplio abanico de eventos

III.2 Hardware

El hardware de los sensores está constituido por el microprocesador *MSP430* de arquitectura *RISC*[19] de 16 bits, de bajo consumo sobre todo en periodos de inactividad, el cual es el encargado de controlar el funcionamiento del dispositivo. Posee 16 Kilobytes de memoria *RAM* que es el máximo disponible en este tipo de sensores, 256 Kilobytes de memoria flash. Opera a una frecuencia de 24 MHz con 11 canales de 12 bits A/D de los cuales 7 permanecen libres para los módulos de expansión. La comunicación entre el microprocesador y el chip de radio 802.15.4 es del tipo serie-síncrona USART con el modo SPI. Ambos componentes comparten reloj maestro.

En la placa base se encuentra un acelerómetro que incorporan todos ellos con 3 ejes (X, Y y Z) y con un rango de ±2 g, ±4 g. ±8 g o ±16g. La sensibilidad es de 1000 LSB/g para ±2 g, que es la comúnmente utilizada.

Además de los dos elementos mencionados anteriormente consta de dos módulos de radio basados en el estándar 802.15 o Bluetooth con diseño Mitsumi WML-C46N CSR y 802.15.4, que pueden funcionar ambos simultáneamente y permanecer desconectados en periodos de inactividad para reducir el gasto del consumo del sensor. Cuando usa el modo 802.15.4, emplea un transceptor CC2420.

Dispone de un led para indicarnos el estado en el que se encuentra el sensor, siendo su color verde para indicar que el dispositivo se encuentra conectado, naranja si está transmitiendo paquetes de datos y rojo si su batería está baja. Además cuenta con una antena de 2.4GHz, una ranura microSD, un pulsador para encender, apagar o reiniciar el dispositivo (manteniéndolo 10 segundos pulsado), un conector externo utilizado para cargar la batería de litio de 450 mAh y y/o actualizar el *firmware*.

[19]Las siglas RISC se corresponden con las palabras inglesas “Reduced Instruction Set Computer” que significa “Ordenador con Conjunto de Instrucciones Reducidas”.

En la siguiente figura se pueden observar las diferentes partes del *hardware* del sensor:

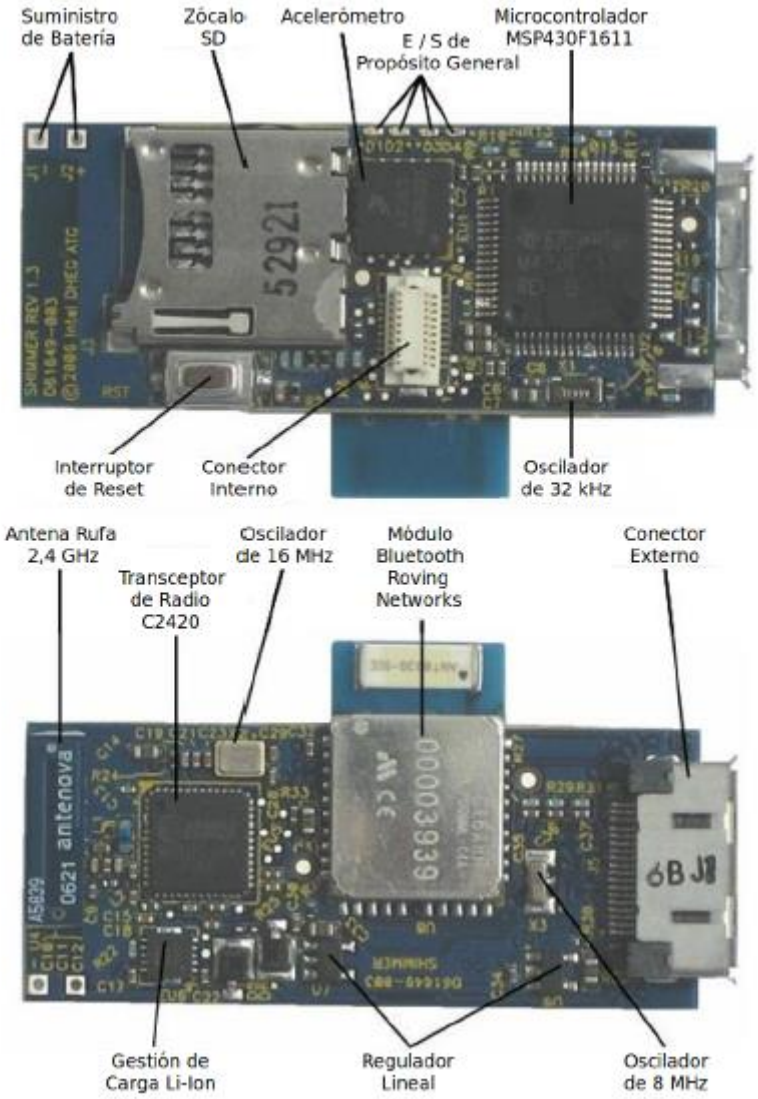


Figura 6 Partes detalladas de un sensor[20]

[20] "Plataforma *Shimmer"* extraída el 12 de agosto de 2016 desde: http://zeitgeistlab.ca/doc/Advanced_WBANs_for_an_Ageing_e-Health_Society.html

III.3 Tipos de sensores

III.3.1 EMG (Electromiograma)

El sensor EMG o electromiograma se encarga de evaluar y registrar la señal de activación de los músculos, siendo esta útil para el diagnóstico de la actividad biomecánica de los mismos. Al mismo tiempo, se estudian los nervios que se encargan de enviar la señal motora al grupo muscular correspondiente. Nuestro sensor nos ofrece una solución inalámbrica, una fácil manera de integrarlo en cualquier grupo además de una disposición ergonómica valiosa.

Consta de dos chips ADS1292R del fabricante *Texas Instruments* con 11 bytes de registros configurables.

Figura 7 Sensor EMG

Las características técnicas del sensor son:

1) Transmisión de datos a una tasa de 10.2, 51.2, 102.4 y 128 Hz . El último modelo permite una velocidad de 125, 250, 500, 1000, 2000, 4000 y 8000 muestras por segundo.

2) Ancho de banda de 8.4 KHz.

3) Conexiones: Dos canales de entrada positivo y negativo respectivamente y uno de referencia. El último modelo del sensor EMG de *Shimmer* dispone de otros dos canales de entrada adicionales.

4) Peso: 32.5 gramos.

5) Dimensiones de 65 x 32 x 12 mm

6) Memoria EEPROM de 2048 bytes.

Es una técnica no invasiva que consiste en la detección de los impulsos por medio de tres electrodos de bajo voltaje desechables situados estratégicamente sobre el tejido muscular sobre el que se quiere realizar el estudio o monitorizar.

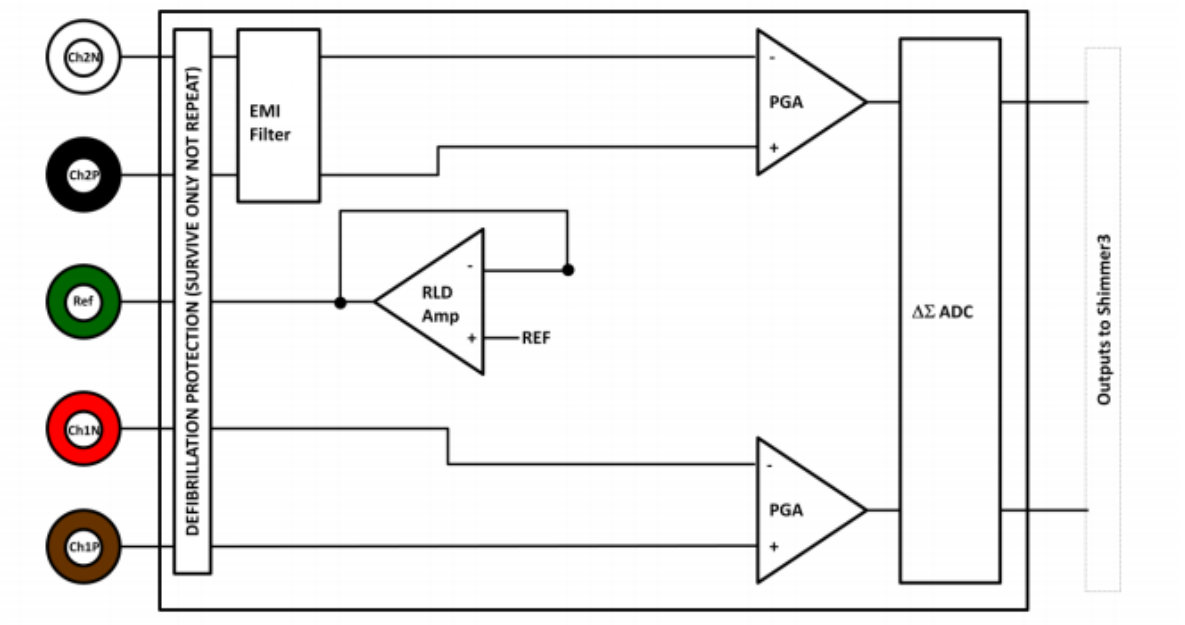


Figura 8 Diagrama de Bloques Simplificado del sensor EMG[21]

El filtro EMI reduce la interferencia electromagnética, tiene un ancho de banda de 3MHz y una ganancia de -3 dB.

El amplificador *RLD* se emplea para contrarrestar interferencias comunes como son las procedentes de líneas de alta tensión, luces fluorescentes, etc…

El amplificador de ganancia *PGA* aumenta la señal de entrada de acuerdo con la calibración que se ha realizado de forma automatizada debido a que está programada en el software en los canales de tal manera que la señal que se obtenga se encuentre en un rango de 2mV pico a pico y una frecuencia de 0.05 Hz y 159 Hz. Puede ser diferente para cada uno de los canales.

El convertidor analógico-digital convierte las señales analógicas de entrada a una representación digital por un total de 24 bit con signo entero para cada una de las muestras obtenidas. Estos datos recibidos en el último bloque del diagrama son lo que se transmitirán por Bluetooth y/o se almacenarán en la tarjeta SD.

[21] "EMG sensor" extraída el 5 de abril de 2016 desde: www.shimmersensig.com

Finalmente, los electrodos desechables en el sensor disponible en el laboratorio tienen tres canales de entrada. El motivo del uso de los tres para capturar la señal es debido a que la amplitud de la señal es muy pequeña en relación con el ruido procedente de fuentes eléctricas cercanas y no es eliminada con el filtro EMI del primer punto. Es decir, la señal captada consta del ruido del entorno y la señal eléctrica local procedente de los músculos con los que está en contacto.

Si restamos la señal procedente tanto del electrodo positivo como del negativo, el ruido será eliminado mientras que las componentes deseadas del EMG se verán amplificadas facilitando además su procesado. Este procesado se denomina de rechazo del modo común (CMR).

La posición de los electrodos es muy importante para una adquisición correcta de la señal. El polo positivo y el negativo deben estar colocados en paralelo con las fibras con tendencia a estar situado en el centro del músculo. El electrodo neutro de referencia ha de ser posicionado en un punto neutro eléctricamente hablando de nuestro cuerpo y lo más alejado posible del músculo que estamos estudiando. Es recomendable situarlo en prominencias como tobillo, muñeca, codo, rodilla etc…

Tenemos que tener en cuenta estas 4 recomendaciones:

1) El material del electrodo debería ser Ag/AgCl y la distancia ideal entre el electrodo positivo y el negativo es de 20 milímetros. El tamaño de electrodo no debería exceder los 10 milímetros.

2) Los electrodos no deben ser situados sobre el tendón del músculo debido a que en ese punto las fibras son más finas y la señal EMG tendría una amplitud menor.

3) Los electrodos no deben ser colocados en el punto motor del músculo. Esto tendría como consecuencia una inexactitud en la señal recibida porque recogería tanto actividad nerviosa, como muscular.

4) Los electrodos no deben ser ubicados en los bordes exteriores del músculo porque captarían señales adyacentes y la principal se vería distorsionada.

III.3.2 ECG (Electrocardiograma)

El sensor ECG o electrocardiograma se encarga de evaluar y registrar la señal eléctrica procedente del corazón en cada latido cardiaco en forma de gráfica continua. Es la herramienta principal de la cardiología cuya importancia destaca en el diagnóstico y monitorización de las enfermedades y afecciones cardiovasculares. Es un sistema empleado en hospitales mediante un dispositivo con unas notables dimensiones que suelen estar las 24 horas conectados al paciente usando entre 3 y 12 electrodos como son los monitores *Holter*.

Figura 9 Monitor *Holter*[22]

Sin embargo la portabilidad y tamaño de nuestro sensor nos facilitará su integración apoyándonos de una cinta ergonómica que viene incluida en el paquete.

Las características del sensor son las siguientes:

1) Ganancia configurable a niveles 1, 2, 3, 4, 6, 8 y 12.

2) Transmisión de datos a una tasa de 10.2, 51.2, 102.4 y 128 Hz. El último modelo permite una velocidad de 125, 250, 500, 1000, 2000, 4000 y 8000 muestras por segundo.

3) Ancho de banda de 8.4 KHz.

4) Peso: 31 gramos.

5) Dimensiones: 65 x 32 x 12 mm

[22] "ECG Monitor" extraída el 5 de abril de 2016 desde: http://monitorbest.elektroshop91.com/windows-monitor/ecg-monitoring-485.html

6) Memoria EEPROM de 2048 bytes.

7) Conexiones de entradas RA, LA, LL y RL normalizadas a 2 mV pico a pico.

Figura 10 Sensor ECG

Es una técnica no invasiva que consiste en la detección de la actividad eléctrica del corazón por medio de cuatro electrodos de bajo voltaje desechables situados estratégicamente sobre la caja torácica para monitorizar la actividad eléctrica del corazón.

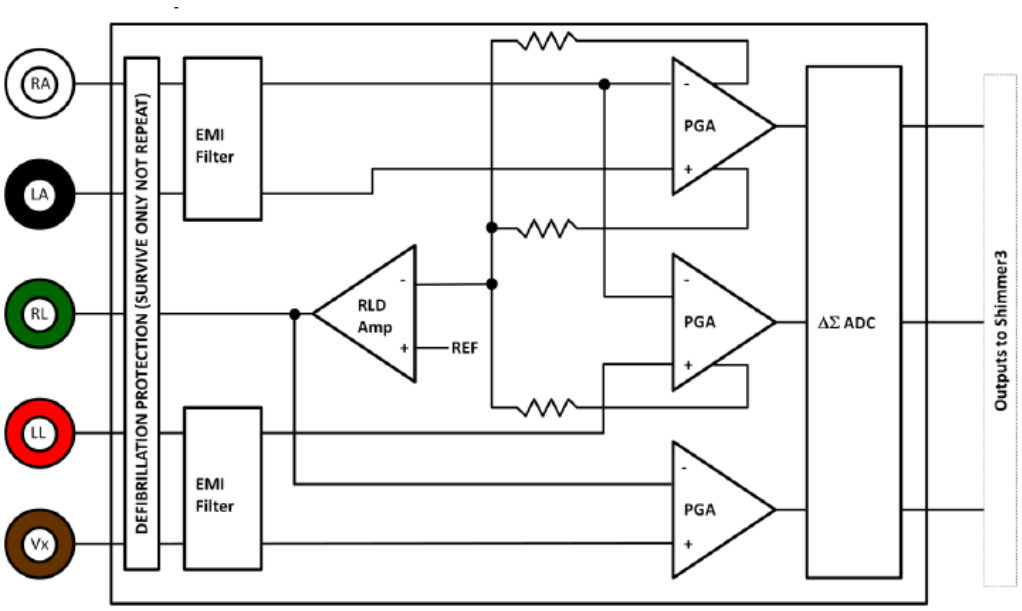


Figura 11 Diagrama de bloques simplificado del sensor ECG [23]

[23] "ECG sensor" extraída el 5 de abril de 2016 desde: www.shimmersensig.com

En primer lugar, la protección frente a la desfibrilación se encuentra presente en las entradas RA y LA con el objetivo de facilitar la demodulación evitando el ruido procedente de la respiración.

Al igual que en el sensor EMG, el filtro EMI reduce la interferencia electromagnética, con un ancho de banda de 3MHz y una ganancia de -3 dB y el amplificador *RLD* se emplea para contrarrestar interferencias comunes como son las procedentes de líneas de alta tensión, luces fluorescentes, etc…

El amplificador de ganancia *PGA* aumenta la amplitud de la señal de entrada de acuerdo con las siete ganancias disponibles teniendo una por defecto. La ganancia se puede configurar a través del *software* adecuándola a nuestras necesidades siguiendo las especificaciones detalladas en la documentación del sensor. La relación entre la señal de salida y la señal ECG en unidades de mVoltios viene dada por la fórmula:

$$(ADC\ Output - ADC\ Offset).ADC\ Sensitivity = ECG\ Signal\ in\ mVolts.Gain$$

Donde, despejando la señal ECG, nos queda:

$$ECG\ Signal\ in\ mVolts = \frac{((ADC\ Output - ADC\ Offset).ADC\ Sensitivity)}{Gain}$$

Teniendo en cuenta que la *ADC Sensivity* es igual a:

$$ADC\ Sensitivity = \frac{Vref}{ADC\ Max} = \frac{2420\ mVolts}{2^{23}-1}.$$

La medida para determinar el Offset de cada canal se lleva a cabo conectando las dos entradas de cada uno y la ganancia se determina mediante la fórmula:

$$Gain = \frac{\left((Max\ ADC\ Output - ADC\ Offset).ADC\ Sensitivity\right)}{Max\ Input\ Signal\ in\ mVolts}$$

Figura 12 Fórmula ganancia sensor ECG

Es recomendable que la señal que se obtenga se encuentre en un rango de 2mV pico a pico y una frecuencia de 0.05 Hz y 159 Hz.

El convertidor analógico-digital convierte las señales analógicas de entrada a una representación digital por un total de 24 bit con signo entero para cada una de las muestras obtenidas. Estos datos recibidos en el último bloque del diagrama son lo que se transmitirán por Bluetooth y/o se almacenarán en la tarjeta SD.

Finalmente, la situación de los electrodos desechables que se conectan al sensor disponible en el laboratorio en sus cuatro canales de entrada con las siguientes derivaciones:

RA = En el brazo derecho (*Right Arm*), evitando prominencias óseas. Su situación es encima del corazón.

LA = En el brazo izquierdo (*Left Arm*), evitando prominencias óseas. Su situación es a la izquierda del corazón.

LL = En la pierna izquierda (*Left Leg*), evitando prominencias óseas. Su situación es en la parte baja izquierda del corazón.

RL = En la pierna derecho (*Right Leg*), evitando prominencias óseas. Su situación es debajo del electrodo LA

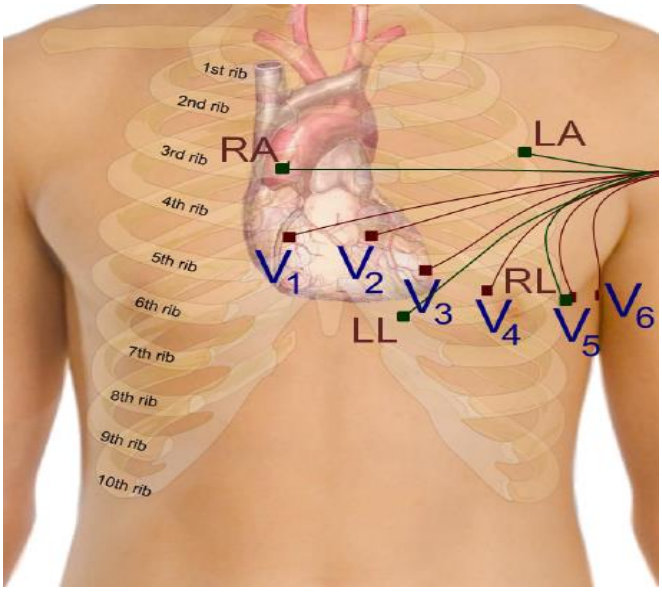


Figura 13 Posición para la colocación de los electrodos[24]

Las derivaciones tienen su origen en el triángulo de Einthoven siendo este el nombre de su investigador Willen Einthoven (1860,1927), planteando que el corazón actúa como un generador de energía eléctrica y el cuerpo humano es un conductor.

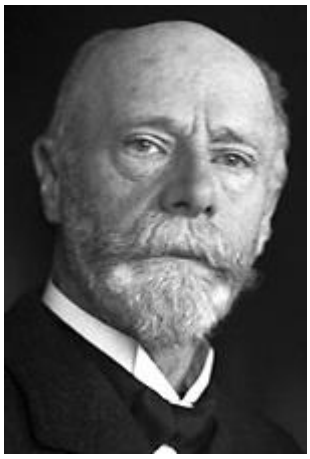


Figura 14 Willen Einthoven[25]

Con esta teoría, construye un triángulo invisible alrededor del corazón donde se proyectan las líneas eléctricas producidas por la contracción y dilatación del músculo cardiaco. A continuación, se establece la polaridad de cada uno de los vértices. Las

[24]"ECG sensor" extraída el 5 de abril de 2016 desde: www.shimmersensig.com.
[25]" Willen Einthoven" extraído el 6 de abril de 2016 desde : http://www.nobelprize.org/nobel_prizes/medicine/laureates/1924/einthoven-bio.html

derivaciones del brazo izquierdo (LA) y la pierna izquierda (LL), presentan una polaridad positiva debido a que reciben un elevado potencial del ventrículo izquierdo y de la cara diafragmática del corazón. En cambio las polaridades de las derivaciones correspondientes a la parte derecha serán negativas porque la base del corazón se proyecta sobre él mismo.

Las primeras derivaciones estándar son la resta de los potenciales de las 3 señales obtenidas con los electrodos situados en las posiciones anteriormente mencionadas siendo D1, D2 y D3:

D1 = Brazo izquierdo – Brazo derecho.

D2 = Pierna izquierda – Brazo derecho.

D3 = Pierna izquierda – Brazo izquierdo.

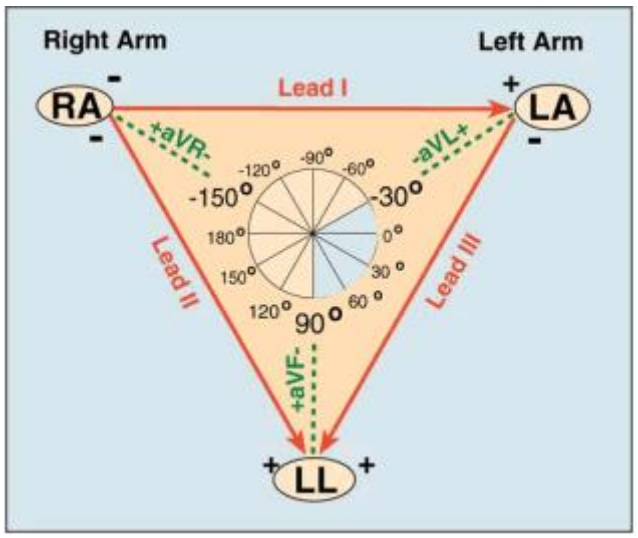


Figura 15 Triángulo de Einthoven[26]

III.3.3 GSR (Galvanic Skin Response)

El sensor GSR o sensor galvánico de la piel se encarga de evaluar y registrar la señal eléctrica procedente de los cambios producidos en la resistencia de la piel en forma de gráfica continua. Depende de ciertos tipos de glándulas sudoríparas presentes en las manos y en los dedos. Es una herramienta utilizada para medir las emociones de

[26] "Triángulo de Willen Einthoven" extraída el 6 de abril de 2016 desde: http://electrocardiogramaymedicinageneral.blogspot.com.es/2013/02/triangulo-de-einthoven.html

las personas antes diferentes estímulos y situaciones de estrés. Por ejemplo esta técnica es utilizada en los detectores de mentiras.

Figura 16 Sensor GSR

Las características del sensor son las siguientes:

1) Consumo de 60 µA.

2) Transmisión de datos a una tasa de 10.2, 51.2, 102.4 y 128 Hz. El último modelo permite una velocidad de 125, 250, 500, 1000, 2000, 4000 y 8000 muestras por segundo.

3) Rango de medidas de: 10 kΩ a 4.7 MΩ.

4) Rango de frecuencia de 15.9 Hz.

5) Peso: 30 gramos.

6) Dimensiones: 65 x 32 x 12 mm

7) EEPROM: 2048 bytes.

8) Dos conexiones de entrada.

Es una técnica no invasiva que consiste en la monitorización de los cambios producidos en la resistencia galvánica de la piel por medio de dos electrodos de bajo voltaje desechables situados estratégicamente sobre las palmas de las manos o las yemas de los dedos.

III.3.4 Shimmer Dock

El *Shimmer Dock* es dispositivo multifunción que puede desempeñar las siguientes funciones:

1) Cargar los sensores.

2) Acceder a la información almacenada en la tarjeta micro SD.

3) Programar internamente el sensor de acuerdo con las especificaciones incluidas en la guía del usuario.

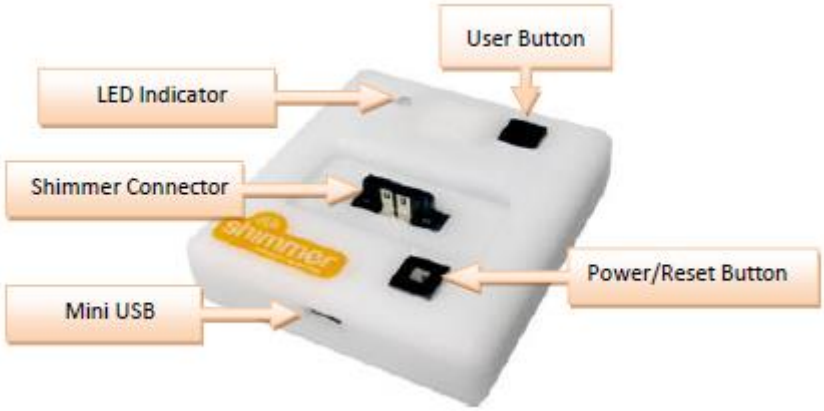


Figura 17 *Shimmer Dock*[27]

El botón Power/Reset se utiliza para encender el sensor o resetearlo mediante un pulsado rápido y para apagarlo manteniéndolo pulsado durante 8 segundos (*Shimmer,* 2015). El led permanecerá de color verde cuando se encuentre encendido.

El botón de usuario se utiliza cuando se quiere aplicar una señal concreta al sensor. El indicador de carga permanecerá de color naranja mientras se esté cargando, apagándose una vez finalizada dicha tarea.

Esta fase tiene tres etapas:

1) Fase 1 o *Preconditioning Phase*: Es una fase de acondicionamiento y se entra en ella cuando el voltaje de la batería ha caído por debajo de un valor umbral. No debe caer por debajo del mismo pero si permanece inactivo durante un tiempo prolongado puede llegar a un valor por debajo del umbral mínimo. Durante esta fase se aplica una corriente baja de unos 12.5 mA hasta que se alcanza el voltaje mínimo deseado. El tiempo en esta fase dependerá del grado de descarga que presente la batería. Es importante destacar que durante esta fase el LED se encontrará apagado.

[27]"*Shimmer Dock*" extraída el 6 de abril de 2016 desde: www.shimmersensing.com.

2) Fase 2 o Primary Charging Phase: Es una fase primaria de carga estándar aplicando una corriente constante de 125 mA. Durante esta fase el LED permanecerá de color amarillo.

3) Fase 3 o *Conditioning Phase*: Es la fase de acondicionamiento de la batería mientras permanezca insertada en el *Shimmer Dock* disminuyendo progresivamente la corriente aplicada hasta que sea menor que el umbral mínimo. Dependiendo como esté programado el firmware que tenga programado el dispositivo el indicador LED permanecerá encendido de color amarillo, intermitente o apagado de nuevo.

La duración estimada de carga es de 4.5 horas.

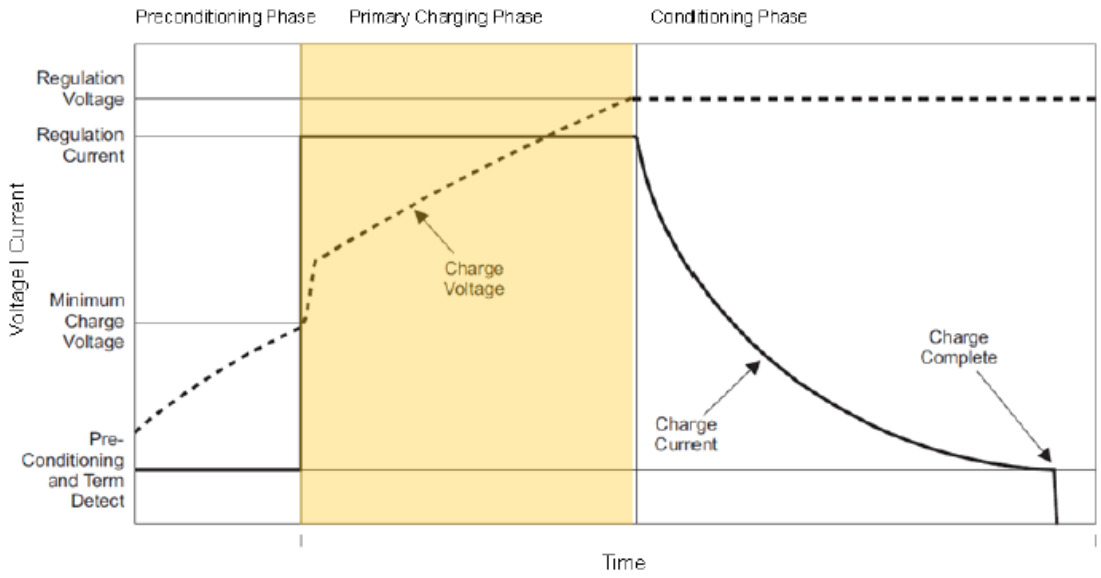


Figura 18 Etapas de carga de los sensores *Shimmer*[28]

El segundo led situado en la esquina izquierda parpadeará de color azul cuando el ordenador haya accedido a la microSD del sensor y de color naranja cuando se programe el BSL o la actividad de la UART.

Internamente dispone de un conmutador DIP utilizado para asegurar la compatibilidad con los dispositivos que se conecten. Trae la configuración por defecto, sin embargo si por alguna circunstancia deseamos cambiarla, tendríamos que abrir el bloque y activar los interruptores de acuerdo con las combinaciones válidas:

[28] "*Shimmer Dock*" extraída el 6 de abril de 2016 desde: www.shimmersensing.com.

Función	Interruptor 1	Interruptor 2	Interruptor 3	Interruptor 4
Configuración recomendada por Shimmer	1	1	1	0
Configuración legal	1	1	0	1
Configuración Solo Acceso tarjeta SD	0	0	0	1

Tabla 1 Configuraciones posibles *Shimmer Dock*[29]

Figura 19 Conmutadores *Shimmer Dock*[30]

III.3.5 Calibrado 9DOF

La calibración del acelerómetro se realiza a través del software proporcionado por la marca llamado *Shimmer 9DoF Calibration v2.8* y siguiendo estrictamente los pasos detallados en la guía de usuario.

En primer lugar debemos seleccionar el rango del acelerómetro que queremos usar y presentamos en siguiente tabla:

Hardware Version	Accelerometer Range	Accelerometer Enabled
Shimmer 2	1.5g,2g,4g,6g	N/A
Shimmer 2r	1.5g,6g	N/A
Shimmer 3	2g	Analog/Low Noise
Shimmer 3r	2g,4g,8g,16g	Digital/Wide Range

Tabla 2 Rango de acelerómetros y tipo de acuerdo con la versión *hardware*[31]

Tenemos que tener cuidado y saber cuál es la versión de hardware de nuestro sensor para no introducir unos parámetros no válidos en la interfaz. Los parámetros de calibrado son específicos para cada rango.

[29] [30] y [31] "*Shimmer Dock*" extraída el 6 de abril de 2016 desde: www.shimmersensing.com.com

Una vez calibrado el sensor si queremos cambiar el rango es necesario una posterior calibración para obtener unos datos con el mínimo error posible.

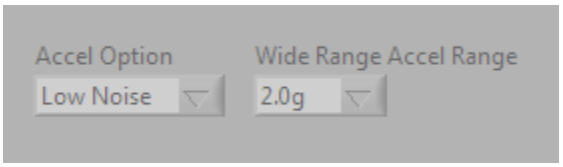


Figura 20 Rango acelerómetro y tipo

Vinculamos el sensor a través de Bluetooth, pulsamos en "Detalles" y apuntamos el número de puerto COM que tiene asignado. En la esquina superior derecha de la interfaz *software* lo introducimos, pulsando a continuación en el botón de color azul "CONNECT". Ahora ya tenemos el dispositivo totalmente vinculado y podemos empezar a transmitir, dándole al botón de superficie verde "START".

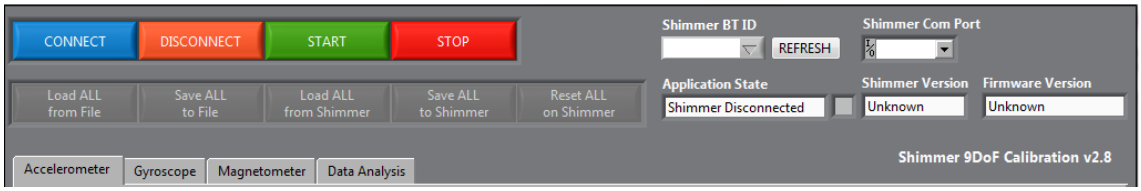


Figura 21 Pantalla del Calibración 9DOF

Una vez realizada esta tarea procedemos a seguir los siguientes pasos:

1) Colocamos el sensor de tal manera que el eje X apunte en dirección del vector de gravedad, es decir, hacia arriba. A continuación, mantenemos dicha posición y pulsamos el botón X+g. Inicialmente se ha puesto de color verde y esperamos hasta que se apague. A continuación colocamos el sensor de tal manera que el eje X apunte en dirección contraria del vector de gravedad, es decir hacia abajo y seguimos los mismos pasos pero pulsando el botón X-g.

2) Colocamos el sensor de tal manera que el eje Y apunte en dirección del vector de gravedad, es decir, hacia arriba. A continuación mantenemos dicha posición y pulsamos el botón Y+g. Inicialmente se ha puesto de color verde y esperamos hasta que se apague. A continuación colocamos el sensor de tal

manera que el eje Y apunte en dirección contraria del vector de gravedad, es decir hacia abajo y seguimos los mismos pasos pero pulsando el botón Y-g.

3) Colocamos el sensor de tal manera que el eje Z apunte en dirección del vector de gravedad, es decir, hacia arriba. A continuación, mantenemos dicha posición y pulsamos el botón Z+g. Inicialmente se ha puesto de color verde y esperamos hasta que se apague. A continuación, colocamos el sensor de tal manera que el eje X apunte en dirección contraria del vector de gravedad, es decir, hacia abajo y seguimos los mismos pasos pero pulsando el botón Z-g.

Figura 22 Calibrado de los ejes del acelerómetro

Si los parámetros situados a la derecha se muestran no válidos o contienen "NaN", contienen un error y no se ha logrado el proceso de calibración por lo que será necesario repetirlo.

Una vez finalizados los 3 pasos, ya tenemos el acelerómetro calibrado y listo para funcionar. También está disponible la calibración del giróscopo y del magnetómetro.

IV. ANDROID

IV.1 Introducción

Es un sistema operativo que fue creado por una empresa que fue adquirida por el gigante *Google* en 2005 y tuvo su éxito a partir del 2008 cuando se unió al proyecto denominado *Open Handset Alliance* formado por 48 empresas dedicadas al desarrollo de *software, hardware* y telecomunicaciones. La alianza se propuso impulsar este software aunque su éxito ha sido fundamentalmente gracias a *Google* por la publicación de su código fuente y *Apache* por ofrecer plataformas a proyectos basados en código abierto.

Es un sistema operativo basado en el *kernel* de Linux para los teléfonos móviles, *tablets*, *netbooks*, reproductores de música mp4 e incluso ordenadores personales. Permite trabajar en un entorno de Java, ejecutar aplicaciones en una máquina virtual con la ventaja respecto a otros sistemas operativos móviles de que cualquier persona con unos conocimientos de programación va a ser capaz de crear nuevas aplicaciones e incluso modificar su propio sistema operativo gracias a que es de código libre.

IV.2 Arquitectura de Android

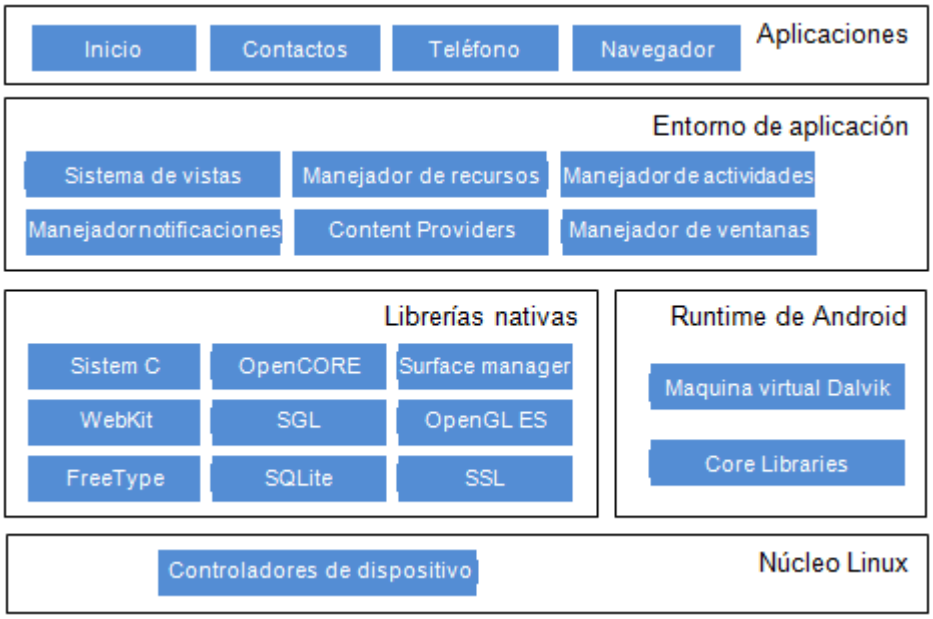


Figura 23 Arquitectura de *Android*[32]

Está constituida por:

1) **Núcleo de Linux:** Dispone del sistema operativo Linux versión 2.6. Se encarga de proporcionar servicios básicos como la seguridad, la gestión de memoria, *drivers* de dispositivos, multiprocesos y la pila de protocolos de red.

2) **Runtime de Android:** Está basado en el concepto de máquina virtual empleado por Java. Utiliza la máquina virtual Dalvik optimizada para dispositivos con poca memoria y capacidad de proceso.

3) **Bibliotecas nativas:** Dispone de bibliotecas en C/C++ usadas por varios componentes de Android y compiladas en el código nativo del procesador.

4) **Entorno de aplicación:** Es la plataforma de desarrollo para las aplicaciones. Su objetivo consiste en simplificar la reutilización de componentes. Las aplicaciones pueden publicar sus capacidades para que otras puedan hacer uso de ellas. Además, ofrece soporte para el desarrollo de aplicaciones basadas en Java con compatibilidad con la mayor parte de las clases del entorno JRE.

[32] "Arquitectura de Android" extraída el 10 de Mayo de 2016 desde: http://www.androidcurso.com/index.php/tutoriales-android/31-unidad-1-vision-general-y-entorno-de-desarrollo/146-las-versiones-de-android-y-niveles-de-api

Proporciona lo siguientes servicios:

4.1) **Views:** Es la parte visual de las aplicaciones.

4.2) **Resource Manager:** Proporciona acceso a los recursos que no son código.

4.3) **Activity Manager:** Gestiona el ciclo de vida de las aplicaciones y proporciona un sistema de navegación entre las partes que la constituyen.

4.4) **Notification Manager:** Permite a las aplicaciones mostrar alertas personalizadas en la barra de estado.

4.5) **Content Providers**: Es un mecanismo sencillo para acceder a los contenidos de otras aplicaciones.

5) **Aplicaciones:** Conforma el conjunto de aplicaciones instaladas en un sistema Android desarrolladas en Java a través del Android SDK o C/C++ usando Android NDK.

La tienda oficial para la compra/descarga de las aplicaciones se llama *Play Store* instalada por defecto en la gran mayoría de los dispositivos. Facilita la búsqueda de aplicaciones por temática, gratuitas o de pago, etc…

Con fecha de 1 de agosto de 2016 exponemos en la siguiente tabla y el siguiente gráfico el tanto por ciento de dispositivos que ejecutan las distintas versiones de Android.

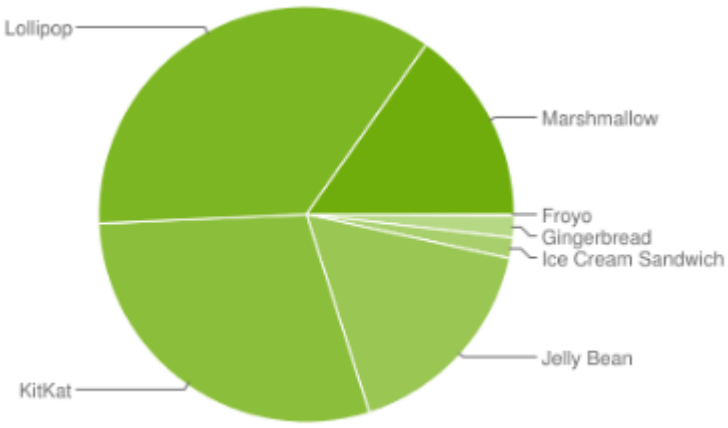


Figura 24 Gráfico con porcentajes de utilización de las distintas versiones de *Android*[33]

Versión	Nombre	API	Distribución
2.2	Froyo	8	0.10%
2.3.3 - 2.3.7	Gingerbread	10	1.70%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.60%
4.1.x	Jelly Bean	16	6.00%
4.2.x		17	8.30%
4.3		18	2.40%
4.4	KitKat	19	29.20%
5	Lollipop	21	14.10%
5.1		22	21.40%
6	Marshmallow	23	15.2

Tabla 3 Versiones de *Android* y porcentaje de utilización

IV.3 Versiones

Es importante elegir para qué versión vamos a programar la aplicación debido a que hay clases y métodos que no están disponibles para las muy antiguas. Antes de empezar a diseñar la app debemos conocer cuál va a ser la versión mínima necesaria.

Cabe destacar de manera importante, que cuando se lanza una nueva plataforma únicamente se le añaden funcionalidades y no tiene problemas de compatibilidad con las versiones anteriores. En caso de eliminar alguna, será declarada obsoleta pero se puede

[33]"Platform Versions" extraída el día 10 de Mayo de 2016 desde: https://developer.android.com/about/dashboards/index.html?hl=es#Platform

continuar utilizando. Todas las plataformas vienen identificadas de tres maneras que son la versión, el nivel de API y el nombre comercial excepto las dos primeras.

IV.3.1 KitKat

Android 4.4 con un nivel de API 19 cuyas incorporaciones más importantes fueron la posibilidad de que el sistema operativo estuviera presente en un número mayor de dispositivos sumando todos aquellos con una memoria RAM inferior a 512 MB. Esto se consiguió con la creación de una API que adapta el comportamiento de la aplicación a dispositivos con menor capacidad. Además, los *WebViews* se basan en el mismo *software* empleado por *Google Chrome* pudiendo mostrar contenido en HTML5, se mejora la conectividad de NFC para poder realizar pagos y la mejora de sensores disminuyendo su consumo. Otro punto a favor es la facilitación del acceso a las aplicaciones a la nube estableciendo un nuevo tipo de almacenamiento hacia vías futuras mediante su *content provider* también conocido como *document provider* y un administrador de impresión para poder enviar documentos a través de la red WiFi a una impresora.

Finalmente, incorpora una máquina virtual ART que logra unos tiempos de ejecución del código escrito en Java muy superiores a la máquina *Dalvik* (utilizada por defecto) a pesar de estar en una fase experimental.

IV.3.2 Lollipop

Android 5.0 con un nivel de API 21 con la novedad de la extensión del sistema operativo a nuevas plataformas como son *Android Wear, Android TV* y *Android Auto*. Se producen cambios significativos en la arquitectura al cambiar definitivamente la máquina virtual *Dalvik* por la ART y soporta dispositivos de 64 bits en los procesadores x86, MIPS y ARM.

Respecto al consumo de batería se incorpora una nueva API que permite la gestión de las tareas programando aquellas más costosas únicamente cuando el dispositivo esté cargando y con otro tipo de condiciones para otras. Cabe destacar que el modo ahorra de batería se encuentra activado por defecto.

Relacionado con el campo gráfico se incorpora el soporte *OpenGL ES* 3.1 permitiendo añadir funcionalidades gráficas más avanzadas y el cambio en el diseño de la interfaz de usuario incluyendo el cambio de los iconos que hacen referencia a "retroceder", "inicio" y "aplicaciones" por un triángulo, un círculo y un cuadrado, respectivamente.

En marzo de 2015 se actualiza a la versión 5.1 con una API de nivel 22 ofreciendo soporte para varias tarjetas SIM en un mismo dispositivo y el permiso para que empresas proveedoras de servicios de telecomunicación puedan subir sus aplicaciones al *Google Play.*

IV.3.3 Mashmallow

Android 6.0 con un nivel de API 23 lanzada en octubre de 2015. Presenta la novedad de administrar los permisos de tal manera que los usuarios puedan retirar ciertos permisos u otorgarlos, dando una mayor protección a la privacidad de sus datos. Se añade autentificación por huella digital a la API, la compartición de documentos más sencilla con la incorporación de *Direct Share* (permitiendo escoger tanto la aplicación como el usuario con el que queremos compartir la información) y la posibilidad de montar y extraer *pen drives*.

Finalmente, se añade la plataforma de pagos combinada con *NFC Android Pay* y un gestor de batería nuevo llamado *Doze.*

IV.4 Entornos de desarrollo

La programación de las aplicaciones en *Android* se realiza normalmente utilizando el lenguaje Java y el conjunto de herramientas de desarrollo *SDK* ofrecidas por *Google* aunque existen otras plataformas disponibles utilizando otros lenguajes de programación y sin un entorno nativo como, por ejemplo, *Basic 4Android* que emplea *VisualBasic* y *Mono para Android* que utiliza *C#* y *.NET*. Por otra parte, el entorno *App*

Inventor permite el desarrollo de aplicaciones de manera visual sin escribir una línea de código aunque obviamente no es posible crear aplicaciones complejas.

Dentro de los entornos de desarrollo disponibles vamos a utilizar el *SDK* de *Android* que consiste en el Eclipse IDE (*Integrated Development Enviroment*) junto a un complemento ADT (*Android Development Tools Plugin*). Integra un depurador de código fuente, las bibliotecas necesarias, ejemplos y tutoriales y un simulador de *Smartphone* permitiendo a su vez la prueba en uno real. Cabe destacar que desde el 8 de diciembre de 2014[34], *Google* liberó la versión *Android Studio 1.0* dejando atrás la expuesta anteriormente y ha pasado a recomendarlo como *IDE*. Sus principales ventajas son dar soporte para programar aplicaciones para el sistema operativo *Android Wear*, la herramienta *Lint* que detecta incompatibilidades entre arquitecturas, usabilidad y compatibilidad entre versiones, la integración de la herramienta *Gradle* para la gestión y construcción de proyectos realizando empaquetado, compilación, etc…, una nueva interfaz para el desarrollo y un nuevo diseño del editor. Además posibilita la vista previa en dispositivos con diferentes características técnicas, versiones de *Android* y resoluciones y con esto, la creación de múltiples variantes de la misma *APK*.

Por otra parte, son necesarios, además de las librerías, dos proyectos uno en *Android* y otro en Java, con el objetivo de simplificar el código cuando se realicen aplicaciones que se van a realizar en código Java y van a ejecutarse en un *PC*. La figura 25 describe cómo es el proceso de comunicación entre las librerías, realizándose de manera transparente para el usuario.

[34] http://domadis.com/2014/12/08/google-libera-android-studio-1-0-ahora-es-mucho-mas-facil-crear-tus-aplicaciones/

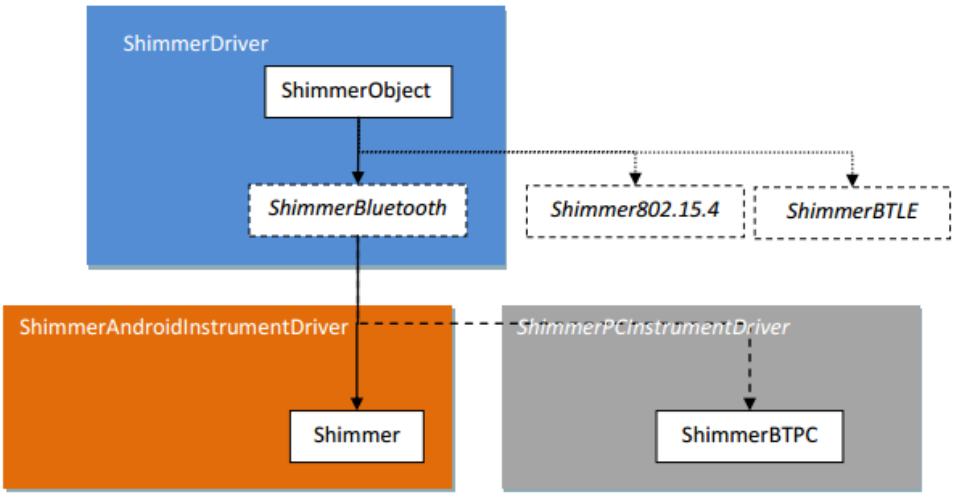


Figura 25 Estructura librería[35]

El primero proyecto llamado *ShimmerAndroidInstrumentDriver* consta de 3 paquetes:

1) *Com.shimmerreseach.android:* Se ocupa de la API de *Android Bluetooth* proporcionando a la aplicación funciones para poderse comunicar con el dispositivo *Shimmer.* Permite una mayor facilidad y flexibilidad en el uso del código para la interfaz *Bluetooth.* Es vital porque sin ella no sería posible la interacción desde nuestra aplicación con el sensor. El tipo de comunicación con el dispositivo emula una conexión *Serie Port Profile (SPP)* de cable sobre una línea inalámbrica estableciendo una distinta para cada sensor conectado simultáneamente.

   Para conseguirlo, cada uno de los sensores tiene que estar representado mediante un objeto *Shimmer.* Cabe destacar que en la mayoría de los casos cuando se utiliza un método de la librería incluye un acuse de recibo o *ACK* por parte del dispositivo Los métodos destacados son *connect, StartStreaming, StopStreaming, writeEnabledSensors, inquiry (*método para saber el estado actual del dispositivo*), writeSamplingRate* y *writeAccelRange.*

[35] "*Android Library*" extraída el 10 de agosto de 2016 desde: www.shimmersensing.com

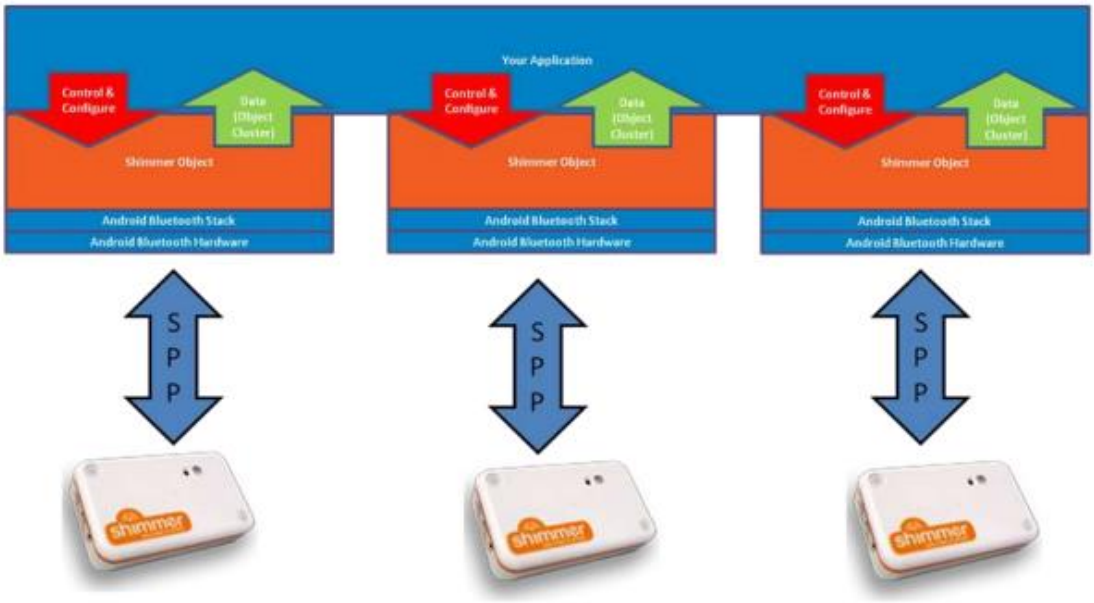


Figura 26 Comunicación de cada uno de los sensores con nuestra aplicación[36]

Todos los datos e instrucciones anteriores son gestionados por la clase *Handler* proporcionada en la librería. Su tarea es la de gestionar una cola para comunicar los datos generados en los *threads* con el *thread* principal. Es muy importante cuando queremos tener los sensores sincronizados transmitiendo y deteniéndose al mismo tiempo de tal manera que cada hilo gestione con el principal que su estado es conectado y están inicializados.

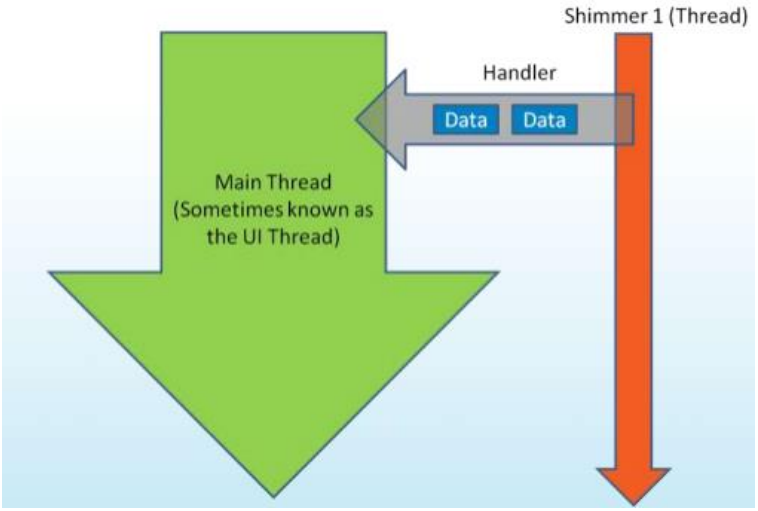


Figura 27 Comunicación datos generados en *threads* adyacentes con el principal

[36] "*Handler"* extraída el 10 de agosto de 2016 desde: www.shimmersensing.com

2) *Com.shimmerreseach.tools:* Contiene ejemplos para la conexión de un sensor con la aplicación

3) *Pl.flex_it.androidplot:* Contiene ejemplos para la representación de gráficas incluyendo constructores para utilizarlos en los proyectos.

El segundo proyecto, con el nombre *ShimmerDriver,* consta también de 3 paquetes:

1) *Com.shimmerresearch.algorithms:* Está formada por la clase necesaria para realizar el filtrado de las señales recibidas y otras dos dedicadas a la orientación del dispositivo basado en un algoritmo de gradiente descendiente.

2) *Com.shimmrresearch.bluetooth:* Esta clase permite la conexión de la aplicación con el *bluetooth* del *Smartphone.*

3) *Com.shimmerresearch.driver:* Está formado por 4 clases. La primera *ShimmerObject* es la principal y más genérica puesto que permite a cualquier sensor estar definido. Dependiendo del estándar de comunicación elegido se podrá extender a otras clases como es nuestro caso, por elegir el *bluetooth.* La segunda *ShimmerConfiguration* define la lista de sensores compatibles, el canal en el que opera cada uno y el rango de operación que se puede seleccionar para el magnetómetro, acelerómetro y sensor galvánico de la piel o *GSR.* La tercera y la cuarta son *FormatCluster y ObjectCluster,* definen la estructura de los datos procedentes de los sensores guardando propiedades como el calibrado, las unidades, tipo de dato etc… y son modificables a través de los mapas de bits mediante su código correspondiente. La importancia de mantener una estructura de datos reside en la escalabilidad de los dispositivos pudiendo añadir funcionalidades de manera sencilla en un espacio de tiempo breve como pueden ser la orientación o la aceleración lineal. Además, una vez establecido el acceso a los datos, será bastante sencillo y cómodo para los programadores.

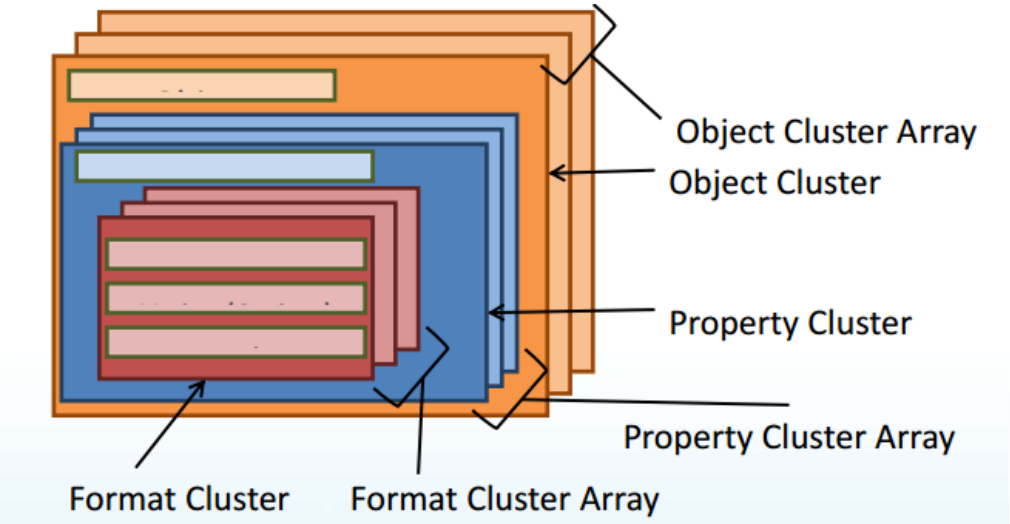


Figura 28 Estructura de los datos de acuerdo con la librería

En nuestro proyecto un *ObjectCluster* estaría constituido por las señales procedentes de cada sensor y el *PropertyCluster* sería el conjunto de las señales separadas de manera individual. Por ejemplo, en el caso del acelerómetro el *ObjecCluster* sería el propio acelerómetro y el *PropertyCluster* las señales de los ejes X, Y y Z.

V. DESARROLLO DE UNA APLICACIÓN PARA LA ADQUISICIÓN Y PROCESAMIENTO DE PARÁMETROS FISIOLÓGICOS EXTRAÍDOS DE SENSORES INALÁMBRICOS MEDIANTE UN DISPOSITIVO ANDROID.

V.1 Introducción

La aplicación va a servir como soporte para la adquisición y procesamiento de las señales procedentes de los sensores SHIMMER mencionados en el punto anterior. Su diseño está enfocado al procesado de las señales en tiempo real. Sin embargo, limitaciones técnicas como el microprocesador propio de un teléfono móvil van a hacer que cierta parte de la información vaya a ser tratada de manera offline tanto en la aplicación como en otro tipo de *software* que dispone de herramientas más precisas para el procesado de señales.

Está organizada en diferentes interfaces llamadas vistas (*View o ViewGroup)*, cada una de las cuales desempeñará una función diferente en la aplicación. Estas vistas estarán constituidas por *layouts* que integran *TextViews, ImageView, EditText,* etc…A continuación describiremos las vistas de la aplicación explicando todos los elementos que las conforman.

V.1.1 Bluetooth Settings

Es la primera actividad de la aplicación dedicada principalmente a la activación y desactivación del *Bluetooth*, exploración y selección de los sensores para capturar la información en las siguientes actividades y acceso a un menú para ver cuáles son todos los ficheros que se encuentran almacenados visualizando a qué usuario pertenecen en caso de haberlo introducido. Consta de un *ActionBar* donde figura el título de la actividad en la que se encuentra el usuario (se establece en el *AndroidManifest.xml*), un menú con la opción *ManageFiles* para gestionar los ficheros almacenados en otras sesiones por los usuarios.

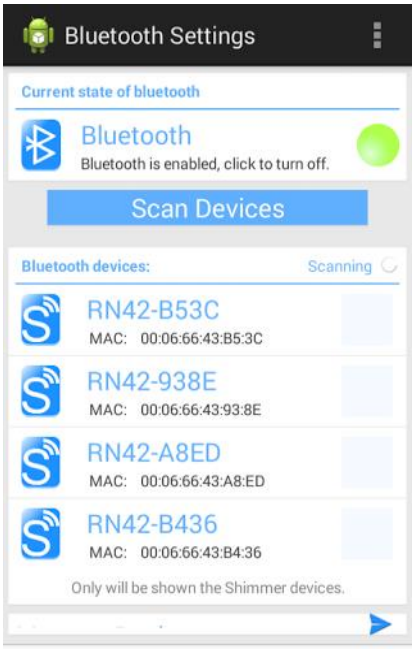


Figura 29 Actividad *Bluetooth Settings*

Por otra parte, la primera sección va a indicar el estado en el que se encuentra el *Bluetooth* del dispositivo, pudiendo estar en 3 estados diferentes. En función del mismo y de acuerdo con las necesidades del usuario, el *TextView* será diferente así como el color del punto situado a la derecha formado por *ImageView* como se aprecia en las siguientes líneas de código. La transición entre los estados está gestionada por el método *BroadcastReceive*r encargado de la recepción de los eventos y ajustando el estado haciendo una llamada *SetBtState().*

```
}else{
    ScanningL.setVisibility(View.INVISIBLE);
    Devices.clear();
    DevicesAdapter.notifyDataSetChanged();
    switch(BtAdapter.getState()){
    case BluetoothAdapter.STATE_ON:     // RECIEN ENCENDIDO
        BtDiscoveringL.performClick();  // Se simula el click para buscar dispositivos.
        DescriptionStateBt.setText("Bluetooth is enabled, click to turn off."); // Se cambia la descripcion del es
        StateBtText.setImageResource(R.drawable.green50);   // Se cambia el color de la imagen de estado.
        BtDiscoveringL.setVisibility(View.VISIBLE); // Se muestra el boton de buscar dispositivos.
        BtSettingsL.setClickable(true);     // Se permite cambiar el estado de bluetooth (encender/apagar).
        break;
    case BluetoothAdapter.STATE_OFF:    // RECIEN APAGADO
        DescriptionStateBt.setText("Bluetooth is disabled, click to turn on.");
        StateBtText.setImageResource(R.drawable.gray50);
        BtDiscoveringL.setVisibility(View.INVISIBLE);
        BtSettingsL.setClickable(true);
        break;
    case BluetoothAdapter.STATE_TURNING_ON:     // Activando Bluetooth
        DescriptionStateBt.setText("Turning on Bluetooth...");
        StateBtText.setImageResource(R.drawable.orange50);
        BtSettingsL.setClickable(false);    // No se permite cambiar el estado de bluetooth, esta en transicion.
        break;
    case BluetoothAdapter.STATE_TURNING_OFF:    // Desactivando Bluetooth
        DescriptionStateBt.setText("Turning off Bluetooth...");
        StateBtText.setImageResource(R.drawable.orange50);
        BtDiscoveringL.setVisibility(View.INVISIBLE);
        BtSettingsL.setClickable(false);
        break;
```

En caso de no existir *bluetooth* o no soportarlo se notificaría al usuario.

```
if (BtAdapter == null){ // Si el dispositivo no soporta bluetooth.dodgerblue

    DescriptionStateBt.setText("Bluetooth NOT supported");
    StateBtText.setImageResource(R.drawable.orange50);
```

Los estados son los siguientes:

1) Apagado: Indicará que se encuentra desactivado y el color del punto permanecerá en color gris. En caso de desactivar el *Bluetooth* desde el terminal y no desde la aplicación también cambiará su estado.

Figura 30 *Bluetooth* apagado

2) Encendido: Indicará que se encuentra encendido y el color del punto permanecerá en color verde. Al igual que en el caso anterior si el *Bluetooth* es activado desde el terminal y no desde la aplicación su estado cambiará.

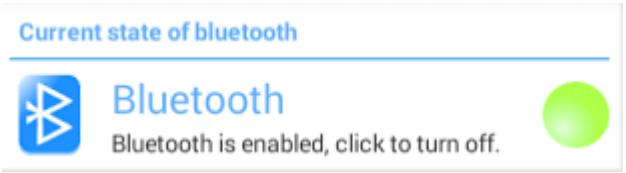


Figura 31 *Bluetooth* encendido

3) Transición encendido a apagado: Indicará que el *Bluetooth* del dispositivo está desactivándose.

Figura 32 *Bluetooth* apagándose

4) Transición apagado a encendido: Indicará que el *Bluetooth* del dispositivo está activándose.

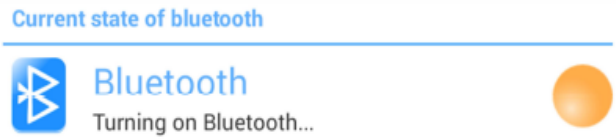


Figura 33 *Bluetooth* encendiéndose

La segunda sección consiste en un botón para la búsqueda de todos aquellos dispositivos que se encuentren dentro del rango de cobertura y no tengan estado oculto. Cabe destacar que solamente será visible cuando el *Bluetooth* se encuentre activado debido a que dicha funcionalidad sería posible si se encontrase apagado.

```
else if (action.equals(BluetoothAdapter.ACTION_STATE_CHANGED)) {
    SetBtState();
}
else if (action.equals(BluetoothAdapter.ACTION_DISCOVERY_STARTED)) {
    ScanningL.setVisibility(View.VISIBLE);  // scanning y el progessbar que indican estado de busqueda.
}
else if (action.equals(BluetoothAdapter.ACTION_DISCOVERY_FINISHED)) {
    ScanningL.setVisibility(View.INVISIBLE);
    if(!isout)print(Devices.size() + " Devices found");
}
```

Una vez pulsado el botón permanecerá visible un *widget* del tipo *ProgressBar* en forma de círculo giratorio indicando que se está llevando a cabo la acción y se irán añadiendo al *ListView* todos aquellos encontrados siempre y cuando su MAC contenga los 3 primeros pares de Shimmer como vemos en el código. El estado durante la búsqueda será de “ En proceso”.

```
if(BluetoothDevice.ACTION_FOUND.equals(action)) {
    BluetoothDevice device = intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
    if(device.getAddress().contains("00:06:66") ||device.getAddress().contains("B4:82:FE")  ){
        boolean added = false;
        //V. Realizamos un bucle, mientras se siguen encontrando dispositivos, mostramos su MAC y nombre
        for(BtDevicesClass Device: Devices){
            if(device.getAddress().equals(Device.getMAC())){
                added = true;
                Device.setName(device.getName());

            }
        }
        if(!added){
    //Permite añadir los dispostivos al array declarado, siguiendo la estructura del metodo BtDevicesClass
    //de la clase con el mismo nombre
            Devices.add(new BtDevicesClass(device.getName(), device.getAddress(), false));
        }
        DevicesAdapter.notifyDataSetChanged();
    }else
        print(device.getAddress()+"\nIt's not a Shimmer Device.");
```

En caso contrario, se lanzará un *toast* indicando la *MAC* del dispositivo encontrado junto un mensaje que nos señala que no es un dispositivo de la gama de los sensores.

Figura 34 Notificación de que no es una MAC perteneciente a *Shimmer*

Una vez finalizada la búsqueda aparecerá una notificación indicando el número de sensores encontrados y podremos seleccionar aquellos sensores que queramos utilizar a través de un *checkbox*. Si no seleccionamos ninguno y pulsamos el botón *Manage* Devices la aplicación nos indicará que debemos seleccionar al menos uno.

```
private OnItemClickListener OnItemClickListenerToSelectDevice = new OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> myAdapter, View myView, int position,
            long id) {
        SelectedItem = (BtDevicesClass) DevicesListView.getItemAtPosition(position);
        mySharedPrefs aux = new mySharedPrefs();
        // Se cambia el estado al pulsarse y se borra/guarda el dispositivo.
        if(SelectedItem.getSelected()){
            Devices.get(position).setSelected(false); // Se cambia el estado al pulsarse.
            aux.removeDevice(getApplicationContext(), SelectedItem.getName());
            numSelected--;
        }else{
            Devices.get(position).setSelected(true); // Se cambia el estado al pulsarse.
            aux.setDevice(getApplicationContext(), SelectedItem.getName(), SelectedItem.getMAC());
            aux.setDefaultConfigFromDevice(getApplicationContext(), SelectedItem.getName(), SelectedItem.getMAC())
            numSelected++;
        }
        DevicesAdapter.notifyDataSetChanged(); // Se notifica el cambio.

    }
};
```

Una vez pulsado dicho botón aparecerá una ventana emergente pidiendo el nombre de usuario (como se indica en la figura 35), el cual deberá contener al menos 4 caracteres. Nos va a permitir clasificar los ficheros por usuarios. Sin embargo, también se puede dejar el campo vacío añadiéndose los ficheros con las muestras obtenidas en la sesión a la carpeta de “otros”.

```
private  OnClickListener OnClickListenerToEnter = new OnClickListener() {
    @Override
    public void onClick(View v) {
        if(BtAdapter != null){
            if(numSelected>0 & BtAdapter.getState()==BluetoothAdapter.STATE_ON){
                final Dialog dialog = new Dialog(MainActivity.this);
                dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
                dialog.setContentView(R.layout.user);
                final EditText username = (EditText) dialog.findViewById(R.id.Username);
                Button accept = (Button) dialog.findViewById(R.id.Accept);
                accept.setOnClickListener( new View.OnClickListener() {
                public void onClick(View v) {
                    if(username.getText().toString().length()>3){
                        NextActivity("/"+username.getText().toString());
                    }else{
                        print("Minimum lenght is 4 characters");
                    }
                }});
                Button omit = (Button) dialog.findViewById(R.id.Omit);
                omit.setOnClickListener( new View.OnClickListener() {
                    public void onClick(View v) {
                        dialog.dismiss();
                        NextActivity("");
                }});
                dialog.show();

            }else
                print("You must select at least one device.");
```

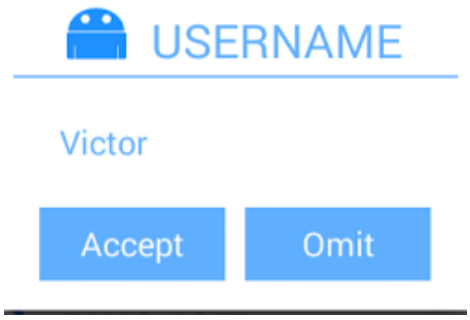


Figura 35 Campo para introducir el nombre de usuario (opcional)

La segunda actividad de la aplicación está relacionada con el manejo de los ficheros almacenados que contienen las señales que se han obtenido en las diferentes sesiones con los usuarios. Los ficheros aparecen agrupados de acuerdo con los propietarios de cada uno y la fecha en la que fueron creados. Para cada uno de ellos, se pueden ver los detalles acerca de las señales que contiene, el tamaño y la fecha de creación. Los ficheros que almacenen las señales procedentes del sensor electrocardiograma se van a poder procesar de manera *off-line*, calculando la tasa de variación entre los latidos y calcular los ratios descritos en la sección V.2.1 con la opción de ser representadas en una gráfica de tal manera que se aprecie de manera visual cuál es la tendencia de los ratios calculados durante dicha sesión y poder describir cuál ha sido el estado del conductor durante la sesión analizada. Otra funcionalidad añadida es la de poder compartir los ficheros a través de la nube, mandarlos por correo

electrónico y por todas aquellas aplicaciones de la misma índole instaladas en nuestro dispositivo.

La estructura jerarquizada de la actividad se adjunta en la figura 36.

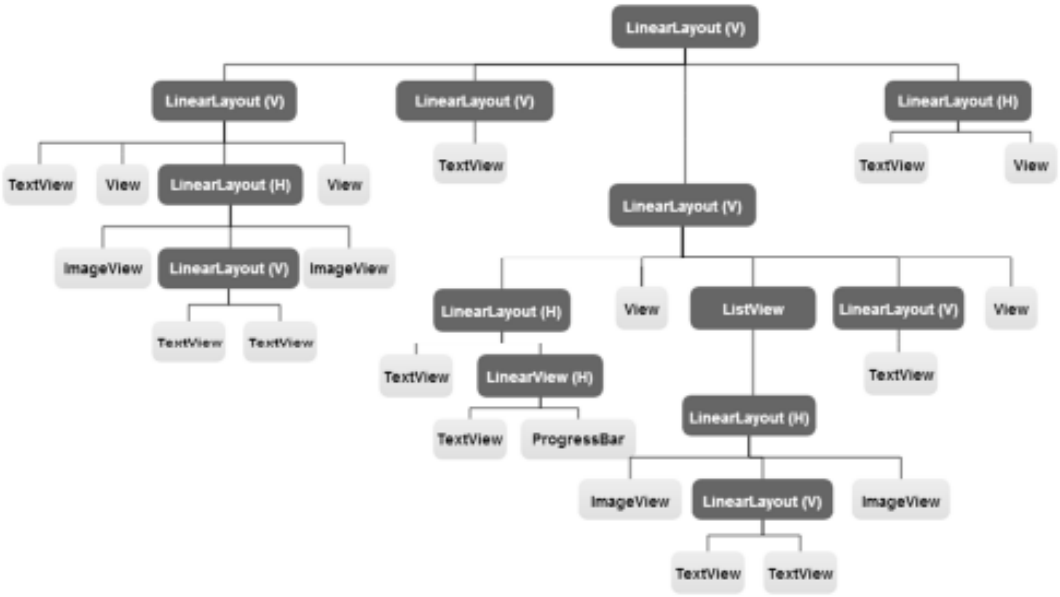


Figura 36 Estructura de las vistas de la actividad *Bluetooth Settings* [37]

V.1.2 Files Management

La actividad *Files Management* se encarga de la gestión de los ficheros almacenados por los usuarios que contienen las señales procedentes de los sensores. Consta de un *ActionBar* que indica el nombre de la actividad en la que nos encontramos y un menú con tres opciones, "*Share all"* para compartir todos los ficheros existentes, *"Delete all"* para borrarlos todos y *"Go home"* para regresar a la actividad *"Bluetooth Settings"*.

[37] Figura obtenida de: Khadmaoui, Amine(2014): "*Desarrollo de una aplicación para el control de sensores fisiológicos mediante un dispositivo Android"* [Proyecto fin de carrera].Valladolid: Universidad de Valladolid, Escuela Técnica Superior de Ingenieros de Telecomunicación .

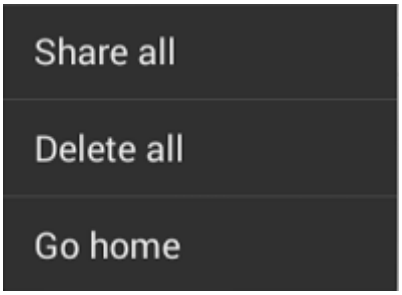


Figura 37 Opciones del menú de la actividad *Files Managment*

La vista principal consta de un *LinearLayout* principal donde haciendo una llamada al método *onCreate()* se invoca la función *CreateListView()* para que muestre tanto el propietario de los ficheros como el nombre, fecha y hora de su almacenamiento y tamaño en *KB* de cada uno de ellos accediendo a la carpeta situada en la memoria de almacenamiento *AK.Shimmer.*

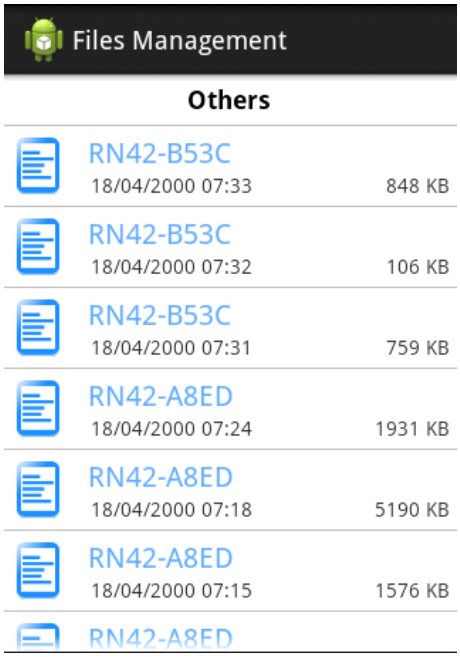


Figura 38 Vista de los ficheros en la memoria de almacenamiento

Pulsando con un toque en cada uno de ellos se desplegará una vista con las opciones para compartir el fichero empleando las aplicaciones de este tipo instaladas en nuestro dispositivo.

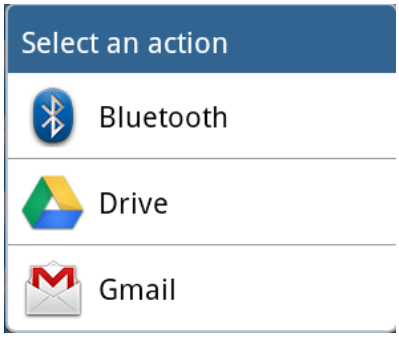


Figura 39 Opciones para compartir el fichero

Manteniendo pulsado cada fichero, se actuará sobre el constructor *FilesListViewOnLongClickListener* activando la función *onItemLongClick()* y se desplegará un menú con las opciones:

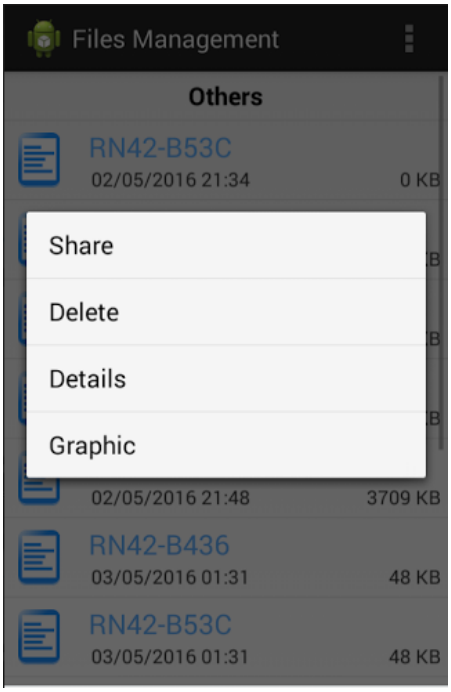


Figura 40 Opciones del menú cuando mantenemos pulsado sobre un fichero

1) *Share* para compartir el fichero a través de la nube, enviarlo por correo electrónico, etc… Se invoca el mismo método que en el párrafo anterior.

2) *Delete* para borrar el fichero. Se realiza una llamada sobre el método *delete()*, al igual que en la opción menú sin embargo en esta ocasión el parámetro enviado es la posición del elemento que se desea

eliminar. Recordemos que en el otro caso se recorrería todo el *array* eliminando todos los archivos uno a uno.

```
private void delete(int position){
    String filePath;
    // Si se tratar de borras todos los ficheros, se recorren todos uno a uno.
    if(position==-100){
        for(int i=0;i<Files.size();i++){
            filePath = Files.get(i).getPath();
            if(!filePath.equals("")){       // Sólo se borra si se trata de un
                if(!(new File(filePath)).delete()){
                    print("An error has occurred");
                }
            }
        }
        CreateListView();   // Se refresca la vista.
    }
    else{   // Cuando se selecciona la opción de borrar un solo fichero.
        filePath = Files.get(position).getPath();
        if(!(new File(filePath)).delete()){
            print("An error has occurred");
        }else{
            CreateListView();   // Se refresca la vista.
        }
    }
}
```

3) *Details* refresca la vista actualizada, "get_*details()"* donde desplegará una vista nueva donde podremos ver el contenido del fichero siendo fijas a todos ellos la señal 1 que hace referencia al acelerómetro y la señal 3 que hace referencia al sello temporal. La señal dos será la procedente del sensor empleado en cada momento. Además figuran detalles como la fecha y hora de creación, tamaño y los ratios en el caso de ser una señal ECG que veremos en profundidad en la sección V.2.1.

4) *Graphic* donde aparecerán representados los ratios de la señal ECG. En primer lugar se crea un *layout* nuevo que contiene el código en lenguaje XML para definir la estructura visual de la interfaz y la propia gráfica donde se van a representar los ratios.

```
    <com.androidplot.xy.XYPlot
android:id="@+id/mySimpleXYPlot"
android:layout_width="fill_parent"
android:layout_height="300px"
android:layout_marginTop="2px"
android:layout_marginLeft="2px"
android:layout_marginRight="2px"
title="HRV"/>
```

En segundo lugar se desarrolla la actividad, importando las librerías *com.androidplot.series.XYSeries, com.androidplot.xy.LineAndPointFormatte, com.androidplot.xy.SimpleXYSeries, com.androidplot.xy.XYPlot* y *android.graphics.Color* junto a la declaración del contructor perteneciente a XYPlot e inicializando el objeto procedente del *layout.*

```
mySimpleXYPlot = (XYPlot) findViewById(R.id.mySimpleXYPlot);
```

A continuación se procede a la recepción de los datos procedentes de la actividad anterior llamada *ManageFiles.* Para ello en esa clase se creará un *Intent* que describe de una manera abstracta la operación que se va a poner en marcha, como va a ser lanzar una actividad nueva. A continuación, se establecen los parámetros que se le van a atribuir mediante la propiedad *putExtra(String name, double[] value)* y se asigna a la clase la actividad desde donde se inicia y la que vamos a poner en ejecución empleando la función *setClass(actividad desde donde se ejecuta, actividad lanzada).* Finalmente se inicia la actividad *startActivity(Objeto Intent)* pasando de una a otra la información que se ha deseado.

```
case 3:
    Intent intent = new Intent();
    intent.putExtra("Objeto", Ratio2);
    intent.setClass(ManageFiles.this, Actividad.class);
    startActivity(intent);
    finish();
    break;
```

En la otra actividad (Actividad.java), se recoge la información utilizando la función *getIntent().getExtras()*, devolviendo el contenido del *Intent().* En nuestro caso, se ha transferido la información mediante el par clave/valor por lo que en la nueva actividad podremos acceder a dicho contenido haciendo únicamente referencia a la clave.

```
public class Actividad extends Activity {

      private XYPlot mySimpleXYPlot;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.actividad);

        // Inicializamos el objeto XYPlot búscandolo desde el layout:
        mySimpleXYPlot = (XYPlot) findViewById(R.id.mySimpleXYPlot);

        double[] RatiosActividad;
        Bundle extras = getIntent().getExtras();
        RatiosActividad = extras.getDoubleArray("Objeto");
```

Una vez realizada esta tarea se procede a la representación gráfica de los datos, añadiendo el *array* de datos a la línea número uno y estableciendo los colores de la línea y de los puntos. Finalmente, se añade al panel y se aprecia la representación por puntos.

```
   // Añadimos Línea Número UNO:
   XYSeries series1 = new SimpleXYSeries(
           Arrays.asList(series1Numbers),  // Array de datos
           SimpleXYSeries.ArrayFormat.Y_VALS_ONLY, // Sólo valores verticales
           "Series1"); // Nombre de la primera serie
```

```
 // Modificamos los colores de la primera serie
 LineAndPointFormatter series1Format = new LineAndPointFormatter(
         Color.rgb(0, 200, 0),                   // Color de la línea
         Color.rgb(0, 100, 0),                   // Color del punto
         null);              // Relleno


    // Una vez definida la serie (datos y estilo), la añadimos al panel
    mySimpleXYPlot.addSeries(series1, series1Format);
```

El último paso consiste en añadir la actividad al *AndroidManifest.xml* para que tenga los permisos necesarios para poderse ejecutar.

```
<activity
    android:name="com.ak.shimmer.Actividad"
    android:label="Actividad"
    android:configChanges="keyboardHidden|orientation|screenSize"
</activity>
```

V.1.3 Devices Managment

La actividad *Devices Managment* es la encargada de gestionar el estado en el que se encuentran los sensores seleccionados por el usuario en la actividad *Bluetooth Settings*, recibir las tramas cuando estén transmitiendo, almacenar los datos en los ficheros, así como la representación de gráficas en tiempo real y la información de las mismas en diferentes *fragments*.

Consta de un *ActionBar* indicando el nombre de la actividad y un menú con tres opciones, que son, *Start all*, para que todos los dispositivos que se encuentran seleccionados pasen al estado *streaming* de manera simultánea, *Disconnect all*, que desconectaría todos los dispositivos, finalizando la transmisión en caso de encontrarse emitiendo y *Go home*, para volver a la primera actividad. A continuación, se muestran las opciones disponibles en el menú con su código fuente.

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch(item.getItemId()){
        case R.id.start_all:
            synchronizing = true;
            SyncDialog = ProgressDialog.show(this, "","Wait, please...", true);
            SyncStarted = System.currentTimeMillis();
            //SyncDialog.setCancelable(true);
            //SyncDialog.setCanceledOnTouchOutside(false);

            SyncDevices();break;
        case R.id.disconnect_all:
            for(int i=0; i<Devices.size();i++){     if(ShimmerDev[i] != null)    ShimmerDev[i].stop(); }
            break;
        case R.id.go_home:
            startActivity(new Intent(ManageDevices.this, MainActivity.class));finish();
            break;
        default:
            return super.onOptionsItemSelected(item);
    }
    return true;
}
```

Está estructurada mediante un *ViewPager,* vista que permite pasar los elementos como si fuesen las páginas de un libro hacia adelante o hacia atrás. Estas páginas son denominadas *fragments* y están en distintos ficheros XML, los cuales se incorporan al *ViewPager.* Es importante destacar que el ciclo de vida del fragmento depende directamente de la actividad y no de él mismo, siendo independientes entre ellos. La ventaja que nos ofrecen los fragmentos es la capacidad de modificar el aspecto y funcionalidades de la actividad en tiempo real. Los *tabs* nos señalan el *fragment* en el que nos encontramos.

```
FragmentManager fm = getSupportFragmentManager();// Se obtiene una referencia al fragmentManager.
MyFragmentPagerAdapter fragmentPagerAdapter = new MyFragmentPagerAdapter(fm);
mPager.setAdapter(fragmentPagerAdapter);
mActionbar.setDisplayShowTitleEnabled(true);
//V. El TabListener permitirá la navegación entre las diferentes pestañas
ActionBar.TabListener tabListener = new ActionBar.TabListener()
{
    @Override
    public void onTabUnselected(Tab tab, FragmentTransaction ft)
    {
    }
    @Override
    public void onTabSelected(Tab tab, FragmentTransaction ft)
    {
        mPager.setCurrentItem(tab.getPosition());
    }
    @Override
    public void onTabReselected(Tab tab, FragmentTransaction ft)
    {
    }
};
// SE AÑADEN LAS DOS PESTAÑAS, SENSORES Y SEÑALES.
//V.Esta pestaña hace referencia a los sensores que tenemos disponibles.
mActionbar.addTab(mActionbar.newTab().setText("SENSORS").setTabListener(tabListener).setTag("SENSORS"));
//V.Esta pestaña hace referencia a las señales representadas a partir de las muestras obtenidas por los sensores.
mActionbar.addTab(mActionbar.newTab().setText("SIGNALS").setTabListener(tabListener).setTag("SIGNALS"));
mActionbar.addTab(mActionbar.newTab().setText("ECG").setTabListener(tabListener).setTag("ECG"));
mActionbar.addTab(mActionbar.newTab().setText("GSR").setTabListener(tabListener).setTag("GSR"));
```

El primer fragmento conforma la vista de los sensores seleccionados en la primera actividad. Una vez constituida la vista y la estructura de la interfaz, se dispone a realizar la obtención de las variables que contienen los dispositivos seleccionados y el nombre de usuario en caso de haberse introducido. Hay que tener en cuenta que estos parámetros se pasan a la actividad y no al fragmento por lo que se ha de tener cautela y no crear la vista del *fragment* en este punto debido a que puede que no haya finalizado su construcción y tengamos un error (*NullPointerException*).

```
Bundle extras = getIntent().getExtras(); //V.Recogemos los parámetros de la actividad anterior.
username = extras.getString("Username");
extras.remove("Username");
//V.Si se obtienen los parametros de la otra actividad, pasamos a añadiros para mostrarlos en la nueva actividad.
//V. Se descarta la posibilidad que no encuentre ninguno porque si no hubiese sido imposible llegar a este punto.
if (extras != null)
{
    //V. Hacemos una serie de iteraciones para obtener los valores que pasamos a esta actividad
    //V. desde la anterior "NextActivity". Nombre del dispositivo y obtener la MAC del mismo.
    Set<String> keys = extras.keySet();
    Iterator<String> it = keys.iterator();
    while (it.hasNext()) {
        String key = it.next();
        //Probar en este método poniendo en ambos String y en ambos toString.
        Devices.add( new ShDevicesClass(key, extras.get(key).toString(), 0));
    }
}
// SE INCLUYE EL ARRAY AL ADAPTAR, ESTE SERÁ EL RESPONSABLE DE CONFIGURA EL LISTVIEW.
DevicesAdapter = new ManageDevicesAdapter(this, Devices){};
```

Una vez incorporada la vista al *ViewPager,* se crea en un proceso independiente al de la actividad con la vista del fragmento seleccionado por defecto para ser el primero que es el número uno. Esta tarea es llevada a cabo por el método *OnViewCreatedListener().*

```
case 0: // Se ha creado el Fragment del listview (SENSORS)
    DevicesListView = (ListView) findViewById(R.id.DevicesListView);
    DevicesListView.setAdapter(DevicesAdapter);
    DevicesListView.setOnItemClickListener(DeviceOnItemClickListener);
    DevicesListView.setOnItemLongClickListener(DeviceOnLongClickListener);
    Button boton = (Button) findViewById(R.id.BotonDesconectar);
    boton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            //Creamos el Intent
            for(int i=0; i<Devices.size();i++){     if(ShimmerDev[i] != null)    ShimmerDev[i].stop(); }
        }
    });

    Button boton2 = (Button) findViewById(R.id.BotonTransmitir);
    boton2.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            synchronizing = true;
            SyncStarted = System.currentTimeMillis();
            SyncDevices2();
        }
    });
    break;
```

Como novedad, se han incorporado dos botones en la interfaz de usuario, uno para transmitir únicamente aquellos dispositivos seleccionados de manera sincronizada y otro para desconectar todos al mismo tiempo evitando de esta manera una deriva temporal, pudiendo capturar señales simultáneamente en el mismo intervalo temporal. Manteniendo pulsado cada dispositivo, aparecerá una ventana emergente que nos permitirá seleccionar el módulo, la frecuencia de muestreo y si deseamos que transmita o no.

```
private OnClickListener LongClickListener = new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int index) {
        dialog.cancel();
        switch(index){
        case 0: // Al pulsar la opcion de cambiar el modulo.
            String[] Modulearray = getResources().getStringArray(R.array.module);
            String Moduleselected = myPrefs.getMODULE();
            int Moduleposition = 0;
            for(int j=0;j<Modulearray.length;j++){  // Se busca en el vector con cual coincide
                if(Modulearray[j].equalsIgnoreCase(Moduleselected))
                    Moduleposition = j;             // Se obtiene la posicion para indicarla en el listview.
            };
            AlertDialog.Builder Modulealert = new AlertDialog.Builder(ManageDevices.this);
            Modulealert.setSingleChoiceItems(R.array.module, Moduleposition, ModuleListener);
            Modulealert.show();
            break;

        ------,
 case 1:     // Al pulsar la opcion de cambiar la f de muestreo.
     String[] SRarray = getResources().getStringArray(R.array.sampling_rates);
     String SRselected = myPrefs.getSAMPLINGRATE();  // Se obtiene la opcion ya guardada.
     int SRposition = 0;
     for(int j=0;j<SRarray.length;j++){  // Se busca en el vector con cual coincide
         if(SRarray[j].equalsIgnoreCase(SRselected))
             SRposition = j;             // Se obtiene la posicion para indicarla en el listview.
     };
     AlertDialog.Builder SRalert = new AlertDialog.Builder(ManageDevices.this);
     SRalert.setSingleChoiceItems(R.array.sampling_rates, SRposition, SRListener);
     SRalert.show();
     break;
 case 2:

     String[] ArraySiNo = getResources().getStringArray(R.array.SiNo);
     int posicionado = DispositivoSeleccionado[SelectedDevice];
     AlertDialog.Builder ArraySiNoAlerta = new AlertDialog.Builder(ManageDevices.this);
     ArraySiNoAlerta.setSingleChoiceItems(R.array.SiNo, posicionado, ArraySiNoListener);
     ArraySiNoAlerta.show();


     break;
```

Como podemos ver en las siguientes líneas de código si seleccionamos el sensor se marcará con un uno y en caso contrario con cero.

```
// Al pulsar una opción del dialogo, se guarda dicha opcion como seleccionada.
private OnClickListener ArraySiNoListener = new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int index) {

        String[] ArraySiNo = getResources().getStringArray(R.array.SiNo);
         Devices.get(SelectedDevice).getName();
         Log.i("Dispositivo Seleccionado",Devices.get(SelectedDevice).getName()+",\n");
        if(index == 0){
            DispositivoSeleccionado[SelectedDevice] = 0 ;

        }else{
            DispositivoSeleccionado[SelectedDevice] = 1;

        }


        dialog.cancel();
    }
};
```

Ya seleccionados todos los dispositivos que se desea que transmitan basta con pulsar el botón “Transmitir” de la interfaz para que se invoque al método *SyncDevices2()* y los pongan en funcionamiento de manera totalmente sincronizada.

```
void SyncDevices2(){
    if(System.currentTimeMillis()-SyncStarted>60000){
        synchronizing = false;
        print("Sorry: Time Out!");
        return; }
    boolean all_connected = true; boolean all_streaming = true;int state;
    for(int i=0; i<Devices.size();i++){
        if( DispositivoSeleccionado[i]==0)
        {
        state = Devices.get(i).getState();
        if(ShimmerDev[i] != null & state == 0 && DispositivoSeleccionado[i]==0){
            Tools.ChangeState(i, 0, 0);}
        if(state < 3) all_connected = false;
    }}
    if(all_connected){
        for(int i=0; i<Devices.size();i++){
            if( DispositivoSeleccionado[i]==0){
            state = Devices.get(i).getState();
            if(ShimmerDev[i] != null & state == 3 && DispositivoSeleccionado[i]==0){ // Fully initialized
                Tools.ChangeState(i, 3, 1); }
            if(state < 5) all_streaming = false;}
    }}
    if(all_connected & all_streaming){
        synchronizing = false;
        //SyncDialog.dismiss();
    }else{
        new CountDownTimer(1000, 1000) {public void onTick(long millisUntilFinished) {}public void onFinish() {SyncDevices2();}
    }
```

Para la representación gráfica en el segundo fragmento de las señales en tiempo real, se va a emplear el modelo *canvas* consistente en una clase para la representación de superficies siguiente un puntero gráfico. En función del número de dispositivos seleccionados creará dos gráficas para cada uno, la primera para el acelerómetro incorporado por defecto en todos ellos y la otra para el módulo, como vemos en las siguientes líneas de código.

```
ScrollView sv = new ScrollView(activity);
sv.setBackgroundColor(getResources().getColor(R.color.whitesmoke));
LinearLayout ll = new LinearLayout(activity);
ll.setLayoutParams(new LayoutParams(LayoutParams.MATCH_PARENT, LayoutParams.MATCH_PARENT));
ll.setOrientation(LinearLayout.VERTICAL);
int numDevices = ManageDevices.Devices.size();  // Segun el numero de los dispositivos se construyen las graficas.
if(numDevices > 0){
    View view = View.inflate(activity, R.layout.plot_fragment_graph, null);
    view.findViewWithTag("DeviceName").setId(R.id.DeviceName0);
    view.findViewWithTag("AccelGraph").setId(R.id.AccelGraph0);
    view.findViewWithTag("ModuleGraph").setId(R.id.ModuleGraph0);
    ll.addView(view);

    if(numDevices > 1){
        view = View.inflate(activity, R.layout.plot_fragment_graph, null);
        view.findViewWithTag("DeviceName").setId(R.id.DeviceName1);
        view.findViewWithTag("AccelGraph").setId(R.id.AccelGraph1);
        view.findViewWithTag("ModuleGraph").setId(R.id.ModuleGraph1);
        ll.addView(view);
        if(numDevices > 2){
            view = View.inflate(activity, R.layout.plot_fragment_graph, null);
            view.findViewWithTag("DeviceName").setId(R.id.DeviceName2);
            view.findViewWithTag("AccelGraph").setId(R.id.AccelGraph2);
            view.findViewWithTag("ModuleGraph").setId(R.id.ModuleGraph2);
            ll.addView(view);
            if(numDevices > 3){
                view = View.inflate(activity, R.layout.plot_fragment_graph, null);
                view.findViewWithTag("DeviceName").setId(R.id.DeviceName3);
                view.findViewWithTag("AccelGraph").setId(R.id.AccelGraph3);
                view.findViewWithTag("ModuleGraph").setId(R.id.ModuleGraph3);
                ll.addView(view);
            }
        }
```

El tercer fragmento consiste en una explicación del procesado realizado sobre la señal del electrocardiograma y el cuarto sobre la señal del sensor galvánico de la piel.

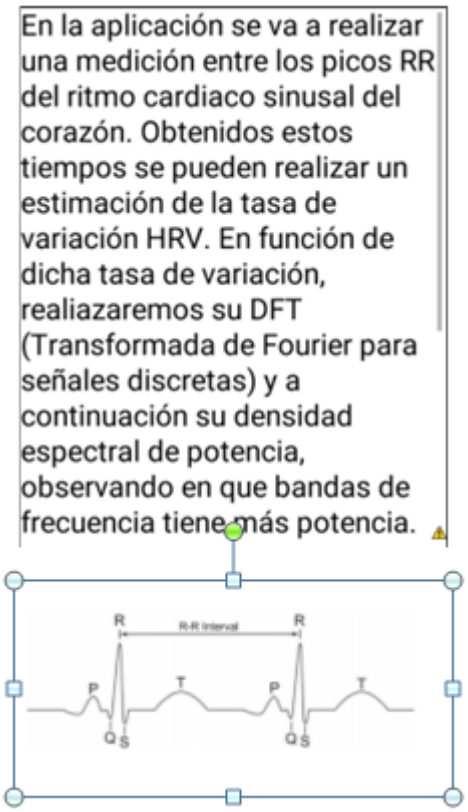


Figura 41 Vista de ECG

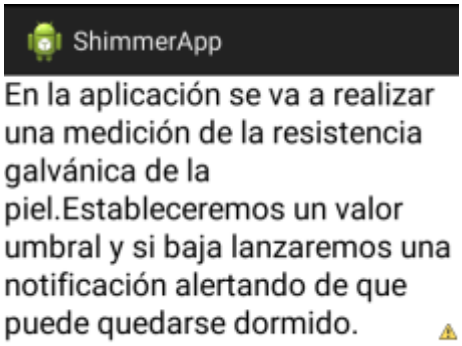


Figura 42 Vista de GSR

V.2 Relación entre la fatiga y las señales recibidas

V.2.1 Relación fatiga y señal ECG

El uso del electrocardiograma en nuestro trabajo ayuda a determinar el estado físico en el que se encuentra el conductor concretamente si está fatigado y/o aproximándose a un estado de somnolencia. Antes de ello vamos a explicar el funcionamiento del corazón y las distintas ondas que obtenemos.

El corazón es un músculo constituido por un tejido denominado miocardio cuya característica principal es su funcionamiento de manera automática en contra de los músculos que constituyen el aparato locomotor.

Internamente se encuentra dividido en cuatro cámaras dos aurículas y dos ventrículos separados por paredes musculares gruesas bien diferenciadas, la izquierda con sangre arterial rica en oxígeno y la derecha con sangre venosa pobre en oxígeno. La sangre arterial proviene de los pulmones donde es oxigenada y lista para ser distribuida por todo el organismo. A su vuelta el corazón la bombea hacia los pulmones para comenzar el ciclo de nuevo. Para realizar estas tareas el corazón se contrae (sístole) y se relaja (diástole) impulsando la sangre por todos los vasos sanguíneos. El ritmo cardiaco, la intensidad y la fuerza de contracción está regulada por el hipotálamo, encargado de elaborar los impulsos nerviosos adecuados para cada estado y necesidades del organismo y las hormonas que actúan sobre el corazón. La actividad eléctrica de cada ciclo latido del corazón es causada en resumidas cuentas por la despolarización de los músculos que lo componen. La fatiga produce cambios en el sistema nervioso central afectando a la actividad cardiaca a través de cambios espontáneos y por ende apreciables en la señal fisiológica.

Las variaciones de la frecuencia cardiaca (*Heart Rate*) nos va a permitir un diagnóstico de la actividad del sistema nervioso autónomo central analizando el cambio en el potencial eléctrico del músculo cardiaco que se ve reflejado con cambios en la señal debido al estrés. Se tomará la frecuencia cardiaca en un intervalo temporal de alrededor de un minuto, en el que se contarán todos los intervalos RR. El objetivo es el cálculo de la densidad espectral de potencia de la variabilidad de la frecuencia cardiaca para relacionarla con el estado físico de los sujetos.

Para ello, en primer lugar realizaremos el cálculo de la transformada de *Fourier*, y a continuación, calcularemos su densidad espectral donde las frecuencias serán divididas en baja (LF, 0,01-0,08), media (MF, 0,08-0,15) y alta (HF, 0,15-0,5) guardando las frecuencias superiores una relación de simetría.

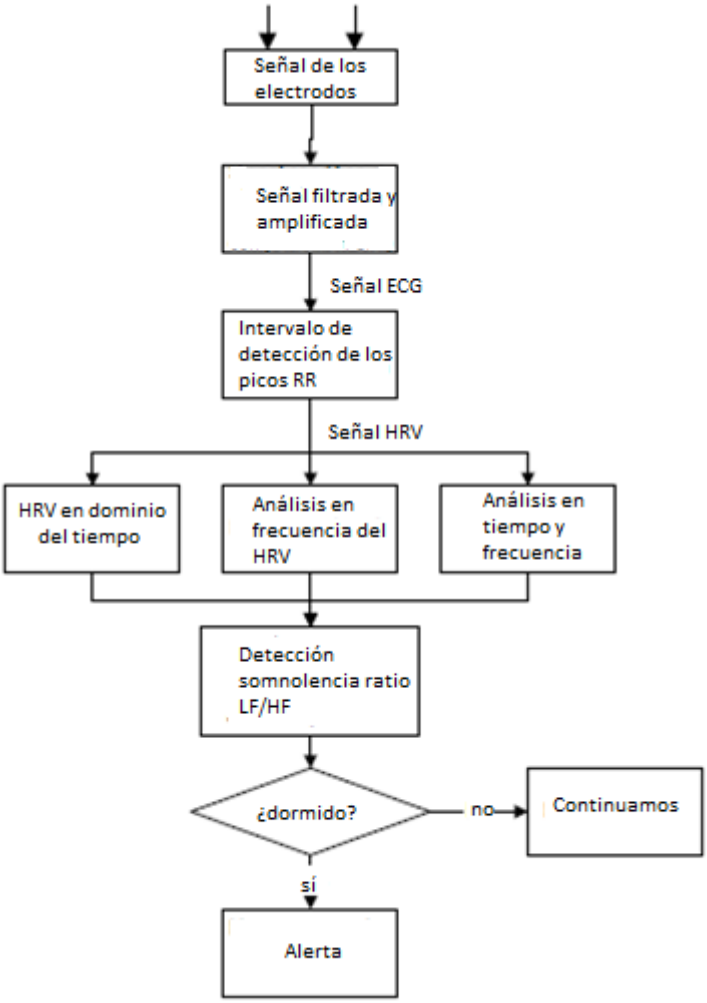


Figura 43 Diagrama de flujo del algoritmo empleado[38]

El diagrama de bloques a seguir para su desarrollo consistirá en un filtro paso bajo con el objetivo de eliminar las frecuencias contaminantes situando el corte en 25Hz, seguido de un filtro paso alto de 1 Hz, que se puede ampliar hasta 5 Hz. Su desarrollo en el lenguaje de programación *Matlab* es el siguiente:

[38] Figura obtenida de: Xun Yu (2009, 5 de Mayo). Real-time Nonintrusive Detection of Driver Drowsiness

```
%Filtrado paso-banda

Wp=[2*1/128,2*25/128];%2*f/fs
R2=-20*log10(1-0.001);
N2=4;
[B2,A2] = cheby1(N2,R2,Wp,'bandpass');

Y2=filtfilt(B2, A2, Y1); % X es la señal en bruto del ECG o
```

Figura 44 Filtro paso-banda MatLab

```
try{
            double tasa_muestreo = Double.parseDouble(myPrefs.getSAMPLINGRATE());
            highpassfilter = new Filter(Filter.HIGH_PASS,tasa_muestreo, mHPFc);

            try
            {
                 ModuleDataArray = highpassfilter.filterData(ModuleDataArray);
                    //Fijamos las características del filtro.
                    try
                    {
       //Obtenemos la tasa de muestreo y la convertimos a double para poder realizar el filtrado paso-bajo
                            tasa_muestreo = Double.parseDouble(myPrefs.getSAMPLINGRATE());
                            lowpassfilter = new Filter(Filter.LOW_PASS, tasa_muestreo, mLPFc);
                            try
                            {   //Devuelve la señal filtrada, la cual tenemos que representar.
                                ModuleDataArray = lowpassfilter.filterData(ModuleDataArray);
```

Figura 45 Filtro Paso-Bajo y Paso-Alto App

De manera *off-line*, el siguiente paso consistirá en un diezmado y un filtrado paso bajo de la señal con el objetivo de reducir el número de muestras de la señal y que su coste computacional sea menor. El filtrado se realiza para evitar la superposición de muestras (*aliasing).* Su desarrollo en el lenguaje de programación *Matlab* se expone a continuación:

```
Wn1=2*16/128;%2*f/fs
N1=4;
[B1,A1] = butter(N1,Wn1);

%Diezmado factor 2
Y3 = decimate(Y2,4);
Y4=filtfilt(B1, A1, Y3);

Wn1=2*32/128;%2*f/fs
N1=4;
[B1,A1] = butter(N1,Wn1);

%Diezmado factor 4
Y5 = decimate(Y2,2);
Y6=filtfilt(B1, A1, Y5); %


Wn1=2*8/128;%2*f/fs
N1=4;
[B1,A1] = butter(N1,Wn1);

%Diezmado factor 8
Y15 = decimate(Y2,8);
Y16=filtfilt(B1, A1, Y15); %
```

Figura 46 Diezmado de las señales ECG

A continuación se muestra en la figura 47 la señal ECG sin ningún filtrado.

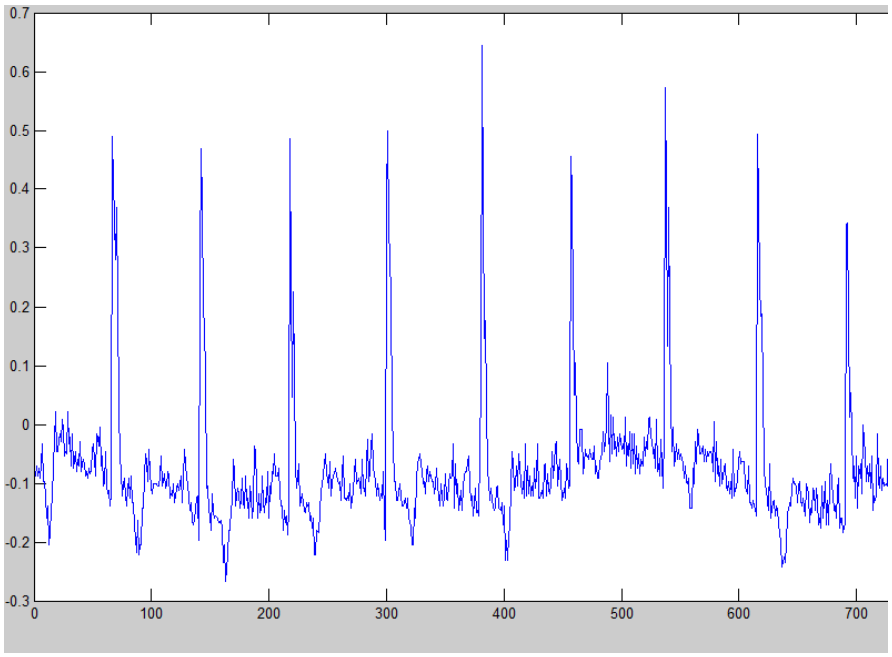


Figura 47 Señal ECG sin filtrar

En la figura 48 se observa la señal filtrada con los filtros descritos anteriormente.

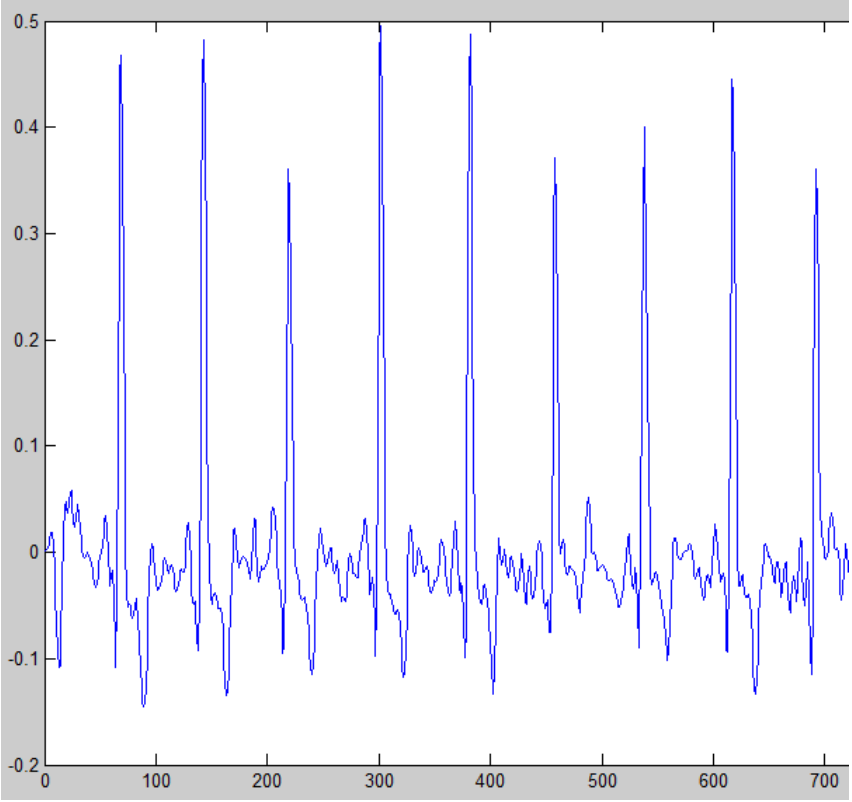


Figura 48 Señal ECG filtrada

La comparación de las señales en la figura 49 hace visible los beneficios del filtrado puesto que la alteración en la señal ECG traería como consecuencia la detección de un falso pico del complejo QRS.

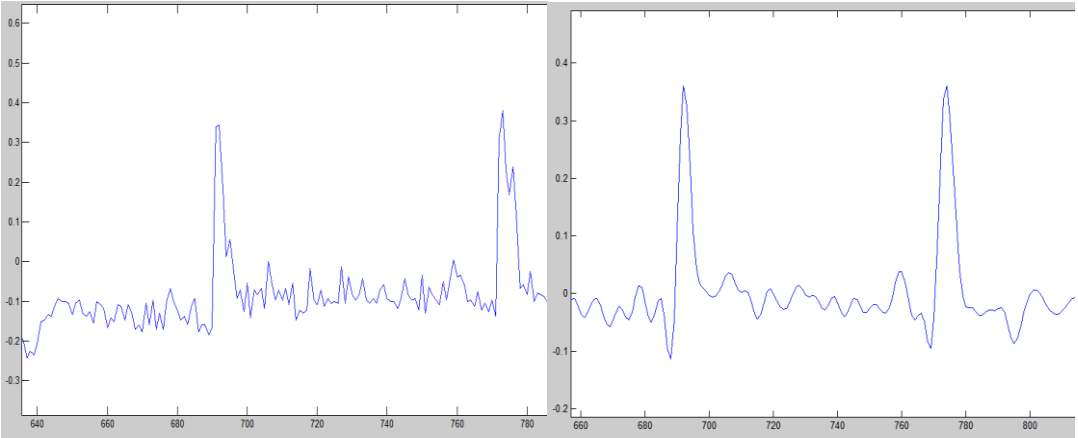


Figura 49 Señal ECG sin filtrar y filtrada

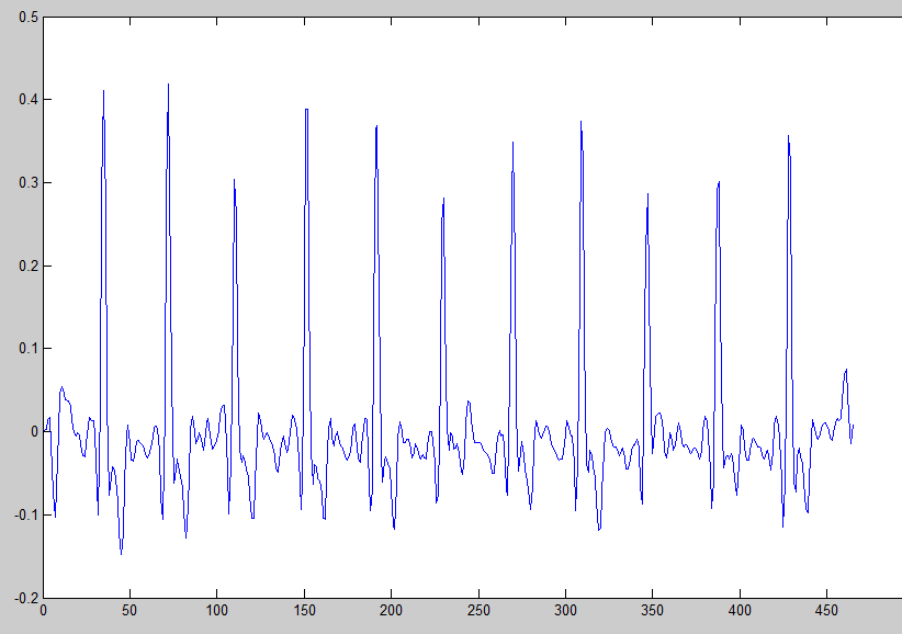


Figura 50 Señal ECG filtrada y diezmada por factor 2

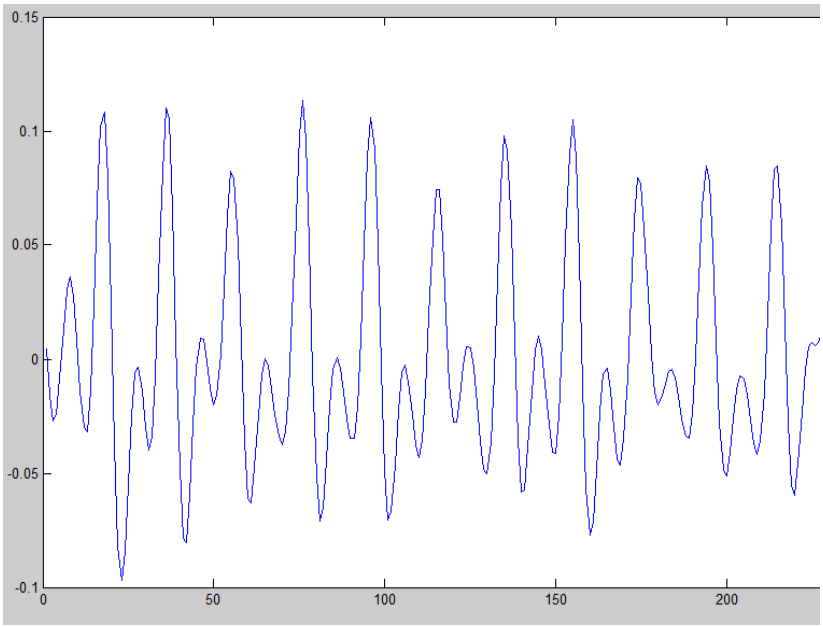


Figura 51 Señal ECG filtrada y diezmada por factor 4

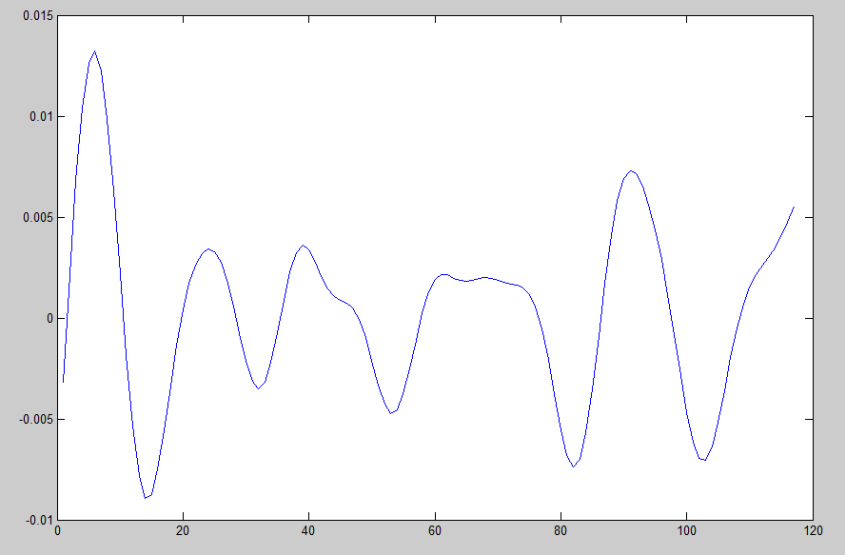


Figura 52 Señal ECG filtrada y diezmada por factor 8

La señal filtrada es uniforme asemejándose a un complejo QRS sin alteraciones como los que presenta la figura 47 y 49 debido al ruido que contamina nuestra señal. El diezmado de la señal visible en las figuras 50 y 51 con la mitad y una cuarta parte de las muestras tomadas en la adquisición de la señal, refleja cómo se puede seguir detectando los picos de la señal. Sin embargo, se aprecia que si se diezma demasiado la señal no se pueden diferenciar los distintos picos y se estaría perdiendo una gran cantidad de información.

El siguiente paso consiste en buscar la posición de los picos y cálculo de los RR para calcular el ratio entre LF y HF:

1) En la aplicación Android utilizaremos un *threshold* bastante amplio (en torno a los 0.2 mV *offline* y 0.3 mV *online*) de tal manera que cuando sea superado se procederá a la detección y almacenamiento de la muestra más elevada hasta que comience a decrecer. Dicho punto será almacenado junto al *timestamp* en el que fue capturado. El motivo por el que se utiliza un valor umbral es debido a que el filtrado realizado en tiempo real en la aplicación no es tan efectivo como el que se puede realizar desde un *software* preparado para el procesado de señales. A continuación se comprueba si es el primer pico detectado para poder calcular el tiempo con el anterior o esperar la llegada del siguiente. Con la detección de cada uno de ellos, se resta su sello temporal con el anterior y se almacena en un vector. Una vez que se han

alcanzado los 70 RR se procede al cálculo de la FFT y su densidad espectral de potencia, calculando el ratio entre LF y HF.

```
if(values[i] >= almacen && values[i] >=threshold)
{
    almacen = values[i];
    TimeStamp[0] = values[1];  //Almacenamos la muestra temporal. Así siempre tenemos la muestra tempora

}else if(values[i] < almacen && values[i] <= threshold ){

    almacen = 0; // Restauramos el valor para buscar el siguiente máximo.
 int xxx = (int) MaximosECG;
TimeStampMaximos[xxx] = TimeStamp[0];
    //Log.i("MAX3",TimeStampMaximos[xxx]+",\n");
    MaximosECG = MaximosECG + 1;
    Log.i("MAX",MaximosECG+",\n");
    if(MaximosECG > 1){
        double aaaa= TimeStampMaximos[xxx];
        double aaaa1= TimeStampMaximos[xxx-1];
             double be = ((aaaa-aaaa1)/*/1000*/);
             RR[xxx-1] = be/1000;
             double resto = MaximosECG % 7;
            if(resto == 0)
            {
             Context context2 = this.getContext();
            int duration = Toast.LENGTH_SHORT;
            Toast toast2 = Toast.makeText(context2, "Las pulsaciones son "+60*1/RR[xxx-1], duration);
            toast2.show();
            }
             Log.i("MAX",RR[xxx-1]+",\n");
    }
```

Figura 53 Búsqueda de máximos y almacenamiento de RR

```
for (int k = 0; k < 59; k++)
    {
        double sumreal = 0.0;
        double sumimag = 0.0;
        for (int t = 0; t < 59; t++) {
            double angle = (2 * Math.PI * t * k);
            angle = angle/59;
            if(k==0)
            sumreal = sumreal + RR[t] * Math.cos(angle);
            sumimag = sumimag - RR[t] * Math.sin(angle);

        }
        realOutA[k] = sumreal;
        imagOutA[k] = sumimag;
    }
```

Figura 54 Cálculo de la FFT

```
for(int cnt=0;cnt < 30/*realOutA.length*/;cnt++){
      String arr = String.valueOf(cnt);
     double cnt2 = Double.parseDouble(arr)  ;
      if(cnt2/59 >= 0.04 && cnt2/59 <= 0.15)
      {
          aq += realOutA[cnt] * realOutA[cnt] + imagOutA[cnt]*imagOutA[cnt];
      }else if((cnt2/59) > 0.25 && (cnt2/59) < 0.4){
          aq1 += realOutA[cnt] * realOutA[cnt] + imagOutA[cnt]*imagOutA[cnt];
      }
      else if(cnt == 29 || cnt2 == 29.0)
      {
          ratio = aq / aq1;
          MaximosECG = 0;
          aq=0;
          aq1=0;
          Context context = this.getContext();
          int duration = Toast.LENGTH_LONG;

          Toast toast = Toast.makeText(context, "El ratio es:"+ratio, duration);
          toast.show();

      }
```

Figura 55 Cálculo de la PSD y del ratio

2) En el *script* de MatLab buscaremos el máximo de una manera sencilla comprobando que la muestra es menor tanto de la anterior como de la posterior estableciendo un umbral pequeño de entre 0.15mV y 0.3 mV para evitar el ruido. Los siguientes pasos son los mismos establecidos en el punto número 1.

```
for k=2:(length(vector)-1)
    if vector(k)>vector(k-1) && vector(k)>vector(k+1) && vector(k)>0.2
        maximos(end+1)=vector(k);
        vectorK(p) = [k]
        p=p+1;
    end
end
```

Figura 56 Búsqueda de máximos Matlab

```
for j=2:length(maximos)

    a = TimeStamp(vectorK(j))-TimeStamp(vectorK(j-1));
    a= a/1000;    %Pasamos a segundos
    RRinterval(j) = [a]; %Almacenamos en el vector los RR
    RR=60/a       %Cálculo del ritmo cardiaco
 end
```

Figura 57 Almacenamiento RR y cálculo del ritmo cardiaco

```
N=length(RRinterval);
XDFTreal=zeros(N,1);
XDFTimag=zeros(N,1);
for k=1:N
    XDFTreal(k)=0;
    XDFTimag(k)=0;
    for n=1:N
        XDFTreal(k)=XDFTreal(k)+RRinterval(n)*cos(2*pi*(n-1)*(k-1)/N);
        XDFTimag(k)=XDFTimag(k)-RRinterval(n)*sin(2*pi*(n-1)*(k-1)/N);
    end
end
       fs=128 N = length(RRinterval);
       S =((1/N)*(fft(RRinterval)));
       S=S(1:ceil(N/2));
       SS = S.*conj(S);
       f = linspace(0,1/2,length(S))
    figure()
    plot(f, abs(S))
       % axis([0 0.5 0 1]);
           figure()
   plot(f, abs(SS));
   axis([0.04 0.4 0 0.0001]);
   figure()
   plot(RRinterval)
   indice1=find(f>=0.04,1)
   indice2=find(f>=0.15,1)
   indice3=find(f>0.25,1)
   indice4=find(f>=0.385,1)
   PotLF=sum(SS(indice1:indice2))
   PotHF=sum(SS(indice3:indice4))
   ratio=PotLF/PotHF
```

Figura 58 Cálculo de la FFT, la PSD y el ratio

La densidad espectral de potencia tiende a ser una delta debido a que la situación ideal sería que la variación entre los intervalos RR fuese nula. Sin embargo, dichas variaciones hacen que la PSD se vea desplazada. A mayor variación del tiempo RR, el desplazamiento de la PSD hacia la derecha sería mayor, aumentando las componentes situadas en alta frecuencia.

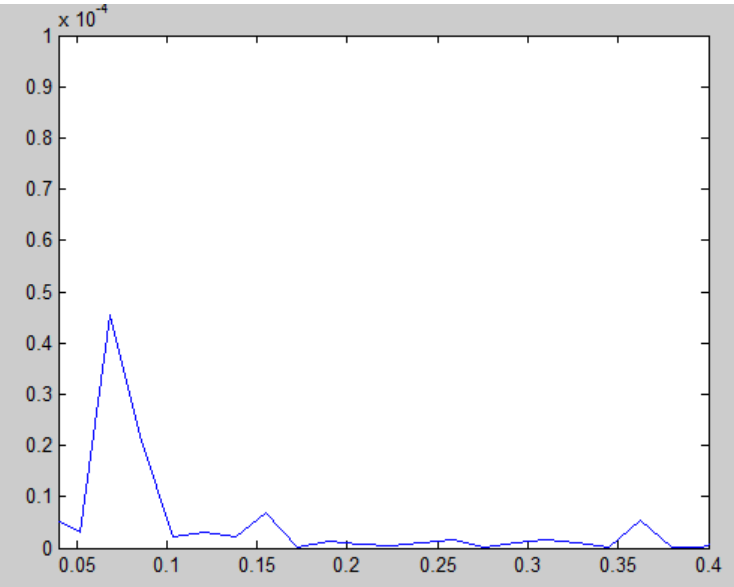


Figura 59 PSD

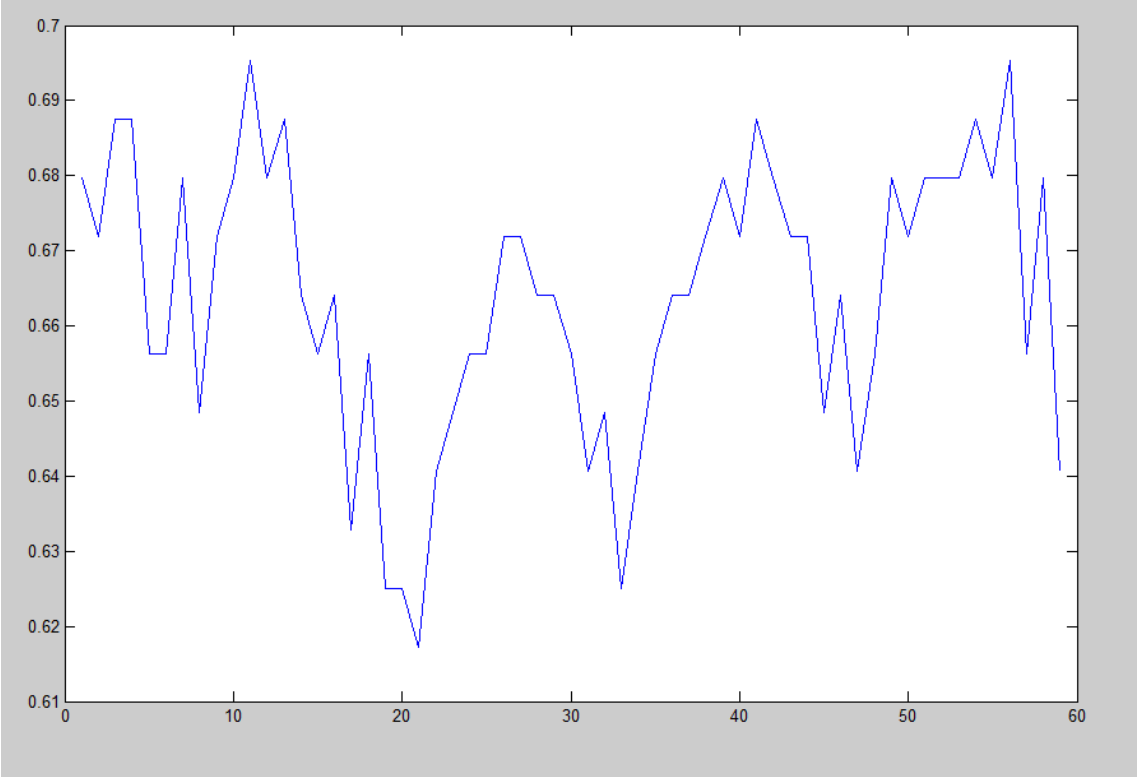


Figura 60 HRV

```
PotLF =

  8.3771e-005


PotHF =

  1.1289e-005


ratio =

    7.4209
```

Figura 61 Resultados obtenidos

Las pruebas realizadas en diferentes estados han arrojado los siguientes ratios:

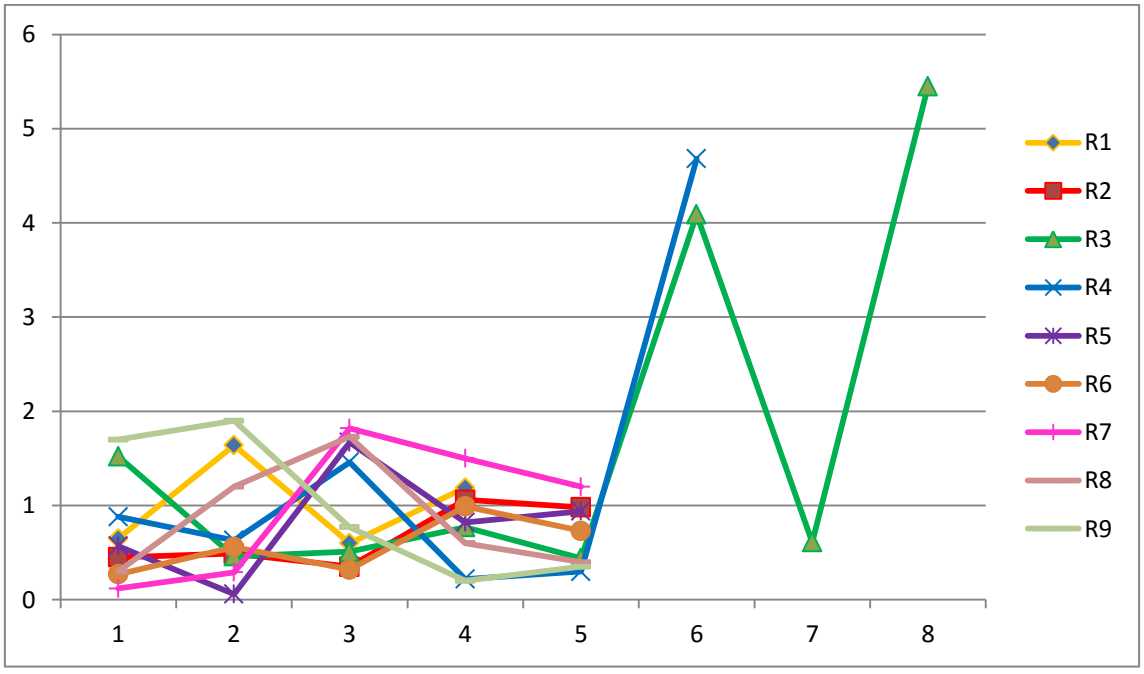


Figura 62. Gráfico fatiga

	Serie 1	Serie 2	Serie 3	Serie 4	Serie 5	Serie 6	Serie 7	Serie 8	Serie 9
1	0,65	0,45	1,52	0,88	0,57	0,27	0,12	0,3	1,7
2	1,64	0,49	0,46	0,63	0,06	0,56	0,29	1,2	1,9
3	0,61	0,35	0,51	1,46	1,67	0,32	1,82	1,73	0,77
4	1,19	1,06	0,77	0,22	0,82	0,99	1,5	0,6	0,2
5		0,98	0,44	0,3	0,94	0,73	1,2	0,397	0,35
6			4,09	4,68					
7			0,61						
8			5,45						
9									

Tabla 4. Gráfico fatiga

Se puede observar que los ratios cuando el sujeto está fatigado son más bajos debido a que el ritmo cardiaco es más irregular. Los casos en los que el individuo se encuentra más fatigado son R2, R5 y R6 debido a que los *ratios* obtenidos en las pruebas son prácticamente inferiores a 1. En cambio en R3 y R4 la somnolencia es menor porque aunque haya valores muy bajos también hay otros altos. En R1, R7, R8 y R9 con valores también muy bajos indican que el individuo se encuentra bastante fatigado pero no tanto como en el primer caso. No obstante, ya sería una situación en la que se debería tomar medidas para descansar y no seguir conduciendo porque se estaría poniendo en riesgo la seguridad vial.

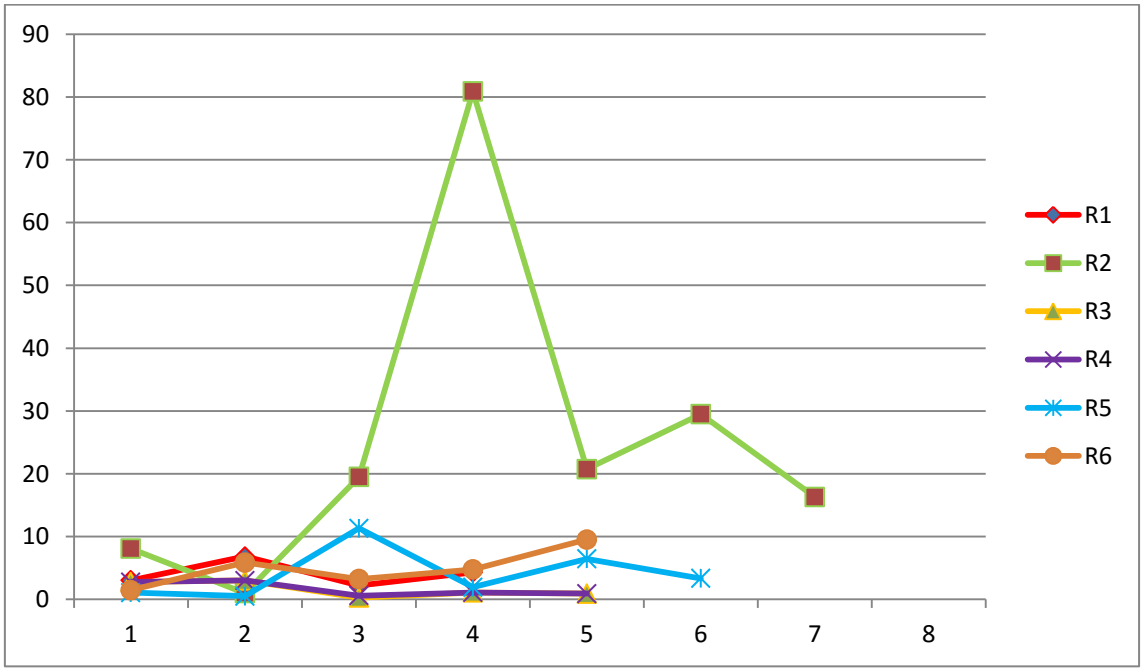


Figura 63. Gráfico usando el móvil

	Serie 1	Serie 2	Serie 3	Serie 4	Serie 5	Serie 6
1	3,04	8,09	2,75	2,75	1,08	1,49
2	6,83	1,01	3	3	0,52	5,87
3	2,19	19,49	0,28	0,58	11,31	3,23
4	4,32	80,86	1,11	1,11	1,93	4,77
5		20,75	0,92	0,92	6,46	9,53
6		29,49			3,349	
7		16,32				

Tabla 5. Usando el móvil

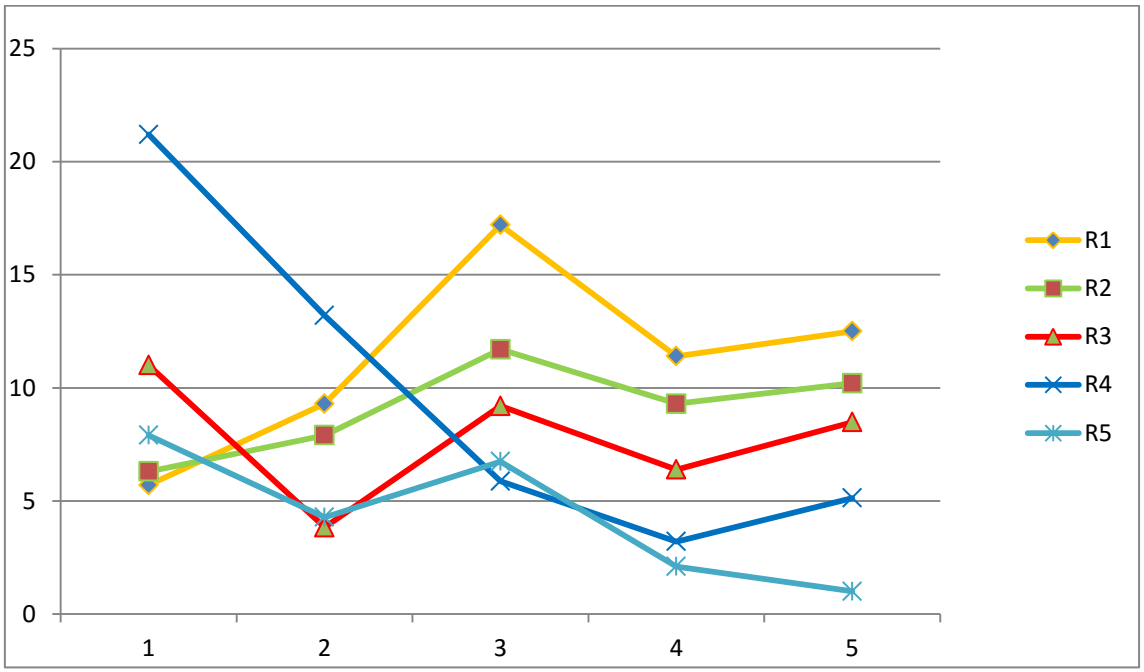


Figura 64. Gráfico normal

	Serie 1	Serie 2	Serie 3	Serie 4	Serie 5
1	5,7	6,3	11	21,2	7,9
2	9,3	7,9	3,83	13,2	4,28
3	17,2	11,7	9,2	5,87	6,74
4	11,4	9,3	6,39	3,2	2,1
5	12,5	10,2	8,48	5,13	1

Tabla 6. Normal

En estos dos últimos casos se observan ratios mayores debido a que se producen menos variaciones en la frecuencia cardiaca y la mayoría de la potencia permanece en la baja frecuencia sin que se produzcan desplazamientos hacia la alta. Esto se debe a que cuando un individuo se encuentra en un estado normal el intervalo temporal entre los picos del complejo QRS es más regular. Cuanto mayor sea este ratio menos

somnolencia tendrá el conductor. Tanto en la tabla 5 (R3 y R4) como en la tabla 6 (R5) existen algunos casos con *ratios* bajos que indican que los primeros signos de fatiga en el individuo. En el resto de casos, el *ratio* es lo suficientemente alto como para asegurar que no hay riesgo de quedarse dormido al volante.

En la figura 65 y 66, se observa como es el procesado *offline* de un señal ECG durante una sesión de conducción.

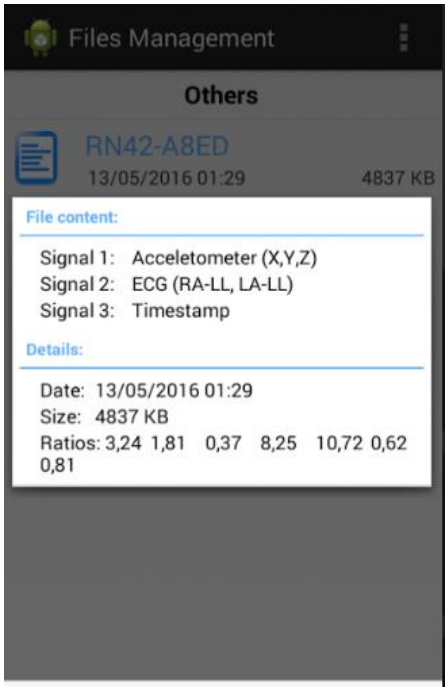


Figura 65 Ratios calculados en la App

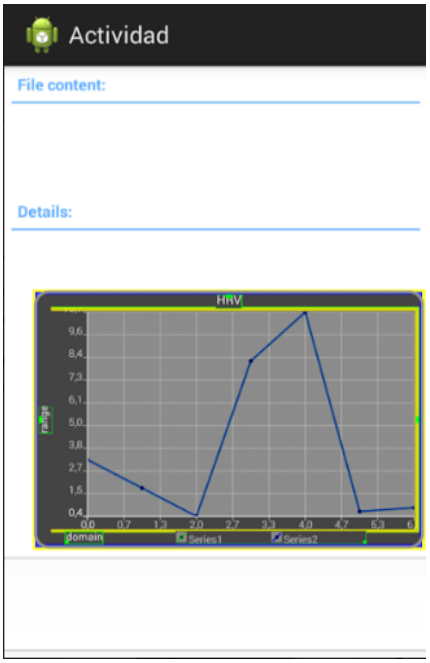


Figura 66 Representación gráfica de los ratios

Las figuras 67 y 68, son una demostración de la aplicación funcionando en tiempo real, representando la señal y mostrando en una notificación los ratios que se van obteniendo.

Figura 67 Ratio calculado en tiempo real

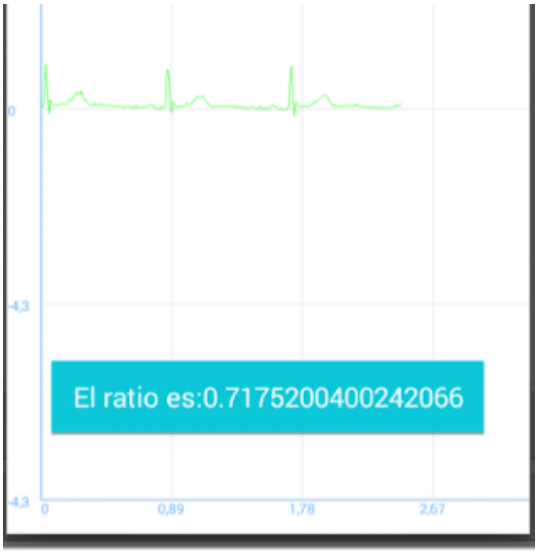


Figura 68 Ratio calculado en tiempo real (2)

V.2.2 Relación fatiga y señal EMG

El uso del electromiograma que registra la actividad del sistema nervioso periférico, en nuestro trabajo ayuda a determinar el estado físico en el que se encuentra el conductor con más detalle si está fatigado y/o aproximándose a un estado de somnolencia.

El sistema nervioso periférico es la torre de control y de comunicaciones de nuestro organismo. Está constituido por un gran número de células llamadas neuronas de diversos tamaños y formas que se comunican con diferentes partes del cuerpo por medio de señales eléctricas, rápidas y específicas y entre ellas mismas mediante la sinapsis. Las neuronas motoras son las responsables de los impulsos eléctricos que activan las fibras musculares. Se distinguen tres partes principales que son el cerebro, la médula espinal y los nervios periféricos.

Los músculos se componen de células especializadas capaces de la contracción y de la relajación y cuya misión principal es la generación de fuerza para producirse los movimientos. El tejido muscular tiene extensibilidad y elasticidad y realiza tres funciones principales que son producir movimiento, proporcionar estabilización y generar calor. De todos los tipos de tejidos musculares, la EMG solo es aplicable al músculo esquelético, el cual se une al hueso y tiene la responsabilidad de mover el cuerpo.

La señal EMG es una señal biomédica que mide la corriente eléctrica generada en un músculo o grupo muscular durante su contracción, es decir, las actividades neuromusculares. Es una señal compleja, diferente para cada músculo debido a que depende de las características anatómicas de los mismos y adquiere una cantidad de ruido considerable en su viaje a través de los diferentes tejidos y debido a la captura de otras señales. Los tipos de ruido que afectan a la señal EMG son:

1) Ruido inherente de los equipos electrónicos: Todos los equipos generan ruido que solo se puede eliminar utilizando componentes de alta calidad.
2) Ruido ambiental: Procede de la radiación eléctrica y magnética imposible de evitar siendo la responsable la superficie de la tierra.
3) Ruido debido al movimiento: El dispositivo introduce información sesgada en el sistema provocando irregularidades en los datos debido al desplazamiento y cable de los electrodos. Es evitable con una buena circuitería y unos cables aislantes.

Debido a que es un método relativamente nuevo presenta limitaciones en la detección y caracterización de la señal y los algoritmos existentes tienen un alto coste computacional y en ocasiones un alta varianza. Los métodos más comunes para el estudio de las características propias son la transformada de *Fourier (FFT)* que se adapta muy bien gracias a que es una señal no estacionaria , la densidad espectral de potencia, la amplitud de la señal y distintos enfoques relacionando las señales tiempo-frecuencia.

Siguiendo el método de la amplitud de la señal, este consiste en establecer un valor umbral determinado por un observador, detectando cada vez que se produce actividad y cuál es el valor de pico. Sin embargo no es un método fiable debido a que los resultados pueden variar en función del umbral establecido y es posible que haya falsas alarmas debido al ruido. Se puede solventar incluyendo un doble umbral para que en caso de superar uno se contemple el siguiente. A pesar de esta mejora el porcentaje de falsas alarmas sigue siendo elevado. En el caso de que el sujeto esté fatigado la amplitud de la señal será mayor debido a que hay que realizar un esfuerzo mayor para llevar a cabo un mismo movimiento. Al mismo tiempo se realiza un análisis en el tiempo relativo a la duración en la que ha estado el músculo activo.

El método de la FFT consiste en realizar la transformada de *Fourier* de la señal y a continuación su densidad espectral de potencia observando en ambas gráficas cuál ha sido la contribución en las frecuencias medias. A medida que el esfuerzo es más intenso y el individuo se encuentra en un estado más fatigado se produce un desplazamiento disminuyendo su contribución a la banda de frecuencia anterior.

En la figura 69 se presentan los resultados de una prueba de conducción de una persona en un estado físico normal, no fatigado. Se observa como la energía de la señal se concentra en las frecuencias intermedias, disipándose tanto en las altas como en las bajas.

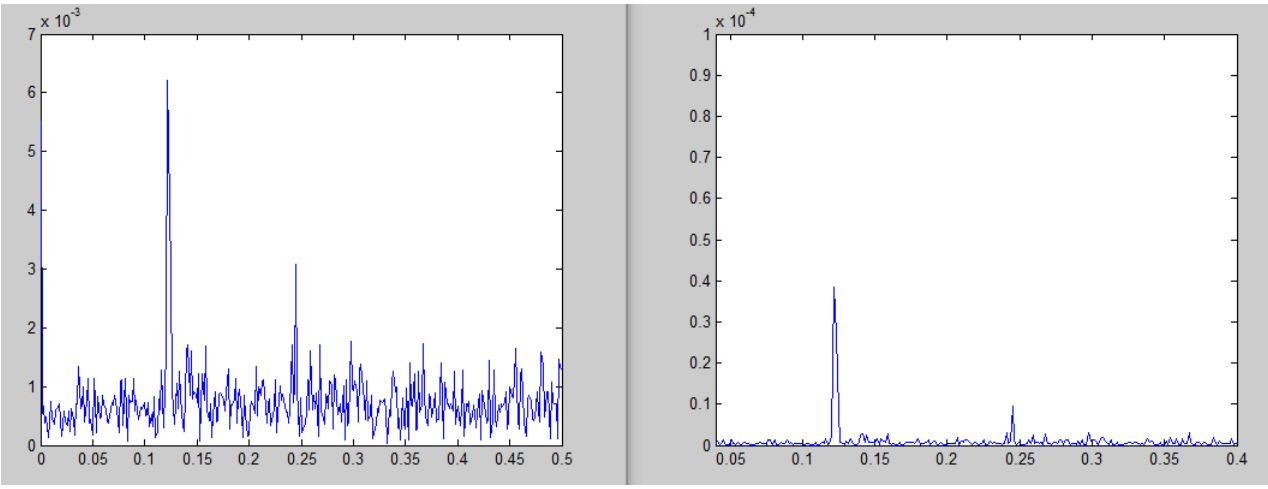


Figura 69 FFT y PSD en estado físico normal

En la siguiente prueba realizada, la fatiga estaba presente de manera leve en el individuo y los resultados arrojados cuando se aplica la transformada de Fourier y la densidad espectral de potencia demuestran que se ha desplazado la banda de energía de la señal en la densidad espectral de potencia y aparecen componentes en baja y alta frecuencia.

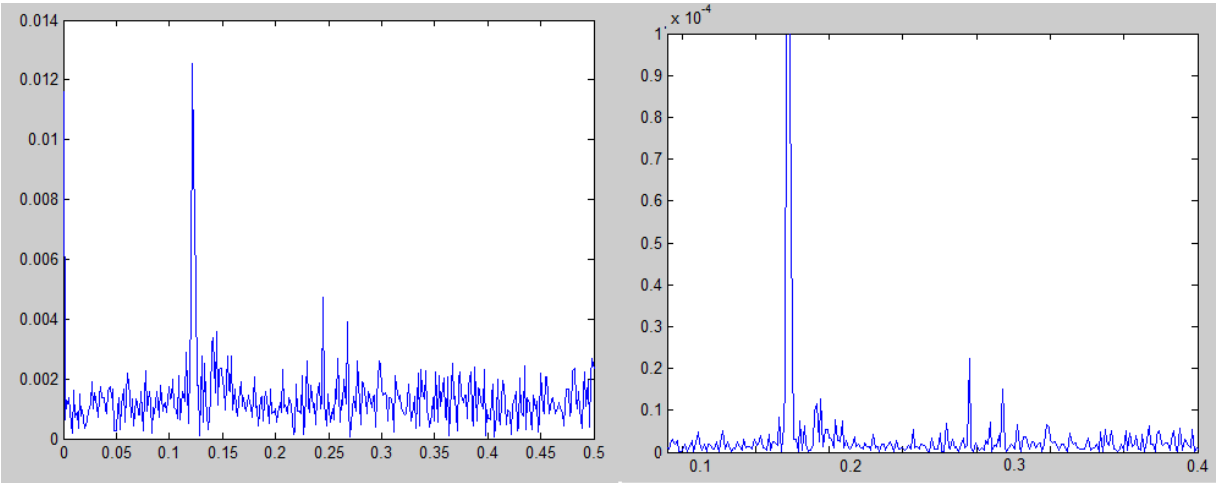


Figura 70 FFT en estado físico un poco fatigado

En la última prueba realizada sobre el individuo la fatiga y el nivel de sueño eran bastante altos y se observa una presencia casi igual de componentes frecuenciales tanto en baja, media y alta. La energía de la señal se encuentra repartida en las tres bandas observándose el desplazamiento hacia las bajas frecuencias principalmente.

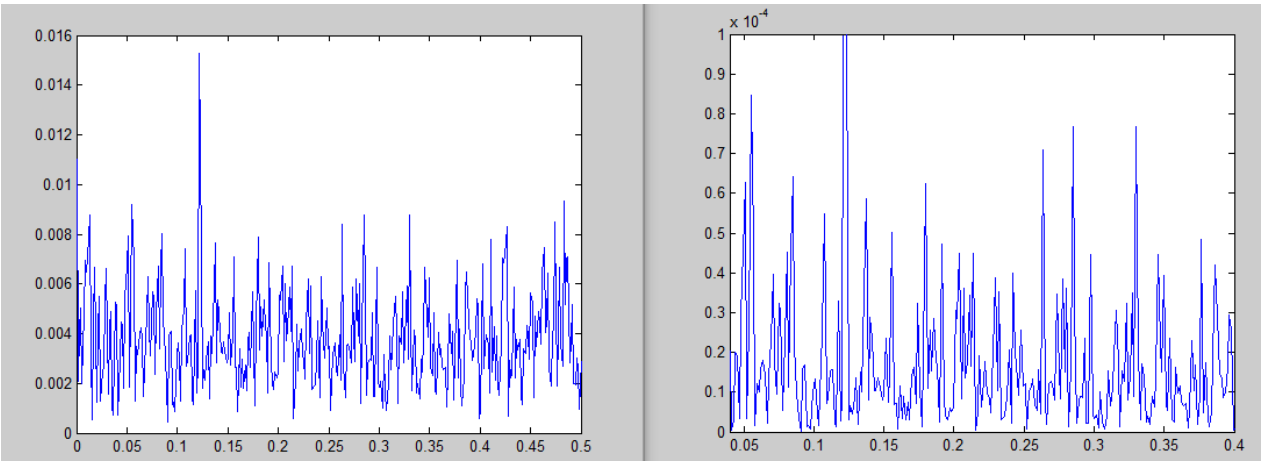


Figura 71 FFT y PSD en estado físico fatigado

Debido a su alto coste computacional, tenemos la limitaciones si realizáramos el procesado en tiempo real desde la aplicación por lo que se procederá a la captura de datos desde la misma y se procesaran *off-line* con *MatLab*.

V.2.3 Relación fatiga y señal GSR

La conductancia de la piel es uno de los mejores métodos para la medición de parámetros relacionados con el sistema nervioso autónomo asociándose a las emociones, las alertas y sobre todo la atención.

La detección de la fatiga basada en la respuesta galvánica de la piel se refleja en los cambios producidos en los niveles de resistencia ofrecidos por la misma debido a la presencia de glándulas sudoríparas ecrinas muy sensibles a los cambios producidos por estímulos de tipo psicológico relacionados con el estado emocional. El sudor es controlado por el sistema nervioso simpático, que es responsable de la estimulación del organismo y que, cuando se está muy excitado, su secreción aumenta.

Hay dos tipos de conductancia en la piel, llamadas tónica y fásica:

- La conductancia tónica tiene lugar cuando hay una ausencia de acontecimientos o estímulos en el medio ambiente que nos rodea. Cada individuo tiene una conductancia en la piel determinada que varía a lo largo de los años y el estado emocional en el que se encuentra.

- La conductancia fásica cambia cuando se ha producido un evento, ya sea por algo visto, un olor, un sonido, o por un trabajo o estrés etc... A continuación regresa levemente a la fase tónica o bien se mantiene en el tiempo dependiendo de cómo sea el estímulo que ha producido el cambio.

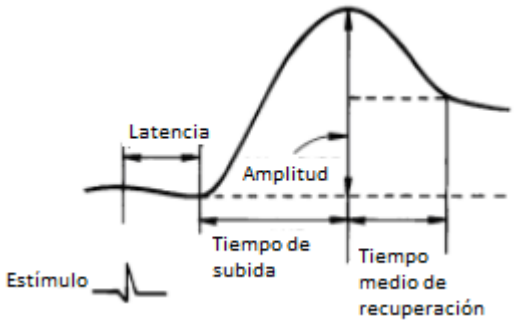


Figura 72 Ejemplo conductancia fásica señal GSR[39]

A medida que el conductor se encuentra más activo y estresado al volante el nivel de resistencia aumenta progresivamente sobre todo en condiciones de tráfico urbanas y acentuándose en horas punta cuando el volumen de tráfico es más elevado y

[39] Figura obtenida de: Milan Stork (s.f). Some methods systems and sensor which are possible for driver's drowsiness estimation.

se producen retenciones. Sin embargo, cuando el conductor se encuentra más cansado el nivel disminuye a cotas inferiores pudiendo establecer un criterio que nos va a permitir conocer cuándo el conductor ha alcanzado un estado de somnolencia, momento en que se puede alertar al mismo de su situación real, evitando de esta manera un posible y probable accidente.

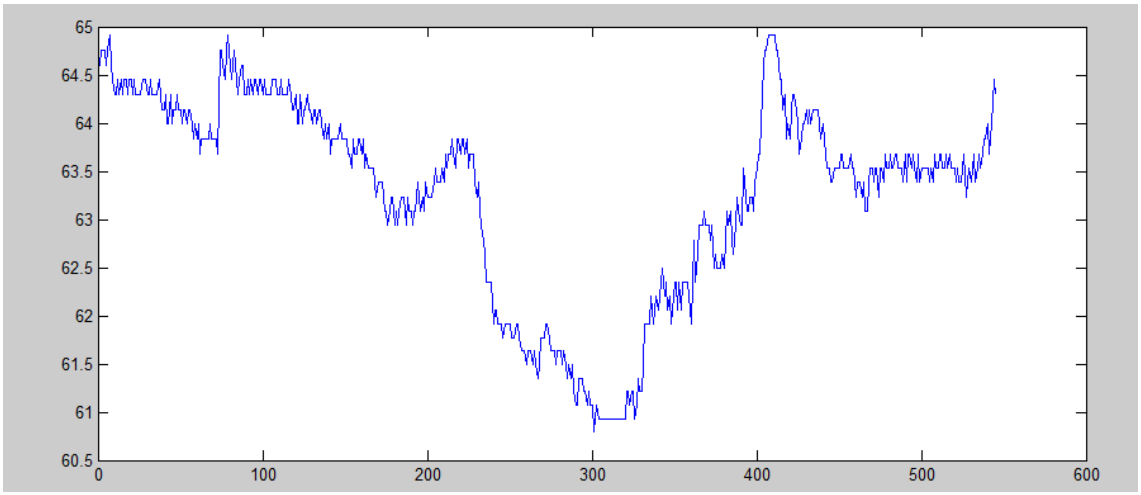
Es muy importante tener en cuenta que, cuando una persona se encuentra fatigada y continúa realizando una actividad, la resistencia galvánica aumenta bruscamente debido a que se necesita una mayor estimulación para desempeñar la tarea. Sin embargo, si se entra en una fase de somnolencia de manera tranquila, la resistencia galvánica disminuye.

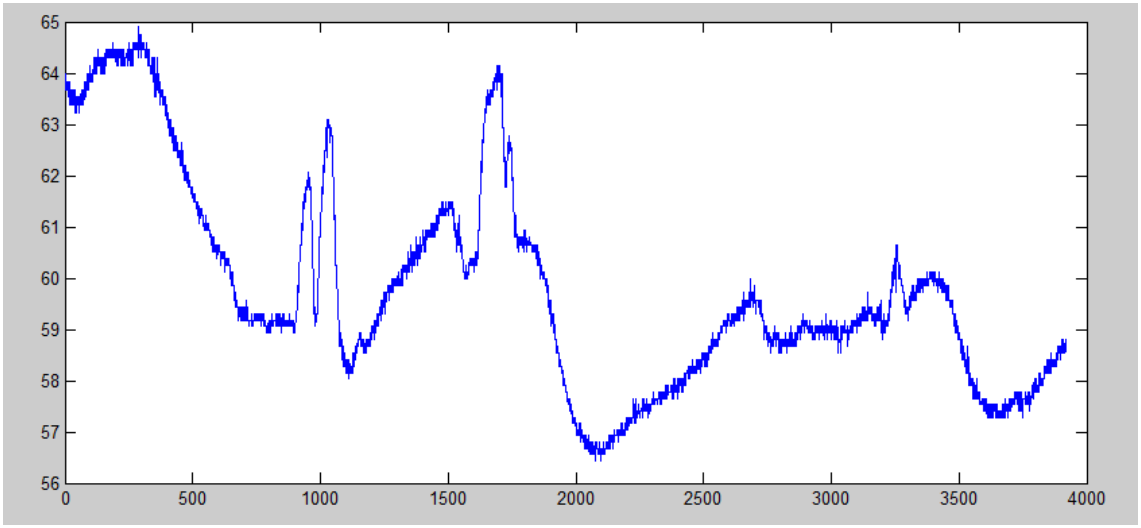
En las diferentes pruebas realizadas en diferentes condiciones psicofísicas y con personas de edades diferentes, se ha podido comprobar que los diferentes umbrales varían en comparación con los presentados en las diferentes fuentes consultadas ya sea por la sensibilidad de los sensores empleados o por la calibración de los mismos. No obstante, se pueden agrupar de nuevo en unos intervalos en función de los diferentes datos obtenidos conociendo el estado físico del conductor previamente y relacionándolos con los artículos.

Las pruebas realizadas con dos personas (referidas como persona 1 y persona 2 más adelante, siendo ambas hombres de edades 24 y 43 años, respectivamente) durante la conducción con una frecuencia de muestreo de 10.2Hz han sido las siguientes:

1) Conducción en autovía y poco tráfico: Los resultados son bastante estables en torno a una frecuencia de 60 kΩ - 65 kΩ. No se produce ninguna estimulación por parte del entorno. Como se puede apreciar lo resultados de la GSR son bastante constantes a lo largo de tiempo, no produciéndose ninguna alteración.

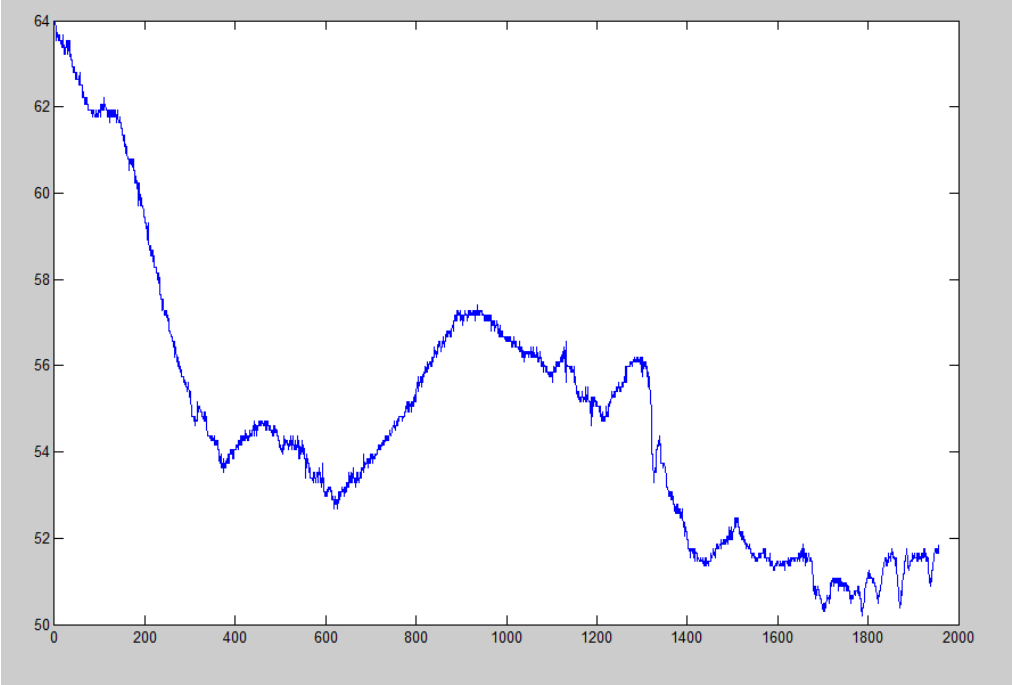
   Prueba realizada con: persona 1

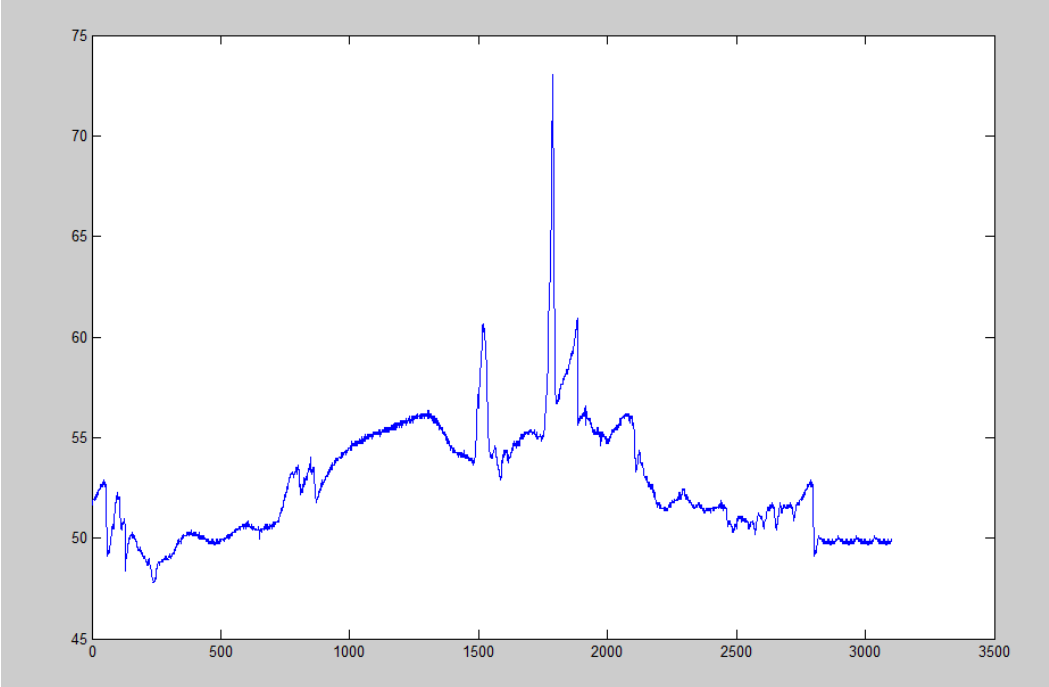




2) Conducción en autovía, poco tráfico y después de haber realizado ejercicio físico intenso: Los resultados son más variables comenzando a decrecer de manera más brusca hacia unas frecuencias de 50 kΩ. En la segunda prueba se produce una alteración por elemento externo como puede ser un reducción de velocidad brusca por la incorporación de un nuevo vehículo, sonido de las líneas laterales de las carreteras para corregir la dirección del vehículo, etc.. ascendiendo hasta 75 kΩ y volviendo a su posición inicial una vez finalizada. Decrece apareciendo signos de somnolencia.

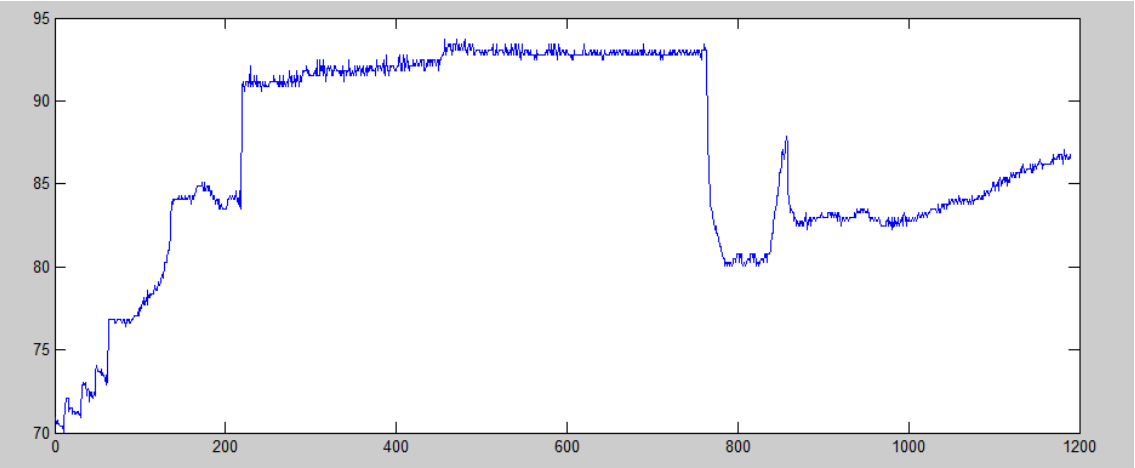
Prueba realizada con: persona 1


64
62
60
58
56
54
52
50
0
200
400
600
800
1000
1200
1400
1600
1800
2000


75
70
65
60
55
50
45
0
500
1000
1500
2000
2500
3000
3500

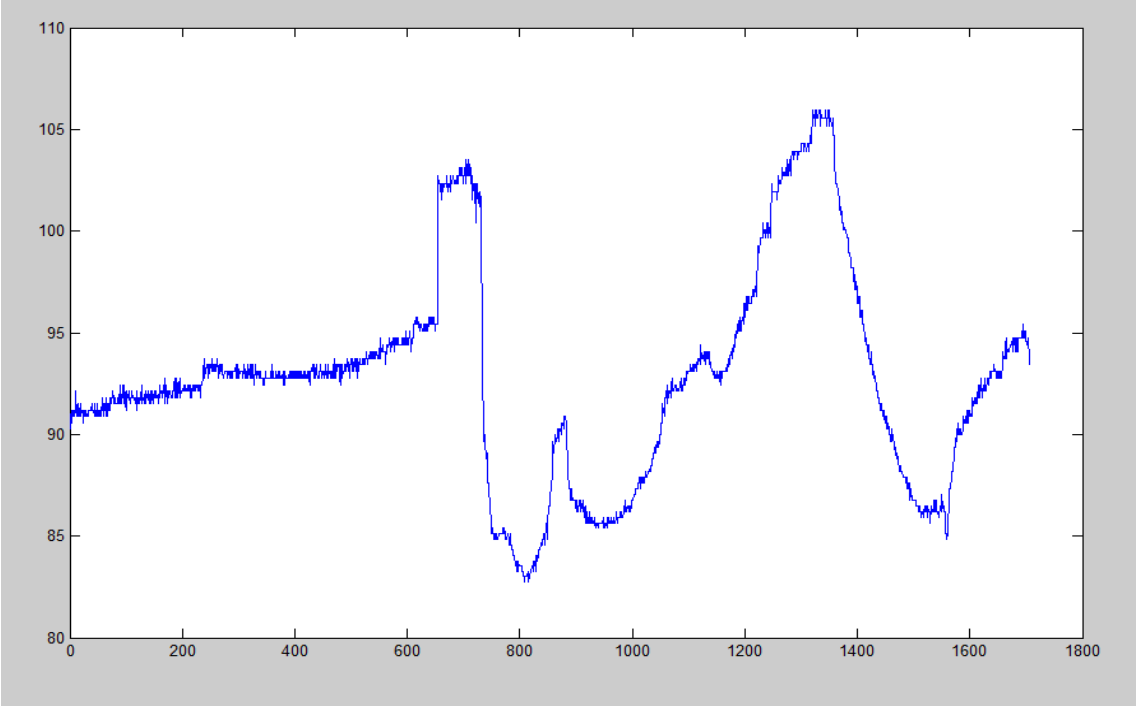
3) Conducción urbana, tráfico medio: En el trascurso de la conducción en un entorno urbano, hay más agentes y más preocupaciones al volante. Partiendo de una resistencia inicial de 70 kΩ aumenta hasta rondar los 95 kΩ, decreciendo posteriormente y teniendo otra vez tendencia ascendente.

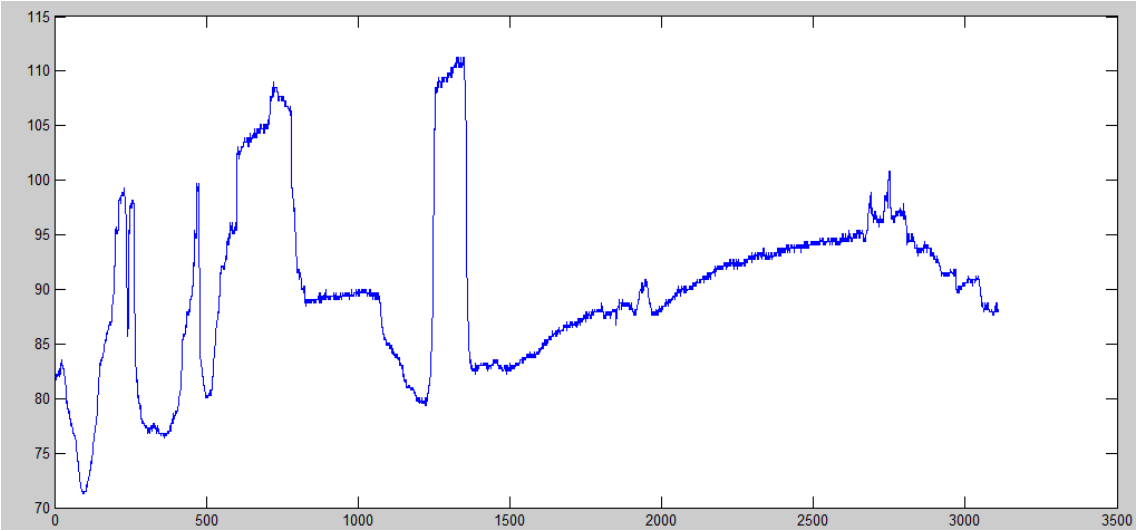
Prueba realizada con: persona 2



4) Conducción urbana, tráfico elevado, sujeto más estresado: En esta situación el estrés es bastante mayor y hay más eventos que hacen que presente mayores variaciones. Se va más pendiente de los coches que circulan a nuestro lado, los semáforos, adelantamientos, pasos de peatones, rotondas, etc…

Prueba realizada con: persona 2





En la aplicación, cuando el valor desciende del umbral establecido, se lanza una notificación para alertar al usuario que puede quedarse dormido en un periodo de tiempo bastante corto, como se puede apreciar en las siguientes líneas de código y se gestiona la notificación.

```
else if(ModuleNames[Dev][0] == "GSR"){

    if(ModuleDataArray[0] <= 725  && GSRbucle != 100)
    {
        GSRbucle = GSRbucle + 1;

    }else if(ModuleDataArray[0] <= 725  && GSRbucle >= 10)
    {
        //Lanzamos mensaje de alerta.
        mBuilder.setContentText("Alerta, Somnolencia!"); //modificamos el texto que aparecerá
        mBuilder.setContentIntent(pendInt);// le indicamos que debe hacer cuando se pulse
        mNotificationManager.notify(105,mBuilder.build()); // la lanzamos.

    }

    ModuleGraph[Dev].setDataGSR(ModuleDataArray);

        //creamos la notificación
mBuilder = new NotificationCompat.Builder(ManageDevices.this).setContentTitle("Alerta, Somnolencia!").setContentTex
        "Alerta, Somnolencia!").setSmallIcon(R.drawable.ic_launcher);

        //intent con el que indicaremos la acción que sucederá cuando pulsemos en la notificación
pendInt = PendingIntent.getActivity(this, 0, new Intent(), PendingIntent.FLAG_UPDATE_CURRENT);

        //gestionamos la notificación
mNotificationManager = (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
```

V.2.4 Relación fatiga y frecuencia respiratoria

La monitorización de la frecuencia respiratoria es muy útil para un seguimiento en las actividades deportivas, de pacientes para la detección de anomalías y para la detección de la somnolencia, en nuestro caso aplicado a la conducción. La variación de esta señal fisiológica es crucial para la aplicación de una contramedida a la somnolencia siendo esta uno de los factores más significativos en la causa de los accidentes de tráfico como hemos visto en el capítulo uno. Los cambios en la frecuencia respiratoria son producidos por alteraciones en el sistema nervioso central a consecuencia del cansancio y la somnolencia. Es comúnmente aceptado que la aparición de la fatiga se acompaña con una disminución de la frecuencia respiratoria (Sun Yu, 2009)

Se realizó un estudio (Clifford, 1982) sobre la tasa de respiración en 19 pacientes (8 hombres con una edad media de 32 años y 11 mujeres con una edad media de 26.5) con un buen estado de salud, sin obesidad mórbida ni ingesta de medicamentos con efectos secundarios relacionados con la somnolencia. Sin privación previa de somnolencia y en la franja horaria entre las 10 pm y las 7am, se determinó que la frecuencia respiratoria disminuye en la fase previa al sueño y durante la primera fase del mismo. Además se realiza un estudio sobre el volumen de aire inspirado y espirado y sus variaciones en otras etapas del sueño y la vigilia.

Cuando un individuo se duerme su frecuencia respiratoria cae bruscamente y es más estable sin embargo cuando está despierto. Si está realizando alguna actividad, presenta más variaciones y tiende a aumentar justa antes de entrar en un estado de fatiga. (Warwick Xiao Chen, s.f)

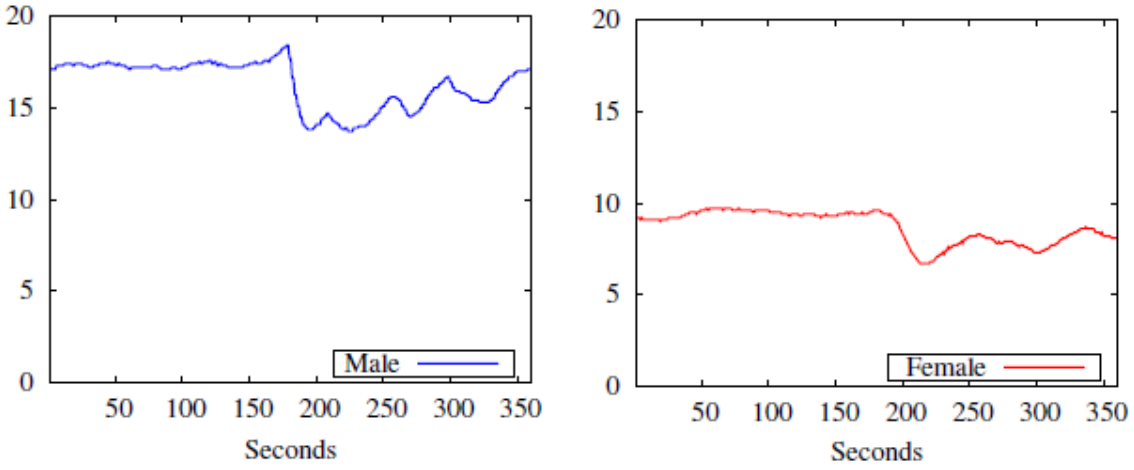


Figura 73 Frecuencia respiratoria cuando un individuo se queda dormido[40]

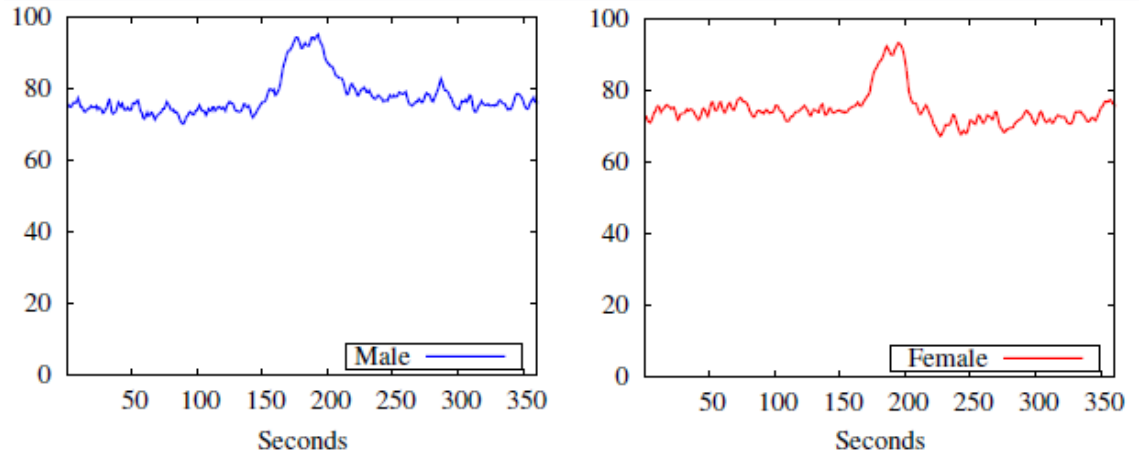


Figura 74 Frecuencia respiratoria fase previa a entrar en fase de fatiga[41]

Mediante los microsistemas electromecánicos o acelerómetros se va a poder medir los cambios en el torso debido a la rotación en la pared del tórax, es decir, movimientos de inclinación y declinación para medir el número de respiraciones por minuto. La señal obtenida por estos dispositivos es la aceleración, caracterizada por ser un vector con una magnitud y una dirección. De los tres ejes (X,Y,Z) solo uno de ellos

[40] [39] Figura obtenida de: Brandy Warwick, Nicholas Symons, Xiao Chen, Kaiqi Xiong (s.f). Detecting Driver Drowsiness using Wireless Wearables.

nos proporcionará la información relativa al movimiento realizado por el diafragma. Sin embargo, vamos a utilizarlos conjuntamente para obtener su inclinación respecto a la dirección de la gravedad. El uso de dos acelerómetros conjuntamente viene motivado por posibles situaciones de insensibilidad debido a rotaciones y traslaciones (tres ejes del acelerómetro paralelos entre sí). Siguiendo esta línea se garantiza que las señales de los acelerómetros durante el movimiento de respiración tienen direcciones opuestas, pudiendo medir la inclinación entre ambos y poder dibujar una señal resultante fruto de la combinación de las seis. No obstante si los acelerómetros no tienen la sensibilidad adecuada, puede ser imposible dicho análisis.

Las pruebas se han realizado con los sensores de la gama *Shimmer* y posteriormente se lleva a cabo el procesado *off-line* de la señal debido a la limitación que presenta en tiempo real desde nuestro dispositivo de pruebas sumado a que no se puede filtrar adecuadamente la señal.

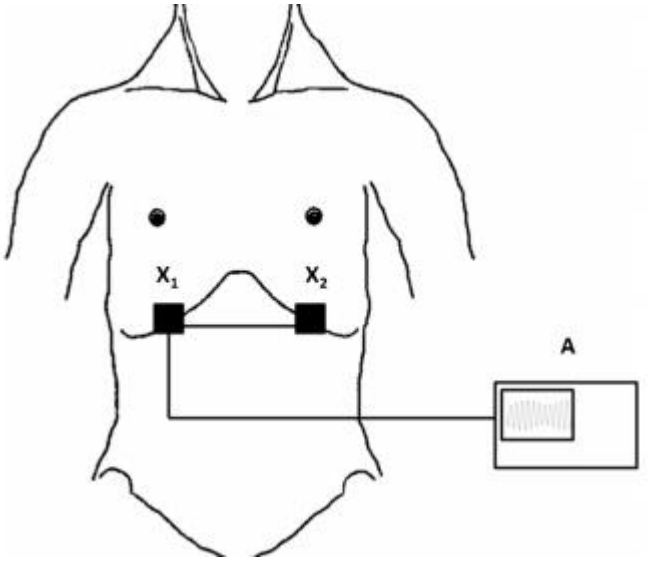


Figura 75 Posición de los acelerómetros[42]

En primer lugar, procederemos a realizar un filtrado paso-alto de la señal a una frecuencia de 1 Hz, y a continuación, un filtrado paso-bajo de 3 Hz con el objetivo de quedarnos con la banda de interés.

[42] Figura obtenida de: Sara Lapi, Federico Lavorini, Giovanni Borgioli, Marco Calzolai, Leonardo Masotti, Massiomo Pistolesi, Giovanni A.Fontana (2013,12 de noviembre). Respiratory rate assessments using a dual-accelerometer device.

```
Wn1=2*3/128;%2*f/fs       %Atento por si cambiamos la frecuencia de muestreo, cambiar el tipo de filtrado para que sea correcto.
N1=4;
[B1,A1] = butter(N1,Wn1);  %Filtrado por defecto paso-bajo

Wn2=2*1/128;%2*f/fs
N2=4;
[B2,A2] = butter(N2,Wn2, 'high');%Filtrado por defecto paso-alto

%Filtramos las señales procedentes de los dos acelerómetros.

X1=filtfilt(B1, A1, X11);
Y1=filtfilt(B1, A1, Y11);
Z1=filtfilt(B1, A1, Z11);

X2=filtfilt(B1, A1, X22);
Y2=filtfilt(B1, A1, Y22);
Z2=filtfilt(B1, A1, Z22);

X1=filtfilt(B1, A1, X1);
Y1=filtfilt(B1, A1, Y1);
Z1=filtfilt(B1, A1, Z11);

X2=filtfilt(B1, A1, X2);
Y2=filtfilt(B1, A1, Y2);
Z2=filtfilt(B1, A1, Z2);
vectorTheta = [];
l=1;
```

Figura 76 *Script Matlab* filtrado

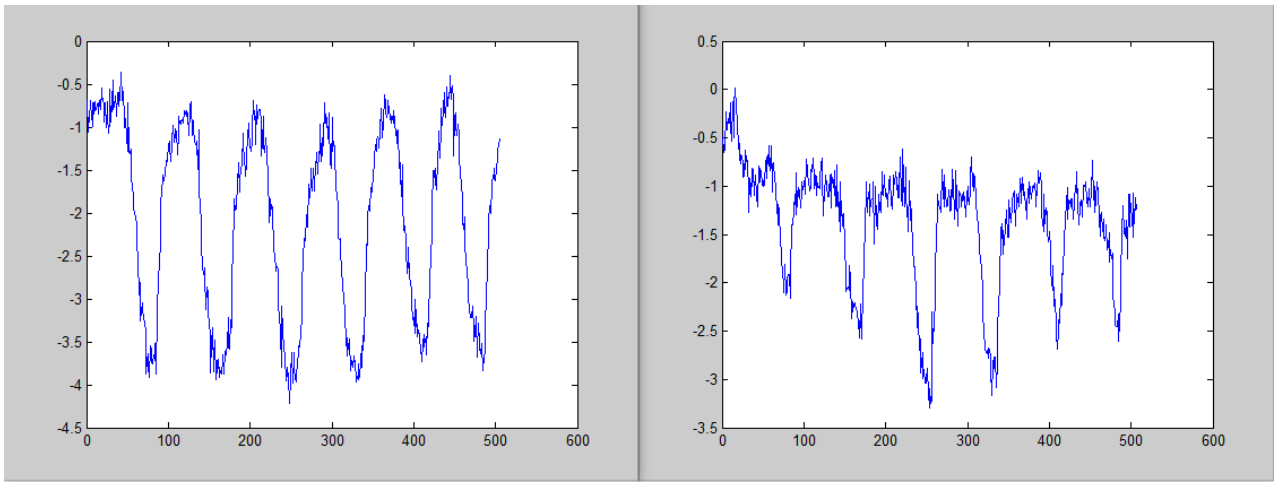


Figura 77 Señal eje Z antes del filtrado

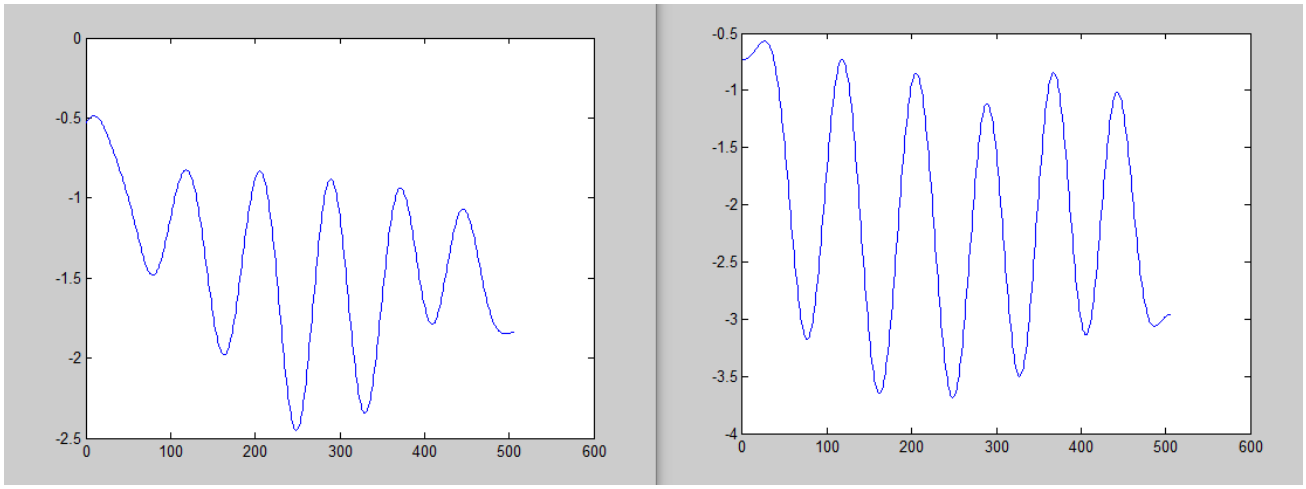


Figura 78 Señal eje Z después del filtrado

Ahora que tenemos las señales limpias de ruido se obtiene el vector diferencia, calculando su módulo como vemos a continuación.

$$|\vec{D}| = D = \sqrt{(g_{1x} - g_{2x'})^2 + (g_{1y} - g_{2y'})^2 + (g_{1z} - g_{2z'})^2}$$

Finalmente, obtenemos el ángulo formado entre los vectores directores de cada uno de los acelerómetros tri-axiales. La inspiración trae como resultado el incremento de este ángulo mientras que la espiración produce el efecto contrario. La relación entre el vector diferencia y el ángulo formado por los vectores g1 y g2 y viene determinado por la siguiente relación:

$$dD = g\sqrt{1 - \frac{D^2}{4g^2}}d\theta \qquad \cos\frac{\theta}{2} = \sqrt{1 - \frac{D^2}{4g^2}}$$

Donde g es :

$$g = \sqrt{g_{2x'}^2 + g_{2y'}^2 + g_{2z'}^2}$$

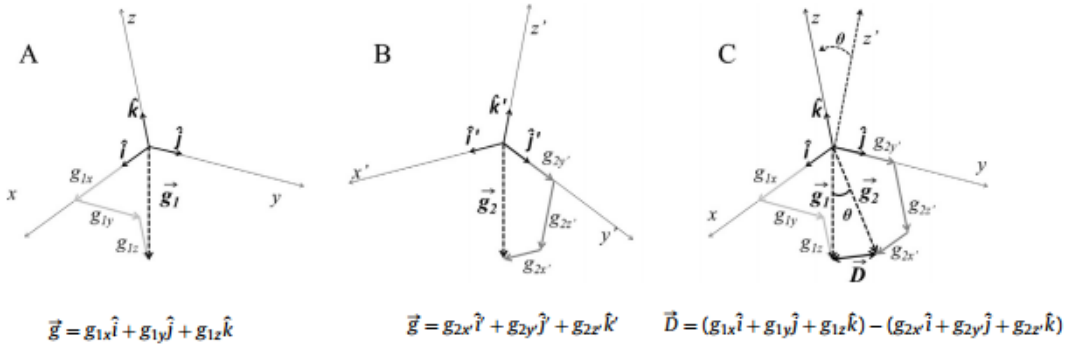


Figura 79 Vectores directores de cada uno de los acelerómetros y vector diferencia[43]

[43] Figura obtenida de: Sara Lapi, Federico Lavorini, Giovanni Borgioli, Marco Calzolai, Leonardo Masotti, Massiomo Pistolesi, Giovanni A.Fontana (2013,12 de noviembre). Respiratory rate assessments using a dual-accelerometer device.

```
%Tenemos que recorrer todo el vector para esto
for i=1:length(X1)

    DX = X1(i)-X2(i);
    DY = Y1(i)-Y2(i);
    DZ = Z1(i)-Z2(i);
    DModulo = sqrt(DX^2 + DY^2 + DZ^2);
    G2Modulo = sqrt(X2(i)^2 + Y2(i)^2 + Z2(i)^2);

    Theta = 2*acos(sqrt(1- DModulo^2/(4*G2Modulo)));
    Theta
   a = Theta;
    vectorTheta(l) = a;
    l=l+1;
end
```

Figura 80 *Script Matlab* cálculo vector diferencia y ángulo *theta*

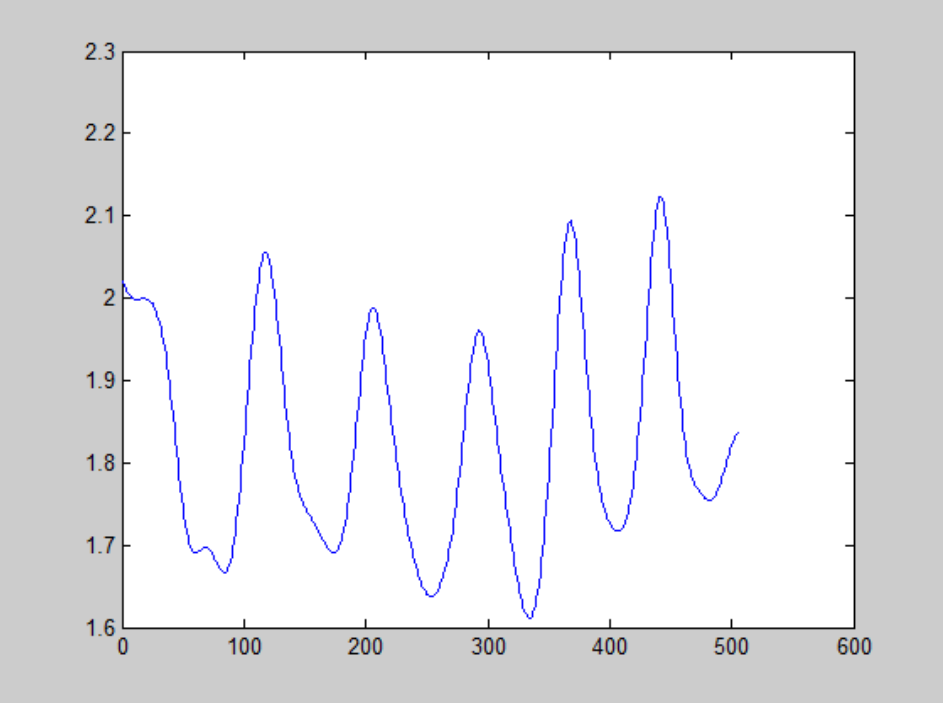


Figura 81 Variaciones producidas en el ángulo *theta* siendo la señal resultante

```
%Contamos los picos de Theta
maximos = [];
p=1
for k=2:(length(vectorTheta)-1)
    if vectorTheta(k)>vectorTheta(k-1) && vectorTheta(k)>vectorTheta(k+1)
        maximos(end+1)=vectorTheta(k);
        vectorK(p) = [k];
        p=p+1
    end
end
```

Figura 82 Búsqueda de máximas en el vector *theta*

Ahora solo falta conocer el instante temporal en el que se encuentra ese máximo y poder calcular la frecuencia respiratoria.

```
P = length(vectorK)
  for j=2:length(maximos)

      a = TimeStamp(vectorK(j))-TimeStamp(vectorK(j-1));
      a= a/1000;
      RespiratoyRate=60/a
  end
figure()
plot(vectorTheta)
```

Figura 83 Cálculo frecuencia respiratoria

Los problemas surgidos para obtener información interesante con este método se deben a la baja precisión de los acelerómetros en el caso de que la respiración no esté muy marcada por una subida y bajada del diafragma.

En la figura 84 se puede observar como no se aprecían los intervalos de inspiración y espiración debido a la baja precisión del sensor y sumado a que la señal está muy contaminada por el ruido. Si el acelerómetro fuera más preciso a pesar de la degradación de la señal se podría distinguir.

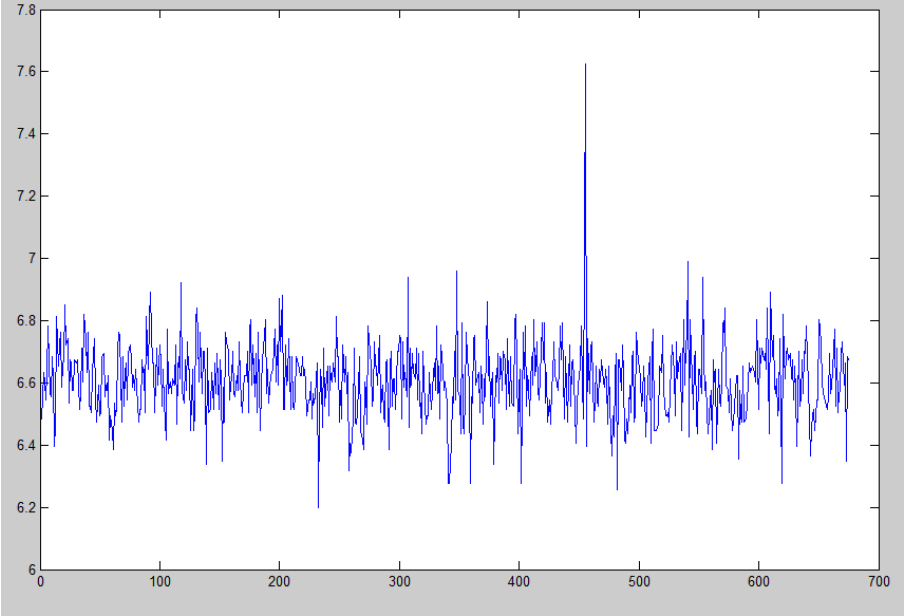


Figura 84 Eje Z sin filtrar

A pesar de realizar un filtrado bastante preciso, resulta imposible determinar cuáles han sido las fases de inspiración y espiración debido a la aparición de picos que no se corresponden con la frecuencia respiratoria real.

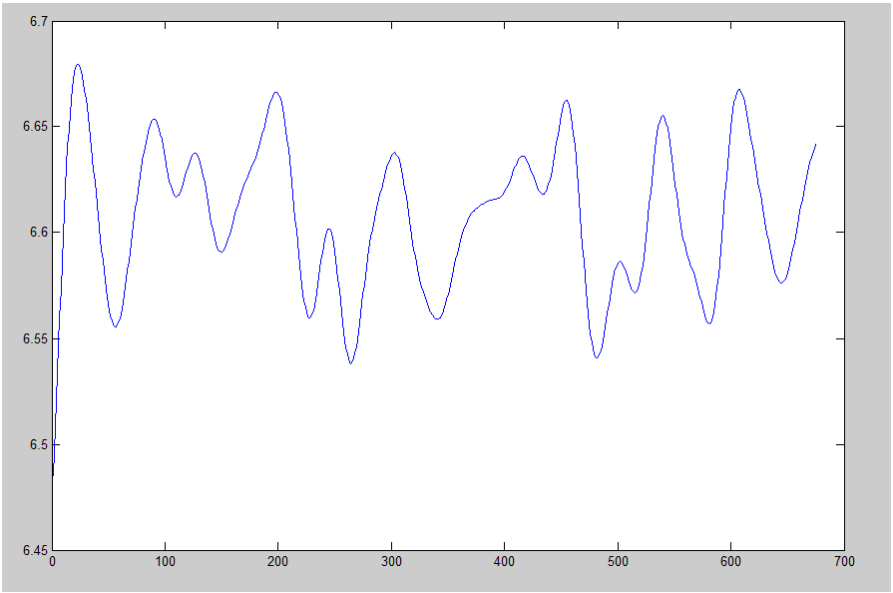


Figura 85 Eje Z filtrada

VI. ANÁLISIS BI-ESPECTAL

Se va a desarrollar un sistema para detectar la hipovigilancia que engloba tanta la somnolencia como la falta de atención mediante el uso del electrocardiograma y electromiogramas. La somnolencia aparece en el conductor con mayor frecuencia en viajes largos y monótonos y la falta de atención cuando manipula un teléfono móvil, GPS o cualquier otro dispositivo electrónico.

Las pruebas realizadas a dos personas (siendo las mismas de las que se han mostrado resultados con el sensor GSR) para el estudio han consistido en la obtención de las señales en distintos entornos, de tal manera que se abarquen las tres categorías de las cuales se desean calcular los momentos de alto orden espectral. En primer lugar se conduce durante 15 minutos aproximadamente sin ningún tipo de distracción. A continuación se utiliza el móvil para contestar a unos mensajes de *Whatsapp* durante alrededor de unos 5 minutos. También se puede interactuar con él preguntando acerca de sus intereses, aficiones de tal manera que siga aumentando su distracción cognitiva. Finalmente, conducir durante un tiempo prolongado y monótono de tal manera que los signos de fatiga aparezcan antes.

Las características de las señales son extraídas con técnicas espectrales de orden superior que consisten en calcular los momentos de orden superior. Los momentos son una medida cuantitativa de un conjunto de puntos y los acumulados son una combinación lineal de los momentos. Las motivaciones para utilizar esta técnica y no la FFT residen en:

1) La información de una señal reside principalmente en la fase y no en la amplitud de su FFT.
2) Los espectros de orden superior conservan la amplitud y la fase de la FFT a diferencia de la densidad espectral de potencia.
3) Los espectros de orden superior son invariantes sirviendo como características invariables para el reconocimiento de patrones.
4) Cuanto mayor sea el orden del espectro más tenderá a cero el valor esperado del ruido de Gauss teniendo una alta inmunidad a este tipo de ruido y siendo más robusto

Los momentos son los siguientes:

1) Primer momento: Es la media de la señal.

$$m_1 = E[X]$$

2) Segundo momento: Es la autocorrelación de la señal.

$$m_2(\tau_1) = E[X(K) \cdot X(K+\tau_1)] = c_2(\tau_1)$$

3) Tercer momento:

$$m_3(\tau_1, \tau_2) = E[X(K) \cdot X(K+\tau_1) \cdot X(K+\tau_2)] = c_3(\tau_1, \tau_2)$$

4) Cuarto momento:

$$m_4(\tau_1, \tau_2, \tau_3) = c_4(\tau_1, \tau_2, \tau_3) + c_2(\tau_1) \cdot c_2(\tau_3 - \tau_2) + c_2(\tau_2) \cdot c_2(\tau_3 - \tau_1) + c_2(\tau_3) \cdot c_2(\tau_2 - \tau_1)$$

La FFT del segundo momento nos proporciona la densidad espectral y del tercer momento el bi-espectro, herramienta útil para la detección de no linealidades cuando la señal de interés no es un proceso gaussiano, que se expresa de la siguiente manera:

$$B(f_1, f_2) = \sum_{\tau_1=-\infty}^{\infty} \sum_{\tau_2=-\infty}^{\infty} c_3(\tau_1, \tau_2) e^{-j(\omega_1 \tau_1 + \omega_2 \tau_2)} \qquad B(f_1, f_2) = E[X(f_1)X(f_2)X*(f_1+f_2)]$$

El bi-espectro para que sea útil tiene que ser calculado en las regiones no redundantes, es decir en su dominio director, siendo este $0 < f_1 < f_1+f_2 < 1$. La región no redundante es la marcada por **Ω**.

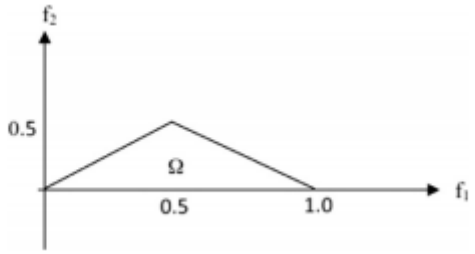


Figura 86 Dominio válido del bi-espectro[44]

[44] Figura extraída de: Nikias & Raghuveer, 1987

Las características que vamos a extraer del bi-espectro de las señales son las siguientes:

1) Suma logarítmica del bi-espectro de las señales.

$$H_1 = \sum_{\Omega} \log\left(|B(f_1, f_2)|\right)$$

2) Suma logarítmica de las amplitudes de los elementos de la diagonal del bi-espectro.

$$H_2 = \sum_{\Omega} \log\left(|B(f_k, f_k)|\right)$$

3) El momento del primer orden espectral de las amplitudes de la diagonal del bi-espectro.

$$H_3 = \sum_{k=1}^{N} k \log\left(|B(f_k, f_k)|\right)$$

Mediante el análisis discriminante lineal (LDA) y cuadrático (QDA) se han obtenido una precisión de un 96.75% y 92.31% de fiabilidad con el estado real del sujeto para señales ECG y EMG respectivamente.

Recopilados todos los datos se procede al desarrollo del script para analizarlos utilizando las características del bi-espectro H1, H2 y H3 y empleando los clasificadores QDA y LDA que delimitarán las áreas en las que se podrá determinar el estado en el que se encuentra el sujeto cuando se realiza la prueba y cuáles son las fronteras entre los tres estados:

Las figuras 87 y 88 muestran los resultados de la suma logarítmica de bi-espectro de las señales y las diferentes regiones determinadas por los clasificadores cuadráticos y lineales respectivamente.

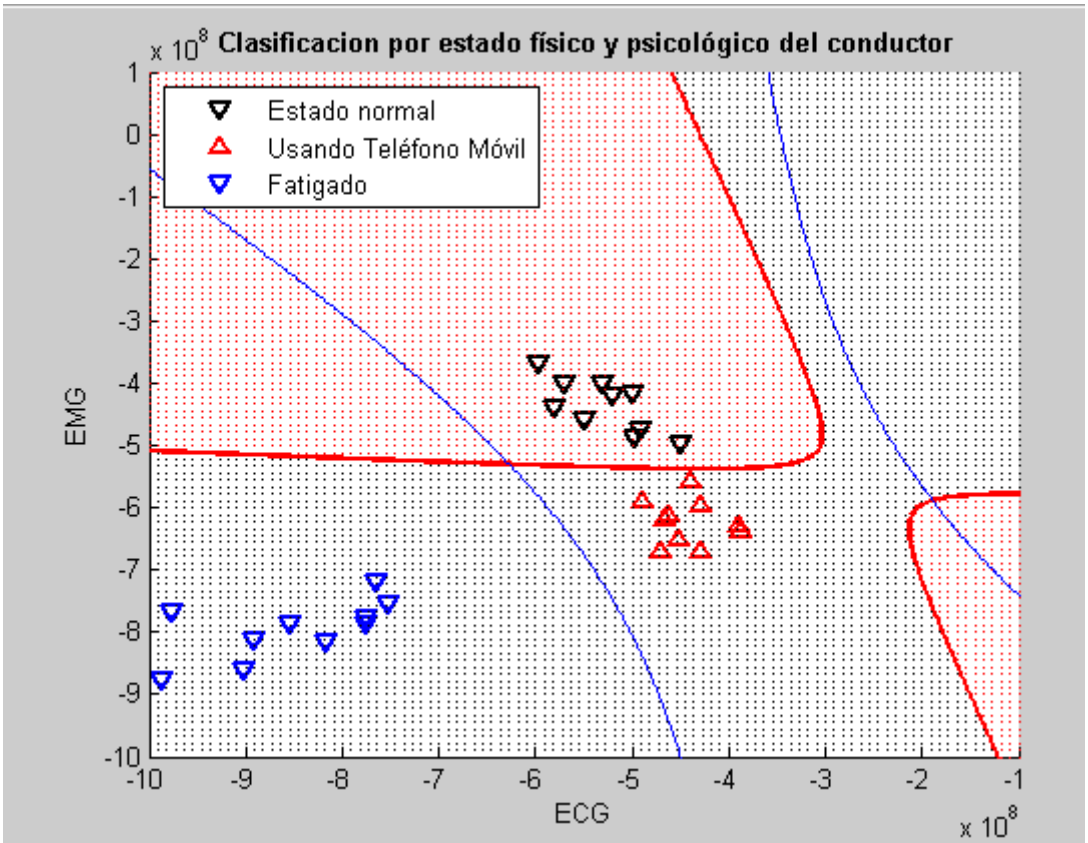


Figura 87 QDA H1

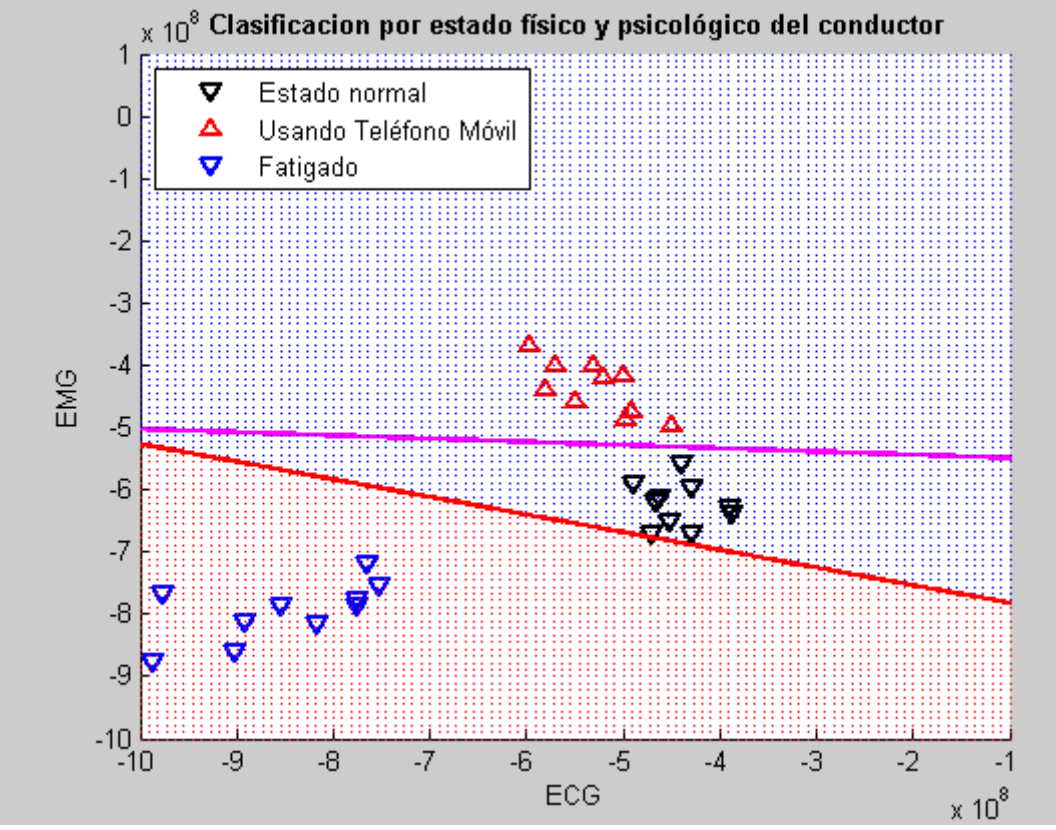


Figura 88 LDA H1

Las figuras 89 y 90 muestran los resultados de la suma logarítmica de los elementos de la diagonal del bi-espectro de las señales y las diferentes regiones determinadas por los clasificadores cuadráticos y lineales respectivamente.

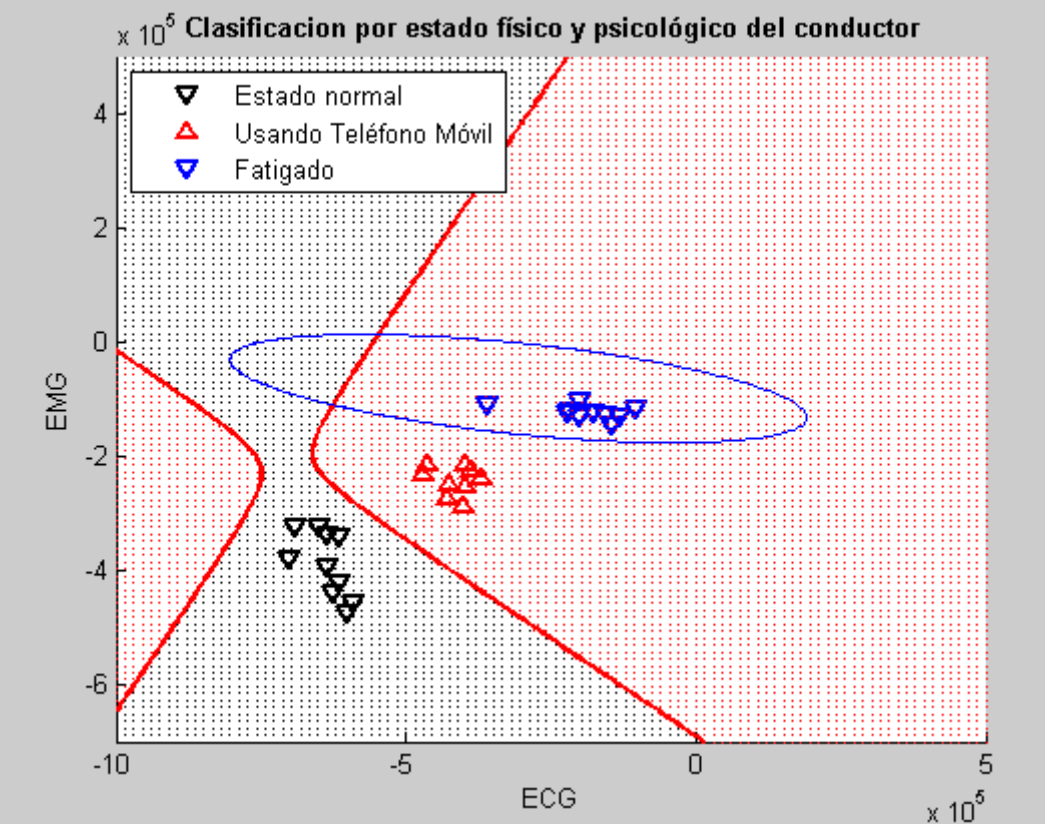


Figura 89 QDA H2

Las figuras 91 y 92 muestran los resultados de la suma logarítmica de los elementos de la diagonal del bi-espectro de las señales y las diferentes regiones determinadas por los clasificadores cuadráticos y lineales respectivamente.

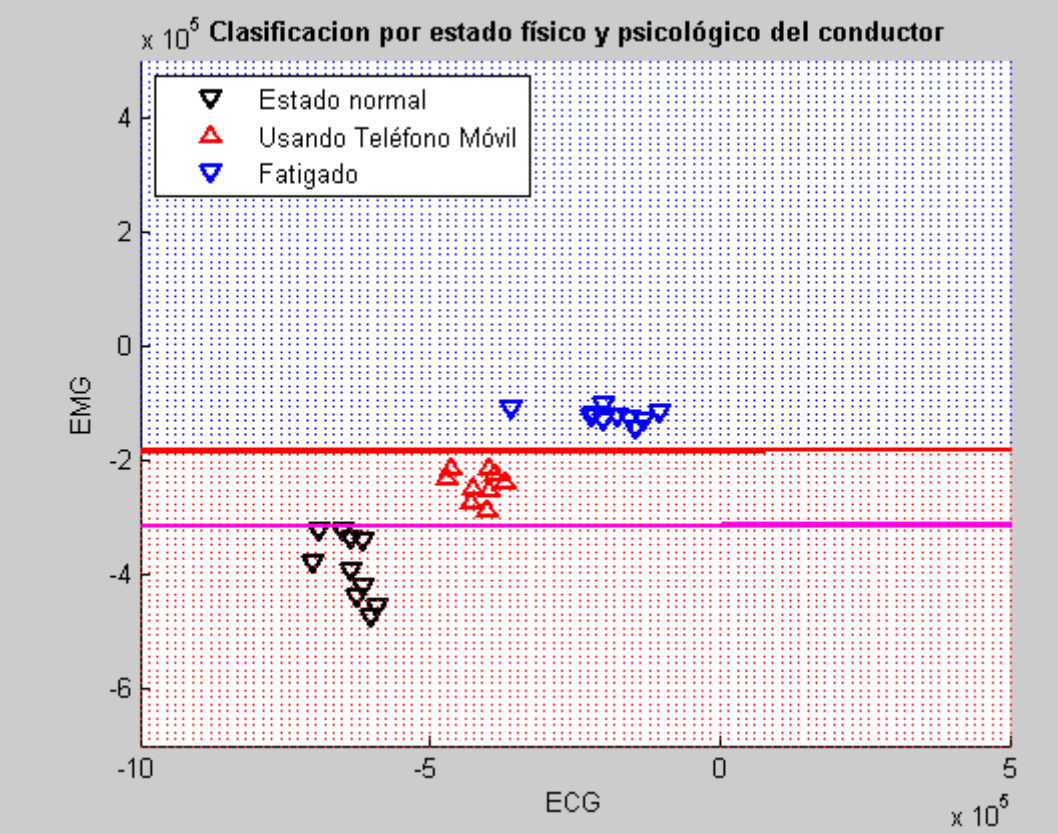


Figura 90 LDA H3

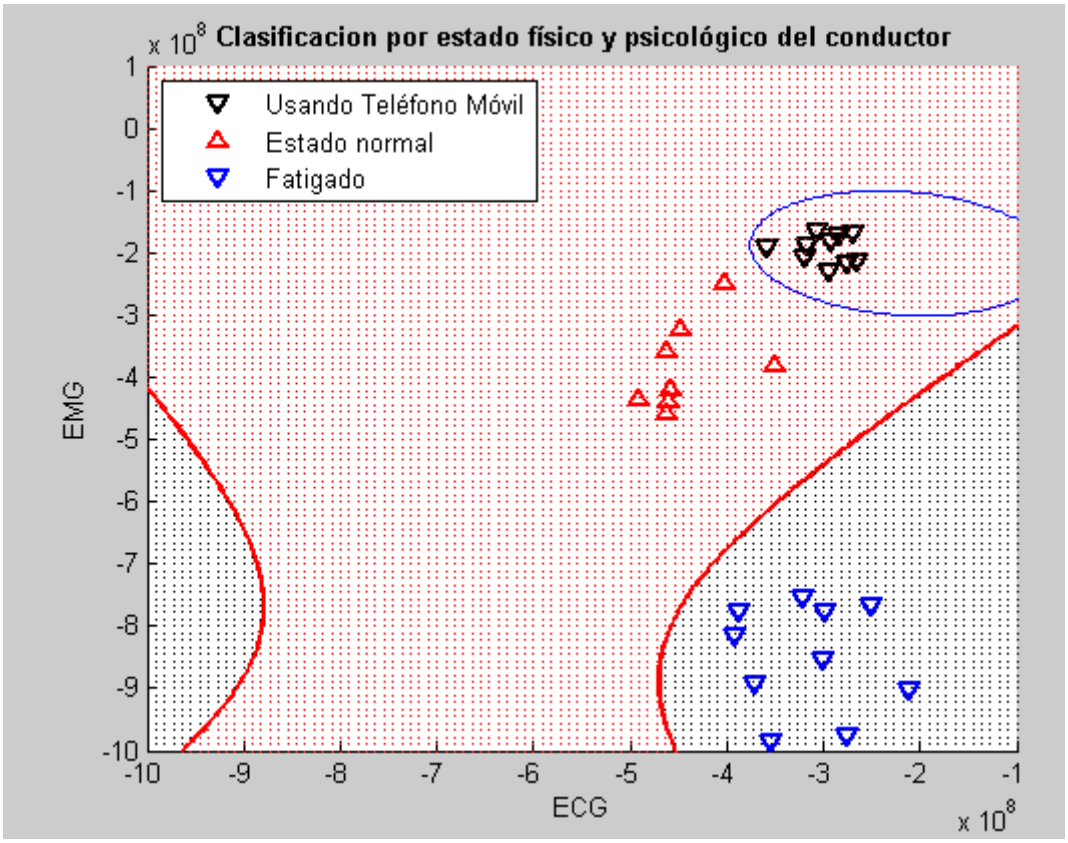


Figura 91 QDA H3

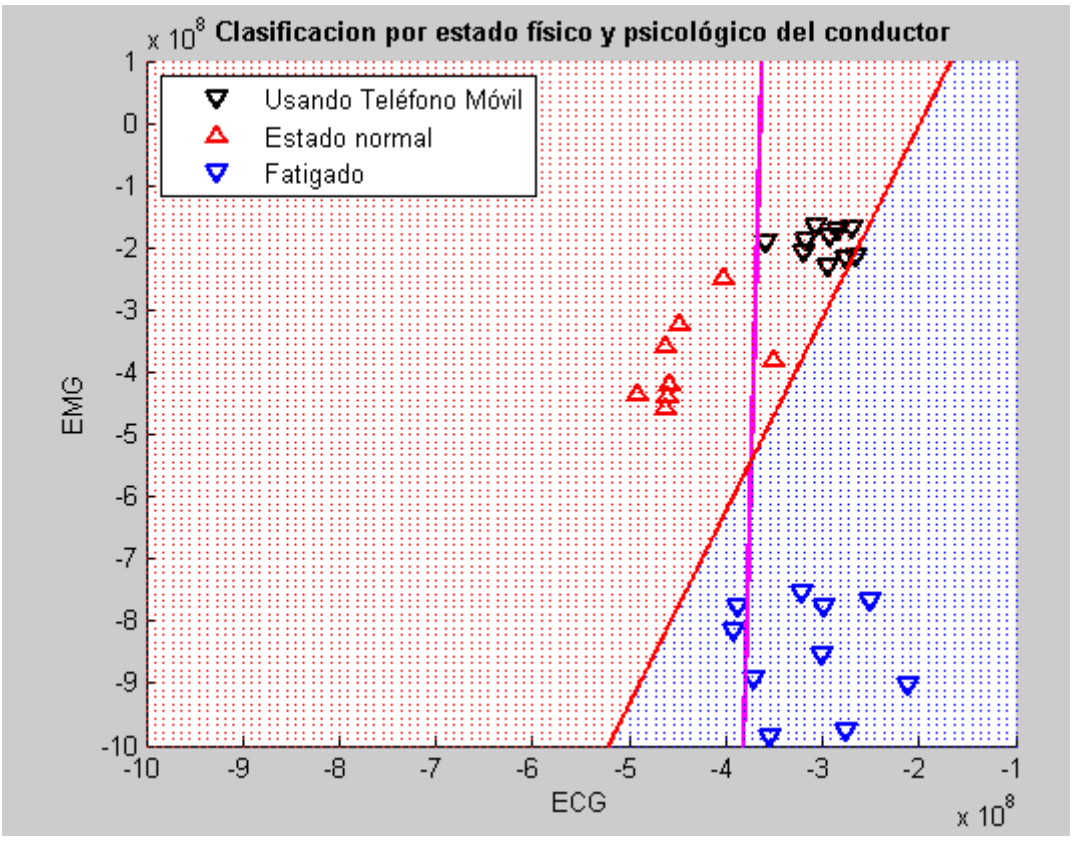


Figura 92 LDA H3

Los datos aparecen representados en las gráficas sin hacer distinción de cuáles son de la persona 1 y persona 2 en cada uno de los estados. Se observa cómo los resultados correspondientes al estado normal y cuando se está utilizando el teléfono móvil se encuentran bastante cercanos pero la diferenciación entre ellos es bastante factible. No obstante, puede darse el caso en el que nuestro clasificador pueda dar un error como ocurre en la figura 92 en el que la clasificación es incorrecta. Sin embargo, el clasificador cuadrático es bastante más preciso y no ofrece a priori este tipo de problemas. La diferenciación cuando la persona se encuentra fatigada es más sencilla debido a que la distancia entre los momentos calculados es mayor y sumado a la precisión dada por los clasificadores hace que no se dé ningún caso lugar a equivocación.

VII. BALANCE ECONÓMICO

El cálculo de coste que tiene la realización de la aplicación y estudio depende de muchos factores. En primer lugar la complejidad, como ha sido este caso, implicando un elevado número de horas.

La realización del trabajo ha supuesto alrededor de 410 horas, buscando información y documentación, escribiendo código, realizando pruebas, redactando la memoria, etc…

Considerando al desarrollador como autónomo y que aproximadamente el salario como programador junior es de 10 euros por hora, el gasto de mano de obra ascenderá a 4100 euros. El tiempo estimado sería 2 meses y medio aproximadamente, trabajando 5 días a la semana, 8 horas.

La cuota de la seguridad social en régimen de autónomos es de 264,44 euros al mes. Durante 2 meses y medio ascendería a un total de 661.11 euros.

Los sensores de la plataforma *Shimmer* utilizados tienen un coste cercano a los 3000 euros y no se ha requerido la adquisición de ningún tipo de licencia.

El presupuesto estimado sería el siguiente:

Mano de obra	**4100**
Seguridad social	**661.11**

Coste Sensores	**3000**
Total	**7761.11**

Tabla 7. Presupuesto estimado

VIII. CONCLUSIONES

La utilización de las TIC en todos los campos que afectan a la monitorización del estado físico y psicológico de las personas es cada vez más frecuentes y necesario teniendo un amplio abanico de parámetros que se deben gestionar, en nuestro caso, durante la conducción. El presente trabajo tenía como objetivo el estudio y desarrollo de una aplicación para Android con el objetivo de monitorizar en tiempo real la situación y nivel de atención de un conductor mediantes sensores fisiológicos *Shimmer* y el almacenamiento de las señales en ficheros para su posterior procesado y clasificación de manera más precisa con el programa *MatLab.*

En primer lugar se ha llevado a cabo un estudio acerca de los dispositivos que se van a utilizar, entendiendo el funcionamiento que siguen, sus configuraciones y las posibles aplicaciones que pueden tener. Seguidamente, se ha buscado información para el desarrollo de aplicaciones compatibles en *Android* utilizando la librería proporcionada por el fabricante de los sensores que incluye clases para procesado de señales. La elección de este sistema operativo viene debido a su gran auge en la última década, su funcionalidad, versatilidad y portabilidad.

En segundo lugar, se ha hecho un estudio de las señales que se obtienen de los sensores a partir de las publicaciones en revistas de carácter científico-técnico, sus posibles procesados y la viabilidad de la implementación de los mismos. Por otra parte se ha llevado a cabo el estudio de método de clasificación lineal y cuadrática basada en los momentos de alto orden espectral.

Durante el desarrollo ha sido imprescindible el estudio de conceptos básicos y avanzados de *Android*, de la librería *Shimmer* y de *Matlab,* recurriendo a las páginas oficiales y adquiriendo una base técnica lo suficientemente amplia para comenzar su desarrollo. La aplicación ya fue realizada en un proyecto fin de carrera anterior por lo que se mejoró y añadieron las clases necesarias para el procesado de las señales en tiempo real, nuevas funcionalidades y en el caso del electrocardiograma su procesado desde la propia aplicación de manera *off-line.*

El trabajo con la señal ECG ha sido relativamente sencillo debido a su alta precisión para la detección de los picos en el complejo QRS, midiendo la distancia entre ellos y generando la señal HRV (*heart rate variability*) y la HR (*heart rate)* o frecuencia cardiaca. A partir del vector de tiempos de RR se calcula su FFT y su PSD con el objetivo de calcular el *ratio* siendo el cociente entre la baja y alta frecuencia. De manera *off-line* se calcula su bi-espectro y se extraen sus características y mediante clasificadores se establecen cuáles son las regiones y fronteras entre los estados del individuo.

El trabajo con la señal EMG ha sido más complicado debido a la gran cantidad de ruido que trae la señal y la limitación que tenemos desde la aplicación tanto en medios como en tiempo para llevarlo a cabo sumado a que es necesario utilizar una tasa de frecuencia bastante alta (mayor de 128 Hz). Además, su sensibilidad es menor por lo que hemos declinado la opción de realizar un procesado de manera *online*. En cambio de manera *offline,* mediante un filtrado más exhaustivo y calculando su FFT y PSD se aprecia que la banda de energía de la señal se desplaza de la media frecuencia cuando el individuo se va fatigando. Finalmente, al igual que en la señal ECG extraemos sus características mediante el bi-espectro y establecemos las regiones y fronteras entre cada uno de los estados.

El trabajo con el sensor GSR ha sido más sencillo, estableciendo un umbral que si se baja se lanzará una notificación al conductor para informarlo que puede quedarse

dormido en un espacio de tiempo bastante breve. De manera *off-line* hemos procedido a la representación gráfica de las señales tomadas en las pruebas realizadas y hemos comprobado la veracidad de lo expuesto en los artículos.

Por último el uso de dos acelerómetro para calcular la frecuencia respiratoria no ha resultado como se esperaba debido a que la señal adquiere bastante ruido debido al movimiento del diafragma y su baja sensibilidad impide capturar la aceleración y deceleración de los movimientos de inspiración y espiración si estos no son muy marcados, es decir, si es una respiración que implica un leve movimiento del diafragma no va a ser detectado.

Durante el desarrollo de esta memoria, se han establecido los objetivos principales para este trabajo fin de grado y las fases y métodos seguidos durante su desarrollo. A continuación se ha hablado de los factores que afectan a la seguridad en la conducción y al estado físico del conductor. Seguidamente se ha abordado un estudio en profundidad de los sensores utilizados y la arquitectura y versiones del sistema operativo. Finalmente se han explicado con detalle el tipo de señales con las que se han trabajado, los procesados, pruebas, *scripts* y clases programadas.

Con este trabajo de investigación espero contribuir a motivar a los investigadores a que desarrollen un interés y un gusto por la detección de la somnolencia en la conducción mediante la utilización de sensores fisiológicos. En este estudio de caso la detección de fatiga mediante señales ECG, EMG, GSR y frecuencia respiratoria, queda mucho por estudiar a fondo y desde diferentes puntos de vista como pueden ser el uso de *software* para el análisis de señales en tiempo real con elementos más precisos e intentar aplicar algoritmos más complejos sin limitación por la capacidad de nuestro microprocesador. Para ello es imprescindible conocer mejor los tipos de señales y la relación que guardan con la somnolencia y nuevas líneas de investigación, a fin de contar con un mayor número de datos. Espero que este trabajo de investigación sirva también, a modo de estudio incipiente, para concienciar de la importancia de este campo de la seguridad en la conducción que afecta a toda la sociedad siendo una de las causas más influyentes en los accidente y para conocer mejor la plataforma de desarrollo Android y las bibliotecas proporcionadas por los fabricantes de los

dispositivos, como elemento imprescindible. Espero además que pueda servir de ayuda como punto de partida a quien se interese por investigar el tema en un futuro.

IX. BIBLIOGRAFÍA Y WEBGRAFÍA

IX.1. Bibliografía

- Khadmaoui, Amine (2014). *Desarrollo de una aplicación para el control de sensores fisiológicos mediante un dispositivo Android.* Proyecto fin de carrera, Ingeniero de telecomunicación, Universidad de Valladolid, España.

-Kurniawan, Hindra, Maslow, Alexadre y Pechenizkiy,Mykola.(Sin fecha). Stress Detection from Speech and Galvanic Skin Response Signals.

- López Ramírez, Jorge Hernán (2006). *La alegría de leer el cardiograma.* Bogotá: Médica Celsus.

-Medicore (Sin fecha). *Heart Rate Variability Analysis System: Clinical Information.*

- Stein, Phyllis, Kleiger, Robert y Rottman, Jeffrey (1997). Differing Effects of age on heart rate variability in men and women. (pp.302-305).

-Sundaraj, Kenneth, Murugappan, Murugappan y Sahayadhas, Arun (2012,7 de Diciembre). Detecting Driver Drowsiness Based on Sensors: A Review. (pp. 16937-16949).

-Zocchi,C, Giusti,A y Rovetta,A (Sin fecha). Biosensor for microsleeps detection during drive simulations.

VI.2 Webgrafía:

- Bonjyotsna,A y Roy,S (2014,Mayo). Correlation of Drowsiness with Electrocardiogram: A Review. 3(5). (pp.9538-9544).

- García Ramiro, Pedro Antonio (2013). *Los seis segundos del ECG. El generador de ritmos cardiacos estáticos y dinámicos.* Universidad de Jaén. Extraído el 31 de mayo de 2015 desde http://www4.ujaen.es/~pgramiro/

-Instituto de Ingeniería Eléctrica y Electrónica: (http://www.ieee.org). Extraído el 23 de marzo de 2015.

-Jorge, Miguel (2012, mayo, 25). *Eric Schmidt: "los gobiernos representan la mayor amenaza para Internet".* Extraído el 31 de mayo de 2015 desde: http://hipertextual.com/2012/05/eric-schmidt-internet-gobiernos

-Ritmo cardiaco recomendado y avalado por Medicina21 y por PortalesDeMedicos.com (2015, 30 de Mayo). Extraído el 31 de mayo de 2015 desde: http://www.my-ekg.com/como-leer-ekg/ritmo-cardiaco.html.

- Revista: *"Science Direct"* (http://www.sciencedirect.com) Extraído el 19 de marzo de 2015.

- "Shimmer" (http://www.shimmersensing.com) Extraído el 17 de marzo de 2015

-"Smartphone OS Market Share Q1 2015" (2015). Extraído el 31 de mayo de 2015 desde: http://www.idc.com/prodserv/smartphone-os-market-share.jsp

ANEXOS

Anexo I. Instalación librería XYPlot

Como se ha comentado con anterioridad en el capítulo V.1.2 se emplea la librería XYPlot para la representación gráfica de los ratos obtenidos después del procesado *off-line* de la señal ECG.

Esta librería genera en ocasiones una serie de errores al importarlas desde el *Android Studio* por lo que vamos a describir los pasos necesarios para incluirla en nuestro proyecto.

1) Descargamos la librería de la página oficial http://androidplot.com/download/

2) Una vez realizada su descarga, las incluimos en la carpeta *libs* de nuestro proyecto que contiene las librerías externas. Si se importa desde fuera dará error debido a que no se encuentra en el mismo *workspace.* Por norma general la carpeta existe por defecto pero en caso de no estar la creamos sin ningún problema. A continuación pulsamos *f5* refrescando el entorno de desarrollo y las añadimos al *build path* de tal manera que ya son ficheros que puedan utilizarse en nuestro proyecto.

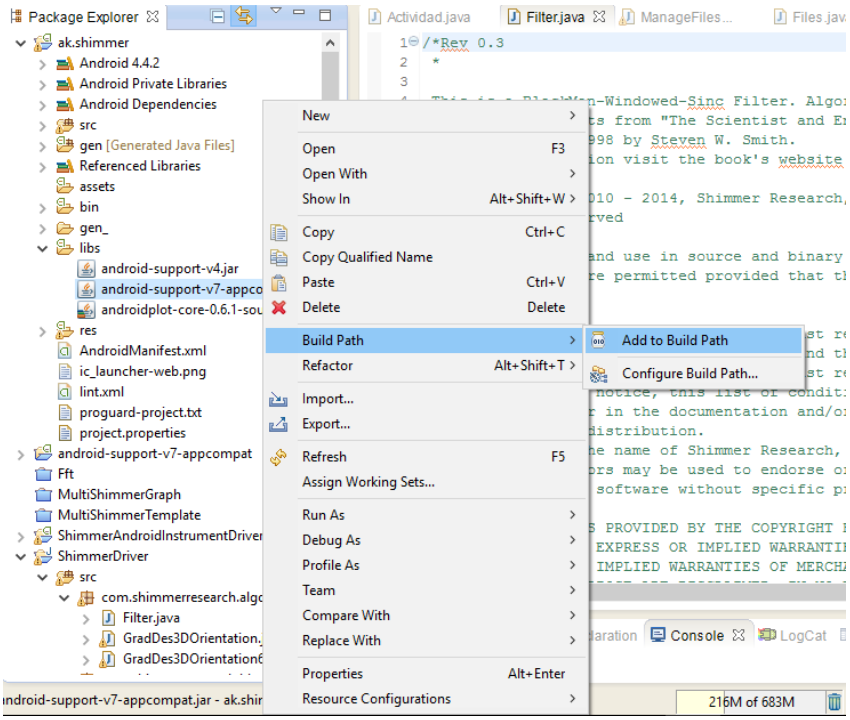


Figura 93 Añadir biblioteca al *build path*

3) Finalmente pulsamos en *Project->Build Project* y los errores del compilador desaparecerían.

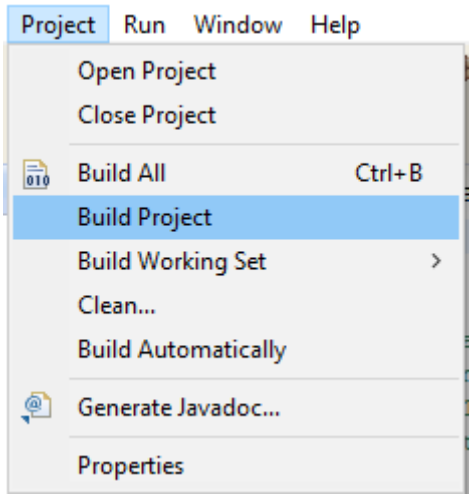


Figura 94 Ruta a seguir para construir el proyecto