Universidad Zaragoza
1542

Trabajo Fin de Grado

Diseño de sistema electrónico para HMI basado en reconocimiento de gestos manuales.

Electronic system design for HMI based on hand gesture recognition.

Autor

Víctor Malumbres Talles

Director/es

Manuel González Bedia

José María López Pérez

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

Año Académico: 2017-2018

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. Víctor Malumbres Talles,

con nº de DNI 78775663L en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado en Ingeniería Electrónica y Automática, (Título del Trabajo) DISEÑO DE SISTEMA ELECTRÓNICO PARA HMI BASADO EN RECONOCIMIENTO DE GESTOS MANUALES,

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 21 de Septiembre de 2018

Fdo: Víctor Malumbres Talles

DISEÑO DE SISTEMA ELECTRÓNICO PARA HMI BASADO EN RECONOCIMIENTO DE GESTOS MANUALES

RESUMEN

El objetivo principal de este proyecto es diseñar un sistema electrónico que establezca una comunicación no verbal basada en gestos manuales entre usuario y máquina, con el propósito de servir como interfaz de usuario en aplicaciones de computador y/o smartphone.

Para ello, en primer lugar, se lleva a cabo una revisión de los diferentes tipos de interfaces usuario-máquina existentes, junto con una revisión de diferentes periféricos para PC, videoconsolas y otros dispositivos electrónicos relacionados con movimientos manuales. Con el fin de marcar un punto de inicio para el proyecto, además de conocer la tecnología relacionada con el área del proyecto.

Después de la revisión del estado del arte, se inicia un análisis de los gestos manuales a utilizar en el proyecto para conocer la características necesarias, magnitudes físicas, para su identificación. Realizada esta fase, se crea un prototipo cableado para la instrumentación de las magnitudes físicas necesarias para iniciar el desarrollo de software de reconocimiento gestual.

En la fase de reconocimiento gestual, se crea software en MATLAB para dicha tarea. El software se conforma de varios scripts en MATLAB que contienen, principalmente, implementado el reconocimiento para gestos estáticos y gestos dinámicos, además de otras funciones útiles en los algoritmos de identificación gestual.

Finalizado el software de reconocimiento gestual, se lleva a cabo un prototipo en placa de circuito impreso (**PCB**) del hardware anteriormente implementado, con el fin de conseguir un prototipo más manejable y cómodo para el usuario.

Para concluir el proyecto, se realiza una pequeña aplicación multimedia en Python basada en el reproductor de código abierto VLC Media, que junto con la identificación gestual desarrollada en el proyecto, sirve para crear el tipo de interfaz usuario-máquina que se buscaba, cumpliendo el objetivo principal del proyecto.

Índice

Índice correspondiente a Anexos

Índice de figuras

Índice de tablas

Índice de extractos

1. INTRODUCCIÓN.

Este trabajo de Fin de Grado en Ingeniería Electrónica y Automática se realiza en colaboración con el dpto. de Informática e Ingeniería de Sistemas junto al dpto. de Electrónica y Comunicaciones.

Este proyecto no está basado en ningún otro trabajo previo, por lo que inicia una nueva temática de estudio.

Como impulso para el desarrollo del proyecto expresamos que el gran desarrollo tecnológico actual está implementado sistemas electrónicos capaces de fusionar la tecnología con el cuerpo humano (*cyborg*), con objetivos como la sustitución sensorial en pérdidas sensoriales o de órganos y extremidades en personas.

Como ejemplo, presentamos el caso del artista *Neil Harbisson* que a través de su modificación corporal puede percibir colores y frecuencias auditivas, imperceptibles por el resto de los seres humanos, a través de una pequeña antena localizada en su cabeza. Acercándonos a mundo de ciencia ficción.

Añadimos que no sólo tienen aplicaciones médicas, sino también han tenido cabida en el arte y en el ámbito militar.

Lo expresado anteriormente, junto con el culto del autor del proyecto por los entornos creados en la literatura y cine de ciencia ficción, como *Blade Runner, Matrix,* etcétera. Han servido de motivación principal a la hora de iniciar este proyecto.

Por lo tanto, nuestro proyecto de título *Diseño de sistema electrónico para HMI basado en reconocimiento de gestos manuales*, presenta como objetivo principal la implementación de un dispositivo electrónico que sirva como periférico HMI con el fin de establecer una interfaz entre usuario y máquina basada en una comunicación no verbal vía gestos manuales.

Expresamos que este proyecto puede presentar gran relevancia práctica en un futuro al establecer una forma novedosa de comunicación con la tecnología para el usuario medio, pero con una posible adaptación para personas con movilidad reducida o con fines similares a los realizados por la sustitución sensorial.

Tras realizar la revisión bibliográfica, apartado *2. ESTADO DEL ARTE*, el desarrollo de este proyecto ha seguido una metodología de trabajo que recoge cuatro fases.

Estas fases se conforman de *análisis de la problemática del proyecto, implementación de soluciones técnicas a la problemática del proyecto*, *implementación del prototipo final,* y, por último, *resultados, conclusiones y mejoras.*

El *análisis de la problemática del proyecto* se recoge en el apartado 3. *ANÁLISIS GESTUAL Y GENERACIÓN DE UNA BIBLIOTECA DE GESTOS,* donde se recoge el análisis gestual realizado, la elaboración de la biblioteca de gestos a utilizar en el proyecto, junto con los problemas a solventar en las fases de diseño de *Hardware* y *Software.*

Respecto a la *implementación de soluciones técnicas a la problemática del proyecto*, expresamos que se conforma de los apartados *4 HARDWARE Y FIRMWARE DE INSTRUMENTACIÓN* y *5. SOFTWARE DE RECONOCIMIENTO GESTUAL.* Donde se expresan el desarrollo de diseño *Hardware* & *Software* y las soluciones técnicas implementadas.

En la fase de *implementación del prototipo final,* se forma de los apartados *6. DISEÑO DE PROTOTIPO* y *7. APLICACIÓN DEL PROTOTIPO COMO INTERFAZ NUI.*

Por último, la fase relacionada con *resultados, conclusiones y mejoras* se conforma de los apartados *8. CONCLUSIONES* y *9. MEJORAS Y FUTUROS TRABAJOS.*

A continuación, se va a presentar una pequeña síntesis de los apartados que conforman este documento, para facilitar así su revisión.

En el primer apartado, *2. ESTADO DEL ARTE,* conlleva una recopilación de diferentes periféricos que basan su funcionamiento en movimientos manuales, además de revisar la tecnología existente en relación con la aplicación que se va a desarrollar en el proyecto.

Respecto al apartado *3. ANÁLISIS GESTUAL Y GENERACIÓN DE UNA BIBLIOTECA DE GESTOS* se presenta el tipo de comunicación que se busca conseguir, a través de gestos manuales, entre usuario y máquina. Además, se exponen el conjunto de gestos elegidos para el proyecto, junto con los problemas que se presentan, de forma previa a la implementación Hardware & Software, en el reconocimiento gestual.

En *4. HARDWARE Y FIRMWARE DE INSTRUMENTACIÓN* se desarrolla la implementación Hardware llevada a cabo para el prototipo cableado, destacando principalmente los sensores utilizados para la medida de las principales magnitudes físicas necesarias para la identificación gestual, incluyendo las ecuaciones matemáticas a implementar para la extracción de estas magnitudes en las unidades que se requieran. En este apartado se adjuntará en forma de flujograma la estructura del Firmware del prototipo cableado.

En cuanto al apartado *5. SOFTWARE PARA RECONOCIMIENTO GESTUAL,* se va a desarrollar todo lo relacionado con el Software que se encargará del procesamiento y análisis de la información obtenida del prototipo cableado vía puerto serial. Además de dar solución a los problemas planteados, expuestos en apartados anteriores, en relación con la identificación de gestos manuales de tipo estático y dinámico.

En *6. DISEÑO DE PROTOTIPO EN PCB* se presentan el conjunto de etapas que conforman el diseño electrónico de un prototipo para reconocimiento gestual. Estas etapas comprenden especificaciones de diseño, diagrama de bloques, esquemático y *layout.*

En relación con el apartado *7. APLICACIÓN DEL PROTOTIPO COMO INTERFAZ NUI,* se presenta el objetivo del proyecto. Relacionar el estudio e identificación de gestos manuales con el sistema electrónico creado, con el fin de obtener una interfaz más natural e intuitiva (**NUI**) en una aplicación de un computador. En el caso del proyecto se trata de introducir este tipo de interfaz en una aplicación multimedia de reproducción de audio y/o vídeo.

Respecto al apartado *8. CONCLUSIONES* se expresan los resultados obtenido tras la realización del proyecto, junto con los problemas principales surgidos durante el desarrollo del trabajo. Además, se presenta una visión personal del proyecto y posibles aportaciones tecnológicas de este.

Para finalizar con la memoria, en *9. MEJORAS Y FUTUROS TRABAJOS* se presentan una serie de mejoras en cuanto al prototipo electrónico y el software de la aplicación como interfaz NUI. También se expresan posibles trabajos futuros en relación con el proyecto desarrollado.

En cuanto a anexos, presentamos el *ANEXO I. CÓDIGO* donde se muestra el código fuente del Firmware de Instrumentación, los scripts en Matlab en relación con el reconocimiento gestual y finalmente, los scripts en Python de la aplicación multimedia con interfaz NUI. También se encuentra el *ANEXO II. DATASHEETS* donde se encuentran extractos de la información técnica más relevante del material utilizado en este proyecto.

2. ESTADO DEL ARTE.

En primer lugar, definimos **HMI** (Human-Machine Interface) o **Interfaz de Usuario** como el medio por el cual el usuario puede comunicarse con una máquina, computadora o dispositivo. [8].

El objetivo principal de las **Interfaces de Usuario** es crear un entorno amigable para el usuario, dando lugar a un proceso comunicativo, entre máquina y usuario, eficiente, intuitivo y, por lo tanto, sencillo.

Las **HMI** se pueden clasificar teniendo en cuenta los factores de *construcción* de la interfaz y *forma de interacción*.

Respecto a la *construcción*, podemos expresar dos tipos de interfaces.

En primer lugar, la **Interfaz Hardware**. Compuesta por el conjunto de dispositivos y controles que permiten el intercambio de información entre usuario y máquina a través de su introducción por medio de botones, palancas, teclas, entre otros. O a través de su visualización en pantallas, medidores, indicadores, etc.

En segundo lugar, tratamos la **Interfaz Software,** que está compuesta por programas o partes de ellos, que permiten expresar y visualizar información.

En cuanto a la *forma de interacción* con el usuario se presentan tres tipos de interfaz.

Por un lado, tenemos la **Interfaz de línea de comandos (CLI)**, es un tipo de interfaz de usuario donde la información se intercambia a partir de texto, usando un conjunto de caracteres alfanuméricos.

Es uno de los tipos más primitivos de comunicación entre usuario-máquina, por lo tanto, conforma un tipo de interfaz poco intuitiva y natural. Por eso, su uso es muy reducido en la actualidad.

Tras lo expuesto en los párrafos anteriores, presentamos que este tipo de interfaz está muy alejada de la idea a desarrollar en el proyecto.

Por otro lado, presentamos la **Interfaz gráfica (GUI)** como una interfaz que se apoya de contenido visual para llevar a cabo la interacción con el usuario. Es el tipo de interfaz más extendida actualmente, presentándose con periféricos como el ratón, teclado, monitor o pantalla, entre otros.

Se observa que este tipo de interfaz es mucho más intuitiva que la mostrada anteriormente, *interfaz de línea de comandos* **(CLI)**.

Como último ejemplo de interfaz según su interacción con el usuario, presentamos la **Interfaz natural de usuario (NUI)**. Interfaz que integra diferentes dispositivos para convertir la interacción usuario-máquina en una experiencia natural e intuitiva. Este tipo de interfaz presenta pantallas táctiles, reconocimiento de voz, reconocimiento de gestos…

En la actualidad, este tipo de interfaz se presenta en periféricos de varias videoconsolas como son EyeToy de Play Station, Kinect de Microsoft, consola Wii de Nintendo, incluso en smartphones y portátiles de última generación.

Una interfaz de tipo **NUI** es lo que buscamos en nuestro proyecto.

Figura 1. Tipos de interfaces

2.1 Periféricos.

A continuación, una vez definido el concepto de **HMI** o **interfaz de usuario,** se van a exponer diferentes periféricos basados en movimientos manuales.

El conjunto de dispositivos se presentará en orden cronológico, con una breve descripción de su funcionamiento, además de sus aspectos positivos y negativos en relación a su uso por parte del usuario, principalmente.

2.1.1 Ratón.

El ratón es un periférico de entrada para PC utilizado como apuntador útil para el manejo de entornos gráficos (**GUI**) a través de un cursor.

Consiste en uno de los primeros periféricos creados para el uso del computador, y el primer dispositivo basado en movimientos manuales.

El primer diseño data de los años 60 en Silicon Valley. Su invención dio lugar al inicio de las interfaces gráficas (**GUI**), siendo las interfaces por líneas de comando (**CLI**) las presentes en aquella época.

Desde su invención, este periférico ha sufrido una gran evolución. Por eso, en la actualidad existen diferentes modalidades de ratones según sus mecanismos de captura de movimiento (mecánico, óptico, láser…) y su conectividad (PS/2, USB, Bluetooth …).

La estética de un ratón común actual se puede observar en la *Figura2. Ratón*.

Figura 2. Ratón

Aunque existan modelos muy variados, todos tienen en común el mismo principio de funcionamiento. Éste consiste en detectar el movimiento relativo de la mano del usuario en un sistema de dos dimensiones (**2D**) formado por la superficie de apoyo del periférico, los movimientos manuales captado se reflejan por pantalla a través de un puntero o una flecha, sirviendo de guía para el usuario en una interfaz gráfica (**GUI**).

En su mayoría, también presentan dos botones (conocidos como botón derecho e izquierdo) y una rueda, entre estos dos botones, para hacer más eficaz su uso. Simplemente comentamos, la existencia de ratones especiales para su uso en videojuegos, donde se incluyen botones que dan la posibilidad de configurar las acciones a realizar.

Al ser un periférico tan exitoso, ya es visto por el usuario como algo intuitivo y fácilmente manejable. Sin embargo, no puede aprovechar una gran cantidad de movimientos y gestos manuales que puede simplificar la interacción usuario-máquina.

Además, puede presentar problemas de uso para personas afectadas por parálisis cerebrales, distrofia muscular, entre otras enfermedades graves y degenerativas.

En este apartado también cabe destacar otros periféricos similares al ratón como son el *touchPad* y el *trackPad*, principalmente, incluyendo también otros tipos de paneles táctiles.

Respecto al *touchPad*¸ comentamos que es un periférico de entrada, normalmente conocido por su amplio uso en los *laptops*, formado por un panel táctil que permite controlar un cursor con el fin de actuar sobre el computador. También, incluye dos pulsadores para realizar las funciones básicas de *clik* derecho y *click* izquierdo, acciones básicas del ratón.

En cuanto al *trackPad,* presentamos que es un periférico muy similar al *touchPad* ya que también se compone de un panel táctil con el fin de controlar un cursor. Sin embargo, las funciones de *clickear* se incluyen dentro del panel táctil, por lo que este periférico no presenta botonera.

Para finalizar con este subapartado, exponemos que estos últimos periféricos comentados tratan de adaptar el ratón a *laptops*, por eso se han mencionado en este subapartado.

2.1.2 Joystick.

Joystick es un periférico basado en una palanca que gira sobre una base, informando sobre el ángulo girado y la dirección de la palanca.

En un principio, la función buscada para este dispositivo era el pilotaje de aviones, pero evolucionó hasta ser un periférico para computador y videoconsolas.

Su primer diseño, como periférico de computador, data de los años 70. Aunque llevaba ya presente desde los años 20 en los aviones.

Al igual que el ratón, presenta diferentes botones, cuyas funcionalidades pueden variar según el videojuego y las preferencias del usuario.

Se presentan diferentes modelos en las siguientes figuras.

Figura 3. Joystick2

Figura 4. Joystick1

Este periférico es bastante reconocido, pero no ha tenido tanta repercusión como el ratón.

Como aspecto positivo frente el ratón es la adaptibilidad para personas afectadas por enfermedades degenerativas. Sin embargo, al igual que el ratón, sólo capta movimientos en **2D** y no permite aprovechar otros movimientos manuales por parte del usuario.

2.1.3 Periféricos basados en sensores inerciales.

En la actualidad, **Wii** de **Nintendo** es una de las consolas más reconocidas por mostrar periféricos basados en sensores inerciales (Acelerómetros, Giróscopos y Magnetómetros), ya que el control de la consola y de la gran mayoría de videojuegos de la consola es a través de movimientos manuales.

El funcionamiento principal del mando de esta consola es la adquisición de medidas de aceleración lineal y velocidad angular en tres dimensiones (**3D**) que sufre el periférico, para así reconocer los movimientos realizados por el brazo-mano del usuario, ejecutando cambios en el videojuego relacionados con los movimientos ejecutados.

En la siguiente imagen se presenta la estética del principal periférico de la consola Wii.

Figura 5. Mando Wii

La tecnología de esta consola data del año 2008.

A diferencia de los periféricos comentados en los apartados anteriores (ratón y joystick), la consola **Wii** tiene la posibilidad de captar los movimientos manuales del usuario en tres dimensiones (**3D**), presentando una interfaz mucho más natural e intuitiva (**NUI**).

Un aspecto negativo que presenta este periférico es que está diseñado principalmente para la consola de **Nintendo** y sus videojuegos, lo que no nos da accesibilidad para poderlo utilizar en otras aplicaciones como telemanipulación, control del PC, etcétera.

Es verdad que en la comunidad maker (comunidad cuyo lema es **DIY**, *Do It By Yourself* en inglés. Traducido al español: *Hazlo tú mismo*) se han realizado pequeños proyectos con este periférico, pero que no tienen cabida en la realización del proyecto.

Por otro lado, presentamos la existencia de un dispositivo experimental (**IMU_Mouse**), basado en sensores inerciales, cuya finalidad es el control de un computador a través de movimientos de la cabeza, pensado para personas con discapacidades físicas y movilidad muy reducida. Este

periférico data del año 2017, no podemos presentar imagen de su estética ya que no es comercial, pero podemos presentar una imagen del concepto del dispositivo. [1]

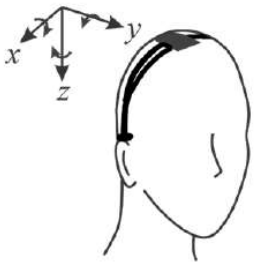


Figura 6. IMU_Mouse

Aunque este periférico no capte movimientos manuales tiene cabida en este apartado por basar su funcionamiento en movimientos de la cabeza medidos a través de sensores inerciales, además de guardar, en cierto modo, una estrecha relación con el proyecto que se quiere realizar.

2.1.4 Periféricos basados en visión artificial.

Los periféricos basados en visión artificial son dispositivos cuyo funcionamiento se basan en el procesamiento y análisis de imágenes.

Por un lado, presentamos el periférico de la consola **PlayStation 2** de **Sony**, **EyeToy**. Este dispositivo se presentó en el año 2003, se forma de una cámara RGB que con técnicas de visión puede reconocer diferentes movimientos con los que el usuario puede interactuar con el videojuego, presentando así una interfaz **NUI**.

Al igual que con los periféricos de **Wii**, **Eyetoy** está diseñado para su uso en consolas **Sony** y en videojuegos específicos.

Por otro lado, presentamos **Kinect** de **Microsoft** que data del 2011. Éste consiste de un periférico para **Xbox 360**, también compatible con sistema operativo **Windows** para PC, basado en una cámara RGB junto a un sensor infrarrojo de profundidad, además de incorporar un conjunto de micrófonos.

A través de este periférico, podemos controlar la consola a través de gestos manuales e incluso usando comandos de voz, presentando una de las interfaces **NUI** más novedosas actualmente.

Presentamos imágenes de la estética de estos periféricos.

Figura 7. EyeToy

Figura 8. Kinect

Uno de los aspectos fuertes de **Kinect** es que no sólo es para videoconsolas, lo que permite un abanico más amplio de uso. De hecho, ha sido utilizada en diversos proyectos de robótica y visión por computador en estos últimos años.

El principal aspecto importante que presentan los periféricos basados en visión es que el usuario no debe llevar ningún artefacto en su cuerpo a diferencia de los periféricos basados en sensores inerciales. Dando la oportunidad al usuario de ejecutar los movimientos necesarios con la mayor naturalidad y comodidad posible.

Sin embargo, el aspecto negativo que presentan estos periféricos es que necesitan unas condiciones de luminosidad adecuadas para una buena adquisición de las imágenes, lo que hace que sean poco robustas en aplicaciones de exterior, donde los periféricos basados en sensores inerciales funcionan mejor. Además, para obtener un buen tracking de los gestos se debe estar siempre delante de la cámara, lo que puede ocasionar alguna ligera molestia dependiendo de la aplicación y localización del periférico.

2.1.5 Periféricos basados en ultrasonidos.

En este apartado presentamos el novedoso proyecto **Soli** de **Google**. [12]

Este proyecto consta de la creación de un chip que conforma un radar de onda milimétrico, con el que es capaz de detectar los gestos realizados con las manos de forma precisa.

En la siguiente imagen se presenta la estética del chip **Soli**.

Figura 9. Soli_Chip

La principal aplicación de este proyecto es la inclusión del chip en periféricos relacionados con realidad virtual, aunque sus desarrolladores también comentan que la identificación de materiales podría ser otra de sus aplicaciones.

A continuación, se muestra en una imagen un resumen de su funcionamiento.

Figura 10. Soli_Radar

Se prevé su salida al mercado entre finales del año 2018 y principio del año 2019.

2.2 Crítica al Estado del Arte

En los subapartados anteriores se ha desarrollado una revisión de los diferentes periféricos existentes que basan su funcionamiento en movimientos manuales. Además, se han añadido aspectos positivos y negativos respecto a cada periférico comentado, con el fin de servirnos como punto inicial para el desarrollo del proyecto.

En general, la mayoría de los periféricos intentan realizar las mismas acciones que plantea un ratón o un mando clásico de videoconsola, control de cursores en menús de interfaces tipo **GUI**. Añadimos que, en el caso de videoconsolas, los periféricos revisados si añaden nuevas funciones dependiendo del videojuego. Sin embargo, en el caso de los periféricos para PC no se plantea ninguna acción complementaria a las aportadas por el ratón o el teclado de un computador.

Tras la revisión del Estado del Arte realizado y el análisis de los aspectos positivos, negativos y funcionalidades de los diferentes periféricos comentados, se observa una *laguna* en relación a las funcionalidades presentes en los periféricos de computador, presentando un camino donde enfocar nuestro proyecto.

3. ANÁLISIS GESTUAL Y GENERACIÓN DE UNA BIBLIOTECA DE GESTOS.

3.1 Introducción.

A modo de introducción, antes de presentar todo el análisis gestual debemos desarrollar los siguientes conceptos relacionados con el tema. [13]

El primer concepto que presentamos es la **comunicación no verbal**, esta consiste en un proceso comunicativo en el que existe envío y recepción de mensajes sin el uso de palabras, a través de indicios, gestos y signos.

Este tipo de comunicación surge con los inicios de la especia humana, antes de la evolución del lenguaje. Aunque sea una de las formas más primitivas de comunicación entre seres humanos, no se encuentra en desuso actualmente, puesto que es el tipo de comunicación más natural e innata al ser humano.

Como ya se ha indicado, este proyecto busca establecer este tipo de comunicación como interfaz entre usuario y máquina, presentando como ventaja principal frente a la comunicación verbal (oral y escrita) un aprendizaje poco extenso y genérico, pudiéndolo utilizar usuarios de diferentes países.

Por otro lado, exponemos la **Kinésica** como disciplina que comprende el estudio del significado expresivo y/o comunicativo de movimientos corporales y gestos aprendidos, no orales, de percepción visual, auditiva, táctil solo o en relación con la situación comunicativa.

Dentro de los diferentes movimientos corporales existentes dentro de la **Kinésica**, nos centramos en los **gestos.** Siendo éstos una opción importante de comunicación no verbal o comunicación no vocal en la que expresiones corporales visibles comunican mensajes determinados.

En relación a la temática del proyecto, simplificaremos el estudio a solamente gestos manuales, cuya ejecución se reduce a movimientos o posturas de las manos.

3.2 Clasificación gestual.

En este subapartado se plantea una pequeña clasificación de los gestos manuales, en dos tipos, en relación a las características necesarias para poder analizarlos y por lo tanto identificarlos.

Por un lado, presentamos los **gestos dinámicos** que son los generados, principalmente, a partir de movimientos de la mano, antebrazo y brazo.

Las características principales por analizar, en esta tipología gestual, son las magnitudes físicas que describan la cinemática que conforma los diferentes movimientos de la mano, antebrazo y brazo. En este proyecto se analizarán posición, velocidad lineal, aceleración lineal y velocidad angular de un sistema de referencia tridimensional (**3D**) solidario a la mano respecto a un sistema de referencia tridimensional fijo u origen.

Por otro lado, exponemos los **gestos estáticos** como los que no conllevan movimientos de las manos. Se conforman de la posición de las manos, juntos con la posición que presentan las articulaciones de los dedos.

Las magnitudes físicas que se analizarán comprenden la orientación de un sistema de referencia tridimensional (**3D**) solidario a la mano respecto a un sistema de referencia tridimensional (**3D**) fijo, junto con el grado de flexión de las articulaciones de los dedos.

Finalmente, en relación a la descripción realizada de los tipos de gestos, presentamos las magnitudes físicas a analizar (y por lo tanto medir con el sistema electrónico a diseñar), junto con sus unidades en Sistema Internacional.

- Posición Lineal. [m]
- Velocidad Lineal. [m/s]
- Aceleración Lineal. [m/s^2]
- Orientación en ángulos RPY. [RAD]
- Velocidad Angular. [RAD/s]
- Grado de flexión. [%]

3.3 Biblioteca gestual.

En este subapartado se van a presentar el conjunto de gestos manuales a utilizar en el proyecto.

En el primer subapartado se van a explicar los criterios empleados para el diseño del vocabulario gestual. A continuación de *3.3.1 Criterios de diseño*, se describirán los gestos empleados según su clasificación.

3.3.1 Criterios de diseño.

El principal criterio de diseño para crear la biblioteca gestual es la búsqueda de la naturalidad en la ejecución de los diferentes movimientos, con el objetivo de crear una interacción entre usuario-máquina más natural e intuitiva que las existentes en la actualidad.

Búsqueda de vocabulario gestual sencillo y fácilmente recordable por el usuario. Tampoco se requiere un vocabulario muy extenso en gestos, ya que no se pretende aburrir al usuario durante el aprendizaje de una multitud de gestos. Además, una gran extensión en la biblioteca afectaría negativamente al criterio de naturalidad.

Se incluye que el vocabulario no deberá recoger gestos muy similares entre sí, facilitando la comunicación usuario-máquina y el reconocimiento gestual.

Para finalizar, el último criterio a utilizar para el diseño de la biblioteca gestual es la relación que guarda el gesto con la futura función y/o acción que podría desempeñar en una aplicación relacionada con una interfacaz **NUI**.

3.3.2 Biblioteca de gestos dinámicos.

A continuación, se van a presentar los gestos dinámicos seleccionados para la biblioteca gestual, junto con una breve descripción de su trayectoria y sus posibles utilidades en una interfaz de usuario.

- Up
    - Su ejecución consiste en realizar un movimiento ascendente hasta una posición superior a la inicial. Después de permanecer en esa posición se realizará un movimiento descendente hasta aproximadamente la posición inicial.
    - Su trayectoria queda representada en la *figura 11. Up*.
    - Las acciones más intuitivas para este movimiento:
        - Ascender en una página web.
        - Aumentar el nivel de volumen en aplicaciones multimedia (audio & vídeo) o del propio computador.
- Down
    - Su ejecución consiste en realizar un movimiento descendente hasta una posición inferior a la inicial. Después de permanecer en esa posición se realizará un movimiento ascendente hasta aproximadamente la posición inicial.
    - Su trayectoria queda representada en la *figura 12. Down*.
    - Las acciones más intuitivas para este movimiento:
        - Descender en una página web.
        - Descender el nivel de volumen en aplicaciones multimedia (audio & vídeo) o del propio computador.
- Left
    - Su ejecución consiste en realizar un movimiento lateral hacia la izquierda desde una posición inicial hasta otra posición final, este movimiento también se completa con el posterior retorno hasta la posición inicial.
    - Su trayectoria que representada en la *figura 13. Left.*
    - Posibles acciones para este movimiento:

- Cambiar en una lista de reproducción de una aplicación multimedia (audio & vídeo) a la pista de audio o vídeo previa a la que se está reproduciendo.
- Rebobinar hacia atrás una pista de audio o vídeo en una aplicación multimedia.
- Cambio a una diapositiva anterior en una aplicación de presentaciones.

- Right
  - Su ejecución consiste en realizar un movimiento lateral hacia la derecha desde una posición inicial hasta otra posición final, este movimiento también se completa con el posterior retorno hasta la posición inicial.
  - Su trayectoria que representada en la *figura 14. Right.*
  - Posibles acciones para este movimiento:
    - Cambiar en una lista de reproducción de una aplicación multimedia (audio & vídeo) a la pista de audio o vídeo posterior a la que se está reproduciendo.
    - Rebobinar hacia delante una pista de audio o vídeo en una aplicación multimedia.
    - Cambio a una diapositiva posterior en una aplicación de presentaciones.

Figura 11. Up

Figura 12. Down

Figura 13. Left

Figura 14.Right

De estos cuatro gestos planteados se pueden presentar diferentes variaciones dependiendo de la estancia en la posición final antes de retornar a la posición inicial. Por ejemplo, realizando el movimiento *up* se aumentaría el nivel de audio en uno. Sin embargo, al mantener la mano en la posición final sin retornar a la inicial, podríamos plantear un aumento mayor en el nivel de audio en proporción al tiempo de estancia en la posición final.
Otra aplicación genérica para estos movimientos es la de genera una línea recta, teniendo en cuenta la dirección del gesto, en aplicaciones de diseño gráfico y/o dibujo.

- Circle
  - Su ejecución consiste en *dibujar* una circunferencia con la mano.
  - Su trayectoria se puede observar en *Figura 15. Circle.*
  - Posibles acciones para este movimiento:

- Generación de círculos en aplicaciones de diseño gráfico y/o dibujo.
- Debido a su similitud con el símbolo de reproducción aleatoria, se podría utilizar para este fin en aplicaciones multimedia, aunque esta acción puede que no sea tan intuitiva.

Figura 15. Circle

Figura 16. Cross

- Cross
  - Su ejecución consiste en dibujar una especie de cruz o *tachón*.
  - Su trayectoria se presenta en la *Figura 16. Cross.*
  - Acciones posibles para este movimiento:
    - *Delete* en cualquier aplicación software que disponga de esta función.
    - Envío de archivos, carpetas y aplicaciones seleccionadas a la *papelera de reciclaje* del PC.
    - Atajo para cerrar sesión en cuentas personales de aplicaciones software, tales como correo electrónico y redes sociales.

3.3.3 Biblioteca de gestos estáticos.

A continuación, se van a presentar los gestos estáticos seleccionados para la biblioteca gestual, junto con una breve descripción de su ejecución y sus posibles utilidades en una interfaz de usuario.

- Stop
  - Su ejecución queda descrita en la *Figura 17 Stop.*
  - Posibles acciones para este gesto:
    - Pausar/Parar la ejecución de una aplicación software.
    - Pause/Stop en aplicaciones software relacionadas con reproducción de contenido multimedia.

- Tick
    - Su ejecución consiste en realizar una *tick* con los dedos pulgar e índice de la mano.
    - El gesto queda descrito en la *Figura 18 Tick.*
    - Posibles acciones para este gesto:
        - Renaudar la ejecución de una aplicación software.
        - Play en aplicaciones software relacionadas con reproducción de contenido multimedia.
        - Salvar o guardar un documento de una aplicación software cuando se ha finalizado.

Figura 17. Stop

Figura 18. Tick

- Up
    - Su ejecución consiste en señalar con el dedo pulgar hacia arriba mientras se mantiene el puño cerrado.
    - Se presenta en la imagen *Figura 19. Up.*
    - Acciones que se le podrían asignar a este gesto:
        - Ascender en una página web.
        - Aumentar el nivel de volumen en aplicaciones multimedia (audio & vídeo) o del propio computador.
        - Dar *like* en una publicación de una red social o a contenido multimedia en páginas de streaming, como por ejemplo youtube.

- Down
  - Su ejecución consiste en señalar con el dedo pulgar hacia abajo mientras se mantiene el puño cerrado.
  - Se presenta en la imagen *Figura 20. Down.*
  - Acciones que se le podrían asignar a este gesto:
    - Descender en una página web.
    - Disminuir el nivel de volumen en aplicaciones multimedia (audio & vídeo) o del propio computador.
    - Dar *dislike* en una publicación de una red social o a contenido multimedia en páginas de streaming, como por ejemplo youtube.

- Left
  - Su ejecución consiste en señalar con el dedo pulgar hacia la izquierda mientras se mantiene el puño cerrado.
  - Se presenta en la imagen *Figura 21. Left.*
  - Acciones que se le podrían asignar a este gesto:
    - Cambiar en una lista de reproducción de una aplicación multimedia (audio & vídeo) a la pista de audio o vídeo previa a la que se está reproduciendo.
    - Rebobinar hacia atrás una pista de audio o vídeo en una aplicación multimedia.
    - Cambio a una diapositiva anterior en una aplicación de presentaciones.

- Right
  - Su ejecución consiste en señalar con el dedo pulgar hacia la derecha mientras se mantiene el puño cerrado.
  - Se presenta en la imagen *Figura 22. Right.*
  - Acciones que se le podrían asignar a este gesto:
    - Cambiar en una lista de reproducción de una aplicación multimedia (audio & vídeo) a la pista de audio o vídeo siguiente a la que se está reproduciendo.
    - Rebobinar hacia delante una pista de audio o vídeo en una aplicación multimedia.
    - Cambio a una diapositiva posterior en una aplicación de presentaciones.

Figura 19. Up

Figura 20. Down

Figura 21. Left

Figura 22.Right

Estos cuatro gestos guardan similitud con los cuatro gestos dinámicos, del mismo nombre.

Son gestos que se les pueden aplicar acciones similares, ya que indican direcciones principalmente, sin embargo, estos gestos estáticos no tienen tanta naturalidad para aplicaciones de diseño gráfico o dibujo debido a que no hay movimiento.

- Fist
  - Su ejecución consiste en cerrar el puño tal como se observa en la *Figura 23. Fist.*
  - Acciones que se le pueden aplicar a este gesto:
    - *Mute* o quitar volumen al dispositivo o en una aplicación de reproducción de archivos multimedia.
    - Selección de archivos, carpetas y/o aplicaciones, con el fin de poder moverlas, junto con gestos dinámicos que indiquen dirección.

Figura 23. Fist

3.4 Problemas a solventar durante la identificación gestual.

En este subapartado, se va a exponer una serie de problemas que se presentan de forma previa a la implementación del hardware y software de reconocimiento gestual.

La solución a estos problemas se expresará en los apartados *4. HARDWARE Y FIRMWARE DE INSTRUMENTACIÓN y 5. SOFTWARE PARA RECONOCIMIENTO GESTUAL*, ya que son los apartados donde se desarrollará la implementación Hardware & Software de la aplicación.

Como se ha indicado en el subapartado *2.2 Clasificación Gestual,* es necesario observar las magnitudes físicas de los movimientos de la mano respecto a un sistema de referencia tridimensional **(3D)** fijo. Por ello, la elección de un sistema de referencia tridimensional **(3D)** fijo u origen y un sistema de referencia tridimensional solidario a la mano es el primer aspecto a solventar.

Un mismo gesto dinámico puede diferir en posición, velocidad y aceleración según la ejecución de cada usuario. Por ejemplo, la ejecución del gesto dinámico *circle* puede presentar diferentes radios, lo que conllevará diferentes aceleraciones lineales, puesto en un movimiento circular la aceleración centrípeta depende linealmente del radio.

Además, dos usuarios que realizan el mismo gesto dinámico, pero a diferentes velocidades, conllevará que uno finalice la ejecución del movimiento antes que otro.

Por lo que identificar gestos dinámicos presenta como problema principal la comparación y/o análisis de secuencias temporales de diferentes duraciones, al poder ejecutarse los movimientos a diferentes velocidades. Además de que estas secuencias temporales no presentan información característica de cada gesto debido a que un mismo gesto puede presentar valores muy diferentes en las magnitudes físicas de estudio, debido a la ejecución realizada por el usuario.

A diferencia de los gestos dinámicos, los gestos estáticos al no depender tan fuertemente del tiempo no presentar los problemas comentados en los párrafos anteriores. Aun así, debemos encontrar las herramientas necesarias para poder comparar este tipo de gestos, definidos por su orientación en ángulos rpy y grado de flexión de los cinco dedos de la mano.

Como último problema, cabe plantear como diferenciar entre los gestos dinámicos y estáticos, en cuanto a su identificación, debido a que las cualidades a estudiar en cada tipo gestual son diferentes. Además, habrá que conocer el inicio y fin de la ejecución de un gesto, sea estático o dinámico.

Finalmente, en relación a lo expresado en este apartado, resumimos los problemas a solventar.

- Elegir un sistema de referencia tridimensional (**3D**) fijo u origen.
- Elegir un sistema de referencia tridimensional (**3D**) solidario a la mano.
- Encontrar herramientas para analizar y comparar secuencias temporales de duraciones diferentes.
- Procesar secuencias temporales con el objetivo de obtener información característica a cada gesto dinámico, aunque éstos sean ejecutados a diferentes velocidades.
- Buscar herramientas para analizar y comparar secuencias procedentes de gestos estáticos.
- Diferenciar la tipología gestual en la fase de reconocimiento gestual.

4. HARDWARE Y FIRMWARE DE INSTRUMENTACIÓN.

4.1 Introducción.

Antes de presentar en detalle la implementación Hardware & Firmware, llevada a cabo en este proyecto, vamos a exponer los sensores utilizados para medir las magnitudes físicas desarrolladas en el apartado *3. ANÁLISIS GESTUAL Y GENERACIÓN DE UNA BIBLIOTECA GESTUAL.*

- **MPU-9250**

Dispositivo formado por una **IMU** (*Inertial Measurement Unit* o *Unidad de medición inercial,* en español) de nueve grados de libertad. La **IMU** se constituye de un acelerómetro de tres ejes, un giroscopio de tres ejes y un magnetómetro de tres ejes. En esta aplicación sólo utilizaremos el acelerómetro y el giroscopio, lo que conlleva seis grados de libertad.

El acceso a la información del dispositivo se realiza mediante comunicación **I2C**, cada medición de un sensor corresponde con un entero de 16 bits.

Su elección se debe a que es un dispositivo muy completo, que no necesita de ninguna etapa de instrumentación complementaria. Añadimos que el propio dispositivo presenta filtros configurables para reducir el ruido en las mediciones.

Presenta rango de escala programable para cada sensor que conforma el dispositivo, permitiendo configurar el dispositivo a los rangos de escala adecuados para nuestra aplicación.

Por último, presentamos que su elección también ha sido condicionada a la disponibilidad del dispositivo en los laboratorios del centro universitario.

- **Flex Sensor**

Consiste en un sensor resistivo cuyo valor óhmico varía con la flexión que sufre. Para este sensor se necesita una etapa previa de instrumentación para poder obtener una tensión proporcional a la resistencia del sensor, relacionada con el grado de flexión que sufre.

Su elección se debe a su simplicidad de uso, aunque sea necesaria una etapa de instrumentación, y a su estética, ya que es idóneo para aplicaciones similares a la del proyecto (ver *Figura 25. FlexSensor*).

Añadimos que están disponibles en los laboratorios del centro universitario.

Adjuntamos dos imágenes donde se observa la estética de los sensores utilizados en el proyecto.

Figura 24. MPU-9250

Figura 25. FlexSensor

En *Datasheets, ANEXO II,* se puede observar la documentación técnica del proyecto, y por tanto de los sensores comentados en este apartado.

A continuación, se va a presentar la solución a los problemas ***Elegir un sistema de referencia tridimensional fijo u origen*** y ***Elegir un sistema de referencia tridimensional solidario a la mano***, comentada en el apartado *3. ANÁLISIS GESTUAL Y GENERACIÓN DE UNA BIBLIOTECA DE GESTOS.* Ya que está estrechamente ligado con la localización que tomará el dispositivo **MPU-9250** en la aplicación.

En la *figura 26. Sistema_Referencia_3D* se puede apreciar la elección tomada como sistema de referencia tridimensional fijo u origen para la aplicación, hay que añadir que el sistema de referencia del dispositivo **MPU-9250** debe coincidir inicialmente con el sistema de referencia fijo planteado.

Lógicamente tras la realización de los diferentes gestos, los sistemas de referencia no coincidirán.

El sistema de referencia del dispositivo **MPU-9250** conforma el sistema de referencia tridimensional solidario a la mano, ya que este dispositivo irá unido a ella.

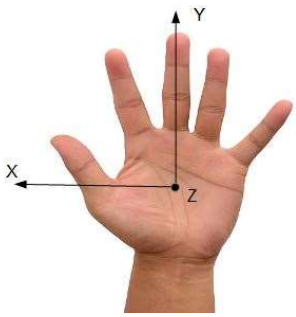


Figura 26. SistemaReferencia_3D

Como en mecánica clásica, el sistema de referencia tomado coincide con un sistema de coordenadas ortogonal bajo las reglas de un espacio euclídeo, esto nos permite describir la relación entre dos sistemas de referencia a través de giros y traslaciones, que nos servirá de bastante utilidad para estudiar, principalmente, los gestos dinámicos propuestos para la aplicación.

4.2 Hardware.

En este subapartado se va a llevar a cabo una breve descripción del hardware que conforma el prototipo cableado utilizado para iniciar el estudio sobre reconocimiento gestual.

Como se ha comentado, no se contará con una descripción muy detallada debido a que en apartados posteriores (*6. DISEÑO DE PROTOTIPO EN PCB*) se desarrollará todo lo relacionado con el Hardware del prototipo final en placa de circuito impreso, ya que esto tiene mayor peso en el proyecto que el prototipo cableado.

La plataforma *Arduino Uno Rev3* será la utilizada para la puesta a punto de este prototipo cableado, debido a su rapidez en montaje y programación al ser una plataforma *open source* y *open hardware* que cuenta con su propio entorno de desarrollo gratuito (*Arduino IDE*) para la creación de firmware en un lenguaje de programación basado en C++, pero encapsulando opciones de bajo nivel típicas en el desarrollo de Firmware para microcontroladores.

El principal problema de otras plataformas hardware es que al no ser *open source* y *open hardware* no se dispone de tantas herramientas gratuitas para el desarrollo de firmware, lo que no nos permite esa rapidez de montaje y desarrollo como la plataforma *Arduino.*

Además de que el precio es generalmente más elevado para otras plataformas que para *Arduino*, aunque disponemos de esta en los laboratorios del centro.

Arduino Uno Rev3 consta del microcontrolador *ATmega328P,* actualmente de la marca Microchip. Este microcontrolador consta de una arquitectura RISC y una CPU de 8 bits.

Otras características importantes para el desarrollo del prototipo son memoria Flash de 35 kB, memoria SRAM de 2 kB, seis canales con capacidad de conversión analógico-digital (Conversor AD de 10 bits), presenta módulos de comunicación UART, I2C, SCI; entre otras.

Se puede encontrar más información técnica sobre esta plataforma Hardware en *ANEXO II, Datasheets.*

Con las características comentadas queremos expresar que esta plataforma es suficiente para la realización del prototipo cableado, ya que se dispone de comunicación I2C para el control del dispositivo **MPU-9250**, canales suficientes A/D para la instrumentación de los Flex Sensors, comunicación UART para conectar el prototipo con otros dispositivos electrónicos como PC, Smartphones, etcétera; además de memoria FLASH y RAM de sobra para el Firmware.

Figura 27. Plataforma Arduino

Ya introducida la plataforma *Arduino*, se va a presentar el conexionado del prototipo cableado.

En relación a los **Flex Sensors**, se debe realizar una etapa previa antes de su conexión a la plataforma *Arduino.* La etapa previa consiste en un divisor resistivo formado por el propio sensor y una resistencia de 22 kΩ, alimentado a 5V procedentes de la propia plataforma *Arduino.* La salida de la etapa da una tensión relacionada con el grado de flexión que sufre el sensor. Al ser la salida una tensión analógica se deberá utilizar los conversores analógico-digital para su lectura. Se necesitarán de cuatro canales de conversión al disponer de cuatro sensores.

En relación al **MPU-9250**, se lleva a cabo una comunicación I2C. El tipo de conexionado se puede observar en la imagen *Figura 28. Conexión_I2C.*

Las líneas SDA y SCL corresponden con la señal de datos y con el reloj de la comunicación, respectivamente.

La comunicación llevada a cabo en el prototipo plantea como dispositivo MASTER la plataforma *Arduino* y como dispositivo SLAVE el dispositivo **MPU-9250.**

El prototipo también presenta una comunicación serial asíncrona cableada con PC a través del cable USB de programación. En la *figura 29. Conexión_UART* se presenta el conexionado que conlleva este tipo de comunicación, donde RX y TX corresponde a la línea de recepción de datos y transmisión de datos de cada dispositivo, respectivamente.

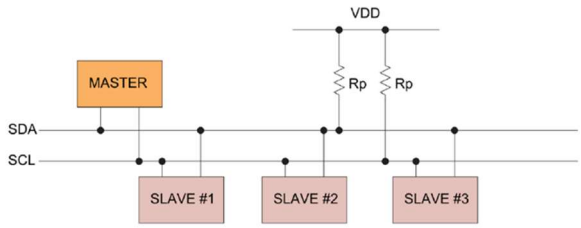


Figura 28. I2C

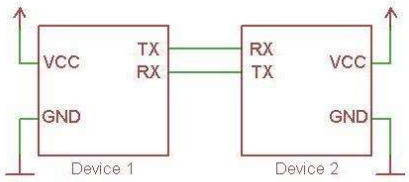


Figura 29. UART

4.3 Firmware.

En este subapartado se va a desarrollar la implementación del Firmware del prototipo. Para ello, primero se va a describir las ecuaciones necesarias para la conversión de la información cruda en información valiosa (unidades de magnitud física) para el reconocimiento gestual.

Luego, se describirá la comunicación serial de nuestro prototipo y una computadora; y por último, se llevará a cabo una descripción global del Firmware a través de un flujograma.

4.3.1 Instrumentación MPU-9250.

El dispositivo **MPU-9250** envía la información de una medición en números enteros de 16 bits, habrá que tener en cuenta que la información relacionada con el acelerómetro viene presentada en complemento a 2, mientras que la información relacionada con el giróscopo no presenta este formato.

En primer lugar, hay que establecer la escala del acelerómetro y giróscopo más adecuada, de las permitidas por el dispositivo, para la aplicación.

La escala más adecuada en relación con el acelerómetro es la que presenta un rango de medición de valores entre -2g y +2g, siendo g la unidad equivalente a 9.8 m/s^2 en Sistema Internacional. Esta escala es muy adecuada ya que los gestos más bruscos y rápidos presentan aceleraciones menores a las máximas que establecen esta escala.

Para el giróscopo se establece la escala de valores entre -1000 º/s y +1000 º/s. Al igual que en el acelerómetro, los gestos más bruscos y rápidos no superan los límites de escala.

Destacamos que el dispositivo **MPU-9250** permite configurar filtros de paso bajo para reducción de ruido. Para el prototipo se configuran los filtros a una frecuencia de corte de 5Hz, siendo la frecuencia de corte más baja que permite el dispositivo.

También, se lleva a cabo una calibración del acelerómetro y del giróscopo realizando la media de 1024 mediciones. Obteniendo un valor de calibración, que se substraerá en las futuras mediciones.

Para finalizar, se presentan las ecuaciones que permiten convertir la información cruda enviada por el dispositivo en información valiosa en unidades de la magnitud física que corresponda.

- Acelerómetro

$$acceleration_{value} = -\frac{raw_{value}}{16384.0} - accel_{offset}\ [g]$$

Siendo el valor *16384* la sensibilidad del sensor, calculada como:

$$sensibilidad_{accel} = \frac{escala_{entero\_16bits}}{escala_{sensor}} = \frac{2^{16} - 1}{+2 - (-2)} = 16384.0$$

- Giróscopo

$$gyroscope_{value} = \frac{raw_{value}}{32.8} - gyro_{offset}\ [g]$$

Siendo el valor 32.8 la sensibilidad del sensor, calculada como:

$$sensibilidad_{accel} = \frac{escala_{entero\_16bits}}{escala_{sensor}} = \frac{2^{16} - 1}{+1000 - (-1000)} = 32.8$$

4.3.2 Instrumentación Flex Sensors.

En este subapartado se va a presentar la estrategia planteada para la instrumentación del grado de flexión.

En primer lugar, cabe destacar que los conversores analógico-digital de la plataforma *Arduino* presentan entero de 10 bits, es decir, números enteros desde 0 hasta 1023. Hay que encontrar una expresión que relacione estos valores enteros con el grado de flexión en %.

La expresión planteada es la siguiente:

$$flex(\%) = -100 * \frac{CAD_{integer} - 300}{Offset - 300} + 100\ [\%]$$

La expresión proviene de llevar a cabo una relación lineal entre el rango 0-100 y el rango Offset-300, donde Offset representa al entero que devuelve el conversor A/D en el caso de no

haber flexión alguna (se mide al inicio del Firmware) y 300 es el entero que devuelve el conversor A/D cuando el grado de flexión es máxima.

4.3.3 Comunicación serial.

En este subapartado, se va a describir el mensaje o la trama que se envía de nuestro prototipo al PC.

El objetivo de esta comunicación serial es el de enviar la información medida por el dispositivo al PC, para ser procesada y analizada por este, con el fin de llevar a cabo una identificación gestual.

La trama serial se conforma de:

1. Cadena de caracteres: ‘ax’.
2. Double con el valor de aceleración en el eje X.
3. Cadena de caracteres: ‘ay’.
4. Double con el valor de aceleración en el eje Y.
5. Cadena de caracteres: ‘az’.
6. Double con el valor de aceleración en el eje Z.
7. Cadena de caracteres: ‘gx’.
8. Double con el valor de velocidad angular en el eje X.
9. Cadena de caracteres: ‘gy’.
10. Double con el valor de velocidad angular en el eje Y.
11. Cadena de caracteres: ‘gz’.
12. Double con el valor de velocidad angular en el eje Z.
13. Cadena de caracteres: ‘f0’.
14. Float con el valor de flexión del dedo 0.
15. Cadena de caracteres: ‘f1’.
16. Float con el valor de flexión del dedo 1.
17. Cadena de caracteres: ‘f2’.
18. Float con el valor de flexión del dedo 2.
19. Cadena de caracteres: ‘f3’.
20. Float con el valor de flexión del dedo 3.
21. Cadena de caracteres: ‘f4’.
22. Float con el valor de flexión del dedo 4.
23. Salto de carro: \r

A continuación, presentamos un ejemplo de una trama obtenida en una de las pruebas de la comunicación serial del prototipo cableado:

ax0.32ay0.73az-0.70gx-10.61gy-2.59gz-1.04f00.00f10.00f20.00f30.30f40.00\r

4.3.4 Flujograma del Firmware.

A continuación, se adjuntan las imágenes *Figura 30. Setup* y *Figura 31. Loop.*

Estas imágenes representan el funcionamiento de nuestro Firmware diferenciando en *setup* y en *loop*.

- *Setup*

Conforma el flujo de programa inicial, por lo tanto, sólo se ejecuta una vez.

Aquí se implementarán las tareas de configuración y calibración.

- *Loop*

Es el flujo de programa que se ejecuta después del *Setup*. También conocido como ejecutivo cíclico. Como expresa su nombre es el flujo de programa que se ejecuta cíclicamente.

Aquí se implementarán las tareas de lectura de datos de la comunicación I2C y de los conversores analógico-digitales, el procesamiento de datos *raw* para obtener datos reales y su posterior envío por comunicación serial.

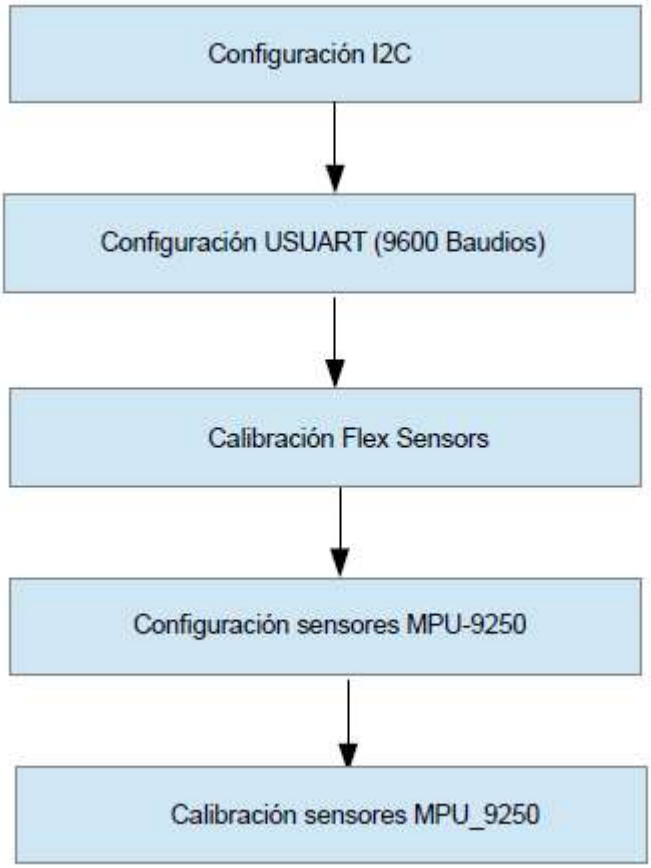


Figura 30. Setup

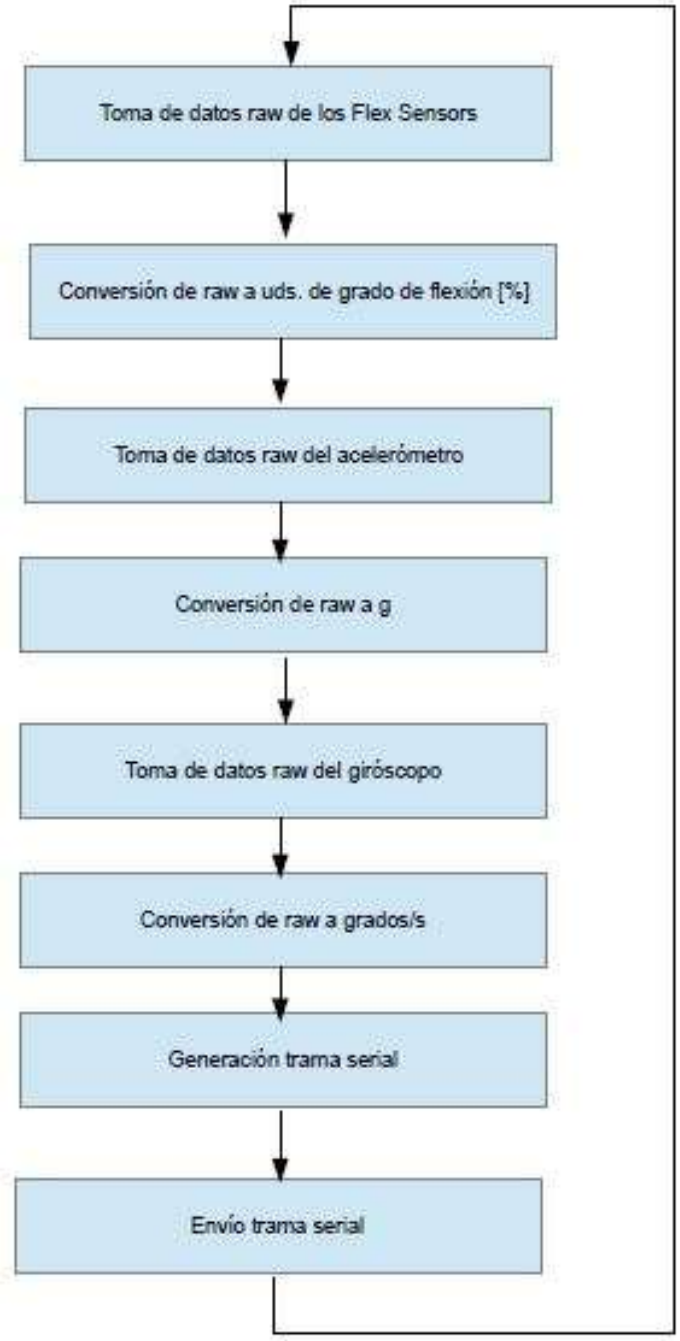


Figura 31. Loop

El código fuente del *Firmware* se encuentra en *ANEXO I, Código.*

5. SOFTWARE PARA RECONOCIMIENTO GESTUAL.

En este apartado se va a desarrollar todo lo relacionado con el Software ejecutado por el PC para procesar y analizar la información obtenida del prototipo cableado, y ver que soluciones se pueden plantear para el reconocimiento de gestos manuales de tipo estático y dinámico.

Para el Software se utilizará scripts de Matlab, principalmente porque el lenguaje de programación de Matlab presenta una sintaxis sencilla. Además, es un lenguaje que facilita el trabajo con matrices y que tiene un gran soporte matemático, lo que nos facilita el procesamiento de la información obtenida debido a al uso de varias ecuaciones matemáticas relacionadas con la trigonometría y geometría.

5.1 Adquisición de información vía puerto serial.

Para la adquisición de la información del prototipo cableado vía puerto serie, en primer lugar, debemos tener un script en Matlab que recoja que puerto serial se va a utilizar para la comunicación, a que velocidad (en baudios) se realiza la comunicación, tipo de paridad y la existencia o no de bit de start y stop. También, se puede configurar más parámetros que no se comentan, ya que la trama serial enviada por la plataforma *Arduino* es sencilla en relación a estos parámetros.

Usando la función *fgetl*, de Matlab, nos devuelve una cadena de caracteres formada por todos los bytes enviados por el puerto serial hasta encontrarse con el carácter, \r, de salto de carro. Esta herramienta de Matlab nos facilita bastante el trabajo, ya que tenemos toda la información enviada por puerto serie en una cadena de caracteres.

Por último, se escribe una función en Matlab (llamada *read_data*) que tiene como argumento una cadena de caracteres (la obtenida vía puerto serial) y como resultado tiene un vector columna donde se presenta la información enviada en el siguiente formato:

[ax; ay; az; gx; gy; gz; f0; f1; f2; f3; f4]

Siendo:

- ax valor de aceleración en el eje X.
- ay valor de aceleración en el eje Y.
- az valor de aceleración en el eje Z.
- gx valor de velocidad angular en el eje X.
- gy valor de velocidad angular en el eje Y.
- gz valor de velocidad angular en el eje Z.
- f0 valor de grado de flexión en dedo 0, índice.
- f1 valor de grado de flexión en dedo 1, corazón.
- f2 valor de grado de flexión en dedo 2, anular.
- f3 valor de grado de flexión en dedo 3, meñique.
- f4 valor de grado de flexión en dedo 4, pulgar.

5.2 Procesamiento de la información.

En este subapartado se va trata sobre el procesamiento de los datos obtenidos a partir del dispositivo (aceleración lineal en tres ejes, velocidad angular en tres ejes y grado de flexión de los dedos), para obtener información sobre otras magnitudes físicas deseables para la identificación gestual.

5.2.1 Orientación

En mecánica clásica, se define orientación a la relación de rotación entre dos sistemas de referencia. En nuestra aplicación tiene importancia conocer la orientación de la mano respecto al sistema de referencia fijo, para tener una idea de cómo está posicionada la mano en el espacio.

Para representar la orientación de un sistema de referencia en esta aplicación se ha utilizado los siguientes formatos, aunque existen otros formatos:

- Ángulos RPY.

Los ángulos RPY se forman de los ángulos Roll, Pitch y Yaw. Estos ángulos indican el giro realizado en cada eje, usando como unidades los radianes [Rad].

Roll es el ángulo girado sobre el eje X. En este proyecto se tomará el símbolo Ψ para identificarlo.

Pitch es el ángulo girado sobre el eje Y. En este proyecto se tomará el símbolo θ para identificarlo.

Yaw es el ángulo girado sobre el eje Z. En este proyecto se tomará el símbolo Φ para identificarlo.

- Matriz de Rotación (R).

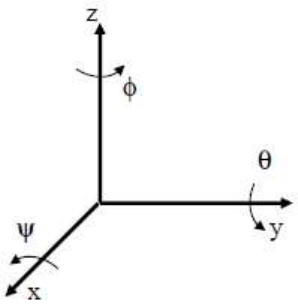
La matriz de rotación es una matriz de tamaño 3 X 3 que nos permite relacionar un punto de un sistema de referencia con un punto de otro sistema de referencia, cuando solamente exista una relación de rotación entre estos. Cuando la relación sea de traslación o traslación junto con rotación, la matriz de rotación, por si sola, no permite describir esta relación.

La relación que presenta la matriz de Rotación es:

$$x_1 = R * x_2$$

$$x_2 = R_T * x_1$$

En la imagen de la *Figura 32. ÁngulosRPY* se muestra la relación existente entre ángulos RPY y matriz de rotación, siendo *C,* la operación trigonométrica coseno y *S,* la operación trigonométrica seno. [7]



Secuencia de **3 rotaciones** entorno a los **3 ejes**

RPY (ϕ, θ, ψ) = Rot (z, ϕ) * Rot (y, θ) * Rot (x, ψ)

$$\text{RPY}(\phi,\theta,\psi)=\begin{pmatrix} C\phi C\theta & C\phi S\theta S\psi - S\phi C\psi & C\phi S\theta C\psi + S\phi S\psi \\ S\phi C\theta & S\phi S\theta S\psi + C\phi C\psi & S\phi S\theta C\psi - C\phi S\psi \\ -S\theta & C\theta S\psi & C\theta C\psi \end{pmatrix}$$

Figura 32. RPY

A continuación, se va a desarrollar la iniciativa implementada en el software para extraer la orientación de la mano a partir de la información obtenida.

1. **Ángulos RPY calculados a partir de información de velocidad angular**.

En mecánica clásica, la velocidad angular se puede describir como la derivada, respecto al tiempo, de la posición angular u orientación.

Así, conociendo la velocidad angular y través de integración se puede obtener información sobre orientación relativa. Siendo la orientación inicial nula, al estar alineados los sistemas de referencia fijo y mano, se obtendrá información sobre orientación absoluta en relación al sistema de referencia fijo.

Al estar trabajando con un sistema digital, por lo tanto, un sistema discretizado, debemos tener en cuenta la integración numérica para la resolución del problema de extracción de la orientación absoluta de la mano.

Cálculos realizados:

$$RPY[k] = RPY\ [k-1] + \Omega * T \quad [Rad]$$

Siendo:

- RPY[k]: ángulos RPY en el instante actual.
- RPY[k-1]: ángulos RPY en el instante anterior.
- Ω: velocidad angular en los tres ejes.
- T: tiempo de discretización o muestreo (10ms en la aplicación).

Las principales ventajas que tiene este método de cálculo es que es sencillo, fácilmente programable y permite observar de forma rápida cambios bruscos en la orientación. Sin embargo, el principal inconveniente es que, ante un error constante (por ejemplo, un error de calibración) repercute gravemente en la medida al estar acumulándose en cada iteración del cálculo, puesto que es un cálculo acumulativo.

2. **Ángulos RPY calculados a partir de información de aceleración lineal.** [2]

El cálculo de la orientación a través de información sobre aceleración lineal tiene relación directa con la acción de la gravedad y la trigonometría.

En la siguiente imagen, se puede apreciar la relación entre orientación y acción de la gravedad.

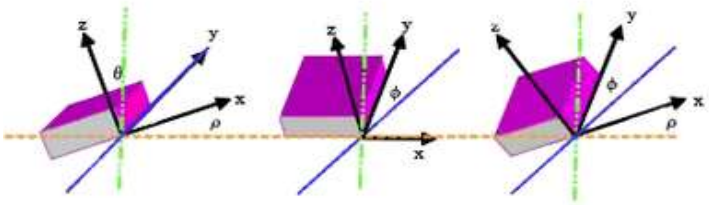


Figura 33. Acceleration Angles

Las expresiones tomadas para el cálculo de la orientación a través de la aceleración lineal:

$$\Psi = \text{atan}\left(\frac{\text{ax}}{\sqrt{\text{ay}^2 + \text{az}^2}}\right) \ [Rad]$$

$$\theta = \text{atan}\left(\frac{\text{ay}}{\sqrt{\text{ax}^2 + \text{az}^2}}\right) \ [Rad]$$

$$\Phi = \text{atan}\left(\frac{\sqrt{ax^2 + ay^2}}{\text{az}}\right) \ [Rad]$$

Siendo:

- Ax: aceleración en eje X.
- Ay: aceleración en eje Y.
- Az: aceleración en eje Z.

La principal ventaja de este método de cálculo es que la información calculada es menos errónea, al no ser un cálculo acumulativo. Sin embargo, presenta desventajas ante cambios bruscos de orientación donde es mejor el uso del giróscopo, además de ser un método de cálculo más complejo que el planteado para el giróscopo.

Hay que añadir que el cálculo del ángulo Yaw es incorrecto en el caso de que ángulos Roll y Pitch sean nulos, siendo Yaw un grado de libertad a calcular a través de la información obtenida por el giróscopo, esto se tiene en cuenta en la implementación realizada.

3. Filtro complementario y alternativas de mejora.

Filtro complementario es un algoritmo proveniente de una simplificación del filtro de Kalman.

Con este algoritmo se busca obtener un cálculo más robusto para la orientación tomando las ventajas que nos plantean los métodos de cálculo de orientación explicados con anterioridad.

Implementación:

$$RPY[k] = A * RPY[k]_{ACEL} + (1 - A) * RPY[k]_{GYRO}$$

Siendo:

- A: Constante del filtro. Su extracción es experimental (en la aplicación se obtuvieron buenos resultados para valores entre 0.85 y 0.9) y su función es básicamente dar más peso a un método de cálculo u a otro.
- RPY[k]_ACEL: Ángulos RPY calculados a partir de la información del acelerómetro.
- RPY[k]_GYRO: Ángulos RPY calculados a partir de la información del giróscopo.

Otra alternativa llevada a cabo para reducir errores en el cálculo de la orientación consiste en observar si una medida de aceleración lineal en un eje se corresponde con la acción de la gravedad (+/- 1g) y en caso afirmativo llevar a cabo acciones relacionadas. Por ejemplo, si se mide una aceleración de -1g en el eje z, quiere decir que los ángulos Roll y Pitch tiene valor nulo.

5.2.2 Aceleración Inercial

En este proyecto, definimos aceleración inercial como la aceleración que sufre el dispositivo, la que realiza la mano, sin tener en cuenta la acción de la gravedad. Es la aceleración con la que podemos reconstruir el movimiento que ejerce la mano sobre el sistema de referencia fijo.

Para extraer la aceleración inercial:

En primer lugar, debemos trasladar las medidas de aceleración obtenidas al sistema de referencia fijo, ya que las medidas que se obtienen con el acelerómetro pertenecen al sistema de referencia de la mano.

Después se resta el efecto de la gravedad en el sistema de referencia fijo. El efecto de la gravedad siempre es constante en este sistema de referencia, y es de valor -1g o -9.8 [m/s^2] en el eje Z.

La información sobre aceleración inercial se debe expresar siempre en el sistema de referencia fijo, principalmente por dos motivos.

Respecto al primer motivo, la reconstrucción de la trayectoria de la mano se debe realizar respecto a un sistema de referencia fijo, por eso necesitamos la información de aceleración inercial expresada en este sistema de referencia.

En cuanto al segundo motivo, consiste en que la extracción de la acción de la gravedad en el sistema de referencia fijo es más sencilla, al ser una acción constante en un solo eje.

Por otro lado, si observásemos el problema desde el sistema de referencia de la propia mano, el problema sería más complejo al afectar la gravedad en los tres ejes del sistema de referencia siendo cambiante en cada instante temporal, ya que se modificaría en función a la orientación de la mano.

A continuación, presentamos la ecuación utilizada para extraer la aceleración inercial:

$$A_{inertial} = R_T * A - G$$

Siendo:

- A_inertial: vector columna de aceleración inercial.
- R_T: matriz de rotación traspuesta.
- A: vector columna de aceleración tomada del dispositivo.
- G: vector columna que expresa la acción de la gravedad. Conformado por el valor 0 en el eje X, el valor 0 en el eje Y, y el valor de -1 [g] ó -9.8 [m/s^2] en el eje Z.

5.2.3 Velocidad y Posición Lineal

La velocidad y posición lineal son las principales magnitudes físicas para llevar a cabo un buen *tracking* de la trayectoria que realiza la mano, con el objetivo de identificar gestos manuales.

En mecánica clásica, se describe la velocidad lineal como la derivada respecto del tiempo de la aceleración lineal, y a su vez, la posición lineal se describe como la derivada respecto del tiempo de la velocidad lineal.

Como quedó expresado en el subapartado *5.2.1 Orientación* el cálculo de la orientación a partir de la integración numérica de la velocidad angular, aplicaremos el mismo concepto de integración numérica en un sistema discretizado para el cálculo de la velocidad y posición lineal a partir de la aceleración inercial extraída, no directamente con la aceleración medida por el dispositivo ya que conlleva información sobre la acción de la gravedad, que implicaría una trayectoria incorrecta.

Por lo tanto, expresamos los cálculos realizados:

$$v[k] = v[k-1] + a_{inertial}[k] * T$$

Donde:

- v[k]: velocidad lineal en el instante actual.
- v[k-1]: velocidad lineal en el instante anterior.
- a_inertial[k]: aceleración inercial en el instante actual.
- T: tiempo de muestreo.

$$p[k] = p[k-1] + v[k] * T$$

Donde:

- p[k]: posición lineal en el instante actual.
- p[k-1]: posición lineal en el instante anterior.
- v[k]: velocidad lineal en el instante actual.
- T: tiempo de muestreo.

Para un buen tracking de la mano, en cuanto a velocidad y posición lineal, debemos tener bien calibrado nuestro dispositivo de medición, ya que un pequeño error de calibración (un error constante) afectará negativamente en el cálculo de velocidad, al ser un método de cálculo integral. Por lo tanto, afectará en peor medida al cálculo de posición lineal al acumular doblemente ese error.

Teniendo en cuenta que las distancias a tomar sobre la trayectoria de la mano no excederán la unidad de metros y la siguiente tabla, *Tabla 1. TrackingErrors,* muestra errores de metros para pequeños errores de aceleración y orientación. Finalmente, se observa que un buen tracking de distancias tan cortas es bastante complejo, por lo que habrá que buscar otros métodos para aplicación. [15]

Angle Error (degrees)	Acceleration Error (m/s/s)	Velocity Error (m/s) at 10 seconds	Position Error (m) at 10 seconds	Position Error (m) at 1 minute	Position Error (m) at 10 minutes	Position Error (m) at 1 hour
0.1	0.017	0.17	1.7	61.2	6120	220 e 3
0.5	0.086	0.86	8.6	309.6	30960	1.1 e 6
1.0	0.17	1.7	17	612	61200	2.2 e 6
1.5	0.256	2.56	25.6	921.6	92160	3.3 e 6
2.0	0.342	3.42	34.2	1231.2	123120	4.4 e 6
3.0	0.513	5.13	51.3	1846.8	184680	6.6 e 6
5.0	0.854	8.54	85.4	3074.4	307440	11 e 6

Tabla 1. IMU Errors

5.3 Reconocimiento gestual.

En este subapartado se van a plantear las soluciones implementadas a los problemas desarrollado en el apartado *3. ANÁLISIS GESTUAL Y EXTRACCIÓN DE UNA BIBLIOTECA DE GESTOS.*

En primer lugar, se van a exponer las soluciones para llevar a cabo la identificación de gestos estáticos, por un lado, y de gestos dinámicos, por otro lado, en dos subapartados.

Al final del apartado, se expondrán las soluciones a cuando utilizar unas técnicas u otras, además de como diferenciar la tipología gestual.

5.3.1 Solución implementada para gestos estáticos.

La solución propuesta para la identificación de gestos estáticos consiste básicamente en comparar los datos recibidos vía el puerto serial con los datos característicos de cada gesto estático de la biblioteca gestual, previamente conformados y almacenados en un script de Matlab.

Los datos característicos de gestos estáticos hacen referencia a magnitudes físicas como orientación y grado de flexión articular, aunque en el caso de la aplicación también se ha utilizado la acción de la gravedad sobre el dispositivo como rasgo característico.

Proponemos un pequeño ejemplo de cómo está estructurado el script de Matlab donde se contienen los rasgos característicos de los gestos estáticos.

Por ejemplo, para el gesto estático *stop* se debe describir que la acción de la gravedad afecta como -1 [g] o -9.8 [m/s^2] en el eje Y, la orientación en ángulos RPY será de [90º, 0º, 0º] y para el grado de flexión articular [0, 0, 0, 0, 0].

Toda esta información queda comprendida en un vector columna llamado *static_stop_data,* cambiando *stop* por el nombre del gesto correspondiente.

Ejemplo:

```
static_stop_data = [0; -9.8; 0; 90; 0; 0; 0; 0; 0; 0; 0]
```

Como es lógico, en el caso práctico, los datos no van a aparecer idénticamente a los propuestos en la biblioteca de gestos estáticos, por eso tenemos que dar un pequeño margen de error. Los márgenes de error utilizados en la aplicación se han obtenidos a través de la experimentación, siendo 0.5 para la aceleración lineal en [m/s^2], 20 [º] para la orientación y entre 15 y 20 [%] para el grado de flexión articular.

Añadimos que los gestos estáticos de nuestra biblioteca muestran orientaciones sencillas (paralelas o perpendiculares al sistema de referencia fijo), por lo que para una identificación más rápida y precisa bastaría con utilizar la información relacionada con la acción de la gravedad.

Finalmente, en el script de Matlab se encuentra implementada la opción tratada en el párrafo anterior.

En la siguiente imagen, se presenta un pequeño flujograma sobre el Software de identificación de gestos estáticos.

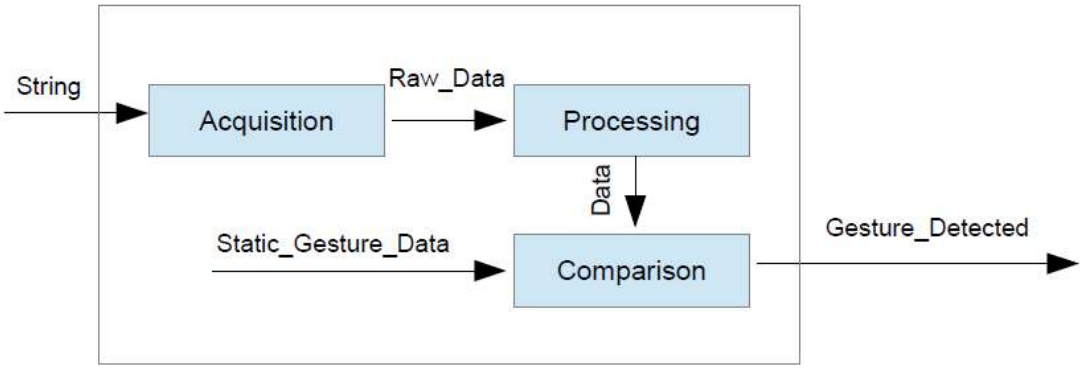


Figura 34. Static Recognition

Los scripts de Matlab relacionados con la identificación de gestos estáticos se pueden localizar en el *ANEXO I, Código.*

5.3.2 Solución implementada para gestos dinámicos

La solución implementada para la identificación de gestos dinámicos es más compleja que la tomada para el reconocimiento de gestos estáticos, debido a que hay más problemas a solventar, por lo que es necesario un mayor análisis hasta llegar a una solución adecuada.

Para el problema planteado ***Encontrar herramientas para analizar y comparar secuencias temporales de duraciones diferentes*** presentamos el algoritmo DTW (Dynamic Time Warping).

El algoritmo DTW es un algoritmo que nos permite medir la similitud entre dos secuencias temporales que varían en tamaño, justo el problema que queríamos solventar. Este algoritmo data de los años 90, y una de sus principales aplicaciones fue el reconocimiento palabras habladas.

La aplicación del algoritmo DTW en nuestro proyecto, consiste en buscar la similitud entre dos secuencias temporales discretizadas, una secuencia de datos correspondiente al gesto realizado por el usuario con una secuencia de datos almacenados previamente en el computador, formando una especie de biblioteca de gestos dinámicos (similar a la iniciativa realizada con los gestos estáticos, pero en vez de almacenar un vector columna de datos consiste en almacenar una matriz donde una fila hace referencia a un dato concreto y una columna hace referencia a un instante temporal concreto).

El algoritmo DTW toma como parámetros de entrada dos secuencias temporales y tiene como salida la distancia de similitud entre las secuencias introducidas, cuanto menor sea la distancia más parecidas son las secuencias introducidas. Una breve descripción de la funcionalidad del algoritmo es la búsqueda de la similitud entre los componentes de las dos secuencias a través del cálculo de su distancia, dando como resultado un valor conformado por una ponderación de las distancias menores obtenidas.

El cálculo de la distancia, en el algoritmo DTW, se puede realizar utilizando el valor absoluto de la resta de los componentes o el cuadrado de la resta de los componentes. En nuestra aplicación se implementa con el cuadrado de la resta de los componentes en vez del valor absoluto, ya que si dos componentes son próximos entre si la distancia será un valor muy pequeño, por el contrario, a dos componentes muy distintos la distancia comprenderá un valor elevado. [3] y [4].

Una vez desarrollado el algoritmo DTW queda conocer que datos van a formar estas secuencias temporales. Para ello, debemos estudiar qué información es característica para describir un gesto dinámico, solventando *el problema Procesar secuencias temporales con el objetivo de obtener información característica a cada gesto dinámico, aunque éstos sean ejecutados a diferentes velocidades.*

Como se ha visto en el subapartado *5.2 Procesamiento de la Información,* la información que se obtiene de velocidad y posición lineal puede acumular errores que harían imposible un tracking adecuado para la identificación de gestos dinámicos, por eso debemos encontrar otro tipo de información que sea característica y que podamos comparar utilizando el algoritmo DTW.

A continuación, se van a plantear una serie de alternativas que se han estudiado en el proyecto, donde, finalmente, se utilizará la alternativa con mejores resultados. Cabe destacar, que en estas alternativas no se utilizará la aceleración inercial, sino la aceleración lineal. Al tener información sobre la acción de la gravedad, puede ser algún detalle característico sobre el gesto a identificar.

En primer lugar, expresamos las alternativas de usar el **signo de aceleración lineal** como información característica en los gestos dinámicos**.** [5]

Esta alternativa consiste en almacenar una secuencia *codificada* de aceleraciones lineales. La codificación consiste en dar el valor -1 a valores negativos de aceleración, el valor 1 a valores positivos de aceleración y el valor 0 a valores iguales a cero (o muy próximas a cero, se tomó un error de 0.01 [m/s^2] en la prueba experimental).

Una ventaja que nos aporta esta codificación es conocer la trayectoria de forma simple, al conocer cuando se acelera, se decelera o se está sin movimiento.

Otra ventaja que ofrece esta codificación es que no se tiene en cuenta la amplitud de toman los valores de aceleración, aspecto que no era característico a la hora de identificar un gesto.

Como aspecto negativo, un valor cero no siempre tiene que indicar una situación de reposo, sino puede ser una situación de velocidad constante. Indicamos que es complejo llevar una velocidad constante al ejecutar un movimiento de la mano, aun así, existen herramientas para solventar este hecho, como, por ejemplo, un seguimiento de velocidad lineal que no es tan erróneo como el de posición.

La principal problemática de esta alternativa es su uso con el algoritmo DTW, ya que al presentar la codificación sólo tres valores las distancias obtenidas son muy parecidas entre sí, por lo que la solución que devuelve el algoritmo no es un valor adecuado para dar como bueno la identificación de un gesto.

Como segunda alternativa estudiada, planteamos la **normalización de aceleración lineal.**

Esta alternativa consiste en normalizar los valores obtenidos de aceleración lineal.

Para ello dividimos la información obtenida de aceleración lineal por eje entre el módulo de aceleración lineal. El módulo de aceleración lineal se calcula como:

$$Módulo = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

Esta opción presentó mejores resultados que la primera alternativa, ya que se consiguieron identificar varios gestos. Sin embargo, en ocasiones daba resultados de forma errónea.

Como última alternativa implementada, estudiamos el uso de la **desviación estándar de aceleración lineal.** [6]

Esta opción consiste en almacenar las desviaciones estándar de las secuencias tomadas de aceleraciones lineales.

Para ello, una vez tomada una secuencia de aceleraciones, se calcula la media y se resta a cada valor de la secuencia, obteniendo así, secuencias de desviaciones típicas.

Esta alternativa se basa en que un valor de aceleración lineal durante la ejecución de un movimiento se forma de la suma de un valor de *Offset* con el valor de desviación.

$$Valor = Offset + Desviación$$

Siendo el valor de *Offset* la media de una secuencia de aceleraciones, siendo un valor dependiente de la velocidad con la que el usuario ejecuta el gesto. Aspecto que no es característico para el reconocimiento gestual, ya que buscamos identificar el movimiento realizado, no su velocidad.

Por otro lado, la *Desviación* depende de la trayectoria ejecutada por el usuario, y no de la velocidad de ejecución del movimiento. Por lo tanto, la *Desviación* será un valor característico para llevar a cabo la identificación gestual.

Exponemos, que esta alternativa es la que mejores resultados ha ofrecido, identificando de manera correcta todos los gestos dinámicos planteados en la biblioteca gestual.

Para implementar esta alternativa se debe almacenar previamente secuencias temporales que almacenen información sobre las desviaciones estándar de la ejecución de los movimientos planteados en la biblioteca gestual.

Para ello se ha desarrollado un pequeño Software que procesa secuencias de información sobre aceleración lineal, substrayéndole la media, y almacenando las secuencias con desviaciones en una carpeta, con extensión .mat.

En la implementación llevada a cabo se identifican los gestos llevados a cabo en el plano XZ, ya que tenemos los datos almacenados de los gestos dinámicos en este plano. Esta característica se utiliza como condición para iniciar el almacenaje de la secuencia gestual en nuestra implementación.

A continuación, se presenta en forma de flujograma la estructura del Software de reconocimiento de gestos dinámicos.

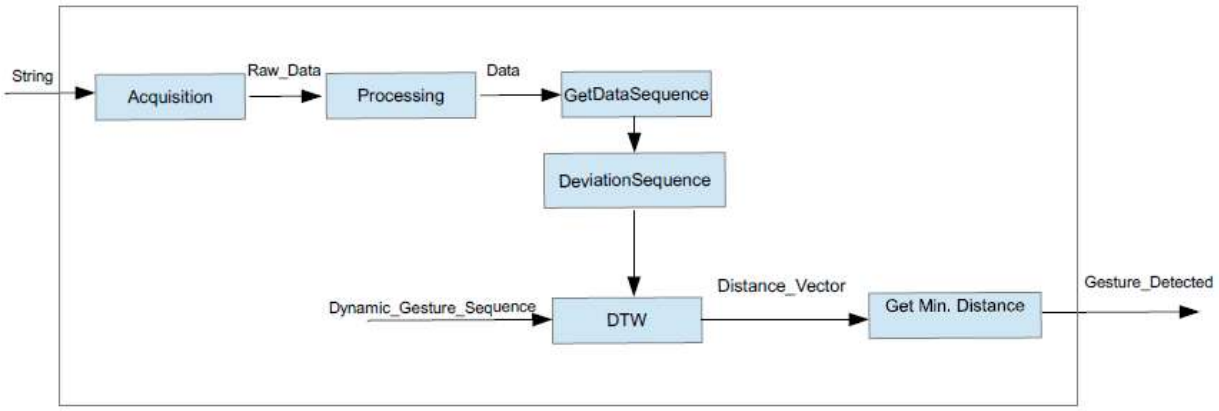


Figura 35. Dynamic Recognition

Los scripts de Matlab relacionados con la identificación de gestos dinámicos se pueden localizar en el *ANEXO I, Código.*

En cuanto al problema ***Diferenciar la tipología gestual en la fase de reconocimiento gestual*** debemos exponer que es un aspecto que no se ha podido resolver con éxito

En el proyecto se han optado dos pruebas para solventar este aspecto.

La primera opción implementada consiste en la ejecución, por parte del usuario, de los gestos en diferentes planos espaciales en relación a su tipología (dinámica o estática). Diferenciando el plano en el que nos encontramos podemos conocer la tipología gestual, y por tanto el tipo de reconocimiento a utilizar.

No es una solución muy generalizada y exitosa, sobre todo si se quieren ejecutar los gestos únicamente en un mismo plano.

Finalmente, se utilizó la aceleración inercial como identificador de la tipología gestual. No obteniendo resultados satisfactorios, ya que muchos gestos dinámicos solo presentan información característica sobre aceleración inercial en tramos concretos, lo que hace complejo generalizar la solución para todos los gestos.

Se concluye que las soluciones planteadas no han logrado solventar este problema planteado, por lo que es un punto claro de mejora en la aplicación.

6. DISEÑO DE PROTOTIPO EN PCB

En este apartado se van a desarrollar, en diferentes subapartados, las fases de diseño tomadas para la creación de un prototipo hardware en PCB para el proyecto.

6.1 Especificaciones de diseño.

Nuestro prototipo debe tener dimensiones reducidas, dentro de lo permitido por los componentes a utilizar, ya que debe ser insertado en la mano del usuario sin que sea pesado y molesto. Añadimos, que no se busca portabilidad en este diseño, debido a que un conjunto de baterías podría dar bastante peso al dispositivo, convirtiéndolo molesto durante su uso.

Necesidad de instrumentar magnitudes físicas tales como aceleración lineal, velocidad angular y flexión de las articulaciones de los dedos. Los dispositivos a utilizar en las mediciones son los comentados en el apartado *4. HARDWARE Y FIRMWARE DE INSTRUMENACIÓN.*

Por último, es necesaria la capacidad de comunicación con PC para llevar a cabo el reconocimiento gestual.

6.2 Diagrama de bloques.

En este subapartado se presenta el Diagrama de Bloques de nuestro prototipo.

El Diagrama de Bloques es una herramienta con la que buscamos describir las diferentes etapas electrónicas de un diseño de manera funcional que satisfacen las especificaciones de diseño planteadas.

En nuestro diseño, se plantean los siguientes bloques:

- Instrumentación de la flexión articular de los dedos (Flex Sensors).
- Instrumentación de aceleración lineal y velocidad angular (MPU-9250).
- Comunicación con PC.

- Unidad de control basada en microcontrolador con conversores analógico-digital y módulos para comunicación I2C y UART. Se utilizará el Atmega328p, encapsulado DIP, ya que es extraíble de la plataforma Arduino UNO.
- Fuente de Alimentación.

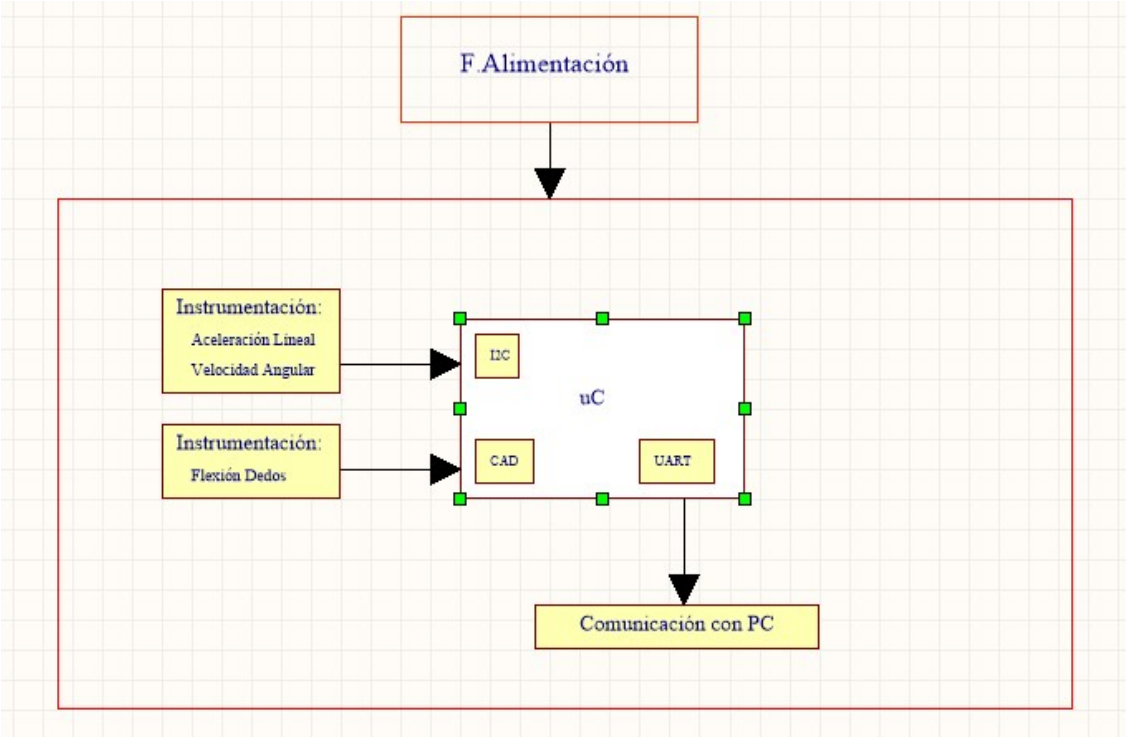


Figura 36. Diagrama De Bloques

6.3 Esquemático.

En este subapartado se presenta el Esquemático.

El Esquemático consiste en un plano donde se describen las interconexiones entre las etapas que conforma el diseño electrónico.

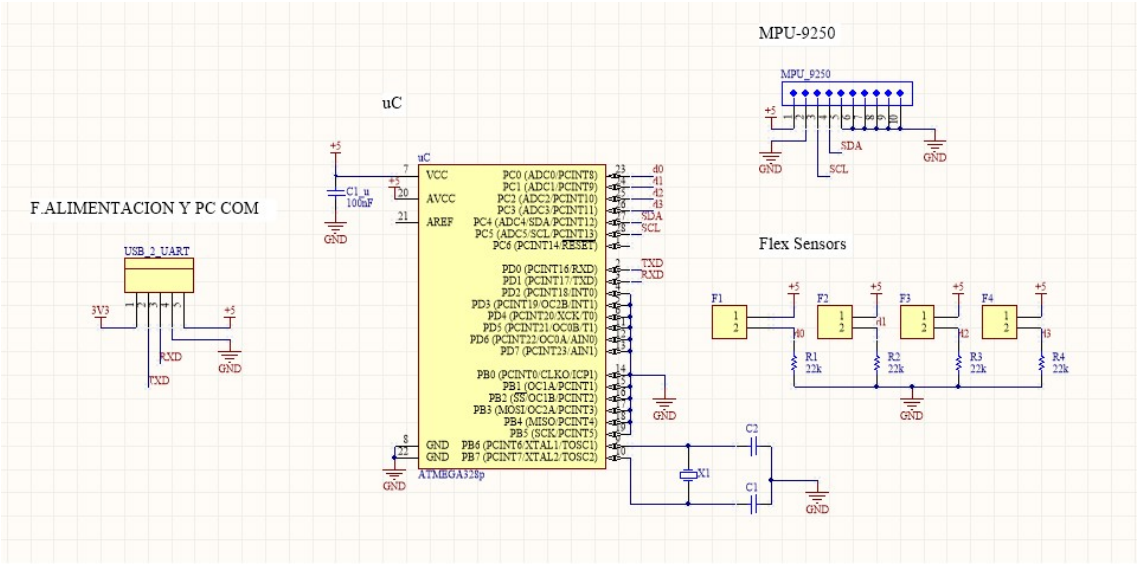


Figura 37. Esquemático

.

A continuación, se va a realizar una descripción de las interconexiones siguiendo como guía los bloques del Diagrama de Bloques.

- **Instrumentación de los Flex Sensor.**

Se utilizan cuatro Flex Sensor para medir la flexión articular de éstos, no mide el dedo pulgar, debido a que no es necesario para los gestos de la aplicación con interfaz NUI. Ahorrándonos un pin necesario para la comunicación I2C con el MPU-9250.

Se lleva a cabo un conexionado formando un divisor resistivo, resistencia de 22 KΩ y alimentación a +5 V.

Los Flex Sensor quedan conectados a la placa a través de dos pines designados en el esquemático como Fi, siendo i el número del dedo correspondiente.

La salida del divisor resistivo se lleva al conversor analógico-digital a través de los pines PC0, PC1, PC2 y PC3 del Atmega 328p.

Finalmente, el consumo que presenta un Flex Sensor en el peor de los casos, cuando se presenta resistencia mínima (22 KΩ), es de 0.23 mA. Teniendo en cuenta los cuatro sensores, 0.92 mA.

- **Instrumentación MPU-9250**.

Uso de comunicación I2C, sólo son necesarias dos líneas de conexión. Éstas son SDA, línea de datos, y SCL, reloj de la comunicación. Se observa que el dispositivo se forma de diez pines, por lo tanto, los pines sobrantes se conectan a masa, a excepción del pin de alimentación +5 V.

Observando la hoja de características de este dispositivo, podemos presentar que su consumo es despreciable.

- **Comunicación USB con PC.**

El Atmega 328p dispone de comunicación UART a través de las líneas TX y RX, estas líneas presentan niveles TTL (0 V – 5 V), lo que hace imprescindible un módulo de conversión de niveles para llevar a cabo una conexión USB con un PC.

Para ello, usamos el módulo PL2303, conexionado a la placa mediante cinco pines.

Pin TX del módulo se conecta con el pin RX del uC, y pin RX del módulo se conexiona con el pin TX del uC.

- **Fuente de Alimentación.**

El prototipo se alimenta a través del PC mediante USB, por lo que se presenta una tensión de alimentación de +5V y un consumo entre 500 y 900 mA que es suficiente para el prototipo, en relación con los consumos presentados por el resto de las etapas electrónicas.

6.4 Layout

En este subapartado se presenta el *Layout* de nuestra PCB.

El *Layout* consiste de una serie de planos que describen la disposición física en la PCB de los dispositivos e interconexiones entre etapas que conforman el diseño electrónico. En nuestro caso, se forma de dos planos, uno para la cara superior de la placa (**TOP**) y otro para la cara inferior de la placa (**BOTTOM**).

Antes de presentar los dos planos descritos, vamos a exponer a través de un pequeño diagrama la disposición de los componentes principales de nuestro diseño (*Placement*).

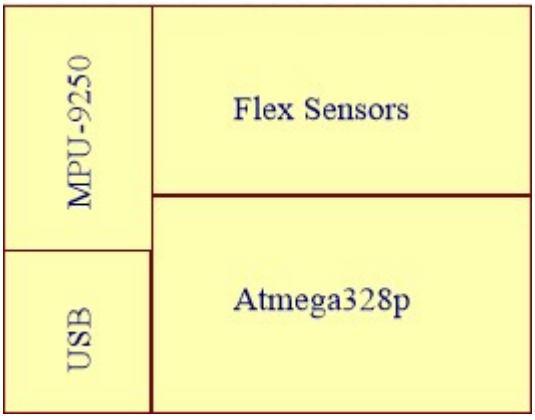


Figura 38. Placement

Esta disposición de los componentes se debe principalmente a que el dispositivo está pensado a usarse con la mano derecha, por eso la electrónica relacionada con la comunicación USB se encuentra a la izquierda de la PCB, haciendo más cómoda la ejecución de los gestos por parte del usuario.

La localización del conexionado de los Flex Sensors en la parte superior de la PCB se debe a que buscamos que estén los más cerca posible de los dedos de la mano. Consiguiendo que su cableado sea más seguro y no muy extenso en longitud.

En cuanto al **MPU-9250** y al **Atmega328p** se localizan lo más próximo posible al conformar una red de comunicación síncrona, **I2C**. Añadimos, que el **MPU-9250** va conexionado directamente a la PCB, ya que esta se dispone directamente en el antebrazo del usuario, teniendo en cuenta la orientación deseada para el **MPU-9250** en el *Placement.*

Respecto al cristal de cuarzo y condensadores de filtrado y desacoplo se ubican lo más cerca posible del microcontrolador, debido a que el cristal de cuarzo consiste de un oscilador de frecuencias elevadas (16 MHz). Gran generador de ruido.

A continuación, se presentan los planos del *Layout* en las caras **TOP** y **BOTTOM.**

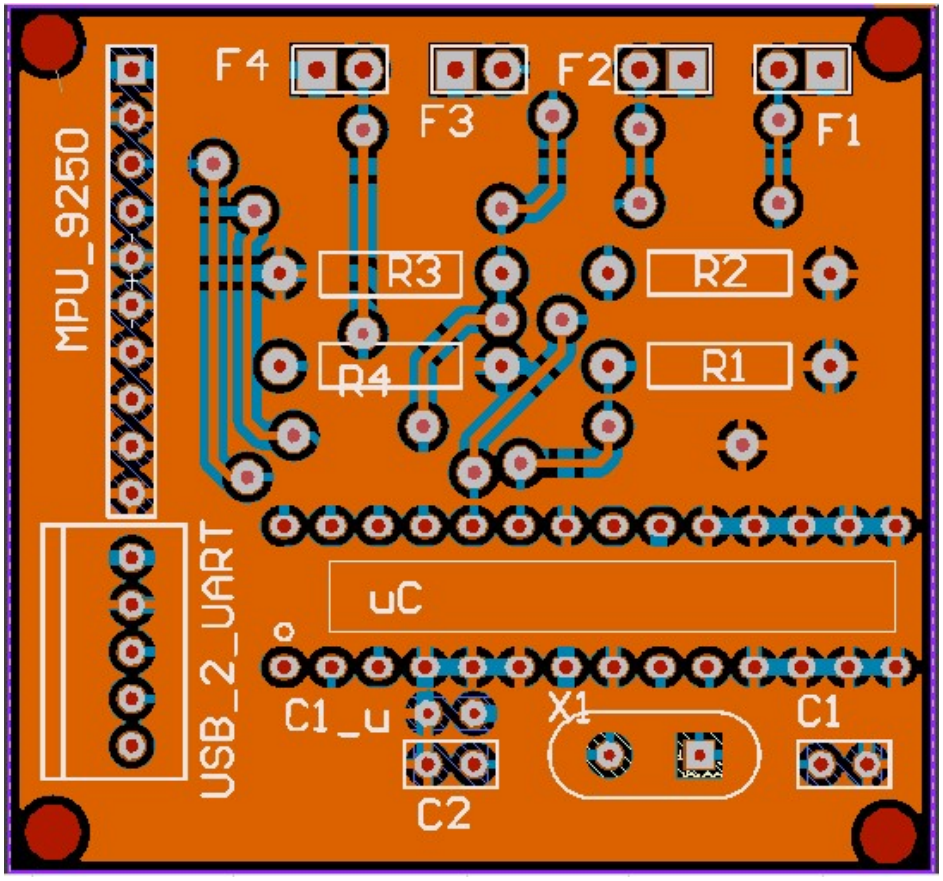


Figura 39. Top

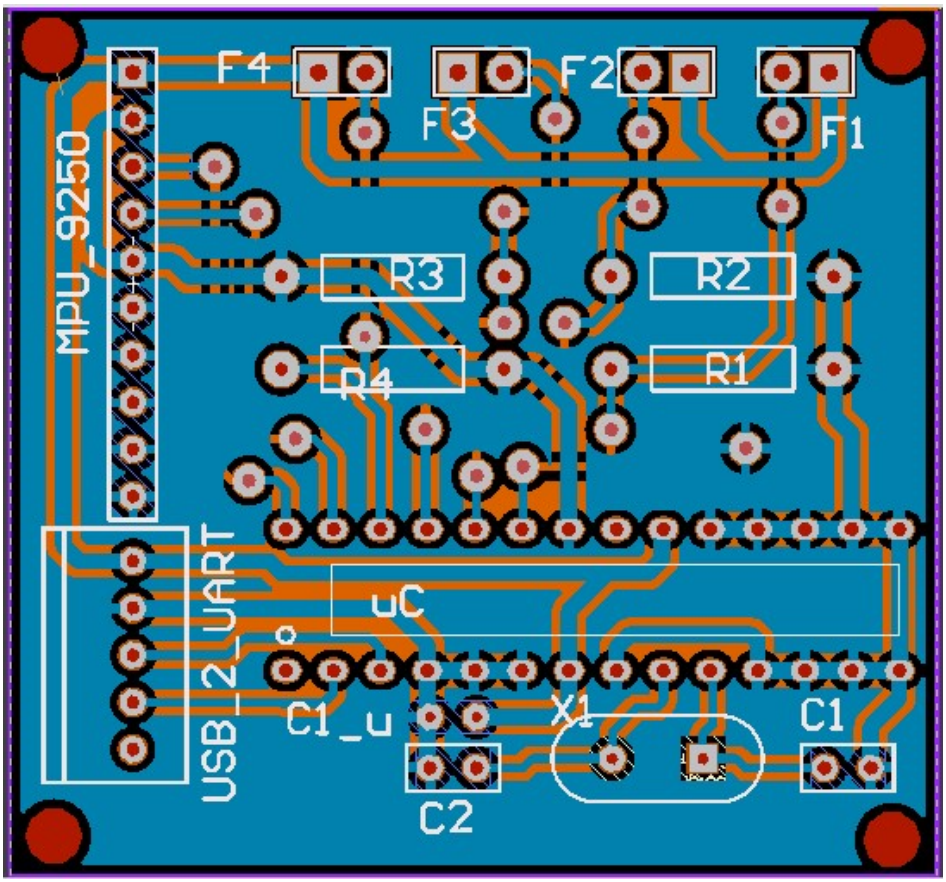


Figura 40. Bottom

En cuanto al *Layout* de la **PCB**, se han tomado grosores de pista de entre 0.90 mm y 1 mm para las pistas de alimentación (por presentan las mayores demandas de corriente). Por otro lado, se han tomado valores entre 0.45 mm y 0.5 mm para el resto de pistas, siendo el valor de 0.4 mm el grosor mínimo permitido por la fresadora del centro.

Añadimos que se ha tomada una separación entre pistas de 0.4 mm.

Respecto a las vías de la **PCB** se han tomado un pad de diámetro de 1.8 mm con un agujero de 0.8 mm, debido a que la placa va a ser soldada a mano. En relación a asignaturas de la titulación se conoce de forma experimental que valores son adecuados.

Para finalizar, exponemos que en primer lugar se traza las pistas de alimentación, ya que buscamos que sean lo más cortas posibles, ya que, al presentar mayores consumos y grosores, disminuimos posibles caídas de tensión, al minimizar la resistencia eléctrica. Después, se realiza el trazado del resto de pistas.

7. APLICACIÓN DEL PROTOTIPO COMO INTERFAZ NUI.

Hasta este apartado, se ha conseguido implementar un periférico para PC capaz de reconocer gestos manuales de tipología estática y de tipología dinámica.

Para conseguir los objetivos planteados en este proyecto debemos unir lo desarrollado a una aplicación para PC, creando una interfaz de tipo **NUI**.

Por el análisis gestual realizado y la biblioteca gestual elaborada creemos que es adecuada incluir este tipo de interfaz en una aplicación de reproducción multimedia. Añadimos que los gestos que conforman la biblioteca presentan relación con las acciones básicas de una aplicación de reproducción multimedia.

La aplicación a desarrollar se implementará en Python. Este lenguaje presenta menos herramientas matemáticas que Matlab, pero existen librerías útiles como soporte matemático.

La elección de Python se debe a que presenta menor complejidad para realizar aplicaciones ejecutables en el PC, además de tener más bibliotecas sobre reproducción de archivos multimedia en comparación con Matlab.

La aplicación consiste en la reproducción de un conjunto de archivos de audio en formato MP3 recopilados en una carpeta en el escritorio del PC llamada *Playlist TFG.* A la aplicación se le dotará de las funciones básicas de reproducción, usando gestos estáticos, como *play* y *pause*; junto con acciones como subir y bajar el nivel de audio, además de la posibilidad de silenciar la reproducción, *mute*. Por último, añadimos que también se ha dotado de acciones para cambiar la reproducción al archivo siguiente o al previo.

Para la gestión de archivos multimedia se utilizará el núcleo del reproductor multimedia de código abierto, VLC Media. En el link [19] se encuentra un enlace a la *web* de VLC Media para desarrolladores, con contenido sobre el uso del núcleo de VLC Media para aplicaciones por parte de terceros, útil para el desarrollo de la aplicación.

A continuación, se presenta el flujograma de la aplicación desarrollada:

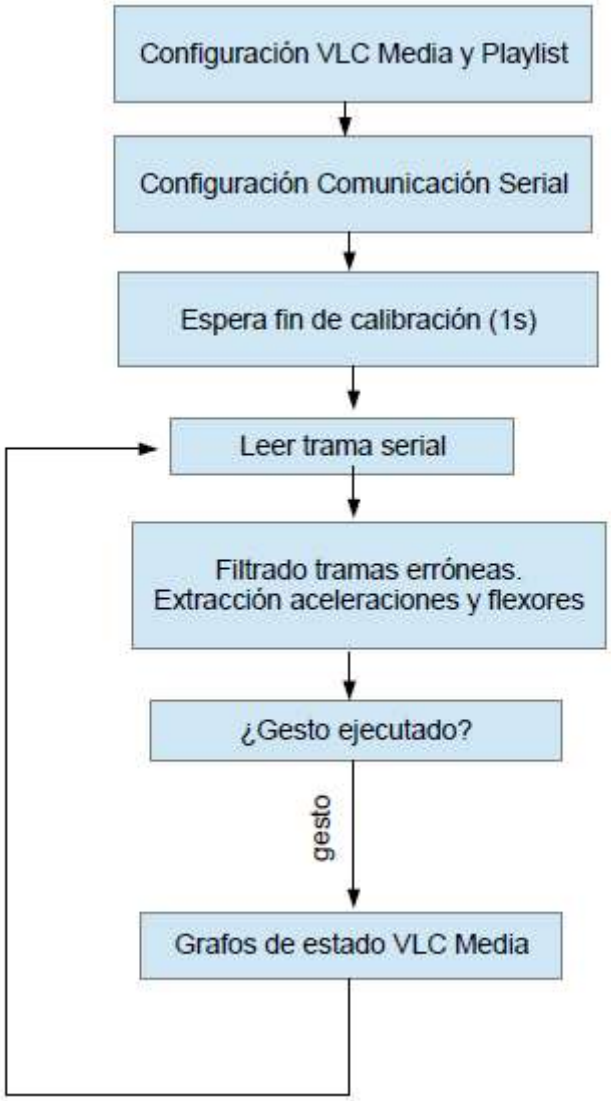


Figura 41. Flujograma Python

Para dotar a la aplicación de las funciones expresadas se ha implementado tres grafos de estado.

A continuación, los presentamos.

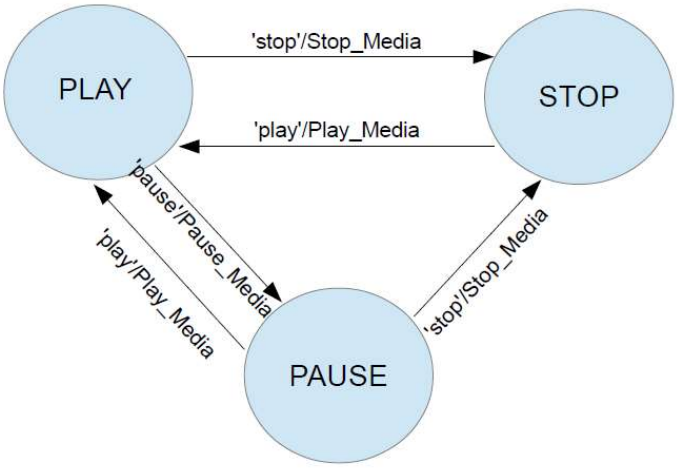


Figura 42. Rep_Grafo

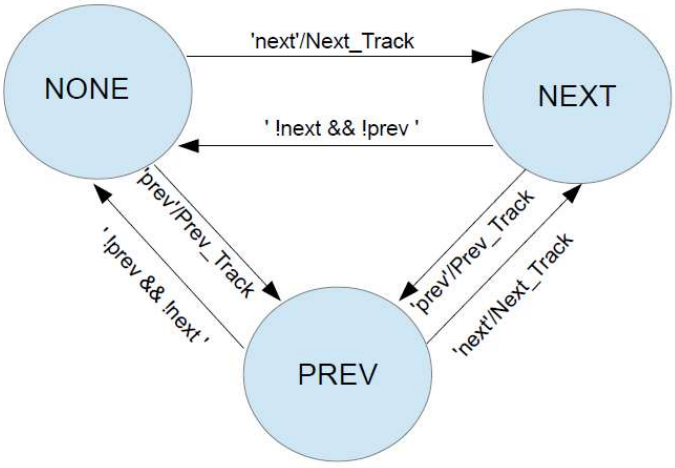


Figura 43. Grafo_Track

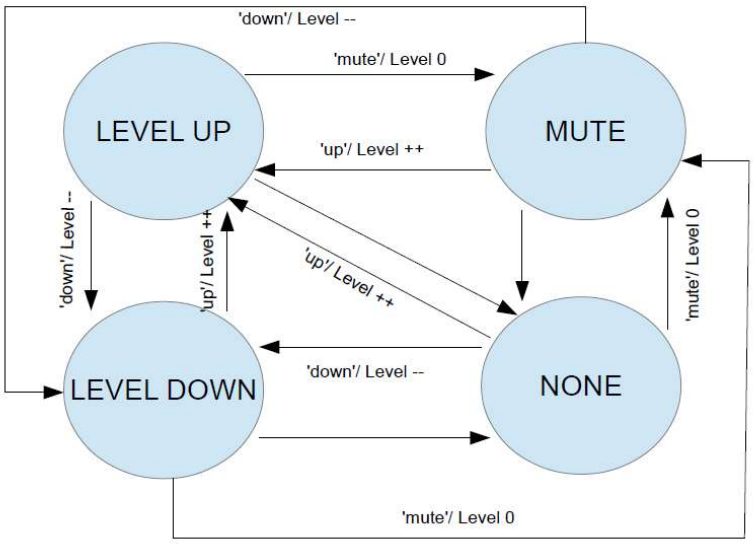


Figura 44. Grafo_Volume

Para el uso de las funciones de la aplicación se ha utilizado los siguientes gestos estáticos.

Gestos estáticos como *play* y *pause* para el manejo de la reproducción, *right* y *left* para el control de las pistas, y por último, los gestos *up*, *down* y *fist* para acciones relacionadas con el nivel de audio de reproducción.

Para finalizar añadimos que el código fuente de la aplicación se encuentra en el *ANEXO I, Código.*

8. CONCLUSIONES

En este apartado indicamos que las diferentes fases del proyecto se han desarrollado conforme a lo previsto, obteniendo un sistema electrónico útil como periférico **HMI** basado en reconocimiento gestual.

Este proyecto también engloba una aplicación informática de reproducción de archivos multimedia con interfaz **NUI**, donde se observa el funcionamiento del periférico desarrollado.

Respecto a problemas que se han planteado durante el desarrollo del proyecto, debemos destacar la existencia de pocos aportes técnicos sobre la identificación de gestos manuales con hardware específico debido a que es algo relativamente novedoso.

Gran parte de la bibliografía técnica tomada está relacionada principalmente con áreas como visión por computador y reconocimiento del habla. Además, la mayoría de los periféricos analizados en la revisión del apartado *2. ESTADO DEL ARTE* no son plataformas libres, siendo un aspecto negativo para el desarrollo del proyecto.

El reconocimiento de gestos dinámicos ha sido otro de los aspectos complejos del proyecto, solventándose a través del algoritmo **DTW** y el uso de secuencias temporales formadas por desviaciones típicas de las aceleraciones medidas.

El reconocimiento gestual dio lugar a otro problema, la identificación de la tipología gestual (dinámica o estática) durante la fase de reconocimiento, con el fin de utilizar de forma indiferente cualquier tipo de gesto en una misma aplicación. Al no lograrse con éxito este aspecto, será uno de los objetivos a estudiar en trabajos próximos (ver *9. MEJORAS Y FUTUROS TRABAJOS*).

En cuanto a una perspectiva más personal sobre el proyecto, expresamos que ha sido un experiencia positiva e interesante donde se ha aprendido a seguir una metodología adecuada a la hora de desarrollar proyectos de gran envergadura en comparación con trabajos y prácticas realizados a lo largo del grado.

Además de cubrir áreas estudiadas a lo largo de la titulación, como el diseño de placas de circuito impreso (**PCB**) y programación de microcontroladores, también han tenido lugar otros conocimientos que no están presentes en la titulación.

Conocimientos relacionados con áreas como la comunicación y la kinésica para el desarrollo del apartado *3. ANÁLISIS GESTUAL Y GENERACIÓN DE UNA BIBLIOTECA DE GESTOS*. Éstos han sido útiles para conocer qué tipos de comunicación se dan entre seres humanos y cuales se pueden desarrollar entre humanos y computadores utilizando gestos manuales.

Incluimos que este apartado también han sido necesarios conceptos básicos de física para obtener las magnitudes físicas a analizar en los gestos. Destacamos esto, porque es un aspecto, que, en ocasiones no se llega a desarrollar con totalidad. Es la aplicación de los conocimientos obtenidos en la titulación en aspectos de la vida cotidiana

También han tenido espacio conocimientos, más técnicos, relacionados con la informática, donde se ha aprendido un nuevo lenguaje de programación, Python. Además, se han reforzado

y ampliado los conocimientos obtenidos sobre comunicación serial al establecer una entre el periférico y el PC, junto con su gestión por parte del PC, tano en Matlab como en Python.

Destacamos el manejo de la documentación de librerías informáticas, útil para el desarrollo de la aplicación multimedia ayudado por librerías de VLC Media en Python.

Por último, expresamos que el proyecto desarrollado proporciona una nueva forma de interfaz entre usuario y máquina, por lo que este trabajo puede aportar nuevos estudios científicos que tenga impacto en el avance tecnológico actual.

9. MEJORAS Y FUTUROS TRABAJOS.

Como se ha indicado en *8. CONCLUSIONES,* queda un gran trabajo para el futuro en relación con esta área. Por lo tanto, vamos a presentar un conjunto de puntos a mejorar en nuestro proyecto, junto con trabajos futuros interesantes relacionados con el tema tratado.

En cuanto a la biblioteca gestual generada, presentamos que se forma de un pequeño conjunto de gestos manuales *básicos*, tanto dinámicos como estáticos. Por lo que planteamos como mejora la ampliación de la biblioteca gestual siguiendo los criterios utilizados en el apartado *3. ANÁLISIS GESTUAL Y GENERACIÓN DE UNA BIBLIOTECA DE GESTOS.*

Respecto al prototipo en placa de circuito impreso (**PCB**) presentamos como aspecto de mejora el desarrollar un prototipo más compacto y reducido, utilizando tecnología electrónica de montaje superficial (**SMD**) en vez de tecnología de agujero pasante (**TH**) y encapsulados **DIP**.

Además, se puede plantear dotarlo de comunicación inalámbrica con PC vía *Bluetooth* o *Wifi*, con el objetivo de lograr mayor comodidad en su uso al no utilizar cables. Este tipo de comunicación obligaría a dar portabilidad al dispositivo a través de pilas y/o baterías, pensando en el peso extra que conllevaría su inclusión, respecto con nuestro prototipo desarrollado.

En cuanto al software implementado, se pueden llevar a cabo mejoras para dar mayor robustez a la comunicación serial entre prototipo electrónico y PC.

Además de un estudio más amplio en cuanto a las técnicas de reconocimiento de gestos dinámicos (ya que es la tipología gestual más compleja de identificar), junto a la búsqueda de soluciones exitosas para diferenciar la tipología gestual durante la fase de reconocimiento con el objetivo de poder emplear de forma indiferente gestos dinámicos y estáticos en una misma aplicación.

En relación a la aplicación implementada en Python, se podría añadir una interfaz visual complementaria para darle un toque más estético. Otro aspecto a mejorar sería la elección de los archivos multimedia a reproducir en vez de la reproducción de una carpeta localizada en el escritorio.

Como el objetivo era realizar una pequeña demo donde mostrar que era posible emplear los gestos manuales para realizar funciones en una aplicación, no se han tenido en cuenta en el desarrollo de la aplicación los aspectos expresados en el párrafo anterior.

Respecto a trabajos futuros, podemos presentar la inclusión de la tecnología desarrollada en aplicaciones de PC diferentes de las multimedia (la llevada a cabo en el proyecto), como por ejemplo en videojuegos, manejo de presentaciones PowerPoint, Prezi; entre otras.

Otro punto importante, sería la adaptación de la tecnología desarrollada a personas con movilidad reducida u otros casos similares, dotando a estas personas de una comunicación más adecuada para ellos en comparación con los periféricos comunes de computado, como son el ratón y el teclado.

Para finalizar este apartado, presentamos que nuestro proyecto inicia una nueva forma de interfaz usuario-máquina, que, junto al éxito de otras áreas científicas como la inteligencia artificial y la realidad virtual, y con el desarrollo notable en la tecnología actual, expresamos que en un futuro no muy lejano aparecerán dispositivos electrónicos novedosos que establezcan entornos y sensaciones similares a los vistos en el cine de ciencia ficción.

Observamos que la evolución científica nos va acercando a estos entornos, tecnológicamente increíbles, de ciencia ficción.

En parte, nuestro proyecto aporta un pequeño granito de arena a este proceso evolutivo, concluyendo que no existen límites en cuanto a posibles trabajos futuros, ya que corresponden con la propia imaginación humana.

Por ello, presentamos la siguiente cita de *Julio Verne* que refleja de forma compacta lo expresado en los dos párrafos anteriores*:*

Todo lo que yo invento, todo lo que yo imagino, quedará siempre más acá de la verdad, porque llegará un momento en que las creaciones de la ciencia superarán a las de la imaginación.

10. BIBLIOGRAFÍA y LINKOGRAFÍA.

- Bibliografía

[1] C. A. Castillo-Benavides, L. F. García-Arias, N. D. Duque-Méndez y D. A. Ovalle Carranza, IMU-Mouse: diseño e implementación de un dispositivo apuntador dirigido al desarrollo de interfaces adaptativas para personas con discapacidad física. *TecnoLógicas*, vol. 21, no. 41, pp. 63-79, 2018.

[2] Nota de Aplicación, AN-1008 Sensors for Orientation Estimation, 20/10/12, CH Robotics.

[3] Hiroaki Sakoe & Seibi Chiba (1978) Dynamic Programming Algorihm Optimization for Spoken Word Recognition, IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING.

[4] Donald J.Berndt & James Clifford (1994) Using Dynamic Time Warping to Find Patterns in Time Series, Information Systems Department, New York University.

[5] Muhammad P. & Anjana Devi S. (2016) Hand Gesture User Interface for Smart Devices Based On MEMS Sensors, 6th International Conference On Advances In Computing & Communications, India.

[6] Ruize Xu, Shengli Zhou and Wen J. Li. (2012) MEMS Accelerator Based Nonspecific-User Hand Gesture Recognition, IEEE Sensors Journal.

[7] Antonio Romeo Tello. (Curso 2016/17) Cinemática de manipuladores. Robótica Industrial. Ingeniería en Electrónica y Automática. EINA, Dpto. Informática e Ingeniería de Sistemas.

- Linkografía

[8] https://es.wikipedia.org/wiki/Interfaz_de_usuario

[9] https://sites.google.com/site/perifericostecnoieselbohio/historia-de-los-perifericos

[10] https://www.um.es/docencia/barzana/IATS/Iats05.html#BM2

[11] https://www.xataka.com/historia-tecnologica/17-mandos-que-demuestran-lo-mucho-que-ha-cambiado-nuestra-forma-de-jugar-a-videojuegos-en-consolas

[12] https://blogthinkbig.com/soli-el-sensor-que-leera-tus-gestos

[13] http://www.analisisnoverbal.com/lenguaje-corporal-y-comunicacion-no-verbal/

[14] http://jujomaruiit.blogspot.com/2015/09/fusion-del-acelerometro-y-giroscopio.html

[15] http://www.chrobotics.com/library/accel-position-velocity

[16] https://store.arduino.cc/arduino-uno-rev3

[17] https://www.invensense.com/download-pdf/mpu-9250-datasheet/

[18] https://www.sparkfun.com/products/8606

[19] https://www.videolan.org/developers/vlc.html

[20] https://docs.python.org/3/

[21] https://www.infobae.com/tecno/2017/07/16/11-tecnologias-que-anticiparon-las-peliculas-de-ciencia-ficcion-y-hoy-son-realidad/