CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITÉCNICO NACIONAL

UNIDAD ZACATENCO

DEPARTAMENDO DE INGENIERÍA ELÉCTRICA

SECCIÓN DE ELECTRÓNICA DEL ESTADO SÓLIDO

"Diseño de un sistema para escritura y dibujo virtuales, basado en un acelerómetro"

T E S I S

Que presenta:

ING. CARMEN CARITINA MUÑOZ GARNICA

Para obtener el grado de:

Maestra en Ciencias

En la especialidad de:

Ingeniería Eléctrica

Directores de Tesis:

Dr. Mario Alfredo Reyes Barranca

Dra. Griselda Stephany Abarca Jiménez

Ciudad de México Febrero, 2021.

AGRADECIMIENTOS

A Mis Padres: Por ser la motivación de cada uno de los pasos que doy día a día, por ser mi más grande apoyo, por estar para mi incondicionalmente y por todos y cada uno de los sacrificios y actos de amor que han hecho por mí, porque sin ustedes yo no sería nada.

A mi esposo Aldo: Por ser mi guía y mi apoyo cada día, por brindarme la fuerza necesaria para salir adelante y estar conmigo en las buenas y malas. Gracias por toda tu ayuda en esta etapa y por ser esa motivación que necesité cada vez que quise darme por vencida, por toda la paciencia y el cariño; sólo puedo estar agradecida contigo por todo lo que haces por mí.

A mi familia: A mis sobrinos, mi hermana y cuñado, por haber sido tan pacientes conmigo durante este tiempo, pero en especial, por su compañía y apoyo, porque siempre han estado ahí para brindarme su cariño cuando los he necesitado.

A mis asesores, el Dr. Alfredo Reyes y la Dra. Stephany Abarca: Por recibirme en su grupo de trabajo y haber puesto su confianza en mí, por el apoyo incondicional durante mi etapa de maestría y por toda la experiencia y la guía aportada durante la elaboración de este trabajo, pero especialmente por su comprensión y entendimiento para conmigo.

Al grupo de VLSI-SEES: Por todas las valiosas contribuciones y enseñanzas que hicieron para poder realizar este trabajo de tesis y por su grata amistad.

A los miembros del jurado: Por cada una de las enseñanzas otorgadas para mejorar y avanzar en el desarrollo de este proyecto y por sus atinados comentarios y aportaciones.

Al CINVESTAV: Por abrirme las puertas y brindarme los recursos necesarios para poder desempeñarme en esta nueva etapa de mi vida profesional y por poner a las personas correctas, junto con extraordinarios profesores en mi camino.

Al CONACyT: Por la beca otorgada para poder realizar mis estudios de Maestría a través del apoyo 489059 con número de registro de becario 634981.

RESUMEN

Este documento de tesis se enfoca en el desarrollo de un dispositivo electrónico capaz de realizar trazos virtuales al aire sin ninguna superficie de apoyo basado en el funcionamiento de un acelerómetro, comprendiendo así el diseño, fabricación y caracterización del sistema electrónico. Y que éste, a su vez, cuente con características de portabilidad, permita la transferencia de los datos en tiempo real, sea versátil, amigable con el medioambiente, y que, mediante un segundo dispositivo conectado de forma inalámbrica, se permita la visualización y el almacenamiento del trazo realizado mediante una interfaz gráfica de usuario; todo realizado en su mayor parte con lenguaje de programación y software libre de licencias.

Los elementos principales que componen a este sistema son una unidad de medición inercial (IMU), el módulo MPU6050, un microcontrolador para el procesamiento de los datos, el PIC18F4550, y un dispositivo de comunicación inalámbrico, el módulo bluetooth HC-05, para enviar la información de manera remota y procesada a un segundo dispositivo electrónico que funcionará como interfaz gráfica de usuario.

Este trabajo está conformado principalmente por cuatro etapas que son: el desarrollo de un algoritmo para el libre trazo, mismo que se basa en un filtro complementario utilizando el cálculo de los ángulos de inclinación de la IMU, el diseño y fabricación del circuito impreso del sistema electrónico, el desarrollo de la interfaz gráfica de usuario y el diseño de la cubierta del dispositivo.

Abstract.

This thesis document focuses on the development of an electronic device capable of making virtual traces in the open air without any support surface based on the operation of an accelerometer, thus understanding the design, manufacturing, and characterization of the electronic system. And that this, in turn, has portability characteristics, allows the transfer of data in real-time, is versatile, friendly to the environment, and that, through a second device connected wirelessly, gives features of visualization and storage of the trace made by means of a graphical user interface; all done mainly with a programming language and license-free software.

The main elements that make up this system are an inertial measurement system, the MPU6050 module, a microcontroller for data processing, the PIC18F4550, and a wireless communication device, the HC-05 Bluetooth module, to send remotely the processed information. to a second electronic device that will function as a graphical user interface.

This work consists mainly of four stages that are: the development of an algorithm for the free trace, which is based on a complementary filter using the calculation of the inclination angles of the IMU, the design and manufacture of the system's printed circuit. electronics, graphical user interface development, and the device case design.

Índice de Contenido

Índice de Figuras

Índice de Tablas

Índice de Ecuaciones

Capítulo 1

Introducción

Las siglas MEMS son un acrónimo para denotar a lo que actualmente se conoce como Sistemas Microelectromecánicos (*Microelectromechanical systems*). Son definidos típicamente como dispositivos de pequeñas dimensiones compuestos por elementos activos y pasivos microfabricados y que realizan diferentes funciones como percepción, procesado de datos, comunicación y actuación sobre el entorno. Los tipos de dispositivos MEMS pueden variar desde estructuras relativamente simples que no tienen ninguna parte móvil, hasta sistemas electromecánicos muy complejos en los que múltiples elementos se mueven bajo el control de la electrónica integrada.

Gracias a los avances en el campo de los semiconductores, los MEMS parten de tecnología que puede aplicarse utilizando una gran diversidad de materiales y técnicas de fabricación; la elección dependerá del tipo de dispositivo que se pretenda fabricar y el sector comercial en el que desee operar. Esta visión de los MEMS, donde microsensores, microactuadores, microelectrónica y otras tecnologías se puedan integrar en un microchip, se espera que sea uno de los avances más importantes del futuro. Los MEMS en general varían en tamaño pues pueden ir desde un micrómetro (una millonésima parte de un metro) a un milímetro (una milésima parte de un metro). En este nivel de escala de tamaño, las construcciones de la física clásica deben abordarse con consideraciones particulares por los efectos que se presentan a estas dimensiones.

Debido a la gran superficie con relación al volumen de los MEMS, llamado comúnmente relación de aspecto, los efectos de superficie como electroestática y viscosidad dominan a los efectos de volumen tales como la inercia o la masa térmica. Por ello, la etapa de diseño y caracterización de los microsistemas será básica de cara al análisis y desarrollo de nuevas aplicaciones.

La motivación para el uso de esta tecnología con respecto a los dispositivos utilizados hasta ahora es que ofrece, además de un costo menor, un menor consumo, un peso más reducido y un alto desempeño.

El trabajo reportado en esta tesis se enfoca en el desarrollo de aplicaciones de dispositivos MEMS. Dentro de las actividades llevadas a cabo en el Grupo de Diseño de Sistemas VLSI de la SEES, se han hecho desde hace años propuestas de estructuras micro-electro-mecánicas con base a transistores MOS de compuerta flotante (FGMOS), que son fabricados con tecnología CMOS. Esto ha dado lugar a propuestas novedosas de acelerómetros, sensores de presión y micromotores lineales y radiales desarrollados aplicando las reglas de diseño de la tecnología CMOS, pero que sacando ventaja de ciertos nichos presentes en estas reglas, se pueden proponer estructuras que pueden ser micromaquinadas posterior a su fabricación, de tal forma que en un mismo dado o chip, se cuente tanto con la estructura tridimensional del dispositivo MEMS sensor o actuador, así como la electrónica de control y procesamiento de señal, logrando lo que se le ha llamado tecnología CMOS-MEMS, sin tener que recurrir a tecnologías MEMS dedicadas, que son muy especializadas, caras y que no tienen la opción de incluir dispositivos electrónicos sobre el mismo chip. Entonces, para continuar sobre la línea de investigación de los MEMS, en esta tesis se hace ahora la propuesta de la aplicación de un dispositivo MEMS comercial, para el desarrollo de un sistema cuyo núcleo sea un dispositivo inercial, complementado con electrónica periférica para la adquisición, procesamiento y transmisión de la señal a considerar. De esta manera, el Grupo de Diseño de Sistemas VLSI de la SEES incursiona en actividades que tienen amplias posibilidades para abordar y proponer y que son diferentes a la actividad de diseño de estructuras propuestas para su fabricación mediante tecnologías CMOS.

1.1. ANTECEDENTES

1.1.1. Sistemas Microelectromecánicos

Los sistemas microelectromecánicos (MEMS) son pequeños dispositivos integrados o sistemas que combinan componentes eléctricos y mecánicos. Su

tamaño varía desde el nivel submicrométrico (o submicrónico) hasta el nivel milimétrico, y puede haber cualquier número, desde unos pocos hasta millones, en un sistema en particular. MEMS extiende las técnicas de fabricación desarrolladas para la industria de circuitos integrados para agregar elementos mecánicos como vigas, engranajes, diafragmas y resortes a los dispositivos.

Ejemplos de aplicaciones de dispositivos MEMS incluyen cartuchos de impresoras de inyección de tinta, acelerómetros, robots en miniatura, micro-máquinas, cerraduras, sensores inerciales, micro-transmisores, microrreductores, microaccionadores, escáneres ópticos, bombas de fluidos, transductores y sensores químicos, de presión y de flujo. Están surgiendo nuevas aplicaciones a medida que la tecnología existente se aplica a la miniaturización e integración de dispositivos convencionales, en áreas tanto industriales, médicas, automotrices, biomédicas, aeroespaciales, de consumo, etc.

Estos sistemas pueden detectar, controlar y activar procesos mecánicos en la microescala, y funcionan individualmente o en matrices para generar efectos en la macroescala. La tecnología de micro fabricación permite la fabricación de grandes conjuntos de dispositivos, que individualmente realizan tareas simples, pero en combinación pueden realizar funciones complicadas.

Los MEMS no se refieren a ninguna aplicación o dispositivo, ni están definidos por un solo proceso de fabricación o están limitados a unos pocos materiales. Se enmarcan en un enfoque de fabricación que transmite las ventajas de la miniaturización, el uso de múltiples componentes y el empleo de la microelectrónica para el diseño y construcción de sistemas electromecánicos integrados. Los MEMS no tienen que ver únicamente con la miniaturización de los sistemas mecánicos, sino que también son un nuevo paradigma para el diseño de dispositivos y sistemas mecánicos. [1]

1.1.2. Acelerómetros.

Un acelerómetro como se intuye por su nombre es una herramienta para medir la aceleración de un objeto al que va unido, lo hace midiendo respecto de una masa inercial interna.

Existen diversos tipos de tecnologías y diseños que, aun cuando todos poseen el mismo propósito que es medir la aceleración tienen la posibilidad de ser bastante diversos unos de otros conforme a la aplicación para la cual van destinados y las condiciones en las que han de desempeñarse. [2]

Al momento de elegir el dispositivo apropiado deben considerarse dos parámetros, que son los rangos de manejo, tanto de temperatura como de frecuencia de operación. Otros parámetros relevantes son, además, la medida y la gravedad; no obstante, si poseen más funcionalidades a hacer, se tiene la posibilidad de agregar la resistencia a golpes, y, desde luego, el costo comercial.

La versatilidad de los acelerómetros los ha llevado a prolongar su aplicación más allá del uso industrial y de la investigación, puesto que en la actualidad permanecen presentes en varios artefactos de uso diario, como lo son el Nintendo, Wii, Footpod y dispositivos portátiles, entre otros.

Con base a lo anterior y en particular, el presente trabajo reporta la aplicación del acelerómetro MPU6050 para el diseño, fabricación y caracterización de un sistema electrónico capaz de escribir de manera virtual y a pulso levantado, sin ser apoyado en alguna superficie, y que además permita la visualización y almacenamiento del trazo realizado en un segundo dispositivo a través de una conexión por bluetooth, controlado por un microcontrolador.

La aportación que se persigue en esta área de conocimiento es desarrollar dispositivos electrónicos capaces de transferir la caligrafía fiel del usuario a sus medios electrónicos por medio de un algoritmo de seguimiento de trayectoria. Se debe entender que la dificultad de este algoritmo recae en que el sistema no cuenta con ningún tipo de locomoción donde se pudiera incluir un sistema de retroalimentación para control. Por lo anterior el reto en este trabajo consiste en implementar un algoritmo de rastreo de trayectoria de la pluma y determinar en dónde inicia el trazo y en dónde termina.

1.1.3. Teoría de las Aceleraciones.

La aceleración se define como la razón entre el cambio de velocidad y el intervalo en el cual ésta ocurre. Es decir, mide qué tan rápidos son los cambios de velocidad de un objeto.

La aceleración es una cantidad vectorial, es decir, tiene una magnitud, la cual define el tamaño de la aceleración y una dirección que define hacia dónde apunta dicha aceleración. Es por eso por lo que se puede decir que:

- Una aceleración grande significa que la velocidad cambia rápidamente.
- Una aceleración pequeña significa que la velocidad cambia lentamente.
- Una aceleración cero significa que la velocidad no cambia.

Si un móvil está disminuyendo su velocidad, entonces su aceleración va en sentido opuesto al desplazamiento. En cambio, si éste aumenta su velocidad, la aceleración tiene el mismo sentido que el propio objeto. [3]

Las técnicas convencionales para identificar y medir aceleraciones se fundamentan en el Principio de Newton (1687). Este principio es además conocido como La Segunda Ley de Newton la cual nos dice: "que la fuerza neta aplicada sobre un cuerpo es proporcional a la aceleración que adquiere dicho cuerpo". Una forma más rigurosa y clara sería su expresión matemática (Ec. 1.1), la cual relaciona tres magnitudes como son fuerza *(F)*, masa *(m)* y aceleración *(a)*:

$$F = m \cdot a \qquad \text{ECUACIÓN 1.1}$$

Muchos acelerómetros operan detectando la fuerza ejercida en una masa por una limitación elástica. Considerando un sistema mecánico simple como el mostrado en la Figura 1.1, que consiste en una masa fija (m), con un muelle o resorte con una rigidez (k) constante. Si la masa se desplaza una distancia x, la aceleración debida a la fuerza restauradora del muelle o resorte es $F = k \cdot x$, conocida como La Ley de Hooke. Para términos prácticos en esta tesis se hace caso omiso al signo menos que se incluye en la ley de Hooke, ya que como recordará el lector, la masa inercial de estudio está referenciada sobre el objeto mismo en movimiento y la aceleración que experimenta es en sentido opuesto a la aceleración del objeto a medir, por lo cual resulta correcta esta simplificación.

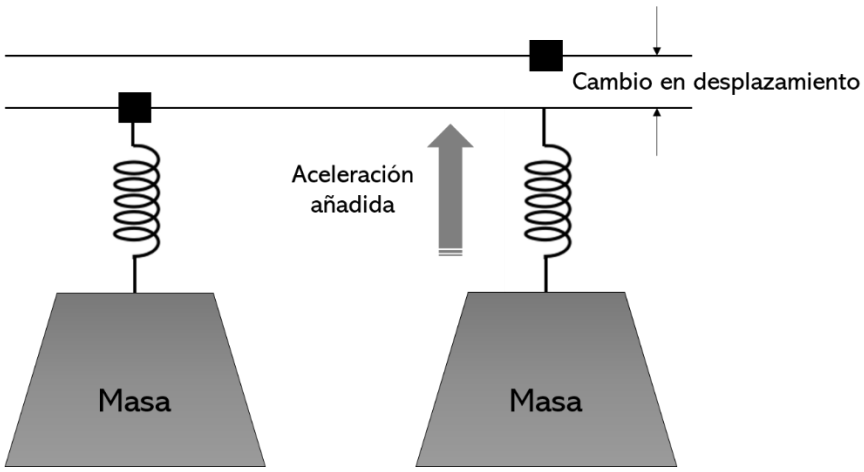


FIGURA 1.1 PRINCIPIO DE LA ACELERACIÓN

Substituyendo en la ecuación de Newton, encontramos que $a = k \cdot x / m$, y es posible derivar la magnitud de la aceleración observando el desplazamiento x de la masa fija. Este principio fundamental se utiliza hasta en el más sofisticado y caro acelerómetro electromecánico; así también trabajan los modernos acelerómetros micro mecanizados. Como ya se había mencionado antes, la aceleración es el cambio de la velocidad y la unidad de medida es: m/s^2, aunque es posible encontrar referencias de acelerómetros cuyo rango de actuación sea de varios g, donde g tiene la equivalencia como $1g = 9.8m/s^2$. [2]

Este tipo de sensores es ampliamente utilizado para determinar tanto la inclinación de un objeto como su vibración, cuando se utiliza en configuración estática. Para el caso de aplicaciones dinámicas, estos sensores se usan para determinar la aceleración traslacional en un sistema de ingeniería. Este tipo de sensores pueden estar basados en el principio de transducción piezorresistivo, piezoeléctrico o capacitivo; sin importar qué tipo de principio de transducción se utilice, el sensor tendrá una salida lineal, esto quiere decir que al impulso o inclinación que reciban a la entrada, se observará una respuesta proporcional a la salida (segunda ley de Newton y ley de Hooke). [4]

En resumen, un acelerómetro consiste en una masa móvil que al percibir aceleración externa genera un desplazamiento proporcional al cociente entre la fuerza aplicada y su rigidez asociada.

1.2. PLANTEAMIENTO DEL PROBLEMA

La idea de este proyecto surge para resolver varios problemas que actualmente presentan los sistemas convencionales como:

- **Portabilidad:** Los sistemas digitales para captura de trazo requieren de un sistema electrónico con una superficie especial de apoyo, este dispositivo está pensado para que funcione de manera inalámbrica y se pueda conectar a cualquier dispositivo electrónico que cuente con la aplicación para su captura y almacenamiento.
- **Permite compartir la idea/trazo en tiempo real:** El trazo se captura en la aplicación y se puede compartir de manera inmediata. El sistema permite capturar de manera digital, permitiendo al mismo tiempo rescatar la caligrafía y estilo del usuario.
- **Siempre conectado:** La interfaz del dispositivo se conectará en tiempo real con la finalidad de estar siempre conectado, sin la necesidad de cargar un teclado, una superficie de apoyo o papel.
- **Versatilidad:** Se busca que el dispositivo tenga la forma de una pluma convencional para facilitar su uso.
- **Ecológico:** Dado que este dispositivo no requiere de tinta, papel, cables o un soporte en especial, la basura generada es mínima.

1.3. OBJETIVO

Diseñar, fabricar y caracterizar un sistema electrónico capaz de escribir en el aire, sin ninguna superficie de apoyo, que sea portable (inalámbrico), que permita la transferencia de los datos (el trazo) en tiempo real, que sea versátil (con un diseño en forma de pluma), y amigable con el medioambiente, y que, mediante un segundo dispositivo conectado de forma inalámbrica, se permita la visualización y el almacenamiento del trazo realizado, basado en el funcionamiento de un acelerómetro.

FIGURA 1.2 DISPOSITIVO EN FORMA DE PLUMA

La idea que se plantea en este protocolo es desarrollar un dispositivo electrónico como el que se muestra en las Fig. 1.2, 1.3 y 1.4. Los elementos principales que componen a este sistema son sistema de medición inercial, un microcontrolador para el procesamiento de los datos y un dispositivo de comunicación inalámbrico para enviar la información procesada al dispositivo electrónico que funcionará como interfaz.

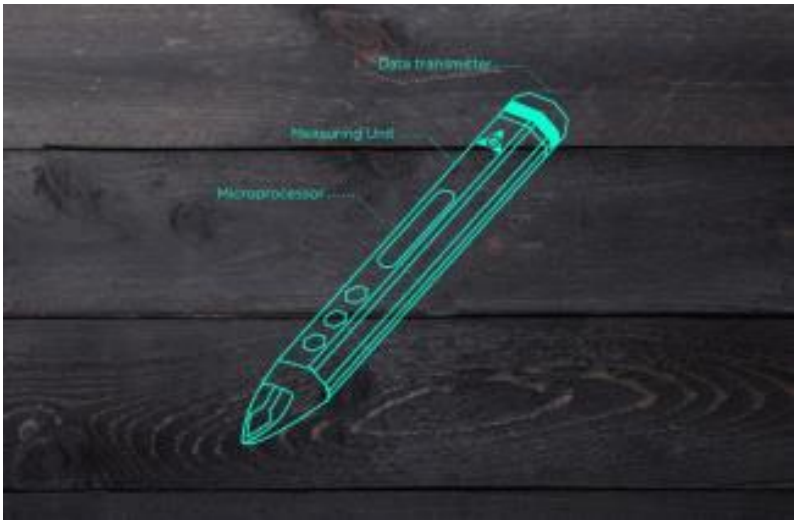


FIGURA 1.3 ELEMENTOS BÁSICOS DEL DISPOSITIVO PARA SU FUNCIONAMIENTO

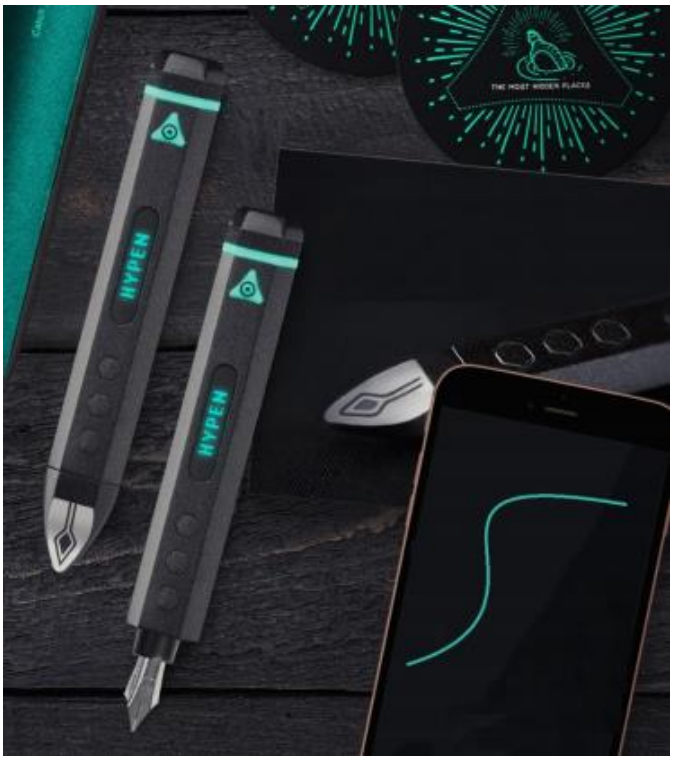


FIGURA 1.4 CONEXIÓN INALÁMBRICA ENTRE EL DISPOSITIVO Y UNA INTERFAZ PARA LA VISUALIZACIÓN Y CAPTURA DEL TRAZO

1.3.1. Objetivos Específicos

Para alcanzar el objetivo principal mencionado, se plantean los siguientes objetivos particulares:

- Familiarizarse con el sensor acelerómetro y su comunicación mediante el protocolo I^2C.
- Desarrollo de una interfaz simple en lenguaje Python para visualizar la señal del acelerómetro.
- Establecer el marco de referencia y el tipo de activación para inicio de trazo.
- Desarrollo del algoritmo para el libre trazo.
- Adaptar y transferir el algoritmo y el acelerómetro a un sistema miniaturizado.

- Desarrollo de una interfaz simple en Python para observar el trazo en forma de PDF/JPEG/etc.

1.4. JUSTIFICACIÓN

El impacto de la integración de sensores, actuadores y elementos electrónicos a nivel micrométrico proporciona soluciones con bajo consumo de energía, reducción en el tamaño de los componentes, mayores anchos de banda de trabajo, mejorando la calidad de vida, desarrollando mejores aplicaciones y dispositivos en diversas áreas entre ellas biomedicina (salud y calidad de vida), telecomunicaciones (radiofrecuencia y redes ópticas), automotriz y aeroespacial por mencionar algunas. Sin embargo, algunos otros, también son utilizados con fines tecnológicos, es decir, su uso se dirige más al área comercial y al desarrollo de dispositivos destinados al entretenimiento o a la prestación de servicios de menor complejidad, pero de mayor utilidad en las actividades diarias.

En este aspecto, es que, siendo los acelerómetros un área importante de estudio dentro del Grupo de Sistemas de VLSI de la SEES, se tomó la decisión de realizar un proyecto con miras tecnológicas y cuyo enfoque principal sea la aplicación de un acelerómetro de tipo comercial en un dispositivo que sea de utilidad para realizar una tarea cotidiana, mostrando así la versatilidad de sus usos.

La propuesta desarrollada en el presente trabajo es que a partir de un algoritmo basado en un filtro complementario que emplea y aprovecha ventajosamente la detección y procesamiento de los ángulos de inclinación del acelerómetro en tres dimensiones y los ángulos de rotación del giroscopio alrededor del eje X y del eje Y sea posible trazar la trayectoria que siga el movimiento del dispositivo, el cual tiene como finalidad la reproducción virtual de la escritura y trazos del usuario, complementando así el desarrollo de un dispositivo que sea capaz de transmitir esta información en tiempo real para su visualización y almacenamiento, lo que haría que presente características innovadoras frente a otros en su tipo.

1.5. ESTADO DEL ARTE

Nombre de la Patente:	Electronic Pen Device
Inventor:	Hsien-Chung Chou
No. de Publicación:	US200502311488A1
Procedencia:	Estados Unidos
Fecha de Publicación:	20 de Octubre de 2005.

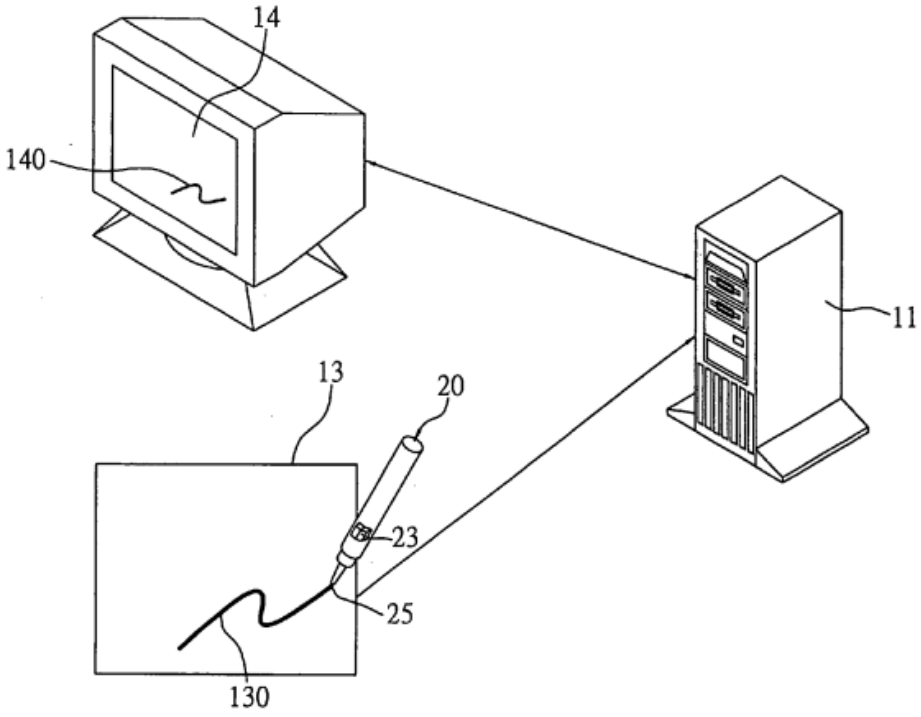


FIGURA 1.5 PATENTE US20050231488A1

Propone un dispositivo de lápiz electrónico, que se puede acoplar a una plataforma informática que tiene una unidad de visualización y una unidad de detección de entrada, lo que permite al usuario introducir manualmente datos y /

o gráficos en la plataforma de la computadora a través del dispositivo de lápiz electrónico. El dispositivo de lápiz electrónico incluye una unidad de microprocesamiento (MPU), una unidad de almacenamiento para almacenar códigos correspondientes al color, una unidad de entrada que permite al usuario establecer un color de visualización para los datos de escritura a mano y / o gráficos que se mostrarán en la pantalla. unidad, y un módulo de salida de señal. Entre sus principales funcionalidades es que permite transmitir el trazo del usuario con una muy alta fidelidad. Esta característica se la da el sistema de sensado que utiliza, en este caso un sensor de posición electromagnético. Por otra parte, entre sus principales desventajas está que requiere una superficie de apoyo, como una pizarra digital de escritura, para que el transductor electromagnético funcione. Se puede observar en la Figura 1.5 el diagrama de aplicación del dispositivo de lápiz electrónico (20) de acuerdo con esta patente, usado con una plataforma informática, que comprende un host (11), una unidad de detección de entrada (13), que puede ser una pizarra o tablero digital de escritura a mano, y una unidad de visualización (14). Tanto la unidad de detección de entrada (13) y la unidad de visualización (14) están conectadas al ordenador central (11).

Nombre de la Patente: Electronic pen, handwriting output system and control state detection method

Inventor: Masaki Morita, 雅紀 森田

No. de Publicación: JP2004199591A

Procedencia: Japón

Fecha de Publicación: 15 de Julio de 2004.

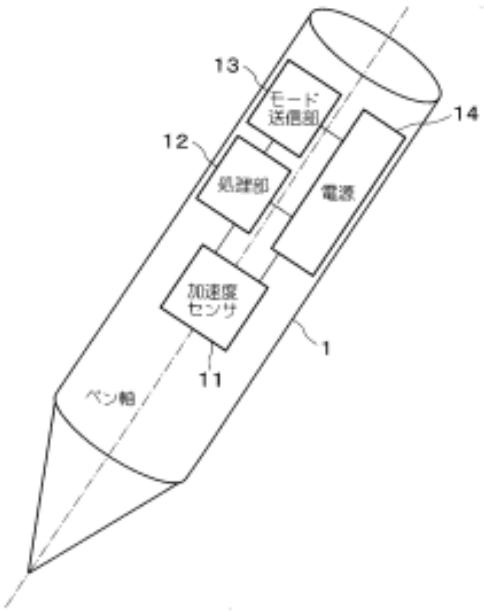


FIGURA 1.6 PATENTE JP2004199591A

Propone un bolígrafo electrónico, un sistema de salida de escritura a mano y un método de detección del estado de control que tiene una función que permite al usuario borrar fácilmente una escritura a mano. En esta patente un sensor de aceleración detecta la aceleración en la dirección del eje de la pluma y una unidad de procesamiento determina que la escritura a mano se borra cuando el valor absoluto de la aceleración es mayor que un valor umbral y cuando el valor absoluto de la aceleración es menor que el valor umbral. Se determina como un estado de entrada de escritura a mano. Entonces, la unidad de transmisión transmite la información del estado de borrado de escritura a mano o la información del estado de entrada de escritura a mano al dispositivo de salida de información de escritura a mano.

El bolígrafo electrónico está conectado al dispositivo de salida de información de escritura a mano a través de un sistema de comunicación inalámbrico para transmitir y recibir información mediante ondas de radio, luz, rayos infrarrojos o similares, o un sistema de comunicación por cable para transmitir y recibir información mediante señales eléctricas. Además, el dispositivo de salida de información de escritura a mano incluye una base para detectar la posición de la punta del lápiz mediante un sensor de presión llamado tableta o similar.

La Figura 1.6 es un diagrama de bloques esquemático que muestra la configuración del bolígrafo electrónico. El número de referencia 1 denota un bolígrafo electrónico, el número de referencia 11 denota un sensor de aceleración para detectar una aceleración unidimensional del lápiz en la dirección del eje del lápiz, el número de referencia 12 denota una unidad de procesamiento, el número de referencia 13 denota una unidad de transmisión para transmitir información sobre el estado de borrado de escritura a mano y el estado de entrada de escritura a mano al exterior según el estado de la pluma electrónica y el número de referencia 14 indica una fuente de alimentación.

Nombre de la Patente: Wireless electronic pen and its handwriting thickness self-adjusting method

Inventor: 彭晓林, 叶耀斌, 张洪哲, 张铭

No. de Publicación: CN101334699B

Procedencia: China

Fecha de Publicación: 13 de Octubre de 2010.

Esta invención proporciona un bolígrafo electrónico inalámbrico, que comprende una carcasa de bolígrafo, un núcleo de bolígrafo, un elemento elástico y un componente de detección de presión, en el que el núcleo de la pluma, el elemento elástico y el dispositivo de detección de presión están dispuestos dentro de la carcasa de la pluma; además, el núcleo de la pluma empuja y se conecta con la carcasa de la pluma mediante el elemento elástico; el bolígrafo electrónico inalámbrico también comprende un dispositivo de autoajuste del grosor de escritura a mano que está conectado con el dispositivo de detección de presión mediante señales inalámbricas; además, el dispositivo de autoajuste del grosor de la escritura a mano está conectado con la pantalla de un ordenador mediante señales. Se basa principalmente en un sistema de medición de presión que entra en contacto con la superficie delimitada para escritura, dicha superficie está

delimitada por el dispositivo sobre el cual se traza. Esta invención utiliza de manera adicional un sistema de sensado óptico que auxilia al sistema de sensado de presión para lograr un mejor desempeño.

Nombre de la Patente: A kind of smart pen and its stroke recognition processing method

Inventor: 李俊峰

No. de Publicación: CN104714666B

Procedencia: China

Fecha de Publicación: 05 de Septiembre de 2017.

Esta invención da a conocer un tipo de bolígrafo inteligente y su método de procesamiento de reconocimiento de trazos. El método que proporciona puede reconocer trazos únicos y trazos múltiples, utilizar un flujo especial mediante el reconocimiento de trazos de la entrada de escritura del usuario en la pantalla táctil para una variedad de trazos geométricos. Figuras tales como línea recta, flecha, rectángulo, rombo, círculo, elipse, flechas de arco circular. El bolígrafo inteligente proporcionado se puede aplicar a una variedad de equipos electrónicos con pantalla táctil, como pizarrones electrónicos. El trazo escrito a mano del usuario se identifica como una variedad de geometrías y se ajusta a formas geométricas estándar.

Nombre de la Patente: Drawing tool for capturing and rendering colors, surface images and movement

Inventor: Kimiko Ryokai, Stefan Marti, Hiroshi Ishii

No. de Publicación: US20060084039A1

Procedencia: Estados Unidos

Fecha de Publicación: 20 de Abril de 2006.

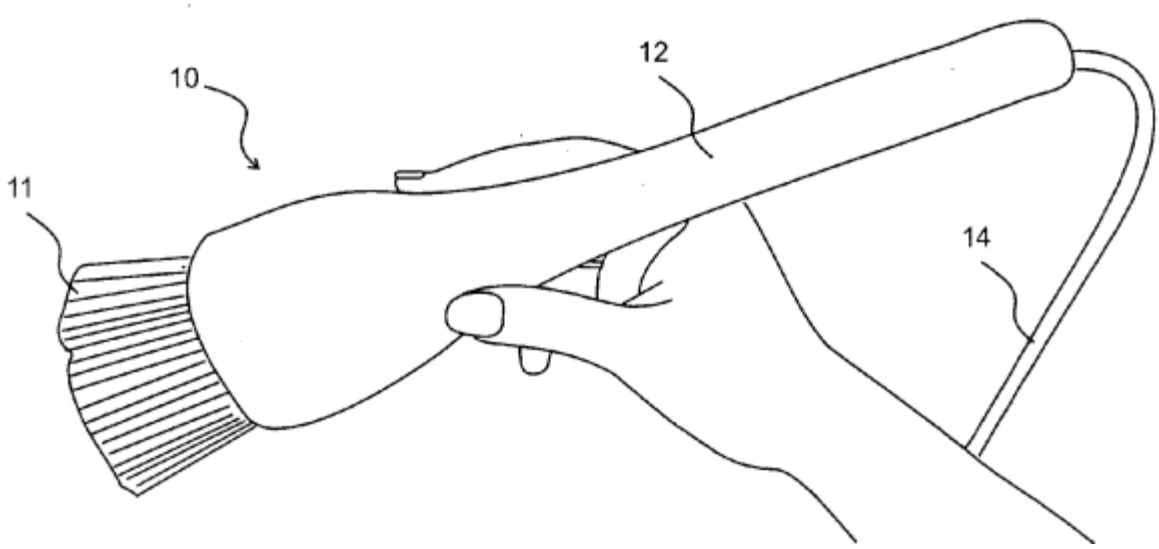


FIGURA 1.7 PATENTE US20060084039A1

Propone una herramienta de dibujo interactiva dirigida a niños pequeños, de cuatro años en adelante, para explorar colores, texturas y movimientos que se encuentran en materiales cotidianos "recogiendo" y dibujando con ellos. La herramienta se ve y se siente como un pincel de mano convencional, pero tiene una pequeña cámara de video con luces y sensores táctiles y de orientación integrados en su interior. Fuera del lienzo de dibujo, el pincel puede recoger el color, la textura y el movimiento de una superficie cepillada, ya sea de objetos o superficies, o de una paleta electrónica que almacena imágenes y colores capturados para un uso repetido y puede implementarse mediante una tableta. En el lienzo, los niños pueden dibujar con la "tinta" especial que acaban de recoger de su entorno inmediato. El lienzo comprende una pantalla de visualización combinada con un sensor de posición del cepillo acoplado a una computadora personal que también recibe imágenes y datos de control del cepillo.

En la Figura 1.7 se puede observar el cepillo o pincel de I/O portátil utilizado en la implementación, la brocha indicada con el número 10 tiene la forma de una brocha de mano grande y tiene cerdas, indicadas con el número 11, en el extremo agrandado de la carcasa del cuerpo, indicada con el número 12. La carcasa del

cuerpo es un cuerpo de madera construido de arce duro girado en un torno que contiene o sostiene todo el cepillo. electrónica, sensores, cámara, cerdas, etc.

1.6. CONCLUSIÓN DEL CAPÍTULO

Este capítulo se realizó con la finalidad de estudiar y explorar algunos de los temas y conceptos que preceden el presente trabajo para tener bases de conocimiento suficientes sobre el área de estudio en el que se incluye el desarrollo del prototipo. Así como, dar a conocer el planteamiento del problema de donde emerge la solución propuesta mediante un conjunto de objetivos a cumplir y su justificación de desarrollo. Además de realizar una búsqueda bibliográfica para mostrar los antecedentes o trabajos similares al dispositivo desarrollado en esta tesis.

Concluyendo así que el trabajo aquí presentado recae en la característica de que la pluma estilográfica inteligente no necesita de ningún tipo de superficie de apoyo, aunque sus tareas también las puede realizar si se tiene superficie de apoyo y los trazos y las órdenes se pueden enviar y recibir sin la necesidad de apoyarse en ningún tipo de superficie específica.

Capítulo 2

Marco Teórico

A partir de la adaptación del micromaquinado para la definición de microestructuras en 1982, se tenía que en aquellos años los primeros acelerómetros eran sistemas de gran complejidad y poco confiables que basaban su funcionamiento en el desplazamiento de una masa inercial sujeta a la aceleración con resortes que contrarrestaban el efecto de la fuerza generada por la masa.

Específicamente, se debe decir que los acelerómetros son sensores inerciales que miden la segunda derivada de la posición, es decir la fuerza de inercia generada cuando una masa se ve afectada por un cambio de velocidad. [2]

2.1. TIPOS DE ACELERÓMETROS

2.1.1. Acelerómetros Mecánicos.

Emplean una masa inerte y resortes elásticos. Los cambios se miden con galgas extensiométricas[1], incluyendo sistemas de amortiguación que evitan la propia oscilación. En este tipo de acelerómetro, una (o más) galgas extensiométricas hacen de puente entre la carcasa del instrumento y la masa inercial (ver Fig. 2.1), la aceleración produce una deformación de la galga que se traduce en una variación en la corriente detectada por un puente de Wheatstone, la deformación es directamente proporcional a la aceleración aplicada al acelerómetro.

[1] Galgas Extensiométricas: Son dispositivos electrónicos que aprovechan el efecto piezorresistivo para medir deformaciones. Ante una variación en la estructura del material de la galga se producirá una variación de su resistencia eléctrica.

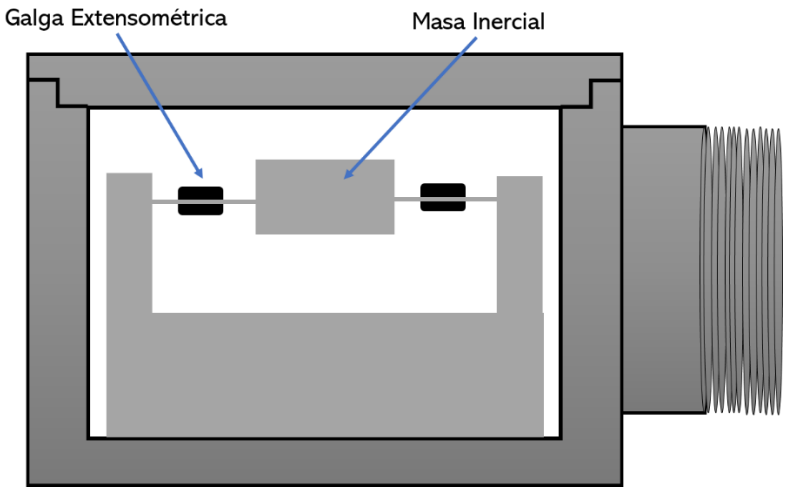


FIGURA 2.1 ESQUEMA DEL ACELERÓMETRO MECÁNICO

Estos acelerómetros también son usados en sistemas rotativos desequilibrados que originan movimientos oscilatorios cuando se encuentran sometidos a aceleración (servo acelerómetros).

2.1.2. Acelerómetros Piezoeléctricos.

Su funcionamiento se basa en el efecto piezoeléctrico. La palabra 'piezo' de origen griego significa "apretar", por lo que se puede deducir su comportamiento: una deformación física del material causa un cambio en la estructura cristalina y así cambian las características eléctricas, (ver Fig. 2.2).

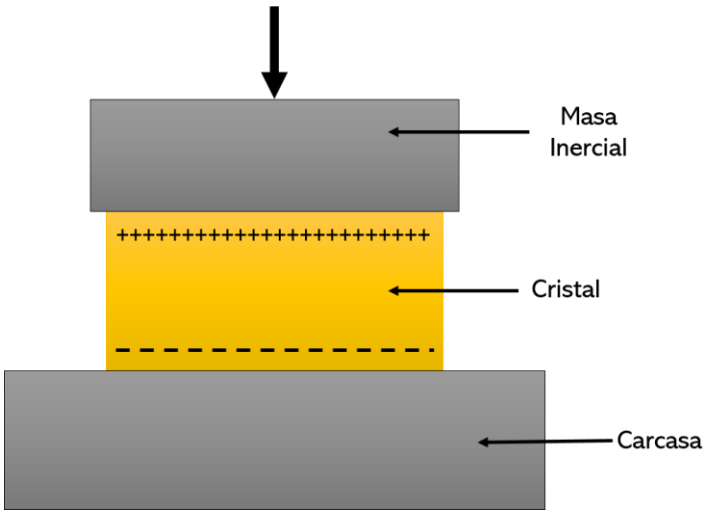


FIGURA 2.2 DIAGRAMA DE UN ACELERÓMETRO PIEZO-ELÉCTRICO

Su principal inconveniente radica en su frecuencia máxima de trabajo y en la incapacidad de mantener un nivel permanente de salida ante una entrada común.

El funcionamiento de este tipo de acelerómetros se basa en las propiedades de los cristales piezo-eléctricos. Estos cristales cuando experimentan alguna fuerza, producen una corriente eléctrica a causa de la variación de su estructura cristalina. Algunos ejemplos de estos materiales serían el cuarzo y la sal.

Así que poniendo un cristal de este tipo entre la carcasa (unida al objeto cuya aceleración se quiere medir) y una masa inercial se producirá una corriente cuando ocurra una aceleración ya que la masa ejercerá una fuerza sobre el cristal. Midiendo esta corriente es posible calcular la aceleración, bien directamente si se trata de un acelerómetro de salida de corriente $(Coulombs/g)$ o bien convirtiéndola a un voltaje de baja impedancia si se trata de un acelerómetro de salida de voltaje.

2.1.3. Acelerómetros Piezorresistivos.

Un acelerómetro piezorresistivo, a diferencia de uno piezo-eléctrico utiliza un sustrato en vez de un cristal piezo-eléctrico. En esta tecnología la fuerza que ejerce la masa sobre el sustrato hace que éste varíe su resistencia, que forma parte de un circuito que mediante un puente de Wheatstone mide la intensidad de la corriente. La ventaja de esta tecnología respecto a la piezo-eléctrica es que puede medir aceleraciones aún con cero *Hz* de frecuencia de operación (ver Fig. 2.3).

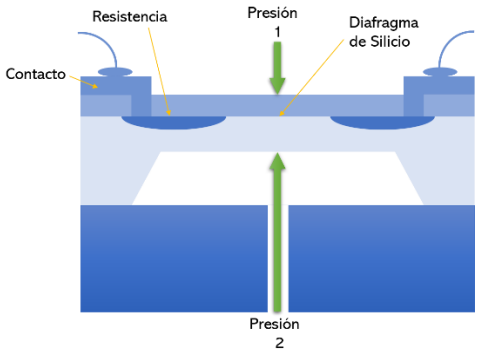


FIGURA 2.3 DIAGRAMA DE UN ACELERÓMETRO PIEZORRESISTIVO

2.1.4. Acelerómetros basados en Tecnología Capacitiva.

En este tipo de dispositivos, la distancia o posición de las placas de un micro condensador de placas paralelas se modifica cuando éste experimenta aceleraciones. Los sensores basados en esta tecnología miden aceleración siempre que se encuentren integrados en un chip de silicio. La integración en chips de silicio reduce diferentes tipos de problemáticas como son la dependencia de humedad, temperatura, capacidades parásitas, número total de terminales, etc.

Cuando se observa el sensor micromecanizado, éste parece una "H". Los delgados y largos brazos de la "H" están fijos al substrato. Los otros elementos están libres para moverse, lo forman una serie de filamentos finos, con una masa central, cada uno actúa como una placa de un condensador variable, de placas paralelas, como se observa en la Fig. 2.4.

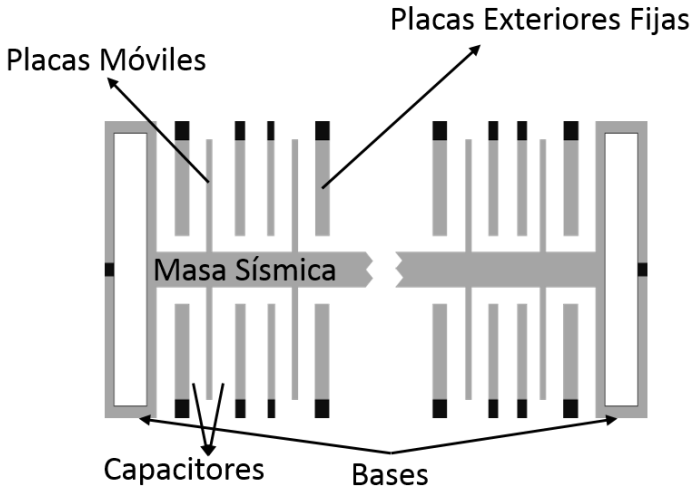


FIGURA 2.4 SISTEMA DE PLACAS PARA SENSOR CAPACITIVO

La aceleración o desaceleración en el eje “sensor”, ejerce una fuerza a la masa central. Al moverse libremente, la masa desplaza las minúsculas placas del condensador, provocando un cambio de capacidad, como se puede observar en la Fig. 2.5. Este cambio de capacidad es detectado y procesado para obtener un voltaje de salida.

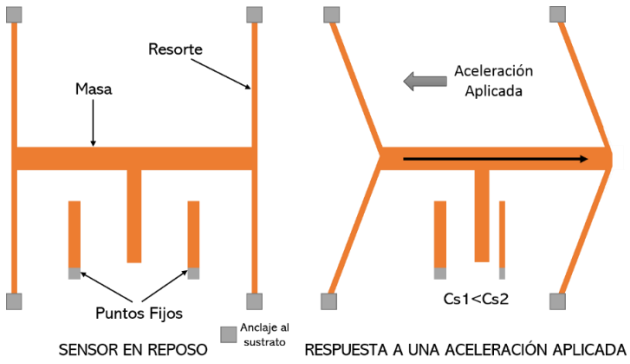


FIGURA 2.5 ESQUEMA DEL ACELERÓMETRO CAPACITIVO

En este tipo de acelerómetros el elemento que conecta la masa inercial con la carcasa es un condensador. Una de las paredes está fija, pegada a la carcasa y la otra a la masa, que tiene movimiento libre. Cuando se presenta una aceleración, la masa se mueve variando en consecuencia la separación entre ambas placas. De este modo, a través de la medición de la capacitancia, se puede hacer la correlación con la aceleración, ver Fig. 2.6. Este tipo de acelerómetros son extremadamente resistentes, pueden soportar aceleraciones de 30000g lo cual permite usarlo, por ejemplo, en mediciones de aceleración de proyectiles de cañón.

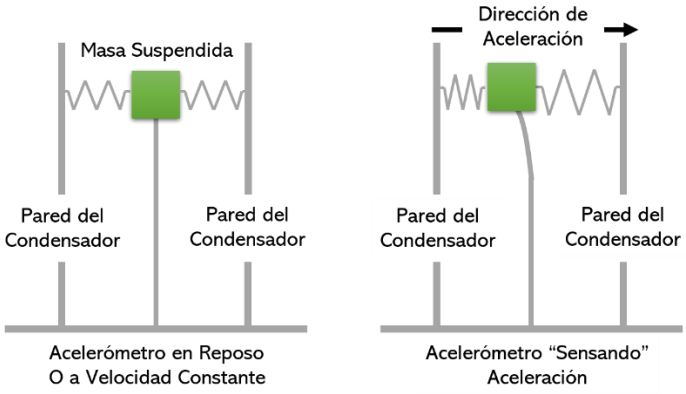


FIGURA 2.6 ESQUEMA DEL PRINCIPIO DE FUNCIONAMIENTO DE UN ACELERÓMETRO DE CONDENSADOR

2.1.5. Acelerómetros Térmicos.

Se trata de un acelerómetro basado en la convección termal. Este tipo de acelerómetro posee un en el cual se hace un hueco para meter una pequeña resistencia que hace de calentador, con dos termopares en los extremos. Con esta

estructura se consigue que se forme una cavidad de aire caliente, que se llama burbuja, sobre los termopares.

La principal característica de estos dispositivos es que tienen sólo un elemento móvil, que consta de la burbuja diminuta de aire caliente, herméticamente sellado dentro de una cavidad existente en el encapsulado del sensor, ver Fig. 11. Cuando una fuerza externa como el movimiento, la inclinación, o la vibración es aplicada, la burbuja de aire caliente se mueve de una forma análoga al mismo. El cambio de estado dentro de la cavidad del integrado produce un voltaje que es función de la diferencia de temperatura y que, tras ser amplificado, condicionado, se proporciona como salida el valor de un voltaje absoluto.

Para el diseño de estos acelerómetros se debe crear una cavidad en la superficie del silicio que conforma el sustrato del sensor. Se coloca una resistencia fabricada con silicio que opera como calentador y que queda suspendida en el centro de la cavidad generada. Se colocan de manera simétrica dos termopares a ambos lados del calentador, ver Fig. 2.7.

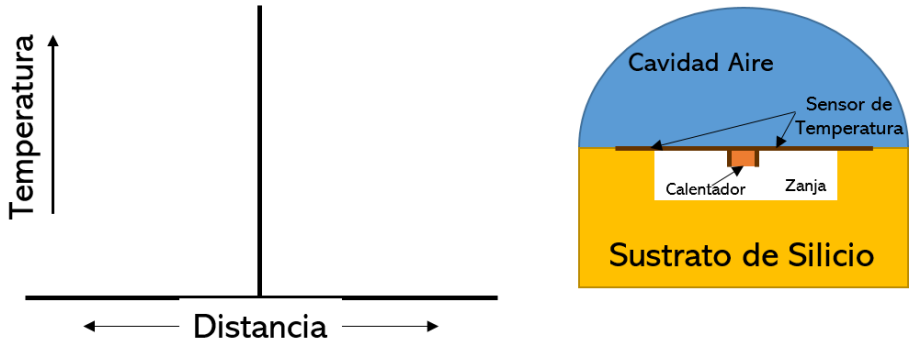


FIGURA 2.7 COMPONENTES DEL SENSOR Y EJES DE REFERENCIA DE LAS VARIABLES ASOCIADAS

Es necesario, además, tener en cuenta al encapsular que se debe dejar una cavidad de aire, o burbuja, sobre la que se producirá la variación de las condiciones de temperatura al producirse movimiento. Este cambio de temperatura entre los termopares creará una señal diferencial que será amplificada y condicionada según las aplicaciones para las que esté diseñado el acelerómetro, obteniéndose como salida de éste.

2.1.6. Acelerómetros basados en Tecnología Microelectromecánica MEMS.

Los avances en tecnología de sistemas microelectromecánicos (MEMS) han permitido la detección del movimiento o los sensores de inercia, conocidos como acelerómetros, para ser puesto en ejecución en muchos usos para las varias industrias.

Los acelerómetros están entre los primeros productos de microsistemas (MST/MEMS) desarrollados. Surgieron en el final de la década de 1980. Sin embargo, para alcanzar un éxito comercial necesitó el desarrollo que surgió durante las décadas de los 70, 80, hasta la del 90 con aplicaciones principalmente en los mercados de la industria automotriz y aeronáutica. Los sensores micrométricos miden el movimiento tal como aceleración, vibración, choque e inclinación. Actualmente, con la tecnología MEMS más madura, gracias a la fabricación en volúmenes muy elevados y a bajo costo, es que los acelerómetros están en gran posición para tener éxito también en aplicaciones como la medicina, la industria y el transporte.

Con relación a la tecnología básica, se distinguen tres categorías principales de acelerómetros de MEMS: el capacitivo de silicio, el piezorresistivo y, finalmente, los acelerómetros térmicos. Hasta el momento, los acelerómetros capacitivos de silicio dominan ampliamente el mercado.

Los sistemas MEMS son básicamente una mejora en los procesos de fabricación, que permite ofrecer ventajas en cuanto a miniaturización, uso eficiente de potencia, alto rendimiento, portabilidad, y fácil interconexión con sistemas múltiples.

Esta tecnología encuentra aplicaciones en una gran diversidad de sectores como de las telecomunicaciones, el automotriz, la informática, electrónica de consumo, y demás dispositivos que utilicen sistemas automatizados como herramienta o producto. La principal característica de los nuevos acelerómetros es la reducción de su factor de forma, logrando además incrementar el rendimiento de los sensores. Disminuir las dimensiones, al mismo tiempo que se incrementa el rendimiento y se ofrece flexibilidad, abren una fascinante ventana de aplicaciones

dentro de las industrias de cómputo y de electrónica de consumo para los sensores basados en MEMS.

2.2. TIPOS DE FILTROS

Dado que inherentemente los acelerómetros producen ruido, el cual aunado a las diferentes frecuencias que la oscilación del sistema masa-resorte tiene, es conveniente el uso de filtros que ayuden a limpiar la señal a procesar. Por lo tanto, en esta sección se ofrece una explicación de los elementos que se pueden usar para el filtraje que debe hacerse para disminuir o eliminar el ruido indeseado. La elección más adecuada de técnicas de filtrado depende de las características de los instrumentos, amplificadores y sistema de adquisición de datos. La integración de los historiales de tiempo del acelerómetro sin un filtrado adecuado producirá una desviación en las velocidades y desplazamientos calculados.

Cuando se trata de Unidades de Medida Inercial (IMU por sus siglas en inglés), la mayoría de éstas tienen 6 DOF (grados de libertad), esto significa que hay 3 acelerómetros monoaxiales y 3 giroscopios monoaxiales dentro de la unidad, pero, aun así, no es posible medir la posición y orientación precisas del objeto al que está conectado, los datos del sensor no son lo suficientemente fiables para usarse de esta manera.

La mejor manera de utilizar una IMU es combinando los datos del acelerómetro y del giroscopio para obtener la posición angular del objeto. El giroscopio puede hacer esto integrando la velocidad angular a lo largo del tiempo, mientras que con el acelerómetro es necesario determinar la posición relativa del vector de gravedad (fuerza g) que siempre está visible en el acelerómetro. Esto se puede hacer fácilmente usando una función arco tangente inversa. Para llevar a cabo esta combinación de una manera ideal es necesario realizar un correcto filtrado de los datos.

Sin embargo, tanto el acelerómetro como el giroscopio presentan inconvenientes o desventajas en su aplicación. Un acelerómetro mide todas las fuerzas que actúan sobre el objeto, por lo que percibe mucho más que sólo el vector de gravedad pues también es susceptible a las aceleraciones producto del movimiento de la IMU o

a fuerzas externas, pero en tiempos largos el ángulo no acumula errores por lo que los datos del acelerómetro son confiables únicamente a largo plazo, por lo que debe usarse un *filtro paso bajo.*

Pero por el otro lado, con el giroscopio, aunque no es susceptible a fuerzas externas, debido a la integración a lo largo del tiempo, la medición tiene la tendencia a desviarse, sin volver a cero cuando el sistema volvió a su posición original por lo que los datos del giroscopio son confiables únicamente a corto plazo, ya que con el tiempo el *drift* (desviación) tiende a aumentar, por lo que debe usarse un *filtro paso alto. [5]*

Uno de los filtros más comúnmente utilizados para llevar a cabo la tarea de combinar el ángulo calculado por el giroscopio (de rotación) y el ángulo calculado por el acelerómetro (de inclinación) es el Filtro Kalman, que, aunque posee muchos atributos, implementarlo resulta muy complejo porque presenta dos grandes problemas que dificultan su uso, estos son:

- Es muy complicado de entender.
- Muy difícil, pero no imposible, de implementar en cierto hardware (microcontroladores de 8 bits).

Otros filtros como el Filtro Complementario, pueden ser la solución a estas dificultades.

2.2.1 Filtro Kalman.

El filtro de Kalman es un algoritmo desarrollado por Rudolf E. Kalman en 1960 que sirve para identificar el estado oculto (no medible) de un sistema dinámico lineal, que sirve además cuando el sistema está sometido a ruido blanco aditivo. [6] En el Filtro Kalman, la ganancia K de realimentación del error se elige de forma óptima cuando se conocen las varianzas de los ruidos que afectan al sistema.

Ya que el filtro de Kalman es un algoritmo recursivo, se tiene que correr en tiempo real usando únicamente las mediciones tanto de entrada actuales, como el estado

calculado previamente y además su matriz de incertidumbre, y no requiere ninguna otra información adicional.

El filtro de Kalman tiene numerosas aplicaciones en tecnología. Una aplicación común es como guía, navegación y control de vehículos, especialmente naves espaciales. Además, se utiliza ampliamente en campos como el procesamiento de señales y la econometría.

El filtro de Kalman es aplicado ampliamente al problema de la mezcla de datos en los sistemas modernos de navegación inercial asistida y se ha convertido en una fuente importante de mejora en la precisión del sistema de navegación. En teoría, el filtro de Kalman hace un uso óptimo de todas las mediciones externas, contabilizando adecuadamente los errores del sistema de medición y de inercia para minimizar los errores del sistema de navegación. Las mecanizaciones prácticas del filtro de Kalman, sin embargo, son invariablemente subóptimas.

Con los requisitos actuales de alta precisión y disponibilidad operativa rápida, el uso óptimo de todas las mediciones externas, contabilizando adecuadamente los errores de medición y del sistema inercial, se ha vuelto obligatorio. Afortunadamente, la base teórica para hacer esto está disponible en las técnicas óptimas de estimación recursiva desarrolladas por Kalman y otros. [7]

Este enfoque, que ha llegado a llamarse filtro de Kalman, es la base para la mezcla de datos en los sistemas modernos de navegación inercial asistida y se ha convertido en una fuente importante de mejora en la precisión de los sistemas de navegación.

La aplicación del filtro de Kalman a los sistemas de navegación inercial comenzó a principios de la década de 1960, poco después de que se desarrollara y publicara la teoría del filtro recursivo óptimo. Dado que los errores en un sistema inercial útil se propagan de manera esencialmente lineal y las combinaciones lineales de estos errores a menudo se pueden detectar a partir de mediciones externas, el filtro de Kalman es ideal para su estimación. También proporciona estimaciones útiles de todas las fuentes de errores del sistema que tienen tiempos de correlación significativos.

Además, el filtro Kalman proporciona un diseño mejorado y flexibilidad operativa. Como filtro variable en el tiempo, puede acomodar fuentes de error no estacionarias cuando se conoce su comportamiento estadístico.

Los cambios de configuración en el sistema inercial se tratan fácilmente mediante simples cambios de programación. El filtro de Kalman proporciona el uso óptimo de cualquier número, combinación y secuencia de medidas lineales externas. Es una técnica para emplear sistemáticamente todas las mediciones externas disponibles, independientemente de sus errores, para mejorar la precisión de los sistemas de navegación inercial.

En pocas palabras, el filtro de Kalman es un algoritmo computacional recursivo que procesa mediciones para estimar el estado de un sistema lineal utilizando el conocimiento del sistema y la dinámica de las mediciones, estadísticas asumidas de ruidos del sistema y errores de medición e información de la condición inicial.

La realización del comportamiento óptimo requiere un conocimiento exacto de la dinámica del sistema y el proceso de medición, y las estadísticas de los ruidos del sistema y los errores de medición. Sin embargo, en la práctica, nunca se dispone de un conocimiento exacto de estas variables. Además, para los sistemas de orden dinámico alto, las ecuaciones de mecanización del filtro de Kalman imponen demandas computacionales sustanciales sobre las capacidades limitadas de las computadoras. Para solucionar este problema, el diseño del filtro a menudo se basa en un modelo dinámico simplificado de orden inferior que conserva las características dominantes del sistema original. Por lo tanto, se considera que se introducen suposiciones incorrectas y a menudo deliberadamente incompletas en el diseño de cualquier filtro Kalman práctico. En estas circunstancias, el filtro resultante ya no es óptimo y se debe realizar un estudio especial para evaluar el rendimiento del error de estimación del filtro subóptimo y la sensibilidad de estos errores a modelos dinámicos y estadísticos incorrectos o incompletos. Este estudio se conoce comúnmente como análisis de sensibilidad.

Para aplicar el filtrado de Kalman a la estimación de errores del sistema de navegación, es necesario derivar una ecuación diferencial de matriz de vector de primer orden estocástica lineal que modele la manera en que el error del sistema interactúa y se propaga en función del tiempo. [8]

2.2.2 Filtro Complementario.

El filtro de complemento o también conocido como filtro complementario, es uno de los más usados por su fácil implementación; éste combina el ángulo calculado por el giroscopio (de rotación) y el ángulo calculado por el acelerómetro (de inclinación).

Debido a las desventajas de trabajar únicamente con el acelerómetro, que es susceptible a las aceleraciones producto del movimiento de la IMU o a fuerzas externas, pero que en tiempos largos el ángulo no acumula errores; o por otro el lado hacerlo únicamente con el giroscopio, que, aunque no es susceptible a fuerzas externas, con el tiempo el *drift* es muy grande y sólo sirve para mediciones de tiempos cortos, es que resulta beneficioso utilizar el filtro complementario, ya que nos brinda "lo mejor de ambos mundos". A corto plazo, utiliza los datos del giroscopio, porque es muy preciso y no es susceptible a fuerzas externas y, además, a largo plazo, usa los datos del acelerómetro, ya que no se desvía. Entonces, la idea es pasar las señales del acelerómetro a través de un filtro de paso bajo y las señales del giroscopio a través de un filtro de paso alto y combinarlas para dar el ángulo final.

El punto clave aquí es que la respuesta de frecuencia de los filtros de paso bajo y paso alto suman 1 en todas las frecuencias. En su forma más simple, la ecuación para calcular el ángulo usando el filtro complemento es (Ver Ec. 2.1):

$$ángulo = A \cdot (ang\_prev + \theta\_gyro\ dt) + B \cdot (\theta\_accel\ ) \qquad \text{ECUACIÓN 2.1}$$

La primera lectura son los datos del giroscopio que se integran en cada paso de tiempo con el valor del ángulo actual. Después de esto, se combina con los datos del filtro paso bajo del acelerómetro (ya procesados con la función arco tangente inversa). A y B son dos constantes que siempre deben de sumar 1 pero, por supuesto, se pueden variar para ajustar el filtro correctamente. [5]

El filtro complementario se comporta como un filtro pasa altas para la medición del giroscopio y un filtro pasa bajas para la señal del acelerómetro. Es decir, la señal del giroscopio manda a corto plazo, y la del acelerómetro a medio y largo plazo, que es exactamente lo que queremos para compensar sus ventajas y defectos.

La función del filtro debe utilizarse en un bucle infinito. En cada iteración, los valores del ángulo de inclinación y rotación se actualizan con los nuevos valores del giroscopio mediante integración a lo largo del tiempo. Luego, el filtro verifica si la magnitud de la fuerza vista por el acelerómetro tiene un valor razonable que podría ser el vector de fuerza *g* real. Si el valor es demasiado pequeño o grande, se sabe con certeza que se trata de una perturbación que no se necesita tener en cuenta. Posteriormente, actualizará los ángulos de inclinación y rotación con los datos del acelerómetro tomando el porcentaje equivalente a la constante A del valor actual y sumando el porcentaje equivalente a la constante B del ángulo calculado por el acelerómetro. Esto asegurará que la medición no se desvíe, pero que será muy precisa a corto plazo.

2.3. MEDICIÓN DE ACELERACIÓN DINÁMICA

La medición de aceleración dinámica con un sensor de este tipo es muy simple, y solo se debe establecer una relación entre la salida que tiene el sensor cuando se le está aplicando una aceleración conocida y establecer una regla de tres (Ec. 2.2).

$$a = \frac{9.81 \times X_{actual}}{X_G}$$ **ECUACIÓN 2.2**

Donde:

X_{Actual} es el valor actual del sensor, que puede estar dado en volts o en digital.

X_G es el valor que entrega el sensor en la posición equivalente a 1 G, éste dependerá de en qué posición se encuentre el sensor.

Para determinar el valor de la aceleración de manera dinámica, solo basta con leer continuamente el valor que entrega el sensor y sustituirlo en la regla de tres. Cabe

aclarar que si se sustituye el valor X_G en volts, el valor de X_{actual} también deberá estar en volts; por el contrario, si se sustituye el valor de X_G en digital, el valor de X_{actual} también deberá estar en digital. Esto se elige según si el acelerómetro es analógico o digital, respectivamente.

La principal dificultad que presenta este tipo de sensado es el ruido presente debido a la vibración del sistema por estimulo del ambiente; por tanto, para explicar este fenómeno se deberá introducir un concepto llamado *sensitividad*[2] cruzada, la cual constituye la respuesta que presenta el sensor a cambios debidos a variables que no necesariamente son la variable física para la que es selectivo. En general se considera que un sensor inercial con baja sensitividad cruzada, es un sensor con mejor desempeño. [4]

2.4. CONCLUSIÓN DEL CAPÍTULO

Este capítulo contempla la revisión teórica de los temas fundamentales implementados en el desarrollo del presente trabajo como, por ejemplo, de los tipos de acelerómetros existentes, ya que como sabemos, este trabajo de tesis basa principalmente el funcionamiento del dispositivo aquí desarrollado en un acelerómetro, aunque es de tipo comercial, se considera de suma importancia conocer la gran gama de existencia de estos dispositivos y su funcionamiento correspondiente. De igual manera se hace referencia a la exploración de los métodos de trabajo que se pueden emplear para la descripción del algoritmo que cumpla con los objetivos establecidos al principio de esta tesis y la elección de uno de ellos que se considera de mayor conveniencia en el desempeño del dispositivo.

[2] Se dice que un sensor es sensitivo al medio, no sensible.

Capítulo 3

Desarrollo

3.1. IDENTIFICACIÓN DE LOS REQUERIMIENTOS DE DISEÑO

Para llevar a cabo el correcto desarrollo del proyecto, se debe ser capaz de verlo como un todo, que estará compuesto por varias partes pequeñas que conllevan a su vez el progreso del proyecto mediante el cumplimiento de los objetivos específicos planteados.

El primer punto que se debe conocer es la función deseada a llevar a cabo por el dispositivo; sabemos que el objetivo que se persigue es ser capaces de realizar trazos virtuales en el aire que se puedan visualizar, por lo que se requiere de un algoritmo que desempeñe esta tarea con ayuda de un circuito electrónico del que al menos basta darse una idea de sus partes. Se conoce que para realizar tareas semejantes a ésta es necesario hacer uso de un acelerómetro, y que se necesita un elemento que sea capaz de procesar esta información y pueda enviarla por comunicación inalámbrica, por lo que se debe tener en cuenta el funcionamiento de estas tres partes en el desarrollo del que llamaremos “Algoritmo de Escritura Libre”, para cuyo inicio se necesita establecer una referencia para enseguida llevar a cabo en primera instancia, la lectura de los datos del acelerómetro, enseguida que procese esta información y, finalmente, la envíe por medio de la comunicación inalámbrica a un dispositivo de visualización. Esto a grandes rasgos es el funcionamiento deseado, buscando en medida de lo posible que la descripción de dicho algoritmo y de la configuración de sus partes se realice con un lenguaje de programación libre de licencias ya que el proyecto está orientado a la elaboración de un producto que puede llegar a ser del tipo comercial.

La segunda actividad a considerar, es configurar nuestro algoritmo de escritura libre para implementarlo, en la medida de lo posible, en un sistema miniaturizado por lo que se propone utilizar montaje superficial para todos los componentes electrónicos requeridos e implementados en cada uno de los módulos utilizados y

descritos anteriormente y hacer las adecuaciones necesarias para que el sistema funcione de la forma esperada. Para esto, se propone construir un circuito electrónico impreso con las menores dimensiones posibles para que éste pueda asemejarse a un lápiz de escritura digital real semejante a los dispositivos comerciales actuales, y así como hacer lo posible para que el manejo por el usuario sea práctico y cómodo.

La tercera parte que conforma este proyecto, es la realización de una interfaz gráfica de usuario (GUI) en la cual se pondrá en marcha el sistema de graficación que originalmente se propone realizar en el lenguaje Python siguiendo los lineamientos de usar software libre de licencias de usuario. Esta interfaz debe permitir ver el funcionamiento del dispositivo en cuanto a conectividad y a la visualización del trazo. Para esta última tarea, se pretende procesar la información del trazo para su almacenamiento en este segundo dispositivo, que por el momento es la estación de trabajo o PC, y que, además, también permita realizar la ejecución de un nuevo trazo a voluntad del usuario.

La última parte de los requerimientos del diseño del proyecto, es la elaboración de la imagen o presentación del producto y con se hace referencia a la elaboración de la cubierta o carcasa del dispositivo y con la cual se presentará como producto terminado, que sea amigable a la vista del usuario y que le ofrezca el mejor confort posible. Este diseño debe procurar primordialmente, la protección del circuito impreso y su correcto manejo por el usuario.

Con esto es posible concluir los requerimientos de diseño del proyecto con el siguiente listado a grandes rasgos:

1. Algoritmo de Escritura Libre
2. Circuito electrónico
3. Interfaz Gráfica de Usuario
4. Diseño del producto

3.2. PRELIMINARES DEL PROYECTO

Como ya se mencionó anteriormente el dispositivo en el que se basa el proyecto está compuesto por tres elementos principales que realizan funciones específicas que son:

1) El módulo MPU6050, que está conformado por un giroscopio de tres ejes y un acelerómetro también de tres ejes y un sensor de temperatura del que no haremos uso para este proyecto,

2) El PIC18F4550, que es un microcontrolador para el procesamiento de los datos, y,

3) El módulo bluetooth HC-05, un dispositivo de comunicación inalámbrico para enviar la información procesada a un segundo dispositivo electrónico que funcionará como interfaz gráfica de usuario.

Inicialmente tanto las configuraciones pertinentes a los módulos implementados, como las primeras pruebas realizadas, e incluso, la descripción original de la rutina para el Algoritmo de Escritura Libre, fueron desarrolladas en el software de Arduino, esto debido principalmente a la facilidad de su uso y a su conocimiento previo. Al realizar el trabajo parte por parte, es una ventaja hacer descripciones individuales y poder modificarlas a conveniencia de la finalidad que se persiga, además de a su vez, poder identificar los errores en cada una de las partes de forma más sencilla.

Para esto se utilizaron a lo largo del desarrollo de la tesis, tres diferentes placas de Arduino, primero se utilizó el Arduino Uno, ya que es uno de los más completos y sencillos de utilizar y no se presentaba ningún inconveniente para poder realizar las configuraciones y pruebas iniciales.

La segunda placa implementada fue el Arduino Nano, que es similar al anterior, buscando reducir las dimensiones del sistema, aun cuando este no sería el sistema final implementado en el trabajo, ya que como sabemos, el objetivo es trasladar la rutina final para el Algoritmo de Escritura Libre a un PIC.

Tanto con el Arduino Uno como con el Arduino Nano, el algoritmo para la Escritura Libre trabaja de forma adecuada, mas no ideal, ya que, al enviar la

información procesada para poder visualizar la gráfica del trazo, se presentó un importante retraso en la graficación de los datos, debido a que utilizamos la misma vía para la obtención de los datos de los módulos implementados y para enviarlos a nuestro segundo dispositivo. Como resultado, se optó por utilizar el Arduino Leonardo, que en características es similar a los anteriores, sin embargo, posee la capacidad de realizar el procesamiento de la información y enviarla de forma separada, logrando así que el bus de comunicación no se sature, solucionando con esto el retraso generado en la graficación de los datos.

3.2.1 Módulo MPU-6050.

FIGURA 3.1 MÓDULO MPU-6050

El módulo MPU6050 es una unidad de medición inercial o IMU (Inertial Measurment Units) de 6 grados de libertad (DoF) pues combina un giroscopio de tres ejes, con el que se puede medir velocidad angular, y un acelerómetro también de 3 ejes, con el que se miden los componentes X, Y y Z de la aceleración. El acelerómetro trabaja sobre el principio piezo-eléctrico, además posee un sensor de temperatura. El módulo se muestra en las Fig. 3.1 y 3.2.

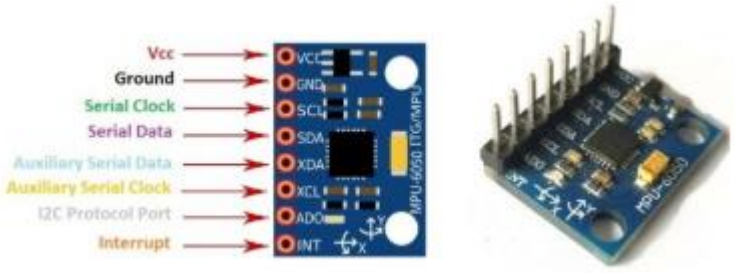


FIGURA 3.2 DESCRIPCIÓN DEL MÓDULO MPU-6050

Los acelerómetros internamente tienen un MEMS que de forma similar a un sistema masa-resorte, permite medir la aceleración. Los giroscopios utilizan un MEMS para medir la velocidad angular usando el efecto Coriolis. El MPU6050 es un sensor de movimiento, que tiene un convertidor analógico-digital (ADC) de 16 bits que convierte los datos a un valor digital; el módulo de giroscopio se comunica con el Arduino a través de la comunicación I^2C a través del reloj serial "SCL" y datos "SDA". El chip MPU6050 necesita 3.3V de polarización, pero un regulador de voltaje en la tarjeta GY-521 le permite alimentarlo hasta 5V, en este caso en un Arduino UNO.

En este dispositivo se tienen dos direcciones I^2C para trabajar, las cuales se muestran en la Tabla 3.1:

TABLA 3.1 DIRECCIONES I^2C

Pin AD0	Dirección I2C
AD0=HIGH (5V)	0x69
AD0=LOW (GND o NC)	0x68

El pin ADDR internamente en el módulo tiene una resistencia a tierra (GND), por lo que, si no se conecta, la dirección por defecto será 0x68.

El procesador interno del IMU es capaz de realizar cálculos precisos de los valores que miden sus sensores internos que son, aceleraciones lineales y angulares, para informarnos de valores útiles como los ángulos de inclinación con respecto a los 3 ejes principales. Un dato importante es que ni la aceleración ni la velocidad lineal afectan la medición de giro.

La dirección de los ejes está indicada en el módulo el cual hay que tener en cuenta para no equivocarse en el signo de las aceleraciones. Como la comunicación del módulo es por medio del protocolo I^2C, esto le permite trabajar con la mayoría de los microcontroladores. En el módulo los pines SCL y SDA tienen una resistencia de tiro (pull-up) en placa para una conexión directa al microcontrolador que se esté utilizando.

3.2.2 Acelerómetro.

El acelerómetro (contenido en el módulo MPU-6050) puede medir la aceleración dinámica, inclinación o vibración y transforma esta magnitud física en otra magnitud eléctrica que será la que se empleará en los equipos estándar de adquisición. Los rangos de medida de este acelerómetro van desde las décimas de g, hasta las decenas de g.

Cuenta con una resolución de 16-bits, lo cual significa que divide el rango dinámico en 65,536 fracciones. Estas características aplican para cada eje X, Y y Z al igual que en la velocidad angular. El sensor es ideal, entre otras cosas, para diseñar control de robótica, medición de vibración, sistemas de medición inercial "IMU", detector de caídas, sensor de distancia y velocidad, y muchas cosas más. El MPU-6050 mediante el protocolo I^2C regresa unos valores conocidos como raw o "crudos" según el registro seleccionado.

A continuación, en la Tabla 3.2, se muestran los rangos de escala y el valor máximo raw, dado en g.

TABLA 3.2 RANGOS DE ESCALA Y SENSITIVIDAD

Rango de Escala completa Giroscopio ***(dps)***	Sensitividad del Giroscopio ***(LSB/dps)***	Rango de Escala Completa Acelerómetro ***(g)***	Sensitividad del Acelerómetro ***(LSB/g)***
±250	131	*±2*	16384
±500	65.5	*±4*	8192
±1000	32.8	*±8*	4096
±2000	16.4	*±16*	2048

Características Técnicas del MPU-6050.

- Salida digital de 6 ejes
- Giroscopio con sensitividad de ±250dps, ±500dps, ±1000dps, y ±2000dps
- Acelerómetro con sensitividad de ±2g, ±4g, ±8g y ±16g
- Algoritmos embebidos para calibración
- Sensor de temperatura digital
- Entrada digital de video FSYNC
- Interrupciones programables
- Voltaje de alimentación: 2.37 a 3.46V
- Voltaje lógico: 1.8V±5% o VDD
- 10000g tolerancia de aceleración máxima

3.2.3 Comunicación Arduino UNO – MPU-6050.

Las bibliotecas con las que se trabajan en algunos de los procesos de implementación para el conocimiento del funcionamiento del MPU6050 y algunas pruebas y ejercicios de experimentación dentro del presente trabajo son:

- La librería desarrollada por Jeff Rowberg para el empleo del MPU6050
- La anterior trabaja con una librería adicional para la comunicación I^2C (I^2Cdev).

TABLA 3.3 CONEXIONES ENTRE ARDUINO Y MPU6050

Mpu6050	Arduino Uno, Nano, Mini	Arduino Mega, DUE	Arduino Leonardo
VCC	5V	5V	5V

GND	GND	GND	GND
SCL	A5	21	3
SDA	A4	20	2

En la Figura 3.3 se muestra el diagrama de interconexión que empata con los datos mostrados en la Tabla 3.3.

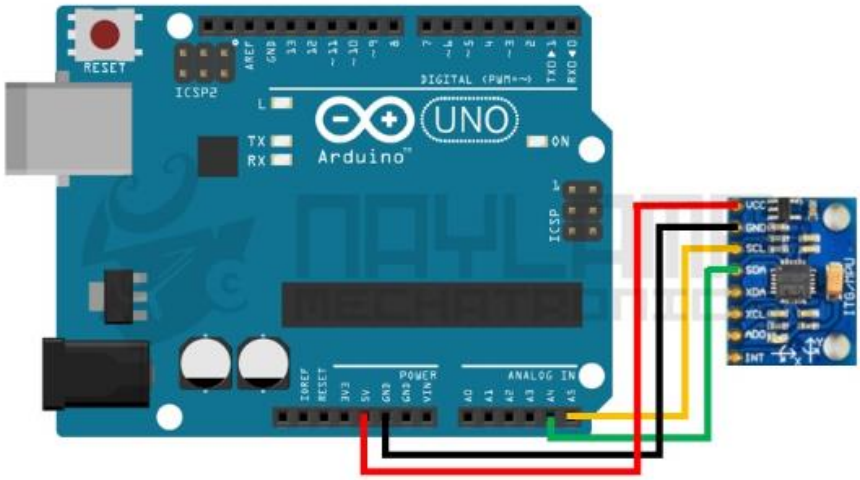


FIGURA 3.3 DIAGRAMA DE CONEXIONES

3.2.4 Mediciones con el MPU-6050.

La primera prueba realizada en el presente trabajo consistió en conocer el funcionamiento del MPU6050 para lo cual se llevó a cabo la implementación de una rutina sencilla (a la que denominamos MPU6050Mediciones) para realizar lecturas tanto del acelerómetro como del giroscopio del módulo. [9]

Para esto se implementaron las librerías descritas anteriormente, utilizando la dirección I^2C 0X68, y la comunicación serial para la visualización de las mediciones.

Al inicializar el sensor, los rangos por defecto serán:

- acelerómetro -2g a +2g
- giroscopio: -250°/sec a +250°/sec

Se deberá tomar en cuenta que la resolución de las lecturas es de 16 bits por lo que el rango de lectura es de -32768 a 32767.

Se realizaron las lecturas y se guardaron en las variables respectivas.

```
// Leer las aceleraciones y velocidades angulares
sensor.getAcceleration(&ax, &ay, &az);
sensor.getRotation(&gx, &gy, &gz);
```

La primera función lee la aceleración de los componentes x-y-z como parámetro, es necesario dar la dirección de las variables como argumento, para lo que se usa: &variable.

La segunda función realiza la lectura de la velocidad angular y guarda las lecturas en sus respectivas variables.

Finalmente se envía por el puerto serial los datos leídos.

Conforme se mueve el sensor los componentes de aceleración irán cambiando en función del ángulo del sensor, puesto que la gravedad siempre es paralela al eje "Z" y se descompondrá en las componentes X, Y, Z del sensor.

3.2.5 Calibración.

Esto es necesario ya que el sensor MPU6050 probablemente de fábrica no se encuentre 100% en una posición horizontal, esto debido a que al ser soldado en el módulo puede estar desnivelado agregando un error en cada componente debido a desalineación con respecto a la base del encapsulado. De igual manera, cuando se instale el módulo en el proyecto, puede estar desnivelado a pesar de que a simple vista lo notemos correctamente nivelado.

Para solucionar este problema, se puede configurar *offsets* en el módulo MPU6050 y de esta forma compensar los errores que se puedan tener.

El programa básicamente está modificando constantemente los offset intentando eliminar el error con la medida real que se desea, en este caso ax=0, ay=0, az=1g y gx=0, gy=0, gz=0.

Inicialmente se leen los *offsets* actuales y se espera que el usuario envíe un caracter por el puerto serie.

Antes de enviar el caracter es necesario ubicar el sensor en posición horizontal y evitar moverlo durante la calibración, dicha posición será el nivel para futuras mediciones.

Después de enviar el carácter, el programa realiza las lecturas tanto del acelerómetro como del giroscopio. Usando un filtro, se estabilizan un poco las lecturas y cada 100 lecturas se verifica si los valores son cercanos a los valores esperados, dependiendo de esto se aumentan o disminuyen los *offsets*. Esto hará que las lecturas filtradas converjan a:

- aceleración: p_ax=0 , p_ay=0 , p_az=+16384
- Velocidad angular: p_gx=0 , p_gy=0 , p_gz=0

Cuando en el monitor serial se observen valores cercanos a los anteriores debemos desconectar o reiniciar nuestro Arduino. Con esto el MPU6050 quedará configurado con el último *offset* calculado en el programa de calibración.

3.2.6 Configuración del Bluetooth.

Para cumplir con una de las especificaciones planteadas como objetivo del presente trabajo, la conectividad inalámbrica, se realiza la conexión entre dispositivos (dispositivo de trazo o pluma y dispositivo de visualización de trazo) mediante conexión bluetooth, para lo cual se hace uso del módulo HC-05 (ver Fig. 3.4). El módulo Bluetooth HC-05 está configurado de fábrica como "esclavo" (slave), pero se puede cambiar para que trabaje como "maestro" (master).

En la Tabla 3.4 se observa la relación de interconexión entre el Módulo Bluetooth HC-05 y el Arduino Uno.

FIGURA 3.4 MÓDULO BLUETOOTH HC-05

TABLA 3.4 CONEXIONES ENTRE EL HC-05 Y ARDUINO

HC-05	Arduino Uno
State	-
RXD	Digital 2
TXD	Digital 3
GND	GND
VCC	5V
EN	-

En este trabajo este módulo quedó asociado al puerto serial denominado "COM9", mismo que cambiará para cada dispositivo que el usuario decida emplear con el receptor, por lo que antes de abrir el Monitor serial, en el IDE Arduino se selecciona dicho "COM".

El módulo HC-05 tiene cuatro estados de trabajo que son:

1) El estado desconectado, entra a este estado tan pronto el módulo es alimentado, y cuando no se ha establecido una conexión bluetooth con ningún otro dispositivo.

2) El estado conectado o de comunicación, entra a este estado cuando se establece una conexión con otro dispositivo bluetooth.

3) El Modo AT 1, en el que para entrar a este estado después de conectar y alimentar el módulo es necesario presionar el botón del HC-05, en este estado, se pueden enviar comandos AT, pero a la misma velocidad con el que está configurado.

4) Y el Modo AT 2, en el que para entrar a este estado el módulo se debe encender con el botón presionado, después de haber encendido se puede soltar y permanecerá en este estado.

Los comandos AT, son la base del conjunto de comandos Hayes, que es un lenguaje desarrollado por la compañía Hayes Communications que prácticamente se convirtió en estándar abierto de comandos para configurar y parametrizar módems. Los caracteres «AT», que preceden a todos los comandos, significan «Atención», e hicieron que se conociera también a este conjunto de comandos como comandos AT. Con ellos, se puede comprobar la disponibilidad del dispositivo, y verificar que todo vaya bien

El siguiente paso es entrar a los comandos de atención, conocidos como comandos AT, en el Modo AT 1 o el Modo AT 2:

- Para entrar al modo AT 1, después de alimentar el módulo y haber encendido, tan solo basta presionar el botón que tiene el módulo HC-05, el LED del módulo seguirá parpadeando rápidamente, por lo que para saber si se ha entrado al Modo AT 1 es necesario enviar comandos AT y esperar la respuesta.

- Para entrar al modo AT 2, antes de alimentar o encender el módulo es necesario presionar su botón, mantenerlo presionado y alimentar el módulo, después que enciende recién, se suelta el botón. Si el LED parpadea lentamente es porque ya está en Modo AT 2. [10]

En este trabajo se envían los comandos AT usando el Modo AT 2, aunque también es posible implementar el Modo AT 1.

En seguida, se abre el monitor serial del IDE de Arduino y se lleva a cabo, mediante comandos AT, una prueba de comunicación y la configuración del módulo. Es posible configurar el nombre del módulo bluetooth, el código de vinculación, la velocidad de comunicación, el rol de trabajo, el modo de conexión e inclusive especificar la dirección del dispositivo al cual se desea conectar; sin embargo, para esta aplicación sólo se cambió del nombre del módulo para su fácil identificación.

3.2.7 Primera Prueba.

Se realizó la descripción de la primera rutina (MPU_Bluetooth) que fue creada con base en el marco de referencia estudiado previamente y en las especificaciones de la hoja de datos del módulo MPU6050 para realizar lecturas exclusivamente del acelerómetro descartando al giroscopio y al sensor de temperatura.

Esta prueba se realizó de forma simple con los componentes y su interconexión entre ellos para poder verificar que la rutina funcione de forma adecuada. Sin embargo, cabe resaltar que ésta junto con las demás pruebas no fueron hechas sobre el prototipo de pluma, ya que únicamente se llevaron a cabo para poder asegurar el cumplimiento de todos los objetivos establecidos. La implementación del sistema sobre el prototipo de pluma fue realizada hasta obtener la rutina correctamente descrita y en lengaje C, empleando SMD en los componentes y un PIC como microcontrolador.

Las características que presenta esta primera prueba son las siguientes:

- Inicialización del módulo MPU6050.
- Conectividad entre el microcontrolador (Arduino Uno) y una estación de trabajo (PC) de manera inalámbrica a través del bluetooth (HC-05).
- Obtención de la señal del acelerómetro a partir de un botón que activa el funcionamiento del sistema y que debe mantenerse presionado mientras se quiera enviar la señal (normalmente abierto).
- Visualización de las lecturas del acelerómetro en los ejes X, Y y Z en la herramienta *TeraTerm*[3].

La Figura 3.5, muestra la pantalla que resulta al leer los datos del acelerómetro con *TeraTerm*, donde las columnas corresponden, de izquierda a derecha con los ejes X, Y y Z.

[3] *TeraTerm:* Es un programa de emulador de terminal (comunicaciones) de código abierto, gratuito e implementado por software que emula diferentes tipos de terminales de computadora.

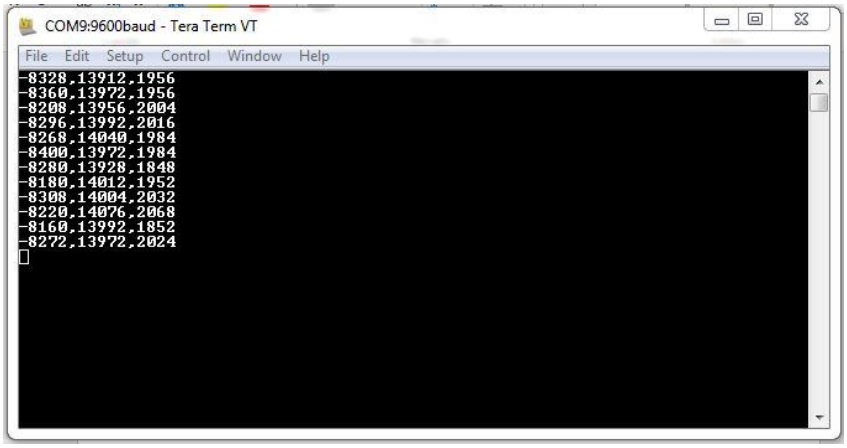


FIGURA 3.5 VISUALIZACIÓN DE LOS DATOS EN LA HERRAMIENTA *TERATERM*

3.2.8 Gráfica de las lecturas de aceleración en Python.

Con el objetivo de ilustrar la dificultad que representa la graficación del trazo que se pretende mostrar, a continuación, se presentan las diferentes estrategias ensayadas en el desarrollo del presente trabajo. Esto ayudó a identificar los inconvenientes presentes en diferentes opciones de rutina, de tal forma que se tuvieran los elementos prácticos que llevaran a elegir el algoritmo más apropiado para lograr los objetivos establecidos.

El primer ejercicio práctico realizado en el lenguaje Python fue obtener los datos y visualizarlos, justo como se hacía en la herramienta *TeraTerm*. La Figura 3.6 muestra los resultados obtenidos.

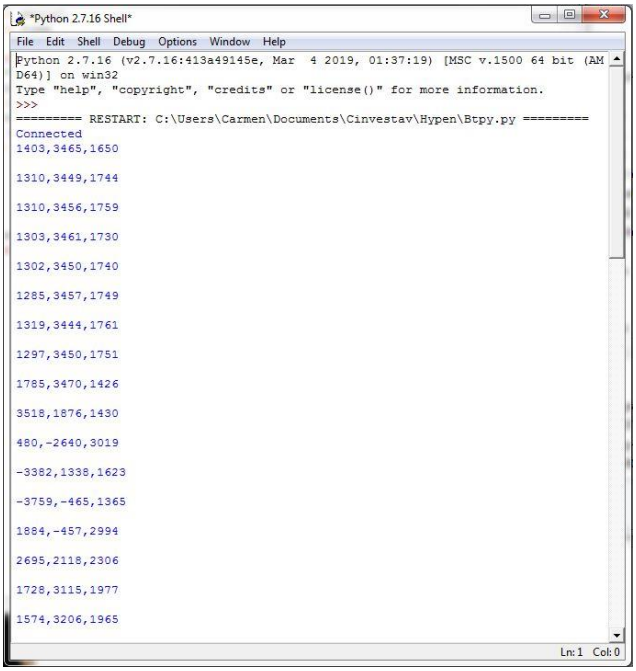


FIGURA 3.6 VISUALIZACIÓN DE LOS DATOS EN PYTHON

A continuación, todas las gráficas aquí descritas se basan en la representación de las lecturas registradas en los ejes X y Y, sin embargo, aunque aún no se realizan las visualizaciones de la aceleración en Z, ésta puede realizarse haciendo cambios de variables en los códigos, ya que la lectura de su valor es conocida y se ha obtenido previamente en las mediciones.

Para obtener una primera gráfica, se realizaron ejercicios de experimentación para conocer el lenguaje de programación, como la creación de una gráfica y de una animación.

- El primer resultado obtenido fue el gráfico del punto de cada lectura del sensor, mismo que al actualizarse generaba una nueva figura, y así por cada lectura realizada, además que la figura tenía una mala representación de los ejes ya que sólo indicaba la ubicación de dicho punto, como se puede ver en la Figura 3.7, en la cual se emplean, como ya se mencionó las lecturas registradas en los ejes X y Y que corresponden a los valores de aceleración en ese momento, en esos ejes.

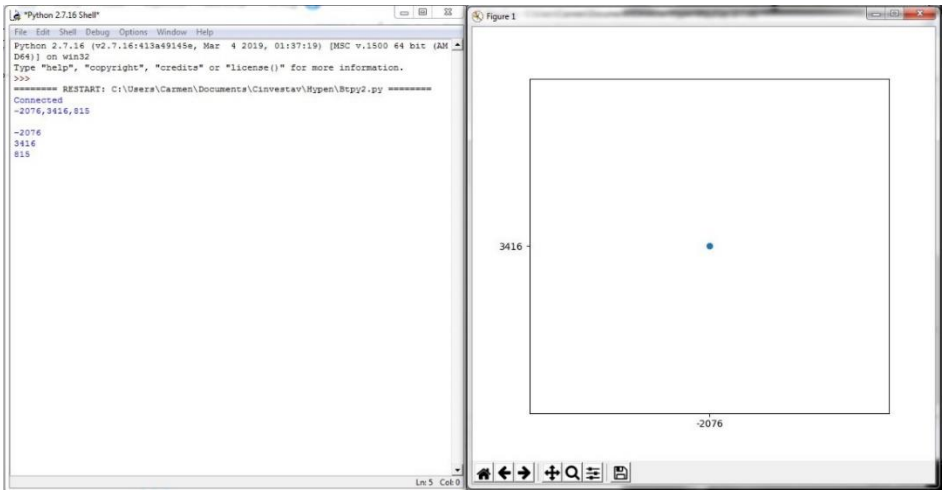


FIGURA 3.7 PRIMER GRÁFICO OBTENIDO (NUEVA FIGURA)

- El segundo resultado obtenido fue similar al anterior, sin embargo, éste ya no se actualizaba en una nueva figura, sino en la misma, pero era imposible ver el punto obtenido anteriormente y así sucesivamente con cada lectura realizada, esto se ve ilustrado en la Figura 3.8.

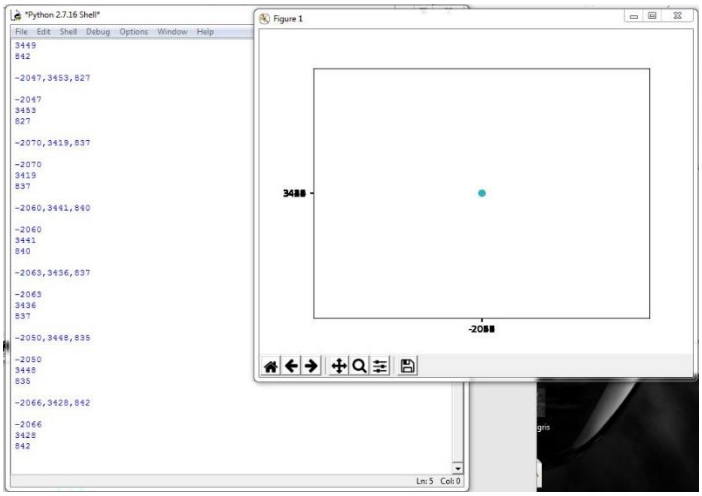


FIGURA 3.8 SEGUNDO GRÁFICO OBTENIDO (ACTUALIZACIÓN EN LA FIGURA)

- El tercer intento tampoco fue exitoso ya que, en la indagación de algún aspecto que llevara a obtener el resultado esperado, se intentó ahora llevar a cabo una descripción que se ajustara con el marco de referencia establecido, así como a los parámetros que ya se han indicado, pero que en esta ocasión almacenaba las lecturas en un archivo de texto .txt y éste a su

vez generaba una gráfica. Sin embargo, el archivo de texto se sobrecargaba y finalmente se encontraba vacío, resultando en la gráfica mostrada en la Figura 3.9.

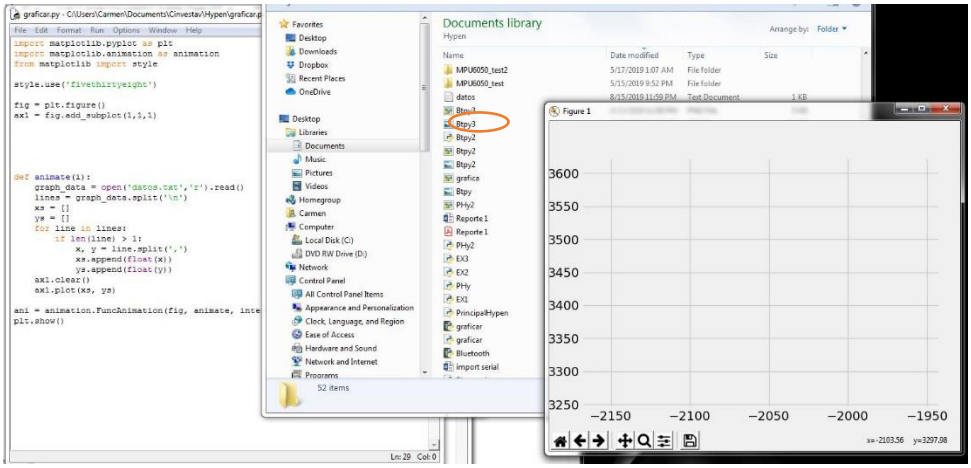


FIGURA 3.9 INTENTO DE GRÁFICA A PARTIR DE UN ARCHIVO DE TEXTO

- La siguiente prueba realizada fue una de las más acertadas ya que el gráfico obtenido era algo muy similar a lo esperado. éste permitía visualizar las aceleraciones y graficar en X y Y, pero, su inconveniente es que es una gráfica en función del tiempo. Por lo cual después de determinadas lecturas, la gráfica era insuficiente por lo que los valores anteriores se eliminaban automáticamente. Esto se puede ver en la Fig. 3.10.

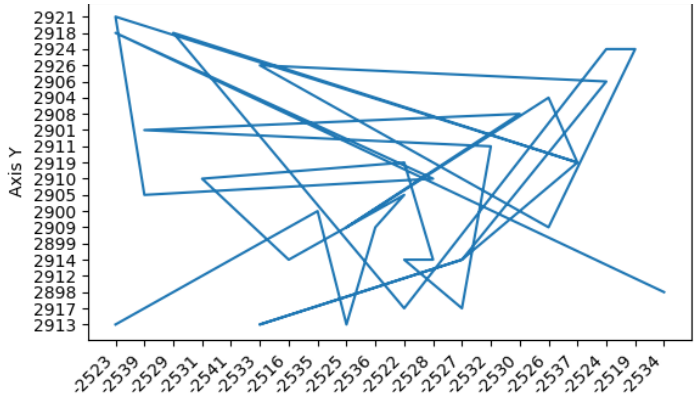


FIGURA 3.10 GRÁFICA EN FUNCIÓN DEL TIEMPO

- Finalmente, la última de las gráficas con la que se concluyó la etapa de exploración para la obtención de la gráfica, es una animación que mantiene fijos los ejes X y Y, que permite que observemos los valores de la aceleración sin perder ningún dato, teniendo así el trazo del acelerómetro en una gráfica en el lenguaje Python, como se observa en la Figura 3.11

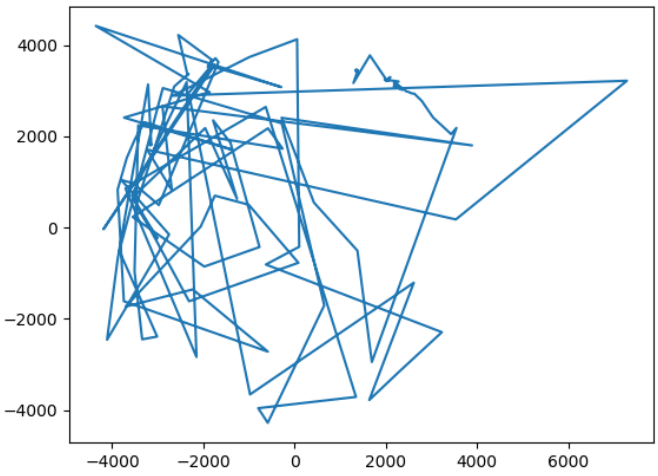


FIGURA 3.11 GRÁFICA FINAL DE ACELERACIÓN

3.3. PLANTEAMIENTO DEL ALGORITMO DE ESCRITURA LIBRE

Continuando con los objetivos establecidos previamente y una vez obtenida la gráfica que permite visualizar la señal del acelerómetro en el lenguaje Python, a continuación, se explica el algoritmo definido para mostrar el libre trazo del sistema de escritura y dibujo virtuales.

Para llevar a cabo esta tarea, en la cual se basará la mayor parte del trabajo, ya que el algoritmo es la parte central en la cual trabajará el sistema, se plantearon tres primeras hipótesis para su desarrollo, las cuales señalan un buen potencial para obtener resultados exitosos que cumplan con el funcionamiento esperado; dichas hipótesis para realizar el trazo son:

1) Partir de la obtención de los ángulos de inclinación del acelerómetro, tomando la lectura anterior como nuevo origen y la siguiente como un segundo punto, trazando un vector entre ambos puntos.

2) Integración de los datos de aceleración.

3) Implementación y uso tanto del acelerómetro como del giroscopio (utilizando también el cálculo de la velocidad angular y el filtro complementario)

Esto sin embargo no descarta que en el camino se formulen nuevas hipótesis o las aquí presentadas se vean modificadas, mezcladas o descartadas.

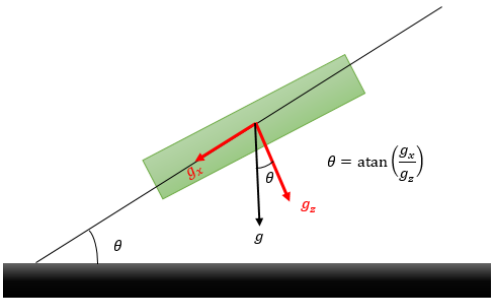
3.3.1 Primera Hipótesis.

Conforme a la implementación de la primera hipótesis, se obtuvo el cálculo de los ángulos de inclinación de la IMU, los cuales se obtuvieron tanto para 2D (empleando la Ec. 3.1), como para 3D (Empleando las Ec. 3.2 y 3.3).

Ángulo en 2D:

$$\theta = \arctan\left(\frac{g_x}{g_z}\right)$$

ECUACIÓN 3.1



Ángulo en 3D:

$$\theta_x = \arctan\left(\frac{a_x}{\sqrt{a_y^2 + a_z^2}}\right)$$

ECUACIÓN 3.2

$$\theta_y = \arctan\left(\frac{a_y}{\sqrt{a_x^2 + a_z^2}}\right)$$

ECUACIÓN 3.3

Se debe señalar, que el cálculo de los ángulos se realizó tanto para 2D como para 3D, porque de primera instancia se planeaba utilizar los ángulos en 2D, pero debido a la perdida de la señal por las asíntotas de la función arcotangente, donde la función no existe, las lecturas entregadas por el sensor son erróneas en dichos

puntos, por lo que la implementación de estos ángulos fue reemplazada por la de los ángulos en 3D.

La Figura 3.12 muestra los datos obtenidos.

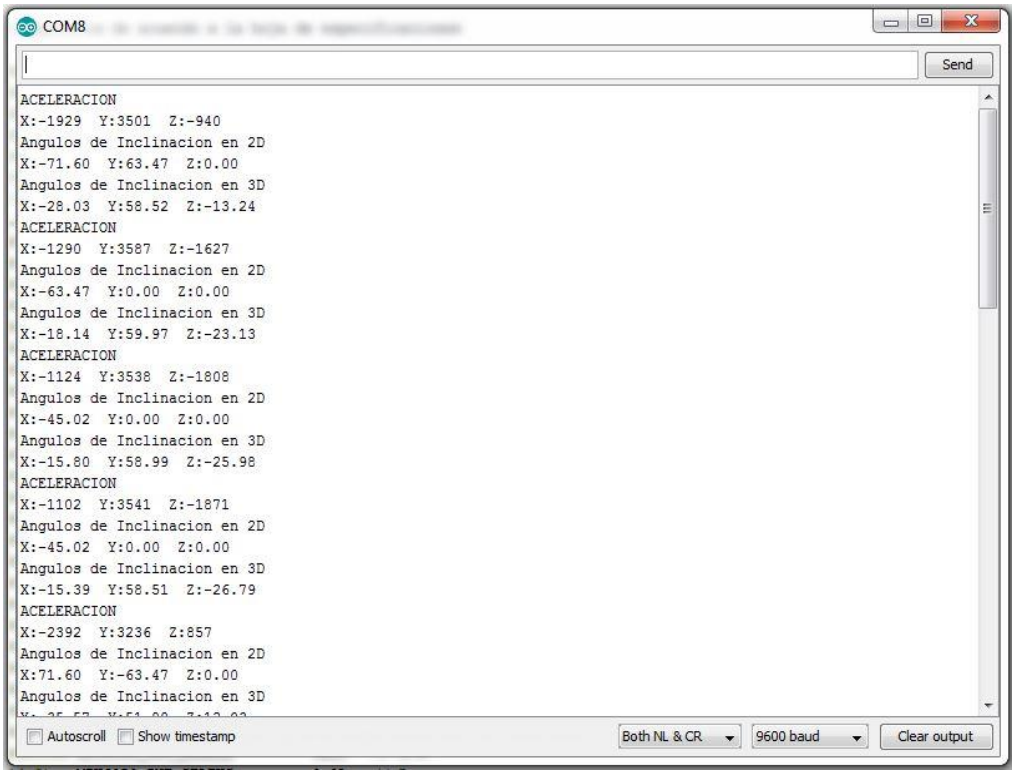


FIGURA 3.12 VISUALIZACIÓN DE LOS VALORES DE LOS ÁNGULOS DE INCLINACIÓN EN 2D Y 3D

La estrategia inicial para esta hipótesis consistió en graficar como puntos los ángulos obtenidos partiendo de la lectura anterior como nuevo origen y trazando un vector entre ambos puntos (ya sea con una longitud unitaria o modificando el módulo con base en medidas experimentales de acuerdo al trazo realizado por el movimiento natural de la mano).

Debido a los inconvenientes presentados (ej. la pérdida de la señal debido a las asíntotas de la función arcotangente, graficación inadecuada utilizando los valores para el ángulo de inclinación en 2D), se optó por implementar nuevas rutas que, si bien corresponden a implementar los ángulos de inclinación de la IMU para la graficación del trazo, modifican esta hipótesis en su forma original.

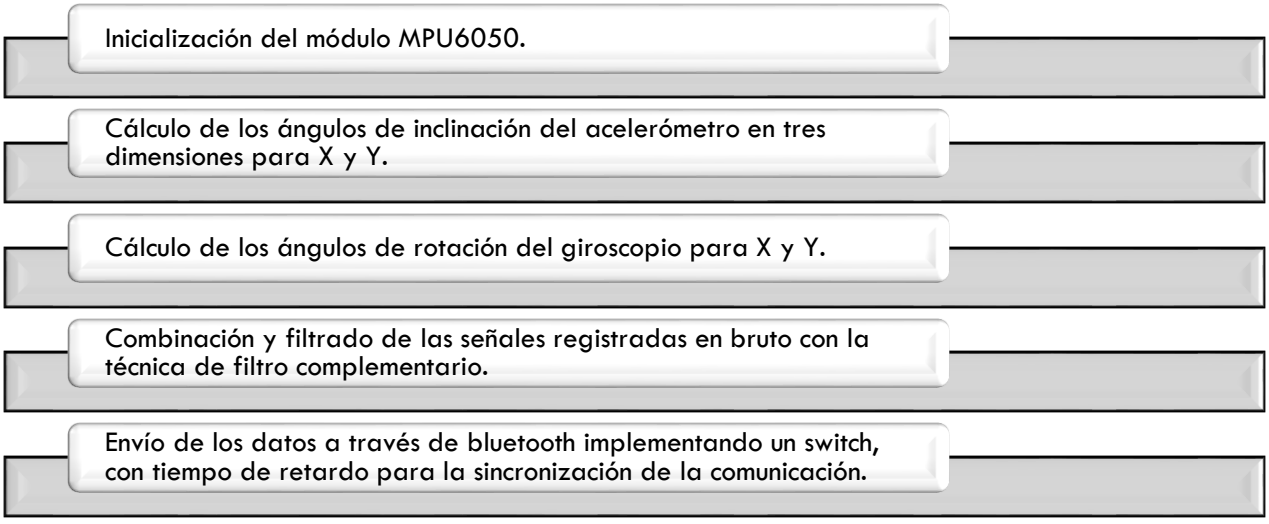
La importancia de la hipótesis planteada radica en que posee un gran valor de innovación y propuesta en cuanto a lo trabajado con anterioridad para realizar la visualización de trayectorias a partir de una IMU.

Por esta razón, se decidió buscar una alternativa que rescate la propuesta de trabajo y que sea capaz de complementar los resultados obtenidos, por lo que, analizando las opciones, se decidió utilizar la tercera hipótesis, que en resumen es una alternativa que permite seguir empleando el ángulo de inclinación.

Se llegó a este resultado sin más que la experimentación.

3.3.2 Algoritmo.

El desarrollo del algoritmo actualmente se basa en la implementación de lo anterior, es decir se puede describir mediante los siguientes puntos:



La implementación de este algoritmo se refiere a trazar la trayectoria realizada por la IMU, a través de cuán inclinada y en qué sentido se encuentre, es decir, que

no corresponde al trazo realizado para la escritura natural, sin embargo, es posible realizar trayectorias que se asemejen a partir de su inclinación.

Tampoco descarta la existencia de otros métodos que sí cumplan con el trazo para la escritura natura, sin embargo, sí los excluye por su falta de valor en propuesta de innovación.

La descripción del algoritmo se realizó originalmente en la interfaz de Arduino con los módulos que se plantearon inicialmente y que fueron descritos en los preliminares, al igual que las primeras pruebas realizadas, pero una vez obtenido un resultado cercano a lo esperado o que nos mostró un funcionamiento adecuado de lo que queríamos ver, se realizó la migración de este programa descrito hacia lenguaje de programación C para utilizar un PIC en lugar de una tarjeta Arduino como microcontrolador, siguiendo los lineamientos de utilizar lenguaje de programación software y elementos de libre acceso. El objetivo que persigue esta actividad es miniaturizar el sistema.

Ambos programas se encuentran en los Anexos de este trabajo de tesis.

3.4. DISEÑO DE LA PLACA DE CIRCUITO IMPRESO

La PCB diseñada para el proyecto está conformada por tres componentes imprescindibles para su funcionamiento, que son, un microcontrolador (PIC18F4550), un módulo de bluetooth HC05, y un módulo MPU6050 (giroscopio, acelerómetro y sensor de temperatura).

Con pruebas experimentales con los Arduino UNO, Nano y Leonardo, se pudo comprobar que el desempeño del algoritmo es bueno, robusto y constante, es decir que se comporta de manera regular sin importar el microcontrolador utilizado (aunque requiere mejoras), sin embargo se presenta una ligera mejoría al separar el envío de los datos de su procesamiento, esto último comprobado en el Arduino Leonardo, por lo que fue posible descartar su uso y reemplazarlo con un microcontrolador con características similares y de dimensiones óptimas para este trabajo, eligiéndose así el PIC18F4550.

A partir de esto, se realizó el diseño esquemático empleando casi en su totalidad dispositivos de montaje superficial por cuestiones de estética, pero principalmente para conseguir el objetivo de llevar a cabo la miniaturización del circuito, por lo que se colocaron cada uno de los dispositivos y sus componentes prescindiendo de la implementación de módulos completos, a excepción del bluetooth para descartar cualquier falla por interferencias y para facilitar su configuración.

Para el diseño del prototipo propuesto, también se utilizaron algunos componentes complementarios, para dar forma al dispositivo esperado, que es de pluma digital, como un led, el interruptor de encendido, y el botón con el que se realizan los trazos virtuales. En la Figura 3.13 se muestra el diagrama esquemático del diseño propuesto, a partir del cual se realiza el diseño del circuito impreso de pluma.

Como se puede observar en el esquemático del circuito, cada uno de los módulos originalmente empleados en el sistema fue reconstruido con todas y cada una de sus partes colocando todos los elementos utilizados en el circuito procurando así que el circuito quedase íntegro y que cada elemento funcione correctamente. Así mismo, es posible observar que se colocaron dos reguladores de voltaje, uno de 3.3V y uno de 5V, como una medida de seguridad y para otorgarle a cada elemento dispuesto en el circuito la alimentación requerida. Por último también se implementaron dos conectores, de 6 pines cada uno, donde el primero estará enlazado con el PIC18F4550, que al ser de tipo SMD sería imposible de reprogramar una vez ensamblada la tarjeta de circuito impreso, por lo que se hace uso de la programación ISP (por sus siglas en inglés, *in-system programming*) o ICSP (por sus siglas en inglés, *in-circuit serial programming*), la cual permite a algunos dispositivos lógicos programables, microcontroladores y otros circuitos electrónicos, que sean programados mientras están instalados en un sistema completo, en lugar de requerir que el chip sea programado antes de ser instalado dentro del sistema. Este conector facilita entonces cargar el programa del algoritmo al microcontrolador con cualquier programador de PICs que disponga de esta función; el segundo de los conectores de 6 pines será utilizado para ensamblar el módulo bluetooth HC-05 una vez que éste esté configurado.

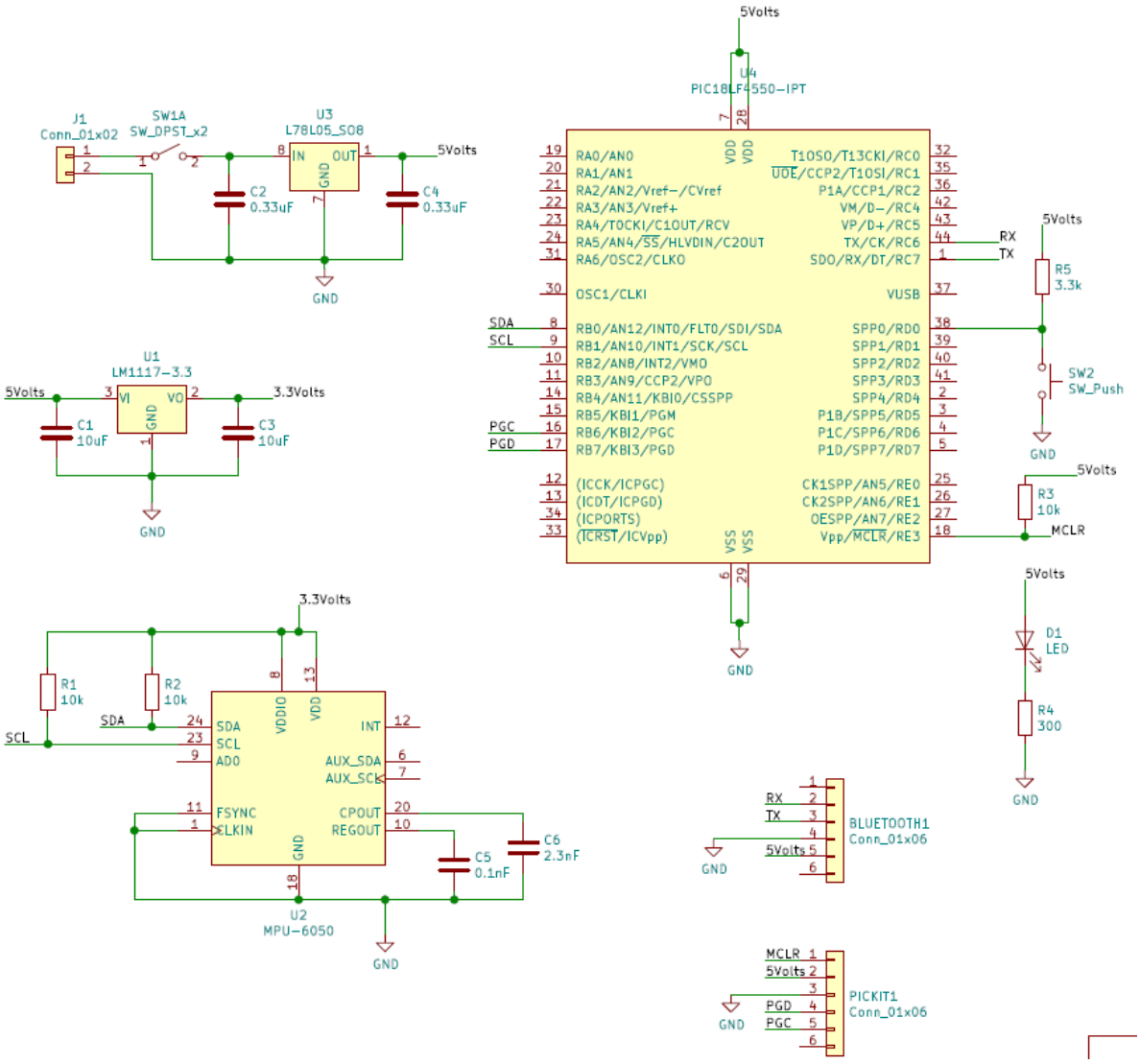


FIGURA 3.13 DISEÑO ESQUEMÁTICO DE LA TARJETA DE CIRCUITO IMPRESO

A continuación, la Tabla 3.5 muestra la lista completa de los elementos que se emplearán en la tarjeta de circuito impreso:

TABLA 3.5 COMPONENTES IMPLEMENTADOS EN LA TARJETA DE CIRCUITO IMPRESO

CANTIDAD	COMPONENTE	TIPO
2	Header o conector de 01x06	Normal

1	Header o conector de 01x02	Normal
2	Capacitor de $10\mu F$	SMD
2	Capacitor de $0.33\mu F$	SMD
1	Capacitor de $0.1nF$	SMD
1	Capacitor de $10nF$	SMD
1	Led blanco ultrabrillante	SMD
2	Resistencia de $10K\Omega$	SMD
1	Resistencia de 300Ω	SMD
1	Resistencia de $3.3K\Omega$	SMD
1	Interruptor deslizable	Normal
1	Botón	Normal
1	Regulador de voltaje de 5V	SMD
1	Regulador de voltaje de 3.3V	SMD
1	Sensor MPU-6050	SMD
1	Microcontrolador PIC18F4550	SMD
1	Módulo Bluetooth HC-05	Normal

Una vez realizada la disposición de todos los componentes indicados arriba, es posible realizar el diseño del circuito impreso (PCB), el cual, para facilitar su elaboración, dispone de dos caras únicamente, una anterior y una posterior.

Este diseño tiene 20mm x 90 mm como dimensiones, esto debido a la disposición de cada uno de los elementos y las pistas de conexión entre ellos, así como los espacios de margen que se pretenden para asegurar que no exista ningún riesgo de cruce entre las conexiones establecidas y puede verse de la siguiente manera en la Figura 3.14:

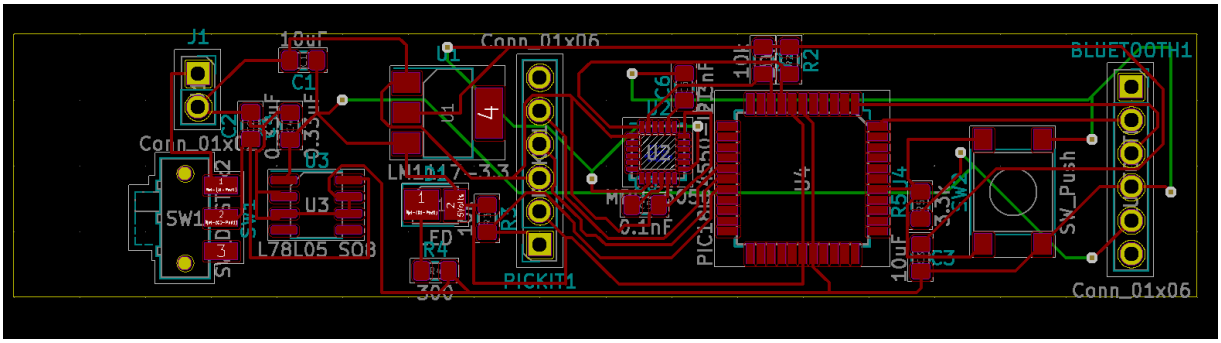


FIGURA 3.14 DISEÑO DE LA TARJETA DE CIRCUITO IMPRESO

La elaboración de las placas del circuito impreso se solicitó con un espesor de 1.6mm dando como resultado la PCB mostrada en la Figura 3.15:

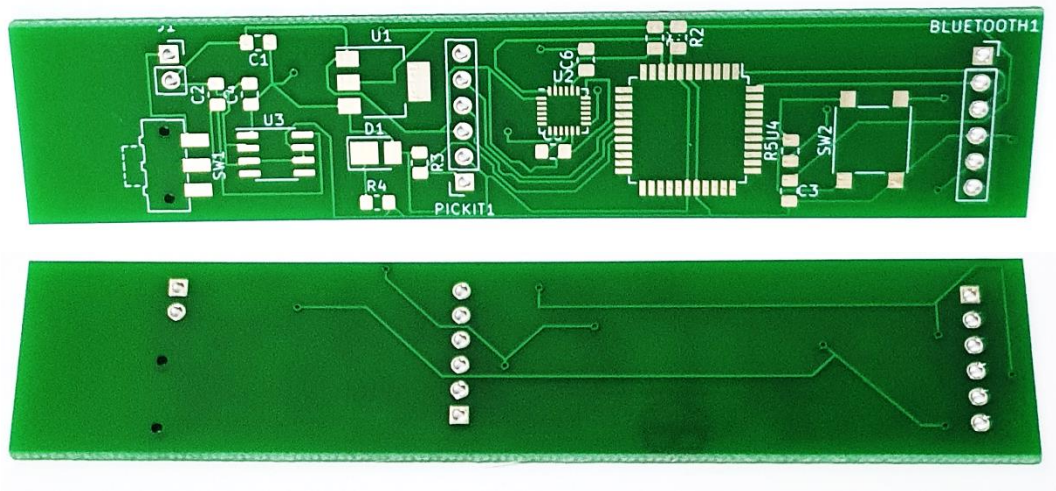


FIGURA 3.15 IMPRESIÓN DE LA TARJETA DE CIRCUITO

Una vez ensamblados los componentes electrónicos, la PCB puede verse de la siguiente manera en la Figura 3.16:

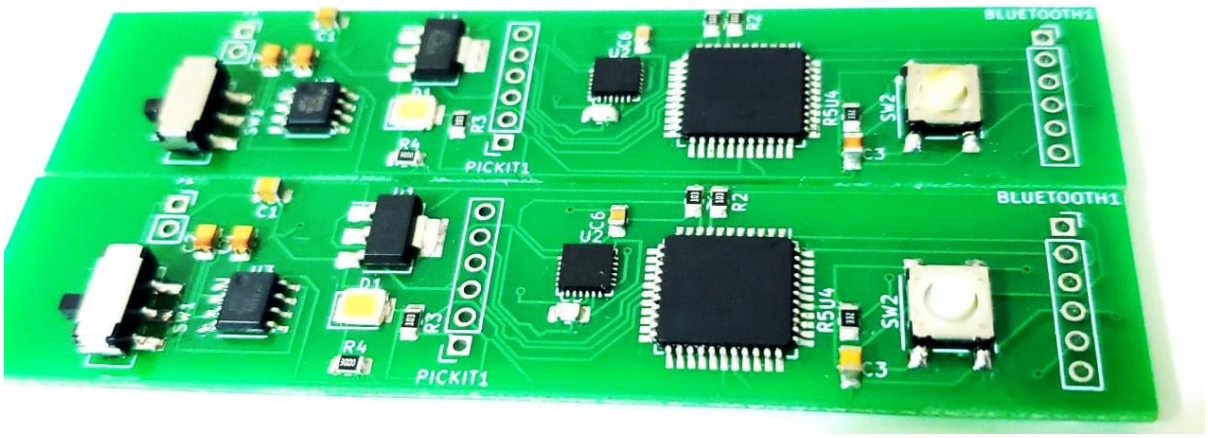


FIGURA 3.16 TARJETA DE CIRCUITO IMPRESO ENSAMBLADA

3.5. DISEÑO DE LA INTERFAZ GRÁFICA DE USUARIO

El desarrollo de la gráfica para la visualización del trazo llevada a cabo en Python consiste, únicamente, de la recepción de los datos enviados y de la creación de un gráfico animado que sitúa los valores en X y Y mediante puntos dentro del plano y su procesamiento y adecuación en cada uno de los ejes, es decir, utilizando signos para modificar el sentido del trazo (de izquierda a derecha), así como, su autoescalamiento debido al posible crecimiento del área del gráfico.

El programa descrito para la graficación de los datos del trazo realizado se puede observar en los anexos de este trabajo de Tesis.

A continuación, se muestran en las Figuras 3.17, 3.18, 3.19, 3.20 y 3.21 los trazos correspondientes a los gráficos resultantes obtenidos con el algoritmo graficando los ángulos en el eje X y en el eje Y resultantes de la aplicación del filtro complementario. En estas figuras se observa que existe un desempeño viable del mismo, mas no el óptimo. Sin embargo, con ellos se puede comprobar que ya es posible realizar trazos más armoniosos, incorporar curvaturas en el mismo y que probablemente con las mejoras adecuadas, permita realizar escritos o figuras que, aunque no sean realizados bajo el trazo natural del usuario puedan ser dibujados con movimientos de inclinación del dispositivo.

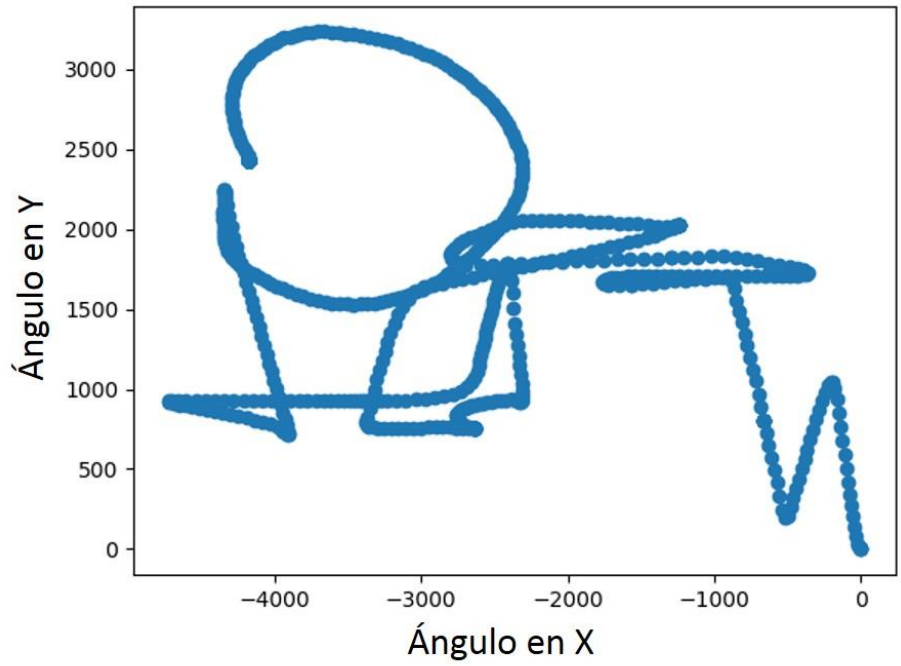


FIGURA 3.17 EJEMPLO 1 DE LOS TRAZOS OBTENIDOS

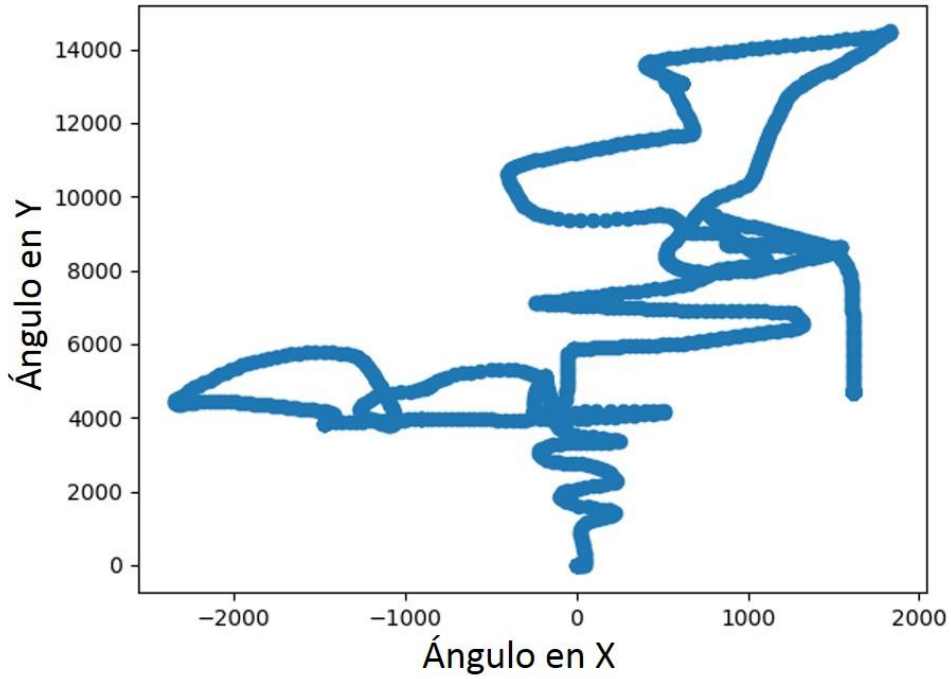


FIGURA 3.18 EJEMPLO 2 DE LOS TRAZOS OBTENIDOS

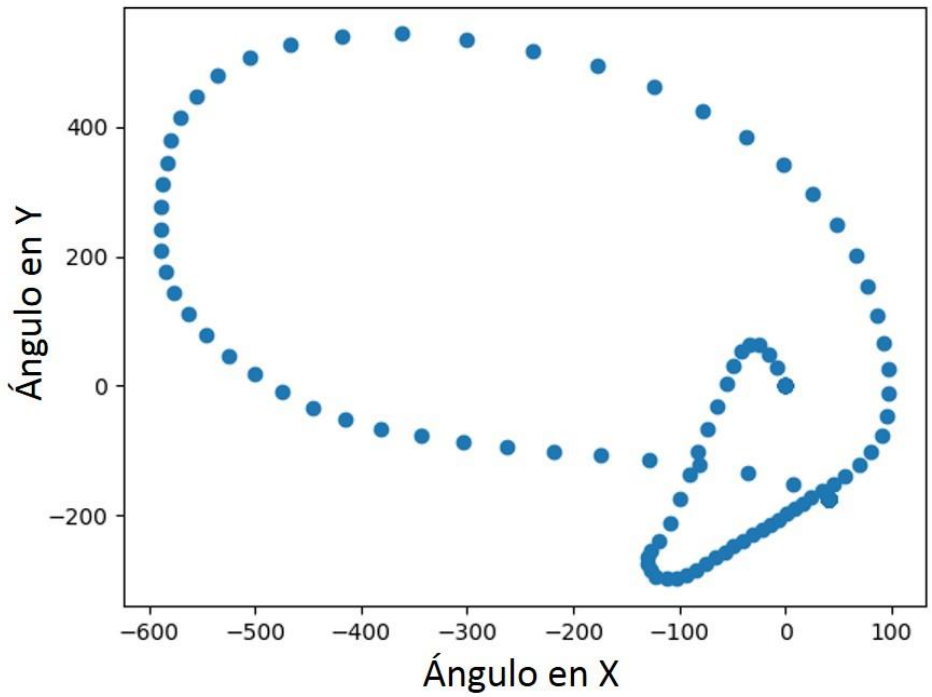


FIGURA 3.19 EJEMPLO 3 DE LOS TRAZOS OBTENIDOS

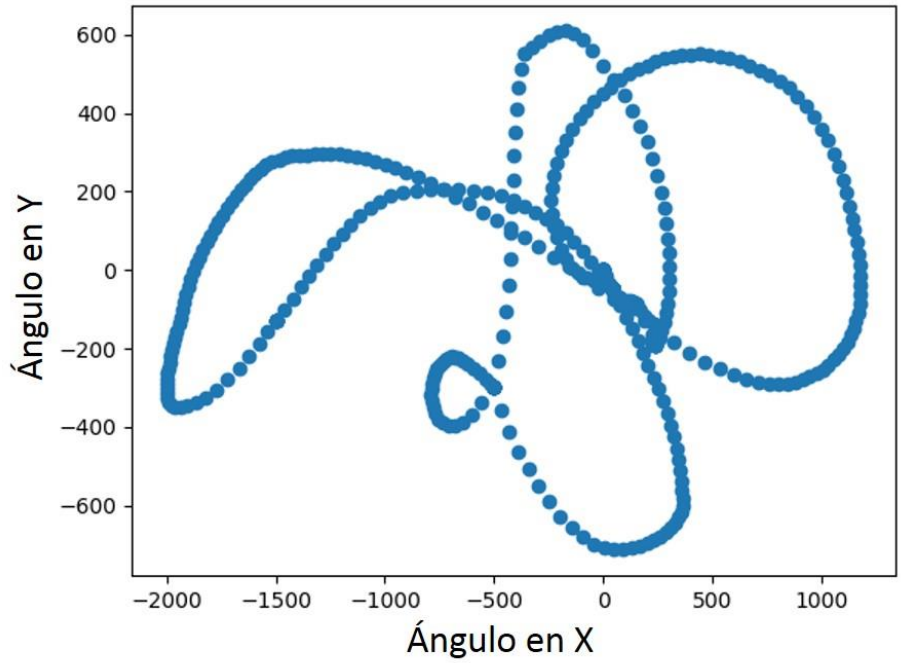


FIGURA 3.20 EJEMPLO 4 DE LOS TRAZOS OBTENIDOS

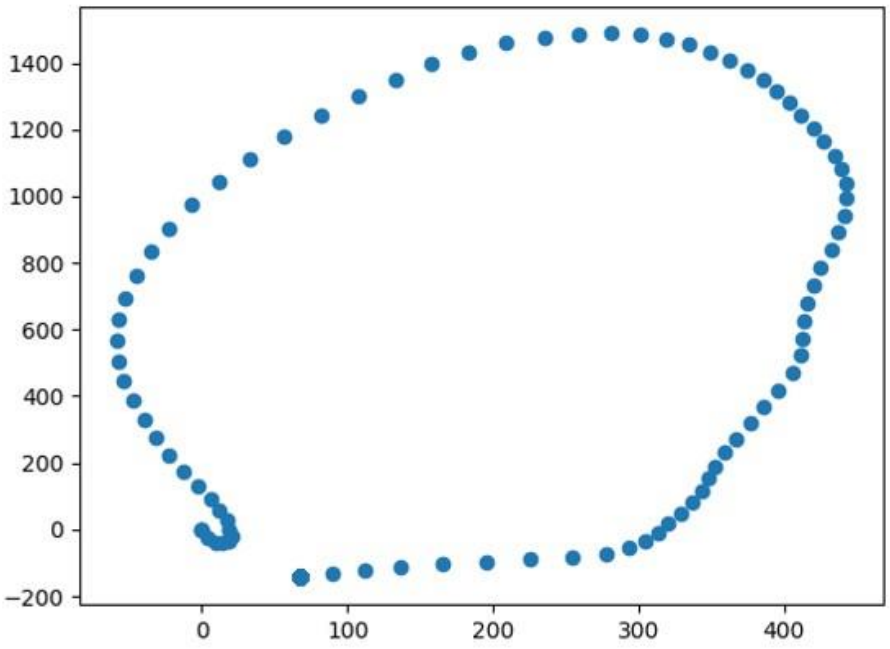


FIGURA 3.21 EJEMPLO 5 DE LOS TRAZOS OBTENIDOS

Una vez obtenida la gráfica del trazo realizado, se desarrolló una interfaz gráfica (GUI) amigable con el usuario, permitiéndole así la correcta visualización del trazo y su almacenamiento y a su vez, permita reiniciar el funcionamiento del dispositivo para crear un nuevo trazo.

La creación de esta GUI, se llevó a cabo con el Diseñador de QT, para permitir así la conexión entre el programa descrito en Python para la graficación y la interfaz creada, además de brindar más herramientas de diseño que vuelvan más agradable a la vista esta interfaz de la aplicación que se le otorgará al usuario. Esta se muestra en las Figuras 3.22 y 3.23.

Esta interfaz cuenta con una primera página para la introducción del producto con su nombre característico, que es, *Hypen*, imágenes demostrativas del dispositivo y un botón de inicio, como se puede observar en la Figura 3.22.

FIGURA 3.22 PÁGINA DE PRESENTACIÓN DE LA INTERFAZ GRÁFICA DE USUARIO

Al oprimir el botón "START", la animación con la cual se grafica se inicializa y es posible observar el trazo que se está realizando mediante el movimiento del dispositivo y la inclinación de éste, como se puede observar en la Fig. 3.23.

Cabe destacar que la interfaz de usuario está directamente ligada mediante Python a la animación descrita bajo este mismo lenguaje.

Esta nueva ventana que contiene la animación también posee la característica de almacenaje una vez que ya no se desee continuar o bien que el usuario haya concluido el trazo y desee hacer uso de él. Esta capacidad de almacenamiento brinda al usuario obtener una versión de su gráfico en diversos formatos, como son: PNG, PDF, etc.

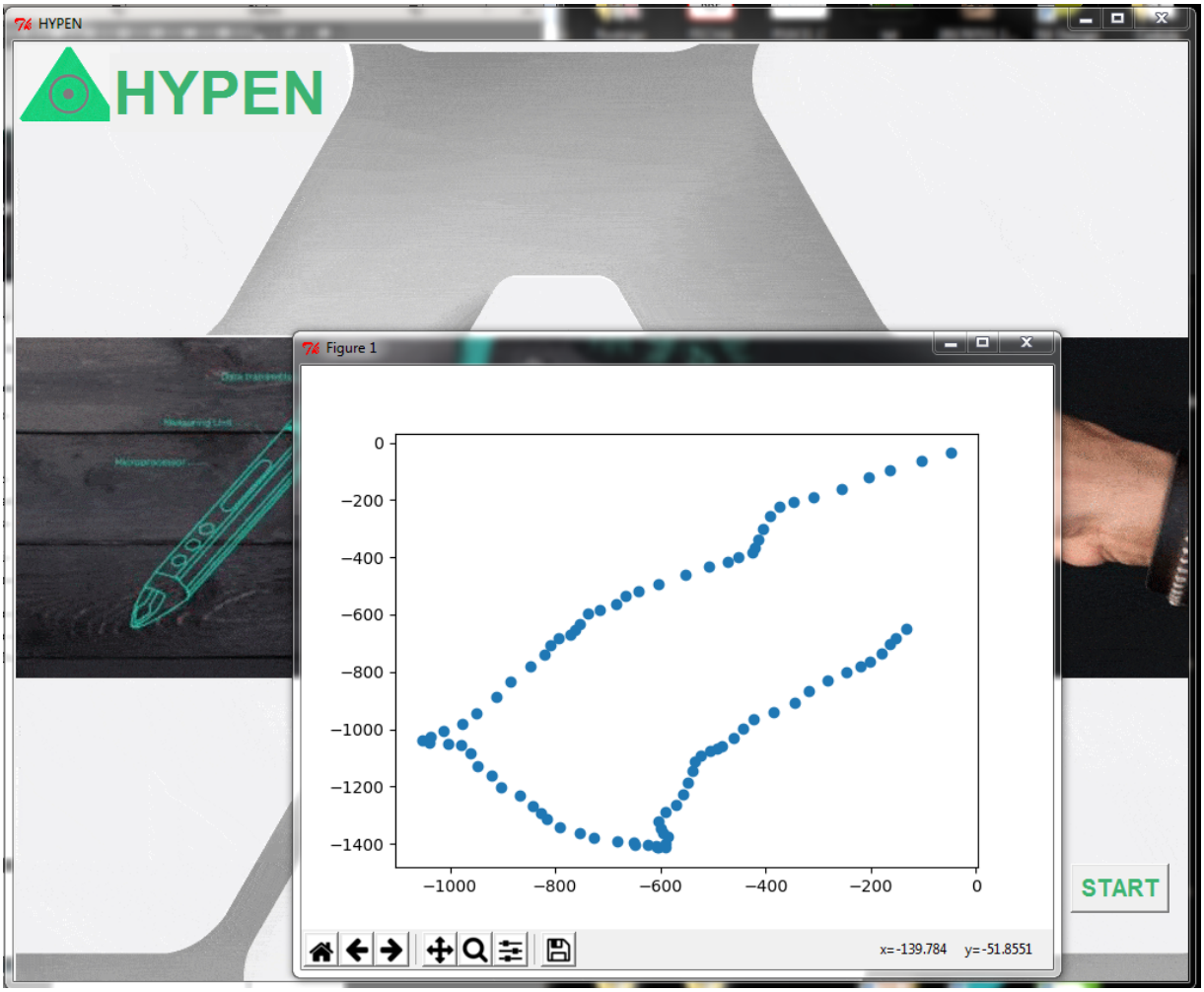


FIGURA 3.23 VENTANA DE VISUALIZACIÓN DE GRÁFICOS DE LA INTERFAZ GRÁFICA DE USUARIO

3.6. DISEÑO DE LA IMAGEN DEL PRODUCTO

Finalmente, la última parte faltante en la composición del proyecto es la elaboración del diseño de la imagen del producto, la cual corre a cuenta del diseño de la cubierta o carcasa del dispositivo. Para el diseño de este elemento se utilizó el software SolidWorks para crear las piezas del objeto con imagen lo más amigable posible para el usuario y que tenga semejanza relevante con el objetivo final del proyecto, por lo que se sugirió que este diseño fuera lo más cercano a un lápiz convencional, el cual consiste básicamente de dos partes: cuerpo y tapa, según se muestra en las Figuras 3.24, 3.25 y .326. Esta cubierta se realizó mediante impresión 3D.

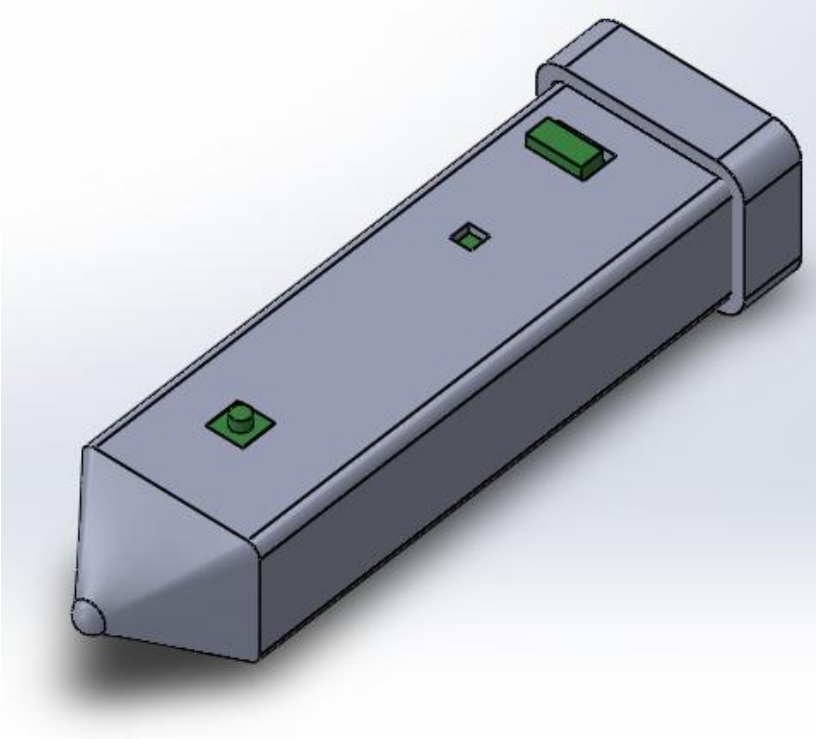


FIGURA 3.24 PROYECCIÓN ISOMÉTRICA DE LA CUBIERTA DEL DISPOSITIVO

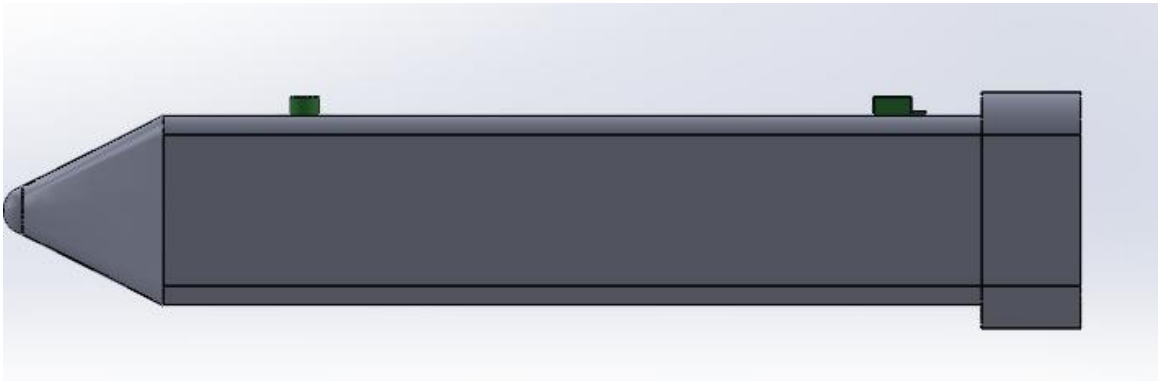


FIGURA 3.25 VISTA LATERAL DE LA CUBIERTA DEL DISPOSITIVO

El cuerpo, de forma general, es una caja rectangular con espesor de paredes de 3mm, que posee una terminación piramidal para dar el aspecto de la punta de lápiz en el extremo que se encuentra cerrado.

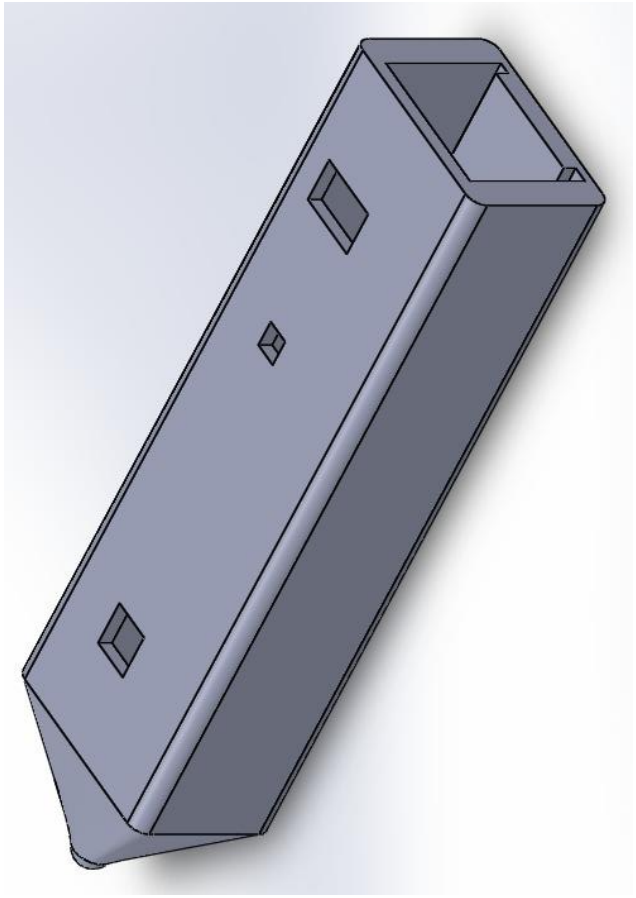


FIGURA 3.26 PROYECCIÓN ISOMÉTRICA DEL CUERPO

Cabe mencionar que, como parte del diseño, conforme a la disposición de los elementos en la tarjeta de circuito impreso, en la cara anterior o superior de la caja se hicieron 3 orificios que facilitan al usuario la operación del dispositivo. El primero que se encuentra en la parte superior cercana a la tapa y sirve para alojar el interruptor de encendido del dispositivo; el siguiente sirve para visualizar el led de operación del dispositivo con lo cual es posible corroborar que se encuentre encendido o apagado; y por último, el inferior más cercano a la punta del lápiz, contiene al botón para realizar los trazos, dando así confort durante la experiencia de uso al usuario ya que de esta forma se obtiene un agarre natural del dispositivo, como se hace con un lápiz o bolígrafo común. Estos detalles se pueden apreciar en la Figuras 3.27 y 3.28.

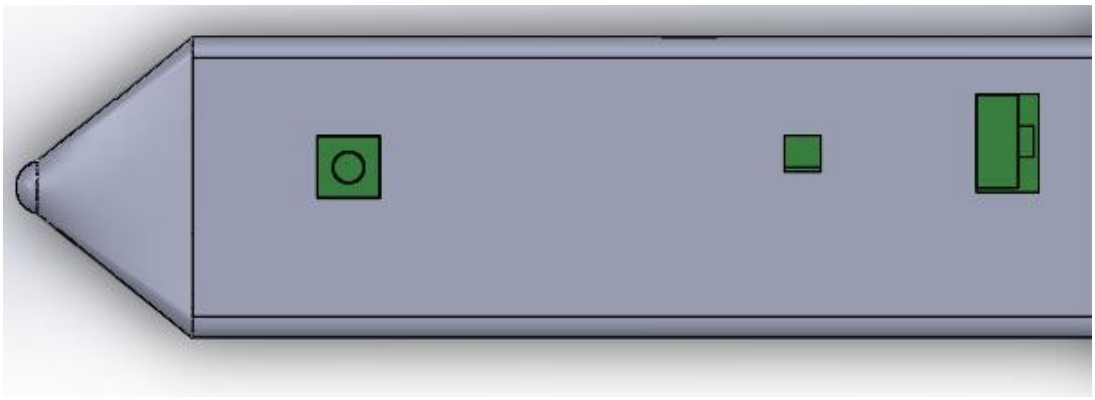


FIGURA 3.27 RANURAS IMPLEMENTADAS EN EL CUERPO DE LA CUBIERTA

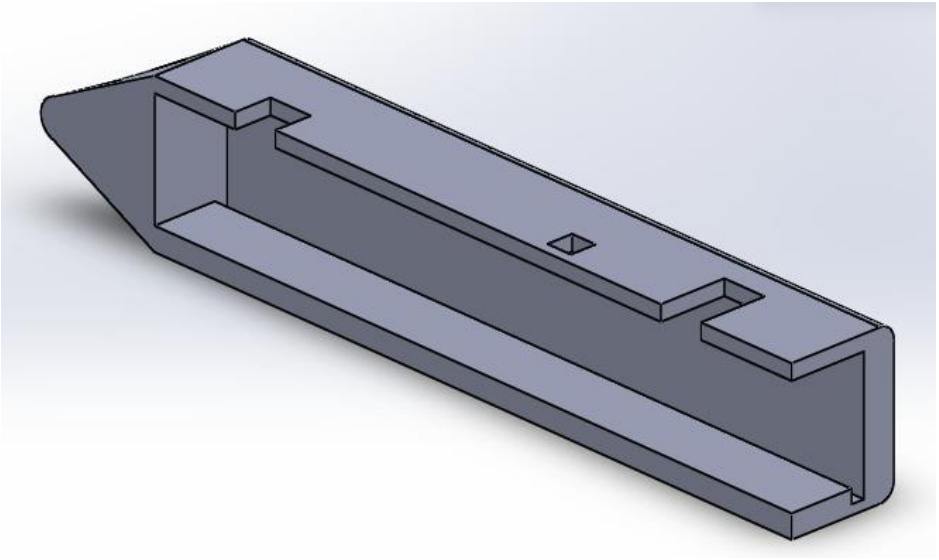


FIGURA 3.28 CORTE LONGITUDINAL DEL CUERPO DE LA CUBIERTA

El cuerpo de la cubierta del dispositivo cuenta adicionalmente con dos ranuras internas dispuestas una en cada lado del rectángulo, mostradas en las Figuras 3.29 y 3.30, cuyo propósito es para introducir dos barras de soporte, una en cada ranura respectivamente, y sirven para colocar y fijar la placa de circuito impreso, es decir, funcionan como una base de apoyo para que la tarjeta quede sujeta a la parte superior y no se mueva y el dispositivo pueda ser manipulable.

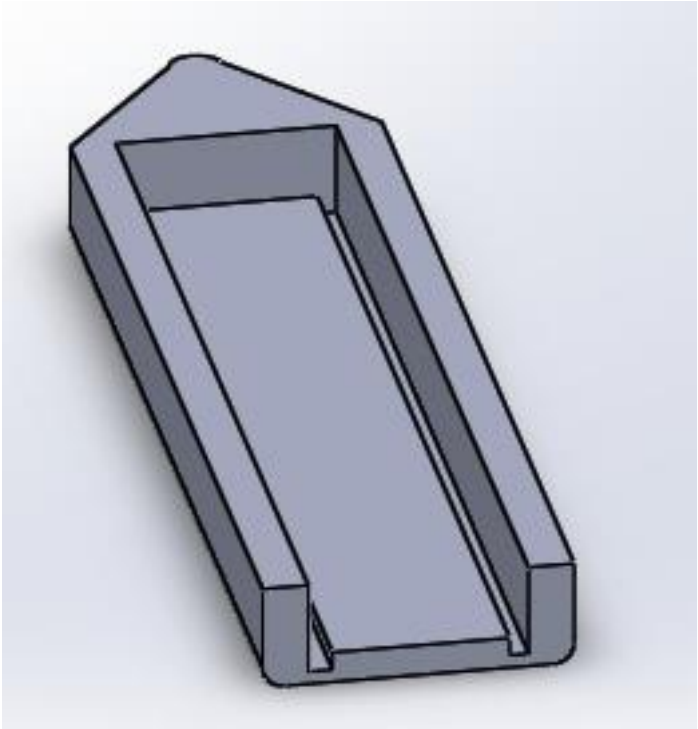


FIGURA 3.29 RANURAS PARA LAS BARRAS DE SOPORTE

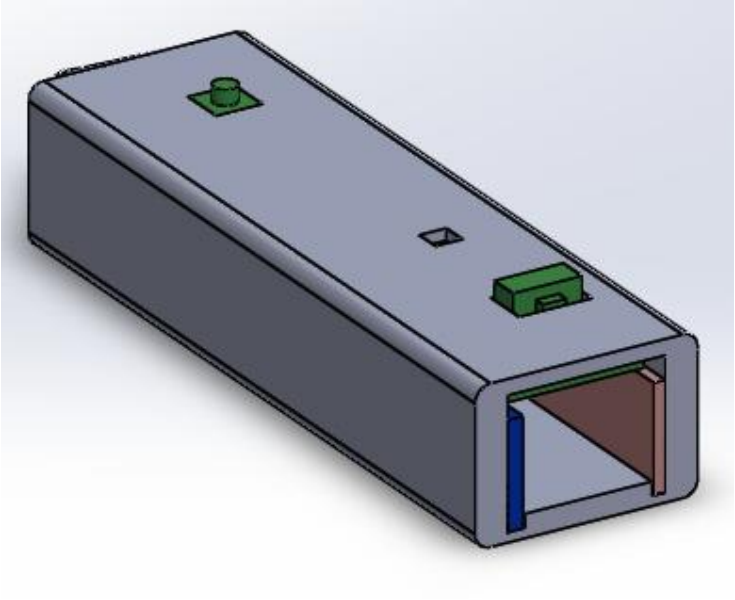


FIGURA 3.30 VISTA DE LAS BARRAS DE SOPORTE PARA LA FIJACIÓN DE LA TARJETA

La otra parte importante de la cubierta o carcasa es la tapa, la cual posee una forma cuadrangular que se ajusta a presión al cuerpo del objeto evitando que la circuitería se salga de posición. Ésta también cuenta con un orificio para

introducir el cable que lleva la alimentación al circuito, como se muestra en la Figura 3.31.

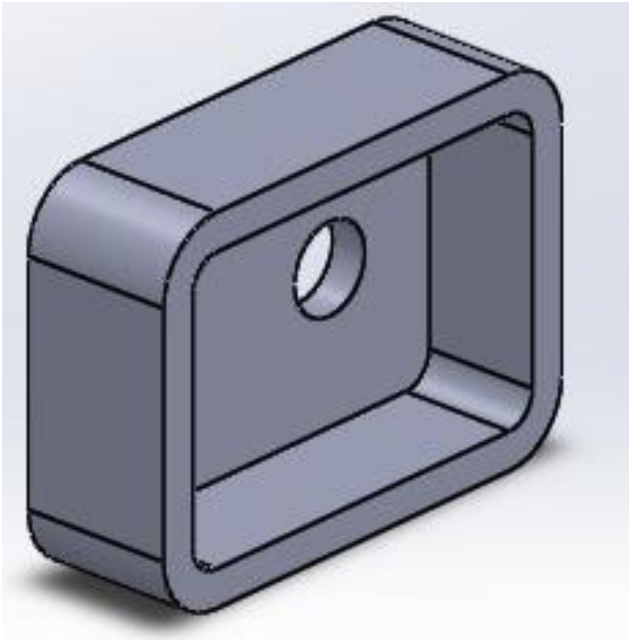


FIGURA 3.31 PROYECCIÓN ISOMÉTRICA DE LA TAPA DE LA CUBIERTA DEL DISPOSITIVO

Finalmente, se muestra en las Figuras 3.32, 3.33 y 3.34 el resultado final de la impresión de la cubierta de la pluma del prototipo.

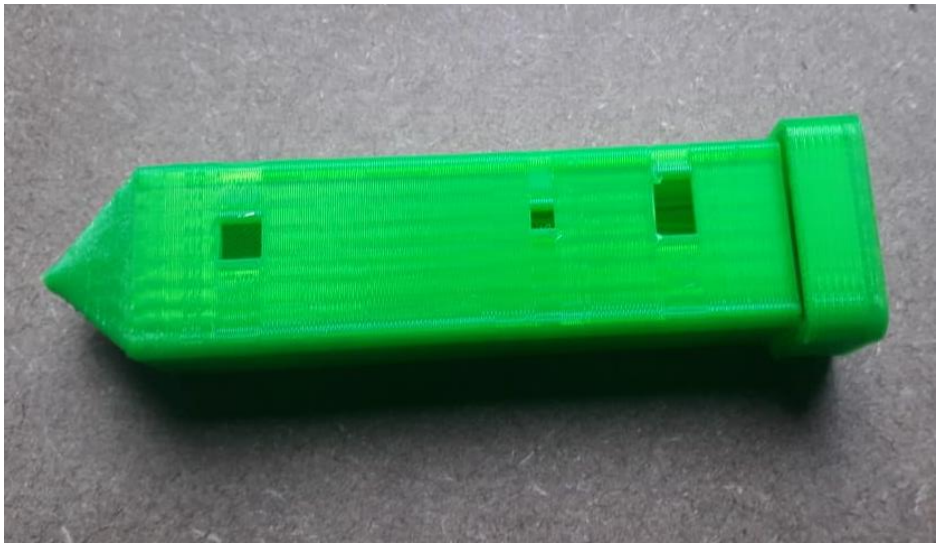


FIGURA 3.32 CUBIERTA DEL DISPOSITIVO IMPRESA

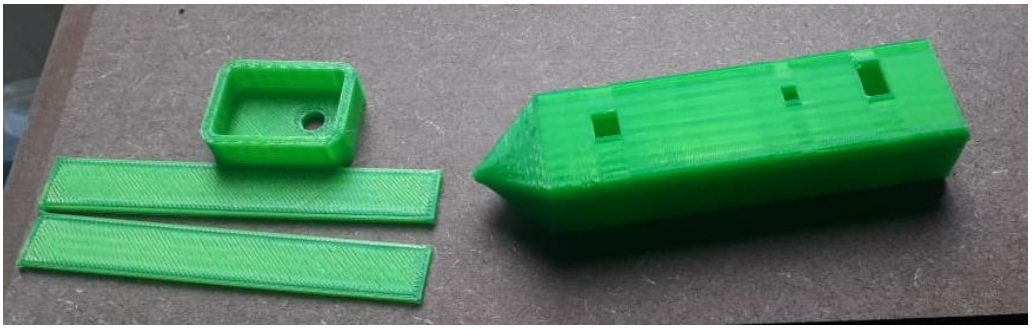


FIGURA 3.33 PIEZAS IMPRESAS DE LA CUBIERTA DEL DISPOSITIVO

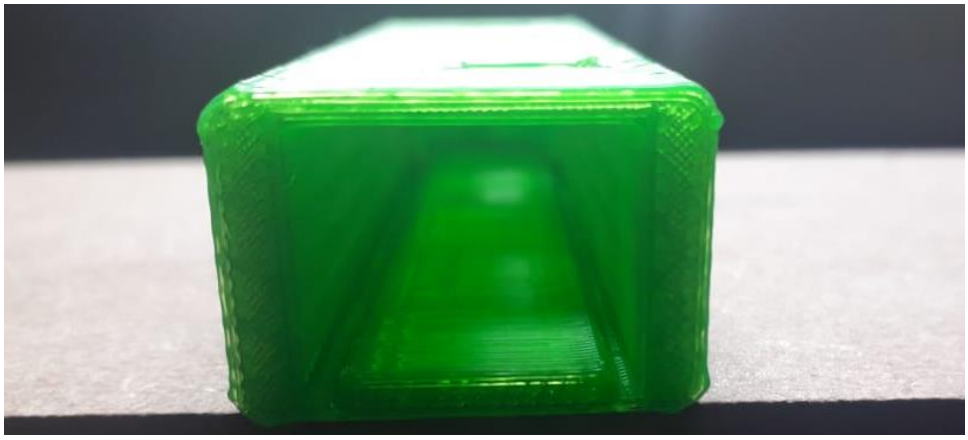


FIGURA 3.34 VISTA DE LAS RANURAS PARA LAS BARRAS DE SOPORTE EN LA PIEZA IMPRESA

3.7. INTEGRACIÓN DE LOS ELEMENTOS

Una vez que se ponen a disposición del mismo objetivo todas y cada una de las partes desarrolladas en este trabajo, la integración de los elementos luce de la siguiente manera, en la Figura 3.35 es posible observar el dispositivo conformado por su cubierta y placa de circuito impreso, así también es posible observar en esta figura, la disposición de algunos de los elementos mencionados anteriormente dentro de la cubierta, como son el botón de funcionamiento, el interruptor de encendido, el led que comprueba el funcionamiento del dispositivo y los cables de alimentación del mismo.

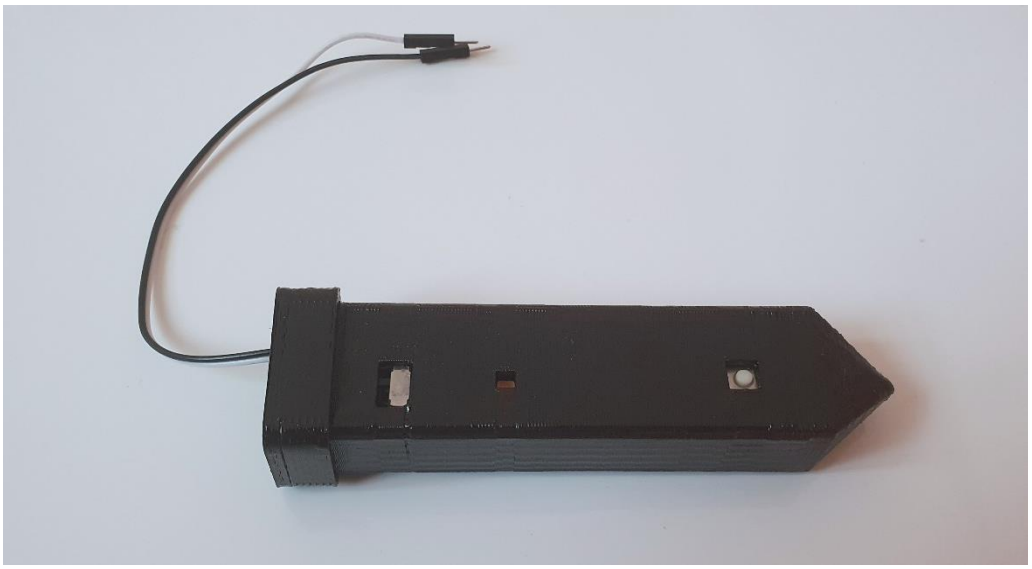


FIGURA 3.35 INTEGRACIÓN DE LOS COMPONENTES

Para una mejor apreciación del dispositivo en funcionamiento, se presenta la Figura 3.36, en la cual se observa al dispositivo conectado a la alimentación, con lo cual es posible comenzar a trabajar con él. Prueba de que se encuentra trabajando el lápiz, es que el led de funcionamiento está encendido.

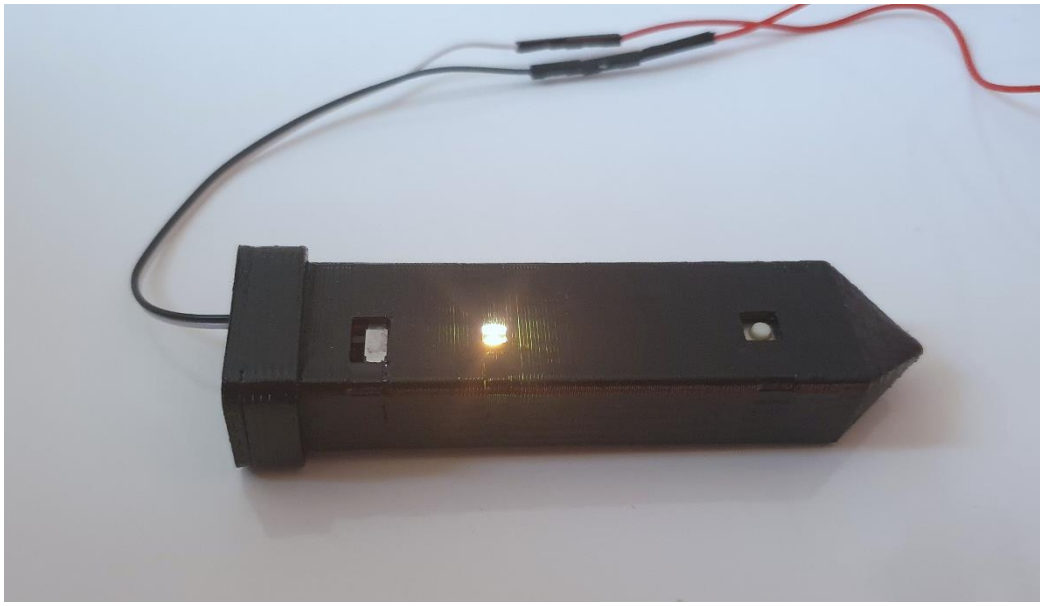


FIGURA 3.36 DISPOSITIVO EN FUNCIONAMIENTO

3.8. CONCLUSIÓN DEL CAPÍTULO

Este capítulo en especial presenta una importante relevancia, ya que muestra el desarrollo parte por parte del presente trabajo de tesis. Aquí es posible conocer cada uno de los elementos que conformaran de forma física e intangible al dispositivo que se realizó, y se explica de manera detallada cada uno de los procedimientos que se llevó a cabo, desde la estructuración y descripción del Algoritmo para la Escritura Libre, el diseño de la placa de circuito impreso, la elaboración de la interfaz gráfica de usuario y por último el diseño de la imagen del producto.

A medida que se desarrolla el capítulo es posible entender la forma de trabajar en cada una de las etapas, los inconvenientes que se presentaron y las soluciones propuestas ante estas dificultades para poder culminar en la integración de las partes del producto para obtener un resultado final optimo cercano a lo esperado y planteado por los objetivos definidos al inicio de este trabajo.

Capítulo 4

Análisis de los Resultados

4.1. PRUEBAS DE FUNCIONAMIENTO

Una vez realizada la integración de todas las partes que conforman al producto, se procede a realizar las pruebas de funcionamiento pertinentes para demostrar que se han alcanzado todos los objetivos planteados al inicio y así poder corroborar el correcto funcionamiento del dispositivo, de tal forma que pueda expresarse la conclusión exitosa del trabajo.

En la Figura 4.1, se puede observar al dispositivo encendido y listo para trabajar, de forma muy similar a la presentada en la Fig. 3.36.

FIGURA 4.1 DISPOSITIVO LÁPIZ EN FUNCIONAMIENTO

A continuación, en las Figuras 4.2 y 4.3 se muestran capturas del funcionamiento del dispositivo, específicamente, la visualización a través de la interfaz gráfica de usuario de su funcionamiento para realizar trazos.

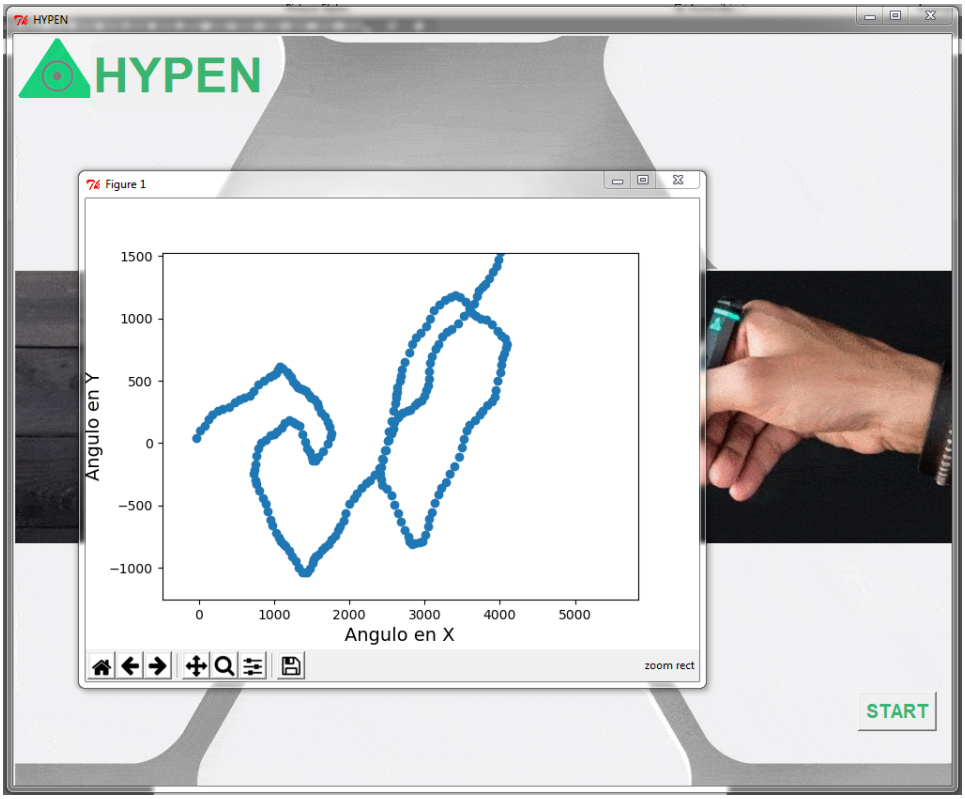


FIGURA 4.2 PRUEBA DE FUNCIONAMIENTO DEL SISTEMA

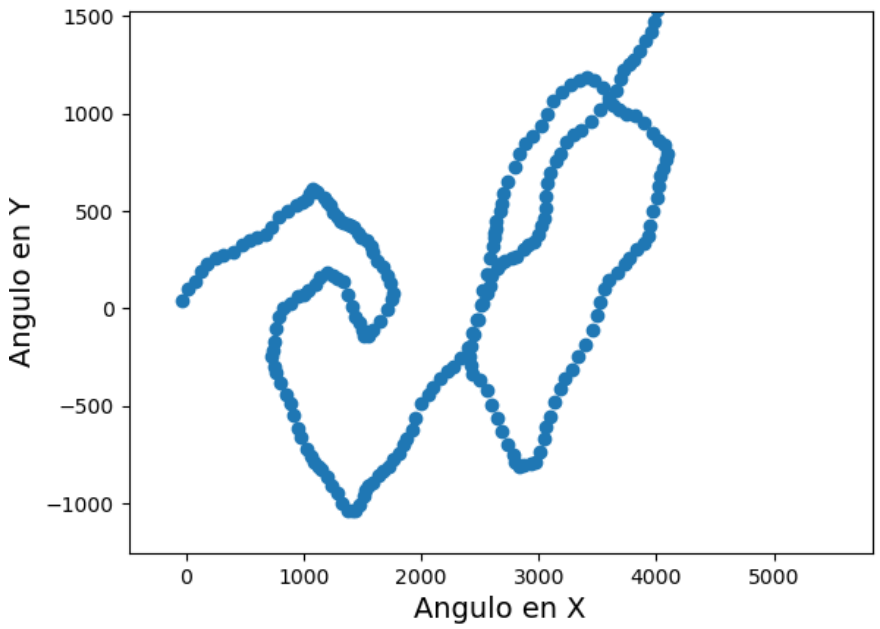


FIGURA 4.3 TRAZO OBTENIDO DEL SISTEMA EN FUNCIONAMIENTO

Por último, la Fig. 4.4 muestra la capacidad del software del dispositivo para almacenar el trazo y, obtener así, un archivo del que el usuario pueda hacer uso según su conveniencia.

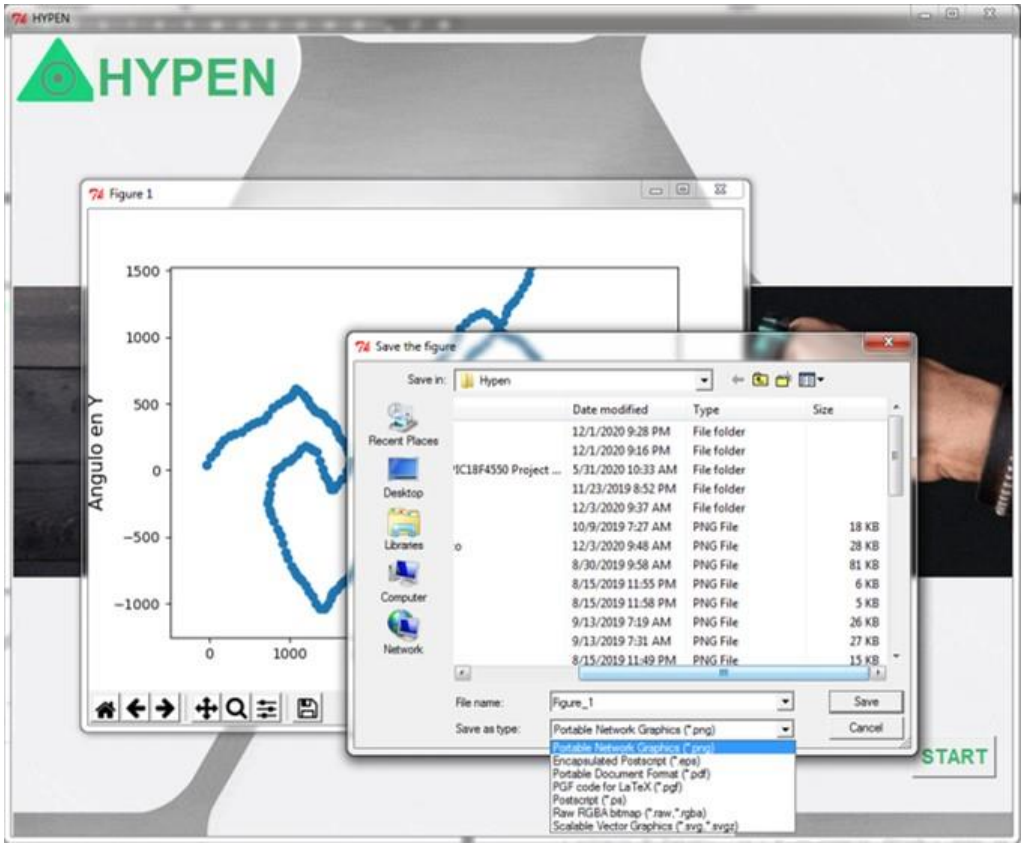


FIGURA 4.4 CAPACIDAD DE ALMACENAMIENTO DEL TRAZO

4.2. DISCUSIÓN DE LOS RESULTADOS

Para finalizar, cabe destacar que el dispositivo en conjunto con cada una de sus partes funcionan de la forma esperada y especificada a lo largo del desarrollo, es decir, el sistema muestra un desempeño optimo según lo planteado en este trabajo; sin embargo, es muy importante hacer mención y recordar que el trazo que se puede llevar a cabo con este dispositivo no asemeja los movimientos realizados con la mano del usuario, sino más bien que depende en su totalidad de la inclinación del dispositivo y que al ser una animación obtenida en tiempo real, no es necesario realizar desplazamientos, ya que el trazo por sí mismo avanza con el tiempo, asegurando así que en todo momento se están obteniendo los valores

reales del ángulo calculado y que no sufre ninguna pérdida de información al graficarse.

También, se debe señalar que este trabajo puede llevarse a otras aplicaciones y enfocarlo en otros usos, o que es posible llevar a cabo la misma tarea mediante distinto desarrollo de software, aunque carezca de innovación, pero que pueda presentar resultados más favorecedores en cuanto a su manipulación y la comodidad de su uso para el usuario, ya que es un dispositivo tan completo, que la electrónica empleada puede decirse universal para casi todas las aplicaciones que se puedan implementar, de igual forma, la interfaz creada cuenta con estas capacidades, por lo que bastaría únicamente con programar al microcontrolador del dispositivo un algoritmo que cumpla con las características deseadas y que realice las tareas solicitadas.

Con todas estas características, se puede decir, que el dispositivo obtenido al final de este trabajo de tesis no sufre de limitaciones para otras implementaciones a futuro, y que la mayoría de las partes que lo componen, de igual manera, poseen esta capacidad, teniendo así, un dispositivo que cumple con todos los requerimientos puntualizados y que además puede modificarse a conveniencia de otras necesidades y usos.

4.3. CONCLUSIÓN DEL CAPÍTULO

En este capítulo se destacan los resultados obtenidos del funcionamiento del sistema y todas las partes que lo conforman, cumpliendo con el objetivo planteado para este trabajo de tesis. Además, se analizan las característica de su desempeño, siendo específicos en la manera en la que el dispositivo trabaja y los resultados que se pueden obtener con su uso, haciendo mención de las posibilidades de realizar modificaciones en el sistema para implementarlo para otras funciones sin necesidad de alterar mayormente sus características, teniendo así como conclusión, que no sólo se logró desarrollar un sistema que cumpla con lo solicitado sino que además puede modificarse para llevar a cabo el trabajo futuro.

Capítulo 5

Conclusiones

De manera muy puntual, dentro del Grupo de Sistemas de VLSI de la SEES, en esta ocasión se propuso seleccionar un tema de Tesis que fuese diferente en su desarrollo a los trabajos realizados habitualmente, ya que se eligió trabajar en la aplicación de uno de sus principales elementos de estudio, un acelerómetro en su tipo comercial, dando paso así, a la idea de llevar a cabo un producto que disponga de este elemento como base de su funcionamiento.

El principal objetivo de este trabajo de tesis radica en cubrir la necesidad de llevar a la materialización el desarrollo y producción de un dispositivo electrónico capaz de realizar trazos virtuales al aire libre, por lo que se puede concluir lo siguiente:

- Dispositivos similares al que elaboró existen actualmente y son comercializados con diferentes finalidades y mercados objetivos distintos, por eso es de gran importancia otorgarle al proyecto un sello de autenticidad que presente innovación ante sus semejantes para que así posea ventajas por encima de ellos. Esto se logró de manera eficaz al alcanzar la independencia total del dispositivo de una superficie de apoyo o de un artefacto secundario que lo auxilie a captar la información del trazo realizado.

- Se desarrolló completa comprensión del funcionamiento del acelerómetro para entender la propuesta de esta aplicación con base en sus propiedades para así realizar la toma de decisiones adecuadas y realizar sugerencias apropiadas que contribuyeran a alcanzar todos los objetivos específicos planteados al inicio.

- La experimentación en más de tres microcontroladores distintos fue útil para conocer cuál de ellos entregaba los mejores resultados o los más cercanos a lo esperado, permitiendo realizar la elección correcta.

- La metodología seguida para definir el algoritmo que se terminó empleando, fue cambiando conforme al desarrollo del proyecto, debido a la experimentación y al análisis de las pruebas realizadas, lo que permitió la implementación de las mejores características o cualidades de cada una de las hipótesis planteadas en este trabajo.

- El algoritmo que se desarrolló se dedica a trazar la trayectoria realizada por la IMU, a través de cuan inclinada y en qué sentido se encuentre, es decir que no corresponde al trazo realizado para la escritura natural, sin embargo, es posible realizar trayectorias que se asemejen a partir de su inclinación., esta es una propuesta diferente a los algoritmos base con los que trabajan otros dispositivos,

- Es importante señalar que con el algoritmo presentado no es posible semejar el movimiento de escritura y dibujo natural del ser humano ya que el funcionamiento del sistema aquí presentado se basa en cuan inclinado se encuentre el dispositivo, es decir, no es posible escribir palabras como si de un lápiz común se tratara, para que se muestre un funcionamiento óptimo es necesario conocer cómo funciona nuestro sistema, que como ya se dijo, es conforme a la inclinación que posee, y requiere de cierta práctica para el dominio de su uso, lo cual puede intuirse con la explicación del algoritmo de libre escritura.

- Otra importante cualidad de este proyecto es que su elaboración física y la descripción del algoritmo están hechas en su mayoría haciendo uso de lenguaje de programación libre (Python y C) y software libre de licencias, lo que permite continuar su desarrollo y realizar mejoras efectivas que permitan en un futuro su materialización para la comercialización.

- Así mismo, es importante señalar que las dimensiones de la tarjeta de circuito impreso se eligieron conforme a los criterios de distribución de los elementos implementados en el dispositivo, para obtener ahorro de espacio, comodidad para el usuario al momento de usar el dispositivo y para poder trazar las pistas de interconexión entre cada uno de los elementos previniendo cualquier tipo de inconveniente en el funcionamiento. Cabe

destacar que la implementación de los módulos y elementos electrónicos de los mismos en tipo SMD brinda ventajas importantes en el ahorro de espacio y le suma atributos de estética a la tarjeta, teniendo así un diseño más prolijo y profesional.

- Finalmente, el dispositivo desarrollado en el presente trabajo cumple en su mayoría con los objetivos planteados al principio del mismo, puesto que se diseñó y fabrico un sistema electrónico capaz de escribir en el aire, que cabe resaltar, no cuenta con una superficie de apoyo ni de realimentación, siendo esta su principal característica volviéndolo un dispositivo único entre los existentes, que es portable (inalámbrico), que permite la transferencia de los datos (el trazo) en tiempo real, que es versátil (con un diseño en forma de pluma), y amigable con el medioambiente, y que, mediante un segundo dispositivo conectado de forma inalámbrica, permite la visualización y el almacenamiento del trazo realizado, basado en el funcionamiento de un acelerómetro.

Trabajo Futuro

Es posible comprender este trabajo de tesis como la suma de todas las partes desarrolladas de este proyecto para la conformación de un todo, que es lo que se busca, desarrollar un sistema completo de trazos y escritura virtual trabajando en cada uno de los detalles desde lo más simple hasta lo más complejo y desde la ingeniería hasta el diseño. Más, sin embargo, la culminación de este proyecto no implica más que sentar bases y precedentes para proyectos que puedan desarrollarse a futuro, puesto que, aunque es un trabajo completo y funcional que cumple con los objetivos planteados al inicio de este trabajo, aún existen posibilidades de crecer su desarrollo y agregarle nuevas características o modificar algunas existentes para empatar con otras necesidades tecnológicas empleadas en distintos enfoques y áreas.

Se puede decir que nuestro proyecto es una primera piedra en el desarrollo de tecnologías de escritura y trazo fiel que puede ser modificado y programado para cumplir con otros fines en esta misma área o incluso en otras como el conocimiento del desplazamiento y ubicación del dispositivo o para el entretenimiento.

Como ya se hizo mención anteriormente, las características con las que cuenta nuestro dispositivo son a grandes rasgos, la electrónica necesaria para desarrollar cualquier función que tenga que ver con ubicar una posición en un plano y poder enviar esta información de forma inalámbrica para su visualización, por lo que es casi seguro que no se requieran mayores modificaciones en ese aspecto, pero si pudiendo variar sus dimensiones, sin dejar de tener la posibilidad de manipular la tarea que desempeñe.

Algunos ejemplos de lo que se puede llegar a desarrollar, cambiar o implementar a partir del dispositivo elaborado se enlistan a continuación:

- Implementación de algoritmos conocidos que pueden ser encontrados en la literatura para obtener el trazo fiel al movimiento de la mano.
- Implementación de un sistema de alimentación ininterrumpida.

- Reconocimiento del trazo y transcripción a un tipo de fuente específico.
- Manipulación del trazo realizado para tener formatos más agradables a la vista del usuario.
- Desarrollo de una aplicación para dispositivos móviles que contenga la interfaz de usuario programada.
- Contemplar las posibilidades de su uso por usuarios que padezcan alguna deformidad o trastornos del movimiento, así como su implementación en terapias de rehabilitación.
- Optimización del diseño y dimensiones de la tarjeta de circuito impreso.
- Localización del usuario.
- Rediseño de su imagen.
- Enfoque del hardware para tareas y findes distintos a la aplicación aquí descrita.

Referencias

1. Historia de los MEMS. Antecedentes. Recuperado de http://www.csa.com/discoveryguides/mems/overview.php
2. Universidad de Sevilla. Diseño Integrado. Capítulo 4 Sensor medidor de Aceleración ACELERÓMETRO. Recuperado de http://bibing.us.es/proyectos/abreproy/11638/fichero/Capitulo+4.pdf
3. Universidad Politécnica de Catalunya. Diseño e implementación de un acelerómetro, velocímetro móvil digital Controlado por un PIC de MICROCHIP. Recuperado de https://upcommons.upc.edu/bitstream/handle/2099.1/7998/Mem%C3%B2ria.pdf
4. Corona Ramírez L. G., & Abarca Jiménez G. S., & Mares Carreño J. (2015). *Sensores y Actuadores. Aplicaciones con Arduino®. México: Grupo Editorial Patria S.A. de C.V.*
5. Pieter-Jan (2013). Reading an IMU Without the Kalman. Recuperado de http://www.pieter-jan.com/node/11
6. Kalman, R.E., "A New Approach to Linear Filtering and Prediction Problems", J. Basic Eng., March 1960.
7. Kalman, R.E. and Bucy, R.S., "New Results in Linear Filtering and Prediction Theory", J. Basic Eng., March 1961.
8. A short Course on Kalman Filtering Theory, The Analytic Sciences Corporation, Reading, Mass., EM-146,1969.
9. Ditecno Makers (2019). Como configurar el acelerómetro y giroscopio del MPU6050. Recuperado de https://ditecnomakers.com/como-configurar-el-acelerometro-y-giroscopio-del-mpu-6050/
10. Naylamp Mechatronics (2016). Configuración del módulo bluetooth HC-05 usando comandos AT. Recuperado de https://naylampmechatronics.com/blog/24_configuracion-del-modulo-bluetooth-hc-05-usa.html

Apéndice

CÓDIGO PARA LA CALIBRACIÓN DEL MÓDULO MPU-6050 EN ARDUINO

```
// Librerias I2C para controlar el mpu6050
// la libreria MPU6050.h necesita I2Cdev.h, I2Cdev.h necesita Wire.h
#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

// La dirección del MPU6050 puede ser 0x68 o 0x69, dependiendo
// del estado de AD0. Si no se especifica, 0x68 estará implicito
MPU6050 sensor;

// Valores RAW (sin procesar) del acelerometro y giroscopio en los
ejes x,y,z
int ax, ay, az;
int gx, gy, gz;

//Variables usadas por el filtro pasa bajos
long f_ax,f_ay, f_az;
int p_ax, p_ay, p_az;
long f_gx,f_gy, f_gz;
int p_gx, p_gy, p_gz;
int counter=0;

//Valor de los offsets
int ax_o,ay_o,az_o;
int gx_o,gy_o,gz_o;

void setup() {
  Serial.begin(57600);   //Iniciando puerto serial
  Wire.begin();           //Iniciando I2C
  sensor.initialize();    //Iniciando el sensor

  if (sensor.testConnection()) Serial.println("Sensor iniciado
correctamente");

  // Leer los offset los offsets anteriores
  ax_o=sensor.getXAccelOffset();
  ay_o=sensor.getYAccelOffset();
  az_o=sensor.getZAccelOffset();
  gx_o=sensor.getXGyroOffset();
  gy_o=sensor.getYGyroOffset();
  gz_o=sensor.getZGyroOffset();

  Serial.println("Offsets:");
  Serial.print(ax_o); Serial.print("\t");
```

```
  Serial.print(ay_o); Serial.print("\t");
  Serial.print(az_o); Serial.print("\t");
  Serial.print(gx_o); Serial.print("\t");
  Serial.print(gy_o); Serial.print("\t");
  Serial.print(gz_o); Serial.print("\t");
  Serial.println("nnEnvie cualquier caracter para empezar la
calibracionnn");
  // Espera un caracter para empezar a calibrar
  while (true){if (Serial.available()) break;}
  Serial.println("Calibrando, no mover IMU");

}

void loop() {
  // Leer las aceleraciones y velocidades angulares
  sensor.getAcceleration(&ax, &ay, &az);
  sensor.getRotation(&gx, &gy, &gz);

  // Filtrar las lecturas
  f_ax = f_ax-(f_ax>>5)+ax;
  p_ax = f_ax>>5;

  f_ay = f_ay-(f_ay>>5)+ay;
  p_ay = f_ay>>5;

  f_az = f_az-(f_az>>5)+az;
  p_az = f_az>>5;

  f_gx = f_gx-(f_gx>>3)+gx;
  p_gx = f_gx>>3;

  f_gy = f_gy-(f_gy>>3)+gy;
  p_gy = f_gy>>3;

  f_gz = f_gz-(f_gz>>3)+gz;
  p_gz = f_gz>>3;

  //Cada 100 lecturas corregir el offset
  if (counter==100){
    //Mostrar las lecturas separadas por un [tab]
    Serial.print("promedio:"); Serial.print("t");
    Serial.print(p_ax); Serial.print("\t");
    Serial.print(p_ay); Serial.print("\t");
    Serial.print(p_az); Serial.print("\t");
    Serial.print(p_gx); Serial.print("\t");
    Serial.print(p_gy); Serial.print("\t");
    Serial.println(p_gz);

    //Calibrar el acelerometro a 1g en el eje z (ajustar el offset)
    if (p_ax>0) ax_o--;
    else {ax_o++;}
    if (p_ay>0) ay_o--;
```

```
    else {ay_o++;}
    if (p_az-16384>0) az_o--;
    else {az_o++;}

    sensor.setXAccelOffset(ax_o);
    sensor.setYAccelOffset(ay_o);
    sensor.setZAccelOffset(az_o);

    //Calibrar el giroscopio a 0°/s en todos los ejes (ajustar el
offset)
    if (p_gx>0) gx_o--;
    else {gx_o++;}
    if (p_gy>0) gy_o--;
    else {gy_o++;}
    if (p_gz>0) gz_o--;
    else {gz_o++;}

    sensor.setXGyroOffset(gx_o);
    sensor.setYGyroOffset(gy_o);
    sensor.setZGyroOffset(gz_o);

    counter=0;
  }
  counter++;
}
```

CÓDIGO PARA OBTENER LAS MEDICIONES DE ACELERACIÓN Y VELOCIDAD ANGULAR DEL MÓDULO MPU-6050 EN ARDUINO

```
// Librerias I2C para controlar el mpu6050
// la libreria MPU6050.h necesita I2Cdev.h, I2Cdev.h necesita Wire.h
#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

// La dirección del MPU6050 puede ser 0x68 o 0x69, dependiendo
// del estado de AD0. Si no se especifica, 0x68 estará implicito
MPU6050 sensor;

// Valores RAW (sin procesar) del acelerometro y giroscopio en los
ejes x,y,z
int ax, ay, az;
int gx, gy, gz;

void setup() {
  Serial.begin(57600);    //Iniciando puerto serial
  Wire.begin();           //Iniciando I2C
  sensor.initialize();    //Iniciando el sensor

  if (sensor.testConnection()) Serial.println("Sensor iniciado
correctamente");
  else Serial.println("Error al iniciar el sensor");
}

void loop() {
  // Leer las aceleraciones y velocidades angulares
  sensor.getAcceleration(&ax, &ay, &az);
  sensor.getRotation(&gx, &gy, &gz);

  //Mostrar las lecturas separadas por un [tab]
  Serial.print("a[x y z] g[x y z]:\t");
  Serial.print(ax); Serial.print("\t");
  Serial.print(ay); Serial.print("\t");
  Serial.print(az); Serial.print("\t");
  Serial.print(gx); Serial.print("\t");
  Serial.print(gy); Serial.print("\t");
  Serial.println(gz);

  delay(100);
}
```

CÓDIGO DE PRUEBA DEL MÓDULO MPU-6050 EN ARDUINO

```
#include <Wire.h>

//Registros de acuerdo a la hoja de especificaciones

#define MPU6050_AUX_VDDIO          0x01   // R/W
#define MPU6050_SMPLRT_DIV         0x19   // R/W
#define MPU6050_CONFIG             0x1A   // R/W
#define MPU6050_GYRO_CONFIG        0x1B   // R/W
#define MPU6050_ACCEL_CONFIG       0x1C   // R/W
#define MPU6050_FF_THR             0x1D   // R/W
#define MPU6050_FF_DUR             0x1E   // R/W
#define MPU6050_MOT_THR            0x1F   // R/W
#define MPU6050_MOT_DUR            0x20   // R/W
#define MPU6050_ZRMOT_THR          0x21   // R/W
#define MPU6050_ZRMOT_DUR          0x22   // R/W
#define MPU6050_FIFO_EN            0x23   // R/W
#define MPU6050_I2C_MST_CTRL       0x24   // R/W
#define MPU6050_I2C_SLV0_ADDR      0x25   // R/W
#define MPU6050_I2C_SLV0_REG       0x26   // R/W
#define MPU6050_I2C_SLV0_CTRL      0x27   // R/W
#define MPU6050_I2C_SLV1_ADDR      0x28   // R/W
#define MPU6050_I2C_SLV1_REG       0x29   // R/W
#define MPU6050_I2C_SLV1_CTRL      0x2A   // R/W
#define MPU6050_I2C_SLV2_ADDR      0x2B   // R/W
#define MPU6050_I2C_SLV2_REG       0x2C   // R/W
#define MPU6050_I2C_SLV2_CTRL      0x2D   // R/W
#define MPU6050_I2C_SLV3_ADDR      0x2E   // R/W
#define MPU6050_I2C_SLV3_REG       0x2F   // R/W
#define MPU6050_I2C_SLV3_CTRL      0x30   // R/W
#define MPU6050_I2C_SLV4_ADDR      0x31   // R/W
#define MPU6050_I2C_SLV4_REG       0x32   // R/W
#define MPU6050_I2C_SLV4_DO        0x33   // R/W
#define MPU6050_I2C_SLV4_CTRL      0x34   // R/W
#define MPU6050_I2C_SLV4_DI        0x35   // R
#define MPU6050_I2C_MST_STATUS     0x36   // R
#define MPU6050_INT_PIN_CFG        0x37   // R/W
#define MPU6050_INT_ENABLE         0x38   // R/W
#define MPU6050_INT_STATUS         0x3A   // R
#define MPU6050_ACCEL_XOUT_H       0x3B   // R
#define MPU6050_ACCEL_XOUT_L       0x3C   // R
#define MPU6050_ACCEL_YOUT_H       0x3D   // R
#define MPU6050_ACCEL_YOUT_L       0x3E   // R
#define MPU6050_ACCEL_ZOUT_H       0x3F   // R
#define MPU6050_ACCEL_ZOUT_L       0x40   // R
#define MPU6050_TEMP_OUT_H         0x41   // R
#define MPU6050_TEMP_OUT_L         0x42   // R
#define MPU6050_GYRO_XOUT_H        0x43   // R
#define MPU6050_GYRO_XOUT_L        0x44   // R
```

```
#define MPU6050_GYRO_YOUT_H        0x45   // R
#define MPU6050_GYRO_YOUT_L        0x46   // R
#define MPU6050_GYRO_ZOUT_H        0x47   // R
#define MPU6050_GYRO_ZOUT_L        0x48   // R
#define MPU6050_EXT_SENS_DATA_00   0x49   // R
#define MPU6050_EXT_SENS_DATA_01   0x4A   // R
#define MPU6050_EXT_SENS_DATA_02   0x4B   // R
#define MPU6050_EXT_SENS_DATA_03   0x4C   // R
#define MPU6050_EXT_SENS_DATA_04   0x4D   // R
#define MPU6050_EXT_SENS_DATA_05   0x4E   // R
#define MPU6050_EXT_SENS_DATA_06   0x4F   // R
#define MPU6050_EXT_SENS_DATA_07   0x50   // R
#define MPU6050_EXT_SENS_DATA_08   0x51   // R
#define MPU6050_EXT_SENS_DATA_09   0x52   // R
#define MPU6050_EXT_SENS_DATA_10   0x53   // R
#define MPU6050_EXT_SENS_DATA_11   0x54   // R
#define MPU6050_EXT_SENS_DATA_12   0x55   // R
#define MPU6050_EXT_SENS_DATA_13   0x56   // R
#define MPU6050_EXT_SENS_DATA_14   0x57   // R
#define MPU6050_EXT_SENS_DATA_15   0x58   // R
#define MPU6050_EXT_SENS_DATA_16   0x59   // R
#define MPU6050_EXT_SENS_DATA_17   0x5A   // R
#define MPU6050_EXT_SENS_DATA_18   0x5B   // R
#define MPU6050_EXT_SENS_DATA_19   0x5C   // R
#define MPU6050_EXT_SENS_DATA_20   0x5D   // R
#define MPU6050_EXT_SENS_DATA_21   0x5E   // R
#define MPU6050_EXT_SENS_DATA_22   0x5F   // R
#define MPU6050_EXT_SENS_DATA_23   0x60   // R
#define MPU6050_MOT_DETECT_STATUS  0x61   // R
#define MPU6050_I2C_SLV0_DO        0x63   // R/W
#define MPU6050_I2C_SLV1_DO        0x64   // R/W
#define MPU6050_I2C_SLV2_DO        0x65   // R/W
#define MPU6050_I2C_SLV3_DO        0x66   // R/W
#define MPU6050_I2C_MST_DELAY_CTRL 0x67   // R/W
#define MPU6050_SIGNAL_PATH_RESET  0x68   // R/W
#define MPU6050_MOT_DETECT_CTRL    0x69   // R/W
#define MPU6050_USER_CTRL          0x6A   // R/W
#define MPU6050_PWR_MGMT_1         0x6B   // R/W
#define MPU6050_PWR_MGMT_2         0x6C   // R/W
#define MPU6050_FIFO_COUNTH        0x72   // R/W
#define MPU6050_FIFO_COUNTL        0x73   // R/W
#define MPU6050_FIFO_R_W           0x74   // R/W
#define MPU6050_WHO_AM_I           0x75   // R
#define MPU6050_ADDRESS            0x68


void setup(void){
  Serial.begin(9600);
  Wire.begin();
  delay(50);
```

```
  MPU6050_configuracion(MPU6050_CONFIG,0x03);   //(EXT_SYNC_SET=0,
DLPF_CFG=3)
  MPU6050_configuracion(MPU6050_GYRO_CONFIG,0x10); //(XG_ST=0,
YG_ST=0, ZG_ST=0, FS_SEL=2 [1000°/s] )
  MPU6050_configuracion(MPU6050_ACCEL_CONFIG,0x10); //(XA_ST=0,
YA_ST=0, ZA_ST=0, FS_SEL=2 [8g] )
  MPU6050_configuracion(MPU6050_PWR_MGMT_1,0x00);
//(DEVICE_RESET=0, SLEEP=0,
 // CYCLE=0, TEMP_DIS=0, CLK_SEL=0 [Internal 8MHz oscillator] )
  delay(50);
}
void loop(void){
  unsigned short int gyro_x_h=0, gyro_x_l=0, gyro_y_h=0,
gyro_y_l=0,gyro_z_h=0, gyro_z_l=0;
  unsigned short int accel_x_h=0, accel_x_l=0,
accel_y_h=0,accel_y_l=0, accel_z_h=0, accel_z_l=0;
  unsigned short int temp_h=0, temp_l=0;
  int gyro_x=0, gyro_y=0, gyro_z=0;
  int accel_x=0, accel_y=0, accel_z=0;
  int temp=0;
      Wire.beginTransmission(MPU6050_ADDRESS);
      Wire.write(MPU6050_ACCEL_XOUT_H);
      Wire.requestFrom(MPU6050_ADDRESS, 14);
      accel_x_h = Wire.read();
      accel_x_l = Wire.read();
      accel_y_h = Wire.read();
      accel_y_l = Wire.read();
      accel_z_h = Wire.read();
      accel_z_l = Wire.read();
      temp_h = Wire.read();
      temp_l = Wire.read();
      gyro_x_h = Wire.read();
      gyro_x_l = Wire.read();
      gyro_y_h = Wire.read();
      gyro_y_l = Wire.read();
      gyro_z_h = Wire.read();
      gyro_z_l = Wire.read();
      Wire.endTransmission();
      accel_x=(((int)accel_x_h)<<8 )  |  ((int)accel_x_l);
      accel_y=(((int)accel_y_h)<<8 )  |  ((int)accel_y_l);
      accel_z=(((int)accel_z_h)<<8 )  |  ((int)accel_z_l);
      temp=(((int)temp_h)<<8 )  |  ((int)temp_l);
      gyro_x=(((int)gyro_x_h)<<8 )  |  ((int)gyro_x_l);
      gyro_y=(((int)gyro_y_h)<<8 )  |  ((int)gyro_y_l);
      gyro_z=(((int)gyro_z_h)<<8 )  |  ((int)gyro_z_l);
  Serial.print("Giroscopio x,y,z: ");
  Serial.print(gyro_x);
  Serial.print(",");
  Serial.print(gyro_y);
  Serial.print(",");
  Serial.println(gyro_z);
  Serial.print("Acelerometro x,y,z: ");
```

```
  Serial.print(accel_x);
  Serial.print(",");
  Serial.print(accel_y);
  Serial.print(",");
  Serial.println(accel_z);
  Serial.print("Temperatura: ");
  Serial.println((float)temp/340 + 36.53); //basado en la hoja de
especificaciones
  Serial.println();
  delay(500);
}

void MPU6050_configuracion(int r1, int r2){
    Wire.beginTransmission(MPU6050_ADDRESS);
    Wire.write(r1);
    Wire.write(r2);
    Wire.endTransmission();
    delayMicroseconds(20);
}
```

CÓDIGO PARA LA CONFIGURACIÓN DEL MÓDULO BLUETOOTH HC-05 EN ARDUINO

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(2, 3); //Crea conexión al bluetooth - PIN 2
A TX y PIN 3 A RX
char  NOMBRE[21] = "CarBluetooth"; //Nombre de 20 caracteres máximo
char BPS = '4'; //1=1200, 2=2400, 3=4800, 4=9600, 5=19200, 6=38400,
7=57600, 8=115200
char PASS[5] = "1234"; //PIN O CLAVE de 4 caracteres numéricos

void setup() {
  // Open serial communications and wait for port to open:
  Serial.begin(9600);

  pinMode(13, OUTPUT);
  digitalWrite(13, HIGH); //Enciende el LED 13 durante 4s antes de
configurar el Bluetooth
  delay(4000);

  digitalWrite(13,LOW); //Apaga el LED 13 para iniciar la
programación

  Serial.print("AT"); //Inicia el comando AT
  delay(1000);

  Serial.print("AT+NAME"); //Configura el nuevo nombre
  Serial.print(NOMBRE);
  delay(1000);          //espera 1 segundo

  Serial.print("AT+BAUD"); //Configura la nueva velocidad
  Serial.print(BPS);
  delay(1000);

  Serial.print("AT+PIN"); //Configura la nueva contraseña
  Serial.print(PASS);
  delay(1000);
}

void loop() { // run over and over
  digitalWrite(13, !digitalRead(13)); // cuando termina de
configurar el Bluetooth queda el LED 13 parpadeando
  delay(300);
}
```

CÓDIGO PARA EL CÁLCULO DEL ÁNGULO DE INCLINACIÓN DE LA IMU EN ARDUINO

```
// Librerias I2C para controlar el mpu6050
// la libreria MPU6050.h necesita I2Cdev.h, I2Cdev.h necesita Wire.h
#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

// La dirección del MPU6050 puede ser 0x68 o 0x69, dependiendo
// del estado de AD0. Si no se especifica, 0x68 estará implicito
MPU6050 sensor;

// Valores RAW (sin procesar) del acelerometro  en los ejes x,y,z
int ax, ay, az;

void setup() {
  Serial.begin(57600);    //Iniciando puerto serial
  Wire.begin();           //Iniciando I2C
  sensor.initialize();    //Iniciando el sensor

  if (sensor.testConnection()) Serial.println("Sensor iniciado
correctamente");
  else Serial.println("Error al iniciar el sensor");
}

void loop() {
  // Leer las aceleraciones
  sensor.getAcceleration(&ax, &ay, &az);
  //Calcular los angulos de inclinacion:
  float accel_ang_x=atan(ax/sqrt(pow(ay,2) +
pow(az,2)))*(180.0/3.14);
  float accel_ang_y=atan(ay/sqrt(pow(ax,2) +
pow(az,2)))*(180.0/3.14);
  //Mostrar los angulos separadas por un [tab]
  Serial.print("Inclinacion en X: ");
  Serial.print(accel_ang_x);
  Serial.print("tInclinacion en Y:");
  Serial.println(accel_ang_y);
  delay(10);
}
```

CÓDIGO PARA EL CÁLCULO DEL ÁNGULO DE ROTACIÓN DE LA IMU EN ARDUINO

```
// Librerias I2C para controlar el mpu6050
// la libreria MPU6050.h necesita I2Cdev.h, I2Cdev.h necesita Wire.h
#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

// La dirección del MPU6050 puede ser 0x68 o 0x69, dependiendo
// del estado de AD0. Si no se especifica, 0x68 estará implicito
MPU6050 sensor;

// Valores RAW (sin procesar) del acelerometro y giroscopio en los
ejes x,y,z
int gx, gy, gz;

long tiempo_prev, dt;
float girosc_ang_x, girosc_ang_y;
float girosc_ang_x_prev, girosc_ang_y_prev;

void setup() {
  Serial.begin(57600);    //Iniciando puerto serial
  Wire.begin();           //Iniciando I2C
  sensor.initialize();    //Iniciando el sensor

  if (sensor.testConnection()) Serial.println("Sensor iniciado
correctamente");
  else Serial.println("Error al iniciar el sensor");
  tiempo_prev=millis();
}



void loop() {
  // Leer las velocidades angulares
  sensor.getRotation(&gx, &gy, &gz);

  //Calcular los angulos rotacion:

  dt = millis()-tiempo_prev;
  tiempo_prev=millis();

  girosc_ang_x = (gx/131)*dt/1000.0 + girosc_ang_x_prev;
  girosc_ang_y = (gy/131)*dt/1000.0 + girosc_ang_y_prev;

  girosc_ang_x_prev=girosc_ang_x;
  girosc_ang_y_prev=girosc_ang_y;

  //Mostrar los angulos separadas por un [tab]
  Serial.print("Rotacion en X:  ");
  Serial.print(girosc_ang_x);
```

```
  Serial.print("tRotacion en Y: ");
  Serial.println(girosc_ang_y);

  delay(100);
}
```

CÓDIGO PARA EL CÁLCULO DE LOS ÁNGULOS DE INCLINACIÓN Y ROTACIÓN DE LA IMU EN ARDUINO

```
// Librerias I2C para controlar el mpu6050
// la libreria MPU6050.h necesita I2Cdev.h, I2Cdev.h necesita Wire.h
#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

// La dirección del MPU6050 puede ser 0x68 o 0x69, dependiendo
// del estado de AD0. Si no se especifica, 0x68 estará implicito
MPU6050 sensor;

// Valores RAW (sin procesar) del acelerometro y giroscopio en los
ejes x,y,z
int ax, ay, az;
int gx, gy, gz;

long tiempo_prev;
float dt;
float ang_x, ang_y;
float ang_x_prev, ang_y_prev;

void setup() {
  Serial.begin(57600);    //Iniciando puerto serial
  Wire.begin();           //Iniciando I2C
  sensor.initialize();    //Iniciando el sensor

  if (sensor.testConnection()) Serial.println("Sensor iniciado
correctamente");
  else Serial.println("Error al iniciar el sensor");
}

void loop() {
  // Leer las aceleraciones y velocidades angulares
  sensor.getAcceleration(&ax, &ay, &az);
  sensor.getRotation(&gx, &gy, &gz);

  dt = (millis()-tiempo_prev)/1000.0;
  tiempo_prev=millis();

  //Calcular los ángulos con acelerometro
  float accel_ang_x=atan(ay/sqrt(pow(ax,2) +
pow(az,2)))*(180.0/3.14);
  float accel_ang_y=atan(-ax/sqrt(pow(ay,2) +
pow(az,2)))*(180.0/3.14);

  //Calcular angulo de rotación con giroscopio y filtro complemento
  ang_x = 0.98*(ang_x_prev+(gx/131)*dt) + 0.02*accel_ang_x;
  ang_y = 0.98*(ang_y_prev+(gy/131)*dt) + 0.02*accel_ang_y;
```

```
  ang_x_prev=ang_x;
  ang_y_prev=ang_y;

  //Mostrar los angulos separadas por un [tab]

  Serial.print("Rotacion en X:  ");
  Serial.print(ang_x);
  Serial.print("tRotacion en Y: ");
  Serial.println(ang_y);

  delay(10);
}
```

CÓDIGO PARA EL ALGORITMO DE ESCRITURA LIBRE EN ARDUINO

```
//PROGRAMA CON MODIFICACIONES PARA ARDUINO LEONARDO
//programa que  emplea switch
// MPU-6050 FILTRO COMPLEMENTARIO
// retardo para sincronizar comunicacion

#include <Wire.h>
#include <SoftwareSerial.h>
SoftwareSerial mySerial(4, 5);

//Registros de acuerdo a la hoja de especificaciones
#define MPU6050_AUX_VDDIO          0x01   // R/W
#define MPU6050_SMPLRT_DIV         0x19   // R/W
#define MPU6050_CONFIG             0x1A   // R/W
#define MPU6050_GYRO_CONFIG        0x1B   // R/W
#define MPU6050_ACCEL_CONFIG       0x1C   // R/W
#define MPU6050_FF_THR             0x1D   // R/W
#define MPU6050_FF_DUR             0x1E   // R/W
#define MPU6050_MOT_THR            0x1F   // R/W
#define MPU6050_MOT_DUR            0x20   // R/W
#define MPU6050_ZRMOT_THR          0x21   // R/W
#define MPU6050_ZRMOT_DUR          0x22   // R/W
#define MPU6050_FIFO_EN            0x23   // R/W
#define MPU6050_I2C_MST_CTRL       0x24   // R/W
#define MPU6050_I2C_SLV0_ADDR      0x25   // R/W
#define MPU6050_I2C_SLV0_REG       0x26   // R/W
#define MPU6050_I2C_SLV0_CTRL      0x27   // R/W
#define MPU6050_I2C_SLV1_ADDR      0x28   // R/W
#define MPU6050_I2C_SLV1_REG       0x29   // R/W
#define MPU6050_I2C_SLV1_CTRL      0x2A   // R/W
#define MPU6050_I2C_SLV2_ADDR      0x2B   // R/W
#define MPU6050_I2C_SLV2_REG       0x2C   // R/W
#define MPU6050_I2C_SLV2_CTRL      0x2D   // R/W
#define MPU6050_I2C_SLV3_ADDR      0x2E   // R/W
#define MPU6050_I2C_SLV3_REG       0x2F   // R/W
#define MPU6050_I2C_SLV3_CTRL      0x30   // R/W
#define MPU6050_I2C_SLV4_ADDR      0x31   // R/W
#define MPU6050_I2C_SLV4_REG       0x32   // R/W
#define MPU6050_I2C_SLV4_DO        0x33   // R/W
#define MPU6050_I2C_SLV4_CTRL      0x34   // R/W
#define MPU6050_I2C_SLV4_DI        0x35   // R
#define MPU6050_I2C_MST_STATUS     0x36   // R
#define MPU6050_INT_PIN_CFG        0x37   // R/W
#define MPU6050_INT_ENABLE         0x38   // R/W
#define MPU6050_INT_STATUS         0x3A   // R
#define MPU6050_ACCEL_XOUT_H       0x3B   // R
#define MPU6050_ACCEL_XOUT_L       0x3C   // R
#define MPU6050_ACCEL_YOUT_H       0x3D   // R
#define MPU6050_ACCEL_YOUT_L       0x3E   // R
#define MPU6050_ACCEL_ZOUT_H       0x3F   // R
```

```
#define MPU6050_ACCEL_ZOUT_L       0x40   // R
#define MPU6050_EXT_SENS_DATA_00   0x49   // R
#define MPU6050_EXT_SENS_DATA_01   0x4A   // R
#define MPU6050_EXT_SENS_DATA_02   0x4B   // R
#define MPU6050_EXT_SENS_DATA_03   0x4C   // R
#define MPU6050_EXT_SENS_DATA_04   0x4D   // R
#define MPU6050_EXT_SENS_DATA_05   0x4E   // R
#define MPU6050_EXT_SENS_DATA_06   0x4F   // R
#define MPU6050_EXT_SENS_DATA_07   0x50   // R
#define MPU6050_EXT_SENS_DATA_08   0x51   // R
#define MPU6050_EXT_SENS_DATA_09   0x52   // R
#define MPU6050_EXT_SENS_DATA_10   0x53   // R
#define MPU6050_EXT_SENS_DATA_11   0x54   // R
#define MPU6050_EXT_SENS_DATA_12   0x55   // R
#define MPU6050_EXT_SENS_DATA_13   0x56   // R
#define MPU6050_EXT_SENS_DATA_14   0x57   // R
#define MPU6050_EXT_SENS_DATA_15   0x58   // R
#define MPU6050_EXT_SENS_DATA_16   0x59   // R
#define MPU6050_EXT_SENS_DATA_17   0x5A   // R
#define MPU6050_EXT_SENS_DATA_18   0x5B   // R
#define MPU6050_EXT_SENS_DATA_19   0x5C   // R
#define MPU6050_EXT_SENS_DATA_20   0x5D   // R
#define MPU6050_EXT_SENS_DATA_21   0x5E   // R
#define MPU6050_EXT_SENS_DATA_22   0x5F   // R
#define MPU6050_EXT_SENS_DATA_23   0x60   // R
#define MPU6050_MOT_DETECT_STATUS  0x61   // R
#define MPU6050_I2C_SLV0_DO        0x63   // R/W
#define MPU6050_I2C_SLV1_DO        0x64   // R/W
#define MPU6050_I2C_SLV2_DO        0x65   // R/W
#define MPU6050_I2C_SLV3_DO        0x66   // R/W
#define MPU6050_I2C_MST_DELAY_CTRL 0x67   // R/W
#define MPU6050_SIGNAL_PATH_RESET  0x68   // R/W
#define MPU6050_MOT_DETECT_CTRL    0x69   // R/W
#define MPU6050_USER_CTRL          0x6A   // R/W
#define MPU6050_PWR_MGMT_1         0x6B   // R/W
#define MPU6050_PWR_MGMT_2         0x6C   // R/W
#define MPU6050_FIFO_COUNTH        0x72   // R/W
#define MPU6050_FIFO_COUNTL        0x73   // R/W
#define MPU6050_FIFO_R_W           0x74   // R/W
#define MPU6050_WHO_AM_I           0x75   // R
#define MPU6050_ADDRESS            0x68

unsigned long timer=0;
long loopTime = 140; //milisegundos !!!!!mayor que interval python
long tiempo_prev;
float dt;
float ang_x, ang_y;
float ang_x_prev, ang_y_prev;

void setup() {
  Serial.begin(9600);
  timer = millis();
```

```
  mySerial.begin(9600);
  pinMode(7, INPUT);
  Wire.begin();
  delay(50);
  MPU6050_configuracion(MPU6050_CONFIG, 0x03);  //(EXT_SYNC_SET=0,
DLPF_CFG=3)
  MPU6050_configuracion(MPU6050_ACCEL_CONFIG, 0x10); //(XA_ST=0,
YA_ST=0, ZA_ST=0, FS_SEL=2 [8g] )
  MPU6050_configuracion(MPU6050_PWR_MGMT_1, 0x00);
//(DEVICE_RESET=0, SLEEP=0,
  // CYCLE=0, TEMP_DIS=0, CLK_SEL=0 [Internal 8MHz oscillator] )
  delay(50);
  tiempo_prev=millis();
}

void loop() {
  unsigned short int gyro_x_h=0, gyro_x_l=0, gyro_y_h=0,
gyro_y_l=0,gyro_z_h=0, gyro_z_l=0;
  unsigned short int accel_x_h = 0, accel_x_l = 0, accel_y_h = 0,
accel_y_l = 0, accel_z_h = 0, accel_z_l = 0;
  int gyro_x=0, gyro_y=0, gyro_z=0;
  int accel_x = 0, accel_y = 0, accel_z = 0;
  int sw;
  int accel_ang_xz, accel_ang_yz, accel_ang_xy, accel_ang_x3,
accel_ang_y3, accel_ang_z3;
  double proyX, proyY;

  //TIEMPO DE RETRASO PARA TRANSMISION
  timeSync(loopTime);

  //ADQUISION DE DATOS
  Wire.beginTransmission(MPU6050_ADDRESS);
  Wire.write(MPU6050_ACCEL_XOUT_H);
  Wire.requestFrom(MPU6050_ADDRESS, 14);
  accel_x_h = Wire.read();
  accel_x_l = Wire.read();
  accel_y_h = Wire.read();
  accel_y_l = Wire.read();
  accel_z_h = Wire.read();
  accel_z_l = Wire.read();
  gyro_x_h = Wire.read();
  gyro_x_l = Wire.read();
  gyro_y_h = Wire.read();
  gyro_y_l = Wire.read();
  gyro_z_h = Wire.read();
  gyro_z_l = Wire.read();

  Wire.endTransmission();
  accel_x = (((int)accel_x_h) << 8 )  |  ((int)accel_x_l);
  accel_y = (((int)accel_y_h) << 8 )  |  ((int)accel_y_l);
  accel_z = (((int)accel_z_h) << 8 )  |  ((int)accel_z_l);
  gyro_x=(((int)gyro_x_h)<<8 )  |  ((int)gyro_x_l);
```

```
  gyro_y=(((int)gyro_y_h)<<8 )  |  ((int)gyro_y_l);
  gyro_z=(((int)gyro_z_h)<<8 )  |  ((int)gyro_z_l);

  dt = (millis()-tiempo_prev)/1000.0;
  tiempo_prev=millis();

  //CALCULO DE ANGULOS

  //Calcular los angulos de inclinacion en 2D:
//  accel_ang_xz = atan2(accel_x, accel_z) * (180.0 / 3.14);
//  accel_ang_yz = atan2(accel_y, accel_z) * (180.0 / 3.14);
//  accel_ang_xy = atan2(accel_y, accel_x) * (180.0 / 3.14);
//  if (accel_ang_xz < 0) accel_ang_xz += 360;
//  if (accel_ang_yz < 0) accel_ang_yz += 360;
//  if (accel_ang_xy < 0) accel_ang_xy += 360;

  //Calcular los angulos de inclinacion en 3D:
  accel_ang_x3 = atan(accel_x / sqrt(pow(accel_y, 2) + pow(accel_z,
2))) * (180.0 / 3.14);
  accel_ang_y3 = atan(accel_y / sqrt(pow(accel_x, 2) + pow(accel_z,
2))) * (180.0 / 3.14);
  accel_ang_z3 = atan(accel_z / sqrt(pow(accel_x, 2) + pow(accel_y,
2))) * (180.0 / 3.14);
//  if (accel_ang_x3< 0) accel_ang_x3 +=360;
//  if (accel_ang_y3< 0) accel_ang_y3 +=360;
//  if (accel_ang_z3< 0) accel_ang_z3 +=360;


  //Calcular los angulos de rotación con giroscopio y filtro
complemento:

  ang_x = 0.7*(ang_x_prev+(gyro_x/131)*dt) + 0.3*accel_ang_x3;
  ang_y = 0.7*(ang_y_prev+(gyro_y/131)*dt) + 0.3*accel_ang_y3;


  ang_x_prev=ang_x;
  ang_y_prev=ang_y;



//  proyX= 0.5*cos(ang_x);
//  proyY= 0.5*sin(ang_x);
  sw = digitalRead(7);




   //ENVIAR DATOS POR PUERTO SERIAL


  if (sw == LOW) {
    mySerial.print(ang_x);
```

```
    mySerial.print(",");
    mySerial.print(ang_y);
    mySerial.print(",");
    mySerial.println(sw);
}


  if (sw == HIGH) {
    mySerial.print(0);
    mySerial.print(",");
    mySerial.print(0);
    mySerial.print(",");
    mySerial.println(sw);
     }

  \\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\Serial.println(dt);

}

void timeSync(unsigned long deltaT)
{
  unsigned long currTime = millis();
  long timeToDelay = deltaT - (currTime - timer);
  if (timeToDelay > 5000)
  {
    delay(timeToDelay / 1000);
    delay(timeToDelay % 1000);
  }
  else if (timeToDelay > 0)
  {
    delay(timeToDelay);
  }
  else
  {
      // timeToDelay is negative so we start immediately
  }
  timer = currTime + timeToDelay;
}



void MPU6050_configuracion(int r1, int r2) {
  Wire.beginTransmission(MPU6050_ADDRESS);
  Wire.write(r1);
  Wire.write(r2);
  Wire.endTransmission();
  delayMicroseconds(10);
}
```

CÓDIGO PARA EL ALGORITMO DE ESCRITURA LIBRE EN C

```
/*
 * MPU6050 Interfacing with PIC18F4550
 * http://www.electronicwings.com
 */

#include <pic18f4550.h>
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "USART_Header_File.h"      /* Include USART header file */
#include "I2C_Master_File.h"
#include "MPU6050_res_define.h"
#include "Configuration_header_file.h"

void MSdelay2(unsigned int);

void MPU6050_Init()
     /* Gyro initialization function */
{
     MSdelay(150);         /* Power up time >100ms */

    I2C_Start_Wait(0xD0);  /* Start with device write address */
     I2C_Write(CONFIG);    /* Write to sample rate register */
     I2C_Write(0x03); /* 1KHz sample rate */
     I2C_Stop();

    I2C_Start_Wait(0xD0);  /* Start with device write address */
     I2C_Write(ACCEL_CONFIG);    /* Write to sample rate register */
     I2C_Write(0x10); /* 1KHz sample rate */
     I2C_Stop();

    I2C_Start_Wait(0xD0);  /* Start with device write address */
     I2C_Write(PWR_MGMT_1);     /* Write to sample rate register */
     I2C_Write(0x00); /* 1KHz sample rate */
     I2C_Stop();

     I2C_Start_Wait(0xD0); /* Start with device write address */
     I2C_Write(SMPLRT_DIV);     /* Write to sample rate register */
     I2C_Write(0x07); /* 1KHz sample rate */
     I2C_Stop();

     /*I2C_Start_Wait(0xD0);
     I2C_Write(PWR_MGMT_1);     /* Write to power management
register */
     /*I2C_Write(0x01);    /* X axis gyroscope reference frequency
*/
     /*I2C_Stop();
```

```
      I2C_Start_Wait(0xD0);
      I2C_Write(CONFIG);    /* Write to Configuration register */
      /*I2C_Write(0x00);    /* Fs = 8KHz */
      /*I2C_Stop();

      I2C_Start_Wait(0xD0);
      I2C_Write(GYRO_CONFIG);     /* Write to Gyro configuration
register */
      /*I2C_Write(0x18);    /* Full scale range +/- 2000 degree/C */
      /*I2C_Stop();

      I2C_Start_Wait(0xD0);
      I2C_Write(INT_ENABLE);      /* Write to interrupt enable
register */
      /*I2C_Write(0x01);
      I2C_Stop();
     */

}

void MPU_Start_Loc()

{
      I2C_Start_Wait(0xD0);                                          /*
I2C start with device write address */
      I2C_Write(ACCEL_XOUT_H);                                       /*
Write start location address from where to read */
      I2C_Repeated_Start(0xD1);                                      /*
I2C start with device read address */
}




void main()

{
      char buffer[20];
      int Ax,Ay,Az,T,Gx,Gy,Gz;
      float Xa,Ya,Za,t,Xg,Yg,Zg;
     int aax3,aay3,aaz3;
     float angx,angy;
     float angxp=0,angyp=0;
     int uno=1, cero=0;
     OSCCON = 0x72;
     TRISCbits.TRISC0=1;
     I2C_Init();
      /* Initialize I2C */
      MPU6050_Init();
      /* Initialize Gyro */
```

```
      USART_Init(9600);
      /* Initialize USART with 9600 baud rate */


      while(1)
      {
            MSdelay2 (140);
         MPU_Start_Loc();                                                  /*
Read Gyro values continuously and send to terminal over USART */
            Ax = (((int)I2C_Read(0)<<8) | (int)I2C_Read(0));
            Ay = (((int)I2C_Read(0)<<8) | (int)I2C_Read(0));
            Az = (((int)I2C_Read(0)<<8) | (int)I2C_Read(0));
            T = (((int)I2C_Read(0)<<8) | (int)I2C_Read(0));
            Gx = (((int)I2C_Read(0)<<8) | (int)I2C_Read(0));
            Gy = (((int)I2C_Read(0)<<8) | (int)I2C_Read(0));
            Gz = (((int)I2C_Read(0)<<8) | (int)I2C_Read(1));
            I2C_Stop();


         aax3=atan(Ax/sqrt(Ay*Ay+Az*Az))*(180.0/3.14);
         aay3=atan(Ay/sqrt(Ax*Ax+Az*Az))*(180.0/3.14);
         aaz3=atan(Az/sqrt(Ax*Ax+Ay*Ay))*(180.0/3.14);

         angx=0.7*(angxp+(Gx/131)*0.001)+0.3*aax3;
         angy=0.7*(angyp+(Gy/131)*0.001)+0.3*aay3;

         angxp=angx;
         angyp=angy;

         if (TRISC0 == 1){
         sprintf(buffer,"%.2f,",angx);
         USART_SendString(buffer);
         sprintf(buffer,"%.2f,",angy);
         USART_SendString(buffer);
         sprintf(buffer,"%i \r \n",cero);
         USART_SendString(buffer);

         }

         if (TRISC0 == 0){
         sprintf(buffer,"%i,",cero);
         USART_SendString(buffer);
         sprintf(buffer,"%i,",cero);
         USART_SendString(buffer);
         sprintf(buffer,"%i \r \n",uno);
         USART_SendString(buffer);

         }


            //sprintf(buffer," Ax = %.2f g\t",Xa);           /* Take
values in buffer to send all parameters over USART */
```

```
            //USART_SendString(buffer);

            //sprintf(buffer," Ay = %.2f g\t",Ya);
            //USART_SendString(buffer);

            //sprintf(buffer," Az = %.2f g\t",Za);
            //USART_SendString(buffer);

            //sprintf(buffer," T = %.2f%cC\t",t,0xF8);           /*
0xF8 Ascii value of degree '°' on serial */
            //USART_SendString(buffer);

            //sprintf(buffer," Gx = %.2f%c/s\t",Xg,0xF8);
            //USART_SendString(buffer);

            //sprintf(buffer," Gy = %.2f%c/s\t",Yg,0xF8);
            //USART_SendString(buffer);

            //sprintf(buffer," Gz = %.2f%c/s\r\n",Zg,0xF8);
            //USART_SendString(buffer);
      }
}

void MSdelay2(unsigned int val)
{
 unsigned int i,j;
 for(i=0;i<val;i++)
     for(j=0;j<165;j++);         /*This count Provide delay of 1 ms
for 8MHz Frequency */
}
```

CÓDIGO PARA LA RECEPCIÓN DE LOS DATOS VÍA BLUETOOTH EN PYTHON

```
import serial
import time
import matplotlib.pyplot as plt
import numpy as np



def Init():
      port="COM9"
      bluetooth=serial.Serial(port, 9600)
      print("Connected")
      bluetooth.flushInput()
      return bluetooth

def Adq(myBlue):
      bluetooth=myBlue
      input_data=bluetooth.readline()
      print(input_data.decode())
      #time.sleep(0.01)
      new_data=input_data.decode().split(",")
      x=float(new_data[0])
      y=float(new_data[1])
      #z=int(new_data[2])
      print(x)
      print(y)
      #print(z)
      return x, y
```

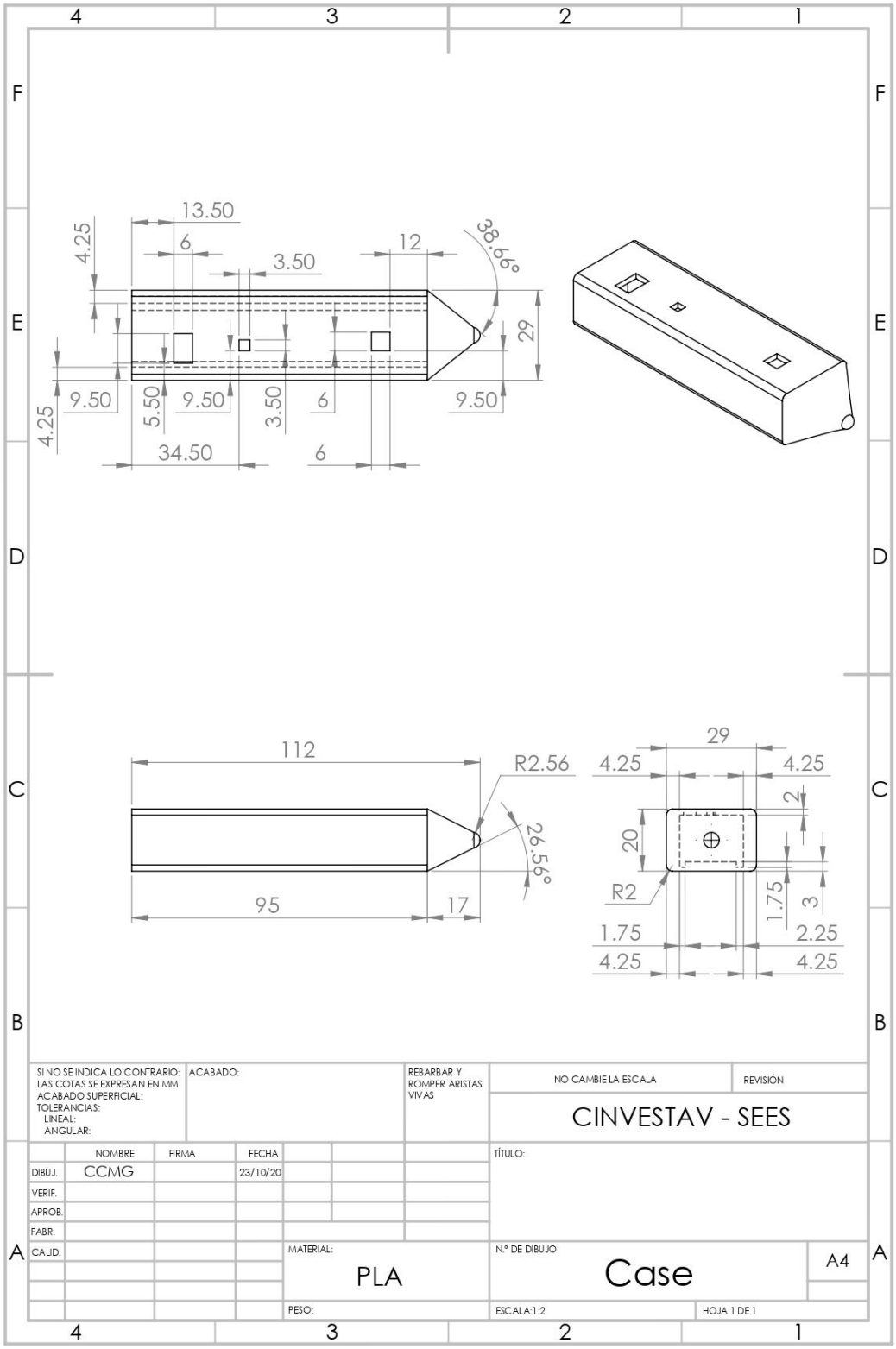
CÓDIGO PARA LA GRAFICACIÓN EN PYTHON

```
import serial
import time
from matplotlib import pyplot
from matplotlib.animation import FuncAnimation
import numpy as np
import Bluetoo
from datetime import datetime

blue=Bluetoo.Init()

x_data=[]
y_data=[]
xoff=0.0
yoff=0.0

figure = pyplot.figure()
line, = pyplot.plot(x_data, y_data, 'o')

def update(frame):
    global xoff, yoff
    xs,ys=Bluetoo.Adq(blue)
    xs=xs+xoff
    ys=ys+yoff
    x_data.append(-xs)
    y_data.append(ys)
    xoff=xs
    yoff=ys
    line.set_data(x_data, y_data)
    figure.gca().relim()
    figure.gca().autoscale_view()
    return line,

animation = FuncAnimation(figure, update, interval=5)

pyplot.show()

blue.close()
print('Disconnected...')
```

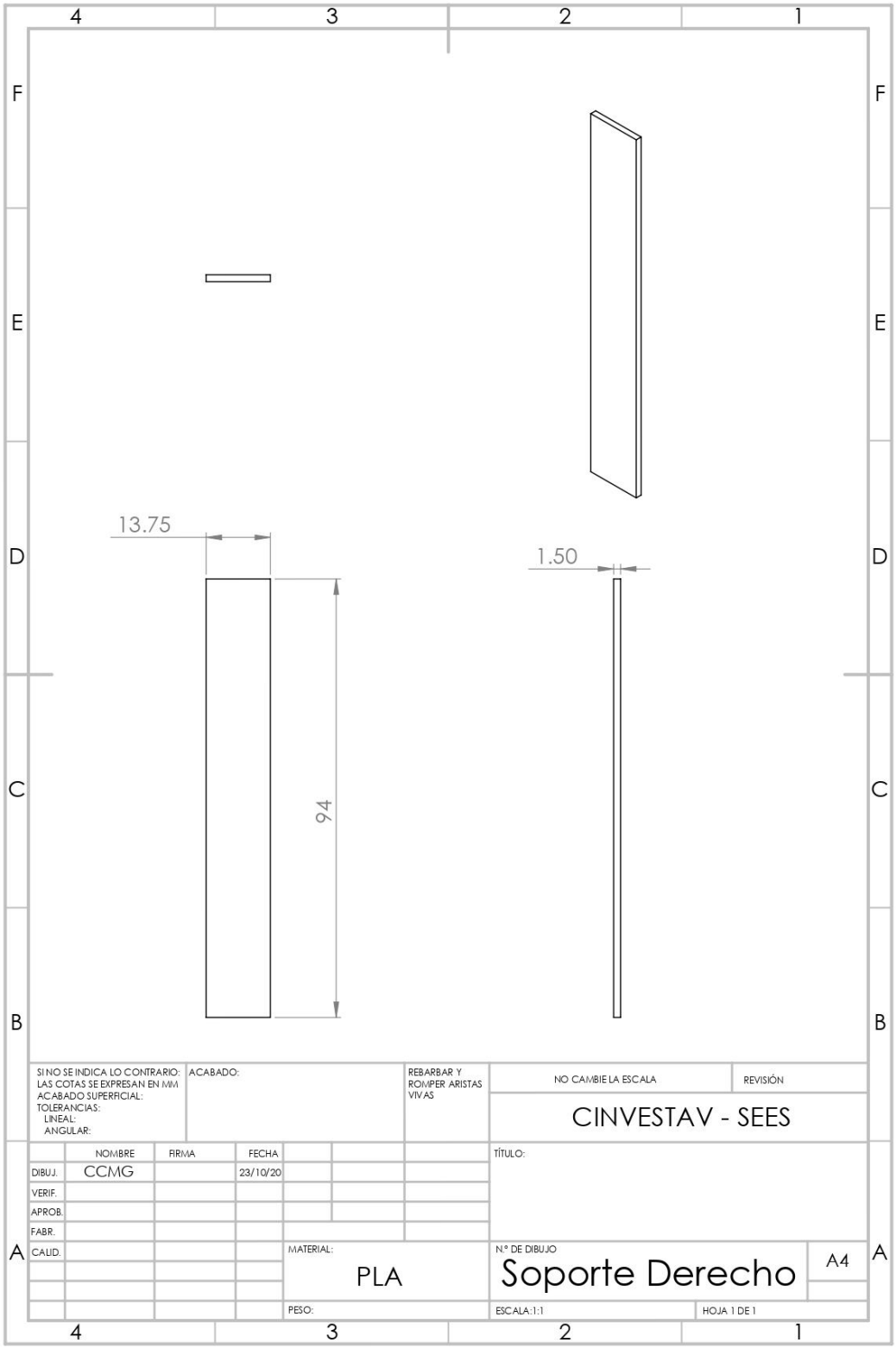
CÓDIGO FUENTE DE LA INTERFAZ GRÁFICA DE USUARIO EN QT CREATOR PARA PYTHON

```
#include "hypen.h"
#include "ui_hypen.h"

Hypen::Hypen(QWidget *parent)
    : QWidget(parent)
    , ui(new Ui::Hypen)
{
    ui->setupUi(this);
}

Hypen::~Hypen()
{
    delete ui;
}


void Hypen::on_pushButton_clicked()
{
    ui->stackedWidget->setCurrentIndex(1);
}

void Hypen::on_buttonBox_accepted()
{

}

void Hypen::on_buttonBox_rejected()
{
   ui->stackedWidget->setCurrentIndex(0);
}
```

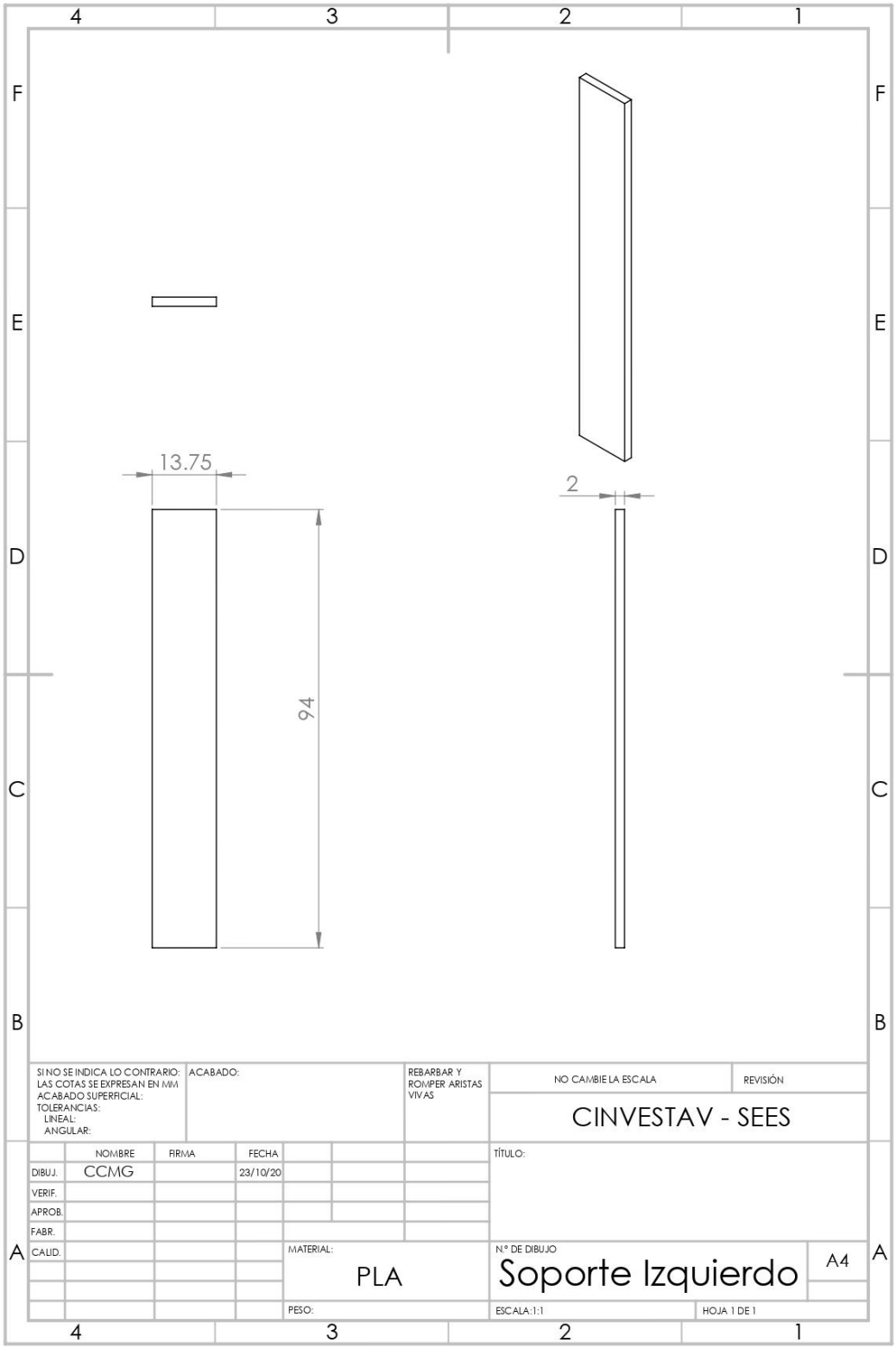
VISTAS FRONTAL, SUPERIOR Y LATERAL DEL CUERPO DE LA CUBIERTA DEL DISPOSITIVO



VISTAS FRONTAL, SUPERIOR Y LATERAL DE LA BARRA DE SOPORTE DERECHA DE LA CUBIERTA DEL DISPOSITIVO



VISTAS FRONTAL, SUPERIOR Y LATERAL DE LA BARRA DE SOPORTE IZQUIERDA DEL CUERPO DE LA CUBIERTA DEL DISPOSITIVO



VISTAS FRONTAL, SUPERIOR Y LATERAL DEL TAPÓN DE LA CUBIERTA DEL DISPOSITIVO