CARACTERIZACIÓN DEL SENSOR IMU (INERTIAL MEASUREMENT UNIT) BNO055

AUTORES:
JULIANA ANDREA RAMOS CIFUENTES
JULIÁN SANTIAGO CASTILLO CORREDOR

Universidad Antonio Nariño
Facultad de Ingeniería Mecánica, Electrónica y
Biomédica
Villavicencio, Colombia
2020

CARACTERIZACIÓN DEL SENSOR IMU (INERTIAL MEASUREMENT UNIT) BNO055

JULIANA ANDREA RAMOS CIFUENTES
JULIÁN SANTIAGO CASTILLO CORREDOR

Proyecto de grado presentado como requisito para optar al título de:

Ingeniero Electrónico

Director:

PhD WILSON HERNANDEZ

Universidad Antonio Nariño

Facultad de Ingeniería Mecánica, Electrónica y Biomédica

Villavicencio, Colombia

2020

Nota de aceptación

_____
_____
_____
_____
_____

_____
Firma del presidente del jurado

_____
Firma del Jurado

_____
Firma del Jurado

Villavicencio, 08- 06-2020.

Agradecimientos

En primera estancia agradezco a Dios y a nuestros padres por ser los principales promotores de nuestro sueño, por anhelar siempre lo mejor para nuestras vidas, de igual modo a todas las personas que de una u otra manera nos aportaron ideas para lograr que este proyecto se hiciera realidad.

Agradezco también al Doctor Wilson Hernández, quien es el director de nuestro proyecto de grado, siempre nos brindó su asesoría en el desarrollo y ejecución del proyecto planteado. Del mismo modo a los docentes de la universidad Antonio Nariño sede Villavicencio, que nos brindaron de la mejor manera posible sus conocimientos y experiencias.

También a todos nuestros compañeros, que nos acompañaron en el trascurso de la carrera, con quienes compartimos nuevas experiencias y siempre buscábamos la manera de superar en grupo las dificultades que se nos presentaran en los trabajos.

Ya, por último, queremos agradecerle a la Sra. Beatriz Montañez quien es la auxiliar de laboratorios, por prestarnos los laboratorios e instrumentos necesarios siempre, permitiéndonos desarrollar nuestro proyecto de grado.

Resumen

La finalidad de este proyecto es la realizar la caracterización de la unidad de medida inercial (IMU) implementando el método de Allan variance, el cual se basa en análisis estadísticos de las medidas obtenidas con el sensor IMU.

Una IMU es un dispositivo compuesto por un acelerómetro, el cual mide los cambios de fuerza producidos por el movimiento del dispositivo; un giroscopio, que se encarga de registrar la variación de la posición de los ejes representamos por la IMU y un magnetómetro, que interactúa con el campo magnético terrestre.
Los datos que proporcionara la IMU serán leídos por medio de una placa raspberry pi, estos son de suma importancia para la navegación de vehículos terrestres coches, camiones y motocicletas de gama alta), aéreos (aviones y helicópteros), marítimos y espaciales (exploración extraterrestre), permitiendo que se orienten y desplacen de la forma más segura y precisa posible.
De la misma manera, el programa que se desarrolla para la gestión del sensor IMU proporciona una herramienta muy útil cuyo objetivo, no es solo la obtención, gestión y manipulación de los datos ofrecidos por la IMU, si no permitirá caracterizarlo aplicando el método estadístico ALLAN VARIANCE.
Para el desarrollo de dicho programa se ha recurrido a uno de los lenguajes de programación más utilizados en este tiempo, como es Python.

Palabras clave: (IMU, Allan variance, python, raspberry pi, giroscopio, magnetómetro, acelerómetro)

Abstract

The purpose of this project is to carry out the characterization of the inertial measurement unit (IMU) by implementing the Allan variance method, which is based on statistical analysis of the measurements obtained with the IMU sensor.

An IMU is a device made up of an accelerometer, which measures the changes in force produced by the movement of the device; a gyroscope, which is responsible for recording the variation in the position of the axes represented by the IMU and a magnetometer, which interacts with the Earth's magnetic field.

The data provided by the IMU will be read by means of a raspberry pi board, these are of utmost importance for the navigation of high-end land vehicles (cars, trucks and motorcycles), air (aircraft and helicopters), maritime and space (extraterrestrial exploration), allowing them to orient and move as safely and accurately as possible.

In the same way, the program that is developed for the management of the IMU sensor provides a very useful tool whose objective is not only to obtain, manage and manipulate the data offered by the IMU, but will also allow it to be characterized, applying the ALLAN VARIANCE statistical method.

For the development of this program, one of the most used programming languages at this time, such as Python, has been used.

Keywords: (IMU, Allan variance, python, raspberry pi, gyroscope, magnetometer, accelerometer)

Contenido

Lista de figuras

Pág.

Lista de tablas

Pág.

Introducción

Desde que los seres humanos iniciaron la exploración por nuevos lugares, la orientación ha sido un requisito indispensable para desplazarse de la manera más segura posible. En los últimos años se ha producido un gran avance en cuanto al desarrollo de sistemas o dispositivos que permitan la navegación en vehículos terrestres, aéreos, marítimos y espaciales. Los sensores que son utilizados para llevar esta labor suelen presentar errores en los datos que proporcionan, esto se debe al complejo movimiento del vehículo o a la cantidad de información que maneja, dado que algunos trabajan con más de una variable. Es por esta acumulación de errores que se decide realizar la caracterización del sensor utilizando una herramienta matemática, en el caso de este proyecto se utilizó el método de varianza de Allan y se realizaran ajustes a la ecuación utilizando el algoritmo Levenberg-Marquardt. Dando a conocer sobre la necesidad de caracterizar un sensor, se estudiaron diferentes proyectos previos lo cuales relacionan los diferentes aspectos que se utilizaron para la implementación y desarrollo del proyecto que se resaltan en los siguientes párrafos.

Este trabajo integral de grado está basado en el informe de Jurado et al [1], donde presentan el desarrollo de una metodología autónoma, que se basa en la regresión para el análisis de Allan variance, utilizando una ecuación base la cual se ajusta con algoritmo Levenberg-Marquardt para poder obtener 5 parámetros que se requieren para la caracterización del sensor (Error de cuantización (□□),caminata aleatoria angular / velocidad (□□□), inestabilidad de sesgo (□□), aceleración / caminata aleatoria de velocidad angular (□□□□), rampa de velocidad (□□□)), ellos realizaron la toma de muestras en un tiempo de 6 horas para observar la variación que se generaba entre las muestras. Para la recolección de los datos arrojados por la IMU se tuvo presente el trabajo de Sergio Martín [2] quien trabajó con este sensor y generó como proyecto de grado la estimación de ángulos de inclinación donde tenía una plataforma que se encontraba sobre dos motores, esto con el fin de hacer que

la plataforma se estabilizará en un ángulo ya referenciado; para llevar a cabo este proyecto se utilizó el sensor IMU para realizar la estimación de la inclinación de la plataforma. Por otra parte [3], sacan un artículo en Coruña, donde explican el diseño para la adquisición, visualización y almacenamiento de los datos trasmitidos por el sensor inercial, también realizan una aplicación la cual es ejecutable en una Raspberry pi, donde esta tiene una pantalla para visualizar los datos obtenidos y posteriormente ser almacenados, la conexión entre el sensor IMU y la Raspberry es por medio de un puerto serie (USB-TTY).

En cuanto a la eliminación del ruido en la IMU. [4] quienes por medio de un globo en el que se transportaba una IMU se percataron que este sensor con el paso del tiempo presentaba acumulación de errores. Utilizando Allan variance y el análisis de densidad espectral para generar una comparación de los resultados, identificaron los ruidos que presenta el sensor, observaron que el ruido que se generaba era de naturaleza gaussiana blanca, el cual por medio de un filtro de Kalman pudieron eliminar en tiempo real. De igual manera [5] se centraron en caracterizar la parte giroscópica del sensor MEMS IMU mediante el análisis de tres diferentes parámetros de error los cuales son: factor de escala, estabilidad de sesgo, caminata aleatoria angular. También utilizaron el análisis de varianza de Allan como aporte primordial para los esquemas de los resultados.

Teniendo en cuenta los avances tecnológicos presentados, se valida la importancia de la caracterización en los sensores. El método de Allan variance consiste en perfeccionar el desempeño del sensor, por esto este proyecto se guía del trabajo de Jurado et al [1], donde lo realizan de manera teórica, en este caso se hará con datos en tiempo real tomados por la IMU.
Consecuentemente, este este proyecto presenta los resultados obtenidos y un documento manual desde la caracterización de sensor IMU utilizando el método de Allan variance por medio de un programa realizado en python, para generar la posibilidad de utilizar el manual guía para realizar la caracterización de cualquier

sensor IMU, facilitando este proceso, desde el planteamiento del proyecto se propone; realizar la caracterización y documentación del sensor IMU perteneciente al robot Génesis basándonos en el trabajo realizado por Jurado et al [1], aplicando el método de varianza de Allan y chi-cuadrado a los datos obtenidos por el sensor, y determinar su ecuación característica facilitando así, su uso en proyectos futuros.

Con el fin de generar un documento manual que permita el caracterizar un sensor IMU de una manera más eficaz, se utilizó el método de Allan variance utilizando como guía el documento de Jurado et al [1], se utilizó el sensor IMU BNO055 que posee 9 grados de libertad y permite leer los datos arrojados por puertos I2C (inter integrated circuits) o por medio de UART (Universal Asynchronous Receiver-Transmitter). El manual guía permitirá realizar la caracterización de sensores inerciales, permitiendo la realización de diversas tareas pensadas a futuro por parte de la comunidad de la UAN.

En la sección 1 se encontrarán las definiciones de los conceptos claves del sensor y conjuntamente los conceptos del método de Allan variance, sus parámetros, el algoritmo Levenberg-Marquardt y acerca de su aplicación, para comprender y entender el tema el desarrollo de este trabajo; En la sección 2 se explican las fases que se desarrollaron a lo largo del proyecto, también cuenta con la explicación del proceso en cada fase, marcando los pasos que se necesitan para el desarrollo de este proyecto; En la sección 3 se presentan los resultados que se obtuvieron al realizar diversas pruebas con el sensor y de igual manera se realiza el análisis correspondiente a cada uno de estos; En la sección 4 se presentan las conclusiones del trabajo, así como también posibles mejoras y sugerencias con el objetivo de presentar una tendencia de mejora continua en este campo explorado en el presente documento.

Objetivos

Objetivo general

Realizar la caracterización por el método de Allan variance y la documentación del procedimiento y los resultados para el sensor IMU BNO055.

Objetivos específicos

•Realizar tomas de datos y archivar los resultados en un documento para ser analizados.

•Implementar el método de Allan Variance para la identificación del error que afecta la IMU.

• Documentar los datos y los métodos.

1. Marco teórico

1.1 Unidad de medición inercial

Una unidad de medición inercial o IMU (del inglés Inertial Measurement Unit) es un dispositivo electrónico con el objetivo de medir los cambios que se presentan en la velocidad, rotación y fuerza gravitacional de un aparato de forma autónoma, la IMU suele estar compuesta por un conjunto de acelerómetros, giroscopios y un magnetómetro, que obtienen datos de uno o más ejes ortogonales [6].

Figura 1: Sensor IMU-10 DOF

Fuente: [7]

1.2 Allan variance

La varianza de Allan es una medida de la estabilidad de frecuencia en relojes, osciladores y amplificadores, llamada así por David Allan. Esta se desarrolló inicialmente para el análisis de las fuentes de error que se encontraban en los relojes atómicos, con el tiempo se descubrió se podía utilizar para identificar las fuentes de error en acelerómetros y giroscopios utilizando el método de pendiente, para analizar las mediciones de varianza de Allan, esta se expresa matemáticamente como $\sigma_{\tau}^{2}(\tau)$ [8].

En general, el Allan variance de una señal de tiempo continuo $y(t)$, es una función de una cantidad llamada tiempo promedio (τ) [9] ,y está dada por el seguimiento:

$$\sigma_{\tau}^{2}(\tau) = \frac{1}{2(N-2n)} \sum_{k=1}^{N-2n} \left[\bar{\theta}_{k+1}(n) - \bar{\theta}_{k}(n)\right]^{2} \qquad \textbf{(1)}$$

Donde (**N**) es el número total de muestras, (**n**) está dada por $n = \frac{\tau}{\Delta\tau}$ y ($\Delta\tau$) es el periodo de muestreo.

A partir de graficar las muestras aplicando Allan variance, se identifican sistemáticamente cinco fuentes de error que afectan al acelerómetro y giroscopio utilizando las pendientes de la relación entre la densidad espectral de potencia (PSD) de una fuente de error y su correspondiente fórmula de varianza [1].

1.2.1 Error de cuantización (Q_q)

La cuantización se define como el acto de muestrear una señal analógica en niveles discretos de tamaño (Δ) durante el proceso de conversión de analógico-digital. Los errores (diferencias entre la señal analógica y la señal digitalizada) causados por dicha cuantización pueden caracterizarse como ruido aditivo [1], que se distribuye uniformemente entre:

$$\sigma_{Q}(\tau) = Q_{q}\sqrt{3\tau^{-1}} \qquad \textbf{(2)}$$

Donde $\sigma_{Q}(\tau)$ es expresión de Allan, (Q_q) es el error de cuantización y (τ) es una función de una cantidad llamada tiempo promedio.

1.2.2 Caminata aleatoria de ángulo / velocidad (N_{RW})

Como su nombre lo indica, la caminata aleatoria de ángulo o velocidad es un proceso de caminata aleatoria observado en la salida de señal de ángulo o velocidad de un sensor inercial [1].

$$\sigma_{R}(\tau) = N_{RW}\sqrt{\tau^{-1}/3} \qquad \textbf{(3)}$$

1.2.3 Inestabilidad de sesgo (□□)

La inestabilidad de sesgo se refiere a la tendencia del sesgo constante de un sensor inercial a cambiar o derivarse durante el uso. Este proceso a menudo se aproxima mediante un proceso de Gauss-Markov de primer orden [1]. Al Seguir el proceso del método de pendiente obtenemos:

$$\square_{\square}(\square) = \square_{\square} \sqrt{\frac{\square\square\square(\square)}{\square}}\,\square^{\square} \qquad \textbf{(4)}$$

1.2.4 Aceleración / velocidad angular caminata aleatoria (□□□□)

Se refiere a un proceso de caminata aleatoria observado en la señal de velocidad del sensor inercial (aceleración o velocidad angular), donde la tasa de caminata aleatoria surge de la integración del ruido blanco [1].

$$\square_{\square}(\square) = \square \qquad \textbf{(5)}$$

1.2.5 Rampa de velocidad (□□□)

Finalmente, la rampa de velocidad se refiere determina, el aumento lineal generalmente a largo plazo de la salida de señal de velocidad del sensor inercial [1].

$$\square_{\square}(\square) = \square \qquad \textbf{(6)}$$

1.3 Método de mínimos cuadrados

El método de mínimos cuadrados se basa en la técnica de análisis numérico enmarcado dentro de la optimización matemática, donde dado un conjunto de pares ordenados, es decir, una variable independiente, una variable dependiente y una familia de funciones, se pretende hallar la función continua, dentro de dicha familia,

donde se aproxime a los datos [10] .

La fórmula de la pendiente se encuentra expresada de la siguiente manera:

$$\square = \square + \square\square \qquad \textbf{(7)}$$

Donde (Y) es el valor proyectado, (a) es el punto en donde la recta corta en el eje, (b) es la pendiente de la recta la tendencia, (x) es cualquier valor de tiempo seleccionado.

Este método es el más utilizado para obtener un ajuste lineal a una serie de datos, donde busca minimizar la suma de cuadrados de las diferencias ordenadas (llamadas residuos) entre los puntos generados por la función y los correspondientes datos [11]. Para encontrar sus variables descocidas a y b se resuelve las siguientes formulas:

$$\square = \frac{(\sum\square\sum\square^2)-(\sum\square\sum\square\square)}{(\square\sum\square)^2-(\sum\square)^2} \qquad \textbf{(8)}$$

$$\square = \frac{(\square\sum\square\square)-(\sum\square\sum\square)(\square\sum\square}{)-(\sum\square)^2} \qquad \textbf{(9)}$$

$\sum\square$ = La sumatoria de los datos dependiente

$\sum\square$ = La sumatoria de los datos independiente

$\sum\square\square$ = La sumatoria de la multiplicación entre los datos dependiente e independiente

$\sum\square^2$= la sumatoria de cada dato independiente al cuadrado

$(\sum\square)^2$ = La sumatoria de los datos independientes al cuadrado

1.4 Método de la pendiente

El método de la pendiente descubre cada coeficiente de intensidad de ruido individualmente, para esto se crea un método autónomo para estimar la fuerza del ruido denominado método de regresión autónoma para la varianza de Allan (ARMAV), este método no solo tiene un rendimiento comparable en términos de precisión de estimación, sino que también es completamente autónomo, estable en condiciones de datos limitadas y adecuado para la caracterización de IMU; la relación combinada entre la varianza total de Allan [1], y las contribuciones de cada una de las fuentes de error están dadas por lo siguiente:

$$\sigma^2(\tau) = \left[\left(\frac{Q}{\tau}\sqrt{3}\tau^{-1}\right)^2 + \left(\frac{B}{\tau}\sqrt{2\log(2)}\right)^2 + \left(\frac{K}{\tau}\sqrt{\frac{\tau}{3}}\right)^2 + \left(\frac{R}{\tau}\sqrt{\frac{\tau}{2}}\right)^2\right] \quad \textbf{(10)}$$

Donde (σ^2) es expresión de Allan, (Q) es el parámetro de cuantización, (N) es el parámetro de caminata aleatoria de ángulo / velocidad, (B) es el parámetro de Inestabilidad de sesgo, aleatoria (K) es el parámetro de Aceleración / velocidad angular caminata, (R) es el parámetro de Rampa de velocidad y (τ) es una función de una cantidad llamada tiempo promedio [1].

1.5 Algoritmo de Levenberg-Marquardt

También conocido como el método de mínimos cuadrados amortiguados, es altamente utilizado en el ambiente de la optimización, ya que brinda la solución al problema de mínimos cuadrados lineales y no lineales, estos problemas de minimización surgen especialmente en el ajuste de curvas de mínimos cuadrados [12]. El algoritmo Levenberg-Marquardt calcula el Jacobiano del modelo teniendo en cuenta los parámetros, y utilizando este busca un mínimo local. El Jacobiano se necesita en cada iteración para poder calcular la dirección hacia el mínimo local; debido a que no es bastante difícil el cálculo analítico el Jacobiano es estimado a través de un algoritmo [13].

1.6 Tarjetas de procesamiento

Son el núcleo del sistema, tanto hardware y como de software, estas proporcionan la funcionalidad de una computadora de escritorio, desde su aparición se han convertido en una fuerza predominante en el mundo del desarrollo, usadas en gran cantidad de proyectos de código abierto. Se encarga de integrar los diferentes dispositivos con los que cuenta el proyecto requerido. Además, contiene un Sistema Operativo y un lenguaje de programación, con el cual se logra una comunicación a nivel de software con elementos internos y externos [14].

1.7 Lenguaje de programación Python

Python es un lenguaje de programación poderoso y de fácil aprendizaje, capaz de analizar y ejecutar otros programas. Cuenta con estructuras de datos eficientes y de alto nivel con un enfoque simple pero efectivo para la programación orientada a objetos. Como una de sus características es el de ser un lenguaje de programación multiparadigma, permitiendo crear programas usando diferentes estilos de programación dependiendo la conveniencia, además cuenta con extensas bibliotecas estándar que están a libre disposición, como también módulos libres de Python de terceros, programas, herramientas, y documentación adicional, todo esto hacen de éste un lenguaje ideal para scripting y desarrollo rápido de aplicaciones en diversas áreas y sobre la mayoría delas plataformas [15].

2. Metodología

El enfoque principal del proyecto propuesto se centra en realizar la caracterización del sensor IMU utilizando el método de Allan variance y el método de Levenberg-Marquardt. La metodología planteada se divide en las siguientes fases:

2.1 Fase 1: Reconocimiento detallado del sensor IMU

Se realizó el reconocimiento del sensor realizando consultas constantemente, en este proyecto se utilizó el sensor del robot Genesis [16].

2.1.1 IMU BNO055

La Unidad de Medida Inercial (IMU) que se utilizó en este proyecto es la placa BNO055 de Adafruit Figura 1, la cual posee 9 grados de libertad y permite leer los datos arrojados por puestos I2C (inter integrated circuits) o por medio de UART (Universal Asynchronous Receiver-Transmitter), esta placa posee dimensiones de 20mm de ancho, 27mm de largo y 4mm de grosor, además posee un regulador de tensión de 3.3V para su funcionamiento [6].

Figura 2: IMU BNO0055

Fuente: Adafruit [6]

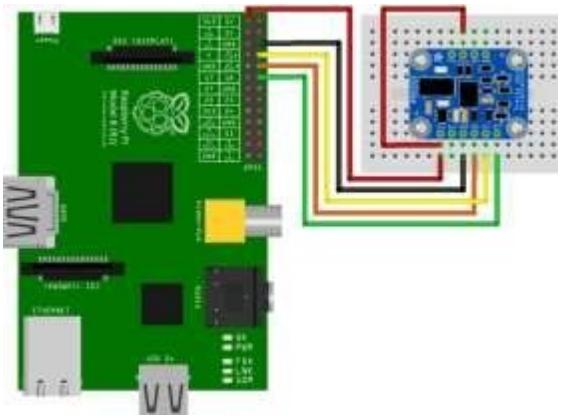
El dispositivo BNO055 puede generar los siguientes datos:

- Orientación absoluta. (Vector de Euler, 100 Hz) Datos de orientación de tres ejes basados en una esfera de 360 °.
- Orientación absoluta. (Quaternion, 100Hz).
- Vector de velocidad angular (100 Hz) Tres ejes de 'velocidad de rotación' en rad / s.
- Vector de aceleración (100Hz) Tres ejes de aceleración (gravedad + movimiento lineal) en m / s ^ 2.
- Vector de fuerza de campo magnético (20 Hz) Tres ejes de detección de campo magnético en micro Tesla (uT).
- Linear Acceleration Vector (100Hz) Tres ejes de datos de aceleración lineal (aceleración menos gravedad) en m / s ^ 2.
- Gravity Vector (100Hz) Tres ejes de aceleración gravitacional (menos cualquier movimiento) en m / s ^ 2.
- Temperatura (1Hz) Temperatura ambiente en grados centígrados.

2.2 Fase 2: Realizar las conexiones entre la raspberry y el sensor.

La conexión del sensor BNO055 se realiza por medio del modo UART en serie siguiendo las especificaciones, dado que este sensor puede presentar problemas cuando se utiliza mediante la su interfaz I2C el cual causa conflictos entre el reloj interno de la IMU y el de la raspberry Pi generando un retraso en la comunicación de estos. Antes de realizar la conexión entre los dos dispositivos, es importante deshabilitar el uso del núcleo de serial RPI; En la figura 2 se observa la conexión entre el sensor inercial y la raspberry Pi.

Figura 3: Conexión IMU BNO055 y Raspberry pi 2B

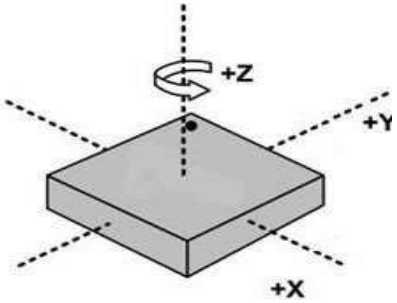


Fuente: Adafruit [6]

Adafruit cuenta con un paquete necesario para la respectiva instalación y configuración del sensor IMUBNO055, este puede descargarse de https://learn.adafruit.com/bno055-absolute- orientation-sensor-with-raspberry-pi-and-beaglebone-black/software,y se puede seguir en el anexo 2.

Cuando los requerimientos que se necesitan para el sensor IMU estén instalados, se procede hacer uso de la librería concedida por Adafruit para poder adquirir los datos de los ángulos de orientación presentados por el sensor, en este trabajo se utilizó como regencia el eje z y se tomaron los ángulos x, y que se pueden observar en la figura 3, debido a la necesidad de conocer únicamente el ángulo de desorientación presentado por robot durante su trayectoria.

Figura 4: Representación ejes de lectura sensor IMU



Fuente: Prometec [16]

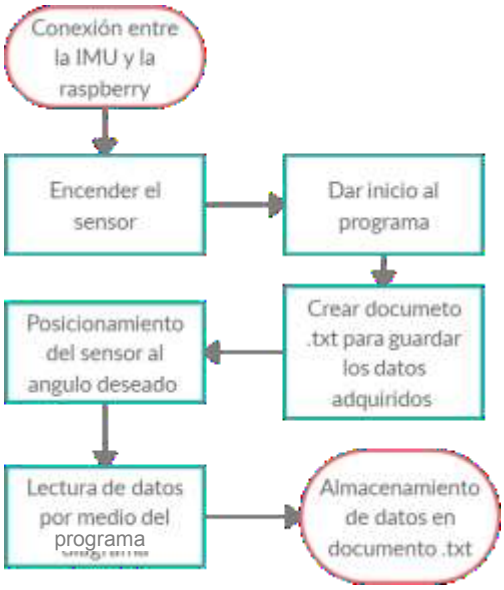
2.3 Fase 3: Toma de muestras

Después de haber realizado la comunicación entre la placa raspberry pi y el sensor IMU, se procede a realizar la toma de datos, estas se realizaron en diferentes rangos de tiempo y ángulos en z.

Para capturar los datos se realizó un programa en python instalando el paquete de Adafruit, se inició con una prueba, donde se tomaron 7200 datos por dos horas segundo a segundo, esto basándonos en el trabajo de Jurado et al [1], donde para la toma de datos usan un tiempo de 6 horas, así que para empezar se tomaron solo dos horas.

En la figura 4 se observa el diagrama de flujo que describe el proceso de adquisición de datos.

Figura 5: Representación el proceso de adquisición de datos.



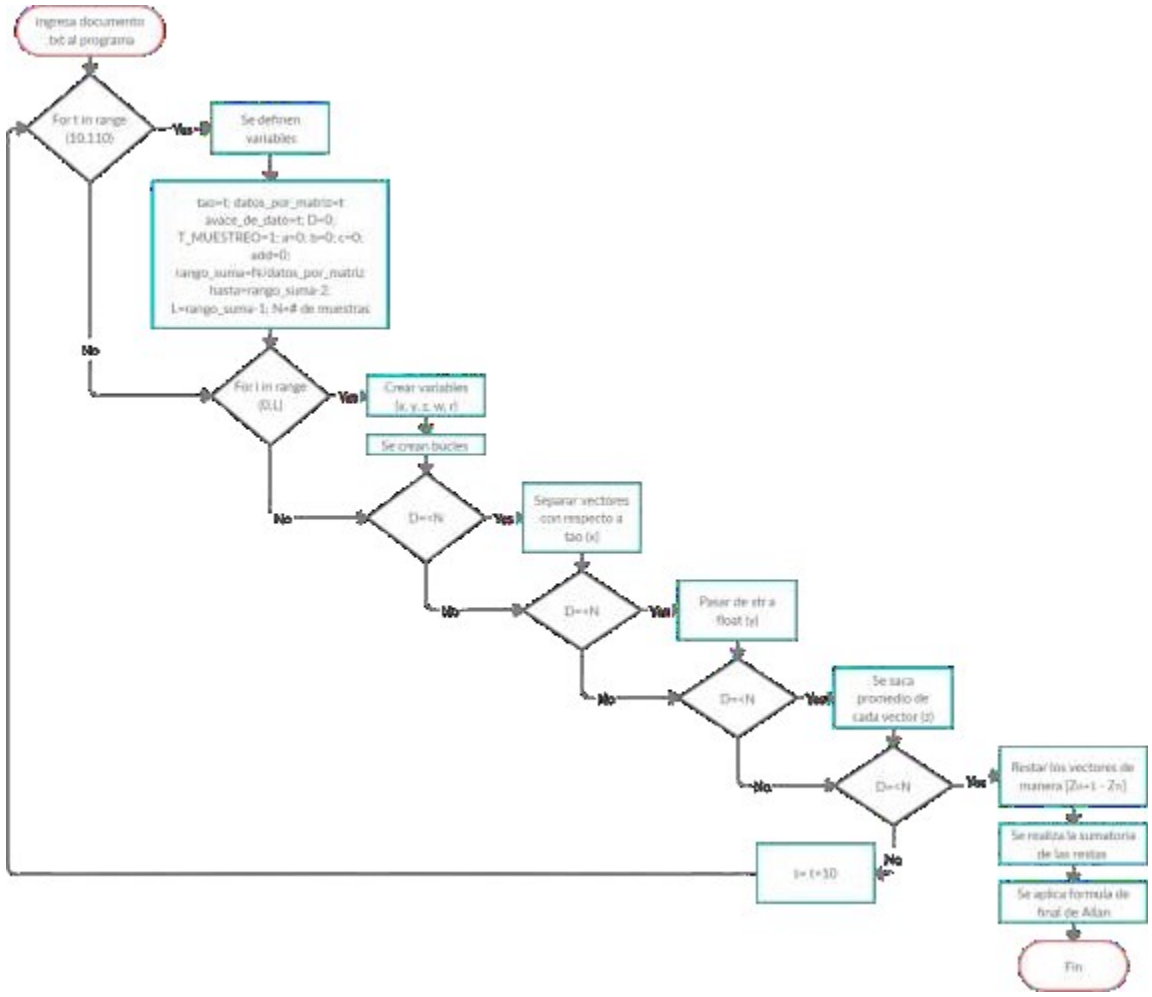
Fuente: imagen propia

2.4 Fase 4: Aplicación de Allan variance y ajuste de ecuación

2.4.1 Allan variance

Luego de haber guardado los datos en un documento .txt se prosiguió aplicar el método de Allan variance, para esto se realizó un programa en python, el cual se explica en la figura 5 donde se observa el proceso por medio de un diagrama de flujo.

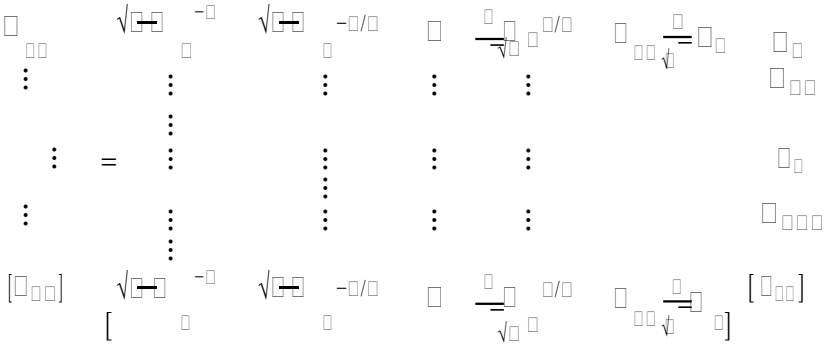
Figura 6: Representación de la aplicación de Allan variance**.**



Fuente: Imagen propia

2.4.2 Regresión lineal

Luego de obtenidos los 7200 puntos, se aplica primero una aproximación lineal, donde se resuelve el siguiente modelo lineal



(11)

Para hallar un valor aproximado para cada parámetro, y así encontrar una suposición inicial para nuestro algoritmo de regresión no lineal, se realiza el procedimiento de mínimos cuadrados, el cual es un método de optimización matemática, que dada una sucesión de medidas, encuentra los parámetros de un modelo de función que mejor ajuste a las medidas conforme al criterio de minimizar la suma de cuadrados de los residuos entre los puntos generados por la función y los datos observados.

Para el desarrollo de la ecuación (11), se implementó la librería en el programa scipy.optimize.curve_fit, donde utiliza el método de mínimos cuadrados no lineales para el ajuste de una función a los datos, la cual se utiliza para aplicar la regresión lineal y encontrar los valores próximos a los 5 parámetros.

2.4.3 algoritmo Levenberg-Marquardt

Ya obtenidos los valores aproximados de los 5 parámetros ($\square_{\square}, \square_{\square\square}, \square_{\square}, \square_{\square\square\square}, \square_{\square\square}$), se remplazan estos valores en la ecuación (10), aplicando el algoritmo Levenberg-Marquardt para poder así obtener los valores de los 5 parámetros requeridos.

3. Resultados

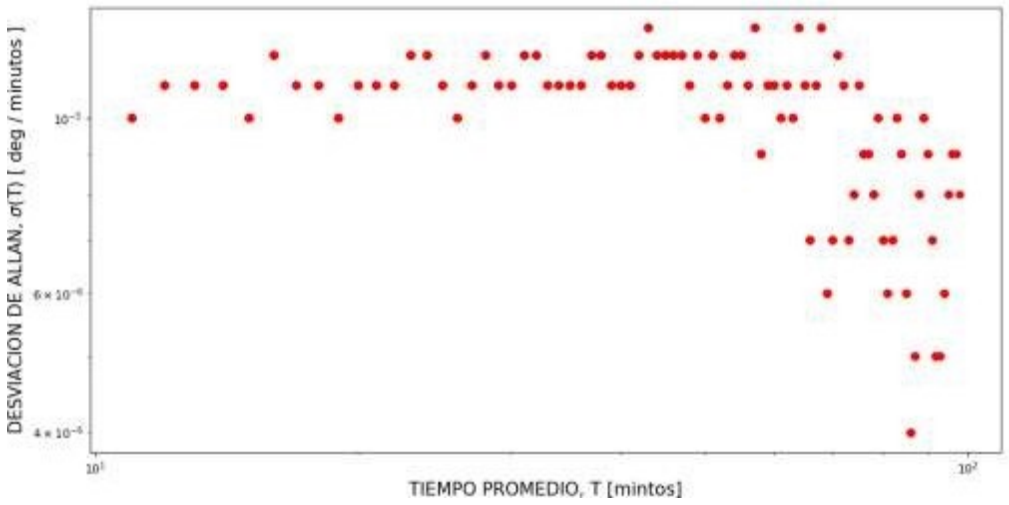
Los resultados obtenidos del sensor luego de implementar el programa de python se obtuvieron al realizar distintas pruebas de tiempo, en los cuales se tienen cuatro experimentos para confirmar la efectividad del programa, para el primer experimento se tomó el sensor y se posicionó a 60° este se dejó por media hora para cada ángulo, posteriormente para el segundo y tercer experimento se posiciono el sensor en 90° pero esta vez se dejó por una hora y por hora y media siguiendo las recomendaciones de nuestro director, para ver su funcionamiento, para el ultimo experimento se dejó el sensor en 90° pero esta vez con un tiempo de dos horas, aun así la curva de Allan variance no se puede observar perfectamente dado que el sensor IMU posee una curva característica que se presenta cuando el tiempo de uso es mayor a 6 horas, este tiempo fue estimado en el trabajo de [1] donde se recomienda dejar al menos 6 horas para que dicha curva aparezca, esto no se pudo realizar debido a que se presento un fallo en el sensor BNO055 y por cuestiones de la cuarentena debido al COVID-19. Los datos obtenidos durante el tiempo de recolección de muestras fueron almacenados en archivos con extensión .txt para posteriormente ser ingresados y analizados en python.

3.1 Experimento 1

En el primer experimento se decidió posicionar el sensor en 60° en un tiempo de media hora (los datos eran arrojados por segundo), esto para ver el comportamiento del sensor y de la fórmula de Allan en este lapso de tiempo.

Se realizo la toma de muestras, se ingresaron los datos al programa en python y se prosiguió a sacar punto por punto aplicando el método de Allan variance, en la Figura 6 se observan los puntos de dados por la fórmula de Allan y el transcurso del tiempo.

Figura 7: Grafica de puntos dados por el metodo de Allan

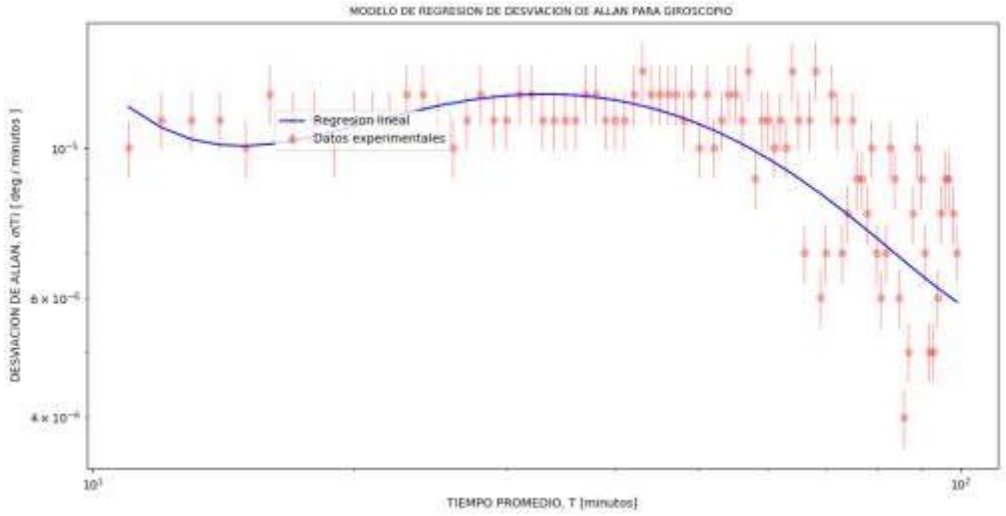


Fuente: Fuente propia

Luego de aplicado el método de Allan variance, se prosigue aplicar el método Mínimos cuadrados no lineales que se utiliza para acomodar un conjunto de (N) muestras con un modelo no lineal en (∅) parámetros desconocidos (N > ∅). La función de este método es aproximar el modelo por uno lineal y estimar un valor aproximado para los parámetros por interacciones continuas [12].

Para realizar el ajuste de a la ecuación (11) se utiliza la librería Curve_fit [17], la cual utiliza el método de mínimos cuadrados no lineales, para hallar el valor aproximado a cada parámetro requerido, para mayor información sobre esta librería consultar el **"DOCUMENTO MANUAL PARA LA CARACTERIZACIÓN DEL SENSOR IMU BNO055 POR EL METODO DE VARIANZA DE ALLAN"**

Figura 8: Modelo regresión lineal de desviacion Allan para giroscopio



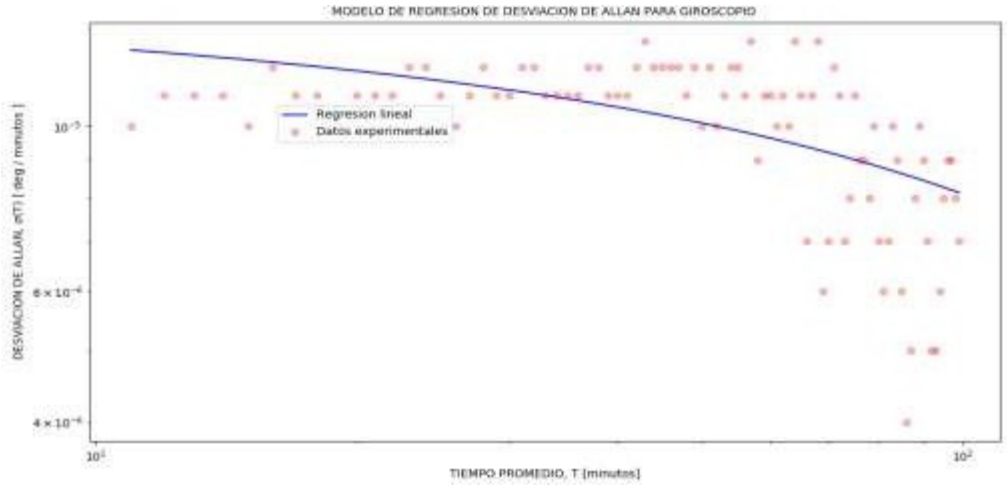
Fuente: fuente propia

Tabla 1: Fuentes de error regresión lineal 1

NOMBRE	SIMBOLO	VALOR
ERROR DE CUANTIZACIÓN	□□	**2.80e-04 -**
CAMINATA ALEATORIO DE ÁNGULO / VELOCIDAD	□□□	**-4.03e-04**
INESTABILIDAD DE SESGO	□□	**1.23e-04**
ACELERACIÓN / VELOCIDAD ANGULAR CAMINATA ALEATORIA	□□□□	**-2.08e-05**
RAMPA DE VELOCIDAD	□□□	**5.51e-07**

Ya obtenidos los valores aproximados de los 5 parámetros se aplica en la ecuación (11) el algoritmo Levenberg-Marquardt para mejorar los valores de los parámetros iniciales y obtener un mejor ajuste en los datos de Allan respecto al to, se obtuvo como resultado tabla 2:

Figura 9: Modelo de Levenberg-Marquardt de desviacion Allan para giroscopio



Fuente: Fuente propia

Tabla 2: Fuentes de error Levenberg-Marquardt 2

NOMBRE	SIMBOLO	VALOR
ERROR DE CUANTIZACIÓN	□□	**-8.38e-08**
CAMINATA ALEATORIO DE ÁNGULO / VELOCIDAD	□□□	**-2.51e-08**
INESTABILIDAD DE SESGO	□□	**6.02e-03**
ACELERACIÓN / VELOCIDAD ANGULAR CAMINATA ALEATORIA	□□□□	**-1.49e-06**
RAMPA DE VELOCIDAD	□□□	**3.31e-10**

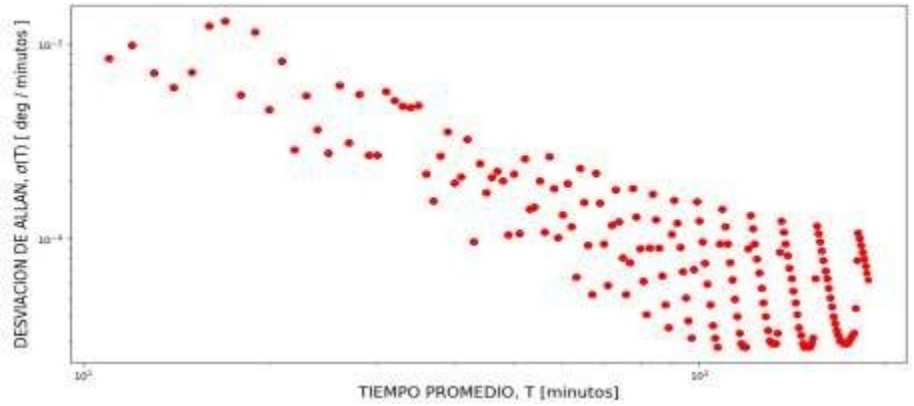
3.2 Experimento 2

Para el segundo experimento se decidió posicionar el sensor en 90° en un tiempo de una hora (los datos eran arrojados por segundo), esto para ver el comportamiento del sensor y de la fórmula de Allan en este lapso de tiempo, comparando con el experimento anterior.

Se realizo la toma de muestras, se ingresaron los datos al programa en python y se prosiguió a sacar punto por punto aplicando el método de Allan variance, en la

Figura 9 se observan los puntos de dados por la fórmula de Allan y el transcurso del tiempo.

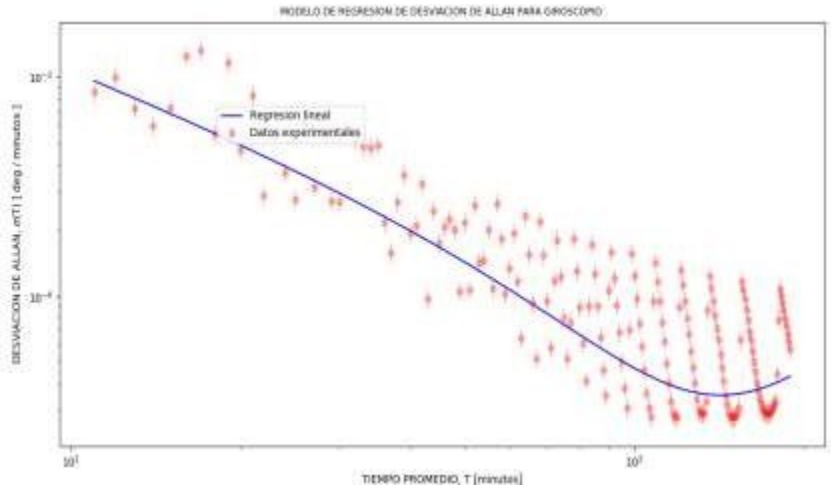
Figura 10: Datos aplicando motodo de Allan variance



Fuente: Fuente propia

Una vez obtenidos los puntos, se utilizó nuevamente la librería Curve_fit, con el fin de hallar el valor aproximado a cada parámetro requerido, que se pueden observar en la tabla 3, esto para realizar la caracterización; En la Figura 10 se puede observar la gráfica luego de la regresión lineal.

Figura 11: Modelo regresión lineal de desviacion Allan para giroscopio



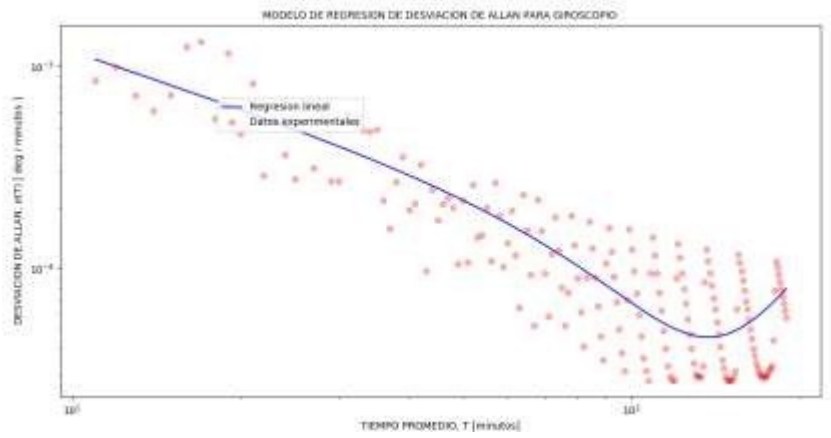
Fuente: Fuente propia

Tabla 3: Fuentes de error regresión lineal 2

NOMBRE	SIMBOLO	VALOR
ERROR DE CUANTIZACIÓN	□□	**1.06e-06**
CAMINATA ALEATORIO DE ÁNGULO / VELOCIDAD	□□□	**-1.021e-01**
INESTABILIDAD DE SESGO	□□	**2.26e-02**
ACELERACIÓN / VELOCIDAD ANGULAR CAMINATA ALEATORIA	□□□□	**-5.86e-05**
RAMPA DE VELOCIDAD	□□□	**1.21e-04**

Ya obtenidos los valores aproximados de las fuentes de error se analizó que estos errores aumentaron, se prosiguió aplicar el algoritmo Levenberg-Marquardt para poder así obtener valores más precisos de las fuentes, se obtuvo como resultado tabla 4:

Figura 12: Modelo de Levenerg-Marquardt de desviacion Allan para giroscopio



Fuente: Fuente propia

Tabla 4: Fuentes de error Levenberg-Marquardt 2

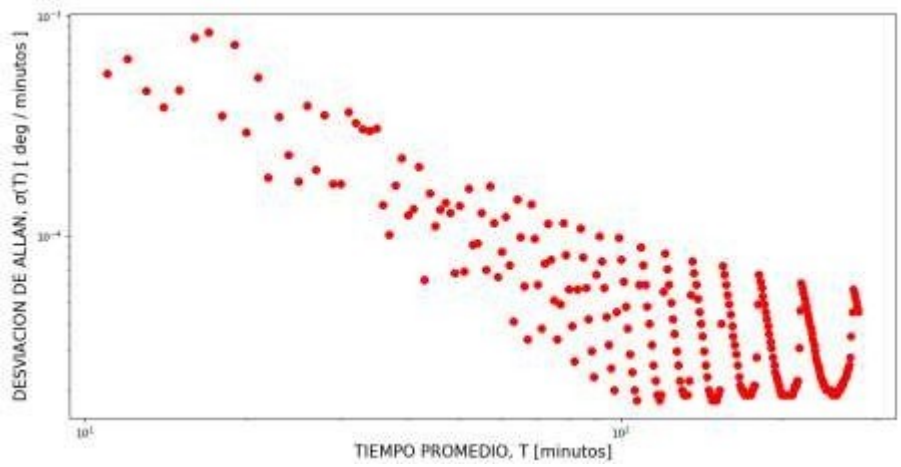
NOMBRE	SIMBOLO	VALOR
ERROR DE CUANTIZACIÓN	□□	**7.79e-03**
CAMINATA ALEATORIO DE ÁNGULO / VELOCIDAD	□□□	**-1.97e-03**
INESTABILIDAD DE SESGO	□□	**3.92e-04**
ACELERACIÓN / VELOCIDAD ANGULAR CAMINATA ALEATORIA	□□□□	**-8.62e-05**
RAMPA DE VELOCIDAD	□□□	**3.04e-06**

3.3 Experimento 3

Para este tercer experimento igual que en segundo se posicionó el sensor en 90° pero esta vez se dejó un tiempo de una hora y media (los datos eran arrojados por segundo), esto para ver el comportamiento del sensor y de la fórmula de Allan en este lapso de tiempo, comparando con los dos experimentos anteriores.

Se realizó la toma de muestras, se ingresaron los datos al programa en python y se prosiguió a sacar punto por punto aplicando el método de Allan variance, en la Figura 12 se observan los puntos de dados por la fórmula de Allan y el transcurso del tiempo.

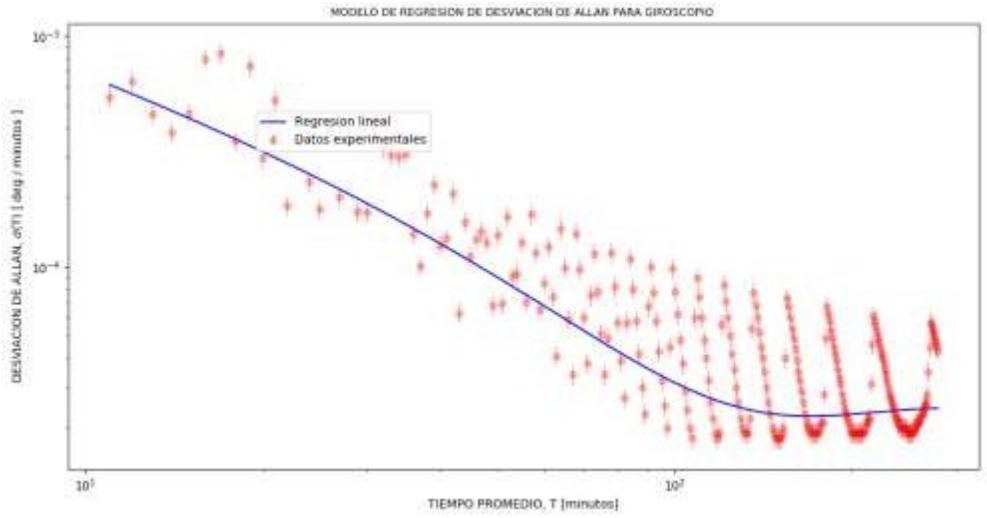
Figura 13: Datos aplicando motodo de Allan variance



Fuente: Fuente propia

Una vez obtenidos los puntos, Una vez obtenidos los puntos, se utilizó nuevamente la librería Curve_fit, con el fin de hallar el valor aproximado a cada parámetro requerido, que se pueden observar en la tabla 5, esto para realizar la caracterización; En la Figura 13 se puede observar la gráfica luego de la regresión lineal.

Figura 14: Modelo regresión lineal de desviacion Allan para giroscopio



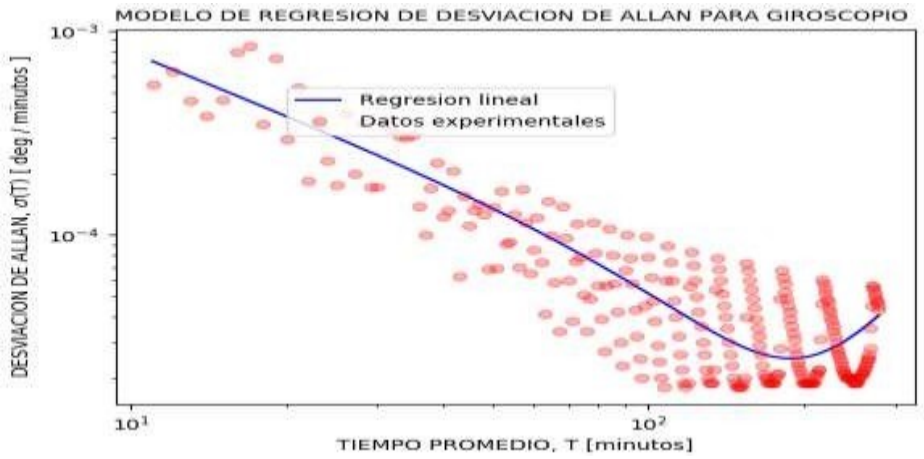
Fuente: Fuente propia

Tabla 5: Fuentes de error regresión lineal 3

NOMBRE	SIMBOLO	VALOR
ERROR DE CUANTIZACIÓN	□□	**1.06e-06**
CAMINATA ALEATORIO DE ÁNGULO / VELOCIDAD	□□□	**-1.02e-01**
INESTABILIDAD DE SESGO	□□	**2.26e-02**
ACELERACIÓN / VELOCIDAD ANGULAR CAMINATA ALEATORIA	□□□□	**-5.86e-05**
RAMPA DE VELOCIDAD	□□□	**1.21e-04**

Ya obtenidos los valores aproximados de las fuentes de error se analizó que estos errores igual que en el experimento anterior aumentaron, se prosiguió aplicar el algoritmo Levenberg-Marquardt para poder así obtener valores más precisos de las fuentes, se obtuvo como resultado tabla 6:

Figura 15: Modelo de Levenerg-Marquardt de desviacion Allan para giroscopio



Fuente: Fuente propia

Tabla 6: Fuentes de error Levenberg-Marquardt 3

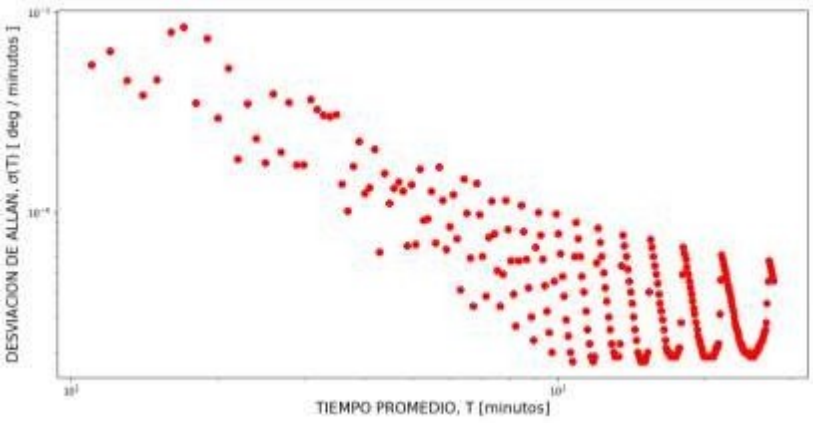
NOMBRE	SIMBOLO	VALOR
ERROR DE CUANTIZACIÓN	□□	**7.79e-03**
CAMINATA ALEATORIO DE ÁNGULO / VELOCIDAD	□□□	**-1.97e-03**
INESTABILIDAD DE SESGO	□□	**3.92e-04**
ACELERACIÓN / VELOCIDAD ANGULAR CAMINATA ALEATORIA	□□□□	**-8.62e-05**
RAMPA DE VELOCIDAD	□□□	**3.04e-06**

3.4 Experimento 4

Para el ultimo experimento al igual que en el segundo y tercero, se posiciono el sensor en 90° pero esta vez se dejó un tiempo de dos horas (los datos eran arrojados por segundo), esto para ver el comportamiento del sensor y de la fórmula de Allan en este lapso de tiempo, comparando con los dos experimentos anteriores.

Se realizó la toma de muestras, se ingresaron los datos al programa en python y se prosiguió a sacar punto por punto aplicando el método de Allan variance, en la Figura 16 se observan los puntos de dados por la fórmula de Allan y el transcurso del tiempo.

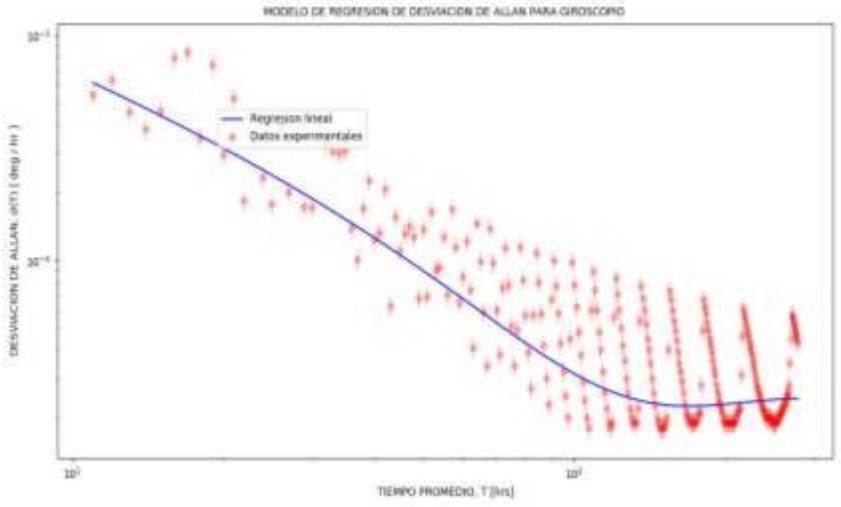
Figura 16: Datos aplicando metodo de Allan variance



Fuente: Fuente propia

Una vez obtenidos los puntos Una vez obtenidos los puntos, se utilizó nuevamente la librería Curve_fit, con el fin de hallar el valor aproximado a cada parámetro requerido, que se pueden observar en la tabla 7, esto para realizar la caracterización; En la Figura 17 se puede observar la gráfica luego de la regresión lineal.

Figura 17: Modelo regresión lineal de desviacion Allan para giroscopio



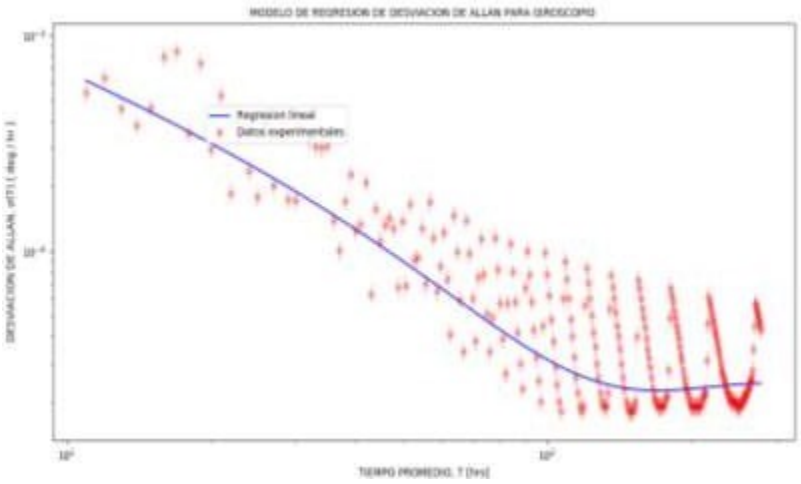
Fuente: Fuente propia

Tabla 7: Fuentes de error regresión lineal 4

NOMBRE	SIMBOLO	VALOR
ERROR DE CUANTIZACIÓN	□□	**-4.03e-06**
CAMINATA ALEATORIO DE ÁNGULO / VELOCIDAD	□□□	**7.77e-02**
INESTABILIDAD DE SESGO	□□	**2.46e-05**
ACELERACIÓN / VELOCIDAD ANGULAR CAMINATA ALEATORIA	□□□□	**-2.82e-07**
RAMPA DE VELOCIDAD	□□□	**-2.16e-05**

Ya obtenidos los valores aproximados de las fuentes de error se analizó que estos errores a medida del tiempo van amentando, se prosiguió aplicar el algoritmo Levenberg-Marquardt para poder así obtener valores más precisos de las fuentes, se obtuvo como resultado tabla 8:

Figura 18: Modelo de Levenerg-Marquardt de desviacion Allan para giroscopio



Fuente: Fuente propia

Tabla 8: Fuentes de error Levenberg-Marquardt 4

NOMBRE	SIMBOLO	VALOR
ERROR DE CUANTIZACIÓN	□□	**-1.50e-04**
CAMINATA ALEATORIO DE ÁNGULO / VELOCIDAD	□□□	**2.93e-03**
INESTABILIDAD DE SESGO	□□	**-5.64e-04**
ACELERACIÓN / VELOCIDAD ANGULAR CAMINATA ALEATORIA	□□□□	**7.18e-05**
RAMPA DE VELOCIDAD	□□□	**-1.34e-06**

4. Conclusiones y recomendaciones

4.1 Conclusiones

Se realizaron 4 experimentos donde se manipularon intervalos de tiempo diferentes para la recolección de datos de cada experimento, generando como análisis que a medida que transcurre el tiempo de recolección de datos, los valores en los 5 parámetros van aumentando, esto se debe a que los sensores inerciales van acumulando errores a medida que pasa el tiempo. De igual manera la diferencia entre las fuentes de erros va disminuyendo.

En este trabajo se implementó el método autónomo basado en Allan variance, para realizar la caracterización de sensor inercial BNO055, realizando un ajuste en la ecuación con el algoritmo de Levenerg-Marquardt, mostrando el metodo de ARMAV permite realizar la calibración a sensores inerciales de manera lineal o autonoma, sin tener conocimiento previos de los parametros de error que pueden afectar al sensor inercial que se desea calibrar.

Se realizó un documento guía donde se explica de manera de tallada el procedimiento que se utilizó para el desarrollo de este trabajo, en este documento se observa los pasos a seguir para aplicar el método de Allan variance y el algoritmo de Levenerg-Marquardt, de igual manera se explica la creación y aplicación del codigo implementado.

4.2 Recomendaciones

Con la implementación y desarrollo del presente trabajo, han surgido posibles mejoras que se pueden realizar, además de trabajos de investigación para el futuro. Algunos de estos son:

- Para mejorar la adquisición de datos por parte del sensor IMU se recomienda dejar un lugar fijo, donde no lo afecte el movimiento, ya que esto puede generar perturbaciones en la toma de datos.

- Para mejorar el desempeño de la carcaterización de los datos se recomienda dejar la recolección de los datos como mínimo seis horas, dado que entre más tiempo será mejor su desempeño.

- Comprobar utilizando IMU de mejor procesamiento para observar si hay una mejora en la recolección de datos y las fuentes de error disminuyen.

- Realizar la toma de muestras en un tipo de 6 horas para comprobar si entre mayor tiempo mejor es el resultado.

Bibliografía

[1] C. S. J. R. Jurado, «A regression-based methodology to improve estimation of inertial sensor errors using Allan variance data,» Wiley Ion, 2018.

[2] S. M. Gómez, «Análisis de unidades inerciales de medida (IMU) y diseño de controlador de ángulo de ataque aplicado a cuadricóptero,» Escuela Politécnica Superior, Alcalá, 2015.

[3] *. A. V. 2. a. J. R. 2. Alvarellos, «Raspberry Pimu: Raspberry Pi Based Inertial Sensor Data Processing System,» proceedings, Coruña, 2018.

[4] A. G. S. J. M. M. S. A. S. A. P. M. S. a. J. M. K. Nirmala, «Noise modeling and analysis of an IMU-based attitude sensor: improvement of performance by filtering and sensor fusion,» aIndian Institute of Astrophysic, India, 2016.

[5] P. Z. ·. Z. ·. Z. L. Zheng, «Error characteristics analysis and calibration testing for MEMS IMU gyroscope,» AerospaceSystems, Shanghai, China, 2019.

[6] Adafruit, «Adafriut,» 2015. [En línea]. Available: https://learn.adafruit.com/bno055-absolute-orientation-sensor-with-raspberry-pi-and-beaglebone-black/hardware. [Último acceso: 15 04 2020].

[7] InvenSense, «InvenSense Inc,» 14 09 2014. [En línea]. Available: https://stanford.edu/class/ee267/misc/MPU-9255-Datasheet.pdf. [Último acceso: 25 05 2020].

[8] M. Matej, «New Experience with Allan Variance».

[9] S. J. J. a. R. J. A. K. CM, «“regression-based methodology to,» vol. 3, nº https://doi.org/10.1002/navi.278., pp. pp. 251-263, 2019.

[10] E. R. H. Cruz, «C´alculo Diferencial e Integral III,» Facultad de Ciencias UNAM, 2016.

[11] I. M. Zambrano, «Estimación de los parámetros de un modelo haciendo uso de correspondencias con incertidumbre,» centro de investigacion en matematicas , Guanajuato, 2011.

[12] Maldonado, «Estimación de los parámetros de un modelo haciendo uso de,» 2011.

[13] I. O. C. Aburto, «Aplicacion del metodo de levenberg del método de levenberg-marquardt y del gradiente conjugado en la estimacion de la generalización de calor de un aparato de placa caliente con guarda,» Morelos, 2004.

[14] D. A. R. M. E. AL., «Implementación de una tarjeta de adquisición y procesamiento de señales para el monitoreo de distorsión armónica y parámetros de estado estable en redes de tensión menor a 1 kv,» BOGOTA, 2017.

[15] Covantec, «Covantec,» 2014. [En línea]. Available: https://entrenamiento-python-basico.readthedocs.io/es/latest/. [Último acceso: 20 04 2020].

[16] Prometec.com, [En línea]. Available: https://www.prometec.net/imu-mpu6050/. [Último acceso: 12 04 2020].

[17] «scipy.optimize.curve_fit — SciPy v1.4.1 Reference Guide,» [En línea]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit. [Último acceso: 20 05 2020].

[18] J. C. M. -. Yand, «Robot móvil de pequeña escala para seguimiento reactivo basado en LIDAR de surcos de cacao en etapa de vivero,» Universidad Antonio Nariño, Villavicencio, 2019.

[19] B. A. M. D. P. J. E. C. Veloz, Diseño, simulación y control de la dinámica de un robot planar de dos grados de libertad, documento, UNITEC, 2014.

[20] B. Sensortec, «BNO055 Intelligent 9-axis absolute orientation sensor,» Bosch, November 2014.

[21] A. Novales, «Estimación de modelos no lineales Contents,» p. 69, 2016.

Anexo 1: Deshabilitación Puerto Serial Rpi

Los comandos necesarios para la deshabilitar el puerto serial son tomados de la pagina https://learn.adafruit.com/bno055-absolute-orientation-sensor-with-raspberry-pi-and-beaglebone-black/hardware

Se comienza ejecutando la herramienta raspi-config ejecutando:

```
1. sudo raspi - config
```

Se navega por el menú “opciones avanzadas” -> “Serie A 8 “opción y cuando se le solicite si desea una shell de entrada por el puerto serie selecciona “No”. Luego se selecciona la opción de menú “Finalizar” para salir de raspi-config.

También puede ser necesario ejecutar lo siguiente para deshabilitar el servicio de terminal de inicio de sesión en el puerto serie:

```
1. sudo systemctl disable serial-getty@ttyAMA0.service
```

Finalmente, se ejecuta el comando de reinicio para reiniciar el Pi y hacer que el cambio tenga efecto:

```
1. sudo reboot
```

Anexo 2: Configuración Sensor BNO055

Para la instalación del software obligatorio en el uso del sensor inercial, es necesario asegurarse de que la Raspberry Pi cuente con el último sistema operativo de Raspbian, además de estar conectada a una red para la descarga.

Se inicia ejecutando los siguientes comandos para instalar las dependencias necesarias:

1. sudo apt-get update
2. sudo apt-get install -y build-essential python-dev python-smbus pythonpip gi

A continuidad, se ejecutan los siguientes comandos para descargar e instalar la última versión del código del módulo BNO055:

1. cd ~
2. git clone https://github.com/adafruit/Adafruit_Python_BNO055.git 3.
cd Adafruit_Python_BNO055
4. sudo python setup.py install

Si la instalación falla, se observa un mensaje de error, toca volver atrás y verificar que se hallan instalado las dependencias anteriores y volver a intentarlo.

Anexo 3: Programa de aplicación Allan variance

```
import pandas as pd
import matplotlib.pyplot as pl
import numpy as np
import promedios as std
numero_de_veces=350
cambio_de_tao=10
sumar=1
periodo_de_muestreo=1
v=0
w=0
while v<numero_de_veces:
 datos=open("90gradosm.txt",'r')
 tao=0
 datos_por_matriz=0
 avance_de_dato=0
 tao=cambio_de_tao
 datos_por_matriz=cambio_de_tao
 avance_de_dato=cambio_de_tao
 linea_a_linea=datos.readlines()
 muestras=[]
 for p in linea_a_linea:
 muestras.append(p.split(' ')[0])
 datos.close()
 N=len(muestras)
 rango_sumar=N/datos_por_matriz
 l=rango_sumar-1
 hasta=rango_sumar-2
 desde=0
 z=0
 x=0
 y=1
 w=0
 suma=0
 for i in range(0,l):
 globals()["a{}".format(i)]=[]
 globals()["b{}".format(i)]=[]
 globals()["c{}".format(i)]=[]
 globals()["d{}".format(i)]=[]
 globals()["e{}".format(i)]=[]
 globals()["f{}".format(i)]=[]
 globals()["s{}".format(i)]=[]
 while datos_por_matriz<=N:
 globals()["a{}".format(z)]=muestras[desde:datos_por_matriz]
 datos_por_matriz=datos_por_matriz+avance_de_dato
 desde=desde+avance_de_dato
 globals()["b{}".format(z)]= np.genfromtxt(globals()["a{}".format(z)], dtype=float)
 globals()["c{}".format(z)]=pd.DataFrame(data= globals()["b{}".format(z)])
 globals()["d{}".format(z)]=globals()["c{}".format(z)].mean(axis=0)
 z=z+1
 while x<=hasta:
 globals()["e{}".format(x)]=(globals()["d{}".format(y)]
 globals()["d{}".format(x)])**2
```

```
 suma=suma+globals()['e{}'.format(x)]
 x=x+1
 y=y+1
 allan=suma/(2*(N-2*(tao/periodo_de_muestreo)))
 cambio_de_tao=cambio_de_tao+sumar

 v=v+1
 resultao=open("tao2.txt","a")
 resultao.write('%s'%tao + '\n' + "\n" )
resultao.close()
resulallan=open("allan1.txt","a")
resulallan.write('%s'%allan +'\n' )
resulallan.close()
nuevo=open('allanm.txt','a')
original=open('allan1.txt','r')
for x in original:
 m=x.replace("dtype: float64","")
 nuevo.write(m)
nuevo.close()
original.close()
nuevo.close()
```

Anexo 4: Programa de aplicación regresión lineal

PARAMETROS INICIALES

```
from scipy.optimize import curve_fit
def funcion_ajuste(x, q, r, o, w, s):
 x=datosx.ix[:,0]
 xq=(np.sqrt(3)/x)
 xr=(1/np.sqrt(x))
 xo=1
 xw=(np.sqrt(x)/np.sqrt(3))
 xs=(x/np.sqrt(2))
 f=(xq*q)+(xr*r)+(xo*o)+(xw*w)+(xs*s)
 return f
datosx=pd.read_csv('tao2.txt', header=0,delim_whitespace=True)
x=datosx.ix[:,0]
print(x)
datosy=pd.read_csv('allanm.txt', header=0,delim_whitespace=True)
y=datosy.ix[:,1]
print(y)
#t = funcion_ajuste(xt,0,0,0,0,0)
st = y*.1
fg, ax = pl.subplots()
ax.errorbar(x, y, yerr=st, fmt='or', label='Datos experimentales',alpha=0.3)
```

```
ax.set_title('MODELO DE REGRESION DE DESVIACION DE ALLAN PARA GIROSCOPIO ', {'fontsize':
10})
ax.set_ylabel(r"DESVIACION DE ALLAN, $\sigma$(T) [ deg / hr ]", {'fontsize': 10})
ax.set_xlabel(r"TIEMPO PROMEDIO, T [hrs]", {'fontsize': 10})
#%%
par, pcov = curve_fit(funcion_ajuste, x, y, sigma=st)
print("Vector de parametros")
print(par) a
= par[0] b
= par[1] c
= par[2] d
= par[3] e
= par[4]
print("Matriz de covarianzas")
print(pcov)
43
sa = np.sqrt(pcov[0, 0])
sb = np.sqrt(pcov[1, 1])
ax.loglog(x, funcion_ajuste(x, a, b, c, d, e), '-b', label=r"Regresion lineal")
ax.legend(loc='lower left', bbox_to_anchor=(.2, .7), frameon=True)
```

PARAMETROS FINALES

```
#%%
def funcion_ajuste2(x, q2, r2, o2, w2, s2):
 xq2=(q2*(np.sqrt(3)/x))**2
 xr2=(r2*(1/np.sqrt(x)))**2
 xo2=(o2*(np.sqrt((2*np.log(2))/np.pi)))**2
 xw2=(w2*(np.sqrt(x)/np.sqrt(3)))
 xs2=(s2*(x/np.sqrt(2)))**2
 f=xq2+xr2+xo2+xw2+xs2
 return f
datosx=pd.read_csv('tao2.txt', header=0,delim_whitespace=True)
x2=datosx.ix[:,0]
datosy=pd.read_csv('allanm.txt', header=0,delim_whitespace=True)
y2=datosy.ix[:,1]
st=y2*.1
fg, ax = pl.subplots()
ax.errorbar(x2, y2, fmt='or', label="Datos experimentales',alpha=0.3)
ax.set_title('MODELO DE REGRESION DE DESVIACION DE ALLAN PARA GIROSCOPIO ', {'fontsize':
10})
ax.set_ylabel(r"DESVIACION DE ALLAN, $\sigma$(T) [ deg / hr ]", {'fontsize': 10})
ax.set_xlabel(r"TIEMPO PROMEDIO, T [hrs]", {'fontsize': 10})
#%%
par2, pcov2 = curve_fit(funcion_ajuste2, x2, y2, p0=par)
print("Vector de parametros")
print(par2)
f = par2[0]
g = par2[1]
h = par2[2]
```

```
i = par2[3]
j = par2[4]
print('Matriz de covarianzas')
print(pcov2)
sa = np.sqrt(pcov2[0, 0])
sb = np.sqrt(pcov2[1, 1])
ax.loglog(x2, funcion_ajuste2(x2, f, g, h, i, j), '-b', label=r'Regresion lineal')
ax.legend(loc='lower left', bbox_to_anchor=(.2, .7), frameon=True)
```

Anexo 5: Librería promedios

```
import numpy as np
from scipy import stats
import matplotlib.pyplot as pl
def val_rep(Xd, Yd, Dx):
 Nr = np.array([])
 sY = np.array([])
 sYr = np.array([])
 Yr = np.array([])
 Xr = np.array([])
 if np.size(np.shape(Dx)) > 0:
 X = Dx
 else:
 X = Dx*np.round(Xd/Dx)
 X_sort = np.sort(X, 0)
 dX = np.diff(X_sort, 1, 0)
 dX = X_sort[1+np.nonzero(dX != 0)[0]]
 dX = np.hstack((np.array([np.min(Xd)]), dX))
 for n in range(np.size(dX, 0)-1):
 # print n
 i_r = np.nonzero(np.logical_and(Xd >= dX[n], Xd < dX[n+1]))[0]
 Nr = np.hstack((Nr, np.size(i_r, 0)))
 sY = np.hstack((sY, np.std(Yd[i_r], 0)))
 sYr = np.hstack((sYr, sY[n]/np.sqrt(Nr[n])))
 Yr = np.hstack((Yr, np.mean(Yd[i_r], 0)))
 Xr = np.hstack((Xr, np.mean(Xd[i_r], 0)))
 pl.loglog(Xr,Yr,"or")
 pl.ylabel(r"DESVIACION DE ALLAN, $\sigma$(T) [ deg / hr ]",
{"fontsize": 15})
 pl.xlabel(r"TIEMPO PROMEDIO, T [hrs]", {"fontsize": 15})
```