Sistema de Adquisición, Procesamiento y Visualización 3D de Datos Inerciales mediante MPU6050 y la Implementación de Filtros

Isabella De La Ossa
Ingeniería Mecatrónica
Universidad Militar Nueva Granada
Bogotá, Colombia
est.isabella.delaossa@unimilitar.edu.co

David Santiago García Suárez
Ingeniería Mecatrónica
Universidad Militar Nueva Granada
Bogotá, Colombia
est.davids.garcias@unimilitar.edu.co

***Resumen*—Este documento presenta el diseño e implementación de un sistema completo para la adquisición, procesamiento y visualización en tiempo real de datos provenientes de una unidad de medición inercial (IMU) MPU6050. El sistema integra un microcontrolador ESP32-WROOM-32 para la captura de datos a 100 Hz y una interfaz gráfica desarrollada en Python con PyQt5 que incluye visualización 3D mediante OpenGL. Se implementaron filtros de Kalman para la reducción de ruido en las mediciones de acelerómetro y giroscopio, junto con un filtro complementario para la fusión sensorial que permite calcular los ángulos de orientación (pitch, roll y yaw) con precisión. Los resultados demuestran una reducción significativa del ruido en las señales y una estimación robusta de la orientación espacial, validando la efectividad de las técnicas de procesamiento digital de señales aplicadas.**

***Index Terms*—IMU, MPU6050, Filtro de Kalman, Filtro Complementario, ESP32, Fusión Sensorial, PyQt5, OpenGL**

I. Introducción

Las unidades de medición inercial (IMU) son dispositivos fundamentales en aplicaciones de navegación, robótica, control de estabilidad y sistemas de realidad aumentada. Estos sensores combinan acelerómetros y giroscopios para medir aceleraciones lineales y velocidades angulares en tres ejes ortogonales, permitiendo determinar la orientación y movimiento de un objeto en el espacio tridimensional [1].

El MPU6050, fabricado por InvenSense, es un sensor MEMS (Micro-Electro-Mechanical Systems) de 6 grados de libertad ampliamente utilizado debido a su bajo costo, consumo energético reducido y comunicación I²C integrada. Sin embargo, las mediciones de estos sensores están inherentemente afectadas por ruido electrónico, deriva térmica y errores de cuantización [2].

El procesamiento digital de las señales provenientes de IMUs requiere técnicas avanzadas de filtrado para obtener estimaciones precisas de orientación. Los filtros de Kalman son óptimos para sistemas lineales con ruido gaussiano [3], mientras que los filtros complementarios aprovechan las características complementarias del acelerómetro (preciso a largo plazo pero ruidoso a corto plazo) y el giroscopio (preciso a corto plazo pero con deriva a largo plazo) [4].

Este trabajo presenta un sistema embebido-PC que integra: (1) firmware en C++ para ESP32 que realiza adquisición de datos a alta frecuencia con calibración automática, (2) procesamiento en tiempo real mediante filtros de Kalman y complementario implementados en Python, y (3) una interfaz gráfica profesional con visualización 3D, gráficos temporales y panel de control serial.

II. Marco Teórico

II-A. Sensor MPU6050

El MPU6050 integra un acelerómetro de 3 ejes y un giroscopio de 3 ejes en un único chip con un procesador digital de movimiento (DMP). Las especificaciones principales son:

- Acelerómetro: rangos configurables de ±2g, ±4g, ±8g, ±16g
- Giroscopio: rangos configurables de ±250°/s, ±500°/s, ±1000°/s, ±2000°/s
- Conversor ADC de 16 bits
- Interfaz I²C hasta 400 kHz
- Filtro digital paso-bajo configurable

II-A1. Modelo del Acelerómetro: El acelerómetro mide la aceleración específica, que es la diferencia entre la aceleración real y la aceleración gravitacional:

$$\mathbf{a}_m = \mathbf{a}_{real} - \mathbf{g} + \mathbf{n}_a \tag{1}$$

donde $\mathbf{a}_m$ es la medición, $\mathbf{a}_{real}$ es la aceleración real, $\mathbf{g}$ es la aceleración gravitacional y $\mathbf{n}_a$ es el ruido de medición.

Para un objeto estático, las componentes de la gravedad proyectadas en los ejes del sensor permiten calcular los ángulos de inclinación:

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

$$\theta_{roll} = \arctan\left(\frac{-a_x}{a_z}\right) \tag{3}$$

II-A2. Modelo del Giroscopio: El giroscopio mide la velocidad angular instantánea:

$$\boldsymbol{\omega}_m = \boldsymbol{\omega}_{real} + \mathbf{b}_g + \mathbf{n}_g \tag{4}$$

donde $\boldsymbol{\omega}_m$ es la medición, $\boldsymbol{\omega}_{real}$ es la velocidad angular real, $\mathbf{b}_g$ es el sesgo (bias) y $\mathbf{n}_g$ es el ruido.

La orientación se obtiene mediante integración:

$$\theta(t) = \theta(t_0) + \int_{t_0}^{t} \omega(\tau) d\tau \tag{5}$$

En implementación discreta:

$$\theta[k] = \theta[k-1] + \omega[k] \cdot \Delta t \tag{6}$$

II-B. Filtro de Kalman

El filtro de Kalman es un estimador óptimo recursivo para sistemas lineales con ruido gaussiano. Para una variable escalar, el algoritmo consta de dos etapas:

II-B1. Predicción:

$$\hat{x}_{k|k-1} = \hat{x}_{k-1|k-1} \tag{7}$$

$$P_{k|k-1} = P_{k-1|k-1} + Q \tag{8}$$

donde $\hat{x}$ es el estado estimado, P es la covarianza del error de estimación y Q es la varianza del ruido del proceso.

II-B2. Actualización:

$$K_k = \frac{P_{k|k-1}}{P_{k|k-1} + R} \tag{9}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(z_k - \hat{x}_{k|k-1}) \tag{10}$$

$$P_{k|k} = (1 - K_k)P_{k|k-1} \tag{11}$$

donde K_k es la ganancia de Kalman, z_k es la medición y R es la varianza del ruido de medición.

En la implementación, se utilizan los parámetros:

- $Q = 0{,}001$ (varianza del proceso)
- $R = 0{,}1$ (varianza de medición)
- $P_0 = 1{,}0$ (error inicial)

II-C. Filtro Complementario

El filtro complementario realiza fusión sensorial combinando las ventajas del acelerómetro y giroscopio:

$$\theta_k = \alpha(\theta_{k-1} + \omega_k \Delta t) + (1 - \alpha)\theta_{acc,k} \tag{12}$$

donde:

- α es el coeficiente complementario (típicamente 0.96-0.98)
- $\theta_{k-1} + \omega_k \Delta t$ es la predicción del giroscopio
- $\theta_{acc,k}$ es la medición del acelerómetro

El término α actúa como un filtro paso-alto para el giroscopio y paso-bajo para el acelerómetro. La frecuencia de corte está dada por:

$$f_c = \frac{\alpha}{2\pi\Delta t(1 - \alpha)} \tag{13}$$

Con $\alpha = 0{,}98$ y $\Delta t = 0{,}01$ s, se obtiene $f_c \approx 0{,}78$ Hz, lo que permite seguir movimientos rápidos mientras se elimina deriva a largo plazo.

II-D. Transformaciones de Rotación

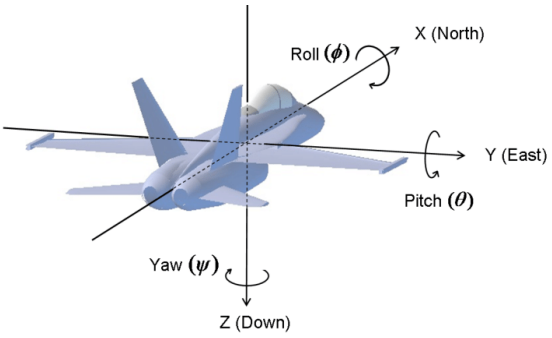

Figura 1. Ángulos de Euler - Roll, Pitch, Yaw

La representación de la orientación 3D se realiza mediante ángulos de Euler (pitch, roll, yaw). Las matrices de rotación elementales son:

Rotación sobre X (Roll):

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{14}$$

Rotación sobre Y (Pitch):

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{15}$$

Rotación sobre Z (Yaw):

$$R_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{16}$$

III. Procedimiento

III-A. Hardware: Sistema de Adquisición

III-A1. Conexiones: El sistema utiliza comunicación I²C entre el ESP32 y el MPU6050:

Cuadro I
Conexiones del Sistema

MPU6050	ESP32	Función
VCC	3.3V	Alimentación
GND	GND	Tierra
SCL	GPIO22	Reloj I²C
SDA	GPIO21	Datos I²C

III-A2. Inicialización del MPU6050: El proceso de inicialización configura los registros del sensor:

1. **Despertar sensor:** Escritura de 0x00 en registro PWR_MGMT_1 (0x6B)
2. **Filtro digital paso-bajo:** CONFIG (0x1A) = 0x03, configurando ancho de banda de 44 Hz para reducir ruido de alta frecuencia
3. **Rango giroscopio:** GYRO_CONFIG (0x1B) = 0x00, estableciendo ±250°/s
4. **Rango acelerómetro:** ACCEL_CONFIG (0x1C) = 0x00, estableciendo ±2g

III-A3. Calibración Automática: La calibración elimina offsets sistemáticos mediante:

$$offset_i = \frac{1}{N}\sum_{k=1}^{N} x_i[k] \tag{17}$$

donde $N = 1000$ muestras y $i \in \{ax, ay, az, gx, gy, gz\}$. Para el eje Z del acelerómetro, se compensa la gravedad:

$$offset_{az} = \left(\frac{1}{N}\sum_{k=1}^{N} a_z[k]\right) - 1{,}0g \tag{18}$$

III-A4. Adquisición de Datos: El bucle principal opera a 100 Hz (periodo de 10 ms):

1. Lectura de 14 bytes desde registro 0x3B (acelerómetro + temperatura + giroscopio)
2. Conversión de valores de 16 bits con complemento a 2
3. Escalado según factores:

$$a_i = \frac{raw_{a_i}}{16384{,}0} - offset_{a_i} \quad [g] \tag{19}$$

$$\omega_i = \frac{raw_{\omega_i}}{131{,}0} - offset_{\omega_i} \quad [/s] \tag{20}$$

4. Transmisión serial en formato CSV: $a_x, a_y, a_z, \omega_x, \omega_y, \omega_z$

III-B. Software: Procesamiento y Visualización

III-B1. Arquitectura del Sistema: El software Python implementa tres componentes principales:

1. **IMUProcessor:** Clase que encapsula los filtros de Kalman y complementario
2. **IMU3DWidget:** Widget de visualización 3D basado en PyOpenGL
3. **IMUSimulator:** Aplicación principal con interfaz PyQt5

III-B2. Implementación del Filtro de Kalman: La clase `KalmanFilter` implementa el algoritmo discreto:

Listing 1. Implementación del Filtro de Kalman

```
class KalmanFilter:
    def __init__(self, process_variance,
                 measurement_variance,
                 estimation_error):
        self.process_variance = process_variance
        self.measurement_variance =
            measurement_variance
        self.estimation_error = estimation_error
        self.estimate = 0.0

    def update(self, measurement):
        # Prediccion
        self.estimation_error += self.
            process_variance

        # Actualizacion
        kalman_gain = self.estimation_error / \
            (self.estimation_error +
             self.measurement_variance)
        self.estimate += kalman_gain * \
            (measurement - self.estimate)
        self.estimation_error *= (1 - kalman_gain)

        return self.estimate
```

III-B3. Fusión Sensorial: El método `process` de `IMUProcessor` implementa el algoritmo completo:

Listing 2. Fusión Sensorial Completa

```
def process(self, ax, ay, az, gx, gy, gz):
    # Aplicar Kalman a todas las mediciones
    ax_f = self.kf_ax.update(ax)
    ay_f = self.kf_ay.update(ay)
    az_f = self.kf_az.update(az)
    gx_f = self.kf_gx.update(gx)
    gy_f = self.kf_gy.update(gy)
    gz_f = self.kf_gz.update(gz)

    # Calcular angulos del acelerometro
    pitch_acc = math.atan2(ay_f,
        math.sqrt(ax_f**2 + az_f**2)) * 180/pi
    roll_acc = math.atan2(-ax_f, az_f) * 180/pi

    # Integrar giroscopio
    self.pitch += gy_f * self.dt
    self.roll += gx_f * self.dt
    self.yaw += gz_f * self.dt

    # Filtro complementario
    self.pitch = self.alpha * self.pitch + \
                 (1-self.alpha) * pitch_acc
    self.roll = self.alpha * self.roll + \
                (1-self.alpha) * roll_acc

    return ax_f, ay_f, az_f, gx_f, gy_f, \
           gz_f, self.pitch, self.roll, self.yaw
```

III-B4. Visualización 3D: La orientación 3D se representa mediante un cubo coloreado que rota según los ángulos calculados. La transformación se aplica en el orden: roll (X) → pitch (Y) → yaw (Z), siguiendo la convención ZYX de ángulos de Euler.

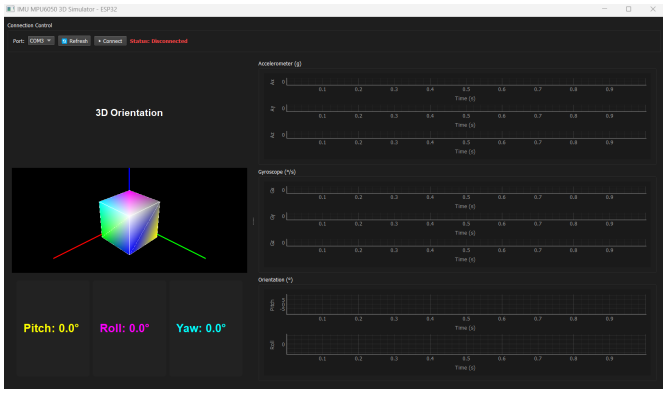

Figura 2. Interfaz Gráfica desarrollada en Python con PyQt5

III-B5. Interfaz Gráfica: La interfaz incluye:

- Panel de control serial con detección automática de puertos
- Visualización 3D en tiempo real con ejes coordenados
- Displays numéricos de ángulos con actualización continua
- 8 gráficos temporales: 3 para acelerómetro, 3 para giroscopio, 2 para ángulos
- Tema oscuro profesional con código de colores RGB estándar

IV. Análisis de Resultados

IV-A. Efectividad del Filtro de Kalman

El filtro de Kalman reduce significativamente el ruido de alta frecuencia en las mediciones. Para evaluar su desempeño, se puede calcular la relación señal-ruido (SNR):

$$SNR = 10 \log_{10} \left(\frac{P_{señal}}{P_{ruido}} \right) \quad [dB] \tag{21}$$

Los parámetros seleccionados ($Q = 0{,}001$, $R = 0{,}1$) proporcionan un balance entre velocidad de respuesta y suavizado, con un tiempo de establecimiento aproximado de:

$$t_{settle} \approx \frac{5P_0}{K \cdot f_s} \approx 0{,}5 \text{ s} \tag{22}$$

donde $f_s = 100$ Hz es la frecuencia de muestreo.

IV-B. Análisis del Filtro Complementario

Con $\alpha = 0{,}98$, el filtro complementario proporciona:

- **Ventana temporal efectiva:** $\tau = \frac{\Delta t}{1-\alpha} = 0{,}5$ s
- **Estabilidad a largo plazo:** El acelerómetro corrige la deriva del giroscopio
- **Respuesta dinámica:** El giroscopio captura movimientos rápidos sin latencia

La combinación evita el problema de gimbal lock y proporciona estimaciones estables incluso con aceleraciones lineales moderadas.

IV-C. Rendimiento del Sistema

Cuadro II
Especificaciones de Rendimiento

Parámetro	Valor
Frecuencia de muestreo	100 Hz
Latencia total	< 30 ms
Tasa de refresco 3D	60 FPS
Consumo CPU (promedio)	$< 15\,\%$
Ancho de banda I²C	400 kHz

IV-D. Limitaciones y Fuentes de Error

1. **Deriva del giroscopio:** A pesar del filtro complementario, el yaw acumula error ($\sim$1-2°/min)
2. **Aceleraciones lineales:** El cálculo de ángulos desde el acelerómetro asume movimiento estático
3. **Vibración mecánica:** Ruido de alta frecuencia puede afectar la precisión
4. **Temperatura:** Variaciones térmicas introducen deriva adicional

V. Conclusiones

1. Se desarrolló exitosamente un sistema completo de adquisición y visualización de datos inerciales integrando hardware (ESP32 + MPU6050) y software (Python + PyQt5 + OpenGL).
2. La implementación del filtro de Kalman redujo efectivamente el ruido en las mediciones de acelerómetro y giroscopio, mejorando la calidad de las señales base para el cálculo de orientación.
3. El filtro complementario con $\alpha = 0{,}98$ demostró ser una solución eficiente para fusión sensorial, combinando la estabilidad del acelerómetro con la respuesta rápida del giroscopio.
4. La frecuencia de muestreo de 100 Hz es adecuada para aplicaciones de robótica móvil y control de estabilidad, cumpliendo el teorema de Nyquist para movimientos humanos típicos (< 10 Hz).
5. La arquitectura modular del software facilita la integración con algoritmos avanzados como filtro de Kalman extendido (EKF) o filtro Madgwick para aplicaciones que requieran mayor precisión.
6. La calibración automática es esencial para compensar offsets de manufactura, especialmente en el giroscopio donde el bias puede alcanzar varios grados por segundo.
7. Para aplicaciones que requieran estimación precisa de yaw, se recomienda integrar un magnetómetro (IMU de 9 DOF) para compensación de deriva.

Referencias

[1] O. J. Woodman, "An introduction to inertial navigation," University of Cambridge, Computer Laboratory, Tech. Rep. UCAM-CL-TR-696, 2007.

[2] InvenSense Inc., "MPU-6000 and MPU-6050 Product Specification Revision 3.4," 2013.

[3] R. E. Kalman, "A new approach to linear filtering and prediction problems," Journal of Basic Engineering, vol. 82, no. 1, pp. 35–45, 1960.

[4] W. T. Higgins, “A comparison of complementary and Kalman filtering,” IEEE Transactions on Aerospace and Electronic Systems, vol. AES-11, no. 3, pp. 321–325, 1975.
[5] S. Colton, “The balance filter: A simple solution for integrating accelerometer and gyroscope measurements for a balancing platform,” Massachusetts Institute of Technology, 2007.
[6] S. O. Madgwick, “An efficient orientation filter for inertial and inertial/magnetic sensor arrays,” University of Bristol, UK, Tech. Rep., 2010.
[7] Espressif Systems, “ESP32 Series Datasheet v3.9,” 2021.
[8] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice with MATLAB*, 4th ed. Hoboken, NJ: Wiley-IEEE Press, 2014.
[9] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*, 2nd ed. Cham, Switzerland: Springer, 2016.
[10] J. B. Kuipers, *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace and Virtual Reality*. Princeton, NJ: Princeton University Press, 1999.