MANUAL DE ESTUDIO · SISTEMAS Y CONTROL EN MECATRÓNICA

Filtro de Kalman, observadores y procesamiento digital

Capítulo 04 · En el video: **minuto 31:38** · Duración 10:29

Objetivo. Estimar estados no medidos y procesar señales de sensores en tiempo real.

Ideas clave

Observador de Luenberger

Copia el modelo y lo corrige con la medición: el error evoluciona como $e[k+1] = (\Phi - LC)e[k]$. Polos rápidos convergen rápido pero amplifican el ruido.

Filtro de Kalman

Elige la ganancia óptima a partir del ruido de proceso Q (cuánto se confía en el modelo) y del ruido de medición R (cuánto se confía en el sensor). Ciclo: predicción ($\hat{x}^- = \Phi\hat{x}$, $P^- = \Phi P \Phi^T + Q$) y actualización ($K = P^- H^T (H P^- H^T + R)^{-1}$, $\hat{x} = \hat{x}^- + K(z - H\hat{x}^-)$).

Velocidad desde un encoder de 400 cuentas/vuelta (Ts = 10 ms), detalle cursosmecatronica.com



La derivada salta entre múltiplos de 1,57 rad/s (una cuenta por periodo). Kalman combina el modelo de velocidad casi constante con las lecturas y reduce el error rms de 0,69 a 0,40 rad/s.

Estimación de estados con el filtro de Kalman.

Aplicaciones

La ganancia converge a un valor fijo (ecuación de Riccati) que puede precalcularse. Se usa para fusionar IMU y encoder, estimar velocidad desde un encoder y, con un estado aumentado, estimar la perturbación de carga. EKF linealiza con jacobianos; UKF propaga puntos sigma y maneja mejor las no linealidades fuertes.

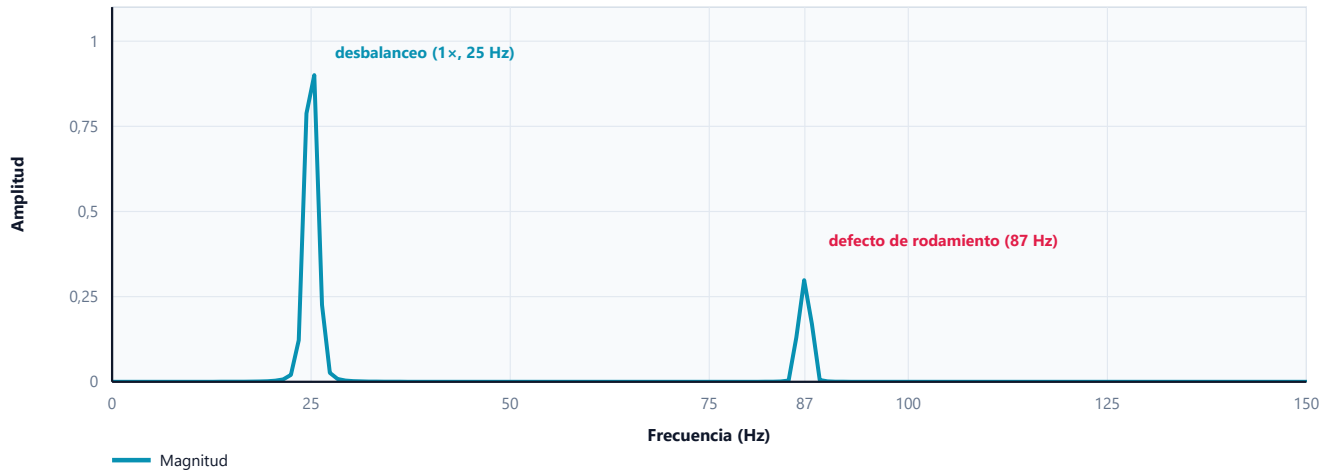
Muestreo y aliasing

Con $f_s > 2f_{máx}$ (Nyquist) la señal se puede reconstruir; si no, las frecuencias altas se pliegan y aparecen como bajas. Por eso se pone un filtro analógico antialias antes del ADC o se sobremuestrea.

Filtros digitales y FFT

FIR: fase lineal exacta pero más retardo. IIR (biquads Butterworth, notch): menos cálculo y retardo, preferidos dentro de los lazos. La FFT calcula el espectro en $N \cdot \log_2 N$ operaciones; la ventana de Hann reduce la fuga espectral. Se usa para diagnosticar rodamientos y motores.

Espectro de la vibración de un motor (FFT de 1024 puntos, ventana de Hann) cursosmecatronica.com



Resolución fs/N = 0,98 Hz. La ventana de Hann reduce la fuga espectral: cada tono queda en unas pocas líneas en vez de esparcirse.

Diagnóstico por análisis espectral.

DSP embebido

Operaciones MAC, aritmética de punto fijo Q15/Q31 con saturación, y DMA en ping-pong para adquirir sin ocupar la CPU. CMSIS-DSP ofrece filtros y FFT optimizados para Cortex-M.

Fórmulas y relaciones

Relación	Expresión	Cómo leerla
Predicción	$\hat{x}^- = \Phi\hat{x} + \Gamma u,\ P^- = \Phi P \Phi^T + Q$	A priori.
Ganancia de Kalman	$K = P^- H^T (H P^- H^T + R)^{-1}$	Óptima con ruido gaussiano.
Actualización	$\hat{x} = \hat{x}^- + K(z - H\hat{x}^-),\ P = (I - KH)P^-$	A posteriori.
Nyquist-Shannon	$f_s > 2 \cdot f_{máx}$	Si no, aliasing.
Resolución de la FFT	$\Delta f = f_s / N$	0,98 Hz con 1024 puntos a 1 kHz.
Biquad (DF2T)	$y = b_0 x + z_1;\ z_1 = b_1 x - a_1 y + z_2;\ z_2 = b_2 x - a_2 y$	Dos estados por sección.

Ejemplo resuelto: velocidad desde un encoder de baja resolución

1. Encoder de 400 cuentas/vuelta: resolución 15,7 mrad; T_s = 10 ms.
2. Derivada directa: cada cuenta equivale a $0{,}0157/0{,}01 = 1{,}57$ rad/s: la velocidad salta en escalones.
3. Modelo de velocidad casi constante con Q de aceleración aleatoria y $R = \text{res}^2/12$.
4. La ganancia converge a $K = [0{,}83;\ 64{,}6]$.
5. Error rms de velocidad: 0,69 rad/s por derivada y 0,40 rad/s con Kalman.

kalman_posicion_velocidad.py implementa el filtro con matrices; dsp_filtros_fft.py diseña el biquad y la FFT del capítulo.

Filtros digitales

Tipo	Fase	Costo	Uso típico
FIR	Lineal (sin distorsión)	Muchos coeficientes	Medición, audio, análisis fuera de línea
IIR Butterworth	No lineal, poco retardo	5 coeficientes por biquad	Filtrar sensores dentro del lazo
Notch	Afecta cerca de f_0	1 biquad	Eliminar una resonancia mecánica o 50 Hz
Media móvil	Lineal	Muy bajo	Suavizado simple, rechazo de una frecuencia

Código del capítulo

Los archivos completos vienen en el paquete ZIP del capítulo (carpeta `codigo/`).

kalman_posicion_velocidad.py — filtro de Kalman de posición y velocidad con un encoder cuantificado. *(extracto: primeras 60 de 118 líneas)*

```
"""
kalman_posicion_velocidad.py
Filtro de Kalman para un eje con encoder de baja resolución (capítulo 04).

Modelo de velocidad casi constante con Ts = 10 ms:
    x = [posición, velocidad],  Phi = [[1, Ts], [0, 1]],  H = [[1, 0]]
    Q = q·[[Ts³/3, Ts²/2], [Ts²/2, Ts]]   (aceleración aleatoria de densidad q)
    R = (resolución²)/12                  (cuantificación del encoder)

Compara la velocidad obtenida por derivada directa del encoder con la del
filtro de Kalman, calcula la ganancia estacionaria iterando la ecuación de
Riccati y la compara con un observador de Luenberger con polos fijos.

Uso:  python kalman_posicion_velocidad.py
"""
import math
import random

def mat(a, b):
    """Producto de matrices (listas de listas)."""
    return [[sum(a[i][k] * b[k][j] for k in range(len(b))) for j in range(len(b[0]))] for i in range(len(a))]


def suma(a, b, s=1.0):
    return [[x + s * y for x, y in zip(fa, fb)] for fa, fb in zip(a, b)]


def esc(a, k):
```

```
    return [[k * x for x in f] for f in a]


def tr(a):
    return [list(f) for f in zip(*a)]


def eye(n):
    return [[1.0 if i == j else 0.0 for j in range(n)] for i in range(n)]


def inv(a):
    n = len(a)
    m = [f[:] + e for f, e in zip(a, eye(n))]
    for c in range(n):
        p = max(range(c, n), key=lambda f: abs(m[f][c]))
        m[c], m[p] = m[p], m[c]
        piv = m[c][c]
        m[c] = [x / piv for x in m[c]]
        for f in range(n):
            if f != c:
                k = m[f][c]
                m[f] = [x - k * y for x, y in zip(m[f], m[c])]
    return [f[n:] for f in m]


def expm(a, t=1.0):
    """e^(A·t) por escalado y elevación al cuadrado con serie de Taylor."""
    n, s = len(a), 0
    norma = max(sum(abs(x) for x in f) for f in a) * abs(t)
    while norma > 0.5:
        norma, s = norma / 2, s + 1
```

dsp_filtros_fft.py — aliasing, biquad Butterworth, notch y FFT con ventana de Hann.

filtro_biquad.ino — biquad IIR en Arduino a 1 kHz para filtrar un sensor.

Errores frecuentes

- Ajustar Q y R al azar: R se mide con el sensor quieto; Q se ajusta según la dinámica esperada.
- Muestrear sin filtro antialias.
- Usar un FIR largo dentro de un lazo rápido: su retardo quita margen de fase.
- Aplicar la FFT sin ventana a una señal que no completa periodos enteros.

Autoevaluación

1. ¿Qué representa R en un filtro de Kalman?
2. ¿Qué frecuencia mínima de muestreo exige una señal de 200 Hz?
3. ¿Por qué se prefieren los IIR dentro de los lazos?
4. ¿Qué resolución tiene una FFT de 2048 puntos a 10 kHz?

Respuestas. 1. La varianza del ruido de medición. **2.** Más de 400 Hz. **3.** Porque tienen menos retardo y cálculo que un FIR equivalente. **4.** Unos 4,9 Hz.