Curso de Sistemas y Control en Mecatrónica — Guía de Estudio Video 04

Módulo del Curso: Módulo 4 — Filtros de Kalman como Observadores de Estado y Procesamiento Digital de Señales (DSP)
Enfoque de Ingeniería: Observador de Luenberger vs. Filtro de Kalman Recursivo (Predicción y Corrección), EKF/UKF, Filtros Digitales FIR/IIR Biquad, FFT y Hardware DSP
Plataforma Oficial: `cursosmecatronica.com`

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - C\cdot\hat{x}_{k|k-1}) \mid K_k = P_{k|k-1} C^T (C P_{k|k-1} C^T + R)^{-1} \mid H_{biquad}(z) = (b0 + b1\, z^{-1} + b2\, z^{-2})/(1 + a1\, z^{-1} + a2\, z^{-2})$$

1. Síntesis de Conceptos Clave y Reglas de Diseño

- Observadores deterministas de Luenberger frente al Filtro de Kalman óptimo estocástico bajo covarianzas de proceso Q y de medición R.
- Algoritmo recursivo de Kalman en dos etapas (Time Update y Measurement Update), fusión sensorial IMU + Encoder, estado aumentado de carga y variantes no lineales EKF (Jacobianos) y UKF (puntos sigma).
- Procesamiento Digital de Señales (DSP) en mecatrónica: teorema de Nyquist-Shannon, prevención de Aliasing, filtros digitales FIR (fase lineal) e IIR en cascada Biquad.
- Análisis espectral en tiempo real mediante FFT con ventaneo Hann/Hamming e implementación embebida con instrucciones MAC en un ciclo, punto fijo Q15/Q31 y DMA Ping-Pong.

2. Parámetros Nominales de Sistemas y Control (Video 04)

Parámetro de Ingeniería	Símbolo	Valor Nominal	Función en el Sistema de Control Mecatrónico
Periodo de Muestreo Digital	`Ts [ms]`	0.25 ms	Intervalo determinista de discretización ZOH y ejecución de Kalman/LQR
Frecuencia Natural y Amortiguamiento	`ωn / ζ`	150 rad/s / 0.72	Ubicación de polos dominantes en lazo cerrado

Parámetro de Ingeniería	Símbolo	Valor Nominal	Función en el Sistema de Control Mecatrónico
Márgenes de Fase y Ganancia	`PM / GM`	60° / 12 dB	Márgenes de estabilidad robusta en respuesta en frecuencia
Covarianzas del Filtro de Kalman	`Q / R`	Q=0.00002, R=0.0008	Matrices de incertidumbre de proceso y de ruido de sensores

3. Desarrollo Académico Diapositiva por Diapositiva

Diapositiva 01: Observadores de Estado Deterministas de Luenberger y el Problema del Ruido

Dinámica del Error de Estimación e[k+1] = (Φ - L·C)e[k] y el Compromiso entre Velocidad de Convergencia y Amplificación de Ruido

Turno 1 (Masculina): Cuando implementamos un controlador por realimentación del vector de estado pero solo medimos una fracción de las variables mediante sensores físicos, construimos en el microcontrolador un sistema virtual en paralelo llamado observador de estado.

Turno 2 (Femenina): El observador determinista de Luenberger simula en tiempo real el modelo de la planta y corrige su estado estimado sumando la innovación, que es la diferencia entre la salida medida real y la salida predicha, multiplicada por una matriz de ganancia L.

Turno 3 (Masculina): La dinámica del error de estimación queda gobernada por los autovalores de la matriz Fi menos L por C; si el sistema es observable, podemos ubicar dichos polos arbitrariamente rápidos.

Turno 4 (Femenina): Sin embargo, aumentar en exceso la ganancia L para acelerar el observador amplifica drásticamente el ruido electrónico de los sensores, planteando la pregunta fundamental que resuelve el Filtro de Kalman: ¿cuál es la ganancia estadísticamente óptima?

Diapositiva 02: Filtros de Kalman como Observadores Dinámicos de Estado: Modelo Estocástico

Ruido de Proceso w_k ~ N(0, Q) e Incertidumbre del Modelo frente a Ruido de Medición v_k ~ N(0, R) de los Sensores

Turno 1 (Masculina): Formulado por Rudolf Kalman en mil novecientos sesenta, el Filtro de Kalman es el observador óptimo en el sentido de mínima varianza del error para sistemas

dinámicos lineales sometidos a perturbaciones aleatorias gaussianas.

Turno 2 (Femenina): El modelo estocástico en espacio de estados incorpora dos vectores de ruido blanco de media cero: el ruido de proceso w sub k con matriz de covarianza Q, y el ruido de medición v sub k con matriz de covarianza R.

Turno 3 (Masculina): La matriz Q cuantifica nuestra desconfianza en el modelo matemático debido a fuerzas no modeladas o discretización, mientras que la matriz R caracteriza la varianza experimental del ruido de cada sensor obtenida en laboratorio.

Turno 4 (Femenina): A partir de estas dos matrices de covarianza, el filtro pondera dinámicamente en cada ciclo de reloj cuánto confiar en la predicción física del modelo frente a cuánto confiar en la nueva lectura ruidosa de los sensores.

Diapositiva 03: Algoritmo Recursivo del Filtro de Kalman: Predicción y Actualización

Ciclo en Tiempo Real de Propagación a Priori ($\hat{x}^-$, P^-) y Corrección Bayesiana a Posteriori (K_k, $\hat{x}^+$, P^+)

Turno 1 (Masculina): La extraordinaria eficiencia computacional del Filtro de Kalman para sistemas embebidos radica en su estructura recursiva de dos etapas por cada periodo de muestreo, sin necesidad de almacenar el historial pasado de mediciones.

Turno 2 (Femenina): En la primera etapa, llamada Predicción o actualización temporal, el algoritmo proyecta hacia adelante el estado estimado a priori empleando el modelo dinámico y propaga la matriz de covarianza del error P sumando la incertidumbre Q.

Turno 3 (Masculina): En la segunda etapa, llamada Corrección o actualización de medición, se calcula primero la Ganancia de Kalman K sub k y luego se actualiza el estado a posteriori combinando la predicción con el residuo de medición.

Turno 4 (Femenina): Finalmente se contrae la matriz de covarianza del error P a posteriori, reflejando que la incorporación de una nueva medida válida de los sensores ha reducido la incertidumbre sobre el estado real de la máquina.

Diapositiva 04: Ganancia de Kalman Estacionaria, Fusión Sensorial y Estimación de Perturbaciones

Ecuación Algebraica de Riccati Discreta (DARE), Fusión IMU + Encoder y Estado Aumentado para Observador de Carga

Turno 1 (Masculina): En sistemas lineales invariantes en el tiempo con covarianzas Q y R constantes, la matriz P y la ganancia K sub k convergen rápidamente hacia valores

estacionarios que pueden precalcularse resolviendo la ecuación de Riccati.

Turno 2 (Femenina): En mecatrónica, el Filtro de Kalman brilla en la fusión sensorial: por ejemplo, al combinar un giroscopio de alta respuesta pero con deriva lenta junto con un acelerómetro o un encoder óptico sin deriva pero con ruido de alta frecuencia.

Turno 3 (Masculina): Otra aplicación industrial poderosa consiste en aumentar el vector de estado añadiendo como variable adicional una fuerza o par de perturbación externa desconocida modelada como constante a trozos.

Turno 4 (Femenina): De este modo, el Filtro de Kalman estima simultáneamente la posición, la velocidad limpia sin derivar numéricamente y el par de carga externo, permitiendo cancelar la perturbación de manera inmediata en el controlador.

Diapositiva 05: Filtro de Kalman Extendido (EKF) y Unscented (UKF) en Mecatrónica No Lineal

Linealización Local mediante Matrices Jacobianas F_k y H_k vs. Propagación Determinista de Puntos Sigma en Robótica

Turno 1 (Masculina): Cuando la cinemática o la dinámica del sistema mecatrónico es fuertemente no lineal, como ocurre en brazos robóticos articulados, drones o vehículos autónomos, las ecuaciones de estado dejan de ser matrices constantes.

Turno 2 (Femenina): El Filtro de Kalman Extendido, conocido como E K F, propaga el estado empleando directamente las funciones no lineales exactas, pero linealiza el sistema alrededor de la estimación actual en cada instante de muestreo.

Turno 3 (Masculina): Para ello evalúa las matrices Jacobianas de derivadas parciales F sub k y H sub k, actualizando la matriz de covarianza P con un costo computacional muy reducido apto para microcontroladores de treinta y dos bits.

Turno 4 (Femenina): Cuando las no linealidades son tan pronunciadas que el Jacobiano de primer orden introduce errores de sesgo, el Filtro de Kalman Unscented o U K F propaga un conjunto determinista de puntos sigma, logrando precisión de tercer orden sin calcular derivadas.

Diapositiva 06: Procesamiento Digital de Señales (DSP) en Mecatrónica: Muestreo y Aliasing

Teorema de Nyquist-Shannon (fs > 2·fmax), Plegamiento Espectral, Filtros Analógicos Antialiasing y Sobremuestreo

Turno 1 (Masculina): El Procesamiento Digital de Señales o D S P para aplicaciones mecatrónicas acondiciona las señales crudas de vibración, corriente y posición antes de que ingresen al algoritmo de control o de diagnóstico.

Turno 2 (Femenina): El Teorema de Muestreo de Nyquist-Shannon establece que la frecuencia de muestreo f sub s debe ser estrictamente mayor que el doble de la componente frecuencial más alta presente en la señal analógica.

Turno 3 (Masculina): Si existen ruidos de conmutación PWM o vibraciones por encima de la frecuencia de Nyquist f sub s medios, ocurre el fenómeno irreversible de Aliasing: dichas frecuencias altas se pliegan y aparecen disfrazadas como oscilaciones falsas de baja frecuencia.

Turno 4 (Femenina): Por ello, toda etapa de adquisición mecatrónica combina un filtro analógico pasabajos antialiasing previo al conversor A D C con técnicas de sobremuestreo y diezmado digital dentro del procesador.

Diapositiva 07: Diseño de Filtros Digitales FIR e IIR para Lazos de Control y Sensores

Fase Lineal Exacta vs. Mínimo Retardo de Fase: Estructuras Biquad Direct Form II, Butterworth y Filtros Notch Digitales

Turno 1 (Masculina): Los filtros digitales en tiempo discreto se dividen en dos grandes arquitecturas: filtros de Respuesta Finita al Impulso o F I R, y filtros de Respuesta Infinita al Impulso o I I R.

Turno 2 (Femenina): Los filtros F I R carecen de realimentación de salidas pasadas, por lo que son incondicionalmente estables y pueden diseñarse con simetría de coeficientes para garantizar fase exactamente lineal, ideal en visión y análisis de ondas.

Turno 3 (Masculina): Sin embargo, dentro de un lazo de control cerrado en tiempo real, un filtro F I R de orden alto introduce demasiadas muestras de retardo; allí preferimos filtros I I R de bajo orden derivados de prototipos analógicos Butterworth o Chebyshev.

Turno 4 (Femenina): Para evitar inestabilidades numéricas por cuantificación de coeficientes, implementamos los filtros I I R de orden superior como una cascada de secciones de segundo orden o Biquads, incluyendo filtros Notch que anulan resonancias mecánicas puntuales.

Diapositiva 08: Análisis Espectral en Tiempo Real: DFT, Algoritmo FFT y Ventaneo Espectral

Complejidad O(N log2 N), Fuga Espectral (Spectral Leakage), Ventanas de Hann/Hamming y Diagnóstico de Rodamientos y Motores

Turno 1 (Masculina): Para identificar experimentalmente frecuencias de resonancia estructural o detectar fallas incipientes en engranajes y rodamientos, transformamos bloques de muestras temporales al dominio de la frecuencia mediante la Transformada Discreta de Fourier.

Turno 2 (Femenina): El algoritmo de la Transformada Rápida de Fourier o F F T de Cooley-Tukey reduce la complejidad computacional de N al cuadrado a N por el logaritmo base dos de N, permitiendo ejecutar espectros de mil veinticuatro puntos en pocos milisegundos.

Turno 3 (Masculina): Dado que truncar una señal continua en una ventana finita de observación produce discontinuidades en los bordes que esparcen energía falsa por todo el espectro, fenómeno conocido como fuga espectral...

Turno 4 (Femenina): ...multiplicamos previamente el bloque temporal por una ventana suave como Hann, Hamming o Flat-Top, logrando aislar con total nitidez los armónicos eléctricos y mecánicos de la máquina.

Diapositiva 09: Arquitectura e Implementación DSP Embebida: MAC, Punto Fijo Q15/Q31 y DMA

Ejecución Determinista sin Bloqueo de CPU mediante Ping-Pong DMA, Saturación por Hardware y Librerías CMSIS-DSP

Turno 1 (Masculina): Llevar algoritmos de filtrado digital y observadores de Kalman a frecuencias de decenas de kilohertz exige aprovechar la arquitectura especializada de los procesadores digitales de señales y microcontroladores con extensión D S P.

Turno 2 (Femenina): El corazón aritmético es la instrucción de Multiplicación y Acumulación o M A C en un solo ciclo de reloj, acompañada de direccionamiento circular en hardware y aritmética con saturación automática que evita el desbordamiento.

Turno 3 (Masculina): Cuando el procesador dispone de unidad de punto flotante F P U operamos en precisión simple de treinta y dos bits; en núcleos enteros empleamos aritmética fraccionaria de punto fijo Q quince o Q treinta y uno rigurosamente escalada.

Turno 4 (Femenina): Además, acoplamos el conversor A D C al controlador de Acceso Directo a Memoria o D M A en configuración de doble buffer Ping-Pong, de modo que

mientras el hardware adquiere un bloque de muestras, la C P U procesa el bloque anterior con cero latencia.

Diapositiva 10: Síntesis del Módulo 4 y Transición al Diseño por Optimización H2 y Control No Lineal

Integración de Observadores Estocásticos de Kalman y Procesamiento Digital de Señales de Alta Velocidad

Turno 1 (Masculina): En este cuarto módulo desarrollamos la teoría y programación de los Filtros de Kalman como observadores dinámicos de estado, abarcando desde el filtro lineal estacionario hasta las variantes E K F y U K F.

Turno 2 (Femenina): Asimismo, estructuramos la cadena completa de Procesamiento Digital de Señales en mecatrónica: muestreo antialiasing, filtros F I R e I I R en cascada Biquad, análisis espectral F F T y optimización en hardware D S P.

Turno 3 (Masculina): Contando ahora con estimaciones óptimas del vector de estado y señales digitales libres de ruido, estamos listos para abordar las estrategias de control más avanzadas.

Turno 4 (Femenina): En el Módulo cinco estudiaremos el diseño de sistemas de control vía optimización H dos y las técnicas de control adaptativo y no lineal.

4. Firmware C++/MCU: Filtro de Kalman, Biquad DSP, LQR (H2) y Modos Deslizantes (control_sistemas_mecatronica_video_04.ino)

```
/*
 * ========================================================================
 * CURSO: SISTEMAS Y CONTROL EN MECATRÓNICA (GEMINI — cursosmecatronica.com)
 * VIDEO 04: Video 04 — Filtros de Kalman como Observadores de Estado y Procesamiento Digital de Señales (DSP)
 * Enfoque: Observador de Luenberger vs. Filtro de Kalman Recursivo (Predicción y Corrección), EKF/UKF, Filtros Digitales FIR/IIR Biquad, FFT y Hardware DSP
 * Ecuación Clave: x̂_k|k = x̂_k|k-1 + K_k(y_k - C·x̂_k|k-1)  |  K_k = P_k|k-1 C^T (C P_k|k-1 C^T + R)⁻¹  |  H_biquad(z) = (b0 + b1 z⁻¹ + b2 z⁻²)/(1 + a1 z⁻¹ + a2 z⁻²)
 * ========================================================================
 */
#include <Arduino.h>

// Parámetros de tiempo real y control en espacio de estados discreto
static const float TS_SEC = 0.0003f;
static const uint32_t TS_US = 250UL;

// Estado estimado por Filtro de Kalman [posición x1, velocidad x2, perturbación d]
```

```
static float x_hat[3] = {0.0f, 0.0f, 0.0f};
static float P_cov[2][2] = {{1.0f, 0.0f}, {0.0f, 1.0f}};
static const float Q_PROC = 0.00002f;
static const float R_MEAS = 0.0008f;

// Ganancias óptimas LQR (H2) + Superficie de Modos Deslizantes (SMC)
static const float K_LQR_POS = 67.50f;
static const float K_LQR_VEL = 9.36f;
static const float LAMBDA_SMC = 52.50f;
static const float K_SW_SMC = 2.80f;
static const float PHI_BOUND = 0.05f;

// Filtro digital IIR Biquad Notch (supresión de resonancia mecánica/hidráulica)
struct BiquadFilter {
  float b0, b1, b2, a1, a2;
  float w1, w2;
};
static BiquadFilter notchFilter = {0.94f, -1.78f, 0.94f, -1.78f, 0.88f, 0.0f, 0.0f};

static inline float stepBiquad(BiquadFilter &f, float x) {
  float w0 = x - f.a1 * f.w1 - f.a2 * f.w2;
  float y  = f.b0 * w0 + f.b1 * f.w1 + f.b2 * f.w2;
  f.w2 = f.w1;
  f.w1 = w0;
  return y;
}

static inline float satBoundary(float s, float phi) {
  if (s > phi) return 1.0f;
  if (s < -phi) return -1.0f;
  return s / phi;
}

void kalmanObserverStep(float u_prev, float y_meas) {
  // 1. Predicción temporal: x_hat^- = Phi * x_hat + Gamma * u
  const float b_fric = 1.25f;
  const float k_gain = 18.5f;
  float x1_pred = x_hat[0] + TS_SEC * x_hat[1];
  float x2_pred = x_hat[1] + TS_SEC * (-b_fric * x_hat[1] + k_gain * u_prev - x_hat[2]);

  // Propagación de covarianza P^- = Phi * P * Phi^T + Q
  P_cov[0][0] += TS_SEC * (2.0f * P_cov[0][1] + TS_SEC * P_cov[1][1]) + Q_PROC;
  P_cov[0][1] += TS_SEC * P_cov[1][1];
  P_cov[1][0]  = P_cov[0][1];
  P_cov[1][1] += Q_PROC * 10.0f;

  // 2. Actualización por medición (Innovación y Ganancia de Kalman K_k)
  float innov = y_meas - x1_pred;
  float S_inv = 1.0f / (P_cov[0][0] + R_MEAS);
  float K0 = P_cov[0][0] * S_inv;
  float K1 = P_cov[1][0] * S_inv;

  x_hat[0] = x1_pred + K0 * innov;
  x_hat[1] = x2_pred + K1 * innov;
  x_hat[2] -= 0.08f * innov; // Estimación adaptativa de carga externa
```

```
  float p00 = P_cov[0][0], p01 = P_cov[0][1];
  P_cov[0][0] -= K0 * p00;
  P_cov[0][1] -= K0 * p01;
  P_cov[1][0] -= K1 * p00;
  P_cov[1][1] -= K1 * p01;
}

void setup() {
  Serial.begin(115200);
  pinMode(9, OUTPUT);
  pinMode(10, OUTPUT);
}

void loop() {
  static uint32_t lastTickUs = 0;
  static float u_cmd = 0.0f;
  uint32_t nowUs = micros();

  if ((uint32_t)(nowUs - lastTickUs) >= TS_US) {
    lastTickUs += TS_US;

    // Lectura de referencia y posición + filtrado DSP Biquad
    float ref_pos = (analogRead(A0) - 512) * 0.01f;
    float raw_pos = (analogRead(A1) - 512) * 0.01f;
    float y_clean = stepBiquad(notchFilter, raw_pos);

    // Estimación óptima de estados con Filtro de Kalman
    kalmanObserverStep(u_cmd, y_clean);

    // Ley de control híbrida: LQR (H2) + Modo Deslizante Suavizado (SMC)
    float e_pos = x_hat[0] - ref_pos;
    float e_vel = x_hat[1];
    float s_surf = e_vel + LAMBDA_SMC * e_pos;

    float u_lqr = -K_LQR_POS * e_pos - K_LQR_VEL * e_vel;
    float u_smc = -K_SW_SMC * satBoundary(s_surf, PHI_BOUND);
    u_cmd = constrain(u_lqr + u_smc + (x_hat[2] / 18.5f), -12.0f, 12.0f);

    int pwmOut = constrain((int)(fabsf(u_cmd) * (255.0f / 12.0f)), 0, 255);
    digitalWrite(10, (u_cmd >= 0.0f) ? HIGH : LOW);
    analogWrite(9, pwmOut);
  }
}
```

5. Simulación Python: Espacio de Estados, Riccati LQR/H2 y Observador de Kalman (simulacion_sistemas_y_control_video_04.py)

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
================================================================================
```

```
CURSO: SISTEMAS Y CONTROL EN MECATRÓNICA (GEMINI — cursosmecatronica.com)
VIDEO 04: Video 04 — Filtros de Kalman como Observadores de Estado y Procesamiento Digital de Señales
(DSP)
Simulación de Espacio de Estados, LQR/H2, Filtro de Kalman, Eje Electrohidráulico
y Optimización Multiobjetivo de Pareto.
================================================================================
"""
import math

TS = 0.0003
WN = 150
ZETA = 0.72
Q_KALMAN = 0.00002
R_KALMAN = 0.0008

def simulate_state_space_lqr_kalman(steps=400, ref=1.0):
    # Estado real [pos, vel] y estado estimado por Filtro de Kalman
    x = [0.0, 0.0]
    x_hat = [0.0, 0.0]
    p00, p01, p11 = 1.0, 0.0, 1.0

    # Ganancias LQR óptimas (síntesis H2)
    k1 = WN * 0.55
    k2 = 2.0 * ZETA * math.sqrt(WN)
    history = []

    for k in range(steps):
        y_meas = x[0]  # Medición del sensor
        # Actualización de Kalman
        innov = y_meas - x_hat[0]
        s_cov = p00 + R_KALMAN
        k_kal0 = p00 / s_cov
        k_kal1 = p01 / s_cov
        x_hat[0] += k_kal0 * innov
        x_hat[1] += k_kal1 * innov

        # Ley de control LQR + compensación no lineal de fricción
        e0 = x_hat[0] - ref
        e1 = x_hat[1]
        u = max(-24.0, min(24.0, -k1 * e0 - k2 * e1))

        # Integración de la planta mecatrónica / electrohidráulica
        acc = -2.0 * ZETA * WN * x[1] - (WN ** 2) * (x[0] - ref) * 0.05 + 15.0 * u
        x[0] += TS * x[1]
        x[1] += TS * acc

        # Predicción de Kalman
        x_hat[0] += TS * x_hat[1]
        x_hat[1] += TS * (-2.0 * ZETA * WN * x_hat[1] + 15.0 * u)
        p00 += TS * (2.0 * p01 + TS * p11) + Q_KALMAN
        p01 += TS * p11
        p11 += Q_KALMAN * 5.0

        if k % 50 == 0:
            history.append((round(k * TS, 4), round(x[0], 4), round(x_hat[0], 4), round(u, 3)))
```

```
    return history

if __name__ == "__main__":
    print("=== Simulación Sistemas y Control en Mecatrónica - Video 04 ===")
    for t_s, pos, pos_est, u_v in simulate_state_space_lqr_kalman():
        print(f"t={t_s:6.3f}s | Pos={pos:7.4f} | Kalman={pos_est:7.4f} | u={u_v:7.3f}V")
```

6. Programa PLC IEC 61131-3 (ST): Control Avanzado de Eje Electrohidráulico (plc_eje_electrohidraulico_control_video_04.st)

```
(* ================================================================================
 * CURSO: SISTEMAS Y CONTROL EN MECATRÓNICA (GEMINI — cursosmecatronica.com)
 * VIDEO 04: Video 04 — Filtros de Kalman como Observadores de Estado y Procesamiento Digital de Señales
(DSP)
 * Norma IEC 61131-3 (Structured Text): Observador de Estado Discreto,
 * Control en Cascada de Eje Electrohidráulico con Amortiguamiento Activo de Presión
 * ================================================================================ *)
PROGRAM PLC_Sistemas_Y_Control_V04
VAR_INPUT
    rRefPos_mm      : REAL := 100.0;   (* Referencia de posición [mm] *)
    rMeasPos_mm     : REAL := 0.0;     (* Sensor magnetoestrictivo / encoder [mm] *)
    rPresionA_bar   : REAL := 70.0;    (* Presión cámara A [bar] *)
    rPresionB_bar   : REAL := 70.0;    (* Presión cámara B [bar] *)
    bEnableAxis     : BOOL := TRUE;    (* Habilitación de seguridad *)
END_VAR
VAR_OUTPUT
    rValveCmd_V     : REAL := 0.0;     (* Comando a servoválvula [-10..+10 V] *)
    rPosEstimada_mm : REAL := 0.0;     (* Posición filtrada por observador *)
    rVelEstimada_mms: REAL := 0.0;     (* Velocidad estimada por Kalman [mm/s] *)
END_VAR
VAR
    rTs             : REAL := 0.0003;
    rErrorPos       : REAL := 0.0;
    rPresionCarga   : REAL := 0.0;
    rPresionAnt     : REAL := 0.0;
    rDerivPresion   : REAL := 0.0;
    rInnov          : REAL := 0.0;
    rKp_Cascada     : REAL := 4.50;
    rKv_Cascada     : REAL := 0.18;
    rKd_Presion     : REAL := 0.035;   (* Amortiguamiento hidromecánico activo *)
END_VAR

IF NOT bEnableAxis THEN
    rValveCmd_V := 0.0;
    RETURN;
END_IF;

(* 1. Observador de estado alfa-beta (Kalman estacionario) *)
rInnov := rMeasPos_mm - rPosEstimada_mm;
rPosEstimada_mm := rPosEstimada_mm + rTs * rVelEstimada_mms + 0.35 * rInnov;
rVelEstimada_mms := rVelEstimada_mms + (0.06 / rTs) * rInnov;
```

```
(* 2. Retroalimentación activa de presión de carga PL = PA - PB *)
rPresionCarga := rPresionA_bar - rPresionB_bar;
rDerivPresion := (rPresionCarga - rPresionAnt) / rTs;
rPresionAnt   := rPresionCarga;

(* 3. Ley de control avanzada con amortiguamiento activo e inversión *)
rErrorPos := rRefPos_mm - rPosEstimada_mm;
rValveCmd_V := (rKp_Cascada * rErrorPos) - (rKv_Cascada * rVelEstimada_mms) - (rKd_Presion *
rDerivPresion);

(* Saturación física de servoválvula proporcional [-10V, +10V] *)
IF rValveCmd_V > 10.0 THEN
    rValveCmd_V := 10.0;
ELSIF rValveCmd_V < -10.0 THEN
    rValveCmd_V := -10.0;
END_IF;

END_PROGRAM
```