UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

ESCUELA DE INGENIERÍA EN MECATRÓNICA

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN MECATRÓNICA

TEMA:

"ALGORITMO PARA DISMINUCIÓN DE DEGRADACIÓN DEL RENDIMIENTO EN SISTEMAS DE CONTROL CON ACTUACIÓN PERIÓDICA"

AUTOR: FAUSTO ISRAEL ORTEGA SALAS

DIRECTOR: CARLOS XAVIER ROSERO CHANDI

IBARRA-ECUADOR
AGOSTO-2021

UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	100377267-8		
APELLIDOS Y NOMBRES:	ORTEGA SALAS FAUSTO ISRAEL		
DIRECCIÓN:	NAZACOTA PUENTO 25-31		
EMAIL:	fiortegas@utn.edu.ec		
TELÉFONO FIJO:		**TELÉFONO MÓVIL:**	0980060838

DATOS DE LA OBRA	
TÍTULO:	"ALGORITMO PARA DISMINUCIÓN DE DEGRADACIÓN DEL RENDIMIENTO EN SISTEMAS DE CONTROL CON ACTUACIÓN PERIÓDICA"
AUTOR (ES):	ORTEGA SALAS FAUSTO ISRAEL
FECHA:	20/08/2021
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	[X] **PREGRADO** [ ] **POSGRADO**
TITULO POR EL QUE OPTA:	INGENIERO EN MECATRÓNICA
ASESOR /DIRECTOR:	XAVIER ROSERO

2. CONSTANCIAS

El autor (es) manifiesta (n) que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 20 días del mes de agosto de 2021

EL AUTOR:

Fausto Ortega

UNIVERSIDAD TÉCNICA DEL NORTE
FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS
CERTIFICACIÓN

En calidad de director del trabajo de grado “ALGORITMO PARA DISMINUCIÓN DE DEGRADACIÓN DEL RENDIMIENTO EN SISTEMAS DE CONTROL CON ACTUACIÓN PERIÓDICA", presentado por el egresado FAUSTO ISRAEL ORTEGA SALAS, para optar por el título de Ingeniero en Mecatrónica, certifico que el mencionado proyecto fue realizado bajo mi dirección.

Ibarra, Agosto del 2021


Firmado electrónicamente por:
CARLOS XAVIER
ROSERO CHANDI

Carlos Xavier Rosero C.
DIRECTOR DE TESIS

Agradecimientos

A mis padres y hermanos por guiarme y apoyarme a lo largo de toda mi vida y en especial en esta etapa tan importante como es mi formación profesional.

A mi esposa Silvana Armas y mi hijo Fausto Ortega por estar presente en toda mi formación con su cariño comprensión y cuidado, apoyándome y siendo el pilar emocional para superar cualquier obstáculo.

A mis compañeros de equipo por el apoyo brindado y las experiencias vividas dentro y fuera de nuestra formación profesional.

A mi director de este trabajo Xavier Rosero por guiarme con sus conocimientos para lograr culminar este proceso.

Fausto Ortega

Dedicatoria

Dedico el presente trabajo a mis amados padres quienes con amor, paciencia, esfuerzo y sacrificio, lograron educar a sus hijos, olvidándose de si mismos y sus necesidades propias. A mis hermanos que siempre estuvieron junto a mi dando fuerza y apoyo en cualquier circunstancia para lograr obtener este Título.

Fausto Ortega

Resumen

Los sistemas de control modernos son implementados en sistemas microprocesados que se desempeñan en tiempo real, ejecutando simultáneamente múltiples tareas. Esto provoca que el rendimiento de la acción de control pueda verse deteriorado por problemas conocidos como: retardos en los tiempo de muestreo/actuación, latencias de entrada/salida y la incertidumbre provocada por el ruido en el sistema. Como resultado, la acción de control es propensa a la degradación de su rendimiento y no cumple con la demanda de periodicidad.

Existen trabajos realizados para enfrentar esta problemática, pero estos requieren que los retardos de control sean constantes. Estos requerimientos se vuelven difíciles de conseguir debido a la latencia existente entre la entrada y salida, además del ruido en el sistema.

Contribuyendo a esta problemática se presenta un algoritmo de control, el cual basa su funcionamiento en la sincronización de los instantes de actuación para afrontar los retardos de tiempos y las latencias existentes entre la entrada y salida. Además, incorpora al modelo un filtro de Kalman para reducir la incertidumbre provocada por el ruido en el sistema. Los resultados obtenidos en las simulaciones confirman que el modelo propuesto presenta un rendimiento adecuado, frente a otros modelos de control.

Abstract

Modern control systems are implemented in microprocessor systems that perform in real time, simultaneously executing multiple tasks. This causes the performance of the control action to be degraded by known problems such as: sample/actuation time delays, input/output latencies and uncertainty caused by noise in the system. As a result, the control action is prone to performance degradation and does not meet the periodicity demand.

Work has been done to address this problem, but it requires that the control delays be constant. These requirements become difficult to achieve due to the latency between input and output, in addition to the noise in the system.

Contributing to this problem, a control algorithm is presented, which bases its operation on the synchronization of the actuation instants to cope with the time delays and latencies between input and output. It also incorporates a Kalman filter to reduce the uncertainty caused by noise in the system. The results obtained in the simulations confirm that the proposed model presents an adequate performance, compared to other control models.

Índice general

Índice de figuras

Introducción

En este capitulo se presenta el planteamiento del problema, el alcance y los objetivos planteados para resolver el mismo.

Problema

El uso de sistemas controlados por computador tiene un incremento drástico en la vida diaria. Procesador y microcontroladores son incrustados cada vez más en los dispositivos que se usan en la cotidianidad. Debido a las restricciones de costo, muchos de estos dispositivos que corren aplicaciones de control son diseñados con bajo espacio, peso y restricciones de energía.[1]

El desarrollo de algoritmos de control efectivo para varios sistemas dinámicos sigue siendo relevante para la teoría de control moderna.[2] Durante las últimas décadas, la programación del tiempo de procesador ha sido un área de investigación muy activa y se han desarrollado varios métodos y modelos de programación diferentes.[3]

Las características de tiempo real en tareas con comportamiento dinámico, junto con restricciones de costo y recursos, crea nuevos problemas que deben abordarse en el diseño de dichos sistemas, en diferentes niveles de arquitectura[4]. Algunos investigadores sugieren que se deben usar nuevas formas de control para distribuir los recursos adecuadamente en base a regulaciones de control orientadas a los recursos como el control disparado por eventos y el control autodisparado[5].

Para muchos sistemas de control, el rendimiento depende en gran medida de las variaciones de retardo en las tareas de control. Dichas variaciones pueden provenir de numerosas fuentes, incluidas la prioridad de las tareas, las variaciones en las cargas de trabajo de las tareas, errores de medición y las perturbaciones en el entorno físico, y pueden causar un rendimiento del sistema de control degradado, como una respuesta lenta y un comportamiento erróneo[6]. Además las propiedades de los algoritmos de programación en tiempo real pueden causar respuestas inesperadas del sistema de control en la implementación de sistemas controlados por procesador en tiempo real. Después de que una tarea ha sido lanzada, tiene que retazar su inicio de ejecución, la acción de control también puede ser remplazada o bloqueada al intentar acceder a recursos compartidos

del procesador, esto significa que los instantes de tiempo de ejecución de la tarea de control no son equidistas en el tiempo de ejecución[7].

Alcance

En el presente proyecto se considerará un modelo en espacio de estados que represente cualquier tipo de planta, donde las matrices de la dinámica se encuentran en función del tiempo de muestreo y el retardo. Ante esto se propone un modelo que considere que el tiempo de muestreo no se mida entre tomas de datos de los sensores sino entre puntos de actuación manteniendo la periodicidad, donde el modelo en espacio de estados discreto depende del tiempo de muestreo, de los retardos y el tiempo entre actuación, permitiendo que a pesar de que existan retardos siempre se tenga un buen desempeño, es decir robustez. Se evaluará el modelo de control con simulaciones a través de software matemático.

Objetivos

Objetivo General

Diseñar un algoritmo para disminución de degradación del rendimiento en sistemas de control considerando actuación periódica.

Objetivos Específicos

- Proponer un modelo de control para disminución de degradación del rendimiento en sistemas de control, basado en la literatura.
- Desarrollar un algoritmo para la implementación del modelo de control.
- Evaluar el desempeño del método propuesto.

Justificación

Los sistemas de control digital constituyen una gran parte de todos los sistemas en tiempo real. A pesar de esto, sorprendentemente se ha hecho poco esfuerzo para estudiar su comportamiento oportuno cuando se implementa como tareas periódicas en su computador. Para muchos sistemas ciber-físicos, la coordinación de inteligencia entre el diseño de control y la implementación

de su computadora correspondiente pueden llevar a su mejor rendimiento de control y/o una reducción de costos. La mayoría de los dispositivos integrados interactúan con el entorno y tienen especificaciones de calidad exigentes, cuya satisfacción requiere que el sistema reaccione de manera oportuna a eventos externos y ejecute actividades computacionales dentro de restricciones de tiempo precisas.

Los sistemas de control basados en computadora utilizan técnicas en tiempo real para resolver problemas reales de ingeniería en mecatrónica. Los fundamentos teóricos obtenidos se convierten en las bases del diseño robótico y mecatrónico brindando la capacidad de análisis y razonamiento matemático para detectar, analizar y resolver problemas de ingeniería que involucren la mecánica, la microelectrónica, la robótica y biomecánica , desarrollando habilidades para una actualización permanente a lo largo del ejercicio profesional.

Capítulo 1

Revisión Literaria

En este capítulo se realiza una revisión de la teoría referente al control automático, técnicas de control y el filtro de Kalman .

1.1. Control automático

Un computador dedicado al control de procesos realiza funciones de control y supervisión. En este tipo de aplicaciones comparte sus recursos entre un cierto número de funciones de supervisión y control en un tiempo estricto de ejecución (hard-real-time). Con un flujo de otras funciones las cuales su tiempo de ejecución no es estrictamente crítico(soft-real-time). Donde una distribución estadística de los tiempos de respuesta es aceptable. En determinadas aplicaciones, no existen trabajos que no sean de tiempo de ejecución estricto y el uso eficiente del computador se puede lograr mediante una cuidadosa programación de las funciones de control[8].

Aplicando técnicas de control, podemos obtener una eliminación parcial del error en los procesos y un aumento considerable en la seguridad de los sistemas. Estos son puntos importantes que destacan la aplicación de las teorías de control en la industria.

A diferencia del control clásico, el procesador es la herramienta principal de un sistema de control moderno. Este permite afrontar problemas tanto SISO (single-output single-input) y MIMO (multiple-input y multiple-output). Dado que su análisis se realiza en la representación en espacio de estados de un sistema y el principio de realimentación de estados, esta operación permite al controlador mantener actualizado con la información de las variables de estado. Con esta información realiza correcciones al proceso cuando sea necesario.

1.2. Tareas de control

En la teoría de control, el muestreo y la actuación se asumen sincrónicos y periódicos, y se considera un cumplimiento preciso en su implementación. Cuando un algoritmo de control es ejecutado por una tarea (o por un conjunto de subtareas) en un sistema en tiempo real, esas suposiciones no se cumplen. Esto provoca una degradación del rendimiento de control e incluso inestabilidad. Además, el algoritmo de programación puede restringir al sistema cuando intenta cumplir con las estrictas restricciones de tiempo que exige la teoría de control, lo que da como resultado una programación deficiente[9].

Una tarea de control se basa en tres partes principales: lectura de datos desde los sensores, cálculo del algoritmo de control y transmisión de salida. En la mayoría de los casos, el control se ejecuta de forma periódica, con un período de muestreo constante determinado por la dinámica del proceso y los requisitos sobre el rendimiento del lazo cerrado. Las restricciones de tiempo básicas de una tarea de control se muestran en la Figura 1.1. La primera es el período que debe ser constante, es decir, sin fluctuación. La segunda restricción involucra la latencia de entrada-salida, también conocida como retardo de control. Esto debe ser lo más pequeño posible y también sin fluctuaciones. Desde una perspectiva de control, las fluctuaciones de fase de muestreo y actuación pueden ser interpretadas como perturbaciones que actúan sobre el sistema de control[3].

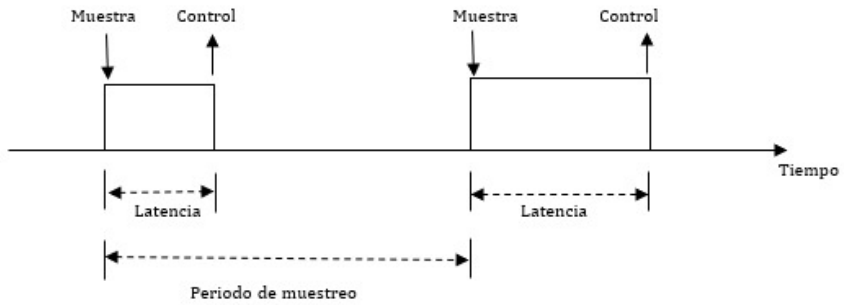

Figura 1.1: Restricciones de tiempo.

La latencia que se encuentra en la entrada y salida del sistema disminuye el margen de estabilidad y limita el rendimiento del mismo.

Al utilizar estrategias de control adecuadas que tengan en cuenta estos requerimientos, el rendimiento del sistema puede mejorar drásticamente. Utilizando plenamente las estrategias de diseño de control más adecuadas, el algoritmo de programación tendrá en algún momento que proporcionar información para cada instante de la tarea de control. Además, estas estrategias de diseño de control proporcionarán nuevas restricciones para cada instancia de la tarea de control, lo que permitirá una mejor programación de todo el conjunto de tareas (tareas de control y de no control)[9].

1.3. Controladores estándar

Los algoritmos de control estándar al implementarse como tareas periódicas en tiempo real, consideran un período de tarea igual al período de muestreo. Las operaciones de muestreo (entrada) y actuación (salida) se especifican para que ocurran al principio y al final de cada ejecución de trabajo[1].

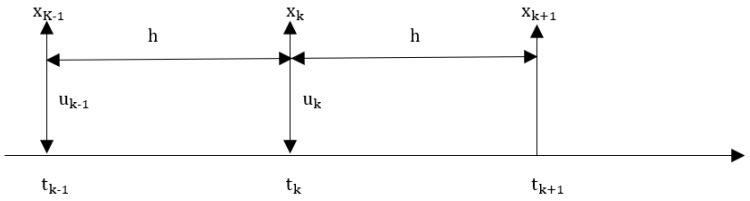

Figura 1.2: Controlador estándar.

El muestreo debe realizarse en el mismo instante de cada período(h), lo que significa un período de muestreo constante. El cálculo del algoritmo de control debe comenzar y terminar tan pronto como sea posible después de que la muestra esté disponible. La actuación debe realizarse inmediatamente después del cálculo del algoritmo, o en un instante fijo después del muestreo, lo que significa un intervalo de actuación constante (dependiendo de cómo se diseñó el controlador). De esta forma las partes de un bucle de control (muestreo, cálculo, actuación) se consideran instantáneas, como se muestra en la Figura 1.2.

El modelo en espacio de estados de un sistema lineal invariante en el tiempo, de tiempo continuo representado en espacio de estados es,

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{1.1}$$

$$y(t) = Cx(t), \tag{1.2}$$

donde $A \in \mathbb{R}^{nxn}$ y $B \in \mathbb{R}^{nxm}$ son matrices que describe la dinámica del sistema y $C \in \mathbb{R}^{pxn}$ es la matriz de salida del sistema. Utilizando técnicas de discretización habituales como,

$$\phi(t) = e^{At} \tag{1.3}$$

$$\Gamma(t) = \int_0^t e^{A(s)} B dt, \tag{1.4}$$

se obtiene la representación del sistema en espacio de estados en tiempo discreto,

$$x_{k+1} = \phi(h)x_k + \Gamma(h)u_k \tag{1.5}$$

$$y_k = Cx_k, \tag{1.6}$$

donde x_k son los estados del sistema u_k e y_k son la entrada y salida del sistema respectivamente.

La señal de control será calculada dentro de cada secuencia de trabajo con una tarea periódica identificada k, como:

$$u_k = Lx_k, \tag{1.7}$$

donde $L \in \mathbb{R}^{1xn}$ es el estado de realimentación, obtenido por métodos de diseño de control.

Cuando se programa un sistema en tiempo real debido a retardos o latencias introducidas, el muestreo no puede ser constante. El cálculo del algoritmo de control puede comenzar más tarde que el instante en el que la muestra está disponible y el cálculo del algoritmo de control puede no ser instantáneo e incluso tener un tiempo de cálculo variable. En la realidad existen retardos de muestreo lo cual implica que la separación entre muestras consecutivas no sea constante. Para evitar la degradación que introducen estos retardos de muestreo, podemos compensar utilizando estrategias de control más adecuadas como proponen en [9].

1.4. Tarea de control flexible en tiempo real

En [10] presentan un modelo en espacio de estados para tareas de control en tiempo real destinadas a resolver las deficiencias mencionadas anteriormente. Este modelo se basa en predicciones que implican calcular la señal de control. Utilizando un vector de estados actualizado (estimado) en el instante de actuación. El cual hace posible eliminar el retardo de tiempo variable entre el muestreó y la actuación. Además el modelo permite instantes de muestreo irregulares, forzando al instante de actuación como único punto de sincronización.

Considerando un modelo en espacio de estados de un sistema en tiempo continuo e invariante en el tiempo con retardo de tiempo τ,

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t-\tau) \\ y(t) &= Cx(t), \end{aligned} \tag{1.8}$$

donde el retardo de tiempo τ, puede aparece debido al cálculo del algoritmo de control. Este puede producir una latencia de entrada/salida.

Se supone que τ es menor o igual que el periodo de muestreo h, de esta forma el modelo del

sistema en espacio de estados en tiempo discreto esta dado por,

$$\begin{aligned} x_{k+1} &= \phi(h)x_k + \phi(h-\tau)\Gamma(\tau)u_{k-1} + \Gamma(h-\tau)u_k \\ y(t) &= Cx(t). \end{aligned} \tag{1.9}$$

La señal de control para el sistema sera calculada mediante realimentación de estados, utilizando el vector de estados actualizado en el instante de actuación $t_{k+\tau}$

$$u_k = -Lx_{k+\tau}, \tag{1.10}$$

donde L es el vector de realimentación del estado el cual se puede obtener por medio de una estrategia de diseño de control, como la técnica de ubicación de polos o el regulador cuadrático óptimo.

El período de muestreo h es el tiempo transcurrido desde $t_{k+\tau}$ hasta $t_{k+1+\tau}$ y u_k se mantiene desde $t_{k+\tau}$ hasta $t_{k+1+\tau}$. Con esto la nueva dinámica del sistema en lazo cerrado se puede modelar mediante 1.1

$$x_{k+\tau} = \phi(\tau)x_k + \Gamma(\tau)u_k. \tag{1.11}$$

El modelo de lazo cerrado en tiempo discreto presentado el cual basa su funcionamiento en las ecuaciones 1.10 y 1.11, elimina los retardos de tiempo utilizando un único punto de sincronización (instante de actuación). El muestreo irregular presenta el beneficio principal de eliminar la degradación introducida por las fluctuaciones. De esta forma logra mejorar el rendimiento de la acción de control[8].

1.5. Controlador con actualización de la señal de control

El diseño de controladores digitales no debe limitarse al caso más simple con estrategias de muestreo síncronas, donde la implementación digital debe tener en cuenta el entorno operativo en tiempo real.

En lugar de aplicar controladores estándar, se puede aplicar un controlador con señal de control actualizada para tener en cuenta el retraso como proponen en [1]. Al contrario de que el cálculo de la señal de control con un vector de estado se vuelva obsoleto en el momento en que se aplica la señal de control, se utiliza un vector de estado actualizado (estimado). Por lo tanto, se calcula la señal de control en términos del estado estimado.

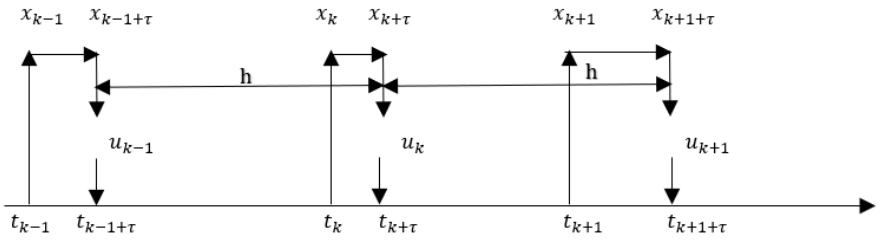

Figura 1.3: Controlador con actualización de la señal de control.

1.5.1. Actualización de la señal de control

En lugar de que el cálculo de la señal de control se obtenga a partir de un vector de estado, el cual se vuelve obsoleto en el momento de aplicar la señal de control. Se aplica un controlador con una señal de control actualizada para tener en cuenta el retardo. Para esto se utiliza un vector de estados estimado (actualizado). Por lo tanto la nueva señal de control se calcula en términos del estado estimado en $t_{k+\tau}$, etiquetado $x_{k+\tau}$ como se muestra en la Figura 1.3.

Además $x_{k+\tau}$ puede ser calculado usando una muestra x_k en cualquier instante de tiempo $t_k \in (t_{k-1+\tau}, t_{k+\tau})$. De esta forma primero el controlador estimará el estado

$$x_{k+\tau} = \phi(\tau_k)x_k + \Gamma(\tau_k)u_{k-1}, \tag{1.12}$$

donde $t = \tau_k$ y τ_k se obtiene de,

$$\tau_k = t_{k+\tau} - t_k, \tag{1.13}$$

y el controlador calculará la señal de control con el estado estimado como:

$$u_k = Lx_{k+\tau} \tag{1.14}$$

donde $L \in \mathbb{R}^{1xn}$ es el estado de realimentación, obtenido por métodos de diseño de control para las matrices $\phi(h)$ y $\Gamma(h)$.

Al compensar las fluctuaciones de latencia mediante el ajuste de los parámetros de tiempo en el controlador, se puede mejorar: tanto el rendimiento del control, como la programación del conjunto de tareas que realiza el computador [1].

1.6. Tarea de control por enfoque de compensación

En [7] propone un algoritmo de control por enfoque de compensación. Donde su intención es compensar la degradación en la respuesta del sistema de control. Mediante las variaciones de una respuesta a otra. El modelo propuesto trata de ajustar en cada tiempo de ejecución de la tarea de

control los parámetros del controlador. Para tener en cuenta tanto la fluctuación en el muestreo como los retrasos en la actuación.

Los parámetros del controlador deben actualizarse en cada ejecución de la tarea de control de acuerdo con la fluctuación en el muestreo h_k y el retardo en la actuación τ_k. Para su demostración se usa un modelo de sistema en tiempo discreto con muestreo irregular, el cual presenta retardos de tiempo variable,

$$\begin{aligned} x(\bar{h}_{k+1}) &= \phi(h_k)x(\bar{h}_k) + \Gamma_0(h_k,\tau_k)u(\bar{h}_k) + \Gamma_1(h_k,\tau_k)u(\bar{h}_{k-1}) \\ y(\bar{h}_k) &= Cx(\bar{h}_k) + Du(\bar{h}_k), \end{aligned} \tag{1.15}$$

la señal de control para el sistema será calculada mediante realimentación de estados. Donde el vector de realimentación de estados será calculado en el instante h_k y el vector de estados actualizado en el instante de actuación $\bar{h}_k$,

$$u(\bar{h}_k) = -L(h_k)x(\bar{h}_k), \tag{1.16}$$

donde $\bar{h}_k$ representa

$$\bar{h}_k = \sum_0^k h_k, \tag{1.17}$$

y los términos $\phi(h_k)$, $\Gamma_0(h_k\tau_k)$, $\Gamma_1(h_k,\tau_k)$ estarán dados por

$$\begin{aligned} \phi(h_k) &= e^{Ah_k} \\ \Gamma_0(h_k,\tau_k) &= \int_0^{h_k-\tau_k} e^{A(s)}Bd(h_k-\tau_k). \\ \Gamma_1(h_k,\tau_k) &= e^{A(h_k-\tau_k)}\int_0^{\tau_k} e^{A(s)}Bd(\tau_k). \end{aligned} \tag{1.18}$$

Aplicando este enfoque, la degradación de la respuesta del sistema debido a la fluctuación de los retardos de tiempo, se elimina como se muestra en la Figura 1.5. Esto se debe a que la acción de control se calcula con las fluctuaciones reales en cada instancia de la tarea de control.

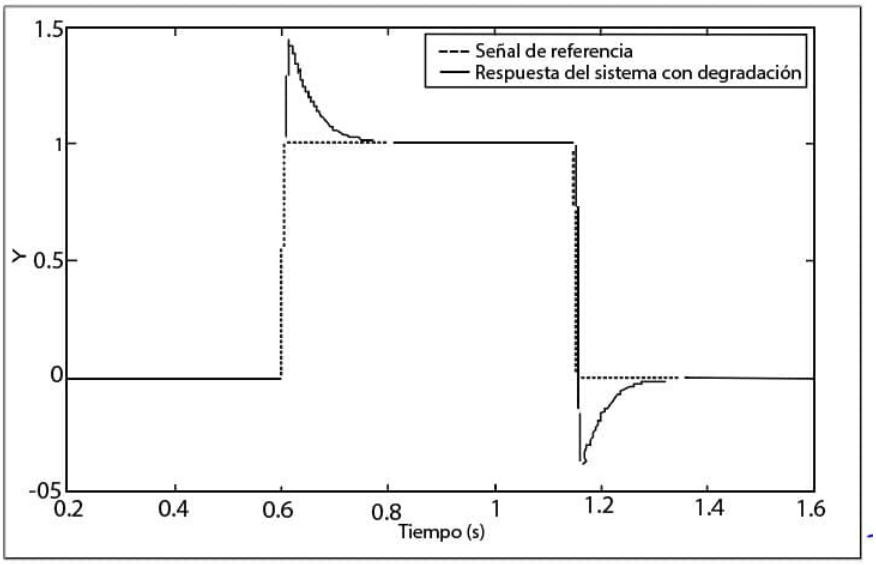

Figura 1.4: Respuesta del sistema con degradación [7]

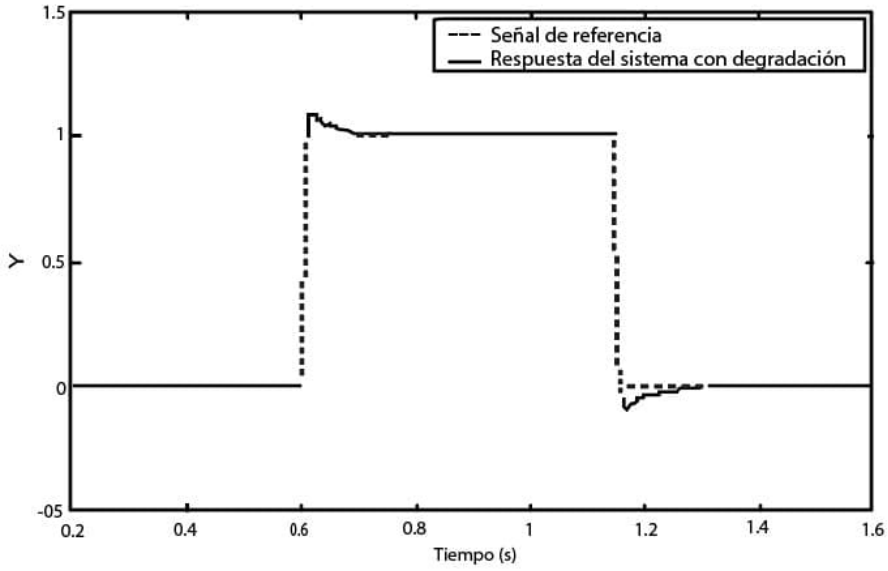

Figura 1.5: Respuesta del sistema con compensación [7].

1.7. Filtro de Kalman

La incertidumbre existente en un sistema de control en lazo cerrado, es provocada en cierta medida por el ruido en la medición. Los sensores sufren errores en el cálculo de sus lecturas debido a causas físicas internas como vibración o cambios bruscos de temperatura. Esto causa que su precisión se vea afectada y se introduzca ruido en el sistema. Afectando al rendimiento del mismo. Existen métodos de corrección los cuales dependen de las características de las perturbaciones. Si el rango de frecuencia de las perturbaciones es diferente del ancho de banda de la señal, un filtrado adecuado en el dominio de la frecuencia elimina las señales no deseadas. Esta etapa suele ser necesaria pero no es suficiente para eliminar los errores del sensor.

En [11] se propone el filtro de Kalman para estimar el estado de un sistema dinámico que tiene incertidumbre en la medición por lo antes mencionado. Para esto el sistema debe describirse en forma de espacio de estados:

$$\begin{aligned} X_{k+1} &= \phi_k \cdot X_k + W_k \\ Z_k &= H \cdot X_k + V_k, \end{aligned} \tag{1.19}$$

donde X_k se llama vector de estado. Está compuesto por cualquier conjunto de variables suficientes para describir completamente un sistema dinámico. Z_k se llama vector de observación. Se trata de datos que se pueden conocer mediante mediciones. W_k y V_k son el ruido en el estado del sistema y la medición respectivamente. ϕ_k es la matriz de transición de estado, H_k la matriz de observación.

El filtro de Kalman basa su funcionamiento en un algoritmo, el cual se ejecuta de forma recursiva. El sistema de control en espacio de estados en tiempo discreto, con un tiempo de muestreo t_k. Donde, la combinación óptima de resultados medidos y estimados viene dado por:

$$\hat{X}_k = \hat{X}_k^- + K_k \cdot (Z_k - H_k \cdot \hat{X}_k^-), \tag{1.20}$$

donde $\hat{X}_k^-$ se denomina la estimación a priori. K_k es la ganancia del filtro de Kalman y se obtiene de :

$$K_k = P_k \cdot H_k^T \cdot (H_k \cdot P_k^- \cdot H_k^T + R_k)^{-1}, \tag{1.21}$$

R viene a ser la covariaza del ruido en la medición. La matriz de covarianzas del error P_k se obtiene de:

$$P_k = (I - K_k \cdot H_k) \cdot P_k^-. \tag{1.22}$$

Dado que el algoritmo de kalman se calcula de forma recursiva, en cada instante de la tarea de control. Las predicciones para la próxima estimación del estado serán dadas por:

$$\begin{aligned} X_{k+1} &= \phi_k \cdot X_k \\ P_{k+1}^- &= \phi_k \cdot P_k \cdot \phi_k^T + Q_K \end{aligned} \tag{1.23}$$

El modelo del filtro Kalman presentado permite la estimación de los estados en sistemas no lineales.

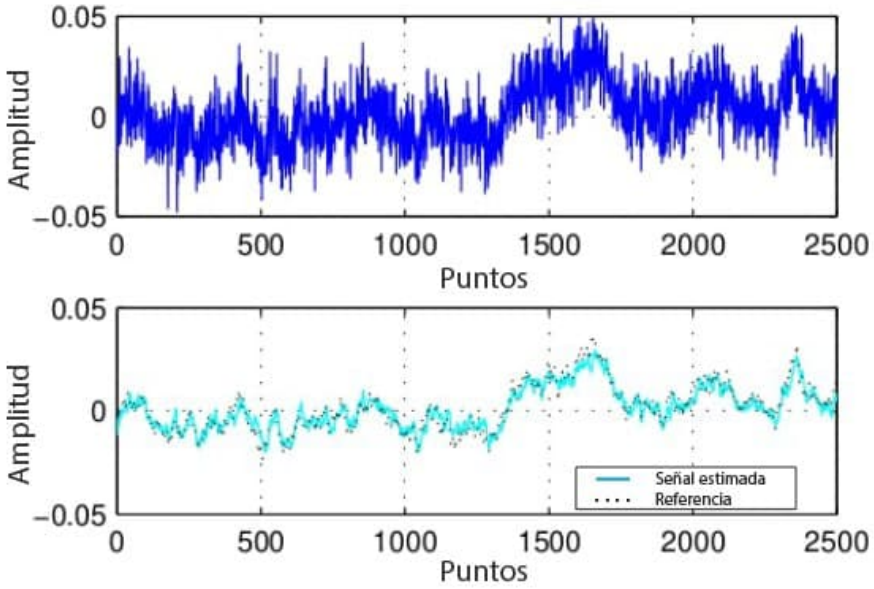

Figura 1.6: Estimación del filtro de Kalman en una señal con ruido. Señal con ruido (arriba), estimación y referencia (abajo)[11].

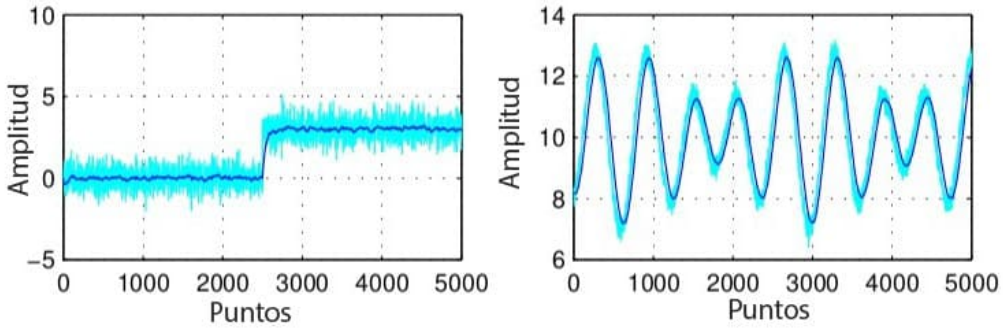

Figura 1.7: Estimación aplicando el filtro de Kalman a una señal con ruido. Señal sinusoidal (izquierda), señal escalón (derecha) [11].

El filtro de Kalman es muy adecuado para eliminar la incertidumbre provocada por el error en los sensores, que es parcial o totalmente aleatorio. El modelo presentado en [11] presenta un correcto funcionamiento estimando la señal afectada por el ruido. De esta forma la incertidumbre se reduce y como resultado la degradación de la acción de control se elimina de forma parcial.

Capítulo 2

Metodología

En este capítulo se desarrolla el modelo de control y el algoritmo que permite su implementación.

2.1. Descripción del sistema

Analizamos un modelo en espacio de estados que represente cualquier tipo de planta. Se considera el modelo de espacio de estados un sistema de tiempo discreto invariante en el tiempo lineal con período de muestreo h[12] ,

$$\begin{aligned} x_{k+1} &= \Phi(h)x_k + \Gamma(h)u_k \\ y_k &= Cx_k, \end{aligned} \tag{2.1}$$

donde $x_k \in \mathbb{R}^{nx1}$, $u_k \in \mathbb{R}^{mx1}$ y $y_k \in \mathbb{R}^{px1}$ son la matriz de estado, entrada y salida respectivamente, y $C \in \mathbb{R}^{pxn}$ es la matriz de salida del sistema, las matrices $\phi(t)$ y $\Gamma(t)$ están dadas por,

$$\begin{aligned} \Phi(t) &= e^{At} \\ \Gamma(t) &= \int_0^t e^{A(s)}B\,dt, \end{aligned} \tag{2.2}$$

donde $A \in \mathbb{R}^{nxn}$ y $B \in \mathbb{R}^{nxm}$ son matrices que describe la dinámica del sistema.

El modelo propuesto en (2.1) se puede modificar para tomar en cuenta un retardo de tiempo modelando una latencia de entrada/salida que aparece debido al calculo del algoritmo de control.

El modelo entándar que incorpora un retardo de tiempo τ, con $\tau \leq h$ es [12]

$$x_{s,k+1} = \Phi(h)x_{s,k}\Phi(h-\tau)\Gamma(\tau)u_{k-1} + \Gamma(h-\tau)u_k \tag{2.3}$$

El modelo presentado (2.3) se ha tomado a menudo como el modelo de tarea de control estándar para el diseño y análisis de sistemas de control en tiempo real. Este modelo asume una referencia de tiempo dada por los instantes de muestreo con un retardo de tiempo fijo desde el muestreo hasta la actuación (Figura 2.1). Es programada para muestrear y actuar periódicamente[13].

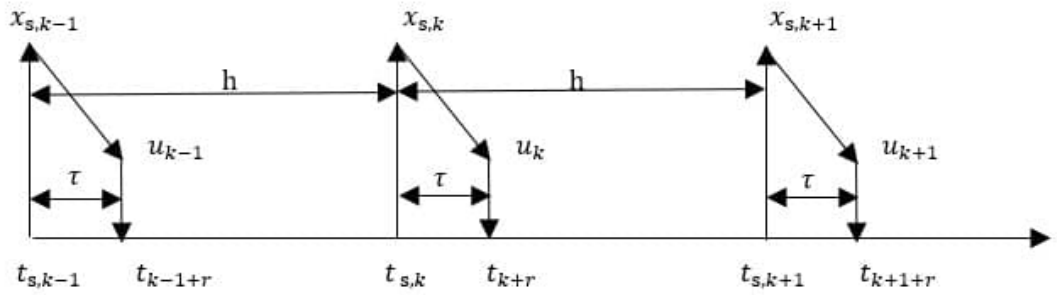

Figura 2.1: Modelo de tarea estándar considerando un retardo de tiempo.

2.2. Modelo de control

Si se analiza un sistema con un modelo de control implementado en tiempo real, la sincronización exigida por el modelo de control dado por (2.3) a menudo se ve violada. Esto se debe a la irregularidad entre el muestreo y la actuación que introducen las fluctuaciones de programación.

Para solucionar este problema desde la perspectiva de un modelo de tarea de control, puede basarse en proporcionar sincronismo en los instantes de actuación, en lugar de los instantes de muestreo, como lo propone en [1].

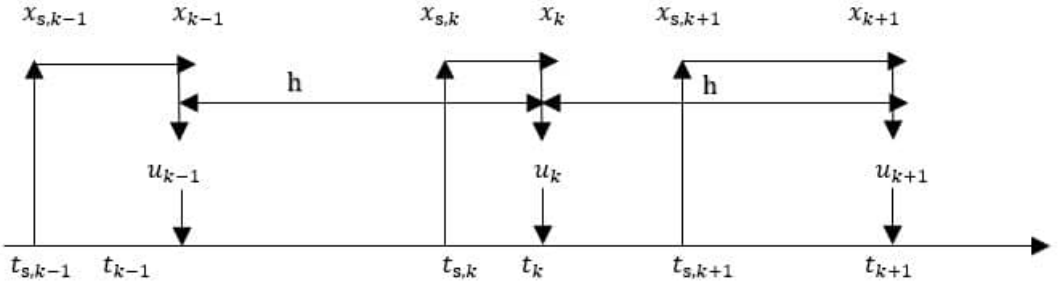

Figura 2.2: Modelo de tarea de control.

2.3. Actuación periódica

En [1], el tiempo transcurrido entre instantes de actuación consecutivos denominados t_{k-1} y t_k como se muestra en la Figura 2.2, es periódico y $h = t_k - t_{k-1}$ es denominado como el período de actuación. Dentro de este intervalo de tiempo se muestrea el estado del sistema denominado

$x_{s,k} \in (t_{k-1}, tk)$ y se registra el tiempo de muestreo $t_{s,k}$. La diferencia entre este tiempo y el siguiente tiempo de actuación es nombrada τ_k y se calcula como

$$\tau_k = t_k - t_{s,k}, \tag{2.4}$$

y es usado para estimar el estado en el instante de actuación

$$\hat{x}_k = \Phi(\tau_k) x_{s,k} + \Gamma(\tau_k) u_{k-1}. \tag{2.5}$$

Con el nuevo estado estimado $\hat{x}_k$ la señal de control sera dada por

$$u_k = L\hat{x}_{k+\tau}, \tag{2.6}$$

donde L es la ganancia de control. La señal de control u_k se aplica a la planta por medio de interrupciones de hardware en cada instante de actuación y se mantiene constante a lo largo de cada periodo de muestreo. Además no se requiere que las muestras sean periódicas porque τ_k puede variar en cada operación de calculo de la acción de control como se muestra en la Figura:2.2.

La Tarea de control con sincronización en los instantes de actuación puede ser implementada sin mayor complejidad. Pero si tomamos en cuenta, que su operación se basa en predicciones calculadas en base a cada muestra $x_{s,k}$. Además, conocemos muy bien que los sistemas están alterados por ruido y que los sensores en un lazo de control no proporcionan lecturas exactas de las variables medidas[13]. El rendimiento del sistema se vera deteriorado rápidamente.

2.4. Propuesta del filtro de Kalman aplicado a la actuación periódica

2.4.1. Revisión del filtro de Kalman

El filtro de Kalman en tiempo discreto resuelve el problema de estimar el estado del sistema de una planta controlada en tiempo discreto. Para implementar el filtro podemos agregar al modelo expresado por la ecuación 2.1, ruido en el proceso y la medición (w_k y v_k).

$$x_{k+1} = \Phi(h) x_k + \Gamma(h) u_k + w_k \tag{2.7}$$

$$y_k = C x_k + v_k. \tag{2.8}$$

El filtro de Kalman aplicado a una planta controlada en tiempo discreto proporciona una estimación óptima de sus estados. La información que usa proviene de dos fuentes: desde mediciones hechas a través de los sensores y desde estimaciones realizadas usando el modelo del sistema. Mezcla estos dos recursos de información de manera óptima considerando las descripciones probabilísticas de su exactitud, es decir, la precisión de los sensores y del modelo[14].

La implementación de filtro de Kalman se divide en dos fases: actualización del tiempo (predicción) y actualización de la medición (corrección).

Fase de predicción

En la fase de predicción, deseamos estimar el siguiente estado del sistema, si hacemos uso de la ecuación 2.7 la estimación a priori del estado del sistema es

$$\hat{x}^{-}_{(k+1)} = \Phi \hat{x}_{(k)} + \Gamma u_{(k)}, \tag{2.9}$$

donde ϕ y Γ representan la dinámica del sistema, y $\hat{x}_k$ es la actual estimación a posterior del estado de proceso y u_k representa la entrada actual. La estimación a priori del error de la covarianza es

$$P^{-}_{(k+1)} = \Phi P_{(k)} \Phi^T + Q, \tag{2.10}$$

donde $P_{(k)}$ es la actual estimación a posteriori del error de la covarianza, y Q es la constante de la covarianza del ruido en el proceso.

Fase de corrección

Se realiza una estimación a posteriori para ser utilizada en la nueva etapa de predicción. La próxima ganancia de Kalman estará dada por

$$K_{(k+1)} = \frac{CP^{-}_{(k+1)}}{(CP^{-}_{(k+1)} C^T + R)}, \tag{2.11}$$

donde R es la constante de la covarianza del ruido en la medición. La estimación a posteriori del próximo estado sera

$$\hat{x}_{(k+1)} = \hat{x}^{-}_{(k+1)} + K_{(k+1)}(y_{(k)} - C\hat{x}^{-}_{(k+1)}), \tag{2.12}$$

donde $y_{(k)}$ es la medición de la salida del sistema como se muestra en la ecuación 2.4.
Para Finalizar la estimación a posteriori del error de covarianza es

$$P_{(k+1)} = (I - CK_{(K+1)}P^{-}_{(k+1)}), \tag{2.13}$$

donde I es la matriz identidad.

Señal de control

Se plantea el calculo de la señal de control por medio de la ley de control para sistemas lineales en espacio de estados, el cual es la realimentación de los mismos[15].

$$u_k = -Lx_{k+\tau}, \tag{2.14}$$

donde $L \in \mathbb{R}^{1xn}$ es la ganancia de retroalimentación del estado obtenida usando métodos de diseño de control óptimo a partir de las matrices en tiempo discreto ϕ y Γ.

2.4.2. Filtro de Kalman sobre la actuación periódica

La propuesta de integración del filtro de kalman sobre la actuación periódica se divide en dos partes.

Primero, la fase de corrección no se puede aplicar, debido a que los valores de las medidas del proceso solo están disponibles en instante de muestreo y no existen en instantes actuación. Por lo tanto, desde el muestreo ($t_{s,k}$) hasta la actuación (t_k) solo se utiliza la fase de predicción (2.9)-(2.10) las mismas que se transforma en

$$\hat{x}^{-}_k = \Phi(\tau_k)\hat{x}_{s,k} + \Gamma(\tau_k)u_{k-1}, \tag{2.15}$$

$$P^{-}_k = \Phi(\tau_k)P_{s,k}\Phi(\tau_k)^T + Q. \tag{2.16}$$

Segundo, de la actuación (t_k) al próximo muestreo($t_{s,k+1}$), se ejecuta la fase de predicción y corrección del filtro. Por lo tanto, si solo aplicamos la fase de predicción del filtro de $t_{s,k}$ a la actuación t_k las ecuaciones (2.9)-(2.10) se redefinen como

$$\hat{x}^{-}_{s,k+1} = \Phi(h - \tau_{k+1})\hat{x}^{-}_k + \Gamma(h - \tau_{k+1})u_k, \tag{2.17}$$

$$P^{-}_{s,k+1} = \Phi(h - \tau_{k+1})P_k\Phi(h - \tau_{k+1})^T + Q, \tag{2.18}$$

a continuación las ecuaciones (2.11),(2.12) y (2.13) de la fase de corrección del filtro se formulan aplicando la misma teoría como

$$K_{s,k+1} = \frac{CP^{-}_{s,k+1}}{(CP^{-}_{s,k+1}C^{T} + R)} \tag{2.19}$$

$$\hat{x}_{s,k+1} = \hat{x}^{-}_{s,k+1} + K_{s,k+1}(y_{s,k+1)} - C\hat{x}^{-}_{s,k+1}) \tag{2.20}$$

$$P_{s,k+1} = (I - CK_{s,K+1})P^{-}_{s,k+1}. \tag{2.21}$$

Si aplicamos la estimación del estado en el instante de actuación la señal de control estará dada por

$$u_k = -L\hat{x}^{-}_{k}. \tag{2.22}$$

2.5. Algoritmo de control

El algoritmo de control propuesto se basa en un controlador con señal de control actualizada como se muestra en la figura 1.3, el cual nos permite una actuación periódica como propone en [1], además el filtro de Kalman discreto para resolver el problema de estimación del estado y a su vez filtrar el ruido del sistema como propone en [16].

Pseudocódigo del controlador

Begin

1. u_k := read_input()
2. t_k := get_time()
3. $t_{k+\tau} := t_{k+\tau} + h$
4. $\tau_k := t_{k+\tau} - t_k$
5. $\hat{x}^-_{s,k+1} = \Phi(h-\tau_{k+1})\hat{x}^-_k + \Gamma(h-\tau_{k+1})u_k$
6. $P^-_{s,k+1} = \Phi(h-\tau_{k+1})P_k\Phi(h-\tau_{k+1})^T + Q$
7. $K_{s,k+1} = \frac{CP^-_{s,k+1}}{(CP^-_{s,k+1}C^T+R)}$
8. $\hat{x}_{s,k+1} = \hat{x}^-_{s,k+1} + K_{s,k+1}(y_{s,k+1)} - C\hat{x}^-_{s,k+1})$
9. $P_{s,k+1} = (I - CK_{s,K+1})P^-_{s,k+1}$
10. $u_k = -L\hat{x}^-_k$

end

Capítulo 3

Implementación y pruebas

En este capítulo, se validará la teoría analizada a través de simulaciones en el software Matlab, además se analizan los resultados obtenidos para evaluar su funcionamiento.

3.1. Plataforma de Simulación

La simulación se realizó en Simulink, el cual es útil para simular el comportamiento de los sistemas dinámicos en un entorno gráfico. En el cual el modelo a simular se construye arrastrando los diferentes bloques que lo constituyen.

Dentro de Simulink, se utiliza el toolbox de True-Time(Figura 3.1), un simulador basado en matlab/simulink. El cual facilita la simulación de la ejecución de tareas del controlador en kernels en tiempo real [17].

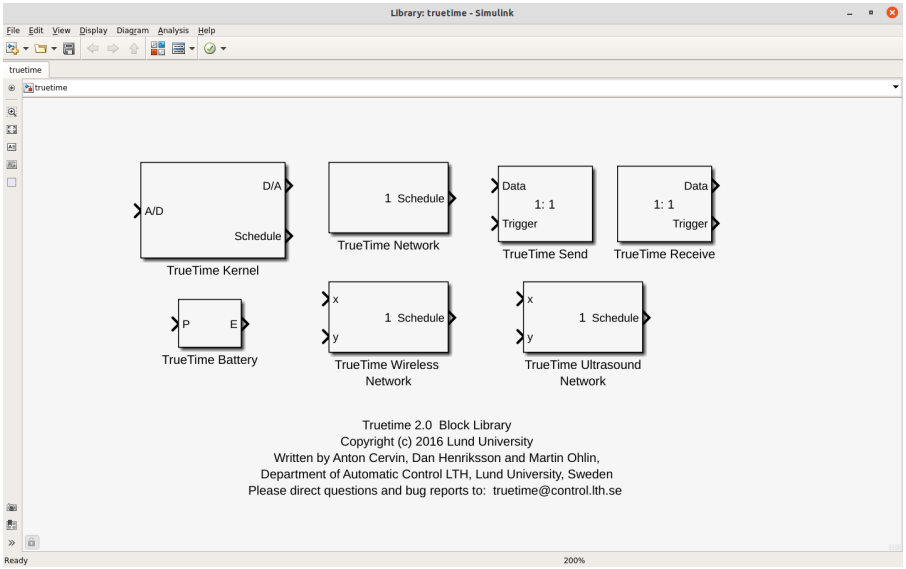

Figura 3.1: Toolbox Truetime.

Dentro del toolbox reluce el bloque TrueTime Kernel. Este nos permite crear tareas periódicas y aperiódicas. En nuestro caso, tareas periódicas, las cuales son creadas por el comando ttCreatePeriodTask. El comando configura un temporizador interno el cual libera periódicamente un trabajo para realizar una tarea de control.

Se configura los parámetros del bloque del kernel a través de su cuadro de dialogo (Figura 3.2). Los parámetros principales a configurar:

- Nombre de la función init (Name of init function): este parámetro hace referencia al programa de inicialización con los parámetros del kernel
- Número de entradas y salidas analógicas (Number of analog input and outputs)

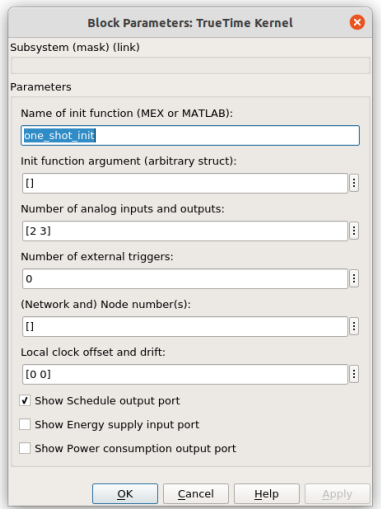

Figura 3.2: Configuración parámetros del Kernel.

La implementación del sistema y el kernel de TrueTime se puede observar es la siguiente Figura.

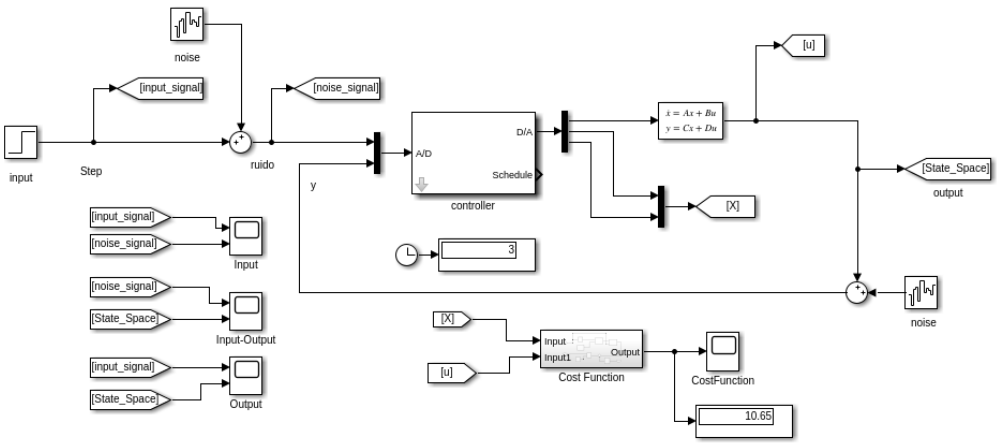

Figura 3.3: Diagrama de bloques del sistema de control.

3.2. Planta

Para la validación del algoritmo propuesto, se analiza un circuito electrónico estabilizador de voltaje en la forma RCRC como se muestra en [1], como la planta objetivo de control, donde su representación en espacio de estados esta dado por:

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -918{,}27 & -90{,}90 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 918{,}27 \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t)$$

El control se diseña para mantener un voltaje de salida deseado, que sigue una onda cuadrada en 0V como nivel bajo y 2.4V como nivel alto.

Como se muestra en el algoritmo de control propuesto, se puede lograr sincronización en los instantes de actuación, en lugar de los instantes de muestreo. El tiempo transcurrido entre instantes de actuación consecutivos denominas t_{k-1} y t_k es periódico y $h = t_k - t_{k-1}$ se define como período de actuación. En el intervalo de tiempo t_{k-1}, t_k se muestrea el estado del sistema $x_k \in (t_{k-1}, t_k)$ y se registra el tiempo de muestreo t_k[14].

La diferencia existente entre el tiempo de muestreo y el tiempo de actuación sera (2.4) este tiempo τ_k es usado para estimar el estado en el próximo instante de actuación (Apéndice B)

3.3. Diseño del controlador

El filtro de Kalman se diseñó teniendo en cuenta las covarianzas de ruido $Q_n = E(w * w^T) = 2 * 10^{-7}$ y $R_n = E(v * v^T) = 8 * 10^{-5}$ donde w y v son la perturbación en la planta y el ruido de medición, respectivamente [1].

Al no ser posible realizar una realimentación exacta del estado, pues esta se desconoce, la señal de control dada por $u_k = -Lx_k$ se remplaza por la realimentación del estado estimada de esta forma la ley de control por realimentación de estados queda como

$$u_k = -L(\tau_k)x_k \tag{3.1}$$

donde L se calcula en cada instante de ejecución del controlador con el valor de tiempo τ_k obtenido previamente de (2.4). La realimentación de estados (L) se obtiene con la ayuda del comando acker de matlab el cual resuelve la fórmula de ackerman para determinar la matriz de ganancia de realimentación del estado L el cual hace uso de las matrices en tiempo discreto ϕ y Γ y ubica los polos en lazo cerrado en $-1{,}5 \pm 1{,}5i$ como se muestra,

$$L = acker(\Phi, \Gamma, [z_1\, z_2]), \tag{3.2}$$

donde z_1 y z_2 son los polos en lazo cerrado en tiempo discreto.

3.4. Pruebas y análisis de resultados

3.4.1. Respuesta al escalón unitario bajo condiciones ideales

La prueba consiste en someter al sistema a una señal de escalón unitario con un retraso de $0{,}5s$ y una amplitud de $2{,}4v$. La Figura 3.4 muestra la respuesta del sistema con una referencia sin ruido. Se puede observar que el algoritmo de control responde de forma adecuada a la referencia.

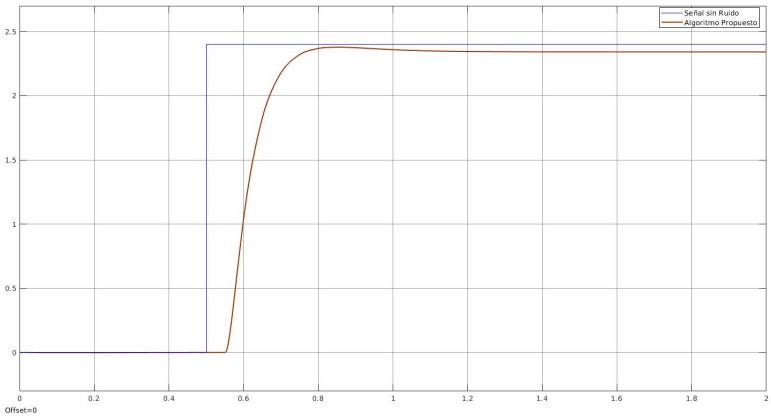

Figura 3.4: Respuesta del sistema a una señal de escalón unitario.

3.4.2. Respuesta del sistema ante una señal con ruido

En la Figura 3.5 se puede observar la respuesta del sistema ante una señal con ruido blanco. Se puede analizar que el algoritmo de control responde de forma adecuada a una señal con perturbaciones.

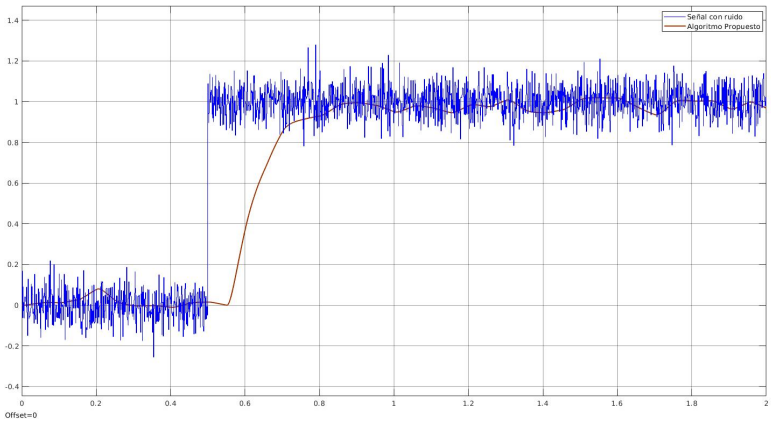

Figura 3.5: Respuesta del sistema a una señal de escalón unitario con ruido.

Además, se analiza el comportamiento del algoritmo frente a una señal cuadrática. La cual es proporcionada por un generador de pulsos programable conectada a la entrada analógica del kernel. Configurada con un retraso de 0,5s, una amplitud de 2,4v en alto, 0v en bajo y un período de 2s.

La Figura 3.6 muestra la comparación entre la señal con ruido sensada en la planta, con la señal resultante de aplicar el algoritmo de control propuesto. El cual utiliza el filtro de Kalman que elimina de forma eficaz la incertidumbre.

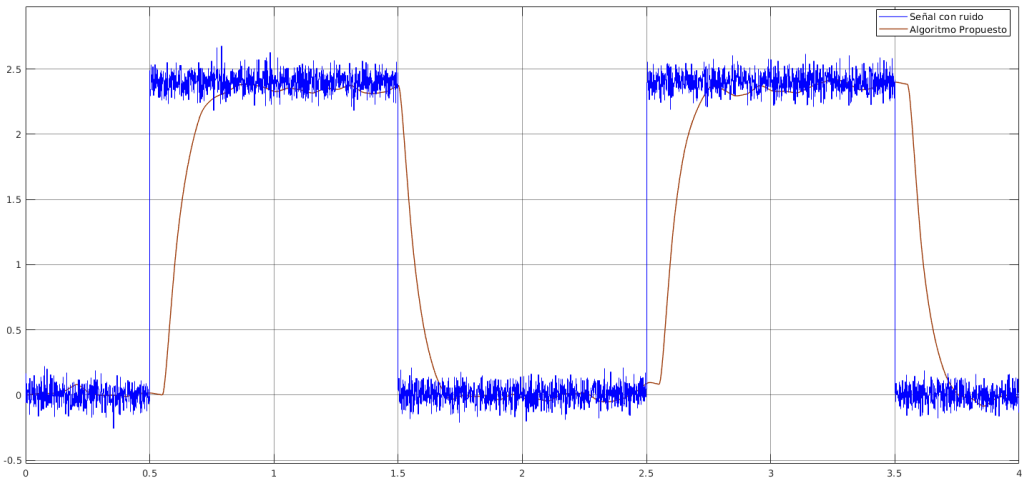

Figura 3.6: Respuesta del sistema a una señal con ruido.

La Figura 3.7 se puede observar la señal resultante de someter al sistema a una señal con ruido, frente a la referencia. Donde presenta una adecuada respuesta al someterse a las condiciones del sistema. Mostrando, un adecuado comportamiento al mantener las condiciones de diseño.

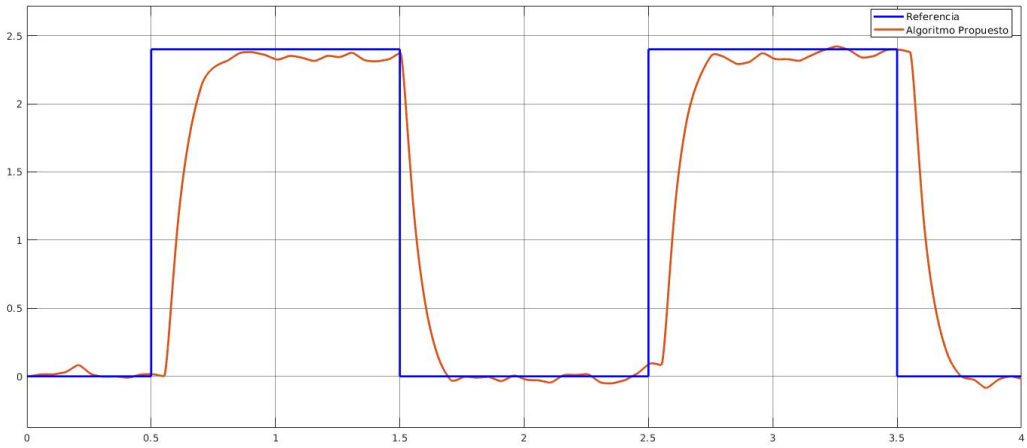

Figura 3.7: Referencia en condiciones ideales, frente a la respuesta del sistema a una señal con ruido.

3.4.3. Evaluación del rendimiento de control

En el sistema propuesto considera variaciones temporales causadas por retardos de tiempo, en el lazo de control, también se considera que además de incluir retardos de tiempo, contiene ruido o perturbación en el proceso y ruido en los sensores.

El rendimiento de control es medido por medio de la entrada y salida de la planta a lo largo del tiempo de simulación(t_{sim}), por medio de un bloque que contiene una función de costo cuadrática

$$J = \int_0^{t_{sim}} [x^T Q_e x(t) + u^T(t) R_e u(t)] dt, \tag{3.3}$$

donde las matrices de peso Q_e y R_e son iguales a la identidad.

Para evaluar su rendimiento se realizó simulaciones del sistema con las mismas condiciones. Se sometió el sistema a un algoritmo de control estándar, donde se considera un muestreo periódico. Además el mismo control estándar junto al filtro de Kalman para eliminar el ruido en la señal, frente al algoritmo de control propuesto el cual presenta una actuación sincronizada, gracias a la actualización de la señal de control, que nos permite calcular la diferencia de tiempo entre el muestreo y la actuación y estimar el próximo vector de estado, todo esto junto al filtro de Kalman en tiempo discreto para eliminar el ruido en la señal de control.

Los resultados obtenidos por la función de costo en el tiempo de ejecución acumulado se muestra en la Figura 3.8

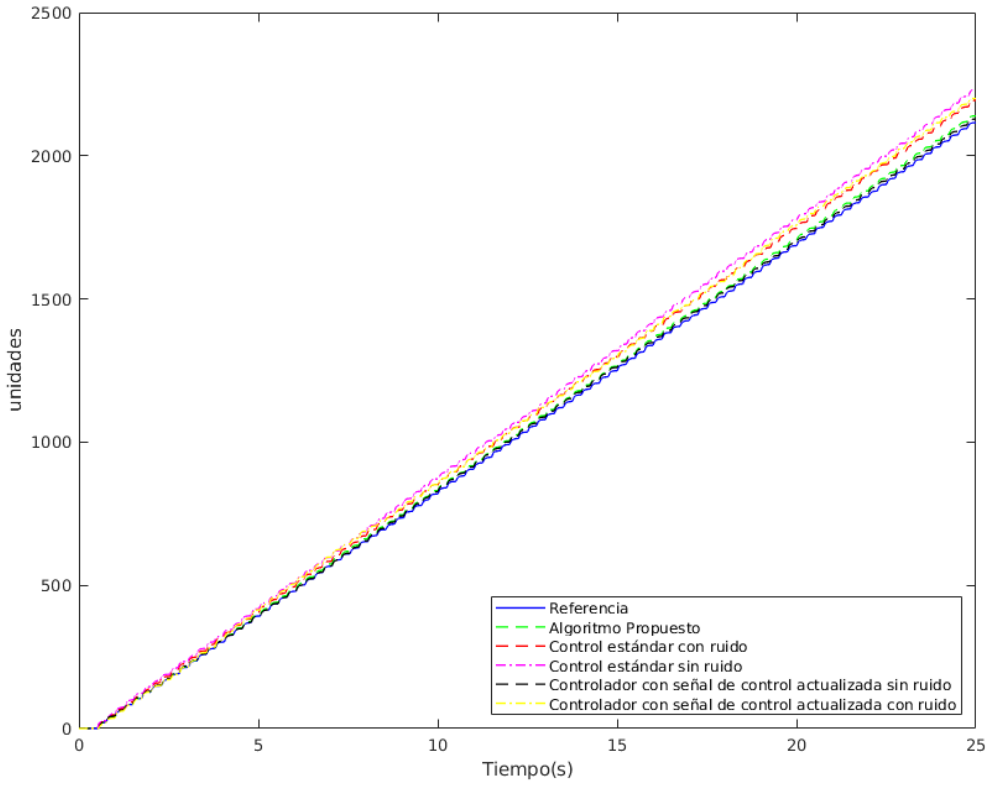

Figura 3.8: Evaluación del rendimiento del algoritmo de control

Capítulo 4

Conclusiones y trabajo futuro

En este capítulo se muestra algunas conclusiones del presente proyecto y se propone algunas posibles aplicaciones de este trabajo.

Conclusiones

- En este trabajo se ha utilizado el filtro de Kalman como técnica de filtrado el cual cumple de forma satisfactoria su propósito de eliminar el ruido y las perturbaciones. Además, de los efectos de estos en la degradación del rendimiento.
- El enfoque presentado muestra periodicidad en el período de actuación del lazo de control en tiempo discreto, eliminando los retardos existentes. Esto permite optimizar el rendimiento de la acción de control al existir sincronismo en la fase de actuación.
- Se logró demostrar el rendimiento adecuado del algoritmo de control propuesto, donde basa su funcionamiento en la sincronización de los instantes de actuación, frente a otros modelos de control estándar el cual presenta la sincronización en el muestreo.
- True-Time nos facilita la programación de tareas de control y su ejecución en kernels que simulan un controlador en tiempo real. De esta forma nos permite evaluar el desempeño de nuestros supuestos, sin ser implementados en un entorno físico.

Trabajo futuro

En el presente trabajo se realizo la implementación del filtro de kalman sobre un sistema con actuación periódica. Por medio de simulaciones se implementó y demostró el correcto desempeño de la teoría presentada. Como trabajo futuro se plantea la implementación de este trabajo en un entorno físico, el cual permita corroborar el funcionamiento y desempeño frente a condiciones físicas reales. Además, evaluar en un sistema de control multitareas en tiempo real el cual ejecute un conjunto de tareas, implementadas con el algoritmo de control presentado en este trabajo.

Bibliografía

[1] C. Lozoya, M. Velasco, and P. Martí, "The one-shot task model for robust real-time embedded control systems," *IEEE Transactions on Industrial Informatics*, vol. 4, no. 3, pp. 164–174, 2008.

[2] E. L. Eremin, L. V. Chepak, and E. A. SHelenok, "Combined adaptive control system for nonlinear periodic action plant," *2015 International Siberian Conference on Control and Communications, SIBCON 2015 - Proceedings*, vol. 1, no. 2, 2015.

[3] K. E. Årzén, A. Cervin, J. Eker, and L. Sha, "An introduction to control and scheduling co-design," *Proceedings of the IEEE Conference on Decision and Control*, vol. 5, pp. 4865–4870, 2000.

[4] G. Buttazzo, "Research trends in real-time computing for embedded systems," *ACM SIGBED Review*, vol. 3, no. 3, pp. 1–10, 2006.

[5] X. Q. Xiao, F. F. Wang, and H. S. Zhao, "Application of self-triggered control in power system excitation control," *2011 International Conference on Electrical and Control Engineering, ICECE 2011 - Proceedings*, no. 51077054, pp. 577–580, 2011.

[6] S. Hong, X. S. Hu, and M. D. Lemmon, "Reducing delay jitter of real-time control tasks through adaptive deadline adjustments," *Proceedings - Euromicro Conference on Real-Time Systems*, pp. 229–238, 2010.

[7] P. Marti, R. Villa, J. M. Fuertes, and G. Fohle, "On real-time control tasks schedulability," *2001 European Control Conference, ECC 2001*, pp. 2227–2232, 2001.

[8] P. Martí and M. Velasco, "Toward flexible scheduling of real-time control tasks: Reviewing basic control models," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4416 LNCS, pp. 710–713, 2007.

[9] P. Martí, J. M. Fuertes, G. Fohler, and K. Ramamritham, "Jitter compensation for real-time control systems," *Proceedings - Real-Time Systems Symposium*, no. January, pp. 39–48, 2001.

[10] C. ans J.W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," pp. 1551–1556, 2003.

[11] H. P. A. F. P. C. Marselli, D. Daudet, "Application of Kalman filtering to noise reduction on microsensor signals Application de filtres de Kalman à la réduction de bruit," no. October, pp. 443–450, 2000.

[12] P. E. Wellstead, "Book Review: Computer Controlled Systems: Theory and Design," *The International Journal of Electrical Engineering Education*, vol. 23, no. 4, pp. 376–376, 1986.

[13] P. S. Maybeck and G. M. Siouris, "Stochastic Models, Estimation, and Control, Volume I," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 10, no. 5, p. 282, 1980.

[14] C. X. Rosero, C. Vaca, L. T. Sub, and M. Gavilanez, "Sobre Sistemas de Control en Red bajo Incertidumbres de Tiempo, Medición y Proceso."

[15] J. Ortiz and M. Reinoso, "Análisis de controladores por realimentación de estados, control óptimo y lógica difusa, para un péndulo de Furuta," *Ingenius*, no. 9, pp. 15–22, 2013.

[16] C. Lozoya, J. Romero, P. Martí, M. Velasco, and J. M. Fuertes, "Embedding kalman techniques in the one-shot task model when non-uniform samples are corrupted by noise," no. July, pp. 701–706, 2010.

[17] D. Henriksson, A. Cervin, and K. E. Årzén, *TRUETIME: Simulation of control loops under shared computer resources*. IFAC, 2002, vol. 15, no. 1. [Online]. Available: http://dx.doi.org/10.3182/20020721-6-ES-1901.00975

Apéndice A

Configuración del kernel TrueTime

```
function init(Arg)

ttInitKernel('prioFP')
%% Datos Tarea Periodica
data.exectime = 0.001;    % control task execution time
starttime = 0.0;        % control task start time
period = 0.05;           % control task period
data.h = 0.05;           % periodo

%% Inicializaci n Variables de Datos
data.u = 0;               % s e a l  de control
data.uk = 0;            % Se a l  de referencia
data.y = 0;          % Salida Anterior

data.tk_tau=0;          % tiempo entre muestras
%% Matrices de la Dinamica del Sistema
data.A = [0 1 ; -918.27 -90.90];    % matris de transicion de estado
data.B = [0;918.27];                % matris de trancision de control
data.C = [1 0];                     % matris de salida
data.D = 0;

%% Datos Kalman Filter
data.Q = 2e-7;       %Covarianza del ruido del poceso
data.R = 8e-5;       % Covarianza del ruido en la medicion
data.P = 1e6;
data.K = 0;
data.X = [0 ; 0];

%% creacion de tarea periodica
ttCreatePeriodicTask('ctrl_task', starttime, period, 'code', data)
```

Apéndice B

Kernel

```
function [exectime, data] = code(segment, data)

switch segment
    case 1
  %% lectura de entradas
        data.uk = ttAnalogIn(1);              % muetreo estado de la Planta
        data.y = ttAnalogIn(2);             % salida anterior de la planta

  %% Calculo de Tiempos
        t_k = ttCurrentTime()                 % obtencion del tiempo de ...
           transcurrido
        data.tk_tau = data.tk_tau + data.h
        tau_k = data.tk_tau - t_k

  %% Llamada a la senal de control
        [data] = control(data,tau_k);%  Tarea de control
        exectime=data.exectime;

    case 2
   %% EScritura en el puerto de Salida
        ttAnalogOut(1, data.u);
        ttAnalogOut(2, data.X(1,1))
        ttAnalogOut(3, data.X(2,1))


        exectime = -1;
end
```

Apéndice C

Tarea de Control

```
function [data] = control(data,t)
%% Espacio de Estados
sys = ss(data.A,data.B,data.C,data.D);

%% Matrices en K
% calculo de las matrices de la dinamica del sistema en t
[fi, T ]= c2d(sys.A,sys.B,t);

%% Ubicacion de los polos
%*************************************************************************
%Ackerman
pole1 = -1.5 + 1.5j; % polo 1 en tiempo continuo
pole2 = -1.5 - 1.5j; % polo 2 en tiempo continuo

% pole1 = -103.93 + 87.1j; % polo 1 en tiempo continuo
% pole2 = -103.93 - 87.1j; % polo 2 en tiempo continuo

z1 = exp(pole1*t); % polo 1 en tiempo discreto
z2 = exp(pole2*t); % polo 2 en tiempo discreto

L = acker(fi,T,[z1 z2]);
%*************************************************************************
%LQR
Q = [1 0; 0 1];
R = 1;
%L = dlqr(fi,T,Q,R);


%% Actualizacion de tiempos
% estimacion a priori des estado del sistema
data.X = (fi * data.X) + ( T * data.uk);
% estimacion a priori de la covarianza
```

```
33 data.P = (fi * data.P * fi') + data.Q;
34
35 %% Actualizacion de medicion
36 % Ganancia de Kalman
37 data.K= ( data.P * data.C' ) / ( ( data.C * data.P * data.C' ) + data.R);
38 % estimacion del estado a posteriori
39 data.X = data.X + data.K*(data.y - ( data.C * data.X) );
40 %disp(data.X(2,1))
41 % estimacion a posteriori de el error de covarianza
42 data.P = (eye(2) - data.K*data.C)*data.P; % calculo de la se al de salida
43 data.u = -L * data.X;
44
45 end
```

Apéndice D

Diseño del controlador

```
1
2  %% Matrices de la Dinamica del Sistema
3  A = [0 1 ; -918.27 -90.90];    % matris de transicion de estado
4  B = [0;918.27];                % matris de trancision de control
5  C = [1 0];                     % matris de salida
6  D = 0;
7
8
9  sys = ss(A,B,C,D);
10
11 %% Datos Kalman Filter
12 Q = 2e-7;       %Covarianza del ruido del poceso
13 R = 8e-5;       % Covarianza del ruido en la medicion
14 P = 1e6;
15 K = 0;
16 X = [0 ; 0];
17 %% Parametros senal de entrada
18 u = zeros(1,200)
19 uk=1
20 Y = step(sys,1); %senal real
21 h=0.005;
22 %% LQR
23 q=[1 0; 0 1];
24 r=1;
25 %% Bucle de lazo cerrado
26 for k=1:length(Y)
27
28     Ys(k) = Y(k)+0.1*(0.1-rand); % senal con ruido
29
30     %% Matrices en K
31     % calculo de las matrices de la dinamica del sistema en t
32     [fi, T ]= c2d(sys.A,sys.B,h);
```

```
33      %lqr
34      pole1 = -1.5 + 1.5j; % polo 1 en tiempo continuo
35      pole2 = -1.5 - 1.5j; % polo 2 en tiempo continuo
36
37      z1 = exp(pole1*h); % polo 1 en tiempo discreto
38      z2 = exp(pole2*h); % polo 2 en tiempo discreto
39
40      L = acker(fi,T,[z1 z2]);
41
42      %% Actualizacion de tiempos
43      % estimacion a priori des estado del sistema
44      X = (fi * X) + ( T * uk);
45      % estimacion a priori de la covarianza
46      P = (fi * P * fi') + Q;
47
48      %% Actualizacion de medicion
49      % Ganancia de Kalman
50      K= ( P * C' ) / ( ( C * P * C' ) + R);
51      % estimacion del estado a posteriori
52
53      X = X + K*(Ys(k) - ( C * X) );
54      % estimacion a posteriori de el error de covarianza
55      P = (eye(2) - K*C)*P;
56      % calculo de la se al de salida
57      %
58      %u(k) = C * X;
59      u(k) = -L * X;
60
61
62  end
63
64  plot(Y)
65  hold on
66  plot(Ys)
67  hold on
68  plot(u)
69  hold on
70  ylim([0 1.5])
71  xlim([0 700])
72  legend('referencia', 'ruido', 'algoritmo de control')
```