Planificación por prioridades para sistemas SISO a través de un controlador

J.A. Jiménez Benítez[1], J.J. Medel Juárez[2]
[1]*Centro de Investigación en Ciencia Aplicada y Tecnología Avanzada, Instituto Politécnico Nacional, Calzada Legaria 694 Col. Irrigación, 11500, México D. F.*
[2]*Centro de Investigación en Computación, Instituto Politécnico Nacional, Calle Venus S/N, Col. Nueva Industrial Vallejo, 07738, México D. F.*

E-mail: alfredojimben@hotmail.com

(Recibido el 2 de Noviembre de 2010; aceptado el 18 de Diciembre de 2010)

Resumen

Comúnmente los procesos de planificación son utilizadas debido a que existe más de una tarea que requiere usar recursos del sistema computacional dentro de un mismo intervalo de tiempo. El planificador puede ser configurado por una gran variedad de algoritmos de planificación. El planificador puede estructurarse teniendo en cuenta la problemática a resolver, de manera que, dependiendo del problema tiene un algoritmo específico. Para este trabajo se emplean dos algoritmos de planificación: el Round Robin (RR) y el de Prioridades Variables (PV). En general el planificador se considera como un sistema en lazo abierto, sin embargo si se toma como un sistema en lazo cerrado utilizando los elementos de realimentación y acción de control, entonces se puede hacer una comparación de la salida que éste genera con alguna referencia deseada, de manera que se lleve mediante el control, a un nivel de error. El control utilizado es del tipo Proporcional-Integral-Derivativo (PID), descrito en diferencias finitas y de manera recursiva. Se presentan las simulaciones del sistema donde se hace una planificación con ambos algoritmos y las simulaciones del planificador cuando ya se hizo el control en lazo cerrado usando el PID recursivo, finalmente se realiza una comparación de los resultados.

Palabras clave: Planificación, Control PID recursivo en diferencias finitas, Round Robin, Prioridades variables.

Abstract

Commonly the scheduler is considered as a tool applied when more a task requires the resources in the same time interval. Into a computer system, the operational resources are housed into CPU. The scheduler can be structured considering the problematic to be solved, so that, depending of the problem has a specific scheduling algorithm. In the present work we use two scheduling algorithms knowing as Round Robin and the Variable Priority. The scheduler could be seen as an open loop system, without actions that solve the convergence to a reference condition; however if the scheduler take a closed loop condition requires a feedback elements and establish the control actions, allow it makes approximation to output generated to a reference, so that, the control strategy carried, minimize the level of error convergence. As a basic application, the control considered is the PID controller knowing as Proportional-Integrate-Derivative. The paper presents simulations of the system where planning is done with both algorithms and then present simulations of the planner when it was control over it, comparing with the system without control.

Keywords: Scheduling, PID Control, Round Robin Algorithm, Priority Variables Algorithm.

PACS: 02.30.Yy; 02.70.Bf, 02.70.-c, 02.60.-x, 89.20.Ff **ISSN 1870-9095**

I. INTRODUCCIÓN

Los sistemas computacionales requieren de un sistema operativo como interfaz entre las tareas que se llevan a cabo y el hardware. El sistema operativo es un conjunto de programas o software que administra los recursos del sistema y que sirve como interfaz al usuario para la realización de tareas sobre el sistema. En suma el sistema de hardware y el sistema operativo o de software forman un sistema computacional [1, 2].

Una *tarea o trabajo* en un sistema computacional es una actividad o elemento que está diseñada para ocupar un recurso con el objetivo de alcanzar un resultado deseado que debe estar definida en términos de acciones concretas para realizarse en tiempos finitos. Una tarea está formada por uno o más procesos, los cuales son instancias de programas. Las tareas pueden tener asociadas propiedades tales como: nombre, recursos que necesita, tiempos de ejecución o realización probables, prioridad, entre otros [3, 4].

Cuando existen dos o más tareas concurrentes para ejecutarse en el sistema, éstas pueden ser planificadas y ejecutadas dependiendo el tipo de sistema donde se vayan a ejecutar. Si el sistema es por lotes, entonces las tareas se introducen al sistema una a una y después de concentrar un conjunto de tareas específico se comienza la ejecución del

lote realizando una tarea a la vez. Si el sistema es multiprogramado, entonces las tareas se concentran en la memoria del sistema listas para ejecutarse en el momento en que el sistema decida realizarlas [5, 6].

La forma en cómo decide el sistema operativo cuál de las tareas que están listas se ejecutará y permitirle ocupar los recursos del sistema, depende del planificador. Un *planificador* es un conjunto de programas escritos en bajo y medio nivel que forman parte del sistema operativo y que elige qué tarea ocupará los recursos del sistema en un intervalo de tiempo determinado. Un algoritmo de planificación está basado en algún conjunto de reglas que dictan una política de planificación. Una *política de planificación* depende del tipo de sistema y éstos pueden ser por lotes, interactivos, en tiempo real entre otros [7, 8].

Un problema a considerar que se presenta con los planificadores es que se comportan como un sistema de control en lazo abierto, como se muestra en la Figura 1, lo cual implica que no existe algún medio de controlar su evolución y por consiguiente su desempeño. Un sistema como éste, es un sistema sin elemento de realimentación que puede presentar una salida no deseada en cualquier intervalo de tiempo.

FIGURA 1. Se muestra el diagrama a bloques de un planificador-controlador donde en la entrada están las solicitudes de las tareas que desean ser atendidas y en la salida el conjunto de tareas organizadas por el sistema.

En un sistema donde se requiere planificación como lo son los sistemas computacionales, el hecho de estar en lazo abierto permite que se puedan generar algunas de las siguientes situaciones: a) el número de tareas que están pidiendo atención del sistema computacional son mayores de los recursos, b) las tareas llegan con exigencia de recursos computacionales al contar con la misma prioridad de ser atendidas, c) el hardware no es el adecuado para cumplir con los trabajos solicitados por la tareas y simplemente se descartan.

En estas circunstancias, el sistema de planificación no puede cumplir con sus objetivos.

En general, los planificadores son algoritmos diseñados para resolver una problemática en específico, tal es el caso del conjunto de tareas en tiempo real que se pueden planificar con el algoritmo EDF (Earlest Deadline First) [9, 10, 11, 12].

Los algoritmos de planificación están diseñados resolver un conjunto de tareas, con propiedades en específico; es decir, no existe un algoritmo universal que permita la planificación de cualquier conjunto de tareas. Razón por la que se tienen sistemas con algoritmos de planificación para trabajar con tareas en tiempo real, otros algoritmos trabajan con tareas de arduo procesamiento y otros más trabajan con tareas de procesamiento interactivo como el RR [13, 14, 15, 16].

Debido a que se puede presentar la posibilidad de que el sistema computacional falle por las circunstancias ya mencionadas y que no existe un algoritmo que planifique cualquier conjunto de tareas, debería existir algún mecanismo que permitiera superar este problema.

Para prevenir este problema el sistema de planificación se debe ajustarse a sí mismo, lo cual requiere de estar observando su comportamiento y acotarlo a un error mínimo de acuerdo a una referencia de uso de los recursos computacionales. Por lo mismo se requiere tener una arquitectura de control realimentada.

Contar con un control integrado en el planificador permite realimentarlo y llevarlo a una condición de operación deseada.

En la Teoría de Control Moderna, el elemento de la realimentación, proporciona una forma de hacer comparaciones de la salida a una referencia, como se puede ver en la Figura 2. Esta comparación permite ajustar los valores de la entrada para que el sistema se aproxime a la referencia. Entre menos error haya con respecto a la señal de referencia, implica que se está cumpliendo con los objetivos del sistema de planificación, siendo estable [17].

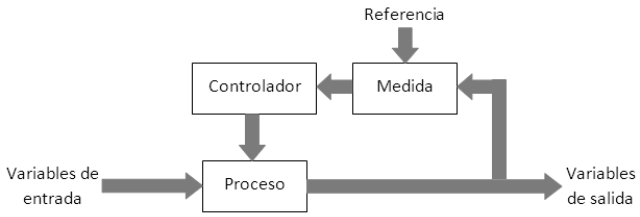

FIGURA 2. Elementos de un sistema de control en lazo cerrado.

La idea de tener un control en los sistemas computacionales se ha visto dentro de los sistemas distribuidos utilizando algoritmos inteligentes [4, 18].

En los sistemas que requieren de planificación, es necesario tener métricas para la comparación en el proceso de control, de manera que exista la referencia adecuada para la corrección del sistema. Se han mostrado algunas métricas en sistemas de tiempo real, como el factor de utilización, el cual es la suma de los cocientes de los tiempos de ejecución entre los periodos de respuesta de cada proceso [11, 12, 13, 14].

II. DESCRIPCIÓN DEL PLANIFICADOR CON UN CONTROLADOR DEL TIPO PID

Un sistema de control descrito a bloques según la Fig. 3, cuenta con un bloque conocido como planta ($g(\tau)$) y un elemento de realimentación ($y(\tau)$), tiene una función de transferencia ($f(\tau)$) como en (1), que relaciona a la salida

($y(\tau)$) con la entrada ($r(\tau)$) que es la señal de referencia, y la señal de error generada ($e(\tau)$) por la diferencia entre $r(\tau)$ y $y(\tau)$.

$$f(\tau) = \frac{g(\tau)}{1+g(\tau)} \qquad (1)$$

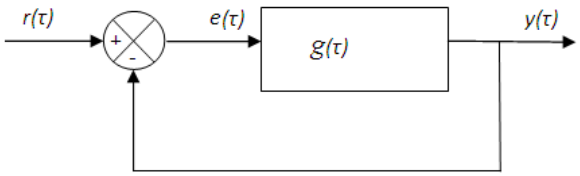

FIGURA 3. Sistema de control clásico con realimentación.

En el caso de los sistemas computacionales, $g(\tau)$ es la función que representa el proceso de planificación expresado por un bloque llamado *Planificador* y que se describirá como $P(\tau)$.

La Fig. 4, muestra un sistema de control formado por una función de planificación y una función de control; contando así con un *Planificador-Controlador* ($P(\tau)$) realimentado.

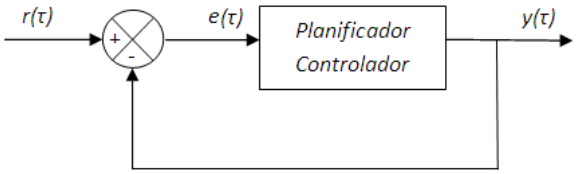

FIGURA 4. Elementos de un sistema de control en lazo cerrado.

La función de planificación $P(\tau)$ se define como el cociente del número de tareas terminadas en un intervalo de tiempo τ entre el número de tareas que entran al sistema y están listas para ser ejecutadas hasta ese momento τ.

El algoritmo que utiliza el planificador puede ser el que el sistema requiera para que se ejecuten el conjunto de tareas a tiempo y con calidad de ejecución. Para el presente trabajo se eligieron dos algoritmos conocidos y utilizados en los sistemas computacionales tradicionales, tal como el RR y el PV. Ambos utilizados tanto en sistemas en Tiempo Real como en sistemas interactivos [4, 9, 10].

Para el sistema mostrado en la Fig. 4, la acción de control sobre el planificador proporciona información del error de planificación. Debido a esto se requiere de un bloque que analice la información del error y realice la acción de control sobre el planificador tal y como se muestra en la Fig. 5, donde $u(\tau)$ representa la acción de control sobre el planificador.

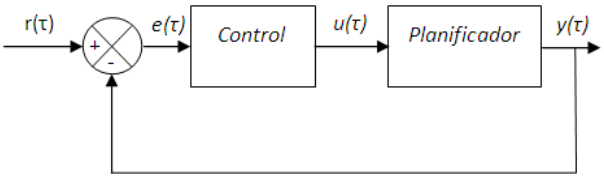

FIGURA 5. Acción de control sobre un Planificador.

Al considerar el bloque de control en el sistema de la Fig. 5, la Ec. (1) se transforma en la Ec. (2).

$$y(\tau) = \frac{C(\tau)P(\tau)}{1 + C(\tau)P(\tau)} \qquad (2)$$

En la Ec. (2) $C(\tau)$ representa al bloque de Control y $P(\tau)$ representa al bloque del Planificador.

III. TEOREMA

La acción de control $u(\tau)$ en diferencias finitas, considerando las propiedades del PID, así como los tiempos de evolución del sistema (ε), de acuerdo con Nyquist y Shannon y a la Fig. 5, es descrito en la Ec. (3).

$$u(\tau) = u(\tau-\varepsilon) + K_p\left(e(\tau)\left[1+\frac{\varepsilon}{T_i}+\frac{T_d}{\varepsilon}\right] - e(\tau-\varepsilon)\left[1+\frac{2T_d}{\varepsilon}\right] + e(\tau-2\varepsilon)\left[\frac{T_d}{\varepsilon}\right]\right) \qquad (3)$$

En la Ec. (3) K_p, T_i y T_d representan las contantes proporcional, integral y derivativa, respectivamente

A. Prueba

El bloque de control en tiempo continuo en la Ec. (4), que modifica la ejecución del sistema planificador está descrito por el funcional error de seguimiento *u(e(t))*, de acuerdo con [17, 19, 20, 21].

$$u(t) = K_p\left(e(t) + \frac{1}{T_i}\int e(t)\,dt + T_d\frac{de(t)}{dt}\right) \qquad (4)$$

El error de seguimiento dentro de un sistema computacional es discreto $e(\tau) := r(\tau) - y(\tau)$. Derivando a la Ec. (4)

$$\frac{du(t)}{dt} = K_p\left(\frac{de(t)}{dt} + \frac{1}{T_i}e(t) + T_d\frac{d^2e(t)}{dt^2}\right) \qquad (5)$$

Considerando que la derivada de la Ec. (5) es descrita por límites en la Ec. (6).

$$\frac{du(t)}{dt} := \lim_{\Delta(t)\to 0}\frac{(u(t+\Delta(t)) - u(t))}{\Delta(t)} \qquad (6)$$

Haciendo la consideración de que un sistema discreto no se conoce el futuro de la excitación, y de que permanece el cociente invariante en el tiempo, se tiene la Ec. (7):

$$\frac{du(t)}{dt} := \lim_{\Delta(t)\to 0} \frac{(u(t) - u(t-\Delta(t))}{\Delta(t)} \tag{7}$$

Cuando la Ec. (7) representará las condiciones de operación de un sistema computacional, $\Delta(t) \to \varepsilon$, de acuedo con Nyquist y Shannon, en donde el incremento de ε representa la diferencia de tiempo entre estados en el espacio discreto y se describe en la Ec. (8).

$$\frac{du(t)}{dt} \approx \frac{(u(t) - u(t-\varepsilon))}{\varepsilon} \tag{8}$$

De igual forma ocurre para el error de seguimiento $e(t)$ expresado en diferencias finitas en la Ec. (9) al considerar la Ec. (8).

$$\frac{de(t)}{dt} \approx \frac{e(\tau) - e(\tau-\varepsilon)}{\varepsilon} \tag{9}$$

De acuerdo con la Ec. (7) al derivarla y al considerar las condiciones de operación de un sistema computacional con $\Delta(t) \to \varepsilon$, la segunda derivada del error de seguimiento es descrita en la Ec. (10).

$$\frac{d^2e(t)}{dt^2} \approx \frac{e(\tau) - 2e(\tau-\varepsilon) + e(\tau-2\varepsilon)}{\varepsilon^2}, \tag{10}$$

Las aproximaciones discretas de las Ecs. (8), (9) y (10) remplazadas en la Ec. (5), y que al organizar términos, se describe a la Ec. (3).

Utilizando el sistema mostrado en la Fig. 5 se realizaron pruebas con los dos algoritmos de planificación mencionados arriba.

IV. SIMULACIÓN

La Figura 6 muestra el accionar de un sistema planificado por un algoritmo del tipo Round Robin en donde se generan los procesos de forma estocástica formando un banco de procesos iniciales, las cuales representan los procesos que todo sistema computacional tiene trabajando en segundo plano. Conforme el tiempo transcurre se van generando procesos de forma aleatoria, las cuales representan los requerimientos por parte de los usuarios o del mismo sistema debido a eventos del mundo exterior.

La Fig. 7 muestra el accionar de un sistema planificado por un algoritmo del tipo Prioridades Variables, y las condiciones de creación de los procesos son lo descritos para el algoritmo Round Robin.

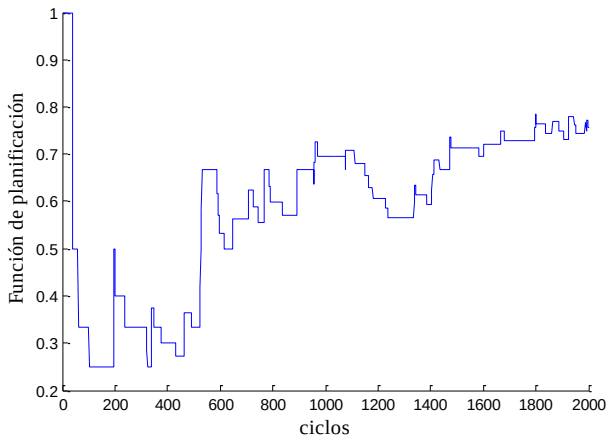

FIGURA 6. Se muestra el desenvolvimiento de la función de planificación del algoritmo Round Robin en 2000 ciclos de ejecución.

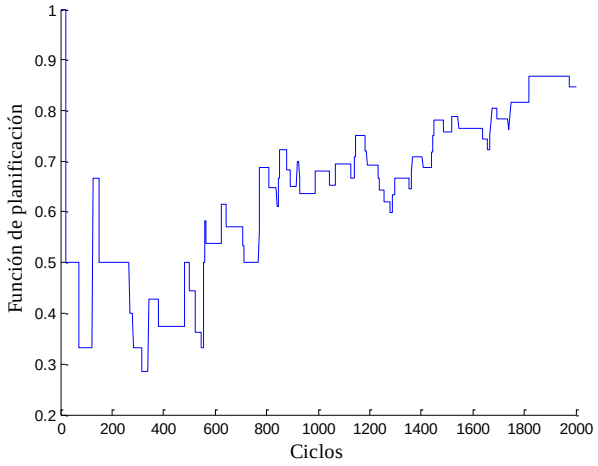

FIGURA 7. Se muestra el desenvolvimiento de la función de planificación del algoritmo Prioridades Variables en 2000 ciclos de ejecución.

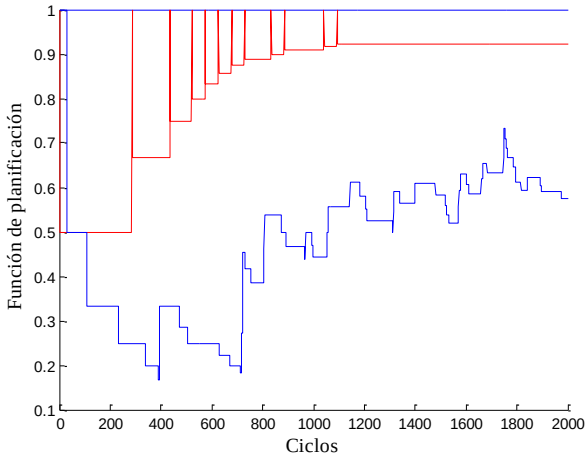

FIGURA 8. Se muestra el desenvolvimiento de la función de planificación del algoritmo Round Robin en 2000 ciclos de ejecución en comparación con el control PID aplicado al mismo algoritmo.

En las Figs. 8 y 9 se muestran los resultados de implementar un control del tipo PID a los algoritmos Round Robin y Prioridades Variables respectivamente. En ambas figuras el color azul representa la función de planificación sin el sistema de control propuesto y el color rojo representa la función de planificación con el control PID modificando el desenvolvimiento del sistema.

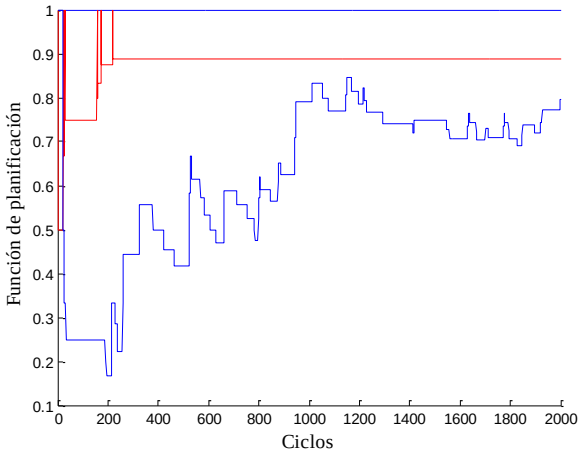

FIGURA 9. Se muestra el desenvolvimiento de la función de planificación del algoritmo Prioridades Variables en 2000 ciclos de ejecución en comparación con el control PID aplicado al mismo algoritmo.

V. CONCLUSIONES

En los sistemas donde la planificación es una herramienta algorítmica para saber cómo ejecutar un conjunto de tareas, como lo son los sistemas computacionales, es importante considerar que la planificación es un sistema de lazo abierto, por lo tanto no se tiene una forma de hacer control ya que no se cuenta con el elemento de realimentación que permite hacer una comparación de la señal de salida con alguna referencia deseada. Un control convencional, como lo es el tipo PID puede hacer que un sistema donde se emplea la planificación pueda ser controlado y llevado a condiciones de planificabilidad deseadas.

En ambos algoritmos aplicar el control PID implica hacer ajustes a las contantes del control ya que la forma en cómo ejecutan las tareas es distinta. El nivel de control que se aplica al planificador con el algoritmo Round Robin es del 5% de permisión de entrada de un procesa al sistema, mientras que el ajuste que se hace al planificador con el algoritmo de Prioridades Variables es del 10%. Esto quiere decir que aunque el control es aplicado a ambos planificadores se requiere hacer ajustes a posteriori.

En lo que respecta al ajuste del control del tipo PID, la constante que varía en la sintonización es la de la parte proporcional, ya que en el planificador con el algoritmo Round Robin es de K_p=1, mientras que en el planificador con Prioridades Variables es K_p>>1, tomandose el valor de 20 para las simulaciones ya presentadas. Los valores de las partes, integral y derivativa se mantienen igual para ambos.

REFERENCIAS

[1] Deitel, H. M., *Introducción a los Sistemas Operativos*, (Addison-Wesley Iberoamericana, México, 1987).
[2] Tanenbaum, A. S., *Sistemas Operativos Modernos*, (Pearson Educación, México, 2009).
[3] Tanenbaum, A. S., *Sistemas Operativos Distribuidos*, (Pearson Educación, México, 1996).
[4] Tanenbaum, A. S. and Steen, M. V., *Sistemas Distribuidos*, (Pearson Educación, México, 2008).
[5] Murdocca, M. J. and Heuring, V. P., *Principios de Arquitectura para Computadoras*, (Pearson Educación, México, 2002).
[6] Dhamdhere, V., *Sistemas Operativos, un enfoque basado en conceptos*, (Mc Graw Hill, México, 2008).
[7] Silberschatz, A., Galvin, P. B. and Gagne, G., *Fundamentos de Sistemas Operativos*, (Mc Graw Hill, México, 2006).
[8] Flynn, A. and McHoes, M., *Understanding Operative Systems*, (Thomson Learning, USA, 2006).
[9] Guevara, P. and Medel, J. J., *Introducción a los Sistemas en Tiempo Real*, (Instituto Politécnico Nacional, Ciudad de México, 2003).
[10] Medel, J. J., Guevara, P. and Cruz, D., *Temas Selectos de Sistemas en Tiempo Real*, (Instituto Politécnico Nacional, Ciudad de México, 2007).
[11] Buttazzo, G. C., *Hard real-time computing systems*, (Scuola Superiore S. Anna, (Kluwer. Academic Publishers, Pisa Italia, 1997).
[12] Liu, J. W. S., *Real-time Systems*, (Prentice Hall, USA, 2000).
[13] Liu, C. L. and Layland, J. W., *Scheduling Algorithms for multiprogramming in Hard Real-Time Environments* Journal of ACM 20, 46-61 (1973).
[14] Burns, A., *Scheduling Hard Real Time Systems: A Review*, Software Engineering Journal **6**, 116-128 (1991).
[15] Audsley, N. and Burns, A., *Real Time Systems Scheduling*, (1995), http://beru.univ-brest.fr/~singhoff/cheddar/publications/audsley95.pdf, colsultado el 10 de Octubre de 2010.
[16] Jeffay, K., *Scheduling Task with Shared Resources in Hard-Real-Time Systems*, Proc. 13th IEEE Real-Time Systems Symposium, Phoenix, AZ, Dec. (1992) pp. 89-99.
[17] Ogata, K., *Ingeniería de Control Moderna*, (Prentice Hall, 2003).
[18] Coulouris, G., Dollimore, J. and Kindberg, T., *Sistemas Distribuidos: Conceptos y Diseño*, (Pearson Educación, México, 2001).
[19] Dorf, R. C. and Bishop, R. H., *Sistemas de Control Moderno* (Pearson, México, 2005).
[20] Kuo, C., *Sistemas de Control Automático*, (Prentice Hall, México, 1996).
[21] Kuo, C., *Sistemas de Control Digital*, (CECSA, México, 1996).