XXV Jornadas de Automática
Ciudad Real, del 8 al 10 de septiembre de 2004

Modelos híbridos de procesos de fabricación

José María González de Durana, José Manuel López
Departamento de Ingeniería de Sistemas y Automática
Universidad del País Vasco, EUITI, Nieves Cano 12, 01006 Vitoria, España
e-mail: jtpgogaj@vc.ehu.es, ccbloguj@sc.ehu.es

Jorge Orellana, Ingeborg Mahla
Departamento de Ingeniería Eléctrica
Universidad de Santiago de Chile, Av. Ecuador 3519, Casilla 10233, Santiago de Chile
e-mail: jorge.orellana@usach.cl, imahla@lauca.usach.cl

Resumen

Los procesos de fabricación se han modelado tradicionalmente en base a las teorías de procesos estocásticos y de colas. Este tipo de modelado resulta idóneo cuando los datos que se desea obtener de la simulación son datos globales del proceso tales como los valores medios de determinadas variables y otros valores de tipo estadístico. Sin embargo, cuando el modelo debe reproducir con precisión ciertos elementos críticos del proceso, de cara a su posterior implementación, dichos métodos no son los más adecuados.

Un proceso de fabricación es claramente un sistema híbrido ya que se compone de una colección de subprocesos conectados entre sí, algunos de los cuales son de tiempo continuo *mientras que otros son* sistemas reactivos*, es decir, sistemas de eventos discretos.*

*Para el modelado de los sistemas reactivos, una opción es utilizar cartas de estado (*StateCharts*), implementadas bajo el nombre de* Stateflow *en Matlab. Con este método es posible el modelado completo de ciertos sistemas híbridos simples en el entorno de Matlab.*

En este trabajo presentamos el modelado híbrido y la simulación en el entorno de Stateflow y Simulink de un proceso simple de fabricación compuesto de un almacén de piezas de entrada, un proceso de tiempo continuo, una cinta trasportadora y un almacén de piezas de salida. El modelo matemático (híbrido) de este sistema podrá ser objeto de un posterior estudio.

Palabras clave: hybrid systems, reactive systems, switching systems, discrete event systems, statecharts, manufacturing proceses.

1 Introducción

Un sistema híbrido es un modelo matemático que se compone de varios sistemas de eventos discretos (también llamados *sistemas reactivos*), de distinta naturaleza, es decir, unos continuos a trozos y otros discretos en el tiempo, conectados entre sí de tal forma que es posible la comunicación entre ellos por medio de datos o mensajes y eventos. Su funcionamiento se puede explicar intuitivamente por medio del diagrama de la figura 1.

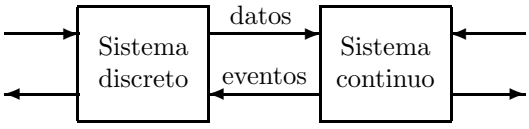

Figura 1: sistema híbrido

Los sistemas híbridos pueden servir para representar una amplia variedad de sistemas físicos: sistemas de tiempo real, software estructurado, robótica, mecatrónica, aeronáutica, automatización de procesos, sistemas electrónicos, sistemas económicos e incluso sistemas biológicos [6].

En los procesos de fabricación, los *sistemas discretos* representan elementos de proceso digital (quizás autómatas programables) mientras que los *sistemas continuos* representan determinados procesos continuos a trozos en el tiempo. La comunicación entre ellos se realiza a través de eventos y/o datos de entrada y salida.

El objetivo de este trabajo es el desarrollo de modelos híbridos computacionales de algunos procesos de fabricación simples y el método se basa en la utilización de cartas de estado (StateCharts).

La descripción formal del un sistema híbrido puede hacerse matemática o computacionalmente, dando lugar a los que denominaremos *modelos híbridos matemáticos* o *modelos híbridos computacionales*, respectivamente.

2 Modelos matemáticos

Un *modelo híbrido matemático* es una descripción matemática del comportamiento en el tiempo de algún ente (físico) y se denomina habitualmente *Sistema Híbrido*. La definición de sistema híbrido dada a continuación se ha tomado de [6].

Definición 2.1 *Un sistema híbrido n-dimensional con k estados es una 6-tupla*

$$\mathbb{H} = (Q, E, \mathcal{D}, \mathcal{X}, \mathcal{G}, \mathcal{R})$$

cuyos elementos son:

- $Q = \{1, \ldots, k\}$ *conjunto finito de estados,*
- $E \subset Q \times Q$ *colección de transiciones,*
- $\mathcal{D} = \{D_i : i \in Q\}$ *colección de dominios donde* $D_i \subset \{i\} \times \mathbb{R}^n$.
- $\mathcal{X} = \{X_i : i \in Q\}$ *colección de campos vectoriales tales que* X_i *es Lipschitz en* D_i *para todo* $1 \leq i \leq k$.
- $\mathcal{G} = \{G(e) : e \in E\}$ *colección de guardas, donde para cada* $e = (i, j) \in E$ *se tiene que* $G(e) \subset D_i$,
- $\mathcal{R} = \{R_e : e \in E\}$ *colección de reinicios, donde para cada* $e = (i, j) \in E$, $\mathcal{R}_e$ *es una relación entre los elementos de* $G(e)$ *y los elementos de* D_j, *es decir,* $R_e \subset G(e) \times D_j$.

Además del de Simić, existen otros posibles esquemas para el modelado híbrido matemático, como puede verse en el libro [7].

La obtención de modelos matemáticos híbridos puede a veces no ser tarea fácil si se pretende abordar directamente, sobre todo si la entidad a modelar es de cierta complejidad ya que los números de estados y transiciones pueden ser elevados. A veces, no obstante, ello puede resultar posible realizando ciertas hipótesis simplificatorias en la descripción de los procesos [4], [5].

En otros casos se puede recurrir al modelado híbrido computacional, el cual, si bien evidentemente no permite realizar análisis matemáticos, sí que resulta apto para la simulación del funcionamiento del sistema, lo que puede ser suficiente en muchas aplicaciones.

3 Modelos computacionales

El lenguaje de las *cartas de estado* (*statecharts*) [1] se ha revelado como uno de los más potentes y eficaces para la descripción de sistemas reactivos, siendo numerosas las publicaciones recientes sobre su aplicación a sistemas complejos de todo tipo, incluso biológicos [2].

En una anterior experiencia con circuitos electrónicos de potencia tuvimos la oportunidad de hallar primero unos modelos híbridos computacionales, basados en cartas de estado [8], que posteriormente nos sirvieron para obtener sus correspondientes modelos matemáticos (sistemas híbridos), procediendo entonces a su estudio matemático [3]. Es por ello que apostamos por aplicar la misma metodología, (es decir, obtener primero el modelo computacional para después, a partir de él, obtener el modelo matemático) a ciertos sistemas de fabricación muy simples cuya descripción se da a continuación.

4 Modelos de procesos simples

La naturaleza de los procesos de fabricación es muy diversa, ya que puede decirse que es diferente para cada producto e incluso que un mismo producto se puede realizar a partir de diferentes procesos.

Un proceso de fabricación es una secuencia de operaciones que se realizan sobre unos materiales, piezas o productos de entrada para obtener uno o varios productos de salida. Cada una de las operaciones que componen un proceso de fabricación es en sí misma un proceso, más simple por supuesto, un subproceso si se quiere, de aquel. Podemos así pensar que un proceso de fabricación se compone de una colección de subprocesos o procesos simples interconectados entre sí, a través de los cuales van pasando los productos hasta la obtención del producto final.

Existe una gran variedad de procesos simples que en la práctica integran los procesos de fabricación, dada la gran variedad de componentes comerciales dedicados a la automatización. Sin embargo, en líneas generales, no es muy grande el número de clases diferentes de los mismos. Los que vamos a considerar en este trabajo son:

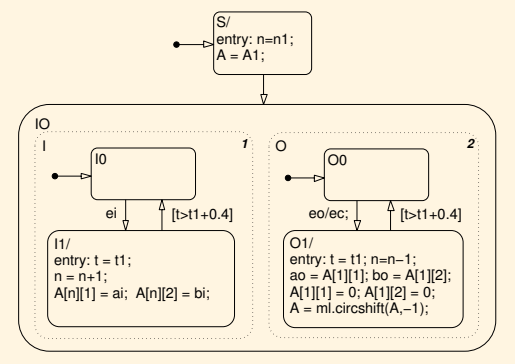

Figura 2: almacén –carta

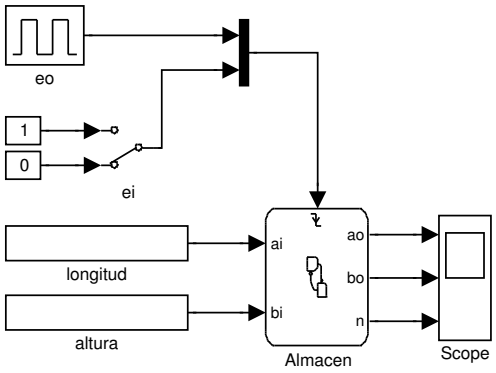

Figura 3: almacén –diagrama

- Manipulador
- Almacén de piezas
- Cinta trasportadora
- Proceso de tiempo continuo
- Autómata programable (PLC)

4.1 Almacén de piezas

Es un recipiente en el que se pueden almacenar un número máximo n_1 de piezas apiladas y que admite dos operaciones: entrada de una pieza por la parte superior, controlada por eventos de la señal e_i, y salida de una pieza por la parte inferior, controlada por eventos de la señal e_o, la cual provoca un desplazamiento vertical, por gravedad, de todas las restantes piezas. Cada pieza tiene una longitud a_i y una altura b_i, para $1 \leq i \leq n$ y supondremos que la primera pieza, para $i = 1$, es la de abajo (figura 8).

El almacén puede estar inicialmente vacío o bien puede contener un número n de piezas.

Para el modelado en Matlab se ha utilizado un bloque *Chart* de Stateflow con 2 entradas de datos a_i, b_i, 3 salidas de datos a_o, b_o, n y 2 entradas de eventos e_i, e_o. Los datos de las piezas se almacenan en una matriz A con n_1 filas, una para cada pieza, y 2 columnas para los dos parámetros a y b de cada pieza. El orden de las filas de A corresponde al orden inverso de las piezas en el almacén: la fila 1 de A contiene los datos de la primera pieza, la colocada en la parte inferior (la última) en el almacén.

La carta de estados (figura 2) tiene dos estados principales llamados S e IO. El estado S es el inicial y se emplea para asignar a la variable n y la matriz A los valores n_1 y A_1 previamente colocados en el *workspace* de Matlab. El estado IO tiene dos estados hijos llamados I y O que funcionan en paralelo, cada uno de los cuales tiene a su vez dos hijos. Los estados hijos de I son I_0 e I_1 mientras que los de O son O_0 y O_1. Si la carta recibe un flanco de bajada en la señal e_i, su estado pasa de I_0 a I_1 en donde se incrementa en una unidad el valor de n y se introducen en en la fila n de la matriz A los valores a_i y b_i de la longitud y anchura de la pieza que ha entrado, pasando entonces el sistema de nuevo al estado I_0, a la espera de recibir un nuevo evento en la señal e_i. En paralelo, si la carta recibe un flanco de bajada en la señal e_o, su estado pasa de O_0 a O_1 en donde los dos valores de la primera fila de la matriz A se asignan a las salidas a_o y b_o, los elementos de la matriz A se desplazan hacia arriba y el valor de n se decrementa en uno, pasando entonces el sistema de nuevo al estado O_0, a la espera de recibir un nuevo evento en la señal e_0.

Para visualizar el funcionamiento del bloque *Almacén* se ha dispuesto el esquema de Simulink de la figura 3. La entrada de piezas se realiza manualmente con el *switch* e_i, de tal forma que en cada flanco de bajada de la señal de e_i se produce la entrada de una pieza. Se han utilizado dos bloques *slider* de Simulink para poner la longitud y la altura de cada pieza. Cada uno de ellos permite introducir un valor numérico desplazando, con el ratón, una marca deslizante (no se aprecia en la figura). Para la salida de piezas se utiliza un generador de señal cuadrada e_o con periodo de 10 s de manera que en cada flanco de bajada de e_o se producirá la salida de una pieza. La transición e_o genera el evento local e_c que se puede retransmitir a otras cartas de estado.

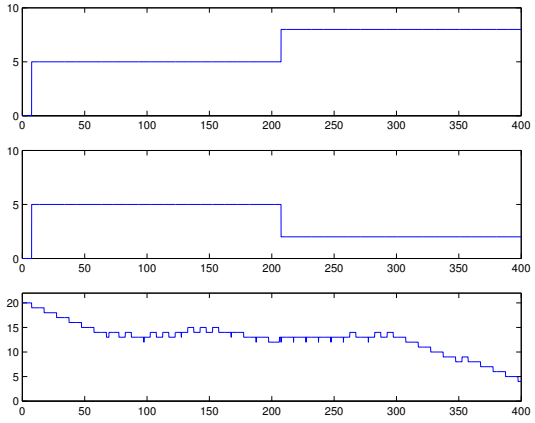

Figura 4: almacén –señales a_o, b_o y n

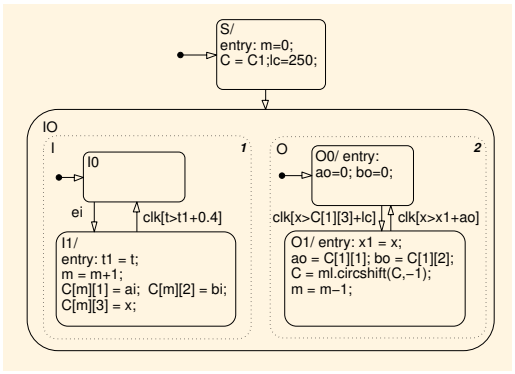

Figura 5: cinta –carta

En la figura 4 se ven las señales de salida a_o, b_o y n. En la simulación realizada, la matriz A contenía 20 piezas iguales con $a = b = 5$ y los *sliders* se mantuvieron fijos con el valor de *longitud* $= 8$ y el de *altura* $= 2$. Se observa que en el instante $t = 205$s las señales a_o y b_o de salida cambian, al haberse agotado todas las piezas iguales y empezar a salir las introducidas mediante los *sliders*. Se puede ver también que la señal n, número de piezas, va disminuyendo de forma más o menos pronunciada dependiendo de la mayor o menor frecuencia con que se accionó el *switch* durante la simulación.

4.2 Cinta trasportadora

Una cinta trasportadora funciona en cierto modo como un almacén de piezas: las piezas entran a la cinta, ésta las almacena temporalmente y finalmente salen de ella. Sin embargo, lo que va a determinar la salida de las piezas no son eventos de entrada, como en el caso del almacén, sino los eventos producidos por las llegadas de las piezas al final de la cinta. Supondremos que la pieza que ocupa el lugar i se caracteriza por tres parámetros, su longitud a_i y su altura b_i, igual que en el caso del almacén, y además el parámetro x_i que denota el desplazamiento acumulado de la cinta en el instante que la pieza entró en ella.

El modelado de la cinta trasportadora en Matlab se ha realizado utilizando un bloque *Chart* de Simulink (figura 5) con 3 entradas de datos a_i, b_i, x_i, 3 salidas de datos a_o, b_o, m y 2 entradas de eventos *Clk*, e_i. Los datos de las piezas se almacenan en una matriz C con m filas, una para cada pieza, 2 columnas para los dos parámetros a_i y b_i de cada pieza y una tercera columna para x_i. La carta de estados tiene dos estados principales llamados S e IO. El estado S es el inicial y se emplea para asignar a la variable m y la matriz C los valores 0 y C_1, éste colocado previamente en el *workspace* de Matlab. El estado IO tiene dos estados hijos llamados I y O que funcionan en paralelo, cada uno de los cuales tiene a su vez dos hijos. Los estados hijos de I son I_0 e I_1 mientras que los de O son O_0 y O_1. Si la carta recibe un flanco de bajada en la señal e_i, su estado pasa de I_0 a I_1 en donde se incrementa en una unidad el valor de m y se introducen en en la fila m de la matriz C los valores a_i, b_i de la longitud y anchura de la pieza que ha entrado y x_i del desplazamiento acumulado de la cinta en ese instante; el sistema pasa entonces de nuevo al estado I_0. En paralelo, la transición entre O_0 y O_1 se produce cuando se cumple la condición $x > C_{13} + lc$, es decir, cuando el desplazamiento x de la cinta es mayor que el desplazamiento C_{13} que tenía la cinta cuando entró en ella la pieza que ahora está en primer lugar más la longitud lc de la cinta; el sistema pasa entonces de nuevo al estado O_0. Los eventos nece-

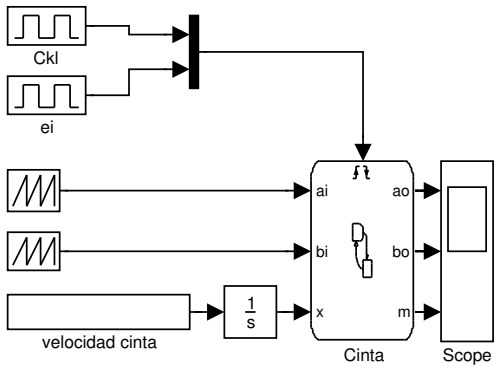

Figura 6: cinta –diagrama

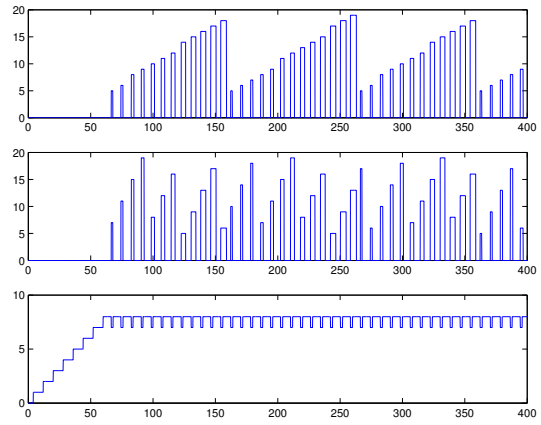

Figura 7: cinta –señales a_o, b_o y m

sarios para la transiciones los proporciona la señal de sincronismo *Clk*.

Para visualizar el funcionamiento del bloque *Cinta* se ha dispuesto el esquema de Simulink de la figura 6. Para la entrada de piezas se utiliza un generador de señal cuadrada e_i con periodo de 8 s de manera que en cada flanco de bajada de e_i se producirá la entrada de una pieza. La longitud y la anchura de cada pieza se introducen mediante dos generadores de señal triangular. La velocidad de la cinta se puede variar manualmente mediante un *slider*.

En la figura 7 se ven las señales de salida a_o, b_o y m. Se observa que al principio va aumentando el valor m de piezas en la cinta hasta que al llegar a 8, transcurridos unos 60 s, comienzan a salir las piezas.

4.3 Controlador

En un proceso de estos el controlador suele ser un autómata programable, un ordenador o un controlador digital de algún otro tipo.

El funcionamiento del controlador lo podemos modelar con una carta de estados. La carta recibe eventos y datos de entrada y emite datos (órdenes) de salida. Los eventos y datos de entrada pueden provenir de cualquiera de los procesos del sistema, según sean las operaciones que se quieran implementar.

4.4 Manipulador

Un manipulador se utiliza para desplazar y colocar las piezas en determinados lugares. En Simulink puede hacerse con cierta facilidad modelos de manipuladores sencillos.

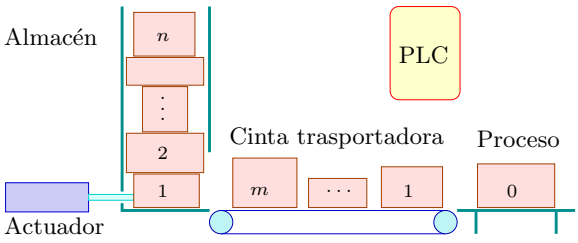

Figura 8: mini-proceso de fabricación

5 Modelado de procesos

Los bloques que hemos desarrollado pueden conectarse entre sí de manera que las mismas piezas que salen de un bloque son las que entran al siguiente. Como hemos visto, la conexión se efectúa a través de señales (datos y eventos) en el entorno de Simulink. Esto nos permite obtener modelos de algunos procesos de fabricación. Como ejemplo, en la siguiente sección vamos a obtener el modelo de un mini-proceso de fabricación.

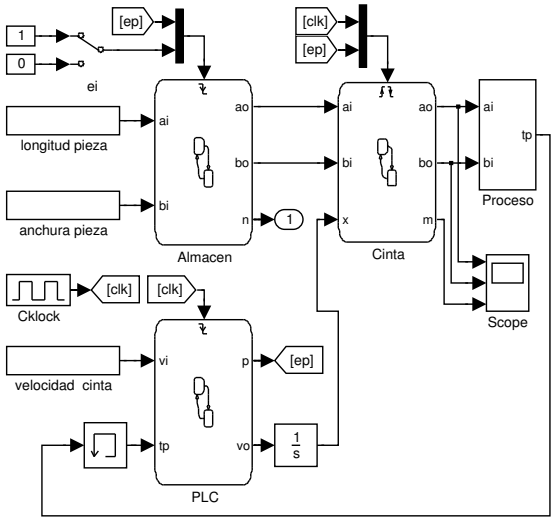

Figura 9: mini-proceso

5.1 Mini-proceso de fabricación

Como ilustración de las posibilidades de los bloques que hemos desarrollado, se ha ideado un proceso de fabricación muy simple, al que denominamos mini-proceso de fabricación, que es el que aparece en la figura 8. Se compone de un actuador, un almacén de piezas, una cinta trasportadora, un

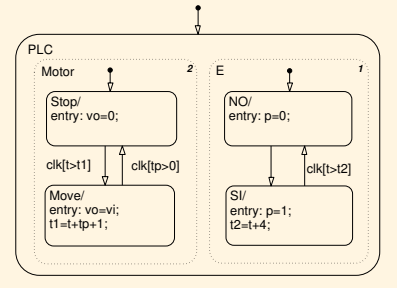

Figura 10: autómata programable –carta

proceso continuo, sensores (no representados en la figura) y un autómata programable PLC.

El mini-proceso es el siguiente. Las piezas se introducen manualmente en el almacén en donde se apilan por gravedad. El actuador, controlado por el autómata, empuja la pieza número 1 y la coloca en la cinta, en la última posición, con lo que el número n de piezas en el almacén se decrementa en una unidad y el número m de piezas en la cinta se incrementa en una unidad. La cinta va llevando las piezas desde el almacén hasta el proceso continuo a una velocidad v_o controlada por el autómata. Al llegar la primera pieza al final de la cinta, pasa al proceso como pieza 0, con lo que el número de piezas en la cinta se decrementa en una unidad.

El bloque *Proceso* simula un determinado proceso que se efectúa sobre cada pieza, cuya duración t_p es función de las entradas a_i y b_i.

El autómata controla la velocidad v_o de la cinta en función del proceso: mientras hay pieza procesando pone la velocidad v_o a cero y cuando termina de procesarse la pieza pone la velocidad v_o de la cinta al valor dado por la entrada v_i.

5.1.1 Observaciones

Hemos tratado de obtener un modelo muy simple de mini-proceso, para destacar los aspectos esenciales del sistema híbrido y facilitar la comprensión del lector, y por ello nos hemos visto forzados a sacrificar algunos detalles. Así, por ejemplo, no se ha tenido en cuenta los tiempos de proceso del *Actuador* que empuja las piezas desde el *Almacén* a la *Cinta* ni de otro posible manipulador (no representado) encargado de llevar la pieza desde la cinta hasta el proceso.

Por falta de espacio no hemos dado una explicación exhaustiva de la implementación. Así, no se

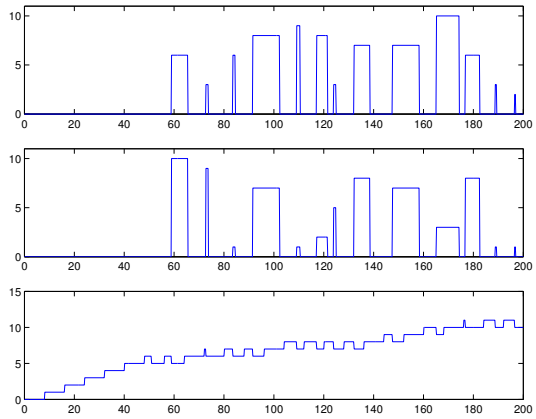

Figura 11: señales de salida

explica algunos detalles de la misma, tales como la retransmisión de eventos y declaración de variables y eventos que es necesario hacer en el entorno de *Stateflow*.

El sistema propuesto puede servir de base para realizar estudios sobre diferentes alternativas y estrategias de fabricación porque, por ser modular, es fácilmente configurable y adaptable a las características del proceso objeto de estudio.

6 Conclusiones

- El lenguaje de las Cartas de Estado ha facilitado notablemente el estudio del funcionamiento de algunos sistemas de fabricación simples, no sólo en lo que respecta al elemento controlador sino también al resto de los elementos que los integran. Ello ha permitido el desarrollo de bloques elementales, asociados a estos procesos simples, que pueden conectarse entre sí para crear modelos de cierta complejidad.
- La inclusión en el entorno Matlab permite un cómodo acceso a los cálculos matemáticos.
- Con los modelos desarrollados es posible modelar un sistema de fabricación completo, con controlador, actuadores, sensores y procesos continuos.
- Además de la simplicidad para programación, los modelos desarrollados también aportan simplicidad conceptual; por ejemplo permiten visualizar rápidamente las ejecuciones (o secuencias de modos continuos) posibles, a través de la visualización de diferentes combinaciones de transiciones que son factibles.

- La generación automática de código de que dispone Stateflow puede permitir una rápida realización del sistema simulado, pudiendo así ayudar en la realización de prototipos rápidos (*rapid prototyping*) del sistema.
- Es destacable la simplicidad de los métodos propuestos, en claro contraste con otros procedimientos basados en realidad virtual que requieren programas específicos, sofisticados y de lenta y difícil elaboración.
- La inclusión en el entorno de Matlab puede facilitar el posterior desarrollo de modelos matemáticos híbridos.
- Con una Red de Petri (o con Grafcet) es posible modelar el sistema reactivo correspondiente al autómata programable. Pero para disparar las transiciones es preciso montar físicamente todo el sistema (autómata + actuadores + sensores + procesos) y hasta entonces es posible apreciar si todo funciona correctamente. Sin embargo con nuestros modelos se puede simular todo el sistema: (autómata + actuadores + sensores + procesos).
- Posponemos para un posterior trabajo el estudio de modelos híbridos matemáticos de estos sistemas. Tendremos entonces la posibilidad de realizar análisis de estabilidad en algunos casos simples (no sabemos si esto último podrá ser válido en sistemas tan complejos como los procesos de manufactura; habría que esperar a terminar el desarrollo para vislumbrar otras ventajas). Asimismo, la especificación de dominios de cada modo podrá permitir inferir desde qué condiciones iniciales se puede alcanzar las ejecuciones deseadas (se puede garantizar la ejecución deseada si se establece que ella es una solución factible y existe convergencia hacia ella).

Referencias

[1] D. Harel. Statecharts: a visual formalism for complex systems. *Science of Computer Programming*, 8:231–274, 1987.

[2] D. Harel. A grand challenge: Full reactive modeling of a multi-cellular animal (invitedtalk). In *Hybrid Systems: Computation and Control, 6th International Workshop, HSCC 2003 Prague, Czech Republic, April 3-5, 2003, Proceedings*, volume 2623 of *Lecture Notes in Computer Science*. Springer, 2003.

[3] I. Mahla, J. Orellana, E. Zulueta, T. Rico, and J.M. González de Durana. Análisis del modelo híbrido para convertidores dc-dc en lazo abierto. In *Actas del XI Congreso Latinoamericano de Control Automático*, La Habana, Cuba, May 2004.

[4] A.S. Matveev and A.V. Savkin. *Qualitative Theory of Hybrid Dynamical Systems*. Birkhäuser, 2000.

[5] J.R. Peerkins and P.R. Kumar. Stable, distributed, real-time scheduling of flexible manufacturing/assembly/disassembly systems. *IEEE Transactions on Automatic Control*, 34(2):139–148, 1989.

[6] S.N. Simić, K.H. Johanson, S. Sastry, and J. Lygeros. Towards a geometric theory of hybrid systems. *Hybrid Systems: Computation and Control*, 1790:421–436, 2000.

[7] A. van der Schaft and H. Schumacher. *An Introduction to Hybrid Dynamical Systems*. Springer, 2000.

[8] E. Zulueta, T. Rico, and J.M. González de Durana. Modelos híbridos de convertidores dc-dc en lazo abierto. In *Actas de las XXIV Jornadas de Automática*, León, España, Sept 2003.