Diseño Sistemático de Controladores Difusos Usando Razonamiento Inductivo

Francisco Mugica Alvarez
Departamento de Ingeniería de Sistemas y Automática
Universidad Politécnica de Cataluña
Av. Diagonal 647, 2da. planta
08028 Barcelona, España
E_mail: mujica@esaii.upc.es

TESIS DE DOCTORADO

Presentada en el Departamento de Lenguajes y Sistemas Informáticos, dentro del Programa de Doctorado en Inteligencia Artificial, para obtener el Título de Doctor en Ciencias con Especialidad en Informática

Marzo, 1995.

Directores:

François E. Cellier
Electric & Comp. Eng. Dept.
University of Arizona

Rafael M. Huber
Instituto de Cibernética
Univ. Politécnica de Cataluña

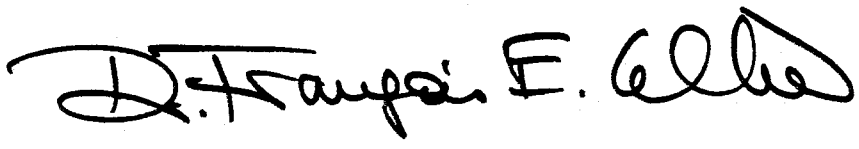
Diseño Sistemático de Controladores Difusos Usando Razonamiento Inductivo

Francisco Mugica Alvarez
Departamento de Ingeniería de Sistemas y Automática
Universidad Politécnica de Cataluña

TESIS DE DOCTORADO

Marzo, 1995.

Aprobación del Director Externo:



François E. Cellier, Ph.D.
Electric and Computer Engineering Department
University of Arizona
Tucson, AZ, 85721 USA.
E_mail: cellier@ece.arizona.edu

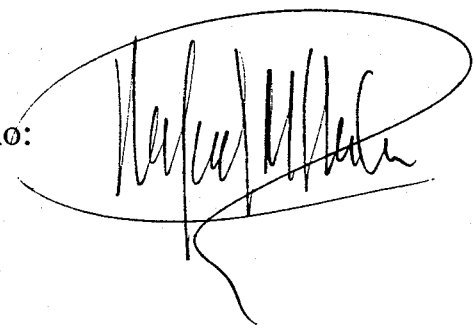
Diseño Sistemático de Controladores Difusos Usando Razonamiento Inductivo

Francisco Mugica Alvarez
Departamento de Ingeniería de Sistemas y Automática
Universidad Politécnica de Cataluña

TESIS DE DOCTORADO

Marzo, 1995.

Aprobación del Director Interno:



Dr. Rafael M. Huber
Instituto de Cibernética
Universidad Politécnica de Cataluña
Av. Diagonal 647, 2da. planta
08028 Barcelona, España
E_mail: huber@ic.upc.es

A los entrañables compañeros de mi vida:

Silvia

Iola Ariana

Francisco Javier

Agradecimientos:

Mi gratitud más grande es desde luego para *François Cellier* quien se ha comportado a lo largo de estos años no solo como un gran director, sino como un compañero científico y un amigo que me ha sabido imbuir de su espíritu investigador, sus ganas de descubrir y que sobretodo, me ha enseñado que la investigación para tener éxito tiene que ser divertida. Gracias François por tu permanente cercanía a más de 15,000 kilómetros. Gracias por tu empuje y por empujarme. En sus propias palabras: "*... your pushing advisor, colleague and friend*".

A *Rafael Huber* quiero darle las gracias por la idea original de desarrollar la tesis en simulación cualitativa, por proporcionarme los medios materiales para desarrollar la investigación y por resolver siempre aquellos problemas insalvables que surgen en todo doctorado.

A *Alvaro de Albornoz*, principal cómplice e instigador de este exilio intelectual. Compañero de investigación y sobretodo de discusiones tan agudas como obstinadas. Aunque en más de una ocasión hubiese querido ahorcarle durante estos largos cinco años de esfuerzos conjuntos ahora me unen muchas más cosas de las que ya teníamos.

A *Angela Nebot*, compañera de doctorado, de artículos, de directores y de desvelos. La catalana más mexicana que conozco, empeñada siempre en ir a comer a un restaurant mexicano, por su invaluable ayuda, comprensión, cariño, enfados y amistad.

A *Sebastián Medina y Josefina López*, los más recientes integrantes del "FAPAS", que toman ahora el relevo en esta aventura a través del mundo cualitativo. Gracias por el empuje y la ayuda desinteresada aunque fuera hasta altas horas de la madrugada. Gracias Fina por tu sonrisa permanente.

A los compañeros del Departamento de ESAII (d'ESAII me corregirían siempre), Toni, Javier y Manel. No os desespereis, algún día hablaré buen catalán.

Por supuesto, tengo que agradecer también al **CO***nsejo* **NA***cional de* **C***iencia* **Y** **T***ecnología - México* y al *Banco de México* por su financiación durante estos años y por brindarme la oportunidad de estudiar este doctorado.

Al *Instituto de Investigaciones Eléctricas* por haberme casi forzado a salir a estudiar al extranjero.

Finalmente, a la *Comisión Interministerial de Ciencia Y Tecnología* por la financiación de la infraestructura necesaria para el desarrollo y publicación de los resultados obtenidos en esta tesis bajo los proyectos CICYT TIC 90–0711 y CICYT TIC 93–0447.

Índice

Lista de Figuras

Capítulo 1

Introducción

Durante los últimos años, prácticamente en todas las ramas de la ciencia se ha experimentado un creciente interés en el uso de los diferentes métodos desarrollados por la Inteligencia Artificial (IA). Aunque el objetivo de la IA — desarrollar máquinas que exhiban un comportamiento inteligente tan parecido al del ser humano como sea posible — está aún lejos de ser alcanzado plenamente, durante las aproximadamente cuatro décadas que lleva de existencia, ha generado un gran número de herramientas que pueden ser de gran valor para otras ramas del conocimiento humano. Paralelamente al crecimiento de la IA, y proporcionándole un gran soporte, la tecnología de la computación digital ha evolucionado en una manera prácticamente exponencial, produciendo computadoras increíblemente veloces, compactas y económicas. Es de esta manera que hoy es posible resolver problemas considerados, hasta hace muy poco, como extremadamente complejos. Este impacto tecnológico ha alcanzado principalmente a todas las ramas de la Ingeniería, siendo de tal magnitud que no sólo ha ampliado el panorama de problemas que pueden resolverse, sino que incluso ha revolucionado la forma en que los problemas son tratados.

Por otro lado, la evolución de los sistemas de ingeniería, como por ejemplo: la aviación, la robótica o las plantas de generación nucleoeléctrica, han alcanzado un grado de complejidad tal, que cada vez es más, no sólo deseable, sino necesario que sean capaces de exhibir un comportamiento *inteligente*. De manera muy relevante, dentro de la Ingeniería de Control existe un importante número de aplicaciones en donde los métodos de la IA podrían implantarse con éxito. Las aplicaciones potenciales pasan por todas las etapas, desde el análisis del comportamiento dinámico, hasta el diseño y desarrollo de los controladores para los procesos. Sistemas basados en conocimiento, razonamiento inductivo,

sistemas difusos y redes neuronales son algunas de las técnicas que han sido usadas en el intento de diseñar controladores inteligentes. De manera recíproca, la IA puede resultar enriquecida y estimulada tanto con la aplicación de sus técnicas en los problemas de ingeniería como con la incorporación de conceptos y herramientas probados durante muchos años. Su aplicación en campos como la identificación de sistemas y el control de procesos, puede ayudar a medir tanto su grado de madurez como sus limitaciones actuales. La evaluación rigurosa de las técnicas de la IA a la luz de aplicaciones en el mundo real, es un elemento fundamental para enfocar correctamente la dirección de los desarrollos futuros. Además, es claro que existen importantes relaciones entre ambas disciplinas como el aprendizaje, la autonomía y la adaptación. Pretender resolver estas cuestiones bajo un único punto de vista es perder la perspectiva y la oportunidad del beneficio del efecto sinergético.

1.1 IA y Control

Se han requerido muchos años para desarrollar los actuales algoritmos para el control de procesos complejos. Basándose en esquemas tales como la teoría de sistemas lineales, las teorías de control óptimo y estocástico y sus extensiones, las teorías de control adaptativo y control robusto, ha sido construida una amplia y sólida base teórica. En todas estas teorías de diseño, tanto el conocimiento del proceso que se desea controlar, como la inteligencia del ingeniero para decidir la estrategia de control correcta, se capturan bajo un esquema definido como "fuera de línea." Existen casos donde el proceso no puede ser descrito mediante modelos lineales, o donde los requerimientos no pueden ser traducidos en un criterio simple para medir la calidad de actuación (como por ejemplo una función de coste cuadrática). En tales situaciones no es posible encontrar una solución analítica basada en los esquemas antes mencionados, y el problema de diseño del controlador debe de ser traducido a un problema de optimización numérica. Aunque en este último caso también son necesarios modelos matemáticos, la experiencia y el conocimiento del experto son requeridos en una fase posterior del ciclo de diseño del controlador cuando la optimización numérica es realizada mediante extensivas ejecuciones de simulación.

Aún así, existen condiciones bajo las cuales, las técnicas de control clásicas no pueden ser aplicadas y puede ser muy ventajoso basar la estrategia de diseño en un enfoque diferente. Algunas de tales condiciones pueden ser:

- no se cuenta con un modelo matemático del proceso a ser controlado, o

éste, sólo puede ser obtenido con gran esfuerzo y costo,

- aunque se tiene un modelo matemático parcial del proceso a ser controlado, la influencia de la dinámica no modelada en la calidad de actuación del controlador es significante y no puede ser despreciada,
- cualesquiera de los parámetros del proceso o del punto de operación cambian de manera imprevisible,
- sólo una parte de la información del proceso se encuentra disponible en forma cuantitativa, mientras que el resto de la información, es asequible únicamente en forma cualitativa,
- los datos que se obtienen del proceso son incompletos y/o imprecisos.

Los puntos mencionados anteriormente no son en manera alguna mutuamente excluyentes. Enfatizan diferentes aspectos de un tema común: control en ausencia del conocimiento completo respecto de la planta a ser controlada, respecto de su entorno, o de ambos.

En situaciones tales como éstas, las aproximaciones matemáticas puramente cuantitativas suelen no trabajar bien y pueden ser reemplazadas con éxito por métodos alternativos de la IA, es decir, por métodos que trabajan utilizando como base una descripción del comportamiento del proceso compuesta por la mezcla de información cualitativa y de la información cuantitativa medida con que se cuenta.

Los métodos alternativos que han sido más comúnmente usados bajo estas situaciones son los siguientes [Verbr 91].

- *Sistemas Basados en Conocimiento (Knowledge–Based Systems – KBS)* o Sistemas Expertos. Representan una aproximación mediante razonamiento simbólico. La idea en la que se basan es la siguiente: a raíz de la aparición de los microprocesadores, los antiguos controladores analógicos han sido reemplazados por controladores digitales (CD) y controladores lógicos programables (CLP). Resulta claro ver que la lógica y el secuenciamiento algorítmico pueden ser convenientemente expresados mediante reglas permitiendo incrementar las capacidades de tales controladores. Este tipo de aproximación es comúnmente usada en los sistemas de consulta y supervisión, en los cuales el conocimiento de especialistas en un campo restringido de experiencia se pone a disposición del usuario [Astrm 86] y [Arsen 89].

- *Redes Neuronales Artificiales (Artificial Neural Networks – ANNs)* son sistemas de aprendizaje basados en conocimiento no estructurado que usualmente operan dentro de un esquema numérico [Ander 89], [Naren 90], y [Kosko 92]. El "perceptrón" multicapa, el mapa autorganizado de Kohonen, las redes de Hopfield y la máquina de Boltzmann son los tipos de redes neuronales más frecuentemente utilizados en sistemas de control. Actualmente la investigación de modelos basados en redes neuronales experimenta un febril desarrollo en todo tipo de aplicaciones en el que por supuesto no faltan los sistemas de control.

- *Control de Articulaciones basado en el Modelo del Cerebelo (Cerebellar Model Articulation Control – CMAC)* Están basados en el método funcional que el cerebelo utiliza para el control de ese tipo de movimiento corporal que, una vez aprendido conscientemente, se lleva a cabo de una manera instintiva o automática. Esta metodología es muy similar a la utilizada en las ANNs, en las que también se aprende el comportamiento del sistema a partir de aplicar valores en las entradas y registrar las correspondientes salidas. Son particularmente útiles para control en tiempo real ya que son extremadamente rápidos y pueden ser desarrollados en circuitos electrónicos digitales ("hardware") de forma más directa que las ANNs [Albus 75], [Krijg 92] y [Kraft 92].

- *Control Difuso (Fuzzy Control – FC)* es una tecnología actualmente muy bien situada que permite utilizar conocimiento de naturaleza heurística para controlar un sistema. Tiene la propiedad de poder manejar imprecisión y vaguedad en la información que utiliza [Lee 90a], [Sugen 85b], [Pedry 93], y [Aoki 90]. Han tenido una gran popularidad ya que, aún sin aprobar del todo la rigurosa métrica que la teoría de control impone, han captado el interés de fabricantes de equipos electrónicos y de control al ser aplicados con éxito a un sinfín de aparatos. Actualmente existen y se continúan diseñando circuitos integrados y microprocesadores especializados para desarrollar este tipo de controladores.

- Sistemas que son producto de cualquier tipo de combinaciones entre los métodos citados anteriormente por ejemplo: FC–ANN's [Beren 92a], KBS-FC-ANN's [Lee 90b] or ANN's-FC [Kosko 92] o bien, combinaciones entre alguno de ellos con controladores convencionales tales como PID, PD, PI o P como en [Kiker 78] o en [Porte 87].

En estas nuevas aproximaciones, conceptos de la teoría de control clásico tales como estabilidad, observabilidad, o controlabilidad no pueden ser

evaluados con facilidad. Es por esta razón que muchos investigadores dentro de la comunidad de la Ingeniería de Control no están convencidos aún del alcance de estos métodos. Sin embargo, miembros más pragmáticos de esta comunidad han sido atraídos por el innegable éxito obtenido al tratar muchas aplicaciones prácticas, y muy particularmente cuando la utilización de los métodos convencionales ha resultado difícil. Añadiendo estas nuevas técnicas al conjunto de opciones disponibles para resolver los problemas de diseño de controladores en aplicaciones prácticas, estos investigadores han creado una nueva y creciente rama de métodos de control llamada *control inteligente*.

Por otro lado, muchos investigadores de la comunidad de IA son aún reticentes a aceptar la idea de que aquellas aproximaciones que no estén basadas en representaciones de datos puramente simbólicas puedan ser llamadas *inteligentes*. Sin embargo, y de manera análoga a la comunidad de control, miembros más prácticos dentro de la comunidad de la IA, se han convencido del impresionante éxito que han tenido, por ejemplo, las redes neuronales especializadas para resolver tareas de la IA como el reconocimiento de patrones, y han considerado estas nuevas técnicas como una importante rama de la IA. Las nuevas generaciones de ingenieros de control y de practicantes de la IA han unido sus fuerzas para resolver problemas que hasta hace muy poco eran considerados como intratables.

En nuestra opinión, la teoría de control clásica desarrollada hasta ahora, se ve imposibilitada de tratar todos los problemas relevantes del mundo real, y nuevas aproximaciones basadas en IA, pueden añadir una nueva faceta a la teoría de control que permitirá ampliar el conjunto de problemas prácticos de control que pueden ser resueltos. Estos métodos alternativos son similares a la forma en la que los seres humanos resuelven las tareas de control, la cual, contrariamente a los algoritmos matemáticos utilizados hasta ahora, no tiene problemas para manejar conocimiento incompleto o impreciso y es realmente capaz de basar las decisiones de control en información cualitativa y frecuentemente incompleta.

Así también, diversas áreas de investigación en IA, como el razonamiento inductivo y la simulación cualitativa, pueden verse reforzadas al incorporar algunos de los más rigurosos y precisos métodos de análisis y diseño que existen, y que han sido probados con éxito durante décadas por los ingenieros de control en la resolución de problemas complejos. Ser capaces de tomar decisiones correctas bajo información incompleta y aún bajo información contradictoria es una gran cualidad, pero no ser capaces de hacer uso de la información cuantitativa con la que se cuenta, justo porque la metodología no permite incorporarla, es notoriamente una gran debilidad. La fusión de métodos

cualitativos y cuantitativos es en sí misma una importante área de investigación que verdaderamente merece ser cultivada.

1.2 Motivaciones y Objetivos de la Investigación

Algunos investigadores en el campo de la IA están llegando a la conclusión de que la inteligencia puede ser entendida mejor si es vista como un sistema integrado de mente y cuerpo [Chand 89]. Por otra parte, campos como la Robótica y el Control Automático intentan dotar de inteligencia a los robots y aumentar las capacidades de desempeño de los controladores que operan sobre mecanismos y sistemas, en tal forma que les permitan convertirse en organismos más autónomos [Minsk 84], [Astrm 86], [Kanad 89]. A diferencia de la Psicología Cognoscitiva, la IA tiene por objetivo no sólo estudiar, sino también construir sistemas que trabajen inteligentemente. Adicionalmente, la IA, la Robótica y la Automática deben tratar problemas del mundo real y, casi siempre, en una escala de complejidad superior.

Así, mientras los psicólogos cognoscitivos abordan la cuestión de la inteligencia de una manera global y, por lo tanto, con una necesaria dosis de vaguedad [Holla 87], los ingenieros de control están forzados a tratar problemas concretos del mundo real, resultando mucho más modesta y específica su visión de la inteligencia artificial. Los investigadores de la IA están quizás situados entre ambos extremos. Los objetivos de esta tesis están ubicados en la región donde se unen la IA y la Ingeniería de Control.

Desde la óptica de la IA, puede verse a los controladores como los agentes inteligentes que son responsables de que los sistemas operen como se desea. Si bien es cierto que, como se ha apuntado antes, existe una vasta cantidad de experiencia y conocimiento respecto de los controladores, su dinámica, su comportamiento y su relación con los procesos y mecanismos que controlan, debe ser desarrollada una nueva generación de controladores con la característica de ser altamente autónomos si pretendemos que los futuros sistemas muestren realmente un comportamiento inteligente. Es en esa nueva generación de controladores, donde la consideración y el uso de las técnicas de la IA resultan imprescindibles.

Aunque las decisiones de los agentes inteligentes artificiales puedan ser vistas como discretas, representadas simbólicamente y esencialmente cualitativas, deben ser tomadas en un mundo que evoluciona en un tiempo y un espacio

continuo. Puede resultar muy útil representar estas decisiones en forma numérica para hacer posible que interaccionen con el entorno de control de manera cuantitativamente coordinada. Una visión conjunta de la IA y del Control Automático puede permitir resolver problemas realmente interesantes en los que se integre la información cualitativa y discreta de los "problem-solving" simbólicos, con la información cuantitativa y continua de la percepción y la acción de los actuadores.

El aprendizaje, el razonamiento con *sentido común* y la dinámica de los procesos son problemas de interés para ambos campos y claves para su desarrollo posterior. En particular la IA ha generado muchas ideas y técnicas útiles que pueden ser usadas para construir sistemas de control más flexibles y robustos. Recíprocamente la IA puede enriquecerse incorporando conceptos como la causalidad temporal y estructural, la dinámica de procesos y el análisis paramétrico procedentes de la Teoría de Sistemas y la Teoría de Control Clásico. **Un primer objetivo** de esta tesis *es colaborar en la construcción de un puente que una los mundos del cálculo cuantitativo y del razonamiento cualitativo.*

Quizás, el problema fundamental que ha impedido la integración de ambos campos en el pasado, es la inherente incompatibilidad que existe entre el esquema simbólico de la IA y la naturaleza numérica de las teorías de sistemas y de control, haciendo que la transferencia de información en la interfaz sea una empresa difícil. Componentes que surgen como naturales en esta interfaz son los *sensores* y los *actuadores* ya que estos han tenido siempre roles similares. Por ejemplo, en los sistemas de control de energía, los sensores traducen las señales de energía en señales de información que pueden ser procesadas por los controladores, mientras que los actuadores traducen las respuestas del controlador (señales de información) en las entradas necesarias a la planta (señales de energía). En los equipos analógicos de control digital, los sensores traducen señales analógicas en señales digitales, las cuales pueden entonces ser procesadas por los controladores digitales, y los actuadores traducen la salida deseada del controlador (señales digitales) en entradas a la planta (señales analógicas). En control inteligente las cosas serán similares. Los sensores traducen la información cuantitativa de la planta que se desea controlar en la información cualitativa que será usada por los razonadores (controladores), preservando aquella valiosa información cuantitativa que sea útil y desechando la información redundante que haría lento e ineficiente el proceso de razonamiento. Los actuadores convierten entonces el resultado cualitativo del proceso de razonamiento (respuesta del controlador) en señales cuantitativas, llevando la planta de su estado actual al punto deseado de operación. Esta tesis muestra un procedimiento sistemático para diseñar y

desarrollar dicha arquitectura de control inteligente, utilizando el esquema básico de los controladores difusos.

En base a resultados principalmente prácticos, la teoría de control ha reconocido, no sin reservas, que los controladores difusos permiten el control de muchas clases de sistemas de una manera robusta e inteligente [Pedry 93], [Ying 90], [Bover 91], [Bover 92], [Aoki 90], [Drian 93], y [Yu 90]. Los Controladores Difusos son básicamente controladores lógicos que utilizan lógica multivaluada. Las funciones de pertenencia son usadas para estimar la calidad de las decisiones lógicas (discretas) y para suavizar las fronteras entre los diferentes valores discretos que durante el proceso de actuación envía el controlador.

Hasta ahora, no obstante, el diseño de controladores difusos, es decir, la determinación de valores límite entre las fronteras de los diferentes valores discretos en la lógica multivaluada y en algunos casos la determinación de la forma de las funciones de pertenencia, ha hecho uso tradicionalmente de técnicas que son predominantemente heurísticas. Aunque siguiendo cierto esquema común, prácticamente cada controlador difuso debe ser diseñado de una manera *ad hoc*. En algunos casos la programación no lineal ha sido usada para minimizar un cierto índice de desempeño en la calidad mostrada por el controlador difuso. En otros casos han sido las redes neuronales las encargadas de optimizar los parámetros de los controladores difusos [Kosko 92]. Los algoritmos genéticos son otra técnica a la que se ha recurrido para alcanzar el mismo objetivo [Goldb 89], [Karr 89] y [Karr 91]. Sin embargo, todas estas técnicas tienen la característica de ser lentas en su velocidad de convergencia y de no garantizar que ésta sea alcanzada.

El objetivo central de la tesis que aquí se propone *es el desarrollo de una nueva aproximación al diseño de controladores difusos*. Esta aproximación se caracteriza por ser sistemática y rápidamente convergente hacia una solución óptima (óptima en términos de un índice especial de comportamiento definido para este propósito). La aproximación tiene como bases de IA el razonamiento inductivo y la lógica difusa y como bases de la teoría de control el estudio del modelado de la dinámica inversa de sistemas no lineales.

Un último objetivo de este trabajo, y no por ello debe considerarse menos importante, *es su aportación en el proceso de estudio, evaluación y desarrollo de la propia herramienta de razonamiento inductivo seleccionada*. La evolución del razonamiento inductivo difuso, tal y como es utilizado en esta tesis, puede verse como un proceso que ha venido desarrollándose en etapas y que ha evolucionado a partir del paradigma de la Teoría General de Sistemas conocido como "General System Problem Solving (GSPS)", [Klir 85], [Uytte

78] y [Uytte 81].

- La primera generación (Cellier, Yandell y Li [Celli 87] y [Li 90]) se dedicó a la tarea de crear un prototipo de implementación computacional — SAPS–II ("System Problem Solver") — que fuese suficiente para llevar a cabo pruebas sencillas y que permitió evaluar adecuadamente la utilidad de la aproximación, es decir, determinar cuánto prometía realmente la metodología. Esta primera fase fue completada en 1990.

- Una segunda generación (Cellier, Nebot, Mugica y de Albornoz [Nebot 94b], [ésta tesis] y [deAlb 95]), inició la tarea de probar que la metodología podría ser usada verdaderamente, tanto en aplicaciones de ingeniería de tamaño industrial como en otro tipo de aplicaciones, ha producido significativos y relevantes resultados, que no pueden obtenerse tan fácilmente con otras tecnologías. Para lograr esto, fue necesario, por un lado, verificar, probar y depurar el código de SAPS–II, así como reformular y añadir algunas funciones para cubrir sus deficiencias. En esta forma, al final de la segunda fase, el prototipo ha sido madurado hasta el nivel de una versión preliminar de explotación. No obstante, debe resaltarse, que el objetivo primordial de esta etapa ha sido contrastar el razonamiento inductivo difuso en aplicaciones realistas de gran escala más que a la metodología en sí misma.

- Durante el desarrollo de la segunda etapa, fueron encontrados nuevos e importantes problemas de la aproximación de razonamiento inductivo difuso. Estos problemas, sin embargo, no pueden ser fácilmente descritos ya que no son fallos o errores simples, sino deficiencias más serias de la metodología. Así que, una tercera generación (López y Medina [López 94] [Medin 94]) agregada a la segunda, intentará dar solución a algunas de estas deficiencias, volcando su atención de nuevo sobre la metodología en sí misma.

- Algunos de los problemas más acuciantes que deberá abordar la tercera generación son los siguientes:

    1. Existe aún mucha heurística. Varios aspectos de la metodología deben ser revisados, en particular, los problemas de segmentación de clases (determinación de los valores de frontera) y el problema de la evaluación de la calidad de las máscaras.
    2. No se cuenta con capacidades de autoverificación. Necesitan ser desarrolladas nuevas herramientas para permitir que SAPS sea

capaz de determinar en que grado sus propias inferencias son razonables.

3. Crecimiento exponencial. Algunos de los algoritmos de SAPS aún tienen patrones de crecimiento exponencial en función del número de variables y del número de clases asociadas a cada variable. Esto limita de manera muy severa las posibilidades del razonamiento inductivo difuso para manejar sistemas de muy grandes dimensiones de una forma eficiente y razonable. La complejidad computacional de los métodos internos de SAPS requiere ser estudiada con mayor detalle para intentar reducirla a un orden polinomial. Este aspecto no sólo se refiere a los algoritmos en sí mismos (complejidad temporal), sino también al uso y necesidades de memoria (complejidad espacial).

1.3 Estructura de la Tesis

La presentación del material de esta tesis seguirá una secuencia de abajo hacia arriba, en forma tal que los elementos de diseño se presentarán de manera progresiva hasta ser completados y reunidos en el capítulo 6 en donde la metodología es completada. Se ha intentado que la tesis sea autocontenida respecto a los elementos y herramientas que la metodología utiliza, y en la medida que fue posible, en cuanto un elemento es presentado a continuación se ilustra mediante un ejemplo.

El capítulo número 2 presenta los elementos fundamentales de la teoría del control difuso. El primer punto considerado es el estado del arte y los principales obstáculos que aún han de ser superados para que esta tecnología pueda ser considerada como una tecnología madura. Aunque de manera únicamente introductoria, se dan los elementos de la teoría de conjuntos difusos que son más relevantes en el campo del modelado de sistemas dinámicos. Posteriormente, se describen, uno por uno, los elementos comunes que deben ser abordados por las diferentes arquitecturas de controladores difusos. A medida que cada elemento es considerado, se resaltan los elementos diferenciales que hacen que nuestra metodología de diseño constituya realmente una alternativa novedosa.

Como se verá más adelante en el capítulo número 6, un elemento indispensable en nuestra metodología de diseño de controladores es el contar con una técnica de modelado cualitativo que satisfaga varios requisitos. La selección de la técnica que resultase más adecuada para la tarea de diseño

de controladores difusos fue una tarea que requirió un esfuerzo considerable que pensamos debería ser reportado. Ello nos llevó a estudiar y probar el estado actual de los diferentes paradigmas con lo que, finalmente, fue posible la elección la más prometedora desde un punto de vista que conjuntase los requerimientos de nuestro objetivo con aquellos que valoran la originalidad de la investigación aún no realizada. Una breve descripción de ellas, seguida de su respectivo análisis de aplicabilidad al problema de control, es dada en el capítulo número 3.

Como resultado del estudio comparativo fue seleccionada una metodología cuya característica más destacable es su capacidad para capturar la causalidad y el conjunto de comportamientos dinámicos de un sistema. Conocida como *Razonamiento Inductivo Difuso*, esta técnica lleva a cabo una formulación de una parte de la *Teoría General de Sistemas* (la inductiva) considerando como marco la *Teoría de Conjuntos Difusos*. A lo largo del capítulo, cada uno de los elementos que constituyen esta metodología es presentado, discutido y finalmente ejemplificado. Ya que ésta es una metodología aún muy poco conocida y en la que muy pocos investigadores trabajan, consideramos pertinente que el nivel de descripción fuera detallado y profundo.

La técnica de modelado cualitativo mediante razonamiento inductivo difuso no sería de ninguna utilidad si no fuera capaz de trabajar en concordancia con las estructuras numéricas de su entorno cuantitativo, lo cual es una cualidad indispensable en el área de control. El desarrollo de una técnica de acoplamiento entre estructuras cualitativas y cuantitativas se presenta en el capítulo número 5, donde, mediante un ejemplo académico pero de buen grado de complejidad, se estudian los detalles de cada uno de los elementos que la componen y se demuestra su factibilidad.

Una vez resueltos los problemas de adquisición, representación y manejo del conocimiento, en el capítulo número 6 se integran las herramientas previamente desarrolladas y añadiendo algunos elementos finales se enuncia la nueva metodología para el diseño de controladores difusos. La idea central de la metodología está basada en la utilización de un modelo en cascada que puede interpretarse como un controlador óptimo en bucle abierto que debidamente identificado mediante un modelo cualitativo puede ser reemplazado para permitir que la configuración del control en bucle abierto sea transformada en una configuración de bucle cerrado. El modelo en cascada está compuesto por un modelo de referencia que hace las veces de operador experto y por un modelo de la dinámica inversa de la planta que estima las entradas que la planta debe obtener para comportarse tal y como el modelo de referencia indica. La formulación resultante es prácticamente independiente del sistema

que se desee considerar. Una vez enunciado en forma completa, el método de diseño es probado e ilustrado mediante el diseño de un sistema de control para una planta del tipo de una entrada y una salida (SISO). El resultado obtenido permite mostrar la validez de la metodología propuesta.

Finalmente, la metodología es probada en un caso de estudio de complejidad del mundo real en el capítulo número 7. El caso de un piloto automático para la conducción de un gran barco de carga fue seleccionado como aplicación de prueba porque cumple con varios requisitos interesantes. Por un lado es una planta cuyo comportamiento dinámico es altamente no lineal y nada trivial de resolver. Por el otro, es un problema relativamente clásico que se ha intentado resolver mediante todo tipo de estrategias (clásicas, adaptativas, difusas) por lo que resulta muy adecuado para los propósitos de comparación permitiendos estimar cuantitativamente la bondad de nuestra metodología.

En el último capítulo se hará, de manera sintética y clara, un resumen de las aportaciones más relevantes de esta tesis, de los aspectos que no fueron posibles de resolver y de los problemas de investigación que se han abierto como resultado de la presente tesis.

Un punto final que deseamos destacar en esta introducción es que, aunque el hilo conductor durante la exposición del material contenido en esta tesis es el diseño de controladores difusos, puede resultar de interés para el lector saber que el desarrollo de la investigación ha sido en alguna forma inverso. Es decir, no fue el diseño de controladores difusos el que motivó originalmente el estudio de las diferentes técnicas y herramientas disponibles, más bien fue la búsqueda de una metodología que permitiera mezclar modelos de estructura simbólica con modelos numéricos convencionales dentro de un esquema de simulación mixta, lo que una vez encontrado, motivó la búsqueda de campos de aplicación lo suficientemente complejos para permitir evaluar el alcance de su potencial. Uno de estos campos de aplicación fue precisamente el desarrollo de una técnica para el diseño sistemático de controladores difusos de la cual ha sido objeto esta tesis. Como siempre suele ocurrir en estos casos, cuando se desea ir más allá de un simple ejemplo académico, la complejidad de la aplicación resulta tal que terminó invirtiendo el orden de interés científico y que es el que finalmente adquirió la presente investigación. Resulta por ello inevitable que, junto con la presentación de la nueva metodología para desarrollar controladores difusos de manera sistemática, se haga continuo énfasis en la estructura y las cualidades de la herramienta metodológica.

Capítulo 2

Control Difuso: Una Revisión

2.1 Introducción

El desarrollo de controladores difusos ha emergido como una de las más prósperas y fructíferas aplicaciones de la teoría de conjuntos difusos. Esta teoría fue postulada por Lotfi Zadeh hace ya más de dos décadas como una generalización de los conjuntos clásicos. Ya en los trabajos seminales, Zadeh [Zadeh 71,73] introducía la idea de formular el problema de control mediante el uso de reglas expresadas con representaciones lingüísticas. Las experiencias diarias de la vida real nos dan muchos ejemplos donde se confirma cómo el entendimiento, el pensamiento y la habilidad humana pueden resolver eficientemente el problema de control para una gran variedad de sistemas sin hacer uso de los sofisticados algoritmos de la teoría de control. Es quizás por eso, que buscando emular las capacidades humanas de control, los primeros trabajos en control difuso fueron desarrollados desde el punto de vista de la inteligencia artificial y no bajo la tutela de la teoría de control. Mamdani y sus colegas son los primeros en aplicar la lógica difusa al control de procesos, particularmente, en un sistema de control para una máquina de vapor [Mamda 74a,74b]. Desde entonces, el número de aplicaciones de los controladores difusos ha ido en aumento, no sólo para el control de procesos, sino prácticamente en todos los campos de la ingeniería. Desde el punto de vista de las aplicaciones pueden diferenciarse dos períodos. En el primero, los esfuerzos se concentraron en experimentos de nivel laboratorio y en el diseño de prototipos, siendo relativamente raros los desarrollos de nivel industrial.

Durante el segundo período, iniciado a finales de los años ochenta, han surgido un gran número de aplicaciones comerciales en lo que se conoce como electrónica de consumo (lavadoras, secadoras, cámaras de video, etc...) así como en aplicaciones industriales de todo tipo aunque con menor auge que en las comerciales. De manera muy notoria ha sido en Japón donde el control difuso ha sido impulsado con mayor fuerza. En este país la lógica difusa se ha convertido en palabra de uso común en respuesta al éxito obtenido al aplicarse y comercializarse en todo tipo de aparatos electrodomésticos e incluso en algunos proyectos industriales. De los casi 200 desarrollos y 700 patentes de aplicaciones basados en lógica difusa reportados hasta 1991 casi el 80% eran japoneses [Garci 91]. La creación de LIFE ("Laboratory for International Fuzzy Engineering Research"), en la ciudad de Yokohama en el año 1989, es otra prueba de los recursos dedicados por este país a consolidar su ventaja en la tecnología difusa. La "moda difusa" ha alcanzado también a EE.UU. y Europa donde se experimenta, también, un fuerte crecimiento. Aunque es cierto que en muchos casos se han exagerado sus capacidades, es indudable que el vaticinado decaimiento de la tecnología difusa, que algunos miembros de la comunidad de control han expresado, no llegará, por lo menos en un futuro próximo.

Los controladores difusos tienen potencialmente un gran número de ventajas cuando las especificaciones del control requieren robustez, adaptabilidad y flexibilidad debido a perturbaciones del entorno o a efectos no modelables de la dinámica del sistema. Los controladores difusos son básicamente controladores *basados en reglas* que discretizan el espacio de operación continuo, no lineal y multidimensional en clases discretas. Utilizando lógica multivaluada, las variables difusas de las reglas preservan la información cuantitativa en los correspondientes valores de clase y el valor de pertenencia a esa clase. El comportamiento del controlador difuso es optimizado mediante una máquina de estados finitos en términos de las clases discretas y finalmente usa la información de la función de pertenencia para hacer una interpolación suave entre los puntos vecinos del espacio de operación continuo.

No es fácil responder a la pregunta de cuándo aplicar, o no, la tecnología de control difuso. Por un lado es una tecnología que actualmente se encuentra en expansión tanto en sus bases teóricas como en sus aplicaciones, no conociéndose muy bien sus límites, por el otro, el número de aplicaciones es aún insuficiente para poder generalizar los resultados alcanzados. Con el objetivo de intentar responder a esta pregunta, en los apartados siguientes expondremos los principales beneficios y desventajas que actualmente exhibe la tecnología de control difuso.

2.2 Beneficios del Control Difuso

Podemos mencionar al menos cinco aspectos que hacen que los controladores difusos sean potencialmente atractivos presentando ventajas sobre los controladores clásicos.

1. **Económicos**. Tres factores deben ser tomados en cuenta cuando se estudia el costo de un controlador:

    a) El número de horas-hombre dedicadas al diseño y al mantenimiento. Durante el desarrollo de un sistema de control, suele haber dos grupos de expertos involucrados. Los que conocen el problema de aplicación y las estrategias de diseño adecuadas para resolverlo, y los expertos en electrónica que conocen los algoritmos numéricos en términos de "bits" y "bytes", así como los detalles de la instrumentación en los microcircuitos correspondientes. Los problemas de comunicación entre ambos grupos tienen un impacto directo en los costos de desarrollo. En [Drian 93] se reporta la construcción en paralelo de dos controles de velocidad, un sistema PID convencional y un sistema de control difuso, para un automóvil cuyo objetivo era mantener 800 rpm., independientemente de las perturbaciones del camino o de cargas adicionales, como por ejemplo el aire acondicionado. Además de las grandes variaciones en los parámetros del sistema, que implican exhaustivos ajustes, debidos primordialmente a la naturaleza de la producción en serie y a otras causas, los problemas de comunicación entre los especialistas mecánicos y de microcontrol, conducen frecuentemente a que los tiempos de desarrollo se prolonguen de manera considerable. En la prueba de estos dos controladores prácticamente no se encontró ninguna diferencia significativa respecto al comportamiento. Sin embargo, mientras que la solución convencional tardó casi dos años-hombre, la solución de control difuso requirió solamente alrededor de seis meses-hombre. Los controladores difusos comerciales ofrecen dos niveles de programación, correspondientes con ambos tipos de expertos: el nivel simbólico, apropiado para describir la concepción de los ingenieros de diseño, y el nivel de compilación que es bien comprendido por los ingenieros electrónicos. Dado que hay una bien definida y formal traducción entre ambos niveles, la tecnología difusa realmente puede ayudar a reducir los problemas de comunicación, lo que repercute en un ahorro de tiempo.

b) El costo de construcción. Un buen número de microprocesadores difusos, tanto analógicos como digitales, así como programas especializados para estos, ha sido desarrollado por empresas comerciales como: Togai Infralogic, Mitsubishi, Micro Devices, Hyper-Logic Corp., entre otras. Estos desarrollos permiten instrumentar controladores difusos a precios muy competitivos y en algunos casos muy inferiores a los de los controladores clásicos tipo PID, PI o PD.

c) Mercadotecnia y patentes. En Japón la palabra "Fuzzy" se ha convertido en un término popular. Todo tipo de aparatos electrodomésticos se comercializan anunciándose como "fuzzy-controlled" bajo asociaciones positivas como siendo modernos, de alta calidad y fáciles de usar. Por ejemplo la lavadora de ropa "un solo botón" constituye el mayor avance respecto a los complejos paneles de control de los anticuados diseños de lavadoras de ropa. En esta máquina se ha incorporado un sensor infrarojo que permite estimar la cantidad y calidad (suciedad) de ropa; una unidad de lógica difusa interpreta los datos del sensor y subsecuentemente un razonador difuso determina, mediante inferencia de su base de reglas, los detalles del programa de lavado. La industria japonesa de aparatos electrodomésticos reportó varios millones de dólares en ventas de productos "fuzzy-controlled". Por otro lado, e independientemente de la mercadotecnia, la situación de patentes hace difícil presentar nuevas soluciones utilizando tecnología convencional sin violar o entrar en conflicto con alguna patente de los competidores produciéndose, en el mejor de los casos, retrasos importantes para entrar en el mercado. Utilizando una solución equivalente difusa se puede (y actualmente lo es) rodear la patente existente.

2. **Robustez**. De manera contraria a los controladores óptimos que son muy sensibles a variaciones de parámetros, un controlador difuso puede tratar con mucha más seguridad una planta cuyos parámetros son variantes en el tiempo. Mientras un piloto aéreo humano es incapaz de calcular una trayectoria de vuelo óptimo utilizando su cerebro para solucionar las ecuaciones de Riccati, es perfectamente capaz de conducirlo de forma segura y exitosa bajo condiciones en las que cualquier piloto automático actual fallaría rotundamente. Cuando una anomalía ha ocurrido, la primera cosa que el piloto humano hará es apagar el control automático. Sin embargo, bajo condiciones normales, el piloto automático puede conducir el avión de una manera más suave y

económica (en términos de combustible) de lo que ningún piloto humano podría hacerlo. En algún sentido, la optimalidad puede ser intercambiada por robustez. Lo mismo es válido para los controladores difusos. Un controlador difuso nunca podrá competir con un controlador óptimo en términos de eficiencia, pero puede ser construido de manera mucho más robusta que cualquier controlador óptimo.

3. **Flexibilidad**. Un controlador difuso puede ser diseñado con muy poco conocimiento del sistema que se supone controlará. Consecuentemente, el mismo controlador, mediante el ajuste de sus parámetros de operación, puede ser usado para controlar diferentes tipos de procesos. Solamente el clásico controlador PID puede competir con los controladores difusos en flexibilidad. Este es un aspecto muy importante para fomentar la formación de técnicos especializados en control difuso, que una vez entrenados en esta nueva tecnología pueden readaptarla fácilmente a un gran número de aplicaciones.

4. **Adquisición del Conocimiento de Expertos**. Una de las mayores cualidades de los controladores difusos es su gran capacidad para capturar de una forma natural el conocimiento que un operador experto tiene de un proceso o sistema. Mediante el uso de variables lingüísticas como 'alto', 'normal', 'muy alto', etc., un experto puede expresar su conocimiento mediante reglas de la forma:

   Si el nivel es alto, **entonces** disminuir flujo de agua.

   Si la presión es baja, **entonces** aumentar flujo de agua.

   Si el consumo es nulo, **entonces** flujo de agua nulo.

   Para diseñar un control difuso, puede no ser necesario conocer, ni ser capaz de expresar mediante ecuaciones diferenciales, un modelo matemático de la dinámica del proceso que se desea controlar. Al no estar basado en un modelo específico, el controlador debe ser más robusto para efectos de la dinámica no modelable de la planta.

5. **Alto grado de automatización**. Normalmente los controladores son diseñados para trabajar dentro de un estrecho margen de operación. En muchos casos de control de procesos industriales, como por ejemplo en la industria química, el grado de automatización normalmente es muy bajo. Aunque suelen instalarse algunos bucles de control convencionales, se requiere la intervención de un operador humano durante las etapas de arranque, cambio de nivel de operación y parada, para ajustar los controladores respecto a cada modo de operación. El conocimiento de

este operador puede ser expresado en proposiciones tales como "si la situación es A o B, debe llevarse a cabo la secuencia C". Bajo condiciones de continuo cambio, el control difuso ofrece un método de representar y de instrumentar el conocimiento de alto nivel del operador experto que resulte compatible con los controladores de bajo nivel. Se conocen aplicaciones, como en la industria del papel, en donde la instalación de un sistema de control difuso en el nivel supervisor de todo el sistema de control de procesos ha conseguido un alto grado de automatización permitiendo obtener sustanciales reducciones tanto de la variación de la calidad del producto, como del consumo de energía y materia prima.

2.3 Desventajas y Limitaciones del Control Difuso

En algunas publicaciones se ha sobregeneralizado e idealizado la utilización del control difuso presentándola como una técnica maravillosa que resuelve todos los problemas de control. En esta sección indicaremos algunas de las desventajas y de los límites actuales de su aplicación.

- **Sistematización**. Actualmente no existen procedimientos que permitan sistematizar el diseño de los controladores difusos [Lee 90a]. Hasta ahora los controladores difusos siempre han sido diseñados heurísticamente y de una manera *ad hoc* basándose en el conocimiento de un operador experto. En muchos casos esto causa que el ajuste de los parámetros del controlador con entradas y salidas múltiples sea una empresa aburrida y desagradable.

- **Adquisición del Conocimiento**. Si bien, como hemos mencionado anteriormente, una de las principales cualidades de los controladores difusos es su facilidad para expresar de manera fácil el conocimiento de un operador experto, cuando el conocimiento a priori y relevante de la operación del proceso no está disponible, es pobre, inadecuado, difícilmente representable bajo el paradigma *basado en reglas*, o no resulta muy consistente, sus benignas cualidades pueden tornarse en serias desventajas y debe de reconsiderarse la idoneidad de la tecnología difusa para estos casos. Por ejemplo, cuando las perturbaciones que resultan de las características de no linealidad y/o variación con el tiempo del proceso son difíciles de predecir, puede producirse una descripción un tanto vaga de cómo el proceso es controlado, descripción

que al incorporarse en el controlador difuso provocará, posiblemente, una pérdida de la estabilidad y una calidad de operación pobre.

- **Estabilidad**. La afirmación de que los controladores difusos siempre son más robustos que los controladores convencionales debe ser matizada. El hecho de que los controladores difusos sean más adaptables que los controladores clásicos no significa que puedan adaptarse arbitrariamente bajo cualquier condición y cambio en la operación del sistema. Mientras que los controladores difusos son usualmente más robustos cuando trabajan con plantas donde se presentan comportamientos no lineales o con el problema de dinámicas no modeladas, se sabe que son más sensibles a cambios en el punto de operación. En general no se cuenta, hasta ahora, con técnicas que permitan diseñar controladores difusos que garanticen ser estables bajo todas las posibles condiciones de operación. Esto está en contraste con los sistemas de control retroalimentados clásicos en los cuales pueden probarse analíticamente las propiedades de estabilidad, al menos cuando son aplicados a sistemas lineales. En el caso de plantas no lineales la situación es diferente. Aunque existen técnicas para analizar las propiedades de estabilidad de ciertos tipos de controladores clásicos cuando se aplican a plantas con cierto tipo de no linealidades, la pérdida de generalidad y de capacidad de análisis matemático hace competitiva la tecnología de controladores difusos cuando la linealidad se pierde (cosa muy frecuente). Además no es cierto, como afirman algunos oponentes al control difuso, que no existan criterios de estabilidad para la metodología difusa. Sin embargo, como ocurre con los controladores no lineales clásicos, no hay métodos generales para probar la estabilidad en todos los casos, pero se cuenta con criterios para un buen número de tipos de controladores difusos.

- **Número de Especialistas**. En muchos proyectos, por ejemplo en industrias europeas, el desarrollo de controladores difusos puede requerir un mayor tiempo de diseño, debido a que los ingenieros de control involucrados en el desarrollo del proyecto no están aún suficientemente familiarizados con la tecnología difusa. La escasez de personal cualificado en control difuso es uno de los actuales *cuellos de botella* para la explotación de esta nueva tecnología. Exceptuando el caso de Japón, la tecnología difusa apenas comienza a incluirse en el contenido de los programas académicos de las universidades y deberán pasar aún varios años para contar con un número suficiente de ingenieros formados adecuadamente que puedan explotarla. Como la industria no puede esperar, debería incrementarse el número de seminarios y cursos para permitir que los actuales técnicos sean capaces de utilizar con más soltura

esta nueva tecnología.

2.4 Tendencias de Investigación y Desarrollo

Considerando los desarrollos teóricos, las aplicaciones existentes, sus beneficios y sus limitaciones actuales, puede decirse que, si bien está fuera de toda duda su utilidad, relevancia y aplicabilidad, la evolución de la tecnología de control difuso no ha hecho más que empezar. Se requiere una segunda generación de controladores difusos que permitan tratar sistemas más complejos, que sean capaces de explotar más a fondo la teoría de conjuntos difusos desarrollada, que tengan una mejor y más genérica estructura y principalmente, que puedan ser desarrollados bajo metodologías más sistemáticas. Los retos actuales más relevantes que deben afrontarse son:

- La investigación de los laboratorios industriales debe transferir a la mayor brevedad posible los avances teóricos a aplicaciones comerciales mediante bancos de pruebas y estudios comparativos con las tecnologías clásicas. No tiene sentido tecnológico encontrar métodos de defusificación más precisos, o nuevas interpretaciones, por ejemplo de la conjunción difusa, si no hay quien las aplique.

- Las universidades deben apresurar la actualización de los planes de estudio preparando a los nuevos ingenieros de control y en electrónica integrando la metodología difusa. Los nuevos desarrollos comerciales no podrán ser eficientemente transferidos a la producción industrial si no hay personal calificado para hacerlo.

- Aunque desde el punto de vista de impacto tecnológico los dos puntos mencionados arriba son los más urgentes, su solución se verá muy pronto obstruida si la investigación académica no es capaz de resolver los principales *cuellos de botella* que impiden la plena explotación de la tecnología de control difuso. En primer lugar es **fundamental** contar con metodologías más sistemáticas de diseño y de análisis que permitan:

  a) superar la gran cantidad de heurística que se aplica en la selección de los valores de los parámetros;

  b) sistematizar la configuración y construcción de la base de reglas;

  c) estudiar las propiedades matemáticas, tales como la estabilidad y la controlabilidad; y

d) estudiar y establecer esquemas de interrelación con los controladores convencionales que permitan amalgamar las mejores propiedades de ambos dentro de un esquema de solución híbrido.

Por supuesto que no se espera contar con un método universal de diseño, que ni en el caso de control clásico existe, pero los métodos y algoritmos alternativos deben estar bien clasificados y categorizados respecto a sus correspondientes aplicaciones potenciales. La ausencia de una coherente y sistemática metodología podría causar, como efecto secundario, la pérdida de los pocos especialistas de control difuso con experiencia y entrenamiento, que habiendo contado con un soporte muy limitado y al no verse reforzados y retroalimentados con la sistematización y evolución de su propia experiencia, se verán incapacitados de continuar y desalentados para difundir los conocimientos adquiridos mediante el trabajo experimental.

2.5 Introducción a los Conjuntos Difusos

La característica más relevante de la lógica difusa, particularmente en lo que se refiere al tratamiento de sistemas de ingeniería, es su capacidad para capturar la incertidumbre e imprecisión de la experiencia humana respecto a la descripción u operación de un sistema. Para efectos del diseño de controladores, es el tratamiento de imprecisión el que más interesa. *La temperatura es alta y la presión normal* es una frase llena de imprecisión pero sin incertidumbre. Recientemente Zadeh escribió: "*Contrary to the expectations, most of the successful applications of fuzzy logic at the juncture relate to control and systems analysis in which there is imprecision but no uncertainty*" [Zadeh 92]. La lógica difusa es una generalización de los conjuntos clásicos. A diferencia de la lógica clásica, en donde un elemento pertenece, o no pertenece a un conjunto — denotándose esto con el conjunto binario de valores de verdad {0,1} o {cierto,falso} —, la lógica difusa permite a sus predicados, adquirir valores de verdad multivaluados, es decir, cualquier valor dentro del intervalo unitario continuo [0,1]. Las bases matemáticas actuales de la teoría de conjuntos difusos y de la lógica difusa son muy amplias y se encuentran aún en continua expansión. No es nuestro propósito profundizar en ellas pero siendo parte de la base de los controladores difusos es importante darle cierta atención. De hecho, hasta ahora, el desarrollo de controladores difusos ha explotado muy pocos fragmentos de estas teorías. Para efectos del material contenido en esta tesis nos limitaremos solamente a presentar un resumen de los operadores y funciones necesarios para entender algunos aspectos del estado

del arte. Si se desea profundizar en la base conceptual de la lógica difusa recomendamos [Kande 86] y [Zimme 85] y [Yager 94] y [Pedry 93] si se desea un enfoque de control.

2.5.1 Conceptos Básicos de Conjuntos Difusos

Para cualquier conjunto clásico C definido en el universo de discurso[1] U es posible definir una *función característica* $Memb_C : U \rightarrow \{0, 1\}$ como:

$$Memb_C(u) = \begin{cases} 1 & \text{when } u \in C \\ 0 & \text{when } u \notin C \end{cases} \tag{2.1}$$

En la teoría de conjuntos difusos, esta función característica es generalizada como una *función de pertenencia* que asigna a cada elemento $u \in U$ un valor en el *intervalo unitario* $[0, 1]$. El conjunto F obtenido en base a tal función de pertenencia es definido como *Conjunto Difuso*.

Definición 2.1: Función de Pertenencia[2]. La función de pertenencia $Memb_F$ del *conjunto difuso* F es:

$$Memb_F : U \rightarrow [0, 1] \tag{2.2}$$

donde, para cada elemento $u \in U$, se asigna el grado de pertenencia $Memb_F(u) \in [0, 1]$. De esta manera, F queda completamente determinado por el conjunto de pares:

$$F = \{(u, Memb_F(u)) \mid u \in U\} \tag{2.3}$$

Mediante la notación de Zadeh, usando + como símbolo de enumeración, el símbolo / para denotar un par y considerando un universo U que sea numerable, podemos definir un conjunto difuso de la siguiente manera:

Definición 2.2a: Conjunto Difuso. Un conjunto difuso en un *universo de discurso* **numerable** U, se define como:

[1] Del inglés "universe of discourse".
[2] Del término en inglés "membership", pertenencia.

$$F = Memb_F(u_1)/u_1 + \cdots + Memb_F(u_n)/u_n = \sum_{i=1}^{n} Memb_F(u_i)/u_i \qquad (2.4)$$

y en donde se asume[3] que, para un mismo elemento r con s valores de pertenencia diferentes, + satisface que:

$$x_1/u_r + \cdots + x_s/u_r = \bigcup_U \{Memb_{F_i}(u_r)/u_r\} = \max(x_1, \cdots, x_s)/u_r \qquad (2.5)$$

Definición 2.2b: Conjunto Difuso. Un conjunto difuso en un *universo de discurso* **no numerable** U, es definido como:

$$F = \int_U Memb_F(u)/u \qquad (2.6)$$

donde el símbolo $\int$ representa una numeración incontable.

Mediante estas definiciones podemos, por ejemplo, representar el conjunto de los números *naturales* cercanos a seis como:

$$F = \tilde{6} = \{0.1/3, 0.2/4, 0.5/5, 1.0/6, 0.5/7, 0.2/8, 0.1/9\}$$

o de manera general para el caso continuo $U = \mathbf{R}$:

$$F = \tilde{6} = \int_U Memb_F(u)/u$$

donde:

$$Memb_{\tilde{6}}(u) = \frac{1}{1 + (u-6)^2}$$

[3]Esta suposición es conocida como el *principio de extensión* que permite: a) combinar conjuntos difusos y no difusos; b) interpretar los conjuntos difusos como *números difusos* para permitir realizar operaciones aritméticas entre ellos como sumar, multiplicar, etc.; y c) representar *conceptos lingüísticos* como subconjuntos difusos del conjunto de números reales.

Podemos ver en este ejemplo que 'cercano a seis' nos define una *noción lingüística* o un *concepto lingüístico* (un *cualificador lingüístico* en la terminología de Zadeh). En modelado de sistemas difusos es frecuente utilizar conceptos lingüísticos representados mediante ciertos conjuntos difusos especiales llamados *números difusos* (ecuación (2.5)), que tienen determinadas propiedades matemáticas[4] especiales. Ya que en control difuso solamente se utiliza este tipo de conjuntos, en el futuro se asumirá que cuando nos referimos a un conjunto difuso nos estamos refiriendo a él.

Los conceptos lingüísticos suelen ser *dependientes del contexto* dentro del dominio físico particular donde se definen. Por ejemplo el concepto lingüístico 'caliente' depende si se refiere a la temperatura de un horno o a la temperatura del medio ambiente. Los conceptos lingüísticos usualmente utilizan una de estas tres formas básicas para definirse, dependiendo de la *monotonía* que guarda su función de pertenencia: creciente, decreciente o unimodal. Por ejemplo, dados los dominios: edad ({1,...,110}), velocidad de un automóvil (km/h), temperatura (^{o}C) y error (%), posibles conceptos lingüísticos crecientes serían: 'viejo', 'rápida', 'calurosa' e 'inaceptable', respectivamente. Para el caso decreciente con esos mismos dominios, posibles conceptos lingüísticos serían: 'joven', 'lenta', 'fría' e 'irrelevante'. Por último, y en forma respectiva, conceptos lingüísticos de forma unimodal podrían ser: 'madura', 'normal', 'agradable' y 'moderado'. Como un ejemplo ilustrativo, en la figura 2.1 pueden verse las funciones de pertenencia para 'calurosa', 'fría' y 'agradable' para los casos de funciones con monotonía a) creciente, b) decreciente y c) unimodal.

Operando el valor de las entradas para producir salidas, un *sistema difuso* puede ser definido mediante una representación abstracta de sus entradas y salidas. Esta representación abstracta es realizada mediante expresiones llamadas *variables lingüísticas o variables difusas*, por ejemplo: 'edad', 'velocidad' o 'temperatura', y cuyos posibles valores son el conjunto de los conceptos lingüísticos definidos sobre el universo de discurso de la variable, por ejemplo: 'joven', 'rápida' o 'calurosa'.

[4]Un número difuso F es un conjunto difuso definido en $\mathbf{R}$, tal que:

(a) F es un conjunto difuso normal (hay al menos un elemento para el cuál $Memb_F = 1$.

(b) F es convexo, $\forall_{\lambda\in[0,1]}\forall_{x\in\mathbf{R}} F[\lambda x + (1-\lambda)y] \geq \min(F(x), F(y))$

(c) Todos los α-cuts de F son intervalos cerrados de $\mathbf{R}$.

(d) F tiene un soporte acotado.

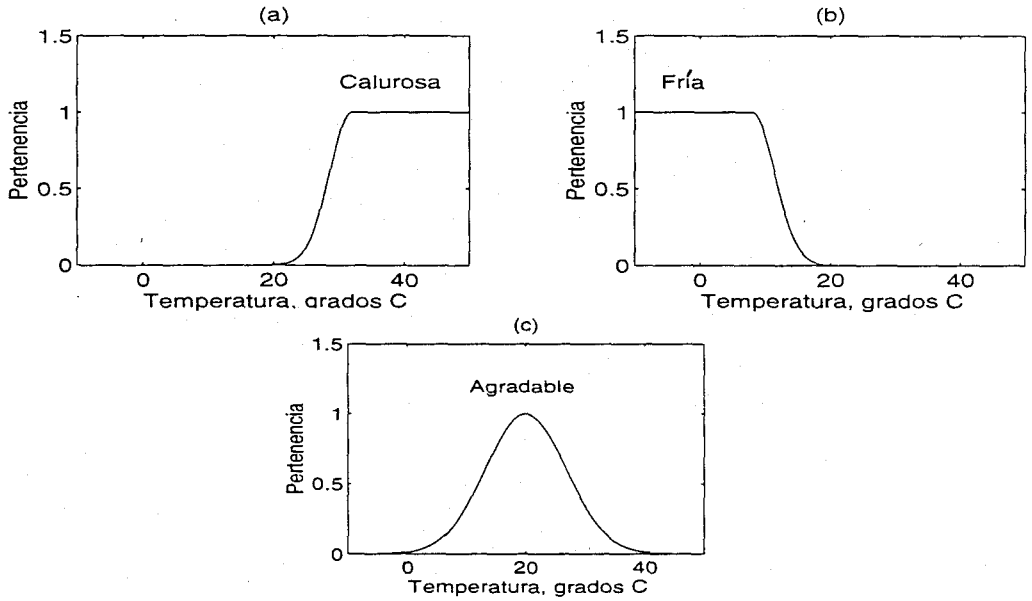


Figura 2.1: Funciones de pertenencia con monotonía a) creciente; b) decreciente; y c) unimodal.

Definición 2.3: Variable lingüística. La variable lingüística u_i es una descripción simbólica constante utilizada para representar, en general, una cantidad variando en el tiempo dentro del intervalo definido por un universo de discurso U_i. Los diferentes valores que esta variable puede adquirir son conocidos como *valores lingüísticos.*

Definición 2.4: Valor lingüístico. Sea la variable lingüística u_i definida sobre el universo de discurso U_i, tal que $u_i \in U_i$ y $U_i = \{F_{i,1}, F_{i,2}, \cdots, F_{i,N_i}\}$. Definimos el valor lingüístico $F_{i,j}$ como el subconjunto difuso que denota el $j^{\text{ésimo}}$ valor que la variable u_i puede adquirir de entre los N_i posibles valores.

Los valores lingüísticos corresponden a los conceptos o nociones lingüísticos mencionados anteriormente. En la terminología de los Sistemas Expertos se les suele referenciar como *clases o categorías*, mientras que en la literatura de "Pattern Recognition" se les conocen como *"clusters"*.

Para ejemplificar estas definiciones consideremos la variable lingüística t_i = *Temperatura interior* en el dominio $T_{Int} = [-10, 50]$, tal que $t_i \in T_{Int}$. Supongamos que nos encontramos desarrollando el modelo de un controlador difuso para un aparato de aire acondicionado y requerimos representar los diferentes valores lingüísticos que puede adquirir la variable lingüística *Temperatura interior*. Utilicemos para ello los valores lingüísticos *calurosa*,

agradable y *fría* representados en la figura 2.1; en nuestra notación: $T_{Int,calurosa}$, $T_{Int,agradable}$ y $T_{Int,fría}$. La descripción del dominio T_{Int} con sus respectivos valores lingüísticos se representa en la figura 2.2.

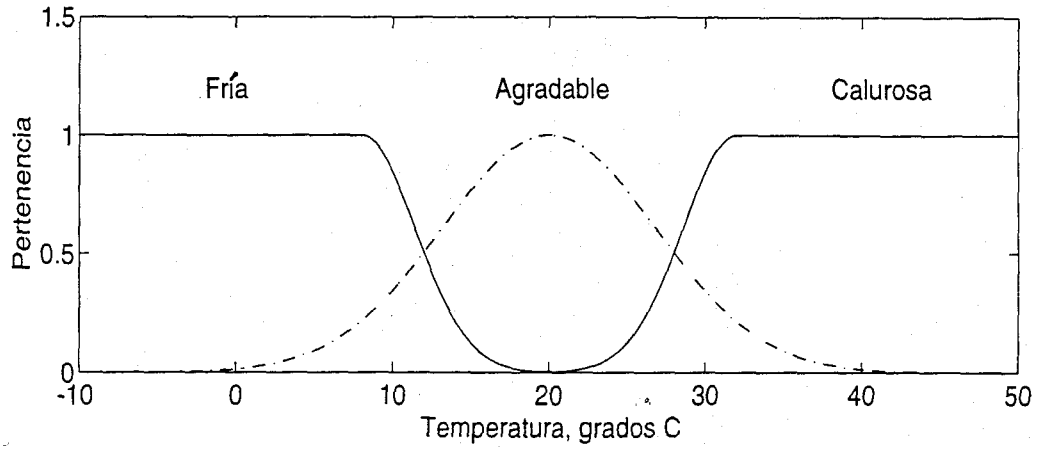


Figura 2.2: Universo de discurso T_{Int} para la variable t_i y sus tres valores lingüísticos.

Supongamos ahora que, por especificaciones de diseño, se necesita que el controlador tenga mayor resolución. Para esto, será necesario añadir más valores lingüísticos dentro de nuestro dominio. Definamos, por ejemplo, los valores $T_{Int,cálida}$ y $T_{Int,fresca}$. La figura 2.3 muestra la inclusión de estos dos nuevos valores lingüísticos al dominio. Como puede verse, el dominio utilizado en este caso es el de los números reales acotado en el intervalo $[-10, 50]$. En control difuso es frecuente definir el dominio de operación en la forma conocida como *dominio normalizado* $[-1, 1]$. Una de las ventajas de la normalización es la posibilidad de disminuir el efecto de la dependencia del contexto. Una desventaja es la pérdida de la relación directa entre la expresividad lingüística y el universo de discurso.

Con éstas definiciones podemos construir *expresiones lingüísticas* del tipo: *La temperatura está agradable.* Si la temperatura del termómetro marca $20°C$ esto lo podemos representar como $t = T_{Int,agradable}$ con $Memb_{T_{Int,agradable}} = 1.0$. Las expresiones lingüísticas son el equivalente de los predicados en lógica.

2.5.2 Operaciones con Conjuntos Difusos Simples

En el modelado de sistemas difusos, es necesario contar con versiones generalizadas de las operaciones básicas de conjuntos, tales como unión,

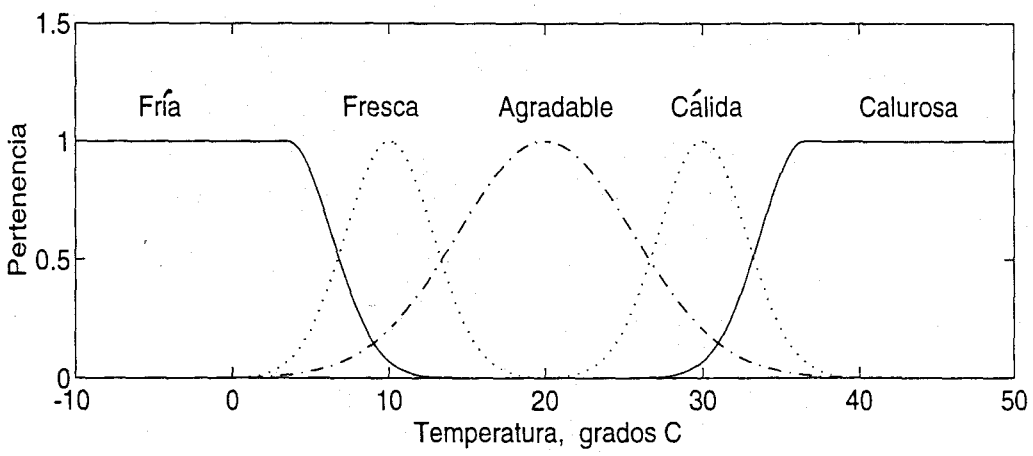


Figura 2.3: Ampliación del número de valores lingüísticos dentro de T_{Int}.

intersección, etc. Comenzaremos con las operaciones básicas suponiendo que los conjuntos (o subconjuntos) son *conjuntos difusos simples*, es decir, se encuentran definidos bajo el mismo universo de discurso. Un punto que debe aclararse respecto a la mayoría de las operaciones con conjuntos difusos es que no existe una única definición, sino que suele haber varias *interpretaciones* dependiendo de la cualidad difusa que se desea preservar. Por ejemplo consideremos las siguientes dos definiciones de la operación unión.

Definición 2.5a: Unión[5]. Sean $F_{i,a}$ y $F_{i,b}$ subconjuntos difusos en el universo de discurso U_i. Definimos la función de pertenencia de su unión, denotada $F_{i,a} \cup F_{i,b}$, como:

$$Memb_{F_{i,a} \cup F_{i,b}} = \min\{1, Memb_{F_{i,a}} + Memb_{F_{i,b}}\} \tag{2.7}$$

Definición 2.5b: Unión (clásica). Sean $F_{i,a}$ y $F_{i,b}$ subconjuntos difusos en el universo de discurso U_i. Definimos la función de pertenencia de su unión, denotada $F_{i,a} \cup F_{i,b}$, como:

$$Memb_{F_{i,a} \cup F_{i,b}} = \max\{Memb_{F_{i,a}}, Memb_{F_{i,b}}\} \tag{2.8}$$

Desde el punto de vista lingüístico, tanto la definición 2.5a como la definición 2.5b pueden ser interpretadas como la conectiva lógica "OR", de tal forma que podemos representar expresiones lingüísticas del tipo: *la*

[5]Esta función de unión también es conocida bajo el nombre de *suma acotada* y representada por el símbolo $\oplus$.

temperatura está fresca **ó** *la temperatura está agradable*: la figura 2.4 muestra la representación bajo ambas definiciones.

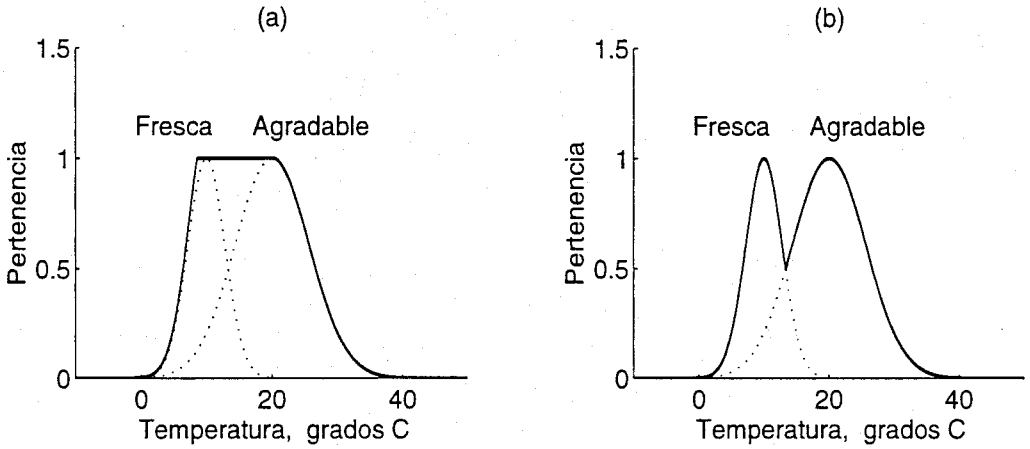


Figura 2.4: Expresión: *la temperatura está fresca* **ó** *la temperatura está agradable* vista desde dos interpretaciones diferentes de la conectiva OR.

Si se observa la región de superposición en ambas gráficas, puede apreciarse que mientras la definición 2.5b pierde cierta información, produciendo un posible resultado contra intuitivo desde el punto de vista lingüístico[6], la definición 2.5a no la pierde. Otra forma, en que pueden verse ambas definiciones de la función de unión, es bajo el nivel de intercambio que se puede hacer entre uno y otro argumento. La unión bajo la definición 2.5a es muy *suave* y permite un perfecto intercambio entre los argumentos. Por el contrario la definición 2.5b es una "OR" *fuerte* seleccionando el mayor de los grados de pertenencia de los conjuntos. Una presentación más amplia de las diferentes familias de operaciones de unión e intersección viene dada en [Klir 88].

Hemos querido destacar esta característica de la teoría difusa para transmitir al lector lo vasto que resulta el campo cuando se comienzan a combinar las diferentes interpretaciones. Aunque la naciente teoría de control difuso no ha explotado sino una muy pequeña porción de los desarrollos matemáticos, los que hay son suficientes para que resulte muy difícil dar revisión a todos ellos con detalle. Si se desea profundizar en las justificaciones para usar una u otra definición desde la óptica de control recomendamos [Yager 94] y [Lee 90a].

[6]Nótese que el conjunto difuso resultante pierde además la calidad de convexidad, y con ello el de número difuso.

Definición 2.6: Intersección. Sean $F_{i,a}$ y $F_{i,b}$ subconjuntos difusos en el universo de discurso U_i. Definimos la función de pertenencia de su intersección, denotada $F_{i,a} \cap F_{i,b}$, como:

$$Memb_{F_{i,a} \cap F_{i,b}} = \min\{Memb_{F_{i,a}}, Memb_{F_{i,b}}\} \tag{2.9}$$

La figura 2.5 muestra el ejemplo de la operación de intersección para el concepto *la temperatura está fresca* **y** *la temperatura está agradable*. La función de intersección entre valores lingüísticos de un mismo dominio es muy poco usada para el modelado de sistemas difusos lingüísticos porque resulta generalmente contra intuitiva.

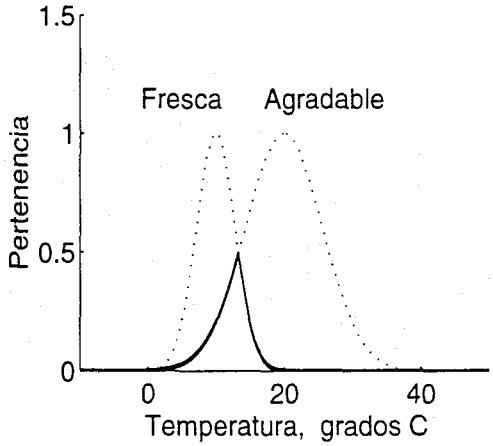


Figura 2.5: Expresión: *la temperatura está fresca* **y** *la temperatura está agradable* como ejemplo de la conectiva AND.

Más natural resulta la operación de complemento dada a continuación.

Definición 2.7: Complemento. Sea F_i un conjunto difuso en el universo de discurso U_i. Definimos la función de pertenencia de su complemento, denotada $\overline{F}_i$, como:

$$Memb_{\overline{F}_i} = 1 - Memb_{F_i} \tag{2.10}$$

La figura 2.6 muestra la representación para el concepto lingüístico: *la temperatura no está agradable.*

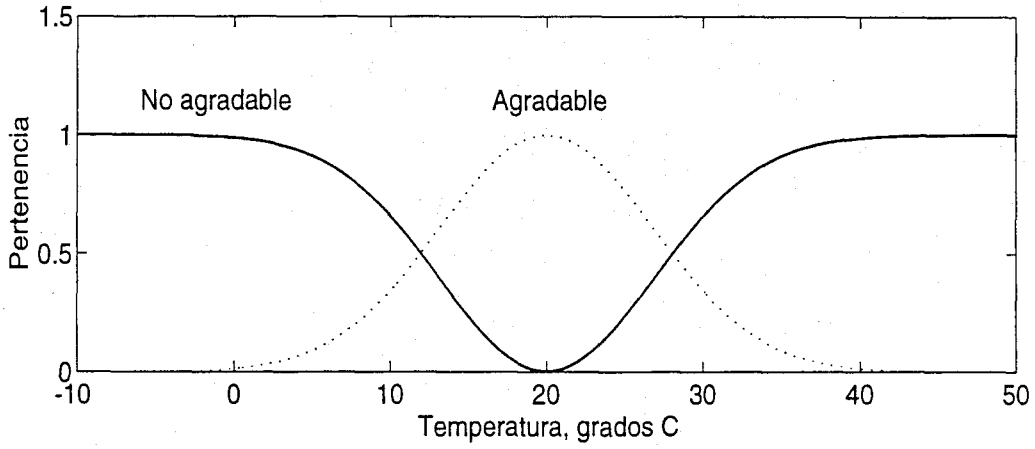


Figura 2.6: Expresión: *la temperatura* **no** *está agradable* como ejemplo de la operación de complemento.

2.5.3 Relaciones entre Conjuntos Difusos

En este apartado consideraremos el caso de operaciones con conjuntos difusos que pueden ser subconjuntos en diferentes dominios (universos de discurso). Este tipo de operaciones son indispensables cuando modelamos sistemas difusos (como por ejemplo controladores difusos) porque permite establecer las *relaciones* entre las diferentes variables lingüísticas. Algunas de las relaciones difusas más importantes desde el punto de vista de su aplicación al control difuso son dadas a continuación.

Definición 2.8: Producto Cartesiano. Sean $F_1, F_2, \cdots, F_n$ conjuntos difusos definidos en sus respectivos dominios $U_1, U_2, \cdots, U_n$, el producto cartesiano de $F_1, F_2, \cdots, F_n$ es un conjunto difuso en el espacio $U_1 \times U_2 \times \cdots \times U_n$, con la función de pertenencia[7] dada como:

$$Memb_{U_1 \times \cdots \times U_n}(u_1, \cdots, u_n) = \min\{Memb_{U_1}(u_1), \cdots, Memb_{U_n}(u_n)\} \quad (2.11)$$

donde $u_1, u_2, \cdots, u_n$ sean elementos en $U_1, U_2, \cdots, U_n$ respectivamente. El producto cartesiano difuso puede verse como la relación difusa más general y es por eso que se utiliza a su vez para definir otras relaciones. En esta forma, una relación binaria cualquiera puede definirse como sigue.

[7]Puede usarse cualquier norma triangular, por ejemplo el producto algebraico: $Memb_{U_1 \times U_2 \times \cdots \times U_n}(u_1, u_2, \cdots, u_n) = \{Memb_{U_1}(u_1) \cdot Memb_{U_2}(u_2) \cdot \ldots \cdot Memb_{U_n}(u_n)\}$.

Definición 2.9: Relación. Sean U_1 y U_2 universos de discurso continuos y $Memb_R : U_1 \times U_2 \rightarrow [0, 1]$. Entonces:

$$R_{U_1,U_2} = \int_{U_1 \times U_2} Memb_R(u_1, u_2)/(u_1, u_2) \tag{2.12}$$

es una relación binaria en $U_1 \times U_2$. Como antes, el símbolo $\int$ denota el conjunto de pares $Memb_R(u_1, u_2)/(u_1, u_2)$ en el universo continuo $U_1 \times U_2$.

Por ejemplo, podemos establecer relaciones binarias de tipo: R_T = *La temperatura externa (t_e) es muy cercana a la temperatura interna (t_i).* Siendo t_e y t_i variables definidas en los universos T_e y T_i respectivamente, podemos describir la relación R_T como:

$$R_T = \int_{T_e \times T_i} \exp^{-\alpha|t_e - t_i|} /(t_e, t_i) \tag{2.13}$$

donde α es un factor de escala.

Otro caso ilustrativo de relación binaria puede ser la *implicación difusa.* Por ejemplo, la expresión: **Si** *la temperatura externa t_e es fresca* **entonces** *la temperatura interna t_i es agradable*, genera la implicación difusa:

$$\textbf{Si} \quad T_{e,fresca} \quad \textbf{entonces} \quad T_{i,agradable},$$

que, representada como una relación difusa, queda como:

$$R = T_{e,fresca} \times T_{i,agradable}$$

con la $Memb_R$:

$$Memb_R(t_e, t_i) = \min\{Memb_{T_{e,fresca}}(t_e), Memb_{T_{i,agradable}}(t_i)\}$$

mostrada en la figura 2.7

Una vez definido el concepto de relación difusa podemos expresar ahora un nivel de complicación más, al considerar operaciones entre relaciones difusas, operaciones que se les conoce bajo el nombre de *composiciones.* La gran importancia de la relación de composición en el modelado de controladores difusos es que con ella es posible definir la operación de **implicación** utilizada

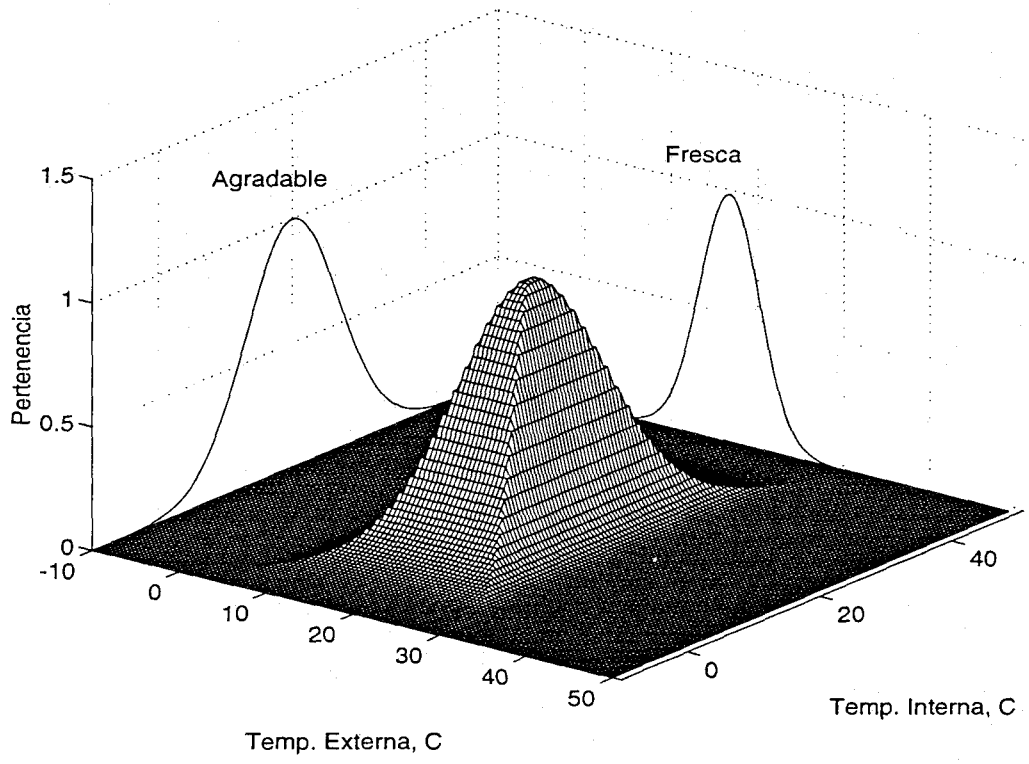


Figura 2.7: Funciones de pertenencia para: *la temperatura externa* t_e *es fresca*; *la temperatura interna* t_i *es agradable*; y $R = T_{e,fresca} \times T_{i,agradable}$.

para representar las **reglas de control**. Se conocen más de cuarenta tipos de implicaciones difusas [Lee 90a] divididas en tres grupos según la función de composición que se utilice: la *conjunción difusa*, que utiliza la composición de unión; *la disjunción difusa*, que utiliza la composición de intersección; y a *implicación difusa*, que utiliza la composición sup–estrella (de "sup–star composition").

Definición 2.10: Composición de Intersección. Si R y S son relaciones difusas en $U_1 \times U_2$ y $U_2 \times U_3$, respectivamente, la intersección de R y S es una relación difusa en $U_1 \times U_3$ representada por $R \cap S$ y definida como:

$$R \cap S = \int_{(U_1,U_3)} [Memb_R(u_1, u_2) * Memb_S(u_2, u_3)]/(u_1, u_3) \qquad (2.14)$$

en donde $*$ puede ser cualquier operador de la norma triangular[8].

[8]mínimo, producto algebraico, producto acotado o producto drástico.

Definición 2.11: Composición de Unión. Si R y S son relaciones difusas en $U_1 \times U_2$ y $U_2 \times U_3$, respectivamente, la unión de R y S es una relación difusa en $U_1 \times U_3$ representada por $R \cup S$ y definida como:

$$R \cup S = \int_{U_1,U_3} [Memb_R(u_1, u_2) + Memb_S(u_2, u_3)]/(u_1, u_3) \tag{2.15}$$

en donde + puede ser cualquier operador de la conorma triangular[9].

Definición 2.12: Composición Sup–estrella. Si R y S son relaciones difusas en $U_1 \times U_2$ y $U_2 \times U_3$, respectivamente, la composición de R y S es una relación difusa en $U_1 \times U_3$ representada por $R \circ S$ y definida como:

$$R \circ S = \int_{U_1,U_3} \mathbf{sup_{u_2}} [Memb_R(u_1, u_2) * Memb_S(u_2, u_3)]/(u_1, u_3) \tag{2.16}$$

Un buen número de posibles composiciones se producen dependiendo de la interpretación de $*$ y de **sup**. Cuando $*$ toma el valor de **min** la definición anterior se convierte en la bien conocida *regla de composición* de Zadeh [Zadeh 73], conocida también como la *regla de composición sup–min.*

Por ejemplo, supongamos la relación R definida como: *humedad* (h_1) alta y temperatura externa (t_e) cálida y la relación S definida como: *temperatura externa* (t_e) cálida y temperatura interna (t_i) calurosa. Si aplicamos la definición 2.12 tomando el **sup** como máximo y $*$ como mínimo, la relación resultante sería:

$$\begin{aligned} Memb_{R \circ S} &= \max_{t_{e,cálida}} [\min(Memb_R(h_1, t_e), Memb_S(t_e, t_i))] \\ &= \max_{t_{e,cálida}} [\min(Memb_{H_{1,alta}}(h_1), Memb_{T_{e,cálida}}(t_e), Memb_{T_{i,calurosa}}(t_i))] \end{aligned}$$

2.6 Razonamiento Aproximado

Lotfi Zadeh presenta por primera vez su teoría del razonamiento aproximado en [Zadeh 73], desarrollándola plenamente en [Zadeh 75] y [Zadeh 79], y proporcionando con ello una poderosa herramienta para razonar bajo

[9] máximo, suma algebraica, suma acotada, suma drástica y suma disjunta.

información imprecisa o incierta. Aunque los fundamentos continúan prácticamente intactos, muchas aclaraciones y valiosas contribuciones se han hecho en los últimos años [Duboi 91,93]. El modelo de Razonamiento Aproximado desarrollado por Zadeh es conocido como el modelo *posibilístico* y cabe aclarar aquí que no es el único, existiendo otros modelos como el de la teoría de la evidencia, desarrollado por Dempster–Shafer [Demps 68], [Shafer 76] y [Bucha 84], o el modelo probabilístico de la teoría de la probabilidad, basado en la inferencia bayesiana, y de la cuál hay un buen número de versiones [Duda 76], [Fried 81], [Reite 81] y [Quinl 83]. En el material que se expone a continuación, sólo nos referiremos al modelo posibilístico.

En razonamiento clásico existen dos tipos de reglas de inferencia, el *Modus Ponens* y el *Modus Tollens*, los cuales pueden ser generalizados para el caso de razonamiento aproximado como:

- *Modus ponens* generalizado.
  Antecedente 1: **Si** u_1 es A **entonces** u_2 es B
  Antecedente 2: u_1 es A'

  Conclusión: u_2 es B'
- *Modus tollens* generalizado.
  Antecedente 1: **Si** u_1 es A **entonces** u_2 es B
  Antecedente 2: u_2 no es B'

  Conclusión: u_1 no es A'

El *modus ponens* generalizado puede ser reducido al *modus ponens* cuando $A' = A$ y $B' = B$. Este tipo de inferencia suele ser conocida como *razonamiento hacia adelante* o como *razonamiento guiado por los datos*, permitiendo ir desde las premisas hacia los resultados. Este razonamiento es particularmente útil cuando se utiliza en control difuso ya que nos permite tomar una decisión de control en base al estado del sistema actual. De manera análoga, el *modus tollens* generalizado puede ser reducido al *modus tollens* cuando $A' = no\ A$ y $B' = no\ B$. A este tipo de inferencia se le conoce como *razonamiento hacia atrás* o como *razonamiento guiado por objetivos* y suele ser más bien usado para generar explicaciones. Este último tipo de inferencia es típica de los sistemas expertos, como por ejemplo, los dedicados a la diagnosis médica. En adelante, solamente nos referiremos a la inferencia *modus ponens*.

Basados en la definición 2.12 podemos encontrar la *regla de inferencia de composición* $R \rightarrow I$ como un *modus ponens generalizado* siguiendo la notación establecida anteriormente como sigue:

Si F_1 es un conjunto difuso elemental en U_1, y R es una relación difusa en $U_1 \times U_2$, el subconjunto difuso de I en U_2 que será inducido por F_1 viene dado por la composición de $F_1 \circ R$, que es:

$$I = F_1 \circ R = \int_{U_2} \mathbf{sup_{u_1}} [Memb_{F_1}(u_1) * Memb_R(u_1, u_2)]/u_2 \qquad (2.17)$$

Usemos de nuevo el ejemplo de las temperaturas interna(t_i) y externa (t_e). Supongamos que conocemos que: *la temperatura interna es agradable*, pero no contamos con la medición de la temperatura externa. Si conocemos alguna regla que nos relacione las variables bajo ese valor lingüístico podemos inferir la temperatura externa. Por ejemplo:

Antecedente 1: **Si**	t_i es *agradable* **entonces** t_e es *cálida*.	
Antecedente 2:	t_i es *agradable*	$(Memb_{T_{i,agradable}}(24^oC) = 0.8)$
Conclusión:	t_e es *cálida*	usando la ec. (2.17) con $*$ como **min**, podemos encontrar $Memb_{T_{e,cálida}}$ como:

$$\begin{aligned} Memb_{T_{i,agradable} \circ T_{e,cálida}} &= Memb_{T_{i,agradable}}(t_i) \circ Memb_{T_{e,cálida}}(t_e) \\ &= \min(Memb_{T_{i,agradable}}(t_i), Memb_{T_{e,cálida}}(t_e)) \\ &= \min(0.8, Memb_{T_{e,cálida}}(t_e)) \end{aligned}$$

lo cuál es representado en la figura 2.8.

Un tipo de operación frecuentemente encontrado en el razonamiento aproximado es lo que se conoce como *cuantificador* o *modificador lingüístico*. Existe una gran variedad de este tipo de modificadores como: muy, casi, poco, siempre, cerca, lejos, parecido, la mayoría, etc. Un buen compendio y discusión de ellos se encuentra en [Zadeh 72]. Un ejemplo de su uso podría ser en la implicación siguiente:

Antecedente 1: **Si**	t_e es *calurosa* **entonces** t_i es *calurosa*
Antecedente 2:	t_e es *MUY calurosa*
Conclusión:	t_i es *MUY calurosa*.

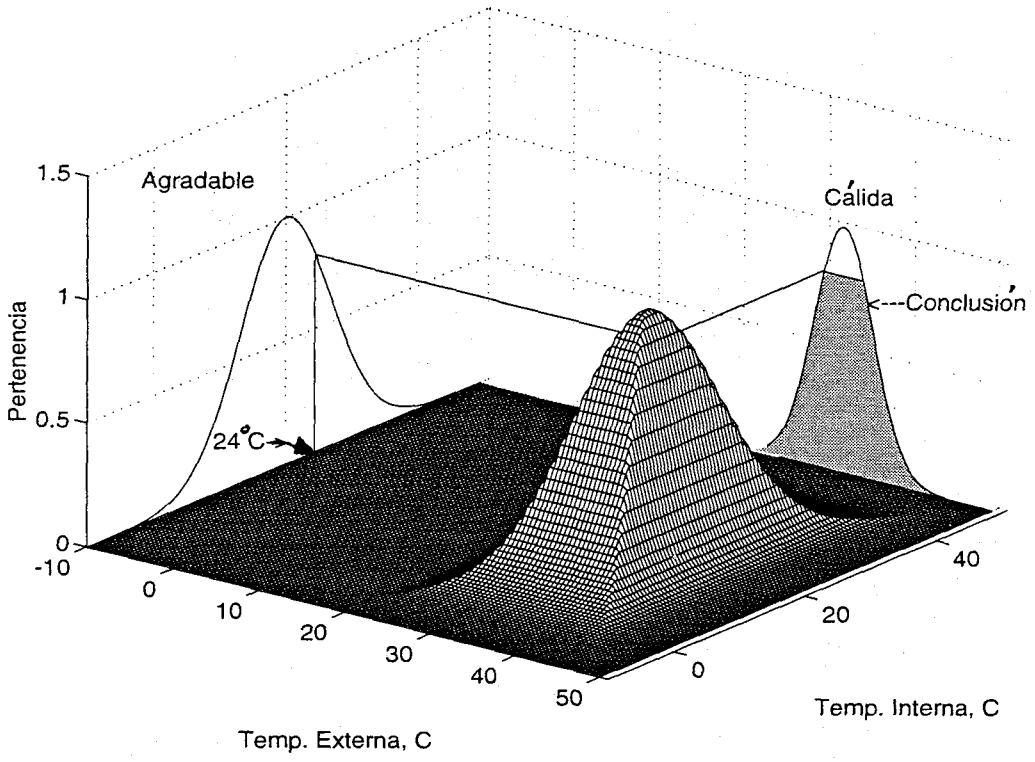


Figura 2.8: Implicación: $T_{i,agradable} \circ T_{e,cálida}$.

Aunque no hemos definido *MUY* calurosa anteriormente, este cuantificador puede verse como una operación difusa sobre el conjunto difuso $F_{T_{e,calurosa}}$, por ejemplo:

$$Memb_{T_{e,MUYcalurosa}}(t_e) = Memb^2_{T_{e,calurosa}}(t_e)$$

El cuantificador "MUY" añade precisión a un conjunto difuso, es decir, 'MUY caluroso" es un poco menos difuso que "caluroso".

En razonamiento aproximado, el conocimiento es representado mediante reglas de inferencia. Cuando se tiene un conjunto de reglas suelen ocurrir dos tipos de situaciones. Una, es la conocida como el *encadenamiento de reglas*, que nos permite utilizar las conclusiones alcanzadas en un momento dado como un *disparo* para otras reglas que a su vez podrían ser premisas para *disparar* otras reglas y así sucesivamente. Su tratamiento puede estudiarse en [Zadeh 75]. La otra situación se presenta cuando las premisas de la regla permiten disparar de manera simultánea más de una regla y es necesario extraer de ellas una conclusión consistente o, si además se presenta simultáneamente con el caso

anterior, resolver el conflicto del orden en que las reglas se ejecutan. El primer fenómeno no se presenta en el caso del control difuso ya que la conclusión de cada regla suele ser directamente una potencial acción de control. El segundo fenómeno, por el contrario, es una situación muy común en control difuso y la tarea a realizar es cómo combinar el efecto de las diferentes conclusiones (potenciales acciones de control).

Existe un último nivel de complicación en lo que se refiere a operaciones difusas requeridas para expresar el conjunto de reglas de un controlador. Los sistemas de control son clasificados según la multiplicidad de entradas y salidas como: SISO (una entrada, una salida), MISO (múltiples entradas, una salida) y MIMO (múltiples entradas, múltiples salidas). La descripción que viene a continuación se referirá a sistemas MISO. La razón para ello es que un sistema MIMO con n_{output} salidas puede siempre descomponerse en n_{output} sistemas MISO independientes; por otro lado, un sistema MISO con n_{input} entradas no puede descomponerse en n_{input} sistemas SISO independientes. El sistema SISO es un caso especial de un sistema MISO con $n_{input} = 1$.

Sea R^i la $i^{\text{ésima}}$ regla de un sistema MISO con N reglas, donde a_m es la $m^{\text{ésima}}$ variable difusa que adquiere el valor lingüístico A^i_m en el **antecedente** de la $i^{\text{ésima}}$ regla, y c^i la variable difusa de salida con el valor lingüístico C^i respectivo, la representación del sistema MISO es:

R^1**: Si** a_1 es A^1_1, **y**, $\cdots$, **y** a_M es A^1_M, **entonces** c^1 es C^1

también,

R^2**: Si** a_1 es A^2_1, **y**, $\cdots$, **y** a_M es A^2_M, **entonces** c^2 es C^2

también,

$\vdots$

también,

R^N**: Si** a_1 es A^N_1, **y**, $\cdots$, **y** a_M es A^N_M, **entonces** c^N es C^N

donde, si interpretamos **también**[10] como una composición de unión y utilizando las ecuaciones (2.11) y (2.15), podemos representar el conjunto de reglas $\mathcal{R}$ del sistema MISO como:

[10]este operador es conocido como el operador *'also'* o como el operador *'or else'*

$$\mathcal{R} = R^1 \cup R^2 \cup \cdots \cup R^N = \bigcup_{i=1}^{N} (A_1^i(a_1) \times \cdots \times A_M^i(a_M) \times C^i(c^i)) \qquad (2.18)$$

cuya función de pertenencia es:

$$Memb_{\mathcal{R}} = \max_i [Memb_{A_1^i}(a_1) * \cdots * Memb_{A_M^i}(a_M) * Memb_{C^i}(c^i)] \qquad (2.19)$$

Supongamos ahora que las variables de entrada al sistema MISO adquieren los valores $a_1 = a_1^*, a_2 = a_2^*, \cdots, a_N = a_N^*$, el problema de *razonamiento aproximado* es el de encontrar el valor apropiado para la variable de salida c. Cada regla tiene un nivel de *disparo* que depende del *grado* de similitud que guarda, el estado actual de las variables de entrada del sistema y la estructura de los antecedentes de cada regla. Existen dos formas para encontrar el *disparo* del conjunto de reglas:

1. La inferencia basada en la operación de composición. En este caso el significado de cada regla es agregada en una relación difusa que combina el conjunto completo de reglas. El *disparo* es conseguido mediante el uso de la operación de composición entre la entrada escalar fusificada y la relación difusa del conjunto de todas las reglas. Como resultado de la composición se obtiene el valor difuso global de la variable de salida.

2. La inferencia basada en cada regla individual. Aquí el *disparo* del conjunto es realizado regla a regla de la siguiente manera: a) se encuentra el *grado de similitud*[11] entre los valores escalares de las variables de entrada y los valores lingüísticos correspondientes a la sección de antecedentes de la regla en cuestión; b) en base a la similitud encontrada en a) se recorta el conjunto difuso del valor lingüístico correspondiente dentro del consecuente de la regla; y c) los valores recortados de los subconjuntos difusos encontrados en b) son agregados en un conjunto difuso global que describe el valor de la salida.

Desde el punto de vista computacional, el método 2 es mucho más eficiente y requiere menos recursos de memoria. La aplicabilidad de uno u otro está en función del tipo de interpretación que se le haya dado a la implicación

[11] degree of match

difusa. Se ha demostrado [Drian 93] que para el tipo de interpretaciones generalmente utilizadas en control difuso, ambas formas son equivalentes por lo que normalmente se prefiere la inferencia basada en el disparo de reglas individuales.

Si ahora definimos α_i como:

$$\alpha_i = Memb_{A_1^i}(a_1) * \cdots * Memb_{A_M^i}(a_M) \tag{2.20}$$

y sustituimos en la ecuación (2.19), obtenemos:

$$Memb_{\mathcal{R}} = \max_i [\alpha_i * Memb_{C^i}(c^i)] \tag{2.21}$$

α_i es conocido como el *nivel de disparo* de la $i^{\text{ésima}}$ regla respecto de los valores $a_1 = a_1^*, a_2 = a_2^*, \cdots, a_N = a_N^*$.

2.7 Arquitectura de Controladores Difusos

Aunque la estructura de todos los controladores difusos es aproximadamente igual, no existe una definición general para sus componentes internos. Cada diseñador debe elegir y decidir cuidadosamente un buen número de cuestiones. Desde cierto punto de vista, la diversidad de posibilidades y de métodos para elegir una solución particular a cada una de las cuestiones dependiendo del caso de que se trata, hace que hasta ahora, el diseño de un controlador difuso sea más un arte que una ciencia.

La estructura general de un controlador difuso es la mostrada en la figura 2.9, en la cual puede apreciarse que consta de cuatro módulos principales:

- *Interfaz de Fusificación.*[12]
- *Base de Conocimiento.*
- *Mecanismo de Inferencia.*

[12]No existe un término en idioma Español que describa el proceso de transformar la información concreta en información difusa. Para efectos de esta tesis, se define el término *fusificación* como significando: a) la codificación de un valor expresado en forma 'escalar' en un(os) valor(es) 'difuso(s)', en el sentido de las definiciones 2.1 y 2.2, y b) el proceso de representar la información *concreta* en información *difusa*.

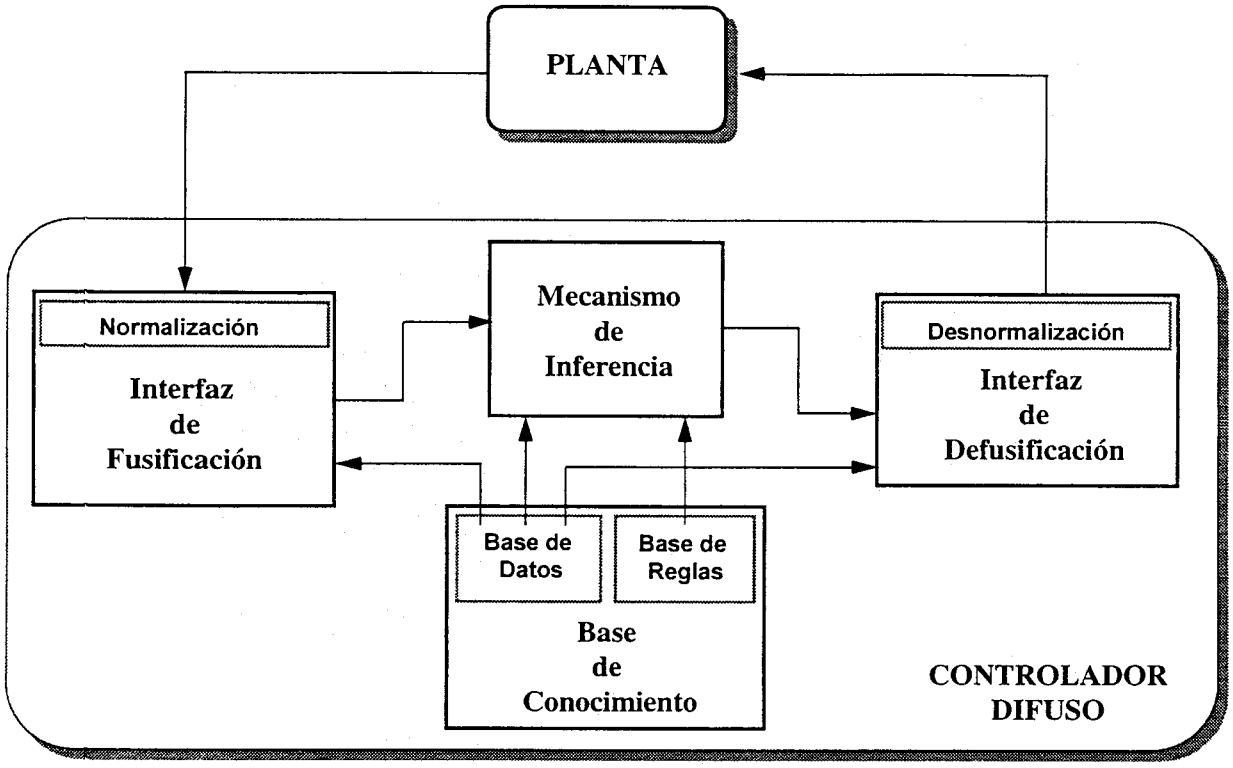


Figura 2.9: Estructura general de un Controlador Difuso.

- *Interfaz de Defusificación.*[13]

En los apartados siguientes daremos un tratamiento más detallado a cada uno de ellos. No obstante, antes de hacerlo es importante tener en consideración que aunque el diseño de un controlador difuso implica la definición de cada uno de los cuatro módulos, se asume que en algún sentido, antes de comenzar la especificación de cada uno de ellos en detalle, ya se conoce (o se ha definido) aproximadamente la forma que tendrá. Esto es debido a que existe una estrecha interdependencia entre todos ellos; por un lado, a medida que se van definiendo los parámetros de un módulo y se avanza hacia la especificación de los otros, un proceso de iteración es requerido; por otro lado, el proceso puede empezar a partir de establecer cualquiera de ellos y avanzar en el diseño, ya sea hacia adelante o hacia atrás, en la determinación de los

[13]Palabra que define el proceso inverso de la fusificación, es decir, el proceso de convertir un valor difuso (o una conclusión difusa) en información concreta expresada mediante valores escalares.

demás. Por ejemplo, no es posible establecer la forma en la que la información medida será convertida en difusa sin conocer previamente la forma en la que se obtendrán las reglas de control. Por ejemplo, si las reglas son obtenidas a partir del conocimiento heurístico de un operador experto, es muy posible que el ingeniero de control decida la selección del número de clases o valores lingüísticos de cada variable difusa en función del conocimiento expresado por el experto. Por el contrario, cuando no se cuenta con información heurística para describir los criterios de operación de la planta, se fijará un número aproximado de clases y la definición de las reglas deberá de limitarse al número predefinido. Si el comportamiento del controlador no es el adecuado, es posible que un aumento en el número de clases permita aumentar también la resolución del controlador. Las ideas de interdependencia y de un proceso de prueba y error deben ser mantenidas en mente durante la descripción de la arquitectura general de los controladores difusos que se da a continuación. El orden de exposición no es el orden de diseño; más bien, es el orden que sigue el flujo de información a medida que las señales de entrada son procesadas para encontrar una señal de control como respuesta.

2.7.1 Interfaz de Fusificación

Hay varias **funciones** que pueden estar incluidas en el proceso llevado a cabo por la interfaz de fusificación:

- Obtener la medida de las variables de entrada. En las aplicaciones de control difuso los valores obtenidos desde los sensores suelen ser valores escalares. Normalmente se asume que estas señales son medidas de manera exacta y que no se introduce imprecisión. Para modelar esto con conjuntos difusos, cada señal de entrada s_0 se representa como un conjunto difuso F con una función de pertenencia dada por (figura 2.10a):

$$Memb_F(s) = \begin{cases} 1 & \text{si } \ s = s_0, \\ 0 & \text{resto de los casos.} \end{cases} \tag{2.22}$$

  Si la lectura del sensor produce ruido, la interfaz de fusificación debe convertir los datos registrados probabilísticamente en números difusos [Duboi 85]. Esto puede ser modelado utilizando una función de pertenencia triangular. El vértice superior del triángulo corresponde a la media del valor registrado y el ancho de su base a la desviación estándar. La medida se representa por la intersección entre la distribución de los

datos medidos y los valores lingüísticos de la variable difusa (almacenados en la base de datos) tal y como se muestra en la figura 2.10 (b) (una aplicación de este caso viene en [Muray 85]).

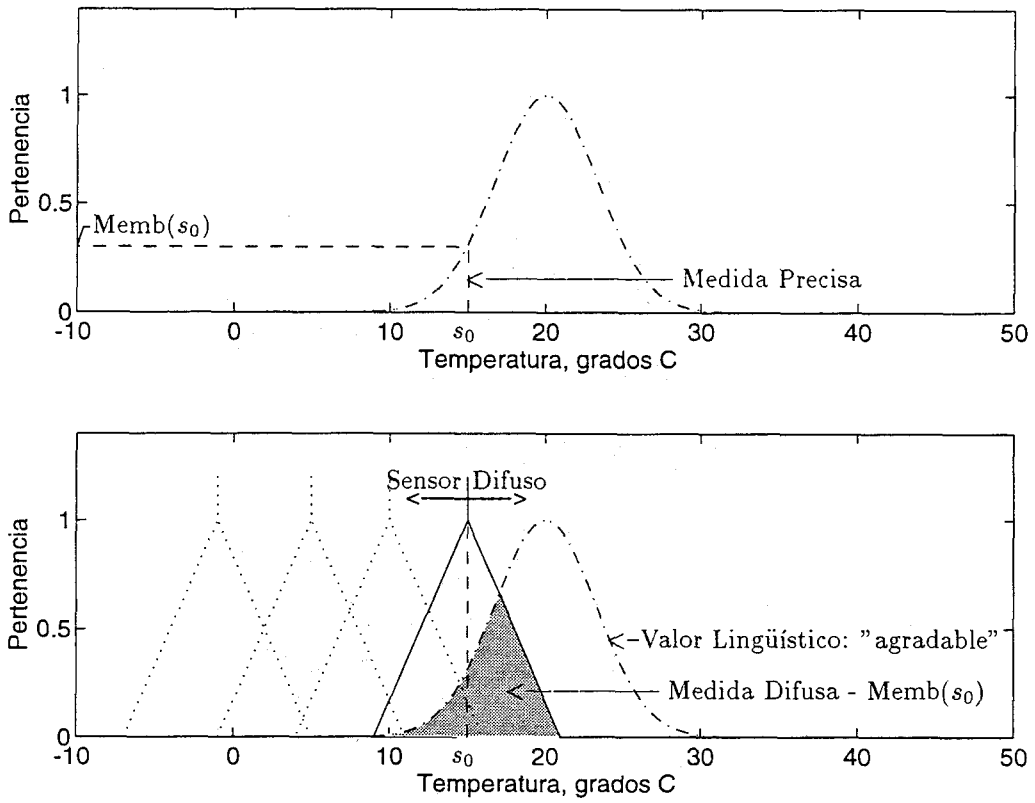


Figura 2.10: Obtención de $Memb(s_0)$: (a) lectura precisa; y (b) lectura difusa del sensor.

- Normalización. Si la gama de valores asignados a cada variable lingüística en los antecedentes de la base de reglas está definida en un dominio normalizado, al valor medido por el sensor debe aplicársele el factor de escala correspondiente (almacenado en la base de datos), que relaciona el universo de discurso de la variable medida con el espacio normalizado de la variable lingüística. Como ha sido comentado antes, una de las ventajas de la normalización es la posibilidad de disminuir el efecto de la dependencia del contexto permitiendo hacer diseños independientes del dominio. La normalización, sin embargo, conlleva algunas desventajas como son: a) la pérdida de la relación directa entre la expresividad lingüística con el universo de discurso, b) el cambio en la sensibilidad

de los parámetros que puede tener serias repercusiones en la estabilidad del controlador, y c) el aumento en el coste computacional por la doble operación normalización y 'desnormalización' de las señales.

- Para cada señal medida (o medida y normalizada) debe encontrarse el conjunto de valores de pertenencia de los valores lingüísticos definidos para la variable difusa correspondiente, utilizando para ello las funciones de pertenencia almacenadas en la base de datos. La manera de hacer esto dependerá del mecanismo de inferencia seleccionado según lo descrito en la sección 2.6. Básicamente la diferencia es la siguiente. En un caso, cada regla es agregada en una relación difusa que describe el conjunto completo de reglas, es preciso encontrar y almacenar los valores del conjunto de las $Memb^{regla_i}$ para el juego completo de reglas. La inferencia es realizada, vía función de composición, con la entrada escalar fusificada y como resultado de la composición se obtiene el conjunto difuso que describe el valor de salida. En el otro caso, cada regla va siendo evaluada por separado en función de los valores escalares de sus entradas que son fusificados y utilizados para encontrar el valor α_i correspondiente a cada regla. Como resultado de la inferencia solamente se almacena el conjunto difuso recortado de la variable de salida. La combinación de la contribución de cada regla en un conjunto difuso es el valor difuso de la variable de salida.

Normalmente cierta cantidad de información es "perdida" durante el proceso de fusificación cuando la variable cuantitativa concreta es transformada en una variable difusa. No obstante, cuando la medida de los sensores son precisas, no existe ninguna limitación que nos impida "recordar" esa información si hubiese alguna forma de utilizarla. La función de codificación difusa utilizada por nuestra metodología de diseño de controladores difusos, expuesta más adelante en el capítulo 3, permite un esquema bajo el cuál esto es posible, dando un grado de precisión mayor a la vez que se preservan las cualidades benignas de la estructura de inferencia difusa.

2.7.2 Base de Conocimiento

Identificación de variables. La primera tarea a llevar a cabo cuando se diseña un control difuso es identificar cuáles serán las variables de entrada y salida del controlador, así como los parámetros que las definirán. La determinación de qué variables son utilizadas para diseñar el controlador quedará, a partir

de este momento, implícita en la configuración del controlador y en la Base de Conocimiento. Respecto a la selección de variables de entrada y salida del controlador, la gran mayoría de los controladores difusos reportados en la literatura asumen la misma estructura. Mientras que resulta obvio que la variable de control (o la variación de ella en el tiempo) viene determinada por el sistema particular que se desea controlar, no lo es necesariamente, cuáles deben ser las variables de entrada. El error entre las señales de consigna y de salida, y la razón de cambio (derivada) de este error, son las variables de entrada usadas por casi todos los diseñadores de controladores difusos. Nuestra metodología selecciona automáticamente las variables de entrada del controlador de entre las variables físicas que describen al sistema (presión, temperatura, posición, etc.) en función de la causalidad (espacial y temporal) que guardan con respecto a la variable de control. Este es una importante diferencia entre la metodología de diseño presentada en esta tesis y los métodos tradicionales de diseño de controladores difusos y cuyos detalles serán presentados en el capítulo 6.

La *Base de Conocimiento* está formada a su vez por dos componentes. El primero es una *Base de Datos* que mantiene los parámetros y las características de cada una de las variables lingüísticas, el segundo es la *Base de Reglas* que almacena el conocimiento respecto a la causalidad entre las variables de entrada y las variables de control. Desde un punto de vista, se puede decir que la *base de datos* da los elementos que permiten la expresión de la *base de reglas*, pero visto de manera inversa, también puede decirse que es la *base de reglas* la que, una vez expresada, se formaliza mediante los parámetros registrados en la *base de datos*.

2.7.2.1 Base de Datos

La base de datos proporciona información a la interfaz de fusificación, al mecanismo de inferencia y a la interfaz de defusificación. Así mismo, proporciona los elementos del *lenguaje* para construir las reglas. Su contenido está estructurado con respecto a cada variable lingüística, proporcionando para cada una:

- El número de valores lingüísticos.
- Los parámetros de la función de pertenencia correspondientes a cada valor lingüístico.

- El factor de normalización y desnormalización (si lo hay) aplicado al universo de discurso.

Número de valores lingüísticos. Una vez determinadas las variables de entrada y de salida del control difuso, debe ser definido el nivel de resolución[14] o el número de valores lingüísticos de cada variable lingüística. Por ejemplo, en el caso del control del aparato de aire acondicionado mencionado en el apartado 2.5.1, dependiendo de la calidad de comportamiento que se desee para el controlador, los valores lingüísticos de la variable T_{Int} podrían ser:

$$\{T_{Int,calurosa}, T_{Int,agradable}, T_{Int,fría}\},$$

o bien:

$$\{T_{Int,calurosa}, T_{Int,cálida}, T_{Int,agradable}, T_{Int,fresca}, T_{Int,fría}\}$$

mostrados anteriormente en las figuras 2.2 y 2.3. La determinación del número de valores lingüísticos es también conocida bajo los nombres de *'clustering'*, *cuantificación*, *partición* o *discretización* del universo de discurso y es una de las operaciones que involucran mayor heurística en el diseño de un controlador difuso. Existen algunos métodos para encontrar la partición óptima del dominio de una variable en función de un conjunto representativo de datos [Krish 93], [Chees 90], [Fishe 90] y [Dubes 79]. Sin embargo, el problema se vuelve sumamente complejo cuando es de tipo multivariable. Los algoritmos suelen tener una estructura iterativa aumentando poco a poco el número de clases y usando una cierta ϵ como criterio de convergencia. Un interesante método particularmente útil en control difuso se presenta en [Sugen 93].

Funciones de Pertenencia. Como fue indicado anteriormente, cada valor lingüístico es definido por su correspondiente función de pertenencia. En el desarrollo de controladores difusos han sido utilizados diferentes 'tipos' de funciones de pertenencia. Debido a la facilidad de su descripción paramétrica y funcional, las funciones más comúnmente usadas son: triangulares, trapezoidales, gausianas y monótonas (figura 2.11). Cuando se ha determinado el tipo de función de pertenencia para cada valor lingüístico $F_{i,j}$, en el universo de discurso U_i con $Memb_F : U \rightarrow [0,1]$, los parámetros que la describen deben de ser fijados. Los más importantes son:

- *Valor Máximo*

$$m_{i,j} \quad \text{tal que} \quad Memb_{X_{i,j}}(m_{i,j}) = 1.0$$

[14] conocido también como grado de 'granularidad'.

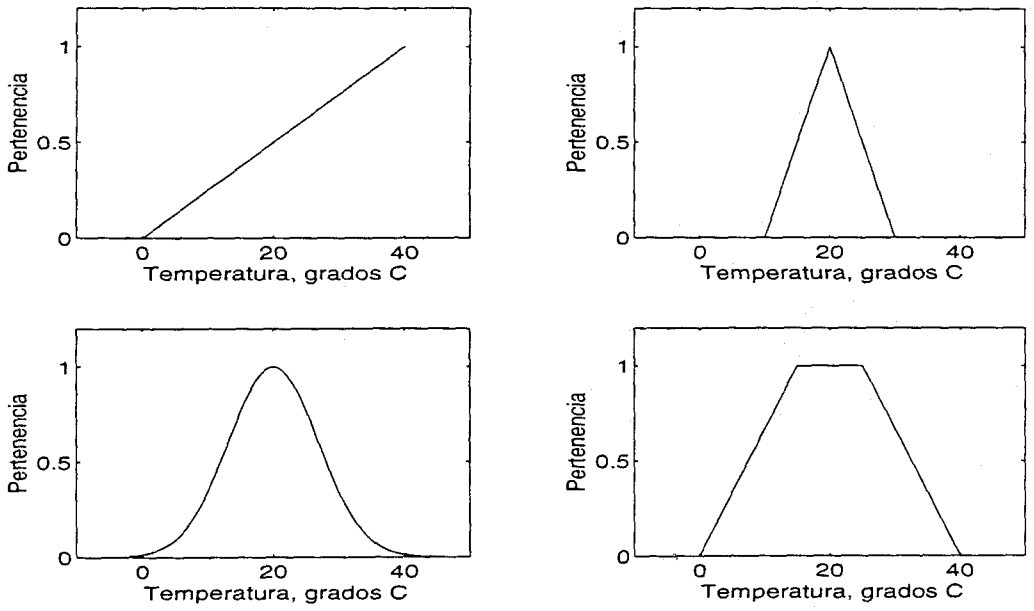


Figura 2.11: Tipos más comunes de funciones de pertenencia.

Si la función es triangular o gausiana el valor máximo es único, si la función es trapezoidal el valor máximo es un intervalo. No debe olvidarse que para cumplir con los requerimientos matemáticos de los números difusos es necesario que el valor máximo sea siempre 1.0.

- *Soporte:*

$$X_{i,j} = \{x_i\} \qquad \text{tal que} \qquad Memb_{X_{i,j}}(x_i) > 0$$

El intervalo de soporte de una función difusa es dividido en dos regiones por el valor máximo, estableciéndose la región izquierda y derecha. Si ambas regiones son iguales, se dice que la función es simétrica.

- *Punto de cruce*

$$c_{i,j} \qquad \text{tal que} \qquad Memb_{X_{i,j}}(c_{i,j}) = \alpha$$

La determinación de este parámetro es muy importante ya que representa el grado de intersección entre dos (o más) valores lingüísticos contiguos. A través de resultados de carácter empírico [Bovie 91], se ha determinado que un valor óptimo para α suele ser 0.5, el cual es el valor utilizado generalmente en la literatura de control difuso.

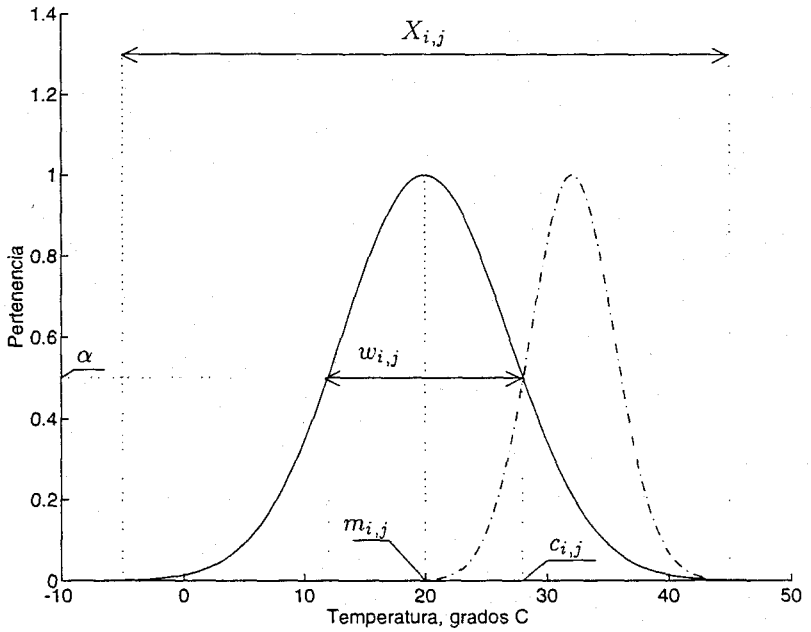


Figura 2.12: Algunos parámetros de las funciones de pertenencia.

- *Ancho de banda*

$$w_{i,j} \quad \text{tal que} \quad \forall z \in w_{i,j}, \; Memb_{X_{i,j}}(z) \geq \alpha$$

La figura 2.12 ilustra estos parámetros. La selección de los parámetros de la función de pertenencia puede afectar de manera significativa el comportamiento y la estabilidad del controlador [Chen 93] y [Bovie 91]. En la mayoría de las aplicaciones reportadas, la selección de ellos es realizada mediante un proceso de prueba y error [Takag 90]. Aunque no son de aplicación general, en algunas publicaciones se reportan técnicas para el ajuste dinámico de los parámetros, basándose en la optimización de índices de calidad de operación [Schar 89] y [Sugen 88].

Factores de escala de la normalización. Cuando el diseño del controlador incluye las operaciones de normalización y 'desnormalización', la base de datos mantiene el registro de los diferentes factores de escala aplicados a cada una de las señales de entrada y de salida permitiendo 'mapear' los valores físicos del sistema real en el dominio normalizado en el que opera el controlador.

2.7.2.2 Base de Reglas

Una vez definidas las variables difusas y sus respectivos valores lingüísticos, es posible expresar el modelo cualitativo del controlador difuso mediante el conjunto de reglas que las relacionan.

Obtención de Reglas. Como en el caso de la base de datos, no existen métodos generales para derivar las reglas. En [Sugen 85a], Sugeno presenta cuatro métodos para derivar las reglas del controlador difuso.

1. Experiencia y conocimiento de un experto.
2. Modelando las acciones de control que ejerce el operador.
3. Modelando un proceso.
4. Auto organización.

Sugeno enfatiza que ninguno de ellos debe ser considerado como mutuamente excluyente con respecto a los demás, pudiéndose combinar y complementar entre ellos. Siendo el más *natural* e intuitivo, el **primer** método ha sido el más ampliamente usado (por lo menos en la primera generación de controladores difusos) [Mamda 75] y [Gupta 81]. Al ser el más heurístico resulta ser el menos estructurado. En este método, el conocimiento empírico de (un) experto(s) es plasmado en un conjunto de reglas mediante la verbalización introspectiva de sus experiencias, ya sea en forma espontánea o por medio de herramientas de adquisición del conocimiento como el uso de cuestionarios o entrevistas. Las principales ventajas que le caracterizan son la expresividad y su facilidad para capturar conocimiento poco estructurado, cualidad de la que hereda su principal desventaja, la dificultad de sistematización. Algunos ejemplos de aplicación de este método pueden ser encontrados en [King 77], [Sugen 85], [Langa 88] y [Efsta 89].

En el **segundo** método, el conjunto de reglas resultante es similar en el contenido y en la forma al obtenido con el método anterior. Sin embargo, en este caso la construcción de la base de reglas es realizada mediante el modelado, por un ingeniero de control difuso, a partir de la observación pasiva de las acciones y condiciones de control que lleva a cabo el operador. Ejemplos de esta forma de derivar el conjunto de reglas son reportados en [Takag 83], [Sugen 85a] y en [Murak 85].

El **tercer** método, aún en sus primeras etapas de desarrollo, está basado en el desarrollo de un modelo difuso del proceso que se desea controlar. Un modelo

aproximado de la planta es configurado mediante implicaciones que describen los posibles estados del sistema. Una vez identificado el modelo difuso de la planta, se construye un controlador difuso, ya sea mediante un conjunto de reglas o mediante una relación difusa explícita, en las que de alguna manera se representan las relaciones inversas entre las salidas y entradas del sistema. El enfoque subyacente en este método es similar en su naturaleza al enfoque que utiliza la teoría de control tradicional. Actualmente este método comienza a gozar de gran aceptación y expansión ya que mediante esta aproximación se alcanzan mejores índices de calidad de operación y de seguridad permitiendo una estructura que, aunque resulta más compleja, es mucho más tratable desde el punto de vista teórico. Algunas herramientas para la identificación de la estructura y de los parámetros del proceso pueden ser consultadas en [Buckl 86], [Sugen 88], [Pedry 91], [Pedry 93] y [Yager 94]. Ejemplos de bases de reglas de controladores obtenidos bajo este método se encuentran en [Sugen 85b] y [Drian 93].

El **cuarto** y último método se refiere al desarrollo de reglas que pueden ser ajustadas sobre el tiempo para mejorar la calidad de operación del controlador. El primer controlador difuso con capacidades de aprendizaje fue desarrollado por Procyk y Mamdani [Procy 79]. A ese tipo de controladores se les conoce como SOC de "Self-Organizing Process". Otros desarrollos interesantes basados en controladores tipo SOC están en [Mamda 81], [Sugen 85a] y [Layne 93]. La idea central es la siguiente: el conjunto de reglas es dividido en dos partes; un primer grupo contiene un conjunto de reglas exactamente iguales a las de un controlador difuso 'normal'; el otro conjunto de reglas, conocidas como metareglas, valora el comportamiento del control en función de un índice de calidad que permite crear nuevas reglas y modificar las existentes. Existe una vertiente alternativa de este mismo método, es la que utiliza redes neuronales para modelar las relaciones entre las variables de entrada y de salida. Se han obtenido diseños neuronales con capacidades similares a los SOC que parecen trabajar con igual calidad [Naren 90] y [Beren 92].

Estructura de las Reglas. Para mostrar el tipo de reglas almacenadas en las bases de reglas, consideremos el siguiente ejemplo que muestra una típica estructura de las reglas obtenidas mediante el tercer método. Como se ha mencionado anteriormente, el error entre las señales de consigna y de salida ($error$), y la razón de cambio (derivada) de este error ($\Delta error$), son las variables de entrada seleccionadas por casi todos los diseñadores de controladores difusos, mientras que la salida es la acción de control ($control$) o bien su cambio ($\Delta control$). Utilizando estas variables es posible construir conjuntos de reglas que representen controladores difusos *similares* a los varios tipos de controladores clásicos conocidos. Si para ejemplificar, suponemos

valores lingüísticos $\{Positivo, Cero, Negativo\}$ para todas las variables, la estructura de las reglas en cada caso será de la forma:

Tipo P:
Si *error* es *Positivo*
entonces *control* es *Positivo*

Tipo PD:
Si *error* es *Cero* **y** $\Delta error$ es *Positivo*
entonces *control* es *Cero*

Tipo PI:
Si *error* es *Negativo* **y** $\Delta error$ es *Positivo*
entonces $\Delta control$ es *Cero*

Tipo PID:
Si *error* es *Negativo* **y** $\Delta error$ es *Positivo* **y** $\delta error$ es *Positivo*
entonces *control* es *Cero*, donde $\delta error$ es la suma de errores acumulados en el tiempo.

Si en vez de usar las variables de error se utilizan las variables de estado más frecuentes en el primer o segundo método, la estructura de las reglas tendrían la forma:

Si *Temperatura* es *Alta* **y** *Humedad* es *Baja*
entonces *Potencia* es *Normal*

Si *Temperatura* es *Alta* **y** *Humedad* es *Alta*
entonces *Potencia* es *Alta*

Los dos ejemplos de reglas descritos anteriormente tienen en común el hecho de que, tanto sus antecedentes como sus consecuentes describen estados particulares del sistema. Para efectos de identificación, a este tipo de reglas les llamaremos reglas del tipo I. Un segundo tipo de reglas, reglas del tipo II o reglas del tipo **Sugeno**, ha sido aplicado con éxito en varios procesos industriales. A diferencia de las del tipo I, el consecuente de las reglas del tipo II tiene la estructura de una función lineal. Es decir, el consecuente de cada regla no es una variable lingüística, sino un valor escalar obtenido mediante una función lineal de los valores de la función de pertenencia de cada antecedente. La combinación de los consecuentes de las reglas *disparadas* se hace mediante un promedio ponderado del escalar obtenido en cada regla y el

resultado de la composición difusa de los antecedentes actuado como factor de contribución. La estructura de las reglas es la siguiente:

Si t_1 es X_1, **y** t_2 es X_2, **y**, $\cdots$, **y** t_n es X_n
entonces control $= f(t_1, t_2, \cdots, t_n)$

donde f es una función lineal de las variables de estado $\{t_1, t_2, \cdots, t_n\}$

Aunque la aproximación utilizada por la metodología de diseño, la cual es objetivo de esta tesis, contiene algunos elementos de los métodos tercero y cuarto, es sustancialmente diferente. De alguna manera, todos los métodos de diseño de controladores (difusos o no) se basan en una aproximación de la dinámica inversa de la planta. Nuestra metodología utiliza un modelo de referencia como un profesor y un modelo directo de la planta para representar la planta real, similar a como se hace en el tercer método. A partir de esto, se genera de manera automática un modelo en cascada del modelo de referencia y de la dinámica inversa de la planta utilizando un potente manipulador algebraico llamado Dymola [Celli 93]. Los modelos anteriores son utilizados para que mediante experimentos de simulación se extraigan datos, a partir de los cuales la metodología de razonamiento inductivo difuso (descrita en el capítulo 3) identificará el modelo de control óptimo. Esto resulta esencialmente en la síntesis automática de la base de reglas del controlador difuso, cuyos detalles se proporcionan en el capítulo 5.

2.7.3 Mecanismo de Inferencia

El corazón de un controlador difuso, y de cualquier sistema basado en lógica difusa, es el mecanismo de inferencia. Aunque, como dijimos en la sección 2.6, el mecanismo de inferencia en los controladores difusos es relativamente simple, con respecto por ejemplo a los sistemas expertos en donde el encadenamiento de reglas complica la estructura del razonamiento aproximado, los requerimientos de precisión y estabilidad hacen que el trabajo de selección y evaluación del método adecuado sea, aún, una labor compleja. Varios estudios teóricos respecto de la implicación difusa y los criterios de selección según el caso aplicación, han sido efectuados [Mizum 82], [Duboi 85b], [Trill 85] y [Pedry 93], algunos específicamente dentro del campo de control difuso [Takag 85], [Bousl 92] y [Pisku 92]. Las conclusiones generales de estos trabajos permiten apreciar la amplitud del campo de posibilidades que deben explorarse aún, particularmente cuando son aplicados a sistemas difusos específicos, tales como

los controladores difusos. Desde el punto de vista práctico, basado en la revisión del estado del arte del control difuso, puede decirse que dentro del gran número de posibles métodos y variantes, han sido sólo unos cuantos los que han sido verdaderamente aplicados y experimentados. Dentro de este apartado nos referiremos únicamente a los que consideramos más relevantes.

Partiendo de la estructura de la base de reglas descrita en la sección 2.6, consideraremos cuatro mecanismos de inferencia:

Mecanismo de Inferencia Sup-Min. Este método es también conocido como el método de Mamdani. Está basado en la interpretación de la implicación difusa como la composición de intersección, definición 2.10, e interpretando el operador $*$ como mínimo, es decir, el **'y'** en el cuerpo de antecedentes es operando con el mínimo. La combinación de reglas es realizada bajo la ecuación (2.18) e interpretando la unión como máximo. Este mecanismo utiliza las reglas del tipo I, descritas en el apartado anterior. Este es, quizás, el método de razonamiento más frecuente en la literatura de aplicaciones de control difuso.

Partiendo de la ecuación (2.21), e incorporando las suposiciones anteriores obtenemos:

$$Memb_{\mathcal{R}} = \max_i[\min(\alpha_i, Memb_{C^i}(c^i))] \tag{2.23}$$

donde:

$$\alpha_i = \min(Memb_{A_1^i}(a_1), \cdots, Memb_{A_M^i}(a_M)) \tag{2.24}$$

El proceso de este mecanismo es ilustrado en la figura 2.13. La obtención de un valor concreto de la señal de control requiere la aplicación de alguno de los métodos de defusificación descritos en la sección siguiente.

Mecanismo de Inferencia Sup-prod. Este método es también conocido como método de Larsen. Igualmente que en el sup-min, este método está basado en la interpretación de la implicación difusa como la composición de intersección dada anteriormente (def. 2.10), pero en este caso, el operador $*$ se interpreta como producto, es decir, el **'y'** que une los antecedentes es evaluado como el producto de sus funciones de pertenencia. La combinación de reglas es realizada según (2.18)$--$ $>$(2.21) bajo el operador de unión usado como máximo. El interés de este método de razonamiento actualmente está aumentando, ya que parece obtener resultados mejores que usando el clásico

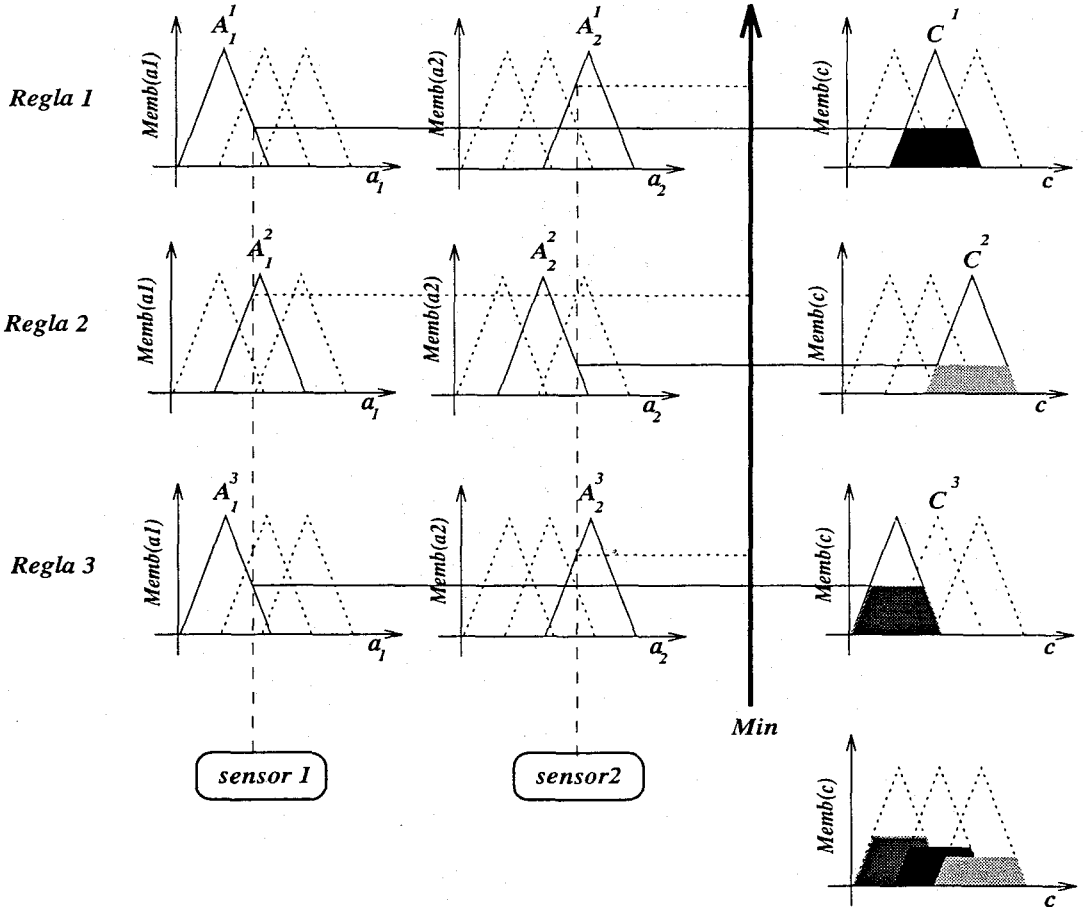


Figura 2.13: Mecanismo de razonamiento sup-min.

sup-min.

Partiendo de 2.21, e incorporando las suposiciones anteriores:

$$Memb_{\mathcal{R}} = \max_i [\alpha_i \cdot Memb_{C^i}(c^i)] \tag{2.25}$$

donde:

$$\alpha_i = Memb_{A_1^i}(a_1) \cdot Memb_{A_2^i}(a_2) \cdot ... \cdot Memb_{A_M^i}(a_M) \tag{2.26}$$

El proceso de este mecanismo es ilustrado en la figura 2.14. La obtención de un valor escalar de salida requiere de aplicar alguno de los métodos de defusificación descritos en la sección siguiente.

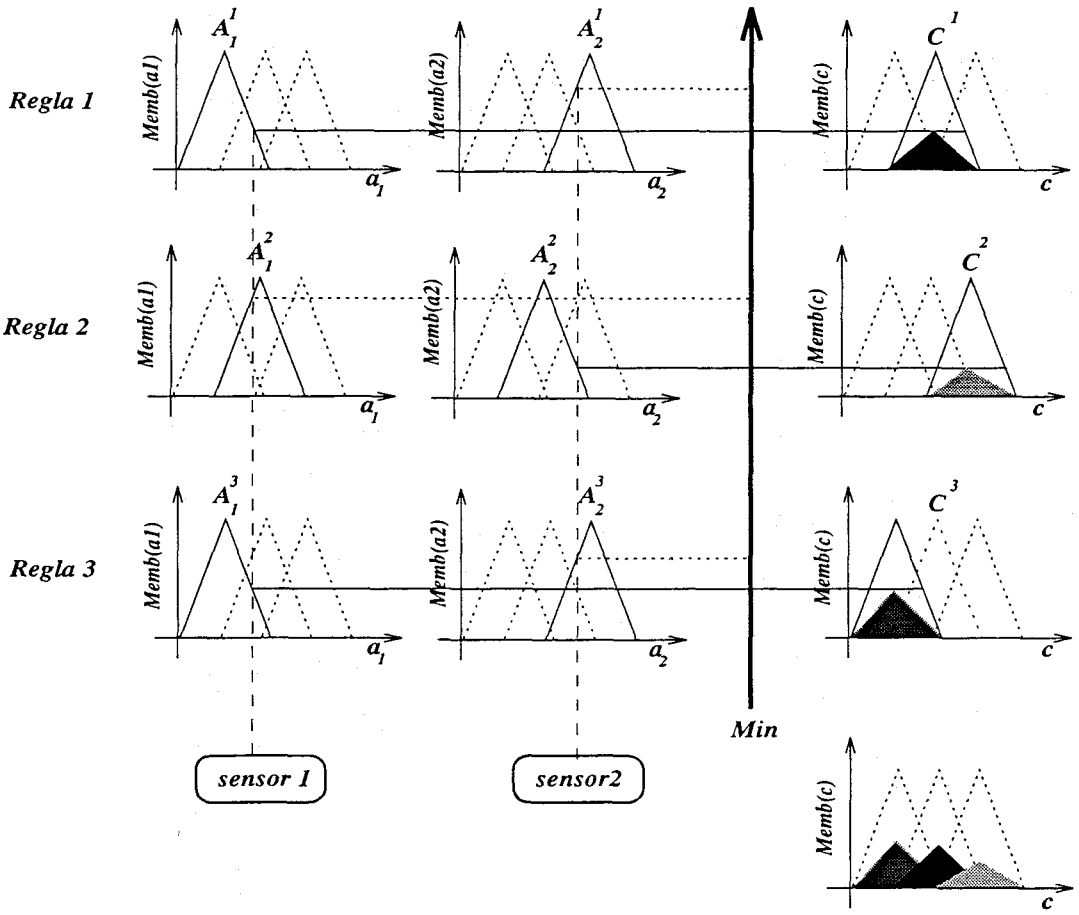


Figura 2.14: Mecanismo de razonamiento sup-pro.

Mecanismo de Inferencia basado en funciones monótonas. Similar a los dos anteriores, este método está basado en la implicación difusa definida como la composición de intersección dada en (2.14). Aquí $*$ es interpretado como mínimo. Su principal particularidad es su uso de funciones monótonas. Fue originalmente propuesto por Tzukamoto [Tzuka 79]. El proceso de inferencia encuentra para la consecuencia de la $i^{\text{ésima}}$ regla un valor escalar y_i, tal que $\alpha_i = C^i(y_i)$. La combinación de reglas es realizada mediante la media ponderada de las N conclusiones como:

$$c^* = \frac{\sum_{i=1}^{N} \alpha_i y_i}{\sum_{i=1}^{N} \alpha_i} \tag{2.27}$$

donde:

$$\alpha_i = \min(Memb_{A_1^i}(a_1), \cdots, Memb_{A_M^i}(a_M)) \tag{2.28}$$

El proceso de este mecanismo es ilustrado en la figura 2.15. A diferencia de los dos métodos anteriores, la obtención de un valor concreto no requiere de aplicar ningún método de defusificación.

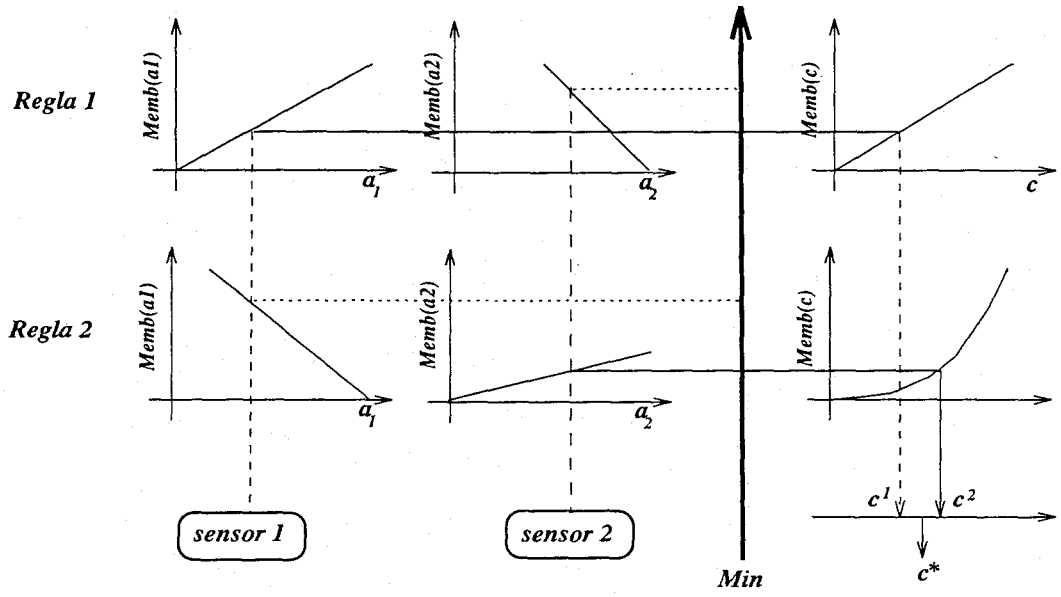


Figura 2.15: Mecanismo de razonamiento usando funciones monótonas.

Mecanismo de Inferencia de Sugeno. Igual que el método anterior, este método está basado en la implicación difusa definida como la composición de intersección dada en (2.14) e interpretando $*$ como mínimo. Aunque suele utilizar funciones monótonas (obligatoriamente en la variable de salida), se diferencia del método anterior en que el cálculo del valor escalar de la variable de control para cada regla es hecho por medio de una función lineal, es decir utilizando las reglas del tipo II descritas en el apartado anterior. Fue originalmente propuesto por Sugeno y Takagi [Takag 83]. La inferencia de la $i^{\text{ésima}}$ regla es dada por $\alpha_i f_i(a_1, \cdots, a_N)$, donde f_i es una combinación lineal de las variables de entrada $a_1, \cdots, a_N$. La combinación de las conclusiones obtenidas por las reglas es realizada mediante la media ponderada de cada una de ellas como:

$$c^* = \frac{\sum_{i=1}^{N} \alpha_i f_i(a_1, \cdots, a_N)}{\sum_{i=1}^{N} \alpha_i} \tag{2.29}$$

donde:

$$\alpha_i = \min(Memb_{A_1^i}(a_1), \cdots, Memb_{A_M^i}(a_M)) \tag{2.30}$$

A diferencia de los dos primeros métodos, y de forma análoga al anterior, la obtención de un valor concreto no requiere de aplicar métodos de defusificación.

En contraste con los métodos de inferencia mencionados anteriormente la tecnología presentada en esta tesis, propone uno nuevo llamado: *método de los cinco vecinos más cercanos (5NN)*[15]. Este nuevo método de inferencia es una combinación de inferencia difusa similar a la expuesta en este apartado, reforzada por la utilización de la comparación de los patrones formados por los antecedentes de las reglas registrados en el pasado. Una de las características más interesantes de ella es la agregación de todas las conclusiones disparadas en un conjunto difuso unitario. El capítulo 5 expone con amplitud esta forma de inferencia.

2.7.4 Interfaz de Defusificación

El proceso de defusificación es el proceso inverso de la fusificación, es decir, es el proceso de convertir un valor difuso (o una conclusión difusa) en información concreta expresada mediante un escalar. Hasta ahora ningún actuador es capaz de interpretar una señal difusa como entrada. Necesariamente las conclusiones difusas que el mecanismo de inferencia produce deben ser recodificadas a un valor escalar. Zadeh fue el primero en percatarse de este problema y sugirió algunas posibilidades para resolverlo [Zadeh 68]. Actualmente existe un buen número de métodos de defusificación, pero como ocurre con otros elementos del diseño de controladores difusos, no existe aún un procedimiento sistemático para seleccionar el más adecuado, dependiendo del caso de aplicación particular. Dentro de esta sección analizaremos los siguientes:

1. Criterio del Máximo (CM).
2. Media de las Alturas (MH).
3. Media de los Máximos (MOM).
4. Suma de los Centros de Área (SCOA).

[15] 'five nearest neighbors'

5. Centro de Área (Gravedad) (COA).

La selección del método de defusificación puede jugar un papel decisivo en la síntesis de modelos difusos para muchas áreas de aplicación. Particularmente dentro del área de control difuso, su influencia puede ser determinante en el comportamiento y la robustez del controlador. De hecho, la principal motivación del estudio del proceso de defusificación tiene su origen en las aplicaciones de la lógica difusa dentro del campo del control de procesos. Sin embargo, aunque el número de aplicaciones de los controladores difusos ha crecido rápidamente, no parece habérsele dado la importancia requerida. La calidad de las conclusiones obtenidas por el mecanismo de inferencia puede deteriorarse significativamente bajo una mala elección del método de defusificación. Algunos de los factores teóricos más relevantes de la defusificación son estudiados en [Yager 94]. La importancia del proceso de defusificación en los aspectos dinámicos del control difuso ha sido estudiado en [Jager 92]. La influencia de seleccionar intervalos difusos irregulares en el proceso de defusificación, y la influencia del número de reglas difusas en la convergencia son reportados en [Renho 91] y [Bousl 92]. En [Braae 78] se presenta un estudio comparativo entre algunos de los métodos más populares (COA y MOM), en [Schar 85] se compara el efecto de esos mismos métodos en el comportamiento dinámico de un brazo de robot con varios grados de libertad y en [Mugic 93] se da un método alternativo a ellos. De esos estudios puede desprenderse que, dependiendo de la aplicación particular, los parámetros que tienen más influencia en la interfaz de defusificación son el perfil de las funciones de pertenencia y el grado de solape entre los diferentes valores lingüísticos, mientras que el criterio de selección más importante es la complejidad computacional. Los métodos dados anteriormente han sido ordenados, y serán presentados, bajo este último criterio.

Asumamos que como resultado del proceso de inferencia fueron disparadas N reglas. Cada una de las $i^{\text{ésimas}}$ reglas ha producido una conclusión en la forma de un subconjunto difuso acotado por el *nivel de activación* α_i. Así que antes de entrar en la descripción de cada uno de los métodos enumerados anteriormente, consideremos la notación siguiente:

N = Número de reglas disparadas.

α_i = Nivel de activación de la $i^{\text{ésima}}$ conclusión (altura).

C^i = Conclusión difusa de la $i^{\text{ésima}}$ regla. Subconjunto del valor lingüístico CL_k en el dominio $\mathcal{C}$ de la variable difusa de salida c, tal que $Memb_{CL_k(c)} \leq \alpha_i$ para toda $c \in \mathcal{C}$.

K = Número de valores lingüísticos de la variable difusa de salida c.

$cmax_k$= Valor escalar correspondiente al máximo del valor lingüístico CL_k tal que $Memb_{CL_k}(cmax_k) = 1.0$.

$cmax^i$= Valor escalar correspondiente al máximo de la conclusión difusa C^i tal que $Memb_{C^i}(cmax^i) = \alpha_i$.

c^* = Valor escalar concreto defusificado de la variable de control.

Criterio del Máximo (CM). Este es el más simple de todos los métodos y el más económico desde el punto de vista computacional ya que considera únicamente la $i^{\text{ésima}}$ conclusión con la mayor α_i cuyo correspondiente $cmax_z$ se encuentra almacenado en la base de datos (sección 2.7.2.1). De este modo:

$$c^* = cmax_z, \quad \text{tal que} \quad \alpha_z = \sup_i \alpha_i \tag{2.31}$$

Si la función de pertenencia no tiene un único $cmax_z$ sino que más bien es un intervalo, como por ejemplo en una función trapezoidal, puede tomarse el valor medio del intervalo como: $cmax_z = (cmax_{z_{sup}} - cmax_{z_{inf}})/2$. La figura 2.16 ilustra este método.

La baja precisión de este método puede mejorarse aumentando el grado de granularidad (valor de K).

Media de las Alturas (MH). Este método es una generalización del anterior ya que en vez de considerar sólo el $cmax_k$ del valor lingüístico con el α mayor, se considera todos los $cmax_k$ correspondientes a las conclusiones de las N reglas disparadas (figura 2.17). La contribución de cada regla al valor de c^* es ponderada mediante la 'altura' (α_i) de la $i^{\text{ésima}}$ conclusión de la siguiente manera:

$$c^* = \frac{\sum_{i=1}^{N} cmax_i \cdot \alpha_i}{\sum_{i=1}^{N} \alpha_i} \tag{2.32}$$

Este es un método bastante mejor al anterior, sobre todo si se utilizan funciones simétricas, ya que sin aumentar mucho el costo computacional permite obtener valores de c^* mucho más plausibles.

Media de los Máximos (MOM). Este método es similar a los anteriores al estar basado en la 'estatura' de los α_is, pero en este caso en vez de

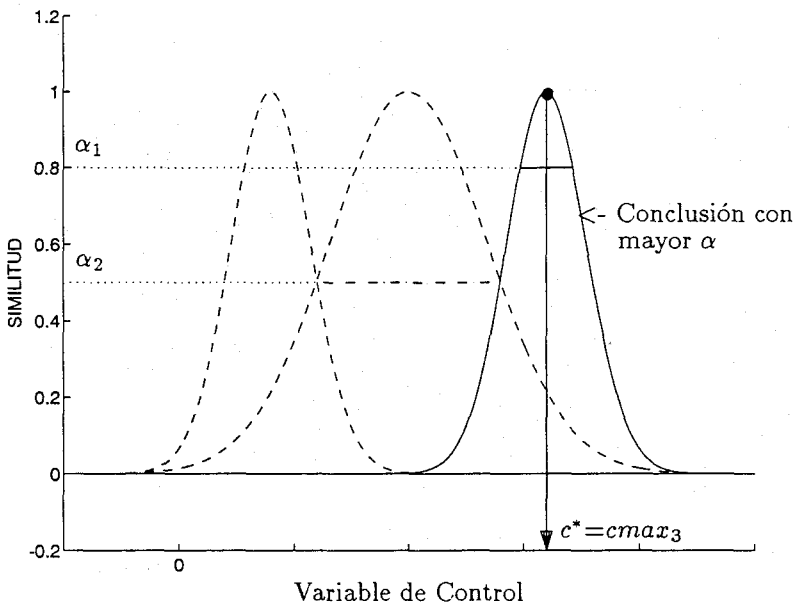


Figura 2.16: Interfaz de defusificación bajo el Criterio del Máximo.

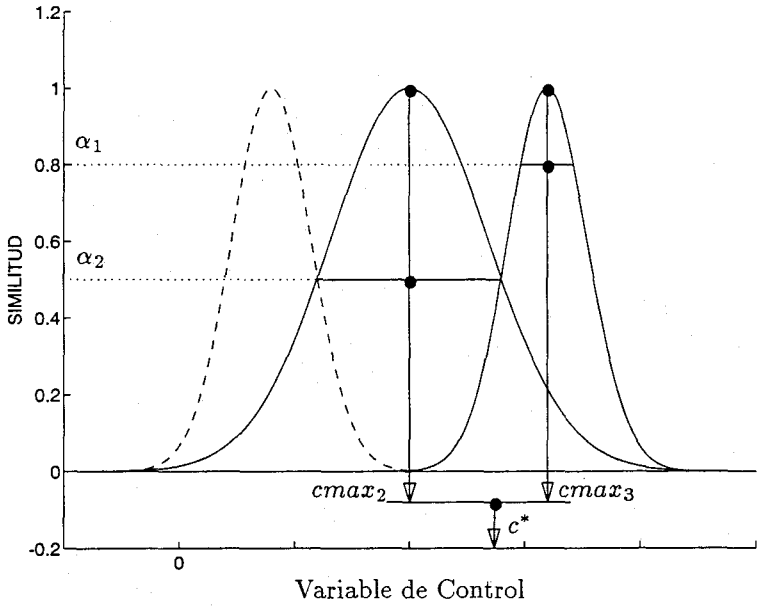


Figura 2.17: Algoritmo de defusificación basado en promedio ponderado de las alturas.

utilizar los $cmax_k$ correspondientes a los máximos de los valores lingüísticos, se consideran los $cmax^i$ correspondientes a los subconjuntos C^i cuyos valores máximos corresponden con los αs de las conclusiones de cada regla. El costo computacional aumenta por tener que calcular cada $cmax^i$ mediante el inverso de la función de pertenencia. Si el algoritmo está basado en la conclusión con α máximo, tenemos que:

$$c^* = cmax^z, \quad \text{tal que} \quad \alpha_z = \sup_i \alpha_i \tag{2.33}$$

Si se da el caso de que $cmax^z$ es único y la función de pertenencia es simétrica, este método puede ser reemplazado por el CM al ser completamente equivalentes. Sin embargo, suele darse el caso de que no lo sea. Por ejemplo, cuando se utiliza la inferencia sup-min y aplicando α_i al valor lingüístico, suele formarse una función semitrapezoidal (figura 2.18a). Para resolver esta complicación puede optarse por elegir el primer máximo (MOM_p), el último ($MOMu$), o la media entre ellos (MOM propiamente dicho). Nuevamente si se elige la media y la función de pertenencia es simétrica, este método puede ser reemplazado por el CM, que tiene un costo computacional menor.

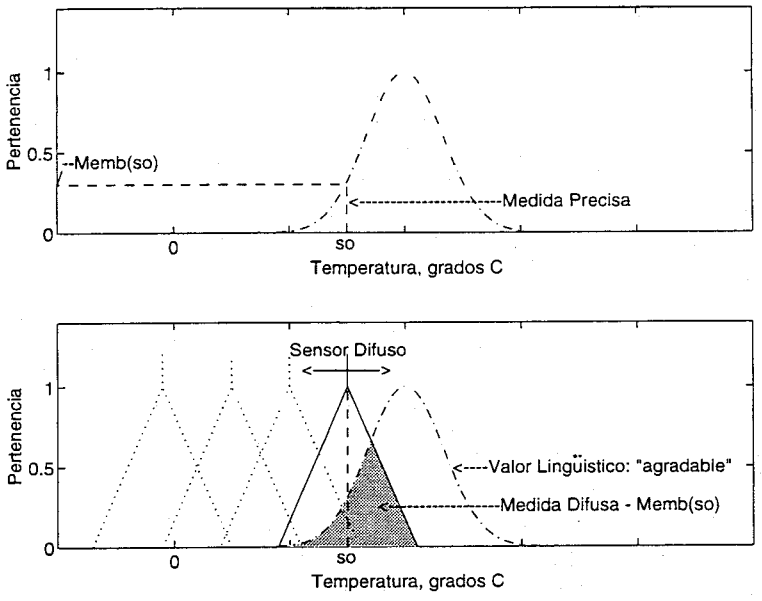


Figura 2.18: (a) $MOMpg$ con inferencia sup-min; y (b) $MOMpg$ con inferencia sup-prod.

Una complicación mayor surge si α_z no es única. Para resolverla, y además evitar la ignorancia de este método a las conclusiones no máximas, es posible hacer una generalización $MOMg$ similar a la de pasar de CM a MH. Para cada una de las N conclusiones obtenidas podemos evaluar un $cmax^z$ como antes, pero sin tomar en cuenta la restricción de que $\alpha_z = \sup_i \alpha_i$. Entonces, podemos encontrar un c^* como:

$$c^* = \sum_{i=1}^{N} \frac{cmax^i}{N} \tag{2.34}$$

Las figuras 2.18a y 2.18b muestran $MOMg$ para dos tipos de inferencias.

Suma de los Centros de Área (SCOA). Ninguno de los métodos considerados hasta ahora toma en cuenta el perfil de las funciones de pertenencia. El método de Suma de los Centros de Área permite añadir esta información. Por supuesto, es inevitable que el costo computacional aumente de manera notoria. La idea de este método es considerar individualmente el centro de gravedad (área cuando es una sola salida) del conjunto difuso obtenido como conclusión de cada una de las reglas disparadas y, posteriormente, efectuar la media ponderada del conjunto total. Esto puede ser calculado como:

$$c^* = \frac{\int_{\mathcal{C}} c \cdot \sum_{i=1}^{N} Memb_{C^i}(c)dc}{\int_{\mathcal{C}} \sum_{i=1}^{N} Memb_{C^i}(c)dc} \tag{2.35}$$

si la función Memb(c) es continua, y mediante:

$$c^* = \frac{\sum_{j=1}^{J} c_j \cdot \sum_{i=1}^{N} Memb_{C^i}(c_j)}{\sum_{j=1}^{J} \cdot \sum_{i=1}^{N} Memb_{C^i}(c_j)} \tag{2.36}$$

si la función es discreta con $\mathcal{C}= \{c_1, c_2, \cdots, c_J\}$.

Como puede apreciarse en la figura 2.19, el área de superposición entre una conclusión y otra es tomada en cuenta dos veces lo que puede producir efectos indeseables, dependiendo del caso de que se trate. Puede verse también que dependiendo del perfil de la función de pertenencia con el que se trabaja el cálculo de las ecuaciones (2.35) o (2.36) puede resultar desde muy simple a muy complejo. Independientemente de la forma de la ecuación (2.35), una gran ventaja de este método es la capacidad para realizar los cálculos de manera

modular, es decir, calcular la contribución de cada conclusión por separado y hacer una operación de media ponderada al final.

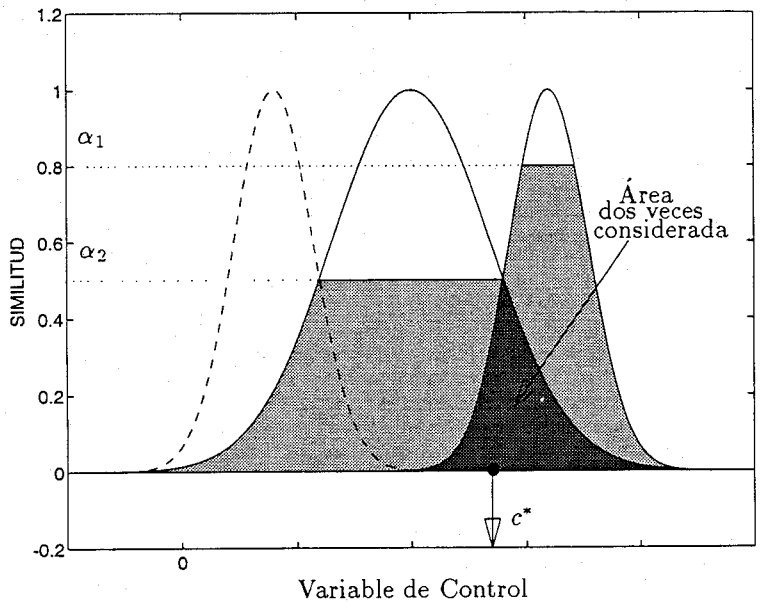


Figura 2.19: Representación gráfica del método: Suma de los Centros de Área.

Una variante más económica de este método, es utilizar el valor de los αs en lugar de las áreas para ponderar los centros de las áreas. Si asumimos que β_i es el centro de área para la $i^{\text{ésima}}$ conclusión, c^* puede calcularse como:

$$c^* = \frac{\sum_{i=1}^{N} \alpha_i \cdot \beta_i}{\sum_{i=1}^{N} \alpha_i} \tag{2.37}$$

Centro de Área (COA). Para corregir el efecto de superposición de las areas del método SCOA, simplemente es necesario tomarlas en cuenta sólo una vez. Esto se hace considerando únicamente el perfil de la función de pertenencia que resulte mayor en cada punto (figura 2.20), es decir:

$$c^* = \frac{\int_C c \cdot \max_i Memb_{C^i}(c) dc}{\int_C \max_i Memb_{C^i}(c) dc} \tag{2.38}$$

si la función Memb(c) es continua, y mediante:

$$c^* = \frac{\sum_{j=1}^{J} c_j \cdot \max_i Memb_{C^i}(c_j)}{\sum_{j=1}^{J} \cdot \max_i Memb_{C^i}(c_j)} \tag{2.39}$$

si es discreta.

Aunque las expresiones matemáticas de COA y SCOA pueden parecer muy similares, el costo computacional del COA es bastante superior ya que requiere calcular o bien los puntos de intersección entre cada una de las funciones que intersectan, o bien, algo aún más costoso, evaluar todas las funciones punto a punto y seleccionar el de mayor valor en cada caso. La posibilidad de modularizar (y quizás paralelizar) los cálculos que permitía el método SCOA no es posible aquí. El COA es considerado como el más preciso, pero más costoso, algoritmo de defusificación. Este método es conocido como *centro de gravedad* en el caso multidimensional, es decir, cuando debe generarse más de una señal de control.

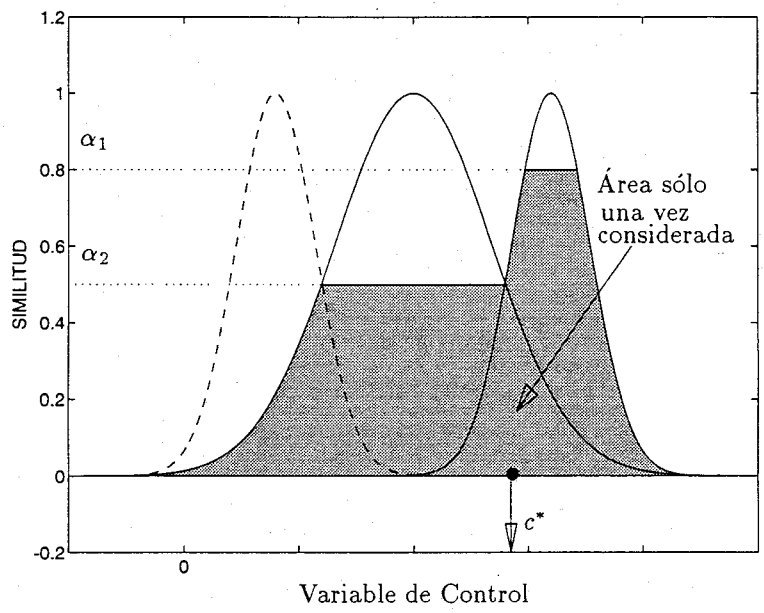


Figura 2.20: Ilustración del método de defusificación Centro de Área.

La interfaz de defusificación utilizada en la metodología presentada en esta tesis es muy simple en comparación con las mencionadas anteriormente. Dado que el mecanismo de inferencia ha producido un conjunto difuso unitario, la conversión a un valor escalar para encontrar una señal de control concreta es justo la función inversa de la operación aplicada en la interfaz de fusificación.

Capítulo 3

Métodos Cualitativos de Modelado y Simulación

3.1 Introducción

Cuando nuestra investigación comenzó, la motivación principal para ella consistía en proponer un esquema de solución general, que después se particularizó al diseño de controladores, al problema de cómo construir modelos continuos en el tiempo de sistemas, o sistemas que incorporan subsistemas, que no pueden ser descritos mediante modelos matemáticos precisos (cuantitativos), debido a su complejidad estructural, a la manera cualitativa en la que están expresados o a que el conocimiento que se tiene de ellos es impreciso o incompleto. La capacidad inherente de los seres humanos para tratar con este tipo de sistemas de una manera natural, sugería que la búsqueda de la solución debería considerar la evaluación de las diferentes técnicas para caracterizar y construir sistemas inteligentes que permiten emular la forma en que los humanos razonamos con respecto a los sistemas físicos.

Nuestra primera hipótesis fue que el diseño de controladores para sistemas altamente no lineales, variantes en el tiempo, o parcialmente desconocidos podría ser realizado mediante técnicas de inteligencia artificial. En el pasado, este tipo de sistemas han sido controlados principalmente por operadores humanos en lugar de usar controladores automáticos, ya que los operadores humanos pueden tratar con comportamientos inesperados o parcialmente desconocidos de una planta de una forma mucho más segura que cualquiera de los actuales controladores automáticos, usando razonamiento cualitativo y sentido común. La investigación en inteligencia artificial ha trabajado

durante años en cómo representar el conocimiento cualitativo en computadoras y en el desarrollo de algoritmos para procesar este conocimiento en forma similar al que realiza el razonamiento humano, generando varios paradigmas que podrían ser aplicables al campo del control automático. Otras áreas del conocimiento, como la Simulación y la Teoría General de Sistemas, también han desarrollado metodologías que son potencialmente útiles para resolver la tarea de modelar y reproducir el conocimiento cualitativo involucrado en el control de dispositivos físicos. Como veremos más adelante, todas las técnicas de modelado y simulación cualitativos son relativamente nuevas y no es fácil conocer *a priori* qué técnica es la más adecuada para cada caso. Ya que, además, la metodología seleccionada como herramienta metodológica es prácticamente desconocida para la comunidad tanto de control como de IA, consideramos que la evaluación y selección de la metodología, que es un tópico de interés científico en sí mismo, puede ser mejor comprendida si se presenta bajo el panorama de las alternativas posibles.

En el primer capítulo de esta disertación, se han mencionado las aproximaciones de IA que más frecuentemente se han intentado aplicar a la teoría de control, resaltando la necesidad de combinar en una estrategia conjunta y cooperativa los campos de la IA y de la teoría de control. En el capítulo segundo se ha mostrado cómo la teoría de conjuntos difusos y la teoría de razonamiento aproximado son un esquema potencialmente adecuado para la representación y procesamiento del conocimiento en el área del control automático y se anticiparon algunas de las características de la metodología particular de controladores difusos que se proponen en esta tesis. Sin embargo, hasta ahora no ha sido explicada la razón por la cual el esquema de control difuso fue seleccionado, qué otros esquemas de solución fueron evaluados ni cuáles son sus características particulares.

En el presente capítulo junto con los siguientes dos, intentaremos responder a estas interrogantes. Ya que la elección de la metodología fue una tarea que requirió un considerable esfuerzo de estudio y evaluación, en este capítulo hemos querido presentar en forma breve el estudio de las diferentes aproximaciones que existen, resaltando sus características principales con el objetivo de justificar la elección del Razonamiento Inductivo Difuso como la herramienta cualitativa de modelado y simulación utilizada para el diseño de controladores. En el capítulo cuatro se hará una descripción detallada de la metodología del razonamiento inductivo difuso. En el capítulo cinco abordaremos el problema de simulación mixta en la que coexisten modelos cuantitativos y modelos cualitativos y mostraremos la capacidad del razonamiento inductivo difuso para tratar este tipo de sistemas. Este último tema es de suma importancia en el campo de control debido al fuerte

acoplamiento que existe entre el controlador a diseñar y la planta que controla, imponiendo un importante requisito a ser satisfecho por las metodologías evaluadas.

3.2 Selección de la Metodología

Inherente a la empresa de dotar de inteligencia a los sistemas está la necesidad de resolver el problema de cómo *representar*, *adquirir* y *procesar* el conocimiento. Estos tres aspectos, piedras angulares de la inteligencia artificial, son fundamentales también para el desarrollo del control inteligente. Cuando se habla de sistemas expertos, redes neuronales, sistemas difusos, modelos cualitativos, modelos cuantitativos, modelos basados en patrones, etc. normalmente se hace referencia más a los esquemas de representación y al procesamiento del conocimiento que a la tarea de adquisición del conocimiento. Casi siempre se parte de la suposición de que el conocimiento es asequible y completamente representable en el esquema elegido. Sin embargo, la realidad es que la etapa de adquisición del conocimiento permanece aún como uno de los actuales *cuellos de botella* no sólo en el diseño de controladores inteligentes sino que es una problemática compartida también con el diseño de los sistemas expertos y en general con el desarrollo de cualquiera de los sistemas computacionales denominados 'inteligentes'. La consecuencia más importante de esta problemática se traduce en una seria dificultad para la sistematización del diseño y cada controlador o sistema inteligente debe ser construido de manera *ad hoc* teniendo en mente el esquema de representación y procesamiento previamente elegido.

El proceso de adquisición de conocimiento respecto de un sistema puede ser visto como el proceso de la construcción del *modelo* que lo representa. A la actividad de experimentar con los modelos bajo las condiciones particulares que se desean controlar o estudiar se le conoce con el nombre de *simulación*. Existe un extenso número de metodologías para la construcción de modelos las cuales varían dependiendo de la naturaleza del conocimiento que se desea representar así como del contexto y área donde se pretenden aplicar. Limitando el campo de estudio a los sistemas físicos y particularmente a aquellos en los que la coordenada sobre la cuál se desarrolla la simulación es el tiempo, podemos establecer dos grupos: los *modelos cualitativos* y los *modelos cuantitativos*. La diferencia entre unos y otros estriba en el tipo de valores que adquieren las variables de estado que describen el modelo. En los modelos cuantitativos los valores de las variables de estado son magnitudes continuas en el dominio de los números reales, mientras que

en los modelos cualitativos los valores son magnitudes discretas entre un número reducido de clases o categorías las cuales pueden ser representadas tanto de manera simbólica como numéricamente [Celli 91a]. Los métodos de modelado cuantitativo y de simulación cuantitativa han alcanzado un alto grado de madurez: tienen un esquema de aplicación perfectamente definidos, gozan de una terminología común y pueden ser perfectamente clasificados según su interpretación de la variable *tiempo* como: continuos, discretos o de eventos discretos (o una combinación de ellos). En contraste, las técnicas de modelado y simulación cualitativas se encuentran aún en sus primeras etapas, experimentando continuos cambios y surgiendo continuamente nuevas aproximaciones, cada una con una terminología diferente por lo que su utilización requiere normalmente una etapa previa de análisis para determinar cuál de ellas es la adecuada en cada caso. La clasificación de los métodos cualitativos es un tanto más compleja y menos definida que la de los sistemas cuantitativos. Atendiendo al requisito de la sistematización y en particular a la tarea de adquisición del conocimiento, una posible clasificación podría ser en función de qué es lo que conocemos del sistema, o qué información queremos utilizar como *fuente de conocimiento*. Podemos definir dos grandes familias:

- *Modelos basados en la estructura.* En este tipo de modelos el conocimiento respecto del sistema en estudio se encuentra bien organizado y suele representarse mediante las relaciones que existen entre las variables que le caracterizan. Las relaciones pueden estar expresadas ya sea en forma verbal (lingüística) o matemáticamente. En el campo de la identificación de sistemas estos modelos son conocidos como modelos de *caja gris*. Desde la perspectiva del campo de la inteligencia artificial, se engloban en esta categoría los sistemas expertos y el conjunto de métodos denotados bajo el término *física cualitativa*. Mientras que los primeros suelen utilizar formas lingüísticas para capturar la experiencia de los operadores expertos, los segundos pueden verse como una abstracción de las ecuaciones diferenciales utilizadas por los métodos cuantitativos. La simulación de sistemas ha intentado algunas aproximaciones al campo cualitativo mediante las técnicas conocidas como: modelos simbólicos de eventos discretos. El o los posibles comportamientos que puede alcanzar un sistema descrito mediante este tipo de modelos se *deducen* a partir de la información de la estructura del sistema que se ha plasmado en el modelo.

- *Modelos basados en el comportamiento.* Cuando se desconoce la estructura del sistema, se sabe muy poco de ella, o simplemente la representación analítica o cualitativa basada en la estructura no es la

adecuada para el objetivo que se busca, la obtención del modelo puede basarse en el comportamiento del sistema más que en su estructura, si se está en la posibilidad de observar y medir la operación del sistema. Estos modelos son conocidos como modelos de *caja negra*. Bajo estas circunstancias el conocimiento del sistema suele encontrarse muy poco estructurado, contándose tan sólo con el registro histórico de los estados que el sistema ha ido adquiriendo en el tiempo. En este caso es necesario razonar *inductivamente* en base a la información registrada para construir un modelo de la estructura del sistema, a partir del cuál se pueden entonces inferir los posibles comportamientos.

3.3 Modelos Basados en la Estructura

3.3.1 Física Cualitativa

Los métodos de la física cualitativa tienen que ver con la representación del mundo físico en términos cualitativos y con el razonamiento respecto de su comportamiento. Sus objetivos son capturar tanto el conocimiento contenido en el sentido común de cualquier persona común y corriente, como el conocimiento tácito que subyace en el conocimiento cuantitativo utilizado por los ingenieros y científicos [Forbu 84]. El gran número de publicaciones y el interés mostrado por la comunidad científica por el formalismo de la física cualitativa le llevaron a constituirse como una rama independiente dentro de la inteligencia artificial. Su gran difusión y sus capacidades potenciales de sistematización, atrajo nuestra atención para considerarlo como la posible herramienta de modelado y simulación cualitativas que requeríamos.

La primera referencia al razonamiento cualitativo con que contamos se remonta a 1977 cuando de Kleer [deKle 77] desarrolla el sistema NEWTON en el que presenta su teoría de 'envisioning'. En NEWTON el comportamiento cualitativo de un mecanismo es representado mediante el grafo de las transiciones entre los diferentes estados cualitativos posibles. Poco después, Hayes publica su manifiesto de la física 'naive' [Hayes 79] en donde, por medio de lógica de primer orden, formaliza gran parte del conocimiento común y corriente que se tiene del mundo físico y axiomatiza la heurística (metainformación) asociada, con el objetivo de inferir los posibles comportamientos de un sistema a partir del conocimiento del mundo físico formalizado. La vida de cada objeto es definida mediante fragmentos espacio–temporales llamados *historias* en las que se registran los diferentes

comportamientos *episódicos* posibles. Después de la publicación del manifiesto de la física 'naive' se desata un frenético interés por el desarrollo de métodos basados en física cualitativa. Las principales aproximaciones que existen se conocen bajo los nombres de: razonamiento cualitativo [Forbu 81], simulación cualitativa [Kuipe 82,84,86], razonamiento causal [deKle 82], teoría de procesos cualitativos [Forbu 84], física *naive* [Hayes 78,85a], física cualitativa [deKle 84], razonamiento con sentido común [McCar 58, 90], [Carbo 85] y modelado de sistemas a partir de conocimiento de nivel profundo [Pan 84]. Cabe aclarar aquí, que existen otras formulaciones de modelado cualitativo hechas desde campos como la biología [Pucci 85], [Levin 74], la economía [Klee 81] o las ciencias sociales en general [Blalo 85], pero dado que nuestro estudio se limitó a sistemas físicos, estas últimas no fueron consideradas.

Una importante característica con respecto a estos métodos es que aunque cada uno de ellos utiliza una terminología diferente, básicamente todos tratan los mismos aspectos. Esto es confuso e innecesario y obliga a invertir un gran esfuerzo por parte de quienes intentan evaluar sus cualidades para adaptarlos a sus respectivos campos de aplicación. Nosotros creemos, que este hecho es en parte responsable del lento progreso que estos paradigmas han mostrado en el pasado, principalmente en lo que se refiere a sus aplicaciones. Existen varias compilaciones y revisiones de tales métodos, entre los que podemos destacar: 'Formal Theories of the Commonsense World' editado por Hobbs y Moore [Hobbs 85], 'Readings in Qualitative Reasoning about Physical Systems', editado por Weld y de Kleer [Weld 90], [Bonis 85] y el número especial dedicado al tema del 'Artificial Intelligence' Vol. 24 [Bobro 84]. Una interesante autocrítica y reorientación del área puede encontrarse en otro número especial de la misma revista, Vol. 51 [Bobro 91]. Un estudio de sus capacidades reales, así como la comparación con otros métodos de simulación cuantitativa se reporta en [Celli 91a]. La discusión de algunos de los conceptos claves que conectan la física cualitativa con la perspectiva de la teoría de sistemas y de la simulación se da en [Fishw 90] donde se propone una reorientación de la investigación en conjunto con la teoría de modelado de sistemas.

Nuestro estudio se concentró en las tres aproximaciones que pueden ser consideradas como las más representativas: i) la teoría de procesos cualitativos de Forbus, basada en el concepto de proceso; ii) la teoría de 'envisioning' desarrollada por de Kleer y Brown, basada en el modelado de *dispositivos* independientes interconectados; y iii) la teoría de modelado cualitativo de Kuipers, basada en ecuaciones diferenciales cualitativas vistas como *restricciones*. En el apéndice A se presenta una breve descripción de los principales elementos de estas metodologías, resaltando los aspectos de *modelado*, *simulación* y *explicación*.

Lamentablemente, no fue posible adaptar ninguna de estas metodologías a los requerimientos de los sistemas de control, al menos al nivel del desarrollo de los controladores. Las principales dificultades encontradas fueron:

- Resulta prácticamente imposible evitar la explosión de las soluciones cualitativas factibles.
- Frecuentemente algunas de esas soluciones no tienen ningún sentido desde el punto de vista físico.
- La naturaleza fuertemente simbólica de los modelos cualitativos que resultan con estas metodologías no permite representar la variable independiente del *tiempo* de manera cuantitativa. No es claro cómo tales modelos puedan ser acoplados con subsistemas modelados de manera cuantitativa.
- Aunque son capaces de capturar incertidumbre e informalidad respecto a la interrelación específica de las variables del sistema, requieren un conocimiento profundo y explícito de la estructura interna del sistema a ser modelado. Este conocimiento puede estar o no disponible.
- Solamente sistemas con estructuras muy simples han podido ser representados con éxito mediante alguno de estos paradigmas. La complejidad de este tipo de métodos no es escalable.
- La existencia de retroalimentación entre las variables de los subsistemas fácilmente deriva en un comportamiento intratable. Únicamente pueden considerarse sistemas con retroalimentación cuando no son dominantes del comportamiento del sistema. Esta característica se encuentra en contradicción directa con los más elementales principios de la teoría de control.

Debe de reconocerse que aunque el paradigma de física cualitativa carece aún de la capacidad de resultar útil para resolver problemas prácticos en el diseño de controladores, está excelentemente bien situado para emular procesos de razonamiento humano de alto nivel. Aunque no parece posible que este tipo de formulaciones permitan por sí mismas alcanzar el nivel de precisión requerido por los controladores, el resultado de nuestro análisis indica que este conjunto de metodologías puede tener una gran efectividad si se aplican en los niveles jerárquicos superiores del control inteligente, como la *planificación*, la *organización* y la *coordinación*.

3.3.2 Sistemas Expertos

Los sistemas expertos, o sistemas basados en reglas, son, quizás, uno de los productos más maduros y exitosos de la inteligencia artificial en cuanto a aplicaciones se refiere y tienen una efectividad probada bajo ciertas condiciones particulares y cuando se cumplen determinados requerimientos. Algunas de sus principales características son su estructura simbólica y la capacidad que tienen para emular ciertas operaciones de la inteligencia humana como el diagnóstico, la explicación y la supervisión. Son particularmente potentes para modelar sistemas en donde el conocimiento de un experto está suficientemente estructurado y se encuentra disponible en la forma requerida por las estructuras simbólicas. Dentro del área de control, se han realizado algunos estudios para determinar el potencial de su utilización. Por ejemplo el sistema de control experto de Åström [Astr 86] propone una arquitectura flexible para combinar algoritmos en tiempo real con lógica para llevar a cabo tareas de supervisión y coordinación. Las arquitecturas de pizarra han demostrado ser una buena solución para instrumentar sistemas expertos supervisores de controladores [Arzen 89]. Un interesante estudio de diferentes arquitecturas de control experto viene en [Salle 89].

Aunque cierta parte de la estructura de los controladores difusos, presentados en el capítulo anterior, parece guardar la misma estructura general de un sistema experto, existen algunos aspectos fundamentales que los diferencian y que deben de ser resaltados para evitar confusiones. El primero, y quizás el más importante, es el hecho de que la información almacenada en la base de conocimiento no guarda la forma simbólica de los sistemas expertos clásicos, cuyas reglas suelen estar expresadas mediante predicados. De lo anterior se desprende otra importante diferencia. Mientras que los sistemas expertos utilizan lógica de primer orden en el procesamiento de la información, la máquina de inferencia de los sistemas difusos requiere utilizar razonamiento aproximado y operaciones de lógica difusa. La idea de encadenamiento de reglas *hacia atrás* y *hacia adelante* tan relevante en los sistemas expertos, no tiene ningún sentido en la estructura de los controladores difusos.

La utilización de los sistemas expertos en control tiene algunos problemas y está limitada a ciertas tareas. Los principales son:

- La prueba y validación de los sistemas basados en reglas suelen ser tareas arduas y complejas.
- Aunque los sistemas expertos cuentan con algunos modelos de razonamiento temporal que permiten representar cualitativamente a la

variable del tiempo, el tratamiento del tiempo no suele incluirse de manera explícita en los modelos de control descritos mediante sistemas expertos. Lo que se suele hacer es utilizar controladores lógicos convencionales para actualizar la información en la pizarra (o en la base de hechos) para disparar las reglas de la base de conocimiento del sistema experto. Las conclusiones alcanzadas, mediante la máquina de inferencia, modifican el estado de las variables lógicas de los controladores por vía de la pizarra (o la base de hechos). Una lógica de secuenciación permite que las acciones sean tomadas en el tiempo real. El hecho de utilizar de manera implícita el tiempo implica que no es posible establecer ningún tipo de estructura de razonamiento causal–temporal.

- La construcción de las reglas continua siendo una fuerte limitación ya que o bien se realiza de manera *ad hoc* o se reduce a una estructura simple y genérica que resulta útil solamente en casos pequeños donde la utilización de este tipo de sistemas no se justifica.

De manera similar al paradigma de física cualitativa, la utilización de este tipo de sistemas está más bien limitada a las tareas de supervisión, coordinación y a la interrelación con los operadores humanos, es decir, tareas que emulan procesos de razonamiento superior. La diferencia entre estas dos aproximaciones estriba en que, mientras que los sistemas expertos están basados en *conocimiento superficial*, expresando en la heurística del experto, los sistemas de la física cualitativa se basan en la utilización del *conocimiento de nivel profundo* de la estructura interna del sistema. De hecho, una de las principales motivaciones para el desarrollo de los sistemas de la física cualitativa fue dar una alternativa al esquema de sistemas expertos que presentan una notoria falta de sentido común causada por la falta de conocimiento de la estructura interna del sistema representado. Mientras que los problemas metodológicos y de sistematización han consumido la mayor parte de los esfuerzos de la física cualitativa, reportando únicamente aplicaciones de sistemas elementales, los sistemas expertos se han concentrado más en las aplicaciones, atendiendo principalmente los problemas particulares de cada caso mediante la heurística del experto, razón por la cual, han obtenido soluciones a un buen número de aplicaciones complejas, incluso dentro del área de control. Desde la perspectiva del control, ninguno de los dos paradigmas parece ser muy adecuado para el diseño de los controladores de bajo nivel desde donde se accionan los actuadores. Ninguno de los dos ofrece el nivel de precisión requerido. Ninguno puede realizar la tarea de control propiamente dicha.

3.3.3 Modelos Simbólicos de Eventos Discretos

No sólo la IA ofrece alternativas para el modelado y simulación cualitativos. El campo de la simulación de sistemas nos provee con el paradigma de modelos de eventos discretos permitiendos representar tanto modelos cuantitativos como modelos cualitativos [Chi 92]. Los modelos simbólicos de eventos discretos difieren sustancialmente de las aproximaciones mencionadas hasta ahora. Una gran diferencia es el tratamiento del tiempo. Mientras que las aproximaciones de razonamiento cualitativo requieren que el tiempo sea utilizado explícitamente y modelado en forma completamente cualitativa, la metodología de modelos simbólicos de eventos discretos modela el tiempo de manera cuantitativa y en forma intrínseca como en el caso del modelado y simulación cuantitativos. Esto significa que tanto el transcurso del tiempo como la modificación de estados del sistema pueden actuar como los agentes que generan la transición entre los estados del sistema. Una descripción detallada de esta metodología puede encontrarse en [Zeigl 89] y en [Zeigl 91].

El concepto central de los modelos simbólicos de eventos discretos es el *evento*. Los cambios en los valores de las variables se realizan de manera *instantánea*. Los intervalos de tiempo que transcurren entre uno y otro evento son variables pero son representados cuantitativamente. Los tiempos para los eventos son formulados como polinomios del tiempo con coeficientes desconocidos o parcialmente conocidos (posiblemente difusos). La base del proceso de simulación es la generación de eventos. Un calendario de eventos contiene los instantes de tiempo cuando el sistema debe transitar de un estado cualitativo a otro. Sin embargo, como los estados cualitativos pueden ser difusos, la especificación de la transición de estados será poco precisa y por lo tanto la definición de los tiempos para los eventos también. La ambigüedad puede surgir, ya que no hay una secuencia de eventos única. Debido a la imprecisión de los tiempos para los eventos puede ocurrir que un evento sucede a otro y viceversa. Para superar la ambigüedad y la intratabilidad de un espacio de trayectorias muy grande, se han intentado la introducción de restricciones causales adicionales [Oleye 89], pero esta solución no ha resultado del todo exitosa.

La simulación simbólica de eventos discretos permite generar todas las trayectorias que son factibles utilizando sus coeficientes difusos, lo cual nos permite predecir todos los posibles comportamientos a partir de un estado. El concepto básico para modelar la imprecisión en la que ocurren los eventos es la *ventana de tiempo* [Wang 91]. La información en una ventana de tiempo puede ser usada subsecuentemente para razonar causalmente con el propósito de diagnóstico de fallos o como un generador de explicaciones.

Nuevamente, la falta de precisión en el modelado del tiempo para definir las secuencias de eventos nos permite realizar tareas de control de alto nivel, principalmente el diagnóstico, pero no nos permite utilizar esta tecnología para el diseño de controladores de bajo nivel.

3.4 Modelos Basados en el Comportamiento

En el apartado anterior se ha mostrado cómo los modelos basados en la estructura no nos han permitido cumplir con los requerimientos para diseñar controladores de manera sistemática. Por un lado, el uso de conocimiento superficial complica la tarea de sistematización, por el otro el uso de conocimiento profundo, mucho más sistematizable, complica en tal grado la tarea de adquisición del conocimiento de la estructura, que se vuelve imposible, por lo menos hasta ahora, capturar estructuras de la complejidad requerida de los sistemas de control. Además de lo anterior debe ser resuelto el problema del manejo del tiempo (o de los eventos) y la exploración de las posibles trayectorias. Siendo poderosas herramientas para representar el conocimiento estructural y heurístico de un sistema, no lo son para hacerlo con el conocimiento temporal y por lo tanto su capacidad de capturar la dinámica de manera precisa es, por lo menos hasta ahora, muy limitada. En cierto sentido puede afirmarse que son capaces de capturar el *conocimiento* superior, el por qué y el cómo, pero no la *habilidad* y el cuándo. El conocimiento teórico involucrado en el lanzamiento de una pelota al aire para luego atraparla antes de caer al suelo es aparentemente fácil de describir y predecir en términos de las leyes causales subyacentes, pero desde el punto de vista de la precisión resulta una tarea extraordinariamente compleja para un robot. Sin embargo, para la habilidad de cualquier niño de cuatro años resulta una tarea simple que no requiere de ningún conocimiento de las leyes físicas involucradas, incluso bajo circunstancias más complejas como por ejemplo la presencia de corrientes de aire. La habilidad humana para tomar acciones de control en tiempo real, como el caso del lanzamiento de la pelota al aire, no ha sido explicada aún por ninguna teoría; sin embargo, parece responder más a un reconocimiento de patrones de comportamiento aprendidos previamente, que a procesos de razonamiento complejos y estructurados. Por ese motivo los modelos cualitativos basados en patrones podrían responder de mejor manera a los requisitos de diseño de controladores al capturar el comportamiento más bien que la estructura de un sistema.

De manera inversa a los métodos basados en la estructura, en los que dado un estado del sistema y un modelo de la estructura, se intentan predecir

su, o sus posibles comportamientos, los métodos basados directamente en los comportamientos aprendidos deben de ser capaces de construir (inducir) un modelo que represente al sistema a partir de los comportamientos observados. El problema de falta de precisión de la que adolecen los métodos basados en estructura puede ser resuelto en base a la información medida de las variables de entrada y de salida del sistema. La calidad de los modelos generados por las diferentes metodologías basadas en comportamiento que existen, dependerá principalmente de dos factores: el primero, que hasta cierto punto es común y externo a las metodologías, es la *riqueza* de la información contenida en los patrones de comportamiento de los que se parte para construir el modelo; el segundo es la capacidad que cada metodología tiene para interpretar, clasificar y/o generalizar los patrones de comportamiento observados para inferir y modelar el comportamiento general del sistema que permita predecir de manera precisa comportamientos futuros. Básicamente hay dos tipos de metodologías: las inductivas y las de identificación de parámetros. Ambas metodologías parten de los datos *crudos* medidos y utilizando algoritmos de clasificación, de correlación y/o de interrelación causal *descubren* mediante un proceso inductivo las relaciones causales y temporales del sistema y a partir de ellas son capaces de proponer un modelo. En contraste, las metodologías de identificación de parámetros proponen un esquema funcional genérico (modelo matemático) con parámetros desconocidos que son *identificados* mediante algoritmos de optimización, en base a la calidad con que el conjunto de parámetros representa el comportamiento que se desea aprender.

3.4.1 Metodologías Inductivas

Una importante característica de aquello que denominamos comportamiento inteligente es la capacidad de adquirir conocimientos generales a partir de hechos específicos. Formular teorías, inferir reglas, descubrir patrones y construir modelos que permitan simular y explicar un sistema a partir de las medidas que obtenemos de él, es una de las tareas más importantes en cualquier campo científico. Denominamos razonamiento inductivo, o simplemente inducción, al proceso de obtener hipótesis generales a partir de los datos de un conjunto de hechos. El problema es extraordinariamente complejo cuando se quiere automatizar, ya que el número de posibles conclusiones inductivas que se pueden obtener a partir del mismo conjunto de datos suele ser muy grande. Existen diferentes paradigmas de razonamiento inductivo, los cuales pueden clasificarse en dos vertientes: los que provienen de la IA y los que provienen de la Teoría General de Sistemas (TGS).

3.4.1.1 Metodologías Inductivas – IA

Las técnicas más interesantes desde el punto de vista de la IA son aquellas desarrolladas dentro del paradigma general conocido como: "Machine Learning" [Carbo 89], [Micha 83]. Lamentablemente, para nuestros propósitos, prácticamente todos los desarrollos han sido enfocados a la adquisición de *conceptos* utilizando para ello esquemas de representación simbólicos. La estructura general de estos sistemas presenta dos problemas principales: primero, dada su estructura *fuertemente simbólica*, su capacidad de adaptación para hacer conexión con mecanismos o modelos cuantitativos es nula o muy pobre; segundo, ninguno de ellos parece ofrecer un marco adecuado para el modelado del tiempo que permitiera sincronizar sus predicciones con el comportamiento dinámico de una planta. Dentro de la variedad de paradigmas de aprendizaje en "Machine Learning", el más afín a nuestros requerimientos es el conocido como: *Aprendizaje Mediante Ejemplos*[1].

El elemento básico detrás del paradigma de aprendizaje mediante ejemplos son los algoritmos de clasificación mediante árboles de decisión. El algoritmo de clasificación mediante inducción de árboles de decisión fue originalmente propuesto por Hunt [Hunt 66] con su sistema CLS ("Concept Learning System"). Quinlan modificó CLS para producir el conocido sistema ID3 ("Iterative Dichotomister") [Quinl 79,86]. La estructura de ID3 ha servido de base para desarrollar un buen número de variantes entre las que se encuentran: ID4 [Schli 86], C4 [Quinl 87], ID5 [Utgof 89] y NewID [Boswe 90]. Las principales variaciones introducidas por los nuevos algoritmos se encuentran en la modificación del algoritmo para: permitir manejar atributos cuantitativos y lógicos binarios, utilizar medidas difusas para representar las clases, permitir aprendizajes incrementales y en aumentar la eficiencia de búsqueda por los árboles para reducir la complejidad, la cual suele ser muy alta.

Un estudio comparativo de algunas de las variantes existentes puede encontrarse en [Minge 89]. En [Hunt 92] se presenta una aplicación de inducción mediante árboles de clasificación para modelar un sistema de control simple. En ese artículo, Hunt propone una reformulación del problema de control como un problema de clasificación el cual es resuelto utilizando NewID. Los ejemplos son generados por un controlador tipo PI. Aunque, como apuntamos anteriormente, estos métodos no están muy bien situados para trabajar con sistemas dinámicos, los últimos avances de estos algoritmos que han permitido incluir atributos cuantitativos y reducir el costo computacional, permite pensar que podrían utilizarse en un futuro en aplicaciones de control.

[1] del inglés: "Learning From Examples"

3.4.1.2 Metodologías Inductivas – TGS

A diferencia de las metodologías de razonamiento inductivo generadas por la IA, concebidas para la adquisición de *conceptos*, las generadas por la TGS están enfocadas al tratamiento de *sistemas*. Estas últimas están mejor situadas para capturar el conocimiento implícito en los sistemas dinámicos, ya que consideran de manera explícita tanto el componente causal como el componente temporal.

EL esquema del que partiremos aquí fue originalmente desarrollado por Klir [Klir 85] como un recurso metodológico para la solución de problemas en la Teoría General de Sistemas (o Teoría de Sistemas Generales) y que designó con el nombre de *"General System Problem Solver" (GSPS)*. La formulación del GSPS tal y como se presenta en [Klir 85] es la agregación del trabajo desarrollado durante casi 20 años por el autor. La primera característica a resaltar del GSPS es que su uso no está restringido ni a una disciplina (ciencias, ingeniería, medicina o administración) particular, ni a un tipo específico de problemas (diseño de sistemas, prueba, diagnosis, toma de decisiones, etc). La única taxonomía que establece el GSPS es una *jerarquía de niveles epistemológicos de sistemas*, es decir, una clasificación basada en los diferentes niveles que guarda la estructura del conocimiento que se tiene del sistema a estudiar. Aunque en el esquema del GSPS se puede ascender (inducción) y descender (análisis) por los niveles jerárquicos, para efectos de dar una explicación breve que muestre su potencial para construir modelos dinámicos cualitativos, describiremos solamente el sentido ascendente (inductivo) y restringiremos la amplitud de algunos términos.

El primer nivel jerárquico es el **nivel 0**, o nivel de *sistemas fuente*. En este nivel se define la fuente de datos empíricos, es decir, se define un marco de estudio en el que decidimos qué información del sistema es relevante para nuestro estudio y cuál es el propósito de la investigación. En términos simples definimos el conjunto de variables que *potencialmente* nos serán útiles. Existen varios criterios para agrupar (si se desea) estas variables. Bajo uno de esos criterios se pueden dividir las variables en *entradas* y *salidas*. Bajo tal partición, los estados de las variables de entrada son vistos como las condiciones que afectan las variables de salida.

El siguiente nivel epistemológico es el **nivel 1**, o nivel de *sistemas de datos*, en el que las variables adquieren valores para proporcionarnos datos. Dependiendo del problema a resolver, los datos pueden obtenerse por observación o medida (si el problema es modelado) o ser definidos como estados deseables (si el problema es de diseño). Los datos son almacenados en filas ordenadas (matrices) respecto a alguna variable de soporte (tiempo, espacio),

en nuestro caso al tiempo. La naturaleza en que los datos son obtenidos y/o almacenados puede ser cualitativa o cuantitativa, pudiendo haber operaciones de transformación.

En el siguiente nivel, el **nivel 2**, o nivel de *sistemas generativos* o *de comportamiento*, la tarea es *descubrir* las relaciones causales[2] entre las variables a partir de las cuales los estados de las variables puedan ser *generados* mediante condiciones iniciales o de frontera particulares. Cuando las relaciones causales son encontradas, es necesario darles una interpretación en el marco de estudio que se investiga. Si la interpretación no es exitosa, es necesario descender en los niveles para efectuar los cambios necesarios. La caracterización de las relaciones espacio–temporales entre las variables que se obtiene en este nivel puede ser vista como un modelo del sistema en estudio. En nuestro caso de modelado, si a partir de este modelo pueden generarse los estados (comportamiento) esperados, en base a condiciones iniciales o de frontera deseados se dice que la interpretación es exitosa y el modelo es válido. Los datos generados pueden ser exactos (determinista) o aproximados en algún sentido (probabilísticos, difusos). Si los datos generados son de naturaleza cualitativa se dice que el modelo es cualitativo.

El elemento primordial para la búsqueda de relaciones causales espacio–temporales se conoce bajo el nombre de *máscara*. Una máscara es simplemente un selector de variables (causas) que, al desplazarse sobre la matriz de datos permite generar los datos de las variables deseadas (efectos). Cada prueba corresponde con una máscara distinta que al desplazarse sobre el total de datos de la matriz, genera una interpretación del comportamiento. La máscara que obtiene la mejor interpretación es conocida como la máscara óptima. La máscara óptima y la interpretación del comportamiento correspondiente son seleccionadas como el modelo del sistema.

Relaciones entre los diferentes modelos son obtenidas en el **nivel 3**, o nivel *sistemas de estructura*, y relaciones entre relaciones de modelos en el **nivel 4**, o *metasistemas*. Sin embargo, para efectos de nuestros requerimientos nos detendremos en el nivel 2, debido a que ya contamos con el modelo buscado que describe adecuadamente el comportamiento del sistema en estudio. Cabe resaltar solamente el gran potencial del GSPS que aún queda por explotar.

La primera implementación del GSPS fue desarrollada por Uyttenhove [Uytte 78], [Uytte 81] bajo el nombre de *Systems Approach Problem Solver (SAPS)*. Esta implementación, un tanto rudimentaria computacionalmente hablando, fue escrita en APL y tenía por objetivo instrumentar computacionalmente las

[2] "support–invariant relational characteristics" en la terminología de Klir.

ideas desarrolladas por Klir hasta el momento. Una segunda versión de SAPS, mucho más flexible y poderosa, fue desarrollada por Cellier y Yandell [Celli 87] implementándose como una biblioteca de funciones en Fortran llamadas desde el programa CTRL–C con el objetivo de aprovechar sus capacidades interactivas y de manipulación de matrices. La nueva versión, llamada SAPS–II, fue enriquecida más tarde con la introducción de medidas difusas [Li 90].

3.4.1.3 Razonamiento Inductivo Difuso

Dado que el paradigma del GSPS es una técnica demasiado extensa y general, hemos querido restringir el campo de estudio al conjunto de tareas que permiten generar un modelo cualitativo (nivel 2) a partir del comportamiento observable (nivel 0) del sistema que se desea representar. En el contexto de la TGS, este particular ascenso por las jerarquías epistemológicas se le conoce como *modelado inductivo de sistemas*. En el desarrollo de esta tesis nos referiremos a esta parte como una metodología independiente que denotaremos bajo el nombre de: Razonamiento Inductivo Difuso y para el cual SAPS–II está particularmente bien adaptado. A partir de ahora nos referiremos al Razonamiento Inductivo Difuso mediante el acrónimo FIR en virtud del que el término fue originalmente acuñado en idioma inglés como *Fuzzy Inductive Reasoning*.

El FIR parece una técnica adecuada para el modelado y simulación cualitativos de sistemas de los que independientemente de la información conocida respecto a su estructura (normalmente representada por medio de ecuaciones algebraicas y/o diferenciales), podemos registrar su comportamiento. El razonamiento inductivo difuso parte de la información medida del sistema y la convierte en información difusa. Mediante un algoritmo de imposición de restricciones, experimenta, una a una, el conjunto de posibles estructuras de modelos cualitativos. Mediante la evaluación de la cantidad de información contenida en las relaciones temporales y espaciales que existen entre las variables de entrada y salida de cada estructura, selecciona el modelo óptimo mínimo que representa al sistema. Finalmente, a partir del modelo seleccionado, es capaz de predecir con gran precisión los estados correspondientes a las condiciones iniciales y de frontera que se deseen, regenerando el estado cualitativo difuso recién predicho en un formato numérico. No nos extendremos aquí en las particularidades de la metodología, ya que al ser seleccionada como la mejor herramienta de modelado y simulación cualitativos bajo los requerimientos previamente mencionados, será descrita con detalle en el siguiente capítulo.

EL FIR ha demostrado ser una herramienta poderosa para el modelado de sistemas cualitativos ya que ha sido exitosamente aplicada a sistemas físicos de comportamiento altamente no lineales [Celli 95b], a sistemas de estructura variable [deAlb 94b], a sistemas de control de sistemas [Celli 95a], [Mugic 94], a la detección de fallos en sistemas de gran complejidad [deAlb 93], [deAlb 94a], y a varios tipos de sistemas biomédicos [Nebot 93,94b]. Al final de esta sección se presentan algunas de sus principales ventajas sobre las demás alternativas.

3.4.2 Algoritmos de Identificación

Existen un buen número de algoritmos de identificación de sistemas, todos ellos basados en la idea de sistema en la forma de señales de entrada y salida. Por un lado se encuentran los pertenecientes a la teoría de identificación clásica, por el otro las teorías emergentes como los algoritmos para el entrenamiento de redes neuronales y los algoritmos genéticos. En los métodos de la teoría de identificación clásica, considerados como métodos cuantitativos de modelado, el modelo es representado mediante funciones matemáticas cuyos parámetros, originalmente desconocidos, son estimados a través de algoritmos de optimización en los que una función de costo es minimizada. Este tipo de métodos requieren un estudio profundo del comportamiento del sistema. Los principales métodos, clasificados de acuerdo a la estructura del modelo matemático, son:

ARX ("Auto Regressive eXternal input"), es considerado como el modelo más sencillo. Utiliza un método de regresión lineal en el que los parámetros son ajustados con el método de mínimos cuadrados.

OE ("Output Error"), similar a ARX ya que utiliza de igual manera el método de mínimos cuadrados, pero a diferencia de los ARX en los que la relación que guarda la función de error con los parámetros a estimar es necesariamente lineal, la relación en el método OE permite una estructura no lineal.

ARMAX ("AutoRegressive Moving Average eXternal input") estima los parámetros optimizando una función de error no lineal de manera similar al método OE. Además de identificar un modelo de la dinámica del sistema, se identifica un modelo de la perturbación, aunque ambos comparten el componente autorregresivo.

BJ ("Box Jenkins") obtiene todos los parámetros del modelo de la dinámica del sistema de manera independiente a los parámetros del modelo del

espectro de perturbación. Este método permite mantener la consistencia en el bucle cerrado superando en precisión al método OE bajo estas condiciones. Sin embargo, en bucle abierto el desempeño de ambos es similar.

Los principales problemas con estos métodos son [Zhu 93]:

- los típicos problemas de convergencia heredados de los algoritmos de optimización que utilizan, problemas que se agudizan a medida que el orden del sistema aumenta.
- funcionan bien únicamente con plantas que presentan un comportamiento lineal y estacionario. En cuanto aparecen no linealidades, los órdenes de este tipo de sistemas suelen dispararse haciéndose muy complejo el proceso de identificación. Se han identificado ciertas familias de no linealidades bajo las cuales la identificación es completamente intratable.
- Su utilización está indicada para estructuras tipo SISO. El costo computacional de la identificación de parámetros es extremadamente alto cuando se intentan tratar estructuras más complejas (como MISO o MIMO). La selección de la configuración orden/estructura es prácticamente exhaustiva.
- Es muy complejo utilizar estos métodos para diseñar modelos que requieran encontrar una entrada óptima. Esta característica es particularmente restrictiva cuando el objetivo de identificación es el de diseño de controladores.
- Es difícil que proporcionen una descripción (cuantificación) del modelo del error, lo cuál es un requisito indispensable para el análisis y diseño en control robusto.

3.4.2.1 Redes Neuronales

Todas las metodologías tratadas hasta ahora, ya sean las basadas en la estructura o las basadas en el comportamiento, tienen la característica común de ser relativamente novedosas y de que al no haber suficientes aplicaciones, no puede afirmarse su aplicabilidad para el desarrollo de controladores. El caso del paradigma de redes neuronales artificiales, o solamente redes neuronales ("Neural Networks" – NN), es sustancialmente diferente. En principio, este paradigma quedaría fuera del marco de esta revisión, ya que no puede ser

clasificado como una metodología de identificación cualitativa. Sin embargo, no hemos podido evitar mencionarlas debido a que mantienen propiedades que aparentemente responden de forma adecuada a los requerimientos planteados al comienzo de este capítulo, tales como los de permitir modelar imprecisión y capturar comportamientos no lineales.

Las NN fueron propuestas hace más de cuarenta años, aunque bajo el nombre de *perceptrón* [McCull 43], pero debido a razones 'desafortunadas' no fue sino hasta hace unos pocos años que la investigación fue reiniciada, esta vez con un gran vigor y popularidad. El número de referencias bibliográficas de redes neuronales, y particularmente las aplicadas al campo de control, es verdaderamente tan grande (por ejemplo los números especiales del *IEEE Control System* en 88, 89 y 90), que resulta indudable su potencial aplicación al diseño de controladores.

Los elementos de procesamiento de una NN son las *neuronas*. La principal propiedad de una neurona es su memoria local que le permite obtener una respuesta a las entradas (estímulos) que recibe. De acuerdo con su memoria local procesa la información que recibe y luego la transmite, mediante canales unidireccionales llamados conexiones, para que otras neuronas continúen el proceso. Funcionalmente la operación de una neurona es la siguiente: la neurona recibe un conjunto de señales de entrada, cada señal es ajustada mediante un factor aritmético, llamado *peso*, las señales ajustadas más un factor de polarización, conocido como "bias", son agregados mediante una suma. Finalmente, el resultado de esa suma es ajustado mediante una función, conocida como función de activación, produciendo el mensaje de salida que se enviará a otro conjunto distinto de neuronas, de acuerdo a la estructura de la red. Un aspecto importante, particularmente en el problema de control, es que la memoria de las neuronas, almacenada en los pesos, es con respecto a sí, es decir, no guarda memoria de las señales que pasan a través de ella. Por esto, si se desea mantener una señal durante algún tiempo, es necesario hacer uso de retardos y realimentaciones; si así lo hace, se le conoce como una *red recurrente*, si no, se le llama una *red hacia adelante*[3] o una red *en cascada*.

El algoritmo que permite encontrar los pesos en función de las señales que pasan a través de ella son conocidos como algoritmos de aprendizaje, los cuales pueden clasificarse en tres grupos [Torra 89]:

- Reglas de minimización del error. Este es un método catalogado como *aprendizaje supervisado*. El algoritmo se desarrolla en dos etapas. En la primera etapa, conocida como etapa de entrenamiento, se proporciona a

[3]en inglés: "feedforward"

la red un conjunto de datos de entrada con sus correspondientes salidas esperadas. Mediante un algoritmo de minimización del error, obtenido entre la salida esperada y la salida real, se ajustan los pesos de la red propagando el error hacia atrás. El proceso se repite para todos los datos y por varias iteraciones hasta que el algoritmo converge y los pesos son determinados. Los algoritmos más conocidos son el algoritmo Delta y el algoritmo de "backpropagation". En la segunda etapa, la etapa de operación, la red produce salidas en respuesta a los nuevos patrones de entrada pero los pesos permanecen fijos.

- Reglas de correlación. Este es un método catalogado como *aprendizaje no supervisado*, es decir, no requiere un conjunto de datos entrada/salida previamente. En este caso, un conjunto de leyes ajusta los pesos entre las conexiones de acuerdo a la correlación entre la activación de las dos neuronas involucradas. Las reglas de aprendizaje actualizan el valor de alguna neurona en la red sumando un término que es proporcional al producto de los valores de activación de las dos neuronas interconectadas. Este tipo de neuronas son típicas en las memorias asociativas y en los modelos de aprendizaje competitivo, en los que las neuronas son organizadas en capas jerárquicas unidas mediante conexiones de excitación. Las neuronas dentro de cada capa son organizadas en grupos ("clusters"); cada grupo compite con los demás grupos de su mismo nivel y mediante una conexión de inhibición el grupo ganador inhibe a los demás.

- Reglas de Reforzamiento. Este tipo de aprendizaje es considerado como un intermedio entre el caso no supervisado y supervisado. Dados unos pesos iniciales, la red calcula la salida en respuesta a la entrada. La red es evaluada de acuerdo a una medida global del comportamiento deseado con lo que se produce una *señal de reforzamiento* el que se realimenta a la red. Los pesos son adaptados de acuerdo a la señal de reforzamiento aumentando los pesos que contribuyeron al buen comportamiento y disminuyendo los que presentaron un desempeño pobre. La red revisa los pesos de tal manera que evita reforzamientos negativos en el futuro.

Los elementos que definen la estructura de una red son las *capas o niveles* y el número de conexiones entre las neuronas. Las neuronas están estratificadas en *capas* y existen tres tipos de ellas: la capa de entrada, las capas internas u ocultas (de las que puede haber más de una), y la capa de salida. Existen tres configuraciones principales de redes:

- Perceptrón Multicapa (MPL). Esta es una red típica de operación hacia

adelante ya que la información siempre fluye de la capa de entrada a la de salida. Como la primera capa no ejerce ningún proceso sobre la información, algunos autores no la consideran como una verdadera capa. La tarea de esta capa es distribuir la señal de entrada a la red al conjunto de neuronas de la siguiente capa. Ésta es siempre una capa de tipo oculta donde la información es procesada y enviada a la siguiente capa, la cuál puede ser nuevamente una siguiente capa oculta o si ya es la última, a la capa de salida. Característicamente este tipo de configuración utiliza algoritmos de aprendizaje supervisados, siendo el más común el de "backpropagation".

- Red de Hopfield. En este caso la red consiste solamente en una capa que recibe el nombre de capa Hopfield. La principal característica de esta configuración es la existencia de conexiones de realimentación por lo que se le conocen también como redes recurrentes. La capa única funciona como capa de entrada, recibiendo información desde el exterior, y como capa de salida enviando directamente la señal a la salida de la red. Además, cada una de las neuronas está completamente interconectada con todas las demás neuronas de la red. Así las señales que llegan a cada neurona pueden ser de dos tipos, las señales que entran a la red desde el exterior y las señales de realimentación procedentes de las demás neuronas. A diferencia de las redes con estructura MPL en la que una vez identificada la relación entre las señales de entrada y de salida (valores de los pesos) permanece fija, las redes de Hopfield permiten que el sistema se adapte dinámicamente al entorno hasta que alcanza la respuesta deseada.

- Red de Kohonen. Conocida también como mapa autorganizado de Kohonen, tiene una estructura similar a la de Hopfield, que se caracteriza porque tiene una sola capa en la que cada una de sus neuronas tienen un alto grado de interconexión tanto con sus neuronas *vecinas*, como consigo misma. Tiene dos conjuntos de pesos para procesar la información que les llega. El primer conjunto se utiliza para controlar la interacción entre las neuronas con las que se interconecta. El segundo grupo de pesos es adaptable, ajustado de acuerdo a las entradas externas y a la calidad de la salida esperada. El proceso de aprendizaje de esta clase de redes es del tipo no supervisado.

En la aplicación de las NN al control cabe hacer notar que la mayoría de las aplicaciones han utilizado redes basadas en el algoritmo de "backpropagation". Dos buenas revisiones del estado del arte de las NN en control son [Mille 90] y [Hunt 92]. En [Hunt 92] encontramos cinco estructuras en las que se han utilizado NN para control:

- Control Supervisado. En este caso un sistema de NN se conecta a un sistema controlado por un operador humano. Una vez entrenada la red con la experiencia del operador, ésta es capaz de reproducir el comportamiento aprendido. El caso de un "pole–cart" es presentado en [Grant 89].

- Control Directo Inverso. En este caso el modelo inverso es puesto en cascada con el sistema a ser controlado para que el sistema compuesto resulte en un 'mapeo' idéntico entre la respuesta deseada (la entrada a la red) y la salida del sistema controlado. La red actúa directamente como el control. Este tipo de aplicaciones es común en aplicaciones de robótica [Mille 90]. Este tipo de sistemas suele carecer de realimentación por lo que no se consideran muy estables en sí mismos.

- Control con Modelo de Referencia. En este caso el comportamiento deseado del sistema en bucle cerrado es proporcionado por un modelo de referencia. El sistema de control intenta que la salida de la planta se tienda a comportar asintóticamente como el modelo de referencia, registrando el error continuamente. En esta estructura la NN, que funge como controlador, es entrenada mediante el error registrado previamente. Algunos trabajos en esta dirección son reportados en [Naren 90].

- Control de Modelo Interno. En este caso un sistema directo y un modelo inverso son usados directamente como elementos dentro del circuito de realimentación. Un modelo de la planta (modelado con una NN) es puesto en paralelo con la planta real y recibe la señal del controlador (una NN modelando un sistema inverso). La diferencia entre ambos modelos es usada como señal de realimentación, la cual es procesada por un subsistema de control (normalmente un filtro lineal) que es usado para producir robustez y estabilidad al cerrar el bucle. Se hace notar que este tipo de estructuras están limitadas a sistemas estables en bucle abierto. Ejemplos de este tipo de NN se encuentran en [Hunt 91].

- Control Predictivo. En este caso una NN previamente entrenada proporciona una predicción de la respuesta futura de la planta sobre un horizonte especificado. Las predicciones de la NN son optimizadas numéricamente mediante un modelo dinámico para calcular una señal de control adecuada.En [Mayne 90] se muestra que el método tiene buenas propiedades de estabilidad.

Las principales cualidades de las NN son:

- La capacidad para trabajar y ser implementadas computacionalmente en paralelo.
- Trabajar con patrones entrada–salida permitiendo que la naturaleza de los datos sea arbitraria.
- Buena capacidad para modelar sistemas no lineales. Según el número de neuronas empleadas en la red la precisión varía arbitrariamente.
- Permite filtrar algunos tipos de ruido que se presente en las señales de entrada.
- No requieren de ningún tipo de conocimiento de la estructura interna del sistema.

Sus principales limitaciones son:

- Las redes neuronales no ofrecen normalmente ninguna evaluación de la calidad de sus predicciones. Eso, en el control de sistemas, puede ser bastante peligroso.
- No ofrecen ninguna interpretación de sus mecanismos de razonamiento. En un sistema de control jerárquico, esa información es importante para los controladores de nivel más elevado.

Aunque las redes neuronales se mostraron como una herramienta muy poderosa para varias aplicaciones incluso el control de sistemas, tienen desventajas importantes. En particular, no ofrecen una buena interfaz con los controladores (o razonadores) de nivel más elevado, por ejemplo los que son responsables para la planificación, la coordinación de subsistemas, y el diagnóstico de fallos. Tiene que ver con la forma en la que las NN adquieren el conocimiento. En forma similar a lo que ocurre con el modo de operación del cerebro humano (como lo comprendemos ahora), prácticamente no es posible conocer nada respecto al proceso de operación; se sabe que trabajan pero no qué es lo que ocurre en su interior. Esto conlleva a la situación de que es prácticamente imposible continuar la cadena epistemológica del conocimiento adquirido por las neuronas, es decir, no podemos establecer ningún tipo de explicación, auditoría ni justificación de por qué una NN alcanza las conclusiones que alcanza.

Además, para el control robusto de un sistema es importante que el controlador sepa como efectúa su salida (la señal de control) la planta. ¿Cuál

es el rango estable de valores de la señal de control? ¿Cuál es el rango (más estrecho) de valores suboptimales? Un controlador inteligente debe de tener un tipo de "conciencia", es decir un conocimiento de sus efectos sobre la planta. Las redes neuronales no son "inteligentes" en este sentido. Una NN *siempre* produce un valor a su salida. Si la señal recibida por sus entradas es conocida, el valor producido por la red será probablemente justificable; si no, el valor que produce la red será una conjetura que no puede defenderse basándose en los datos disponibles. Esto en control simplemente es peligroso.

Lo que nos atrajo más a la solución con FIR que con NN es el hecho que la metodología FIR nos ofrece un buen compromiso entre la computación numérica y la simbólica, que ofrece un razonamiento interno que es interpretable y que contiene mecanismos internos para la validación de sus propias decisiones. FIR, como los sistemas basados en conocimiento, organiza su información interna en reglas simbólicas de las que se puede continuar trabajando en forma ascendente en los aspectos cognoscitivos, los cuales son claves para el desarrollo de controladores más inteligentes; sin embargo, FIR, como las redes neuronales, produce un valor cuantitativo y numérico a su salida que puede aplicarse directamente a una planta real y que puede usarse en el análisis cuantitativo de la actuación del sistema.

Hay otras razones más para nuestra decision de trabajar con FIR. Un aspecto tiene que ver con los resultados que pueden esperarse de la investigación. El número de investigadores en NN es del orden de cientos; las publicaciones, congresos y "workshops" se han venido multiplicando geométricamente y, aunque es posible que ahora se encuentre estabilizado, resulta muy difícil hacer alguna aportación significativa sin contar con un conocimiento de respaldo avalado por años de trabajo o restringirse a un aspecto muy estrecho de la investigación. El FIR por el contrario es una tecnología virgen con un gran potencial y en donde solamente se ha empezado a trabajar. Pensamos que la línea de investigación abierta puede producir contribuciones más significativas (más básicas) no sólo en control sino además en muchos otros campos de la ingeniería y de las ciencias.

3.5 Conclusiones

A lo largo de esta sección ha sido presentado, de manera breve, el abanico de posibilidades con que se cuenta para modelar un sistema de manera cualitativa. Dando énfasis a los problemas de sistematización, de acoplamiento y de precisión requerido por los controladores, se ha intentado mostrar los criterios

para la elección del FIR como metodología para la sistematización del diseño de controladores.

En términos generales puede decirse que los métodos basados en la estructura, como la física cualitativa y los sistemas expertos, son paradigmas mejor situados para emular procesos de pensamiento más elevados y que dentro del campo del control automático pueden dirigirse a desarrollar tareas como la supervisión, la coordinación y la detección de fallos, más que al control propiamente dicho. En concordancia con esto, Saridis [Sarid 89] ha desarrollado una formulación analítica del *principio de incremento de precisión con la disminución de la inteligencia*. De acuerdo a este principio, tres niveles de control deben de ser considerados: el nivel de organización, el nivel de coordinación y el nivel de ejecución. Mientras que en el primer nivel se requieren grandes volúmenes de conocimiento del dominio, así como la capacidad para procesarlo, es decir más inteligencia, en el nivel de ejecución los controladores requieren secuencias específicas para llevar a cabo las tareas, lo que implica mayor precisión. El nivel de coordinación, el cuál actúa como una interfaz entre los otros dos niveles, puede asignar probabilidades para cada acción. Las probabilidades son usadas para calcular la entropía del sistema de control. Saridis demuestra con su principio que minimizando la entropía se alcanza una ley de control óptima. Mucho del conocimiento de alto nivel de control (planificación, diagnóstico de fallos, secuencias de acciones, etc) puede ser mejor representado por reglas simbólicas o reglas lingüísticas. El control de bajo nivel puede ser realizado mediante una combinación de controladores convencionales (por ejemplo PID), controladores difusos o controladores neuronales. La mayor parte del nivel de coordinación, que desde otro punto de vista puede verse como un acoplamiento entre estructuras simbólicas y numéricas, suele ser absorbido por los niveles de organización o de ejecución dependiendo de la estrategia seleccionada.

Trazando un espectro unidimensional de las diferentes teorías de modelado disponibles, podemos localizar en un extremo del espectro las técnicas puramente cualitativas de la IA, y en el extremo opuesto se encontrarían las técnicas puramente cuantitativas de la Teoría de Control. Los Sistemas Difusos, que se ubican en medio de los dos extremos, pueden servir como una metodología puente que permite combinar las estructuras simbólicas de la IA con las estructuras cuantitativas de la teoría de control. El núcleo de estos sistemas puede tomar la forma de una base de conocimiento o de una red neuronal. Las interfaces de fusificación y defusificación permiten convertir información cuantitativa en información cualitativa y viceversa. Las estructuras de representación del conocimiento, los algoritmos de reconocimiento de patrones y los mecanismos de inferencia de esos sistemas

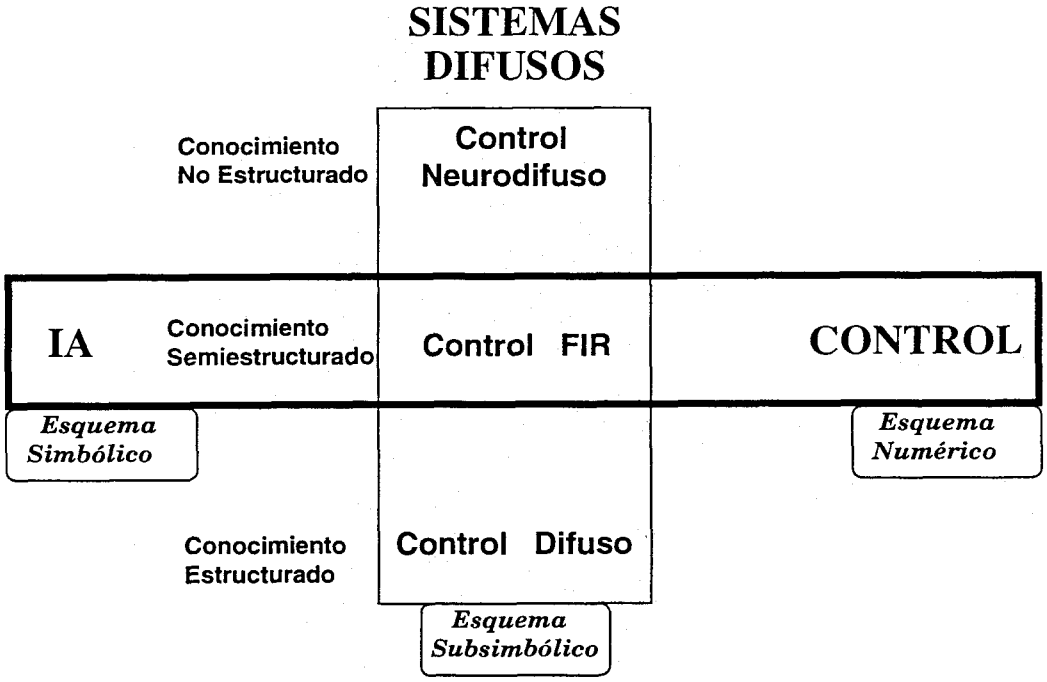


Figura 3.1: IA y Control

difusos emulan la experiencia y las capacidades de inducción y deducción de la mente humana, que conforman a las bases de los procesos del pensamiento humano y de la capacidad para tomar decisiones.

La forma en que el conocimiento es representado establece otro eje. Este eje denota cuán estructurado es el conocimiento que captura el modelo cualitativo que representa al controlador difuso. La figura 3.1 muestra las relaciones entre IA, Sistemas Difusos, Teoría de Control por un lado, y sus relaciones con las Redes Neuronales, Razonamiento Inductivo y Sistemas Expertos por el otro. En la parte superior del gráfico puede encontrarse el conocimiento no estructurado de la aproximación neurodifusa. En la parte inferior puede verse el conocimiento altamente estructurado de los sistemas expertos determinados mediante conocimiento heurístico. En el centro, usando una aproximación semiestructurada para caracterizar el conocimiento, la metodología de FIR construye un conjunto de reglas difusas tomando como base datos numéricos de la planta y un conjunto de relaciones causales (tanto estructurales como temporales) entre las variables derivadas a partir de los datos medidos.

Algunas de las ventajas de la metodología FIR son las siguientes:

1. Los razonadores inductivos permiten, en contraste con los modelos cualitativos puros, tratar el tiempo como un variable continua (cuantitativa). Esto es de una importancia primordial si deseamos modelar y simular sistemas mixtos cuantitativos y cualitativos.

2. La técnica puede ser aplicada para cualquier sistema disponible en que se pueda realizar experimentación y observación. El razonamiento inductivo es completamente basado en patrones, por lo que no es necesario conocer la estructura interna del sistema bajo estudio. Con respecto a esto, el razonamiento inductivo es similar a las redes neuronales.

3. La metodología contiene un mecanismo inherente de validación dentro del método de simulación, que evita que se alcancen conclusiones que no son justificables en base a los hechos observados. Respecto a esto, los razonadores inductivos son similares a los sistemas basados en conocimiento.

4. El razonamiento inductivo opera internamente de manera cualitativa, justo como los razonadores basados en conocimiento. Por lo tanto, es posible aplicar metaconocimiento para mejorar la calidad de la máquina de inferencia.

5. El razonamiento inductivo opera internamente de manera cualitativa justo como los razonadores basados en conocimiento. Por lo tanto, es posible seguir el rastro del proceso de razonamiento si se desea.

Capítulo 4

Razonamiento Inductivo Difuso

4.1 Introducción

La herramienta de modelado y simulación cualitativos que subyace en nuestra metodología de diseño de controladores difusos es el Razonamiento Inductivo Difuso o FIR[1], el cual será explicado con todo detalle en esta sección. La investigación con FIR está completamente soportada, computacionalmente hablando, por el paquete SAPS–II [Celli 87]. SAPS–II es la implementación más completa que existe del paradigma de "General System Problem Solving" [Klir 85] y del cual FIR es sólo una parte. SAPS–II se encuentra codificado como una biblioteca de funciones (escritas en Fortran) que pueden ser llamadas desde un programa, o interactivamente, ya sea desde CTRL–C [SCT 85] o desde MATLAB [Mathw 92]. También, para efectos del modelado mixto cuantitativo/cualitativo se cuenta con una versión reducida de SAPS-II, sólo con las funciones requeridas, en forma de una biblioteca de funciones para el paquete de simulación numérica ACSL [MGA 86].

La metodología FIR consta esencialmente de dos procesos: el proceso de modelado cualitativo y el proceso de simulación cualitativa, los cuales son realizados mediante cuatro funciones de SAPS-II. Estas cuatro funciones, que constituyen el núcleo conceptual del FIR y algorítmico de SAPS-II, son:

(a) Codificación Difusa,

[1] **F**uzzy **I**nductive **R**easoning

(b) Búsqueda de Máscaras Óptimas,

(c) Predicción Difusa, y

(d) Regeneración (escalar).

Para el proceso de modelado cualitativo se requieren las funciones de codificación difusa y de búsqueda de máscaras óptimas. Para el proceso de simulación cualitativa son necesarias las funciones de codificación difusa, de predicción difusa, y de regeneración escalar.

El núcleo de FIR ha sido objeto de continuos refinamientos y extensiones que se han producido en el transcurso de la investigación (comenzada hace más de tres años) que dió lugar a la elaboración de tres tesis de doctorado desarrolladas en la Universidad Politécnica de Cataluña, tal y como fue descrito en el capítulo 1 sección 1.2. Aunque las hipótesis de investigación de las tres tesis difieren completamente, convergen en el objetivo común de evaluar y madurar la tecnología del FIR. La exposición del FIR que sigue a continuación describe el estado actual de la metodología.

4.2 Modelado Cualitativo

Modelado cualitativo se refiere al proceso de identificar el conjunto de reglas, una máquina de estados finitos, que caracterice de mejor forma el comportamiento del sistema bajo investigación en términos cualitativos.

4.2.1 Obtención de Datos

El primer paso en el modelado de sistemas cualitativos es definir cuál es el sistema bajo estudio. Este paso no puede ser automatizado completamente, por lo menos no en el caso general, debido a su naturaleza altamente heurística. Sin embargo, pueden establecerse de manera sistemática cada uno de los elementos que deben de satisfacerse. Se debe:

- Seleccionar el conjunto de variables que potencialmente pueden resultar de interés. No es necesario en este paso indicar exactamente cuáles variables (señales) determinan el comportamiento, lo único que debe de cumplirse es que esas variables (cualesquiera que sean) se encuentren dentro del grupo de variables seleccionadas como potencialmente representativas. En una etapa posterior, la

metodología FIR será capaz de extraer aquel grupo de variables que contribuyan de manera relevante a la dinámica del sistema y desestimar el resto.

- Según el objetivo del estudio, deben de indicarse cuáles son las variables de salida. Esto significa decidir qué variables van a definir el comportamiento del sistema. Si hubiera más de una variable de salida cada una formará un subsistema del tipo MISO. En un sistema de control, por ejemplo, siempre se tiene el conocimiento *a priori* de cuál es, o son, las variables de salida del controlador que ejercerán la acción de control sobre la planta.
- La información proporcionada debe ser la más rica posible, en el sentido de que se encuentren contenidos el mayor número de los comportamientos que el sistema pueda exhibir. En este punto, hay dos factores implicados: el intervalo de muestreo de las señales y el tipo de excitación dada al sistema. No siempre es posible mantener bajo control estos dos factores, por ejemplo, no se puede excitar de manera arbitraria el sistema cardiovascular humano mediante impulsos eléctricos para registrar la gama de sus posibles comportamientos, lo mismo sucede con una central nuclear. Sin embargo, debe tenerse presente siempre, que resulta imposible *aprender* la dinámica de un sistema, si ésta no es previamente *observada*.

Intervalo de muestreo. Para ser capaz de capturar la dinámica del sistema bajo estudio, es imprescindible que se obtengan de alguna manera las constantes de tiempo más rápida y más lenta que le caracterizan. La constante más rápida nos permitirá establecer el intervalo de muestreo, δt, con que las señales del sistema serán registradas. La más lenta será utilizada más adelante para determinar el número de retardos temporales que deben ser considerados en la caracterización del modelo. La determinación de δt en una simulación mixta cuantitativa/cualitativa tendrá un efecto determinante en la caracterización del modelo cualitativo para que puedan acoplarse debidamente los subsistemas involucrados. No existe todavía una manera automática para determinar el valor adecuado para este parámetro, siendo actualmente objeto de una intensa investigación. Según el teorema de muestreo de Shannon, el intervalo de muestreo no debe ser mayor que la mitad de la constante de tiempo más rápida $t_{rápida}$:

$$\delta t \leq \frac{t_{rápida}}{2} \tag{4.1}$$

En la práctica esta constante de tiempo se obtiene como sigue. Si contamos con un sistema físico abierto a la investigación, o un modelo matemático de él, podemos experimentar diferentes frecuencias de excitación para obtener un diagrama de Bode del sistema. A través de él podemos determinar las frecuencias propias[2], y en particular, la más corta y la más larga. La frecuencia propia más corta ω_{baja} es la frecuencia más lenta, en la cuál el comportamiento tangencial de la amplitud del diagrama de Bode cambia en $-20\ dB/década$ y el valor propio más largo ω_{alta} es la frecuencia más alta en donde esto ocurre. La constante de tiempo más larga, o tiempo de establecimiento, t_s, y la más corta, $t_{rápida}$, del sistema pueden calcularse como sigue:

$$t_s = \frac{2\pi}{\omega_{baja}} \quad ; \quad t_{rápida} = \frac{2\pi}{\omega_{alta}} \tag{4.2}$$

En el caso de que se disponga de un modelo analítico del sistema que se estudia, las constantes pueden ser obtenidas directamente de él.

Excitación del sistema (experimentación). Si se cuenta con la posibilidad de excitar al sistema (o un modelo analítico de él) como es el caso de la mayoría de sistemas físicos en ingeniería, debe tenerse una estrategia que garantice la obtención de la gama completa de estados dinámicos. Una de las maneras de hacer esto es utilizando señales de *ruido blanco*[3]. Este tipo de señales tiene un espectro plano y por lo tanto permite excitar al sistema en todo el margen de frecuencias a las que responde el sistema a modelar, por lo que la información registrada bajo este tipo de excitación contendrá la dinámica completa del sistema. Lamentablemente, en la práctica no es fácil producir este tipo de señales, e incluso, dependiendo de la naturaleza del sistema, resulta imposible aplicarlas. Por estas razones, resulta más frecuente (y sistemático) el uso de ruido blanco pseudoaleatorio[4] que prácticamente permite recoger las mismas características del ruido blanco.

Una última consideración debe hacerse con respecto al problema de excitación del sistema. Si bien este tipo de señal es ideal para capturar la dinámica del sistema, no lo es para capturar todos los estados. Esto significa que el sistema captura únicamente situaciones dinámicas extremas que se presentan raramente en el comportamiento real de los

[2]del inglés: eigenfrequencies

[3]Aunque estrictamente hablando este método es sólo válido para sistemas lineales, puede ser extendido sin demasiado error a cualquier tipo de sistemas.

[4]Señal que adquiere sólo dos valores (normalmente 0 y 1) que conmutan de manera aleatoria.

sistemas. Deben incluirse en la estrategia de excitación, por decirlo de alguna manera, situaciones intermedias que permitan al razonador inductivo interpolar entre ellos, y capturar así la gama completa de estados del sistema.

Una vez obtenida la información de la dinámica del sistema como un conjunto de datos muestreados, ésta será almacenada en una matriz que denominaremos *matriz primitiva de datos* o simplemente *matriz primitiva*, en la que las columnas serán las diferentes variables y los renglones los registros de esas variables a diferentes instantes de tiempo, obtenidos con un intervalo de muestreo δt.

4.2.2 Codificación Difusa

El siguiente paso en la construcción de un modelo cualitativo es convertir las variables cuantitativas en variables cualitativas. Las variables cualitativas son variables que asumen valores cualitativos. En el capítulo 2 nos referimos a las variables y a los valores cualitativos como variables y valores lingüísticos. El término cualitativo es un término más general para denotar la información no numérica. El término lingüístico, se refiere más bien al conocimiento cualitativo verbalizado por un ser humano, el cuál además de capturar correctamente la imprecisión de la información, suele introducir un cierto componente de sujetividad imposible de representar. Aquí solamente utilizaremos el término lingüístico cuando la fuente del conocimiento cualitativo provenga de un ser humano. Mientras que las variables de los modelos cuantitativos describen comportamientos mediante *trayectorias*, se dice que las variables de los modelos cualitativos describen comportamientos *episódicos*. La matriz primitiva de datos puede verse como una compilación de trayectorias; la operación de codificación difusa convierte las trayectorias en episodios.

La función de codificación del FIR nos permitirá convertir una variable cuantitativa en una variable cualitativa. En general alguna información se pierde durante el proceso de convertir una variable cuantitativa en una variable cualitativa. Obviamente, una temperatura de 30^oC contiene más información que el valor cualitativo definido por el valor de **clase** *calurosa*. Utilizando el formalismo de conjuntos difusos, explicado previamente en la sección 2.5, la temperatura de 30^oC puede representarse cualitativamente mediante el par de valores **clase/pertenencia**, por ejemplo *calurosa/0.8* con una pérdida de información menor. En la sección 2.7, definimos este proceso bajo

el nombre de fuzificación. La razón de introducir el nuevo término de *codificación difusa*, es con el objetivo de diferenciarlo del término introducido anteriormente, ya que, aunque son muy parecidos, no son exactamente iguales.

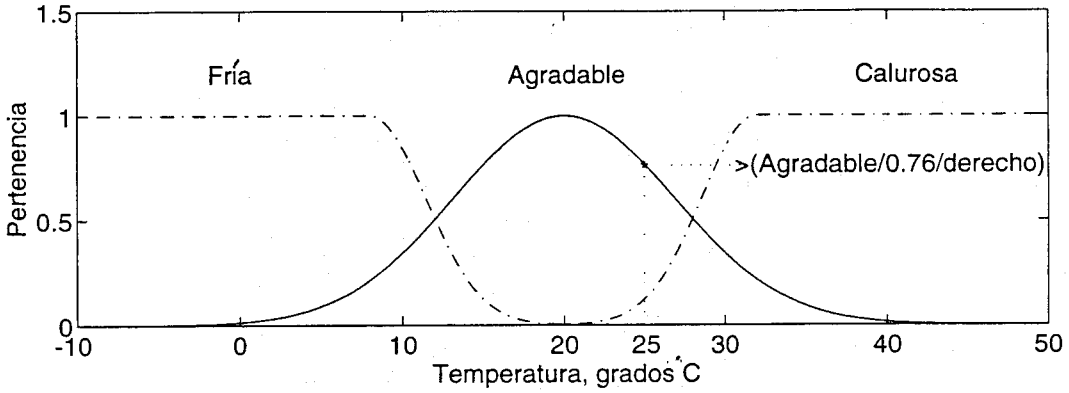


Figura 4.1: Funciones de pertenencia de la variable *Temperatura*

La figura 4.1 muestra la codificación difusa de la variable "Temperatura". Por ejemplo, un valor de la variable temperatura de $25°C$ es codificado como un valor cualitativo *agradable* con una función de pertenencia de 0.76 de igual manera que la operación de fuzificación, pero la codificación difusa incluye un atributo más, el valor de *lado* que, en este ejemplo, es *derecho*. El atributo lado indica si el valor cuantitativo en proceso de conversión se encuentra a la izquierda, a la derecha o en el máximo de la función de pertenencia. En esta forma la función de codificación difusa convertirá cada valor cuantitativo en el trío de atributos **clase/pertenencia/lado**. En el ejemplo de la figura 4.1 *agradable/0.76/derecho*. Obviamente, el trío cualitativo contiene la misma información que la variable cuantitativa original. El valor cuantitativo puede ser regenerado de manera precisa, es decir, sin pérdida de información, a partir del triple cualitativo. El hecho de mantener un atributo más en nuestra representación de conjuntos difusos no impide que los elementos teoricos presentados en las secciones 2.5 y 2.6 sigan siendo válidos para el desarrollo del FIR que se describe en este capítulo.

El lector puede preguntarse: ¿Cuál es la razón de transformar una representación cuantitativa en una cualitativa en la que no se pierda información?. La respuesta plena a esta pregunta es precisamente el objetivo de este capítulo y por ahora sólo daremos una respuesta muy general a la pregunta. En principio debemos de reconocer que no es un requisito de la transformación de la información cuantitativa→cualitativa

el perder información. Por supuesto que, a medida que el proceso de razonamiento inductivo generaliza, tiende a perder especificidad en la información que maneja. Se intercambia precisión por generalidad. La cuestión a resaltar es que no es menester *olvidar* lo que sabemos. La capacidad de generalizar (o abstraer lo que es lo mismo) nos permite tomar decisiones aún cuando no poseemos un conocimiento completo de la situación, pero no se debe volver al revés la lógica de este sentido. No debemos olvidar lo que ya sabemos para poder generalizar. Además, el proceso de generalización nos permite tomar decisiones, el de *recordar* nos ayuda a llevarlas a cabo. El primero nos da *conocimiento*, el segundo *habilidad*. Como se verá más adelante, este tipo de representación nos permitirá mantener compatibilidad con la teoría de conjuntos difusos, a la vez que nos capacita para utilizar algoritmos basados en el reconocimiento de patrones.

Dos tipos de funciones de pertenencia han sido definidas en SAPS-II: gausianas y triangulares. Prácticamente en todas las aplicaciones desarrolladas hasta ahora se han utilizado las funciones gausianas de la forma que describiremos a continuación. Se han realizado algunas pruebas comparativas para determinar el efecto de utilizar unas u otras. Los resultados parecen mostrar que, bajo nuestra metodología, es prácticamente indistinto cuál de las dos se utilice. La razón de usar las funciones gausianas, aunque son mś complejas computacionalmente hablando, es histórica: la implementación de las funciones triangulares en SAPS-II es mucho más reciente. Se espera que, en el futuro, este tipo de funciones de pertenencia se usará más frecuentemente.

Las funciones de pertenencia gausianas pueden ser fácilmente calculadas mediante la ecuación:

$$Memb_i = \exp(-\tau_i \cdot (x - \mu_i)^2) \tag{4.3}$$

donde x es la variable continua a ser codificada, i es la $i^{\text{ésima}}$ clase del conjunto difuso F, μ_i es la media algebraica entre dos *valores límite* ("landmarks") vecinos, y τ_i es determinada de tal forma que la función de pertenencia, $Memb_i$, alcance el valor de 0.5 en los valores límite. La selección del punto de intersección 0.5 entre dos funciones de pertenencia vecinas es relativamente frecuente en las aplicaciones de sistemas difusos. Como puede verse, los conjuntos difusos definidos en SAPS-II son conjuntos *normalizados*, ya que siempre al menos uno de sus elementos (la media en nuestro caso) alcanza el valor máximo posible (1.0 en nuestro caso) de la función de pertenencia. Otra característica importante de los

conjuntos difusos de SAPS-II es que de hecho son conjuntos difusos del tipo *alpha-cut* [¿¿¿ref¡¡¡] tales que $F_\alpha = \{x \in X \mid Memb_{F,i}(x) \geq \alpha, \forall i$. La manera de codificar la información difusa en SAPS-II y la forma de llevar a cabo la inferencia nos permite utilizar esta peculiaridad ya que existe muy poca pérdida de información en el proceso.

La siguiente cuestión a definir es cuántos niveles o clases deben de usarse para describir cada variable, y en dónde deben de ubicarse los valores límite que determinen donde termina una clase y comienza la otra.

A partir de consideraciones estadísticas, se sabe que en cualquier análisis de clases, cada estado discreto válido debe ser registrado por lo menos cinco veces [Law 90]. Así, existe una relación entre el número de estados válidos y el número aproximado de puntos que deben ser registrados para soportar al modelado:

$$n_{\text{reg}} \geq 5 \cdot n_{\text{val}} = 5 \cdot \prod_{\forall i} k_i \tag{4.4}$$

donde n_{reg} denota el número total de registros, es decir, el número total de puntos muestreados para cada variable, n_{val} denota el número total de estados válidos distintos, i es un índice que itera sobre el número de variables, y k_i el número total de niveles definido para la $i^{\text{ésima}}$ variable. Como el número de variables se conoce normalmente, y el número de registros está frecuentemente predeterminado, entonces, el número mínimo de niveles, n_{niveles}, para el total de las variables puede encontrarse mediante la siguiente ecuación:

$$n_{\text{niveles}} = \text{round}\ (\ \sqrt[n_{\text{var}}]{n_{\text{reg}}/5}\ ) \tag{4.5}$$

suponiendo que todas las variables son clasificadas en el mismo número de niveles. Por razones de simetría es preferido un número impar de niveles. Estados anormales ('muy bajo', 'muy alto' y 'bajo', 'alto') son dispuestos simétricamente con respecto al estado normal ('medio').

El número de niveles seleccionado para cada variable es un factor muy importante por varias razones. Este número afecta directamente la complejidad computacional de la etapa de inferencia. Usualmente los controladores difusos requieren entre siete y trece niveles para cada variable [Aliev 92], [Maiers 85], [Wu 92]. Una búsqueda exhaustiva en tal espacio multi–dimensional discreto puede resultar muy costoso, por lo que el número de niveles debe ser tan reducido como sea posible, para permitir que la velocidad de la optimización pueda ser suficientemente

rápida. En [Mugic 93] ha sido probado que la técnica de inferencia difusa utilizada por el FIR, permite reducir el número de niveles, usualmente a entre tres y cinco, un número confirmado por varias aplicaciones prácticas [deAlb 93a,93b], [Celli 91] y [Vesan 89].

El número de niveles de cada variable determina la *expresividad* y la *capacidad de predicción* del modelo cualitativo. La expresividad de un modelo es una medida del contenido de información que el modelo permite dar. La capacidad de predicción es una medida que determina la longitud del intervalo sobre el cuál se puede utilizar el modelo para predecir el comportamiento futuro del sistema. Si todas las variables se codifican en un solo nivel, el modelo cualitativo exhibirá un solo estado válido. A este modelo se le conoce como el *modelo nulo*. El modelo nulo predice perfectamente el comportamiento futuro (cualitativamente hablando) del sistema sobre un intervalo de tiempo de longitud infinito (dentro del marco de referencia de la resolución del modelo). Sin embargo, esta predicción no es de ninguna utilidad, pues siempre se predecirá el mismo estado. Así, el modelo nulo se caracteriza por una capacidad de predicción infinita y una expresividad cero. Por el contrario, si codificamos cada variable en un número de niveles muy grande, el modelo exhibirá una infinidad de estados válidos. La expresividad (o resolución) de tal modelo será excelente. Cada estado contiene una gran cantidad de información respecto del sistema real. Sin embargo, la capacidad de predicción en este último caso será muy pobre o nula a menos que el número de datos registrados sea mucho mayor, lo cual resulta imposible por el costo computacional de almacenamiento. Cualquier algoritmo que resuelva el número de niveles, deberá aceptar un compromiso entre ambos factores.

Cuando el número de niveles de cada variable ha sido seleccionado, el valor límite que separa dos regiones adyacentes debe ser determinado. Existen varias formas para encontrar un conjunto de valores límite significativo. A través de los casos de aplicación en donde se ha experimentado, la forma más efectiva de hacerlo está basada en la idea de que la expresividad del modelo (o contenido de información) será máxima si cada nivel es observado con la misma frecuencia. Con el objetivo de encontrar un conjunto de valores límite para cada variable que distribuya los valores observados de las trayectorias de igual manera entre los distintos niveles, usaremos el procedimiento siguiente:

(a) Ordenar el vector de valores de cada variable de manera ascendente.

(b) Dividir el vector de valores de cada variable en n_{niveles} segmentos de igual longitud (mismo número de valores).

(c) Calcular cada uno de los $n_{\text{niveles}} - 1$ valores límite de una variable como la media aritmética entre el valor final de un segmento y el valor inicial del segmento adyacente siguiente.

Una vez que el comportamiento de las trayectorias cuantitativas (almacenadas en la matriz primitiva de datos) ha sido convertido en episodios cualitativos, el resultado se almacena en una matriz primitiva de datos *cualitativa* que se llama la *matriz de comportamiento episódico*. La matriz primitiva de comportamiento episódico, compuesta a su vez de tres matrices paralelas, almacena los triples resultantes del proceso de codificación difusa. La primera matriz almacena los valores de clase. Los valores de clase se representan en SAPS–II mediante un conjunto de enteros positivos, por ejemplo $\{\ 1, 2, 3, 4, 5\ \}$ correspondientes con el número de niveles asignados a cada variable. La segunda matriz almacena el correspondiente valor de pertenencia (un valor real entre 0.0 y 1.0). La tercera matriz almacena el valor de lado mediante los símbolos $\{\ -1, 0, +1\ \}$ para representar la izquierda, el centro o la derecha del punto donde la función de pertenencia alcanza el valor máximo.

4.2.3 Causalidad Dinámica Cualitativa

En este punto el comportamiento de las trayectorias cuantitativas ha sido codificado y se encuentra disponible para la etapa de modelado. Se ha asumido que las entradas y salidas del sistema real pueden ser medidas y que son conocidas. El comportamiento de las trayectorias cuantitativas (la matriz primitiva de datos) puede ser separado en un conjunto de trayectorias de entrada, u_i, concatenadas por la derecha con el conjunto de trayectorias de salida, y_i, tal y como se muestra en el siguiente ejemplo que consta de dos entradas y tres salidas:

$$\begin{array}{lc} \textit{tiempo} & \begin{array}{ccccc} u_1 & u_2 & y_1 & y_2 & y_3 \end{array} \\ \begin{array}{l} 0.0 \\ \delta t \\ 2 \cdot \delta t \\ 3 \cdot \delta t \\ \vdots \\ (n_{\text{reg}} - 1) \cdot \delta t \end{array} & \left(\begin{array}{ccccc} \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \ldots & \ldots & \ldots & \ldots & \ldots \end{array} \right) \end{array} \tag{4.6}$$

donde n_{reg} es el número de registros almacenados y δt es el intervalo de muestreo, tal y como fueron definidos anteriormente.

Par evitar posibles ambigüedades, definiremos que los términos *entrada(s)* y *salida(s)* cuando son usados sin ningún calificativo, siempre se refieren a las variables de entrada y salida del subsistema a ser modelado por el razonador cualitativo.

En el proceso de modelado, se desea descubrir las relaciones entre las variables codificadas de entrada y de salida (en forma de un autómata de estados finitos) que haga que las matrices de transición sean tan deterministas como sea posible. Si podemos encontrar un conjunto de relaciones de este tipo para cada variable de salida, entonces el comportamiento del sistema puede predecirse por medio de la iteración de las matrices de transición de estados. Entre más deterministas sean las matrices de transición de estados, mayor será la certeza en que la predicción resultante sea correcta.

Tomando el caso del subsistema de dos entradas y tres salidas anterior, un ejemplo de una posible relación (cualitativa) entre las variables cualitativas podría ser de la siguiente forma:

$$y_1(t) = \tilde{\mathrm{f}}(y_3(t-2\delta t), u_2(t-\delta t), y_1(t-\delta t), u_1(t)) \tag{4.7}$$

La ecuación (1.7) es representada en FIR como:

$$\begin{array}{l} {}_t\backslash^x \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} \begin{array}{ccccc} u_1 & u_2 & y_1 & y_2 & y_3 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & -1 \\ 0 & -2 & -3 & 0 & 0 \\ -4 & 0 & +1 & 0 & 0 \end{pmatrix} \end{array} \tag{4.8}$$

En FIR, a este tipo de representación de las relaciones entre las variables cualitativas se le define como una *máscara*. Los elementos negativos en la máscara se refieren a las m–entradas. M–entradas denotan argumentos de entrada a la relación funcional cualitativa. M–entradas pueden corresponder tanto a entradas como a salidas del subsistema a ser modelado cualitativamente y pueden haber sido registradas en tiempos diferentes. El ejemplo anterior contiene cuatro m–entradas. La secuencia de numeración no tiene ningún significado especial, simplemente suelen ser numeradas de izquierda a derecha y de arriba a abajo. El valor positivo se denota la m–salida. En el ejemplo, la primera m–entrada, i_1, corresponde a la variable de salida y_3 dos intervalos de muestreo anteriores: $y_3(t-2\delta t)$; la segunda m–entrada, i_2, se refiere a la variable de entrada u_2 un intervalo de muestreo en el pasado: $u_2(t-\delta t)$; de

manera similar para la tercera, i_3, y cuarta, i_4, m–entradas; la m–salida, o_1, corresponde con la variable y_1 en el tiempo presente.

Una máscara describe las interrelaciones dinámicas entre las variables cualitativas. Una máscara es una matriz que tiene el mismo número de columnas que la matriz de comportamiento episódico a la que se aplicará. El número de filas de la matriz de la máscara es conocido como la *profundidad* de la máscara. La profundidad de la máscara representa el intervalo de tiempo en el que las relaciones causales aún se verifican. Una máscara con profundidad p cubre un intervalo de tiempo Δt de longitud temporal:

$$\Delta t = (p-1) \cdot \delta t \tag{4.9}$$

La experiencia ha mostrado que la máscara debe cubrir al menos la constante de tiempo más larga, t_s, de la dinámica del sistema que se modela. De esta forma la profundidad de la máscara puede determinarse como:

$$p = \text{round}\,(\frac{2 \cdot t_s}{t_{rápida}}) + 1, \quad \text{ó} \tag{4.10}$$

$$= \text{round}\,(\frac{\Delta t}{\delta t}) + 1 \tag{4.11}$$

siempre y cuando este cociente no sea mucho mayor de 3 o 4. Si esto no se cumple, el razonador inductivo no trabajará muy bien, ya que el esfuerzo computacional crecerá factorialmente con el tamaño de la máscara. El trabajo con múltiples frecuencias en FIR es un área de investigación abierta y de gran importancia en el campo de control. Se enseñará más adelante en la tesis como atacamos este problema.

La máscara es usada para *aplanar* el tiempo, convirtiendo las relaciones dinámicas en relaciones estáticas. La máscara se desplaza de arriba a abajo, fila por fila, sobre la matriz de comportamiento episódico. En cada desplazamiento se extraen las m–entradas y la m–salida correspondientes con la máscara, y se escriben de manera contigua, una tras otra, como una fila de una nueva matriz que llamaremos *matriz de entrada/salida*. La figura 4.2 ilustra este proceso.

Cada fila de la matriz de entrada/salida representa un *estado* del sistema. Cada de estos estados estado está compuesto de un *estado de entrada* y un *estado de salida*. El estado de entrada corresponde con el vector de

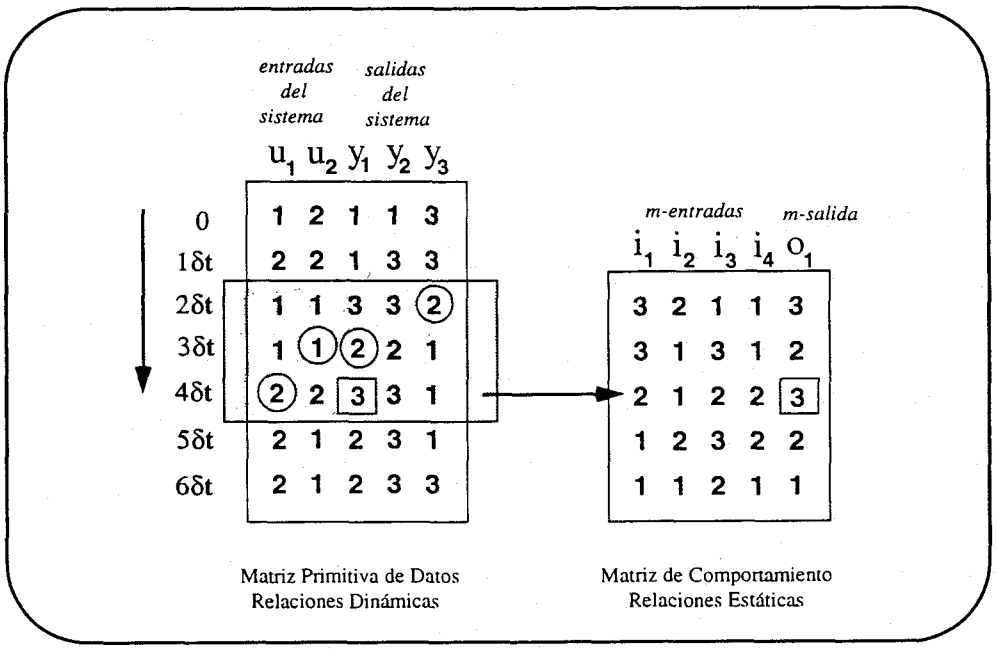


Figura 4.2: Construcción de la Matriz de entrada/salida

valores de todas las m–entradas de la máscara, colocadas al principio del estado; el estado de salida corresponde con la m–salida de la máscara, como el valor dispuesto en la posición final del estado. El conjunto de todos los estados diferentes presentes en la matriz de entrada/salida es conocido como el conjunto de *estados válidos* del modelo cualitativo.

4.2.4 Máscara Óptima

En el apartado anterior se ha utilizado una máscara para representar las relaciones causales, temporales y espaciales, del sistema en estudio con el objetivo de construir un modelo cualitativo. Sin embargo, antes de continuar debe darse un paso atrás para explicar el proceso de selección de la máscara que nos permita construir el mejor modelo cualitativo posible. ¿Cómo podemos encontrar, entre el conjunto de todas las máscaras posibles aquella que represente la matriz de transición de

estados más determinista? Esa máscara optimizará la capacidad de predicción del modelo.

En FIR, el concepto de una *matriz de máscaras candidatas* ha sido introducido. Una matriz de máscaras candidatas es un ensamble de todas las posibles máscaras a partir de las cuales será seleccionada la mejor mediante un mecanismo de búsqueda. En el estado actual de SAPS–II esta búsqueda es exhaustiva y ha permitido resolver los problemas prácticos planteados hasta ahora. Sin embargo esto es poco eficiente y debe ser corregido. Actualmente se trabaja en la solución de este aspecto utilizando estrategias de búsqueda heurísticas. La matriz de máscaras candidatas contiene elementos -1 en donde la máscara tiene una potencial m–entrada, un elemento $+1$ donde la máscara indica cual es la m–salida, y elementos 0 para indicar relaciones prohibidas. Así, una buena matriz de máscaras candidatas para el ejemplo de cinco variables presentado anteriormente para la variable de salida y_1 puede ser:

$$
\begin{array}{c}
{}_t\backslash^x \\
t-2\delta t \\
t-\delta t \\
t
\end{array}
\begin{array}{c}
\begin{array}{ccccc} u_1 & u_2 & y_1 & y_2 & y_3 \end{array} \\
\begin{pmatrix} -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & +1 & 0 & 0 \end{pmatrix}
\end{array}
\tag{4.12}
$$

El conocimiento *a priori* de las relaciones causales o temporales del sistema puede ser incluido en la matriz de máscaras candidatas colocando o eliminando elementos -1 y 0. Por ejemplo, los elementos 0 de la matriz anterior indican que conocemos que la variable y_1 debe ser encontrada antes de las variables y_2 y y_3 prohibiendo su relación causal en el tiempo t. Si no conocemos nada del sistema podemos permitir todas las posiciones, con la lógica excepción de la salida, colocando elementos -1 en toda la matriz. Aunque esto aumentará el costo computacional considerablemente, permitirá que el FIR encuentre de manera automática las variable relevantes. Ya que este es un proceso que se realiza fuera de línea, no resulta tan crítica la falta de optimalidad en la búsqueda de la máscara óptima, siempre y cuando el problema no se convierta en intratable por el tamaño y la complejidad de la máscara.

En SAPS–II, la rutina *foptmask* determina la máscara óptima partiendo de la matriz primitiva de datos cualitativa, una matriz de máscaras candidatas y un parámetro que limita la complejidad máxima de la máscara, es decir, el mayor número de elementos distintos de cero que la máscara óptima debe contener. *Foptmask* busca a través de todas las máscaras válidas de complejidad dos (todas las máscaras con una

m–entrada y la m–salida) y encuentra la mejor; luego procede con la búsqueda a través de todas las máscaras válidas de complejidad tres (todas las máscaras con dos m–entradas y la m–salida) seleccionando la mejor; el proceso continua hasta que se alcanza la complejidad máxima permitida. En todos los ejemplos prácticos, la calidad de la máscara ha crecido a medida que la complejidad aumenta, entonces un máximo es alcanzado, a partir del cual decae rápidamente. Una estimación sencilla para determinar la complejidad máxima de la máscara candidata es utilizar aproximadamente el mismo número de variables que se han incluido en la matriz primitiva de datos. En el caso del ejemplo anterior, un buen valor será cinco o seis.

Cada una de las posibles máscaras deberá compararse con las otras de acuerdo al mérito potencial que tenga. La optimalidad de la máscara se evalúa con respecto a la maximización de su poder predictivo.

Para determinar la incertidumbre asociada con la predicción de un estado de salida específico dado cualquier estado de entrada válido, FIR utiliza la medida de entropía de Shannon. La entropía relativa de Shannon para un estado de entrada es calculado a partir de la ecuación:

$$H_i = \sum_{\forall o} \mathrm{p}(o|i) \cdot \log_2 \mathrm{p}(o|i) \tag{4.13}$$

donde $\mathrm{p}(o|i)$ es la probabilidad condicional de que un particular estado de salida, (m–salida) ocurra, dado que el estado de entrada i (m–entrada) ha ocurrido. El término probabilidad es usado en un sentido estadístico más bien que en un sentido estrictamente probabilístico. Denota el cociente de la frecuencia con que se observa un particular estado dividido por la frecuencia más alta posible de que ese estado sea observado.

La entropía total de la máscara es calculada como la suma:

$$H_m = -\sum_{\forall i} \mathrm{p}(i) \cdot H_i \tag{4.14}$$

donde $\mathrm{p}(i)$ es la probabilidad de que el estado de entrada ocurra. La entropía más alta posible $H_{\max}$ es obtenida cuando todas las probabilidades son iguales; una entropía cero se encuentra cuando las interrelaciones de entrada/salida son totalmente deterministas.

Una reducción de la entropía total normalizada H_r es definida como:

$$H_r = 1.0 - \frac{H_m}{H_{\max}} \tag{4.15}$$

H_r es obviamente un número real en el rango entre 0.0 y 1.0, y en donde los valores altos usualmente indican un aumento del poder de predicción. La máscara óptima entre un conjunto de máscaras candidatas se define como aquella con la mayor reducción de entropía que implica tener el mayor poder de predicción.

El valor de pertenencia asociado con el valor de una variable cualitativa (difusa) es una *medida de confianza*. En el cálculo de la matriz de entrada/salida, el valor de confianza de cada estado cualitativo (una fila en la matriz de entrada/salida), puede ser calculado utilizando la operación de composición de intersección, según la definición 2.10, e interpretando el operador $*$ como la norma triangular min. Así, en SAPS–II el valor de pertenencia conjunta de una fila de la matriz de entrada/salida está dada como:

$$Memb_{\text{joint}} = \bigcap_{\forall i} Memb_i = \inf_{\forall i}(Memb_i) \stackrel{\text{def}}{=} \min_{\forall i}(Memb_i) \tag{4.16}$$

expresando la confianza que se tiene en un particular estado de la matriz de entrada/salida.

El *comportamiento básico* de la matriz de entrada/salida puede calcularse ahora. El comportamiento básico es definido como el conjunto ordenado de todos los estados distintos observados, junto con la medida de confianza que se tiene de cada estado. Ya que cada estado se ha observado posiblemente más de una vez (en teoría alrededor de cinco veces), debemos de calcular la confianza para cada estado. Es lógico que la confianza de un determinado estado aumentará en la medida que más veces se haya observado. Más que contar la frecuencia de observación (como sería en el caso de una medida probabilística), la confianza en un estado particular será la confianza acumulada de las confianzas de cada observación de ese estado. Sin embargo para mantener la compatibilidad con la entropía de Shannon, que es una medida *probabilística* de la cantidad de información contenida, durante el cálculo de la máscara óptima, las confianzas acumuladas pueden ser transformadas en valores que pueden interpretarse ya sea como probabilidades condicionales o bien como conjuntos difusos obtenidos mediante una función de agregación de la familia de operadores OWA[5] definidos por Yager [Yager 93]. En esta forma, la confianza de todos los estados que contengan el mismo patrón de entrada será normalizada con respecto a su suma acumulada. Aunque la aplicación de la entropía de Shannon como una medida de confianza

[5]Ordered Weighted Averaging operator

ha probado tener validez aún en muchas áreas de aplicación, puede ser que no resulte ser siempre el mejor índice de medida de confianza. Así que otros índices [Klir 89], [Shafe 76] pueden reemplazar la entropía de Shannon en el cometido de estimar la incertidumbre asociada a un estado particular. No obstante, desde un punto de vista práctico, un número significativo de experimentos de simulación parece mostrar que la entropía de Shannon trabaja satisfactoriamente en contextos de medidas difusas.

Hay un problema más que debe ser resuelto respecto a la medida de calidad de una máscara. El tamaño de la matriz de entrada/salida se incrementará a medida que la complejidad de la máscara crece, y consecuentemente, el número de estados válidos crecerá rápidamente. Dado que el número de registros n_{reg} permanece constante, la frecuencia de observación de cada estado caerá rápidamente y con ella la capacidad de predicción del modelo. La medida de reducción de la entropía no toma en cuenta este problema. Con el incremento de la complejidad, H_r simplemente crecerá más y más. Muy pronto se llegará a la situación de que cada estado ha sido observado sólo una vez, lo que nos lleva a tener una matriz de transición de estados completamente determinista y H_r alcanzará el valor de 1.0. Sin embargo, la capacidad de predicción puede ser muy pobre ya que fácilmente (muy probablemente) arribaremos a un estado de entrada (m–entradas) que no se ha observado nunca anteriormente y la predicción se detendrá. Esta paradoja de un H_r alto y una capacidad predictiva baja debe ser resuelta.

Como fue mencionado antes, desde el punto de vista estadístico, cada estado debería de ser observado al menos cinco veces [Law 90]. Por lo tanto, para medir en qué grado esta suposición es satisfecha, introducimos un factor que nos permita conocer la calidad de medida total, al cual denotaremos como cociente de observación. Será calculado como:

$$R_o = \frac{5 \cdot n_{5\times} + 4 \cdot n_{4\times} + 3 \cdot n_{3\times} + 2 \cdot n_{2\times} + n_{1\times}}{5 \cdot n_{\text{val}}} \tag{4.17}$$

en donde:

n_{val} = número de estados de entrada válidos;
$n_{1\times}$ = número de estados de entrada observados sólo una vez;
$n_{2\times}$ = número de estados de entrada observados dos veces;
$n_{3\times}$ = número de estados de entrada observados tres veces;
$n_{4\times}$ = número de estados de entrada observados cuatro veces;
$n_{5\times}$ = número de estados de entrada observados cinco o más veces.

Si cada estado de entrada válido se ha observado al menos cinco veces, entonces R_o será igual a 1.0, y la medida de reducción de entropía anterior no se alterará. Si esto no ocurre, entonces R_o puede utilizarse como una medida de calidad para ajustar o corregir la medida de reducción de entropía.

La calidad total de la *calidad de la máscara*, Q_m, se define como el producto de la medida de reducción de entropía, H_r y el cociente de observación R_o:

$$Q_m = H_r \cdot R_o \tag{4.18}$$

La *máscara óptima* es la máscara que obtenga el mayor valor de Q_m.

En SAPS–II, la función *Foptmask* devuelve la mejor máscara encontrada en la optimización como la matriz: *Mask*; un vector fila que contiene la entropía de Shannon de la mejor máscara para cada una de las complejidades consideradas (máscaras subóptimas): H_m; otro vector fila que contiene las medidas de reducción de la entropía para cada una de las máscaras subóptimas: H_r; y un vector fila más con la lista de las medidas de calidad correspondientes a cada una de las máscaras subóptimas, Q_m. Finalmente, *Foptmask* también regresa la *matriz histórica de máscaras*, una matriz que consiste de la concatenación horizontal de todas las máscaras subóptimas. Una de estas máscaras es la máscara óptima, la cual, por razones de conveniencia, es regresada también en forma separada.

Una vez que la máscara óptima ha sido determinada, ésta puede ser aplicada a la matriz primitiva de datos cualitativos para producir una matriz de entrada/salida específica. La matriz entrada/salida puede ser ordenada de forma alfanumérica atendiendo a la secuencia de los valores del estado de entrada (m–entradas) de cada una de sus filas. El resultado de esta operación es llamado *matriz de comportamiento* del sistema. La matriz de comportamiento es una máquina de estados finitos. Para cada estado de entrada, se puede conocer cuál de los estados de salida será el

más probablemente observado. La matriz de comportamiento junto con la máscara óptima conforman el modelo cualitativo del sistema.

4.3 Simulación Cualitativa

Basándose en el modelo cualitativo, el proceso de predicción (simulación) es directo. La máscara es colocada en el segmento final de la matriz de datos primitiva; los valores de las m–entradas son leídos mediante la máscara; y la matriz de comportamiento es usada para determinar el siguiente valor de la m–salida, el cual es copiado a la matriz de datos primitiva. En la predicción difusa es esencial que, junto con el valor cualitativo, los valores de pertenencia y de lado sean también predichos. El proceso de predicción difusa encuentra un triple cualitativo que puede ser regenerado cuantitativamente cuando se desee.

En el proceso de predicción difusa, los valores de los triples cualitativos correspondientes al nuevo estado de entradas, son comparados con cada uno de los registros almacenados en la matriz de comportamiento que coincidan con el mismo patrón de valores de clase en el estado de entradas.

Para este propósito se realiza una operación de normalización. Esta operación es un tanto peculiar ya que se realiza en forma segmentada. La idea es proyectar cada una de las clases de las variables cualitativas en el intervalo $[0.0, 1.0]$ de forma que podamos realizar comparaciones compatibles para establecer la similitud que guardan los patrones de las variables de estado de entradas contenidas en la matriz de comportamiento y las recién generadas. Para cada conjunto difuso, representado por cada clase de cada variable, se encuentra el corte–α de 0.5, proyectándolo sobre el intervalo $[0.0, 1.0]$. De esta manera podemos realizar operaciones aritméticas compatibles. Las funciones de pertenencia, representadas genéricamente por la ecuación 1.3, se recalculan modificando el valor de τ_i, asumiendo una media μ_i de 0.5 y un ancho de banda[6] de 1.0. Con estas nuevas funciones de pertenencia podemos llevar a cabo una pseudoregeneración en el intervalo unitario $[0.0, 1.0]$. En SAPS–II, esta operación es realizada por la función "MAPE". La función aplicada en MAPE es:

$$psr_i = B \cdot \sqrt{\log Memb_i} + 0.5 \tag{4.19}$$

[6]Tal y como se definió en la página 49.

donde: $B = (4 \cdot \log 0.5)^{-1/2}$.

Para las clases especiales de los extremos, las funciones se calculan como sigue. Para el extremo izquierdo:

$$psr_i = C \cdot \sqrt{\log Memb_i} \tag{4.20}$$

con: $C = (\log 0.5)^{-1/2}$. Para el extremo derecho:

$$psr_i = 1 - C \cdot \sqrt{\log Memb_i} \tag{4.21}$$

La pseudoregeneración se aplica a cada uno de los *R*egistros de la **M**atriz de **C**omportamiento, en los valores del estado de entrada y en el valor de salida; el resultado se almacena en la matriz $PSRMC$ para las entradas y en la $PSRMC_{out}$ para las salidas. Esta pseudoregeneración es aplicada también en cada iteración de predicción a los N m–entradas del nuevo estado y son almacenados en el vector PSR. A continuación, como una medida de distancia, se calcula la $\mathcal{L}_2$ norma de la diferencia para cada una de las R filas de la matriz $PSRMC$ con el vector PSR mediante la siguiente ecuación:

$$d_j = \sqrt{\sum_{i=1}^{N} (PSR_i - PSRMC_{j,i})^2} \tag{4.22}$$

De los R registros, sólo los cinco con la $\mathcal{L}_2$ norma más pequeña serán conservados de manera ascendentemente ordenada. A este conjunto de registros le llamaremos los cinco vecinos más cercanos[7] (5NN)[8]. Sus características estarán almacenadas en: K, el número de vecinos que normalmente es cinco; el vector d_k, las distancias recién calculadas y donde nos refiramos a la del último de ellos (el más lejano) como d_{max} o d_K ; en la matriz $PS5NN_{k,i}$, los valores pseudoregenerados de las N m–entradas de la matriz de comportamiento y en $PC5NN_{out_k}$ el vector de valores de cada una de las K m–salidas.

La contribución de cada vecino a la estimación de la predicción del siguiente estado será en función de su cercanía y será expresada mediante pesos absolutos calculados como sigue:

$$w_{\mathrm{abs}_k} = (\frac{d_{\max} - d_k}{d_{\max}})^x \tag{4.23}$$

[7]Si no hubiera cinco vecinos, el método puede trabajar con un número menor.
[8]Del inglés: Five–Nearest–Neighbors.

en donde el índice k itera sobre los K vecinos más cercanos, y para cualquier l y m se cumple que: $l < m$; $d_l \leq d_m$; y $d_{\max} = d_K$; donde K normalmente tiene un valor de 5. El exponente x es un factor que gobierna la exactitud que deseamos en la dispersión de los patrones. Dicho de otra manera, entre más grande sea x, el peso de los vecinos más cercanos (primero o segundo) será aún mayor, utilizando los demás de una manera secundaria. Si se tienen elementos para saber que la información almacenada es muy precisa respecto al episodio que se predice (por ejemplo si sabemos que una trayectoria específica fue registrada), sabemos que la confianza en el primer o segundo vecino es muy grande y podemos informarlo al método mediante el parámetro x. Si por el contrario sabemos que nuestros patrones registrados representan aproximaciones no muy exactas del caso que deseamos simular, como el modelado de un sistema biomédico de un individuo similar del que registramos el comportamiento, un exponente x con valor 1.0 o aún menor permite aumentar el efecto de vecinos no tan cercanos de tal manera que la interpolación entre estados será más relajada.

Los pesos absolutos son números entre 0.0 y 1.0. Usando la suma de los K pesos absolutos:

$$s_w = \sum_{k=1}^{K} w_{\text{abs}_k} \tag{4.24}$$

es posible calcular los pesos relativos:

$$w_{\text{rel}_k} = \frac{w_{\text{abs}_k}}{s_w} \tag{4.25}$$

Los pesos relativos son, también, números entre 0.0 y 1.0, y cuya suma es igual a 1.0. De esta forma, los pesos relativos pueden ser interpretados como porcentajes. Usando esta idea, los valores del nuevo estado de salida, en el espacio normalizado, pueden ser calculados como una suma pesada de los valores de los estados de salida previamente observadas de los cinco vecinos más cercanos. De esta forma un estado cualitativo puede ser calculado como:

$$Estado_{out_{nuevo}} = \sum_{k=1}^{K} w_{rel_k} \cdot \frac{clase_{out_k} + PS5NNout_k}{K} \tag{4.26}$$

En particular para el valor de clase:

$$clase_{out_{nueva}} = IFIX(Estado_{out_{nuevo}}) \tag{4.27}$$

y para el valor de pertenencia pseudoregenerada:

$$ps5NN_{out_{nueva}} = Estado_{out_{nuevo}} - clase_{out_{nueva}} \tag{4.28}$$

Finalmente aplicando el inverso de la función de normalización segmentada, podemos obtener el valores de pertenencia y lado desnormalizados, lo cual es obtenido en SAPS–II con la función "REMAPE".

La función de predicción difusa usualmente dará una predicción más precisa que la función de predicción probabilística. Un estudio comparativo de los métodos de inferencia más comúnmente usados y el método de los cinco vecinos más cercanos que es detallado en [Mugic 93] es presentado de manera resumida en la siguiente sección. Este método permite recuperar señales de salida pseudo–continuas con una buena calidad utilizando la función *regenerate* de SAPS–II. Esto significa que también puede obtenerse la predicción de señales continuas en el tiempo [Celli 91a]. Debe de hacerse notar por último, que la operación de la función *regenerate* es justamente el proceso inverso de la función *recode*.

4.4 Un Caso de Estudio

4.4.1 El Experimento

Para ilustrar el uso y las capacidades de la metodología FIR, en esta sección desarrollaremos el modelado y simulación cualitativos de un sistema lineal con tres salidas y una entrada. El sistema a identificar será la abstracción cualitativa del siguiente sistema de ecuaciones diferenciales:

(4.29)

$$\dot{\mathbf{x}} = \mathbf{A}\cdot\mathbf{x} + \mathbf{b}\cdot u = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -2 & -3 & -4 \end{pmatrix}\cdot\mathbf{x} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\cdot u$$

(4.30)

$$\mathbf{y} \;=\; \mathbf{C}\cdot\mathbf{x}+\mathbf{d}\cdot u \;=\; \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\cdot\mathbf{x}+\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}\cdot u$$

El sistema ha sido implementado en Matlab. Con el objetivo de caracterizar el comportamiento del sistema se simularon cuantitativamente 900 segundos (300 pasos de muestreo) en el ambiente SAPS–II/Matlab, almacenándose posteriormente en una matriz primitiva de datos. Los primeros 270 pasos se utilizaron para la identificación del modelo cualitativo. La entrada del sistema fue excitada mediante señales de ruido binario por lo que la codificación difusa no será necesaria para la variable de entrada que será representada por dos niveles. La codificación difusa fue aplicada a las variables de salida utilizando tres clases. Los últimos 30 pasos de la simulación de tiempo–continuo serán usadas como etapa de simulación cualitativa para validar el modelo cualitativo.

El apéndice B presenta el conjunto de programas LINEAL#.M, desarrollados en Matlab, en donde puede seguirse con facilidad el procedimiento de modelado y simulación:

Lineal0 En este programa se generan los datos del sistema lineal de ecuaciones diferenciales. Las variables son muestreadas cada 3 segundos de acuerdo a su diagrama de Bode y tomando las consideraciones dadas al respecto en la sección anterior.

Lineal1 Una vez que se cuenta con el conjunto de datos muestreados y almacenados en la matriz primitiva de datos *meas*, este programa convierte los datos cuantitativos en información cualitativa utilizando la función *recode* de SAPS–II y los almacena en las tres matrices *raw1*, *Memb1* y *side1*. Puede verse que fueron seleccionadas 3 niveles para cada variable y la forma en que fueron encontrados los valores límite de cada una.

Lineal2 Seleccionando sólo 270 puntos del comportamiento episódico, se identifican las máscaras óptimas para cada variable de salida utilizando la función *foptmask* de SAPS–II. Es interesante ver que son seleccionadas tres máscaras para el modelo de cada variable. Aunque en este ejemplo particular no llegan a usarse la segunda y tercera máscara alternativa (subóptima) la intención de introducirlas es con el fin de mostrar el uso de modelos alternativos que pueden ser utilizados para superar una falta de predicción (la base de reglas no cuenta con el patrón de entradas que intenta predecir), para verificar la predicción (las predicciones de un

segundo y/o tercer modelo pueden ser usadas para confirmación) o para subsumir tres comportamientos 'parecidos'. Las máscaras óptimas encontradas que se almacenan en las variables *m1a*, *m2a* y *m3a* son:

$$\begin{array}{c} {}_t\backslash^x \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} \begin{array}{cccc} u_1 & y_1 & y_2 & y_3 \end{array} \\ \begin{pmatrix} -1 & 0 & 0 & -2 \\ -3 & 0 & 0 & 0 \\ -4 & +1 & 0 & 0 \end{pmatrix} \end{array} \tag{4.31}$$

para la primera variable,

$$\begin{array}{c} {}_t\backslash^x \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} \begin{array}{cccc} u_1 & y_1 & y_2 & y_3 \end{array} \\ \begin{pmatrix} -1 & 0 & 0 & -2 \\ -3 & 0 & 0 & 0 \\ -4 & 0 & +1 & 0 \end{pmatrix} \end{array} \tag{4.32}$$

para la segunda variable, y

$$\begin{array}{c} {}_t\backslash^x \\ t-2\delta t \\ t-\delta t \\ t \end{array} \begin{array}{c} \begin{array}{cccc} u_1 & y_1 & y_2 & y_3 \end{array} \\ \begin{pmatrix} -1 & -2 & 0 & 0 \\ -3 & 0 & 0 & 0 \\ -4 & 0 & 0 & +1 \end{pmatrix} \end{array} \tag{4.33}$$

para la tercera variable.

Lineal3 Si ahora aplicamos las máscaras a las tres matrices de datos cualitativos correspondientes a cada variable, podemos terminar de obtener el modelo cualitativo al 'aplanar' el comportamiento dinámico en un conjunto de reglas estáticas. Esto es realizado por las funciones *fiomodel2* y *fbehavior2*. El conjunto de reglas queda entonces almacenado en las variables matriciales *[b1a Mb1a sb1a]* para la primera salida, *[b2a Mb2a sb2a]* para la segunda y *[b2a Mb2a sb2a]* para la tercera.

Lineal4 Terminada la etapa de modelado, podemos pasar a la etapa de simulación. En este programa, se muestra el proceso de simulación cualitativa durante 30 intervalos (90 segundos) que **no** han sido observados previamente. EL resultado de este proceso son triples de variables cualitativas que se almacenan en variables matriciales, por ejemplo, para la primera variable en *[frcdat1 Mfrcdat1 sfrcdat1]*. Otro punto interesante mostrado en este programa es uno de los posibles usos de los modelos alternativos mencionados anteriormente cuando describimos la búsqueda de

máscaras. Observando el programa puede verse una llamada en cascada al la función *fforecast2* de SAPS–II. Si por alguna razón el modelo cualitativo óptimo no es capaz de predecir, dos modelos de respaldo pueden sustituirlo en ese punto en particular. Aunque quizás con un poco menos de precisión, el modelo alternativo produce una predicción y el proceso puede continuar. El resultado de la simulación cualitativa puede observarse también en el apéndice B, y puede constatarse que solamente ocurrió un error.

Lineal5 El paso final consiste en regenerar las variables cualitativas encontradas en el programa anterior. Esto es realizado por la función *regenerate* de SAPS-II. El resultado es almacenado en las variables *rvar11*, *rvar12* y *rvar13*, las cuales son graficadas y comparadas en la figura 4.3. La calidad de la predicción habla por sí misma.

4.4.2 Evaluación Comparativa

Con el objetivo de comparar la capacidad de FIR, y utilizando el experimento mostrado anteriormente, se realizó un estudio comparativo [Mugic 93] con los métodos de inferencia difusa[9] más comúnmente usados para la tarea de control. Los métodos comparados serán el método de la Media de los Máximos (MOM) y el Método del Centro de Área (COA) que fueron descritos con detalle en la sección 2.7.4. Estos métodos fueron codificados como métodos alternativos al algoritmo de los cinco vecinos más cercanos e integrados a la función de predicción *fforecast2* de SAPS-II.

El comportamiento del algoritmo de inferencia de la Media de los Máximos se muestra en la figura 4.4:

El comportamiento tan pobre del método de inferencia MOM es causado predominantemente por el pequeño número de niveles de discretización. Tres clases para cada variable de salida no dan suficiente capacidad de discriminación para permitir estimaciones decentes a la salida del proceso de defuzificación. Un aumento en el número de clases por variable podría resolver este problema, pero a expensas de requerir un número considerablemente mayor de registros, n_{reg}, para la identificación

[9]En la literatura es muy común encontrar que se refieren a los métodos de defuzificación como métodos de inferencia. La razón es que al utilizar siempre como base del proceso de inferencia la composición Sup–Estrella de Zadeh, los métodos suelen diferenciarse más bien por el algoritmo de defuzificación que usan y el cual termina nombrando la variante del método de inferencia.

del modelo. Con el número actual de discretizaciones por variable y manteniendo cinco observaciones para cada patrón, como recomienda la estadística, se requieren al menos de $5 \times 2 \times 3 \times 3 \times 3 = 270$ registros para la identificación del modelo cualitativo. Si se aumenta a siete el número de niveles por variable, como recomienda el método MOM, el número de registros deberá aumentarse a $5 \times 2 \times 7 \times 7 \times 7 = 3430$.

La capacidad de predicción del método de inferencia del Centro de Área es mostrada en la figura 4.5.

En esta figura resulta notorio que el funcionamiento del algoritmo de inferencia COA es sustancialmente mejor que el presentado por el MOM. Esta observación puede ser explicada por el hecho de que el algoritmo COA toma en cuenta el perfil de la curva de la función de pertenencia, mientras que el MOM trabaja únicamente con los valores máximos. Además, el método COA intenta capturar las diferencias entre las funciones de pertenencia, utilizando un cierto tipo de asignación de pesos, lo cual es ignorado por completo por el MOM que funciona mejor con funciones de pertenencia iguales. Desafortunadamente el método COA resulta muy costoso computacionalmente, aunque, como hicimos notar en la sección 2.7.4, existen algunas alternativas que lo abaratan no sin afectar la precisión. Para efectos del estudio comparativo presentado aquí el método COA fue implementado en SAPS–II prefiriendo la representación continua a la solución discreta mediante un algoritmo trapezoidal de integración con el objetivo de evitar perder precisión numérica.

El error presentado por los diferentes métodos de inferencia difusa es graficado conjuntamente en la figura 4.6.

El éxito del algoritmo 5NN puede ser explicado por el hecho de que es fuertemente dirigido por datos. En el caso de las técnicas del MOM y del COA, los datos medidos solamente son usados en la etapa de modelado cualitativo, es decir, en la identificación de una máscara óptima. Cuando la máscara, en nuestro caso, o la base de reglas, en el caso de los modelos tipo sistemas expertos, ha sido encontrada, los datos no vuelven a usarse más, y únicamente se preserva la información capturada por la función de pertenencia. En contraste con esto, el algoritmo de los 5NN explota el conocimiento disponible en su máxima extensión tanto en la etapa de modelado como en la de simulación. Esto tiene pleno sentido en el ámbito de razonamiento inductivo. ¿Por qué echar un conocimiento precioso cuando es requerido para diferentes propósitos y se encuentra tan a la mano?

Por otro lado, muchas aplicaciones de sistemas difusos están basadas

más en un modelado de tipo deductivo que de tipo inductivo. Las relaciones cualitativas entre las salidas y las entradas están determinadas en base a metaconocimiento, es decir, un entendimiento general de cómo el dispositivo o sistema a modelar se supone que funciona, en lugar de un conjunto de datos medidos. En éste tipo de aplicaciones, los métodos MOM y COA pueden seguir siendo usados, mientras que el método 5NN quedará descartado si no cuenta con el registro de datos que requiere.

4.5 Conclusiones

En esta sección se ha presentado una nueva metodología par el modelado de sistemas cualitativos que requieren ser acoplados con modelos o sistemas que requieren trabajar en forma cuantitativa. El núcleo de la metodología intenta preservar la información precisa de los registros medidos, a la vez que infiere un modelo cualitativo sobre las relaciones causales, tanto espaciales como temporales, que le permite generalizar sobre todos los comportamientos posibles del sistema que representa. Los puntos débiles de la metodología son principalmente dos: (1) como todas las técnicas inductivas, la riqueza de la información con la que trabaja, determinará por completo su capacidad de funcionamiento; (2) aunque es muy eficiente en cuanto a la velocidad de procesamiento y es capaz de seleccionar la mejor información disponible, requiere considerablemente más memoria secundaria que las metodologías difusas estándares.

Mediante el ejemplo de estudio se ha podido mostrar que la técnica de inferencia opera mejor que los métodos de inferencia (o defuzificación) más comúnmente usados en el contexto de las aplicaciones de razonamiento inductivo. Las principales ventajas de la metodología son: (1) trabaja excelentemente bien en el ámbito de operación guiado por datos; (2) en contraste con el resto de técnicas de inferencia inductiva, que requieren un número considerablemente mayor de niveles de discretización, el método de inferencia de los cinco vecinos más cercanos permite trabajar con un número pequeño de estados discretos; (3) es muy económico con respecto al número de crunching y un poco menos en el requerimiento de memoria; (4) el proceso de inferencia inductiva difusa se encuentra claramente separada del proceso de defuzificación, lo que le da la capacidad potencial (aún no explotada) de combinar su inferencia con otro tipo de conocimiento (por ejemplo de un experto o de un sistema supervisor) y de generar explicaciones del proceso.

Aunque el ejemplo discutido anteriormente es lineal, la selección fue hecha bajo consideraciones de sencillez y de ilustrar las características de la metodología para trabajar con razonadores inductivos múltiples. La linealidad de la aplicación no tiene ninguna influencia en el éxito de la técnica. Varias aplicaciones no lineales de simulación cualitativa para detección y diagnóstico de fallos, sistemas biomédicos, así como para el desarrollo de controladores difusos han sido exitosamente resueltas utilizando SAPS–II como implementación de FIR. Otras aplicaciones fuertemente no lineales, como un control hidráulico, el control de un barco de carga y otro de un brazo robot de tres grados de libertad, serán presentadas en los capítulos subsiguientes.

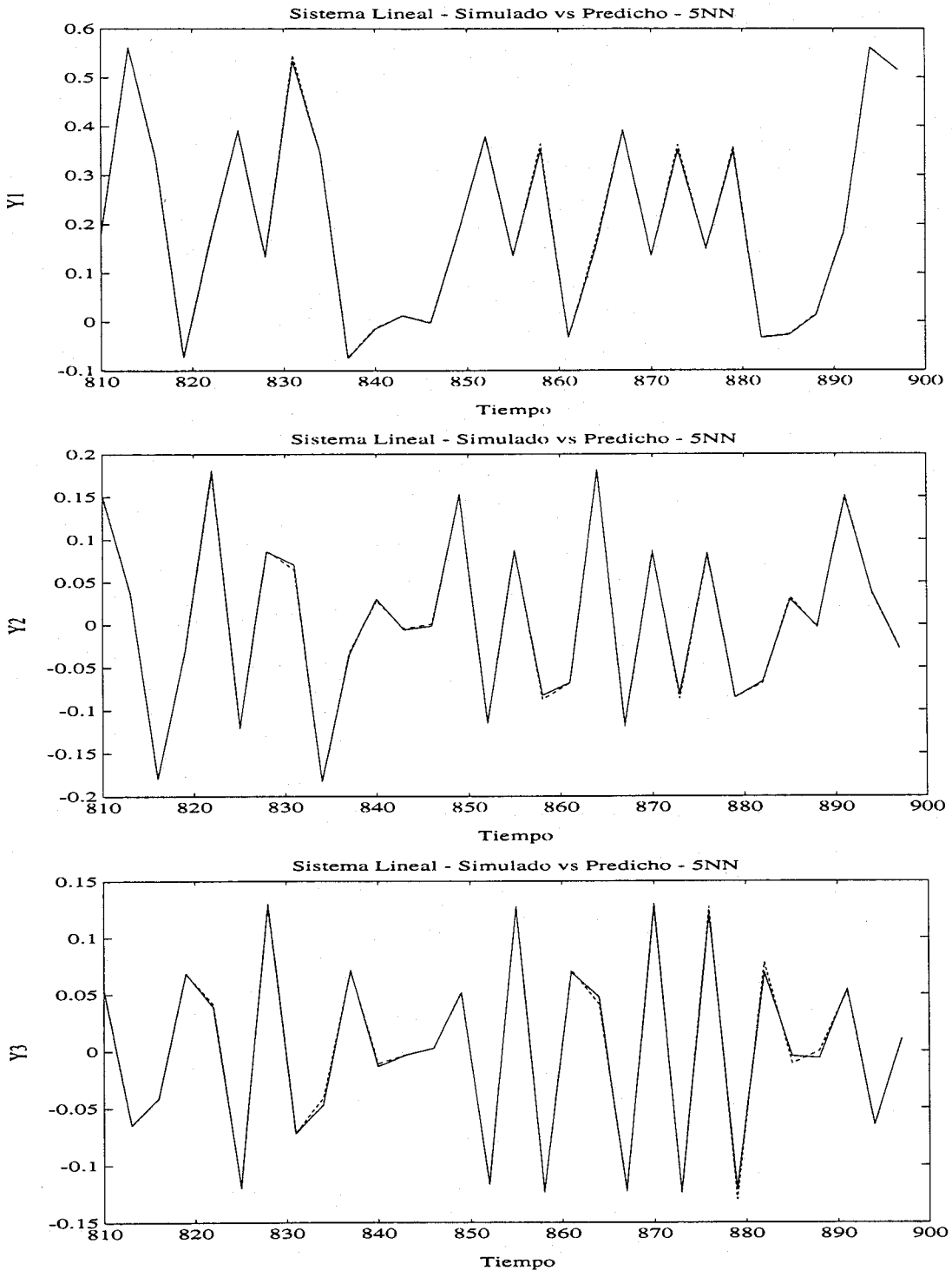


Figura 4.3: Predicción de las variables Y1, Y2 y Y3

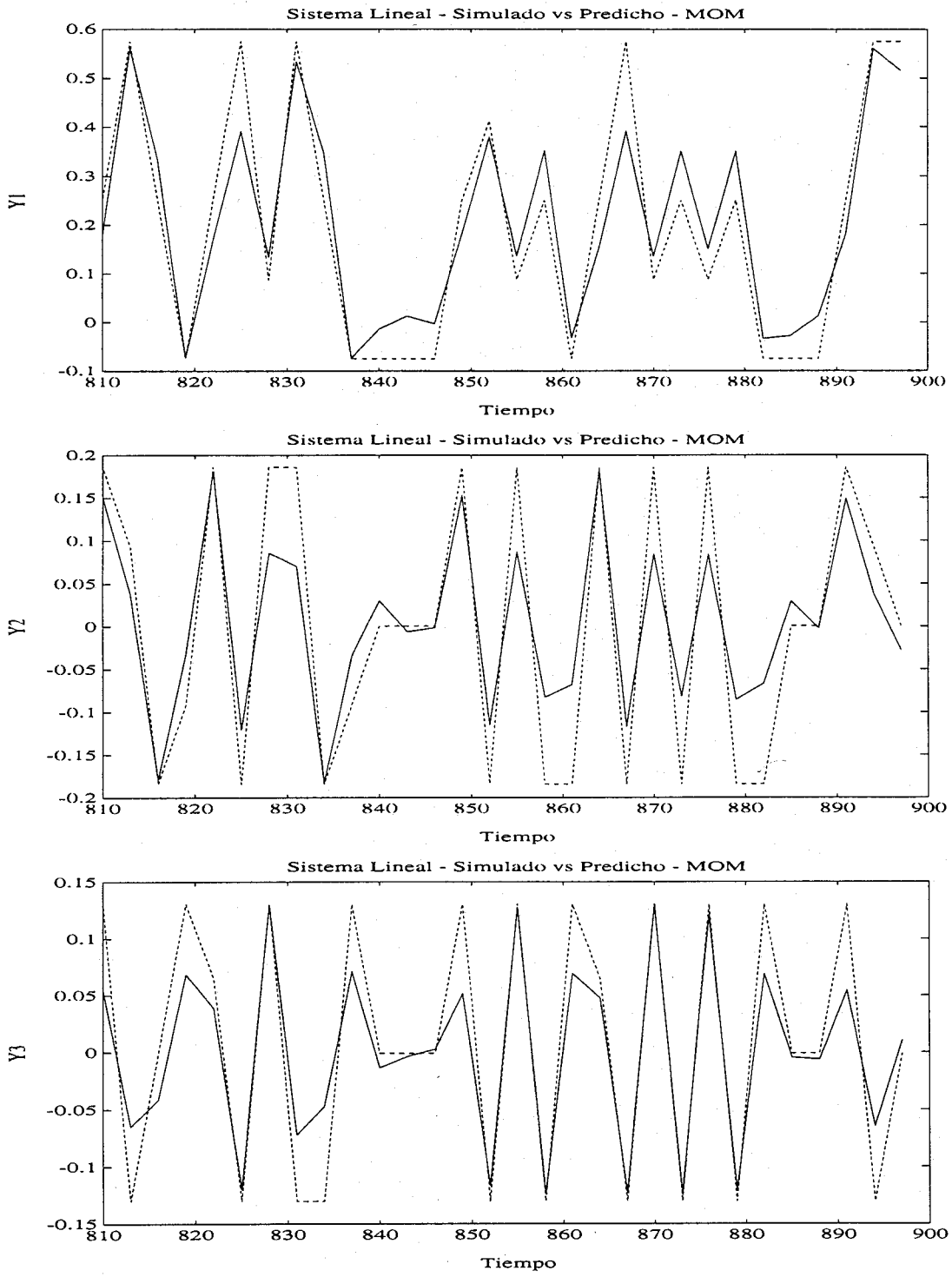


Figura 4.4: Capacidad de predicción del método de inferencia MOM.

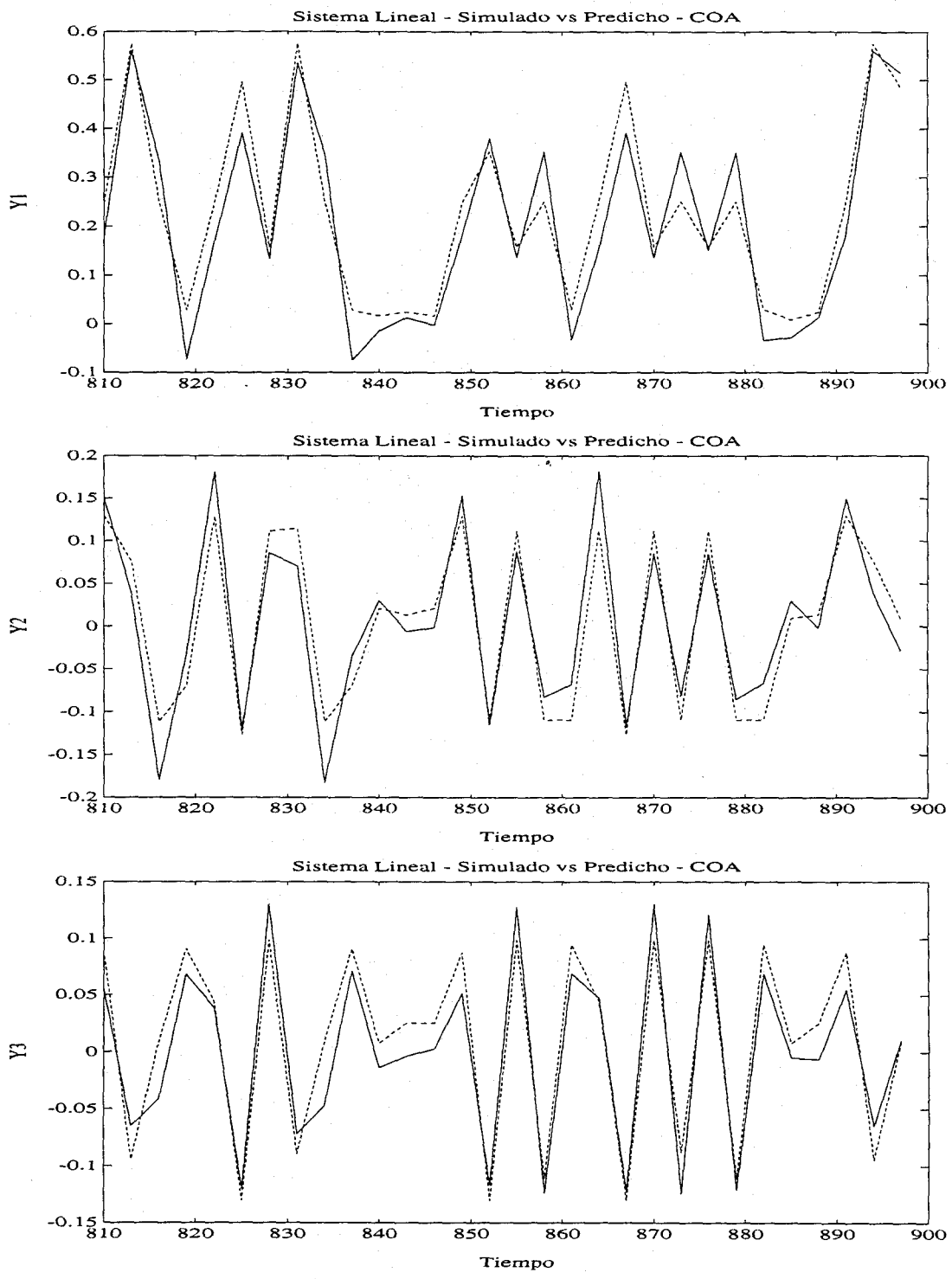


Figura 4.5: Capacidad de predicción del método de inferencia COA.

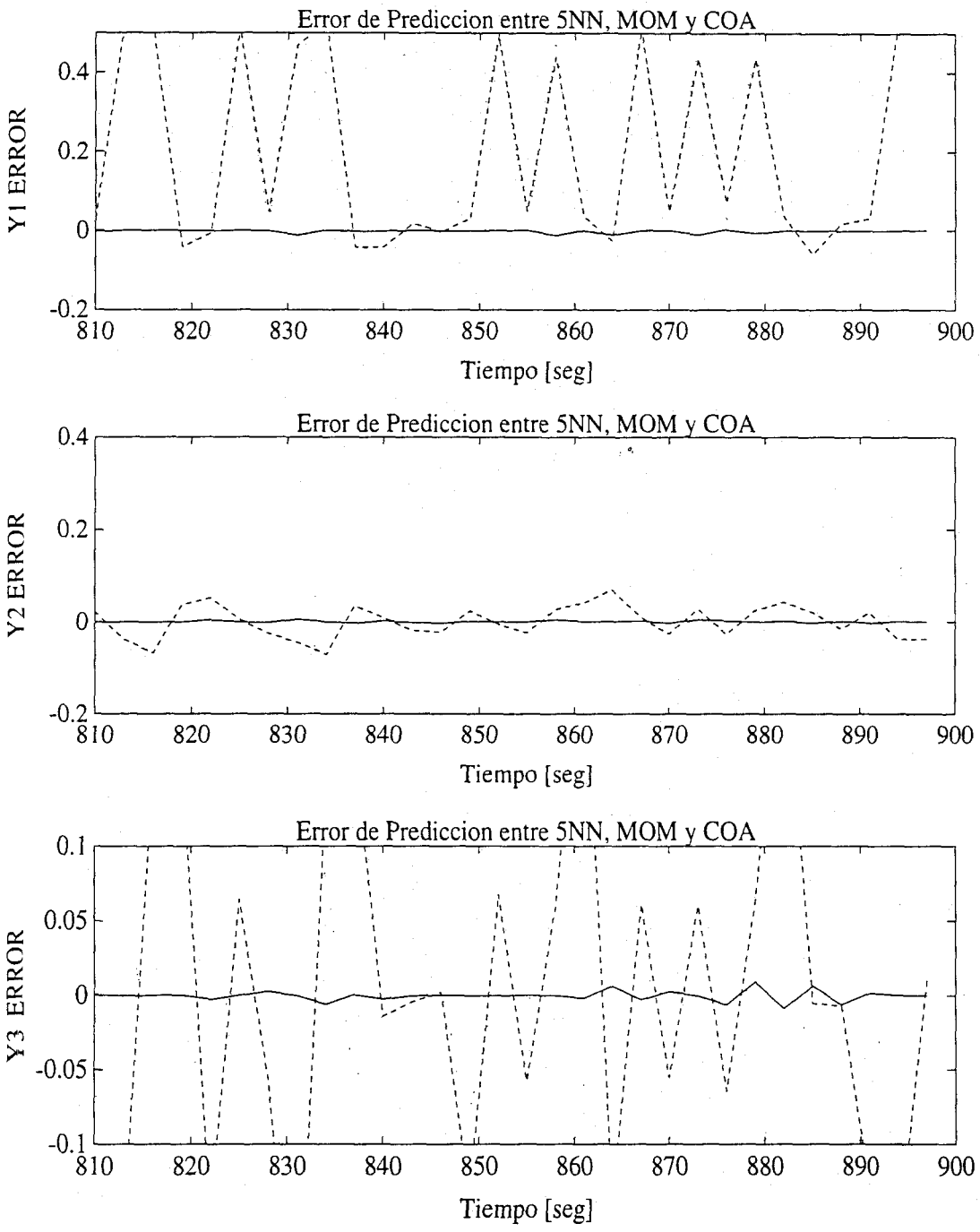


Figura 4.6: Comparación de error exhibido por los métodos MOM, COA y 5NN.

Capítulo 5

Simulación Mixta Cuantitativa/Cualitativa

5.1 Simulación Mixta: Objetivos y Problemas

El objetivo general de toda simulación, ya sea cuantitativa, cualitativa o mixta, es la predicción del comportamiento de un sistema. En el caso de los sistemas dinámicos, en donde la variable independiente es el tiempo, el comportamiento se describe mediante *trayectorias* cuando se trabaja con modelos cuantitativos, y mediante *episodios* cuando se trabaja con modelos cualitativos. En el primer caso, el manejo de la variable tiempo requiere ser cuantitativa necesariamente. En el segundo, el tiempo puede ser manejado tanto en forma cuantitativa como cualitativamente, dependiendo del propósito para el que se realiza la simulación. A su vez, puede ser que el interés en la simulación sea conocer todos los comportamientos factibles a los que un sistema puede arribar, o por el contrario, predecir solamente un comportamiento específico y único. Bajo estas consideraciones, las aplicaciones de la simulación mixta, cuantitativa cualitativa, pueden ser clasificados en varias familias de problemas a estudiar.

Como fue mostrado en el capítulo 3, el propósito de la mayoría de las aproximaciones de modelado cualitativo ha sido enumerar todos los posibles comportamientos episódicos de un sistema bajo todas las condiciones de operación factible. Esta es una cualidad deseable si, por ejemplo, lo que se desea es predecir en términos muy generales los posibles escenarios de comportamiento de un sistema que ha sufrido

un fallo, o si se desea explicar el por qué un sistema ha producido un comportamiento particular. Lamentablemente, como fue analizado anteriormente, la posibilidad de conseguir que estos sistemas operen razonablemente bien para sistemas de complejidad no trivial, aún parece lejana. La utilización de esquemas de modelado mixto podría ayudar a superar las principales limitaciones de este tipo de aproximaciones mediante la combinación con modelos cuantitativos que le permitan disminuir la vaguedad en el seguimiento de los episodios. Algunos de los intentos de reconducir las aproximaciones del tipo de la física cualitativa empiezan a considerar que su integración en entornos de simulación mixta, tales como la interpretación de resultados de simulación, continúa.

Si, en vez de buscar todos los posibles episodios, la simulación se orienta a buscar uno solo, por ejemplo el más probable, la situación de indecisión se relaja, aunque en otro sentido puede complicarse. Mantener el seguimiento de un único episodio implica partir de unas condiciones particulares y deducir entre los posibles comportamientos cuál es el correcto. La vaguedad implícita en la estructura cualitativa de los estados hace muy difícil cumplir esta tarea. Algunos sistemas expertos son ejemplos típicos de sistemas de episodios únicos. La simulación del modelo cualitativo va desarrollándose a medida que la máquina de inferencia predice la secuencia de episodios en función de la instanciación de variables. Debido a que el tiempo es una variable cualitativa, con frecuencia ocurre que no puede ser definido cuál es el orden correcto de eventos, y un comportamiento divergente puede presentarse provocando que la secuencia de cambios en los objetos representados sea imposible de predecirse. Esto es conocido como el "frame problem". Se ha intentado, sin mucho éxito aún, la formulación de lógicas temporales que permitan decidir qué elementos, en qué tiempo y bajo qué ordenamiento han sufrido los cambios y así encontrar un escenario final consistente. Bajo ciertas características, el uso de la simulación mixta puede ayudar para la resolución de este tipo de conflictos, por ejemplo, permitiendo que el cálculo de algunas variables se realice mediante modelos cuantitativos que acoten el grado de bifurcación de comportamientos episódicos. Los modelos mixtos pueden ejecutarse en un tiempo relativo (pero cuantitativo), y los resultados (las trayectorias cuantitativas) pueden utilizarse para decidir cuál es el escenario correcto, permitiendo que el predicado que esperaba resolución sea instanciado. En la realidad este caso de modelado mixto suele ser muy complejo y no ha podido ser sistematizado aún.

Los casos de comportamientos cuantitativos múltiples no pueden existir,

ya que, dadas unas condiciones específicas iniciales, solamente puede existir una única trayectoria cuantitativa como solución.

Por último y en el extremo opuesto, al caso de episodios múltiples, los comportamientos cualitativos y únicos son el dominio de la simulación cuantitativa pura. Su principal limitación es que no permiten manejar ningún tipo de imprecisión que no sea la de sus propios algoritmos de resolución numérica. Esto hace que resulten muy rígidos y que su aplicación sea reducida a aquellos casos en los que se cuenta con una información muy completa y precisa de la estructura del sistema a modelar, así como de la información de sus parámetros y constantes. El trabajo de ajuste de parámetros y de validación suele ser enorme. Si, por alguna razón en algún subsistema o componente del sistema, se pierde la capacidad de formalizarlo, la simulación mixta puede ser un factor clave para permitir aún construir modelos válidos que nos permitan predecir una trayectoria particular. Quizás es en este tipo de casos donde la utilización de la simulación mixta puede resultar más provechosa, y es precisamente en ellos donde se centrará nuestra investigación. Existen varias razones por las que se requiere incorporar modelos cualitativos en un sistema de esas características. Algunas de ellas pueden ser:

1. Los detalles cuantitativos respecto a un (sub)sistema no están disponibles. Por ejemplo, mientras que las propiedades mecánicas del funcionamiento del corazón humano son bien conocidas y pueden ser fácilmente descritas por medio de un modelo cuantitativo de ecuaciones diferenciales, los efectos que una gran variedad de sustancias químicas o los impulsos eléctricos provenientes del cerebro puedan ocasionar en el comportamiento del corazón, son entendidos solamente de una manera vaga y general, y resulta muy difícil cuantificarlos. Un modelo mixto puede ser utilizado para representar las partes del sistema que son bien comprendidas mediante modelos de ecuaciones diferenciales, mientras que aquellos aspectos o componentes de los que se sabe o se entiende poco, podrían ser descritos mediante modelos cualitativos. Los sistemas sociales, biológicos y biomédicos podrían beneficiarse mucho con la aplicación de la simulación mixta en sus sistemas de estudio.
2. Los detalles cuantitativos pueden limitar la robustez de un (sub)sistema para reaccionar a condiciones previamente no experimentadas. Por ejemplo, mientras que un piloto humano es incapaz de calcular mentalmente la trayectoria óptima de vuelo, él o ella pueden controlar el avión de una forma mucho más robusta que cualquier piloto automático moderno. La optimalidad puede ser

canjeada por robustez. Un controlador difuso es un buen ejemplo de un subsistema cualitativo diseñado para operar bajo diferentes tipos de condiciones experimentales de manera subóptima pero robusta. El modelo cuantitativo de la planta, o la planta misma, operará en un esquema mixto junto con el modelo cualitativo del controlador.

Sin embargo, existen una serie de aspectos incompatibles entre los subsistemas cualitativos y los subsistemas cuantitativos que deben ser resueltos antes de que la simulación mixta pueda realizarse, resultando cuestionable que la simulación mixta sea del todo factible bajo este contexto. ¿Cómo podría una simulación mixta manejar el hecho de que los subsistemas cualitativos traten la variable independiente, *el tiempo*, como una variable cuantitativa, mientras que los subsistema cuantitativos traten a la misma variable cuantitativamente? ¿Cuándo ocurre un evento cualitativo en términos de tiempo cuantitativo? ¿Cómo pueden los subsistemas cualitativos tomar en cuenta las condiciones experimentales explícitas que requieren los subsistemas cuantitativos?

En la sección siguiente, se intentará dar respuesta a estas interrogantes, presentando el marco de solución del FIR para construir un esquema de simulación mixta que permita integrar conjuntos de modelos cualitativos y modelos cuantitativos. A continuación se ilustrará la metodología de la simulación mixta utilizando como ejemplo un motor hidráulico controlado mediante una servoválvula.

5.2 Simulación Mixta en FIR

El desarrollo de la metodología de simulación mixta cuantitativa cualitativa, que se presenta en este capítulo, ha sido el resultado del trabajo realizado en la etapa inicial del desarrollo de tres tesis de doctorado de la UPC, que aunque partieron de este origen común, divergieron hacia objetivos de investigación sustancialmente diferentes. La primera tesis se enfocó al estudio y tratamiento de sistemas biomédicos [Nebot 94], la segunda desarrolla técnicas para el monitoreo de fallos y soporte de decisiones en sistemas de gran complejidad, mientras que la presente se avocó al diseño sistemático de controladores difusos. Aunque en diferente medida las tres tesis requirieron del uso de este tipo específico de simulación mixta como herramienta de soporte [Celli 94], no ha constituido el elemento central en ninguna de ellas.

Tomemos como punto de partida el sistema genérico que se muestra en la figura 5.1. Las cajas grises de esquinas rectas representan subsistemas

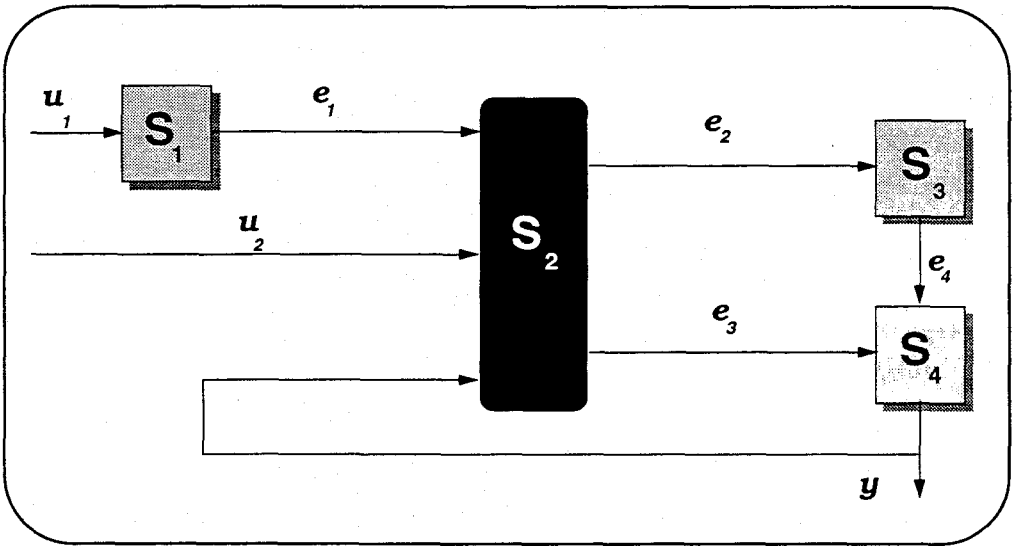


Figura 5.1: Sistema Genérico.

de naturaleza cuantitativa, es decir, de sistemas de los que conociendo su estructura peuden ser modelados mediante ecuaciones diferenciales y algebraicas. La caja negra de esquinas curvas representa un subsistema de naturaleza cualitativa del cuál no se conoce nada con respecto a su estructura interna. Se conoce solamente la interrelación con otros subsistemas a través de sus variables de entrada y de salida.

Supongamos que el modelo S_2 ha sido modelado cualitativamente bajo la suposición de que su comportamiento, variables e_2 y e_3, depende de las variables de entrada e_1, u_2 y la realimentación y. La identificación de este subsistema puede haberse realizado, por ejemplo, mediante la metodología de modelado cualitativo que fue desarrollada en el capítulo anterior. Así, SAPS–II hubiera sintetizado dos razonadores inductivos difusos, uno para la variable e_2 y otro para la variable e_3 para representar cualitativamente al sistema S_2.

Ya que los razonadores inductivos requieren de variables cualitativas en sus entradas y predicen valores también cualitativos, será necesario utilizar las funciones de SAPS-II de codificación difusa y de regeneración escalar, para poder *conectar* el modelo cualitativo recién identificado con el resto de los subsistemas. Incorporando estas funciones en el

sistema general, el esquema de simulación mixta de los subsistemas cuantitativos, S_1, S_3 y S_4, y del subsistema cualitativo, S_2, guardará la topología mostrada en la figura 5.2. Las cajas *Codifica* y *Regenera* corresponden con las funciones de SAPS–II, *recode* y *regenerate*, para convertir información cuantitativa en triples cualitativos y viceversa, respectivamente. Las señales cuantitativas, e_1, u_2 y y, deberán de ser convertidas en los triples cualitativos, e_1^*, u_2^* y y^*, mediante la función de codificación difusa. Las variables cualitativas, e_2^* y e_3^*, obtenidas por los razonadores cualitativos, tendrán que ser regeneradas a las señales cuantitativas, e_2 y e_3, mediante la función de regeneración. El esquema de simulación mixta de la figura 5.2 representa un caso más o menos típico en sistemas dinámicos, como por ejemplo el diseño de controladores difusos.

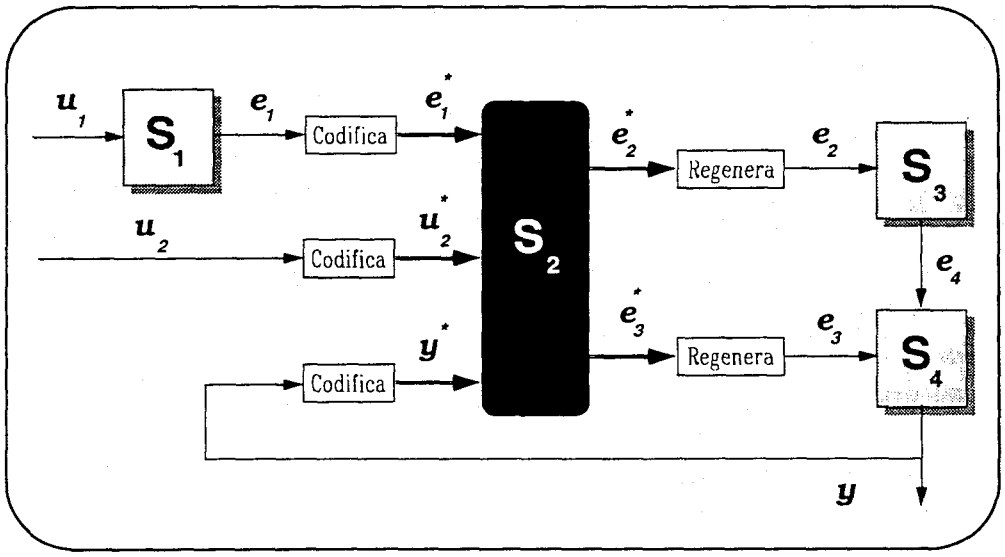


Figura 5.2: Esquema de Simulación Mixta.

La implementación de este tipo de sistemas puede realizarse utilizando la interfase ACSL/SAPS-II diseñada para soportar la simulación mixta del FIR. El lenguaje de simulación continua ACSL [MGA 86], considerado como un estándar, es particularmente poderoso para describir los modelos cualitativos mediante sistemas de ecuaciones diferenciales y algebraicas. Los modelos de naturaleza cualitativa son representados

utilizando el grupo de rutinas de la interfase ACSL/SAPS-II. Esta interfase consiste de la adaptación de las funciones de codificación difusa, (*recode*), de predicción difusa, (*forecast2*), y de regeneración escalar, (*regenerate*), para que puedan ser llamadas desde ACSL. Las funciones, codificadas en Fortran, son incluidas como una biblioteca de rutinas que pueden ser invocadas directamente desde ACSL. El manejador de eventos discretos de ACSL permite realizar la tarea de interconexión entre las variables de los modelos cuantitativos y las variables de los modelos cualitativos de una manera sencilla. Una rutina genérica, denominada D–SAPS, es la encargada de realizar las operaciones y llamadas a la biblioteca de rutinas de SAPS-II. D–SAPS es implementada como un bloque discreto que se ejecuta cada intervalo de muestreo, *tsaps*. Para efectos de dar una descripción ilustrativa de la estructura de D–SAPS, supongamos que el sistema S_2 ha sido identificado cualitativamente mediante SAPS-II y que las máscaras óptimas encontradas para las variables e_2^* y e_3^* han sido:

$$
\begin{array}{c} {}_t\backslash^x \\ t-2\delta t \\ t-\delta t \\ t \end{array}
\begin{array}{c} \begin{array}{ccccc} e_1 & u_2 & y & e_2 & e_3 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & -1 \\ 0 & -2 & -3 & 0 & 0 \\ -4 & 0 & 0 & +1 & 0 \end{pmatrix} \end{array} \tag{5.1}
$$

$$
\begin{array}{c} {}_t\backslash^x \\ t-2\delta t \\ t-\delta t \\ t \end{array}
\begin{array}{c} \begin{array}{ccccc} e_1 & u_2 & y & e_2 & e_3 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & -1 & 0 \\ -2 & 0 & -3 & 0 & 0 \\ 0 & -4 & 0 & 0 & +1 \end{pmatrix} \end{array} \tag{5.2}
$$

con un δt de 0.5 segundos. Las bases de reglas, *rules1* y *rules2*, del comportamiento episódico correspondiente a las máscaras son almacenadas en ficheros externos. El código, dado a continuación, representa una versión simplificada del código actual para el bloque D–saps correspondiente al sistema de la figura 5.2.

```
 DISCRETE D-saps
   INTERVAL tsaps = 0.5   !segundos
   PROCEDURAL(var_e2,var_e3 = var_e1,var_u2,var_y)
!                                                    1 - INICIALIZACION
     IF (primera_vez) THEN
        LECTURA('LANDMARKS.DAT', val_lim_e1, val_lim_u2,
                                 val_lim_e2, val_lim_e3,
                                 val_lim_y)
        LECTURA('INIRAW.DAT', clas, memb, side)
        LECTURA('FILE1.DAT', rules1, mask1, depth1)
        LECTURA('FILE2.DAT', rules2, mask2, depth2)
        primera_vez = .F.
```

```
    ENDIF
!                                                          2 - CODIFICACION
    RECODE(clas_e1, memb_e1, side_e1, var_e1,
                        val_lim_e1, 3)
    RECODE(clas_u2, memb_u2, side_u2, var_u2,
                        val_lim_u2, 3)
    RECODE(clas_y, memb_y, side_y, var_y,
                        val_lim_y, 3)
!                                                      3 - AVANZAR LA MASCARA
    nvar1 = 5
    DO kk = 1, depth1-1
       DO m = 1, nvar1
          clas(kk,m)  = raw(kk+1,m)
          memb(kk,m)  = memb(kk+1,m)
          side(kk,m)  = side(kk+1,m)
       END ! DO
    END ! DO
    clas(1,depth1)  = clas_e1
    memb(1,depth1)  = memb_e1
    side(1,depth1)  = side_e1
    clas(2,depth1)  = clas_u2
    memb(2,depth1)  = memb_u2
    side(2,depth1)  = side_u2
    clas(3,depth1)  = clas_y
    memb(3,depth1)  = memb_y
    side(3,depth1)  = side_y
!                                                          4 - PREDICCION
    FORECAST2(clas_e2, memb_e2, side_e2,
            clas, memb, side, mask1, ndepth1, rules1)
    clas(4,depth1)  = clas_e2
    memb(4,depth1)  = memb_e2
    side(4,depth1)  = side_e2
    var_e2 = REGENERATE(clas_e2, memb_e2, side_e2,val_lim_e2)

    FORECAST2(clas_e3, memb_e3, side_e3,
            clas, memb, side, mask2, ndepth2, rules2)
    clas(5,depth1)  = clas_e3
    memb(5,depth1)  = memb_e3
    side(5,depth1)  = side_e3
    var_e3 = REGENERATE(clas_e3, memb_e3, side_e3,val_lim_e3)
    END !of PROCEDURAL
  END !of DISCRETE D-saps
```

Como puede verse en este código, el bloque discreto D–SAPS está compuesto de cuatro partes:

1) La primera parte es ejecutada únicamente en la inicialización del modelo cuando D–SAPS es llamado por primera vez. Para cada razonador inductivo, i, se carga desde los ficheros externos, *file1.dat* y *file2.dat*, las respectivas matrices de datos que contienen las máscaras, $mask1$ y $mask2$, con sus respectivas profundidades, $depth1$ y $depth2$, así como las bases de reglas de comportamiento, $rules1$ y $rules2$. Los datos de los valores límite (los "*landmarks*") para las variables involucradas se leen desde el fichero externo *landmarks.dat*. Finalmente, el estado inicial del sistema es

proporcionado por el fichero *iniraw.dat* que contiene los valores iniciales de las variables correspondientes a la profundidad de las máscaras.

2) En la segunda parte, que como el resto de las partes es ejecutada en cada llamada al bloque discreto, las variables de las m–entradas de las máscaras correspondientes al instante t, son transformadas por la función de codificación difusa, *recode*.

3) La tercera parte corresponde conceptualmente al proceso de desplazamiento de la máscara un intervalo de tiempo, *tsaps*, y su función es actualizar los valores de las m–entradas para conformar el patrón de entradas que será requerido por la rutina de predicción. Lamentablemente la asociación de variables debe realizarse de manera explícita, lo que hace que la rutina D–SAPS no sea genérica.

4) En la cuarta parte del bloque discreto, se predicen los correspondientes valores cualitativos de salida para cada razonador inductivo mediante la rutina *forecast2*. Finalmente, los estados cualitativos son regenerados a valores cuantitativos mediante la rutina *regenerate*, actualizando las variables para que el bloque continuo de ACSL pueda iterar durante un nuevo intervalo *tsaps*.

A diferencia de la implementación de las interfaces MATLAB/SAPS-II o CTRL-C/SAPS–II, que son completamente genéricas y flexibles, la interfase ACSL/SAPS-II debe ser programada un tanto *ad hoc* para cada caso particular de simulación mixta, lo que resulta un proceso un poco aburrido. Las consideraciones de eficiencia y velocidad de ejecución son las responsables en gran medida de que esto sea así. Actualmente se desarrolla una nueva versión de la interfase ACSL/SAPS-II para permitir que el proceso de implementación de los sistemas de simulación mixtos resulte más sencillo y amigable.

5.3 Un Ejemplo Ilustrativo

5.3.1 Descripción del Sistema

La complejidad del ejemplo presentado aquí no es muy grande pero es significativa de una aplicación real.

La figura 5.3 muestra un motor hidráulico accionado por una servoválvula de cuatro vías. Los flujos que van desde la línea de alta presión hacia la servoválvula, q_1 y q_4, así como los flujos de regreso desde la servoválvula a

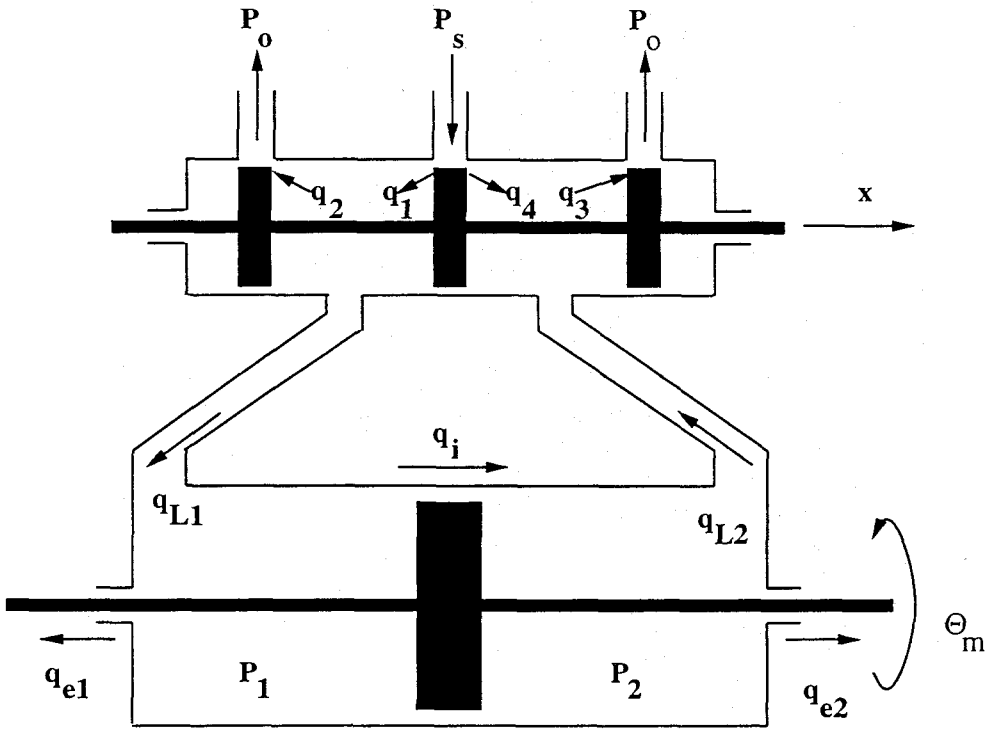


Figura 5.3: Motor Hidráulico y Servo Valvula.

la línea de baja presión, q_2 y q_3, son turbulentos, por lo tanto, la relación entre los flujos y la presión será cuadrática:

$$\begin{aligned} q_1 &= k(x_0 + x)\sqrt{P_S - p_1} \\ q_2 &= k(x_0 - x)\sqrt{p_1 - P_0} \\ q_3 &= k(x_0 + x)\sqrt{p_2 - P_0} \\ q_4 &= k(x_0 - x)\sqrt{P_S - p_2} \end{aligned} \tag{5.3}$$

Los parámetros seleccionados que describen el sistema son: línea de alta presión, $P_S = 0.137 \times 10^8$ N m^{-2}; línea de baja presión, $P_0 = 1.0132 \times 10^5$ N m^{-2}; posición de referencia de la servoválvula, $x_0 = 0.05$ m; y una constante de relación de $k = 0.248 \times 10^{-6}$ kg$^{-1/2}$ m$^{5/2}$.

El cambio de presión entre las cámaras de presión es proporcional al flujo efectivo entre las dos cámaras de la válvula y puede obtenerse como:

$$\dot{p}_1 = c_1(q_{L1} - q_i - q_{e1} - q_{ind})$$

$$\dot{p}_2 \quad = c_1(q_{ind} + q_i - q_{e2} - q_{L2}) \tag{5.4}$$

con $c_1 = 5.857 \times 10^{13}$ kg m^{-4} seg^{-2}. La pérdida interna de flujo, q_i, y las pérdidas externas de flujo, q_{e1} y q_{e2}, pueden calcularse como:

$$\begin{aligned} q_i \quad &= c_i \cdot p_L = c_i(p_1 - p_2) \\ q_{e1} \quad &= c_e \cdot p_1 \\ q_{e2} \quad &= c_e \cdot p_2 \end{aligned} \tag{5.5}$$

$c_i = 0.737 \times 10^{-13}$ kg^{-1} m^4 seg, y $c_e = 0.737 \times 10^{-12}$ kg^{-1} m^4 seg.

El voltaje inducido, q_{ind}, es proporcional a la velocidad angular del motor hidráulico, ω_m:

$$q_{ind} = \psi \cdot \omega_m \tag{5.6}$$

con $\psi = 0.575 \times 10^{-5}$ m^3, mientras que el par de torsión producido por el motor hidráulico es proporcional a la presión de carga, p_L:

$$T_m = \psi \cdot p_L = \psi(p_1 - p_2) \tag{5.7}$$

La parte mecánica tiene una inercia, J_m, de 0.08 kg m^2, y una fricción viscosa, ρ, de 1.5 kg m^2 seg^{-1}.

El motor hidráulico está incorporado en el circuito que se muestra en la figura 5.4. Para efectos de la simulación mixta cuantitativa y cualitativa, los componentes mecánicos y eléctricos del sistema de control serán modelados mediante ecuaciones diferenciales algebraicas, mientras que los componentes hidráulicos serán representados por un razonador inductivo difuso.

En la simulación mixta se supondrá que no se cuenta con ningún conocimiento respecto de la dinámica de la parte hidráulica del sistema que permita modelarla cuantitativamente mediante ecuaciones diferenciales. Todo lo que se sabe es que el par de torsión del motor hidráulico, T_m, depende "en alguna forma" de la señal de control, u, y de la velocidad angular, ω_m.

Para fines de validación, los resultados de la simulación mixta serán comparados con los resultados obtenidos previamente por un modelo completamente cuantitativo. El modelo completamente cuantitativo del sistema mostrado en la figura 5.4 fue codificado como un programa

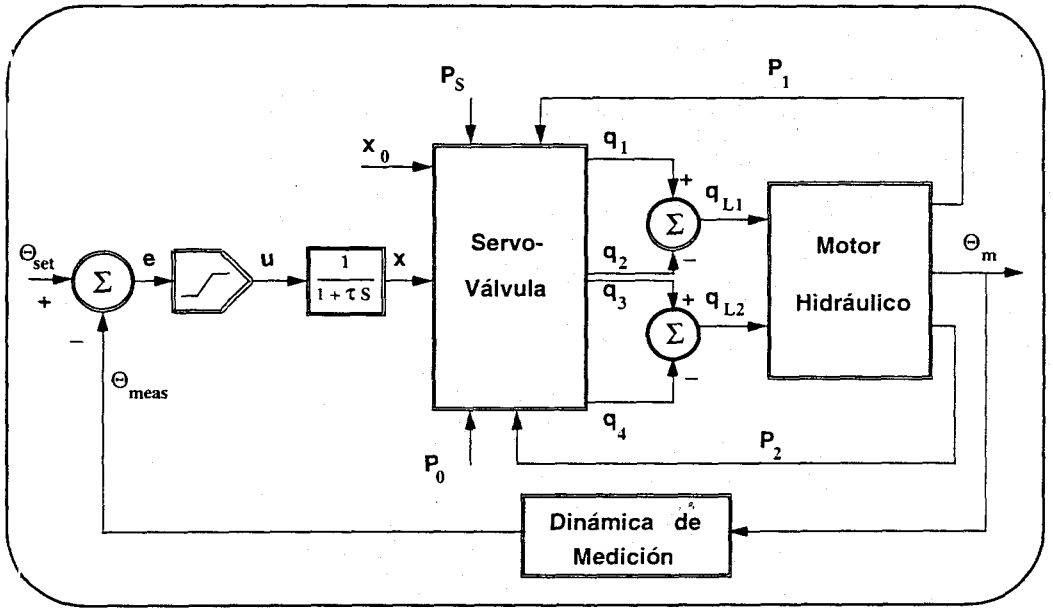


Figura 5.4: Circuito de Control de Posición del Motor Hidráulico.

en el lenguaje de simulación continua ACSL [MGA 86]. Se simularon 2.5 segundos en los que se aplicó al sistema una señal de entrada binaria aleatoria.

Los valores de la señal de control, u, la velocidad angular, $\omega_m = \dot{\theta}_m$, y el par de torción, T_m, para los primeros 2.25 segundos de la simulación cuantitativa serán transformados por la función de codificación difusa y almacenados para construir el razonador inductivo difuso del motor hidráulico.

Los valores de los últimos 0.25 segundos de la simulación cuantitativa también serán almacenados para llevar a cabo la validación del modelo. La validación se realizará comparando los resultados de simulación del modelo mixto resultante con los resultados del modelo completamente cuantitativo que será utilizado en lugar de "datos medidos".

5.3.2 Construcción del Modelo Cualitativo

Como fue descrito en el capítulo anterior, sección 4.2, el modelado de un sistema requiere de dos pasos. En el primero, los datos son registrados

y transformados en información cualitativa mediante la función de codificación difusa; en el segundo, se encuentra la causalidad dinámica cualitativa mediante la búsqueda de la máscara óptima utilizando para ello una máscara candidata y los datos debidamente muestreados y codificados.

Codificación Difusa

La primera cuestión que debe ser resuelta en el proceso de codificación difusa es la selección de un intervalo de muestreo, o intervalo de comunicación en el lenguaje de simulación, que sea adecuado para registrar las variables continuas. Los datos pueden proceder del registro de las medidas o, como en este ejemplo, del estudio de simulación con un modelo cuantitativo. En el ejemplo que desarrollamos aquí, el valor del intervalo cubierto por la máscara puede deducirse directamente a partir de la constante de tiempo más lenta (es decir, de la inversa del valor propio más pequeño del Jacobiano). Como el valor propio es -20, la constante de tiempo será 0.05 segundos. De acuerdo con la ecuación 4.11, si se elige una máscara de profundidad 3, las tres variables u, ω_m y T_m deberán de ser muestreadas una vez cada 0.025 segundos.

FIR sólo es capaz de producir un solo valor de T_m cada intervalo de muestreo, por lo que el comportamiento de la simulación mixta cuantitativa cualitativa será tal como el de un sistema de control que toma una muestra cada 0.025 segundos. Lamentablemente, la estabilidad del sistema de control será perdida, porque el intervalo de muestreo es demasiado lento para seguir los cambios del sistema. Aunque desde el punto de vista de capturar los estados dinámicos del sistema el muestreo cada 0.025 segundos es correcto, desde la perspectiva de un sistema de control es necesario que la variable de control se actualice con una rapidez considerablemente mayor. Ya que el sistema continuo (cuantitativo) sigue evolucionando en el tiempo, la actualización tardía de la variable de control, que es interpretada como pequeñas señales escalón, termina por hacer que el sistema derive hacia un comportamiento inestable. Para resolver este problema, se codificó un programa en el lenguaje de simulación ACSL con el objetivo de estudiar el efecto de diferentes intervalos de comunicación entre los dos subsistemas en la estabilidad del sistema completo. La idea del programa es muy simple. Introduciendo en el modelo puramente cuantitativo un retardo de tiempo en el cálculo del par de torsión con un valor inicial de 0.025 segundos, se va disminuyendo el retardo hasta que la estabilidad se recobre. Mediante

este procedimiento se determinó que el retardo más largo que se puede tolerar sin que la estabilidad se pierda es de 0.0025 segundos, es decir, diez veces más corto que el intervalo de muestreo recomendado anteriormente. Consecuentemente, la profundidad de la máscara deberá de aumentarse de 3 a 21.

El siguiente paso en el proceso de modelado es encontrar el número de niveles discretos en que cada variable debe de ser codificada. Para este ejemplo, y en función de lo dicho en la sección 4.2.2, se decidió que las tres variables pueden caracterizarse suficientemente bien con tres niveles. Una discretización de las variables en estos términos implica que el número teórico de estados permitidos sea de $3 \times 3 \times 3 = 27$. Como se explicó anteriormente, es deseable que cada uno de esos estados válidos sea observado por lo menos 5 veces. Por lo tanto serán necesarios un mínimo de 135 registros que corresponden a un tiempo de simulación de 0.325 segundos. Sin embargo, tomando en cuenta que la observación de estados recomendada por FIR corresponde a un tiempo de muestreo mayor que el usado actualmente por las características del controlador, será necesario considerar un número de datos bastante mayor, aproximadamente en un orden de magnitud. Se decidió utilizar un tiempo de simulación de 2.5 segundos, de los cuales 2.25 segundos se utilizarán a la identificación del modelo, y los últimos 0.25 segundos serán destinados para la tarea de validación. De esta manera, la etapa de identificación contará con 900 registros y la de validación con 100. Los resultados de la simulación serán almacenados en dos grupos de acuerdo al experimento mostrado en la figura 5.5.

Determinación de la Máscara Óptima

Con los datos debidamente codificados, tal y como se describió antes, SAPS-II puede encontrar un modelo cualitativo del subsistema, utilizando la función de búsqueda de la máscara óptima (subsección 4.2.4). Para combinar los modelos cualitativos y cuantitativos de manera adecuada debe de resolverse la cuestión de la estabilidad del sistema al mismo tiempo que se respete la causalidad involucrada dentro de un intervalo equivalente a la constante de tiempo más lenta. Para resolver esto en el apartado anterior, se resolvió elegir una δt de 0.0025 segundos y una máscara de profundidad 21. Sin embargo, una máscara candidata para este caso con 63 ($= 21 \times 3$) posiciones de interrelación representa un problema sin solución, computacionalmente hablando. ¿Cómo puede resolverse tal situación? La respuesta a esta pregunta pasa por reinter-

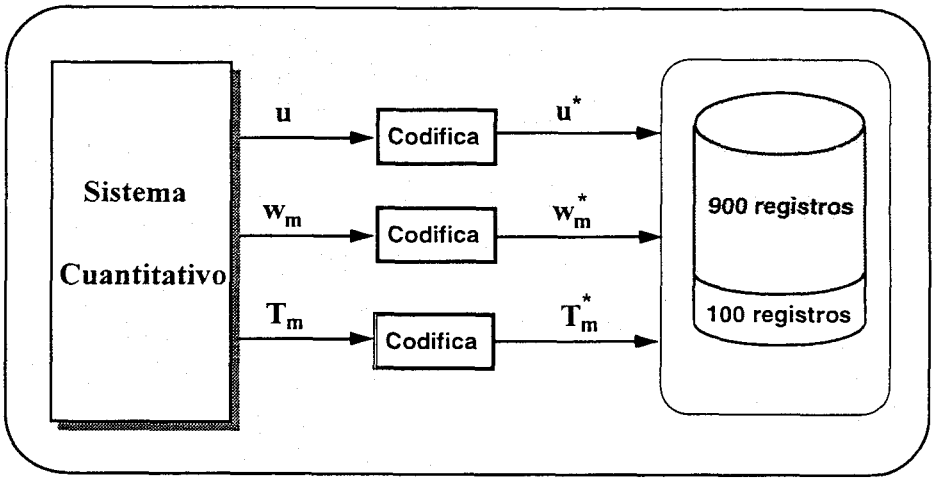


Figura 5.5: Experimento # 1.

pretar las sugerencias del razonamiento inductivo sin olvidar las de la teoría de control. Visto desde un punto de vista superior, es decir, del razonador cualitativo, las facetas importantes de la dinámica del proceso pueden ser capturadas en un intervalo de tiempo de 0.05 segundos, y debería bastar considerar las variables una vez cada 0.025 segundos. Sin embargo, desde el punto de vista de la estabilidad del bucle cerrado del sistema de control, no podemos permitir retardos de más de 0.0025 segundos. Podemos satisfacer ambos requerimientos sin aumento de la complejidad computacional de la máscara candidata si prohibimos la gran mayoría de las posiciones de la máscara candidata colocando ceros. En una máscara de 21 niveles de δt de profundidad, no es necesario que la máscara investigue las relaciones causales entre todos los niveles, solamente será necesario que considere los reglones cada 0.025 segundos de tiempo. Consecuentemente, si sólo seleccionamos entradas a la máscara en los niveles primero, 11^{avo} y 21^{avo}, y prohibimos el resto de las relaciones colocando ceros en sus posiciones, la estructura de interrelación descenderá a 9 con una complejidad computacional muy tratable. La máscara candidata resultante permitirá un muestreo suficientemente rápido para evitar inestabilidad, mientras que permite la evaluación exhaustiva de

todas las máscaras factibles sin explosión computacional.

La máscara candidata entonces puede describirse como:

$$
\begin{array}{l|ccc}
{}_t\backslash^x & u & \omega_m & T_m \\
t-20\delta t & -1 & -1 & -1 \\
t-19\delta t & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots \\
t-11\delta t & 0 & 0 & 0 \\
t-10\delta t & -1 & -1 & -1 \\
t-9\delta t & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots \\
t-\delta t & 0 & 0 & 0 \\
t & -1 & -1 & +1
\end{array}
\tag{5.8}
$$

Así, el par de torsión mecánico, T_m, en el tiempo t dependerá solamente de los valores de u y ω_m en el mismo tiempo, así como también de los valores de u, ω_m, y T_m en los tiempos $t - 0.025$ segundos y $t - 0.05$ segundos. Sin embargo, un nuevo valor de T_m se calculará cada 0.0025 segundos, satisfaciendo así los requerimientos de la estabilidad del sistema de control.

Utilizando esta máscara candidata y el conjunto de datos debidamente codificados, SAPS–II encontró la siguiente máscara óptima:

$$
\begin{array}{l|ccc}
{}_t\backslash^x & u & \omega_m & T_m \\
t-20\delta t & 0 & -1 & -2 \\
t-19\delta t & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots \\
t-11\delta t & 0 & 0 & 0 \\
t-10\delta t & 0 & 0 & 0 \\
t-9\delta t & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots \\
t-\delta t & 0 & 0 & 0 \\
t & -3 & 0 & +1
\end{array}
\tag{5.9}
$$

que, en términos funcionales, se representa como:

$$
T_m(t) = \tilde{\mathrm{f}}(\omega_m(t-0.05), T_m(t-0.05), u(t)) \tag{5.10}
$$

5.3.3 Validación del Modelo Cualitativo

Una vez que la máscara óptima ha sido encontrada, y antes de integrarla en el esquema de simulación mixta, debe de verificarse su capacidad de predicción. Para tal efecto, se realiza el experimento de validación mostrado en la figura 5.6.

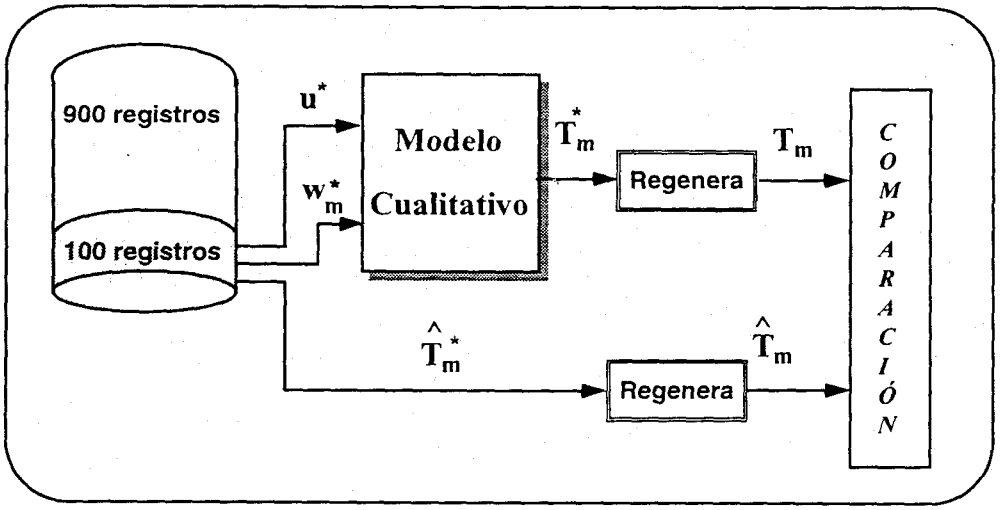


Figura 5.6: Validación del Modelo Cualitativo.

Como puede observarse en la figura, el experimento compara los valores de T_m obtenidos a partir de la simulación cuantitativa con los valores predichos y regenerados del razonador inductivo difuso. Como se dijo antes, los primeros 900 registros almacenados se usaron como datos históricos para encontrar la máscara óptima y el modelo cualitativo. La función de predicción difusa de SAPS-II (subsección 4.3) ha sido utilizada para predecir nuevos estados (triples) cualitativos para la variable T_m durante 100 intervalos más, equivalentes a 0.25 segundos no vistos anteriormente por el sistema. Los estados cualitativos pueden ser regenerados por la función de regeneración escalar de SAPS–II, y el resultado puede ser contrastado con los 100 registros almacenados para fines de validación.

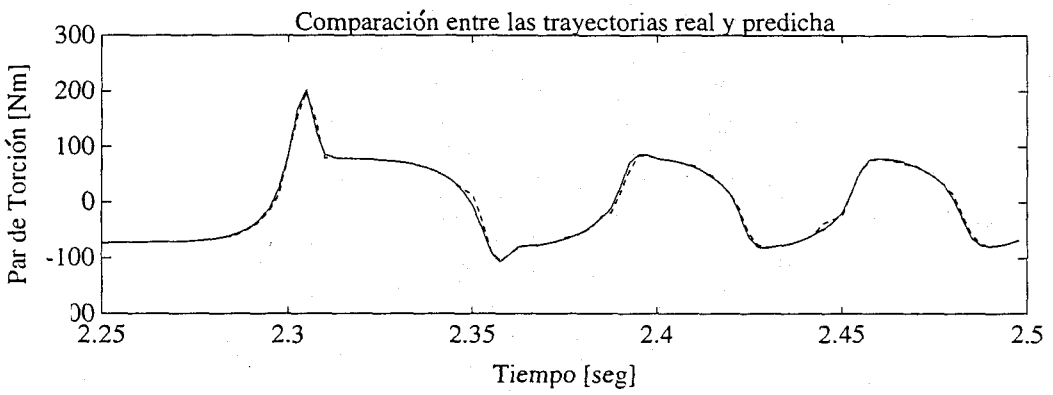


Figura 5.7: Resultados de Validación del Razonador Inductivo.

La figura 5.7 compara los valores correctos de T_m obtenidos de la simulación del modelo completamente cualitativo (línea continua) contra los valores obtenidos desde el razonador inductivo difuso (línea punteada). Los resultados son suficientemente buenos para permitir afirmar que el razonador inductivo difuso es un modelo cualitativo válido. Es claro que el modelo cualitativo contiene suficiente información respecto del comportamiento del subsistema hidráulico como para permitir reemplazar al modelo cuantitativo de ecuaciones diferenciales por el razonador inductivo difuso, a pesar de haberse utilizado solamente tres niveles de cuantización para cada variable. Es importante resaltar que el razonador inductivo difuso fue construido únicamente en base a los datos registrados. El único conocimiento del funcionamiento del subsistema hidráulico que se requirió fue el asumir que el par de torción, T_m, depende dinámicamente de la señal de control, u, y de la velocidad angular, ω_m.

5.3.4 Simulación Mixta Cualitativa Cuantitativa

Cuando la capacidad de predicción del modelo cualitativo ha sido verificada, el razonador inductivo difuso, en substitución del modelo de ecuaciones diferenciales del subsistema hidráulico, puede ser incorporado en un esquema de simulación mixta donde los componentes eléctricos y mecánicos son aún modelados mediante ecuaciones diferenciales, mientras que el subsistema hidráulico es ahora modelado cualitativamente mediante un razonador inductivo. El modelo mixto se muestra en la figura 5.8. La señal de control, u, es convertida

en el triple cualitativo, u^*, utilizando la función de codificación difusa (subsección 4.2.2). De igual manera, la velocidad angular cuantitativa, ω_m, del motor hidráulico es transformada en una señal cualitativa, ω_m^*. A partir de estas dos señales cualitativas, el razonador inductivo puede calcular el triple cualitativo para la señal del par de torción del motor hidráulico, T_m^*, usando la función de predicción difusa (subsección 4.3). Finalmente, esta señal cualitativa es convertida en la señal cuantitativa, T_m, mediante la función de regeneración escalar (subsección 4.3).

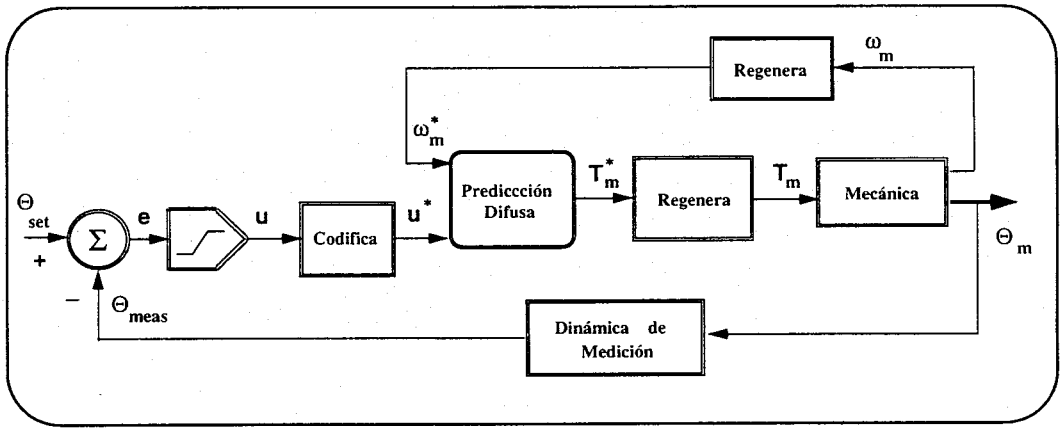


Figura 5.8: Simulación Mixta.

La predicción fue desarrollada para los últimos 100 intervalos de muestreo, es decir, partiendo desde el tiempo 2.25 segundos hasta el tiempo de 2.5 segundos. La figura 5.9 compara las gráficas de la posición angular, θ_m, del motor hidráulico que se calculó desde el modelo completamente cuantitativo (línea continua) contra la obtenida en el esquema de la simulación mixta cuantitativa cualitativa (línea punteada).

Como era de esperar, el modelo mixto se comportó como un sistema de control con muestreo de datos[1]. La simulación mixta exhibe una amplitud de oscilación que es ligeramente mayor y una frecuencia de

[1] del inglés: "sampled–data control system"

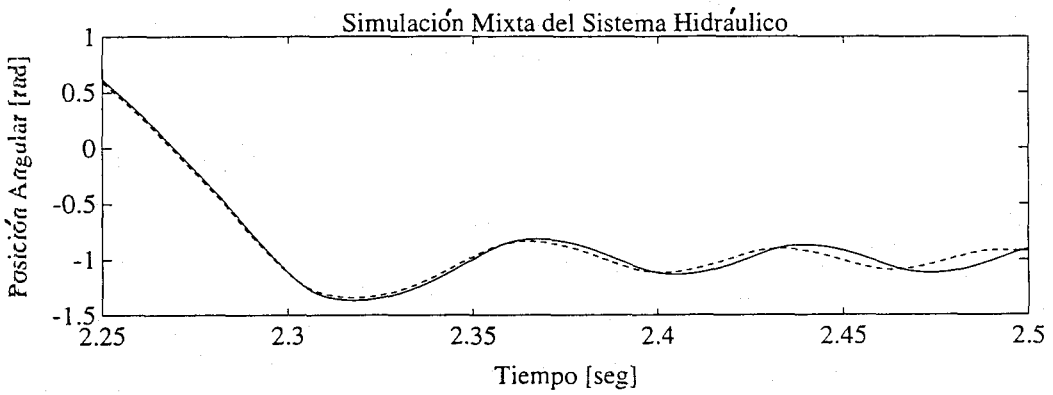


Figura 5.9: Resultados de Predicción de la Simulación Mixta.

oscilación que es ligeramente menor que la que muestra la simulación completamente cuantitativa. Sorprendentemente, las características de amortiguación del modelo mixto son ligeramente mejores que las del modelo completamente cualitativo.

5.4 Conclusiones

El esquema de simulación mixta es en efecto similar al esquema de simulación de sistemas muestreados; la *codificación difusa* toma el lugar de los convertidores analógico/digitales; y la *regeneración escalar* el de los convertidores digital/analógicos. Sin embargo, aquí es donde la similitud termina. Los sistemas muestreados operan en una representación exacta de las señales digitales. Los convertidores normalmente son convertidores de 12 *bits*, correspondiendo a una discretización de 4096 niveles. En contraste, el modelo de razonamiento inductivo difuso empleado en el ejemplo anterior codifica las tres variables en variables cualitativas solamente con tres niveles. La información cuantitativa es retenida en las funciones de pertenencia difusa que acompañan a la señal cualitativa. Debido al pequeño número de niveles discretos, el resultado de la máquina de estados finitos es extremadamente simple. La predicción de la pertenencia difusa ha mostrado ser muy efectiva en la inferencia de información cuantitativa respecto del sistema bajo investigación en términos cualitativos.

El esquema de simulación mixto descrito en esta sección ha sido utilizado en cada uno de los controladores difusos desarrollados en

esta tesis que serán presentados más adelante, y constituyen un elemento indispensable para su implementación. Los requerimientos de integración, particularmente el que se refiere al tratamiento de la variable independiente del tiempo, que han sido planteados anteriormente, son bien satisfechos por este esquema que en términos generales resulta simple de utilizar. Sin embargo, una versión más genérica de la interfase ACSL/SAPS–II debe de ser construida para permitir el uso completamente automatizado de ella que libere al usuario de la codificación *ad hoc* del bloque D–SAPS.

Capítulo 6

Diseño Sistemático de Controladores Difusos

6.1 Introducción

Una vez consideradas las características que hacen de los controladores difusos un esquema alternativo de control versátil y potente (capítulo 2), mostradas las cualidades del FIR como metodología de modelado cualitativo particularmente eficaz para el tratamiento de sistemas dinámicos (capítulo 4), y probado que la técnica de simulación mixta cuantitativa y cualitativa funciona adecuadamente (capítulo 5), se han establecido los elementos que servirán de marco metodológico en el diseño sistemático de controladores difusos.

En el capítulo 2 se dió una revisión a la tecnología de control difuso prestando la mayor atención a los aspectos de *representación* y de *procesamiento* del conocimiento. En los capítulos 3 y 4, el énfasis fue dado al proceso de *adquisición* del conocimiento, seleccionándose una aproximación inductiva cuyo punto de partida es el conjunto de datos registrados del comportamiento de un sistema. Finalmente en el capítulo 5 atendimos al requisito de integración entre las estructuras cualitativas de los controladores propuestos y las estructuras cuantitativas de los sistemas (o modelos) sobre los que opera. Sin embargo, la *fuente* misma del conocimiento utilizada para el diseño del controlador no ha sido estudiada aún.

Hasta este punto contamos con la herramienta que nos permite *extraer* de los datos la información contenida *descubriendo* las relaciones causales y dinámicas que guardan entre ellos; disponemos de los algoritmos

para representar adecuadamente el conocimiento obtenido y *construir* modelos representativos de los sistemas de los cuales fueron extraídos; finalmente contamos con la maquinaria de razonamiento que permite *emular* o *predecir* el comportamiento futuro de los sistemas bajo los escenarios deseados. Si el objetivo fuera solamente *modelar* un sistema, el trabajo terminaría aquí. Sin embargo, nuestro objetivo es bastante más complejo, es *diseñar*, es *controlar*. Cuando en los capítulos 4 y 5 nos hemos referido a un *sistema*, no se ha especificado si nos referimos al sistema de control, al sistema a ser controlado o al sistema que forman unidos. ¿Cuál es exactamente la fuente del conocimiento expresado por los datos? Si asumimos que los datos corresponden al sistema de control, ¿para qué deseamos diseñar otro controlador? ¿Cómo diseñar entonces para sistemas que no tengan previamente un controlador del cual aprender? Si los datos corresponden al sistema a ser controlado, es claro que en ellos no existe ninguna información respecto a la acción de control. ¿De dónde se extraerá entonces el conocimiento que permita diseñar el controlador? Hasta ahora se ha presentado una metodología para sistematizar el modelado cualitativo. ¿Cómo se va a sistematizar el diseño de los controladores difusos?

El objetivo de este capítulo es proponer una solución lo más amplia posible a la última de las preguntas. Se comenzará, sin embargo, respondiendo a la primera de ellas sobre el sentido de la palabra *sistema*. Cuando se describió la metodología del razonamiento inductivo difuso en el capítulo cuarto, la palabra sistema hacía referencia a "cualquier" sistema del cual se pudiera medir el comportamiento. Esta misma observación puede hacerse respecto a la técnica de simulación mixta descrita en el capítulo anterior. No obstante, desde el punto de vista del diseño de controladores es importante acotar el uso del término sistema. Dentro del campo de control, podemos considerar dos tipos de sistemas que pueden estudiarse por separado. En un primer caso, el sistema es tomado como el conjunto de acciones de control que un operador humano lleva a cabo sobre una planta específica. La metodología FIR está plenamente capacitada para capturar la dinámica de este *controlador humano* y posteriormente emular sus respuestas para reemplazarlo. Lo mismo puede hacerse en cualquier sistema de control convencional existente y del cuál se pueda obtener el espectro completo de su comportamiento bajo los diferentes estados que la planta a ser controlada pueda adquirir. Aunque puede dudarse del sentido o de la utilidad de este tipo de reemplazo, el controlador difuso obtenido de este tipo de sistemas mediante la metodología FIR puede ser justificado bajo los objetivos de aumento de robustez y de adaptabilidad sobre las

arquitecturas originales. No obstante, el trabajo realizado en este proceso es más de modelado que de diseño, y en principio no existen limitaciones para llevarlo a cabo siempre y cuando se cuente con el controlador del cual aprender. Otra variante de este primer caso, utilizada con mucha frecuencia en el diseño de controladores difusos, es la situación en la que un experto enuncia las reglas de control bajo un esquema (estructura de las reglas) predeterminado que debe de capturar el controlador que se diseña. Mediante un ajuste de parámetros las reglas son estructuradas convenientemente hasta que demuestran operar adecuadamente. En esta variante, el controlador es la abstracción del conocimiento experto en un sistema difuso que puede interpretarse como un controlador. En un segundo caso, un poco más clásico y más frecuente, se cuenta con una planta cuyo comportamiento es más o menos conocido y el objetivo es construir un sistema de control para él. Aunque es claro que el FIR permite modelar cualitativamente la planta capturando su estructura causal y dinámica, el objetivo es proponer un esquema de control adecuado, lo cual requerirá un proceso considerablemente más complejo y laborioso. Sistematizar este proceso es el objetivo no sólo de este capítulo, sino de la tesis en general. Para evitar confusión en el uso de los términos, nos referiremos al sistema que se desea controlar como a "la planta", al módulo de control como "el controlador" y al conjunto formado por los dos como "sistema de control".

6.2 Filosofía de Diseño

Cuando se diseña un controlador de cualquier tipo (clásico, adaptativo, o difuso), el problema intrínseco siempre es el mismo: dada una planta (sistema) y una trayectoria deseada de la señal de salida, ¿cuál es la mejor trayectoria de entrada que al ser aplicada a la planta provoque que la salida del sistema resulte tan similar como sea posible a la trayectoria de salida que se desea? Todos los diseños de controladores, ya sean directos o indirectos, están basados en un conocimiento parcial de la *dinámica inversa de la planta*, porque al menos conceptualmente, el modelo de la dinámica inversa nos da una respuesta a la pregunta planteada anteriormente. Si la dinámica inversa de la planta es conocida perfectamente, el problema de control resultará trivial. Examinemos una aproximación *simplista* de control, en el que se propone un *modelo de referencia* como la salida de la planta que se desea obtener, y en el cual se utiliza la *dinámica inversa de la planta* en la forma descrita por la figura 6.1.

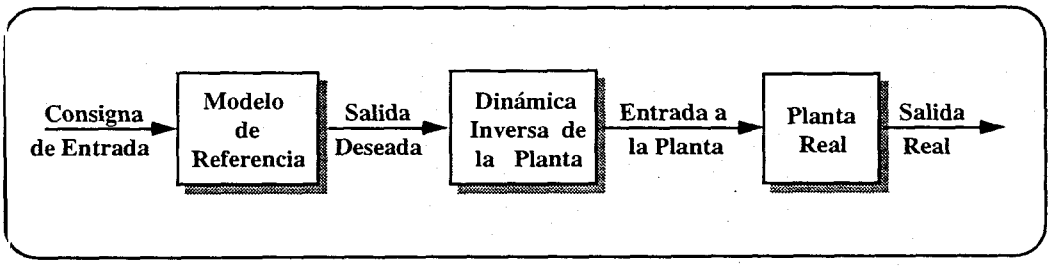


Figura 6.1: Controlador Simplista

Si la dinámica de la planta, tanto la directa como la inversa, se conocen de manera explícita, los dos módulos localizados más a la derecha se cancelarán mutuamente, y el conjunto del sistema completo se comportará exactamente como el modelo de referencia. La salida real de la planta será igual a la salida deseada. Para demostrar este concepto y su validez en el marco de modelado cualitativo y simulación mixta, se aplicó la técnica del FIR a un *controlador simplista* en el que la obtención de la dinámica inversa de la planta es trivial. El desarrollo de esta aplicación será presentado más adelante en la sección 6.4 como un primer ejemplo utilizado para aclarar algunos de los conceptos implicados en la metodología que se describirá en detalle en la sección 6.3. Como podrá constatarse después, los resultados obtenidos fueron francamente alentadores y abrieron el paso a la etapa de sistematización de la metodología para poder dar tratamiento a plantas más complejas.

La primera pregunta que surge es: ¿si el modelo de la dinámica inversa nos permite resolver el problema de control, cómo sistematizar su búsqueda para cualquier tipo de planta? Lamentablemente, el *controlador simplista* trabaja solamente en algunos casos muy sencillos, ya que el hecho de encontrar el modelo de la dinámica inversa no es una cuestión normalmente trivial sino más bien compleja e inclusive, en algunos casos, irresoluble. Comencemos la respuesta tomando el caso de una planta lineal. La razón de elegir una planta lineal es que permite utilizar intuitivamente conceptos bien conocidos en la teoría de control clásica que resultan claros y fáciles de comprender. No obstante, los conceptos utilizados en la explicación siguiente pueden ser extendidos a plantas no lineales y de cualquier grado de complejidad. Los siguientes aspectos deben ser considerados y resueltos si se espera lograr que la metodología funcione:

- La mayoría de las plantas son *estrictamente propias*, es decir, las funciones de transferencia que relacionan las entradas con las salidas contienen más polos que ceros. Aunque el concepto de "polos y ceros de una planta" es un concepto relacionado con el dominio de frecuencia y, por eso, es limitado a sistemas lineales, la propiedad de una planta de ser "estrictamente propia" puede ser extendida sin problemas a plantas no lineales también. Las plantas estrictamente propias no tienen un acoplamiento directo entrada/salida, lo que significa que si se aplica un impulso escalón a la entrada esto no conduce a obtener un impulso escalón en la salida. Es claro que si la planta directa es *estrictamente propia* entonces la planta inversa será *no propia* y exhibirá un comportamiento diferencial.
- En el caso de que además, la dinámica de la planta resulte ser de *fase no mínima*, conteniendo ceros en el lado derecho del plano complejo, la inversa de la planta tendrá una dinámica inestable. Si esto ocurre, la conjunción del modelo de la dinámica inversa de la planta con el modelo de la dinámica directa de la planta provocará la cancelación de los pares polo/cero inestables, una solución que funciona solamente sobre papel, porque en realidad cada tipo de inexactitud entre la planta real y el modelo de su dinámica inversa resultará en un residuo asociado con un valor propio inestable que no puede ignorarse como, dentro de corto tiempo, dominará el comportamiento del sistema. Encontramos el mismo problema si la planta es inestable y, por consiguiente, el modelo de la dinámica inversa de la planta resulta ser de fase no mínima.
- Como el controlador está en bucle abierto, el control no podrá corregir la dinámica no modelada de la planta y/o las perturbaciones, y el *controlador simplista* fallará, porque los errores (las diferencias entre el comportamiento de la planta real y el de su modelo inverso) se acumularán y la salida de la planta real tendrá a alejarse de la señal de referencia.

El problema del diseño del controlador simplista no es el hecho de estar basado en la dinámica inversa de la planta, sino más bien de la forma en la que este conocimiento se ha utilizado.

Para dar solución al primer problema, lo único que se debe hacer es darse cuenta que tanto el modelo de referencia como el modelo de la dinámica inversa son precisamente eso, son *modelos*. Son conjuntos de ecuaciones diferenciales que no tienen por que ser tratados de manera independiente. Podemos combinar los modelos de una forma diferente.

Podemos construir un *modelo en cascada*, que consista del modelo de referencia unido en cascada con el modelo de la dinámica inversa de la planta, de tal manera que dando al modelo de referencia un número suficientemente mayor de polos que de ceros, se pueda lograr que el sistema de control combinado resulte ser por lo menos un sistema *propio*. Si el modelo inverso *no propio* tiene k ceros más que polos, el modelo de referencia debería tener al menos $(k+1)$ polos más que ceros para que el sistema en cascada combinado resulte aún en un sistema *estrictamente propio*.

En esta forma, el problema anterior de encontrar el modelo de la dinámica inversa puede ser transformado en el problema de encontrar el modelo en cascada. Este problema es más fácil de resolver, dado que siempre es posible asignar al modelo de referencia un suficiente número de polos para conseguir que el modelo en cascada sea *propio* o *estrictamente propio*. Si el modelo en cascada puede ser encontrado, el problema de control estará solucionado. De esta forma, el controlador simplista tomará la estructura que se muestra en la figura 6.2.

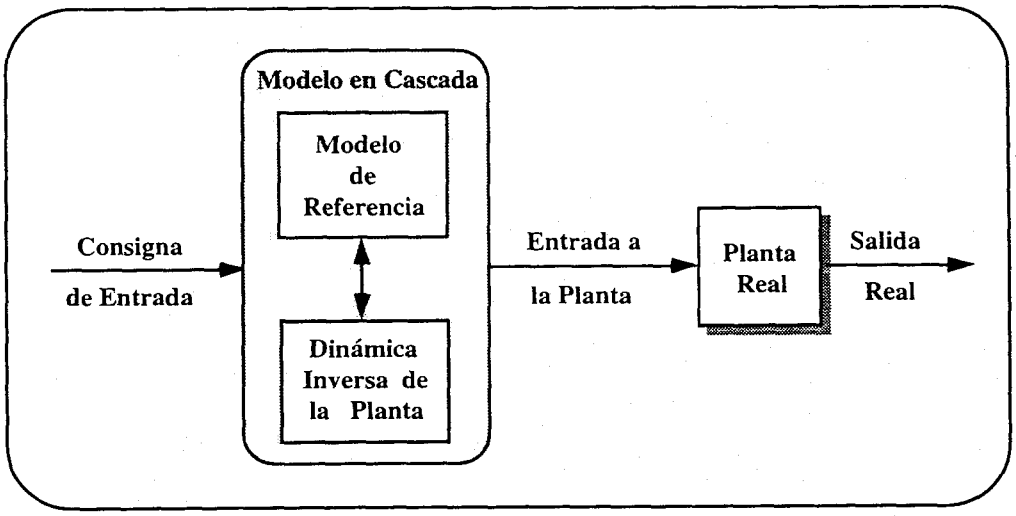


Figura 6.2: Modelo en Cascada

Discutimos ahora el segundo problema. Si ocurre que el modelo original es de fase no mínima, entonces se puede incluir un controlador local para desplazar todos los ceros a la parte izquierda del plano complejo. Ya que no serán impuestas nuevas demandas sobre el diseño del controlador, el desplazamiento de ceros puede lograrse mediante un(os) controlador(es)

clásico(s) de un tipo conveniente. La planta con el controlador local será considerada como la "nueva planta" que se utilizará para obtener el modelo en cascada. De esta manera se evita el problema de la cancelación de pares inestables de polos y ceros.

Si la planta original es inestable, se puede introducir un controlador clásico con la única intención de estabilizar la planta, es decir, de mover todos los polos a la parte izquierda del plano complejo. La planta con el estabilizador entonces será considerada como la "nueva planta" que se utilizará en el proceso de identificación del modelo de cascada.

Una vez encontrado el modelo en cascada, este puede ser identificado como un modelo cualitativo utilizando la metodología del FIR y, posteriormente ser insertado en un esquema de control realimentado en una forma similar al ejemplo tratado en el capítulo 5. El tercer problema es un problema clásico del diseño de controladores que, al menos para plantas lineales, fue resuelto de una vez por todas por Wolovich [Wolov 74]. Es el problema de mover partes del compensador en serie hacia el interior de un bucle de realimentación. Sin embargo y en contraste con la solución ofrecida por Wolovich, nuestro método no es limitado a los sistemas lineales y además es más simple y elegante en su implementación.

En la sección siguiente, se proporcionarán los detalles de la metodología de diseño que se ha expuesto de una manera un tanto intuitiva en esta sección. Como ya fue mencionado antes, en una sección posterior se demostrará la factibilidad de la filosofía de diseño, aplicándose a un sistema de *control en cascada* para el cual el problema de la obtención de la dinámica inversa de la planta es trivial.

6.3 Metodología de Diseño

La metodología de diseño de un *Controlador Difuso Basado en Razonamiento Inductivo (FIR–control)* puede ser descompuesta en un proceso que consta de las siguientes cinco etapas:

(a) Obtención de un modelo en cascada como una fusión de los modelos de referencia y de la dinámica inversa de la planta. El modelo en cascada será interpretado como un controlador.

(b) Obtención del modelo cualitativo del controlador o controladores difusos mediante la metodología de Razonamiento Inductivo Difuso descrita en el capítulo 4.

(c) Validación de el, o los, controladores en bucle abierto.

(d) Integración del controlador o controladores difusos con la planta en una configuración de bucle cerrado utilizando la técnica de modelado mixto descrita en el capítulo 5.

(e) En el caso de ser requerido para la estabilidad, añadir bucles de control simples a la estructura global del controlador.

En los apartados que siguen se describirán, una por una, cada etapa del proceso de diseño, enfatizando los aspectos heurísticos que por varias razones aún no han podido ser completamente sistematizados.

6.3.1 Obtención del Modelo en Cascada

En principio, siempre que los modelos de referencia y de la dinámica de la planta sean conocidos, es posible obtener analíticamente un modelo en cascada que en forma implícita contenga la dinámica inversa de la planta. En la literatura de control, este proceso es conocido como *linealización exacta*. Ejemplos de este proceso pueden ser encontrados en [Isido 89]. Sin embargo, en la mayoría de los casos, derivar manualmente el modelo en cascada resulta ser un proceso laborioso y complejo. Es por esta razón que hasta ahora, este tipo de transformación haya sido poco utilizada en aplicaciones de complejidad real. Afortunadamente, contamos entretanto con herramientas computacionales, tales como los paquetes de manipulación algebraica simbólica, que permiten automatizar la obtención de un modelo en cascada. Una de estas herramientas, particularmente indicada para el tratamiento de sistemas dinámicos, es Dymola [Elmqv 78], [Cell 91], [Cell 93]. Dymola es un lenguaje de modelado orientado a objetos que permite desarrollar modelos de sistemas dinámicos tanto continuos como discretos.

Se pueden definir dos *clases* de modelos independientes, uno para el modelo de referencia y otro para el modelo de la dinámica de la planta en términos de conjuntos de ecuaciones diferenciales y algebraicas. En código Dymola esto sería algo como:

model class *Planta*

terminal $u_1,\ \cdots,\ u_{entradas}$,
terminal $y_1,\ \cdots,\ y_{salidas}$,

parameter $K_1,\ \cdots,\ K_n$,

$\vdots$

model class *Referencia*

terminal $r_1,\ \cdots,\ r_{consignas}$,
terminal $ydes_1,\ \cdots,\ ydes_{salidas}$,

parameter $P_1,\ \cdots,\ P_m$,

$\vdots$

Expresiones Algebraicas	Expresiones Algebraicas
⋮	⋮
end	**end**

en donde u_i corresponde a la $i^{\text{ésima}}$ entrada a la planta, y_j a la $j^{\text{ésima}}$ salida de la planta, $ydes_k$ a la $k^{\text{ésima}}$ salida del modelo de referencia y r_l a la $l^{\text{ésima}}$ consigna de la planta. Los vectores K y P son los conjuntos de parámetros de los modelos de la planta y de referencia respectivamente. La notación parece implicar que las variables u_i son las *entradas* del modelo de la planta y las variables y_k son sus *salidas*, indicando que se trata en este código de un modelo de la dinámica directa de la planta. Sin embargo, eso no es el caso. La implicación de los nombres es de naturaleza semántica no sintáctica. La instrucción **terminal** informa al manipulador de fórmulas los puntos donde se pueden *interconectar* los modelos. No indica ningún sentido del flujo de información. En Dymola todas las ecuaciones especificadas son de naturaleza declarativa. El manipulador determina la *causalidad* de cada una de las ecuaciones usando el conocimiento de las entradas y salidas globales del modelo interconectado. La causalidad de una ecuación determina para cuál de sus variables debe de resolverse esta ecuación, es decir, cuál entre ellas debe de ponerse al lado izquierdo de la asignación en el programa de simulación generado por Dymola. Por consiguiente, el mismo modelo puede representar tanto la dinámica directa de la planta como también su dinámica inversa.

Puede definirse una nueva clase llamada: *EnCascada* para definir el modelo en cascada. La clase *EnCascada* invoca instanciaciones de objetos de la clase *Planta* y de la clase *Referencia*, de forma tal que el usuario simplemente declara las entradas al modelo de referencia como las entradas del modelo en cascada, entonces *conecta* las salidas del modelo de referencia a las salidas del modelo de la planta, es decir, el modelo de la planta se conecta dentro del modelo en cascada invirtiendo el flujo de información. En código Dymola, esto se puede expresar como:

model $EnCascada$

submodel $(Referencia)\ Ref$
submodel $(Planta)\ Plant$

input $r_1,\ \cdots,\ r_{consignas},$
output $u_1,\ \cdots,\ u_{entradas},$

$Ref.r_1 = r_1,\ \cdots,\ Ref.r_l = r_l$

$Plant.y_1 = Ref.ydes_1, \ \cdots, Plant.y_k = Ref.ydes_k,$
$u_1 = Plant.u_1, \ \cdots, u_i = Plant.u_i$

end

Como puede verse, el modelo *EnCascada* incorpora dos submodelos, uno de clase *Referencia* llamado *Ref* y el otro de clase *Planta* llamado *Plant*. El modelo *EnCascada* direcciona la causalidad de las ecuaciones por media de entradas externas (funciones de manejo), declaradas como *input*, y salidas globales, declaradas como *output*. El modelo conecta los submodelos dentro de sí mediante las instrucciones $Plant.y_k = Ref.ydes_k$ significando que cada una de las variables y_k de la planta deberá de ser igualada a la variable $ydes_k$ respectiva en el modelo de referencia y conecta los modelos con las entradas y salidas globales por media de las instrucciones $Ref.r_l = r_l$ y $u_i = Plant.u_i$. Las conexiones, *aparentemente en sentido erróneo*, obligan a que el flujo de evaluación del modelo de la planta se invierta, produciendo de manera implícita la aparición del modelo de la dinámica inversa de la planta.

Evidentemente, si el modelo de cascada aún no representa el modelo global sino solamente un modelo de más alto nivel en la jerarquía de modelos, un modelo que se interconectará después con otros modelos para formar un modelo de aún más alto nivel jerárquico, sería mejor declarar también sus variables externas, las variables r_l y u_i, como *terminal* en lugar de *input* y *output*, retrasando la determinación de la causalidad hasta el modelo del más alto nivel, es decir, el modelo global. Las conexiones, como se presentan en el modelo *EnCascada*, son conexiones primitivas. Se verá más adelante que Dymola ofrece construcciones más avanzadas para conectar múltiples variables con una única instrucción.

Este tipo de conexión, entre los modelos de la planta y de referencia, invariablemente conduce a que se genere un modelo del tipo *ecuaciones algebraicodiferenciales (DAE)*[1] de un índice de perturbación alto [Bren 89], debido a las restricciones que existen entre las ecuaciones de las salidas de los integradores de los dos modelos. Afortunadamente, Dymola es suficientemente poderoso para reducir automáticamente el índice de perturbación de los modelos de esta naturaleza obtiendo un modelo en el formato de espacio de estados. Aplicando el algoritmo de Pantelides [Pante 88], Dymola itera hasta que el modelo en cascada es reducido a índice 1, diferenciando simbólicamente las ecuaciones de restricción y añadiendo en cada ciclo el resultado al conjunto de ecuaciones del modelo [Celli 93]. Una vez en este punto suelen permanecer bucles algebraicos

[1] del inglés: Differential Algebraic Equations

en un conjunto de ecuaciones fuertemente acopladas, los cuales pueden ser resueltos mediante manipulaciones tanto algebraicas (simbólicas) como numéricas. El algoritmo de Tarjan [Tarja 72] es utilizado para resolver el problema de la asignación de la causalidad determinando el conjunto mínimo de ecuaciones algebraicas acopladas. En esta forma, el manipulador de fórmulas algebraicas de Dymola permite obtener un modelo analítico cerrado en el espacio de estados del sistema en cascada a partir de las descripciones independientes de los modelos de referencia y de la planta, equivalente al acoplamiento de modelos de referencia y de la dinámica inversa de la planta del controlador simplista mostrado en las figuras 6.1 y 6.2. La figura 6.3 intenta ilustrar la estructura de este nuevo proceso para el caso de un sistema SISO.

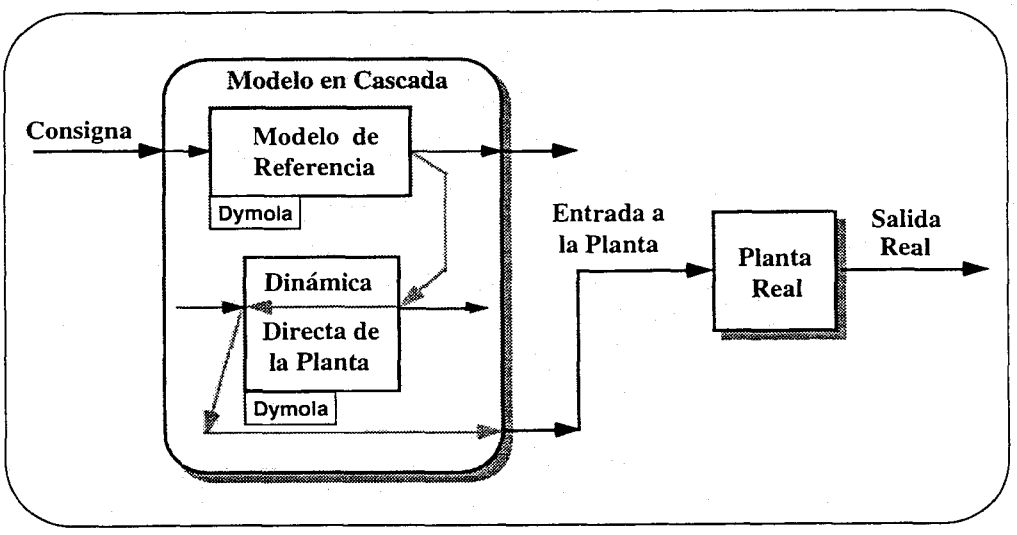


Figura 6.3: Obtención del Modelo en Cascada mediante Dymola

Resueltos los problemas de la reducción del índice alto de perturbación y de la asignación de la causalidad, el modelo en cascada se genera automáticamente en el lenguaje de simulación deseado. El siguiente archivo, en código Dymola, produce el modelo del sistema en cascada en el lenguaje de simulación ACSL [MGA 86].

enter model
@cascada.dym

differentiate

variable state $cascada.varestado_1, \cdots, cascada.varestado_M$

partition

language *acsl*
outfile *Cascada.csl*
output model

stop

La instrucción *differentiate* activa la reducción automática del índice de perturbación del modelo iterando tantas veces como sea necesario para alcanzar un índice de orden 1. Después de la diferenciación deben seleccionarse las variables de estado. Como Dymola añade las ecuaciones diferenciadas al modelo en lugar de reemplazar las restricciones por ellas, se reduce el número de variables de estado. Normalmente se selecciona un subconjunto de las salidas de los integradores, pero, como no son todos, este proceso no es automatizado en Dymola. La instrucción *variable state* permite al usuario determinar las variables de estado que quiere que se usen en el modelo. Finalmente, la instrucción *partition* permite encontrar la asignación de la causalidad computacional determinando el conjunto de ecuaciones que formarán el modelo de simulación.

6.3.2 Obtención del Modelo Cualitativo del Controlador

Puede pensarse que, una vez que el sistema en cascada ha sido obtenido, el problema de control ha quedado resuelto, ya que ahora es posible calcular las señales de entrada a la planta más adecuadas en función del tiempo. Sin embargo, esto no es así. El problema con el diseño basado en el modelo en cascada es que la arquitectura del sistema de control mediante el sistema en cascada es una estructura de bucle abierto. Consecuentemente, el sistema de control no mostrará ninguna reducción de la sensibilidad a perturbaciones externas y/o a dinámicas no modeladas de la planta, lo cual es una de las más importantes ventajas del control realimentado.

La cuestión es, ¿qué tanto de la dinámica del compensador en cascada puede ser movida hacia el interior del bucle de realimentación de la planta para darle la robustez requerida? El problema no es nuevo. Wolovich [Wolov 74] desarrolló un método para convertir los diseños de compensadores en cascada para sistemas multivariables lineales a un diseño de estructura con realimentación utilizando matrices polinomiales. La obtención del compensador en cascada es la parte fácil del trabajo. Una vez diseñado, convertir el compensador en cascada en un controlador

realimentado es un proceso normalmente muy complejo y que requiere numerosas etapas.

La situación en nuestra metodología es similar, excepto que en nuestro caso el diseño del compensador realimentado puede ser arbitrariamente complejo y arbitrariamente no lineal. El propósito del diseño del controlador difuso es precisamente hacer posible la transformación de un compensador en cascada que trabaja estrictamente hacia adelante en un controlador predominantemente realimentado. Además, se intenta que el diseño sea lo más sistemático posible y que permita tratar estructuras arbitrariamente complejas (SISO, MISO, MIMO) y arbitrariamente no lineales.

El proceso de diseño de un FIR–controlador es similar al procedimiento descrito previamente en el capítulo 4 para el modelado de un sistema cualitativo. Una vez que el modelo en cascada es encontrado, se aplican las señales apropiadas (por ejemplo ruido binario aleatorio) a cada una de las r_l entradas del sistema de control (modelo de referencia). Las reacciones de la(s) salida(s) del modelo de referencia, $ydes_k$ y de la(s) salida(s) del modelo en cascada, u_j, son observadas (muestreadas) y registradas según las recomendaciones contenidas en la sección 4.2.1. El conjunto de estas señales, entradas y salidas, son almacenadas dentro de una *matriz primitiva de datos* en la que cada columna representa una variable y cada renglón un registro de datos en el tiempo (punto muestreado). En un primer experimento, el sistema es simulado durante un lapso suficiente de tiempo, tal que permita excitar todos los modos del sistema en todas las frecuencias que sean de interés. Utilizando *codificación difusa*, como se explicó en la sección 4.2.2, cada valor cuantitativo de las r_l, $ydes_k$ y u_i señales es convertido en el *triple cualitativo* r_l^*, $ydes_k^*$ y u_i^* correspondiente que contiene un valor de clase, un valor de pertenencia y un valor de lado. Como resultado de esta transformación se obtienen tres matrices de datos; una contiene los valores de clase, la segunda contiene los valores de pertenencia, y la tercera los valores de lado. Estas tres matrices conforman la matriz primitiva de datos cualitativos que será utilizada en las siguientes etapas del proceso de razonamiento inductivo difuso. Una porción mayoritaria de los datos almacenados (aproximadamente el 90%), se destinará para identificar el modelo cualitativo del sistema en cascada, mientras que el resto se reservará para la etapa de validación. El procedimiento del experimento $^{\#}1$ es mostrado en la figura 6.4 para el caso de un sistema de estructura SISO.

Con las variables debidamente codificadas (fuzificadas) y almacenadas,

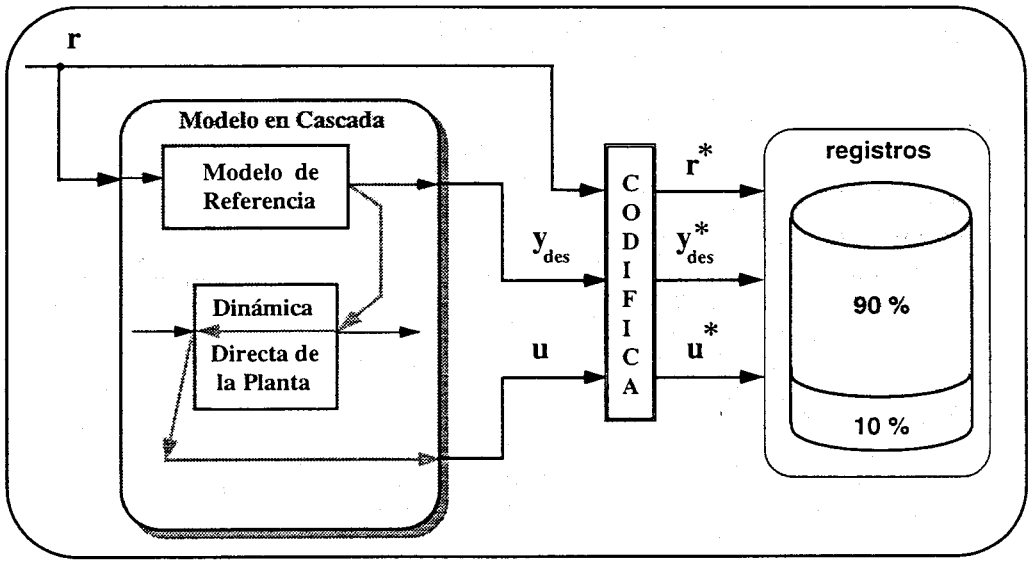


Figura 6.4: Experimento #1 para la Obtención del Modelo Cualitativo

puede pasarse a la identificación de los modelos cualitativos utilizando *análisis de máscaras óptimas* tal y como fue descrito en la sección 4.2.4. La máscara óptima representa el conjunto de relaciones causales más adecuado que se puede establecer entre todas las variables de entrada y una variable de salida particular (que en este caso es la variable de control). Cuando la máscara es desplazada sobre la matriz primitiva de datos cualitativa, la dependencia causal temporal entre las variables cualitativas es transformada en una dependencia estática generándose una base de reglas de comportamiento. Esta base de reglas, junto con la máscara correspondiente, constituye un modelo cualitativo el cual puede ser interpretado como la versión cualitativa del modelo en cascada o como el mismo controlador difuso.

La metodología de diseño de controladores FIR no puede ser aplicada para obtener un controlador simple que genere salidas múltiples, por lo que en el caso de que la planta que se desea controlar, requiriese de más de una señal de entrada proveniente del sistema de control, el problema debe descomponerse en casos simples. La metodología FIR permite identificar tantos modelos con estructura MISO independientes como número de entradas requiera la planta. Así, la arquitectura de FIR–control para una planta con n–entradas estará compuesta por n controladores difusos diferentes.

6.3.3 Validación del Modelo Cualitativo

En la tercera etapa del diseño, el modelo cualitativo recién obtenido es simulado en bucle abierto durante el lapso de tiempo reservado para la validación. Los valores cualitativos obtenidos por el razonador inductivo se convierten a valores cuantitativos mediante la función de Regeneración de SAPS–II, que será denotada como "Regenera" en las figuras siguientes y que ha sido descrita previamente en la sección 4.3. Los valores cuantitativos obtenidos se compararán contra los valores previamente obtenidos en la simulación cuantitativa del modelo de referencia según se presenta en la figura 6.5.

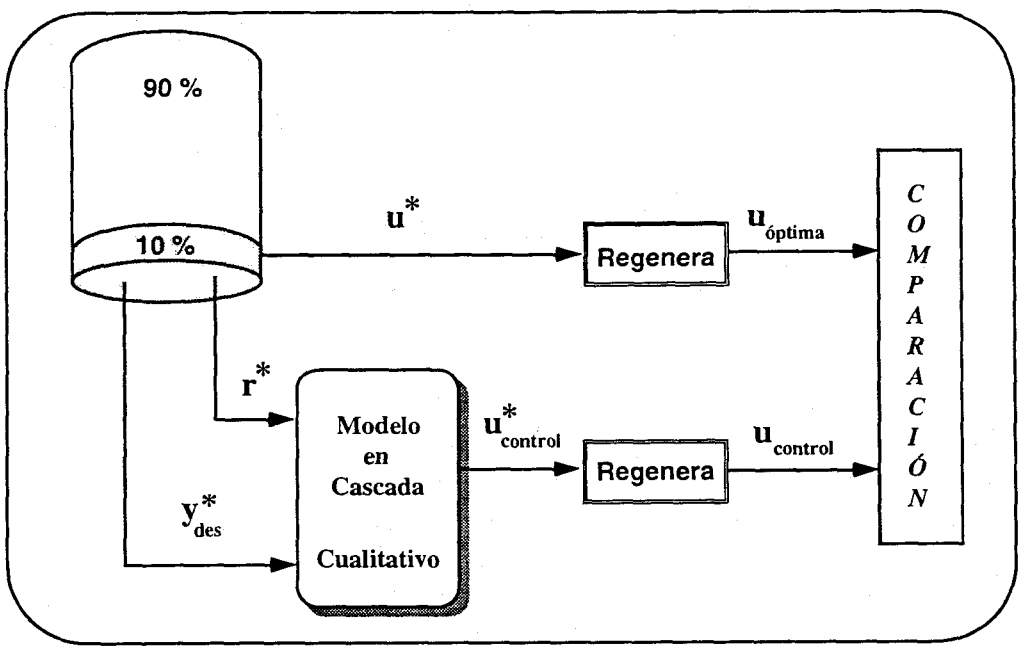


Figura 6.5: Validación del Modelo Cualitativo

Si la comparación entre los dos grupos de datos es buena, se puede pasar a la siguiente etapa. En el caso contrario, el diseño del controlador difuso debe modificarse ya sea mediante la consideración de máscaras subóptimas disponible en la historia de máscaras, o por la modificación de las funciones de entrada para excitar los modelos, o cambiando el intervalo de muestreo. Esta parte del diseño no ha sido automatizada aún completamente, pero los elementos de cómo cambiar las funciones de entrada y de cómo seleccionar el intervalo de muestreo han sido discutidos

en la sección 4.2.1 así como en [Celli 91], [Nebot 94a]. Siguiendo estas sugerencias, normalmente se obtienen modelos cualitativos decentes.

6.3.4 Integración del Controlador Difuso al Sistema

Cuando se ha verificado que los errores en la etapa anterior son suficientemente pequeños, el bucle de control deberá de cerrarse. El controlador difuso se integrará ahora con la planta en un sistema de control que incluya ambos modelos. Esta es una operación relativamente simple. El truco para hacerlo, aunque el modelo cualitativo fue diseñado utilizando la(s) salida(s) del modelo de referencia, $ydes_k$, ahora será usado con la(s) salida(s) del modelo de referencia reemplazadas por la(s) salidas reales, y_j, de la planta. De esta mánera, el modelo cualitativo diseñado previamente se ha convertido ahora en el controlador difuso de una configuración similar a la de la figura 6.6. En esta configuración realimentada, el controlador difuso será capaz de corregir los efectos de perturbaciones del entorno y de la dinámica no modelada de la planta.

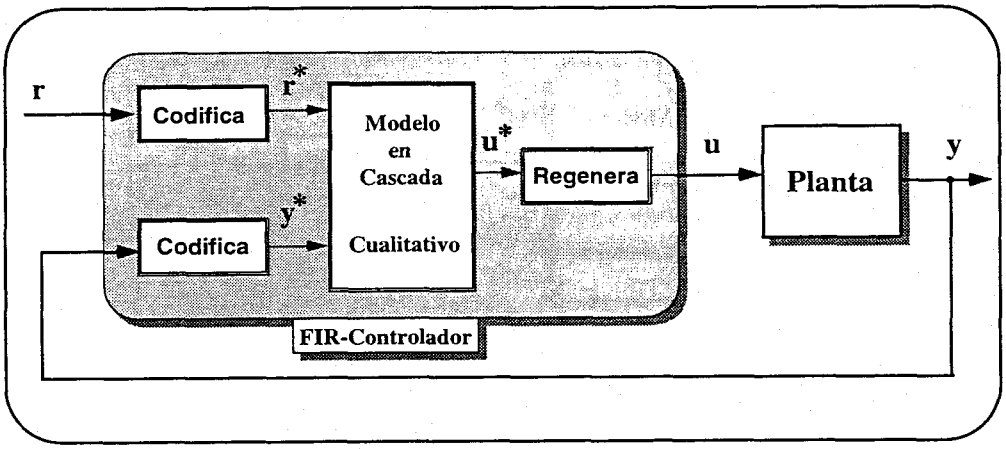


Figura 6.6: Estructura de un FIR-Controlador

En el sistema SISO de la figura 6.6, la entrada escalar, r, se convierte en una variable difusa, r^*, por medio de la función de Codificación Difusa (sección 4.2.2). Similarmente, la salida del sistema, y, es convertida en la variable difusa, y^*. El controlador difuso utiliza estas dos variables difusas para calcular la señal de control, u^* mediante la función

de Predicción Difusa (sección 4.3). Finalmente, la señal de control obtenida por el razonador inductivo, expresada de manera cualitativa, es convertida en el valor escalar, u, mediante la función de transformación "Regenera".

Sin embargo, si los efectos de la perturbación externa toman dimensiones mayores, el comportamiento del control puede deteriorarse de manera importante. Aún en este caso, la metodología permite obtener una solución simple a este problema. Todo lo que se necesita hacer en tal caso es aumentar el modelo de la planta en forma tal que tome en cuenta también los efectos de esas variables de entrada adicionales. Dymola incluirá automáticamente esas variables en la obtención del modelo en cascada. La matriz primitiva de datos tendrá una columna más, y el modelo cualitativo obtendrá una entrada adicional, el resto permanecerá igual. Por supuesto, se requerirá un número de registros de entrenamiento considerablemente mayor para la base de datos con la que se construye el razonador inductivo que representará al controlador difuso. No obstante, todo este proceso es realizado de una manera fuera de línea por lo que no afecta la velocidad de respuesta del FIR–controlador.

6.3.5 Estabilización, Seguimiento, y Ajuste

Hasta ahora no se ha mencionado nada respecto al hecho de que la salida de la planta debe guardar una cercanía, lo más estrecha posible, con la salida del modelo de referencia. Aunque la información de la estructura del modelo de referencia ha sido utilizada en la identificación del modelo en cascada para construir el FIR–control, la señal de salida actual del modelo de referencia en sí misma no es utilizada aún. Por lo tanto es posible que la salida de la planta en la configuración realimentada derive lenta pero progresivamente de la trayectoria deseada sin que el FIR–controlador se percate de ello. Si esto último ocurre, pueden añadirse bucles adicionales de estabilización y seguimiento.

Si resulta necesario, debe introducirse un bucle de control de seguimiento, en el cual la diferencia entre la salida de la planta deseada y la salida real sea realimentada a la variable de control. El control de seguimiento puede hacerse mediante un controlador muy simple del tipo P (o PI). Dado que el control de seguimiento solamente corrige errores muy pequeños, es completamente insensible a no linealidades de la planta o a cambios en el punto de operación, por lo que no requiere ser reajustado. El controlador difuso es realmente el que lleva a cabo todo el trabajo. La

señal de realimentación que viene desde el bucle de control de seguimiento es mucho más pequeña en magnitud que la señal que viene del controlador difuso. Todo lo que hace es proveer un estímulo adicional para que la planta siga la trayectoria deseada. En configuraciones más complejas es posible que sea necesario incluir otros bucles de control de seguimiento y/o estabilidad, lo cual puede realizarse en igual forma y bajo las mismas consideraciones dadas anteriormente.

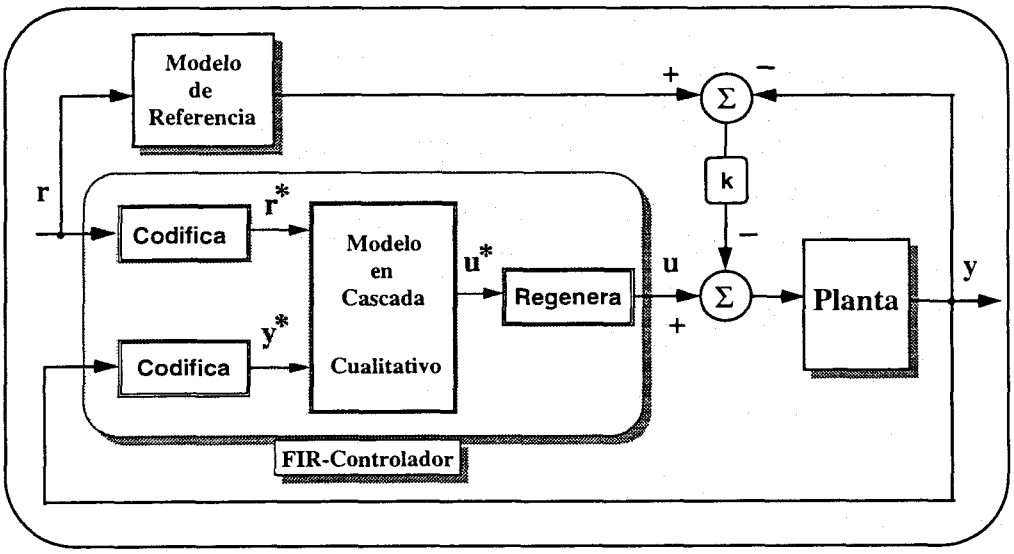


Figura 6.7: Ajuste del FIR–Controlador.

La figura 6.7 utiliza un sistema SISO para mostrar la forma en la que bucles adicionales de estabilidad y/o seguimiento pueden ser añadidos dentro de la arquitectura propuesta por la metodología de diseño presentada aquí.

6.4 Diseño de un FIR-Controlador para una Planta Simple

Con el propósito de mostrar, paso a paso, la forma en la que la metodología basada en razonamiento inductivo difuso propuesta en esta tesis puede ser utilizada, en esta sección presentaremos un primer ejemplo particularmente claro por lo sencillo de su estructura. El desarrollo de este controlador difuso fue presentado originalmente en

la *IEEE Intelligent Control Conference* [Celli 92b] y posteriormente, en una versión extendida, fue aceptado para publicarse en el *Journal of Intelligent and Fuzzy Systems: Applications* que aparecerá en el número 3 del año actual. El objetivo de esta aplicación era realizar una primera estimación de la factibilidad que la aproximación tiene para sistematizar el desarrollo de controladores difusos. A la luz de los resultados tan alentadores que se obtuvieron, el siguiente paso fue formalizar la metodología para poder dar tratamiento a plantas más complejas, tal y como fue presentado en las secciones anteriores. El ejemplo que se resuelve a continuación, intenta dar énfasis a la utilización del FIR que en nuestra metodología de diseño juega un papel central. Los detalles de los puntos (1) y (5) de la metodología no podrán ser estudiados con esta aplicación, ya que la obtención de la dinámica inversa para la planta de este caso es trivial, y no será requerido el uso de controladores auxiliares de ajuste y seguimiento. En los capítulos posteriores se proporcionarán ejemplos para plantas con un nivel de complejidad real, donde podremos profundizar el estudio de esos otros aspectos.

6.4.1 Descripción del Sistema

Se tiene una planta lineal de estructura SISO con la siguiente función de transferencia:

$$G(s) = \frac{s^2 + 3 \cdot s + 7}{s^2 + 5 \cdot s + 10} \tag{6.1}$$

Como puede verse en este caso, la estructura de la planta seleccionada tiene una función de transferencia *propia pero no estrictamente propia*, por lo que el cálculo de la dinámica inversa resultará trivial. El objetivo es diseñar un controlador difuso para esta planta tal que el sistema completo se comporte similarmente a un sistema lineal de referencia con la función de transferencia siguiente:

$$G_{\text{ref}}(s) = \frac{1}{s+1} \tag{6.2}$$

Obviamente, este objetivo puede ser alcanzado de manera precisa utilizando el controlador clásico mostrado en la figura 6.8.

donde:

$$G_{\text{control}}(s) = \frac{s^2 + 5 \cdot s + 10}{s \cdot (s^2 + 3 \cdot s + 7)} \tag{6.3}$$

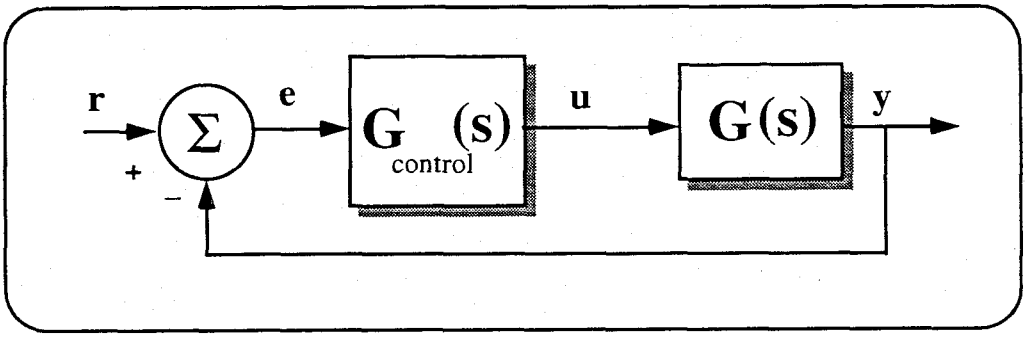


Figura 6.8: Diseño de Control Clásico

En lugar de esto, se utilizará un FIR-controlador con la misma estructura que fue mostrada en la figura 6.6.

6.4.2 Obtención del Modelo en Cascada

Ya que en esta aplicación la obtención de la dinámica inversa resulta trivial (utilizando el inverso de la función de transferencia de la planta), la etapa de generación del modelo en cascada es innecesaria y podemos pasar directamente a la etapa de generación de los datos para la obtención del modelo cualitativo del controlador.

6.4.3 Obtención del Modelo Cualitativo del Controlador

6.4.3.1 Obtención de Datos

El objetivo de esta etapa es la obtención de los datos a partir de los cuales el proceso de modelado se inicia. Como resultado de esta etapa se obtendrá la *matriz primitiva de datos cualitativos.* Esta matriz es obtenida partiendo de las señales observadas y medidas y posteriormente convertidas por la función de codificación difusa de SAPS-II según el experimento que se ilustra en la figura 6.9. Siguiendo las reglas dadas en la sección 4.2.1 para la obtención de datos, una señal aleatoria binaria, r, es aplicada a la entrada del modelo formado por el sistema de referencia, $G_{ref}(s)$, que indica el comportamiento que se desea del

sistema controlado en bucle cerrado, y_{des}, junto con el modelo de la dinámica inversa de la planta, $G^{-1}(s)$, que produce la señal de control óptima, u_{opt}, que debe ser aplicada a la planta. Este tipo de señal aleatoria binaria excita al sistema óptimamente bien a lo largo de todo el rango de frecuencias a las que responde el sistema. El sistema será simulado durante un período determinado de tiempo y muestreado con un intervalo seleccionado previamente. Las tres señales, r, y_{des}, y u_{opt}, serán codificadas y almacenadas en una *matriz primitiva de datos cualitativos.*

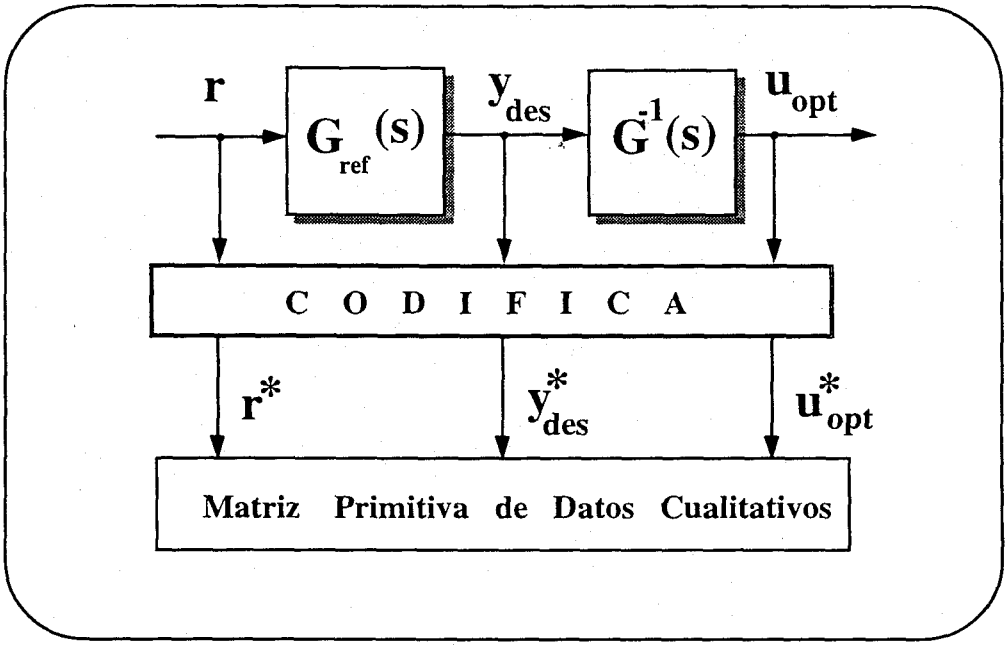


Figura 6.9: Experimento para la Obtención de los Datos.

La selección del intervalo de muestreo se realiza de acuerdo a las normas mencionadas en la sección 4.2.1. Para este ejemplo, el valor será obtenido a partir de la constante de tiempo más larga del sistema (es decir, la inversa del valor propio más lento del Jacobiano). Como la máscara óptima deberá cubrir la constante de tiempo más lenta del sistema en bucle cerrado (1.0 segundo), una máscara de profundidad de 3 sugiere que se debe usar un intervalo de comunicación de 0.5 segundos, es decir, la matriz primitiva de datos cualitativa deberá contener entradas cada 0.5 segundos para capturar la causalidad y el comportamiento cualitativo del sistema.

Desafortunadamente, la rutina predictora del FIR será capaz de proporcionar solamente un valor de la señal de control, u, cada intervalo de muestreo, causando que el sistema modelado, se comporte como un sistema de control digital de datos muestreados que reacciona discretamente a cada intervalo de muestreo, 0.5 segundos, en este caso. Desde la prespectiva de la teoría de control, esta reacción es muy lenta en relación con la dinámica de la planta, por lo que el sistema debe de ser muestreado con una rapidez mucho mayor si se desea que la estabilidad no se pierda. Siguiendo la técnica descrita en la sección 5.3.2 una reducción del muestreo a cada 0.05 segundos resultó ser suficiente para permitir que el controlador reaccione adecuadamente.

El siguiente paso en la obtención de los datos es encontrar el número de niveles discretos en los que debe de codificarse cada una de las variables. Para este ejemplo se decidió que un número de tres niveles permitiría caracterizar suficientemente bien cada una de las tres variables en cuestión. Una discretización de las variables de esta forma implica que el número de estados legales del sistema a codificar será de 27 ($3 \times 3 \times 3$). Como fue explicado en la sección 4.2.1, es deseable que cada uno de los estados legales sea observado al menos cinco veces. Consecuentemente, se requerirá un mínimo de 130 registros los cuales corresponden a una simulación de 65 segundos tomando en cuenta el intervalo de muestreo de la causalidad. Sin embargo, debido a la discrepancia señalada anteriormente entre el intervalo de muestreo indicado por la causalidad del modelo cualitativo (0.5 segundos) y el intervalo de muestreo requerido por las características de controlabilidad de la planta (0.05 segundos), serán necesario un número considerablemente superior de registros muestreados con intervalo más rápido. Basándose en esto, se decidió que el experimento de obtención de datos de la figura 6.9 tuviera un tiempo total de simulación de 100 segundos, con muestreo cada 0.05 segundos, donde 90 segundos serán utilizados para la identificación del modelo cualitativo del controlador, y se reservarán 10 segundos para la validación del modelo, es decir, que el módulo de máscaras óptimas dispondrá de los primeros 1800 registros para llevar a cabo la identificación del controlador difuso, mientras que los 200 últimos registros deberán de ser predecidos por el módulo de predicción difusa durante la etapa de validación. El juego de datos mostrado a continuación en la figura 6.10 fue obtenido mediante un programa de simulación en ACSL:

La codificación difusa fue realizada utilizando un programa en CTRL–C que hace las llamadas a la biblioteca de funciones de SAPS–II y que se muestra a continuación:

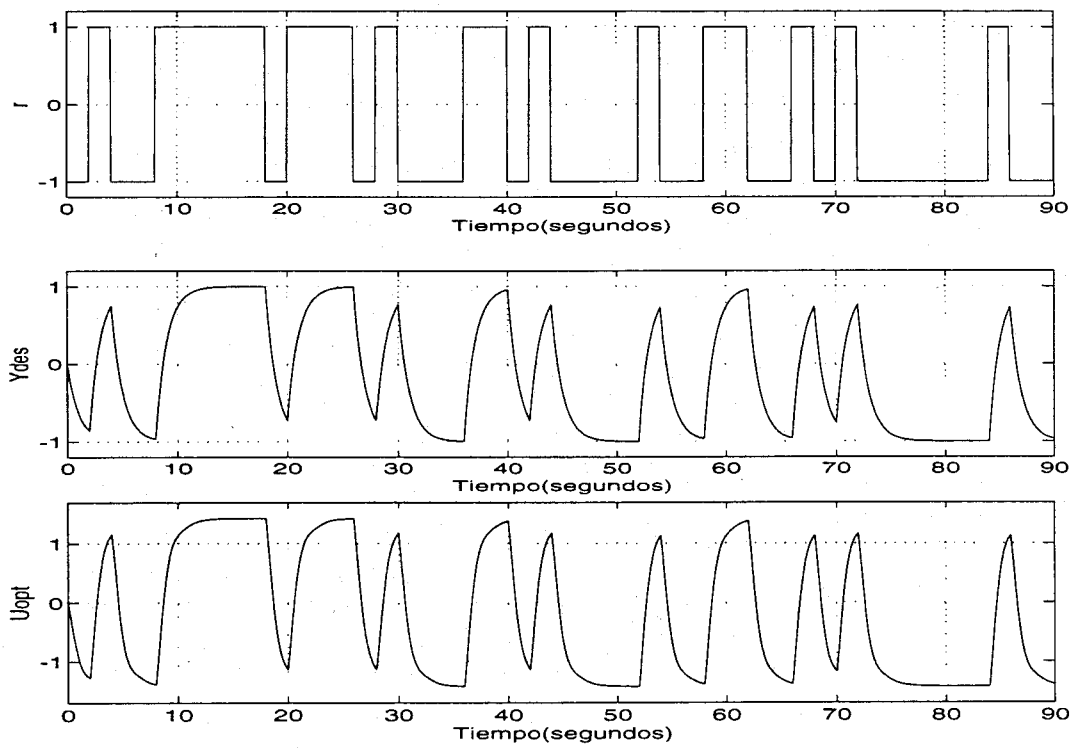


Figura 6.10: Juego de datos para el ejemplo

```
//  Codifica el sistema de manera optima

//  Se ordenan ascendentemente los valores de las trayectorias observadas

meas = yfile(1:2001,2:4);
m = meas;
FOR i=1:3, ...
  [indx,mi] = SORT(meas(:,i)); ...
  m(:,i) = mi; ...
END

//  Se cortan los vectores ordenados en n_lev segmentos de igual longitud y
//  se seleccionan los valores limite entre los segmentos vecinos

LM = [ m(1,:)
       0.5*(m(666,:) + m(667,:))
       0.5*(m(1333,:) + m(1334,:))
       m(2001,:) ];

//  Codifica los valores de las trayectorias observadas
```

```
to = 1:3;
FOR i=1:3, ...
  from = [ LM(1:3,i) , LM(2:4,i) ]'; ...
  [r,m,s] = RECODE(meas(:,i),'fuzzy',from,to); ...
  raw1(:,i) = r;  Memb1(:,i) = m;  side1(:,i) = s; ...
END
```

6.4.3.2 Obtención de la Máscara Óptima

Con los datos registrados y debidamente transformados en un formato cualitativo requerido por la función de síntesis de máscaras óptimas (sección 4.2.4), es posible obtener el conjunto de máscaras óptimas que caracterizan los mejores modelos posibles para el controlador difuso. Para tal efecto se decidió utilizar la siguiente matriz candidata:

$$
\begin{array}{l|ccc}
{}_t\backslash^x & r & y & u \\
t-20\delta t & -1 & -1 & -1 \\
t-19\delta t & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots \\
t-11\delta t & 0 & 0 & 0 \\
t-10\delta t & -1 & -1 & -1 \\
t-9\delta t & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots \\
t-\delta t & 0 & 0 & 0 \\
t & -1 & -1 & +1
\end{array}
\tag{6.4}
$$

de profundidad 21.

De acuerdo con la teoría de control, el valor seleccionado para el intervalo de muestreo δt ha sido de 0.05 segundos. Además, tal como sugiere la metodología de razonamiento inductivo, la entrada, u, al tiempo t dependerá de los valores pasados de r, y y u a los tiempos $t - 0.5$ y $t - 1.0$. La estructura de esta máscara candidata ha seguido las condiciones establecidas para la aplicación de la técnica de simulación mixta explicadas en la sección 5.3.2.

La máscara óptima encontrada con esta matriz candidata fue:

$$
\begin{array}{l\backslash x} & r & y & u \\
t-20\delta t & 0 & 0 & 0 \\
t-19\delta t & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots \\
t-11\delta t & 0 & 0 & 0 \\
t-10\delta t & 0 & 0 & 0 \\
t-9\delta t & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots \\
t-\delta t & 0 & 0 & 0 \\
t & 0 & -1 & +1
\end{array}
\tag{6.5}
$$

En otras palabras:

$$u(t) = \tilde{\mathrm{f}}(y(t)) \tag{6.6}$$

El código en CTRL–C que generó esta máscara óptima se muestra enseguida, en el se pueden observar las llamadas a la función FOPTMASK en la biblioteca de rutinas de SAPS–II:

```
//  Operacion del Analisis de Mascaras Optimas

//  Se extraen los primeros 1800 registros para la identificacion

rraw = raw1(1:1800,:);
MMemb = Memb1(1:1800,:);
sside = side1(1:1800,:);

//  Seleccion de la mascara candidata

mcan = ZROW(21,3);
mcan(1:10:21,:) = -ONES(3,3);
mcan(21,3) = 1;

//  Determinacion de la mascara optima

[mask,HM,HR,Q,mhis] = FOPTMASK(rraw,MMemb,mcan,6)
```

Desafortunadamente, esta máscara *óptima* no funcionará. Debido al acoplamiento que existe entre la entrada de la planta, u, y la salida de la planta, y, la máscara óptima sugiere que el conocimiento del valor presente de la salida de la planta, y, es suficiente para predecir el valor

óptimo de control a la entrada de la planta, u. En un esquema en bucle abierto, esto es correcto. Si $y(t)$ es dado, entonces $u(t)$ puede estimarse con exactitud mediante esta máscara óptima. Sin embargo, esto es como el problema del huevo y la gallina. Si $y(t)$ es dado, el valor de $u(t)$ puede ser obtenido, pero hasta no obtener $u(t)$, el valor de $y(t)$ no podrá ser calculado. Existe un bucle algebraico entre estas dos variables.

El hecho de que la planta fuera seleccionada como una función de transferencia *propia pero no estrictamente propia* hace que la solución del problema de encontrar la dinámica inversa de la planta sea una tarea fácil, pero, al mismo tiempo, convierte el problema de control difuso en un problema considerablemente más difícil. El algoritmo de máscaras óptimas optimiza la máscara para la configuración del sistema en bucle abierto. Si la planta tiene características de *filtro paso bajo*[2], la máscara óptima trabajará adecuadamente en una configuración de bucle cerrado. Sin embargo, en el ejemplo dado, algunas de las máscaras triviales (tales como la máscara *óptima* anterior) exhibirán un comportamiento pobre de seguimiento y otras mostrarán problemas de estabilidad.

En este caso, fue necesario buscar entre la historia de máscaras, es decir, entre el conjunto de máscaras subóptimas. Se encontró que la segunda máscara de complejidad cuatro, presentaba buen comportamiento tanto de estabilidad como de seguimiento. Tal máscara es:

$$
\begin{array}{l\backslash x} & r & y & u \\
t-20\delta t & 0 & 0 & -1 \\
t-19\delta t & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots \\
t-11\delta t & 0 & 0 & 0 \\
t-10\delta t & 0 & 0 & 0 \\
t-9\delta t & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots \\
t-\delta t & 0 & 0 & 0 \\
t & -2 & -3 & +1
\end{array}
\tag{6.7}
$$

o, expresado por una relación funcional:

$$u(t) = \tilde{\mathrm{f}}(u(t-1.0), r(t), y(t)) \tag{6.8}$$

[2]del inglés: low pass filter

6.4.4 Validación del Modelo Cualitativo

Como fue mencionado antes, los primeros 1800 registros (90 segundos) de la matriz primitiva de datos cualitativos fueron usados como datos históricos para la obtención de la máscara óptima. Para comprobar la calidad de predicción de esta máscara, la función de predicción difusa será utilizada para predecir los nuevos triples cualitativos para los últimos 200 registros (10 segundos) de la matriz primitiva de datos cualitativos. A partir de los triples cualitativos predichos, los correspondientes valores escalares pueden ser obtenidos mediante la función de regeneración de SAPS–II según se muestra en el siguiente código:

```
//  Prediccion del sistema durante 200 pasos

//  Copia de datos

rraw = raw1(1:2000,:);
MMemb = Memb1(1:2000,:);
sside = side1(1:2000,:);

//  Se destruyen los ultimos 200 registros

rraw(1801:2000,3) = ZROW(200,1);
MMemb(1801:2000,3) = 0.75*ONES(200,1);
sside(1801:2000,3) = ONES(200,1);

//  Se predicen nuevos valores para los ultimos 200 puntos

[frcst,Mfrcst,sfrcst] = FFORECAST(rraw,Mmemb,sside,mask,1800);

//  Se extraen los datos predichos.

meas = yfile(1801:2000,4);

//  Se regeneran las senyales continuas.

from = 1:3;
to = [ LM(1:3,3) , LM(2:4,3) ]';
rmeas = REGENERATE(frcdat,Mfrcdat,sfrcdat,from,to);
```

La figura 6.12 compara los valores "medidos" reales de u que se obtuvieron de la simulación original (línea continua) con los valores predichos y regenerados obtenidos mediante el FIR (línea discontinua) en bucle abierto, es decir, las trayectorias "medidas" en el tiempo de

$r(t)$ y $y(t)$ que fueron codificadas y almacenadas en las señales $r^*(t)$ y $y^*(t)$ fueron utilizadas por el módulo predictor de FIR para estimar la señal difusa de $u^*(t)$. Mediante la función de regeneración de FIR, se reconstruyó la señal escalar de $u(t)$, y ésta fue comparada con los valores de las trayectorias "medidas" previamente de $u_{\text{opt}}(t)$. La configuración de este experimento es la misma que fue presentada en la figura 6.5. En el programa en código CTRL-C del párrafo anterior pueden observarse las llamadas a las funciones de prediccción difusa y de regeneración en la biblioteca de funciones de SAPS-II.

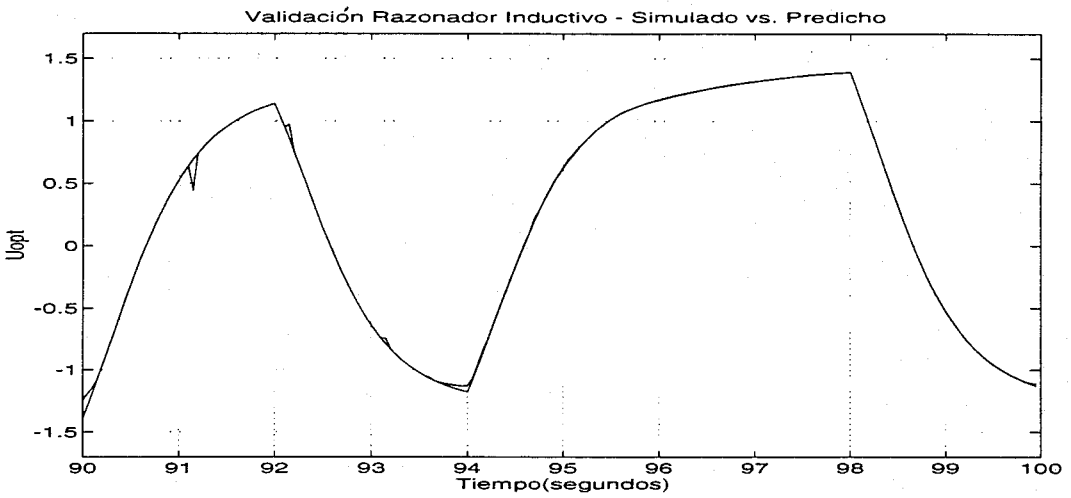


Figura 6.11: Validación del FIR-Controlador en bucle abierto.

Los resultados son realmente alentadores. Prácticamente no existe ninguna diferencia entre las trayectorias óptimas, u_{opt}, y la salida del controlador difuso, u, en bucle abierto. Es completamente claro que la máscara óptima contiene suficiente información para ser utilizada como un sustituto de la dinámica inversa real. Debe de resaltarse que el modelo FIR ha sido construido únicamente en base a los datos medidos.

6.4.5 Integración de los Modelos de la Planta y del FIR-Controlador

Verificada la capacidad predictiva del controlador difuso en bucle abierto, este puede ser insertado en dentro del sistema completo de control tal y como se mostró previamente en la figura 6.6. La señal de consigna de entrada al control, r, se convierte en el triple cualitativo, r^*, mediante la función de codificación difusa. De igual manera, la señal

escalar de la salida de la planta, y, es convertida en el triple cualitativo, y^*. A partir de estas dos señales cualitativas se predice el triple de valores cualitativos de la señal de entrada a la planta, u^*, por medio de la función de predicción difusa. Mediante la función de regeneración, esta señal cualitativa es reconvertida a la señal escalar, u. La planta en sí misma es descrita por medio de un modelo de ecuaciones diferenciales. Este experimento ha sido codificado en ACSL/SAPS de acuerdo a la descripción presentada en la sección 5.3.4. Mientras la dinámica continua de la planta será modelada utilizando un programa convencional escrito en ACSL, el subconjunto de funciones FIR de la interfase ACSL/SAPS permitirá el modelado del control difuso cualitativo.

La función de predicción difusa fue restringida a los últimos 200 intervalos de muestreo registrados, es decir, al tiempo comprendido entre los 90.0 y los 100.0 segundos. La figura 6.12 compara el valor de la salida deseada de la planta, $y_{\text{des}}(t)$, obtenida desde la simulación completamente cuantitativa (línea continua) contra el valor de la salida, y, del modelo cualitativo del controlador difuso (línea discontinua).

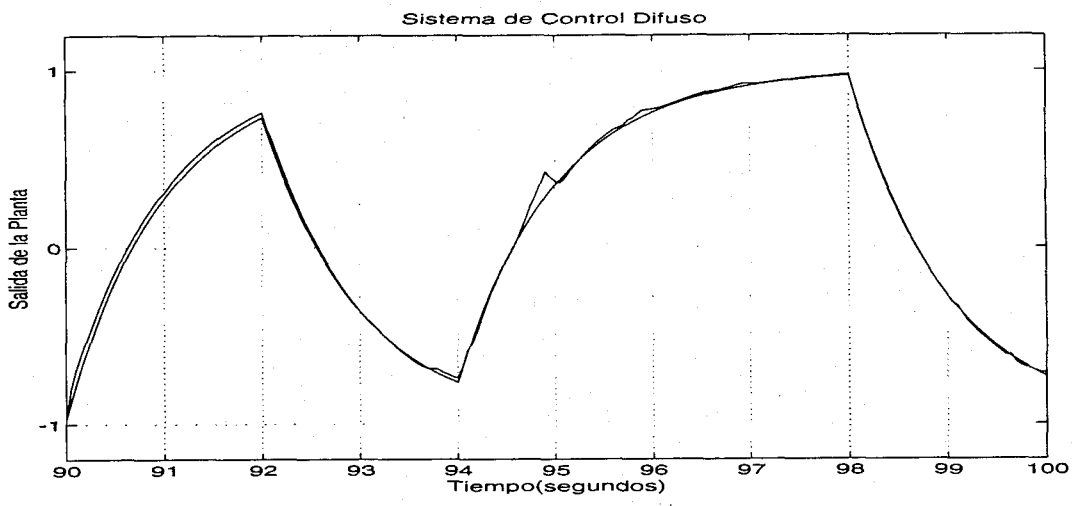


Figura 6.12: Comportamiento del FIR-Controlador en Bucle Cerrado.

Como puede verse en la figura, la planta con el controlador difuso, se comporta en efecto casi exactamente igual como la señal $1/(s+1)$ deseada. La nueva aproximación de diseño trabaja satisfactoriamente, aunque el acoplamiento directo entrada/salida en la planta ha hecho la tarea de diseño significativamente más difícil. El controlador difuso no presenta ningún problema ni de seguimiento ni de estabilidad, por lo que la etapa alternativa de ajuste mediante controladores convencionales simples no es necesaria en este ejemplo.

6.5 Conclusiones

En este capítulo se ha descrito, etapa por etapa, la metodología de diseño de controladores difusos utilizando como herramientas de diseño el razonamiento inductivo difuso (capítulo 4) y la técnica de modelado y simulación mixtos (capítulo 5). El ejemplo expuesto en la última sección, aunque simple, ha permitido probar la factibilidad de la aproximación propuesta para el diseño de controladores difusos.

A pesar de los buenos resultados alcanzados hasta este punto, la aplicabilidad de la metodología debió de ser probada en una forma más profunda mediante la aplicación de esta técnica a ejemplos más sofisticados. Como ya hemos apuntado antes, el razonamiento inductivo difuso también ha sido exitosamente aplicado para el modelado cualitativo de sistemas complejos y altamente no lineales [Albor 93a,93b], [Celli 92], [Nebot 93] aunque no en el contexto de diseño de controladores. Los esfuerzos posteriores se enfocaron en la aplicación de esta aproximación al diseño de controladores difusos como por ejemplo para la conducción de un barco de carga de gran tamaño. En el capítulo sigiente se darán más detalles de cómo la metodología puede ser aplicada para diseñar controladores para sistemas de ingeniería de complejidad real. Particularmente serán tratados los temas de síntesis del modelo en cascada y la adición de controladores convencionales para el ajuste fino de las propiedades de seguimiento y de estabilidad.

Algunos elementos del conocimiento utilizado en la implementación de los modelos mixtos, tales como la selección del intervalo de muestreo más apropiado o la estrategia de excitación de los sistemas, adolecen de la falta de sistematización al tener aún elementos de naturaleza heurística. Estos aspectos requieren ser estudiados con mayor profundidad para garantizar que la metodología es totalmente automatizable. Por otro lado, la aplicación de esta técnica para el diseño de controladores difusos para sistemas rígidos (sistemas cuyas constantes de tiempo más lenta y más rápida varían en órdenes de magnitud) necesita ser revisada con mayor detalle debido a los problemas de resolución de frecuencia múltiple inherentes en este tipo de sistemas. Aunque es una solución válida, no es práctico simplemente aumentar cada vez más y más la profundidad de la máscara.

Otro aspecto pendiente de solución sistemática es la determinación de aquellos elementos que deben de ser proporcionados al algoritmo de búsqueda de máscaras óptimas, para que ciertos tipos de máscaras no válidas desde el punto de vista estructural, como por ejemplo aquellas

que contienen bucles algebraicos, sean eliminadas como máscaras potencialmente óptimas, evitando así que opaquen las posibilidades de otras máscaras dentro del conjunto de máscaras subóptimas que guardan méritos superiores.

Capítulo 7

FIR-Controlador para un Barco de Carga

7.1 Introducción

En este capítulo se aplicará la metodología de diseño sistemático de controladores difusos, introducida en el capítulo anterior, al diseño de un piloto automático para un barco de carga de gran tamaño con el objetivo de demostrar que la metodología funciona bien cuando es aplicada a plantas altamente no lineales. La aplicación también nos permitirá mostrar el uso de razonadores inductivos acoplados en paralelo para resolver un sistema de control de tipo MIMO mediante múltiples controladores del tipo MISO. Finalmente, se usarán circuitos de control clásico para la estabilización y el seguimiento del sistema global.

Como ya hemos apuntado antes, el control de procesos complejos, tales como los sistemas altamente no lineales, los sistemas variantes en el tiempo o los sistemas de estructura variable, es todavía un tópico de gran interés ya que los controladores clásicos, tales como el PID, no trabajan adecuadamente bajo tales condiciones. Cuando se llega a utilizar este tipo de controladores para operar en plantas fuertemente no lineales, en sistemas sujetos a fuertes perturbaciones ambientales o en plantas con amplios puntos de operación, es necesaria una continua recalibración de los parámetros de cada controlador para que el rendimiento del control del sistema no se deteriore.

Los controladores adaptativos [Astro 89] han sido introducidos con el fin de automatizar el trabajo de reajustar los parámetros del controlador. Sin embargo, la primera generación de controladores adaptativos estaba

fundamentada en la idea de que la planta a controlar era lineal. Actualmente, una segunda generación de controladores adaptativos que incorporan técnicas de inteligencia artificial [deSil 91], técnicas de sistemas difusos [Layne 93] y/o tecnologías de redes neuronales [Naren 90] han sido introducidas para solucionar las limitaciones de la primera generación. Aunque estos controladores de la nueva generación tienen ventajas potenciales cuando las especificaciones de control requieren robustez, adaptabilidad y flexibilidad a perturbaciones del entorno o efectos no modelados de la dinámica de la planta, la aplicación de estas tecnologías presenta algunos inconvenientes. Uno de ellos es que, hasta ahora, este tipo de controladores suelen ser diseñados heurísticamente y de una manera *ad hoc* basándose en el conocimiento experto de un ingeniero de control o de un operador humano. A pesar de que algunos investigadores ven esta propiedad como una *ventaja* más que un inconveniente, ya que, al no estar basados ni en teorías sofisticadas ni en un modelo de la planta, son más robustos a efectos de dinámicas no modeladas, es precisamente esto lo que hace que la sintonización de controladores con múltiples entradas y múltiples salidas es una tarea aburrida y complicada.

Así, en este capítulo continuaremos abordando el diseño *sistemático* de controladores difusos, preservando las propiedades benignas de esta tecnología, simplificando el proceso de diseño y reduciendo drásticamente el tiempo necesario para la sintonización del controlador difuso para un rendimiento óptimo de control. En el capítulo anterior, la viabilidad de la técnica de diseño propuesta fue demostrada por medio del diseño de un controlador difuso simple para una planta lineal. En el presente capítulo, la metodología será aplicada al diseño del piloto automático de un barco de carga para demostrar que este enfoque también trabaja para plantas no lineales encontradas en aplicaciones del mundo real. La aplicación del barco de carga ha sido seleccionado debido al amplio número de publicaciones previas que tratan este sistema [Amero 75], [Astro 89], [Kalls 79,81] y [Layne 93]. Esto hace posible comparar los resultados obtenidos mediante esta nueva técnica de diseño de controladores difusos con los resultados obtenidos previamente y recientemente resumidas en [Layne 93].

7.2 Dinámica de la Dirección del Barco

7.2.1 Modelo Directo

La dinámica de un barco puede ser descrita por un modelo con dos variables de entrada, el ángulo del timón, δ, y el empuje del motor, f, y dos variables de salida, el curso del barco, ψ, y su velocidad, u. Sin embargo, para efectos de diseñar un piloto automático, este modelo es innecesariamente complejo. La mayoría de los pilotos automáticos para grandes barcos son diseñados para mantener el curso del barco en una dirección predeterminada a la vez de minimizar el consumo de combustible. Para ello, es suficiente trabajar con un modelo simplificado que tome en cuenta el ángulo de la posición del timón, δ, y la velocidad del barco, u, como entradas y el curso del barco, ψ, como única salida. El modelo fue propuesto originalmente por [Nomot 57]. La notación del sistema de coordenadas para el barco es ilustrada en la figura 7.1.

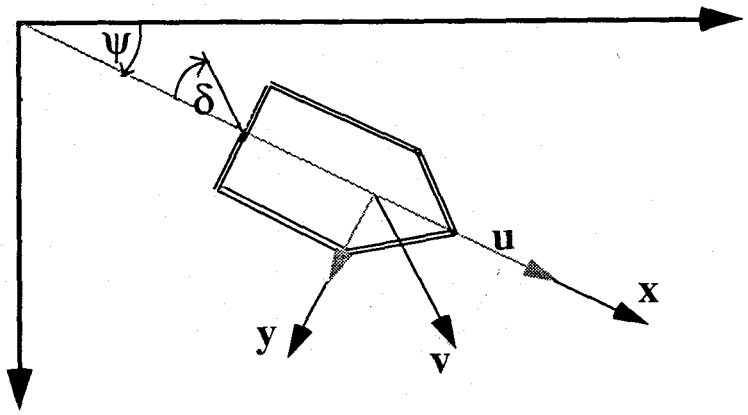


Figura 7.1: Sistema Coordenado del Movimiento del Barco

La dinámica del barco es capturada por una ecuación simple no lineal de tercer orden respecto al curso del barco, ψ. La dinámica del timón, $\dot{\delta}$, también es tomada en cuenta.

$$\tau_1 \tau_2 \dddot{\psi} + (\tau_1 + \tau_2)\ddot{\psi} + \dot{\psi} = K (\delta + \tau_3 \dot{\delta}) \tag{7.1}$$

donde τ_1, τ_2 y τ_3 son tres constantes que dependen de la velocidad del barco, u, y de la longitud del barco, l. K es un valor de ganancia que depende también de las mismas cantidades. Sin embargo, el modelo de Nomoto no es válido excepto para maniobras muy suaves con un ángulo de timón, δ, de menor de 5^o. [Bech 69] extiende esta formulación para permitir ángulos de timón superiores a los 5^o y modelar cursos inestables en la dinámica del barco introduciendo un término de amortiguación no lineal en función de ψ:

$$H(\dot{\psi}) = a\dot{\psi}^3 + b\dot{\psi} \tag{7.2}$$

en tal forma que, dividendo la ecuación 7.1 entre τ_1 y τ_2 y sustituyendo $(1/K)\psi = H(\dot{\psi})$, obtenemos:

$$\dddot{\psi} + (\frac{1}{\tau_1} + \frac{1}{\tau_2})\ \ddot{\psi} + (frac1\tau_1\ \tau 2)\ H(\dot{\psi}) = fracK\tau_1\ \tau 2(\delta + \tau_3\ \dot{\delta}) \tag{7.3}$$

En nuestro caso, esto se representará mediante la definición de la clase modelo *CargoShip* en lenguaje de Dymola [Elmqv 78], [Celli 91] mediante el siguiente código:

model class *CargoShip*

terminal *delta*, *u*, *psi*
local *psid*, *psi*2*d*, *psi*3*d*, *deltad*
local *H*, *K*, *tau*1, *tau*2, *tau*3
local *tau*1*inv*, *tau*2*inv*, *tau*12*inv*, *lu*

parameter $K0 = -3.86$, $tau10 = 5.66$
parameter $tau20 = 0.38$, $tau30 = 0.89$
parameter $l = 161.0$, $a = 1.0$, $b = 1.0$

$tau1inv = 1.0/tau1$
$tau2inv = 1.0/tau2$
$tau12inv = tau1inv * tau2inv$
$lu = l/u$

$K = K0/lu$
$tau1 = tau10 * lu$
$tau2 = tau20 * lu$
$tau3 = tau30 * lu$

$psid = \mathrm{der}(psi)$
$psi2d = \mathrm{der}(psid)$
$psi3d = \mathrm{der}(psi2d)$
$deltad = \mathrm{der}(delta)$

$H = (a * psid * psid + b) * psid$

$psi3d + (tau1inv + tau2inv) * psi2d + tau12inv * H- >$
$= K * tau12inv * (tau3 * deltad + delta)$

end

que es fácil de leer y comprender. El lector debe acordarse que, en Dymola, las ecuaciones especificadas son de tipo declarativo, es decir, la utilización del operador *der* no implica que $\dot{\psi}$ es calculado desde ψ por medio de una diferenciación numérica. Dymola, un compilador de modelos, puede generar un modelo de simulación en forma de ecuaciones de espacio de estados para varios lenguajes de simulación. En nuestro caso utilizaremos ACSL [MGA 86] para la simulación.

7.2.2 Modelo de Referencia

Supondremos que se desea que el comportamiento del sistema de control del barco resulte similar al que describe el siguiente modelo de referencia lineal de segundo orden:

$$\ddot{\psi}_m + 0.1\ \dot{\psi}_m + 0.0025\ \psi_m = 0.0025\ \psi_r \tag{7.4}$$

donde ψ_m representa la respuesta deseada del barco a la consigna, ψ_r, del curso. En lenguaje Dymola, se define una nueva clase de modelo llamada *Reference* mediante el código siguiente:

model class $Reference$

terminal $psir,\ psim$
local $psimd,\ psim2d$

$psimd = \mathrm{der}(psim)$
$psim2d = \mathrm{der}(psimd)$

$psim2d + 0.1 * psimd + 0.0025 * psim = 0.0025 * psir$

end

Tanto el modelo del barco (ecuación 7.3) como el modelo de referencia (ecuación 7.4) se utilizan de igual forma que los que fueron empleados en [Layne 93].

7.3 Obtención del Modelo en Cascada

En la sección 6.2 estudiábamos que para resolver el problema de la dinámica inversa de la planta teníamos que introducir en nuestro sistema de control un modelo de referencia que tuviese un número de polos suficientemente mayor que de ceros para que, cuando operase junto al modelo de la dinámica inversa, el sistema formado por ambos (el modelo en cascada) resulte por lo menos en un modelo *propio* (pero no necesariamente *estrictamente propio*). Ésta es precisamente la razón por la que se introdujeron modelos de referencias en los diseños de controladores (aunque muchas técnicas de diseño de controladores no lo dicen explícitamente). En el ejemplo del capítulo anterior, la solución de la dinámica inversa era trivial. La ejemplificación de la obtención del modelo en cascada se ha postergado hasta ahora en donde el modelo de la dinámica inversa de la planta es no propio y por lo tanto tendrá un comportamiento dinámico diferencial.

La estructura del modelo de la dinámica del barco expresada por la ecuación 7.3 consiste de tres polos y un cero. Por lo tanto, el modelo de referencia deberá contener dos polos más que ceros para conseguir que el sistema en cascada resulte al menos propio. Es por esta razón, que se decidió el modelo de referencia de la ecuación 7.4 cuya estructura tiene justamente dos polos y ningún cero. Tal y como fue descrito en la sección 6.3.1, podemos encargar a Dymola que realice el trabajo de encontrar el modelo en cascada mediante la siguiente definición de modelo *CascadeSystem*:

model *CascadeSystem*

submodel (*Reference*) *Ref*
submodel (*CargoShip*) *Plant*

input *psir*, *u*
output *delta*

Ref.psir = *psir*
Plant.psi = *Ref.psim*
Plant.u = *u*
delta = *Plant.delta*

end

El modelo *CascadeSystem* invoca instanciaciones de dos submodelos, el modelo *Ref* de la clase *Reference* y el modelo *Plant* de la clase *CargoShip*. El modelo *CascadeSystem* tiene dos variables de entradas externas (o funciones de manejo), *driving functions* en lenguaje Dymola), ψ_r y u,

y una variable de salida δ. Los subsistemas son *conectados* dentro del sistema mediante la instrucción $Plant.psi = Ref.psim$ que significa que las variables ψ de la planta y ψ_m del modelo de referencia deberán de ser igualadas, la dirección de la asignación, $Plant.psi \rightarrow Ref.psim$ o $Plant.psi \leftarrow Ref.psim$, será la que Dymola establezca como correcta de acuerdo a lo que se ha declarado como entrada y como salida. El mismo sentido guardan las demás asignaciones.

La planta será conectada con el modelo en cascada en una forma *aparentemente errónea* solicitando a Dymola que manipule la formulas algebraicas para que, tomando como entradas ψ_r que es la salida *natural* de la planta y u que es la velocidad de avance del barco (normalmente constante), se obtenga como salida δ que es la entrada *natural* de la planta. Sin embargo, antes de dar a Dymola la tarea de encontrar el modelo en cascada debemos abrir un paréntesis en el desarrollo de nuestro diseño para considerar un problema que aún está pendiente. En la formulación del modelo dinámico directo de la planta, la deflexión del timón, δ, fue empleada como una entrada. Sin embargo, el modelo en sí mismo contiene también la variable $\dot{\delta}$ la cual debe de ser calculada en alguna manera. Una primera solución a este problema es obligar al programa de simulación a diferenciar numéricamente la variable δ. Sin embargo, no debemos de perder de vista que δ será una variable calculada cualitativamente cuando el modelo en cascada haya sido obtenido, el controlador difuso haya sido identificado y el bucle se haya cerrado. Aunque esta es de hecho una solución válida, sus resultados serán dudosos, ya que esta situación es una fuente potencial de inestabilidad.

En algunas referencias el problema ha sido abordado ignorando el término de la dinámica del timón, es decir, fijando $\dot{\delta} = 0$, basándose en la suposición de que la dinámica del timón es considerablemente más rápida que la dinámica del barco por lo que puede ser despreciada. Sin embargo, cuando hemos adoptado esta solución, se observa que las deflexiones del timón exceden frecuentemente los límites físicos permisibles de 37°. La deflexión del timón usada para controlar el barco debe ser mantenida dentro del rango permisible mediante el uso de limitadores fuertes entre el controlador y la planta. Por supuesto, esto introduce un error acumulativo que degrada el rendimiento del control por debajo de los límites aceptables.

Una segunda solución consistiría en reemplazar la variable de entrada, δ, por su derivada, $\dot{\delta}$, de tal manera que podemos obtener δ mediante la integración numérica de $\dot{\delta}$ dentro del modelo de la planta. Este enfoque, no obstante, no trabaja muy bien, debido a que el acoplamiento entre $\dot{\delta}$

y la salida de la planta, ψ, es menos fuerte que el acoplamiento entre $\dot{\delta}$ y δ. Como será mostrado más adelante, esta aproximación debilitará el desempeño del controlador difuso más allá de lo recuperable.

Un tercer enfoque es quitar la relación entre δ y $\dot{\delta}$ dentro del modelo de la planta, eliminando la ecuación $der(delta) = deltad$. *Delta* y *deltad* serán tratadas como dos variables de entrada "independientes", diseñándose dos controladores separados (paralelos): uno obtendría los valores de *delta*, y el otro encontraría los valores de *deltad*. Esta solución, además de trabajar muy bien, nos permitirá mostrar el uso de múltiples razonadores inductivos trabajando en paralelo para la acción de control de un mismo sistema, ya que de hecho la planta se ha convertido ahora en una planta tipo MISO.

Ante esta solución, un lector observador puede preguntarse cómo pueden quitarse las relaciones entre δ y $\dot{\delta}$ en un barco real. ¿Cómo deberíamos de mover el timón para lograr *este* truco? Es obvio, que esto no se puede hacer. Sin embargo, es posible diseñar un controlador tipo PID que tome las señales de entrada *delta* y *deltad*, sugeridas por los dos controladores difusos, y generar una sola señal de salida $\dot{\delta}$, tal que $\dot{\delta}$ se acerque lo más posible a *deltad*, permitiendo llevar δ hacia *delta*. Este controlador PID es innocuo dado que sólo corrige pequeños errores de segundo orden. Consecuentemente no se requerirá efectuar ningún ajuste para diferentes puntos de operación. La inserción de este controlador se realizará más adelante cuando se integre el controlador difuso con la planta.

Una vez resuelto el problema, podemos cerrar el paréntesis abierto y avanzar en el diseño de nuestro controlador difuso en el punto donde nos quedamos. Formulados los modelos de la planta y de referencia, y habiéndolos conectado en sentido *aparentemente erróneo*, permitamos ahora que Dymola obtenga el modelo en cascada. Debido a la conexión *aparentemente errónea* de los modelos se producen restricciones en las variables diferenciadas, y el modelo resulta en un sistema DAE[1] de índice de perturbación alto [Bren 89]. Como las variables involucradas no pueden ser variables de estado con sus propios valores iniciales independientes, es necesario averiguarlo. El índice de perturbación del sistema, que en este caso es de 4, nos dice cuántas veces ciertas ecuaciones requieren ser diferenciadas para permitir resolver las derivadas. Dymola reduce en forma automática el índice de perturbación de los modelos, generando un nuevo modelo de índice reducido, para lo cual determina qué ecuaciones deben ser diferenciadas y, una vez hecho, las añade al

[1] Differential–Algebraic system of Equations

conjunto original de ecuaciones extraídas de los modelos interconectadas, hasta que se obtenga un sistema de ecuaciones de índice 1 [Celli 93].

Asumiendo que los modelos de la planta, de referencia y en cascada, definidos anteriormente, se han almacenado en el fichero *ship.dym*, observemos paso a paso el proceso que Dymola realiza. Si se desean consultar más detalles respecto a cada operación, en el apéndice C se incluyen los listados de cada parte en los cuales se incluye el estado de solución de las ecuaciones del sistema conjunto.

enter model
@*ship.dym*

partition

El comando *partition* dice a Dymola que encuentre la secuencia de ecuaciones de sistemas mínimos que requieren ser solucionados simultáneamente. Normalmente (a menos que se desee hacerlo) todas las ecuaciones triviales del tipo $b = a$ son eliminadas, y solamente el conjunto de ecuaciones no triviales y variables no redundantes permanece. En el caso que nos ocupa Dymola produce los resultados siguientes:

PARTE 1

— Additional trivial constraint equation relating known variables:
system. $ship.psi = reference.psim$
— The number of non–trivial equations is 12.
— The number of unknown variables is 13.

— Unassigned variables:
system.delta

El sistema es irresoluble al contar con 12 ecuaciones y 13 incógnitas. El problema es que Dymola asume que las salidas de los integradores son variables de estado y, por eso, son conocidas. Por consecuencia descubre que una de las ecuaciones no contiene ninguna variable desconocida (la ecuación de restricción entre las variables de estado) y enseguida le falta una ecuación para evaluar una de las variables incógnitas.

La variable sin resolver en nuestro caso es precisamente la que requerimos del sistema en cascada, pero no se pudo encontrar ninguna ecuación para ella. Las variables que conectan a los subsistemas *ship.psi* y *ship.psim* se suponen conocidas, ya que al estar diferenciadas, por defecto, se asume

que son variables de estado. El sistema hace *lo que puede* y elimina todas las ecuaciones de derivadas resolviendo para el caso trivial las ecuaciones restantes. En el apéndice C parte 1 puede verse la solución. Esta situación sugiere que el sistema es de índice de perturbación alto. Debemos informar al sistema de que proceda a reducir el índice mediante las siguientes instrucciones:

set LogDeriv on

differentiate

El resultado de estas instrucciones es el siguiente:

PARTE 2

Equation needs to be differentiated:
system. $ship.psi = reference.psim$

Derivative:
$ship.derpsi = reference.derpsim$
Checking if differentiated equations need to be differentiated.

⋮

⋮

Equation needs to be differentiated:
system. $ship.derderpsi = reference.derderpsim$

Derivative:
$ship.derderderpsi = reference.derderderpsim$
Checking if differentiated equations need to be differentiated.

El comando *set LogDeriv on* instruye a Dymola que debiera reportar las ecuaciones y sus derivadas durante el proceso de diferenciación. Sólo se han incluido la primera y la última de las operaciones aquí. Los detalles pueden consultarse en el apéndice C parte 2. Dymola tuvo que diferenciar tres veces, lo que indica que el índice de perturbación del sistema de ecuaciones original era de 4. Cada diferenciación reduce el índice por uno, hasta que resulte un sistema de ecuaciones de índice 1. En este momento, aún pueden encontrarse bucles algebraicos de ecuaciones en variables que deben de evaluarse simultáneamente, pero no se encuentran más restricciones entre variables de estado.

Como Dymola ha incluido un conjunto nuevo de ecuaciones producto del proceso de diferenciación, se requiere utilizar nuevamente el comando:

partition

para recalcular la solución del sistema. Esta vez obtenemos:

PARTE 3

— The number of non–trivial equations is 13.
— The number of unknown variables is 15.

— Unassigned variables:
reference.psim
system.delta

Los detalles del estado de la solución se encuentran en el apéndice C parte 3. La nueva partición nos dice que el sistema no ha podido ser resuelto, pues aún existen algunas problemas que resolver. La razón es que Dymola se da cuenta que, como había restricciones entre variables de estado antes, solamente un subconjunto de las salidas de los integradores deben de declararse como variables de estado. Como no puede decidir cuáles son estas variables, deja esta decisión al usuario. Dymola no hace ninguna suposición más respecto a qué variables deben de ser tomadas como variables de estado después del proceso de reducción del índice de perturbación del sistema.

El mensaje nos dice que faltan dos ecuaciones. Pues dos variables deben de declararse como variables de estado para que Dymola las trate como conocidas. Así que mediante los comandos siguientes:

variable state *ship.psi*, *ship.psid*

partition
output solved equations

asignamos las variables de estado pertinentes. Como puede verse, *ship.delta* no fue incluida en la lista de variables de estado debido a que, como se explicó anteriormente, la interrelación entre δ y $\dot{\delta}$ fue eliminada. El comando *partition* no envía ningún mensaje ya que el sistema de ecuaciones fue particionado satisfactoriamente.

La instrucción *output solved equations* nos permite ver el sistema de ecuaciones resuelto:

PARTE 4

SORTED AND SOLVED EQUATIONS

ship. $H = (a * psid * psid + b) * psid$

system. $u = 5.0$

ship. $lu = l/system.u$
$K = K0/lu$
$tau1 = tau10 * lu$
$tau2 = tau20 * lu$
$tau3 = tau30 * lu$
$tau1inv = 1/tau1$
$tau2inv = 1/tau2$
$tau12inv = tau1inv * tau2inv$

reference.
$ship.derpsid = 0.0025 * system.psir -$
$(0.1 * ship.psid + 0.0025 * ship.psi)$
$derderderpsim = 0.0025 * system.derpsir -$
$(0.1 * ship.derpsid + 0.0025 * ship.psid)$

ship. $system.delta = (reference.derderderpsim +$
$(tau1inv + tau2inv) * derpsid +$
$tau12inv * H)/(K * tau12inv)$

END OF SORTED AND SOLVED EQUATIONS

En este caso, no aparecieron bucles algebraicos después de la diferenciación.

Finalmente, mediante el conjunto de instrucciones:

language *acsl*
outfile cascada.csl
output model

Dymola genera un programa de simulación en ACSL [MGA 86] para el sistema en cascada, después de conectar en forma apropiada las funciones de manejo (las "driving functions") tal y como se puede consultar en el apéndice C parte 5.

7.4 Identificación del Modelo Cualitativo del FIR-Controlador

El manipulador simbólico de fórmulas de Dymola ha sido capaz de derivar un modelo analítico cerrado del sistema en cascada, en base a

las descripciones independientes de la dinámica directa de la planta y del modelo de referencia deseado. Encontrado el modelo en cascada, que puede interpretarse ya como un controlador, un modelo cualitativo del modelo en cascada debe de ser identificado y validado para que se convierta en un controlador difuso que pueda ser integrado en el sistema de control en bucle cerrado.

7.4.1 Obtención de Datos

En esta segunda etapa del diseño, los dos modelos cualitativos de los controladores deben ser identificados. Lo primero que debe hacerse es obtener los datos del comportamiento de estos modelos en todos los rangos de operación y para todas las frecuencias de respuesta. Para hacerlo llevaremos a cabo el experimento ilustrado en la figura 7.2:

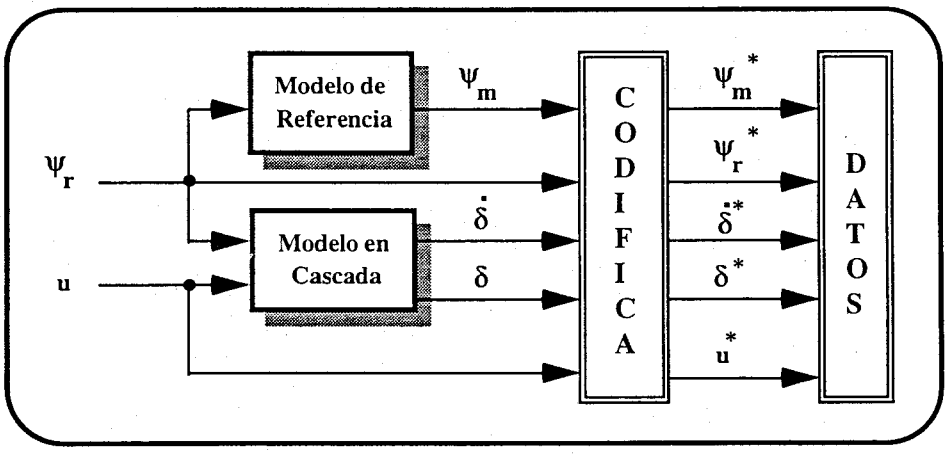


Figura 7.2: Identificación de los Modelos Cualitativos - Exp. 1.

De acuerdo a los requerimientos generales de identificación de modelos cualitativos dados en la sección 4.2.1, una señal de ruido binario aleatorio se aplica a la primera entrada del sistema, ψ_r, y una excitación sinusoidal a la segunda entrada del sistema, u. Se observan las reacciones del modelo de referencia, ψ_m, y de los modelos en cascada, δ y $\dot{\delta}$. Estas cinco señales son registradas y convertidas a señales cualitativas mediante la función de codificación difusa de SAPS–II descrita en la sección 4.2.2. Como resultado de esta transformación, obtenemos tres matrices de valores, una conteniendo los valores de clase, la segunda acumulando los valores

de las funciones de pertenencia y la tercera constituida por los valores de lado. Estas tres matrices conforman la *matriz primitiva de datos cualitativos*, en donde cada columna representa una variable y cada fila almacena un muestreo en el tiempo debidamente codificado.

De manera análoga a como se ha hecho en los casos anteriores, el intervalo de muestreo debe tomar en cuenta tanto los factores de causalidad como los de la estabilidad numérica. Desde el punto de vista causal sabemos que las constantes de tiempo lentas de la dinámica del barco son del orden de 2.0 segundos por lo que un intervalo de muestreo de 1.0 segundo será suficiente si suponemos una máscara de profundidad 3. Desde el punto de vista del control es necesario verificar la frecuencia con la que la planta debe ser alimentada para que la estabilidad no se pierda. Mediante un experimento similar al descrito en la metodología de simulación mixta en la sección 5.3.2, se encontró que el sistema debe recibir una señal al menos cada 0.2 segundos. Así que fijando el intervalo de muestreo en 0.2 segundos, podemos llevar a cabo el registro de los datos de entrenamiento mediante la ejecución del modelo de simulación en código ACSL, generado por Dymola en la sección anterior, con una duración de 7000 segundos. En este caso, los primeros 6000 segundos serán usados para identificar los dos modelos cualitativos, mientras que los 1000 segundos restantes del banco de datos de entrenamiento serán reservados para llevar a cabo las etapas de validación. Las señales utilizadas para formar el banco de datos de entrenamiento son mostradas en la figura 7.3.

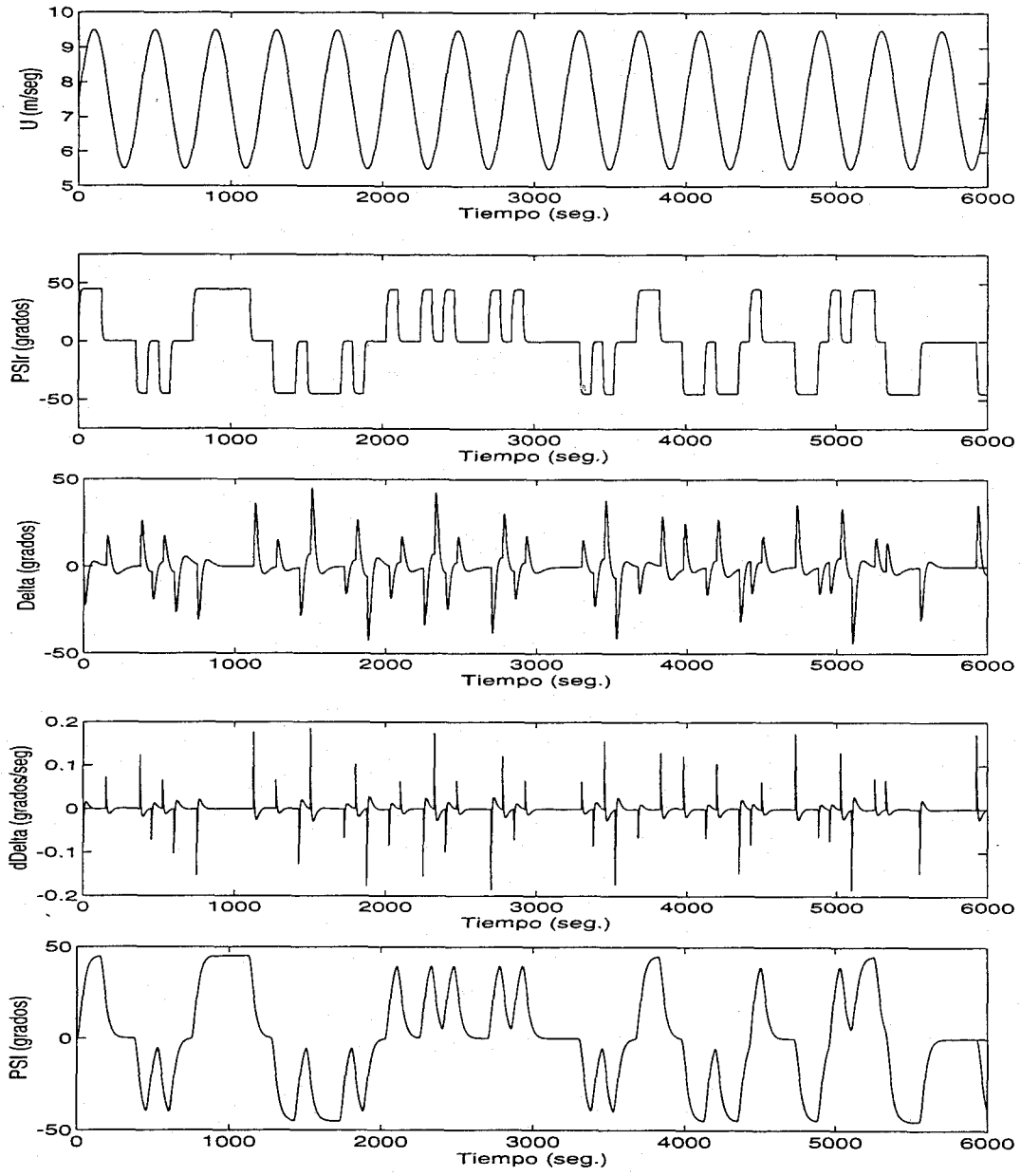


Figura 7.3: Banco de Datos de Entrenamiento.

7.4.2 Obtención de la Máscara Óptima

En la siguiente etapa, los dos modelos cualitativos, cuya topología se muestra en la figura 7.4, serán generados usando el análisis de máscaras óptimas, tal como se explica en las secciones 4.2.3 y 4.2.4. Como ahí se dijo, una máscara óptima es una relación de la causalidad temporal entre las variables de entrada y las variables de salida. La función de optimización está basada en la potencia de predicción de las posibles máscaras, computada a través del uso de la medición de la entropía de Shannon. Para satisfacer los criterios de causalidad y de estabilidad estudiados en la sección anterior, la profundidad de la máscara debió crecer a 11 para cubrir los 2.0 segundos de la constante lenta del sistema con un muestreo de 0.2 segundos, así que cada elemento de las filas primera, sexta y undécima de la matrices candidatas de los dos modelos cualitativos a identificar deberán de tener un valor -1 (con la lógica excepción de la variable de salida que se pone en $+1$) para conservar la recomendación de muestreo del aspecto de interrelación causal/temporal cada 1.0 segundo.

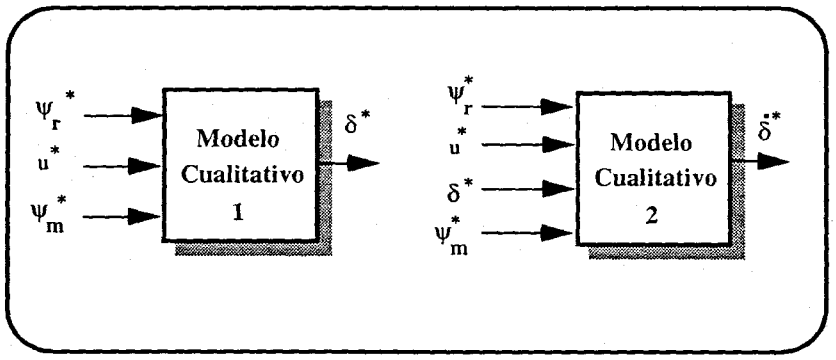


Figura 7.4: Topología de los Modelos Cualitativos

Las matrices óptimas encontradas reflejan que el triple cualitativo de la primera entrada requerida por la planta, δ^*, será obtenido como una función cualitativa de las dos entradas al sistema, ψ_r^* y u^*, y la salida del modelo de referencia, ψ_m^*, previamente almacenados en el banco de datos cualitativos. El triple cualitativo de la segunda entrada deseada a la planta, *deltad*, es encontrado como una función cualitativa de las dos entradas del sistema, ψ_r^* y u^*, y la variable δ^* previamente almacenados

en el banco de datos. Debe notarse que la máscara óptima para obtener *deltad* no utiliza la salida del modelo de referencia, ψ_m, es decir, no reconoce una dependencia causal muy fuerte. Ésta es la razón por la que en las soluciones previas de reemplazar δ por $\dot{\delta}$ como entrada de control para la planta, no funcionará bien.

Las máscaras óptimas encontradas son las siguientes. Para $\dot{\delta}^*$:

$$
\begin{array}{c}
{}_t\backslash^x \\
t-20\delta t \\
t-19\delta t \\
\vdots \\
t-11\delta t \\
t-10\delta t \\
t-9\delta t \\
\vdots \\
t-\delta t \\
t
\end{array}
\begin{pmatrix}
\psi_r & u & \psi & \delta & \dot{\delta} \\
0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -2 & 0 \\
0 & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & -3 & 0 \\
-4 & 0 & 0 & 0 & +1
\end{pmatrix}
\tag{7.5}
$$

para δ^*:

$$
\begin{array}{c}
{}_t\backslash^x \\
t-20\delta t \\
t-19\delta t \\
\vdots \\
t-11\delta t \\
t-10\delta t \\
t-9\delta t \\
\vdots \\
t-\delta t \\
t
\end{array}
\begin{pmatrix}
\psi_r & u & \psi & \delta & \dot{\delta} \\
-1 & 0 & -2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -3 & 0 \\
0 & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 \\
-4 & -5 & 0 & +1 & 0
\end{pmatrix}
\tag{7.6}
$$

7.5 Validación del Modelo Cualitativo

En la tercera etapa del diseño, los resultados de 1000 segundos de simulación cualitativa de ambos modelos serán comparados con los últimos 1000 segundos de la simulación cuantitativa desarrollada previamente con el fin de verificar la calidad de la predicción. El proceso es esquematizado en la figura 7.5 donde los bloques *CODIFICA* y *REGENERA* son los conversores Cuantitativo → Cualitativo y

Cualitativo → Cuantitativo de SAPS-II. Ya que estos últimos 1000 segundos nunca se han visto anteriormente, nos permiten evaluar en la misma prueba tanto la calidad (diversidad) de los datos muestreados como la capacidad de predicción del modelo en sí misma. Si cualquier de los dos aspectos fuera incorrecto se reflejaría justamente en esta etapa.

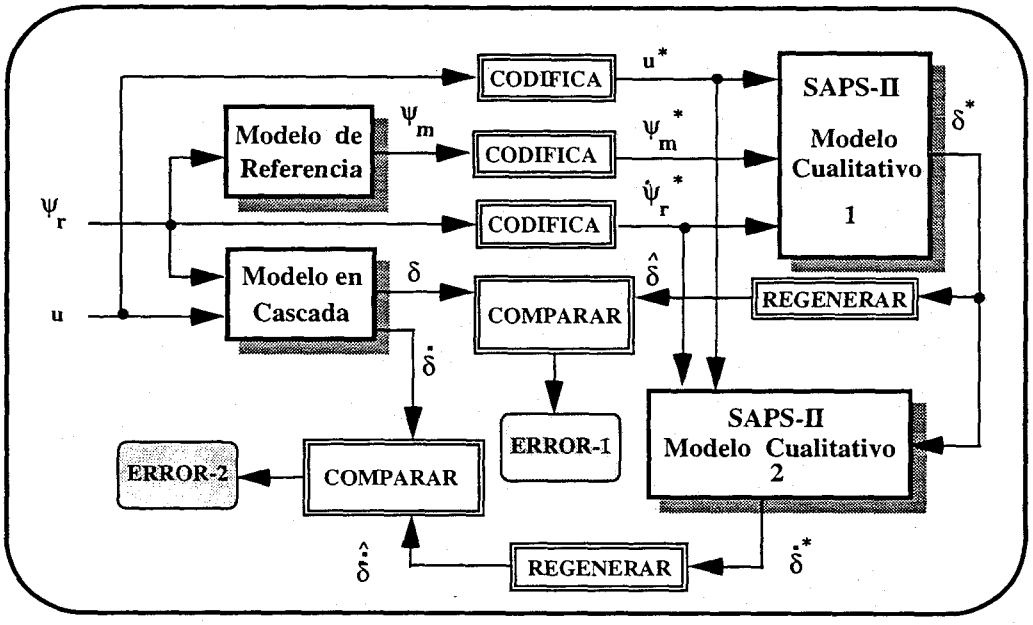


Figura 7.5: Esquema del Proceso de Validación

Los resultados de la etapa de validación se presentan en la figura 7.6. Las señales de error son mostradas dado que las señales originales son prácticamente indistinguibles una de otra. Los excelentes resultados obtenidos garantizan la calidad de predicción de ambos modelos cualitativos. Ya que esta es una operación en bucle abierto, es muy difícil eliminar del todo los errores que, como se muestra en la figura, tienen un pico de crecimiento justo en la perturbación (la máxima posible del sistema) y rápidamente tienden a decrecer.

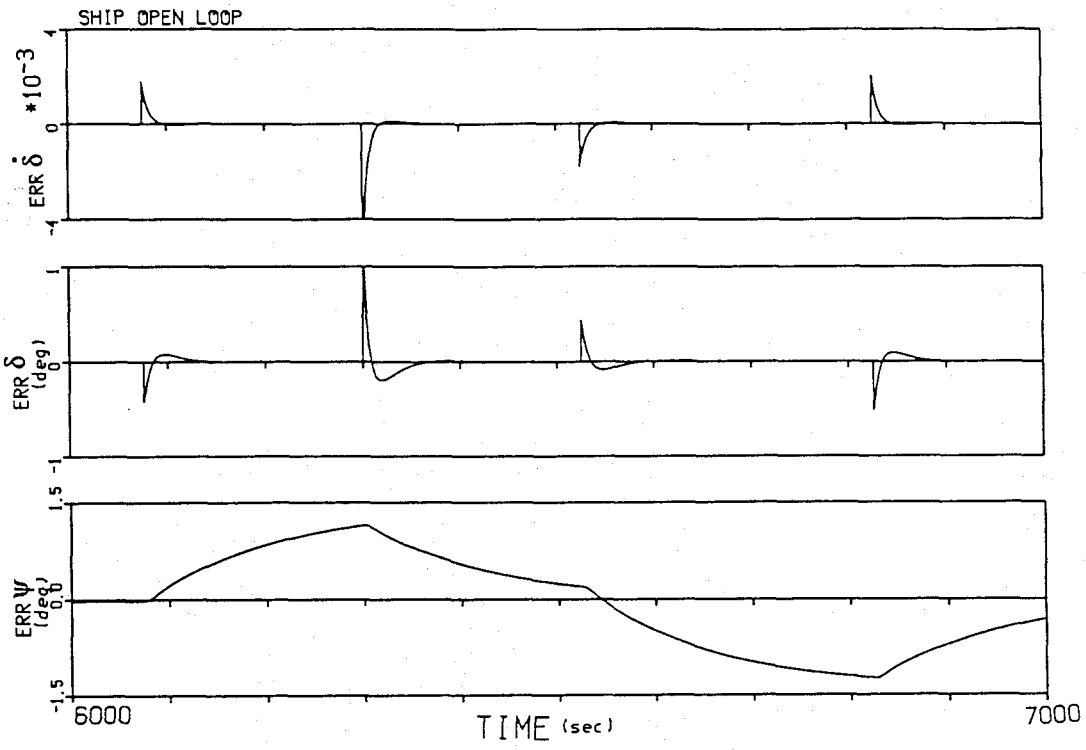


Figura 7.6: Resultados del Proceso de Validación

7.6 Integración de los FIR-Controladores con la Planta

Una vez que los errores en los experimentos anteriores lleguen a ser lo suficientemente pequeños, el bucle de control podrá ser cerrado de acuerdo al esquema mostrado en la figura 7.7.

Los resultados muestran que no existe un error directo en el cálculo entre la salida deseada, ψ_m, y la salida real, ψ, como en la mayoría de las otras configuraciones de control. El detalle es que, aunque el modelo cualitativo fue diseñado usando ψ_m en bucle abierto, usando ψ para reemplazar a ψ_m permite cerrar el bucle manteniendo el mismo objetivo global pero con nuevas propiedades relacionadas a la supresión de perturbaciones y reducción de la sensibilidad de los parámetros de la planta. En esta forma los modelos cualitativos diseñados anteriormente se han transformado en controladores difusos baja la nueva configuración.

Por otro lado, ¿que podemos decir sobre el comportamiento de fase no mínima? Los barcos grandes tienen, intrínsecamente, una tendencia a

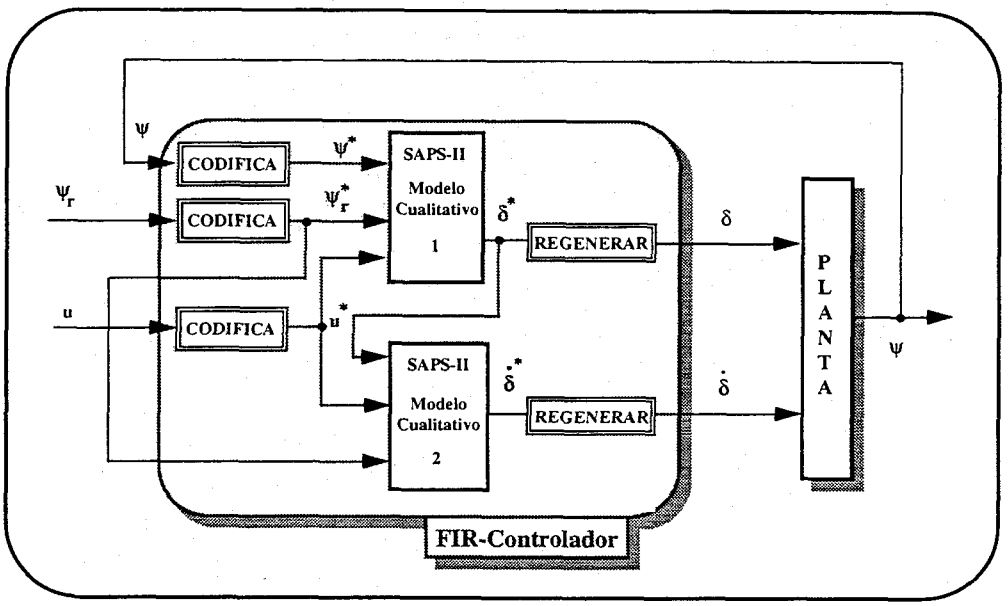


Figura 7.7: Configuración del Sistema FIR–Control en Bucle Cerrado

exhibir un comportamiento de fase no mínima. Cuando los pilotos giran el volante a la izquierda, el barco reacciona girando a la derecha, por lo que el piloto tiende a girar más a la izquierda y el barco reacciona girando más a la derecha [Clark 84]. La solución es nuevamente muy simple: se requiere un controlador local que mueva los ceros inestables hacia el semiplano izquierdo.

Un último punto queda aún pendiente. Bajo esta configuración no hay forma de decirle al controlador difuso que es importante que ψ se mantenga tan parecido a ψ_m como sea posible, ya que los FIR–controladores no tienen forma de conocer la señal ψ_m. Consecuentemente, la posición del barco irá poco a poco alejándose del curso deseado. Para resolver esto, podemos introducir un bucle de control de seguimiento, dentro del cuál, la diferencia entre la posición deseada del barco, ψ_m, y la posición real del mismo, ψ, sean realimentadas a la deflexión del timón.

El lector perspicaz podría sospechar que una versión del anticuado controlador tipo P ha sido introducido por la puerta trasera para favorecer que todo funcionará, mientras que el controlador difuso fue puesto en el sistema para justificar el nuevo método elegante y sofisticado.

Esto no es verdad. Aquí ocurre lo mismo que en una escuela de conducción. Es realmente el estudiante quien realiza toda la conducción del vehículo. El profesor sólo atiende de vez en cuando para efectuar una corrección en el volante y evitar que el coche tenga un encuentro poco amistoso con los árboles cercanos. Es realmente el controlador difuso que realiza todo el trabajo. La señal de realimentación que llega por el bucle de control de seguimiento es mucho más pequeña en magnitud que la señal que llega del controlador difuso. Tan sólo proporciona un incentivo adicional para que la planta permanezca en la ruta. La ganancia de realimentación, k_1, tiene un valor de $k_1 = 0.2$ que fue encontrado experimentalmente. Dado que el controlador P sólo corrige errores pequeños de segundo orden, no se requieren ajustes del punto de operación.

La misma argumentación en el párrafo anterior vale para la adaptación del controlador a la configuración real de las entradas a la planta mediante otro pequeño controlador tipo P. Simplemente medimos desde la planta la deflexión del timón y la comparamos con la obtenida por el controlador difuso respectivo. La ganancia de realimentación , k_2, también fue encontrado experimentalmente obteniendo un valor aún más pequeño de $k_2 = 0.1$. El resultado es agregado a la salida del control de seguimiento y añadido a la señal de control del otro controlador difuso. La configuración final del sistema se muestra en la figura 7.9 donde se destaca el componente difuso del controlador mediante un sombreado.

En la configuración del bucle de realimentación, el controlador difuso está preparado para corregir efectos de perturbaciones del entorno y dinámicas no modeladas de la planta. Sin embargo, si estos efectos llegan a ser considerables, el control puede deteriorarse. Por ejemplo, la dirección y velocidad del viento tienen un efecto sustancial en el comportamiento del barco, y consecuentemente una corrección indirecta podría no ser suficientemente buena. La solución es todavía posible y simple. Lo único que se debe hacer es modificar el modelo de la planta *CargoShip* de forma tal que éste contenga los efectos de las dos variables como entradas adicionales. Dymola, incluiría estas variables en el modelo en cascada tratándolas exactamente como se hizo con la velocidad del barco, u. La matriz primitiva de datos presentará dos columnas adicionales y el modelo cualitativo también, pero los datos anteriores seguirán siendo los mismos. Es obvio que, dado que el modelo cualitativo tiene ahora cuatro entradas diferentes, requeriremos de más datos de entrenamiento para el conjunto de la base de datos para el controlador difuso. Sin embargo, ya que todo esto se realiza fuera

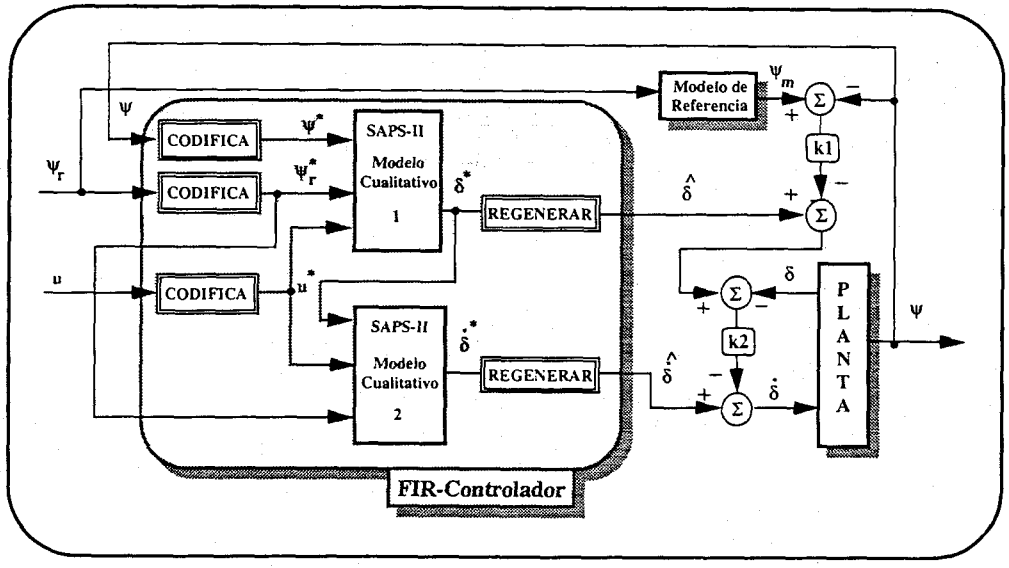


Figura 7.8: Configuración Final del Sistema FIR–Control

de línea, no trae demasiadas consecuencias a la operación del FIR–controlador.

Para comparar el rendimiento del FIR–controlador con otros diseños recientes tales como el *Fuzzy Model Reference Learning Control (FMRLC)* presentado en [Layne 93], se reprodujo el mismo experimento que emplearon Layne y Passino para medir el rendimiento de su diseño. Las características de este experimento, son las siguientes:

- constante de la velocidad del barco: 5.0 m/s,
- seis ciclos de perturbación de ψ_r como sigue:
  (1) curso de 45° hacia la izquierda durante 250 segundos,
  (2) curso de 0° durante los siguientes 250 segundos,
  (3) curso de 45° hacia la derecha en los siguientes 250 segundos, y
  (4) curso de 0° en los últimos 250 segundos del ciclo.

Los resultados de este experimento en bucle cerrado se muestran en la figura 7.19. Ambas respuestas lucen prácticamente idénticas, y las dos trabajan sustancialmente mejor que los controladores adaptativos con modelo de referencia de gradiente y Lyapunov [Layne 93]. Ambas técnicas evitan las grandes y poco realistas deflexiones del

timón. Además, tanto el FMRLC como el FIR–controlador han sido identificados con perturbaciones tanto en u como en ψ_r, y no se requiere reidentificar si la velocidad del barco cambia.

Con respecto a la cantidad de energía empleada por el controlador para obtener con exactitud el seguimiento del curso de referencia y utilizando la definición de $E = \delta^2$ [Layne 93], el FMRLC requirió $E = 17.3368$ durante 6000 segundos del experimento, mientras que el FIR–controlador solamente necesitó de $E = 13.111$ durante el mismo período.

7.7 Conclusiones

La arquitectura del controlador obtenido no es realmente un esquema de control con aprendizaje. Este es más bien un esquema de control óptimo que construye fuera de línea un conjunto óptimo de reglas difusas mediante a partir de un conjunto de datos de entrenamiento. La utilización del conocimiento disponible en la forma de un modelo directo cuantitativo de la planta es una ventaja. La dinámica directa de la planta es bastante fácil de adquirir, y en la medida en que el proceso de generar un modelo en cascada a partir de él el controlador difuso pueda ser automatizado, no hay nada incorrecto en utilizar este enfoque.

Más aún, debe ser fácil colocar sobre la arquitectura del controlador difuso presentado en esta tesis un esquema de adaptación de las funciones de pertenencia, similar al presentado en [Layne 93], en forma tal que el controlador pueda adaptarse a sí mismo para variaciones lentas de la dinámica de la planta. Tal y como se pudo sentir a lo largo de la exposición de este capítulo, de igual manera que en otras técnicas de diseño de controladores difusos, la metodología de los FIR–controladores sufre aún de cierto grado de heuristicismo.

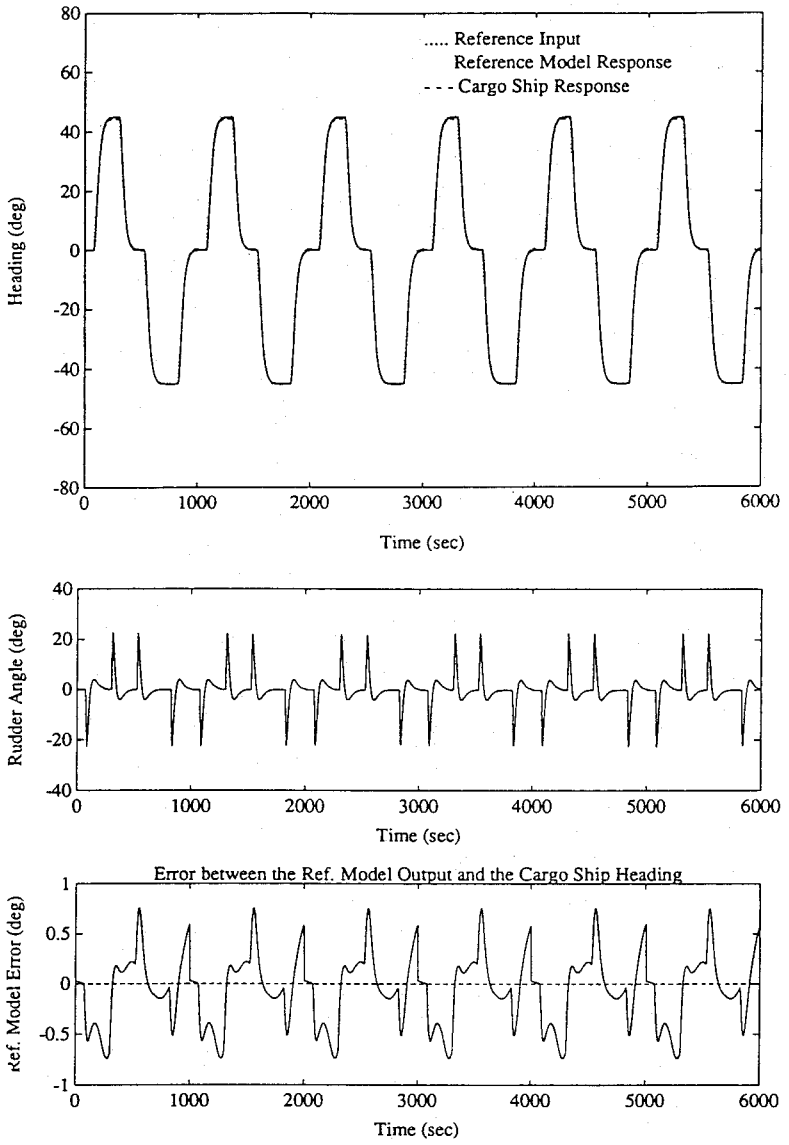


Figura 7.9: Comparación del Desempeño del FIR–Control

Capítulo 8

Conclusiones e Investigación Abierta

8.1 Introducción

La primera generación de controladores con realimentación fue desarrollada con el propósito de mejorar las características de estabilidad y precisión en torno al estado estacionario y para reducir la sensibilidad a las perturbaciones del entorno de sistemas tales como aviones o sistemas para la generación y distribución de energía eléctrica. Aunque la operación de estos controladores resultó adecuada, no existe aún ninguna teoría de diseño que sea sistemática excepto para plantas lineales o cuasilineales. Las no linealidades de la gran mayoría de plantas reales resultaron en un reto formidable para los ingenieros de control. Un método para tratar este tipo de problemas es mediante la parametrización de la estructura del controlador e identificar los valores óptimos de los parámetros del controlador por medio de algoritmos de optimización.

Esta aproximación paramétrica se intentó generalizar mediante la introducción de arquitecturas basadas en redes neuronales. En lugar de asumir una estructura del controlador específica para el proceso que debe de ser controlado, se plantea una estructura muy general y flexible que, al menos en principio, permita representar cualquier sistema que exhiba un comportamiento arbitrario, es decir, se propone una estructura con suficientes parámetros de forma tal que le permitan operar bajo cualquier situación. La solución de esta arquitectura tiene que ver con el entrenamiento de los pesos de las neuronas, o dicho de otra manera, con

la optimización de los parámetros del controlador, la cuál puede resultar extremadamente lenta. Desafortunadamente no se conoce ninguna teoría que garantice la convergencia de los algoritmos de optimización para cada caso y que converja en un óptimo global.

La solución mediante redes neuronales resulta poco satisfactoria porque el proceso es costoso, lento y poco fiable. En contraste con esta solución, los operadores humanos son capaces de controlar procesos altamente no lineales de una forma fácil, natural, fiable y rápida, sin necesidad de resolver complejos problemas de optimización. La búsqueda de emular estas capacidades humanas mediante alguna estructura de controladores con realimentación llevó a la introducción de los controladores difusos.

Los controladores difusos operan de manera similar a como lo hacen los controladores de lógica programables[1], con la excepción de que en vez de utilizar los principios de la lógica clásica, utilizan la lógica difusa. La utilización de la lógica difusa capacita a estos controladores para operar también bajo condiciones de conocimiento impreciso o incompleto, permitiéndoles la posibilidad de interpolar entre los diferentes valores discretos de la salida, lo que en la técnica de control difuso se conoce como proceso de *defusificación de la salida*.

Las propiedades de los controladores difusos son en efecto similares a las de los operadores humanos. Son controladores no paramétricos que funcionan bien para plantas altamente no lineales. Al menos para plantas del tipo SISO con un comportamiento de fase mínima, resulta bastante fácil e intuitivo el desarrollo de un conjunto adecuado de reglas bajo las cuales el controlador funcione correctamente, aunque no necesariamente de una manera óptima, sin necesidad de optimizar el diseño. Con el objetivo de mejorar aún más el comportamiento de los controladores difusos, se han introducido conceptos como la postoptimización del contorno de las funciones de pertenencia, en la que una parte de la estructura del controlador es parametrizada por la puerta trasera. Este proceso es similar al entrenamiento de un piloto humano quien, aunque ya conoce la teoría del vuelo, requiere de la experiencia práctica de pilotar un avión durante muchas horas antes de que pueda conducir el avión con confianza y recibir el permiso de volar con pasajeros.

Sin embargo, desarrollar el conjunto de reglas difusas para plantas de fase no mínima o para plantas con múltiples entradas y salidas (MIMO), no resulta nada obvio. El número de reglas que se necesitan para controlar plantas del tipo MIMO crece rápidamente, con el agravante de que, en

[1] del inglés: "programmable logic controllers" (PLC).

el caso de plantas de fase no mínima, las reglas resultantes pueden ser contraintuitivas. Bajo tales condiciones, aún un operador humano tiene exactamente las mismas dificultades para resolver el problema de control. Es precisamente por esta razón que los vehículos que se diseñan para ser operados por seres humanos, siempre han de ser sistemas de fase mínima y, si el vehículo tiene múltiples entradas, los efectos de control por media de estas entradas deben ser desacoplados como sea posible.

La necesidad de superar las dificultades que los controladores difusos del pasado han tenido para tratar adecuadamente sistemas tipo MIMO y/o sistemas de fase no mínima motivó el desarrollo de esta tesis de doctorado que propone una nueva metodología para el diseño sistemático de controladores difusos para plantas con propiedades arbitrarias.

8.2 Resumen de los Resultados Obtenidos

En el capítulo de introducción, nos habíamos planteado como objetivo central de esta tesis desarrollar una metodología que permita la construcción de controladores difusos de una manera sistemática, de tal modo que, sin perder las cualidades benignas inherentes a los modelos cualitativos como la *flexibilidad* y *robustez*, el sistema de control presente un comportamiento suficientemente preciso para mantener la estabilidad y el seguimiento del estado deseado. Como se ha visto a lo largo de la tesis y resumido en la sección anterior, la falta de sistematización en el desarrollo de controladores difusos es quizás la desventaja más reportada de esta tecnología. Es claro que toda nueva metodología que surja debe contar con algún tipo de aportación en esta dirección.

¿Qué es lo que hace tan difícil sistematizar el diseño? Revisando los principios subyacentes del diseño de controladores, tal y como se expuso en el capítulo 6, nos hemos percatado que todas las técnicas de diseño, de una u otra manera, requieren resolver el problema de la dinámica inversa de la planta, es decir, dado un comportamiento deseado, debemos decidir, qué entrada debe de aplicarse a la planta, para que se comporte como se desea que lo haga. Este es el problema que un operador humano resuelve a cada momento cuando debe de aplicar su *conocimiento* de la planta y su *habilidad* para controlarla. Este es también el problema subyacente en las técnicas clásicas y adaptativas. Es claro que un proceso sistemático de diseño de controladores no puede partir de la base de que siempre encontrará disponible el conocimiento y la habilidad de un experto que le transfiera su resolución particular del problema de

la dinámica inversa de la planta y que este conocimiento pueda ser debidamente representado. Buscando rodear este problema, la mayoría de las actuales propuestas para diseñar controladores difusos han optado por regresar a los conceptos y estructuras básicos de los controladores clásicos, de tal manera que la dinámica inversa de una planta genérica es postulada bajo la regla genérica: *a mayor desviación del comportamiento deseado, la acción de control ejercida deberá de ser mayor*. Así que de una manera casi trivial, para el caso más simple se construye una base de reglas basada en una tabla en la que se relaciona cada *estado* posible del error de la planta ('grande+', 'medio+', 'cero', 'medio-','grande-') con cada una de las posibles acciones de control discretizadas similarmente. La relación es establecida mediante un peso, o una confianza, o el valor difuso que debe ser estimado por el diseñador del controlador. Tanto la discretización como la asignación de pesos se lleva a cabo de forma completamente heurística, lo que impide su sistematización. En casos más complejos, los pesos son conjuntos difusos que se estiman de acuerdo al nivel de correspondencia entre los patrones de entrada y las reglas. No existe más conocimiento de la dinámica inversa de la planta que la prueba y el error de los parámetros que el diseñador experimenta.

Por el contrario, la metodología desarrollada en esta tesis es capaz de combinar el conocimiento de los modelos analíticos con la habilidad que reflejan las técnicas difusas para tratar imprecisión. A diferencia del caso mencionado anteriormente, esta metodología encuentra un modelo 'real' de la dinámica inversa de la planta, aunque expresado de manera implícita dentro de otro modelo, que denominamos 'modelo en cascada'. Dentro del modelo en cascada se encuentran, en conjunción y en una estructura de *bucle abierto*, tanto un modelo de referencia que representa el comportamiento deseado del sistema, como el propio modelo de la dinámica inversa de la planta. Esta parte del proceso, además de *sistemática*, es diferente del caso de los controladores difusos mencionados anteriormente, porque no funciona en base a la medida del error. Opera *directamente* en base a los estados de las variables de la planta, con lo cual ni pierde sensibilidad respecto a los puntos de operación, ni padece de la falta de persistencia en la excitación. Una vez obtenido el modelo inverso de la planta, el modelo en cascada es caracterizado como un modelo cualitativo. Esta transformación permite moverse de una arquitectura en bucle abierto a una de bucle cerrado. Esta parte del proceso también está suficientemente sistematizada. Salvo en plantas extremadamente complejas o inherentemente inestables, el proceso no requerirá más que de añadir en algunos casos controladores tipo P innocuos, tal y como fue justificado en la descripción de la metodología en el capítulo 6.

La hipótesis fundamental de esta tesis era que los controladores difusos pueden ser desarrollados sistemáticamente mediante la metodología del *razonamiento inductivo difuso*[2]. Desde otra perspectiva, el controlador difuso desarrollado en esta tesis puede verse como un controlador discreto del tipo PLC con un mecanismo de interpolación interno que utiliza las funciones de pertenencia difusas. El controlador óptimo es buscado en el espacio discreto de las clases. Ya que tanto los valores de clase de las variables de entrada del controlador difuso como el valor de clase de salida son señales discretas con pocos valores, se puede interpretar al controlador como una *máquina de estados finitos*[3]. Para capturar la dinámica del controlador, se pueden añadir valores pasados de las entradas y la salida como entradas adicionales de la FSM. Si se usan suficientes variables como entradas a la FSM, debe ser posible encontrar un comportamiento de la FSM totalmente determinista. Si no se usan suficientes variables, el comportamiento de la FSM será ambiguo. Por otra parte, si se utilizan demasiadas variables, la FSM no será exhaustiva a menos que cuente con una gran cantidad de datos. En FIR se usa un *mecanismo de abstracción* (que llamamos búsqueda de máscaras óptimas en el capítulo 4) para identificar un subconjunto de variables que permite encontrar un comportamiento que es casi tan determinista como exhaustivo, reduciendo a un mínimo el volumen de datos necesarios para la identificación del controlador.

En el caso de plantas inherentemente inestables o de fase no mínima (con polos o ceros del lado derecho del plano), la situación es un poco diferente ya que la sistematización es menos completa. Sin embargo, es complejo para todas las metodologías. Los polos o ceros **deben** necesariamente ser movidos hacia la izquierda. En nuestra metodología, esto se hace identificando un nuevo modelo en cascada, modificando la dinámica mediante la inclusión de algún controlador clásico adecuado. Aunque el esquema de actuación en este caso es general, el mecanismo detallado no está suficientemente sistematizado aún.

A través de los ejemplos de aplicación presentados, puede concluirse que la metodología de diseño desarrollada trabaja razonablemente bien, incluso muy bien en comparación a otras técnicas convencionales. Sin embargo a pesar de los buenos resultados alcanzados, la aplicabilidad de la metodología debe aún de ser probada en una forma más profunda mediante su aplicación a ejemplos más sofisticados y a más casos. Aunque el grado de sistematización es alto, se requieren generalizar aún

[2]del inglés: "fuzzy inductive reasoning" (FIR).
[3]del inglés: "finite state machine" (FSM).

varios de los aspectos del diseño.

Otro objetivo perseguido por esta tesis fue el de evaluar y desarrollar la herramienta del FIR para caracterizar sistemas dinámicos. Como la metodología de FIR no estaba suficientemente desarrollada cuando se empezó la investigación, fue necesario aumentarla mediante la implementación de varios módulos nuevos los cuáles fueron descritos en el capítulo 4 de esta tesis. A juzgar por los resultados alcanzados, no cabe duda del enorme potencial que la metodología contiene. Los casos tratados por la herramienta de razonamiento inductivo difuso, además de los presentados aquí, permiten suponer que la metodología puede ser aplicada para diseñar controladores para sistemas de ingeniería de complejidad real.

Otro aspecto a destacar de la presente tesis es su aportación al desarrollo de una metodología funcional de simulación mixta cuantitativa y cualitativa cuyos elementos han sido descritos en el capítulo 5. El ejemplo presentado ahí ha sido la primera vez que dicha metodología ha sido aplicada. Dado el fuerte acoplamiento que existe entre las estructuras cualitativas del controlador difuso con las estructuras numéricas de su entorno (planta y señales de entrada), la habilidad de operación mixta es imprescindible para el desarrollo mediante simulación de controladores difusos del tipo FIR.

Los resultados obtenidos pueden resumirse de la manera siguiente:

- Diseño sistemático de controladores difusos multientrada/multisalida.
- Una nueva metodología para obtener la dinámica inversa de la planta.
- El desarrollo de un poderoso entorno para modelado y simulación mixta cuantitativa y cualitativa de sistemas dinámicos.
- Adquisición de conocimiento basado en patrones para el diseño de sistemas difusos.

8.3 Temas para la Investigación en el Futuro

En la medida que la investigación fue progresando, surgieron diversos aspectos, tanto de las herramientas metodológicas como de la aproximación del diseño en sí misma, que requieren ser investigados con

mayor profundidad. Los temas pueden ser agrupados en función de los tres objetivos planteados en el capítulo introductorio como:

- **Metodología de diseño.** Aunque los ejemplos de complejidad realista de problemas típicos de la ingeniería han demostrado la factibilidad de la metodología propuesta, aún no puede presentarse como una herramienta que pueda ser ofrecida a un ingeniero de control como una caja negra. Tenemos que automatizar una cantidad de decisiones que aún deben tomarse a mano mediante conocimiento de naturaleza heurística. Se espera que, en un par de años, será posible escribir un libro de control difuso que pueda usarse como un manual del diseño sistemático de controladores difusos mediante la metodología de FIR. Esta tesis aún está muy leja de poder servir como ese manual. Los principales puntos a desarrollar aún son:
  (a) Debe automatizarse y simplificarse la inclusión de bucles de control de seguimiento y ajuste.
  (b) Deben de estudiarse los criterios para decidir cuándo los bucles de control clásicos deben de incluirse fuera o dentro del modelo cualitativo.
  (c) Aunque debido al comportamiento diferencial que presentan, es difícil utilizar modelos cualitativos, deben de estudiarse los casos en los cuales la estructura de la planta sea no propia. También debe determinarse el límite de validez de la metodología en función de la estructura de la planta a ser controlada.
  (d) Una posible alternativa para la obtención del modelo de la dinámica inversa de la planta es utilizando directamente un modelo cualitativo identificado de manera directa pero utilizado en forma inversa.
  (e) Como se estudió en el capítulo 7, la arquitectura de los FIR-Controladores puede extenderse a un esquema adaptativo en el que los parámetros de las funciones de pertenencia se adapten a las condiciones en que el modelo cualitativo opera.
  (f) Un esquema con diferentes razonadores inductivos para cada FIR-Controlador puede ser otra extensión para dar tratamiento a sistemas de estructura variable.
- **Evaluación de la metodología FIR.** Respecto a la herramienta de razonamiento inductivo difuso se debe de:

(a) Automatizar la selección del intervalo de muestreo más apropiado.

(b) Sistematizar una estrategia de excitación de los sistemas que permita capturar el comportamiento dinámico del sistema no sólo desde el punto de vista frecuencial sino también que permita mantener una base de datos suficientemente rica de los diferentes estados posibles del sistema.

(c) Mejorar el algoritmo de búsqueda de máscaras (sub)óptimas mediante la inclusión de técnicas de búsqueda heurística que permiten utilizar el cálculo de la entropía de Shannon como función de costo para la optimización.

(d) Intentar la utilización de medidas de incertidumbre alternativas [Klir 88], [Yager 93] para la evaluación y búsqueda de máscaras (sub)óptimas.

(e) Experimentar otros algoritmos de selección de número de clases y de determinación de los valores límite entre ellas.

(f) La interfaz de simulación mixta con ACSL debe de ser mejorada para permitir un entorno más amigable.

- **Unión entre la IA y el Control.**

(a) Debe desarrollarse una técnica para permitir incluir conocimiento *a priori* que sea compatible con el manejo de la base de datos obtenida automáticamente.

(b) De acuerdo con el principio de Saridis [Sarid 89], debe formularse un esquema en el cual los FIR-Controladores puedan trabajar en colaboración con etapas de control de jerarquía superior. Tanto la utilización de los niveles epistemológicos de Klir [Klir 85] como los niveles jerárquicos de Saridis deben ser considerados para la realización de una generalización del conocimiento mayor.

(c) Se debe de estudiar con mayor profundidad la solución de la metodología de modelado y simulación mixtos para satisfacer tanto los criterios causales como los de la estabilidad numérica. Puede no resultar práctico simplemente aumentar más y más la profundidad de la máscara, por ejemplo en sistemas rígidos (sistemas cuyas constantes de tiempo más lenta y más rápida varían en órdenes de magnitud). El problema de resolución de frecuencia múltiple inherentes en este tipo de sistemas debe ser considerada.

(d) Una evaluación comparativa contra los resultados obtenidos por los esquemas conexionistas debe de ser realizada con

el propósito de investigar las ventajas (o desventajas) de la metodología FIR contra la de las Redes Neuronales.

Como una nota final, deseamos participar nuestra convicción de que la metodología de razonamiento inductivo difuso tiene un gran potencial que aún debe de ser explorado. Lamentablemente el grupo que lo desarrolla es aún muy pequeño. Respecto al caso particular del método expuesto a lo largo de esta tesis, sentimos que el trabajo no ha hecho más que comenzar.

Apéndice A

Paradigmas de Física Cualitativa

En este apéndice consideramos principalmente el conjunto de metodologías y formalismos desarrollados como una nueva rama de la inteligencia artificial bajo los nombres de: razonamiento cualitativo [Forbu 81], simulación cualitativa [Kuipe 82,84,86], razonamiento causal [deKle 82], teoría de procesos cualitativos [Forbu 84], física *naive* [Hayes 78,85a], física cualitativa [deKle 84], razonamiento con sentido común [McCar 58,90], [Carbo 85] y modelado de sistemas a partir de conocimiento de nivel profundo [Pan 84]. En el futuro nos referiremos a ellos genéricamente como métodos de razonamiento cualitativo. Existen varias compilaciones y revisiones de tales métodos. Algunos de ellos son 'Formal Theories of the Commonsense World' editado por Hobbs y Moore [Hobbs 85], 'Readings in Qualitative Reasoning about Physical Systems', editado por Weld & de Kleer [Weld 90], [Bonis 85] y los números especiales dedicados a Razonamiento Cualitativo del 'Artificial Intelligence' Vol. 24 [Bobro 84] y Vol. 51 [Bobro 91]. Una comparación con los métodos de simulación cuantitativa se reporta en [Celli 91a]. Cabe aclarar que nuestra revisión está limitada a sistemas físicos y no contempla otras formulaciones de modelado cualitativo hechas desde campos como la biología [Pucci 85], [Levin 74], la economía [Klee 81] o las ciencias sociales en general [Blalo 85].

La primera referencia al razonamiento cualitativo con que contamos se remonta a 1977 cuando de Kleer [deKle 77] desarrolla el sistema NEWTON en el que presenta su teoría de 'envisioning'. En NEWTON el comportamiento cualitativo de un mecanismo es representado mediante el grafo de las transiciones entre los diferentes estados cualitativos

posibles. Dos años después, Hayes publica su manifiesto de la física 'naive' [Hayes 79] en donde utiliza la lógica de primer orden para formalizar una gran parte del conocimiento que común y corrientemente se tiene del mundo físico y para axiomatizar la heurística (metainformación) con el objetivo de inferir los posibles comportamientos de un sistema a partir del conocimiento del mundo físico formalizado. La vida de cada objeto se define mediante fragmentos espacio–temporales llamados *historias*. A partir de entonces surgen una gran variedad de métodos cualitativos entre los que podemos destacar tres aproximaciones que pueden ser consideradas como las más representativas: i) la teoría de procesos cualitativos de Forbus, basada en el concepto de proceso; ii) la teoría de 'envisioning' desarrollada por de Kleer y Brown, basada en el modelado de *dispositivos* independientes interconectados; y iii) la teoría de modelado cualitativo de Kuipers, basada en ecuaciones diferenciales cualitativas vistas como *restricciones*. En el siguiente se presenta una breve descripción de los principales elementos de estas metodologías, resaltando los aspectos de *modelado*, *simulación* y *explicación*.

A.1 Razonamiento Cualitativo — Procesos

La teoría de Procesos Cualitativos (TPC), desarrollada por Forbus, parte de las ideas de Hayes de construir una versión cualitativa de la física clásica. La idea básica es que el mundo físico es una colección de objetos con ciertas propiedades e interrelaciones tales que todo cambio en las propiedades de los objetos o en sus interrelaciones es causado por un *proceso*. Es la aproximación que se considera más elaborada conceptualmente hablando, y es la más complicada de instrumentar computacionalmente. Las principales referencias son [Hayes 78,85] y [Forbu 81,84,85].

Modelado. En la teoría de procesos cualitativos un sistema queda modelado cuando se han definido los objetos, todas las posibles vistas de los objetos y los procesos. Los elementos de modelado son:

- Tiempo. El tiempo es modelado cualitativamente mediante *intervalos*, los cuales son definidos con tres funciones: *comienzo*, *duración* y *terminación*. Cuando se habla de *instantes* se habla de intervalos con duración cero.

- Historia. El concepto de historia es introducido en la teoría de Hayes y plenamente desarrollado por Forbus. Una historia es un registro de los estados adquiridos por un objeto en el transcurso del tiempo. Una historia está formada por intervalos (episodios) y por instantes (eventos). Una historia indica qué procesos (o vistas individuales) están activos en cada momento.
- Objetos. Los objetos son descritos mediante *parámetros* y los procesos actúan sobre los objetos cambiando esos parámetros. Los parámetros se representan mediante entidades formados por una *cantidad* y una *derivada*, cada una de las cuales se expresa mediante un *signo* $[-1, 0, +1]$ y una *magnitud*. Las magnitudes a su vez se representan mediante símbolos ordenados 'parcialmente'. Dos símbolos contiguos se denominan símbolos vecinos y en base a ellos se determina cuando comienza y termina un proceso. Los diferentes estados que adquieren los objetos en el tiempo son denominados *vistas individuales*.
- Proceso. Un proceso es especificado mediante cinco elementos:
  (a) *Individuos*. Lista de objetos afectados por el proceso.
  (b) *Precondiciones*. Condiciones externas (no son resultados de la dinámica) que deben de cumplirse para que el proceso se active. Por ejemplo: *Válvula 1: ABIERTA*.
  (c) *Condiciones cuantitativas*. Afirmaciones que deben verificarse para activar el proceso respecto a desigualdades entre los valores de los parámetros de los individuos que participan del proceso y/o acerca del estado del proceso. Por ejemplo: 'El nivel del contenedor ha alcanzado el valor máximo'.
  (d) *Relaciones*. Como se relacionan los parámetros de los individuos. Por ejemplo $Q_1 \propto_{Q+} Q_2$ nos dice que Q_1 se relaciona con Q_2 de manera monótonamente creciente.
  (e) *Influencias*. Influencia que tendrá el proceso en los valores de los parámetros de los individuos. La sintaxis es: $I \pm (p, n)$, en donde p es el parámetro influenciado por el proceso n. Por ejemplo, supongamos el proceso *flujo de fluidos* entre dos contenedores cuando una válvula se ha abierto. La influencia del proceso *flujo* en el contenedor que se vacía (C1) será negativa, denotado como $I - (cantidad(C1), A[flujo])$; la influencia del proceso *flujo* en el contenedor que se llena (C2) será positiva, denotado como $I + (cantidad(C2), A[flujo])$.
- Vista individual. Las primeras cuatro características de un proceso caracterizan una vista individual describiendo un objeto o el estado

de un objeto. Un proceso puede ser visto como el estado de un conjunto de objetos más un conjunto de influencias.

Simulación. El proceso de simulación, es decir la búsqueda de los comportamientos posibles, queda reflejado en la construcción de historias y puede descomponerse en cinco etapas:

(a) Instanciación de procesos posibles y vistas. En función de los procesos (biblioteca de procesos) y de los componentes (o sus vistas individuales) se encuentran las instancias de los procesos y las instancias de las vistas individuales.

(b) Evaluación de precondiciones y condiciones cuantitativas. Se determina qué procesos se activarán, es decir, se determina qué está pasando.

(c) Resolución de influencias. El proceso efectúa los cambios en los parámetros de los componentes en función del valor del signo de sus derivadas (incrementa, decrementa o permanece estable).

(d) Análisis de valores límite. Se determinan los posibles comportamientos válidos del sistema. Una vez establecidos los cambios en los valores de los parámetros, se valora la consistencia de dichos cambios revisando que los demás cambios sean consistentes y que el principio de continuidad se verifique.

(e) Ambigüedad. Forbus [Forbu 84] explica que una situación ambigua puede presentarse si: a) han ocurrido varios procesos simultáneamente; b) el espacio cuantitativo no está ordenado y puede ocurrir que exista más de un vecino; y c) un proceso puede influir más de un parámetro.

Explicación. La capacidad de explicación de la TPC está basado en el concepto intuitivo de la *hipótesis de direccionalidad causal* que establece que 'las influencias directas de los procesos se propagan linealmente por la proporcionalidad que guardan las variables cualitativas entre sí, expresadas por medio de sus dependencias funcionales'. Razonar causalmente es establecer el orden en que los eventos se suceden unos después de otros. Mientras que con un sistema de ecuaciones podríamos deducir, utilizando la fórmula de la segunda ley de Newton, que la masa ha disminuido o la fuerza ha aumentado porque la aceleración aumenta, en la TPC se establece que la única causalidad posible es justamente la relación inversa, es decir, es un cambio en la aceleración el resultado de cambiar la fuerza o la masa de un sistema.

A.2 Razonamiento Cualitativo — Dispositivos

A diferencia de la TPC que basa la descripción de la estructura de un sistema en el concepto de *proceso*, esta aproximación captura la estructura de un sistema centrándose en el modelado de sus componentes y las relaciones que guardan unos con respecto a otros. El comportamiento del sistema es obtenido mediante una composición o agregación de los comportamientos individuales de cada componente, los cuales se almacenan como comportamientos genéricos (independientes de cualquier situación particular) junto con la descripción de los parámetros de cada componente. El razonamiento cualitativo basado en componentes fue iniciado por de Kleer [deKle 77,80], desarrollándose de manera independiente varias aproximaciones diferentes. Las principales son: la aproximación de de Kleer y Brown [deKle 82,84,86], la propuesta por Davis [Davis 82,84], la de Genesereth [Genes 84], y la postulada por Iwasaki y Simon [Iwasa 86,91].

Modelado. Los elementos básicos de modelado son las *confluencias*. Las confluencias son ecuaciones diferenciales cualitativas que normalmente son encontradas en base a la ecuación diferencial de la física real que rige el comportamiento del componente. Hay tres clases principales de entidades: *materiales*: energía, sustancias, etc.; *componentes*: que transforman los materiales al actuar sobre sus propiedades; y *conexiones*: que transportan los materiales entre los componentes. El modelo de un sistema puede verse como un grafo de nodos (componentes) unidos por conexiones. Los elementos más importantes de modelado son:

- Tiempo. El tiempo es modelado cualitativamente mediante *intervalos* e *instantes*. Dos estados consecutivos pueden corresponder a instantes contiguos de tiempo sin necesidad de alternar con intervalos.
- Confluencias. Las confluencias o ecuaciones diferenciales cualitativas son descritas en base a variables cualitativas las cuales corresponden a las variables físicas del sistema o a sus derivadas, utilizándose incluso derivadas de orden superior a dos. Las variables están defininidas sobre intervalos completamente ordenados en el espacio cuantitativo que cubren el dominio de los números reales. El espacio cuantitativo típico está formado por los intervalos: $(-\infty, 0)$, $[0, 0]$ y $(0, +\infty)$, con lo que una variable cualitativa puede adquirir los valores $[-, 0, +]$. Una confluencia se expresa como $\sum_{i=1}^{T} t_i =$

constante, donde cada término t_i puede ser una variable, la negación de una variable o el producto de una constante por una variable. Un conjunto de valores satisface una confluencia si: a) todos sus términos están instanciados (tienen algún valor) y b) utilizando el cálculo cualitativo se mantiene la igualdad. Las confluencias siempre son lineales no estando permitido el producto entre variables cualitativas.

- Estado. Se define el concepto de estado para describir las diferentes condiciones o comportamientos de un dispositivo bajo las cuales determinadas confluencias son aplicables. Los diferentes estados se definen por las confluencias que se aplican y por sus especificaciones. Por ejemplo, los estados de una válvula simple pueden describirse cómo:

  Estado: *ABIERTA*

  precondición: $\{\acute{a}rea = \acute{a}rea_{max}\}$

  confluencias: $\{presión = 0, \delta presión = 0\}$

  Estado: *OPERANDO*

  precondición: $\{0 \leq \acute{a}rea \leq \acute{a}rea_{max}\}$

  confluencias: $\{presión = flujo, \delta presión + \delta área - \delta flujo = 0\}$

  Estado: *CERRADA*

  precondición: $\{\acute{a}rea = 0\}$

  confluencias: $\{flujo = 0, \delta flujo = 0\}$

- Conexiones. Cada conexión es definida por un conjunto de nodos a los que se les asocia un parámetro de flujo (por ejemplo m_i) y por alguno de dos tipos especiales de confluencias (y sus derivadas), la primera llamada *confluencia de continuidad* para representar conceptos como los de la conservación de materia y energía ($m_1 + m_2 + \cdots + m_n = 0$); y la segunda la *confluencia de compatibilidad* para representar las bifurcaciones entre dos conexiones.

Simulación. De Kleer utiliza el término *'envisioning'* para denotar su método de simulación cualitativa. El 'envisioning' es el proceso de *predecir* los posibles comportamientos del sistema, inferidos a partir del estado actual de cada uno de los componentes y del modelo cualitativo (descripción de la estructura) del sistema. Debemos distinguir dos partes del proceso de simulación: la predicción del comportamiento *dentro* de un estado y la predicción del comportamiento *entre* dos estados.

- Comportamiento dentro de un estado. El estado de un sistema en un determinado momento es la suma de los estados de cada uno de los dispositivos que lo componen. La primera tarea a realizar en esta etapa es identificar, para cada dispositivo, los posibles estados a los que se puede arribar partiendo del estado actual; una vez encontrados los estados posibles, deben de ser resueltas las confluencias mediante satisfacción de las restricciones (cálculo cualitativo) con el propósito de encontrar los valores cualitativos de las derivadas de las confluencias. En la terminología de de Kleer esto es definido como un *episodio*. Desafortunadamente la propagación simple de las restricciones para la resolución de confluencias no es suficiente para determinar los valores de las variables cualitativas, formándose algo similar a un sistema de ecuaciones en los que unas variables están en función de las demás. Para resolver esto, un proceso de prueba y verificación es utilizado para encontrar las soluciones a todas las confluencias. A cada predicción resultante para un mismo conjunto de confluencias se le conoce como una *interpretación*.
- Comportamiento entre estados. Cuando los comportamientos de cada componente para cada estado han sido determinados se aplican leyes de continuidad, derivadas del cálculo cualitativo, para encontrar el conjunto de transiciones permitidas y se determina con ello el conjunto de los posibles nuevos estados. Las leyes de continuidad son del tipo: no se puede ir del estado A al estado C sin pasar por el estado B. El resultado del proceso de 'envisioning' es un *diagrama de estados* en donde el comportamiento del sistema es representado.

Explicación. Mientras que la capacidad de predicción de la metodología desarrollado por de Kleer se considera satisfactoria, la capacidad de explicación se ha calificado como deficiente. El sistema utiliza un esquema de deducción natural para explicar la causalidad mediante pruebas lógicas utilizando mecanismos poco intuitivos como la reducción al absurdo y la introducción de premisas. Aunque se pueda demostrar por que el sistema se comporta de una cierta manera, no se explica cómo lo hace ni las causas que provocaron ese comportamiento. Intentando resolver este problema, de Kleer y Brown introducen el concepto de *causalidad mítica*. La idea central de la causalidad mítica es redefinir el cambio del sistema como una interacción entre estados de equilibrio (el sistema siempre debe partir de un estado de equilibrio) y situaciones inestables. En un estado inestable las restricciones son propagadas

hasta que se encuentre un nuevo estado estable. Todas las acciones emprendidas entre un estado estable y otro ocurren en un *tiempo mítico*. La proposición de la causalidad mítica requirió la introducción de algunas heurísticas para resolver conflictos como el de bloqueo de la propagación de restricciones. Este tipo de soluciones (un tanto forzadas) provocaron muy fuertes críticas, particularmente de Iwasaki y Simon [Iwasa 86a,86b], los cuales propusieron un sistema alternativo mejor fundamentado introduciendo su teoría de orden causal entre las magnitudes que describen el sistema.

A.3 Razonamiento Cualitativo — Restricciones

Comparado con los dos métodos anteriores de razonamiento cualitativo, el método de Kuipers, basado en restricciones, es el más simple desde el punto de vista conceptual y es quizás el que ha sido instrumentado computacionalmente de mejor manera (más eficiente). Se le conoce mejor como *simulación cualitativa* posiblemente porque le da un mayor énfasis a la simulación que al modelado. Es también el más referenciado en los campos de ingeniería. Su descripción detallada puede ser encontrada en [Kuipe 82,84,86,89,91].

Modelado. Podría decirse que Kuipers no da demasiada atención al problema de modelado cualitativo limitándose a tomar el conjunto de ecuaciones diferenciales que describen el proceso físico como fuente de conocimiento de la estructura del sistema. De esta manera aprovecha las poderosas herramientas desarrollada por físicos, matemáticos e ingenieros. Para Kuipers el mundo físico se puede modelar completamente mediante un conjunto homogéneo de restricciones, que pueden ser vistas básicamente como la interpretación cualitativa de las ecuaciones diferenciales ordinarias encontradas analíticamente. El proceso de modelado consiste en: a) establecer las variables cualitativas conocidas como *parámetros* y sus valores límite; b) definir las relaciones (restricciones) entre ellos y c) determinar los intervalos de validez de las restricciones. La terminología utilizada se presenta a continuación.

- Tiempo. De manera análoga a los dos sistemas descritos antes, el tiempo es modelado cualitativamente, en este caso mediante puntos temporales e *intervalos* descritos mediante dos puntos temporales adyacentes. El tiempo *transcurre* mediante la alternancia sucesiva

entre puntos temporales e intervalos (abiertos entre los puntos temporales que los definen).

- Variables Cualitativas. Los valores de las variables cualitativas son representados mediante un conjunto completamente ordenado y definido de *valores límite*. El *estado* cualitativo de una variable se define para un punto temporal o para un intervalo mediante el par [Qvalor, Qdirección]. *Qvalor* es representado por un valor límite único L_i si se trata de un punto temporal, y por dos valores límite contiguos L_i, L_{i+1} cuando se trata de un intervalo. *Qdirección* describe la dirección del cambio de la variable (derivada cualitativa) y puede adquirir los valores {inc, std, dec} indicando que Qvalor está aumentando, estable o disminuyendo respectivamente.
- Restricciones. Las restricciones son obtenidas en forma directa de las ecuaciones diferenciales analíticas. Una restricción establece la relación funcional cualitativa que guardan dos parámetros o variables cualitativas en el tiempo. Se han definido tres tipos de restricciones:

  Aritméticas: ADD(x,y,z), MULT(x,y,z) y MINUS(x,y)
  Funcionales: $M^+(x, y)$ y $M^-(x, y)$
  Derivativas: DERIV(x,y)

  Algunos ejemplos de relaciones físicas familiares son:

  DERIV(velocidad,aceleración),
  MULT(masa,aceleración,fuerza) y
  ADD(total,salida,entrada).
- Regiones de operación. Para expresar el rango de aplicabilidad de un conjunto de restricciones se utiliza el concepto de *región de operación*. Una región de operación es definida mediante un conjunto de *rangos legales* cada uno de los cuales se corresponde con una de las variables cualitativas implicadas en el conjunto de restricciones y es establecido mediante dos puntos terminales expresados con valores límite. Los puntos terminales se asocian con transiciones a otras regiones de operación en donde un nuevo conjunto de restricciones es aplicable.

Simulación. El proceso de simulación cualitativa es descrito mediante *transiciones* entre estados del sistema. Existen dos tipos de transiciones: las *I–transiciones* y las *P–transiciones*. Las P–transiciones definen los cambios que se suceden cuando se va de un punto temporal a un intervalo y las I–transiciones definen los cambios cuando se va de un intervalo

a un punto temporal. La transición de estados es originada cuando el conjunto de reglas de transición, correspondiente al momento, es aplicada a cada una de las variables cualitativas según el Qvalor y Qdirección que mantienen. Al poderse aplicar más de una regla a cada parámetro, pueden producirse estados alternativos como posibles transiciones. Las tablas de reglas de transiciones son la versión cualitativa del cálculo numérico utilizando las versiones cualitativas de los teoremas del valor medio y valor intermedio.

La instrumentación computacional del proceso de simulación es conocida como el algoritmo QSIM. El algoritmo de simulación de QSIM se describe brevemente a continuación.

Se coloca el estado inicial en la lista de estados ACTIVOS y se repite el ciclo hasta que no haya ningún estado activo:

(a) Seleccionar el siguiente estado cualitativo de la lista de ACTIVOS.

(b) Para cada variable cualitativa determinar el conjunto de transiciones posibles desde el estado cualitativo actual, usando la tabla de transiciones correspondientes (punto temporal ó intervalo).

(c) Para cada restricción obtener los conjuntos de t-uplas de transición correspondientes a cada una de las combinaciones de los posibles estados de sus argumentos.

(d) Filtrar las t-uplas de transición mediante criterios de consistencia eliminando las t-uplas inconsistentes.

(e) Generar todas las posibles interpretaciones globales (conjuntos consistentes de variables cualitativas) mediante las transiciones no eliminadas. Si no quedan transiciones, marcar el comportamiento como inconsistente. Crear un nuevo estado cualitativo para cada interpretación global y hacerlo sucesor del estado actual.

(f) Aplicar las reglas de filtrado global (reglas: no cambio, ciclo y divergencia) a los estados recién generados y colocar a los no eliminados en la lista de ACTIVOS.

Dos son las críticas más fuertes al método de modelado cualitativo basado en restricciones: la falta de capacidad para generar explicaciones y la incapacidad del algoritmo para eliminar los comportamientos no reales. Este último problema está intentándose resolver mediante el uso de derivadas de orden superior [Kuipe 91].

Explicación. El único tipo de explicación que puede ofrecerse con esta metodología es un árbol de los posibles estados (que puede llegar a ser

muy grande) pero en el cuál no hay información de cuáles estados son reales y cuáles son espúreos.

Apéndice B

Programas de la Aplicación Lineal

```
//  Nombre: lineal0.m
//
//  Esta rutina genera los datos de un sistema lineal
//       de ecuaciones diferenciales en MATLAB
//  ---------------------------------------------
//
//  Por:  Francisco Mugica
//
echo off;
global qualms repo;
qualms = 1;
repo = 0;
//
//  Se define el Sistema:
//
a = [ 0 1 0 ; 0 0 1 ; -2 -3 -4 ];
b = [ 0 ; 0 ; 1 ];
c = eye(size(a));
d = zeros(size(b));
//
//  Se simulan 900 segundos del Sistema (cuantitativo)
//                    en MATLAB:
//
t = 0:3:900;
rand('seed',1);
u = round(rand(301,1));          // Se genera el ruido binario
x0 = zeros(3,1);
[y,x] = lsim(a,b,c,d,u,t,x0);    // Simulacion
meas = [ u , y ];
save lineal0 t meas
```

```
//  Nombre: lineal1.m
//
//  Esta rutina convierte la informacion cuantitativa en
//  informacion cualitativa utilizando la funcion de
//  SAPS-II  Codificacion Difusa.
//  ------------------------------------------------
//  Por:  Francisco Mugica
//
echo off;
global qualms repo;
qualms = 1;
repo = 0;
//
//  Se aplica la codificacion difusa a las variables
//
load lineal0;
m = meas;
veps = 0.0001*ones(1,4);
for i=2:4,
  [mi,indx] = sort(meas(:,i));
  m(:,i) = mi;
end
LM = [ m(1,:)
       0.5*(m(100,:) + m(101,:))
       0.5*(m(200,:) + m(201,:))
       m(301,:)+veps ];
LM(3,1) = LM (3,1) - 1.0E-4;
raw1 = meas;
Memb1 = ones(size(meas));
side1 = ones(size(meas));
for j=1:301,
    if raw1(j,1) == 0
       raw1(j,1) = 1;
       side1(j,1) = 0;
       Memb1(j,1) = 1;
    else
       raw1(j,1) = 3;
       side1(j,1) = 0;
       Memb1(j,1) = 1;
    end;
end;
to = 1:3;
for i=2:4,
  from = [ LM(1:3,i) , LM(2:4,i) ]';
  [r,m,s] = recode(meas(:,i),'fuzzy ',from,to);
  raw1(:,i) = r;  Memb1(:,i) = m;  side1(:,i) = s;
end
save lineal1 meas raw1 Memb1 side1 LM
```

```
//
echo on;
//  Nombre: lineal2.m
//
//  Esta rutina encuentra las mascaras optimas para
//  las variables de salida utilizando la funcion de
//  SAPS-II --> FOPTMASK.
//  ------------------------------------------------
//
//  Por:  Francisco Mugica
//
echo off;
global qualms repo;
qualms = 1;
repo = 0;
load lineal1;
//
//  Encuentra las mascaras optimas usando solo 270 puntos
//
rraw = raw1(1:270,:);
MMemb = Memb1(1:270,:);
sside = side1(1:270,:);
//
//  Para la primera variable de salida
//
mcan = [-1 -1 -1 -1 ; ...    // Mascara candidata
        -1 -1 -1 -1 ; ...
        -1  1  0  0];
[mask1,hm1,hr1,q1,mhis1] = foptmask(rraw,MMemb,mcan,5);
mask1
[Q,indx] = sort(-1 * q1);
-Q(1)
m1a = mhis1(:,4*(indx(1)-1)+1:4*indx(1));
m1b = mhis1(:,4*(indx(2)-1)+1:4*indx(2));
m1c = mhis1(:,4*(indx(3)-1)+1:4*indx(3));
m1 = [m1a,m1b,m1c];
//
//  Para la segunda variable de salida
//
mcan = [-1 -1 -1 -1 ; ...    // Mascara candidata
        -1 -1 -1 -1 ; ...
        -1  0  1  0];
[mask2,hm2,hr2,q2,mhis2] = foptmask(rraw,MMemb,mcan,5);
mask2
[Q,indx]  = sort(-1 * q2);
-Q(1)
m2a = mhis2(:,4*(indx(1)-1)+1:4*indx(1));
m2b = mhis2(:,4*(indx(2)-1)+1:4*indx(2));
m2c = mhis2(:,4*(indx(3)-1)+1:4*indx(3));
```

```
m2 = [m2a,m2b,m2c];
//
//  Para la tercera variable de salida
//
mcan = [-1 -1 -1 -1 ; ...    // Mascara candidata
        -1 -1 -1 -1 ; ...
        -1  0  0  1];
[mask3,h13,hr3,q3,mhis3] = foptmask(rraw,MMemb,mcan,5);
mask3
[Q,indx]  = sort(-1 * q3);
-Q(1)
m3a = mhis3(:,4*(indx(1)-1)+1:4*indx(1));
m3b = mhis3(:,4*(indx(2)-1)+1:4*indx(2));
m3c = mhis3(:,4*(indx(3)-1)+1:4*indx(3));
m3 = [m3a,m3b,m3c];
save lineal2 m1a m1b m1c m2a m2b m2c m3a m3b m3c rraw MMemb sside
```

```
//
//  -------- MASCARAS  OPTIMAS  ---------  RESULTADOS -----
//
mask1 =    -1     0     0    -2
           -3     0     0     0
           -4     1     0     0


calidad  =  0.9610


mask2 =    -1     0     0    -2
           -3     0     0     0
           -4     0     1     0


calidad  =  0.9491


mask3 =  -1    -2     0     0
         -3     0     0     0
         -4     0     0     1

calidad  = 0.8945
```

```
//
echo on;
//  Nombre: lineal3.m
//
//  Esta rutina encuentra las matrices de entrada/salida
//  y de comportamiento para los modelos cualitativos
//  utilizando las funciones de SAPS-II:
//                                        --> FIOMODEL2
//                                        --> FBEHAVIOR2.
//  ------------------------------------------------------
//
//  Por:  Francisco Mugica
//
echo off;
global qualms repo;
qualms = 1;
repo = 0;
load lineal2;
//
//  Encuentra los modelos para la primera variable
//
[io,mio,sio] = fiomodel2(rraw,MMemb,sside,m1a);
[b1a,Mb1a,sb1a] = fbehavior2(io,mio,sio);

[io,mio,sio] = fiomodel2(rraw,MMemb,sside,m1b);
[b1b,Mb1b,sb1b] = fbehavior2(io,mio,sio);

[io,mio,sio] = fiomodel2(rraw,MMemb,sside,m1c);
[b1c,Mb1c,sb1c] = fbehavior2(io,mio,sio);

//
//  Encuentra los modelos para la segunda variable
//

[io,mio,sio] = fiomodel2(rraw,MMemb,sside,m2a);
[b2a,Mb2a,sb2a] = fbehavior2(io,mio,sio);

[io,mio,sio] = fiomodel2(rraw,MMemb,sside,m2b);
[b2b,Mb2b,sb2b] = fbehavior2(io,mio,sio);

[io,mio,sio] = fiomodel2(rraw,MMemb,sside,m2c);
[b2c,Mb2c,sb2c] = fbehavior2(io,mio,sio);

//
//  Encuentra los modelos para la tercera variable
//

[io,mio,sio] = fiomodel2(rraw,MMemb,sside,m3a);
[b3a,Mb3a,sb3a] = fbehavior2(io,mio,sio);
```

```
[io,mio,sio] = fiomodel2(rraw,MMemb,sside,m3b);
[b3b,Mb3b,sb3b] = fbehavior2(io,mio,sio);

[io,mio,sio] = fiomodel2(rraw,MMemb,sside,m3c);
[b3c,Mb3c,sb3c] = fbehavior2(io,mio,sio);

save lineal3 b1a Mb1a sb1a b1b Mb1b sb1b b1c Mb1c sb1c ...
             b2a Mb2a sb2a b2b Mb2b sb2b b2c Mb2c sb2c ...
             b3a Mb3a sb3a b3b Mb3b sb3b b3c Mb3c sb3c
```

```
//
echo on;
//  Nombre: lineal4.m
//
//  Esta rutina simula los modelos cualitativos durante
//  30 pasos obteniendo los comportamientos cualitativos
//  predichos y compara el error (cualitativo) de la
//  prediccion utilizando la funcion de SAPS-II:
//                                          --> FFORECAST2
//  ---------------------------------------------------
//
//  Por:  Francisco Mugica
//
echo off;
global qualms repo;
qualms = 1;
repo = 0;
load lineal1;
load lineal2;
load lineal3;
//
[filas,columnas] = size(LM);
nclas = (filas-1)*ones(1,columnas);
LM = [nclas;LM];
limit = 271;
fin = 301;
inpt = raw1(limit:fin,1);
minp = Memb1(limit:fin,1);
sinp = side1(limit:fin,1);
[depth,col] = size(m1a);
fmask = raw1(limit-depth+1:limit-1,:);
Mmask = Memb1(limit-depth+1:limit-1,:);
smask = side1(limit-depth+1:limit-1,:);
xf1 = [fmask;inpt,zeros(fin-limit+1,3)];
xM1 = [Mmask;minp,0.75*ones(fin-limit+1,3)];
xs1 = [smask;sinp,ones(fin-limit+1,3)];
def = 1;
[dx,cx] = size(b1a);
jbbm = b1a(1:dx)';
n = fin - limit + 1;
for i=1:n,
//
//  Predice los episodios para la primera variable
//
  i,
  f1 = xf1(i:i+depth-1,:);
  M1 = xM1(i:i+depth-1,:);
  s1 = xs1(i:i+depth-1,:);
  [f2,M2,s2] = fforecast2(f1,M1,s1,b1a,Mb1a,sb1a,def,m1a,LM,jbbm);
```

```
  if f2 == f1,
    [f2,M2,s2] = fforecast2(f1,M1,s1,b1b,Mb1b,sb1b,def,m1b,LM,jbbm);
    if f2 == f1,
      [f2,M2,s2] = fforecast2(f1,M1,s1,b1c,Mb1c,sb1c,def,m1c,LM,jbbm);
      if f2 == f1,
         f2(depth+i-1,2) = round(rand(1,1));
         M2(depth+i-1,2) = 0.51;
         s2(depth+i-1,2) = 0;
      end,
    end,
  end,
//
//  Predice los episodios para la segunda variable
//
  f1 = f2; M1 = M2; s1 = s2;
  [f2,M2,s2] = fforecast2(f1,M1,s1,b2a,Mb2a,sb2a,def,m2a,LM,jbbm);
  if f2 == f1,
    [f2,M2,s2] = fforecast2(f1,M1,s1,b2b,Mb2b,sb,def,m2b,LM,jbbm);
    if f2 == f1,
      [f2,M2,s2] = fforecast2(f1,M1,s1,b2c,Mb2c,sb2c,def,m2c,LM,jbbm);
      if f2 == f1,
         f2(depth+i-1,3) = round(rand(1,1));
         M2(depth+i-1,3) = 0.51;
         s2(depth+i-1,3) = 0;
      end,
    end,
  end,
//
//  Predice los episodios para la tercera variable
//
  f1 = f2; M1 = M2; s1 = s2;
  [f2,M2,s2] = fforecast2(f1,M1,s1,b3a,Mb3a,sb3a,def,m3a,LM,jbbm);
  if f2 == f1,
    [f2,M2,s2] = fforecast2(f1,M1,s1,b3b,Mb3b,sb3b,def,m3b,LM,jbbm);
    if f2 == f1,
      [f2,M2,s2] = fforecast2(f1,M1,s1,b3c,Mb3c,sb3c,def,m3c,LM,jbbm);
      if f2 == f1,
         f2(depth+i-1,4) = round(rand(1,1));
         M2(depth+i-1,4) = 0.51;
         s2(depth+i-1,4) = 0;
      end,
    end,
  end,
  xf1(i+depth-1,:) = f2(depth,:);
  xM1(i+depth-1,:) = M2(depth,:);
  xs1(i+depth-1,:) = s2(depth,:);
end
//
//  Compara los resultados cualitativos
```

```
//
simdat = raw1(271:300,:);
Msimdat = Memb1(271:300,:);
ssimdat = side1(271:300,:);
frcdat1 = xf1(3:32,:);
Mfrcdat1 = xM1(3:32,:);
sfrcdat1 = xs1(3:32,:);
error1 = simdat - frcdat1;
[ simdat , frcdat1 , error1 ]
save lineal4 frcdat1 Mfrcdat1 sfrcdat1 simdat Msimdat ssimdat
```

```
//
//  -------   PREDICCION CUALITATIVA   --------  RESULTADOS  -------------
//
//  |---- Simulados -----|  |---- Predichos ----| |- Error Cualitativo --|
//
//   x1    y1    y2    y3    x1    y1    y2    y3  errx1 erry1 erry2 erry3
     3     3     3     1     3     3     3     1     0     0     0     0
     1     2     1     2     1     2     1     2     0     0     0     0
     1     1     2     3     1     1     2     3     0     0     0     0
     1     1     2     2     1     1     2     2     0     0     0     0
     3     2     3     3     3     2     3     3     0     0     0     0
     3     3     3     1     3     3     3     1     0     0     0     0
     3     3     2     2     3     3     2     2     0     0     0     0
     1     2     1     1     1     2     1     1     0     0     0     0
     1     1     2     3     1     1     2     3     0     0     0     0
     1     1     2     2     1     1     2     2     0     0     0     0
     3     2     3     3     3     2     3     3     0     0     0     0
     3     3     3     1     3     3     3     1     0     0     0     0
     1     2     1     2     1     2     1     2     0     0     0     0
     1     1     2     3     1     1     2     3     0     0     0     0
     1     1     2     2     1     1     2     2     0     0     0     0
     3     2     3     3     3     2     3     3     0     0     0     0
     3     3     3     1     3     3     3     1     0     0     0     0
     3     3     2     2     3     3     2     2     0     0     0     0
     3     3     2     2     3     3     2     2     0     0     0     0
     1     2     1     1     1     2     1     1     0     0     0     0
     1     1     2     3     1     1     2     3     0     0     0     0
     1     1     2     2     1     1     2     2     0     0     0     0
     1     1     2     2     1     1     2     2     0     0     0     0
     3     2     3     3     3     2     3     3     0     0     0     0
     1     3     1     1     1     3     1     1     0     0     0     0
     3     2     3     3     3     2     3     3     0     0     0     0
     1     2     1     1     1     3     1     1     0    -1     0     0
     3     2     3     3     3     2     3     3     0     0     0     0
     3     3     3     1     3     3     3     1     0     0     0     0
     1     2     1     1     1     2     1     1     0     0     0     0
```

```
//
echo on;
//  Nombre: lineal5.m
//
//  Esta rutina regenera los episodios (comportamientos
//  cualitativos) y encuentra las trayectorias cuantitativas
//  utilizando la funcion de SAPS-II:  --> REGENERATE.
//  Finalmente realiza una grafica comparativa
//  de las senyales predichas contra las esperadas
//  y el error correspondiente.
//  ------------------------------------------------
//
//  Por:  Francisco Mugica
//
echo off;
global qualms repo;
qualms = 1;
repo = 0;
load lineal1;
load lineal4;
//
//
//  Regenera la primera variable cualitativa
//

from = 1:3
to = [ LM(1:3,2) , LM(2:4,2) ]';
rvar11 = regenerate(frcdat1(:,2),Mfrcdat1(:,2),sfrcdat1(:,2),from,to);

//
//  Regenera la segunda variable cualitativa
//
from = 1:3;
to = [ LM(1:3,3) , LM(2:4,3) ]';
rvar12 = regenerate(frcdat1(:,3),Mfrcdat1(:,3),sfrcdat1(:,3),from,to);

//
//  Regenera la tercera variable cualitativa
//
from = 1:3;
to = [ LM(1:3,4) , LM(2:4,4) ]';
rvar13 = regenerate(frcdat1(:,4),Mfrcdat1(:,4),sfrcdat1(:,4),from,to);
save lineal5  rvar11 rvar12  rvar13
linealg1
```

```
echo on;
//
//  Nombre: linealg1.m
//
//  Esta rutina grafica los resultados de la simulacion
//  cualitativa.
//  ---------------------------------------------------
//
//  Por:  Francisco Mugica
//
echo off;
global qualms repo;
qualms = 1;
repo = 0;
load lineal0;
load lineal1;
load lineal5
//
t = t(1:30);
meas1 = meas(271:300,2);
meas2 = meas(271:300,3);
meas3 = meas(271:300,4);
//
//  Compara resultados variable 1, graficando
//  juntos los valores simulados y predichos.
//
xx1 = LM(1,2)*ones(30)';
xx2 = LM(2,2)*ones(30)';
xx3 = LM(3,2)*ones(30)';
xx4 = LM(4,2)*ones(30)';
echo off
plot(t,[meas1,rvar11,xx1,xx2,xx3,xx4]);
title('Sistema Lineal - Simulado vs. Predicho');
ylabel('Y1');
xlabel('Time');
pause;
//meta grafy1
//
//  Compara resultados variable 2, graficando
//  juntos los valores simulados y predichos.
//
xx1 = LM(1,3)*ones(30)';
xx2 = LM(2,3)*ones(30)';
xx3 = LM(3,3)*ones(30)';
xx4 = LM(4,3)*ones(30)';
plot(t,[meas2,rvar12,xx1,xx2,xx3,xx4]);
title('Sistema Lineal - Simulado vs. Predicho');
ylabel('Y2');
xlabel('Time');
```

```
pause;
//meta grafy2
//
//  Compara resultados variable 3, graficando
//  juntos los valores simulados y predichos.
//
xx1 = LM(1,4)*ones(30)' ;
xx2 = LM(2,4)*ones(30)' ;
xx3 = LM(3,4)*ones(30)' ;
xx4 = LM(4,4)*ones(30)' ;
plot(t,[meas3,rvar13,xx1,xx2,xx3,xx4]);
title('Sistema Lineal - Simulado vs. Predicho');
ylabel('Y3');
xlabel('Time');
pause;
err1 = rvar11-meas1;
err2 = rvar12-meas2;
err3 = rvar13-meas3;
plot(t,[err1,err2,err3]);
title('Sistema Lineal - Simulado vs. Predicho');
ylabel('ERROR');
xlabel('Time');
title('Sistema Lineal - ERROR - Simulado vs. Predicho');
//meta grafy3
pause
plot(t,meas1,'-',t,rvar11,'-',t,err1*100,':');
title('Sistema Lineal - Simulado vs. Predicho y Error*100');
pause
plot(t,meas2,'-',t,rvar12,'-',t,err2*100,':');
title('Sistema Lineal - Simulado vs. Predicho y Error*100');
pause
title('Sistema Lineal - Simulado vs. Predicho y Error*100');
plot(t,meas3,'-',t,rvar13,'-',t,err3*100,':');
```

Apéndice C

Código relacionado al Control del Barco

C.1 Obtención del Modelo en Cascada

C.1.1 Parte 1

```
 - outfile tesis2.rep
 - set LogModel on
 - set Statistics on
 -
 - enter model
enter model
 = @tesis.dym
@tesis.dym
model type reference

  terminal psim, psir
  local psimd, psim2d

    psimd = der(psim)
    psim2d = der(psimd)

    psim2d + 0.1*psimd + 0.0025*psim = 0.0025*psir

end

model type ship

  terminal psi, delta, u
  local psid, psi2d, psi3d, H, K, tau1, tau2, tau3
```

```
  local tau1inv, tau2inv, tau12inv, lu

  parameter K0 = -3.86, tau10 = 5.66, tau20 = 0.38, tau30 = 0.89
  parameter l = 161.0, a = 1.0, b = 1.0

    psid = der(psi)
    psi2d = der(psid)
    psi3d = der(psi2d)

    tau1inv = 1.0/tau1
    tau2inv = 1.0/tau2
    tau12inv = tau1inv*tau2inv
    lu = l/u

    K = K0/lu
    tau1 = tau10*lu
    tau2 = tau20*lu
    tau3 = tau30*lu

    H = ( a*psid*psid + b )*psid

    psi3d + ( tau1inv + tau2inv )*psi2d + tau12inv*H  ->
      = K*tau12inv*delta

end

model system

  submodel reference
  submodel ship

  local u

  input psir
  output delta

    u = 5.0

    reference.psir = psir
    ship.psi = reference.psim
    ship.u = u
    delta = ship.delta

 - partition
--- Additional trivial constraint equation relating known variables:
  system.    ship.psi = reference.psim

--- The number of non-trivial equations is 12.
--- The number of unknown variables is 13.
```

```
--- Unassigned variables:
system.delta

 - output solved equations

SORTED AND SOLVED EQUATIONS
  reference.
             derpsimd = 0.0025*system.psir - (0.1*psimd + 0.0025*psim)
  ship.      H = (a*psid*psid + b)*psid
  system.    u = 5.0
  ship.      lu = 1/system.u
             K = K0/lu
             tau1 = tau10*lu
             tau2 = tau20*lu
             tau3 = tau30*lu
             tau1inv = 1/tau1
             tau2inv = 1/tau2
             tau12inv = tau1inv*tau2inv
END OF SORTED AND SOLVED EQUATIONS

ELIMINATED STATE DERIVATIVES AND OUTPUTS
             reference.derpsim = reference.psimd
             ship.derpsi = ship.psid
             ship.derpsid = ship.psi2d

STATISTICS

  Number of variables:                            100
  Number of unknown variables:                     13
  Number of equations:                             58
  Number of non-trivial equations:                 12
  Number of systems of equations:                   0

  Number of sub-expressions:                        0
  Number of multiplications and divisions:         14
  Number of additions and subtractions:             3
  Number of function evaluations:                   0
  Number of other operations                        0
```

C.1.2 Parte 2

```
 - set LogDeriv on
 - differentiate
Equation needs to be differentiated:
  system.    ship.psi = reference.psim
Derivative:
             ship.derpsi = reference.derpsim
```

```
Checking if differentiated equations need to be differentiated.
Equation needs to be differentiated:
  reference.
              psimd = derpsim
Derivative:
              derpsimd = derderpsim

Equation needs to be differentiated:
  ship.       psid = derpsi
Derivative:
              derpsid = derderpsi

Equation needs to be differentiated:
  system.     ship.derpsi = reference.derpsim
Derivative:
              ship.derderpsi = reference.derderpsim

Checking if differentiated equations need to be differentiated.
Equation needs to be differentiated:
              reference.psir = psir
Derivative:
              reference.derpsir = derpsir

Equation needs to be differentiated:
  reference.
              psim2d + 0.1*psimd + 0.0025*psim = 0.0025*psir
Derivative:
              derpsim2d + 0.1*derpsimd + 0.0025*derpsim = 0.0025*derpsir

Equation needs to be differentiated:
              psim2d = derpsimd
Derivative:
              derpsim2d = derderpsimd

Equation needs to be differentiated:
  ship.       psi2d = derpsid
Derivative:
              derpsi2d = derderpsid

Equation needs to be differentiated:
  reference.
              derpsimd = derderpsim
Derivative:
              derderpsimd = derderderpsim

Equation needs to be differentiated:
  ship.       derpsid = derderpsi
Derivative:
```

```
             derderpsid = derderderpsi

Equation needs to be differentiated:
  system.    ship.derderpsi = reference.derderpsim
Derivative:
             ship.derderderpsi = reference.derderderpsim

Checking if differentiated equations need to be differentiated.
```

C.1.3 Parte 3

```
 - partition
--- The number of non-trivial equations is 13.
--- The number of unknown variables is 15.

--- Unassigned variables:
reference.psim
system.delta

 - output solved equations

SORTED AND SOLVED EQUATIONS
  system.    u = 5.0
  ship.      lu = 1/system.u
             tau1 = tau10*lu
             tau1inv = 1/tau1
             tau2 = tau20*lu
             tau2inv = 1/tau2
             tau12inv = tau1inv*tau2inv
             K = K0/lu
             tau3 = tau30*lu
END OF SORTED AND SOLVED EQUATIONS

STATISTICS

  Number of variables:                          109
  Number of unknown variables:                   15
  Number of equations:                           69
  Number of non-trivial equations:               13
  Number of systems of equations:                 0

  Number of sub-expressions:                      0
  Number of multiplications and divisions:        8
  Number of additions and subtractions:           0
  Number of function evaluations:                 0
  Number of other operations                      0
```

C.1.4 Parte 4

```
 - variable state ship.psi ship.psid
 - partition
 - output solved equations

SORTED AND SOLVED EQUATIONS
  ship.      H = (a*psid*psid + b)*psid
  system.    u = 5.0
  ship.      lu = 1/system.u
             K = K0/lu
             tau1 = tau10*lu
             tau2 = tau20*lu
             tau3 = tau30*lu
             tau1inv = 1/tau1
             tau2inv = 1/tau2
             tau12inv = tau1inv*tau2inv
  reference.
             ship.derpsid = 0.0025*system.psir - (0.1*ship.psid + 0.0025*
               ship.psi)
             derderderpsim = 0.0025*system.derpsir - (0.1*ship.derpsid +
               0.0025*ship.psid)
  ship.      system.delta = (reference.derderderpsim + (tau1inv + tau2inv)*
               derpsid + tau12inv*H)/(K*tau12inv)
END OF SORTED AND SOLVED EQUATIONS

ELIMINATED STATE DERIVATIVES AND OUTPUTS
             ship.derpsi = ship.psid

STATISTICS

  Number of variables:                          109
  Number of unknown variables:                   13
  Number of equations:                           69
  Number of non-trivial equations:               13
  Number of systems of equations:                 0

  Number of sub-expressions:                      0
  Number of multiplications and divisions:       21
  Number of additions and subtractions:           8
  Number of function evaluations:                 0
  Number of other operations                      0
```

C.1.5 Parte 5

```
 - language acsl
 - outfile ship.csl

!  - output model

! ACSL model generated by Dymola.

PROGRAM system

! --- File not found: system.dec
  VARIABLE Time, StartTime=0.0
  CONSTANT StopTime=1.0


  INITIAL

  CONSTANT K0=-3.86, tau10=5.66
  CONSTANT tau20=0.38, tau30=0.89
  CONSTANT l=161.0, a=1.0
  CONSTANT b=1.0


  END ! of INITIAL

  DYNAMIC

    DERIVATIVE der

      PROCEDURAL

! SORTED AND SOLVED EQUATIONS
!   ship.
      H = (a*psid*psid + b)*psid
!   system.
      u = 5.0
!   ship.
      lu = l/u
      K = K0/lu
      tau1 = tau10*lu
      tau2 = tau20*lu
      tau3 = tau30*lu
      tau1inv = 1.0/tau1
      tau2inv = 1.0/tau2
      tau12inv = tau1inv*tau2inv
!   reference.
      derpsid = 0.0025*psir - (0.1*psid + 0.0025*psi)
      derderderpsim = 0.0025*derpsir - (0.1*derpsid + 0.0025*psid)
```

```
!   ship.
      delta = (derderderpsim + (tau1inv + tau2inv)*derpsid +   &
        tau12inv*H)/(K*tau12inv)
! END OF SORTED AND SOLVED EQUATIONS

! ELIMINATED STATE DERIVATIVES AND OUTPUTS
      derpsi = psid

! STATISTICS
!
!   Number of variables:                          109
!   Number of unknown variables:                   13
!   Number of equations:                           69
!   Number of non-trivial equations:               13
!   Number of systems of equations:                 0
!
!   Number of sub-expressions:                      0
!   Number of multiplications and divisions:       21
!   Number of additions and subtractions:           8
!   Number of function evaluations:                 0
!   Number of other operations                      0
!
!

 EXIT .. CONTINUE
      END ! of PROCEDURAL

      CONSTANT initpsi=0
      psi = INTEG(derpsi, initpsi)
      CONSTANT initpsid=0
      psid = INTEG(derpsid, initpsid)

    END ! of DERIVATIVE

    TERMT (Time .GE. StopTime)

  END ! of DYNAMIC

END ! of PROGRAM


! --- File not found: system.app
!  - output variables
!
! reference.psim                              differentiated  = ship.psi
! reference.psir                              differentiated  = psir
! reference.psimd                             differentiated  = ship.psid
! reference.psim2d                            differentiated  = ship.derpsid
! reference.derpsim                           differentiated  = ship.psid
```

```
! reference.derpsimd                              differentiated  = ship.derpsid
! ship.psi              psi             known differentiated
! ship.delta                                     terminal        = delta
! ship.u                                         terminal        = u
! ship.psid             psid            known differentiated
! ship.psi2d                                     differentiated  = ship.derpsid

! ship.psi3d                                     local           = reference.derder
! ship.H                H                       local
! ship.K                K                       local
! ship.tau1             tau1                    local
! ship.tau2             tau2                    local
! ship.tau3             tau3                    local
! ship.tau1inv          tau1inv                 local
! ship.tau2inv          tau2inv                 local
! ship.tau12inv         tau12inv                local
! ship.lu               lu                      local
! ship.K0               K0              known parameter       = -3.860000
! ship.tau10            tau10           known parameter       = 5.660000
! ship.tau20            tau20           known parameter       = 0.380000
! ship.tau30            tau30           known parameter       = 0.890000
! ship.l                l               known parameter       = 161.000000
! ship.a                a               known parameter       = 1.000000
! ship.b                b               known parameter       = 1.000000
! ship.derpsi           derpsi                  differentiated  = ship.psid
! ship.derpsid          derpsid                 differentiated
! ship.derpsi2d                                  derivative      = reference.derder
! u                     u                       local
! psir                  psir            known input
! delta                 delta                   output
! Time                  Time            known independent
! reference.derderpsim                           differentiated  = ship.derpsid
! ship.derderpsi                                 differentiated  = ship.derpsid
! derpsir               derpsir         known input
! reference.derpsir                              derivative      = derpsir
! reference.derpsim2d                            derivative      = reference.derder
! reference.derderpsimd                          derivative      = reference.derder
! ship.derderpsid                                derivative      = reference.derder
! reference.derderderpsimderderderpsim           derivative
! ship.derderderpsi                              derivative      = reference.derder
!
!  -
!  - {stop}
```

Apéndice D

Bibliografía

D.1 Publicaciones

[**Celli 92a**] Cellier, F.E., Nebot, A., Mugica, F., and De Albornoz, A. (1992), "Combined Qualitative/Quantitative Simulation Models of Continuous–Time Processes Using Fuzzy Inductive Reasoning Techniques," in: *Proceedings SICICA'92, IFAC Symposium on Inteligent Components and Instruments for Control Applications*, Málaga, Spain, May, pp. 589–593.

[**Celli 92b**] Cellier, F.E. and Mugica, F. (1992) "Systematic Design of Fuzzy Controllers Using Inductive Reasoning", in: *Proceedings of the IEEE International Symposium on Intelligent Control*, Glasgow, Scotland, U.K. August, pp. 198–203.

[**Celli 94a**] Cellier, F.E., Nebot, A., Mugica, F. and De Albornoz, A. (1994), "Combined Qualitative/Quantitative Simulation Models of Continuous–Time Processes Using Fuzzy Inductive Reasoning Techniques." Article accepted for publication in: *International Journal of General Systems* (to be published during 1994).

[**Celli 94b**] Cellier, F.E. and Mugica, F. (1994), "Inductive Reasoning supports the Systematic Design Fuzzy Controllers ", in: *Journal of Intelligent and Fuzzy Systems* (to be published during 1994).

[**Mugic 93**] Mugica, F. and Cellier, F. E. (1993), "A New Fuzzy Inferencing Method for Inductive Reasoning," in: *Proceedings of the Sixth International Symposium on Artificial Intelligence, Intelligent Systems in Industry and Business*, Monterrey, México, September, pp. 372–379.

[**Mugic 94**] Mugica, F. and Cellier, F.E. (1994), "Automated Synthesis of a Fuzzy Controller for Cargo Ship Steering by Means of Qualitative Simulation," in: *Proceedings ESM'94, European Simulation MultiConference*, Barcelona, Spain, June pp. 523–528.

D.2 Referencias

[**Albus 75**] Albus J.S. (1975), "A New Approach To Manipulator Control: The Cerebellar Model Articulation Controller (CMAC)", in: *Transactions of the ASME*, September, pp. 220–227.

[**Aliev 92**] Aliev, R.A., Aliev, F.T. and Babaev, M.D., (1992), "The Synthesis of a Fuzzy Coordinate-Parametric Automatic Control System for an Oil-Refinery Unit," in: *Fuzzy Sets and Systems*, **47**, pp. 157–162, 1992.

[**Amero 75**] Amerogen, J.V., and Cate, A., (1975), "Model Reference Adaptative Autopilots for Ships," in: *Automatica*, **11**, pp. 441–449.

[**Ander 89**] Anderson, C.W. (1989), "Learning to Control an Inverted Pendulum Using Neural Networks" in: *IEEE Control Systems Magazine*, April.

[**Aoki 90**] Aoki, S., Kawachi, S., and Sugeno, M. (1990), "Application of Fuzzy Control Logic for Dead-time Processes in a Glass Melting Furnace", in: *Fuzzy Sets and Systems*, **38**, pp. 251–265.

[**Arsen 89**] Årsén (1989), K.E., "An Architecture for Expert System Based Feedback Control", in: *Automatica*, **25**(6), pp. 813–827.

[**Astrm 76**] Åström, K.J., and Kallström, C.G. (1976), "Identification of Ship Steering Dynamics,"in: *Automatica*, **12**(1), pp. 9–22.

[**Astrm 86**] Åström, K.J., Anton, J.J., and Årzén, K.E. (1986), "Expert Control," in: *Automatica*, **22**, pp. 277–286.

[**Astrm 89**] Åström, K.J., and Wittenmark, B., (1989), *Adaptative Control*, Addison-Wesley Publishing Company.

[**Bech 69**] Bech, M., and Smitt, L., (1969), *Analogue Simulation of Ship Maneuvers*, Technical Report, Hydro– og Aerodynamisk Laboratorium, Danske Tekniske Høyskole, Lyngby, Denmark.

[**Beren 92a**] Berenji, H.R. (1992), "An Architecture for Design Fuzzy Controllers Using Neural Networks", in: *International Journal of Approximate Reasoning*, **6**(2), pp. 267–292.

[**Beren 92b**] Berenji, H.R., and Khedkar, P. (1992), "Learning and Tuning Fuzzy Logic Controllers through for reinforcements", in: *IEEE Transactions on Neural Networks*, **3**(5).

[**Blalo 85**] Blalock, H.M. (1985), "Causal Model in Panel and Experimental Designs". Adline Publishing Co.

[**Bobro 84**] Bobrow, D.G., Ed. (1984), "Qualitative Reasoning about Physical Systems", in: Special Volume *Artificial Intelligence*, **24**.

[**Bobro 91**] Bobrow, D.G. and de Kleer, J. eds., "Special Volume on Qualitative Reasoning about Physical Systems I", *Artificial Intelligence*, **51**.

[**Bonni 85**] Bonnisone, P.P., and Valavanis, K.P. (1985), "A comparative study of different approaches to Qualitative Physics Theories," in: *CH2215–2/85/0000/0236$01.00 1985 IEEE*.

[**Boswe 90**] Boswell R.A. (1990), "Manual for NewID ver. 4.1", Tech. Rep. TI/P2154/RAB/4/2.3, The Turing Institute.

[**Bousl 92**] Bouslama, F. (1992), "Fuzzy Control and Their Natural Control Laws," in: *Fuzzy Sets and Systems*, **48**, pp. 65–86.

[**Bover 91**] Boverie, S., Demaya, B., and Titli, A. (1991), "Fuzzy Logic Control Compared With Other Automatic Control Approaches", in: *Proceedings of the 30th IEEE Conference on Decision Control*, pp. 1212–1216.

[**Bover 92**] Boverie, S., Demaya, B., Ketata, R., and Titli A., (1992), "Performance Evaluation of Fuzzy Controller," in: *Proceedings of the IFAC SICICA'92 Symposium*, Málaga, Spain, May, pp. 105–108.

[**Braae 78**] Braae, M. and Rutherford, D.A. (1978), "Fuzzy Relations in Control Setting". in *Kibernetes 7*, pp. 185–188.

[**Brena 89**] Brenan, K.E., Campbell, S.L., and Petzold L.R. (1989), "Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations". North Holland, New York.

[**Bucha 84**] Buchanan, B.G. (1984),"Rule–Based Expert System. The MYCIN Experiments of the Standord Heuristic Programing Project". Addison–Wesley Publishing Company, USA.

[**Buckl 86**] Buckley, J., Siler W., and Tucker (1986), "A Fuzzy Expert System", in: *Fuzzy Sets and Systems*, **20**, pp. 1–16.

[**Carbo 85**] Carbonell, J., and Minton, S. (1985), " Metaphor and Commonsense Reasoning", in: *Formal Theories of the Commonsense World*, (Hobbs and Moore, eds.). Ablex Publishing Corporation., Norwood, N.J., USA.

[Carbo 89] Carbonell, J.G. (1989), "Paradigms for Machine Learning", in:, *Artificial Intelligence*, Vol. **40**, pp. 1–9.

[Celli 87] Cellier, F.E., and Yandell, D.W. (1987), "SAPS–II: A New Implementation of the Systems Approach Problem Solver," in: *International Journal of General Systems*, **13**(4), pp. 307–322.

[Celli 91a] Cellier, F.E. (1991), "Qualitative Modeling and Simulation: Promise or Illusion," in: *Proceedings WSC'91, Winter Simulation Conference*, Phoenix, Ariz., December 8–11, pp. 1086–1090.

[Celli 91b] Cellier, F.E. (1991), "Continuous System Modeling". Springer-Verlag, N.Y., USA.

[Celli 93] Cellier, F.E., and Elmqvist H., (1993). "Automated Formula Manipulation Supports Object–Oriented Continuous–System Modeling", in: *IEEE Control Systems*, **13**(2), pp. 28–38.

[Chand 89] Chandrasekaran (1989), "A roundtable discussion: Present and Future directions: Trends in Artificial Intelligence," in: *OE Reports, SPIE*, September.

[Chees 90] Cheeseman, P., Kelly, J., Self, M., Stuts, J., Taylor, W., and Freeman, D. (1990), "AutoClass: A Bayesian Classification System", in: *Readings in Machine Learning*, (Shavlik, J.W., and Dietterich, T.G., Eds.). Morgan Kaufmann Publishers, California, USA.

[Chen 93] Chen, C.L., Chen, P.C., and Chen, C.K. (1993), "Analysis and Design of Fuzzy Control System", in: *Fuzzy Sets and Systems*, bf 57(1), pp. 125–140.

[Chi 92] Chi, S.D. (1992), Modeling and Simulation for High Autonomy Systems, Ph.D. dissertation, University of Arizona, Tucson Arizona, U.S.A.

[Clark 84] Clark, D. W., (1984), "Self–Tuning Control of Nonminimum–Phase Systems," in: *Automatica*, **20**, pp. 501–517.

[deAlb 93a] de Albornoz, A., and Cellier, F.E. (1993),"Qualitative Simulation Applied to Reason Inductively about the Behavious of a Quantitatively Simulated Aircraft Model", in: *Proceedings QUARDET'93, Qualitative Reasoning and Decision Technologies*, Barcelona, Spain, pp. 711–721.

[deAlb 93b] de Albornoz, A., and Cellier, F.E. (1993),"Variable Selection and Sensor Fusion in Automatic Hierarchical Fault Monitoring of Large Scale Systems", in: *Proceedings QUARDET'93, Qualitative Reasoning and Decision Technologies*, Barcelona, Spain, pp. 722–734.

[**deAlb 94a**] de Albornoz, A., and Cellier, F.E. (1994),"Building Intelligence into an Autopilot — Using Qualitative Simulation to Support Global Decision Making", in: *Simulation*, **62**(6), pp. 354–364.

[**deAlb 94b**] de Albornoz, A., Sardá, J., and Cellier, F.E. (1994), "Structure Identification in Variable Structure Systems by Means of Qualitative Simulation", in: *Proceedings ESM'94, European Simulation MultiConference*, Barcelona, Spain, pp. 486–491.

[**deAlb 95**] de Albornoz, A. (1995), "Inductive Reasoning and Reconstruction Analysis: Two Complementary Tools for Qualitative Fault Monitoring and Decission Support in Large–Scale Systems". Ph.D. thesis in preparation, L.S.I., Universidad Politécnica de Cataluña, Barcelona, Spain.

[**Davis 82**] Davis, R. (1982), "Diagnosis Based on Description of Structure and Function", in: Proceedings of the National Conference on Artificial Intelligence (AAAI-82).

[**Davis 84**] Davis, R. (1984), "Diagnostic Reasoning Based in Structured Behavior", in: Artificial Intelligence 24(1-3), pp. 347–411.

[**deKle 77**] de Kleer, J. (1977), "Multiple Representations on Knowledge in a Mechanics Problem Solver" in: Proceedings of International Joint Conference in Artificial Inteligence.

[**deKle 82**] de Kleer, J., and Brown, J.S. (1982), "Foundations of Envisioning", in: *Proceedings of the National Conference on Artificial Intelligence (AAAI-82)*, August.

[**deKle 84**] de Kleer, J., and Brown, J.S. (1984), "A Qualitative Physics Based on Confluences," in: *Artificial Intelligence*, **24**.

[**Demps 68**] Dempster, A. (1968), "A Generalization of Bayesian Inference", in: *Jour. Stat. Soc*, bf 30(b), pp. 205–247.

[**deSil 91**] de Silva, C., (1991), "An Analytical Framework for Knowledge–Based Tuning of Servo Controllers," in: *Engineering Aplications of Artificial Intelligence*, **4**(3), pp. 177–189.

[**Drian 93**] Driankov, D. (1993), "Introduction to Fuzzy Control". Springer Verlang.

[**Dubes 79**] Dubes, R. and Jain, A.K. (1985), "Validity Studies in Clustering Mehodologies", in: *Pattern Recognition*, bf 11, pp. 235–254.

[**Duboi 85a**] Dubois, D. and Prade, H. (1985), "Unfair Coins and Necessity Measures: Toward a Possiblistic Interpretation of Histograms", in: *Fuzzy Sets and Systems*, bf 10(1), pp. 15–20.

[**Duboi 85b**] Dubois, D. and Prade, H. (1985), "The Generalized Modus Ponens Under Sup-min composition — A Theoretical Study", in: *Approximate Reasoning in Expert Systems*, (Gupta, M.M., Kandel, A., Bandler, W., and Kiszka, J.B. eds.), North Holland, Amsterdam, pp. 217–232.

[**Duboi 91**] Dubois, D. and Prade, H., (1991), "Fuzzy Sets in Approximate Reasoning, Part 1 and 2", in: *Fuzzy Sets and Systems*, **40**, pp. 143–202 and pp. 203,244.

[**Duboi 93**] Dubois, D., Prade, H., and Yager R.R., (1993), "Readings in Fuzzy Sets for Intelligent Systems", Morgan Kaufmann, San Mateo, CA.

[**Duda 80**] Duda, R.O., Gaschnig, J., and Hart P.E., (1980), "Model Design in the Prospector Consultant System for Mineral Explotation", in: *Expert Systems in the Micro Electronic Age*, (Michie, D, ed.), Edinburgh University Press., pp. 153–167.

[**Elmqv 78**] Elmqvist, H. (1978), *A Structured Model Language for Large Continuous Systems*. Ph.D. dissertation, Report CODEN: LUTFD2/(TFRT–1015), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden, 1978.

[**Efsta 89**] Efstathiou J. (1989), "Expert Systems in Process Control", (Flanagan T. P., Ed.). Longman, U.K.

[**Fishe 90**] Fisher, D.H. (1990), "Knowledge Acquisition Via Incremental Conceptual Clustering", in: *Readings in Machine Learning*, (Shavlik, J.W., and Dietterich, T.G., Eds.). Morgan Kaufmann Publishers, California, USA.

[**Forbu 81**] Forbus, K.D. (1981), "Qualitative Reasoning About Physical Processes", in: *Proceedings of the Seventh Intl. Joint Conf. on Artificial Intelligence (IJCAI-81)*, Vancouver, B.C., Canada.

[**Forbu 84**] Forbus, K.D. (1984), "Qualitative Process Theory", in: *Artificial Intelligence*, **24**, pp. 85–168.

[**Forbu 85**] Forbus, K.D. (1985), "The Role of Qualitative Dynamics in Naive Physics", in: Formal Theories in the Commonsense World, (Hobbs and Moore, eds.), Ablex Publishing Corporation, Norwood, N.J., USA.

[**Fried 81**] Friedman, L. (1981), "Extended Plusible Inference", in: *Proceedings of the Seventh Intl. Joint Conf. on Artificial Intelligence (IJCAI-81)*, pp. 487–495, Vancouver, B.C., Canada.

[**Garci 91**] García, A (1991), "Aplicaciones Actuales de la Lógica Borrosa", in: *Automática e Instumentación*, **216**, pp. 113–119.

[**Genes 84**] Genesereth, M.R. (1984), "The Use of Design Description in Automated Diagnosis", in: Artificial Intelligence 24(1-3), pp. 411–436.

[**Grant 89**] Grant, E., and Zhang, B. (1989), "A Neural Net Approach to Supervised Learning of Pole Balancing", in: *IEEE Int. Symposium on Intelligent Control*, pp. 123–129.

[**Gupta 81**] Gupta, M.M. (1981), "Feedback Control Applications of Fuzzy Set Theory: A Survey", in: *Proceedings 8th IFAC World Congress*, ?.

[**Hayes 79**] Hayes, P.J. (1979), "The Naive Physics Manifesto", in: *Expert Systems in the Micro-Electronic Age*, (D. Mitchie, ed.). Edinburgh University Press, Edinburgh, Scotland,UK.

[**Hayes 85**] Hayes, P.J. (1985), "The Second Naive Physics Manifesto", in: *Formal Theories of the Commonsense World*, (Hobbs and Moore, eds.). Ablex Publishing Corporation, Norwood, N.J., USA.

[**Hobbs 85**] Hobbs and Moore, eds. (1985), "Formal Theories of the Commonsense World". Ablex Publishing Corporation, Norwood, N.J., USA.

[**Holla 87**] Holland, J.H., Holyoak, K.J., Nisbett, R.E. and Thagard, P.R. (1987), "Induction, Processes of Inference, Learning and Discovery", (Feldman, J.A., Hayes, P.J. and Rumelhat, D.E., eds.). MIT Press, London, England.

[**Hunt 66**] Hunt, E.B., Marin, J., and Stone, P. (1966), "Experiments in Induction", Academic Press, New York.

[**Hunt 91**] Hunt, K.J., Sbarbaro, (1991), "Neural Networks for Non-linear Internal Model Control", in: *Proc. IEE Pt. D.*, **138**, pp. 431–438.

[**Hunt 92a**] Hunt, K.J. (1992), "Induction of Decision Trees for Rule–based modelling and Control", in: *Proceedings of the IEEE International Symposium on Intelligent Control*, Glasgow, Scotland, U.K. August, pp. 306–311.

[**Hunt 92b**] Hunt, K.J., Sbarbaro, D., Zbikowski, R., and Gawthrop, P.J. (1992), "Neural Networks for Control Systems–A Survey", in: *Automatica*, **28**(6), pp. 1083–1112.

[**Isido 89**] Isidori A. (1989), "Nonlinear Control Systems: an Introduction". Springer Verlang, Berlin.

[**Iwasa 86a**] Iwasaki, Y. and H. Simon, "Causality in Device Behavior", Artificial Intelligence, 29(1) pp. 3–32.

[**Iwasa 86b**] Iwasaki, Y. and H. Simon, "Causal Ordering Analysis: Reply to de Kleer and Brown", Artificial Intelligence, 29(1) pp. 33–61.

[**Iwasa 91**] Iwasaki, Y., "Causal Ordering Analysis", (Fishwick and Luker, eds.) Qualitative Simulation Modeling and Analysis, Springer-Verlag, USA.

[**Jager 92**] Jager, R., Verbruggen, H.B., and Bruijn, P.M. (1992), "The Role of Defuzzification Methods in the Application of Fuzzy Control", in: *Proceedings of the IFAC SICICA'92 Symposium*, Málaga, Spain, May, pp. 111–116.

[**Kalls 79**] Kallström, C.G., Åström, K.J., Thorell, N.E., Eriksson, J., and Sten, L., (1989), "Adaptative Autopilots for Tankers," in: *Automatica*, **15**, pp. 241–252, 1979.

[**Kalls 81**] Kallström, C.G., and Åström, K.J., (1981), "Experiences of System Identification Applied to Ship Steering," in: *Automatica*, **17**, pp. 187–198.

[**Kanad 89**] Kanade, T. (1989), "A roundtable discussion: Present and Future directions: Trends in Artificial Intelligence", in: *OE Reports, SPIE*, September.

[**Kande 86**] Kadel, A. (1986), "Fuzzy Mathematical Techniques with Applications". Addison–Wesley Publishing Company, USA.

[**Karr 89**] Karr, C.L. Freeman, L.M., and Meredith, D.L. (1989), "Improved Fuzzy Process Control of Spacecraft Autonomous Rendezvous Using Genetic Algorithm", in: *Proceedings SPIE Intelligent Control and Adaptative Systems Conference, Philadelphia*, Penn., pp. 274-288.

[**Karr 91**] Karr, C.L. (1991), "Genetic Algorithms for Fuzzy Controlers", in: *AI Expert*, February.

[**Kiker 78**] Kikert, W.M.J., and Mamdani, E.H. (1978), "Analysis of a Fuzzy Logic Controller", in: *Fuzzy Sets and Systems*, **1**(1).

[**King 77**] King, P.J. and Mamdani, E.H. (1977), "The Application of Fuzzy Control System to Industrial Processes," in: *Automatica*, **13**(3), pp. 235–242.

[**Klee 81**] Klee, V., and Ladner, R. (19815), "Qualitative Matrices: Strong Sign-solvability and Weak Satisfiability", in: Computer–Assisted Analysis and Model Simplification,pp. 293–320, Academic Press.

[**Klir 85**] Klir, G.J. (1985), "Architecture of Systems Problem Solving". Plenum Press, New York.

[**Klir 88**] Klir, G.J., and Folger, T.A. (1988), "Fuzzy Sets, Uncertainty and Information". Engle–wood Cliffs, Prentice Hall, New Jersey, USA.

[**Kosik 87**] Kosikov, V.S., and Kurdyukov, A.P. (1987), "Design of a Nonsearching Self–adjusting System for Nonlinear Plant", in: *Automatika i Telemakhanika*, **4**, pp. 58–65.

[**Kosko 92**] Kosko, B. (1992). "Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence". Prentice Hall, USA.

[**Kraft 92**] Kraft, L.G., Miller, W.T., and Dietz, D. (1992). "Development and Application of CMAC Neural Network-Based Control," in: *Handbook of Intelligence Control*, (D.A. White and Donald A. Sofge, Eds.). Van Nostrand Reinhold, New York.

[**Krijg 92**] Krijgsman A.J. (1992), "Associative Memories: The CMAC approach", in: *Application of Artificial Intelligence in Process Control*, (L. Boullart, A. Krijgsman and R.A. Vingerhoeds, Eds.). Pergamon Press, UK.

[**Krish 93**] Krishnapuram, R., and Keller, J.M. (1993), "A Possibilistic Approach to Clustering", in: *IEEE Transactions on Fuzzy Systems*, **1**(2).

[**Kuipe 82**] Kuipers, B. (1982), "Getting the Envisionment Right", in: *Proceedings of the National Conference on Artificial Intelligence (AAAI-82)*, August.

[**Kuipe 84**] Kuipers, B. (1984), "Commonsense Reasoning About Causality: Deriving Behaviour From Structure",in: *Artificial Intelligence*, **24**.

[**Kuipe 86**] Kuipers, B. (1986). "Qualitative Simulation", in: *Artificial Intelligence*, **29**(3).

[**Kuipe 91**] Kuipers, B., C. Chiu, D.T. Dalle Molle and D.R. Throop, "Higher-order derivative constraints in qualitative simulation", Artificial Intelligence, Vol. 51, Numbers 1-3, October 1991.

[**Langa 88**] Langari G., and Tamizuka, M. (1988). "Fuzzy Linguistic Control of Arc Welding", in: *Sensors and Controls for Manefacturing, ASME.*

[**Law 90**] Law A., and D. Kelton, *Simulation Modeling and Analysis*, 2nd Edition, McGraw–Hill, New York, 1990.

[**Layne 93**] Layne, J., and Passino, K. (1993), "Fuzzy Model Reference Learning Control for Cargo Ship Steering", in: *IEEE Control System*, **13**(6), pp. 23–34, 1993.

[**Lee 90a**] Lee C.C. (1990), "Fuzzy Logic in Control Systems: Fuzzy Logic Controller-Part I and 2", in: *IEEE Trans. on Systems, Man, and Cybernetics*, **20**(2), pp. 404–418 and **20**(2) pp. 419–435.

[**Lee 90b**] Lee C.C. (1990), "A Self–Learning Rule–Based Controller With Approximate Reasoning and Neural Nets", in: *Proceedings 11th IFAC World Congress*, Tallinn, Estonia.

[**Lee 94**] Lee, M., Lee, S., and Park, C. H. (1994), "Neuro-Fuzzy Identifiers and Controllers", in: *Journal of Intelligent and Fuzzy Systems*, **2**(1), pp. 1–14.

[**Levin 74**] Levins, R. (1974), "Qualitative Analysis of Partially Specified Systems", in: Annals of the New York Academy of Sciences 231 123–138.

[**Li 90**] Li, D., and Cellier, F.E. (1990), "Fuzzy Measures in Inductive Reasoning", in: *Proceedings 1990 Winter Simulation Conference*, New Orleans, LA, pp. 527–538.

[**López 94**] López J. (1994). "Contribución a la Predicción de Series Temporales Utilizando Técnicas de Razonamiento Inductivo Borroso". Ph.D. thesis proposal, I.C., Universidad Politécnica de Cataluña, Barcelona, Spain.

[**Maier 85**] Maiers, J. and Y.S. Sherif,"Applications of Fuzzy Set Theory", *IEEE Transactions on Systems, Man and Cybernetics*, **SMC-15**, pp. 175–186, 1985.

[**Mamda 74a**] Mamdani, E.H. (1974), " Application of Fuzzy Algorithms for Control of Simple Dynamic Plant", in: *IEEE Proccedings*, **121**(12).

[**Mamda 74b**] Mamdani, E.H., and Assilian, S. "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller", in: *Fuzzy Reasoning and its Applications* (Mamdami, E.H., and Gaines, B.R. eds.). Academic Press, New York, USA, pp 311-323.

[**Mamda 75**] Mamdani, E.H., and Assilian, S. "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller", in: *Int. J. Man Machine Studies, 7*(1), pp. 1–13.

[**Mamda 81**] Mamdani, E.H., and Assilian, S. "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller", in: *Fuzzy Reasoning and its Applications* (Mamdami, E.H. ed.) London Academic Press, pp. 311–323.

[**Mathw 92**] "MATLAB: High Performance Numeric Computation and Visualization Software — Reference Guide", Natick, Mass., USA.

[**Mayne 90**] Mayne, D.Q., and Michalska, H., (1990), "Receding Horizon Control of Nonlinear Systems", in: *IEEE Transactions on Aut. Control*, **35**, pp. 814–824.

[**McCull 43**] McCulloch, W.S. and Pitts, W. (1943), "A logical Calculus of the Ideas Immanent in Nervous Activity," in: *Bulletin of Mathematical Biophisics*, **9**, pp. 127–147.

[**Medin 94**] Medina, S. (1994), "Una Contribución a la Generalización del Conocimiento. Aplicación al Razonamiento Inductivo Difuso". Ph.D. proposal thesis, E.S.A.I.I., Universidad Politécnica de Cataluña, Barcelona, Spain.

[**MGA 86**] "ACSL: Advanced Continuous Simulation Language – User Guide and Reference Manual", Mitchel and Gauthier Assoc., Concord, Mass. USA.

[**Micha 83**] Michalsky, R.S. (1983) "A Theory and Methodology of Inductive Learning", in: *Artificial Inteligence*, **20**, pp. 111–161.

[**Minge 89**] Mingers, J. (1989), "An Empirical Comparison of Selection Measures for Decision–tree Induction," in: *Machine Learning*, **3** pp. 319–342.

[**Minsk 84**] Minsky, M.L. (1984), "Adaptative Control: From Feedback to Debugging, Adaptative Control of Ill-Defined Systems". Plenum Press, New York, pp. 115-126

[**Mille 90**] Miller, W.T., Sutton, R.S., and Werbos, P.J. (1990), "Application of a General Learning Algorithm to the Control of Robotic Manipulator", MIT Press, Cambridge, MA.

[**Mizum 87**] Mizumoto, M. and Zimmermann, H. (1985), "Comparison of Fuzzy Reasoning Methods", in: *Fuzzy Sets and Systems*, bf 8, pp. 253–283.

[**Murak 85**] Murakami, M., and Maeda, M. (1985), "Application of Fuzzy Controlled to Automobile Speed Control System", in: *Industrial Applications of Fuzzy Control*, (M. Sugeno, ed.)pp. 105–124, Elsevier, North Holland, Amsterdam.

[**Muray 85**] Murayama, Y. and Terano, T. (1985), "Optimising Control of Disel Engine,"in: *Industrial Applications of Fuzzy Control*, (Sugeno, M. ed.) pp. 63–72, Elsevier, North–Holland, Amsterdam.

[**Naren 90**] Narendra, K.S., and Parthasarathy, K. (1990), "Identification and Control of Dynamical Systems Using Neural Networks", in: *IEEE Transactions on Neural Networks*, **1**(1), pp. 4–27.

[**Nebot 93**] Nebot, A., Cellier, F.E., and Linkens, D.A. (1993), "Controlling an Anaesthetic Agent by Means of Fuzzy Inductive Reasoning", in: *Proceedings QUARDET'93, Qualitative Reasoning and Decision Technologies*, Barcelona, Spain, pp. 345–356.

[**Nebot 94a**] Nebot, A., Medina, S., and Cellier, F.E. (1994), "The Causality Horizon: Limitations to Predictability of Behavior Using Fuzzy Inductive Reasoning", in: *Proceedings ESM'94, European Simulation MultiConference*, Barcelona, Spain, June 1-3, pp. 492-496.

[**Nebot 94b**] Nebot A. (1994). "Qualitative Modeling and Simulation of Biomedical Systems Using Fuzzy Inductive Reasoning". Ph.D. thesis, L.S.I., Universidad Politécnica de Cataluña, Barcelona, Spain.

[**Nomot 57**] Nomoto, K. (1957) "Response Analysis of Manoeuvrability and its Applications and its Designs", in: *6oth. Aniversary Series, II, Soc. Of Naval Arch of Japan.*

[**Oleye 89**] Oleyeleye, O.O., and, Kramer, M.A. (1989). "The Role of Causal and Noncausal Constraints in Stady–State Qualitative Modeling", in: *Artificial Intelligence, Simulation and Modeling*, (Widman, L.A., Loparo, K.A., and Nielsen, N.N. eds.), John Wiley, N.Y., pp. 257–274.

[**Pan 84**] Pan, Y.C. (1984). "Qualitative Reasoning with Deep-Level Mechanism Models for Diagnoses of Dependent Failures". Ph.D. thesis, University of Illinois at Urbana-Champaign.

[**Pante 88**] Pantelides, C. C. (1988), "The Consistent Initialization of Differential–Algebraic Systems", in: *SIAM J. Sci. Stat. Comput.*, **9**(2), pp. 213–231.

[**Pedry 91**] Pedrycs W. (1991), "Fuzzy Modelling: Fundamentals, Construction and Evaluation," in: *Fuzzy Sets and Systems*, **41**, pp. 1–15.

[**Pedry 93**] Pedrycs W. (1993), "Fuzzy Control and Fuzzy Systems". 2nd extended, edition, Research Studies Press, UK.

[**Pisku 92**] Piskunov, A. (1992), "Fuzzy Implication in Fuzzy Systems Control," in: *Fuzzy Sets and Systems*, **45**, pp. 25–35.

[**Porte 87**] Porter, A.H.J., and McKeown, C.B. (1987), " Real–Time Expert Tuners for PI Controllers", in: *IEEE Proccedings*, **134**(4)Part D, pp. 260–263.

[**Procy 79**] Procyk, T.J., and Mamdani, E.H. (1979), "A Linguistic Self-Organizing Process Controller", in: *Automatica*, **15**(1), pp. 15–30.

[**Pucci 85**] Puccia, C.J., and Levins, R. (1985), "Qualitative Modeling of Complex Systems". Harvard University Press, Cambride, Massachusetts, USA.

[**Quinl 79**] Quinlan, J.R. (1979), "Discovering Rules from Large Colections of Examples: A Case Study," in: *Expert Systems in the Microelectronic Age* (D. Michie, ed.) Edinburgh University Press, Edinburgh, UK.

[**Quinl 83**] Quinlan, J.R. (1983), "Inferno: A Cautious Approach To Uncertain Inference," in: *The Computer Journal*, **26**, pp. 255–269.

[**Quinl 86**] Quinlan, J.R. (1986), "Induction of Desicion Trees," in: *Machine Learning*, **1** pp. 81–106.

[**Quinl 87**] Quinlan, J.R. (1987), "Simplifyng Desicion Trees," in: *International Journal on Man–Machine Studies*, **27**, pp. 221–234.

[**Renho 91**] Renhong, Z., and Govind, R. (1991), "Defuzzification of Fuzzy Intervals," in: *Fuzzy Sets and Systems*, **43**, pp. 45–55.

[**Reite 81**] Reiter, J. (1981), "AL/X: An Inference System for Probabilistic Reasoning", *M.Sc. Thesis*, Departament of Computer Science, University of Illinois at Urbana–Champaign.

[**Salle 89**] Sallé, S.E., and Årzén, K.E. (1989), "A Comparison Between Three Development Tools for Real–time Expert Systems: Chronos, G2, and Muse," in: *IEEE Control Systems Society Workshop on Computer–Aided Control System Design (CACSD)*, Tampa, Fl.

[**Sarid 89**] Saridis, G.N. (1989), "Analytic Formulation of the Principle of Increasing Precision with Decreasing Intelligence for Intelligent Machines," in: *Automatica*, **25**(3), pp. 461–467.

[**Schar 85**] Scharf, E.M., and Mandic, N.J. (1985), "The Application of a Fuzzy Controller to the Control of a Multi–degree–freedom Robot Arm", in: (M. Sugeno, ed.) *Industry Applications of Fuzzy Control*, North Holland, pp. 41-62.

[**Schli 86**] Schlimmer, J.C., and Fisher, D. (1986), "A Case of Incremental Concept Induction," in: *Proceedings of the Fifth National Conference on AI*, Morgan Kaufmann, Philadelphia, pp. 496–501.

[**SCT 85**] "CTRL-C: A Language for the Computer-Aided Design of Multivariable Control Systems, User's Guide", Systems Control Technology Inc., Palo Alto, Ca, USA.

[**Shafe 76**] Shafer, G. (1976), "A Mathematical Theory of Evidence", Princeton Univ. Press, New Jersey, USA.

[**Sugen 85a**] Sugeno, M. and Murakami, M.(1985), "An Experimental Study on Parking Control Using a Model Car", in: *Industrial Applications of Fuzzy Control*, (M. Sugeno, ed.)pp. 125–138, Elsevier, North Holland, Amsterdam.

[**Sugen 85b**] Sugeno, M. (1985), "An Introductory Survey of Fuzzy Control", in: *Information Science*, **36**(1), pp. 59–83.

[**Sugen 88**] Sugeno, M., and Kang, G.T. (1988), "Structure Identification of Fuzzy Model", in: *Fuzzy Sets and Systems*, **28**, pp. 15–33.

[**Sugen 93**] Sugeno, M. and Yasukawa, T. (1993) "A Fuzzy Logic Based Approach to Qualitative Modeling", in: *IEEE Trans. on Fuzzy Systems*, **1**(1) pp. 7–31.

[**Takag 83**] Takagi, H. and Sugeno M. (1983), "Derivation of Fuzzy Control Rules from Human Operator's Control Actions", in: *IFAC Symposium on Fuzzy Information, Knowledge Representation and Decision Analysis*, Marseille, France. pp. 55–60.

[**Takag 85**] Takagi, H. (1985), "Fuzzy Identification of Systems and Its Application to Modeling and Control", in: *IEEE Trans. on Systems, Man, and Cybernetics*, **15**(1),pp. 116–132.

[**Tarja 72**] Tarjan, R. (1972), "Depth-First Search and Linear Graph Algorithm", in: *SIAM Computation*, **1**(2),pp. 146–160.

[**Torra 89**] Torras, C. (1989), "Relaxation and Neural Learning: Points of Convergence and Divergence," in: *Journal of Parallel and Distributed Computing*, **6**(1),pp. 217–244.

[**Trill 85**] Trillas, E. and Valverde, L. (1985), "On mode and Implication in Approximate Reasoning", in: *Approximate Reasoning in Expert Systems*, (Gupta, M.M., Kandel, A., Bandler, W., and Kiszka, J.B. eds.), North Holland, Amsterdam, pp. 157–166.

[**Tzuka 79**] Tzukamoto, Y. (1979), "An approach to Fuzzy Reasoning Method", in: *Advances in Fuzzy Set and Applications*, (Gupta, M.M., Ragade, R.K., and Yager, R.R. eds.), North Holland, Amsterdam.

[**Utgof 89**] Utgoff, P.E. (1989), "Incremental Induction of Desicion Trees," in: *Machine Learning*, **4** pp. 161–186.

[**Uytte 78**] Uyttenhove, H.J. (1978), "Computer–aided System Modeling Readings: An Assemblage of Methodological Tools for System Problem Solving". Ph.D. Dissertation, School of Advanced Technology, SUNY-Binghamton, New York.

[**Uytte 81**] Uyttenhove, H.J. (1981), "SAPS (System Approach Problem Solver): An Introduction and Guide". Computing and Systeme Consultants, Binghamton, New York.

[**Vesan 89**] Vesanterä, P., and Cellier, F.E. (1989), "Building Intelligence into an Autopilot – Using Qualitative Simulation to Support Global Decision Making", in: *Simulation*, **52**(3), pp. 111–421.

[**Verbr 91**] Verbruggen, H.B., Krijgsman, A.J., and Bruijn, P.M. (1991), "Towards Intelligent Control", in: Journal A, **32**(1).

[**Wang 91**] Wang, Q., and Cellier, F.E. (1991), "Time Windows: Automated Abstraction of Continuous-Time Models into Discrete Event Models in High Autonomy Systems", in: *Int. J. Gen. Syst.*, **19**(3), pp. 241–262.

[**Weld 90**] Weld, D.S., and de Kleer, J. eds. (1990), " Readings in Qualitative Reasoning about Physical System". Morgan Kaufman, San Mateo, CA, USA.

[**Willi 91**] Williams, B.C., and de Kleer, J. (1991), "Qualitative Reasoning About Physical Systems: A Return to Roots", in: Special Volume on Qualitative Reasoning about Physical Systems II *Artificial Intelligence*, **51**(1-3), pp. 1–473.

[**Wolov 74**] Wolovich, W.A. (1974), "Linear Multivariable Systems". Springer-Verlag, New York, USA.

[**Wu 92**] Wu, Z.Q., Wang, P.Z., and Teh, H.H. (1992), "A Rule Self–Regulating Fuzzy Controller", in: *Fuzzy Sets and Systems, 47*, pp. 13–21.

[**Yager 94**] Yager, R.R. and Fliev, D.P. (1994), "Essentials of Fuzzy Modeling and Control". Jhon Wiley & Sons, Inc. New York, USA.

[**Ying 90**] Ying, H., Siler, W., and Buckley, J. (1990), "Fuzzy Control Theory: A Nonlinear Case", in: *Automatica*, **26**(3), pp 513–520.

[**Yu 90**] Yu, C., Cao,Z., and Kandel, A. (1990), "Application of Fuzzy Reasoning to the Control of an Activated Sludge Plant", in: *Fuzzy Sets and Systems*, **38**, pp. 1–14.

[**Zadeh 68**] Zadeh, L.A. (1968), "Fuzzy Algorithm", in: *Informat. Control*, 8, pp 338–353.

[**Zadeh 71**] Zadeh, L.A. (1971), "A rationale for Fuzzy Control", in: *ASME, J. Dynam. Syst. Measur. Control*, **94**, pp. 3–4.

[**Zadeh 72**] Zadeh, L.A. (1972), "A Fuzzy Set Theoretic interpretation of linguistic headges", in: *Journal of Cybernetics*, **2**, pp. 4–34.

[**Zadeh 73**] Zadeh, L.A. (1973), "Outline of a New Approach to the Analysis Complex Systems and Decision Processes", in: *IEEE Trans. Sys. Man Cybern.*, **SMC-3**, pp. 28–44.

[**Zadeh 75**] Zadeh, L.A. (1975), "The Concept of a Linguistic Variable and its Applications to Approximate Reasoning I, II, III", in: *Informat. Sci.*, **8**, pp. 199–251, pp. 301–357 and **9**, pp. 43–80.

[**Zadeh 79**] Zadeh, L.A. (1979), "A Theory of Approximate Reasoning", in: *Machine Intelligence*, (Hayes, J., Michie, D., and Mikulich, L.I. eds.) Halstead Press, New York; **9**, pp. 149–194.

[**Zadeh 92**] Zadeh, L.A. (1992), Foreword of Chapter 4: "Fuzzy Logic in Control Engineering" in: *Handbook of Intelligence Control*, (D.A. White and Donald A. Sofge, Eds.). Van Nostrand Reinhold, New York.

[**Zeigl 89**] Zeigler, B.P. (1989) "DEVS Representation of Dynamical Systems: Event–Based Intelligent Control" in: *Conference on AI and P Autonomy Systems*, Florida, USA.

[**Zeigl 91**] Zeigler, B.P. and Chi, S.D. (1991) "Symbolic Discrete Event System Specification" in: *IEEE Proceedings, 77*(1), pp. 72–80.

[**Zimme 85**] Zimmermann, H.J. (1985), "Fuzzy Set Theory and Its Applications". Kluwer Academic Publishers, Boston, USA.

[**Zhu 93**] Zhu, Y., and Backx, T., (1993), "Identification of Multivariable Industrial Processes", Springer–Verlang, London, UK.