13

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMA DE CONTROL DE ALTA RESOLUCIÓN PARA UN MOTOR DE PASOS

T E S I S

Que para obtener el Título de

INGENIERA MECÁNICA

P r e s e n t a n

acompañado de un disco compacto

Laura Adriana Oropeza Ramos

Teresa Alejandra Vidal Calleja

Director de Tesis: Dr. Enrique Geffroy Aguilar

México, D. F. 2000

A mis padres,

a Iñaki,

a mi hermano y a mi abuelita.

TERE

A mis padres y a mi hermano.
Porque simplemente les
debo todo lo que soy,
y porque este logro es
también de ustedes.

A Líber, porque juntos
encontramos el camino.
¡Hasta el infinito y mas allá!.

LAURA

AGRADECIMIENTOS

Al **Dr. Enrique Geffroy** porque no sólo ha sido un gran asesor, sino que además ha sido un verdadero amigo mostrando siempre disposición a escuchar, a dar un buen consejo y porque nos brindó su apoyo incondicional en los momentos más difíciles.

Al **Instituto de Investigaciones en Materiales** que nos alojó durante todo este tiempo.

A la **Facultad de Ingeniería** porque más allá de darnos una excelente formación académica nos permitió desarrollar otras actividades como la fundación de la Sociedad Astronómica SAFIR y la publicación de la revista Sidereus nuncius.

A **nuestra banda de la FI** con la que nos desvelamos por todas aquellas razones por las que se puede hacerlo: Juan Pablo, el Amigo, Pablito, Pillín, Vivi, el Germen, Toño, Iñaki, Tair, Toña, Yusen, el Mimoso, Claudia, Farah, Mumu, el Primo, el Skid, Mauricio, Carrera, el Bern.

A Yukihiro Minami por su apoyo durante la carrera.

A nuestros maestros Ubaldo Márquez, Ignacio Juárez, Rolando Peralta, Margarita Navarrete, Moisés Mendoza por sus excelentes clases.

A Pablo García y Colomé porque aunque no compartimos el aula nos brindó una gran amistad.

Y principalmente a la **UNAM**, que es y seguirá siendo la máxima casa de estudios; por lo que ella representa, por su diversidad de ideas, porque nos permitió no sólo estudiar una carrera universitaria sino que además en ella hemos encontrado razones esenciales por las cuales seguir luchando.

A nuestros amigos con los que no compartimos la carrera pero que siempre estuvieron ahí.

Claudia, Lizette, Vivian, Diana, Coneja, Mariana, el Yuri, Cristina, Maru, Choche, Jorge, Diana Martha, mis primas Wendy y Karla, Isela, Luis, Begoña, Juan Antonio, Ximena, Carlos, mi abuelo Juan, a toda mi familia y especialmente a los que ya no están conmigo.

Tere

Gracias mamá, papá y Adrián. Por su apoyo, por ser mi familia, por lo que hemos vivimos juntos, por lo que me han enseñado.

A Natalia y Alondra, mis grandes amigas.

A Mariana, Paula, Tania, Genoveva, Maru, Pável, Quique Gavilán y Sancho.

A mi banda del Madrid: Natzin, Claus, Tato, Jano, Mónica, Hugo, el Huevo, Zara, el Hongo, Daniel, el Nerd y Mots.

A mis primos, tíos y a mis abuelos que siempre creyeron en mí.

Laura

ÍNDICE

OBJETIVO

El objetivo de este proyecto es llevar a cabo un sistema de control de alta resolución que permita generar las deformaciones necesarias para producir una serie de flujos fuertes por medio de un molino de dos rodillos. De esta manera se completa una parte del experimento de Birrefringencia Bicolor que permite el estudio de la anisotropía óptica de los materiales poliméricos sujetos a grandes deformaciones, lo que permite obtener mayor información de las propiedades mecánicas de los polímeros a lo largo de direcciones preferenciales. Ello representa una mejora sustancial sobre las técnicas establecidas en reología-óptica y así significa una importante aportación para la ciencia y tecnología de los materiales.

DESCRIPCIÓN DEL PROBLEMA

Lograr el acoplamiento de la instrumentación en el Laboratorio de Reología-Óptica, que se refiere a una computadora central, un controlador avanzado de movimiento, un motor de pasos y un protocolo de comunicación entre la computadora y el controlador.

Elaborar los códigos de programación que permitan realizar dos tareas esenciales:

(1) Crear una adecuada interfaz de operación entre el usuario y el sistema de control.

(2) Generar y enviar una secuencia compleja de instrucciones al controlador para que éste maneje el movimiento del motor de pasos con base en las necesidades del experimento de Birrefringencia Bicolor.

HIPÓTESIS

Por medio de un lenguaje de programación gráfico y del control digital se busca generar diversas historias de deformación bien caracterizadas, con una alta precisión en la generación de anisotropías ópticas a partir de las herramientas mencionadas en la descripción del problema. El lenguaje a utilizar permite establecer los dispositivos de comunicación al usuario y del usuario-motor en forma inteligente, que evite errores innecesarios así como eventualidades no previstas en el trabajo experimental.

OBJETIVO

El objetivo de este proyecto es llevar a cabo un sistema de control de alta resolución que permita generar las deformaciones necesarias para producir una serie de flujos fuertes por medio de un molino de dos rodillos. De esta manera se completa una parte del experimento de Birrefringencia Bicolor que permite el estudio de la anisotropía óptica de los materiales poliméricos sujetos a grandes deformaciones, lo que permite obtener mayor información de las propiedades mecánicas de los polímeros a lo largo de direcciones preferenciales. Ello representa una mejora sustancial sobre las técnicas establecidas en reología-óptica y así significa una importante aportación para la ciencia y tecnología de los materiales.

DESCRIPCIÓN DEL PROBLEMA

Lograr el acoplamiento de la instrumentación en el Laboratorio de Reología-Óptica, que se refiere a una computadora central, un controlador avanzado de movimiento, un motor de pasos y un protocolo de comunicación entre la computadora y el controlador.

Elaborar los códigos de programación que permitan realizar dos tareas esenciales:

(1) Crear una adecuada interfaz de operación entre el usuario y el sistema de control.

(2) Generar y enviar una secuencia compleja de instrucciones al controlador para que éste maneje el movimiento del motor de pasos con base en las necesidades del experimento de Birrefringencia Bicolor.

HIPÓTESIS

Por medio de un lenguaje de programación gráfico y del control digital se busca generar diversas historias de deformación bien caracterizadas, con una alta precisión en la generación de anisotropías ópticas a partir de las herramientas mencionadas en la descripción del problema. El lenguaje a utilizar permite establecer los dispositivos de comunicación al usuario y del usuario-motor en forma inteligente, que evite errores innecesarios así como eventualidades no previstas en el trabajo experimental.

OBJETIVO

El objetivo de este proyecto es llevar a cabo un sistema de control de alta resolución que permita generar las deformaciones necesarias para producir una serie de flujos fuertes por medio de un molino de dos rodillos. De esta manera se completa una parte del experimento de Birrefringencia Bicolor que permite el estudio de la anisotropía óptica de los materiales poliméricos sujetos a grandes deformaciones, lo que permite obtener mayor información de las propiedades mecánicas de los polímeros a lo largo de direcciones preferenciales. Ello representa una mejora sustancial sobre las técnicas establecidas en reología-óptica y así significa una importante aportación para la ciencia y tecnología de los materiales.

DESCRIPCIÓN DEL PROBLEMA

Lograr el acoplamiento de la instrumentación en el Laboratorio de Reología-Óptica, que se refiere a una computadora central, un controlador avanzado de movimiento, un motor de pasos y un protocolo de comunicación entre la computadora y el controlador.

Elaborar los códigos de programación que permitan realizar dos tareas esenciales:

(1) Crear una adecuada interfaz de operación entre el usuario y el sistema de control.

(2) Generar y enviar una secuencia compleja de instrucciones al controlador para que éste maneje el movimiento del motor de pasos con base en las necesidades del experimento de Birrefringencia Bicolor.

HIPÓTESIS

Por medio de un lenguaje de programación gráfico y del control digital se busca generar diversas historias de deformación bien caracterizadas, con una alta precisión en la generación de anisotropías ópticas a partir de las herramientas mencionadas en la descripción del problema. El lenguaje a utilizar permite establecer los dispositivos de comunicación al usuario y del usuario-motor en forma inteligente, que evite errores innecesarios así como eventualidades no previstas en el trabajo experimental.

INTRODUCCIÓN

A lo largo de las últimas décadas del siglo veinte se han desarrollado, de manera sostenida, avances tecnológicos de gran importancia e impacto en la ingeniería y en la ciencia. El empleo de instrumentación y sistemas de control precisos ha permitido llevar a cabo innovaciones importantes dentro de una diversidad de procesos.

Sin embargo, la historia de los sistemas de control data desde la época antigua con el desarrollo de sistemas de transportación y servicios básicos, pasando por innumerables trabajos notables como los de James Watt (1789) con su máquina de vapor con regulación de velocidad y la labor de Charles Babbage en 1822 para construir un dispositivo de análisis aritmético, hasta llegar a nuestros días donde sin duda el uso de computadoras ha revolucionado a las sociedades, incluso sus costumbres y modos de vida.

El empleo de computadoras para la ejecución de experimentos o el control de instrumentos es cosa común hoy en día, en particular enfatizando dos tipos de aplicaciones; por una parte, para experimentos muy complejos en donde se procesan una gran cantidad de datos y mediciones, y por otra parte, para reconocer estados anómalos que se pueden presentar de manera inesperada y que requieren la toma de decisiones con criterios posiblemente nunca antes considerados. En general, en situaciones complejas necesitamos desarrollar sistemas de control donde una computadora centralice las operaciones de comunicación y control y los dispositivos con los que se cuenta sean programables.

En los procesos industriales actuales, con frecuencia se requiere conocer las propiedades reológicas de los materiales plásticos, con el propósito de alcanzar un uso de éstos más competitivo. Desde el punto de vista en ciencia de polímeros, la necesidad industrial planteada requiere conocer y entender los mecanismos mediante los cuales los materiales poliméricos relajan sus esfuerzos después de haber sido deformados. Los esfuerzos son el resultado de los cambios en la microestructura del material, siendo estos últimos resultado de las deformaciones debidas al procesado del material.

Este trabajo de tesis está centrado alrededor de aplicaciones de control por computadora para la realización de una colección de procesos complejos. En particular la aplicación se emplea en estudios reológicos de fluidos no-newtonianos mediante la técnica

de Birrefringencia Bicolor Inducida por Flujos (BBIF) del Laboratorio de Reología Óptica del Instituto de Investigaciones en Materiales de la UNAM. Las actividades de dicho laboratorio enfatizan el estudio de la dinámica microestructural de dichos materiales, aplicando historias de deformación desde lineales hasta aquellas fuertemente no lineales. Lo anterior es sólo posible si se genera una diversidad de historias que enfatizan todos los aspectos de la reología de materiales poliméricos.

El objetivo de este proyecto es el llevar a cabo un sistema de control que permite generar las deformaciones necesarias por medio de un molino de dos rodillos. Para esto es necesario, por una parte, el acoplamiento de la instrumentación con la que el Laboratorio de Reología ya cuenta, que se refiere a una computadora central, un controlador avanzado de movimiento, un motor de pasos y un protocolo de comunicación entre la computadora y el controlador. Por otra parte, la elaboración de códigos de programación que permitan realizar dos tareas esenciales: una que es la de crear una adecuada interfaz de operación entre el usuario y el sistema de control de tal forma que libere al usuario de las complejidades e ideosincracias propias del procesador, de la interfaz de comunicación, del lenguaje de programación del dispositivo, etc; y otra que se refiere a generar y enviar una secuencia compleja de instrucciones al controlador para que éste maneje el movimiento del motor de pasos de acuerdo a las necesidades previstas por el operador.

En el Capítulo 1 se describe en forma general el experimento de Birrefringencia Bicolor Inducida por Flujos dando énfasis al dispositivo de flujo para controlar el movimiento de un motor de pasos que produce diferentes historias de deformación en fluidos poliméricos por medio de un molino de dos rodillos.

En el Capítulo 2 se presentan las características y formas de operación de los componentes del sistema de control: el controlador con su *driver* integrado, el motor de pasos de la serie UE7_PP y el lenguaje de programación gráfica HP VEE.

El Capítulo 3 explica el funcionamiento del módulo de control del motor de pasos SMI al igual que todas las rutinas que lo incluyen, tales como la de Flujo de Estado Estacionario, Inicio y Cese de Flujo y Flujo de Doble Escalón, que representan las diferentes historias de deformación necesarias para el estudio minucioso no-newtoniano de un fluido. Así como un Formato Libre que utilizando las instrucciones básicas del controlador del motor, permite evaluar el desempeño de los códigos programables y ejecutables.

CAPÍTULO 1

Dispositivo de Flujo en el Experimento de Birrefringencia Bicolor

1.1. Antecedentes de Dinámica de Polímeros

Un sistema polimérico es aquel constituido de pequeñas unidades moleculares simples *denominadas monómeros.* En la práctica, un polímero puede alcanzar un gran peso molecular cuando existe una enorme concatenación de monómeros, y dado que entre ellos existen grados de libertad en su orientación, entonces la cadena tiene una dinámica muy compleja. Esta dinámica compleja puede observarse en una enorme variedad de características en el material polimérico, que va desde cadenas poliméricas rígidas hasta las muy flexibles, que se comporta como líquidos simples o como sólidos cristalinos, etc. El comportamiento observado no sólo depende de las características de la unidad química (monómero) sino que también es importante la física de las cadenas o redes de monómeros, y por lo tanto, de su dinámica microscópica.

En el campo de las aplicaciones industriales y de producción de polímeros, el conocimiento de la dinámica y la microestructura es de enorme relevancia. Por ejemplo, para mejorar la capacidad a la tensión y ruptura de una fibra polimérica se ha observado que es necesario un alineamiento de las macromoléculas. Durante el procesado del material, las deformaciones que producen un fuerte alineamiento macromolecular o el estiramiento de la cadena polimérica en direcciones preferenciales son un requisito indispensable para mejorar la capacidad a la tensión de la fibra. Ello implica que el ordenamiento microestructural, que induce una anisotropía en el material, es la característica deseable, que sin embargo debe establecerse aún cuando a nivel microscópico la dinámica del material tenderá a destruir dicha anisotropía.

1.1.1. Fluidos Anisotrópicos No Lineales

Una forma de estudiar la *dinámica de sistemas poliméricos* es por medio de experimentos de relajación cuando éstos son deformados. La *dinámica lineal* se tiene cuando las deformaciones son esencialmente infinitesimales. Por *dinámica no lineal* debe entenderse

los mecanismos de relajamiento que se observan cuando el sistema se perturba de manera notable como resultado por ejemplo de una deformación finita. En la práctica, la aplicación de deformaciones pequeñas ocurre en limitadas ocasiones, por lo que resulta entonces de vital importancia el estudio y conocimiento de la dinámica no lineal de los sistemas poliméricos.

En la práctica, es posible inducir deformaciones finitas por medio de flujos. En especial, los *flujos fuertes* son capaces de generar grandes deformaciones microscópicas; son fundamentalmente elongacionales aunque pueden presentar cierto grado de vorticidad. Este tipo de flujos puede generarse en una *celda de flujos de dos rodillos*, en la cual, la rotación de éstos genera flujos bidimensionales. La importancia de los flujos no lineales, y de los flujos fuertes en general, es que tienen el potencial para deformar la estructura de un líquido polimérico desde un estado moderado de cuasi-equilibrio (como bola de estambre) hasta uno altamente anisotrópico (es decir, con una dirección bien definida), condiciones que son imposibles de generar en flujos débiles, tales como los viscométricos [1,2].

Figura 1.1. *Diferencia entre los estados isotrópico y anisotrópico en un material polimérico. En (a) no existen esfuerzos deviatorios, por lo que no presenta direcciones definidas, mientras que en (b) se presenta una deformación promedio inducida mediante esfuerzos por lo que la dirección del flujo establece una dirección preferencial. La birrefringencia del material en el estado anisotrópico es la diferencia entre los índices de refracción en las direcciones $n_{\perp}$ y $n_{\parallel}$.*

1.1.2. Anisotropía Óptica Inducida por Flujos

Los cambios conformacionales en el flujo polimérico se observan porque producen anisotropías ópticas que se pueden medir experimentalmente. Cuando un material isotrópico, en reposo, se somete a esfuerzos mecánicos, se puede volver ópticamente anisotrópico; es decir, el material presenta diferentes valores del índice de refracción,

dependiendo de la dirección de propagación de la luz. La forma de representar esta condición óptica del material es mediante un índice de refracción tensorial, cuyos elementos son números complejos. La parte real de los componentes del tensor es una medida de la velocidad de propagación de las ondas electromagnéticas en el medio, y la parte imaginaria corresponde a la atenuación de las ondas por absorción y difracción. Cuando los valores principales del tensor difieren en su parte real, se dice que el material es *birrefringente*, y cuando sus partes imaginarias son diferentes, el material es *dicroico* [3].

Entonces, la anisotropía estructural del líquido polimérico está asociada con la anisotropía óptica del material, y ésta a su vez repercute en la transmisión de ondas electromagnéticas. Utilizando luz polarizada y analizando los cambios de polarización e intensidad que ésta sufre al atravesar el medio polimérico, se puede determinar los valores principales del índice de refracción. De las diferencias en la parte real e imaginaria de los valores principales se deduce la anisotropía óptica del material, y de esta última se estima la estructura anisotrópica del fluido[1].

1.2. Descripción del Experimento de Birrefringencia Bicolor de Alta Resolución

El experimento de Birrefringencia Bicolor de Alta Resolución se emplea para conocer la anisotropía microestructural de materiales poliméricos. Las anisotropías microestructurales tienen asociadas esfuerzos deviatorios y anisotropías ópticas, y son resultado de deformaciones macroscópicas. Esto es, para poder determinar los primeros se requiere inducir una deformación en el material como las que se producen con un flujo fuerte. A continuación se presentan las ideas básicas tanto de la forma de inducir las deformaciones como la manera de medir las anisotropías ópticas.

1.2.1. La Técnica de Birrefringencia Bicolor de Alta Resolución

Dentro de los métodos experimentales comúnmente utilizados para evaluar los cambios microestructurales de los polímeros, se encuentran: (1) las técnicas mecánicas que

[1] Las soluciones poliméricas muestran un alto grado de birrefringencia, y un grado de dicroísmo muy pequeño. Por ello, se considera únicamente la contribución de la birrefringencia para deducir la anisotropía del material.

determinan los esfuerzos resultado de aplicar deformaciones a la muestra bajo estudio, utilizando los llamados reómetros; y (2) aunque menos utilizadas, las técnicas que miden la anisotropía óptica del medio de interés directamente en el procesado de plásticos. En general, los métodos ópticos resultan más ventajosos tanto técnica como científicamente, aunque son más complejos de establecer en el laboratorio. Además, los parámetros experimentales de medidas reológicas que usualmente se reportan corresponden a promedios macroscópicos evaluados en la frontera del flujo (esto es, sobre las paredes que definen la celda de flujos utilizada). Por lo tanto, resultan medidas más adecuadas para deformaciones infinitesimales, pues para deformaciones finitas no se puede considerar siempre válido que el valor de un parámetro dentro del flujo sea el mismo que su valor en la frontera. Por ello, los métodos ópticos son una alternativa a las medidas mecánicas que, además resultan de gran utilidad cuando es necesario evaluar una respuesta rápida del sistema, a la vez que proveen una indicación del comportamiento del material que es local en la escala del flujo.

La metodología experimental que se utiliza para medir la anisotropía óptica inducida por las moléculas del polímero es la Birrefringencia Bicolor Inducida por Flujos (de la literatura científica *Two Color Flow Birefringence*, TCFB), y se basa en el dispositivo óptico denominado elipsómetro de nulos[2]. Con esta técnica se determina, de manera simultánea, el grado de anisotropía del medio y la orientación de los ejes principales del tensor del índice de refracción. Para esto, utiliza dos rayos de luz de diferente longitud de onda (dos rayos láser de diferente color) y los analiza de manera independiente por medio de los dos elipsómetros de nulos, uno para cada color. Las ventajas principales de la técnica son: (a) permite realizar mediciones puntuales con una resolución de aproximadamente 50μm; (b) es capaz de analizar flujos transitorios realizando mediciones con una frecuencia de 2×10^4 datos por segundo.

Este tipo de técnicas ópticas son ventajosas, en particular, cuando los cambios conformacionales en la microestructura polimérica son inducidos por un flujo no homogéneo, como es el caso de aquellos generados por un molino de dos rodillos. Además, las técnicas ópticas no son invasivas, es decir, que no alteran el flujo mientras se lleva a cabo la medición. La principal desventaja de las técnicas ópticas es que únicamente

[2] Un elipsómetro de nulos es un instrumento que mide los estados de polarización inicial y final de un haz de luz que atraviesa el medio en el estudio.

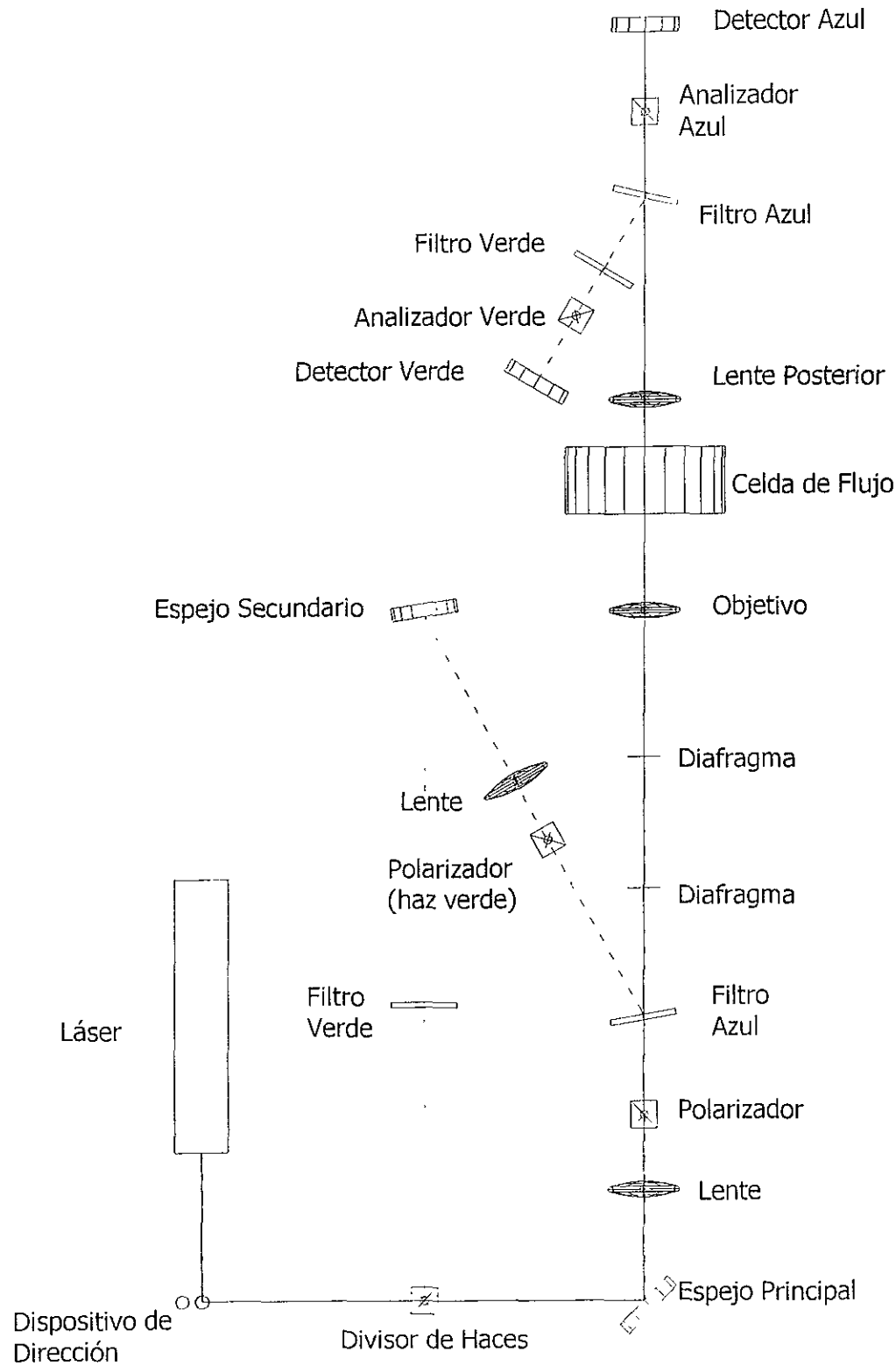


Figura 1.2. *Arreglo óptico para el experimento de Birrefringencia Bicolor Inducida por Flujos (BBIF). [De la tesis de J.P. Jiménez UNAM 2000, 4].*

permiten investigar fluidos transparentes. Los fluidos que contienen partículas suspendidas, por ejemplo, no pueden ser analizados por estos métodos [4]. Afortunadamente los polímeros sólidos, que son opacos a temperatura ambiente debido a su estructura cristalina, se vuelven transparentes por encima de su temperatura vítrea.

El diseño experimental propuesto por Geffroy [3] utiliza un láser de argón *Spectra Physics* modelo 2020, el cual emite dos haces luminosos de longitudes de onda diferentes, azul y verde respectivamente, así como el tren óptico [4] compuesto de polarizadores, lentes, espejos, etc. que se muestra en la Figura 1.2.

1.2.2. El Arreglo Experimental para la Caracterización de Polímeros

Para el desarrollo del experimento dedicado al estudio de la dinámica no lineal de polímeros, se requiere de las siguientes condiciones: (a) controlar con precisión la temperatura del líquido bajo estudio, (b) tener controlada la fuente de luz verde y azul por medio de la computadora, (c) contar con la electrónica que permita convertir y amplificar las señales luminosas provenientes del arreglo para la determinación de la birrefringencia bicolor inducida por flujos, y (d) *realizar un conjunto de diferentes historias de deformación con una celda de flujos del tipo de molino de dos rodillos, cuyo control de movimiento se lleve a cabo mediante un controlador de motores de pasos programable.*

El trabajo de tesis que aquí se presenta, resuelve el Inciso (d) antes mencionado, presentando el control para un motor de pasos que genera las diferentes historias de deformación dentro de la celda de flujo de un molino de dos rodillos. Este controlador es semiautónomo, para lograr gran confiabilidad y versatilidad en la ejecución de las historias de deformación, así como una extrema precisión y repetitividad para tales historias.

Los requerimientos (a), (b) y (c) antes mencionados se desarrollaron en el trabajo de tesis de Mónica Hochstein [6]. En éste se propone que todas las acciones de control de la instrumentación y la adquisición de datos se lleven a cabo a través de una Unidad de Control y Adquisición de Datos HP 3852A y una estación de trabajo HP 382, comunicadas entre sí vía una interfaz IEEE-488 (HP-IB). El control y corrección de la temperatura de la muestra se hace utilizando un voltímetro de 5 ½ dígitos HP 44701A y un convertidor digital analógico HP 44724. La sección de adquisición de datos se compone por una tarjeta de amplificación HP 44736A para las señales luminosas, un multiplexor HP 44713 que sirve para alimentar las señales provenientes de los amplificadores correspondientes a los

distintos detectores y un voltímetro de alta velocidad HP 44704A, el cual digitaliza las señales de voltaje de los detectores en forma secuencial. Por otro lado, la unidad de láser también es parte de los instrumentos bajo control de la computadora, pues resulta indispensable revisar tanto la estabilidad de la fuente de luz azul y verde, así como monitorear los problemas que en ella puedan presentarse. Hochstein propuso la programación necesaria para el arreglo experimental, que se compone de 16 módulos más el programa principal MHGMAIN, que se encarga de la sincronización de los diferentes eventos que se realizan en los instrumentos que se describieron. El módulo de control del motor de pasos se integra a MGHMAIN.

1.2.3. Dispositivo de Flujo de un Molino de Dos Rodillos

En la Sección 1.1.1 se menciona que cuando un flujo induce deformaciones que producen alineamiento y estiramiento de las cadenas poliméricas, las propiedades del material se ven modificadas de manera importante. Para que esto suceda se deben generar flujos fuertes sobre el material polimérico. El experimento de Birrefringencia Bicolor de Alta Resolución del Laboratorio de Reología Óptica del Instituto de Investigaciones en Materiales cuenta con una celda de flujo especial para generar estos flujos fuertes[3] (Figura 1.3).

Esta celda se compone de dos rodillos paralelos, separados por una distancia determinada, y que giran con velocidades de la misma magnitud y sentido. De esta forma, se genera un punto de estancamiento en el punto medio de la separación de los dos rodillos, y en la región alrededor de dicho punto prevalecen las condiciones de flujo elongacional con vorticidad. La combinación de punto de estancamiento y condiciones cinemáticas de flujo fuerte capaces de producir grandes deformaciones son la principal ventaja del molino. Las líneas de flujo para las condiciones cinemáticas (diferentes radios e igual separación entre los ejes se presentan en la Figura 1.3. El punto de estancamiento se muestra del lado derecho de la ilustración. El dispositivo permite variar las condiciones cinemáticas en la región de flujo fuerte tan sólo cambiando los diámetros de los rodillos para así cubrir una amplia variedad de flujos fuertes a través del parámetro de flujo λ, de acuerdo con la siguiente expresión:

[3] El diseño de la celda de flujos del Laboratorio de Reología Óptica, se encuentra a cargo de Juan Pablo Jiménez de la Rosa y de Marco Antonio Reyes Huesca [4,7].

$$\frac{1+\lambda}{1-\lambda} = \frac{|E|}{|\Omega|} = \frac{rapidez\ de\ formación}{vorticidad}$$

En el punto de estancamiento las cadenas poliméricas estancadas tienden a alinearse y estirarse en direcciones determinadas.

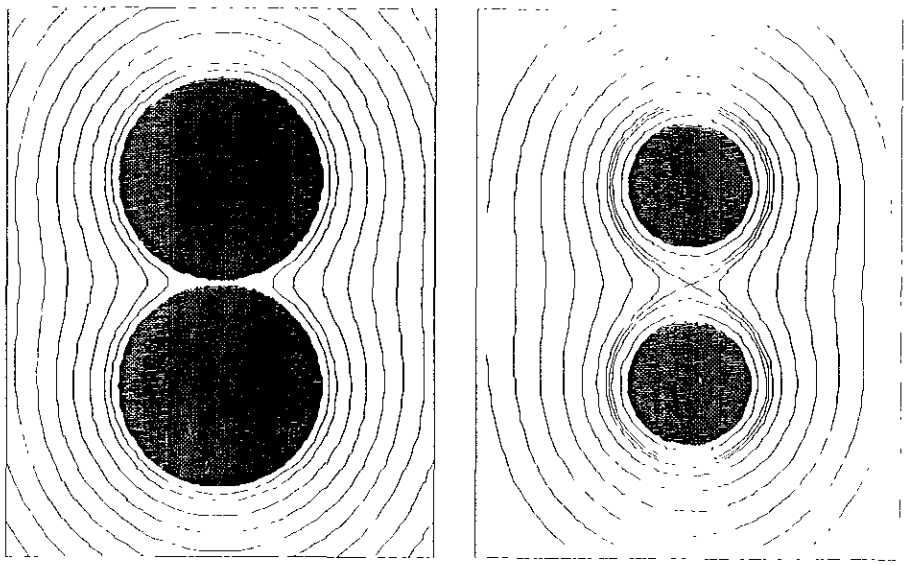


Figura 1.3. *Flujo generado por un molino de dos rodillos. Los parámetros de velocidad de deformación y vorticidad son función de la geometría del molino y de la velocidad de giro de los rodillos* [7].

La Figura 1.4 muestra un esquema de la celda de flujo del arreglo experimental. Para los experimentos que se consideran en el Laboratorio de Reología, el molino de dos rodillos debe cumplir con tres importantes requisitos: (a) debe permitir el control y monitoreo de la temperatura del fluido, ya que, en la mayoría de las soluciones poliméricas, pequeñas variaciones en su temperatura alteran la viscosidad y modifican las características del espectro de relajación; (b) debe permitir mediciones ópticas a lo largo de los rodillos, en la región central entre los mismos, para realizar las medidas en los cambios de la conformación; y (c) debe permitir una amplia variedad de posibles historias de deformación con el objetivo de facilitar el estudio de diversos aspectos de respuesta no lineal en líquidos poliméricos.

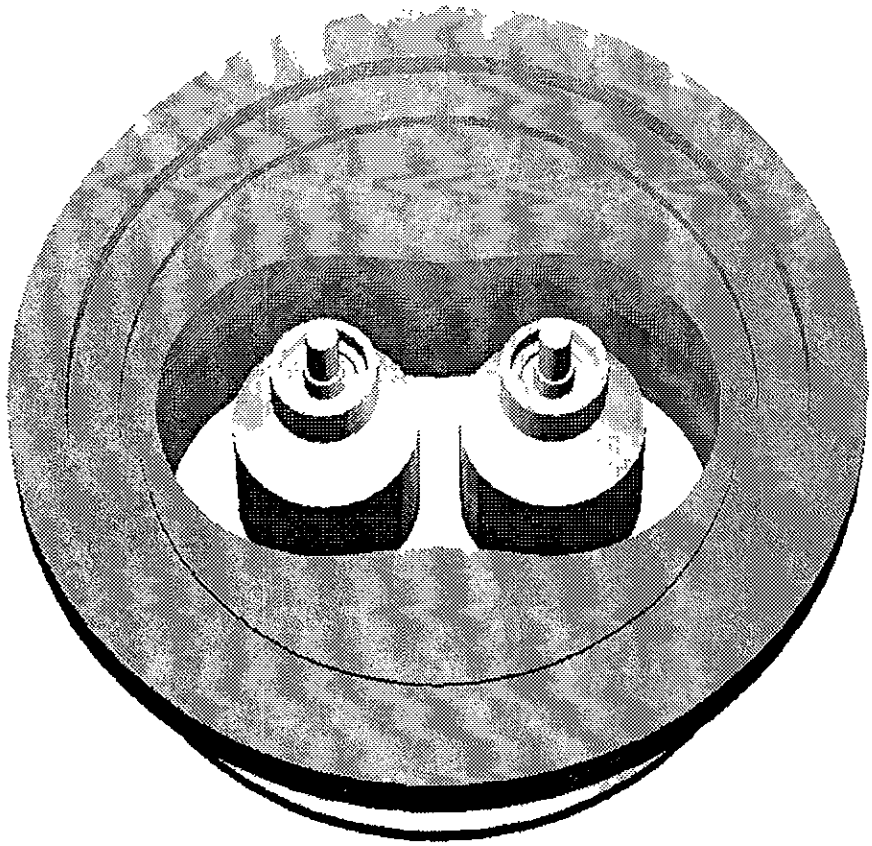


Figura 1.4. *Celda de flujo del arreglo experimental. El contorno en forma de cacahuate es resultado de la optimización del espacio que se logra con la solución analítica de* [7,8].

Los requerimientos para lograr un campo de flujo estable, simétrico, bidimensional, reproducible y exacto dependen del diseño y ensamblado de los rodillos. Estos requerimientos se cubren sólo cuando se garantizan las mismas características físicas (como radio y excentricidad) y una verdadera sincronización en la rotación de ambos rodillos, de tal manera que la evolución temporal de la velocidad de puntos simétricos en el flujo debe ser la misma en cada momento. La necesidad de sincronizar un par de rodillos se resuelve utilizando un sólo motor de pasos para manejar a ambos, con el acoplamiento de un tornillo sin fin – engrane para conectarlos con el motor como se muestra en la Figura 1.5.

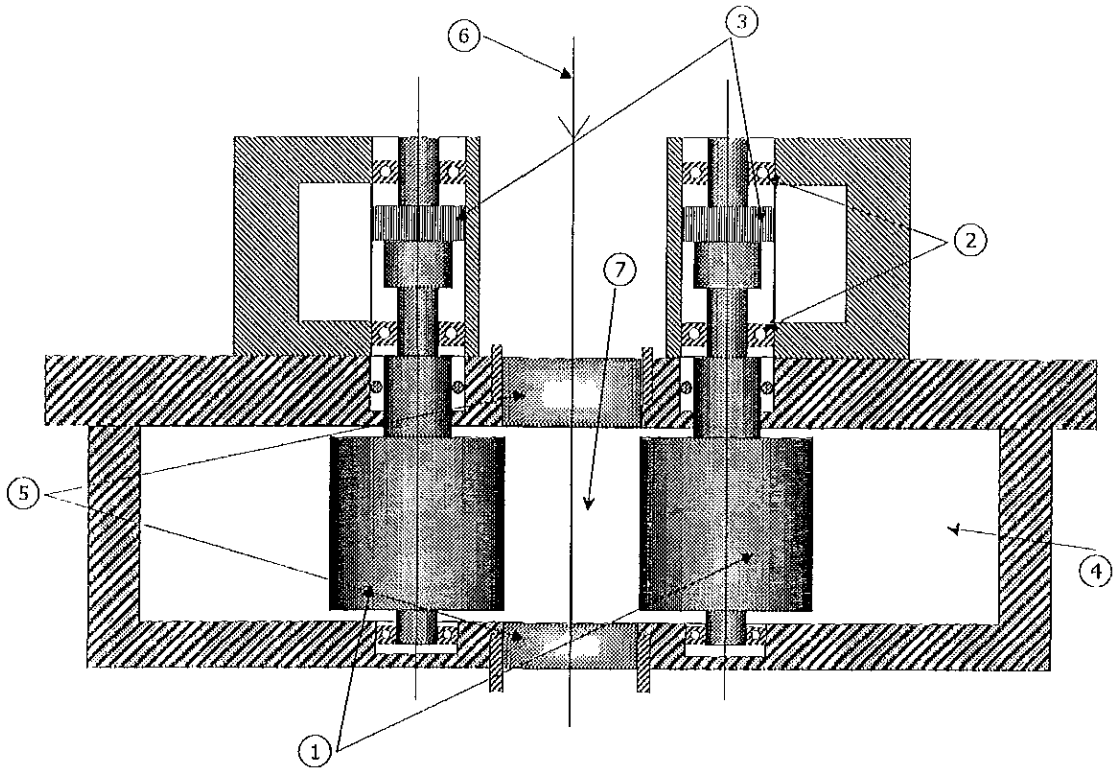


Figura 1.5. *Descripción del molino de dos rodillos: (1) rodillos, (2) rodamientos, (3) engranes, (4) cámara, (5) ventanas, (6) eje óptico, (7) zona de observación. Los rodillos poseen ejes paralelos y giran en la misma dirección, a una misma velocidad. En esta figura no aparecen el motor de pasos, el tornillo sin fin ni el sistema de control de temperatura* [9].

De esta manera, manejando los rodillos con un motor que es capaz de *variar la velocidad de rotación como función del tiempo* en una forma controlada y reproducible, el molino de dos rodillos puede generar una gran variedad de flujos fuertes transitorios en los cuales ahora la velocidad de corte $\dot{\gamma}$ es dependiente del tiempo $\dot{\gamma}(t)$. El motor de pasos utilizado para producir el flujo es controlado por un programa de computadora, de tal forma que se pueden ejecutar fácilmente diferentes historias de deformación. El utilizar flujos transitorios, permite explorar con mayor detalle la *dinámica de relajación* de las soluciones poliméricas cuando éstas se encuentran sujetas a grandes deformaciones. A continuación se presentan los diferentes flujos que se pueden generar con este dispositivo.

1.3. Historias de Deformación

El conocer la respuesta del líquido viscoelástico cuando se somete a una rapidez de deformación constante tiene interés tanto teórico como práctico. Teórico porque se parte de un modelo simple e independiente en el tiempo que frecuentemente permite predecir las propiedades de un material, y práctico, porque tiene aplicabilidad tecnológica ya que los procesos industriales con frecuencia utilizan una velocidad de procesado constante. Por ello, existe la necesidad de la medición de anisotropías inducidas por un flujo con rapidez de deformación constante, al que se denomina *flujo de estado estacionario* (*steady state*). Además del estado estacionario, existen otros tipos de flujos que resultan útiles como estudio de los tiempos de relajación de los materiales, tales como el de *inicio y cese de flujo* (*start-stop flow*) y el *flujo de doble escalón* (*double-step flow*) [3]. La totalidad de estos flujos aportan información útil para el estudio de la dinámica no lineal del material. En este trabajo se codifican las subrutinas específicas para generar los flujos y se muestran detalladamente en el Capítulo 3.

1.3.1. Flujo de Estado Estacionario

En el molino de dos rodillos corrotacionales, se genera un flujo estacionario cuando los rodillos giran a una velocidad angular constante durante un tiempo suficientemente largo, de manera que el fluido alcanza una condición de equilibrio con las fuerzas generadas por el flujo. Razón por la cual, para los líquidos viscoelásticos con tiempos característicos hasta de varios segundos, se requiere mantener el flujo durante al menos unas decenas de segundos.

Debido a que resulta útil conocer la anisotropía inducida en el fluido para toda una serie de valores para la velocidad de deformación, los experimentos del flujo de estado estacionario consisten de una sucesión de mediciones sobre estados estacionarios. Para obtener cada uno de ellos, se aplica en los rodillos una velocidad angular constante. La serie de flujos, comienza con una velocidad lenta, y ésta se incrementa una cantidad predeterminada, después de un periodo de tiempo largo. El procedimiento se repite hasta alcanzar la máxima velocidad (ver Figura 1.6). El número de incrementos en la velocidad que conforman la serie, el valor de la velocidad inicial, el valor de cada incremento y el tiempo de permanencia en cada velocidad constante son los parámetros que el usuario aporta en el programa vía la computadora, previo a la realización del experimento.

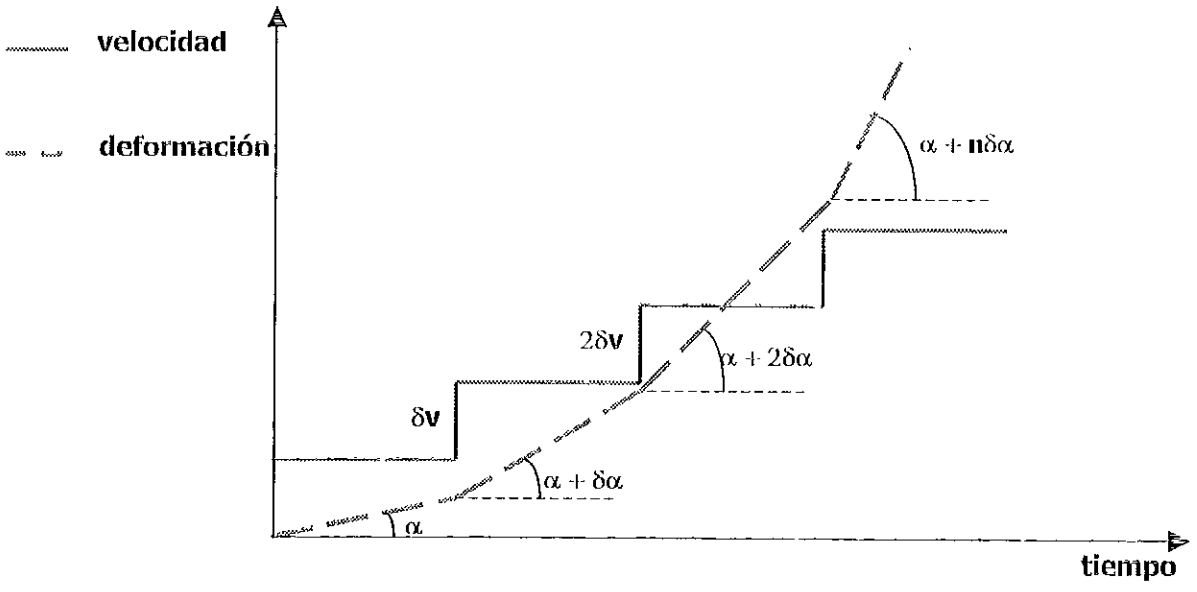


Figura 1.6. *Gráfica del comportamiento de la velocidad y la deformación de un polímero sujeto a un Flujo de Estado Estacionario.*

1.3.2. Inicio y Cese de Flujo

El inicio de flujo consiste en hacer girar los rodillos súbitamente y observar el aumento de la birrefringencia hasta que ésta alcanza un estado estacionario. El cese de flujo es esencialmente la condición opuesta al inicio y dada una birrefringencia en estado estacionario de no equilibrio, los rodillos se detienen abruptamente, lo que ocasiona el decaimiento posterior de la birrefringencia hasta que ésta alcanza un valor cercano a cero, como se ilustra en la Figura 1.7. El inicio y cese son dos de los flujos transitorios más simples que pueden proveer información acerca de las escalas de tiempo del fluido a nivel molecular.

La relajación de la birrefringencia y su ángulo de orientación después del cese de flujo se han considerado desde tiempo atrás como una medida directa de la dinámica interna del fluido. Las escalas de tiempo de relajación dependen únicamente de la conformación inicial del polímero provocada por el estado del flujo anterior. Ello depende entonces, de la razón de cambio de velocidad con respecto a la posición (magnitud del gradiente de velocidad $\dot{\gamma}$) impuesta por el esfuerzo cortante, de la duración del flujo y del tipo de flujo (elongacional o rotacional).

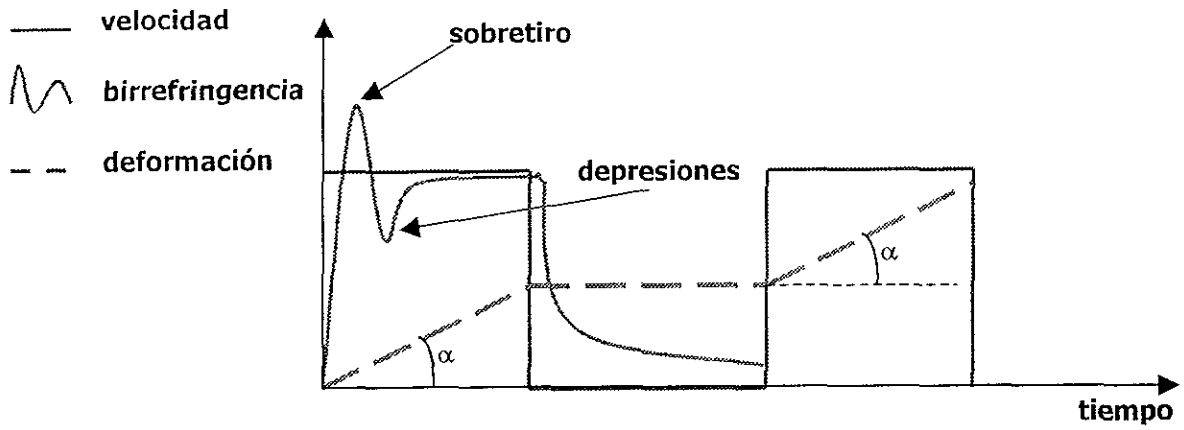


Figura 1.7. *Gráfica del comportamiento de la velocidad, la deformación y la birrefringencia de un polímero sujeto a un proceso de Inicio y Cese de Flujo. Para este flujo, a diferencia de los demás, se puede predecir el comportamiento de la birrefringencia.*

En contraparte, el inicio de flujo ha sido reconocido hasta hace poco tiempo, como otra técnica valiosa para estudiar la microestructura de un fluido. Para estos flujos existen varias características de importancia, dependiendo de las propiedades del comportamiento observado. En particular se han observado sobretiros (del inglés *overshoots*) en la evolución de la anisotropía no lineal (resultado de deformaciones finitas) con un pico amplio antes de alcanzar el valor (menor) del estado estacionario de no equilibrio. Esta característica de sobretiro algunas veces viene acompañada de depresiones (o *undershoots*) antes de alcanzar el estado estacionario. El tiempo necesario para alcanzar los valores máximos de perturbaciones y el tiempo en el que éstas decaen, etc., son escalas de tiempo que también brindan información acerca de la dinámica microestructural, específicamente acerca de la evolución de la conformación (inicial y final) de las moléculas.

Los parámetros que afectan en mayor medida el comportamiento de los flujos de inicio son la aceleración con la que alcanza el estado estacionario, el gradiente de velocidad, el tipo de flujo y el tiempo en el que el fluido se mantuvo en reposo antes de que el campo de flujo se iniciara. Todos estos parámetros pueden ser modificados con el sistema de control adaptado al molino de dos rodillos.

1.3.3. Flujo de Doble Escalón

Por último, el flujo de doble escalón consiste de una combinación de dos arranques de flujo, el segundo a partir de la velocidad de giro de la primera etapa. Este procedimiento juega un importante papel en el estudio de la dinámica de polímeros cuando los estados inicial y final no se encuentran en equilibrio y existen fenómenos no lineales, como histéresis, en la respuesta, etc.

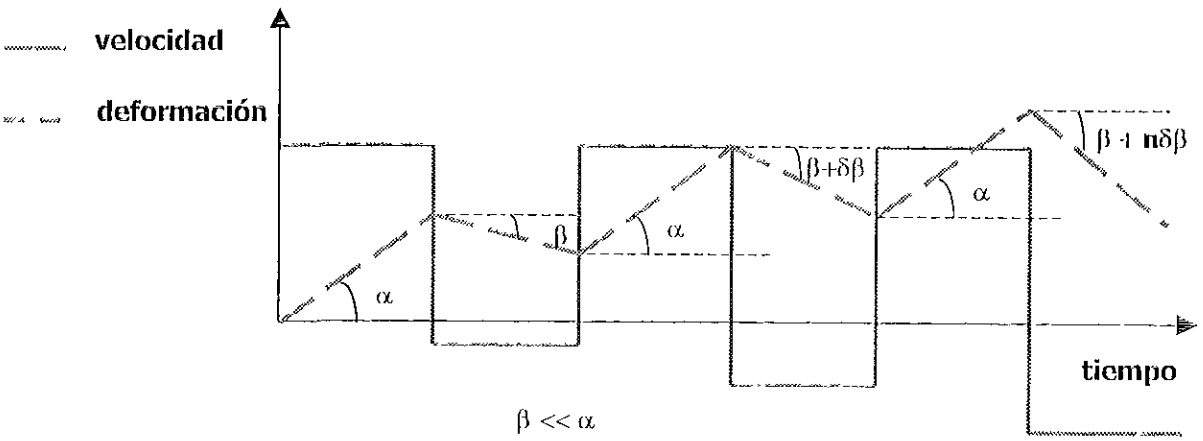


Figura 1.8. *Gráfica del comportamiento de la velocidad y la deformación de un polímero sujeto a un Flujo de Doble Escalón. El comportamiento de la birrefringencia varía dependiendo de las características físicas del polímero, por lo que no es posible establecer una curva representativa de la misma.*

Estos flujos imponen sobre el fluido una razón de corte continua y alternada desde un estado con una razón de corte alta a una razón de corte pequeña, aunque diferente de cero (o viceversa). En estos casos, las escalas de tiempo de la dinámica del polímero, pueden sufrir manifestaciones como la presencia de *perturbaciones* o *depresiones* cuyas características dependen de las razones de corte. La etapa previa del flujo se utiliza para establecer la conformación inicial de la microestructura del polímero. El tiempo en el que el flujo permanece en el primer paso se considera lo suficientemente largo como para que éste alcance un estado estacionario de no equilibrio, de tal manera que el campo de flujo sea altamente reproducible. Entonces, con este tipo de flujos transitorios, los cambios de los fenómenos observados se sitúan únicamente en función de una alta y baja razón de cortante, aunque bajo circunstancias particulares también es importante la longitud del tiempo en el que éstas se llevan a cabo. El dispositivo de flujo que se utiliza en este experimento, es capaz de variar todos estos parámetros.

CAPÍTULO 2

Sistema de Control para el Motor del Molino

2.1. Arreglo de Control

Un sistema de control puede definirse como una interconexión de varios componentes que trabajan juntos para desarrollar una función específica, que en la mayoría de los casos se refiere al control de variables físicas, las cuales pueden ser termodinámicas y cinemáticas. Para el control de las variables cinemáticas, son los motores los que se utilizan para operar un sinnúmero de aplicaciones, y el motor de pasos en específico ha sido el móvil básico de una gran cantidad de sistemas de control de movimiento de precisión (Apéndice A) [11].

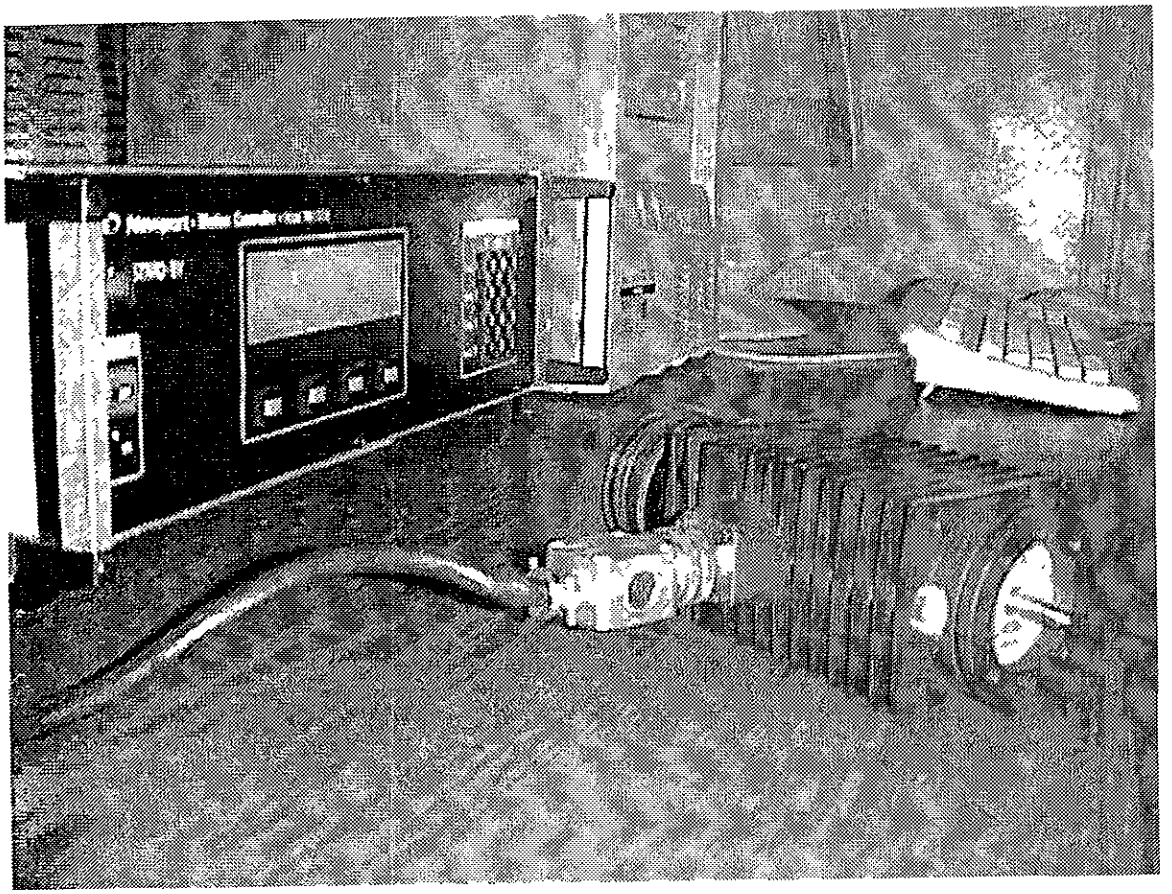


Figura 2.1. *Fotografía del motor (al frente) conectado al controlador MM4000 (izquierda) y la computadora (al fondo). El frente del motor muestra la flecha, y en la parte posterior se encuentra el tambor y el codificador del motor.*

El sistema de control de este proyecto genera el movimiento de un molino de dos rodillos que se emplea en el experimento de Birrefringencia Bicolor, el cual se presenta con detalle en el Capítulo 1. De esta manera, para que el molino de dos rodillos pueda generar diferentes historias de deformación en un fluido polimérico, se requiere de un sistema capaz de manipular, en forma controlada, el comportamiento del flujo polimérico por medio del molino.

El objetivo del proyecto es entonces desarrollar este sistema de control preciso que involucra recursos computacionales, electrónicos y mecánicos para llevar a cabo una parte de dicho experimento ya que requiere del movimiento angular de un molino de dos rodillos por el que pasa un fluido polimérico para ser deformado.

De esta manera, el trabajo consiste en dos etapas principales. Una, realizar los programas computacionales para generar los datos que el experimento requiere, a partir de diferentes parámetros que son introducidos por el usuario. La otra parte se refiere a lograr una correcta comunicación entre los dispositivos con los que cuenta el laboratorio, estableciendo un sistema de control preciso que funcione al servicio de la técnica que utiliza el experimento, para analizar propiedades de fluidos poliméricos en circunstancias que serán expuestas posteriormente.

El sistema de control, consiste de un equipo de cómputo, una interfaz de comunicación, un controlador de movimiento y un motor de pasos (ver Figura 2.1). Por medio de la computadora se realiza el programa utilizando el lenguaje de programación gráfico HPVEE que calcula y estandariza los datos que él mismo envía al controlador de movimiento MM4000, vía una interfaz de comunicación por el puerto serial, para que el controlador regule la acción del motor de pasos UE73PP.

2.2. Controlador MM4000 para Motor de Pasos

El *MM4000* es un avanzado controlador con capacidad para manejar 4 ejes de movimiento en forma simultánea y combinar motores de CD y motores de pasos. Soporta operaciones de lazo cerrado para motores de pasos y junto con el algoritmo de *alimentación adelantada* (que se explica en la Sección 2.2.1) es capaz de producir un movimiento suave y preciso.

Este controlador cuenta con dos diferentes modos de operación. El *modo local* que se realiza a través del teclado en el panel frontal del controlador y que permite seleccionar

el menú y las operaciones que pueden desarrollarse sin el uso de una computadora u otro dispositivo externo. La realización de estas operaciones depende de si los motores se encuentran encendidos o apagados y una actualización de la configuración general del controlador no se puede realizar si el motor se encuentra en movimiento. En cambio, si la operación es en *modo remoto*, el controlador debe conectarse por medio de una interfaz a la computadora, ya sea por una RS-232 o una IEEE-488. En este modo, todos los comandos son recibidos por vía remota y el controlador los ejecuta instantáneamente.

El MM4000 cuenta con más de 129 comandos posibles de utilizar, que se pueden enviar por estas vías para que se ejecuten en caso de que el motor se encuentre encendido [12,13].

2.2.1. Lazos de Control

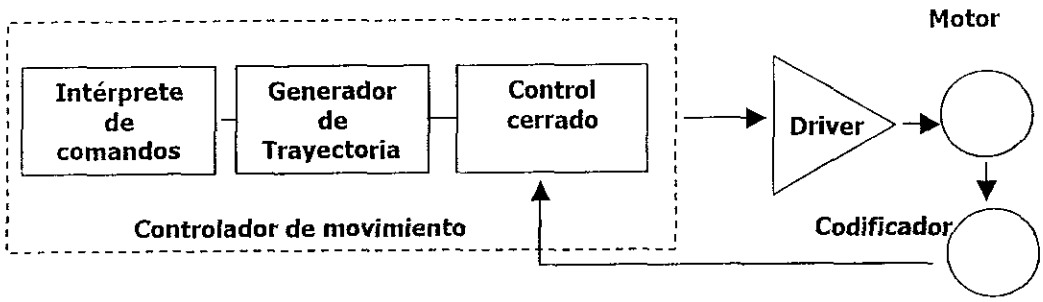


Figura 2.2. *Diagrama de un lazo cerrado.*

Cuando se habla de sistemas de control de movimiento, una de los aspectos más importantes es el tipo de lazo que se vaya a utilizar. La primera distinción es entre lazo cerrado y lazo abierto. El diagrama básico de un lazo cerrado se muestra en la Figura 2.2, junto con el intérprete de comandos. Las dos partes principales del controlador de movimiento son el generador de trayectoria y su controlador. El primero genera la trayectoria deseada y el segundo controla al motor siguiendo a ésta lo más cerca posible. El MM4000 utiliza un lazo cerrado PID con alimentación hacia adelante (*feedforward*) de velocidad tanto para motores de CD como para motores de pasos, conceptos que se explican a continuación.

Lazo PID

El término PID proviene de los factores de *ganancia proporcional*, *integral* y *derivativo*, los cuales son las bases del cálculo de lazos de control. La ecuación común que describe al lazo PID, está dada por:

$$s = K_p \bullet e + K_i \int e dt + K_d \bullet \frac{de}{dt} ;$$

donde s es la señal de control, K_p es el factor de ganancia proporcional, K_i es el factor de ganancia integrador, K_d es el factor de ganancia derivativo y e es el error (ver Apéndice A). El problema más común es la interpretación de esta fórmula, especialmente cuando se trata de modificar los valores de los tres factores de ganancia para obtener una respuesta específica del sistema, tarea que se dificulta si no se comprende su comportamiento. A continuación se presenta la explicación de los componentes del PID y sus operaciones.

Lazo P

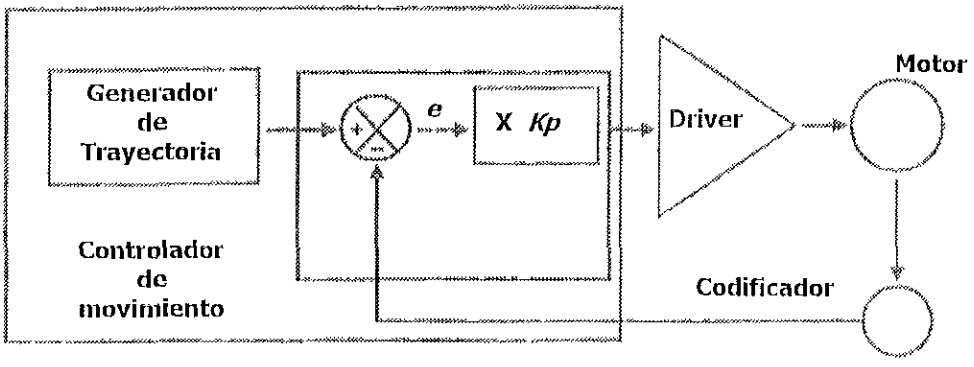


Figura 2.3. *Lazo P compuesto de los elementos de un lazo cerrado más el elemento del factor de ganancia proporcional K_p.*

El lazo cerrado más simple es el proporcional P y su diagrama se presenta en la Figura 2.3. En cada ciclo, la posición actual como la reporta el codificador, se compara con la posición establecida por el generador de trayectoria. La diferencia e es el error de posición. Amplificándolo al multiplicar por K_p, se genera una señal de control *proporcional* al error que, convertida a una señal analógica, se envía al *driver*[1] del motor.

[1] Para el controlador MM4000 de Newport, el *driver* es una tarjeta electrónica, que funciona como un circuito de potencia en el caso de los motores de pasos.

Lazo PI

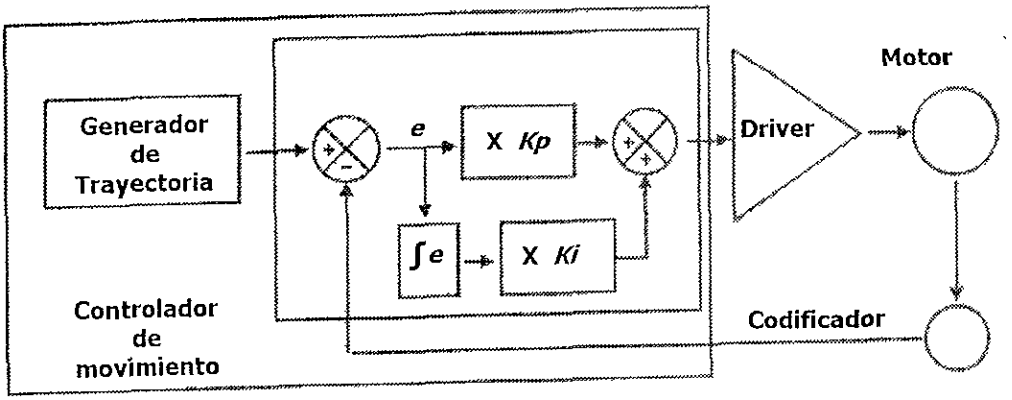


Figura 2.4. *Lazo PI compuesto por los factores de ganancia K_p y K_i así como de los elementos del lazo cerrado.*

Para eliminar el error que se produce al suspender el movimiento y durante movimientos prolongados a velocidad constante, usualmente llamado error de *estado estable*, un término *integrador* se agrega al lazo. Este término realiza la operación de integración al valor del error producido en cada ciclo, que multiplicado por el factor de ganancia K_i se suma a la señal de control. El resultado es que el término que involucra la operación de integración, se incrementa hasta reducir el error a cero. Durante un movimiento largo a velocidad constante también convierte el error a cero, una característica que puede resultar importante para algunas aplicaciones. Al detenerse, el error de posición se reduce a su mínimo valor, teniendo así el efecto deseado.

Lazo PID

El tercer término del lazo PID es el *derivativo*. Se define como la diferencia entre el error del ciclo actual y el del anterior. Si el error no cambia, entonces el término derivativo es cero, de tal forma que su función es verificar que efectivamente la señal de control corrija el error de cada ciclo en vez de incrementarlo, previniendo así, oscilaciones y cambios bruscos de posición. La Figura 2.5 muestra el diagrama del lazo PID. El término *derivativo* se suma a los términos *proporcional* e *integrador* para formar el sistema de control PID, siendo éste uno de los algoritmos de mayor uso en los sistemas de movimiento. Cuando se produce un error o una diferencia entre la posición actual y la deseada en un sistema de movimiento, se utiliza el lazo PID para corregir la señal de

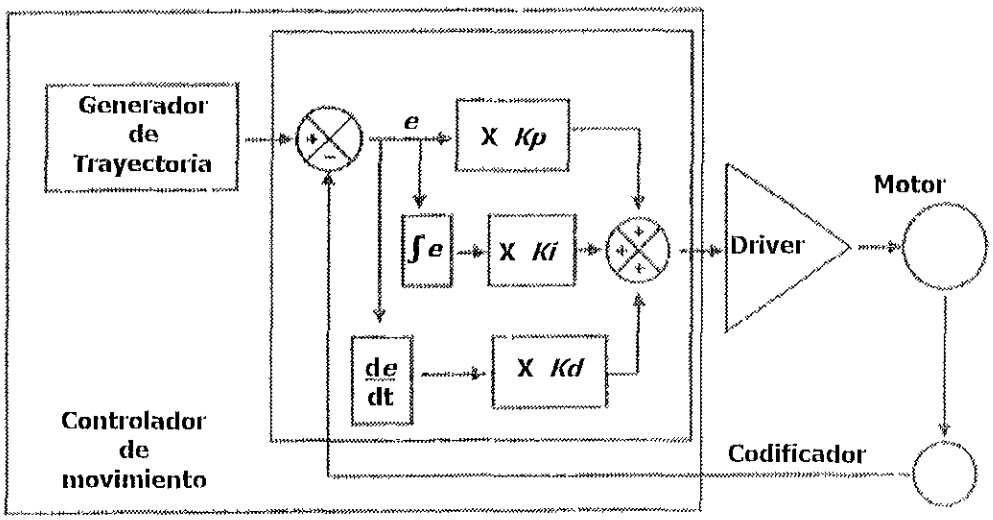


Figura 2.5. *Lazo PID compuesto por los factores de ganancia K_p, K_i y K_d así como de los elementos del lazo cerrado.*

entrada en cada ciclo y anular el error. Sin embargo, en la realidad, el valor del error nunca es cero [11,14].

Lazo de Alimentación Adelantada

El esquema de alimentación *adelantada* (del inglés *feed forward*), como una solución para sistemas que presentan tiempos de retraso, ha sido desarrollado para superar los efectos desestabilizadores que se ocasionan en un control de retroalimentación ya sea por estos tiempos de retraso o por oscilaciones del sistema mismo [15,16].

El objetivo del lazo de alimentación adelantada es entonces minimizar el valor del error, ya que predice el funcionamiento futuro del sistema y lleva a cabo correcciones basadas en estas estimaciones. Este lazo funciona a partir de que el generador de trayectoria calcula la velocidad deseada, por lo que su valor se conoce con anterioridad. Esta información se multiplica por el factor de escala *Kvff* y el resultado es el que alimenta al *driver* del motor. Si el escalamiento fue correcto, el valor de la señal se envía al motor, para lograr de esta manera, la velocidad que se requiere sin necesidad de un lazo cerrado. Este procedimiento conocido como velocidad de alimentación hacia adelante, es más bien considerado como un lazo abierto. Sin embargo si se adhiere esta señal al lazo cerrado, reduce de manera significativa el trabajo del PID para disminuir todo el error. El PID en

realidad, únicamente corrige el error residual que deja la señal de lazo de alimentación adelantado.

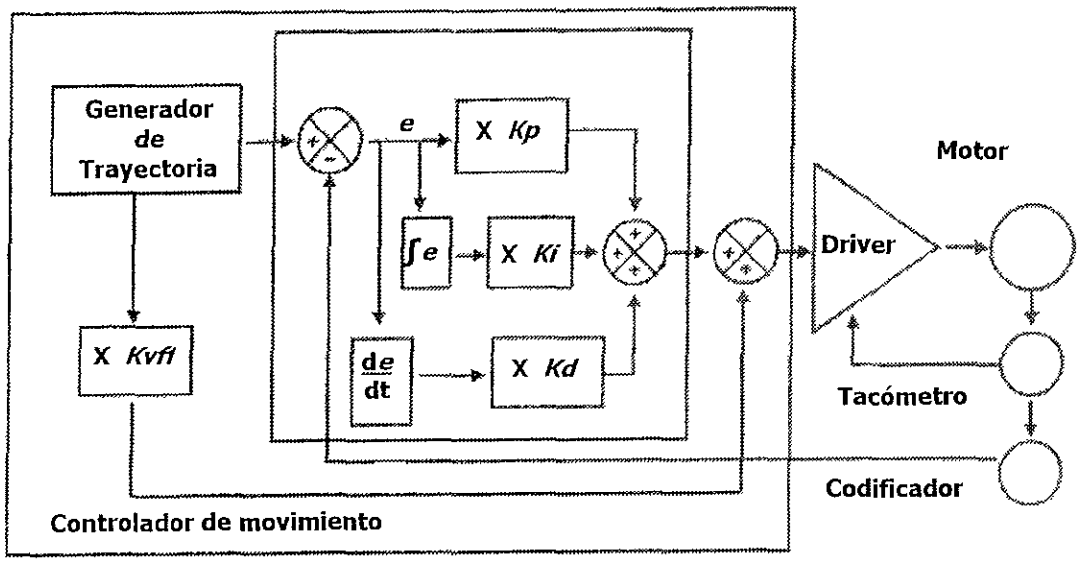


Figura 2.6. *Lazo PIDF con un* driver *basado en un tacómetro.*

Una característica especial del controlador MM4000 es que el *driver* no es un simple amplificador de voltaje y/o de corriente, sino que cuenta además con su propio lazo retroalimentado desde un tacómetro. El tacómetro es un dispositivo que produce una señal de voltaje proporcional a la velocidad. Usando esta señal, el *driver* puede mantener la velocidad proporcional a la señal de control. Sin embargo, los tacómetros resultan muy costosos y muchas veces poco prácticos, así que el *driver* tiene un circuito especial adaptado para cada tipo de motor, que es capaz de calcular la velocidad de manera individual.

El resultado del uso del circuito especial es prácticamente igual al del tacómetro de retroalimentación pero a una fracción de su costo. El inconveniente es que cada motor necesita una tarjeta especial, por lo que cada tarjeta del MM4000 está diseñada para trabajar en un sistema predefinido utilizando motores conocidos. Esta operación es totalmente transparente al usuario, pues todas las tarjetas están previamente diseñadas para una serie de motores conocidos, tanto de CD como de pasos, por lo que no se requieren ni se permiten ajustes.

El controlador MM4000 es capaz de definir los valores de las ganancias para cada tipo de motor y proporcionar al usuario un valor por default para cada uno. Para el control del motor de pasos que se utiliza en este trabajo, el controlador determina que el término integrador produce un severo efecto desestabilizador en el lazo cerrado, por lo que el valor de K_i resulta ser igual a cero, como se muestra en la tabla de configuración de los parámetros del eje 4 (ver Tabla 2.1) que se presenta posteriormente.

2.2.2. Perfil de Movimiento

Para poder obtener un movimiento suave del motor a altas velocidades, es necesario que el controlador cambie de velocidad de una manera prudente para alcanzar óptimos resultados. Esto se logra utilizando perfiles de velocidad que limiten las aceleraciones y las desaceleraciones.

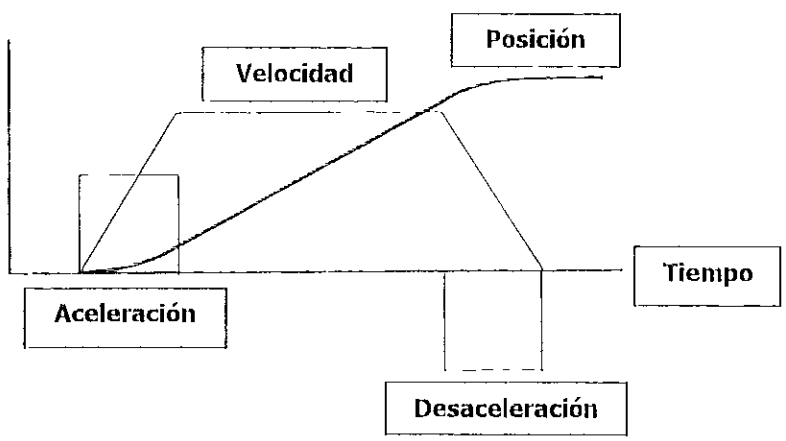


Figura 2.7. *Perfiles de posición y velocidad. El perfil trapezoidal corresponde a la evolución de la velocidad con dos regiones de aceleración constante (positiva y negativa). El desplazamiento corresponde al área bajo el trapecio: curva de posición.*

Cuando se ejecuta un comando de movimiento, el dispositivo acelera hasta alcanzar una velocidad preestablecida. Posteriormente, comenzará a desacelerarse hasta que el motor se haya detenido en la posición correcta. La gráfica de velocidad de este tipo de movimiento tiene una forma trapezoidal; por esta razón, a este tipo de movimientos se le conoce como movimiento trapezoidal. Los perfiles de posición y de aceleración relativos a la velocidad se muestran en la Figura 2.7.

Además de la posición final, es posible especificar la aceleración y la velocidad antes de cada comando de movimiento. Por ello, se considera al MM4000 como un controlador

avanzado, ya que permite al usuario definir estos valores durante el movimiento y verifica que el parámetro a cambiar esté dentro de los límites seguros permitidos. De no ser así, se ignora el comando y el movimiento continúa como fue establecido inicialmente. Existen otros tipos de perfiles de velocidad, pero el trapezoidal es el óptimo para un posicionamiento de punto por punto.

2.2.3. Configuración del Controlador

El Controlador MM4000 se compone de dos modos principales de operación, el local y el remoto como se menciona en la Sección 2.2. En modo local el controlador es operado a través del teclado del panel frontal y en modo remoto es necesario conectar el controlador a una computadora por medio de alguna de sus interfaces (RS-232 ó IEEE-488).

CONFIGURACIÓN GENERAL	
Language	English
Controller	Standard
Speed Scaling	100 %
Comm. Timeout	5.00 [s]
Home Timeout	90.00 [s]
Terminator	LF
Communication	RS-232
IEEE Address	2
IEEE SRQ Used	NO
Baude Rate	9600
Parity	None
Word Length	8
Stop Bits	1

Tabla 2.1. *Configuración General del Controlador.*

Para la utilización del Controlador MM4000, como para cualquier otro dispositivo, es necesario configurarlo previamente. Esto se hace por medio del modo local, aunque algunos de los parámetros pueden ser cambiados en el modo remoto. La Tabla 2.1 muestra la configuración general que se establece al controlador para el caso específico de este trabajo. Este menú de configuración hace posible la comunicación con la

computadora. Es necesario que los parámetros que se le asignan al controlador, también se asignen cuando se configura el interfaz por medio del lenguaje de programación.

La Tabla 2.2 muestra la configuración de los parámetros para el eje 4 del controlador asignado al motor de pasos UE73PP. Los valores de los parámetros son asignados por el controlador y dependen directamente del tipo de motor. Algunas de las definiciones de los parámetros que se presentan en esta sección se encuentran en el Apéndice A.

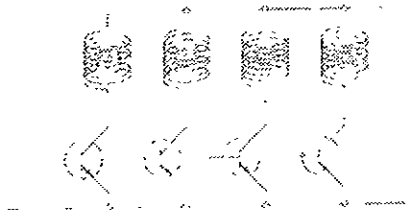
MODIFYING AXIS PARAMETERS			
Units	Inc	**Acceleration**	80 000 [Inc/s]
Motion Type	Real	**Min Position**	-75 000 [Inc]
Home Type	Simulated	**Max. Position**	75 000 [Inc]
Motor Type	Stepper	**Home Preset**	0
Control Loop	Closed	**Kp**	0.015
Periodicity	NO	**Ki**	0
Motor Increment	0.001 [mm]	**Kd**	0.0001
Encoder Resolution	0.001 [mm]	**Ks**	1
Scaling Speed	24 000 [Inc/s]	**Maximum Error**	1 000 [Inc]
Maximum Speed	20 000 [Inc/s]	**Backlash**	0 [Inc]
Manual Speed	10 000 [Inc/s]	**Dsply. Resol**	0
Home Speed	10 000 [Inc/s]		

Tabla 2.2. *Configuración de los parámetros del eje 4 designado al motor de pasos UE73PP.*

2.2.4. Driver

El controlador incluye al *driver* como uno de sus componentes. La parte de control es común para cualquier configuración, sin embargo, el *driver* corresponde al hardware electrónico que optimiza la operación del motor de acuerdo a sus necesidades de potencia y características electromecánicas. El controlador traduce los requerimientos recibidos mediante comandos y optimiza las secuencias básicas para que cualquier motor alcance su objetivo. Es por lo anterior que el controlador consiste de tantos *drivers* como motores tiene el sistema y cada *driver* se conecta al respectivo motor mediante un cable y conector de 25 líneas (pines). El contar con un dispositivo con controlador y *driver* integrados, no

solamente ofrece ventajas de optimización de espacio y costo, sino que también ofrece una mejor coordinación entre las dos unidades, ya que facilita que el controlador pueda



monitorear y controlar las operaciones efectuadas por el *driver*. A continuación se analiza las funciones del mismo.

Figura 2.8. *Driver simple para un motor de pasos.*

Para manejar un motor de cuatro fases, únicamente se requieren cuatro interruptores (transistores) controlados directamente por un CPU (ver Figura 2.8). Sin embargo, cuando se requiere de altas velocidades se presentan inconvenientes como los que se presentan enseguida.

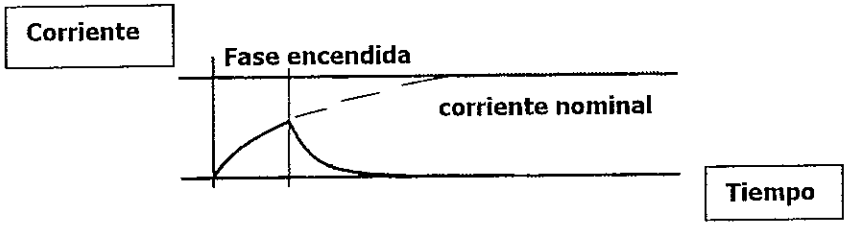


Figura 2.9. *Efecto en la corriente provocado por un tiempo corto.*

Cuando el voltaje es aplicado a una bobina del motor, la corriente (y por lo tanto el torque) se aproxima al valor nominal en forma exponencial. Si la velocidad de pulso es rápida, la corriente no cuenta con el tiempo suficiente para alcanzar el valor deseado antes de que el dispositivo se apague, por lo que el torque total adquiere un valor menor igual a una fracción del nominal como se observa en la Figura 2.9.

La rapidez con que la corriente alcanza su valor nominal depende de tres factores: de la inductancia y la resistencia de las bobinas del motor, y del voltaje aplicado. La inductancia y la resistencia son fijas pues dependen de las características electromecánicas del motor, pero el voltaje puede ser temporalmente incrementado para que la corriente alcance rápidamente el nivel deseado. La técnica más comúnmente utilizada para lograr

este efecto, es la del *Interruptor de Alto Voltaje*. Así, si por ejemplo un motor de pasos requiere de 3V para alcanzar la corriente nominal, se conecta momentáneamente a un potencial de 30V. De esta manera alcanzará la misma corriente en aproximadamente 1/10 del tiempo. Una vez que se obtiene el valor necesario de corriente, se activa un circuito limitante que mantiene un valor de la corriente cercano al valor nominal; Figura 2.10.

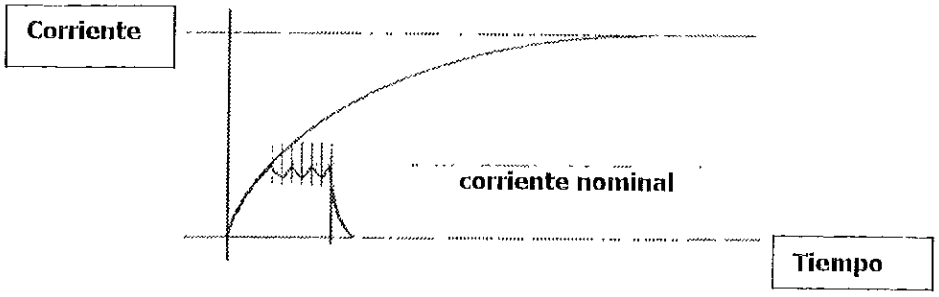


Figura 2.10. *Pulso del Motor con el Interruptor de Alto Voltaje. El torque máximo alcanzado es proporcional a la corriente nominal.*

El controlador emplea dos diferentes circuitos para dos familias de motores de pasos. La tarjeta MM78PP es la que está diseñada para manejar cuatro motores de pasos magneto permanentes.

2.3. Motor de Pasos de Cuatro Fases

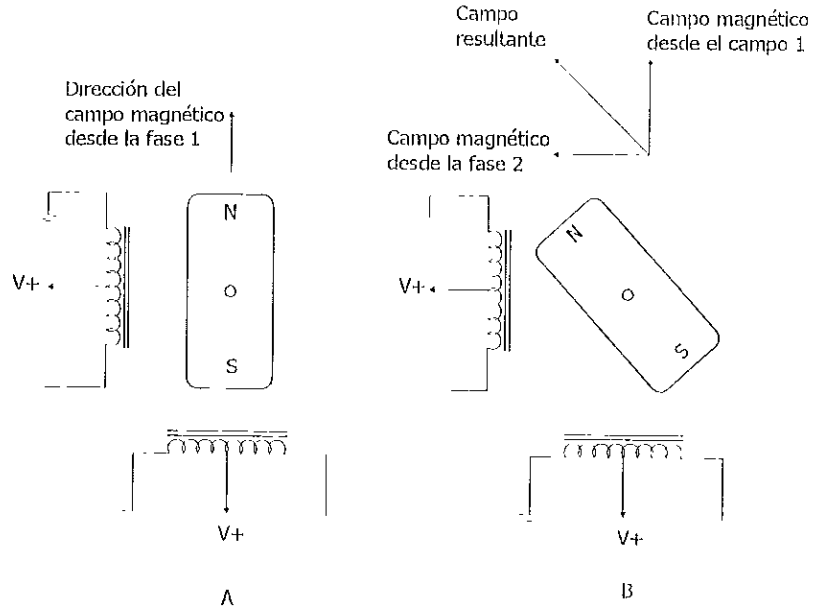


Figura 2.11. *Representación simplificada de un motor de pasos de 4 fases. Cuando Φ1 es energizada, el rotor se alinea como se muestra en A. Cuando Φ1 y Φ2 son energizadas, el rotor se mueve a la posición que se muestra en B.*

La característica principal de un motor de pasos es que cada ciclo de movimiento tiene un número de posiciones estables. Esto significa que si se aplica corriente a una de las bobinas o fases, el rotor tratará de encontrar uno de estos puntos estables para quedarse en él. Para generar una velocidad de rotación, una secuencia en el tiempo entre estados estables (o posiciones estables) se debe establecer. Esto es, dado un impulso a una posición estable, a continuación otra fase se debe energizar para que el motor avance al nuevo punto estable, logrando un pequeño giro y así sucesivamente. Sin embargo, si el rotor es un magneto permanente, dependiendo de la polarización de la corriente, el estator moverá al diente del rotor en un sentido o en el otro. Esta es la mayor distinción entre dos diferentes tecnologías en motores de pasos: los motores de *reluctancia variable* y los *magneto permanentes*. Los de reluctancia variable son usualmente pequeños, de bajo costo y con un ángulo de paso grande. La tecnología de magneto permanentes se utiliza para motores grandes o de mayor precisión, como lo requerido para este proyecto.

La Figura 2.11 muestra un modelo conceptual de un motor de pasos de 4 fases típico, donde el rotor es un magneto permanente que gira al aplicársele un campo magnético. Energizando una bobina del estator a la vez, el rotor puede orientarse hacia cualquiera de las cuatro posiciones, cada una a 90° (Figura 2.11.A). Energizando 2 fases adyacentes de manera simultánea, el rotor puede ser orientado en otras 4 posiciones, a la mitad de las primeras cuatro (Figura 2.11.B). Alternando entre los dos esquemas, es posible obtener el doble de pasos por revolución. Este es conocido como el modo de "medio-paso".

En la práctica, la mayoría de los motores de pasos que se utilizan en dispositivos ópticos están diseñados para girar con desplazamientos angulares de 1.8º por paso. Cuando se requieren grandes torques y bajas velocidades entonces se acopla una caja reductora y el número de pasos por rotación del engrane motor es igual al número de pasos del motor multiplicado por la relación de reducción.

2.3.1. Características de la Serie UE7_PP de Motores de Pasos

El motor específico que se utiliza en este dispositivo de control de movimiento es el UE73PP de la serie UE7_PP de motores de pasos Newport, los cuales son motores magneto permanentes de 4 fases cuyas características generales se presentan en la Tabla 2.3.

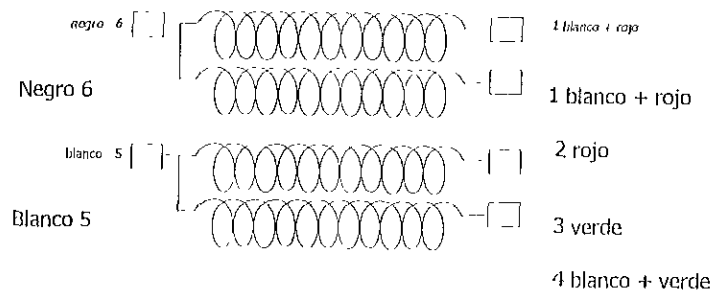


Figura 2.12. *Embobinado y salidas.*

Para producir una rotación dextrorsa de la flecha del motor (en sentido horario) el orden de activación de los *switches* asociados a cada fase debe ser 1-3-2-4, como se muestra en la Figura 2.12. Esta serie UE7_PP, cuenta con dos componentes adicionales. Una cabeza de engrane con codificador incremental y un tambor graduado con 2 x 100 divisiones que provee una indicación visual de la operación del motor, ambos elementos se observan en la fotografía de la Figura 2.1. La posición de la marca puede ser ajustada desde la salida de la flecha. El codificador genera 10 incrementos por cada paso del motor para operaciones de minipaso o un incremento por paso para operaciones de paso completo. Para efectos de este trabajo, siempre se manejan minipasos.

Características		UE73PP
Inductancia por fase	mH	2.9
Resistencia por fase	Ω	1.16
Corriente nominal	A	2.9
Potencia mecánica	W	60
Número de pasos/rev		200
Torque		
◎ Sin engrane	Ncm	50
◎ Con engrane		
1/3	Ncm	130
1/5	Ncm	210
> 1/5	Ncm	300
Máxima velocidad	pasos/s	4000
Máxima aceleración	pasos/s^2	8000

Tabla 2.3. *Características del motor de pasos UE73PP.*

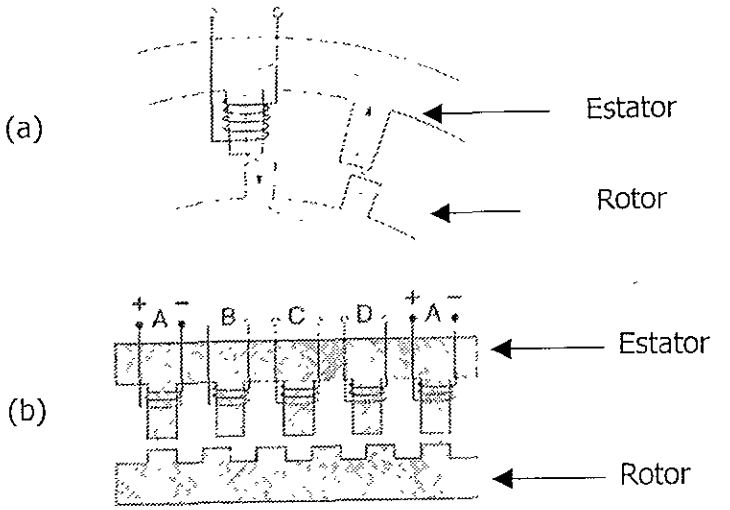


Figura 2.13. *Operación de un motor de pasos.*

2.3.2. Funcionamiento del Motor UE73PP

Con el propósito de optimizar los desplazamientos de manera que las velocidades de giro sean lo más suaves y continuas posibles, es conveniente modificar la señal que alimenta a los embobinados del motor. Esto es, el motor genera el movimiento avanzando a una nueva posición estable por medio de varias fases del estator que tiene los dientes ligeramente separados unos de otros. La Figura 2.13 muestra un motor de pasos con cuatro fases dibujado en forma lineal para facilitar su seguimiento.

Las cuatro fases, desde la A a la D, se energizan una a la vez (la fase A se muestra 2 veces). Los dientes del rotor se alinean con la primera fase energizada. Si se apaga la corriente de la fase A y después se energiza la fase B, el diente del rotor más cercano y a la izquierda de B será atraído por éste y el motor se moverá un paso de 1.8°. En cambio, si después de energizada A se energiza la fase D, el diente del rotor más cercano está en la posición opuesta, lo que provocará que el motor se mueva en sentido contrario. La fase C no debe energizarse inmediatamente después de A porque se encuentra exactamente entre dos dientes, ya que su aplicación provocaría la indeterminación en la dirección del movimiento. Para lograr el movimiento en una dirección, la corriente en las cuatro fases debe tener el diagrama de la Figura 2.14.

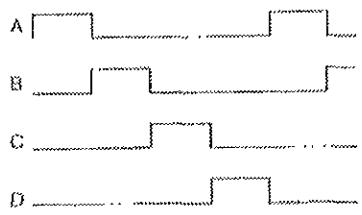


Figura 2.14. *Diagrama de fases.*

Para avanzar un diente completo del rotor se necesita realizar un ciclo completo de 4 pasos. Con este procedimiento se logra lo que se conoce como un paso completo (*fullstep*), que es el incremento de movimiento más grande que puede realizar el motor de pasos. Para lograr un giro completo del rotor se necesita un número de pasos igual a cuatro veces el número de dientes del rotor. Para nuestro motor esto da como resultado 50 dientes.

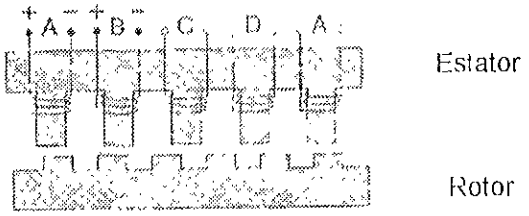


Figura 2.15. *Dos fases energizadas en forma simultánea.*

Si dos fases vecinas se energizan simultáneamente, como se muestra en la Figura 2.15 ambas fases atraen por igual y el rotor encuentra una posición estable justo a la mitad del paso completo. Si simultáneamente se energizan dos fases a la vez, el motor continuará realizando el paso completo. En cambio, si se activan una y dos fases alternadamente el motor avanzará en medios pasos. La ventaja de esta secuencia de activación es que permite giros con el doble de resolución angular con un ligero aumento en la complejidad del sistema de control. El diagrama de sincronización para medio paso se muestra en la Figura 2.16.

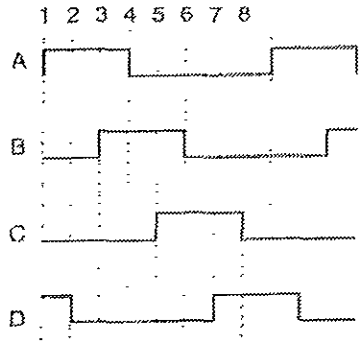


Figura 2.16. *Diagrama de fases para medio paso.*

Otra posibilidad de giro factible en este tipo de motores se logra energizando dos fases simultáneamente, pero con diferentes corrientes. Por ejemplo, se le aplica a la fase A la corriente completa y a la fase B sólo la mitad. Esto significa que la fase A atrae al diente del rotor dos veces más fuerte que lo que lo hace B, por lo tanto, el diente del rotor se detendrá más cerca de A, en algún lugar entre el paso completo y el medio paso (Figura 2.17).

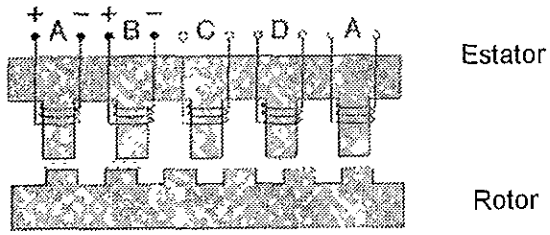


Figura 2.17. *Dos fases energizadas con diferentes intensidades.*

La conclusión es que, variando la relación entre las corrientes de ambas fases, es posible posicionar al rotor en cualquier parte entre dos pasos completos. Para lograrlo, el motor necesita recibir señales analógicas, similares a las que se muestran en la Figura 2.18. Con este método se pueden lograr pasos muy pequeños o mini-pasos. Por cada comando de paso, el motor se moverá únicamente una fracción del paso completo. A medida que se reducen los pasos del movimiento, se incrementa la resolución de éste y se disminuye el movimiento de rizo o ruido.

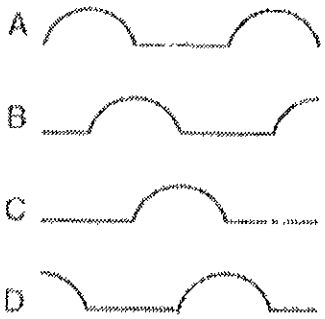


Figura 2.18. *Diagrama para un movimiento continuo idealizado.*

2.3.3. Solución al Error de Posicionamiento

El *driver* del MM4000 usa la técnica de mini-pasos para dividir el paso completo en 10 mini-pasos, incrementando así la resolución del motor por un factor de 10. Sin embargo, los mini-pasos tienen un costo. Primero, el driver electrónico debe generar señales de corriente significativamente más complejas. Segundo, el torque de sujeción para un paso completo se reduce por el factor de mini-paso. Es decir, que si tenemos un cierto valor de torque, al utilizar mini-pasos de 1/10, el torque de sujeción se reduce a 1/10 de su valor total, introduciendo un error de posicionamiento en el motor equivalente a un mini-paso.

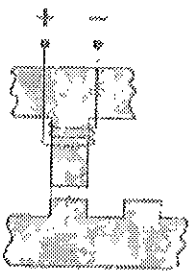


Figura 2.19 *Fase simple energizada.*

Cuando una fase atrae un diente del rotor, asumiendo que no se presentan cargas externas, el motor no desarrolla ningún torque, manteniéndose en una posición estable como se muestra en la Figura 2.19. En cambio, si se presenta un torque externo, el rotor intentará girar (Figura 2.20.A), oponiéndose al flujo magnético, de manera que se presenta

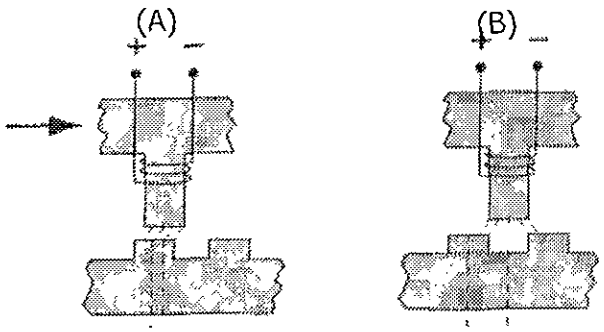


Figura 2.20. *(A) Aplicación de una fuerza externa. (B) Posición inestable.*

un desalineamiento entre los dientes. Este desalineamiento aumenta hasta que el torque alcanza su valor máximo, comienza a disminuir y cuando el estator está situado exactamente entre los dientes del rotor, el torque se vuelve cero (Figura 2.20.B). Esta posición corresponde a un punto inestable y a una desalineación inducida por fuerzas externas que causa que la posición angular del motor brinque de un diente a otro. Esto se conoce como pérdida de pasos y es una de las características intrínsecas del dispositivo que se consideran como una de sus mayores desventajas. El diagrama del torque en función de la posición relativa de dientes estator-rotor se muestra en la Figura 2.21, el cual es el mismo que para los casos de medio-paso y mini-paso. El máximo torque se volverá a presentar después de un paso completo a partir de la posición estable.

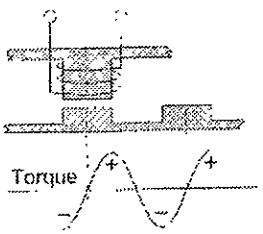


Figura 2.21. *Torque y alineación de dientes.*

Cuando los motores funcionan en mini-pasos o en medios-pasos en sistemas de lazo abierto, la pérdida de pasos se traduce en un error inherente, sin embargo, sistemas de control avanzados como con el que cuenta el MM4000 pueden controlar motores de pasos, como se presentó en la Sección 0, con operaciones de lazo cerrado justamente para eliminar este inconveniente.

2.3.4. Análisis de Motores

Para algunas aplicaciones se requiere realizar una selección entre servomotores y motores de pasos. Los dos tipos de motores ofrecen oportunidades similares para lograr un posicionamiento de precisión, sin embargo difieren en varios aspectos [11,17].

- **Servomotores de Corriente Directa.** Estos motores ofrecen ciertas ventajas sobre los servomotores de CA, como su alta confiabilidad, tamaño pequeño y bajo costo. Un típico servomotor de CD dispone de un rango de potencia que va desde fracciones de HP hasta cientos de HP y son compatibles con elementos electrónicos que permiten fácilmente su control, sin embargo, no se recomiendan para aplicaciones de potencias bajas (10 HP o menos).
- **Servomotores de Corriente Alterna.** Estos motores se utilizan en aquellas aplicaciones que requieren un control suave de velocidad y su campo de aplicación es mayor en aplicaciones industriales estacionarias, debido a la disponibilidad de corriente alterna. Sin embargo, la construcción de este tipo de motores provoca diversas desventajas, tales como la rotación no uniforme de la armadura (en los motores de jaula de ardilla) o la baja relación torque-inercia (en los motores con rotores sólidos de acero). Con frecuencia, los servomotores de CA presentan una eficiencia muy baja (5-20%) y necesitan de enfriamiento externo. Además, requieren un sistema analógico de control retroalimentado, lo que normalmente involucra un potenciómetro que retroalimenta la posición del rotor y un circuito para manejar la corriente que pasa por el motor, de tal forma que esta sea inversamente proporcional a la diferencia entre la posición deseada y la posición actual.
- Los **motores de pasos** poseen ventajas inherentes sobre los servomotores convencionales: (1) son compatibles con procesos digitales; (2) es posible el control de lazo abierto; (3) el error de paso no es acumulativo; y (4) el diseño de las escobillas facilita el mantenimiento y la resistencia. La desventaja principal de los motores de pasos es su baja eficiencia, lo cual restringe su uso en aplicaciones con potencias menores a 1 HP.

Una de las características que se requiere tomar en cuenta para seleccionar un motor de pasos o un servomotor, es la repetibilidad de posicionamiento. La repetibilidad que se logra con un motor de pasos depende de la geometría del rotor, mientras que la que se logra con un servomotor depende generalmente de la estabilidad del

potenciómetro y de otros componentes analógicos dentro de un circuito de retroalimentación.

Los motores de pasos se utilizan en sistemas simples de lazo abierto, lo que resulta adecuado para operaciones a bajas aceleraciones con cargas estáticas. Para aceleraciones altas resulta indispensable el uso de un lazo de control cerrado, particularmente si la operación involucra cargas variables. Si un motor de pasos en un lazo abierto es sobrecargado, se pierde toda la información sobre la posición del rotor y el sistema se reinicializa. Los servomotores no están sujetos a este problema.

Después de presentar este análisis general sobre los motores que son capaces de brindar alta precisión en los sistemas de movimiento, se puede afirmar que el motor de pasos sigue presentando mejores características de funcionamiento de acuerdo con las características del proyecto. Es decir, el motor que se utiliza va a trabajar a altas aceleraciones con un esquema de control digital y en donde se requiere de una alta repetibilidad de posición ya que forma parte de un dispositivo experimental complejo. Sin embargo, el dispositivo mismo presenta la ventaja de que es posible cambiar el motor de pasos por un servomotor en caso de que sea necesario, tan sólo haciendo algunos cambio mínimos a la configuración del controlador y a los comandos de éste.

2.4. HP VEE: Lenguaje de Programación

En esta sección se explica brevemente el lenguaje de programación gráfica llamado HP VEE (*Visual Engineering Environment*) utilizado para generar los programas de las rutinas. HP VEE permite crear de una manera óptima tareas con control, adquisición, análisis y presentación de datos experimentales, principalmente. Este lenguaje permite reducciones considerables en el tiempo de desarrollo de una prueba, debido a que los programas se construyen por medio de la conexión de iconos u objetos en una interfaz gráfica (ver la Figura 2.22), obteniendo así un programa que se asemeja a un diagrama de bloques. De esta manera, con HP VEE se reducen las complejidades de la programación textual, ya que su metodología resulta inherentemente intuitiva para el programador [18].

La Figura 2.23 muestra la ventana principal (*Main*) del módulo SMI (el cual se detalla en el Capítulo 3). El programa que se muestra, permite al usuario del controlador del motor seleccionar una de cuatro posibles rutinas del experimento de Birrefringencia

Bicolor, cada una de las cuales genera una historia de deformación en particular. A partir de este ejemplo, se explican algunos conceptos fundamentales.

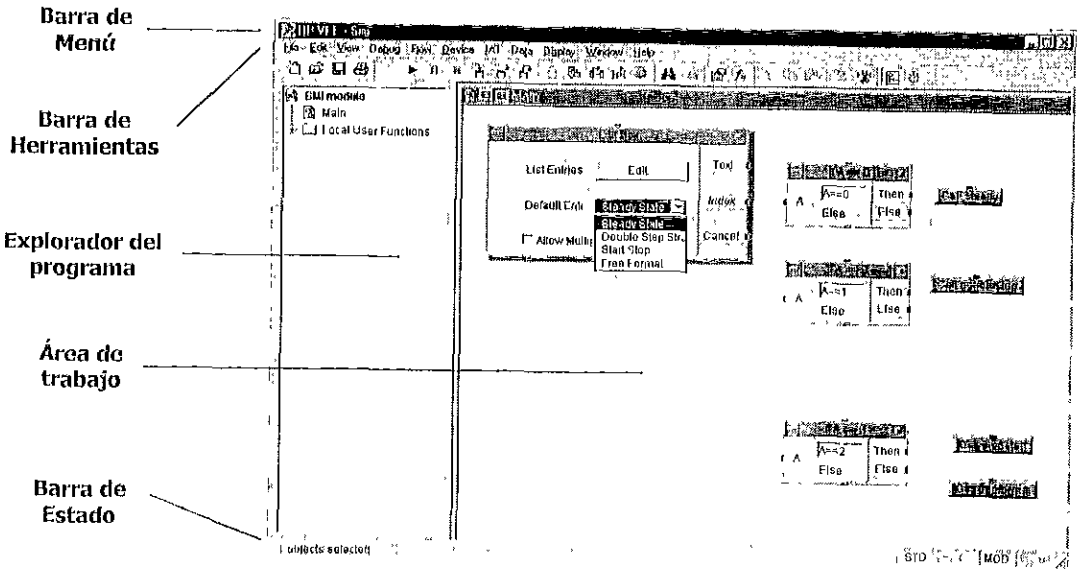


Figura 2.22. *Ambiente de trabajo en HP VEE. En esta ventana se observa la función* Main *del módulo SMI.*

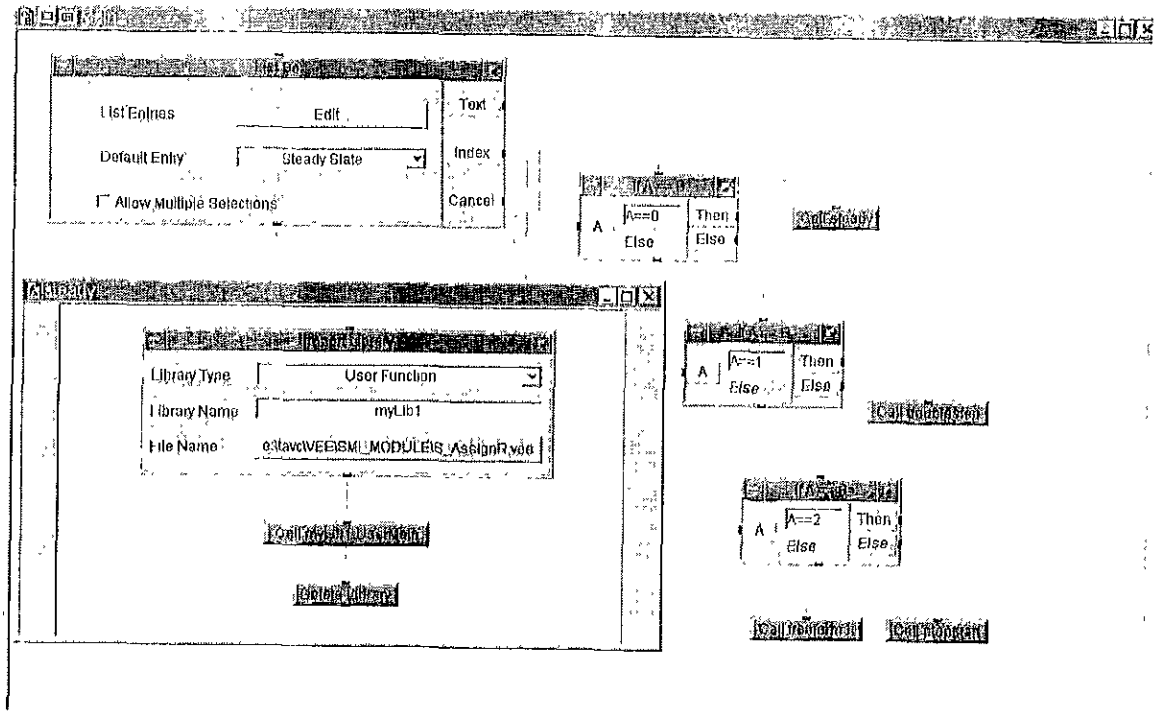


Figura 2.23. *Ventana dentro de la cual se realiza el procedimiento para importar la biblioteca* S_AssignR.vee *que lleva a cabo la rutina de Estado Estacionario* (Steady State).

La mayoría de los lenguajes de programación utilizan funciones para crear subprogramas que desarrollan tareas específicas. En este lenguaje existen dos formas de crear estos subprogramas o subrutinas: por medio del Objeto de Usuario (*User Object*) o por la Función de Usuario (*User Function*) [2]. El Objeto de Usuario es, como su nombre lo indica, un objeto definido por el usuario que se utiliza como cualquier otro objeto de HP VEE. Si se requiere utilizar en algún otro lugar del programa, se puede copiar cuantas veces sea necesario; sin embargo, si se le asigna alguna característica a uno de ellos, mediante un valor asociado a un parámetro o variable, ésta se debe asignar también en cada uno de los objetos copiados. En cambio, la Función de Usuario se almacena para poderla requerir desde el objeto *Call* o algún otro campo de expresión. Los cambios que se hagan a la Función de Usuario se llevarán a cabo en todas las instancias del programa donde se llama a esa función. En ambos casos, el programador puede asignar o personalizar los títulos de cada objeto según la tarea que realicen.

Además, se puede crear una colección de Funciones al salvar externamente varias Funciones de Usuario dentro de un mismo archivo. Esta colección puede activarse en un programa por medio del objeto *Import Library* [19]. Por esta razón y para efectos de este trabajo, al ejercicio de utilizar archivos de Funciones de Usuario dentro de los programas se le denomina importar bibliotecas, lo cual se ejemplifica en la Figura 2.23.

Un beneficio adicional del programa en HP VEE es la facilidad para crear interfaces de comunicación con el usuario, una vez que se ha terminado la parte funcional del programa. Estas interfaces se realizan por medio de lo que se conoce como Vista de Panel (*Panel View*). En la Vista de Panel sólo aparecen los objetos (sin líneas o conectores) que requieren la interacción con el usuario al momento de ejecutar el programa, tales como datos de entrada, gráficas, advertencias, errores, etc. (ver Figura 2.24).

HP VEE maneja cuatro tipos de objetos para controlar instrumentos. La Tabla 2.4 contiene información básica sobre las diferencias que existen entre estos objetos para controlar instrumentos. Con los objetos *To/From, VXIplug&play, Panel Driver* y *Component Driver* se pueden controlar instrumentos sin conocer los detalles de la sintáxis y términos nemotécnicos de la programación de instrumentos. En cambio, si se prefiere la

[2] Cada función y cada rutina se construyen dentro de su propia ventana, por lo que cuando se ejecuta un programa, se observan múltiples ventanas al mismo tiempo, como se observa en la Figura 2.23.

comunicación con éstos enviando términos nemotécnicos de bajo nivel, o si el driver no está disponible para los instrumentos se utiliza el *Direct I/O.*

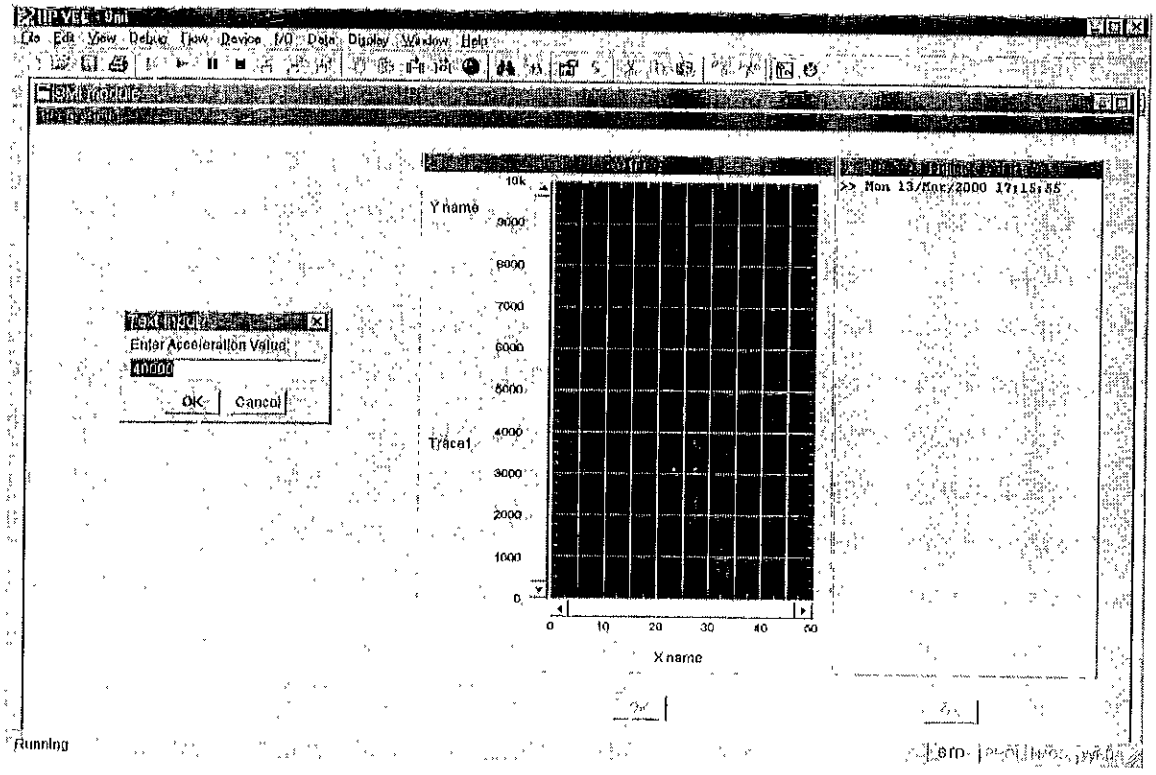


Figura 2.24. *Interfaz gráfica en la que el usuario puede asignar valores a las variables y observar el proceso de graficación.*

Objeto de HP VEE	Acceso al Instrumento	Beneficios Principales	Interfaces que soporta
Direct I/O	Se comunica directamente con cualquier instrumento.	Rápido I/O. Puede controlar cualquier instrumento.	HP-IB o GPIB, serial, GPIO, VXI y LAN
To/From VXIplug&play	Requiere un driver VXIplug&play suministrado desde el fabricante del instrumento. Requiere VISA para ser instalado.	Rápido I/O. Los drivers pueden ser usados por múltiples aplicaciones de software.	HP-IB o GPIB y VXI
Panel Driver	Requiere un Driver de instrumento HP que proporciona el HP VEE.	Fácil de usar.	HP-IB o GPIB y VXI
Component Driver	Requiere un Driver de instrumento HP que proporciona el HP VEE	I/O más rápido que con Panel Driver	HP-IB o GPIB y VXI

Tabla 2.4. *Comparación de los objetos que permiten controlar instrumentos en HP VEE.*

En este lenguaje de programación visual existen tres posibles formas de comunicación con instrumentos utilizando transacciones I/O:

1. El objeto *Direct I/O* que ya se describió con anterioridad.
2. El objeto *MultiDevice Direct I/O* que permite desarrollar transacciones I/O directamente con múltiples instrumentos, a partir de un simple objeto.
3. El objeto *Interface Operation* que puede enviar mensajes, comandos y datos de las interfaces HP-IB, GPIB o VXI.

Para efectos de este trabajo, la comunicación con el controlador se realiza por medio del objeto *Direct I/O*. Este objeto permite leer y escribir datos a los instrumentos en la misma forma en la que se pueden leer datos desde el instrumento y almacenarlos en un archivo. Para poder lograr esta comunicación con los instrumentos, es necesario realizar la configuración adecuada del protocolo de comunicación. La Figura 2.25 muestra el cuadro de diálogo para configurar el puerto serial de comunicación del controlador de motores de pasos MM4000.

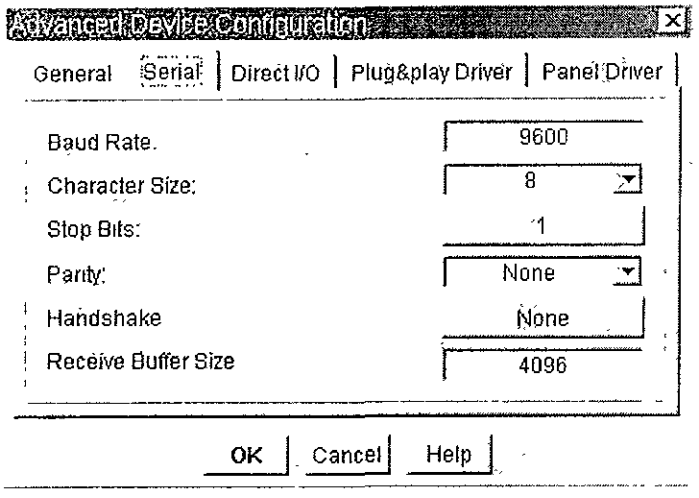


Figura 2.25. *Cuadro de diálogo para configurar el puerto de comunicación serial.*

Para enviar comandos al instrumento, se pueden utilizar transacciones WRITE, que pueden ser las que se muestran en la Figura 2.26. Las más importantes para enviar comandos vía el HP-IB, GPIB, VXI y puerto serial son las transacciones *WRITE TEXT, WRITE BINBLOCK* y *WRITE STATE.* Para instrumentos con GPIO solamente se utilizan *WRITE BINARY* y *WRITE IO CONTROL. WRITE REGISTER* y *WRITE MEMORY* son

transacciones que únicamente se utilizan para comunicación con bases de registros de instrumentos VXI. En cambio, la mayoría de los instrumentos que utilizan HP-IB, bases de mensajes VXI y Puerto Serial utilizan cadenas de texto legibles para los seres humanos como comandos de programación. Estos comandos son fáciles de enviar utilizando la transacción *WRITE TEXT*[20].

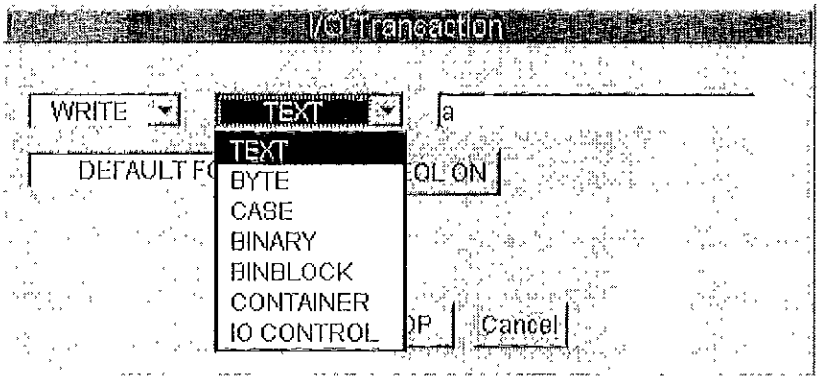


Figura 2.26. *Ventana para seleccionar una de las diferentes posibilidades de transacciones* WRITE.

En este capítulo se han descrito brevemente los elementos básicos de la teoría de control, las características de los motores de pasos así como del controlador y del lenguaje de programación utilizados en la realización de este trabajo.

CAPÍTULO 3

Programación de las Rutinas del Motor

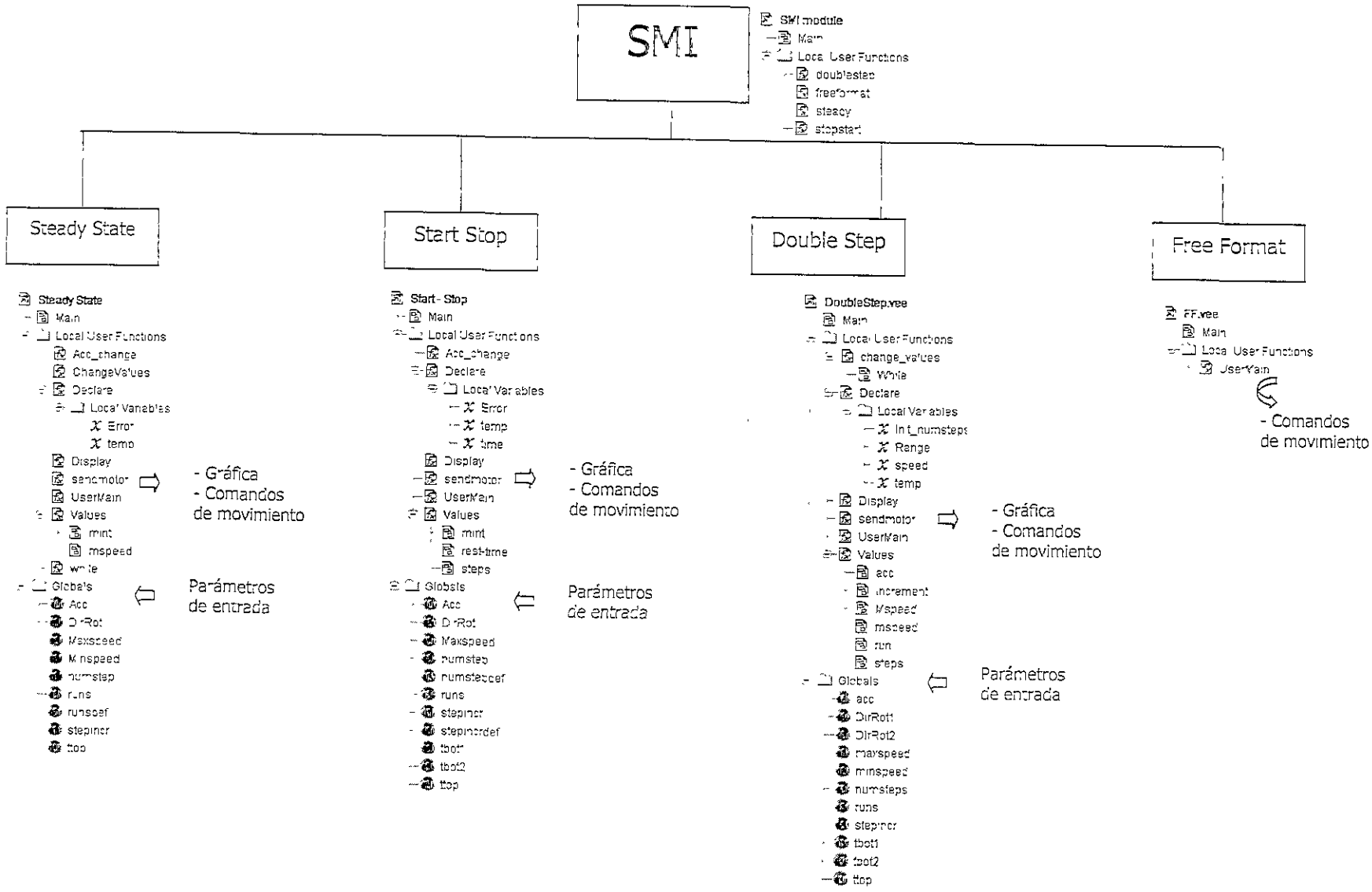
Como ya se mencionó, el problema de este trabajo reside en primer lugar en lograr el acoplamiento entre los instrumentos que se encuentran en el Laboratorio de Reología-Óptica, que se refiere a una computadora central, un controlador avanzado de movimiento, un motor de pasos y un protocolo de comunicación entre la computadora y el controlador.

Como segundo punto se requiere elaborar los códigos de programación que permitan crear una adecuada interfaz de operación entre el usuario y el sistema de control, así como generar y enviar una secuencia compleja de instrucciones al controlador para que éste maneje el movimiento del motor de pasos con base en las necesidades del experimento de Birrefringencia Bicolor.

El experimento requiere de diferentes módulos para su control total, entre ellos se encuentra el Módulo SMI, que tiene como función mover el motor de pasos de forma que se generen las diferentes historias de deformación que se necesitan para analizar al fluido en estudio. Este módulo es un programa que permite al experimentalista introducir los parámetros requeridos por cada historia de deformación, tales como aceleración, velocidad, tiempo, incrementos de velocidad o de tiempo, dirección de rotación, entre otros, y a su vez da como resultado una secuencia de movimiento del motor, así como graficas en pantalla acerca del comportamiento del mismo. El Cuadro 3.1 esquematiza la composición del módulo SMI.

Las rutinas o subprogramas que componen este módulo tienen restricciones y limitaciones, ya sea de *hardware* o de *software*, que se deben al controlador, al motor y a la misma física del experimento, por lo que los parámetros que introduce el experimentalista se verán modificados en repetidas ocasiones.

En este capítulo se explica detalladamente cómo se da solución a la problemática específicamente de *software*, es decir, se describe con detalle el funcionamiento de los programas que generan las diferentes historias de deformación.



Cuadro 3.1. *Estructura del Módulo SMI. Las flechas hacia la izquierda indican los parámetros que introduce el usuario y las flechas hacia la derecha los datos que genera el programa.*

3.1. Módulo SMI

El módulo SMI (*Stepping Motor Interface*) contiene el procedimiento encargado de programar y controlar la interfaz del motor de pasos. En un principio y como ya se explicó en el capítulo anterior, el módulo permite escoger entre cuatro condiciones de movimiento para inducir: Flujo en Estado Estacionario, Flujo de Doble Escalón, Inicio y Cese de Flujo y un Formato Libre (ver Sección 2.4). Cada una de estas rutinas genera en el molino de dos rodillos flujos con diferentes historias de deformación. Los nombres de las rutinas corresponden a aquellos flujos que se desean analizar y que se exponen en la Sección 1.3.

Para cada rutina (o tipo de flujo) se requiere definir los valores para varios parámetros de la ejecución del experimento y del flujo (o historia de deformación a analizar). La rutina recibe del usuario los valores para los parámetros requeridos y calcula entonces los valores más aceptables que el controlador puede ejecutar, de tal forma que cubran con un intervalo ligeramente más amplio que lo solicitado por el usuario o el experimentalista. Esto es, los valores del usuario no necesariamente tienen sentido para las características del motor y del experimento, y por lo tanto, se requiere generar un intervalo de valores para los parámetros iniciales, que sean factibles y que además sean de interés al experimentalista. Estos valores calculados se guardan en un archivo, para que posteriormente, se puedan utilizar por otros módulos del experimento, si así lo desea el operario. Las rutinas a su vez, envían los comandos necesarios al controlador para que el motor de pasos comience el movimiento y se genere así la trayectoria optimizada.

El módulo se compone de la función *Main* y de cuatro *UserFunctions: steady*, *doublestep*, *startstop* y *freeformat*. Cada una de estas Funciones de Usuario importa las bibliotecas que contienen los programas que producen cada historia de deformación, ya que cada una constituye un programa independiente. La función que se llama en todos los casos para que ejecute cada rutina es la *UserMain*. Es decir, se crearon cuatro programas que se pueden ejecutar de forma independiente:

Steady State (Rutina de Flujo en Estado Estacionario)
Double Step (Rutina de Flujo de Doble Escalón)
Start Stop (Rutina de Inicio y Cese de Flujo)
Free Format (Formato Libre)

Para ejecutar el módulo SMI es necesario importar estos cuatro programas. Sin embargo, al ejecutarse este módulo, importa sólo las bibliotecas de la rutina que el usuario ha seleccionado y que se encuentran en un archivo externo (ver Figura 2.28). Al final del programa elimina aquellas que no fueron seleccionadas para el ahorro de espacio de dicho archivo.

3.2. Formato Libre

Como se menciona con anterioridad, las rutinas de Estado Estacionario, de Flujo de Doble Escalón y de Inicio y Cese de Flujo, generan historias de deformación específicas, las cuales han sido diseñadas para estudios de la física de polímeros como se describe en el Capítulo 1. Sin embargo, el *Formato Libre* no representa la generación de algún flujo específico, sino que brinda al usuario la libertad de incursionar en flujos de características diferentes a los tres anteriores, ya que consiste de un programa por medio del cual se envían comandos al controlador desde la computadora, en caso de que se requieran realizar trayectorias diferentes a las anteriores, o simplemente se necesite cambiar de posición al motor, etc.

Se hace notar que el uso de Formato Libre no es trivial puesto que para poder generar una rutina es necesario el conocimiento de los comandos del Controlador MM4000 [12,13]. El Formato Libre consiste solamente de una función *UserMain*, que es la que manda los datos de entrada al controlador por medio de una transacción *Direct I/O* como se muestra en la vista detallada de la Figura 3.1 y genera la interfaz gráfica con el usuario que se muestra en la Figura 3.2. Los comandos se introducen por medio del cuadro de diálogo de título *Text Input* y en caso de necesitar más de un comando se deben separar por comas como se muestra en la Figura 3.2. El monitor del *bus*[1] muestra, al igual que en el resto de las rutinas, los comandos que son enviados al motor por medio de la interfaz de comunicación entre los dispositivos de control.

[1] Se le llama bus al conjunto de conductos que transportan señales eléctricas generalmente de carácter digital.

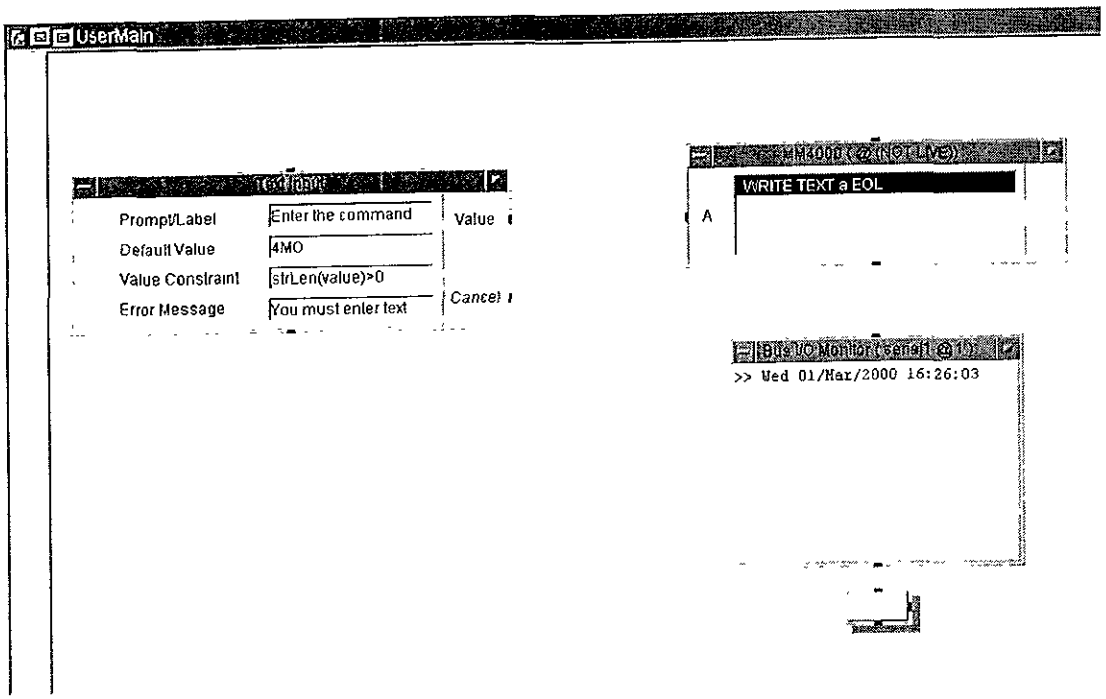


Figura 3.1. *Vista detallada de la función* UserMain *del Formato Libre.*

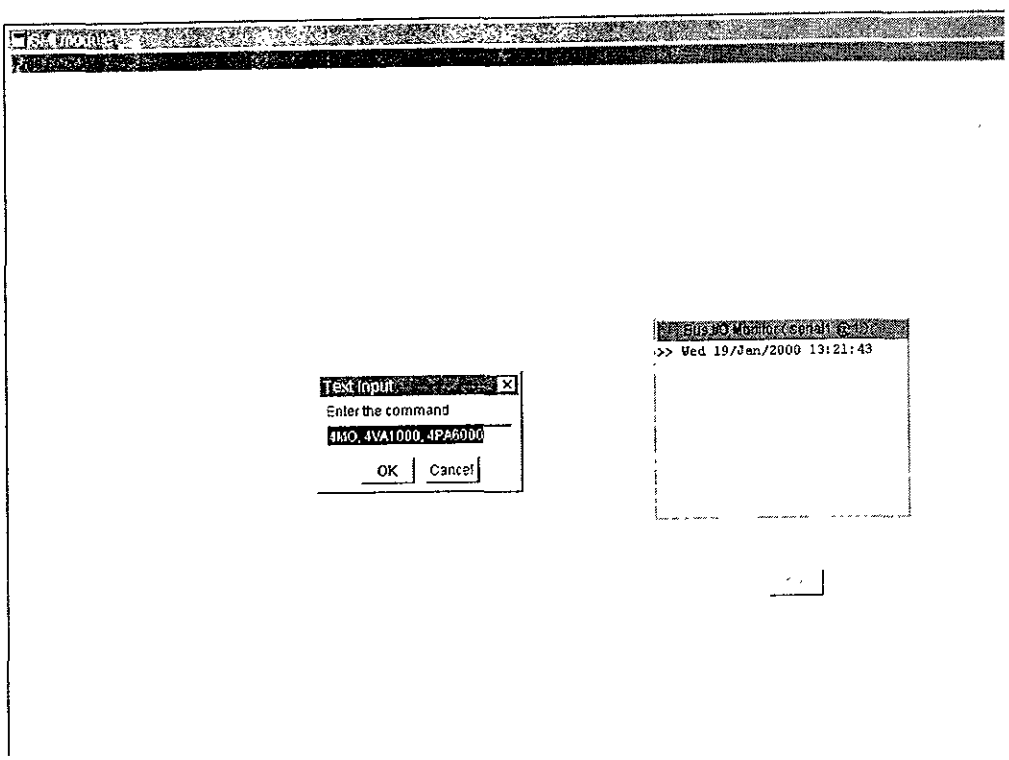


Figura 3.2. *Interfaz gráfica de la función* UserMain *del Formato Libre. El cuadro de diálogo* Text Input *permite al usuario manejar el motor a partir de los comandos del Controlador MM4000.*

3.3. Estructura General de las Rutinas Específicas

Para generar distintas historias de deformación sobre un polímero en el experimento de Birrefringencia Bicolor Inducida por Flujos, es necesario realizar como primer paso los algoritmos que envían una sucesión de datos al motor de pasos a través del controlador. Así, cada rutina consiste de un programa independiente (aunque todas ellas tienen un estructura similar); sin embargo, la función principal denominada *UserMain* es la misma para las subrutinas *Steady State, Start Stop* y *Double Step*, mientras que los datos de entrada y de salida, son diferentes en cada caso. Los objetivos de la función *UserMain* son:

- Ofrecer (al usuario) valores que sean aceptables (de default).
- Ofrecer (al usuario) el conjunto de valores para el experimento con base en valores antes utilizados.
- Ofrecer la posibilidad de que el usuario asigne un nuevo conjunto de valores.
- Una vez que el conjunto de valores se tiene, evaluar si éstos son adecuados para el motor y el experimento, y en su caso, calcular los valores más cercanos a los iniciales que cumplen con los criterios establecidos para el experimento.
- Con base en valores aceptables ofrecer (al usuario) la ejecución del experimento.

Los códigos y el diagrama de las acciones se presentan en la Figura 3.3. El primer icono (esquina superior izquierda) es la función o subrutina *Declare* y tiene como tarea definir las variables del programa, ya que se pueden utilizar dos tipos de variables: *globales* y *locales*. Las primeras pueden utilizarse en cualquier lugar dentro del programa, y las segundas se utilizan dentro del bloque interno de cada función en particular. Los siguientes iconos, etiquetados como *0* y *JTC* (*Junction*) en la Figura 3.3, conforman la salvaguarda para restablecer un estado inicial en caso de que las modificaciones de los valores de las variables del programa no resulten aceptables de acuerdo al criterio del experimento o del usuario. El icono *Last or New parameters?* se utiliza para consultar al usuario, si quiere utilizar valores nuevos u otros que hayan sido almacenados en un intento anterior. Si los valores que se requieren son los almacenados, el icono *Call Display* llama a la función que despliega los valores que el usuario ha introducido con anterioridad y que el programa avala como aceptables.

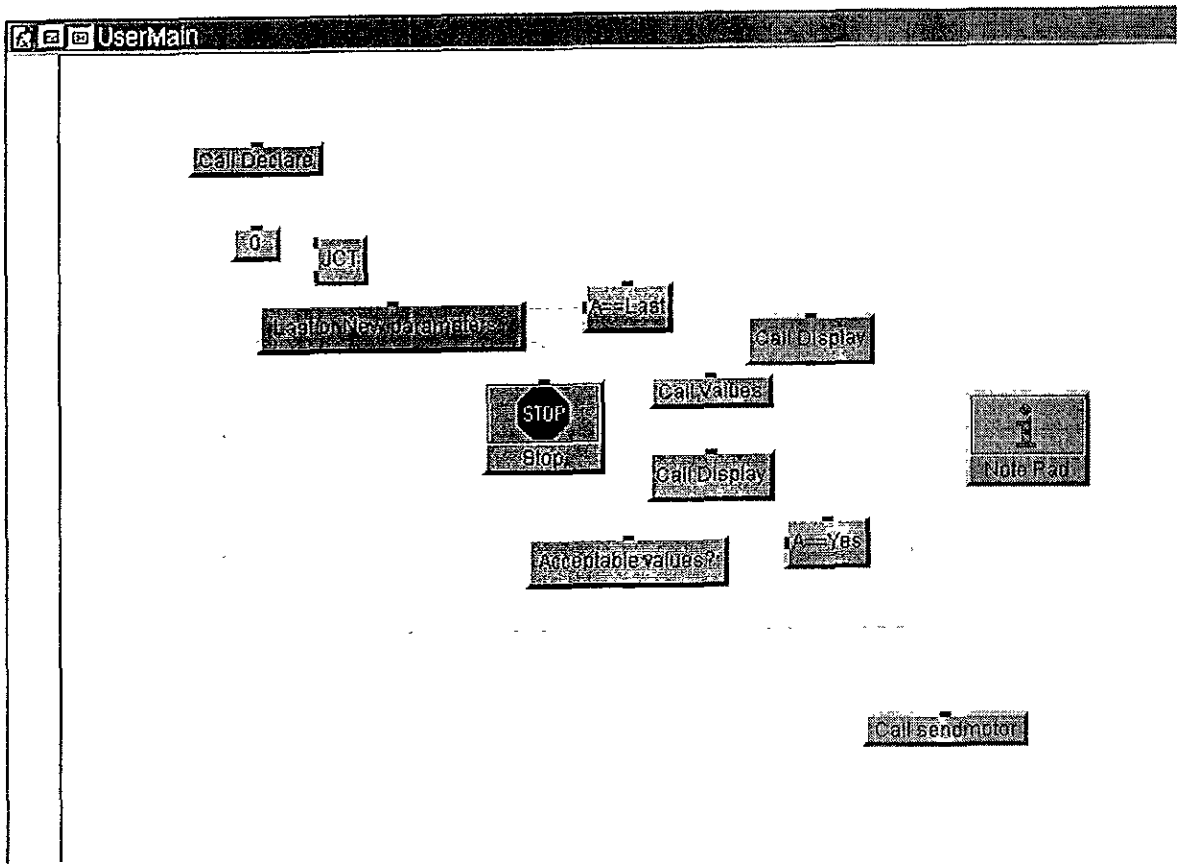


Figura 3.3. *Vista detallada de la función* UserMain *de las rutinas de Estado Estacionario, Inicio y Cese de Flujo y Flujo de Doble Paso.*

Cuando aparece un icono con *Call* antecediéndole, ello significa que se llama a la función cuyo nombre le sigue. Si los valores son nuevos, el objeto *Call Values* llama a la función donde se asignan los valores de las variables. Esta asignación se lleva a cabo por medio de cuadros de diálogo en los que se pueden escoger valores por default o escribir otros diferentes. A su vez, la función *Values* llama a otras funciones objeto o subrutinas para realizar los cálculos necesarios con el objeto de obtener los valores aceptables para el experimento. Posteriormente se ejecuta la función *Display* la cual se encarga de presentar en pantalla tanto los datos de entrada como los valores aceptables para que el usuario pueda cotejar los cambios de estos valores después de los cálculos de optimización (ver Figura 3.4).

Acceptable values es un cuadro de diálogo en el que se pregunta si los datos mostrados por la función *Display* son aceptables. Si la respuesta es negativa la condición

A=Yes resulta ser falsa y entonces se regresa al punto inicial (icono JCT). Si la respuesta es **sí**, el programa continúa y ejecuta la Función de Usuario *sendmotor.*

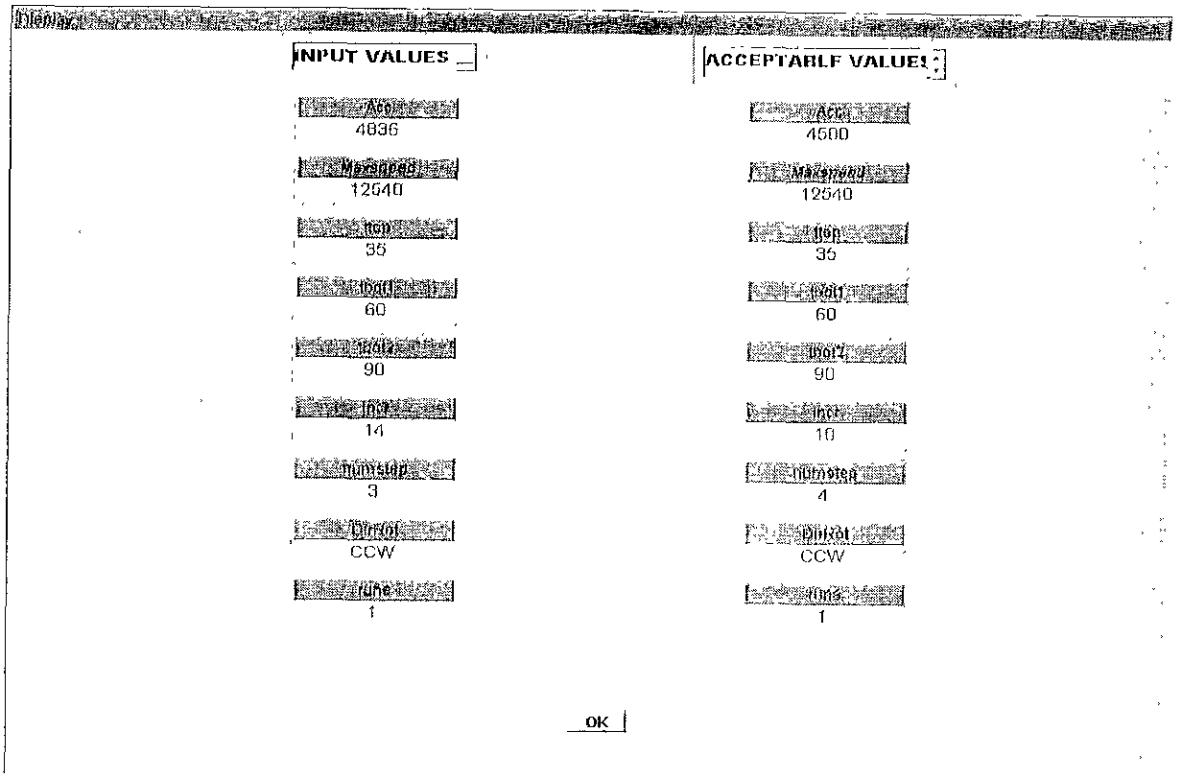
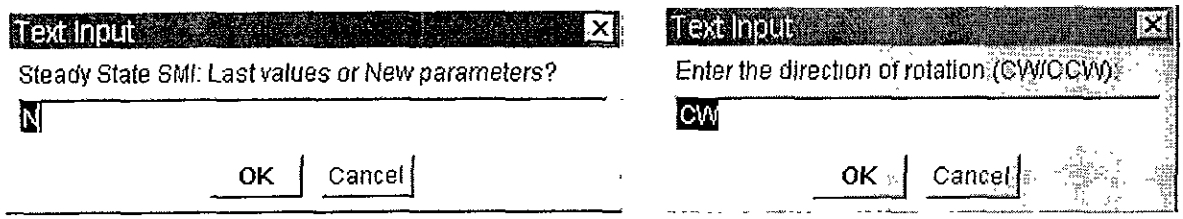


Figura 3.4. *Interfaz gráfica de la función* UserMain *de la rutina de Inicio y Cese de Flujo. Esta interfaz permite al usuario comparar los valores que asignó a cada variable con los que el programa calcula como aceptables.*

Entre las bondades del HP VEE que se mencionan en el Capítulo 2, se incluye precisamente la posibilidad de generar *interfaces gráficas* como la de la Figura 3.4, las cuales se utilizan para mostrar sólo los objetos que interactúan con el usuario, como son los cuadros de diálogo (donde se asignan valores a las variables), los cuadros que muestran las gráficas y las advertencias, entre otros, etc. De tal forma que cuando se ejecutan las diferentes rutinas, aparecen interfaces gráficas de algunas funciones que proveen un gráfico de ayuda visual para el usuario, lo cual representa uno de los objetivos de este trabajo.



(a) (b)

Figura 3.5. *Cuadros de diálogo que forman parte de la interfaz gráfica de la función* UserMain *de la rutina de Estado Estacionario. En (a) se pregunta al experimentalista si desea realizar el experimento con el último paquete de valores o si prefiere establecer nuevos valores, en cuyo caso aparecen diversos cuadros de diálogo como el de (b) en donde tiene la posibilidad de ingresar datos diferentes.*

La Figura 3.5 muestra los cuadros de diálogo que aparecen en la interfaz gráfica de la función *UserMain* y que permiten al usuario definir el valor de los parámetros que requieren cada una de las rutinas. Por último, según la secuencia de la función *UserMain* de la Figura 3.3, el programa manda al controlador los comandos y los valores aceptados por medio de la función *sendmotor*. En esta función los datos se envían a través del puerto serial (interfaz RS-232).

Este esquema se utiliza para todas las historias de deformación específicas (Estado Estacionario, Flujo de Doble Escalón e Inicio y Cese de Flujo). Lo que varía en cada rutina son las variables a definir, así como los intervalos de valores para aquellas que son razonables. Igualmente cambian las interfaces gráficas y los comandos enviados al controlador, es decir el contenido de las Funciones de Usuario *Values*, *Display* y *sendmotor*. Finalmente, los iconos de *OK* se utilizan para hacer una pausa en la ejecución del programa, con el objeto de que el usuario verifique los valores desplegados ya sea en forma de gráfica o de tabla.

Nótese que para todas las variables las unidades de desplazamiento, velocidad y aceleración están dadas en mini-pasos, mini-pasos/s y mini-pasos/s^2 respectivamente. Los mini-pasos se refieren a incrementos angulares en la flecha del motor que equivalen a 2000 incrementos por revolución. Si bien una revolución equivale a 200 pasos en motores de pasos comunes, la necesidad de obtener mayor precisión en desplazamientos, requiere el uso de mini-pasos en lugar de pasos. Un mini-paso es igual a 0.1 pasos completos de 1.8°. A continuación se amplía la información del proceso que realiza cada rutina específica.

3.4. Rutina de Estado Estacionario

Esta rutina genera una secuencia de flujos estacionarios en el fluido polimérico. La secuencia es la siguiente. El motor comienza a la velocidad mínima y se mantiene el tiempo que se le ha especificado; después cambia de velocidad con una aceleración preestablecida y con base en el incremento calculado por el programa, repite la operación hasta alcanzar el número de escalones a velocidad constante solicitados hasta lograr la máxima velocidad. La velocidad angular se mantiene constante durante el mismo periodo de tiempo para cada escalones. Los parámetros que se le introducen al programa son:

Aceleración para cambiar de velocidad (*acc*)

Velocidad mínima o inicial (*minspeed*)

Velocidad máxima o velocidad tope del experimento (*maxspeed*)

Número de escalones a diferente velocidad (*numstep*)

Corridas o número de veces que se repite la secuencia (*runs*)

Tiempo o duración del flujo a velocidad constante (*ttop*)

Incremento de velocidad (en pasos o mini-pasos) entre escalones (*stepincr*)

Dirección del giro (*DirRot*)

La Figura 3.6 explica gráficamente esta rutina y las variables involucradas.

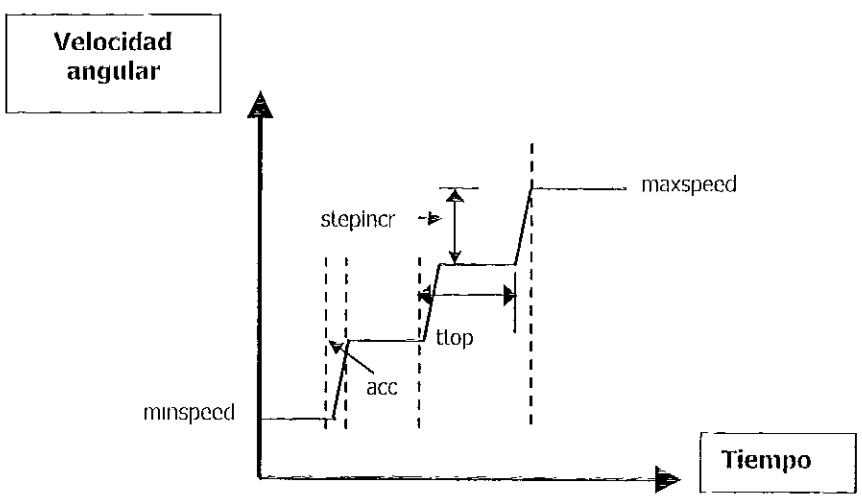


Figura 3.6. *Gráfica de la Velocidad angular en función del tiempo para la rutina de Estado Estacionario. La DirRot establece la dirección de rotación.*

3.4.1. Limitaciones Experimentales

Uno de los objetivos de esta rutina es establecer valores para los parámetros que se introducen al programa que sean aceptables para el experimento, con base principalmente en las restricciones y consideraciones del experimento en sí, del motor, del controlador y de la interfaz. Una de las principales restricciones se refiere al comportamiento transitorio del fluido, y debido a que la rutina busca generar un estado estacionario, es necesario que el tiempo durante el cual se mantiene cada velocidad sea mayor a 10 y menor a 120 segundos. Para evitar que el programa se detenga a causa de un valor fuera del rango, se utiliza la siguiente ecuación:

$$\mathbf{ttop} = 10*\text{ORDINAL}(ttop<=10) + 120*\text{ORDINAL}(ttop>=120) + \\ + ttop*\text{ORDINAL}((10<ttop)\ \text{AND}\ (ttop<120)) \qquad (3.1)$$

La función ORDINAL es una operación lógica que si se cumple da como resultado uno y si es falsa da cero, por lo que si *ttop* es menor o igual a 10 el resultado es 10, si *ttop* es mayor a 120 el valor que el programa envía es 120; para los demás casos ttop=ttop.

La velocidad base o inicial no debe ser muy alta, por lo que si ésta sobrepasa el 80% de la máxima velocidad posible, el experimento comenzará desde cero. De esta forma, el programa no permite que la velocidad mínima sea mayor o igual a la máxima, puesto que resulta ser una inconsistencia para el experimento. Si la diferencia entre ambas velocidades es menor al número de pasos, el programa marca un error y se detiene para que el usuario tenga la posibilidad de ingresar un nuevo valor. Puesto que los valores de número de escalones e incrementos no son independientes, se calculan con base en los valores de la velocidad máxima y mínima, tratando de aproximarse lo más posible a los valores propuestos inicialmente por el operario.

Otra de las restricciones se debe a los valores límite que soporta el motor de pasos y el controlador. La velocidad máxima permitida por el controlador para el motor de pasos UE73PP es de 20000 mini-pasos/s (600 RPM) y la aceleración máxima es de 80000 mini-pasos/s^2, en consecuencia, el programa no permite que se introduzcan valores mayores a los anteriores.

3.4.2. Desarrollo del Código

La función *UserMain* para la rutina de estado estacionario, al igual que la de las otras rutinas específicas, es equivalente a la de la Figura 3.3. Las funciones que conforman esta rutina y hacen posible la rotación del motor son: *Declare*, *Display*, *Values* y *SendMotor*.

Declare es la función donde se definen los parámetros que se mencionan en la Sección 3.4 (variables globales) y la variable *temp* (variable local), que se emplea únicamente para cuestiones de cálculo.

La función *Values* solicita al usuario los valores de entrada como son la aceleración para cambiar de velocidad, la velocidad máxima (final) y la mínima (inicial), el número de escalones a diferente velocidad, el incremento de velocidad entre escalones, la dirección de giro, el número de corridas o veces que se repite la secuencia completa y el tiempo en el que se mantiene el flujo a velocidad constante; algunos de estos se muestran en la Figura 3.6. Estos parámetros se almacenan en un registro (*record* [18]), así como en el archivo llamado *Inputpar*. *Values* también tiene como objetivo analizar si los valores son correctos para el motor y el experimento, y para ello utiliza algunas subrutinas que evalúan los datos introducidos y que se explican a continuación. Los valores aceptables son enviados a otro registro y a su vez un segundo archivo llamado *Acceptablepar*. El objetivo de guardar toda la información, tanto la inicial como aquella calculada, es la posibilidad de que el operario tenga acceso a ella en caso de que requiera analizarla junto con los datos que le brindan las otras secciones de código del experimento de birrefringencia.

La Figura 3.7 representa la función *Values*. El icono *Until Break* representa un ciclo iterativo que termina cuando todos los parámetros de entrada son aceptables para el operario. Al terminar el ciclo, estos parámetros son almacenados en otro registro y luego en el archivo *Acceptablepar*. Los iconos que representan a cada una de las variables en la segunda columna (de la izquierda) de la Figura son los cuadros de diálogo que permiten al usuario introducir el valor de éstos. En cambio, en la tercera columna se pueden observar dos tipos de objetos diferentes: los llamados *Set* "variable" son los que almacenan el valor que se introduce en los cuadros de diálogo y aquellos con los títulos *mspeed*, *Mint*, *Change Values* y *while* son Objetos de Usuario o subrutinas que evalúan los datos iniciales como se expone a continuación.

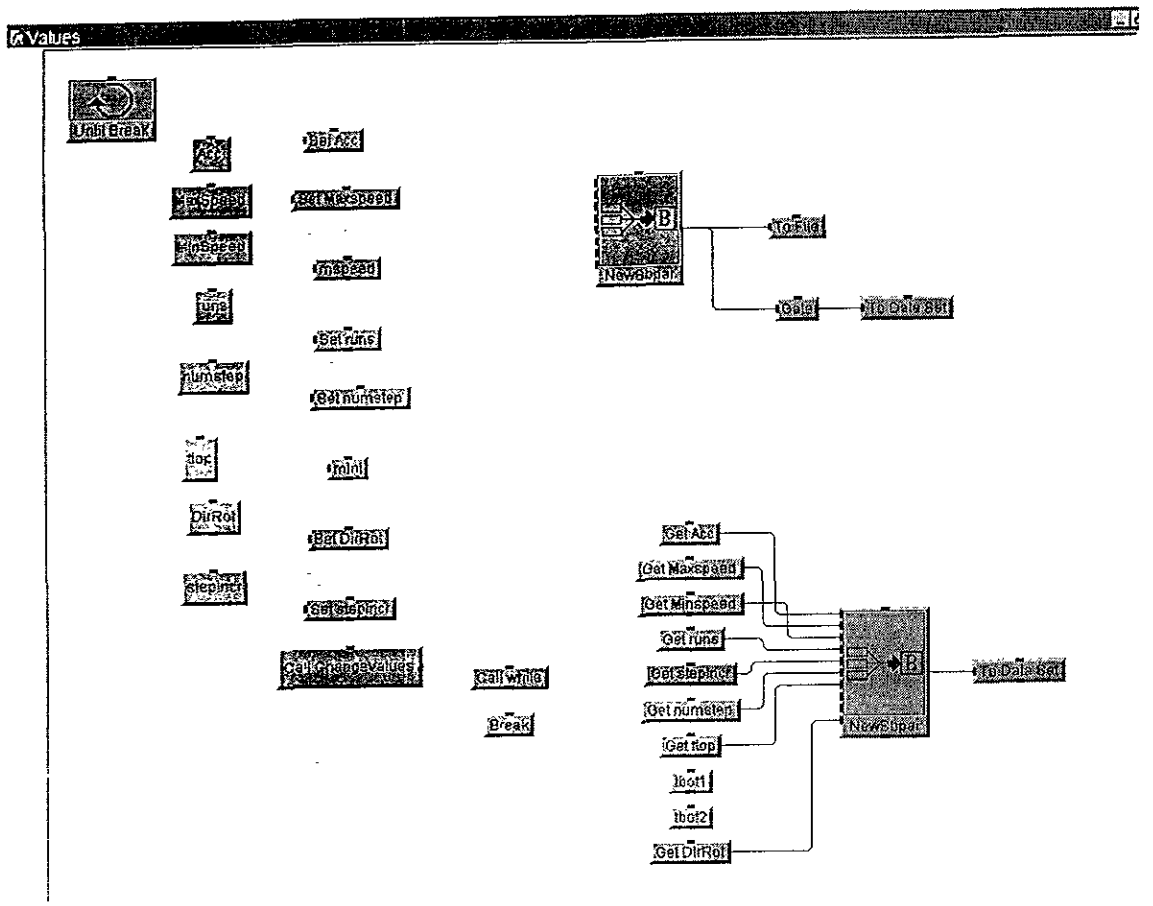


Figura 3.7. *Vista detallada de la función* Values *de la rutina de Estado Estacionario.*

Cabe aclarar, que cuando el usuario considera que los valores aceptables calculados por el programa no son adecuados para el experimento, *Until Break* permite introducir un nueva serie de valores. Los objetos que evalúan los parámetros del flujo de estado estacionario son:

- *Mspeed* se observa en la Figura 3.7 y se encarga de evaluar si la velocidad mínima es menor al 80% de la máxima, de tal forma que si es mayor, envía el aviso que se presenta en la Figura 3.8. Este mensaje se guarda en el archivo de errores para su consulta posterior y a continuación se cambia el valor de la velocidad a cero. Para el caso en el que la velocidad se encuentre dentro del rango permisible, únicamente se archiva el valor de la variable.
- *Mint* utiliza la Ecuación 3.1 en función de que el valor del tiempo sea aceptable. El aviso que manda si este valor no es admisible y que se guarda en el archivo de

errores tiene el mismo formato que el de la Figura 3.8, pero con el texto: *WARNING Transient Behavior Expected!.*

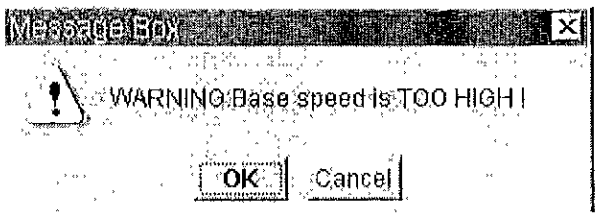


Figura 3.8. *Aviso que genera la función* mspeed *en caso de que la velocidad mínima sea mayor al 80% de la velocidad máxima.*

- *Change Values* asigna el valor de la velocidad máxima permitida por el controlador (en caso de que el valor asignado por el usuario haya sobrepasado el límite), revisa que los incrementos no sean muy pequeños (e.g., fracciones de mini-pasos) y calcula tanto el número de escalones como el incremento con base en las velocidades.
- Dentro de la función *while* se encuentra el objeto que revisa la compatibilidad entre la aceleración y las velocidades.

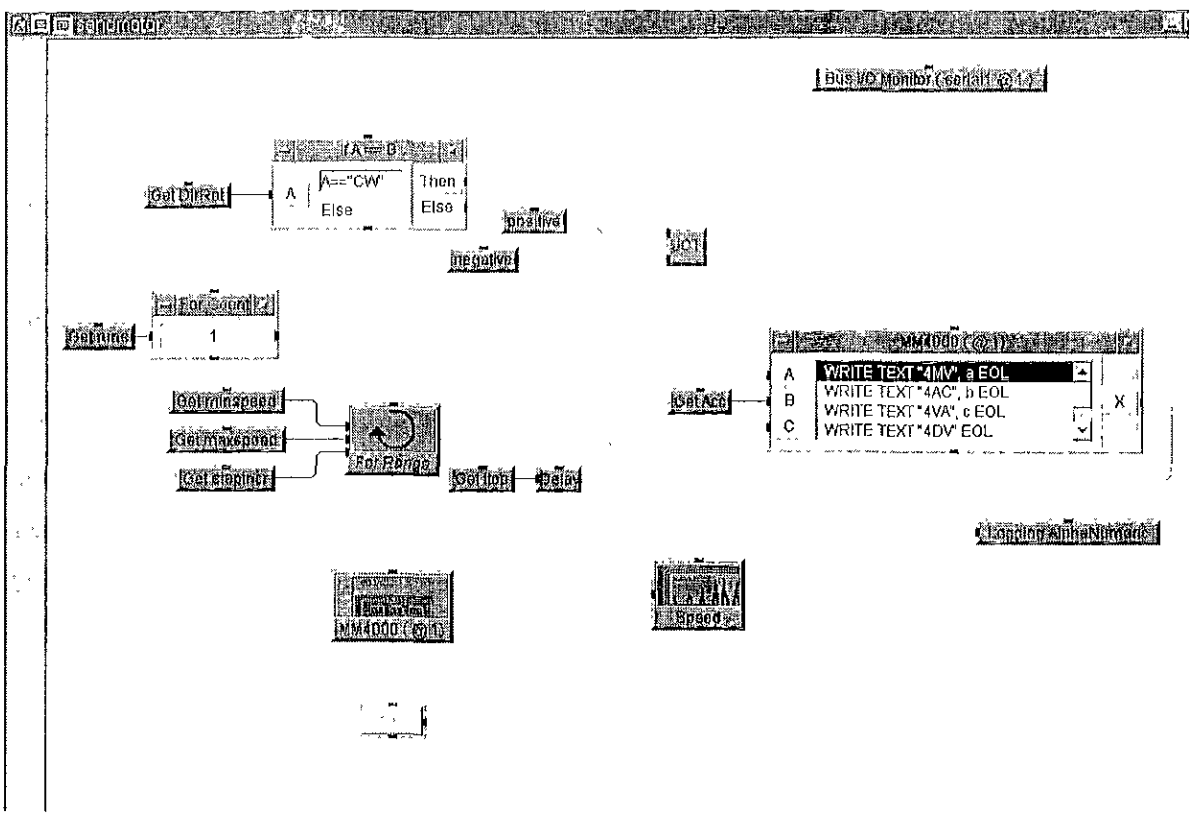


Figura 3.9. *Vista detalla de la función* sendmotor *de la rutina de Estado Estacionario.*

Cuando el programa sale de la Función de Usuario *Values,* pasa inmediatamente al objeto *Display* de la función *UserMain* (ver Figura 3.3). Este objeto tiene como tarea abrir los dos registros que se crearon en *Values* e imprimirlos en pantalla como *INPUT VALUES* y *ACCEPTABLE VALUES* (ver Figura 3.4).

La última función que el programa ejecuta es *sendmotor* (ver Figura 3.9) y desarrolla dos tareas principales, de tal forma que envía al controlador (vía la interfaz) los datos que necesita el motor de pasos para realizar la secuencia elegida por medio de la transacción Directa E/S (ver la Sección 2.2.3) e incluye el procedimiento para generar la interfaz que grafica las velocidades y muestra el monitor del *bus* de datos.

Para la primera tarea es necesario enviar, en primer lugar, la dirección del giro. Si el valor de la variable DirRot es CW, el dato que ingresa a la transacción Directa E/S, por medio de A (Figura 3.10), es "+" y si el valor es CCW ingresa en A un "-". La aceleración se asigna en la variable B, y para el caso de la velocidad fue necesario crear un ciclo que mande la velocidad de base a C, se detenga el tiempo deseado, e incremente este valor conforme a los parámetros antes mencionados (*stepincr* y *numstep*) hasta alcanzar el valor de la velocidad máxima.

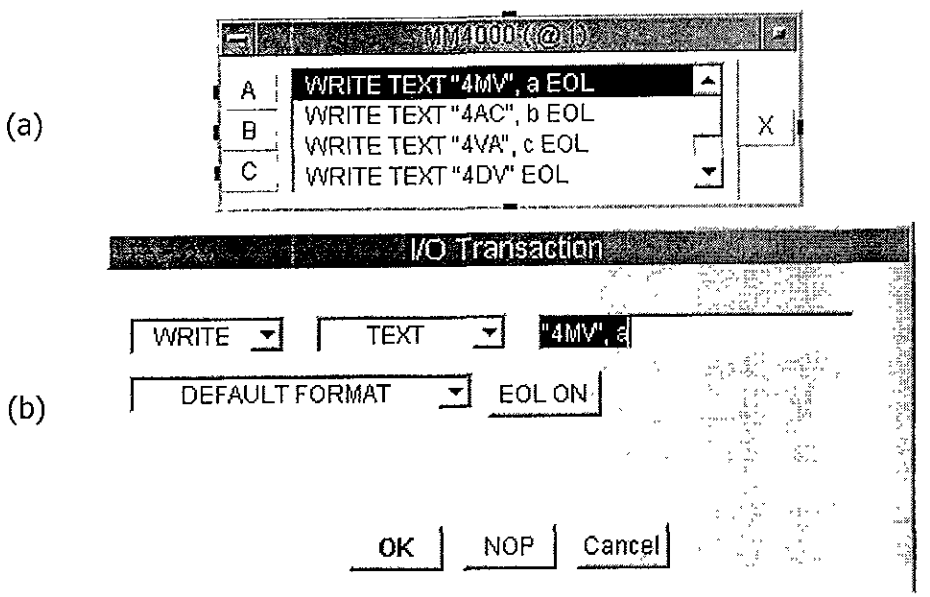


Figura 3.10. (a) *Transacción* Direct I/O *con los comandos que se envían al controlador.* (b) *Cuadro de diálogo de Transacción I/O.*

Los comandos que recibe el controlador a través de la RS-232 se envían en código ASCII, por lo que se utiliza la opción de escribir texto (*WRITE TEXT*) en el cuadro de diálogo de la transacción de entrada/salida, como se muestra en la Figura 3.10. Por medio de esta transacción se generan los comandos que se le envían a la interfaz, sin embargo, también se pueden leer los datos que tiene ésta en el *bus* de E/S, lo que permite monitorear la información que el controlador está recibiendo. Los comandos enviados se explican más adelante.

3.4.3. Presentación de Resultados

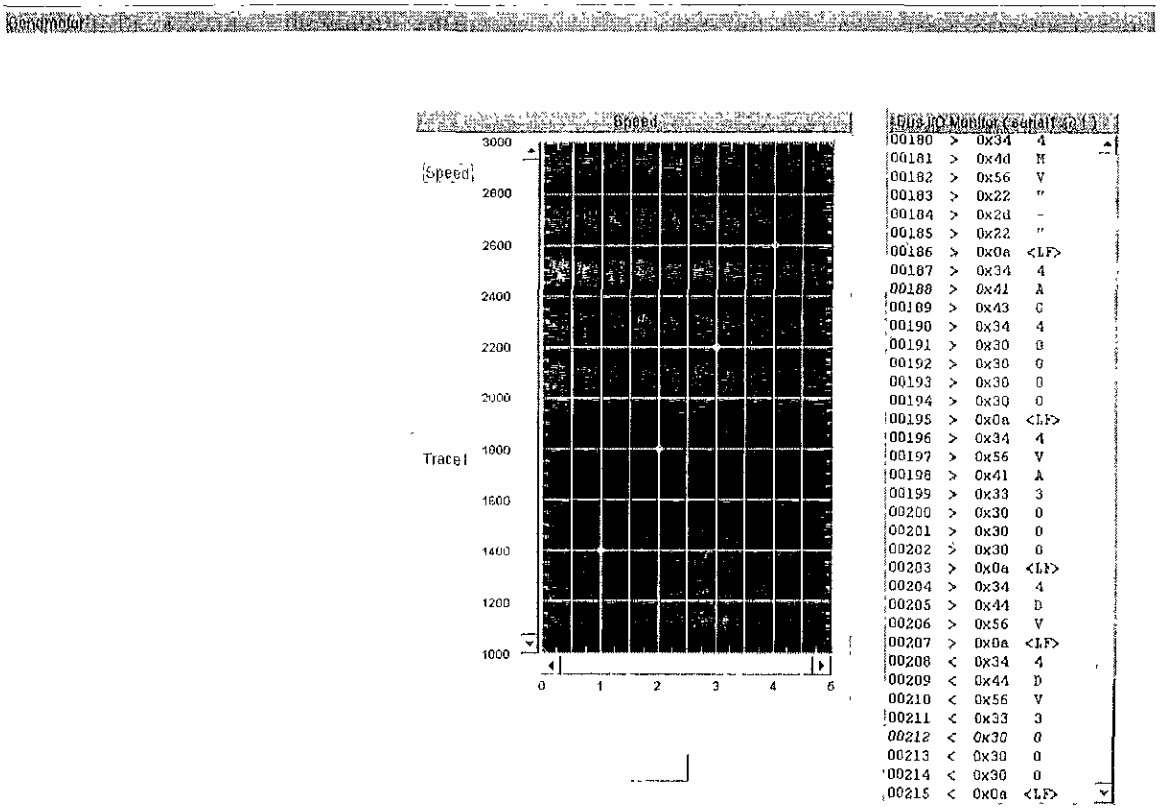


Figura 3.11. *Interfaz de comunicación con el usuario que permite observar la gráfica de la velocidad angular y el Monitor del* Bus *E/S del puerto serial RS-232.*

El programa de la rutina de Estado Estacionario muestra, en principio, los valores que el usuario introduce y los valores aceptables calculados por el programa mismo, en una pantalla equivalente a la de la Figura 3.4. Además imprime en pantalla una gráfica de las

velocidades angulares que se han ejecutado, lo que permite conocer las velocidades en el momento en que las ejecuta el motor (ver Figura 3.11).

Una de las tareas principales del programa es la de enviar los comandos necesarios al controlador por medio de la interfaz para que el motor genere la rutina de estado estacionario (ver Apéndice B). Por lo tanto, en función de observar los estados o condiciones en la que se encuentra el motor, se utiliza un monitor que despliega los datos del *bus* de entrada/salida. Del lado derecho de la Figura 3.11, se muestra el cuadro que monitorea al *bus* y en el que se observan los comandos en el momento en que se transfieren al controlador, de tal forma que cuando se ejecuta el programa se puede observar también el instante en el que cada velocidad cambia de valor. Este Monitor del Bus E/S muestra cuatro columnas para el RS-232:

- Columna 1 - Número de Línea.
- Columna 2 - Salida de datos (>) o entrada de datos (<).
- Columna 3 - El valor hexadecimal del byte transmitido.
- Columna 4 - El caracter ASCII correspondiente al byte transmitido.

En la última columna, los valores se presentan en forma vertical. Los primeros caracteres de arriba hacia abajo indican el comando y los que siguen indican el valor de la aceleración, velocidad, posición, etc, dependiendo del comando. LF indica el cambio de línea, pero en este caso, se envía un solo comando por línea.

Los comandos que se envían al controlador por medio de la interfaz y que están descritos en el Apéndice B son: 4**MV**+/-, 4**AC**xx, 4**VA**xx y 4**DV** (ver Figura 3.9). Ahora se describe el segundo tipo de flujo.

3.5. Rutina de Inicio y Cese de Flujo

Esta rutina genera los códigos para el motor requeridos para producir una condición de arranque y paro abrupto del flujo. En el arranque de flujo se hacen girar los rodillos súbitamente para poder observar el comportamiento de la birrefringencia como se menciona en el Capítulo 1. Opuesto al arranque, el cese detiene abruptamente los rodillos. El tiempo durante el cual el flujo se mantiene (*ttop*) permite conocer la respuesta del líquido polimérico en un estado estacionario de no equilibrio a velocidad *maxspeed.* Generalmente, la señal (anisotropía óptica) comienza desde cero, crece hasta alcanzar un valor máximo y decrece hasta el valor típico del estado estacionario. Es claro que el valor

de la birrefringencia alcanzada dependerá de cuánto tiempo permanece encendido el flujo (*ttop*), y sólo alcanza el estado estacionario cuando el tiempo característico del material es mucho menor que *ttop*. En el caso de cese del flujo, la birrefringencia del material decrece desde el valor típico del estado estacionario y puede llegar a cero si el tiempo de reposo (*tbot1*) es mayor que el tiempo característico del material. En estos experimentos se utiliza la más alta aceleración posible, aunque para propósitos del experimento es deseable poder controlar explícitamente este parámetro.

También se utilizan otros parámetros secundarios para dar mayor capacidad de análisis al operario. Por ejemplo, dado que el cese de flujo está directamente relacionado con el tiempo característico del material, entonces se puede analizar el decaimiento de la señal en función de la duración del tiempo de reposo. Para ello, se propone una secuencia de esos tiempos que inicia con *tbot1* con incrementos *incr* hasta alcanzar *tbot2*. Por estas razones los parámetros que se introducen al programa son los siguientes:

Aceleración (*acc*)
Velocidad (*maxspeed*)
Tiempo de flujo (*ttop*)
Tiempo de reposo mínimo (*tbot1*)
Tiempo de reposo máximo (*tbot2*)
Incremento (*incr*)
Número de pasos (*numstep*)
Número de corridas (*runs*)
Dirección del giro (*DirRot*)

En la Figura 3.12 se observan las variables que utiliza esta rutina, así como su trayectoria. Una vez que el usuario introduce todos los parámetros antes mencionados, el motor comienza a girar con la velocidad *maxspeed* durante el tiempo *ttop*. Se detiene *tbot1* segundos y repite esta operación el número de corridas (*runs*) establecido. Para iniciar un segundo experimento, comienza a girar otra vez durante *ttop* y se detiene *tbot1* + *incr* segundos. Repite la operación el número de corridas deseadas y aumenta otra vez el *tbot1*. Esta operación continúa hasta llegar a *tbot2* y cumplir el número de incrementos (*numstep*) en el tiempo de relajación. La dirección de giro puede ocurrir en ambos sentidos.

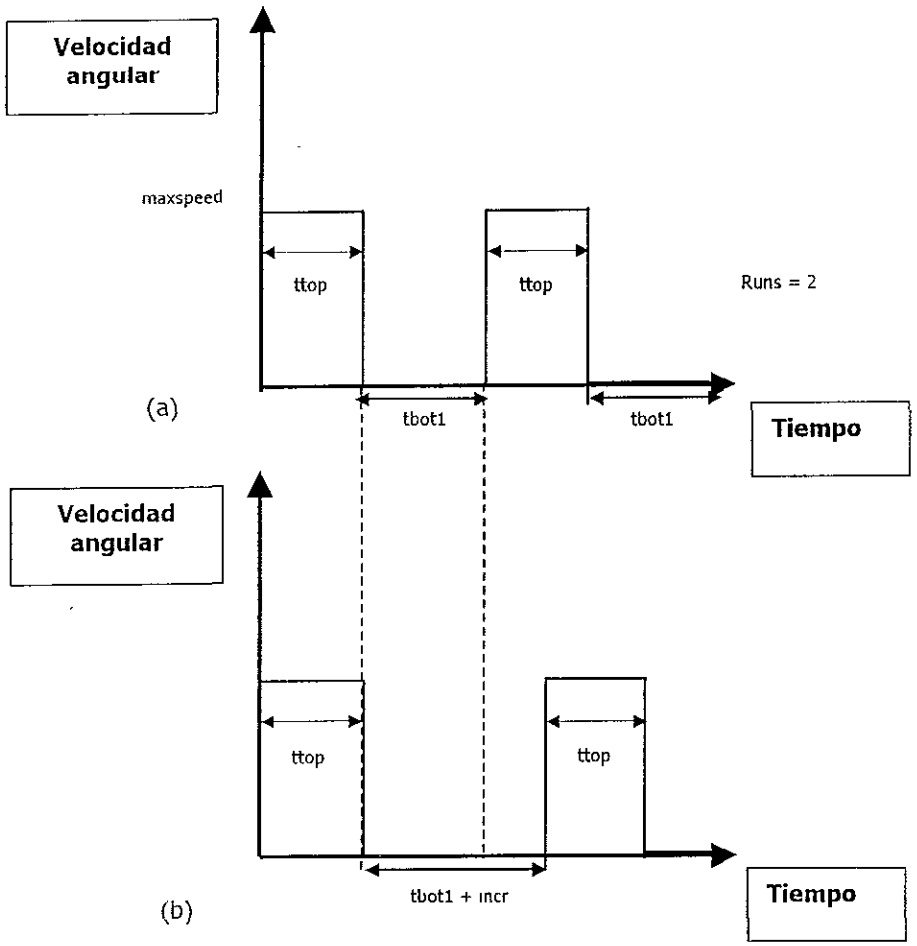


Figura 3.12. *Gráficas de velocidad angular en función del tiempo de la rutina de Inicio y cese de flujo, considerando una aceleración infinita. En (a) se representa la gráfica para cuando el tiempo de reposo es* tbot1*, en cambio, en (b) el tiempo de reposo es* tbot1 *más el primer incremento* incr.

3.5.1. Limitaciones Experimentales

Para el caso de esta rutina se requiere analizar el fluido una vez que alcanza un estado estacionario, por lo que la primera consideración consiste en permitir tiempos cortos pero no demasiado pequeños respecto a la rapidez de cambio posible con el motor y a la dinámica del polímero. Igualmente, en la práctica, el tiempo máximo tampoco puede ser muy largo, pues los experimentos requieren de mucho tiempo y capacidad de almacenamiento de datos. Por ello, los tiempos aceptables están acotados por la Ecuación 3.1, con la que cualquier número menor a 10 tomará el valor de 10 y lo mismo sucederá para los valores mayores a 120.

Si el usuario introduce un valor *tbot1*>*tbot2* el programa detecta la incompatibilidad e invierte estos valores. La resta de *tbot2* menos *tbot1* dividida por el incremento no debe ser menor al número de incrementos, ya que de ser así el experimento brindaría menor información que la originalmente requerida por el operario. En caso de presentarse esta situación, se cambiará el valor del incremento y si es necesario el de *tbot2*.

El programa no permite que la aceleración sea mayor a 80000 mini-pasos/s^2, ya que como se menciona anteriormente, el motor tiene este valor como límite máximo para la aceleración. Es suficiente con introducir el valor del número de pasos o el del incremento, sin embargo, si se introducen los dos y son incompatibles, el programa modifica los valores del *numstep*, el *incr* y *tbot2* para que éstos sean adecuados.

3.5.2. Desarrollo del Código

En esta rutina se tienen el mismo número de funciones así como de variables que en las otras. Las variables locales que se declaran son: *temp* y *time*, y las globales son las que se mencionan en la Sección 3.5.

La función *Values* solicita los valores de la aceleración, velocidad, tiempo de flujo, tiempos de reposo, número de pasos, número de corridas y dirección de giro. Esta función es casi la misma que la de la rutina de Estado Estacionario (ver Figura 3.7) ya que también cuenta con una subrutina llamada *mint* que utiliza la Ecuación 3.1 para cambiar el valor de *ttop*, sin embargo tiene tres subrutinas más: *rest-time*, *steps* y *acc_change*, las cuales se explican a continuación.

- *Rest-time* compara los valores de los tiempos de reposo. Si *tbot2* es menor que *tbot1*, cambia los valores utilizando la variable *temp* que se declara en un inicio y manda un aviso: *WARNING: Incompatible Rest Times. NOW tbot2 > tbot1.*
- *Steps* es la subrutina que calcula los incrementos, el número de pasos y el tiempo de reposo para que sean compatibles entre sí y aceptables para el experimento. Evalúa la posibilidad de que los tiempos de reposo uno y dos sean iguales y si lo son da el valor de cero a *incr* y de uno a *numstep*. El número de pasos por default tiene que ser mayor a la diferencia entre *tbot2* y *tbot1* divida por los incrementos. En caso contrario se presenta una incongruencia de los tiempos de reposo, por lo que el programa manda un aviso y automáticamente cambia los valores según las ecuaciones que se muestra en la Figura 3.13.

$$\mathbf{acc} = \text{maxspeed} * 100 * \text{ordinal}(\text{maxspeed}*100<80000) + \\ + 80000 * \text{ordinal}(80000<=\text{maxspeed}*100) \qquad (3.2)$$

- *Acc_change* utiliza la Ecuación 3.2 que cambia la magnitud de la aceleración de acuerdo a la velocidad máxima, de tal forma que si la velocidad máxima multiplicada por cien es mayor a 80,000 (que es la máxima aceleración que el motor puede ejecutar) el valor de *acc* será igual a 80,000, buscando no rebasar la limitante del motor. Y si en cambio *maxspeed**100 es menor a 80,000, entonces *acc* asumirá el valor de la velocidad máxima multiplicada por el factor de 100, de modo que se garantiza que el perfil de velocidad se asemeja a una función escalón con un error máximo del 1%.

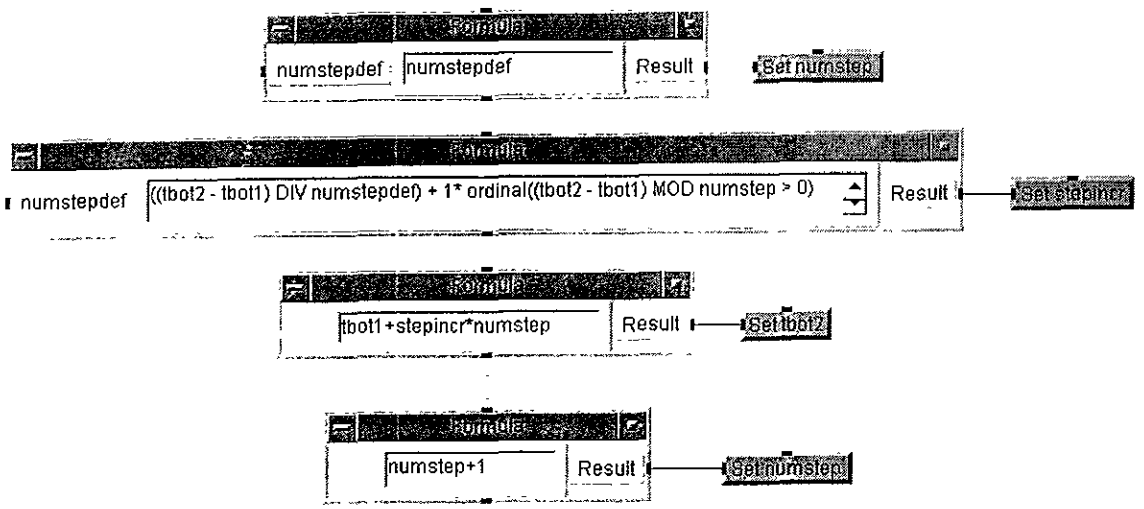


Figura 3.13. *Cambio de valores en la subrutina* step *de la rutina de Inicio y Cese de Flujo.*

La función *Display* es prácticamente equivalente a la de estado estacionario, de tal forma que muestra en pantalla tanto los datos de inicio como los aceptables calculados por el programa como se muestra en la Figura 3.4. La diferencia entre una y otra rutina radica en que los parámetros para cada flujo son diferentes.

SendMotor tiene tres funciones principales. En primer lugar, genera un ciclo iterativo que incrementa el tiempo de reposo cierto número de veces. Para ello utiliza el objeto *For Range* al cual se le especifican el valor inicial (*tbot1*), el valor final (*tbot2*) y el incremento (*stepincr*), de tal forma que produce una serie de valores distribuidos equitativamente

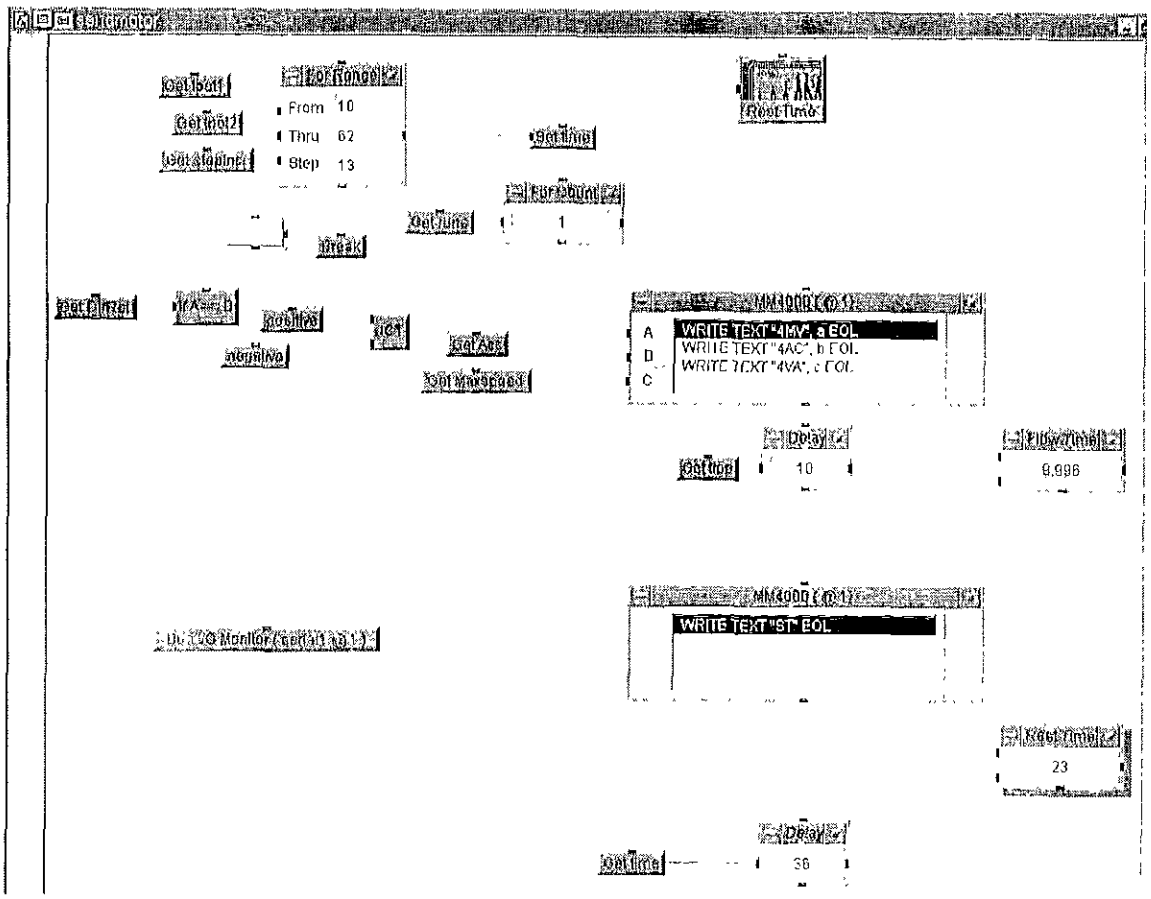


Figura 3.14. *Vista detalla de la función* sendmotor *de la rutina de Inicio y Cese de Flujo.*

entre el primero y el último valor, por lo que no es necesaria la especificación del número de pasos. Para establecer el número de corridas (*runs*) que el usuario requiere para cada tiempo de reposo, se utiliza el objeto *For Count* como se muestra en la Figura 3.14. La segunda función consiste en enviar los comandos al controlador por medio de dos transacciones directas, una para girar al motor durante cierto tiempo y otra para detenerlo el tiempo necesario. *Delay* es el objeto que permite que la acción dure el tiempo especificado, el cual a su vez, se monitorea con otro objeto llamado *Timer* que mide el tiempo transcurrido entre dos eventos. En este caso, mide el tiempo que pasa entre el inicio del giro y el tiempo en el que se detiene el movimiento, y viceversa. Además para el Inicio y Cese de Flujo se utiliza una interfaz gráfica que muestra el tiempo transcurrido para el flujo (*Flow Time*), el tiempo para el cese (*Rest Time*) y el monitor del bus de E/S del puerto serial (ver Figura 3.15). Por último, *sendmotor* grafica los tiempos de reposo en el momento en que se están ejecutando.

3.5.3. Presentación de Resultados

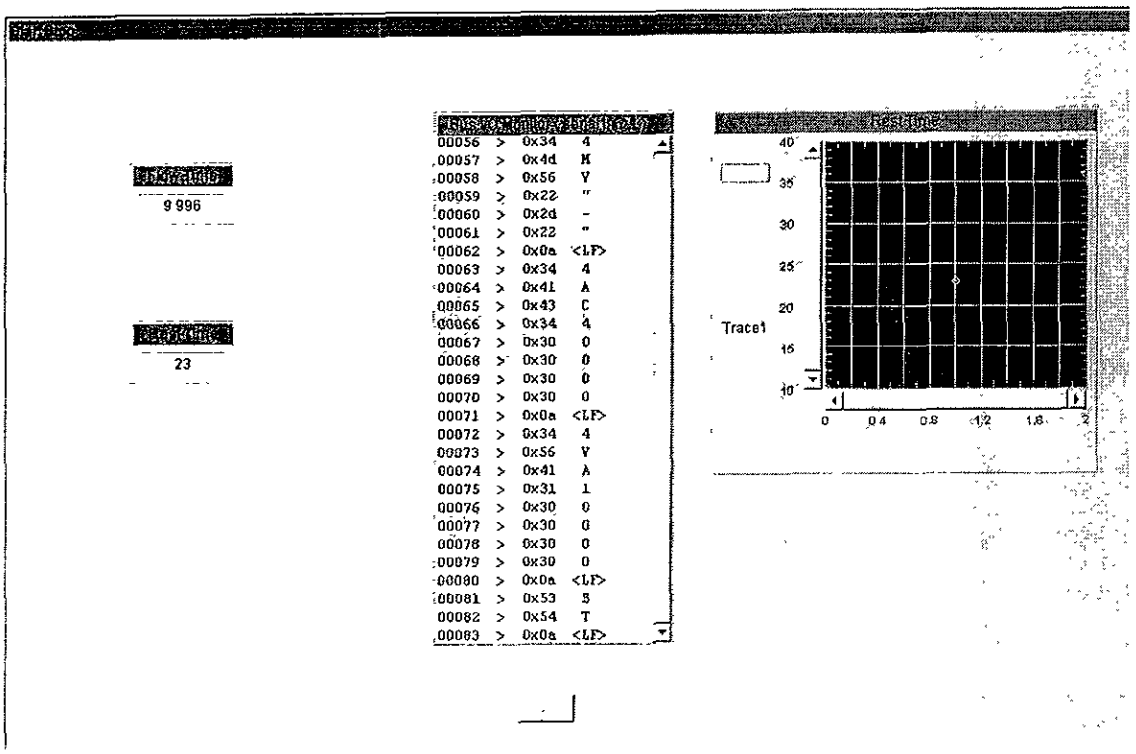


Figura 3.15. *Interfaz gráfica de la función* sendmotor *de la rutina de Inicio y Cese de Flujo durante la ejecución.*

En esta rutina como en las anteriores, se genera una interfaz gráfica en forma de tabla por medio de la función *Display,* para comparar los valores de entrada y los aceptables. A diferencia de las otras rutinas, ésta no gráfica las velocidades; en cambio, permite la observación de los tiempos de flujo y de reposo durante su ejecución, así como el monitor del *bus,* como se muestra en la Figura 3.15. El tiempo *Flow Time* que se observa se mide desde que el motor comienza a moverse hasta que se detiene y el *Rest Time* es el tiempo en el que el flujo cesa, desde que se envía el comando para detener el movimiento hasta que comienza a girar otra vez.

La transacción que se utiliza es la *Direct I/O,* en la cual los datos se envían al controlador en formato ASCII y los comandos que se utilizan se presentan en el Apéndice B. En la Figura 3.15 se presenta el monitor del *bus* en el que se observa el signo de la dirección del giro después del comando **MV.** Si el signo es positivo el motor gira en sentido dextrorso y si es negativo gira en el sentido opuesto. También se muestran los

valores tanto de la velocidad como de la aceleración siguiendo a su respectivo comando. Las columnas de este monitor son las mismas que en la rutina de Estado Estacionario.

Al analizar el Apéndice B, se puede observar que los comandos que se envían al controlador no son muy diferentes en las tres rutinas, la diferencia radica en la forma en la que se envían los valores de las variables, es decir, si se manda un solo dato o un arreglo de éstos, así como si éstos se incluyen en ciclos iterativos o no. Esta rutina utiliza varios ciclos para ejecutar más de una corrida y para incrementar los tiempos de reposo, como se muestra en la Figura 3.14. Los comandos que se envían al controlador son: 4**MV**+/-, 4**AC**xx y 4**VA**xx.

3.6. Rutina de Flujo de Doble Escalón

La rutina de Flujo de Doble Escalón genera dos flujos a diferentes velocidades, de tal manera que se logra imponer en el fluido un cambio brusco a partir de una razón de corte alta a una razón de corte pequeña (y viceversa), situaciones que permiten el análisis tanto de perturbaciones como de histéresis en los tiempos característicos de los fluidos poliméricos.

Como se muestra en la Figura 3.16, la secuencia comienza haciendo girar a los rodillos con una velocidad alta (*maxspeed*), la cual se mantiene el tiempo deseado (*ttop*). Posteriormente cambia bruscamente a la velocidad baja (*minspeed*) y conserva este valor un cierto tiempo (*tbot1*), estableciendo así la conformación inicial de la microestructura del polímero. Este primer ciclo se repite el número de corridas (*runs*) que el usuario haya definido, al igual que ocurre con los ciclos subsecuentes en donde la velocidad alta se mantiene siempre constante a diferencia de la velocidad mínima, la cual aumenta su valor de acuerdo al valor del parámetro *stepincr* en cada corrida. Esta secuencia se repite hasta haber cumplido con el número de pasos requeridos. Los parámetros utilizados son:

Aceleración (*acc*)
Velocidad máxima (*maxspeed*)
Velocidad mínima (*minspeed*)
Tiempo de flujo (*ttop*)
Tiempo de reposo (*tbot1*)
Número de escalones (*numstep*)
Incremento (*stepincr*)
Número de corridas (*runs*)
Dirección de giro 1 (*DirRot1*)
Dirección de giro 2 (*DirRot2*)

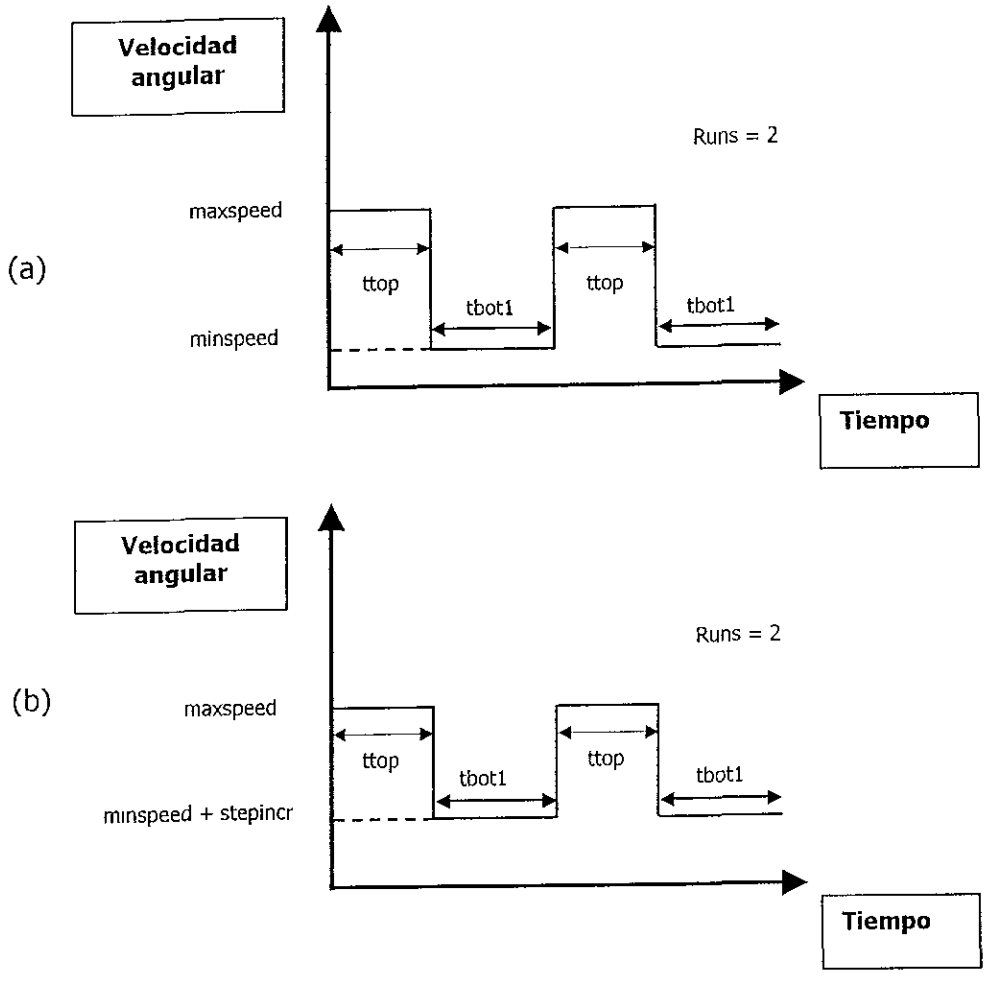


Figura 3.16. *Gráficas velocidad angular en función del tiempo de la rutina de Flujo de Doble Escalón, considerando una aceleración infinita. En (a) se representa la gráfica que representa dos corridas del primer ciclo en donde la velocidad baja es* minspeed. *En (b) se ejecuta el segundo ciclo en donde la velocidad mínima aumenta de valor y la máxima se mantiene constante.*

3.6.1. Limitaciones Experimentales

Para generar esta combinación de flujos se tienen que tomar en cuenta diferentes consideraciones, sin embargo, la más importante es establecer una cota superior para el cálculo de la velocidad mínima que se va incrementando en cada paso. Para lograr esto, se utiliza dentro de la programación de esta rutina el parámetro *Range* que está en función de la *minspeed*, el *stepincr* y el *numstep*, de tal manera que este valor *Range* nunca rebasa el valor de *maxspeed*. En el caso en que el usuario introduzca un valor para la velocidad menor que el establecido para la velocidad mínima, el programa invierte automáticamente los valores.

Por otra parte, se considera un intervalo relativamente amplio para definir el número de escalones (*numstep*), de tal manera que el experimentalista pueda obtener información relevante sobre el comportamiento de la birrefringencia producida por esta combinación de dos flujos.

El objetivo de ejecutar varias corridas de un mismo ciclo radica únicamente en lograr cierta repetibilidad en los flujos para así facilitar una buena adquisición de datos. Esta repetiblidad está limitada a un número máximo de 10 corridas.

Finalmente, con la intención de que la suma de la velocidad mínima (o de base) más cada incremento tenga un valor tal que genere un cambio considerable en el flujo, es necesario que el incremento sea de cuando menos 50 mini-pasos/s. Por lo tanto el programa optimiza los valores introducidos de tal manera que se garantiza esta condición.

3.6.2. Desarrollo del Código

Las Funciones de Usuario que se diferencian de las otras dos rutinas son *Declare*, *Values* y *sendmotor*. En la función *Declare* se establecen como variables locales *temp*, *Init_numstep*, *speed* y *Range*, y como en los casos anteriores, los parámetros del experimento están definidos como variables globales. Los valores que solicita y analiza la función *Values* son *Acceleration*, *maxspeed*, *minspeed*, *runs*, *flow time*, *Rest time*, *stepincr*, *numstep*, *DirRot* del tiempo de flujo y *DirRot* del tiempo de reposo. En la podemos observar una estructura similar a la de la función *Values* para el Flujo de Doble Paso. Los Objetos de Usuario que se utilizan para evaluar y cambiar los valores de los parámetros de entrada, en caso de ser necesario, se explican a continuación:

- En *mspeed* se evalúa la posibilidad de que *maxspeed* sea menor a *minspeed*, en cuyo caso invierte los valores y presenta el siguiente aviso: *WARNING: incompatible min & max speeds!*.
- *Run* es un Objeto de Usuario que tiene como tarea cambiar el valor del parámetro *runs* si el valor de éste introducido por el usuario es mayor a 5.
- Steps utiliza **numstep** = 20* Ordinal (numsteps>=20) + numsteps * Ordinal (numsteps<20) para el análisis del número de escalones (*numstep*), de tal manera que su valor siempre sea menor o igual a 20. Además el objeto *Steps* asigna este mismo valor a *Init_numsteps* que se utiliza posteriormente.

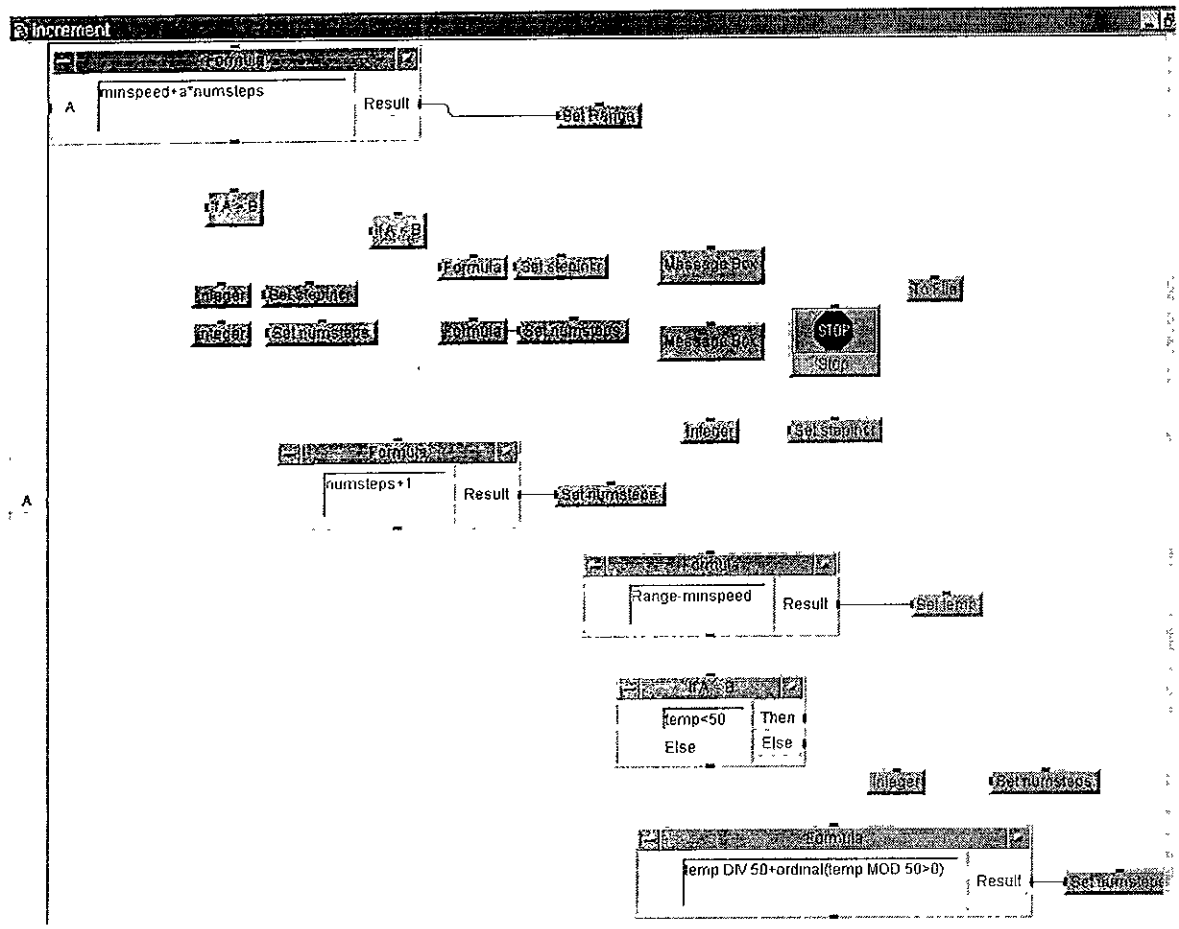


Figura 3.17. *Vista detalla del Objeto de Usuario* Increment *de la rutina de Flujo de Doble Escalón.*

- El objeto llamado *Increment* realiza dos tareas. En primer lugar, calcula el valor de la variable local *Range*, que representa el valor máximo que alcanza la velocidad baja que se va incrementando y posteriormente evalúa si el *stepincr* es menor a 50. En caso de que esta condición resulte verdadera el programa genera un aviso (*WARNING: Speed increment is too small; Instead, stepincr is being used*) y calcula el número de pasos necesarios para obtener el mismo valor de *Range* calculado en un inicio, como se muestra en la Figura 3.17.
- *Change Values* es la función encargada de generar el ciclo para analizar y cambiar el valor de *Range* (en caso de que éste sea mayor a *maxspeed*) disminuyendo el número de pasos. Si *numsteps* resulta menor que uno, el programa detecta el error, manda el siguiente aviso *FATAL ERROR: Inconsistent velocities for steping motor!* y

asigna el valor de cero para el número de pasos y el de la velocidad máxima para *Range*.

- La función *Display* al igual que en las otras rutinas, presenta los valores de entrada y los aceptables, pero además muestra en pantalla el valor de *Range*, que aunque no es un valor establecido por el usuario, brinda importante información.

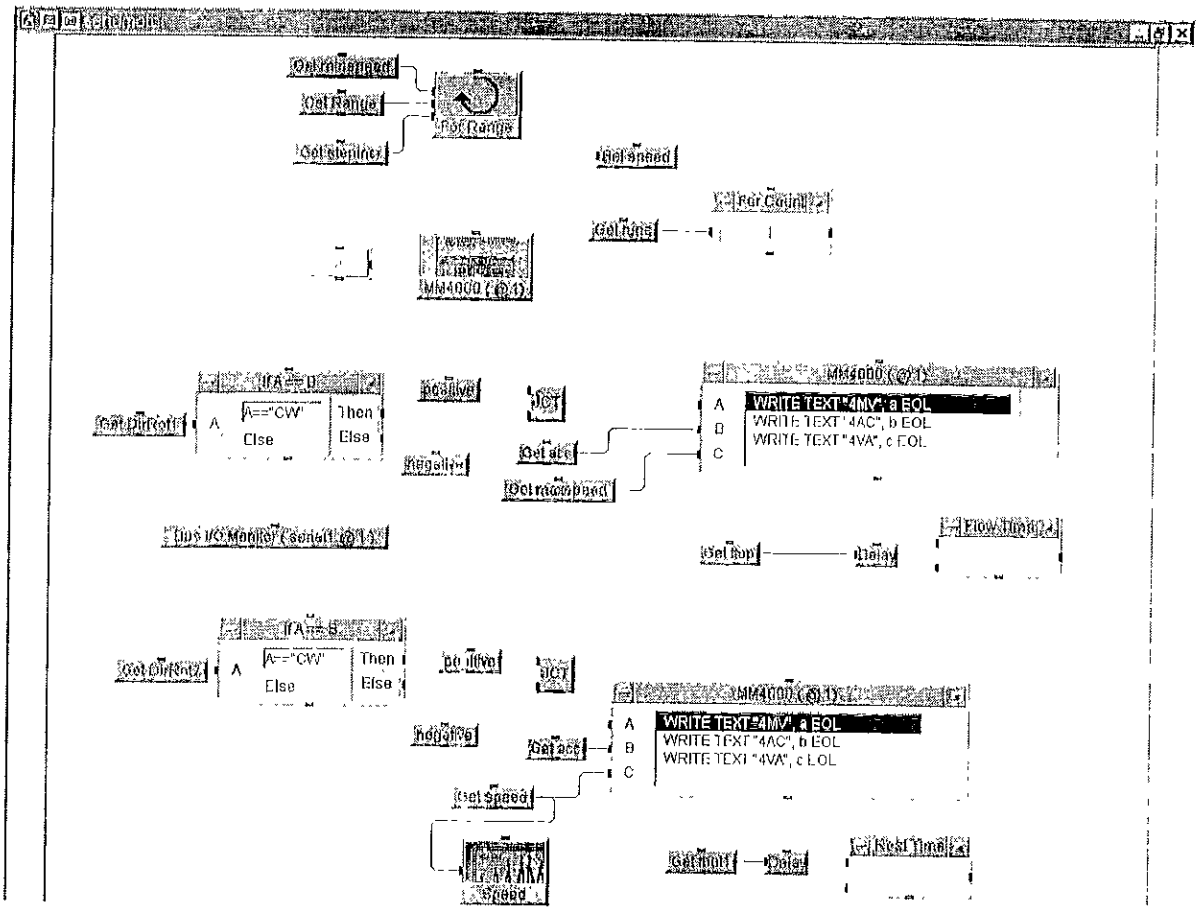


Figura 3.18. *Vista detalla de la función* sendmotor *de la rutina de Flujo de Doble Escalón.*

Para finalizar, esta rutina ejecuta la función *sendmotor* que se muestra en la Figura 3.18, que de manera similar con la rutina de Inicio y Cese de Flujo, realiza tres tareas primordiales. La primera consiste en realizar el ciclo iterativo necesario para calcular, en cada paso del experimento, los valores de la velocidad mínima. Esto lo hace por medio del objeto *For Range* al cual se le especifican los valores *minspeed, Range* y *stepincr*. Cada valor de la velocidad mínima que se va incrementando se asigna a la variable local *speed*. La segunda tarea consiste en enviar al controlador los comandos necesarios para lograr el

movimiento del molino y por último *sendmotor* contiene el código para producir la interfaz que muestra al usuario la graficación de las velocidades mínimas a lo largo del experimento y le permite la observación de los tiempos *ttop* y *tbot1*.

3.6.3. Presentación de Resultados

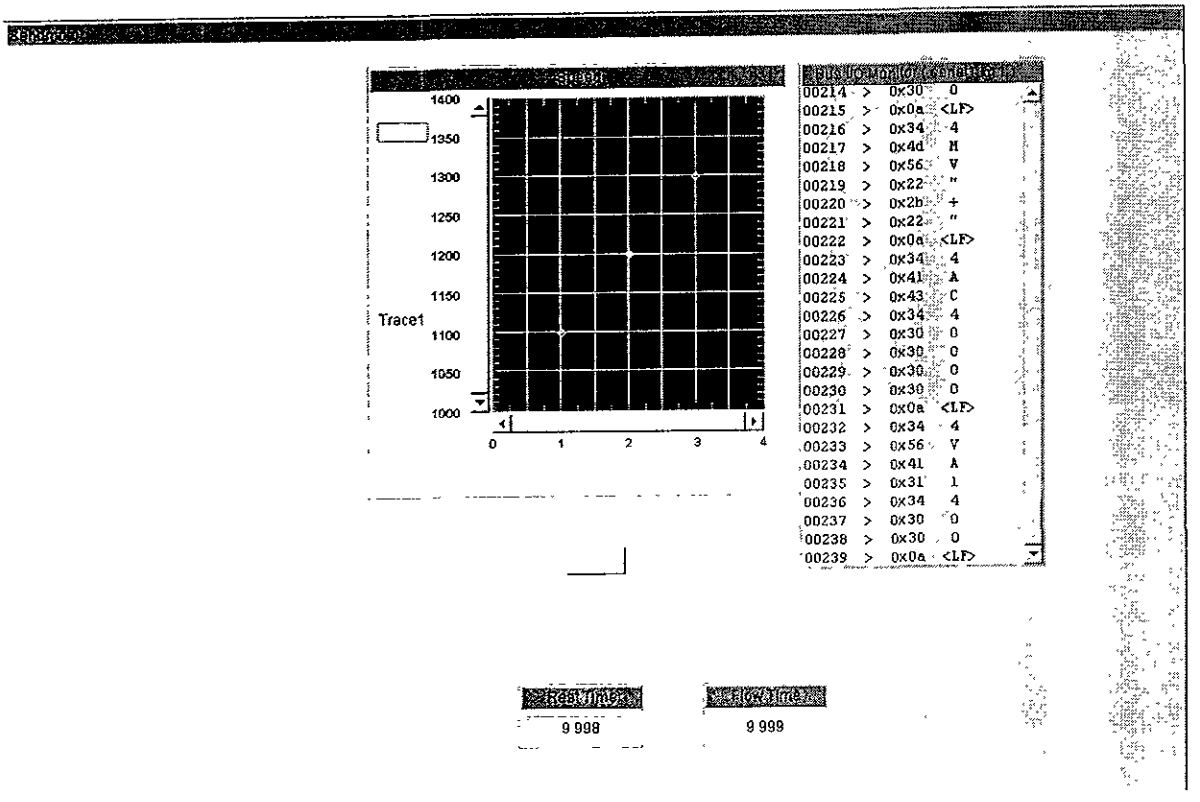


Figura 3.19. *Interfaz de usuario que grafica la evolución de la velocidad mínima que se va incrementando (cuando* run *= 1) y permite monitorear el* Bus *E/S del puerto serial RS-232, el tiempo del flujo a velocidad máxima y el tiempo del flujo a velocidad mínima.*

Las interfaces que este código genera para que el usuario pueda introducir valores a los parámetros del experimento y compararlos con los que el programa optimiza, son prácticamente equivalentes a las descritas en las rutinas de Estado Estacionario e Inicio y Cese de Flujo.

Finalmente, cuando comienza el giro de los rodillos, aparece en pantalla la interfaz que muestra la evolución de la velocidad mínima y cómo se va incrementando en cada ciclo. En este caso, no aparece la gráfica de la velocidad máxima porque su valor nunca cambia. En la Figura 3.19 se observan tres puntos que representan los primeros tres

incrementos en la velocidad mínima. La diferencia de ésta gráfica con la de la Figura 3.20 es simplemente que en ésta última se ejecutan tres corridas en vez de una, como es el caso del experimento graficado en la Figura 3.19. Esta interfaz también muestra el monitoreo del tiempo que transcurre en cada flujo.

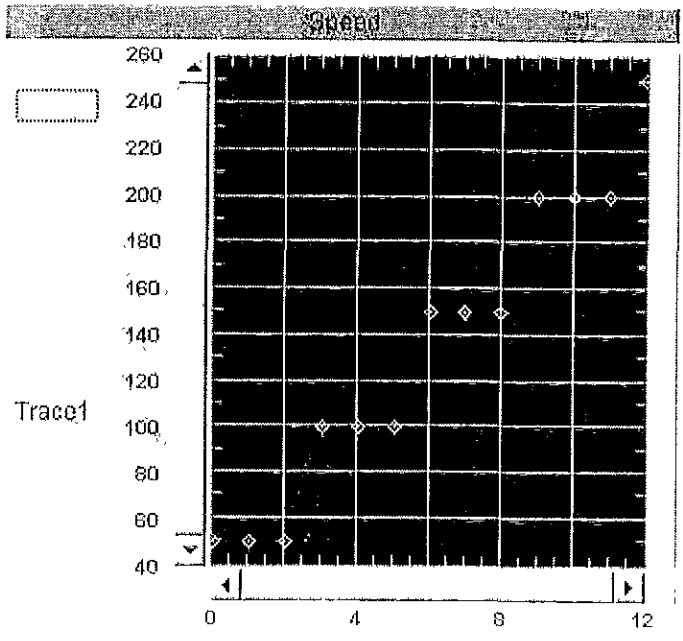


Figura 3.20. *Gráfica que muestra la forma en la que se incrementa el valor de la velocidad mínima cuando se ejecutan tres corridas* (run = 3).

Los comandos enviados al controlador son los mismos que en la rutina de Inicio y Cese de Flujo, y se muestran en la Transacción Directa localizada en la parte baja de la Figura 3.18.

CONCLUSIONES

En este trabajo se presenta una descripción completa de la programación desarrollada para el control del motor de pasos que genera el movimiento de un molino de dos rodillos para el experimento de Birrefringencia Bicolor Inducida por Flujos que se lleva a cabo en el Laboratorio de Reología Óptica del Instituto de Investigaciones en Materiales de la UNAM.

Para el control del experimento completo es necesario: (a) definir y realizar un conjunto de diferentes historias de deformación, (b) controlar la temperatura del líquido bajo estudio, (c) contar con la electrónica para la conversión y amplificación de las señales luminosas provenientes del arreglo para la determinación de la Birrefringencia Inducida por Flujos y (d) controlar por medio de la computadora la fuente de luz azul y verde. Nuestro trabajo consiste en generar los códigos necesarios para la primera de las cuatro partes del experimento que se mencionan con anterioridad, para lo cual se utilizó como base la programación que con propósitos equivalentes existía en PASCAL, con el Sistema Operativo *System P* y procesadores Motorola 68030. Una parte importante del trabajo ha consistido en proponer una solución al día en cuanto al lenguaje de programación, el sistema operativo así como el procesador maestro. Ello conlleva una mayor estabilidad para los códigos generados, pero requiere conocer en detalle los ambientes antiguos así como las equivalencias para un sinnúmero de dispositivos y elementos computacionales, electrónicos y de algoritmos en el nuevo ambiente. Asimismo, el incorporar herramientas actualizadas ofrece la posibilidad de generar una importante riqueza de flujos con mayor precisión. Este es el escenario que se ha incorporado en este trabajo, de manera que futuras mejoras o ampliaciones del mismo resulten factibles, con un menor esfuerzo al ya realizado.

Las rutinas (flujos) que pueden generarse son el flujo estacionario, arranque y paro súbito de flujo estacionario y flujo en dos escalones. Asimismo se han hecho las pruebas básicas para deformaciones en pasos dobles, lo cual representa una posibilidad nueva hasta ahora no explorada. El esquema presente, dado que puede modificar las aceleraciones instantáneamente, aun cuando el sistema está en movimiento permite la

realización de flujos oscilatorios que no se exploró en la práctica y que queda como tarea por realizar.

Como se menciona en el Capítulo 2, el lenguaje de programación gráfica que se utiliza es el HP-VEE 4.0. Este lenguaje permite ahorrar una gran cantidad de tiempo de programación y permite realizar una mejora considerable en la interfaz con el usuario, puesto que ahora se utiliza un sistema de ventanas y cuadros de diálogo para la introducción de datos y la presentación de gráficas de resultados. Además resulta relativamente más sencilla la programación de la transmisión de datos a cualquier interfaz de comunicación y el almacenamiento de datos y resultados en archivos. Una de las limitaciones en cuanto al lenguaje de programación es que los archivos ocupan más espacio en memoria, puesto que el HP VEE es gráfico y por tanto de más alto nivel que el PASCAL.

El problema actual es la necesidad de convertir todos los códigos realizados anteriormente en PASCAL al lenguaje de programación gráfica HP-VEE, puesto que sólo se han realizado en este lenguaje los programas que generan las historias de deformación. La conversión de los códigos es imperativa debido a que el equipo, el sistema operativo y el lenguaje de programación utilizados anteriormente resultan ya obsoletos.

Este experimento presenta la facilidad de que puede utilizarse tanto un motor de pasos como un servomotor de CD y el controlador del motor permite cambiar fácilmente de una tecnología a otra. Sin embargo, debido a que el motor de pasos ya había sido adquirido, y atendiendo a la necesidad de utilizar los dispositivos disponibles en el Laboratorio, utilizamos un motor de pasos UE73PP con codificador sujeto a un control de lazo cerrado. Los motores de pasos pueden utilizarse en un simple sistema de control de lazo abierto, que es generalmente adecuado para sistemas que operan a bajas aceleraciones con cargas estáticas, ya que si el motor de pasos es sobrecargado en un sistema de lazo abierto, pierde la posición (pierde pasos) y el sistema se reinicializa. En cambio, si el sistema se somete a aceleraciones altas, particularmente si se presentan cargas variables, es esencial utilizar un lazo de control cerrado.

Una de las bondades para el flujo con esta nueva configuración es que tanto para la aceleración como para la velocidad se tiene una resolución del motor de 1 mini-paso/s^2 y de 1 mini-paso/s respectivamente, y con la anterior configuración la resolución (ver Apéndice A) era de 40 pasos/s para la velocidad y de 500 pasos/s^2 para la aceleración.

Razón por la cual, a este sistema de control se la ha denominado de alta resolución, ya que permite una resolución 400 veces mayor para el caso de la velocidad y de 500 veces mayor para la aceleración. Esta resolución es necesaria bajo condiciones experimentales frecuentes, las cuales a priori no se conocen, pero que afectan los resultados experimentales. Por ello, la precisión de la generación de desplazamientos de ± 1 mini-paso/s es deseable y algunas veces necesaria.

En lo que se refiere a la transmisión de datos entre la computadora y el controlador del motor de pasos se emplea un protocolo de comunicación RS-232 de 9-pins, por lo que se utiliza el puerto serial con una razón de transmisión de 9600 bytes/s. Sabiendo que un protocolo IEEE-488 transmite *byte-serial/bit-parallel* a diferencia del protocolo RS-232 que transmite únicamente *byte-serial* y que el IEEE-488 tiene la posibilidad de alojar hasta 15 instrumentos distintos y el RS-232 solamente uno, podemos concluir que esta razón de transmisión es más lenta y se podría mejorar utilizando una interfaz IEEE-488 que la incrementaría hasta 1 Megabyte/s. Sin embargo, uno de los problemas que tuvimos fue la incompatibilidad entre los protocolos de comunicación de las tarjetas del controlador del motor y de la computadora, que creemos se debió a la diferencia de años entre estas interfaces. A pesar del inconveniente, el tiempo actual que tarda en enviar los comandos al controlador, así como el que ocupa en realizar una operación es del orden de milisegundos, similar a lo que se lograba con la configuración anterior.

Los valores de los parámetros *Kp, Ki, Kd* y *FeedForward* del algoritmo digital PID que el controlador establece por default son de uso común, seguros, libres de oscilación y minimizan el error, pero para lograr el mejor funcionamiento del sistema para cada aplicación se deben de modificar de acuerdo a la carga, aceleración, etc. Para el caso específico de nuestra aplicación no se cambió ningún parámetro puesto que hasta que el molino de dos rodillos no sea adaptado, no es necesario cambiar los valores por default. Además, el controlador MM4000 permite monitorear la respuesta del sistema por medio del modo *Trace*, en el cual el controlador es capaz de registrar la posición real y la posición deseada simultáneamente y los valores pueden ser actualizados cada 0.0005 segundos. El error absoluto de posición para el sistema sin carga es de 4 mini-pasos, lo que nos da mayores elementos para mantener los parámetros por default.

Este trabajo puede tener posibles mejoras como las que se mencionan anteriormente y es necesario ampliarlo con nuevas rutinas (flujos), pero la versatilidad que

se logró en el programa permite realizar estos cambios con sencillez y rapidez. Gran parte del trabajo realizado es independiente del motor o controlador, por lo que la programación efectuada puede ser de utilidad para una gran variedad de otras aplicaciones. Además, es factible incorporar con facilidad nuevos controladores y motores, lo que representa una mayor flexibilidad y potencial de uso de esta programación.

APÉNDICE A

Conceptos Básicos

- **Error**: se refiere al error instantáneo percibido por el controlador, que es una comparación entre la trayectoria ideal y la posición de retroalimentación del codificador.
- **Exactitud**: es una variable que representa la desviación entre el valor obtenido y el valor requerido. Empezando de un punto de referencia, ordenamos al controlador moverse cierta distancia, cuando el movimiento es completado, medimos la distancia que recorrió con un instrumento externo preciso. La diferencia representa la **exactitud** en el posicionamiento para el caso particular del movimiento.
- **Resolución** es el movimiento más pequeño que el controlador intenta realizar. Para todos los motores de CD y para cualquier motor de pasos estándar soportados por el controlador, la resolución está dada por el codificador. Tomando en cuenta que el lazo de control es digital, la **resolución** puede ser el incremento de posición mas pequeño que el controlador puede manejar.
- La **repetibilidad o precisión** es la máxima variación en el posicionamiento cuando ejecutamos el mismo perfil de movimiento, es decir cuando la misma secuencia de movimiento es repetida un buen número de veces. Es un error relativo entre movimientos idénticos.
- La **velocidad máxima** que se puede utilizar en un sistema de control de movimiento depende del dispositivo disponible (el motor ha utilizar) y del *driver*, y corresponde a una velocidad límite menor que aquella que el *driver* y el motor pueden soportar. De esta manera, la velocidad máxima establece una cota superior para un uso confiable y duradero de los componentes electromecánicos y electrónicos involucrados.
- La **velocidad mínima** depende del sistema de control, pero también trata de establecer una velocidad de regulación aceptable, que en primera instancia implica que el controlador asigne el menor incremento de velocidad capaz de realizar. La resolución del codificador determina el tamaño del incremento de movimiento y después, la aplicación determina un límite en el máximo de la velocidad.

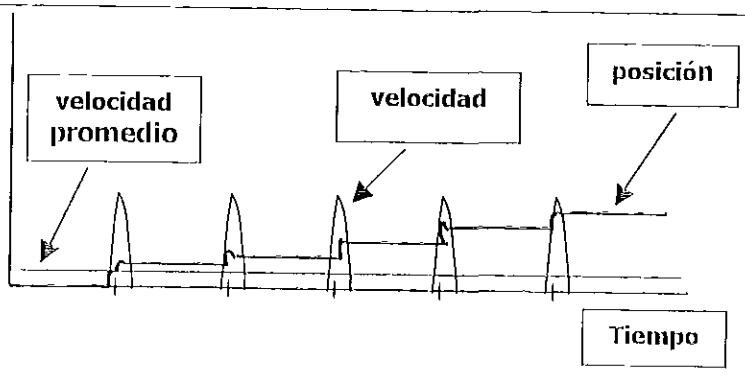


Figura A.1. *Posición, velocidad y velocidad promedio.*

La velocidad promedio es baja, pero el pico de velocidad (o velocidad de rizo) es muy alto y dependiendo de la aplicación, éste puede ser aceptable o no. Cuando la velocidad aumenta, el pico disminuye y la velocidad se vuelve más suave. Esto ocurre comúnmente en motores de pasos. El ruido típico se debe a una rápida transición desde una posición a otra y el pico de velocidad, en este caso, es significativamente alto (ver Figura A.1).

Aunque el controlador realice un trabajo perfecto en la primera ejecución al obtener un error de cero, las imperfecciones mecánicas (fricción, variación de rizo, etc.) generarán un pico de velocidad que puede generar problemas en la regulación de velocidad.

Depende de la aplicación específica el que un motor sea tecnológicamente mejor que otro. En lo que concierne al controlador, el motor de pasos es ideal para obtener un buen promedio en la velocidad de regulación ya que sigue inherentemente de manera precisa la trayectoria deseada. El único problema es el pico provocado por su mismo sistema de pasos. En cambio, lo que puede hacer el mejor controlador para un motor de CD es, aproximarse al desarrollo de un motor de pasos en cuanto a la regulación de velocidad, pero tiene la ventaja de que reduce significativamente el pico de velocidad, por su propia estructura, y a través de implementar un lazo PID.

- La **aceleración máxima** es un parámetro complejo que depende en gran medida, tanto del sistema de control de movimiento como de la aplicación que se requiere. Para motores de pasos, lo principal es no perder pasos (o la sincronización) durante la aceleración. A la par de la ejecución del motor y del control del mismo, la carga inercial juega un papel importante [12].

APÉNDICE B

Descripción de Comandos Básicos del Controlador MM4000

En este apéndice se describen algunos de los comandos del controlador MM4000 que se utilizaron para mover el motor de pasos UE73PP, así como su sintaxis.

Sintaxis

Figura B.1. *Formato de los comandos del controlador.*

El formato general de los comandos es un doble caracter nemotécnico (AA), los cuales pueden ser letras tanto mayúsculas como minúsculas. Dependiendo del comando, se especifica el parámetro *xx* precediendo al comando y el parámetro *nn* al final de éste, como se muestra en la Figura B.1.

Los comandos se ejecutan línea por línea. Una línea consiste de uno o varios comandos y el controlador interpreta estos comandos en el orden en el cual son recibidos para después ejecutarlos. Usualmente tarda algunos microsegundos en realizar esta operación.

En una línea de comandos se permiten hasta 110 caracteres como máximo. Aquellos que se encuentran en una misma línea deben separarse por una coma (,) o por punto y coma (;). Cada línea debe contar con un terminador de línea, el cual se define en el modo *GENERAL SETUP* (Configuración general). El controlador soporta todas las combinaciones de cambio de línea, ya sea LR (Line Feed) o CR (Carriage Return): LF, CR, LF/CR y CR/LF.

Comandos

- **MO** : Este comando enciende el motor.
- **MF** : Apaga el motor.

- *xx***MV+** / *xx***MV–** : Inicia un movimiento continuo (infinito), en sentido horario (+) o en sentido antihorario (–).
- *xx***PA***nn* : Inicia un movimiento absoluto. El eje de movimiento *xx* se moverá con la aceleración y velocidad predefinidas a la posición específica *nn*.
- *xx***PR***nn* : Inica un movimiento relativo. El eje de movimiento *xx* se moverá con la aceleración y velocidad predefinidas *nn* unidades de la posición actual.
- *xx***AC***nn* : Define el valor de la aceleración o desaceleración (*nn* en unidades de mini-pasos/s^2), para el eje *xx*. El valor por default es 40 000, debido a que es la mitad del valor máximo que se le puede asignar al motor de pasos.
- *xx***VA***nn* : Establece el valor *nn* de la velocidad en mini-pasos/s para el eje *xx*. El valor por default es 10000 mini-pasos/s.
- *xx***DV** : Este comando lee la velocidad del eje *xx* definida anteriormente por **VA**.
- *xx***ST** : Detiene un movimiento en progreso en uno de los ejes. Si el parámetro *xx* tiene un valor de 0 o ningún valor, el movimiento en todos los ejes se detiene.

Nota: los comandos **AC** y **VA** se ejecutan en forma inmediata, es decir, que los valores tanto de aceleración como de velocidad se cambian en cuanto se procesa el comando, aún cuando el movimiento esté en progreso.

BIBLIOGRAFÍA

[1] MACOSKO, Ch., *Rheology principles, measurements and applications*, Wiley-VCH, 1993.

[2] BAUTISTA, E., *Evaluación de la técnica de Anisotropía Bicolor Inducida por Flujos para muestras con dicroismo, birrefringencia o depolarización residual*, Tesis de Físico, UNAM, 1994.

[3] GEFFROY, E., *Birefringence of Polymer Solutions in Time Dependent Flow*, PhD Thesis, California Institute of Technology, 1990.

[4] JIMENEZ, J.P., *Diseño del Tren Óptico para el experimento de Birrefringencia Bicolor de alta Resolución*, Tesis de Ingeniero Mecánico, FI-UNAM, 2000.

[5] JANESCHITZ-KRIEGL, H., *Polymer Melt Rheology and Flow Birefringence*, Springer-Verlag, Germany, 1983.

[6] HOCHSTEIN, M., *Birrefringencia Bicolor de Alta Resolución*, Tesis de Físico, UNAM, 1997.

[7] REYES, M. & GEFFROY, E., *A Corotating Two-Roll Mill for Studies of Two-Dimensional, Elongational Flows with Vorticity*, en Physics of Fluids to appear in 2000.

[8] REYES, M. & GEFFROY, E., *Study of low Reynolds number hydrodynamics generated by symmetric corotating two-roll mills*, en Revista Mexicana de Física, No.46, 2000.

[9] MARIN, C., *Formalismo unificado de anisotropía óptica para la descripción de materiales no-homogéneos*, Tesis de Físico, UNAM, 1999.

[10] CORONA, C., *Medición de Señales luminosas de alta resolución para estudios de anisotropía de flujos en sistemas poliméricos,* Tesis de Físico, UNAM, 1997.

[11] NACHTIGAL, *Instrumentation and Control*, Wiley, 1990.

[12] NEWPORT, *MM4000-4 Axis Motion Controller/Driver, User's Manual*, Newport, 1995.

[13] NEWPORT, *MM4000-4 Axis Motion Controller/Driver, Addendum*, Newport, 1995.

[14] OGATA, K., *Ingeniería de control moderna*, Prentice Hall, México, 1993.

[15] SÖDERSTRÖM, T. *Feedforward, correlated disturbances and identification*, en Automatica, editorial Elsevier, Suecia, No. 35, 1999, p. 1567-1571.

[16] MENON, K., KRISHNAMURTHY, K., *Control of low velocity friction and gear backlash in a machine tool feed drive system*, en Mechatronics, USA, No. 9, 1999, p. 33-52.

[17] HUBERT, Ch., Electric Machines. *Theory, operation, applications, adjustment and control*. Macmillan, USA, 1991.

[18] HELSEL, R., *Visual Programming with HP VEE*, Third Edition, Prentice Hall, 1998.

[19] HEWLETT PACKARD, *HP VEE. Advanced Programming Techniques*, HP, 1998.

[20] HEWLETT PACKARD, *Controlling Instruments with HP VEE*, HP, 1998.

[21] SCHNEIDER, M, *An introduction to programming and problem solving with Pascal*, Second Edition, Wiley, 1982.

[22] HEWLETT PACKARD, *Pascal 3.2 Workstation System. Programming and Configuration Topics*, volume 2, Hewlett Packard, 1991.

[23] McDONALD, A. & LOWE, H., *Feedback and Control Systems*, Prentice-Hall, 1981.