UPC ESAII
Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial

EUPVG

Tecnologia de sistemes de control

Introducción y aplicación en LabVIEW para el control de un proceso contínuo mediante PID

LabVIEW™ 5.0
Graphical Programming for Instrumentation

Alumnes : Sanchez Rallo, Joan
Sanchís Aicart, Miquel Angel
Sanz Carnicero, Esteban

Director Projecte : Pere Ponsa -ESAII

Curs 1998/99 - 2Q

ÍNDICE

1 Introducción al LabVIEW

LabVIEW *es un software suministrado por la empresa National Instruments orientado a la programación de instrumentos virtuales en un entorno gráfico.*

1.1 Instrumento virtual (VI)

A un instrumento virtual lo podemos definir como un módulo de software que simula el funcionamiento de un instrumento físico. Este software debe poseer un hardware controlado por el ordenador que le permita acceder a los datos externos al instrumento. Este hardware podría ser una tarjeta de adquisición de datos, una tarjeta DSP, o un instrumento controlado mediante GPIB (Global Purpose Interface Bus), mediante RS-232 o mediante el bus XVI. De este modo el usuario del instrumento virtual puede observar en la pantalla de un computador los datos recibidos por el instrumento. Al mismo tiempo puede analizarlos con un programa realizado por el mismo LabVIEW. De los resultados del análisis de los datos y mediante el hardware el programador de LabVIEW puede llegar a controlar un sistema. [1]

1. 2 Entorno gráfico

Cuando hablamos de un entorno gráfico nos referimos tanto a una representación del panel frontal del instrumento virtual como de la programación en si. Para visualizar los datos del instrumento utilizamos una serie de objetos como puede ser leds, botones, pantallas y demás tipos de controles o visualizadores. Existen tanto indicadores como controladores de muchos tipos, a los cuales se les pueden cambiar el tamaño, el color, la sensibilidad, y la escala.

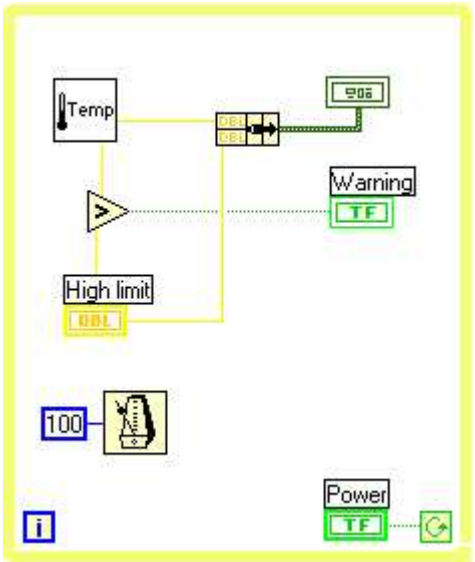


Figura 1.1 Diagrama de bloques (Tipo)

La Programación gráfica se consigue gracias a unos iconos o bloques que representan tanto cifras como operaciones. Estas cifras pueden ser constantes, datos introducidos por el panel de control, o datos capturados por el hardware del ordenador. En cuanto al tipo de operaciones que se pueden realizar son las típicas de cualquier lenguaje de programación más unas cuantas propias del LabVIEW. Todos estos bloques se colocan en el diagrama de bloques y se conectan entre si mediante un cable virtual.

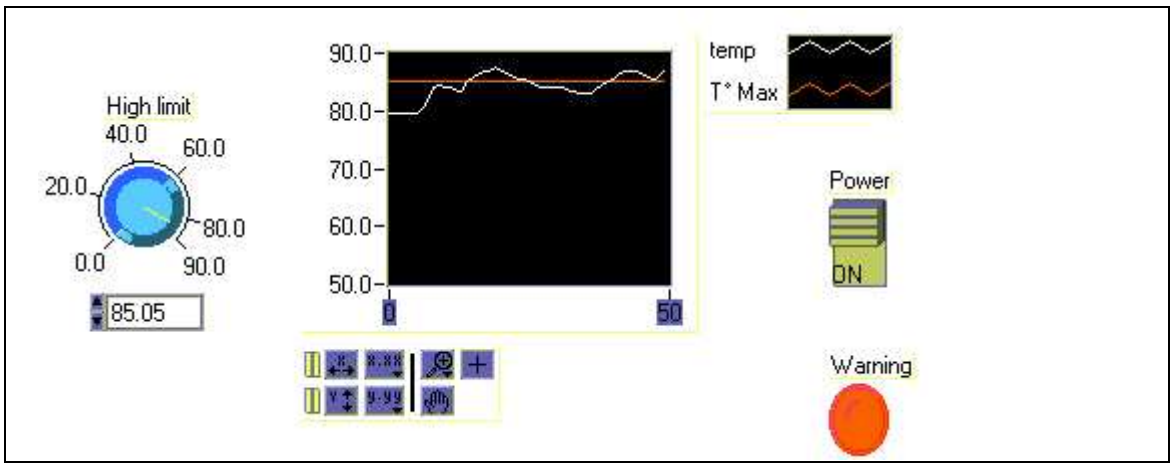


Figura 1.2 Panel frontal (Tipo)

1. 3 Otros lenguajes de programación

En general al hablar de un lenguaje de programación o un programa informático estamos pensando en una plataforma de tipo texto, es decir , las órdenes se basan en una serie de palabras claves denominadas órdenes o comandos; ejemplo de ello nos podemos remitir a los lenguajes como C, PASCAL, COBOL, BASIC.

Si hacemos referencia a los lenguajes COBOL y BASIC se puede apreciar que fueron programas no diseñados para mejorar el trabajo de los programadores, ni para mejorar la fiabilidad del código producido, ni siquiera para aumentar la velocidad con la que se pudiera escribir el código. En el caso concreto de COBOL fue diseñado para permitir que los no programadores pudieran leer y presumiblemente (aunque improbable) comprender el programa. En el caso de BASIC fue creado esencialmente para permitir a los no programadores programar una computadora para resolver problemas relativamente sencillos.

El lenguaje C fue inventado e implementado por primero vez por Dennis Ritchie en un DEC PDP-11 usando unix como sistema operativo. Este lenguaje de programación nació en la década de los setenta como desarrollo del lenguaje B (Ken Thompson). En esta misma década nació el microprocesador y gracias a él se desarrollaron muchas implementaciones de C. Siendo este lenguaje considerado de nivel medio, porque combina elementos de lenguajes de alto nivel con el funcionalismo del lenguaje ensamblador. Hay que hacer especial mención a la virtud de su portabilidad, es decir, significa que es posible adaptar el software escrito para un tipo de computadora en otra. En contra podemos mencionar que tiene una pésima librería gráfica y no está orientado a trabajar con objetos (OOP). [5]

1.4 Evolución del LabVIEW

LabVIEW v.1.0 era un lenguaje basado en el flujo de datos entre diversos componentes, junto con un control del flujo mediante un código gráfico desarrollado sobre una plataforma Macintosh.

Debido a las características de esta primera versión los usuarios se veían obligados a hacer correr sus aplicaciones de cierta complejidad en grandes máquinas, como la mencionada anteriormente. Frente a esta dificultad LabVIEW v.2.0 optó por hacer uso de un compilador, frente al intérprete de la versión previa.

> Para entender esta diferencia pasaremos a explicar brevemente los términos *compilador e intérprete* que se refieren a la forma en la que se ejecuta un programa. Ambos son simplemente programas sofisticados que trabajan sobre el código fuente del programa, por ejemplo podemos citar BASIC que es de carácter interpretado frente al C que es normalmente compilado. Cabe mencionar que la forma en que se ejecuta un programa no viene definida por el lenguaje en que se haya escrito.
>
> En el caso de los interpretes se lee el código fuente del programa línea a línea, realizando las instrucciones específicas contenidas en esa línea. Por el contrario el compilador lee el programa entero y lo convierte a *código objeto,* que es una traducción del código fuente del programa a una forma que puede ser ejecutada directamente por la computadora. El código objeto también se puede denominar código binario o código máquina.

La mejora propuesta por National Instruments para LabVIEW v.3.0 fue hacer los programas del mismo más portables, con lo que se tuvo que procurar hacer más independiente el código del programa frente al procesador y del sistema operativo de la plataforma. *Corriendo en la actualidad la versión cinco sobre los sistemas operativos WindowsNT Windows95 PowerPC* [3]

2 Control PID

2.1 Introducción.

El control PID es el algoritmo de control más empleado en la industria con más diferencia, ya que proporciona una gran flexibilidad, no sólo en algoritmo de control, sino también en lo que se refiere al tratamiento de la señal de referencia. Presenta diferentes estructuras de control y algoritmos: Es la base teórica de todos los métodos de sintonía automática más habituales en los reguladores industriales.

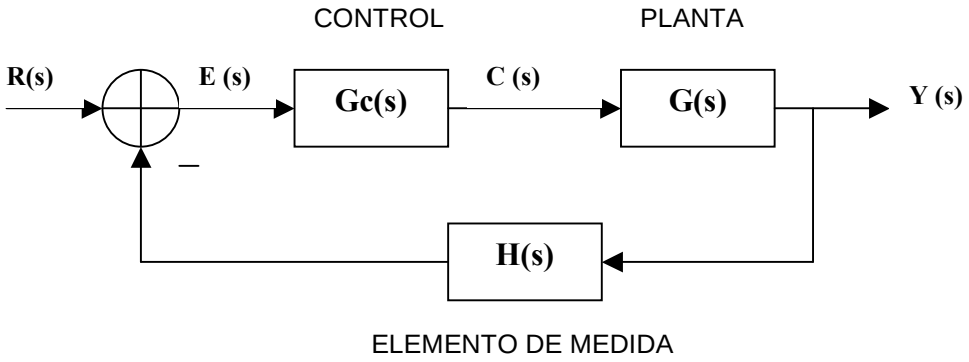


Figura 2.1 : Lazo de control de una planta

Este tipo de controladores no se ha visto desplazado por los modernos algoritmos de control fruto del desarrollo de las áreas electrónica e informática. El motivo radica en las dos grandes ventajas que proporcionan, como la robustez y las intuitivas relaciones entre sus parámetros y la respuesta del sistema; ello no quiere decir que halla quedado al margen de los avances en las áreas tecnológicas de control, sino, que debido a su flexibilidad el control PID se ha podido beneficiar de ello. [2][4]

PID- El acrónimo corresponde a las acciones que encierra este algoritmo de control, como son la acción proporcional, la acción integrativa y la acción derivativa.

$$u(t) = K_p\left(e(t)+\frac{1}{T_i}\int e(t)dt + T_D\frac{de(t)}{dt}\right) \quad \text{Formula 1}$$

2.1.1. Acción de control proporcional.

Este controlador genera a la salida una señal de control que es proporcional a la señal de error. De este modo:

U(t)=K·e(t) => U(s)=K·e(s)

U(t): señal de control

K: sensibilidad proporcional o ganancia proporcional
E(t) error

Figura 2.2 : salida de un proceso con control proporcional

Cuanto mayor es la ganancia del control proporcional mayor es la señal de control generada para un mismo valor de señal de error. Desde otro punto de vista se puede decir que para una señal de control determinada, cuando mayor es la ganancia de control proporcional, menor es la señal de error actuante. Por lo que un aumento de la ganancia del control proporcional permite reducir el error en estado estacionario hasta cierto límite. Pero teniendo en cuenta que hace al sistema más sensible a perturbaciones y menos estable. El error cometido se denomina error estacionario. En sistemas que poseen una diferencia entre el grado del denominador y el numerador mayor de dos en su función de transferencia (la mayoría) el aumento de la ganancia de control proporcional lleva generalmente a un empeoramiento de la respuesta transitoria en lazo cerrado:

- Aumento del sobreimpulso.
- Disminución del tiempo de pico.

2.1.2.Acción de control integral.

Genera una señal de control proporcional a la integral de la señal de error:

$$u(t) = \frac{K_p}{T_i} \int_o^t e\, dt \qquad \text{Formula 2}$$

La característica más importante de este tipo de control es que la acción correctora se efectúa mediante la integral del error; ello permite saber que el control integral proporciona una señal de control en función de la propia historia de la señal de error. También permite obtener una señal de control diferente de cero aunque la señal de error sea cero; e(t)=0 no implica M(t)=0.

El control integral permite obtener error estacionario nulo en un sistema de control mediante la introducción de un elemento integrador en la función de transferencia de lazo abierto.

La disminución abusiva de la constante de tiempo integral T_i puede provocar un efecto desestabilizador alargando el transitorio y obteniendo mayor sobreimpulso.

Figura 2.3 Salida de un proceso con control PI

2.1.3. Acción de control derivativa

La presencia de la acción derivativa lleva generalmente asociada una mejora de la estabilidad, lo que permite valores más elevados de la ganancia proporcional con la siguiente mejora del estacionario. Pero esta acción no es recomendable cuando la salida del proceso esta afectada de ruido ya que éste sería amplificado por la acción derivativa.

$$u(t) = K_c T_d \frac{de}{dt}$$ Formula 3

La acción de control derivativa genera una señal de control proporcional a la derivada de la señal de error. De este modo el control derivativo, obteniendo la derivada de la señal de error, "conoce" las características dinámicas de la misma (crecimiento o decrecimiento), produciendo una corrección antes de que la señal de error se haga excesiva. A este efecto se le denomina acción anticipativa.

Resumiendo la acción de control derivativa añade sensibilidad al sistema y tiene un afecto de aumento de estabilidad relativa. Sin embargo, el control derivativo no puede utilizarse en solitario ya que es incapaz de responder a una señal de error constante.

$e(t)=cte \Rightarrow U(t)=0$

En conclusión, con un control derivativo un sistema no alcanzaría nunca el estado estacionario, y el control derivativo siempre debe utilizarse en combinación con otros controles por su influencia estabilizadora mediante la acción anticipativa. [2] [4]

Figura 2.4 Salida de un proceso con control PD

2.1.3.1. Filtro en la acción derivativa.

El cálculo en la acción derivativa en la práctica es imposible de realizar, y se sustituye por una pseudoderivada. La parte derivativa se sustituye por la siguiente función de transferencia.

$$\frac{U_d(s)}{E(s)} = \frac{K_p T_D s}{\alpha T_D s + 1} \qquad \text{Formula 3}$$

Siendo **α** el factor de filtro derivativo. Este factor toma valores típicos entre 0.05 y 0.1 . Esto se entiende como un filtro de primer orden, cuya constante de tiempo es **α·Td**. La nueva acción derivativa actuará como derivada solo a baja frecuencia, y su ganancia a alta frecuencia como máximo valdrá **Kp/α** Luego, el ruido a altas frecuencias será amplificado como mucho por ese valor y no por un valor elevado como en el caso ideal. Esta acción también es llamada red de adelanto. Esto es un ejemplo de cómo tratar las funciones derivativas en un caso concreto de una estructura de control no interactiva. Para una estructura interactiva usaremos la siguiente red de adelanto:

$$\frac{U_d(s)}{E(s)} = \frac{K_p T_D s}{\alpha T_D s + 1} \qquad \text{Formula 4}$$

2.2. Estructures de control

Modificaciones del algoritmo PID

En la actualidad en el mercado se ofrecen distintas versiones del algoritmo de control PID donde cualquiera de ellas se puede asociar a uno de los tres grupos de controladores PID : No-Interactivos, Interactivos y Paralelos.

El motivo de esta nomenclatura es que en el controlador No Interactivo estándar reconocido por la ISA (Instrumentation Society of America) la constante de tiempo integral (Ti) no influye en la parte derivativa y la constante de tiempo derivativa Td no influye en la parte integral.

2.2.1 Algoritmo No Interactivo

Es el algoritmo de control por excelencia , de carácter general y reconocido por la ISA. Con esta denominación se desea manifestar que las acciones de control integral y derivativa son independientes de los parámetros de los otros controladores pues actúan cada uno de ellos sobre la misma señal de error. El controlador proporcional, sin embargo es el último en actuar. Afecta a la señal de error, a la derivada del error y a su integral. Este algoritmo es más sencillo, a nivel conceptual, que el Interactivo.

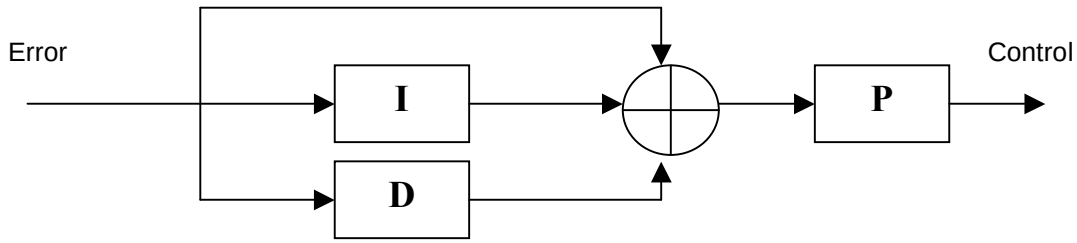


Figura 2.5 : PID no interactivo

2.2.2. Algoritmo Interactivo.

Se denomina algoritmo interactivo (serie o clásico) a aquel que cualquier modificación en uno de sus parámetros conlleva cambios en todas las constantes de tiempo, integral, proporcional y derivativa. Aparentemente ésto complica la sintonía manual, pero en realidad no sucede así. Hay motivos históricos para preferir el control interactivo: Los primitivos controladores eran más fáciles de construir en esta forma. Cuando los fabricantes de controladores cambiaron la tecnología de neumática a eléctrica-analógica y finalmente a técnicas digitales, conservaron el esquema interactivo. Esta forma además es muy utilizada en los controladores de lazo simple.

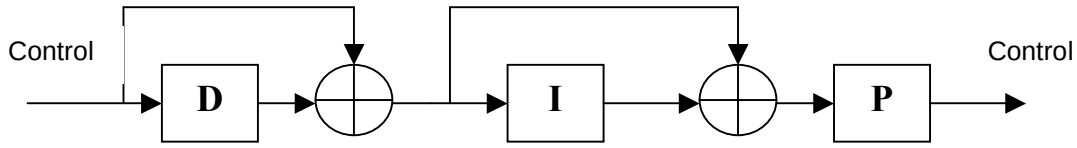


Figura 2.6 : PID Interactivo

Una ventaja de los algoritmos interactivos frente a los algoritmos no interactivos, es que los primeros tan sólo requerían dos amplificadores a diferencia de los otros que necesitan tres. Este fenómeno era debido a que la implementación de un tercer amplificador analógico más encarecía el producto en buena medida.

No obstante por la facilidad de sintonía del PID no interactivo es frecuente transformar el controlador interactivo al no interactivo con propósito de hacer la sintonía analíticamente.

2.2.3. Algoritmo Paralelo.

Es la forma más general de realizar el algoritmo PID. Las tres acciones de control actúan directamente sobre la señal de error. Es también la fórmula más flexible, pero los parámetros así obtenidos no tienen una interpretación física muy directa. Cada acción de control puede ser variada por separado sin que ello pueda inducir a cambios en las demás acciones de control.

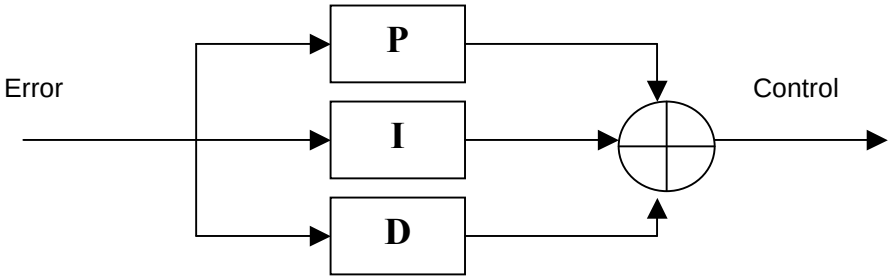


Figura 2.7 : PID Paralelo

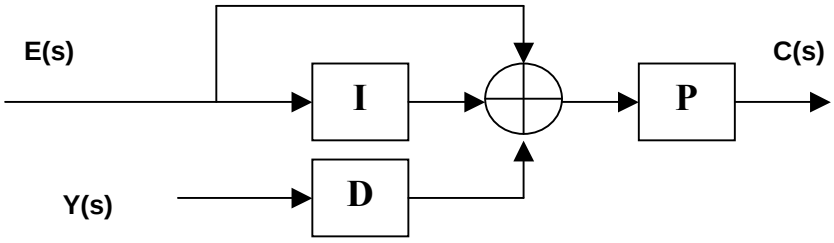
2.3. Arquitecturas de control PID

Partiendo de la estructura PID no interactiva, podemos efectuar variantes en la misma para intentar mejorar su funcionamiento. Cuando aparecen perturbaciones a la entrada de una planta que no han pasado por el control, nos encontramos que el controlador actuará en función de una señal "no actual", es decir, estamos trabajando con cierto desfase.[2] [4] Para paliar este efecto podemos recurrir a estructuras alternativas como:

2.3.1 PI-D:

La acción derivativa actúa únicamente sobre la salida del proceso y no sobre la señal de error. De esta forma se evita respecto a la estructura PID que aparezcan señales de control muy bruscas o excesivamente elevadas cuando la consigna varia bruscamente. Se ha demostrado que esta estructura no mejora de forma sustancial ni la respuesta ni la señal de control.

Figura 2.8 : Estructura PI-D



2.3.2. I-PD:

En esta estructura tanto la parte derivativa como la proporcional actúan sobre la salida y no sobre el error. Atenúan además las perturbaciones y mejoran el transitorio, puesto que la señal de control es mucho más suave.

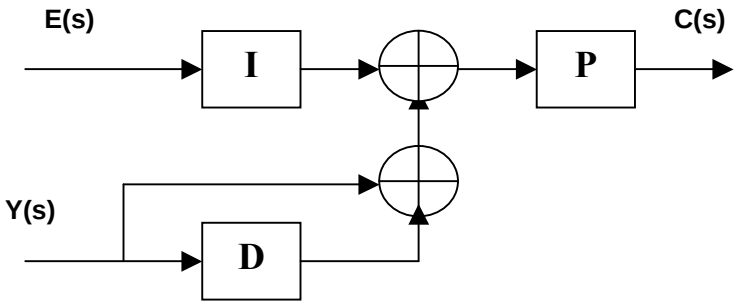


Figura 2.9 : Estructura I-PD

3 PID Software VI's

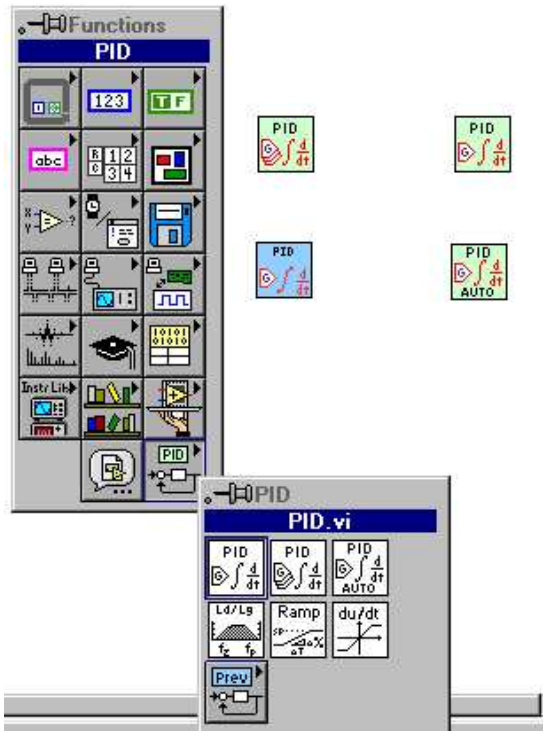


Figura 3.1 ventana e iconos de PID.VI

3.1 Introducción

Los dispositivos de control PID son de carácter analógico, requiriendo para su funcionamiento una variable de entrada denominada variable del proceso. Su salida vendrá condicionada por los parámetros de sus acciones proporcional, derivativa e integral, así como del tipo de PID usado

LabVIEW 5.0 no ofrece por defecto en sus librerías el software de PID's. Por ello es necesario instalar las librerías PID's que ofrece National Instruments en su Toolkit.

En el LabVIEW podemos encontrar estos dispositivos en la barra de funciones. Escogiéndose uno de los representados de acuerdo con las características de la aplicación

3.2 PID.VI

Este icono es el que LabVIEW proporciona como estándar.

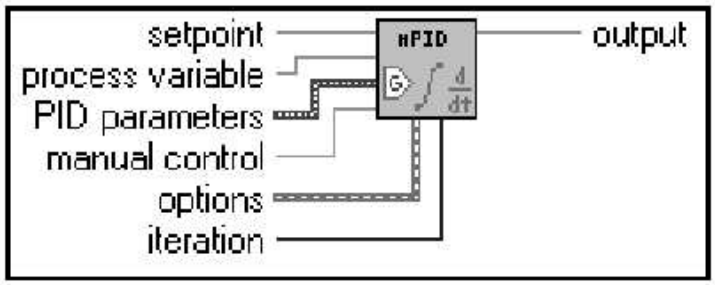


Figura 3.2 Conexionado del PID.VI

- **Setpoint** : Es la consigna del PID.
- **Process variable :** Es el valor de la variable del control de realimentación.

- **PID parametres :** Conjunto de las ganancias de las tres acciones, Proporcional, integral, derivativa.
  - **Kc** : Ganancia proporcional.
  - **Ti** : Constante de tiempo integral. Expresado en minutos. Valor 0 inhibe la acción.
  - **Td** : Constante de tiempo derivativa. Expresado en minutos. Valor 0 desactiva la acción.

- **Manual control** : Control relativo de la variable de salida.
- **Options** : Grupo de accionamientos que especifican el algoritmo de control.

  - **Sp low** : Valor mínimo para la variable del proceso y consigna. Por defecto 0.
  - **Sp high** : Valor máximo para la variable del proceso y consigna. Por defecto 100.
  - **Out low** : Valor mínimo de salida del controlador. Por defecto -100.
  - **Out high** : Valor máximo de salida del controlador. Por defecto 100.
  - **Hold (F)** : Cuando es cierto pone el controlador en modo hold. Resetea las acciones y congela las salidas.
  - **Auto (T)** : Cuando es cierto selecciona el control automático. Pone el control en modo manual cuando es falso.
  - **Pro.band (F) :** Selecciona si: el valor proporcional de los parámetros de entrada del PID es con ganancia proporcional o con banda proporcional. Por defecto es 0, seleccionando banda proporcional. Kc=100/BP. (BP=banda proporcional).
  - **Reverse Acting (T)** : Si es cierto selecciona la actuación inversa, el modo de trabajo habitual de los controladores (la salida decrece si la entrada es mayor que la consigna).
  - **Beta** : Es el nivel relativo del rechazo a las perturbaciones en el punto de consigna. Por defecto 1, que es apropiado para la mayoría de aplicaciones. Un valor pequeño entre 0-1 puede ser usado para incrementar el rechazo frente perturbaciones.
  - **Linearity :** Especifica la linealidad de la señal de error entre 0-1. El valor de 1 proporciona una respuesta plana, mientras que 0,1 aproximadamente proporciona un respuesta parabólica.
  - **Dt (s)** : Es el intervalo (en sg) en el que el VI es llamado. Si el intervalo menor o igual a 0, un temporizador interno usa un milisegundo de resolución.

- **Iteration** : Es el número de iteración del lazo de control.
- **Output** : Salida del algoritmo de control.

[3]

3.3 PID (gain scheduling).VI

La ganancia programable (Gain scheduling) se refiere a un sistema donde los parámetros controlados cambian dependiendo de las condiciones de operación medidas. Por ejemplo, la variable programable puede ser la consigna, la variable de proceso, la salida del controlador o una señal externa. Por motivos históricos la palabra 'ganancia programable' se usa incluso si los parámetros que cambian son la constante de tiempo integrativa o derivativa. La ganancia programable controla realmente un sistema cuya dinámica cambia con las condiciones de operación.

Se usa este VI para implementar un PID con ganancia programable para las ganancias proporcional, integral e derivativa.

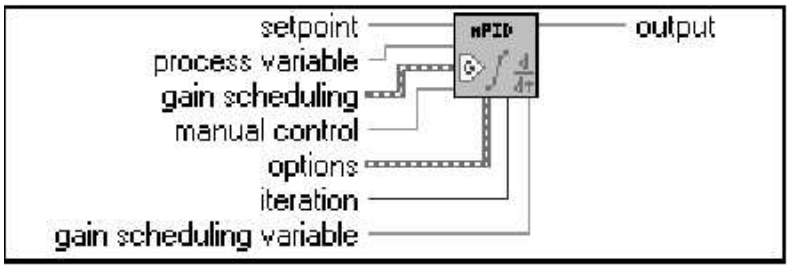


Figura 3.3 Conexionado del PID(Gain schedule).VI

- **Setpoint** : Es la consigna del PID.
- **Process variable** : Es el valor de la variable del control de realimentación.
- **Gain Scheduling** : Define el valor de las ganancias programables para los parámetros porporcional, integral y derivativo.

    - **Gain schedule** : Define el rango de la variable gain scheduling, para los respectivos parámetros del PID. Cada valor especifica el máximo valor de ese rango.

    - **PID parametres** : Conjunto de las ganacias de las tres acciones, Proporcional, integral, derivativa. Cada set de parámetros corresponde a un rango de la entrada gain schedule.

        - **PID param** :Es el conjunto de ganancias proporcional, integral y derivativa.

            - **Kc** :Ganancia proporcional.
            - **Ti** :Constante de tiempo integral. Expresado en minutos. Valor 0 inhibe la acción.
            - **Td**: Constante de tiempo derivativa. Expresado en minutos. Valor 0 desactiva la acción.

        - **Scheduling input**: Define que valor es usado para la variable "gain scheduling". Escogiendo "gain scheduing input" indica que el valor cableado a la variable de entrada "gain scheduling" debe ser usada para la ganancia programable.

- **Manual control** : Control relativo de la variable de salida.
- **Options** :Grupo de accionamientos que especifican el algoritmo de control.

    - **Sp low** : Valor mínimo para la variable del proceso y consigna. Por defecto 0.
    - **Sp high** : Valor máximo para la variable del proceso y consigna. Por defecto 100.
    - **Out low** : Valor mínimo de salida del controlador. Por defecto -100.
    - **Out high** : Valor máximo de salida del controlador. Por defecto 100.
    - **Hold (F)** : Cuando es cierto pone el controlador en modo hold. Resetea las acciones y congela las salidas.
    - **Auto (T)** : Cuando es cierto selecciona el control automático. Pone el control en modo manual cuando es falso.
    - **Pro.band (F):** Selecciona si: el valor proporcional de los parámetros de entrada del PID es con ganancia proporcional o con banda proporcional. Por defecto es 0, seleccionando banda proporcional. Kc=100/BP. (BP=banda proporcional).

  - **Reverse acting (T)** : Si es cierto selecciona la actuación inversa, el modo de trabajo habitual de los controladores (la salida decrece si la entrada es mayor que la consigna).
  - **Beta** : Es el nivel relativo del rechazo a las perturbaciones en el punto de consigna. Por defecto 1, que es apropiado para la mayoría de aplicaciones. Un valor pequeño entre 0-1 puede ser usado para incrementar el rechazo frente perturbaciones.
  - **Linearity** : Especifica la linealidad de la señal de error entre 0-1. El valor de 1 proporciona una respuesta plana, mientras que 0,1 aproximadamente proporciona un respuesta parabólica.
  - **Dt (s)** : Es el intervalo (en sg) en el que el VI es llamado. Si el intervalo menor o igual a 0, un temporizador interno usa un milisegundo de resolución.

- **Iteration** : Es el número de iteración del lazo de control.
- **Gain scheduling variable** : Valor usado para determinar los parámetros PID de la variable "gain schedule" .Esta entrada es usada sólo cuando la variable "scheduling input" esta en modo "gain scheduling input". La consigna, variable del proceso, salida del controlador, o un valor introducido por el usuario puede actuar como variable de ganancia programable.
- **Output:** Salida el algoritmo de control.

[3]

3.4 PID (Compatible)

Se usa este VI con cualquiera de los PID VI's de la anterior caja de herramientas PID para lograr la compatibilidad. Las entradas y las salidas son idénticas a los anteriores PID. No obstante, este VI usa el algoritmo PID revisado. Para usar este VI en una aplicación existente hacer doble click en el PID VI y escoge reemplazar para seleccionar este VI. Hay una entrada adicional. La entrada de iteración debe ser cableada desde el terminal de iteración del lazo de control para permitir la inicialización del algoritmo PID cuando el control de la aplicación esta en funcionamiento.

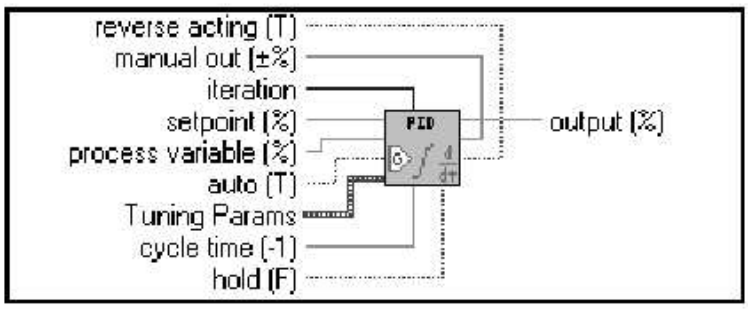


Figura 3.4 Conexionado del PID(Compatibility).VI

- **Reverse acting (T)** : Si es cierto selecciona la actuación inversa, el modo de trabajo habitual de los controladores (la salida decrece si la entrada es mayor que la consigna).
- **Manual out (+-%)** :Es el parámetro para la salida manual, expresado en %
- **Iteration** : Es el número de iteración del lazo de control.
- **Setpoint (%)** :Consigna del proceso expresada en porcentaje.
- **Process variable (%)** :Es el valor de la variable del control de realimentación.
- **Auto (T) :** Si es cierta selecciona el control automático, si es falsa control manual.
- **Tuning Params** : Conjunto que contiene los siguientes parámetros:

  - **Kc** : banda proporcional (%). La ganancia del controlador es 100/BP.

- **Ti :** tiempo integral en minutos por reset. 0=desactiva el reset.
- **Td** : Es la tasa en minutos por repetición. 0=desactiva acción derivativa.

- **Cycle time (-1)**: Es el intervalo en segundos en que es llamado el VI. Si este valor es menor o igual a cero, es usado un temporizador interno con una resolución de un milisegundo.
- **Hold (F):**Si es cierto activa el hold mode. Desactiva la acción de reset y congela las salidas.
- **Output (%)** : Salida del algoritmo de control expresada en %.

[3]

3.5 PID (autotuning).VI

Se usa este VI como PID VI con ganancia programable. Este VI usa entradas para especificar e iniciar un proceso de autosintonia. Con un valor cierto en la entrada autotuning se inicia el asistente de autosintonia.

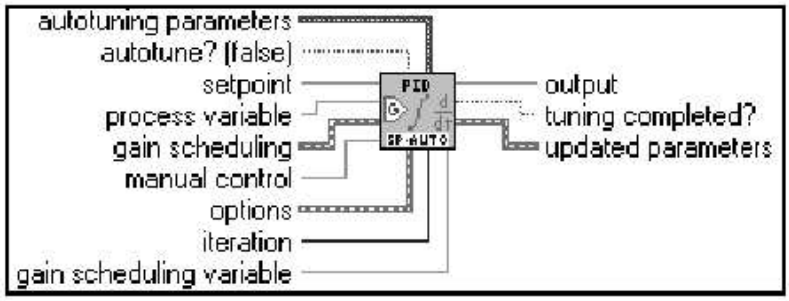


Figura 3.5 PID(autotuning).VI

- **autotuning parameters**: Se pueden seleccionar estos valores manualmente gracias al asistente.

  - **Controller Type** : Especifica si usaremos un control P,PI,PID.
  - **# Cycles:** Especifica el número de ciclos de consigna relacionados usados para determinar la ganancia y periodo final. Más ciclos conllevan una mejor estimación del parámetro. No obstante los sistemas lentos pueden necesitar mucho tiempo de respuesta si se usan muchos ciclos.

  - **Relay amplitude**: Determina la amplitud de la acción relativa de consigna. Esta consigna relativa esta en el rango (consigna-amplitud relativa) y (consigna-amplitud relativa).
  - **Control design:** Especifica la realización de la respuesta de los parámetros PID deseados determinados por el proceso de autosintonia. Una respuesta rápida, generalmente proporciona menor tiempo de subida, pero las respuestas lentas proporcionan menor sobrepico.
  - **noise level**: es una estimación del nivel de ruido obtenida en la medida de la variable de proceso
- **autotune? (F)** :Si es cierto indica que puede empezar el proceso de autosintonia y se llama al asistente de autosintonia.
- **setpoint** : Es el valor deseado para la variable del proceso.
- **process variable**: Es el valor de la realimentación del lazo de control.
- **Gain scheduling** : Define la ganancia programable para los parámetros proporcional, integral, derivativo.

- **Gain schedule :** Define el rango de la variable gain scheduling, para los respectivos parámetros del PID. Cada valor especifica el máximo valor de ese rango.
- **PID parametres** : Conjunto de las ganancias de las tres acciones, Proporcional, integral, derivativa. Cada set de parámetros corresponde a un rango de la entrada gain schedule.

    - **PID param** : Es el conjunto de ganancias proporcional, integral y derivativa.

        - **Kc** : Ganancia proporcional.
        - **Ti** : Constante de tiempo integral. Expresado en minutos. Valor 0 inhibe la acción.
        - **Td** : Constante de tiempo derivativa. Expresado en minutos. Valor 0 desactiva la acción.

- **Scheduling input :** Define que valor es usado como ganancia programable en la variable "gain scheduling". Escogiendo "gain scheduing input" indica que el valor cableado a la variable de entrada "gain scheduling" debe ser usada para la ganancia programable.
- **Manual control** : Control relativo de la variable de salida.
- **Options** : Grupo de accionamientos que especifican el algoritmo de control.

    - **Sp low** : Valor mínimo para la variable del proceso y consigna. Por defecto 0.
    - **Sp high** : Valor máximo para la variable del proceso y consigna. Por defecto 100.
    - **Out low** : Valor mínimo de salida del controlador. Por defecto -100.
    - **Out high** : Valor máximo de salida del controlador. Por defecto 100.
    - **Hold (F) :** Cuando es cierto pone el controlador en modo hold. Resetea las acciones y congela las salidas.
    - **Auto (T) :** Cuando es cierto selecciona el control automático. Pone el control en modo manual cuando es falso.
    - **Pro.band (F) :** Selecciona si: el valor proporcional de los parámetros de entrada del PID es con ganancia proporcional o con banda proporcional. Por defecto es 0, seleccionando banda proporcional. Kc=100/BP. (BP=banda proporcional).
    - **Reverse acting (T) :** Si es cierto selecciona la actuación inversa, el modo de trabajo habitual de los controladores (la salida decrece si la entrada es mayor que la consigna).
    - **Beta** Es el nivel relativo del rechazo a las perturbaciones en el punto de consigna. Por defecto 1, que es apropiado para la mayoría de aplicaciones. Un valor pequeño entre 0-1 puede ser usado para incrementar el rechazo frente perturbaciones.
    - **Linearity** Especifica la linealidad de la señal de error entre 0-1. El valor de 1 proporciona una respuesta plana, mientras que 0,1 aproximadamente proporciona un respuesta parabólica.
    - **Dt (s)** Es el intervalo (en sg) en el que el VI es llamado. Si el intervalo menor o igual a 0, un temporizador interno usa un milisegundo de resolución.

- **Iteration** : Es el número de iteración del lazo de control.
- **Gain scheduling variable** : Valor usado para determinar los parámetros PID de la variable "gain schedule" .Esta entrada es usada sólo cuando la variable "scheduling input" esta en modo "gain scheduling input". La consigna, variable del proceso, salida del controlador, o un valor introducido por el usuario puede actuar como variable de ganancia programable.
- **Output** : Salida el algoritmo de control.
- **tuning completed?** Indica si se ha completado el proceso de sintonía. Se puede usar para 0determinar cuando la actualización de la ganancia del PID es correcta..

- **updated parameters**: Son parámetros del gain scheduling una vez finalizado el proceso de sintonía del PID. Normalmente la salida es reflejo de la ganancia del scheduling de la entrada de los parámetros.
  - **Gain schedule :** Define el rango de la variable gain scheduling, para los respectivos parámetros del PID. Cada valor especifica el máximo valor de ese rango.
  - **PID parametres** : Conjunto de las ganancias de las tres acciones, Proporcional, integral, derivativa. Cada set de parámetros corresponde a un rango de la entrada gain schedule.

    - **PID param** : Es el conjunto de ganancias proporcional, integral y derivativa.

      - **Kc** : Ganancia proporcional.
      - **Ti** : Constante de tiempo integral. Expresado en minutos. Valor 0 inhibe la acción.
      - **Td** : Constante de tiempo derivativa. Expresado en minutos. Valor 0 desactiva la acción.

    - **Scheduling input**: Define que valor es usado para la variable "gain scheduling". Escogiendo "gain scheduing input" indica que el valor cableado a la variable de entrada "gain scheduling" debe ser usada para la ganancia programable.

4 Aplicación introductoria al LabVIEW

Control de temperatura

Objetivo:	Introducción a la programación en LabVIEW.
Aplicación.	Control de un proceso continuo para la visualización de la temperatura.
Herramientas :	Software de National Instruments.
Entorno de trabajo:	Cuando se crea un VI en LabVIEW trabajamos con dos ventanas: Una en la que se implementará el panel frontal (Figura 2) y otra que soportará el nivel de programación gráfica (figura 1).

Para la programación del panel frontal se dispone una librería de controles e indicadores de todo tipo y la posibilidad de crear más controles según las necesidades del propio usuario.

Para crear una variable de control tendremos que ir previamente a la barra del menú horizontal, bien en la pantalla de diagrama de control o de bloques. Desplazándose con el cursor en la opción control Show tool palette, y clicando sobre la misma aparecerá la paleta de herramientas. Para visualizar la tabla de funciones escogeremos la opción 'show function table' que está en el diagrama de control. [1]

Objetos utilizados de la paleta de control: Controlador numérico (High limit).

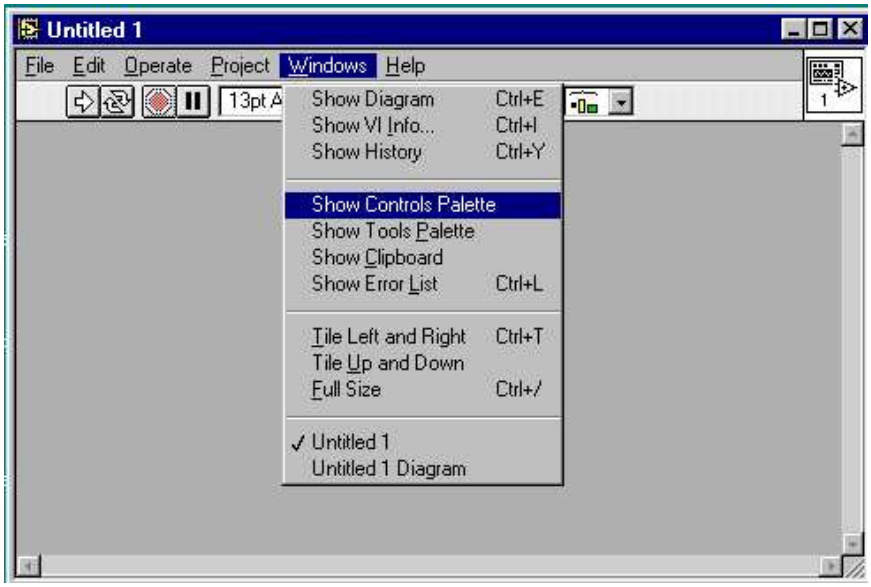


Figura 4.1 Menú panel frontal

Figura 4.2 : Barra de herramientas

Figura 4.3 : Barra de control

Las barras de herramientas y control se pueden activar mediante el menú horizontal de la pantalla del panel frontal.

La barra de funciones se puede activar mediante el menú horizontal de la pantalla del diagrama de bloques.

Al situar el puntero sobre algún icono se despliega un menú asociado con todas las opciones relacionadas con ese icono.

Figura 4.4 : Barra de funciones

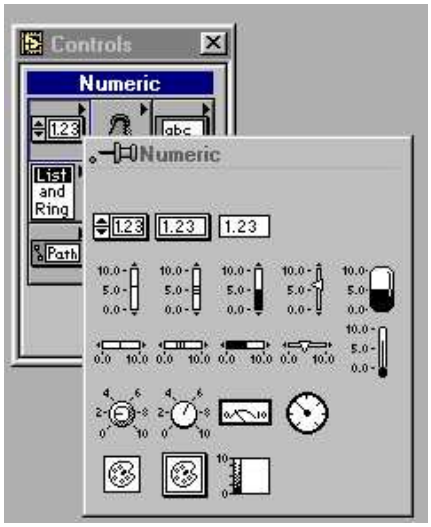


Figura 4.5 Menu de Controles Numéricos

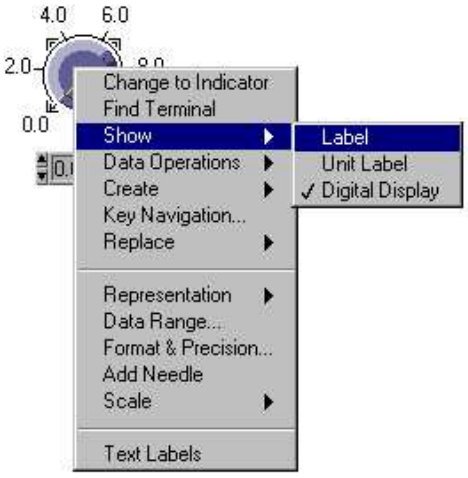


Figura 4.6 Ejemplo de un control numérico

Control numérico: Una vez hemos seleccionado el control numérico deseado lo arrastramos hasta situarlo en la pantalla de diagrama de control. Y a continuación procederemos a realizar un pop-up para poder ver las opciones que presenta dicho control. Una vez presionemos sobre label aparecerá sobre el control un recuadro para que nosotros introduzcamos un título o etiqueta.

La asignación del valor de control podremos fijarlo desplazando el potenciómetro con el cursor o bien situándonos sobre las flechas del rectángulo del dispositivo de control.
Al mismo tiempo que realizamos esta operación (crear una variable de control sobre el panel frontal) se crea su variable terminal en el diagrama de bloques, la que relacionaremos con otros elementos.

Tanto los indicadores como los controladores tienen un icono cuya forma depende del tipo de información que nos suministra o representa desde el diagrama del control.

Control tipo booleano (Power)

Indicador tipo booleano (Warninig)

Indicador tipo entero (Ejemplo)

Control tipo entero (High limit)

Representación de una variable de control booleana tal como se vería en la pantalla del panel frontal

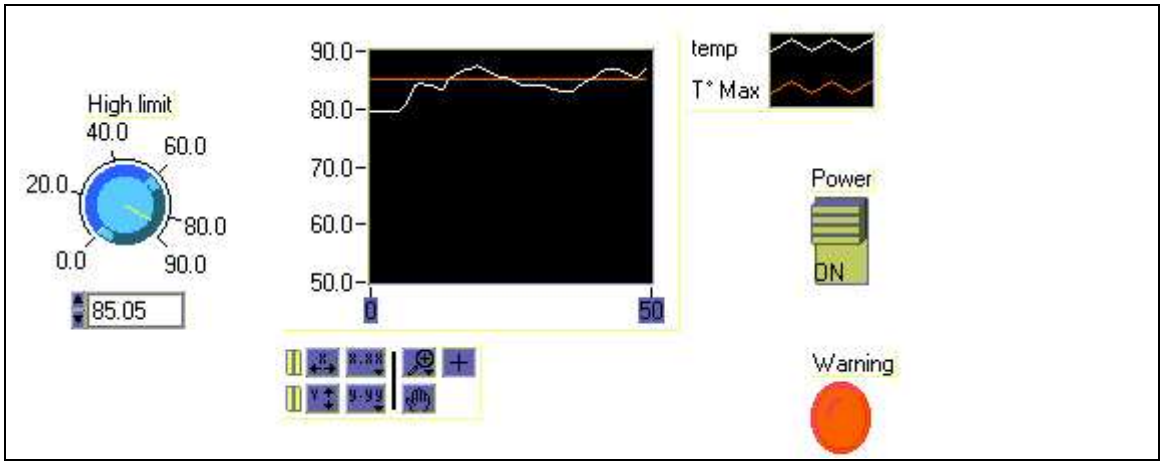


Figura 4.7 Panel frontal del control de temperatura.

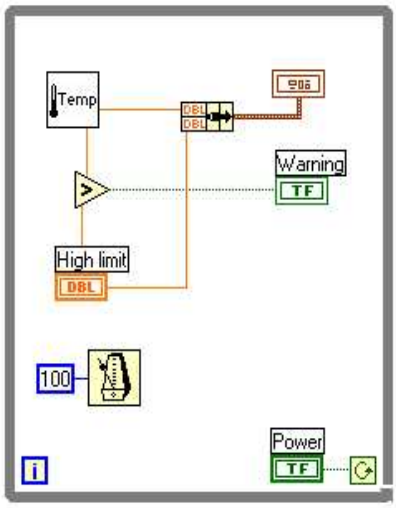


Figura 4.8 Diagrama de bloques del control de temperatura

Dentro de una estructura while relacionamos los iconos mediante "cables". La estructura while se realizará hasta que power adquiera el valor 'false'. La temperatura está siendo comparada con el High limit. Dando 'true' cuando la temperatura da un valor superior a high limit. Todos estos valores se van representando en la gráfica.

El 'while' se realiza cada incremento de 100 milisegundos. Este valor lo introducimos mediante el temporizador.

5. Aplicación de un control PID, sobre un proceso continuo:

5.1 Descripción del proceso a controlar

Tenemos un planta química que prepara una solución compuesta por dos elementos (A,B), y en función del producto final requerido variamos los porcentajes de los compuestos.
A tal efecto disponemos de dos tanques que albergan los compuestos, y una serie de válvulas de control para regular el caudal de entrada de los líquidos, para obtener la solución adecuada.
Para controlar el caudal de salida de los tanques, disponemos de un controlador lineal para poder modificar la proporción entre un compuesto respecto al otro.
Cada uno de los tanques presenta un controlador para poder variar en tiempo real el nivel de líquido presente en cada tanque.
La salida final del producto requerido esta disponible en una canalización de salida final, para posterior almacenaje o distribución.
Para controlar el proceso disponemos de unas gráficas que muestran la evolución del proceso.
Tales gráficas constan de las siguientes leyendas: nivel deseado del tanque, nivel real del tanque, y caudal de entrada en el tanque.
El caudal de entrada de los tanques esta regulado por válvulas, que a su vez son accionadas por un control tipo PID.

5.2 Descripción del diagrama de bloques

La estructura básica del programa es un 'while'. Dentro de esta estructura se dan todos los procesos de la aplicación. La aplicación se irá realizando hasta que la condición de repetición sea 'false'. Para ello el pulsador de parada tendrá que estar pulsado dando un 'true' que a su vez estará negado por un 'nor'. Dentro del bucle tendremos un 'timer' para que el 'while' se repita cada 250 milisegundos.

Los PID.VI que utilizaremos para controlar los tanques serán estándares. Los controladores PID's constan de dos clusters de datos las Opciones y los Parámetros de PID. Estas variables están explicadas en el capítulo 3 (software PID's). Las variables de ambos clusters están controladas desde el panel frontal. También introducimos en los controles PID las consignas, es decir, el nivel de solución que deseamos que posea cada tanque, al igual que introducimos el valor real que poseemos. Por último entramos en el control PID el valor de la iteración que estamos realizando. La salida que nos proporcionan los PIDs es el caudal de entrada a los tanques, es decir la posición de la válvula de entrada.

Para poder controlar el proceso en tiempo real de lo que esta sucediendo en los tanques, utilizaremos el entorno gráfico que nos proporciona LabVIEW. Para ello tendremos que utilizar un indicador 'Waveform Chart' que encontraremos en el panel de controles. En esta gráfica dispondremos del nivel real del tanque, el nivel deseado, y de la entrada a el tanque. Para poder introducir en el 'Waveform' estos tres datos que queremos representar tendremos que utilizar un 'bundle' que los unirá en un solo clúster. El 'bundle' realiza la tarea de agrupación de varios elementos en un solo cluster.

La planta.vi es la encargada de simular el proceso que controla el controlador PID. (Este VI será explicado más extensamente en el apartado 5.4). A la Planta le introducimos un nivel inicial de compuesto en el tanque de 0, un ruido de 5 unidades, que puede ser debido tanto a la evaporación de la solución como a imperfecciones en la válvula de entrada. A el 'process deadtime' le asignamos una constante de valor '2', es decir, cuando el bucle 'while' haya realizado la segunda iteración la planta entrará en funcionamiento. Esto será a los 500ms. El 'process gain' (que vale 1.5) multiplica a la ganancia del sistema de primer orden. El 'process load' es la entrada del caudal a los tanques, que varía entre 0 y 100, dependiendo de la posición del potenciómetro de proporcionalidad entre el 'compuesto 1' y el 'compuesto 2'. La variable 'iteration' debe estar conectada a las iteraciones del bucle 'while'. El 'Time Lag' es la constante de tiempo (en minutos) del polo del sistema, a la que se otorga un valor de 0,3 . El 'deadband', que tiene asignado el valor '1', representa los márgenes mínimos de cambio de la salida de la planta. La salida no cambiará hasta que la entrada varíe el tanto por ciento indicado por 'Deadband' de su valor inicial.

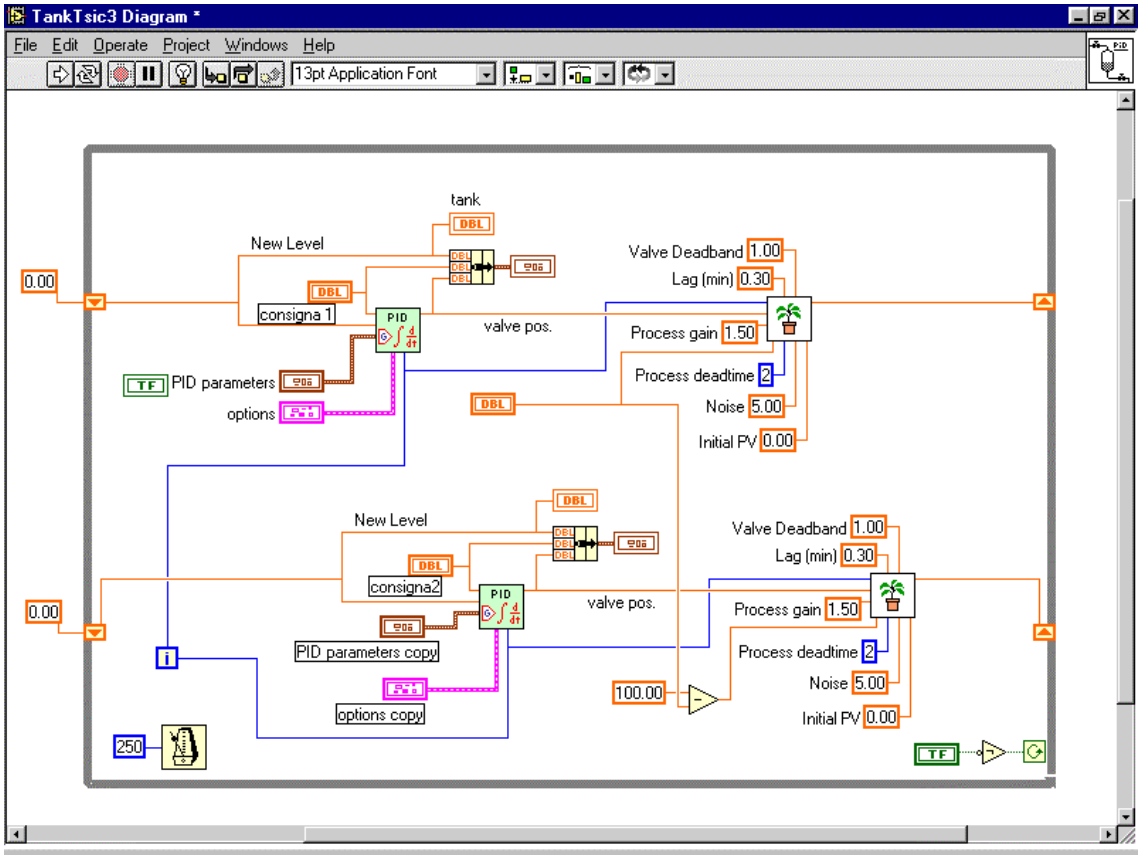


Figura 5.1 Diagrama de bloques del Control PID en un proceso continuo.

5.3 Panel frontal de la aplicación

El panel frontal es una pantalla de labVIEW donde están representados los indicadores y los controladores del proceso. Dichos controladores e indicadores tienen sus correspondientes

etiquetas que describen su funcionalidad. Por ejemplo el potenciómetro que controla la consigna del tanque 1 tiene una leyenda que dice *Nivel del tanque deseado del compuesto 1*.
Para poder obtener cambios en la respuesta del controlador PID, y que éste no se estabilice con facilidad se dispone de ciertas variables a las que puede acceder el usuario, como las consignas de los tanques, o la salida de estos mediante el potenciómetro que controla los porcentajes.

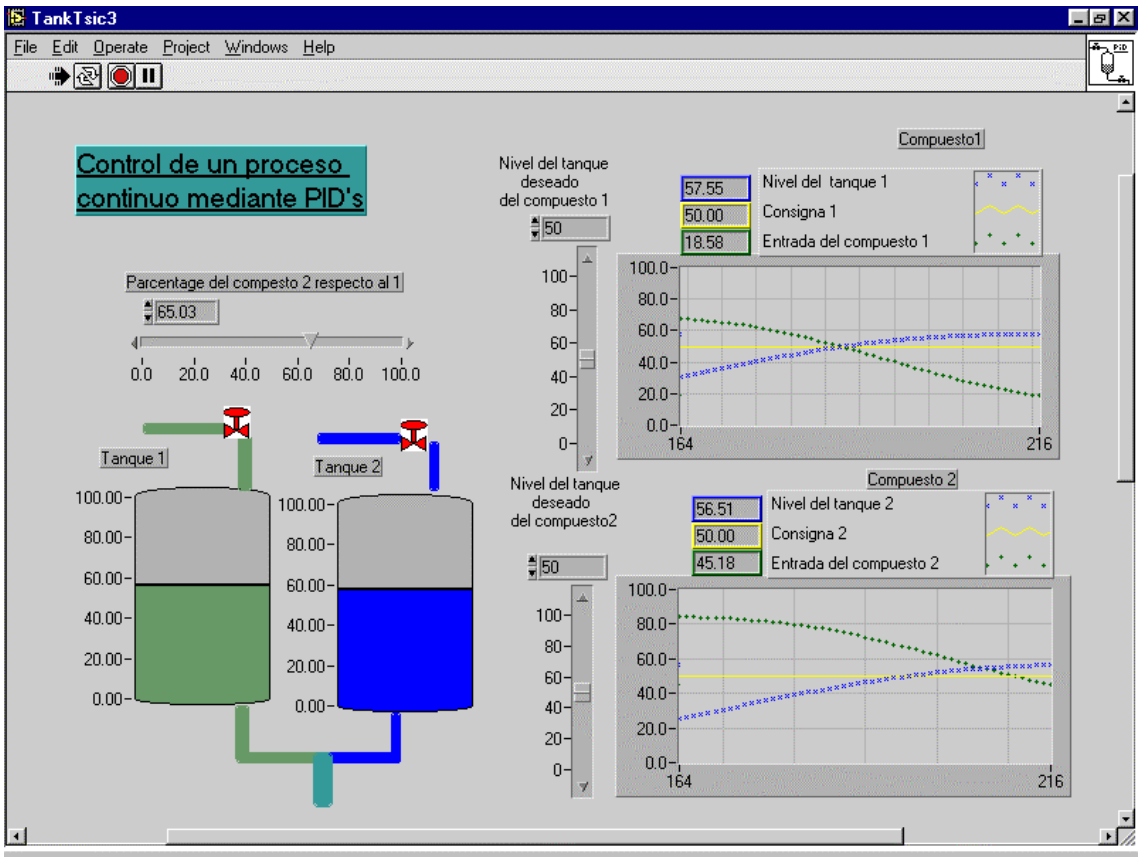


Figura 5.2 Panel frontal del Control PID en un proceso continuo

Para poder variar los parámetros PID tenemos unos controladores numéricos que nos permiten modificar su valor en tiempo real (figura 5.3) Al mismo tiempo también disponemos de un conjunto de interruptores y controladores numéricos que nos permiten variar las opciones que nos proporcionan los PIDs (figura 5.4).

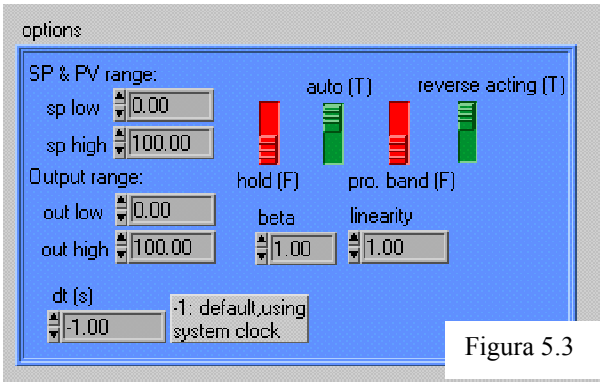


Figura 5.3

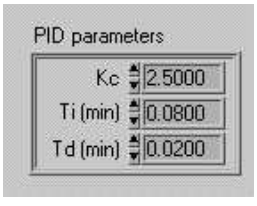


Figura 5.4

5.4 Planta VI

La PLANTA VI simula la respuesta física de una planta industrial cualquiera. Este VI mide la variable del proceso (PV) en %. Para usar este VI con un lazo de realimentación, hay que llamar al VI con el conmutador "Update PV" (actualizar variable de proceso), en posición falso e implementar el algoritmo PID. Finalmente llamar al VI con el "Update PV" a cierto. Para calcular el nuevo valor de variable de proceso hay que proporcionar todas las características de la planta y conectar la salida del PID hacia la variable "manipulada", como se muestra en la figura 5.5. Cuando el valor de "actualizar PV" esta en falso el último valor de la variable de proceso pasa a la salida. Cuando "actualizar PV" este en cierto, el VI lee y retiene la variable de proceso y la escala conforme al valor de la variable 'Process Gain' (Ganancia de proceso). El VI añade ruido y la respuesta resultante del proceso al antiguo valor, a través de un filtro de primer orden, añadiendo una constante de tiempo razonable a la respuesta aparente del tanque. La salida del filtro representa la nueva variable de proceso.[3]

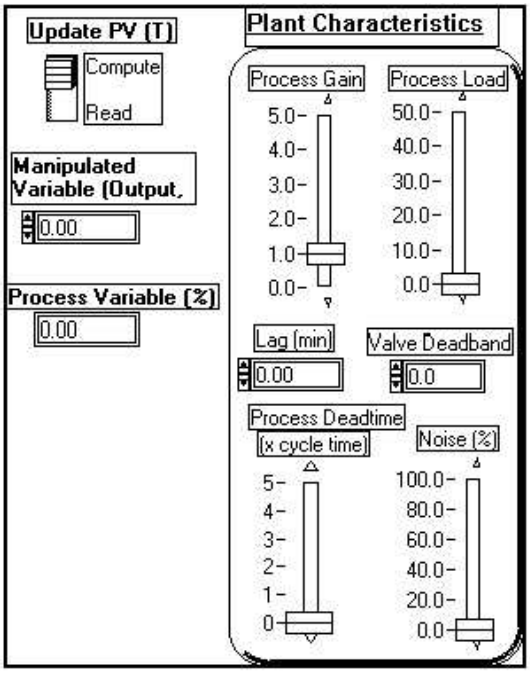


Figura 5.5 Panel frontal de PLANTA VI

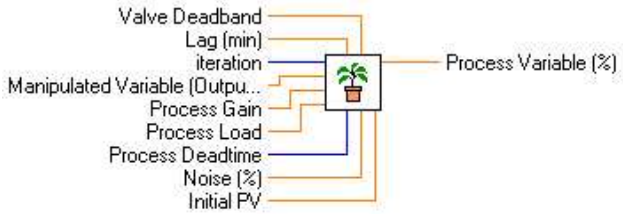


Figura 5.6 : Conexionado de la PLANTA VI

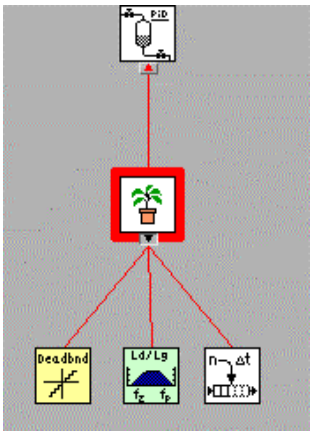


Figura 5.7 : Elementos de "la planta VI":

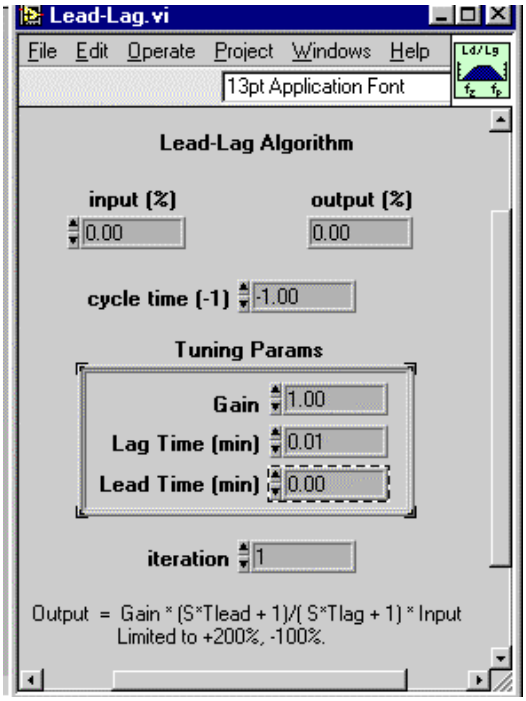


Figura 5.8 : Panel frontal de Lead-Lag

La Planta VI, agrupa una serie de acciones como:

-Deadband Simulator VI.
-Lead-Lag VI.
-Shift Register Delay VI.

El VI más significativo de los VI´s anteriores, es el Lead-Lag VI:
Este Vi corresponde a un sistema de primer orden con función de transferencia estándar, donde:

Parámetros de sintonía:

-Gain: Ganancia del sistema de primer orden.
-Lead Time: Constante de tiempo en minutos del cero del sistema de primer orden.
-Lag Time: Constante de tiempo en minutos del polo del sistema.
-Output: Salida del sistema, expresada en %.
-Input: Entrada del sistema expresada en %.

Otro VI muy importante es el Deadband.VI . Éste proporcional os màrgenes mínimos de cambio de la planta. La salida no cambiará hasta que la entrada varíe el 'Deadband%' de su valor.

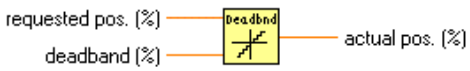


Figura 5.9 : Deadband.VI

Como muestra el diagrama de bloques, nuestra aplicación necesita dos VI's que a su vez contienen los otros VI's usados. Éstos VI's son facilitados por National Instruments para el control de cualquier tipo de plantas (el PID) y aprendizaje (Planta). En una aplicación real usaríamos una tarjeta de adquisición de datos que substituiría los valores dados por la palnta.VI.

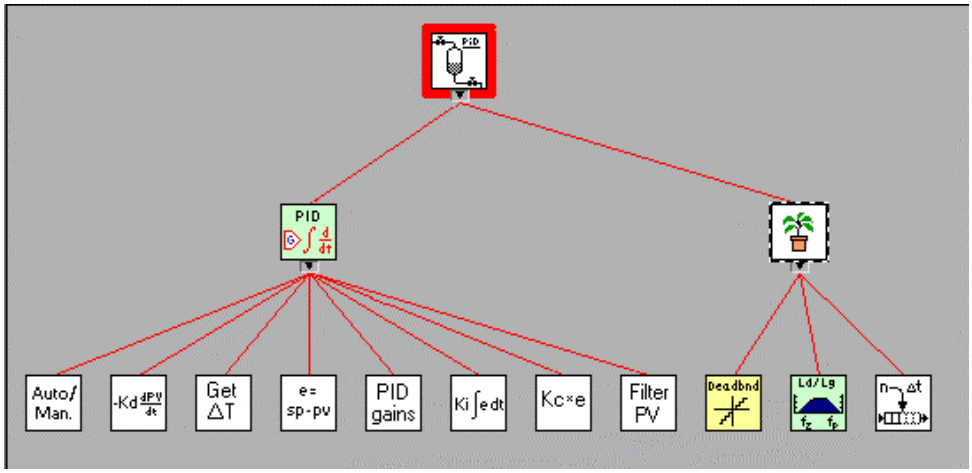


Figura 5.10 Estructura y elementos de la aplicación

6. Otras Herramientas de programación gráfica

1. Diferencias entre LabVIEW i LabWindows/CVI.

Las diferencias entre ambos programarios radican básicamente en la metodología de desarrollo en la programación.

Por un lado LabVIEW presenta una programación en lenguaje G (gráfico) más actual, y aceptando incluso la programación orientada a objetos (OOP).

En contraste, LabWindows/CVI esta diseñado para la extensa base de programadores formados en lenguajes más tradicionales como "C" y Basic. Aunando los conocimientos de los mismos y las innumerables librerías que están implementadas en estos lenguajes.

La decisión de decantarse por una de estas dos opciones, viene en función de las preferencias de los usuarios/programadores en función de sus habilidades o conocimientos en programación.

2. Lookout.

Software de automatización industrial basado en objetos y de elevadas prestaciones, entre ellas cabe destacar:

-Arquitectura orientada a objetos.
-Crear y conectar objetos.
-Configuración on-line, permitiendo modificar y borrar sin interrumpir el proceso.
-Ejecutar acciones frente un evento.
-Expandible I/O.
-Fast polling.
-Animación.
-Comunicación a distancia: PLC, RTU, FieldPoint, DAQ, SCXI, Fieldbus devices...
-Incluye drives DLL.
-Capacidad de comunicación por radio o módem.
-Tipos de objetos:

- Continuos.
- Secuencial.
- Objetos de control discreto.
- Históricos.
- SPC.
- Trending.
- SQL.
- Seguridad.
- Alarma.
- Animación.
- Dirección.

Cabe destacar que es un software altamente portable, capaz de ejecutarse en cualquier PC y bajo cualquier sistema operativo: Windows NT,95,3.X,MS-Dos.
Esta provisto de una interface gráfica HMI (Interface Machine Human), propia de los sistemas SCADA (Supervisory Control and Data Adquisition), permitiendo grácias a estas características la supervisión y control de procesos, sistemas de telemetría y porcesos batch.

Las aplicaciones en Lookout, las podemos encontrar en innumerables industrias como: Industria Química, petroquímicas, alimentación, farmacéutica, tratamiento de aguas, paperera, transporte, plástico, energía, automatización etc...

3. BridgeVIEW.

Este paquete software sigue las directrices de los anteriores, aunque el echo más reseñable es que presenta el denominado G Wizard, que no deja de ser una aplicación soporte para implementar un interface HMI o SCADA de una forma más ágil y eficaz.
También permite la conectividad con ODBC, SQL, TCP/IP, DDE, ActiveX.
Como echo singular hay que mencionar que BridgeVIEW no corre bajo MS-DOS, sino que es un software que fue ideado para ejecutarse en un sistema operativo Windows NT/95; con lo que nos beneficiamos del echo de trabajar con aplicaciones de 32 bits, multitasking, comunicación entre procesos y trabajo en red.

Entre las compañías que utilizan este tipo de producto están: Allen-Bradley, Device Net, GE Fanuc, Mitsubishi, Omron y Siemens entre otras.

4. Softwarwe Toolkits for Process Monitoring and Control.

A LabVIEW y LabWindows/CVI se les han añadido una serie de toolkits, algunas de ellas específicas para procesos de supervisión (monitorización de procesos) y control de aplicaciones. Ofreciendo ambos “toolkits PID control”.

LabVIEW --SPC Toolkit.
LabWindos/CVI --SQL Toolkit.

5. Control Toolkit.

Este toolkit añade sofisticados algoritmos de control a LabVIEW, BridgeVIEW y LabWindows/CVI. Con este paquete, se puede fácilmente construir rápidamente sistemas de control y proceder a su sintonía.

Características:
-P,PI,PID.
-Error-squared PID.
-External Reset Fdbk PID.
-PID with external-reset feedback.
-Lead-lag compensation.
-Setpoint ramp generation.

-Integrador Anti Wind.up.
-Multiloop cascade control.
-Feedforward control.
-DAQ board example.
-Override (min/max selector) control.
-Ratio/bias control.

Podemos aplicar este tipo de control de procesos para controlar el nivel de tanques o simulación de plantas, entre otros.

6. Overview

El PID Control Toolkits añade sofisticados algoritmos de control al software base de instrumentación para el desarrollo de sistemas. Combinando el PID Control Toolkit con funciones matemáticas y funciones lógicas de LabVIEW, BridgeVIEW, LabWindows/CVI; para desarrollar programas para el control de automatización.

7. Fuzzy Logic Control Designer.

La lógica Fuzzy puede ser usada para acelerar el desarrollo del control para sistemas altamente complejos o no lineales. Pudiendo ser combinado con NI-DAQ, PID Control Toolkit y SPC (statistical process control).

Sólo disponible para aplicaciones bajo LabVIEW/BridgeVIEW.

7. Bibliografía

[1] **Antonio Mánuel Lázaro.**
LabVIEW. Programación Gráfica para el control de instrumentación.
Ed. Paraninfo.

[2] **K. Aström y T. Häglud.**
PID Controllers: Theory, Design and Tunning.
2ª Edición. 1994

[3] **National Instruments**
PID Control toolkit reference manual
National Instruments 1998 [Libro electrónico formato PDF]

[4] **Pere Ponsa**
Apunts de Classe Tecnologia de Sistemes de Control
Ahlens S.L. 1999
.

[5] **Herbert Schild**
C/C++: Manual de referencia.
McGraw-Hill 2ª Edición. 1992.

Otras fuentes :

Pàgina Internet de Natinonal Instruments http://www.natinst.com