UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ciencias y Humanidades

OPTIMIZACION DE SISTEMAS DE CONTROL

MANUEL ANTONIO FERNANDEZ MELGAR

Trabajo de investigación presentado para optar
al grado académico de
Licenciado en Ingeniería Electrónica

Guatemala

1993

OPTIMIZACION DE SISTEMAS DE CONTROL

Vo. Bo. :

(f) Ing. Rolando Mata.

Tribunal:

(f) Ing. Rolando Mata

(f) Lic. Roberto Tejada

(f) Manuel A López V.
Dr. Manuel López

Fecha de aprobación: 7 de octubre de 1993.

Dedico este trabajo a:
Dios,
A mi padre
Ing. Manuel Fernández
A mi madre
Ft. Rosita M. de Fernández
mis hermanos
Ada y José

CONTENIDO

RESUMEN

El avance de la tecnologia es producto de las necesidades que deben satisfacer los seres humanos, y los sistemas de control vienen a contribuir de una manera determinante al desarrollo tecnológico humano. Sus aportes más significativos apuntan a la automatización industrial en el área de robótica y control de procesos, dando como resultado economias de escala; también brinda aportes al área de investigación, navegación y defensa.

En vista de la importancia que juegan los sistemas de control en general, esta área ha sido desarrollada a través de los años y se ha convertido en una ciencia. Este proyecto tiene como finalidad convertirse en una herramienta para el ingeniero o proyectista de control, proporcionándole varios métodos de optimización para aplicarlos a sus sistemas de control, con el propósito de encontrar el máximo rendimiento y mejor respuesta dependiendo de las condiciones que rodeen a dicho sistema.

I. INTRODUCCION

La necesidad de un arreglo de componentes físicos conectados de tal manera que se pueda comandar, dirigir o regular a si mismo o a otro sistema es lo que dio origen a los sistemas de control, como modernamente se les conocen. En el sentido más abstracto es posible considerar cada objeto fisico como un sistema de control, sin embargo en el área de ingenieria se restringe el significado al aplicarlo a esos sistemas, cuya función principal es comandar, dirigir o regular dinámica o activamente.

Con la ayuda de sistemas computarizados, los sistemas de control se pueden simular y, por lo tanto, planear las características y tipos de respuestas que estos vienen a tener. Este proyecto consiste primordialmente en la elaboración de un paquete que sirva de herramienta, tanto en el diseño como en la optimización.

A lo largo de este trabajo se muestra la teoria necesaria para comprender el proceso de optimización, asi como la información necesaria para utilizar el paquete.

II. SISTEMAS DE CONTROL

A. Tópicos del proyecto

La principal razón de este proyecto es brindar una herramienta al proyectista de control en el diseño y optimización de sistemas de control. En este proyecto se presentan varios métodos para poder llevar esto a cabo.

El principal objetivo de un sistema de control es que éste sea estable (se refiere a la estabilidad cuando a una señal de entrada limitada produce una salida limitada). La consideración ahora es respecto qué parámetros el sistema es estable y cuál es su grado de estabilidad.

Algunos métodos únicamente nos indican si el sistema será absolutamente estable, otros, por el contrario, que es el objetivo del presente trabajo, tratan sobre los grados de estabilidad y sugieren formas para poder convertir un sistema inestable en uno estable con la variación de parámetros u operación en determinadas regiones.

Los métodos que se presentan y se implementan en el paquete son los de Lugar de las Raices, Nyquist, Nichol y variables de estado. También se presenta en el programa un sistema para poder diseñar diagramas de bloques. De igual manera se plantea a lo largo de este trabajo la descripción teórica de los métodos y cómo con ejemplos se le puede sacar provecho con el paquete de computación.

B. <u>Ventajas y Desventajas de los Métodos</u>.

No todos los métodos nos indican en una forma sistemática todos los aspectos del sistema que se está analizando, es necesario en algunos casos aplicar más de un método para tener un informe más completo y detallado de lo que se analiza.

Es por esto la necesidad de mostrar toda una gama de métodos que a lo largo de su descripción en este trabajo el lector encontrará el camino más apropiado para el análisis.

No se puede en ningún momento indicar que un método sea más completo que otro cuando de estabilidad relativa se habla, sino que depende de los parámetros actuales y los requerimientos de la optimización.

III. ANALISIS

A. Método de Lugar de las Raices

1. Introducción.

W.R Evans desarrolló un método simple para hallar las raices de la ecuación caracteristica y se utiliza extensamente en el área de Ingenieria de control.

Este método consiste en trazar todas las raices de la ecuación caracteristica para todos los valores de un parámetro del sistema. Habitualmente este parámetro es la ganancia. La idea básica, es que los valores de s que hacen a la función de transferencia alrededor del lazo igual a -1, deben satisfacer la ecuación caracteristica del sistema. Este método permite encontrar los polos de lazo cerrado partiendo de los polos y ceros de lazo abierto. Para sistemas de controles lineales, el método de lugar de las raices resulta útil, pues indica la forma en que hay que modificar los polos y ceros de lazo abierto para que la respuesta cumpla con las especificaciones de comportamiento de sistema.

2. <u>Variación de Polos</u>.

La forma canónica del sistema por retroalimentación está definido por su función de transferencia de lazo cerrado, esto es:

$$\frac{C}{R}=\frac{G}{(1+GH)}$$

ECU I

La función de transferencia de lazo abierto GH se puede presentar por:

$$GH=\frac{KN(s)}{D(s)}=\frac{K(s^{m}+a_{m-1}s^{m-1}+\ldots\ldots+a_0)}{(s^{m}+b_{n-1}s^{n-1}+\ldots b_0)}$$

ECU II

En este caso K es el factor de ganancia de lazo abierto.

Los polos de lazo cerrado son las raices de la ecuación caracteristica:

$$D(s)+KN(s)=0$$

ECU III

La localización de estas raíces en el plano s cambia a medida que se varia el factor de ganancia en el lazo abierto K de $-\infty$ a $+\infty$. El lugar de estas raices representado en el plano s como una función de K se llama el **lugar geométrico de las raíces**, el cual se puede dividir en las siguientes categorías:

1. Lugar geométrico positivo
2. Lugar geométrico complementario
3. Contornos de las raíces

Este último se refiere cuando se varía más de un parámetro.[1]

3. <u>Criterios de Angulo y Magnitud</u>.

Para que una rama pase por un punto en particular del plano s es necesario que sea una raíz de la ecuación característica, esto es que:

$$GH(s) = KN(s)/D(s) = -1$$

ECU IV

Por lo tanto el número complejo debe tener un ángulo de fase de 180° + 3601°, donde 1 es un entero arbitrario. En consecuencia K debe tener el valor particular que satisface el **criterio de magnitud**:

$$|GH(s)| = 1$$

ECU V

El número de ramas del lugar de las raíces es igual al número de polos de la función de transferencia de lazo abierto GH. Con estos conceptos en mente podemos proceder a la construcción del lugar de las raíces para el eje real:

[1] Kuo,B. Sistemas Automáticos de Control

Para K > 0:

Hay puntos del lugar de las raices sobre el eje real que quedan hacia la izquierda de un número impar de polos y ceros finitos.

Para K < 0:

Los puntos quedan hacia la izquierda de un número par de polos y ceros finitos.

Para K = 0:

Los puntos del lugar geométrico son los polos de G(s)H(s).

El lugar geométrico de las raices completo es simétrico respecto del eje real del plano s. En general, el lugar geométrico de las raices completo es simétrico a los ejes de simetria de los polos y ceros de G(s)H(s). Para valores grandes de s ($k \geq 0$), el lugar geométrico positivo es asintótico a lineas rectas o producen asintotas con ángulos dados por:

$$\Phi_k = \frac{(2k+1)\pi}{n-m}$$

Para el Lugar Geométrico Negativo, $k \leq 0$, los ángulos de las asintotas son:

$$\Phi_k = \frac{2k\pi}{n-m}$$

donde n y m representan el número de polos y ceros de G(s) H(s), respectivamente.

4. Márgenes de Ganancia y Fase.

El **margen de ganancia** es el factor por el cual el valor de diseño del factor de ganancia k se puede multiplicar antes que el sistema de lazo cerrado se vuelva inestable.[2] Este se puede determinar a partir del lugar de raíces usando la siguiente ecuación:

$$\text{margen de ganancia} = \frac{\text{Valor de k cruce eje imaginario}}{\text{valor de diseño de k}}$$

Para encontrar el margen de fase es necesario encontrar el punto jw1 sobre el eje jw para el cual $|GH(jw1)|=1$ para el valor de diseño de k, luego el margen de fase se calcula de la siguiente manera:

$$\phi_{PM}=180°+argGH(jw_1)$$

A través del lugar de las raíces se puede encontrar los factores de amortiguamiento, que es el factor de ganancia requerido para dar una razón de amortiguación especificada:

$$\phi=\cos^{-1}\xi$$

El factor de ganancia en el punto de intersección

---

[2] Distefano, S. Retroalimentación y Sistemas de Control

con el lugar de las raices es el valor requerido de K.

5. Diseño con Lugar de las Raices.

Este diseño se lleva a cabo en la forma más simple escogiendo el valor de k que resulte de un comportamiento satisfactorio de lazo cerrado. A esto se le llama compensación del factor de ganancia. Si la configuración de polos y ceros es tal que no pueda satisfacer las especificaciones del sistema variando el factor de ganancia de lazo abierto, se puede añadir al sistema un compensador en cascada. El objetivo es de reemplazar polos indeseados por polos deseados.

Estos compensadores en cascada pueden afectar o compensar la fase o la magnitud. Típicas redes de atraso y adelanto de fase se muestran a continuación:

$$P_{Adelanto}=\frac{s+a}{s+b},\ 0\le a<b$$

$$P_{Atraso}=\frac{a}{b}\left(\frac{s+b}{s+a}\right),\ 0\le a<b$$

Para compensar la magnitud se pueden sintetizar dipolos de baja frecuencia, utilizando un compensador proporcional o integral.

Otra forma de diseñar utilizando el lugar de las raices, es por medio del método del punto, donde conocemos según las especificaciones del sistema un punto s1, por lo tanto se puede forzar a que una rama del lugar pasará por el punto s1 utilizando para ello compensaciones de fase y magnitud.

B. Análisis de Nyquist.

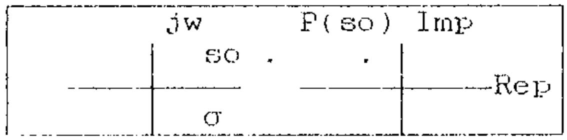
1. Introducción.

El criterio de estabilidad de Nyquist está basado en teoremas de la teoria de las variables complejas y determina la estabilidad absoluta y relativa de los sistemas de control de lazo cerrado. Este método tiene la ventaja en ser utilizado cuando una señal se retrasa T segundos en alguna parte del lazo del sistema; asimismo aparecen términos exponenciales en la ecuación caracteristica.

$$e^{-Ts}=1-Ts+\frac{T^2s^2}{2!}-\ldots$$

La ventaja de este método reside en que se puede utilizar el término exponencial, en lugar de su aproximación a series de potencias. Las técnicas de Nyquist son útiles para obtener información acera de las funciones de transferencia de componentes o sistemas a partir de datos experimentales. Este rasgo caracteristico es muy útil en la determinación de las caracteristicas de estabilidad del sistema, cuando las funciones de transferencia de las componentes no están disponibles en forma analitica o cuando los sistemas fisicos se deben probar y evaluar experimentalmente.

Ilus 1



2. <u>Representaciones de funciones</u>.

Una función compleja no se puede representar sobre un conjunto único de coordenadas, la variable compleja $s=\sigma+jw$ al ser aplicada a una función P(s) también tendrá su parte real e imaginaria; por lo tanto se necesita un par de planos como el de la ilustración 1.

3. <u>Trayectoria de Nyquist</u>.

La trayectoria de Nyquist es un contorno cerrado en el plano s que rodea completamente todo el semiplano derecho del plano s. Para que la trayectoria no pase por cualquier polo de P(s), se requieren en la trayectoria pequeños semicirculos a lo largo del eje imaginario.

4. Representación de Estabilidad.

El sistema de control de lazo cerrado, cuya función de transferencia de lazo abierto sea GH(s), es estable si:

$$N = -Po \leq 0$$

donde Po ≡ número de polos en el plano derecho de GH y N número de circunvalaciones antihorario del punto (-1,0), por lo tanto el sistema es absolutamente estable si N=0. si N >0, el número de ceros Zo está dado por:

$$Zo = N + Po$$

Hay que poner en claro que las técnicas de diseño de Myquist se usan rara vez por si solas debido a que las representaciones de estabilidad de Nyquist son difíciles de generar en detalle y por consiguiente, se usan en general para complementar otros métodos. En el diseño de Nyquist se pueden aplicar redes de adelanto, la escogencia apropiada de polos y ceros permite un aumento en el factor de ganancia k, proporcionando mayor exactitud (y a veces estabilidad), sin afectar adversamente el rendimiento bajo transitorios. Generalmente un tipo de compensación de adelanto aumenta el ancho de banda del sistema. Cuando se aplica una compensación de atraso generalmente ocurre:

1- El ancho de banda del sistema decrece

2- La constante de tiempo del sistema τ generalmente aumenta, produciendo un sistema más lento.

3- Para una estabilidad relativa dada, el valor de la constante de error aumenta.

4- Para un valor dado de la constante de error, la estabilidad relativa mejora.

$$P_{Adelanto}=\frac{s+a}{s+b}$$

$$P_{Atraso}=\frac{a}{b}\left(\frac{s+b}{s+a}\right)$$

Otros sistemas de compensación pueden ser utilizados, entre ellos está el uso de compensación por retroalimentación.

C. Cartas de Nichol.

1. Introducción.

La representación sobre las cartas de Nichols tienen dos ventajas sobre la representación polar:

a. Se puede representar en un intervalo de amplitudes mucho más amplio porque $|GH(jw)|$ se representa sobre una escala logaritmica.

b. La representación de GH(jw) se obtiene mediante la suma algebraica de las magnitudes individuales y contribuciones de fase.

Esta técnica es útil para obtener directamente la representación de C/R(jw).

2. Representación de magnitud en DB contra fase.

La forma polar de respuesta de frecuencia de lazo abierto es:

$$GH(jw)=|GH(jw)| \ \arg GH(jw)$$

Por lo tanto, la representación de la magnitud contra el ángulo de fase de GH(jw) es una gráfica de |GH(jw)| en db contra arg GH(jw), en grados, en un sistema de coordenadas rectangulares con w como parámetro.

3. Estabilidades Relativas.

Los márgenes de ganancia y fase se determinan fácilmente de la representación de la magnitud en db contra el ángulo de fase de GH(jw).

La frecuencia de **cruce de fase** $w\pi$ es la frecuencia en a la cual la curva de GH(jw) intersecta la línea de -180° sobre la representación de la magnitud db contra el ángulo de fase. El **margen de ganancia** en db está dado por

$$margen{\cdot}de{\cdot}ganancia = -20\log_{10}|GH(jw_\pi)|\, db$$

El cual se obtiene directamente de la gráfica de db contra ángulo de fase. La frecuencia del **cruce de ganancia** w1 es la frecuencia a la cual la curva de GH(jw) intersecta la linea de 0 db sobre la gráfica de la magnitud db contra el ángulo de fase. El margen de fase está dado por:

$$margen{\circ}de{\circ}fase = 180^\circ + argGH(jw_1)$$

En la mayoria de casos, los márgenes positivos de ganancia y fase aseguran la estabilidad; sin embargo, la estabilidad absoluta deberá establecerse por algún otro medio.[8]

4. <u>La Carta de Nichols</u>.

La función de respuesta de lazo cerrado de un sistema con retroalimentación unitaria en forma polar es:

$$\frac{C}{R}(jw) = \frac{|G(jw)|\angle\phi_G}{1+|G(jw)|\angle\phi_G}$$

haciendo |C/R(jw)|=M = constante tenemos que

---

[8] Distefano, S. Retroalimentación y Sistemas de Control. 1972.

$$|G(jw)|^2+\frac{2M^2}{M^2-1}|G(jw)|\cos\phi_G+\frac{M^2}{M^2-1}=0$$

Para un valor fijo de M, este lugar se puede obtener en 3 pasos: 1) se escogen valores numéricos para $|G(jw)|$; 2) se resuelven las ecuaciones resultantes para ϕ, excluyendo los valores de $|G(jw)|$ para los cuales $|\cos \phi|>1$; y 3) se localizan los puntos obtenidos sobre una gráfica de la magnitud en db contra el ángulo de fase.

El lugar de los puntos sobre una gráfica de magnitud en db contra el ángulo de fase para los cuales el arg C/R(jw) es una constante

$$\tan[arg\frac{C}{R}(jw)]=N$$

se define por la ecuación

$$|G(jw)|+\cos\phi_G-\frac{1}{N}sen\phi_G=0$$

Para un valor fijo de N, este lugar de puntos se puede obtener en tres pasos : (1) se escogen valores para ϕ; (2) se resuelven las ecuaciones resultantes para G(jw); y (3) se localizan los puntos obtenidos sobre una gráfica de magnitud en db contra el ángulo de fase.

5. Diseño con Cartas de Nichol.

Las representaciones de las cartas de Nichols son las más apropiadas para determinar los ajustes del factor de ganancia.

En este diseño existen diversos métodos de compensación , como el método de curva de amplitud constante, el cual se utiliza para encontrar Kb (ganancia de Bode), especificando un máximo de resonancia Mr. Esta técnica es de tipo manual y se necesita una tablilla por lo que mayor detalle se deja a la referencia bibliográfica.

D. Análisis de Variables de Estado.

1. Introducción.

El método de la función de transferencia es muy conveniente en el análisis de la frecuencia de un sistema, sin embargo su principal desventaja es que no considera las condiciones iniciales. Por lo tanto, en el proyecto moderno de sistemas de mando, que se realiza normalmente en el dominio temporal es necesario utilizar un método distinto para caracterizar el sistema.

Cuando se habla de estado se refiere al pasado, presente y futuro del sistema. Las variables de estado de un sistema lineal pueden definirse como un conjunto mínimo de variables, x1(t), x2(t)... cuyo conocimiento en un instante cualquiera to, más la información de la excitación de entrada subsiguiente es suficiente para determinar el estado del sistema en cualquier tiempo t >to[4]. Las variables de estado tienen una representación matricial de la siguiente forma:

$$\frac{dx(t)}{dt}=Ax(t)+Br(t)$$

donde x(t) es el vector columna de las variables de estado que se denominan normalmente vector de estado; r(t) es el vector columna de las variables de entrada, denominado vector de entrada. En otras palabras,

$$x(t)=\begin{bmatrix} x_1(t) \\ x_2(t) \\ \cdot \\ \cdot \\ \cdot \\ x_n(t) \end{bmatrix}$$

y A es una matriz de N * N.

---

[4] Kuo,B. 1991. Sistemas Automáticos de Control.

A la solución de la ecuación de estado se le llama ecuación de transición. Existen varios métodos para resolver dicha ecuación, como el método de utilizar la transformada de Laplace. Sin embargo estos métodos son muy difíciles de implementar por cuanto se utilizan métodos de solución de ecuaciones diferenciales de orden mayor como el método de Runge Kutta [5] para sistemas de ecuaciones diferenciales de primer orden. Este sistema se puede aplicar para la solución numérica de ecuaciones diferenciales de orden mayor sujetas a condiciones iniciales. Consiste el método en la transformación de una ecuación de orden mayor en un sistema de ecuaciones diferenciales de primer orden:

$$\frac{d^n c(t)}{dt^n} + a_1 \frac{d^{n-1} c(t)}{dt^{n-1}} + \ldots + a_n c(t) = r(t)$$

se puede representar por ecuaciones de estado,

$$\frac{dx_1(t)}{dt} = x_2(t)$$

$$\frac{dx_2(t)}{dt} = x_3(t) \ldots$$

$$\frac{dx_{n-1}(t)}{dt} = X_n(t)$$

$$\frac{dx_n(t)}{dt} = -a_n x_1(t) - a_{n-1} X_2(t) - \ldots - a_2 X_{n-1}(t) - a_1 x_n(t) + r(t)$$

[5]Richard, L. Análisis Numérico. 1985.

La ecuación de salida será siempre c(t)=x1(t).

2. Elaboración de la Matriz.

A partir de las ecuaciones de estado se puede pasar a la elaboración de la matriz de estado y los vectores de estado, llegando a una matriz de este tipo:

$$\begin{bmatrix} \frac{dx_1(t)}{dt} \\ \frac{dx_2(t)}{dt} \\ \cdot \\ \cdot \\ \cdot \\ \frac{dx_n(t)}{dt} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & 0 & 1 & 0 & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & 0 & 0 & 1 & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdot & \cdot & \cdot & 1 \\ -a_n & -a_{n-1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -a_1 \end{bmatrix} * \begin{bmatrix} x_1(t) \\ x_2(t) \\ \cdot \\ \cdot \\ x_n(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ 1 \end{bmatrix} r(t)$$

La solución de las ecuaciones de estado se denomina ecuación de transición de estado o simplemente ecuación de transición. La ecuación de estado

$$\frac{dx(t)}{dt} = Ax(t) + Br(t)$$

puede resolverse por el método clásico, o bien por la transformada de Laplace.

3. Método de resolución.

Entre los método de resolución de la ecuación de estado está el de Laplace, sin embargo se utilizan dentro de este proyecto métodos numéricos para su resolución.

Para aproximar la solución del sistema de orden m de problemas de valor inicial de primer orden, se utiliza el algoritmo de Runge Kutta para sistemas de ecuaciones diferenciales de primer orden. En este método se convierte una ecuación de orden *m*, a un sistema matricial como el anteriormente descrito. Este método está completamente implementado en el programa, por lo que se hará una referencia específica más adelante.

E. Sistema de Diagrama de Bloques.

1. Introducción.

Los diagramas de bloques son una simplificación, una representación gráfica de un sistema físico que ilustra las relaciones funcionales entre los componentes del sistema.[6]

Los diagramas de bloques, entonces, nos sirven para esclarecer y representar los sistemas fisicos, además nos ayudan para simplificarlos o bien para manipularlos de una manera tal que tengan un comportamiento deseado. Por medio de los diagramas de bloques podemos colocar nuevos elementos y pasar luego a la simulación o evaluación, observando qué parámetros se deberán modificar para

---

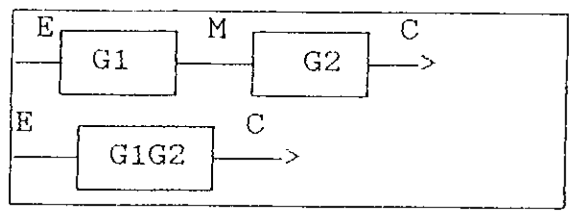
[6] Distefano, S. 1972. Retroalimentación y Sistemas de Control.

tener la respuesta adecuada.

2. Explicación de los Diagramas.

En general, un diagrama en bloque consta de una configuración especifica de cuatro tipos de elementos-bloques, puntos de suma, puntos de toma y flechas que representan el flujo unidireccional de señales.

Las cantidades en el dominio del tiempo se representan con letras minúsculas, y las letras mayúsculas se utilizan para las transformadas de Laplace. Cualquier número finito de bloques se puede combinar algebraicamente por medio de multiplicación.



Como se ha observado en el ejemplo anterior lo bloques vienen a ejemplificar un diseño físico, y por lo tanto se puede aplicar toda una álgebra de bloques para minimizarlos o simplificarlos, por lo que en este proyecto sólo se hace un programa para poder dibujarlos, asi que para el álgebra en si se debe consultar alguna de las bibliografias.

IV. MANUAL DEL USUARIO

A. <u>Generalidades del programa</u>.

El programa está construido en lenguaje C/c++, se utiliza un compilador Borland, su código objeto está orientado a procesadores 80286; sin embargo éste fue elaborado en 80386DX40. A nivel fuente es portable siempre y cuando se utilicen los mismos compiladores. Partes de los programas utilizan ambientes orientados a objetos, incluyendo técnicas como recursión y programación por módulos. Se intenta utilizar las técnicas más avanzadas para que toda persona en un futuro pueda modificar los patrones de los programas sin encontrarse con un programa obsoleto. Se ha escogido el lenguaje de programación C porque es un lenguaje de nivel intermedio, que es una ventaja por algunas aplicaciones en el hardware, no obstante la razón principal, es que es un lenguaje que tiene la tendencia a sobrevivir. Se refiere a sobrevivir porque hay una clara inclinación a que los sistemas del futuro sean computadores basados en procesadores RISC, operados la mayoría de ellos en ambiente UNIX, por lo tanto software basado en C es el que más se adapta a estos tipos de sistemas. Todo el programa que incluye sus códigos fuente ocupan aproximadamente 1.2 Mega. Las gráficas de los programas corren en monitores tipo EGA,VGA,SVGA. Se recomienda que los monitores sean a colores para poder identificar mejor los detalles y contrastes.

El programa está realizado modularmente, esto primordialmente es por la extensión del mismo y por el tiempo que consume cada parte del programa en su compilación (aunque se utilizan proyectos sin embargo al unirlos el tiempo es considerable). El programa está diseñado de una forma intuitiva para que el usuario le sea fácil navegar por él. También es necesario que la computadora donde se utilice, tenga aplicaciones con mouse y una impresora con capacidad gráfica; ésto es para la parte de del programa donde se le da la opción al usuario para hacer los diagramas de bloques.

Los tiempos de corrida del programa son muy variados y dependen de la complejidad de las ecuaciones; sin embargo los tipos de ecuaciones que se pueden ingresar no pueden ser en extremo complejas debido a que el método de evaluación de las mismas es recursivo, y por la segmentación del programa se deja un área fija para la recursión, entonces si se excede la persona de ésta área los programas no funcionarán apropiadamente. En lo particular hay aplicaciones que funcionando a 40 Mhz se observan un poco lentas, ésto se debe a que se utilizan métodos numéricos para encontrar las raíces de las ecuaciones como el algoritmo de Müller que busca raíces complejas. Estos métodos son complicados por lo que requieren tiempo de máquina. Una ventaja que tiene éste programa sobre programas similares es que se pueden ingresar ecuaciones no polinomiales, es decir se pueden ingresar funciones como cosenos, senos logaritmos exponente. Lo principal es que se pueden ingresar funciones en paréntesis como es usual que vengan en este tipo de análisis ej. (x+3,2j)(s+3)cos(2x^2+4), como se observa también se pueden ingresar expresiones complejas (números complejos), lo que hace a éste programa tan útil y flexible. Sin embargo el costo de ésta gran ventaja es el tiempo que se consume de máquina. El tiempo de operación se puede disminuir con la ayuda de un coprocesador, lo que es relativamente raro que alguna computadora de uso personal posea. Con el advenimiento de procesadores más completos como el Pentium, éstas aplicaciones serán mucho más rápidas. Algunas de las aplicaciones del programa contienen sub-menús para que el usuario pueda por ejemplo hacer énfasis en algún intervalo de interés. Sin embargo hay que hacer la aclaración que el usuario debe de tener conocimientos de los métodos antes de pasar a la aplicación.

1. Navegación por el programa.

Se refiere a la forma de poderse desplazar por todas la opciones del programa. El programa cuenta con una serie de menús y sub-menús para hacer de una forma fácil y rápida su utilización. Generalmente para poder salirse de una opción se utiliza la tecla de Escape (ESC), en vez de utilizar una función de un menú donde indique fin. Esto se hace así para darle una forma mas intuitiva y en algunas aplicaciones importantes el programa le exige al usuario confirmar el fin de la opción. Para correr el programa sólo se escribe RUN ej C:\>run ; en el apéndice A se muestra un ejemplo de lo que aparece en la pantalla.

B. Sistema gráfico.

Este sistema gráfico es la rutina del programa que tiene a su cargo los dibujos de los diagramas de bloques. Al entrar en la rutina principal, el programa se bifurca en dos opciones:

1. Análisis.
2. C.A.D.

La rutina que se describe a continuación es la correspondiente a la opción 2.

Al escoger la opción 2 no aparecerá éste número en la pantalla, de lo contrario, aparecerá inmediatamente la opción para hacer los diagramas. Esta sub-rutina tiene el objetivo de poder dibujar los diagramas, par ello se cuentan con sub-opciones. Si el usuario en algún momento no se recuerda de los comandos para dibujar, puede presionar la tecla F1. Se ha escogido esta tecla puesto que en muchos programas como hojas electrónicas utilizan esta tecla como ayuda (Help). Al presionar esta tecla aparece una serie de comandos que se pueden aplicar en esta sub-rutina, éstos comandos están descritos en las tabla 4.1. La rutina permite hacer planos de hasta 8 hojas, que es suficiente para las aplicaciones; sin embargo si se desean hacer planos superiores a esta capacidad se pueden hacer por grupos de 8 hojas.

Se debe tener en mente que esta rutina es complementaria a las rutinas de análisis por lo que únicamente está orientada a sistemas de diagramas de control. No se pretende otro fin más que el de tener una rutina de dibujo de acceso rápido y orientado al análisis que se realice. Esto es en parte debido que cuando el usuario encuentra una solución u optimización a su programa puede ir a dibujarlo rápidamente y luego imprimirlo para tener un plano con las modificaciones que se le hacen a su sistema.

Tabla 4.1

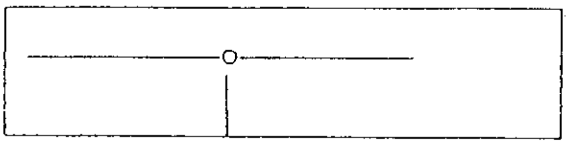
Comandos de la rutina de CAD.

Comando	Función
c	Dibuja un círculo
x	Permite el ingreso de texto
l	Dibuja líneas
b	Dibuja cuadrados
t	Dibuja triángulos
p	Puntos de unión
8	Flecha hacia arriba
6	Flecha a la derecha
2	Flecha hacia abajo
4	Flecha hacia izquierda
e	Borrar
+	Aumenta la velocidad del cursor
-	Disminuye la velocidad del cursor
s	Guarda en el disco
g	Obtiene del disco
r	Borra y redibuja la pantalla
n	Función de impresión.

En la tabla 4.1 se observa cómo se toman en cuenta muchos aspectos necesarios para poder hacer los diagramas, como los círculos, cuadrados etc. Esta rutina también tiene implementado el uso de mouse para hacer más rápido el movimiento del cursor. Al comenzar la rutina se observará una flecha blanca que es el indicador del mouse y con sólo presionar un botón el cursor se trasladará a donde está la flecha del mouse. Si por alguna razón no se tiene en disposición un mouse se puede incrementar la velocidad del cursor presionando la tecla de "+" o bien, para ciertas aplicaciones de precisión, se puede disminuir la velocidad presionando la tecla de "-". Los tamaños de los elementos de dibujo se han escogido de tal forma que confronten los tamaños que usualmente se utilizan en los libros de texto. Se tienen convenientes aspectos en la rutina como el uso de triángulos, éstos aparecen viendo hacia la derecha con un símbolo de integral dentro, para ser convenientemente utilizados como integradores dentro de la simbología de diagramas. Sin embargo, si se requiere hacer un triángulo viendo hacia el inusual lado izquierdo se puede elaborar por medio del comando de lineas, el cual está convenientemente diseñado para su uso continuo, es decir que no únicamente realiza líneas entre dos puntos sino que también se pueden dibujar diversos tipos de polígonos o figuras. Este comando actúa así:1) se oprima la tecla "l", lo que hace que se active la rutina, luego se mueve el cursor ya sea por medio del mouse o las flechas convencionales hasta el punto deseado, estando en este punto se oprime la tecla de "Enter" la cual marca el punto como perteneciente al poligono. 2) se traslada el cursor hacia otro punto de interés marcándolo con "Enter". 3) Luego de escoger varios puntos pertenecientes al polígono se oprime la tecla de "d" (draw), la cual hace que todos los puntos escogidos sean interconectados con líneas. Es decir que a la vez de funcionar en el dibujo de lineas convencionales también se puede utilizar para el diseño de diversas formas.

El ingreso de texto dentro de un sistema de diagramas es de vital importancia, por lo que se incluye esa opción dentro de los comandos. La manera de ingresar un texto se hace por medio de la tecla "x", la cual activa la rutina de ingreso de texto. Los caracteres especiales se pueden ingresar utilizando la tecla de ALT seguida por el número en consideración, luego de ingresar la cadena de simbolos se presiona la tecla de ENTER, la que hará que el texto aparezca en la forma ingresada.

Il. 4.1



En la ilustración 4.1 se indica en qué forma se colocan las bifurcaciones en las líneas por medio de puntos, este punto se ingresa por medio del comando p el cual hará que aparezca un pequeño círculo. Combinando los diversos comandos se pueden obtener diagramas de bloques completo. Las flechas se utilizan para mostrar el flujo de las señales y se recomienda utilizarlas para dar la claridad específica a los diagramas.

En el apéndice A se muestra un ejemplo de un diagrama de bloques.

1. Comandos de dibujos.

La tabla 4.1 da los resúmenes de los comandos utilizados, aquí se da una explicación de cada uno. Para que el programa funcione correctamente hay que colocar los comandos en minúsculas.

"c" Circle.

Este comando tiene la función de dibujar un círculo en la pantalla, el programa también tiene la opción que si el círculo queda entre páginas, se dibujarán circulos en cada una de ellas para que se complementen entre páginas. El círculo tiene en sistemas de control la función de agrupar señales por lo que se requiere generalmente junto con los circulos los dibujos de lineas con sus respectivas flechas para indicar los sentidos de los flujos. También se requiere el uso de texto para poder aclarar los signos de los flujos.

"x" Text.

En esta opción se activa la sub-rutina para el ingreso del texto, por lo que una vez activada, se procede a escribir el mensaje deseado, asi también se pueden utilizar caracteres especiales o símbolos especiales utilizando la tecla ALT seguido por la serie de números que en ASCI tengan una representación.

"l" Line.

Este comando es más completo que hacer una simple línea. También con este comando se pueden hacer figuras como polinomios. 1) se utiliza la tecla "l" para activar esta subrutina. 2) se utiliza la tecla de "ENTER" para marcar los puntos de los polígonos. 3) se presiona la tecla "d" (draw) para hacer que los puntos marcados sean unidos por líneas. De esta manera se observa cómo se pueden hacer figuras complicadas por medio de la combinación de comandos. Se debe hacer notar que cuando se utiliza una línea, ésta debe ir finalizada por un indicador del sentido de flujo y es cuando se deben de aplicar los comandos que dibujan flechas.

"b" Box.

Este comando tiene como objetivo dibujar cuadros, los cuadros son utilizados en sistemas de control para indicar algún proceso, por lo que no se deben utilizar aisladamente, sino combinados con los comandos de texto para ingresar dentro de los cuadros los procesos que se desean representar. El tamaño del cuadro es fijo y está basado en los tamaños usuales que se necesitan para los diseños de control. Si se desean cuadros más grandes se puede proceder a utilizar el comando de "l" para dibujar poligonos y de esta manera crear cuadros del tamaño adecuado a las necesidades del usuario.

"p" Point.

Este comando hace pequeños circulos que se utilizan para unir las líneas, como en la ilustración 4.1. Sirven entonces para hacer más claro los diseños y saber con exactitud de donde provienen los flujos.

"8" Up.

Para poder hacer más claros las direcciones de los flujos se utilizan pequeñas flechas que van generalmente al final de las líneas. Se escoge entonces la tecla "8" para representar la flecha hacia arriba, se tiene ésta tecla porque convencionalmente puede substituir la tecla de flecha Up arrow (flecha arriba).

Para utilizar esta tecla, primero se debe de colocar el cursor en la posición donde se debe de colocar la flecha y luego presionar la tecla "8".

"6" Right.

Esta tecla, como la anterior, sirve para hacer flechas hacia la derecha.

"2" Down.

Sirve para dibujar flechas hacia abajo.

"4" Left.

Nos sirve para hacer flechas hacia la izquierda.

"r" redraw.

Esta función nos va a servir para poder redibujar la pantalla. Lo que sucede en este caso es que al pasar el cursor sobre cierto elemento, el cursor a su paso puede dejar cierto rasgo, por lo que se utiliza esta tecla para volver a colocar la pantalla como antes. También este comando tiene otra función, que es borrar el o los elementos donde está el cursor, es decir que cuando se desee borrar todo un elemento sólo se requiere colocar el cursor por encima del elemento a borrar y presionar la tecla r.

"e" Erase.

Esta función sirve para borrar sectores de los dibujos, es decir puede o no borrar total o parcialmente un elemento. Esta tecla no se utiliza sola, va combinada con otros funciones. 1) se presiona la tecla "e" para activar la sub-rutina. 2) se coloca el cursor en la esquina superior izquierda del elemento a borrar y se presiona la tecla "Enter". 3) se coloca el cursor en la esquina inferior izquierda y se presiona "Enter", esto dará como resultado la eliminación de un cuadro desde el punto marcado por el primer "Enter" hasta el segundo punto.

"+" More.

Esta función sirve para aumentar la velocidad del cursor, esto tiene la función de hacer más rápida la navegación dentro del plano del diseño. La tecla hay que presionarla varias veces para darle la velocidad requerida al cursor. En cierta forma el proceso es que el desplazamiento del cursor será más amplio en cuanto más se presione repetidamente esta tecla.

"-" Less.

Esta función es lo contrario a la anterior, lo que hace ésta es disminuir la velocidad del cursor; de forma que al presionar repetidamente la tecla, así irá disminuyendo la velocidad del cursor. Esta tecla tiene aplicaciones cuando se desee colocar o borrar un símbolo que se encuentra en una posición precisa, por lo que hay que bajar la resolución de movimiento al cursor para poder alcanzar dicha posición.

"s" Save.

Esta instrucción nos servirá para guardar el plano actual en el disco. Lo cual es muy útil para próximos usos o modificaciones de los planos que se ejecuten. Si se desea guardar en diferentes unidades de disco o sub-directorios se debe colocar todo el patrón que identifique la unidad de almacenamiento o directorio. Se recomienda utilizar la extensión "cnt" (control) para colocarla en los archivos. Esto hará a la larga más identificable el directorio para el usuario.

"g" Get.

Este comando sirve para sacar del disco y directorio seleccionado algún proyecto o plano de interés.

"t" Triangle.

Esta instrucción nos servirá para dibujar triángulos. El triángulo en este programa representará un proceso de integración por lo que al utilizar esta tecla aparecerá un triángulo con una integral en su interior. El triángulo tendrá una orientación hacia la derecha, que es el sentido usual de los integradores en sistemas de control. Si el usuario necesita un triángulo con una orientación diferente puede recurrir a las funciones de línea para hacer los polígonos.

"n" **Print.**

Esta función se encarga de mandar lo que tiene la hoja actual del proyecto a la impresora. Esta función es una de las más lentas pues se tiene que cambiar la impresora a una alta resolución lo que hace el proceso de impresión más lento de lo normal.

2. Consideraciones sobre los diagramas.

Se debe tomar en cuenta que esta rutina de diagramas es de carácter complementario al análisis, por lo que se ha diseñado para ser una herramienta para dibujo. Si el usuario necesita hacer alguna aplicación que requiere mayor grado de especialización debe recurrir a software más complejo. Sin embargo lo que hasta aquí está implementado es suficiente para la elaboración de los sistemas de diagramas de bloques más usuales o convencionales.

C. Descripción de la rutina de Lugar de las Raíces.

La rutina corresponde propiamente a la parte de análisis de los sistemas de control. El objetivo de esta rutina es encontrar los polos y ceros de GH, el centro de asíntotas, obtener las asíntotas, graficar para distintos valores de k, y a la vez hacer una aproximación para los coeficientes de amortiguamiento.

Se recuerda aquí que la función a evaluar será

$$GH=\frac{KN(s)}{D(s)}$$

$$\frac{C}{R}=\frac{G}{1+GH}$$

donde k es una constante de diseño que acompaña al numerador de la ecuación. Estas raices son graficadas en el plano s. El factor de amortiguación viene siendo el factor de ganancia k requerido para dar una razón de amortiguación especificada τ donde

$$\phi=\cos^{-1}\tau$$

Así se obtiene la recta desde el origen a un ángulo de más o menos ϕ con el eje real negativo. El factor de ganancia en el punto de intersección con el lugar de las raíces es el valor requerido de K. Ver apéndice A.

1. Obtención de la gráfica.

Para llegar hasta esta rutina es necesario escoger entre dos opciones:

1. ANALISIS

2. CAD

La opción a utilizar es la 1. que se refiere al análisis. Luego aparecerá otro menú que es propiamente del análisis de sistemas:

ANALISIS DE SISTEMAS

1.- Lugar de raíces

2.- Nyquist

3.- Nichol

4.- Estado

Se escoge en este caso la opción número 1, con la cual podremos ingresar a lo qué es la rutina del lugar de las raíces. En el apéndice A se muestran los menús y los ejemplos de las distintas rutinas.

Al escoger la opción 1 tendremos un pantalla con un recordatorio de que tipo de ecuaciones se están analizando, así como un pequeño diagrama de bloques. Para pasar a la siguiente pantalla sólo se oprime la tecla de "Enter", se le preguntará al usuario en que intervalos desea trabajar, ésto es necesario para que el programa sepa dónde buscar las raices de las ecuaciones, cómo es un método numérico donde las funciones pueden ser en su totalidad complejas. Se aplica en este caso el algoritmo de Müller para encontrar las raíces en general; el algoritmo se basa en tres aproximaciones iniciales con la cual se construye una parábola y la siguiente aproximación la considera en intersección de la parábola con el eje x.[7] Cuando se ingresan los intervalos, el programa se encarga de encontrar las aproximaciones y elaborar las parábolas. En el apéndice A se muestra un ejemplo para resolver

$$GH=\frac{K}{s(s+1)(s+3)(s+4)}$$

se observa del ejemplo que el intervalo de trabajo es de -5 a 5, aunque todas las raices están en el lado negativo (así debe ser). Se hace la advertencia que se necesita un monitor a colores para poder observar las asíntotas para cuando $K > 0$ y $K < 0$; las asíntotas para $K > 0$ son de color azul, en cambio para $K < 0$ son de color café. Las asíntotas son los lugares donde tiende a estar las raíces para valores grandes de k. También el programa da el centro de asíntotas, que es el lugar donde convergen todas las asíntotas.

Esta rutina tiene un sub-menú:

ingrese k = K; esc; a = amortigua

---

[7] Richard, L. 1985. Análisis Numérico.

ingrese k:

Esta función tiene como objetivo ingresar un valor de k, luego calcular las raíces y representarlas en el plano complejo. Para activar la función sólo se debe presionar la tecla "k"; luego el programa le preguntará algún valor de k que el usuario desee para su diseño, entonces el programa calculará las raíces de esta nueva posición y la representará en el plano complejo por medio de pequeñas cruces.

esc:

La opción tiene como objetivo poder salirse de la rutina o terminarla.

a= **amortiguación:**

Esta opción tiene como finalidad encontrar el factor de ganancia requerido K para dar una razón de amortiguamiento τ. Este coeficiente de amortiguación debe de estar comprendido entre -1 y 1. Para activar la sub-rutina se debe de presionar la tecla "a", entonces el programa le preguntará el coeficiente de amortiguación; el programa luego calcula el ángulo correspondiente a este factor de amortiguación y graficará una recta. El valor de diseño de k es la intersección de esta recta con el lugar de las raíces.

D. Descripción de la rutina de Nyquist.

La rutina de Nyquist tiene como objetivo estudiar la estabilidad, para ello necesitamos la función de transferencia de lazo abierto GH(s), a la cual se le hará el análisis; el objetivo es pues primero obtener cual será la trayectoria de Nyquist. La trayectoria de Nyquist comprenderá el semiplano derecho, luego de encontrar la trayectoria procederemos a pasarla al plano, cuyo eje vertical es Im GH y el horizontal es el Re GH. Cuando se encuentra una singularidad en la trayectoria inicial de Nyquist hace un pequeño rodeo a la singularidad; el rodeo de que se habla es simplemente un semicírculo de radio infinitesimal.

Al obtener la representación en el plano de la trayectoria de Nyquist procedemos al criterio de la estabilidad en este caso lo que realmente observamos es que sí se ha rodeado al punto -1,0.

1. Ingreso de función.

En el apéndice A se muestra un ejemplo de la utilización de la rutina de Nyquist. Lo primero que pregunta es el intervalo de trabajo, que nos servirá para encontrar los polos y ceros y poder dibujar la trayectoria de Nyquist. Luego le pide al usuario que ingrese el numerador de la función de transferencia GH. En el ejemplo del apéndice A la función a analizar es:

$$\frac{e^{-1s}}{1s+0.333}$$

donde el numerador es la función exponencial, la función exponencial está ejemplificada para demostrar cómo, dentro de éste método, se pueden aplicar funciones que impliquen retardos. En algunos casos los retardos en sistemas muy complicados no se pueden expresar especificamente, y para simplificar el análisis se puede representar el retardo por funciones del tipo exponencial con constantes de tiempo provenientes del análisis experimental.

Al ingresar en la rutina la función de transferencia, el programa procede a encontrar la trayectoria de Nyquist, si encuentra una singularidad trazará un pequeño semicírculo alrededor el punto en cuestión.

La siguiente pregunta que hace la rutina es: cuál es el intervalo de estudio. Este intervalo de estudio viene a representar el área que se trazará con una mayor precisión. La precisión del trazo es la siguiente pregunta que viene a continuación, la precisión la debe ingresar el usuario según su conveniencia, sin embargo se debe decir que a mayor precisión se requiera, mayor es el tiempo de consumo de máquina. Ha que tomar en cuenta que el programa se especializa en la solución de funciones del tipo complejo (que contienen expresiones y números complejos), lo que hace su ejecución un poco más lenta; sin embargo, a la larga resulta un costo necesario para el correcto análisis.

Tabla 4.2

Operaciones al ingresar la función.

Operaciones	Función
(exp1)+(exp2)	Suma el contenido de exp1 con exp2
(exp)(exp)	Multiplica las expresiones
(exp1)/(exp2)	Divide las expresiones
(exp1)^(exp2)	Eleva la exp1 a la exp2
EXP(exp)	Obtiene el exponente de la exp
LN(exp)	Obtiene el logaritmo natural de exp
SE(exp)	Obtiene el seno de la expresión
CO(exp)	Obtiene el coseno de la expresión

En la tabla 4.2 se muestra un condensado de los tipos de funciones que se pueden ingresar para todas las rutinas.

Dentro de la rutina de Nyquist existe un sub-menú donde se pregunta,

individual i.

Otro o.

Escape.

Individual.

Dentro de esta opción se le pregunta al usuario un número en particular, que se representa dentro del plano por medio de una equis.

Otro.

Esta opción sirve para ingresar una nueva función y pasar a obtener su representación.

Escape.

Para poder salir de la opción se debe presionar la tecla de escape (ESC).

2. Resultados y Optimizaciones.

En este caso de la rutina de Nyquist, se puede observar de las gráficas cómo es que se contiene o no al punto -1,0; para ello, la rutina graficará un círculo con centro en el origen que abarque el punto para facilitar al usuario la verificación visual del criterio. Para Optimizar el sistema se pueden hace modificaciones al diseño y , por lo tanto, a su función de transferencia para que no contenga al punto -1,0.

E. Descripción de la rutina de Nichols.

El análisis de la estabilidad relativa es muy práctica utilizando este método. La ventaja de la representación al utilizar las cartas de Nichols es que se pueden representar un intervalo mucho más amplio de amplitudes, ya que se utiliza una escala logarítmica.

1. Obtención de las cartas.

Para poder activar la rutina se debe escoger explícitamente la opción de las Cartas de Nichols. Lo primero que pregunta el programa es si desea las cartas o no, esto se refiere a los gráficos para los cuales se da las siguiente condiciones:

$$\left| \frac{C}{R}(jw) \right| = M = constante$$

$$\tan\left(arg\frac{C}{R}(jw)\right) = N = constante$$

que representan superficies de los lugares de la magnitud en db y el ángulo de fase constante. Sin embargo el programa debe construir estas superficies lo que puede consumir una gran cantidad de tiempo, esta es la razón principal de la pregunta.

2. Ingreso de la función.

Luego se procede a ingresar la función a evaluar, se puede ingresar atendiendo a la tabla 4.2. Luego que se ha ingresado la función comienza a realizar las gráficas, obteniendo los márgenes de frecuencia y fase que nos darán una indicación de la estabilidad relativa.

Al terminar las gráficas aparecerá un sub-menú donde se le preguntará al usuario

Otra fun. o.

Escape.

Si el usuario presiona la tecla "o", el programa permitirá el ingreso de una nueva función, si se ingresa escape (tecla ESC), se saldrá de la rutina. Se debe tomar en cuenta, que la forma de hacer la rutina es por medio de transformaciones de coordenadas, en el apéndice A se muestra cómo se ingresa la función de transferencia.

$$\frac{6}{6-4w^2+3wj-w^3j}$$

De este ejemplo se puede observar qué tan poderoso es la posibilidad de ingresar números y funciones complejas. La rutina tiene como objetivo, intentar que el usuario se sienta cómodo a la hora de ingresar las funciones en una forma que se intenta de lo más natural.

Del apéndice A se muestra cómo se debe ingresar una función del tipo complejo. Si se quiere ingresar el número complejo 3wj, a la hora de ingresar la función es de esta forma 0,3w y el programa analizará en una forma recursiva la evaluación de lo ingresado. Existen ciertas limitantes en lo que se puede ingresar, cómo las rutinas están orientadas a la solución de números complejos, todo número o función se evalúa como un número complejo; de tal forma que si se desea ingresar la función -x^2 se debe ingresa como +-1x^2, se debe recalcar que se debe anteponer un signo menos a cualquier número complejo para identificar que es un número real. si por ejemplo se desea ingresar la función -x^2j se debe ingresar así: +0,-1x^2. De este último ejemplo se demuestra como todas las funciones se toman como si fueran complejas, ésto es debido a que las exigencias de los sistemas de control piden en su mayoría funciones de éste tipo.

3. Resultados y Optimizaciones.

A partir de los márgenes de ganancia y fase se puede decidir sobre la estabilidad relativa del sistema, y con ayuda de pequeñas modificaciones e puede encontrar mejores soluciones a los sistemas. DE acuerdo con esto la rutina tiene una gran aplicación en la optimización pies por un método de ir buscando la solución que más se adecue a nuestras necesidades, podemos ensayar varias veces nuestras soluciones en el programa

F. Descripción de las Variables de Estado.

Esta rutina es una de las más poderosas en la investigación sobre la respuesta en el tiempo de un sistema, en general un sistema se puede representar, por medio de un conjunto diferencial de ecuaciones, definida por variables de estado del sistema. Con el conjunto de ecuaciones diferenciales se puede obtener un sistema matricial de ecuaciones de estado. Este sistema matricial se puede resolver por medio de transformadas de Laplace, sin embargo el objetivo de la rutina es resolverlos por métodos clásicos de ecuaciones diferenciales.

El método utilizado en la rutina es el de Runge-Kutta para sistemas de ecuaciones, el método escogido es el de " Cuatro pasos" que es una variante del de Runge Kutta que es más complicado pero más exacto. Los demás métodos de análisis hasta ahora estudiados, se preocupaban mucho de la estabilidad relativa y como optimizar los sistemas, sin embargo, el método de las variables de estado nos da la respuesta real de comportamiento del sistema, con ayuda de la rutina se pueden examinar los sobre-tiros, tiempo de recuperación y tiempo de estabilización de una respuesta.

Una ventaja más de este método es que aquí podemos ingresar condiciones iniciales de operación que es algo común en sistemas generales físicos. Por medio de las condiciones iniciales de operación se puede trazar el comportamiento del sistema utilizando para ello aproximaciones sucesivas.

1. Descripción de la rutina.

La rutina se activa con la previa selección del usuario. Al activarse la rutina comienza una serie de preguntas al usuario sobre el sistema, se puede hacer un conveniente análisis del funcionamiento de la rutina con un pequeño ejemplo:

Supongamos una ecuación diferencial de éste tipo:

$$1/2y'+y''=-\frac{1}{2}e^{-\frac{1}{2}x}\cos(x)$$

Luego el usuario si es el caso de una ecuación como la anterior procede a elaborar las matrices de estado:

$$x1=y$$
$$\frac{dx1}{dt}=x2$$
$$\vdots$$

luego se puede construir todo el sistema:

$$\begin{bmatrix} \dfrac{dx_1}{dt} \\ \dfrac{dx_2}{dt} \end{bmatrix} = \begin{bmatrix} 0 & +1 \\ -1 & -\dfrac{1}{2} \end{bmatrix} * \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -\dfrac{1}{2} e^{-\frac{1}{2}x} \cos(x) \end{bmatrix}$$

recordamos siempre que la solución de dicho sistema es:

$$y = e^{-\frac{1}{2}} sen(x)$$

El programa en un principio nos pide la matriz de estado recordando su formulación.

$$mat. = .Ax + Bt$$

luego pregunta la matriz A, preguntando cada una de sus posiciones en este caso la matriz A está dada por:

$$\begin{bmatrix} 0 & 1 \\ -1 & -\dfrac{1}{2} \end{bmatrix}$$

luego pregunta el vector que depende de "t" en este caso también podemos sustituir t por x.

La ventaja de la rutina es que pregunta individualmente el vector y se pueden ingresar funciones y no únicamente coeficientes, así el usuario se ahorra ingresar muchos vectores para diferentes funciones.

A continuación el programa pregunta sobre el intervalo de trabajo, el intervalo de trabajo como su nombre lo indica es el segmento donde se aplicará el análisis.

Luego viene la pregunta sobre un entero N, éste entero N nos indica la precisión que tendrá el programa en el análisis dentro del intervalo, es decir cuanto mayor sea el número más preciso será el análisis.

Por último pregunta sobre las condiciones iniciales que es donde se observa la capacidad de esta rutina, podemos obtener las aproximaciones iniciales de la ecuación diferencial, o si es de un sistema físico de las condiciones en la que lo forcemos a trabajar. Al final nos pregunta sobre los topes en el eje Y que sirve al programa para aislar mejor el segmento que se requiere de estudio.

En el apéndice A se muestran algunos ejemplos de aplicación.

La rutina tiene como objeto hacer las diferentes gráficas del comportamiento del sistema en el tiempo por lo que dibuja varias gráficas según el sistema de colores, comenzando a dibujar la respuesta al sistema diferencial como sus derivadas.

IV CONCLUSIONES Y RECOMENDACIONES

- En el estudio, los diferentes métodos presentan ventajas y desventajas por lo que no todos los métodos son adecuados para un análisis determinado.

- El estudio de la estabilidad relativa para que sea completo debe de utilizarse más de un método de análisis.

- Para obtenerse una panorámica total de estudio, luego de aplicar los métodos, debe ser complementado con un análisis de estado.

- No existe un sistema óptimo general sino que debe de efectuarse un proceso de ensayo de la optimización comprobando según los requerimientos del sistema cual es la solución o intervalo conveniente.

- Cuando es posible hacer una variación de los parámetros dentro del sistema diseñado se debe ensayar redes de atraso o adelanto para optimizar los sistemas.

V. BIBLIOGRAFIA

Burdem, R. Análisis Numérico.
1985 Iberoaméricana, México

Distefano, J. Retroalimentación y Sistemas de
1981 Control. Mc Graw Hill. México.

Dorn, W. Numerical Methods with Fortran IV case
1976 Studies. Willey. USA.

Friedland, A. Control System Design.
1986. Mc Graw Hill. USA.

Kuo, B. Sistemas Automáticos de Control.
1991 CECSA. México. 824 pp.

Ogata, K. Ingeniería de Control Moderna.
1980 Prentice Hall. México. 902 pp.

Oualline, S. Advance C Programming.
1992 Brady Publishing. USA.

Schildt, H. C/C++ The Complete reference.
1992 Mc. Graw Hill. USA.

Schildt, H. Born to code C.
1991 Mc. Graw Hill. USA.

Schildt, H. The Art of C .
1991 Mc. Graw Hill. USA.

Schildt, H. ANSI C.
1991 Mc. Graw Hill. USA.

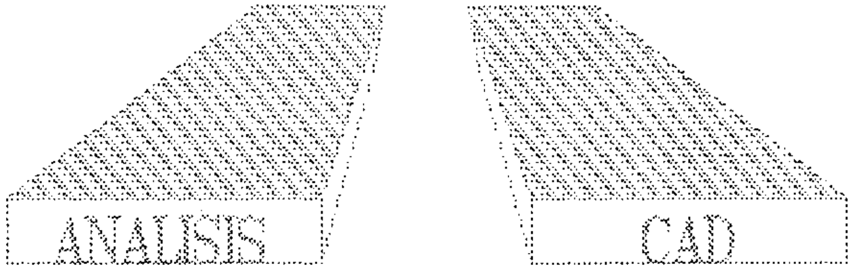
Vosu, G. Object Oriented Programming.
1991 Mc. Graw Hill. USA.

APENDICE A

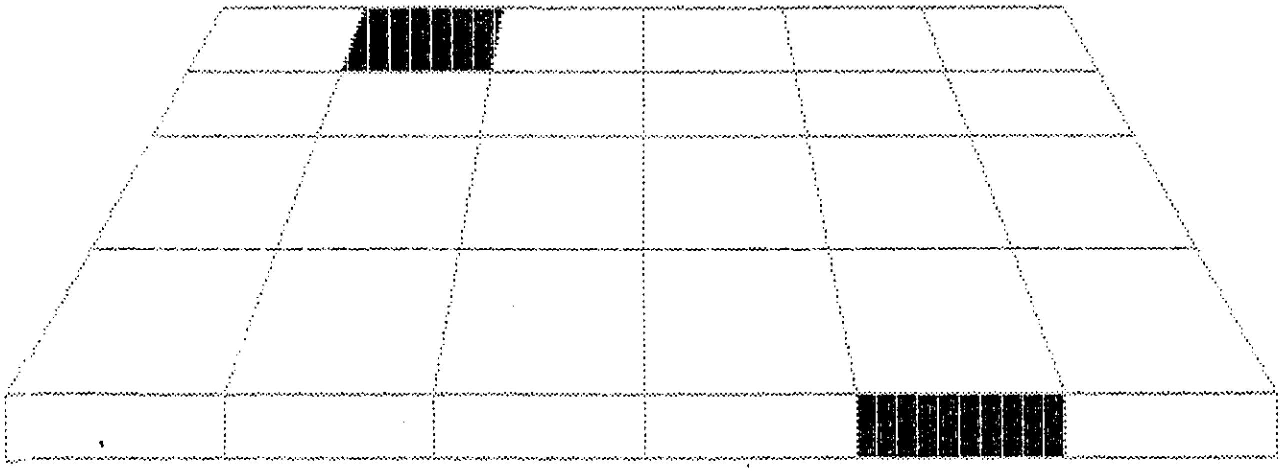
Ejemplos de corridas de las diferentes rutinas.

GERDAN PROYECT

By Manuel Fernández



Optimización De Sistemas



√7

Analisis De Sistemas

1–Lugar de raices

2–Nyquist

3–Nichol

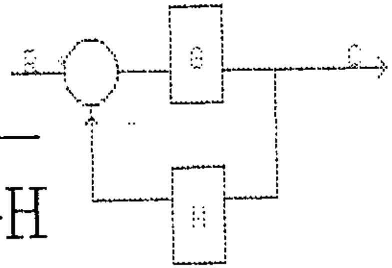
4–Estado

Cual es su opcion –>

A. Ejemplo de Lugar de las Raíces

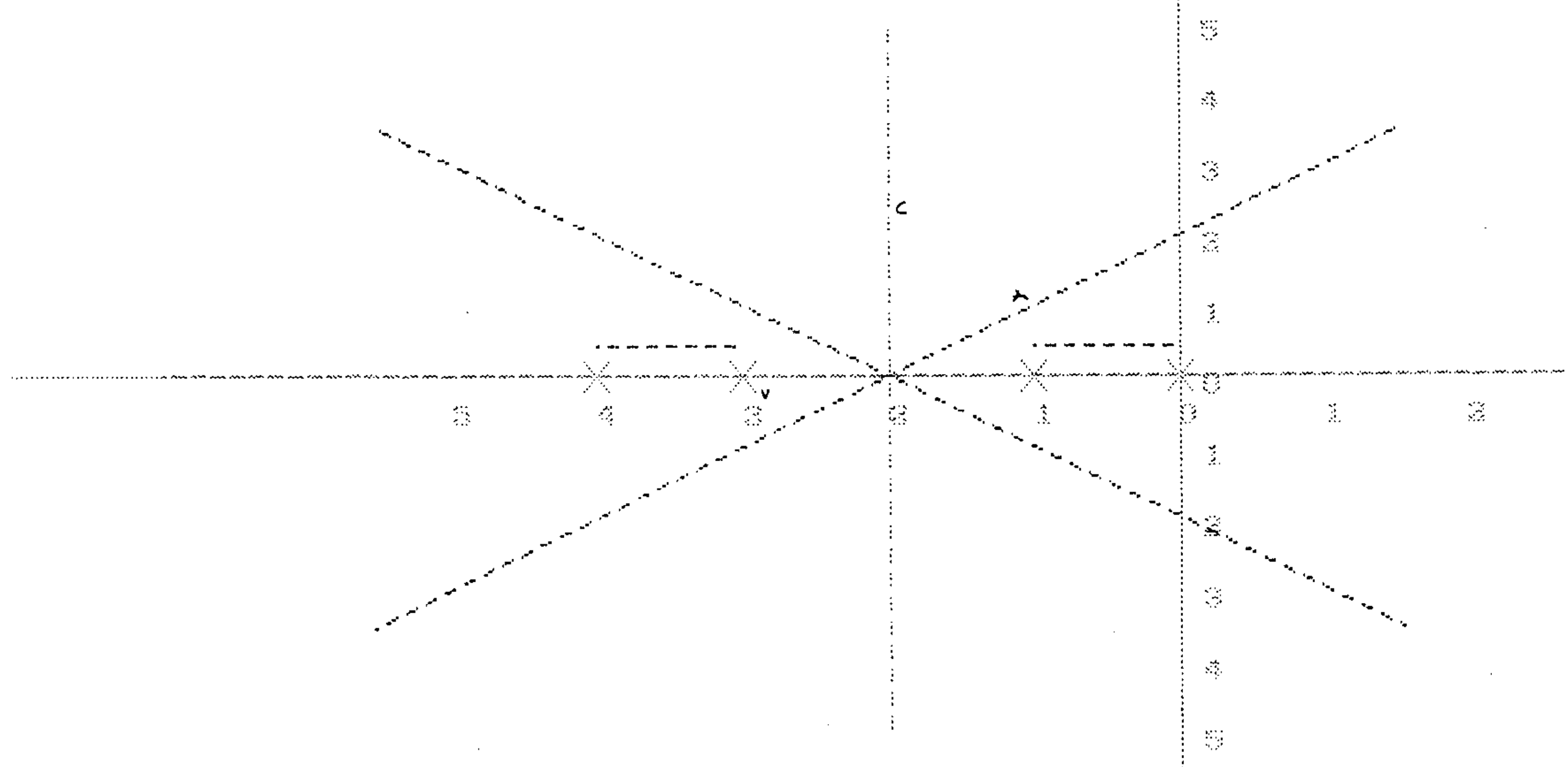
Lugar de las Raices

$$\frac{C}{R} = \frac{G}{1 + GH}$$



$$GH \equiv \frac{K\,N(s)}{D(s)}$$

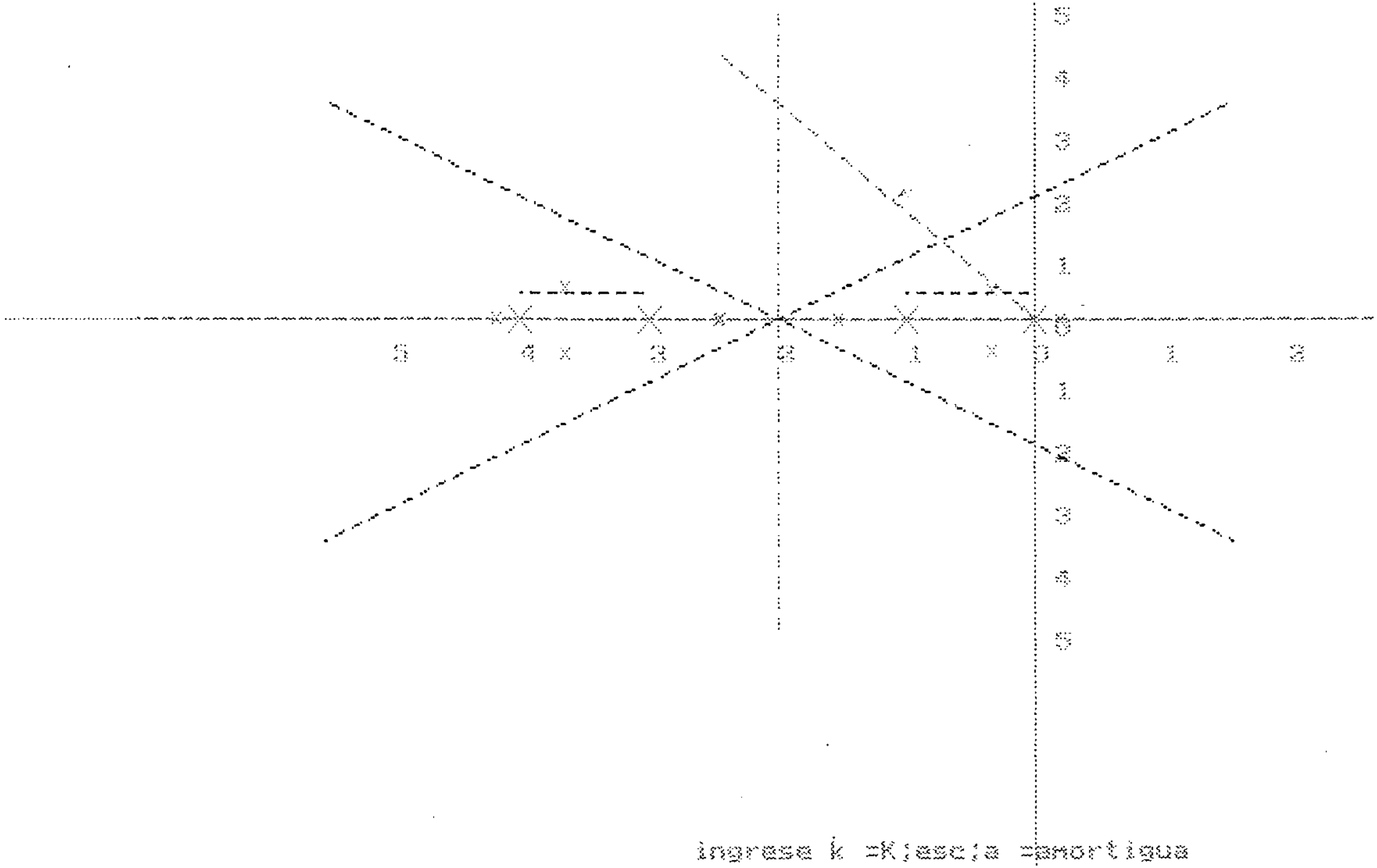
ingrese (entero)inter.. [a...b] a? 3
b?5
ingrese N(s)1
ingrese D(s)(1s)(1s+1)(1s+3)(1s+4)
solucion de N(s)-5-4-3-2-1012345no tiene solucsolucion de D(s)-5-4-3-2-1012345(-4.000042, 0)(-3.000108, 0)(-1.000302, 0)(0, 0)
centro asintotas(-2.000113, 0)



INGRESE K ; ESC ; a= amortiguación

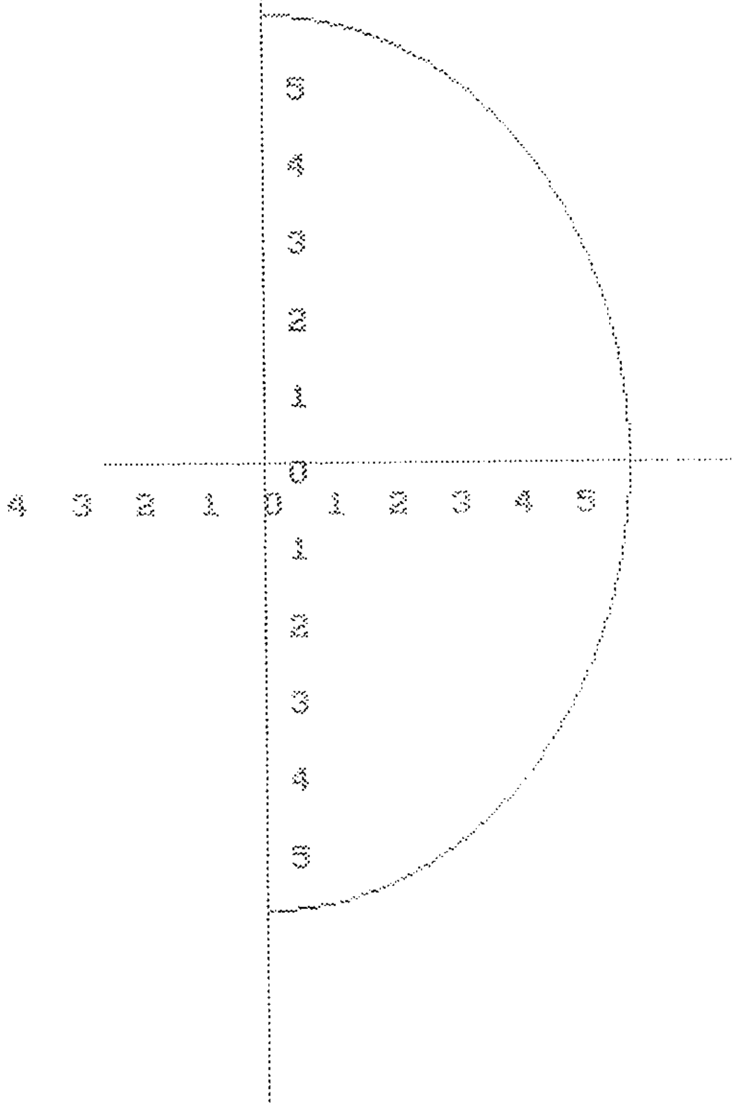
61

4.000042, 0)(-3.000108, 0)(-1.000302, 0)(0, 0)
centro asintotas(-2.000113, 0)ingrese k5
-5-4-3-2-1012345ingrese k-3
-5-4-3-2-1012345ingrese amortigua -1,1.5



B. Ejemplo de Nyquist

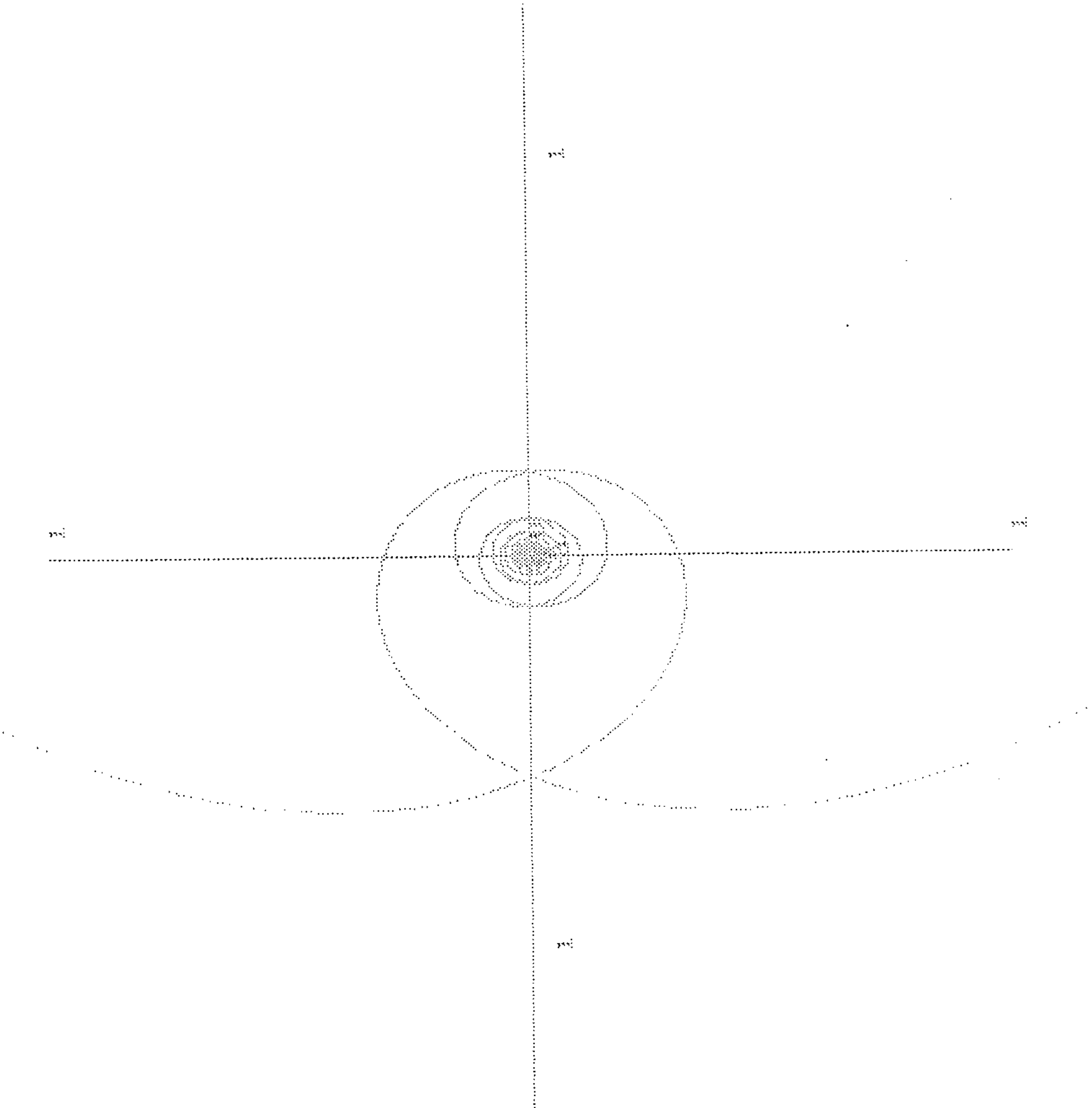
```
ingrese N(s)EXP(-1s)
ingrese D(s)(1s+0.333)
solucion de D(s)-3-2-101232
(-0.333, 0)nueva Funcion enter(EXP(-1s))/((1s+0.333))) termina<
```



6a

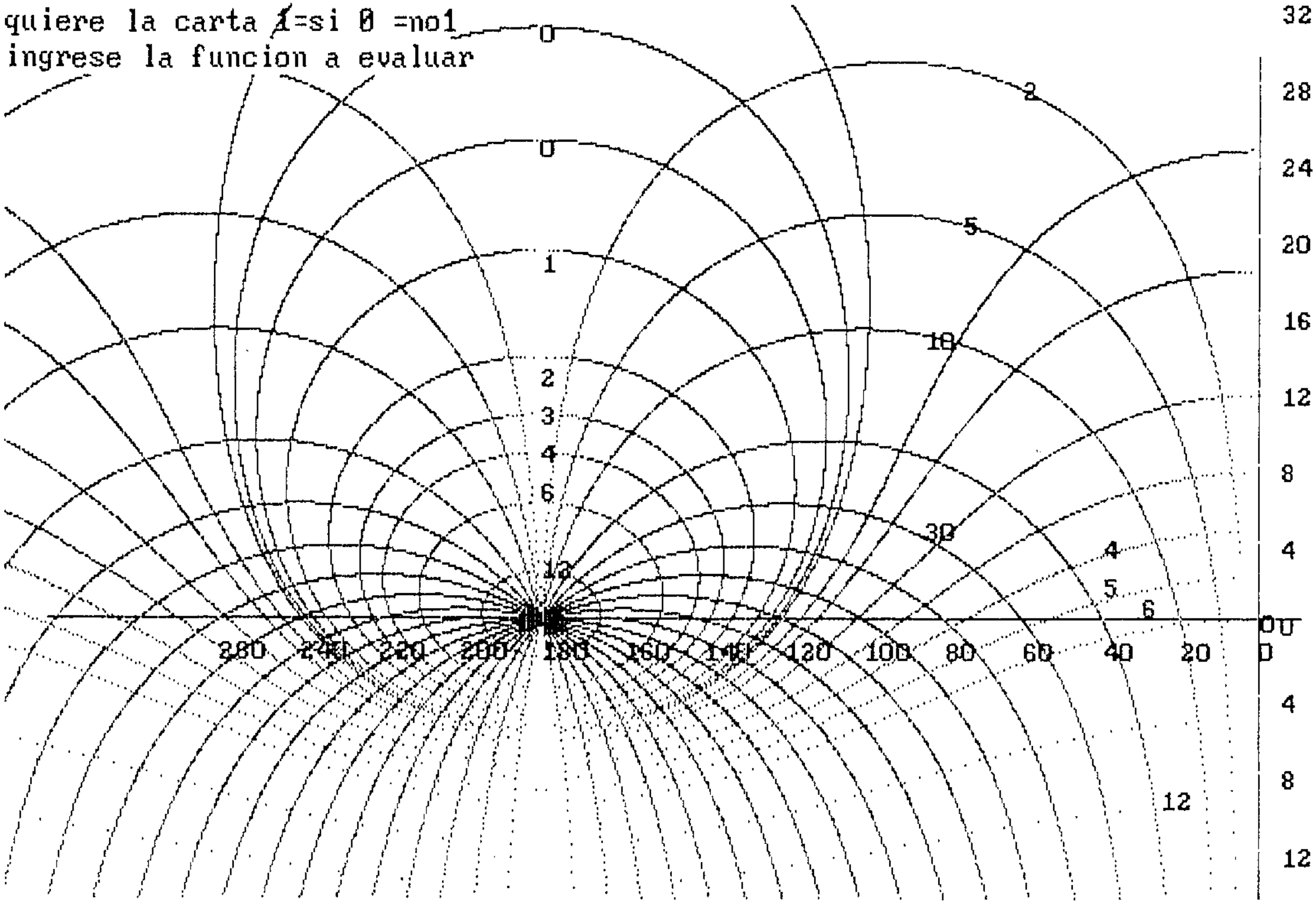
```
e intervalo de precision[a,b] incremento ia->-100
b->100
incremento.01
```

65

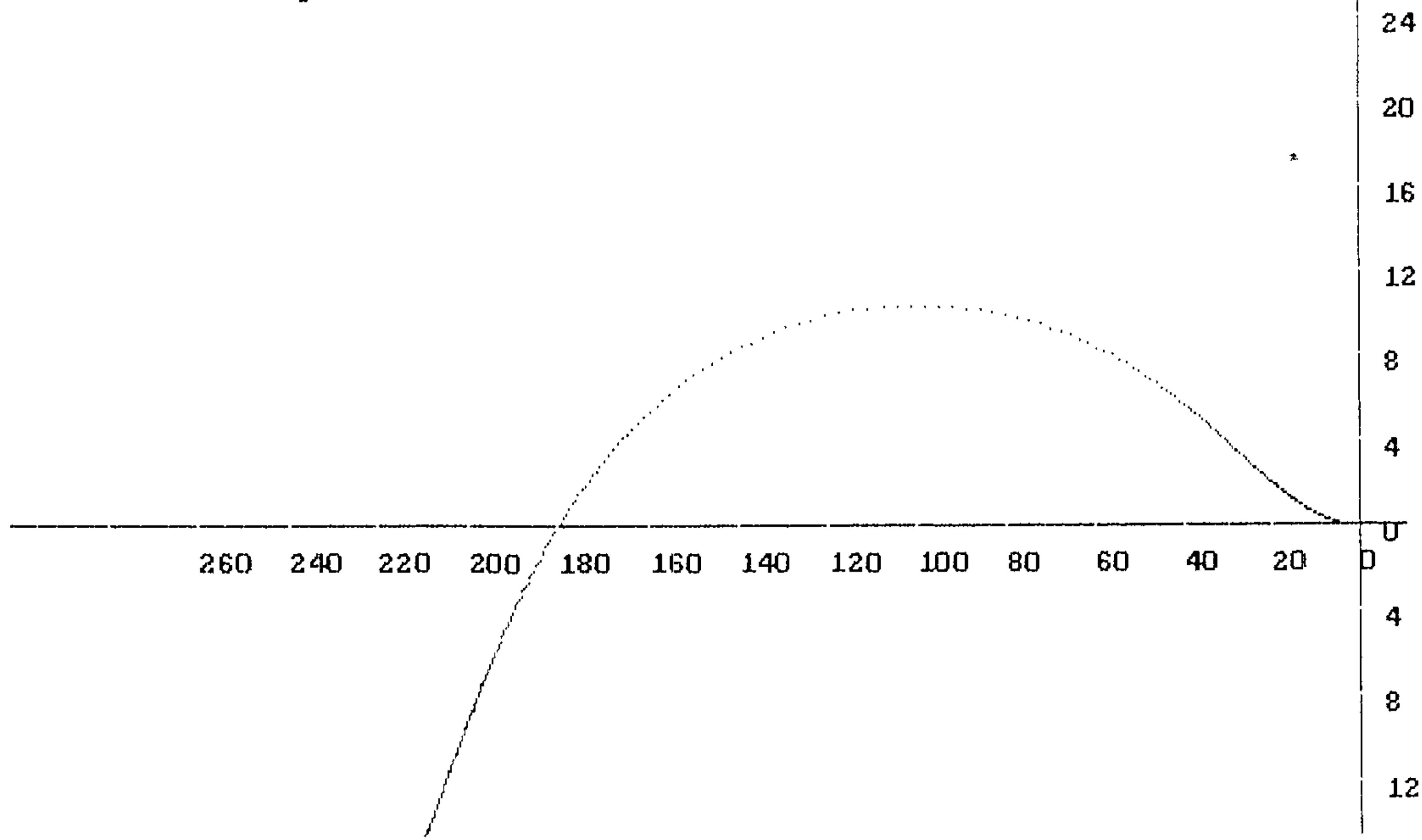


C. Ejemplo de Nichols

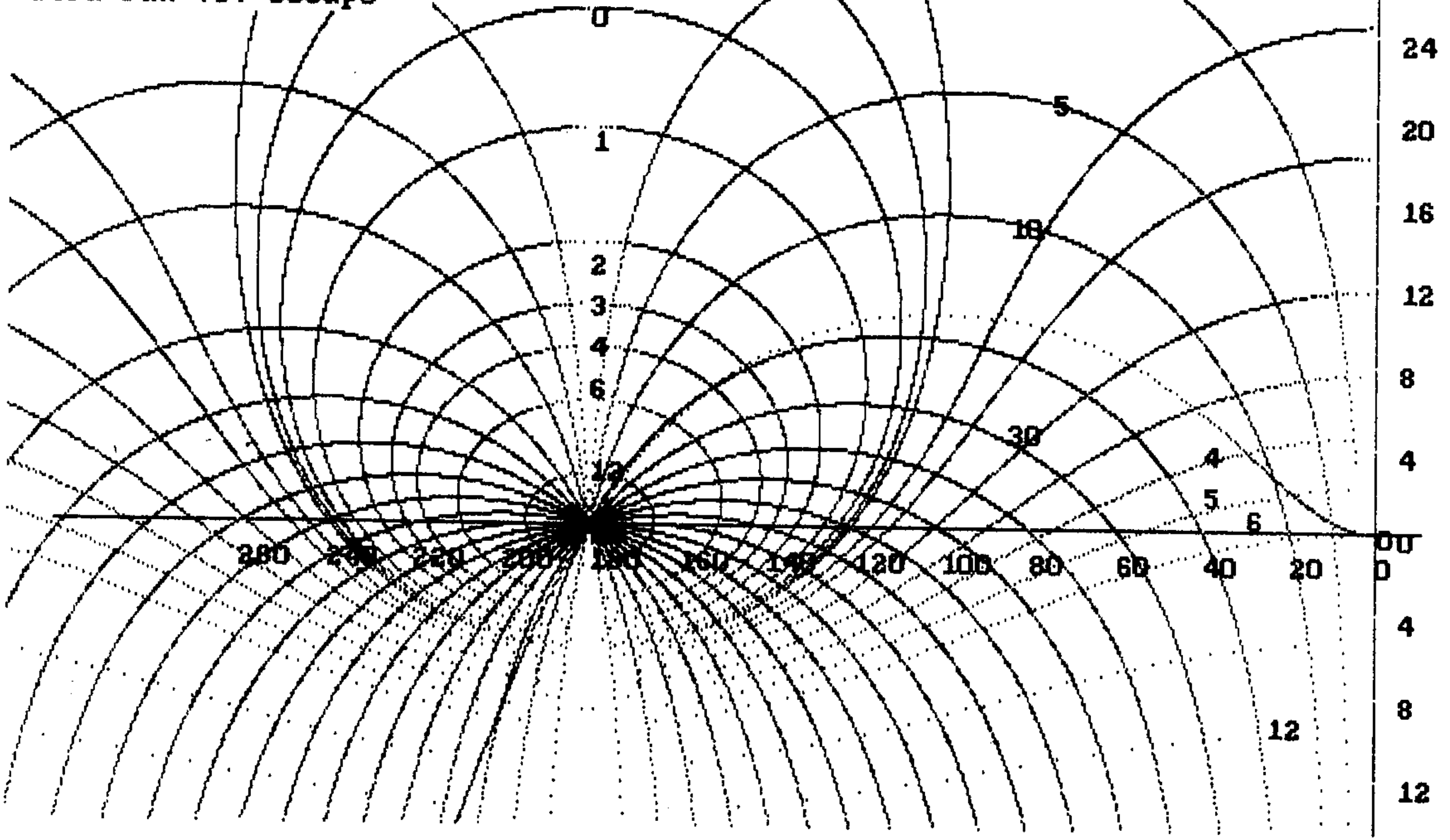
quiere la carta 1=si 0 =no1
ingrese la funcion a evaluar



quiere la carta 1=si 0 =no0
10
ingrese la funcion a evaluar(6)/(6+-4w^2+0,3w+0,-1w^3)
grecuencia de cruce -180 =0cruce por cero deb =1.729999quiere individual .i.
otra fun .o. escape

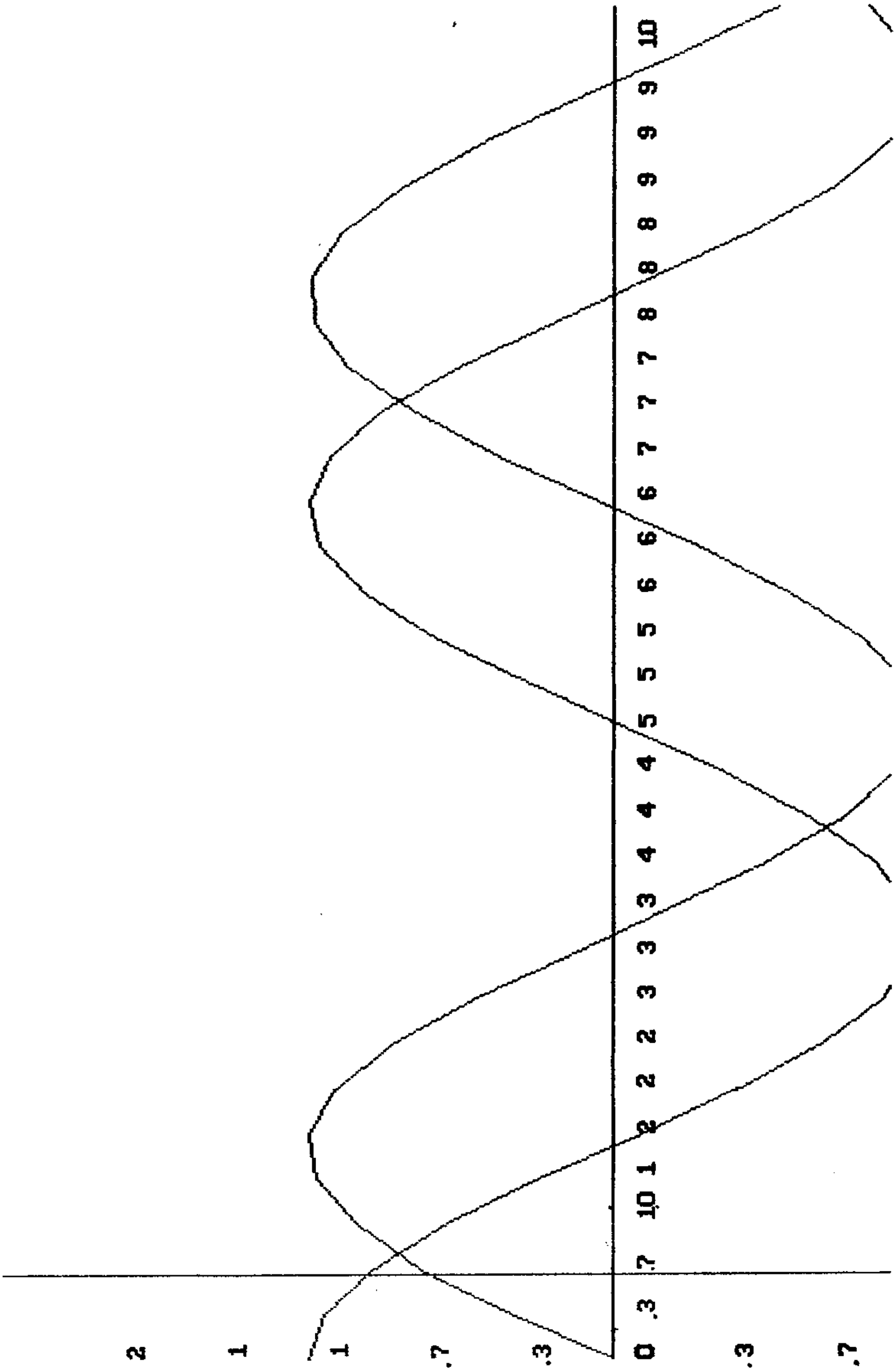


quiere la carta 1=si 0 =no1
ingrese la funcion a evaluar(6)/(6+-4w^2+0,3w+0,-1w^3)
grecuencia de cruce -180 =0cruce por cero deb =1.729999quiere individual .i.
otra fun .o. escape

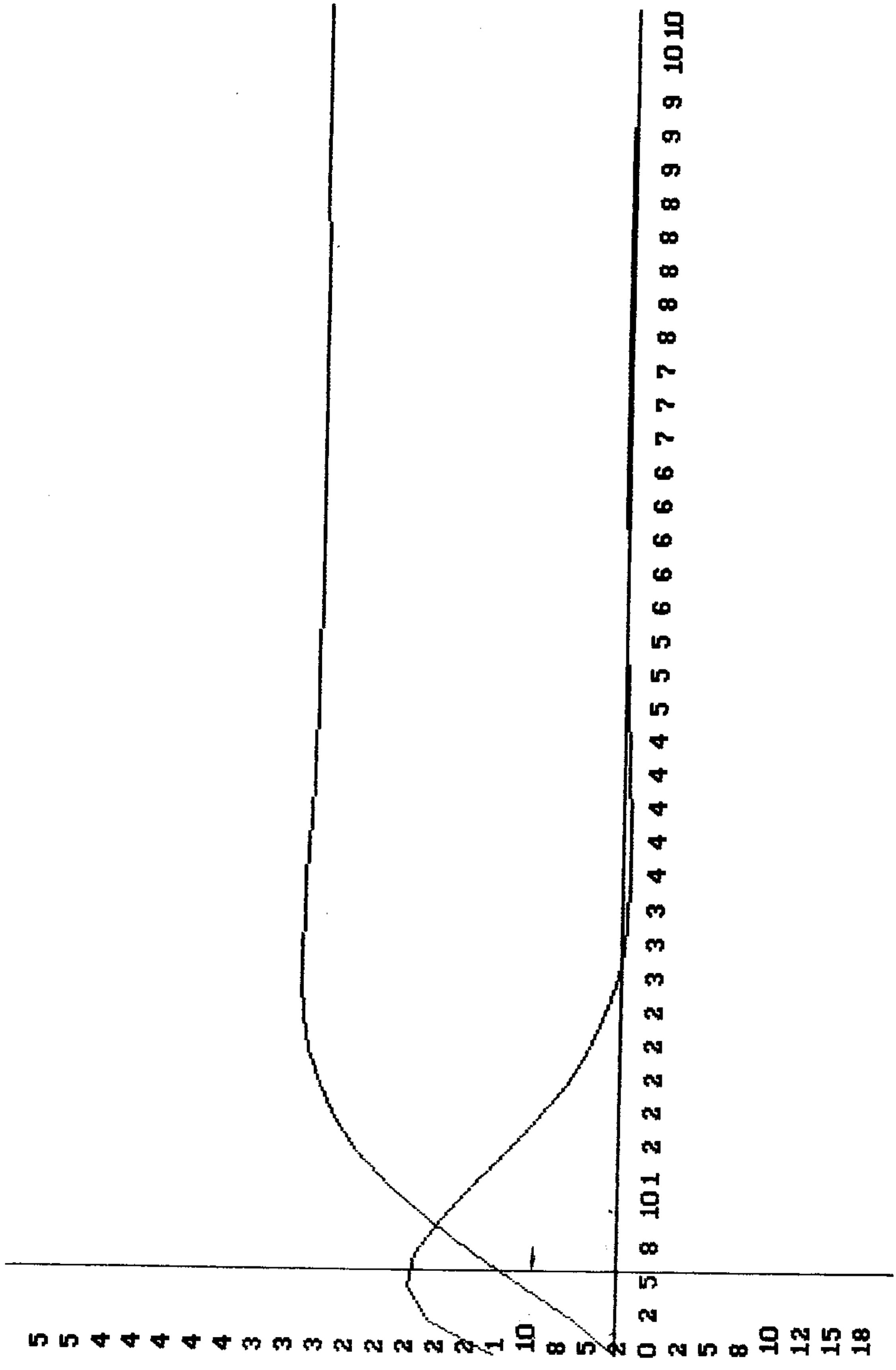


D. Ejemplo de Variables

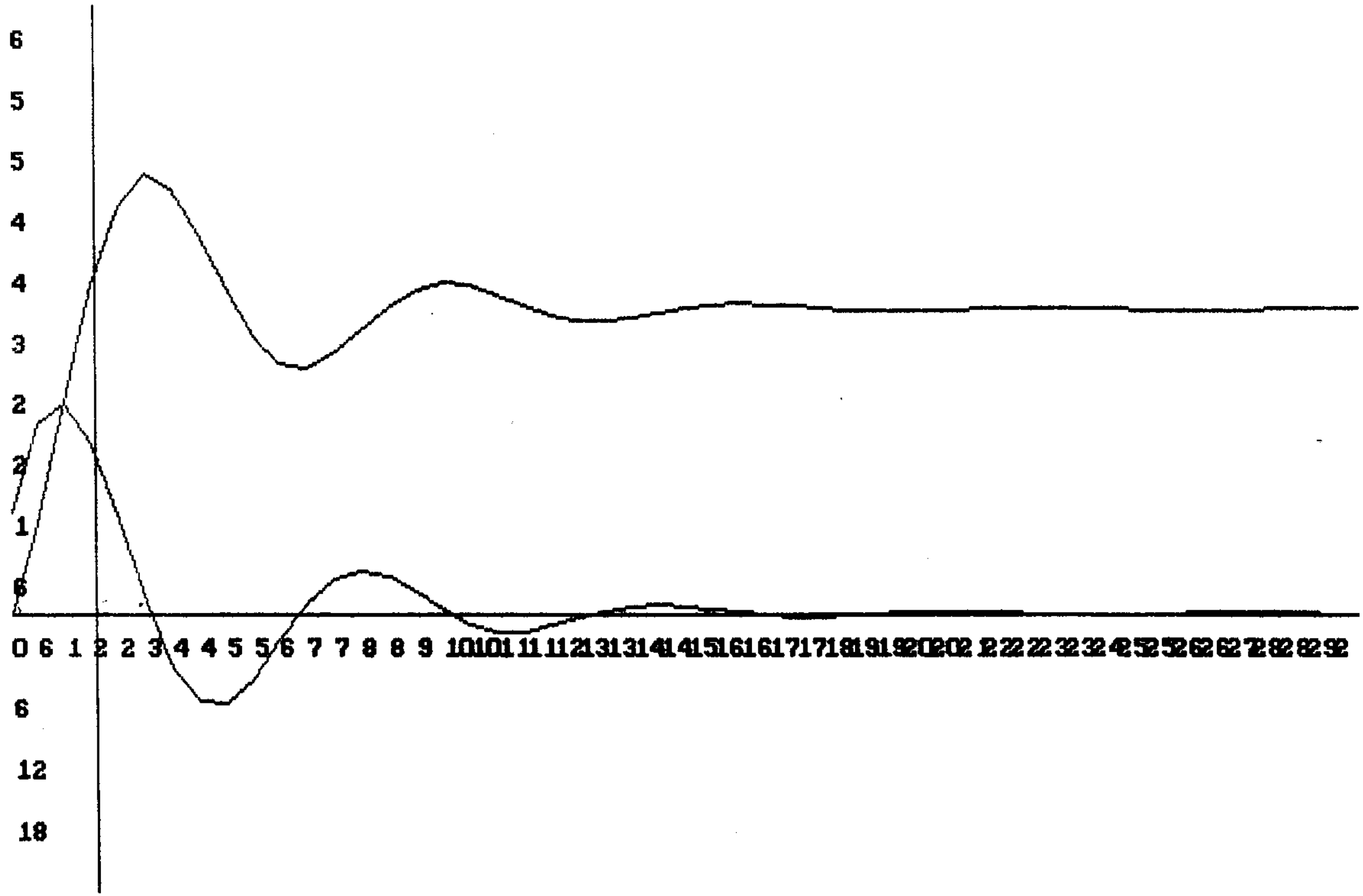
```
 ingreso dela matriz de estado
 mat = AX+Bt  la mat A es de n,n cuanto es n ?..
2
es la matriz de esta forma convencional 1=si
posicion 1 10
posicion 1 21
posicion 2 1-1
posicion 2 20
 ingrese las f(t)  en las respectivos vectoresingrese el vector 1
0
ingrese el vector 2
0
 cual es el intervalo de trabajo  ?  [a,b]
 ingrese a ..  --- >0
 ingrese b ... ---->10
 entero N para h=(b-1)/n
30
 ingrese las  aproximaciones de las ecuaciones
 aproximacion a la ecuacion 1
0
 aproximacion a la ecuacion 2
1
ingrese los topes para la grafica en y2
```


0 .3 .7 10 1 2 2 2 3 3 3 4 4 4 5 5 5 6 6 6 7 7 7 8 8 8 9 9 9 10
2 1 1 .7 .3 0 .3 .7

```
 ingreso dela matriz de estado
 mat = AX+Bt  la mat A es de n,n cuanto es n ?..
2
es la matriz de esta forma convencional 1=si
posicion 1 10
posicion 1 21
posicion 2 1-2
posicion 2 2-2
 ingrese las f(t)  en las respectivos vectoresingrese el vector 1
0
ingrese el vector 2
5
 cual es el intervalo de trabajo  ?  [a,b]
 ingrese a ..  --- >0
 ingrese b ... ---->10
 entero N para h=(b-1)/n
30
 ingrese las  aproximaciones de las ecuaciones
 aproximacion a la ecuacion 1
0
 aproximacion a la ecuacion 2
1
ingrese los topes para la grafica en y5
```

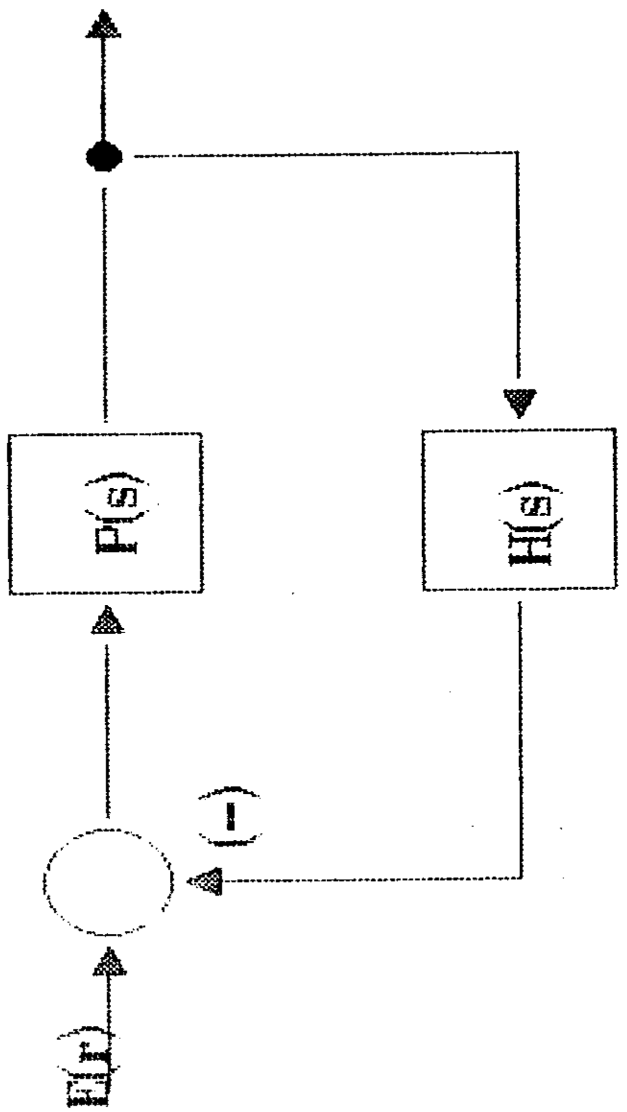


```
 ingreso dela matriz de estado
 mat = AX+Bt  la mat A es de n,n cuanto es n ?..
2
es la matriz de esta forma convencional 1=si
posicion 1 10
posicion 1 21
posicion 2 1-1
posicion 2 2-0.5
 ingrese las f(t)  en las respectivos vectoresingrese el vector 1
0
ingrese el vector 2
(+-0.5)(EXP(+-0.5x))(CO(1x))+(3)
 cual es el intervalo de trabajo  ?  [a,b]
 ingrese a ..  --- >0
 ingrese b ... ---->30
 entero N para h=(b-1)/n
50
 ingrese las  aproximaciones de las ecuaciones
 aproximacion a la ecuacion 1
0
 aproximacion a la ecuacion 2
1
ingrese los topes para la grafica en y6
```



6
5
5
4
4
3
2
2
1
6
0 6 1 2 2 3 4 4 5 5 6 7 7 8 8 9
6
12
18

E. Ejemplo de CAD



F. Ejemplo de Programa

```
ude <alloc.h>
ude <ctype.h>
ude <stdio.h>
ude <string.h>
ude <dos.h>
ude <iostream.h>
ude <stdlit.h>
ude <graphics.h>
uoe <conio.h>
ude <time.h>
ass Location{
blic:
t X,Y;
cation(int x,int y){
x,Y=y;};

ass shape :public Location{
    private:
    int Page;
    int m1[5];
    public:
    shape():Location(X,Y){};
    void Ppage(int page){Page=page;};
    int Gpage(void) {return Page;};
    int Gcor(int x,int y) {
     if (x>=m1[1] && y>=m1[2] && x<=m1[3] && y<= m1[4]) return 1;
     else
     return 0;
    };
    void Pcor(int a,int b,int c,int d){
       m1[1]=a;m1[2]=b;m1[3]=c;m1[4]=d;
    };
    void virtual Show(void){};
    void virtual Hide(void){};
    void virtual MKPoly(int Xo,int Yo,int Number){};
    void virtual MKBorrar(int exs,int eys,int eis){};


ass Nothing:public shape{
    public:
    void Show(){};


ass Text :public shape{
    int eq,ye;
    char M[20];
    public:
    Text(int x,int y, char* Me){eq=x;ye=y;
    strcpy(M,Me);
    cout<<Me;};

    void Show(){
    outtextxy (eq,ye,M);
    };

ass Circle :public shape{
    int R;
    public:
    Circle(int equis,int ye, int radius){
    X=equis;
    Y=ye;
    R=radius;
    };
```

```
  X=x,Y=y;};
  };

  class shape :public Location{
        private:
        int Page;
        int m1[5];
        public:
        shape():Location(X,Y){};
        void Ppage(int page){Page=page;};
        int Gpage(void) {return Page;};
        int Gcor(int x,int y) {
         if (x>=m1[1] && y>=m1[2] && x<=m1[3] && y<= m1[4]) return 1;
         else
         return 0;
        };
        void Pcor(int a,int b,int c,int d){
           m1[1]=a;m1[2]=b;m1[3]=c;m1[4]=d;
        };
        void virtual Show(void){};
        void virtual Hide(void){};
        void virtual MKPoly(int Xo,int Yo,int Number){};
        void virtual MKBorrar(int exs,int eys,int els){};
 };

 class Nothing:public shape{
       public:
       void Show(){};
 };

 class Text :public shape{
       int eq,ye;
       char M[20];
       public:
       Text(int x,int y, char* Me){eq=x;ye=y;
       strcpy(M,Me);
       cout<<Me;};

       void Show(){
       outtextxy (eq,ye,M);
       };
};
 class Circle :public shape{
       int R;
       public:
       Circle(int equis,int ye, int radius){
       X=equis;
       Y=ye;
       R=radius;
       };
       void Show(){
       circle(X,Y,20);
       };
 };
class Rect :public shape{
     int Xa,Ya,Fa,Fb;
      public:
        Rect (int Xo,int Yo, int Xf,int Yf){Xa=Xo,Ya=Yo.Fa=Xf.Fb=Yf;}
        void Show(){
        rectangle(Xa,Ya,Fa.Fb);
       };
       };
```

```
class Borrar :public shape{
        int pv[5];
        int numera;
        public:
        Borrar (int l){
         numera=l;
        };
        void MKBorrar(int exs,int eys,int els){
         pv[els]=exs;
         pv[els+1]=eys;
        };
        void Show(){
           for (int iu=pv[1];iu<=pv[3];iu++){
            for (int yu=pv[2];yu<=pv[4];yu++){
              putpixel(iu,yu,0);
            };
          };


        };
};

class Inte :public shape{
       int poly2[10];
       int X1o,Y1o;
       public:
        Inte (int Xo,int Yo){
        X1o=Xo;Y1o=Yo;
        poly2[0] =Xo-30;
        poly2[1] =Yo-30;
        poly2[2] =Xo+30;
        poly2[3] =Yo;
        poly2[4] =Xo-30;
        poly2[5] =Yo+30;
        poly2[6] =Xo-30;
        poly2[7] =Yo-30;
        poly2[8] =Xo-30;
        poly2[9] =Yo+30;
        };
        void Show(){
         setcolor(3);
         outtextxy(X1o-16,Y1o-9,"⌠");
         outtextxy(X1o-20,Y1o+4,"⌡");
         setcolor(4);
         drawpoly (4,poly2);
        };
 };

class Poly :public shape{
      int poly[50];
      int Numero;
      public:
      Poly (int Number){
       Numero =Number;
       };
      void MKPoly(int Xo,int Yo,int Number){
       Numero = Number;
       Number = Number *2;
       poly [Number]=Xo;
       poly [Number+1]=Yo;
      };
      void Show(){
      drawpoly(Numero+1,poly);
```

```
      int eq,ye;

      public:
       Point (int x, int y){eq=x;ye=y;};
       void Show(){
        setfillstyle(SOLID_FILL,1);
        circle (eq,ye,5):
        floodfill(eq,ye,4);
       };
};
class APoint:public shape{
        int eq,ye,pos;
        public:
        APoint (int x,int y,int posi){eq=x,ye=y,pos=posi;};
        void Show(){
        setfillstyle(SOLID_FILL,2);
        if (pos ==1){
         line (eq+5,ye,eq-5,ye-5);
         line (eq+5,ye,eq-5,ye+5);
         line (eq-5,ye-5,eq-5,ye+5);
        };

        if (pos ==2){
         line (eq+5,ye+5,eq,ye-5);
         line (eq-5,ye+5,eq,ye-5);
         line (eq-5,ye+5,eq+5,ye+5);
        };
        if (pos ==3){
         line (eq-5,ye,eq+5,ye-5);
         line (eq-5,ye,eq+5,ye+5);
         line (eq+5,ye-5,eq+5,ye+5);
        };

        if (pos ==4){
         line (eq-5,ye-5,eq,ye+5);
         line (eq+5,ye-5,eq,ye+5);
         line (eq-5,ye-5,eq+5,ye-5);
        };
        floodfill(eq,ye,4);
};
};
    class Cursor:public Location{
     protected:
     int Dx,Dy;
     public:
     Cursor(int x,int y):Location(x,y){};
     void Pos(int x,int y){
      X=x;Y=y;};
     void Show(){
      setcolor(4);
      line (X-5,Y-5,X+5,Y+5);
      line (X-5,Y+5,X+5,Y-5);
     }
     void Hide(){
      setcolor(getbkcolor());
      line (X-5,Y-5,X+5,Y+5);
      line (X-5,Y+5,X+5,Y-5);
     };

    };
class Mouse :public Location{
 union REGS r;
 public:
```

```
        int86(0x33, &r, &r);
       r.x.ax=1;
       int86(0x33,&r,&r);
      };
      void MouseOff(){
       r.x.ax=2;
       int86(0x33,&r,&r);
      };
      void MousePos(){
       r.x.ax=3;
       int86(0x33,&r,&r);
       X=r.x.cx;
       Y=r.x.dx;
      };
     int Mgetx(){
      return X;
      };
     int Mgety(){
      return Y;
     };
  };
oid hel(void);
oid M(void);
oid Analisis(void);
oid Cad(void);
nsigned long sample[100];
ain()
   {
   int gdriver=DETECT,gmode;
   initgraph(&gdriver,&gmode,"c:\borlandc\bgi...");
enu:    cleardevice();
    Cad();
    };
/ ****************************RUTINA DE CAD DE CONTROL ***************

oid Cad(){
ILE *fp;
nion REGS r;
nt poly[20];
har mes[20];
har blank[20]={""};
nt x,y,quit,flag,conta,A,page,sac,i,fig,fig1,fig2,band,sample[100],tex,figa,s
a;
nt m1[5];
hape *p[50];
ircle Cir(20,20,20);
hape u;
nt (shape::*pa)(void);
oid (shape::*pp)(int pag);
oid (shape::*tc)(int a,int b, int c,int d);
nt (shape::*gc)(int x,int y);
c=&shape::Pcor;
c=&shape::Gcor;
a=&shape::Gpage;
p=&shape::Ppage;
or (i=1;i<100;i++)sample[i]=0;
ettextjustify (CENTER_TEXT,CENTER_TEXT);
ettextstyle (TRIPLEX_FONT,HORIZ_DIR,1);
=1;
ex=1;
age=0;
loba=0;
 =320;
```

```
Mouse M;
setactivepage(page);
setvisualpage(page);
Cursor Kurs(x,y);
cleardevice();
Kurs.Show();
M.MouseOn();
quit = 0;
flag =0;
band =0;
page=0;
sac=0;


while (!quit){
 if (kbhit()){
  switch(getch()){
     case 72:{y-=A;
      Kurs.Hide();
      Kurs.Pos(x,y);
      Kurs.Show();
      break;
     };
     case 80:{y+=A:
      Kurs.Hide();
      Kurs.Pos(x,y);
      Kurs.Show();
      break;
     };
     case 75:{x-=A;
      Kurs.Hide();
      Kurs.Pos(x,y);
      Kurs.Show();
      break;
     };
     case 77:{x+=A;
      Kurs.Hide();
      Kurs.Pos(x,y);
      Kurs.Show();
      break;
     };
     case 'c':{

//      sample[iex]=1;
//      sample[iex++]=x;
//      sample[iex++]=y;
//      sample[iex++]=page;
      p[fig]=new Circle(x,y,30);
     (p[fig]->*pp)(page);
      (p[fig]->*tc)(x-30,y-30,x+30,y+30);
      p[fig]->Show();
      fig++;
/       if ((640-x)<30){
/        p[fig]=new Circle(x-640,y,30);
/        (p[fig]->*pp)(page+1);fig++;
/       };
/       if ((480-y)<30){
/       p[fig]=new Circle (x,y-480,30);
/       (p[fig]->*pp)(page+4);fig++;
/       };
/       if ((640-x)<30 && (480-y) <30){
/       p[fig]=new Circle (x-640,y-480,30);
//      (p[fig]->*pp)(page+5);fig++;
//      };
```

```
      p[fig]=new Circle(x,480+y,30);
      (p[fig]->*pp)(page-4);fig++;
     };
     if (x<30 && page >0){
     p[fig]=new Circle (640+x,y,30);
     (p[fig]->*pp)(page-1);fig++;
     };
     if (x<30 && y<30 && page >4 ){
     p[fig]=new Circle (640+x,y+480,30);
     (p[fig]->*pp)(page-5);fig++;
     };

     if ((x+30)>640 && y<30 && page >3 ){
     p[fig]=new Circle (x-640,480+y,30);
     (p[fig]->*pp)(page-3);fig++;
     };

     if (x<30 && (y+30)>480 && page >0 ){
     p[fig]=new Circle (640+x,y-480,30);
     (p[fig]->*pp)(page+3);fig++;
     };

  break;
 };

  case 'b':{
   p[fig]=new Rect(x-30,y-30,x+30,y+30);
   p[fig]->Show();
   (p[fig]->*pp)(page);
   (p[fig]->*tc)(x-30,y-30,x+30,y+30);
   fig++;

    if ((x+30)>640){
     p[fig]=new Rect(x-30-640,y-30,x+30-640,y+30);
     (p[fig]->*pp)(page+1);fig++;
    };
    if ((x-30)<0 && page>0){
   p[fig]=new Rect(640+(x-30),y-30,640+x+30,y+30);
   (p[fig]->*pp)(page-1);fig++;
    };
    if ((y+30)>480){
    p[fig]=new Rect(x-30,(y-30)-480,x+30,(y+30)-480);
    (p[fig]->*pp)(page+4);fig++;
   };

     if ((y-30)<0 && page >3){
     p[fig]=new Rect(x-30,480+y-30,x+30,480+y+30);
     (p[fig]->*pp)(page-4);fig++;
    };
    if ((x-30)<0 && (y+30)>480){
    p[fig]=new Rect(640+x-30,y-30-480,x+30+640,y+30-480);
    (p[fig]->*pp)(page+3);fig++;
    };
    if ((x+30)>640 && (y+30)>480){
    p[fig]=new Rect(x-30-640,y-30-480,x+30-640,y+30-480);
    (p[fig]->*pp)(page+5);fig++;
    };
   if ((x+30)>640 && (y-30)<0){
   p[fig]=new Rect(x-30-640,480+y-30,x+30-640,480+y+30);
   (p[fig]->*pp)(page-3);fig++;
   };
   if ((x-30)<0 && (y-30)<0){
   p[fig]=new Rect(640+x-30,480+x-30,640+x+30,480+y+30);
   (p[fig]->*pp)(page-5);fig++;
```

```
case '6':{
 p[fig]=new APoint(x,y,1);
 p[fig]->Show();
 (p[fig]->*tc)(x-10,y-10,x+10,y+10);
 (p[fig]->*pp)(page);
 fig++;
 break;
};
case '8':{

 p[fig]=new APoint(x,y,2);
 p[fig]->Show();
 (p[fig]->*pp)(page);
 (p[fig]->*tc)(x-10,y-10,x+10,y+10);
 fig++;
 break;
};
case '4':{
 p[fig]=new APoint(x,y,3);
 p[fig]->Show();
 (p[fig]->*pp)(page);
 (p[fig]->*tc)(x-10,y-10,x+10,y+10);
 fig++;
 break;
};
case '2':{
 p[fig]=new APoint(x,y,4);
 p[fig]->Show();
 (p[fig]->*pp)(page);
 (p[fig]->*tc)(x-10,y-10,x+10,y+10);
 fig++;
 break;
};

case 'x':{

  *mes=*blank;
   cin >> mes;
   p[fig] =new Text(x,y,mes);
    p[fig]->Show();
    (p[fig]->*pp)(page);
    (p[fig]->*tc)(x-10,y-5,x+10,y+5);
    fig++;
    break;
  };

case 't':{
  p[fig]=new Inte(x,y);
  p[fig]->Show();
 (p[fig]->*pp)(page);
 (p[fig]->*tc)(x-20,y-20,x+20,y+20);
  fig++;
  break;
};
case 'p':{
 p[fig]=new Point(x,y);
 p[fig]->Show();
 (p[fig]->*pp)(page);
 (p[fig]->*tc)(x-5,y-5,x+5,y+5);
 fig++;

 break;
```

```
case 59:
hel();
break;

case 'e':{
  if(band !=1){
   p[fig]=new Borrar(0);p[fig]->MKBorrar(x,y,1);band=1;fig1=fig;
  }
  else{
   p[fig1]->MKBorrar(x,y,3);band=0;p[fig1]->Show();
   (p[fig1]->*pp)(page);fig++;
  };
  break;
};


case 'l':{
    if (flag !=1){p[fig]=new Poly(0);fig2=fig; conta=0;flag=1;};
    break;
};
case 13:{
   if(flag== 1){
     p[fig2]->MKPoly(x,y,conta);
     (p[fig]->*tc)(x-5,y-5,x+5,y+5);
     conta ++;
   };
   break;
};

case 'd':{
 if (flag == 1){
     p[fig2]->Show();
     (p[fig2]->*pp)(page);
     flag =0;conta=0;fig++;
 };
 break;
 };

 case 's':{ gotoxy(1,1);sac=1; cout << "save option";
  if (getch()==27) break;else{
   cout << "Name to save...";
   cin >> mes;
   if((fp=fopen(mes,"wb"))==NULL){cout<<"cannot open file";break;};
   fwrite(&fig,sizeof(int),1,fp);
   for (i=0;i<fig;i++){
     fwrite(p[i],sizeof(p[i]),1,fp);
    };
   fclose(fp);
   break;
  };
 };
 case 'g':{gotoxy(1,1);sac=1;cout << "retrieve option";
  if (getch()==27) break;else{
   cout << "Name to retrieve..";
   cin >> mes;
   if((fp=fopen(mes,"rb"))==NULL){cout<< "cannot open file";break;};
   fread(&fig,sizeof(int),1,fp);

   for (i=0;i<fig;i++){
   fread(p[i],sizeof(p[i]),1,fp);};
   fclose(fp);
   getch();
```

```
      case 'n':{gotoxy(1,1);sac=1;cout << "printing options":
        if (getch()==27) break;else{
       globa=1; sac=1;
       break;
       };
      };
      case '+':{A++;break;};
      case '-': {if (A>1) A--;break;};
      case 27:{cout<< "are you sure<enter>";
      if (getch()==13) quit =1;break;};

      case 'r':{
        for (i=0 ;i<fig;i++){
        if ((p[i]->*pa)()==page){
           if((p[i]->*gc)(x,y)){
             p[i]=new Nothing;
            cout << "match";
           };
        };
        };
        cleardevice();
               for (i=0 ;i<fig;i++)
                  if ((p[i]->*pa)()== page)p[i]->Show();

        break;
      };


 };/// cierra el switch
};// sierra el if
  r.x.ax=3;
  int86(0x33,&r,&r);
  if(r.x.bx ==1){M.MousePos();x=M.Mgetx();y=M.Mgety();
    Kurs.Hide();
    Kurs.Pos(x,y);
    Kurs.Show();

  };
 if (x>=640){sac=1;x=1;page++;};
 if (x<=0 && page >0) {sac=1;x =639;page--;};
 if (y>=480){sac=1;y=1; page+=4;};
 if (y<=0 && page >3) {sac=1;y=479;page-=4;};
 if (sac==1 && fig!=0){
       cleardevice();
       gotoxy(0,0);
/      cout << page;

       for (i=0 ;i<fig;i++){
       if ((p[i]->*pa)()==page)p[i]->Show();
       };
       sac=0;
 };
 if (globa==1){globa=0;int86(0x5,&r,&r);};

; //cierrra el while
losegraph();

; // cierra la funcion
```

```
setcolor(3);
line (80,100,560,100);
line(80,100,80,300);
line (80,300,560,300);
line (560,300,560,100);
setcolor(1);
settextstyle (TRIPLEX_FONT,HORIZ_DIR,4);
outtextxy (320,60,"HELP");
setcolor(2);
settextstyle(SMALL_FONT,HORIZ_DIR,5);
outtextxy(320,110," c  = Circle                    t= triangle");
outtextxy(320,130," t = text; solo escriba el mensaje y luego enter");
outtextxy(320,150," l = line; marque los puntos con Enter y luego oprima l");
outtextxy(320,170," b = box                        p = point o tag");
outtextxy(320,190,"8= up arrow,6= right arrow,2= down arrow,4=left arrow");
outtextxy(320,210, " e = erase;marque los puntos con Enter y luego e");
outtextxy(320,230, " + = aumenta la speed , - = disminuye speed");
outtextxy(320,250, " s= save , g= get file");
outtextxy(320,270, " r= redraw and erase the current figure");
outtextxy(320,290, " n= prints  de plane");
setvisualpage(1);
getch();
setactivepage(0);
setcolor(4);
settextstyle(TRIPLEX_FONT,HORIZ_DIR,1);
setvisualpage(0);


};

A:\>
```