Universidad Nacional Autónoma de México

FACULTAD DE CIENCIAS

"UN PAQUETE DE GRAFICACION PARA EL MODELADO DE CURVAS Y PARA GRAFICAS FUNCIONES DE DOS VARIABLES"

TESIS PROFESIONAL

Que para obtener el Titulo de

M A T E M A T I C A

P r e s e n t a

MARIA CONCEPCION ANA LUISA SOLIS GONZALEZ COSIO

México, D. F. 1984

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

INDICE

INTRODUCCION.

CAPITULO

I.- INTRODUCCION A LA GRAFICACION.

1.1 INTRODUCCION.

1.2 TERMINOLOGIA COMUN EN GRAFICACION.

1.3 USOS REPRESENTATIVOS DE LA GRAFICACION POR COMPUTADORA.

1.4 CLASIFICACION DE LAS APLICACIONES.

1.5 ALGUNOS ASPECTOS DE LA GRAFICACION DESDE EL PUNTO DE VISTA USUARIO.

II.- DESARROLLO HISTORICO DE LA GRAFICACION POR COMPUTADORA.

2.1 INTRODUCCION.

2.2 DESARROLLO DE LA GRAFICACION DESDE EL PUNTO DE VISTA DE LA PROGRAMACION ("SOFTWARE").

2.2.1 TECNICAS.

i)Paquetes de Subrutinas.

ii)Paquetes de Aplicacion y "Turnkey Systems".

2.2.2 ALGORITMOS.

i)Los primeros algoritmos en Graficacion.

ii)Algoritmos orientados a aplicaciones.

iii)Algoritmos orientados a Raster Graphics.

2.2.3 EL ASPECTO DE LA ESTANDARIZACION.

III.- TECNICAS Y ELEMENTOS MATEMATICOS PARA EL MODELADO DE CURVAS.

3.1 INTRODUCCION.

3.2 TRANSFORMACIONES GEOMETRICAS EN DOS DIMENSIONES.
i)Translacion.

ii)Escalamiento.

iii)Rotacion.

3.3 COORDENADAS HOMOGENEAS Y SU REPRESENTACION MATRICIAL PARA LAS TRANSFORMACIONES EN 2D.

i) Tecnicas de Coordenadas Homogeneas.

ii)Composicion de Transformaciones en 2D.

3.4 TECNICAS BASICAS DE GRAFICACION.

i)Ventana y Puertos de Vision.

ii)Recorte.

3.5 METODOS DE INTERPOLACION PARA EL MODELADO DE CURVAS.

i)Representacion de Curvas.

ii)Curvas de Bezier.

iii)Curvas B-spline.

IV.- ELEMENTOS MATEMATICOS PARA LA REPRESENTACION DE OBJETOS EN TRES DIMENSIONES EN UN ESPACIO DE DOS DIMENSIONES.

4.1 INTRODUCCION.

4.2 TRANSFORMACIONES GEOMETRICAS EN TRES DIMENSIONES.

4.2.1 Representacion Matricial de Transformaciones en 3D.

4.2.2 Composicion de Transformaciones en 3D.

4.3 SOLUCION A LA REPRESENTACION DE OBJETOS EN TRES DIMENSIONES EN UN ESPACIO DE DOS DIMENSIONES.

4.3.1 Introduccion.

4.3.2 Clasificacion de las Proyecciones Geometricas Planas.

4.3.3 Bases Matematicas para las Proyecciones Geometricas Planas.

4.3.3.1 Introduccion.

4.3.3.2 Proyeccion Ortografica.

4.3.3.3 Proyeccion Oblicua.

4.3.3.4 Proyeccion en Perspectiva.

4.3.3.5 Proyeccion Estereopar.

V.- ELIMINACION DE LINEAS Y SUPERFICIES OCULTAS.

5.1 INTRODUCCION.

5.2 ESPACIO DEL OBJETO Y ESPACIO DE LA IMAGEN.

5.3 ALGORITMOS DE PROFUNDIDAD DE MEMORIA.

5.4 ANALISIS GEOMETRICO.

5.5 ALGORITMOS DE COHERENCIA DE LINEA DE BARRIDO.

5.6 ALGORITMO DE COHERENCIA DE AREA.

5.7 ALGORITMO DE PRIORIDAD.

5.8 LA ELECCION DE UN ALGORITMO.

5.9 ALGORITMO DE LINEAS OCULTAS USADO EN EL PAQUETE.

VI.- UN PAQUETE DE GRAFICACION PARA EL MODELADO DE CURVAS Y PARA LA REPRESENTACION DE FUNCIONES DE DOS VARIABLES.

6.1 INTRODUCCION.

6.2 DISEÑO DEL PAQUETE.

6.3 USO Y EJEMPLOS DEL PAQUETE DE GRAFICACION.

CONCLUSIONES

ANEXO A: DESCRIPCION DEL GRAFICADOR CALCOMP MODELO 1012 Y DEL SOFTWARE BASICO.

BIBLIOGRAFIA

INTRODUCCION.

El presente trabajo fue desarrollado con el objetivo de presentar un Paquete de Graficacion creado para el Graficador de Papel ("Plotter"), Calcomp Modelo 1012 , el cual se encuentra conectado a la computadora PDP 11/34 del Laboratorio de Computacion del Departamento de Matematicas de la Facultad de Ciencias.

La aplicacion que le fue dada , es la del desarrollo de rutinas para el modelado de curvas y para graficar superficies ($f: R^2 \rightarrow R$) , con distintos tipos de proyecciones y lineas ocultas.

Se pretende que el usuario del Paquete de Graficacion no necesite saber programar , solamente conocer el sistema de la PDP 11/34 a nivel de usuario (MCR , EDI o TECO ,etc.)

Para obtener esta facilidad se ha utilizado el manejo de archivos de comandos indirectos.

Ademas , hubo la necesidad de crear una biblioteca con las rutinas del Paquete con el Sistema de la PDP 11/34 , con el proposito de reducir el tamaño de la tarea que permite obtener la salida grafica en el graficador Calcomp.

alsun dispositivo de entrada para hacerlo interactivo.

5.-Computadoras personales , en blanco y negro o en color.Generalmente tienen muy baja resolucion en los que los puntos individuales que pueden ser facilmente vistos , son usados para formar caracteres, lineas y areas solidas.

6.-Microprocesadores orientados exclusivamente para juegos (video games processors).

La diferencia entre ellos esta en el tipo y calidad de imagen y el grado en el que el usuario puede controlar la imagen dinamicamente. Todos ellos comparten solo una propiedad: el dibujo de alsun objeto u objetos es creado y manipulado por un procesador digital.

Asi se puede decir que la Graficacion por Computadora es la creacion, almacenamiento y manipulacion de modelos de objetos y su trazado via la computadora.

C A P I T U L O I

CAPITULO I

INTRODUCCION A LA GRAFICACION

1.1 Introduccion.
1.2 Terminologia comun en graficacion.
1.3 Usos representativos de la Graficacion por Computadora.
1.4 Clasificacion de las Aplicaciones.
1.5 Algunos aspectos de la Graficacion desde el punto de vista usuario.

1.1.-Introduccion.

La Graficacion por Computadora se divide en Graficacion Pasiva y Graficacion Interactiva . Esta clasificacion esta relacionada con el tipo de dispositivo de graficacion en el cual se va a obtener la salida grafica.

La Graficacion "Pasiva" , se da en el sentido de que no hay dispositivos que permitan al usuario modificar el dibujo en tiempo real.

En cambio en la Graficacion Interactiva cuenta con una serie de dispositivos donde el usuario puede influir en el dibujo.

Se puede presentar varios ejemplos donde la graficacion por computadora quizas sea familiar.

1.-Impresoras y terminales de papel (Fig. 1.1 y Fig. 1.2).

2.-Graficadores de rodillos o planos(Drum o Flatbed Plotters),usados para la generacion de dibujos por lineas en dos dimensiones 2D y tres dimensiones 3D (diagramas de pastel , histogramas en 2D y 3D , diagramas de flujo ,diagramas para arquitectura, dibujo de circuitos, etc.)

3.-Films o "Video recorder" para producir dibujos con alta calidad y objetos.

4.-"Storage tube display" y terminales de pantalla,terminales con teclado y cursor manejado por

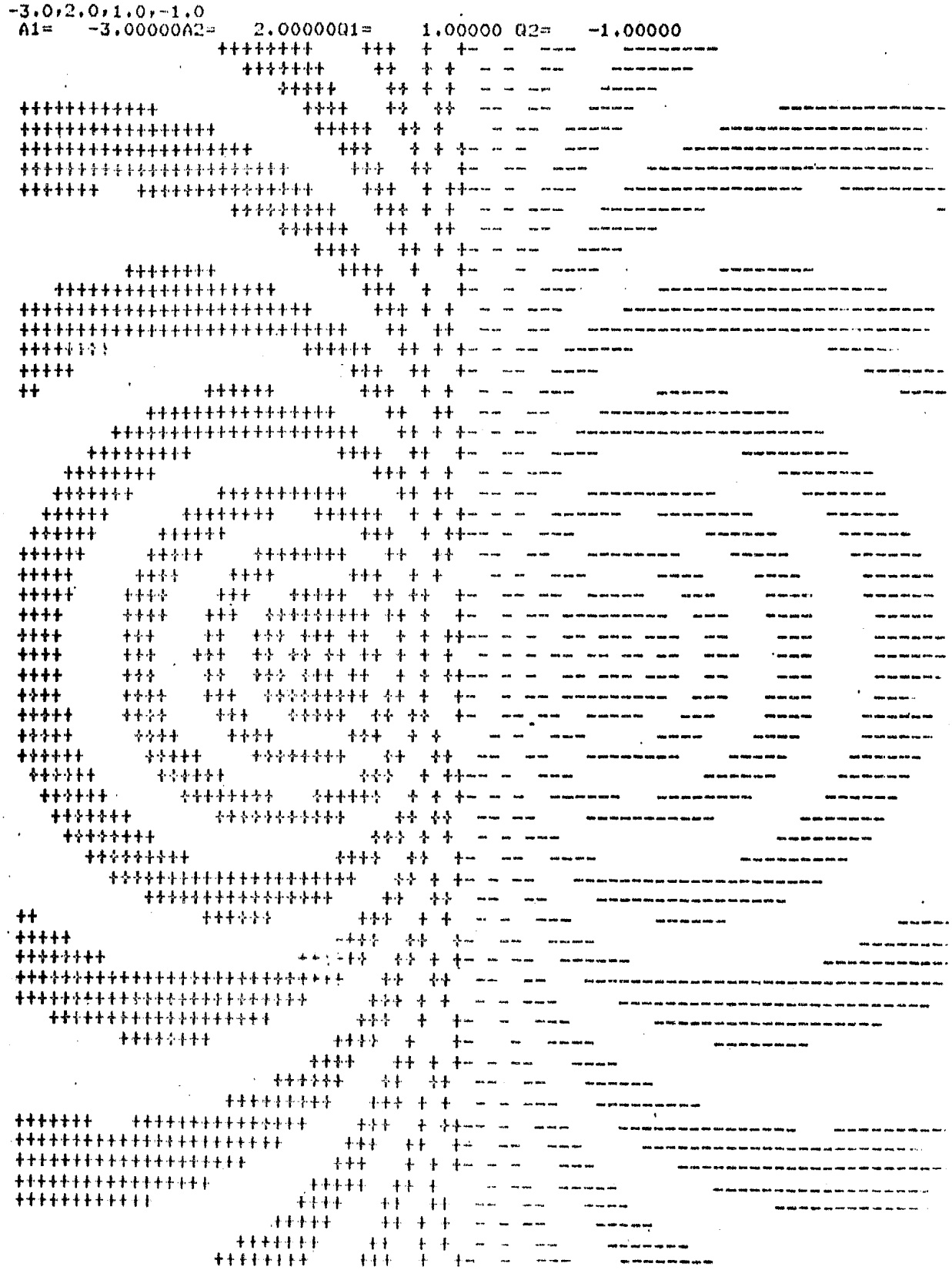


Fig 1.1

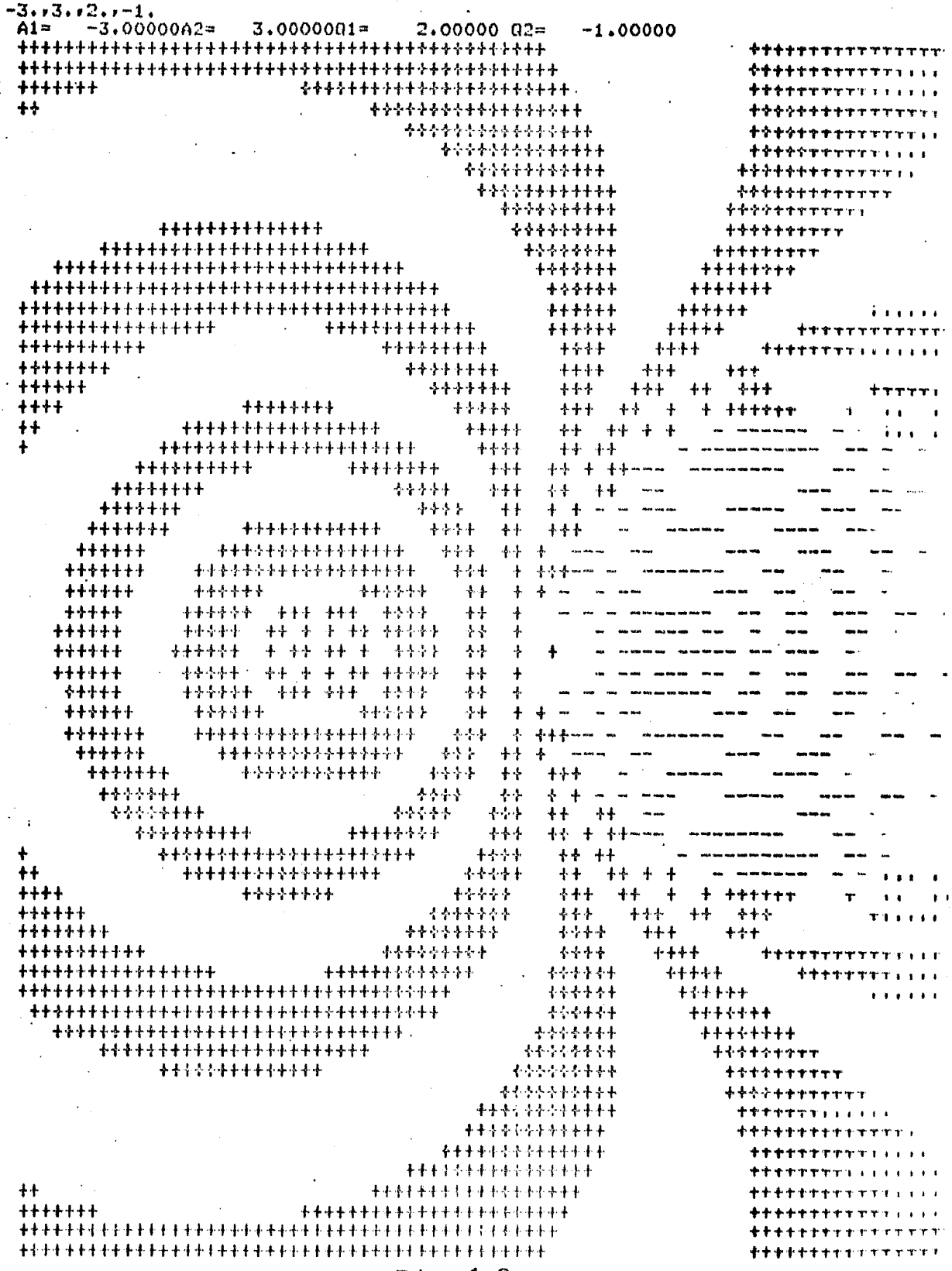


Fig. 1.2

1.2.- Terminologia comun en Graficacion.

Ya que la graficacion por computadora es relativamente una nueva tecnologia, es necesario clarificar la terminologia comun en este campo.

Un numero de terminos y definiciones son usados con soltura en esta area.

En particular , Computer Aided Design (CAD) , Interactive Graphics (IG), Computer Graphics (CG) y Computer Aided Manufacturing (CAM) son usados frecuentemente de manera incorrecta , ya que causan confusion su significado.

CAD seria el mas general ,CAD puede definirse como cualquier uso de la computadora para ayudar en el diseño de una parte individual, un subsistema, o un sistema total.

El uso de la computadora para diseñar no tiene que involucrar necesariamente a la graficacion.El diseño de procesos puede ser a un nivel del concepto de sistema o al nivel de diseño de alguna parte.

Esto puede tambien involucrar al CAM . Computer Aided Manufacturing (CAM) es el uso de la computadora para ayudar en la manufactura o produccion de una parte exclusiva del proceso del diseño.

Hay una relacion directa entre los resultados de una aplicacion del CAD y la parte necesaria de programacion usando lenguajes como APT (Automatic Programmed Tool) y UNIAPT (UNITED's APT) , un ejemplo seria la direccion de una maquina usando una minicomputadora.

Computer Graphics (CG) es el uso de una computadora para definir , almacenar, manejar y presentar una salida grafica o pictorica.

Esto es esencialmente una operacion pasiva.La computadora prepara y presenta la informacion almacenada a un observador en la forma de dibujos. El observador no tiene control directo sobre el dibujo presentado.

La aplicacion puede ser tan simple como la presentacion de la grafica de una funcion sencilla usando una impresora o una terminal de papel asi como hacer simulacion de algo complejo como el aterrizaje de una capsula espacial.

Interactive Graphics (IG) tambien usa la computadora para preparar y presentar material pictorico. En la graficacion interactiva el observador puede influir en el dibujo como este se vaya presentando , es decir, el observador interacciona con el dibujo en tiempo real.

Para ver la importancia de la graficacion en tiempo real , consideremos el problema de rotar un dibujo complejo tridimensional de 1000 lineas , que

estara rotando con una razon de 15 srados /sesundo ,suPondiendo que las 1000 lineas del dibujo se almacenaran en una matriz que contendra los Puntos finales de las lineas , se requiere de una matriz de 1000X4 , y la rotacion se lograra multiPlicando la matriz antes descrita Por una matriz de transformacion de 4X4 , dicha rotacion requiere de 16000 multiPlicaciones , 12000 sumas y 1000 divisiones .Si estas oPeraciones fueron hechas Por Prosramas , la tarea de rotar una srafica resultaria imPortante Por el tiemPo que consumiria.

La forma como se ha suelto es usando comPutadoras que tienen circui que realizan dichas oPeraciones y el tiemPo se reduc a microsesundos , ademas Permiten en un tiemPo suficientemente PequeMo el redibujar (refrescar) al menos 30 veces cada sesundo, evitando de esta manera el ParPadeo o temblor momentaneo de la srafica. De no hacerse en tiemPo real , resultaria imPosible llevar a cabo esta tarea.

1.3.- Usos representativos d la Graficacion por Computadora.

La graficacion es usada en chas areas tales como la industria, negocios, educacic. entretenimiento y lo mas reciente en el hogar.

Pero esta lista se va incrementado cada dia.

Una lista representativa seria:

- Ciencia y tecnologi (insenieria , matematicas, fisica, economia, etc`

- Administrativas y servicios.

- Educacion.

- Cartografia . Mapas geografic s ,mapas de relieves, mapas de exploracion, oceonografi s , de contornos ,para exploracion ,ya sea minera ,petrolifera y mapas de densidad de poblacion.

- Diseño ayudado por Computadora (Computer Aided Design (CAD)).La graficacion es usada para el diseño de componentes y partes mecanicas, electricas, electromecanicas y dispositivos electronicos, microcircuitos.Construccion de edificios ,plantas quimicas ,carroceria,barcos,aviones,etc. sistemas opticos, telefonicos,arquitectura,etc.

Simulacion y Animacion.No solo podemos estudiar objetos matematicos sino fisicos (hidraulicos,nucleares,fenomenos quimicos y biologicos).

Simular movimiento y crear objetos para cine,educacion y publicidad.

Otra aplicacion de animacion , es la simulacion de un vuelo.Los simuladores generan vistas no solo de un mundo fijo en el cual un vehiculo es movido , sino tambien de efectos especiales como nubes , niebla,contaminacion , luces y artefactos de varias formas y tamaños.

Control de Procesos.- Controlar ,corregir procesos , control de trafico aereo.

Arte y Comercio.- El arte por computadora se ha ido introduciendo en estos ultimos años y seguira desarrollandose ya que ofrece muchos atractivos a los artistas como otra herramienta mas para expresar su creatividad.

En lugar de utilizar pinceles y otro tipo de materiales,el dispositivo de gra icacion le ofrece una gama de colores muy extensa y la manera facil de creacion de objetos(sistemas diseñados para crear objetos graficos).

No solo es la creacion de los objetos sino el poder representar luces y sombras y otros efectos. Esto es utilizado ampliamente en publicidad para obtener efectos en los mensajes presentados.

1.4 .- Clasificacion de las Aplicaciones.

Los diversos usos de la graficacion por computadora difieren considerablemente en una variedad de formas.

Puede utilizarse muchos criterios para categorizar las aplicaciones.

1er. Criterio.- Tipo de objeto y dibujo ha ser producido.

Dentro de este criterio se incluyen el dibujo con lineas de objetos en dos dimensiones , el dibujo con lineas para objetos en tres dimensiones (algunas veces llamados "wire-frame pictures") , el dibujo de objetos en tres dimensiones con lineas ocultas removidas , imagenes en color en dos dimensiones , representacion en tres dimensiones de objetos solidos sombreados con superficies ocultas removidas.

Algunos de estos objetos son claramente abstractos ,algunos muy reales; similarmente los dibujos pueden ser puramente simbolicos(una grafica en 2-D) o muy realistas.

El mismo objeto puede , por supuesto , ser representado de muchas maneras.

2o.Criterio .-Tipo de interaccion y grado de control sobre el objeto o su imagen.

Dentro de este criterio se incluye tres niveles.El mas bajo es aquel en el que con una base de datos predefinida se produce una salida grafica con la ayuda de un programa de aplicacion.El segundo nivel es aquel en el cual hay una interaccion con el programa de aplicacion.Esta interaccion es a traves de parametros los cuales permiten al usuario modificar el dibujo.De esta manera se grafica, se altera los parametros y se vuelve a graficar.

Un tercer nivel,es el diseño interactivo , es en al cual el usuario empieza con la pantalla en "blanco" , define un objeto,tipicamente de componentes predefinidas y luego se alterara ya sea acercandonos , o alejandonos del objeto creado para conseguir la vista deseada.

3er. Criterio .- Papel que juega el dibujo.

Es la importancia que tiene la representacion visual en algun proceso.

Esta importancia va desde ser parte intermedia de algun proceso , hasta el ser el producto final.

Como ejemplos de productc finales tenemos la Cartografia , en Publicidad o histogramas , una animacion,planos arquitectonicos,etc.

En cambio en procesos como el diseño de automoviles , de diseño de objetos son parte minima en el proceso de fabricacion de estos objetos.

El diseño con la ayuda de la Computadora , ayuda a reducir el tiempo que llevaria en un diseño , ya que las modificaciones se hacen al instante , sin necesidad de repetir algun diseño hecho en papel.

1.5 Algunos aspectos de la Graficacion desde el punto de vista Usuario.

La graficacion por computadora como fue definida puede ser muy compleja y puede ser aplicada en diversas materias.

Esta abarca campos de estudio tan diversos como el diseño en electronica y diseños mecanicos usados en sistemas de graficacion , etc.; y los conceptos de listas y estructuras de arboles son usados para preparar y presentar los dibujos al observador en el sistema de graficacion.

Vesmos algunos aspectos de la graficacion desde el punto de vista usuario.

Primero, hay que familiarizarse con los dispositivos graficos a los cuales uno va a tener acceso.

Segundo, pensar que cualquier problema en graficacion por computadora se reduce en saber como especificar puntos en un espacio de dos dimensiones.

Ahora bien , como usuario , la graficacion puede ser dividida en las siguientes areas:

- Representacion de los dibujos a ser presentados.
- Preparacion del dibujo para la presentacion.
- Representacion previa de los dibujos preparados.
- Interaccion con el dibujo.

Aqui la palabra 'dibujo' , es usada en el sentido de una coleccion de lineas, puntos, textos , etc. a ser desplegados en el dispositivo de graficacion.

Un dibujo puede ser tan simple como una sola linea o curva , o representacion compleja de una superficie de alguna funcion o algun objeto.

a)Representacion de dibujos a ser presentados.

Fundamentalmente los dibujos presentados en graficacion pueden ser considerados como una coleccion de lineas , puntos y textos.

Una linea puede ser representada por las coordenadas de sus puntos extremos (x1,y1,z1) y (x2,y2,z2) , un punto puede ser una sola coordenada (x2,y2,z2) y los textos una coleccion de lineas y puntos.

b)Preparacion del dibujo para la presentacion.

Los dibujos consisten de puntos. Las coordenadas de estos puntos son generalmente almacenados en un archivo (arreglo) a ser usado para representar el dibujo.

Este archivo (arreglo) es llamado una base de datos. Para dibujos muy complejos se requiere de una base de datos muy compleja que envuelve estructuras de anillo, estructuras de arbol, etc., y la base de datos misma puede contener puntos ,sub-estructuras, y otro tipo de datos no graficos.

El diseño de estas bases de datos y los programas que hacen uso de estas es un tema de investigacion.

Quizas , en muchas aplicaciones de graficacion se quiera dibujos sencillos en donde el usuario pueda facilmente inventar una estructura de base de datos sencilla la cual pueda ser facilmente accesada.

Los puntos son bloques de construccion basicos de una base de datos.

Hay tres metodos basicos o instrucciones para manejar un punto como una entidad geometrica:mover el punto de la pantalla(la pluma , cursor o la cabeza del graficador a otro punto).Dibujar una linea a otro punto ; o dibujar un punto.

Fundamentalmente hay dos maneras para especificar la posicion de un punto : coordenadas absolutas o relativas.

En coordenadas relativas , la posicion del punto es definido dando el desplazamiento del punto respecto al punto previo.

El especificar la posicion de un punto en alguna de las coordenadas absolutas o relativas requiere de un numero . Este puede causar dificultades si una computadora tiene un tamaño de palabra muy limitada.Generalmente toda una palabra es usada para especificar una posicion.

El entero mas grande que se puede especificar es de 2**n-1 , donde n es el numero de bits de la palabra.Para una computadora de 16 bits frecuentemente usa pantallas para graficacion de 32767 posiciones.

Para muchas aplicaciones esto es aceptable.Quizas, las dificultades son encontrados cuando un numero muy grande (entero) es requerido. En primera debemos superar esta dificultad usando coordenadas relativas para especificar un numero ,por ejemplo 60000, es decir , usando coordenadas absolutas especificando la posicion del cursor (pluma,etc) en (30000,30000) y entonces especificar las coordenadas relativas a (30000,30000)de esta posicion a la posicion(60000,60000) que es el punto final deseado.

Pero esta no es la mejor manera para solucionarlo.

Otro metodo es usar coordenadas homogeneas. El

uso de coordenadas homogeneas introduce alguna complejidad adicional, se puede perder rapidez , y algo de resolucion. Quizas, estas desventajas son compensadas con la ventaja de representar numeros muy grandes en computadoras con el tamaño de palabra limitada.Por esta razon , y por otras que seran vistas adelante , la representacion en coordenadas homogeneas son generalmente usadas.

La idea de las coordenadas homogeneas es pasar de un espacio n-dimensional a un espacio con n+1 dimensiones ,es decir, los datos en tres dimensiones donde la posicion de un punto es dado por la terna (x,y,z) es representado por cuatro coordenadas (hx,hy,hz,h) , donde h es un numero arbitrario.

Si cada una de las posiciones de las coordenadas representadas en una computadora de 16- bits es menos de 32767 , entonces h debera ser igual a 1 y las coordenadas de las posiciones se representan directamente.

Si una de las coordenadas es mayor a 32767 , digamos x=60000 , entonces el poderio de las coordenadas homogeneas se vuelve aparente.

En este caso podemos hacer que h=1/2 , y las coordenadas del punto son entonces (30000,1/2y,1/2z,1/2) , todos los numeros son aceptados por una computadora de 16 bits.

Quizas , alguna resolucion se pierde ya que si x=60,000 y x=59,995 , ambas son representadas por las mismas coordenadas homogeneas.

En efecto la resolucion se pierde en todas las coordenadas aun si solo una de ellas se excede al maximo numero expresable de una computadora particular.

c)Representacion final de los dibujos preparados.

Con los comentarios acerca de la base de datos en mente es necesario notar que la base de datos usada para preparar el dibujo para la presentacion casi nunca es el mismo que la imagen que esta almacenada en memoria ("archivo del display") usado para representar el dibujo.La base de datos representa todo el dibujo mientras que el archivo del display representa alguna porcion , o alguna vista , o alguna escena del dibujo.

El archivo del display es creado por la transformacion de la base de datos.

El dibujo contenido en la base de datos puede ser redimensionado , rotado ,trasladado o parte de este , removido o la vista de un punto particular obtener una perspectiva antes de ser desplegado.

Muchas de estas operaciones pueden llevarse a cabo

usando simplemente transformaciones lineales las cuales son realizadas usando multiplicacion de matrices.

En muchas aplicaciones en graficacion frecuentemente solo algunas porciones de la base de datos son desplegadas . Este proceso de solo desplegar parte de la base de datos es llamada "windowing".

Es necesario convertir las coordenadas de los datos , los cuales pasan por el proceso del "windowing" ,en las coordenadas del pantalla de tal manera que el dibujo aparezca en alguna area especifica sobre el pantalla , llamada "viewport".

Un requerimiento adicional para muchos dibujos es la presentacion de caracteres alfanumericos.Hay en general dos metodos de generacion de caracteres -software y hardware-.Si los caracteres son generados en software usando lineas , estos son tratados de la misma manera como cualquier otro elemento del dibujo.La generacion de caracteres por hardware es menos flexible ya que no permite transformaciones como las antes mencionadas.

d)Interaccion con el Dibujo.

La interaccion con el dibujo requiere de algun tipo de dispositivo interactivo para la comunicacion con el programa mientras este esta ejecutandose.Numerosos dispositivos han sido usados para esta tarea.

Lo mas simple es ,por supuesto, el teclado alfanumerico.Dispositivos mas sofisticados incluyen plumas luminosas ,"joysticks",mouse" ,tabletas analogicas ,etc.

C A P I T U L O I I

CAPITULO II

DESARROLLO HISTORICO DE LA GRAFICACION.

2.1 INTRODUCCION.
2.2 DESARROLLO DE LA GRAFICACION DESDE EL PUNTO DE VISTA DE LA PROGRAMACION ("SOFTWARE").
  2.2.1 TECNICAS.
  i)Paquetes de Subrutinas.
  ii)Paquetes de Aplicacion y "Turnkey Systems".
  2.2.2 ALGORITMOS.
  i)Los algoritmos en Graficacion.
  ii)Algoritmos orientados a aplicaciones.
  iii)Algoritmos orientados a "Raster Graphics".
  2.2.3 El aspecto de la Estandarizacion.

2.1 Introduccion.

La Graficacion por Computadora ha evolucionado conforme va desarrollandose, de manera paralela, el software y el hardware para Graficacion.

Desde el punto de vista hardware , este se ve reflejado en el desarroll de pantallas para Graficacion y las facilidades r ra la interaccion con la salida grafica.

Y desde el punto de vista de softuare en el desarrollo de tecnicas y algoritmos.

Podria decirse que en 1950, aparecio el primer sistema de graficacion por computadora .La computadora Whirlwind del MIT ,tenia conectada una pantalla CRT que fue usada para generar dibujos muy sencillos.

A mediados de los 50's surje un sistema de control y comando de la defensa aerea SAGE.

SAGE convierte la informacion del radar en dibujos generados por computadora. SAGE introduce tambien la pluma luminosa , la cual permite al operador seleccionar informacion simplemente apuntado la localidad deseada sobre el CRT.

Esta es la primera vez que se uso la pluma luminosa en graficacion.

Al final de la decada (1957) , aparece la segunda generacion de computadoras en la cual se desarrollaron maquinas como TX0 y TX1 del MIT , donde la interaccion es posible y el interes en la graficacion empieza a incrementarse rapidamente.

Un ejemplo son las consolas de graficacion DEC modelo 30.

Al principio de los 60's ocurre un solo evento que hizo que la graficacion surgiera como un nuevo campo muy importante, fue la publicacion en 1962 de una tesis brillante de Ivan F.Sutherland de doctorado del MIT. Esta tesis, titulada "Sketchpad: A Man-Machine Graphical Communications System",mostro a muchos que la graficacion era un campo de investigacion viable , util y excitante.

2.2.-Desarrollo de la Graficacion desde el punto de vista del Software.

En sus inicios los interese en Graficacion estuvieron centrados en el desarrollo de tecnicas y algoritmos.Gradualmente , algunos principios basicos fueron reconocidos y fueron aceptados por la mayoria .En años recientes , se ha hecho mas enfasis en los principios y menos en las tecnicas.El desarrollo de algoritmos continua estimulado especialmente por el reciente interes en "Raster Graphics".

Los desarrollos mas importantes se dieron en las tres siguientes areas: tecnicas,algoritmos y principios.

Tecnicas.- Numerosas aplicaciones fueron desarrollandose con tecnicas basicas y se demostro la utilidad de la graficacion.

Actualmente hay tres tipos de productos: paquetes de subrutinas , paquetes de aplicacion y los "Turnkey Systems" (un sistema en donde rutinas de graficacion estan implementadas en "hardware").

Algoritmos .- El desarrollo de algoritmos ha sido paralelo al desarrollo de tecnicas y principios.

Las rutinas enfocadas a graficacion son aquellas rutinas del Sistema que hace posible la representacion grafica de los datos , es decir, subrutinas para generar curvas , llenar regiones , etc.

Las rutinas enfocadas a l aplicacion son aquellas rutinas del sistema que hace que sea posible el manejo de datos especificos a una aplicacion ,por ejemplo , medelado de superficies por curvas para partes de un automovil.

Principios.-El interes por la estandarizacion empezo en 1974 con el trabajo acerca de la independencia de dispositivo .Estos esfuerzos dieron resultados significativos a mediados de la decada y para 1982 aparecieron un gran numero de articulos de autores de muy diversos paises acerca de la estandarizacion en graficacion.

2.2.1 .- TECNICAS.

La contribucion mas famosa en graficacion ocurrio en 1962 , con la tesis doctoral de Ivan Sutherland:"Sketchpad".

Se mostro que se podia dibujar diagramas y ser manejados sobre la superficie de un pantalla CRT. El CRT junto con la pluma luminosa, permitia interaccionar con la computadora.La pluma luminosa permitio al usuario manejar el diagrama y ver los cambios casi inmediatamente.

El uso de elementos geometricos , es decir, puntos,lines y circulos ,y la obligacion de usar estos elementos para hacer todo diagrama ,fue el nucleo del "Sketchpad".

El trabajo inicial en 2D de Sutherland fue extendido a tres dimensiones por Johnson en 1963. Johnson partio al CRT en cuatro porciones , una mostrando un frente , un lado, la vista de arriba y la otra vista de perspectiva de algun objeto.

Un cambio hecho en una vista deberia ser hecho en las otras vistas.

Ahora algunos de los sistemas Turnkey usan tecnicas similares.

Johnson tambien mostro el dibujo de curvas y algunas aplicaciones [JOHN].

En 1962 , los investigadores de los laboratorios de Investigacion de General Motors dirijieron sus recursos al diseño de automoviles (Computer Aided Automobile Design).

Asi , el diseño para automoviles de la GM (GM's Design Augmented by Computer) tambien llamado Proyecto DAC-I , desarrollo tecnicas de graficacion para emular y mejoro los metodos tradicionales para el diseño de automoviles.

El Proyecto DAC-I , fue importante porque se mostro la utilidad en areas de diseño y manufactura.

Otras investigaciones hechas al principio de la graficacion fueron las de Chasen y sus colegas en la Compañia Lockheed - Georgia. Sus dos principales aplicaciones fueron la produccion de cintas para trabajar en maquinas herramientas de control numerico y la solucion de problemas de analisis estructural.

Tecnicas de Graficacion Interactiva fueron usadas por diseñadores para definir formas de partes a ser producidas.

Un cuarto ejemplo final de las primeras aplicaciones por computadora es el trabajo de Fetter y sus colaboradores en el comercial del Boeing de una compañia aerea el cual uso graficadores de papel para producir los cuadros para la animacion.

Estos cuadros fueron dibujados en fuera de linea y se realizaron manualmente (el color fue añadido por ejemplo) antes de ser filmados. Este fue una contribucion muy importante para la graficacion pasiva con tecnicas convencionales de animacion.

Se siguio con los esfuerzos de pioneros , investigadores y muchos otras firmas de industriales ,

de institutos y de Universidades y empezaron muchos proyectos sobre graficacion por computadora .

i)Paquetes de Subrutinas.

Mientras que las primeras aplicaciones convencieron a mucha gente de la utilidad de la graficacion , no se produjo software de graficacion que fuera portatil y de muchos propositos.

La situacion fue pobre , el programador tenia que desarrollar tanto el software de graficacion como las rutinas de graficacion de la aplicacion , el desarrollo para las aplicaciones fue mas lento y mas caro y el programador para las aplicaciones fue forzado a ser un experto en graficacion.

Entonces, se necesito de que existiera y aun ahora todavia se necesita, del desarrollo de un buenas rutinas para ser usado por un programador de aplicaciones.

Tales rutinas han sido desarrolladas por muchas fuentes , por vendedores de hardware para dar soporte a sus productos , por usuarios para dar apoyo a sus aplicaciones ,por investigadores de graficacion para alcanzar sus objetivos.

Un buen paquete de graficacion de proposito general puede ser usado para desarrollar una variedad de aplicaciones.

El paquete proporciona una interface entre el programador y el hardware para graficacion.

Para usar el paquete de proposito especial se requiere tambien de una interface,pero a un nivel mayor.Generalmente , el programador trabajara en un area particular y el paquete tendra esta orientacion y el programador no tendra que saber de graficacion.

El extender algun lenguaje de programacion ha sido muy atractivo y una alternativa por años .Las extensiones pueden proporcionar una interface mas satisfactoria entre el programador y la capacidad de graficacion que el mecanismo tipico de subrutinas.

El desarrollo de lenguajes de graficacion de proposito general seria muy poderoso como todo lenguaje de programacion.

Los paquetes de subrutinas de proposito general han recibido mucha atencion por años y probablemente viene a la mente mas rapidamente cuando uno habla de software de graficacion.

El primer paquete de este tipo fue introducido por Calcomp al principio de los 60's para sus graficadores de papel.

Paquetes de graficacion y otro software de proposito especial fue construido encima de rutinas basicas de Calcomp.

Sistemas que son compatibles con Calcomp aun existen en muchas instalaciones, frecuentemente soporta pantallas que son mucho mas sofisticados que un

graficador de papel para el cual originalmente se creo el software.

GINO (Graphical Input and Output), fue desarrollado por el grupo Computer Aided Design en la Universidad de Cambridge.Es un ejemplo de uno de los primeros paquetes con otra clase de subrutinas.GINO fue desarrollado para ejecutarse en una PDP-7 con una pantalla DEC 340 [GINO].

El desarrollo de GINO descubrio el valor de un paquete que es independiente del dispositivo de graficacion y de la maquina ,tambien proporciono facilidades para graficar en 2D y 3D.

En los siguientes años , muchos otros paquetes de subrutinas se han desarrollado y han alcanzado un grado de popularidad , como el GKS.

Otros , como el GPGS y el IG fueron concebidos y desarrollados como paquetes de subrutinas de proposito general que son independientes de la maquina [GPGS].

Mientras que estos paquetes existen en un numero de maquinas y soportan varios dispositivos , ninguno ha alcanzado la popularidad que ha tenido las rutinas de Calcomp.

Mas recientemente , los paquetes de proposito general han sido desarrollados con las bases propuestas por el grupo SIGGRAPH Graphics Standars Planning Commitee (GSPC).

Ejemplos de estos paquetes estan la DI-3000 de Precision Visuals ,Inc.; VGM de Bell-Northern Research y Template de Megatek.

ii)Paquetes de Aplicacion y Turnkey Systems.

Un paquete de aplicacion es frecuentemente un programa interactivo .Aqui el usuario necesita solo conocer como se ejecutan e interaccionar con el programa.

A este nivel el factor humano se vuelve muy importante .Si la interface entre el usuario y el sistema no fuera bien diseñada e implementada , el usuario no podria usar el sistema.

La graficacion es una area en la cual se cuenta con muchos paquetes de arlicacion .Tektronix , por ejemplo, ofrece el IPG , un programa para usuarios sin experiencia en computacion para obtener graficas de datos y diagramas de pastel o histogramas.

Como el uso de la graficacion en los negocios ha crecido , el numero de tales paquetes se ha incrementado.

Los programas de Graficacion estan disponibles para aplicaciones en el area electrica , mecanica e ingenieria civil.

Los Turnkey Systems representan la capa mas alta del software de graficacion. Hace algunos años , un Turnkey System fue frecuentemente menos que una variedad de hardware.Ahora ,los componentes del hardware de un sistema Turnkey es integrado , es empaquetado dependiendo de la demanda de la aplicacion y las necesidades del operador.

Estos sistemas son aplicables en insenieria electrica , en insenieria mecanica (3D).

Estas areas proporcionara un mayor mercado de los sistemas Turnkey en los 80's.

2.2.2 Algoritmos.

Los algoritmos de graficacion se ha desarrollado en paralelo con el "hardware" de graficacion, "software" y aplicaciones.

Durante años , varios problemas han estimulado a los programadores en graficacion para encontrar nuevos o mejores algoritmos.

Por ejemplo, al principio se presento el problema de trazar segmentos de recta sobre pantallas en los cuales se disponia de solo puntos.

Ahora , los programadores se enfrentan a las limitaciones de los dispositivos de rastreo y se buscan formas de sobreponerlas.

La necesidad de los programadores para hacer aplicaciones ha motivado tambien para desarrollar algoritmos , por ejemplo la necesidad de diseñar automoviles , se necesito de representar superficies suaves y poder interaccionar con ellas.

i) Los primeros algoritmos en Graficacion.

Los algoritmos para generar un conjunto de puntos descritos que pudieran representar un segmento de recta fue entre los primeros algoritmos desarrollados.

Un algoritmo clasico para generar una sucesion de puntos fue desarrollado por Bresenham.

Este algoritmo no requiere de multiplicaciones o divisiones , lo cual lo convierte en muy economico.

Se ha desarrollado algoritmos muy importantes desde que surgio el "Sketchpad".De nuevo , el objetivo ha sido incrementar el realismo sin incrementar el costo.

Los algoritmos que mas se acercan a esta realidad son llamados algoritmos de lineas ocultas.Estos algoritmos eliminan de la descripcion del objeto los segmentos de linea que deben ocultarse del observador.

En 1963 , justamente un año despues del desarrollo del "Sketchpad" , Roberts fue el primero que desarrollo el problema de lineas ocultas.

Tambien se ha desarrollado algoritmos para el manejo eficiente de la informacion grafica sobre pantallas.

Hay dos tipos de algoritmos mas importante , algoritmos para recorte (clipping) y los algoritmos basados en el concepto de coordenadas homogeneas.

El uso de coordenadas homogeneas en Graficacion fue introducido por Roberts en 1965.

El concepto no ha sido ampliamente entendido y apreciado , esto proporciona una base conveniente para describir y realizar las transformaciones de los objetos en la escena.

Translaciones en tres dimensiones , y en dos dimensiones, escalamientos, rotaciones y proyecciones pueden manejarse simultaneamente en una sola matriz de 4X4 formada por la concatenacion de operaciones individuales.

ii)Algoritmos orientados a aplicaciones.

Quizas lo mas importante de estos algoritmos son que se crearan para la descripcion y manipulacion de curvas y superficies.

Numerosas aplicaciones en areas como el diseño de automoviles y el diseño grafico , son utilizadas.

Son pocos los nombres que pueden mencionarse que han contribuido en esta area.

Coons empezo a reportar sus desarrollos en 1964 , cuando creo un sistema para modelado de superficies.

Desafortunadamente , su tecnica desarrollada para especificar a lo que el llama "Coons patches" , donde superficies complejas son construidas uniendo superficies(patches) , fue dificil y no intuitiva.

Al principio de 1970 , Bezier desarrollo un sistema alternativo ampliamente aceptado actualmente,con el cual puede describirse curvas y superficies.

Los algoritmos de Bezier tiene varias caracteristicas los cuales contribuyeron a su popularidad , es facil especificar las curvas y superficies y tienen un buen comportamiento.

Al mismo tiempo , Riesenfield desarrollo la tecnica de B-splines , la cual tambien describe curvas y superficies.

Se ha dado una atencion considerable a los algoritmos para la descripcion y manejo de modelos geometricos tridimensionales.

La idea es producir modelos de volumenes mas que

de superficies y realizar operaciones tales como la union e interseccion de estos modelos.

Forrest ha revisado el modelado de curvas y superficies y en el modelado geometrico en 3D.

iii) Algoritmos orientados al "Raster Graphics."

La creacion de imagenes por computadora usando equipo de tipo "Raster Graphics" o Graficacion por Rastreo es el equivalente electronico de la tecnica de puntillismo usado por Seurat y otros pintores.Estos pintores para crear sus obras aplicaban la pintura colocando puntos sobre el lienzo.

Todas las pantallas de tipo raster contienen un arreglo de "puntos" o elementos de dibujo (picture elements) llamados pixels.

Una imagen o dibujo puede ser creada sobre la pantala controlando el valor de cada uno de los pixels.

En el caso de una pantalla en blanco y negro de tipo raster , un pixel es blanco o negro.

En una pantalla en color ,cada pixel puede tomar un ranso de colores.

En una pantalla de tipo vectorial ("vector display") , el electron se mueve en linea recta de un punto a otro, mientras que en una pantalla de graficacion de tipo raster trabaja como un aparato de television.

El electron pasa por toda la matriz de pixels , a esta operacion se le llama rastrear.

Muchos algoritmos desarrollados han sido asociados con "raster graphics".

Se han desarrollado los algoritmos en cuatro areas: conversion a rastreo, superficies ocultas , sombreado y manejo del color.

Conversion a rastreo.-Esta conversion consiste en cambiar la informacion vectorial en informacion por rastreo. Esto es hecho frecuentemente por hardware.

La popularidad de las pantallas de tipo raster ha revivido el interes en los algoritmos de generacion por puntos.

Tales algoritmos aparecieron en los 60's. Se desarrollaron articulos sobre generacion de rectas , circulos y poligonos.

El raster graphics es util para desplegar superficies.

Knowlton ha desarrollado un algoritmo , con operaciones sobre esferas y cilindros ,con los cuales ha encontrado

un amplio uso en aplicaciones tales como modelado de moleculas.

Con "raster displays" , es facil de intesificar y colorear areas , las superficies no necesitan representarse solo con lineas , sino que pueden ser sombreadas .En muchas aplicaciones en 2D , en el dibujo de diagramas y otro tipo de dibujos, polisonos son llenados con un solo color o intesidad , el cual es escosido por el usuario para el efecto visual deseado.

En aplicaciones en 3D , se escose el color e intensidad apropiada el cual es determinada por el algoritmo.

2.2.3.-El aspecto de la Estandarizacion.

El periodo de 1963 al 1974 fue caracterizado por el desarrollo aislado de proyectos y trabajos.

Las sociedades profesionales de graficacion y video fueron formadas durante este periodo.

La sociedad "Society for Information Display", fue fundada en 1963.

El comite de interes especial de ACM para Graficacion fue formado mas tarde en 1966 y se convirtio en un grupo de interes especial (SIGGRAPH) en 1967.

Habia dos grupos cuyas aplicaciones tenian distintas necesidades de equipo.Para necesidades sencillas usaban graficadores de papel y el otro grupo con necesidades mas complejas hacia uso de procesadores de pantalla para graficacion interactiva.

Pero hubo un cambio en la tecnologia de pantallas que hizo que estos dos grupos se acercaran.Se introduce en 1966 el direct view storage tube (DVST) ,el cual satisfacia las necesidades de ambos grupos, y surge la idea de la estandarizacion para la graficacion por computadora en el sentido de independencia de dispositivos , estandarizacion que podria satisfacer las necesidades de ambos grupos.

Uno de los primeros usos fue que el usuario podria descubrir errores rapidamente sobre el DVST sin consumir tiempo en graficadores de papel o tener que esperar.Asi muchas corridas en graficadores de papel que consumieran horas en un DVST podia hacerse en segundos.

En 1968, el DVST fue ampliamente aceptado no solo por su bajo precio (de $14,000.00 bajo a $3,500.00 dolares) sino porque podia ser conectado simple y directamente a pequeñas computadoras y sistemas de tiempo compartido.

Y de esta manera fueron mas los usuarios que tuvieron acceso a dispositivos de graficacion.

Blinn en la Universidad de Michigan creo el paquete IG ,que no solo incluso propiedades de estructuras de datos sino que proporciono independencia de dispositivo.

Una consecuencia de esto , es una serie de articulos que aparecieron con la discusion acerca del diseño de programas de aplicacion. Por ejemplo , Newman y Sproull resumentaron que la estructura de datos para una aplicacion debe ser separada del sistema de graficacion.

A mediados de los 70's , la proliferacion de las aplicaciones en graficacion empezaron a interesar a la comunidad del area de graficacion.

Muchas de estas aplicaciones desaparecieron tan pronto como el hardware sobre el cual se basaron se

volvio obsoleto o fallo.

Tambien poca atencion fue puesta en la transportabilidad , en la independencia de dispositivo e independencia de computadora.

En 1974 , un trabajo sobre independencia de dispositivo fue llevado al National Bureau of Standards en Gaithersburg ,Maryland.

Trabajadores en el campo de la graficacion discutieron de la necesidad de una programacion estandarizada para reducir la dependencia de dispositivo. Pocos resultados surgieron del esfuerzo inicial.

Estas discusiones fuero llevadas a cabo por Robert M. Dunn (Presidente del grupo de trabajo de SIGGRAPH) y se creo el Graph s Standards Planning Committe (GSPC).

Bajo la supervision del IF un grupo de trabajo se reunio de Mayo de 1976 en Se lac ,Francia.

Cerca de 30 expertos se re ieron bajo el tema de "Metodologia en la Graficacion p r Computadora".

Estos dieron algunas ideas:

a)Tener claro el conjunto esencial y fundamental de los conceptos en graficacion.

b)Identificar que areas se necesitan de mas investigacion.

c)Donde sea posible ,dar una serie aceptable de recomendaciones.

d)Diseminar el trabajo a la comunidad para producir un documento.

Se tomo con entusiasmo las ideas de Seillac y se regreso con ellas en el Siggraph 76.

Al mismo tiempo , el subcomite empezo a trabajar primero en el Reporte del GSPC el cual fue presentado en el Siggraph 77.

El GSPC produjo un segundo reporte en 1979.

En la Republica Federal de Alemania surge un esfuerzo similar.

Este grupo es conocido como el Graphics Kernel System (GKS) y auspiciado por la organizacion de estandarizacion Alemana (DIN).

La organizacion Internacional de Estandarizacion convino en crear un grupo de trabajo para considerar la estandarizacion para el software de graficacion.

Este grupo se le asigno la coordinacion del GKS y del GSPC para minimizar las diferencias y exponer el material en una audiencia internacional.

El trabajo del GSPC hasta ahora ha finalizado

(1981) y el comite ha desaparecido.

El Instituto Nacional Americano de Standarizacion ha formado un grupo llamado Technical Commitee on Computer Graphics Lanstandes,el cual se ha hecho cargo de producir la estandarizacion fundamentalmente de funciones para la graficacion con dispositivo independiente.

Es prematuro decir cuales serian estas recomendaciones ,pero puede decirse que los documentos del GSPC han tenido una amplia aceptacion en el desarrollo de rutinas para graficacion.

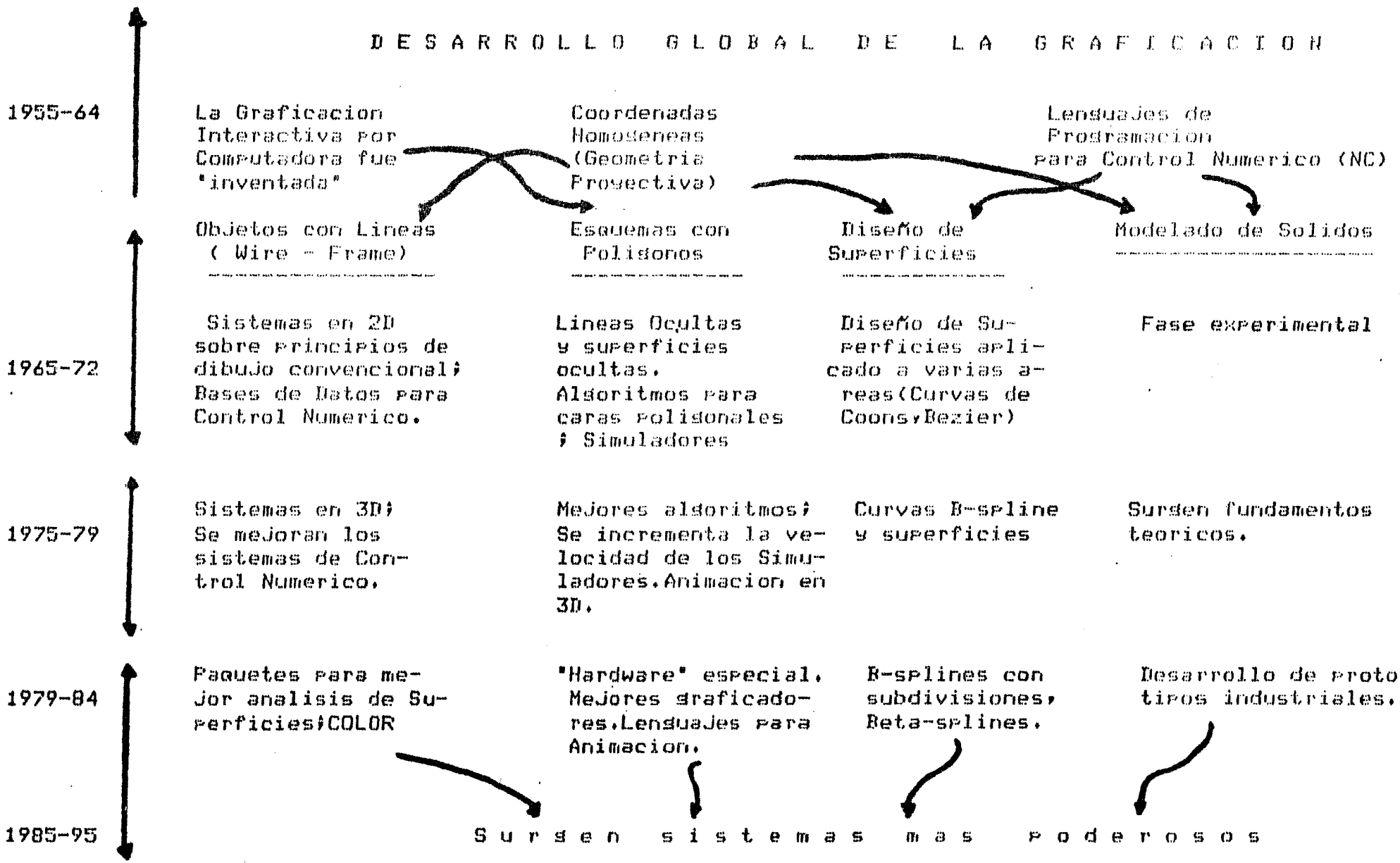

DESARROLLO GLOBAL DE LA GRAFICACION
1955-64
La Graficacion Interactiva por Computadora fue "inventada"
Coordenadas Homogeneas (Geometria Proyectiva)
Lenguajes de Programacion para Control Numerico (NC)
Objetos con Lineas (Wire - Frame)
Esquemas con Poligonos
Diseño de Superficies
Modelado de Solidos
1965-72
Sistemas en 2D sobre principios de dibujo convencional; Bases de Datos para Control Numerico.
Lineas Ocultas y superficies ocultas. Algoritmos para caras poligonales ; Simuladores
Diseño de Superficies aplicado a varias areas(Curvas de Coons,Bezier)
Fase experimental
1975-79
Sistemas en 3D; Se mejoran los sistemas de Control Numerico.
Mejores algoritmos; Se incrementa la velocidad de los Simuladores.Animacion en 3D.
Curvas B-spline y superficies
Surgen fundamentos teoricos.
1979-84
Paquetes para mejor analisis de Superficies;COLOR
"Hardware" especial. Mejores graficadores.Lenguajes para Animacion.
B-splines con subdivisiones, Beta-splines.
Desarrollo de prototipos industriales.
1985-95
Surgen sistemas mas poderosos

TABLA 1.1

C A P I T U L O I I I

CAPITULO III

TECNICAS Y ELEMENTOS MATEMATICOS PARA EL MODELADO DE CURVAS.

3.1 INTRODUCCION.
3.2 TRANSFORMACIONES GEOMETRICAS EN DOS DIMENSIONES.
  i)Translacion.
  ii)Escalamiento.
  iii)Rotacion.
3.3 COORDENADAS HOMOGENEAS Y SU REPRESENTACION MATRICIAL PARA LAS TRANSFORMACIONES EN 2D.
  i)Tecnicas de Coordenadas Homogeneas.
  Composicion de Transformaciones en 2D.
3.4 TECNICAS BASICAS DE GRAFICACION.
  I)VENTANA Y PUERTOS DE VISION.
  RECORTE.
3.5 METODOS DE INTERPOLACION PARA EL MODELADO DE CURVAS.
  i)Representacion de Curvas.
  ii)Curvas de Bezier.
  iii)Curvas B-spline.

3.1 .- Introduccion.

Existen metodos basicos que son parte importante de cualquier software de graficacion , es decir, estos algoritmos son independientes de la orientacion que tenga el programa de aplicacion, ya sea que este orientado a manejo de datos , Cartografia , Arquitectura, etc..

Entre estos metodos basicos se encuentran las transformaciones geometricas en dos dimensiones , el metodo de recorte (clipping) y el uso de la ventana (windowing) y los puertos de vision (viewport).

Para tres dimensiones , se tiene las transformaciones en tres dimensiones y las proyecciones.

3.2.-TRANSFORMACIONES GEOMETRICAS EN DOS DIMENSIONES (2D).

Las transformaciones geometricas son una parte esencial en la Graficacion.

La translacion , cambio de escala y rotacion son transformaciones que son el corazon de muchas aplicaciones en Graficacion.

Las transformaciones son aplicadas ya sea dentro del paquete de graficacion o bien directamente del prograna de aplicacion.

Una aplicacion directa de estas transformaciones , es el mapeo de las coordenadas del objeto a las coordenadas de la pantalla (translacion y cambio de escala).

i)TRANSLACION.

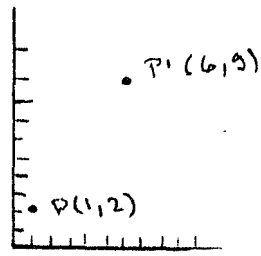
Para obtener la transalacion en 2D , lo que se hace , es simplemente sumar ciertas cantidades para obtener nuevas posiciones.

Para cada punto $P(x,y)$, se requiere obtener un nuevo punto $P'(x,y)$,
el cual se puede escribir como:

$$x' = x + Dx$$

$$y' = y + Dy$$

Para cada punto $P(x,y)$ es movido Dx unidades con respecto a x y Dy
unidades con respecto a y .



3.1 Translacion de un Punto

Definiendolos como vectores se tiene:

$$P = [x \; y] \quad P' = [x' \; y'] \quad T = [Dx \; Dy]$$

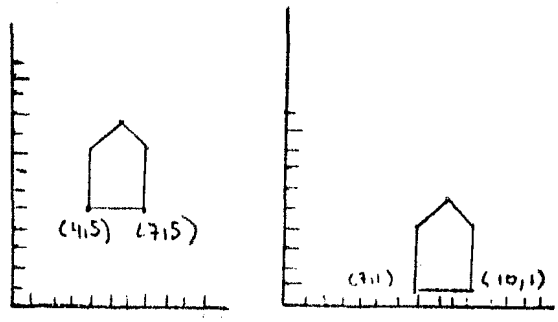
en la forma vectorial se tiene:

$$[x' \; y'] = [x \; y] + [Dx \; Dy]$$

aun mas conciso:

$$P' = P + T$$

Una transformacion puede ser aplicada a cada punto que compone el objeto . Afortunadamente solo es necesario aplicar la transformacion a los puntos extremos y dibujar una nueva linea.



3.2 Translacion de un Objeto

ii) CAMBIO DE ESCALA.

Otra transformacion es el cambio de escala .Los puntos son transformados por Sx y Sy a lo larso de x y y.

$$X' = x \cdot Sx \qquad y' = y \cdot Sy$$

Definiendo a S como

$$S = \begin{bmatrix} Sx & 0 \\ 0 & Sy \end{bmatrix}$$

se puede escribir en forma matricial :

$$[x' \quad y'] = [x \quad y] \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

ó

$$P' = P \cdot S$$

Hay que notar que el cambio de escala es respecto al origen.

Si Sx=1/2 y Sy=1/4 , el objeto es mas pequeño y cercano al origen.

Para un cambio de escala uniforme se debe cumplir que Sx=Sy para que no se altere los proporciones.

iii) ROTACION.

La rotacion es aplicada a cada punto de una figura a traves de un angulo THETA con respecto al origen.La rotacion es definida matematicamente como:

$$x' = x \cdot \cos\theta - y \cdot \operatorname{sen}\theta$$

$$y' = x \cdot \operatorname{sen}\theta + y \cdot \cos\theta$$

En la forma matricial , se tiene:

$$[x' \quad y'] = [x \quad y] \begin{bmatrix} \cos\theta & \operatorname{sen}\theta \\ -\operatorname{sen}\theta & \cos\theta \end{bmatrix}$$

ó

$$P' = P \cdot R$$

3.3.-COORDENADAS HOMOGENEAS Y SU REPRESENTACION MATRICIAL PARA LAS TRANSFORMACIONES en 2D.

La representacion matricial de las transformaciones vistas son :

$$P' = P+T$$

$$P' = P \cdot S$$

$$P' = P \cdot R$$

Hay que observar que se tiene dos multiplicaciones y una suma.

Si las transformaciones pudiesen ser expresadas de manera homogenea, podrian ser combinadas facilmente.

Si podemos expresar los puntos en terminos de coordenadas homogeneas, las tres transformaciones pueden ser tratadas como multiplicaciones

Las coordenadas homogeneas fueron desarrolladas en geometria y luego fueron aplicadas en graficacion.

Numerosos paquetes de graficacion y algunos procesadores de pantalla trabajan con coordenadas homogeneas.

i) TECNICAS DE COORDENADAS HOMOGENEAS.

La representacion de puntos, lineas,planos y superficies en coordenadas homogeneas son de gran utilidad para describir y manejar objetos graficos.

El termino homogeneo es aplicado a la representacion para una clase de objetos que que noi contienen constantes de manera explicita.

For ejemplo la ecuacion de una recta en dos dimensiones es y=mx+b.

La ecuacion homogenea de la misma recta es mwx=yw+bw=0 , una ecuacion con tres variables .Esta ecuacion tiene su representacion de la linea en un espacio de tres dimensiones.

La representacion homogenea fue desarrollada como una herramienta geometrica para probar teoremas de geometria proyectiva.

Un problema en un espacio-n tiene un correspondiente en un espacio n+1. Los resultados en un espacio-(n+1) son obtenidos mas facilmente que los planteados en el espacio-n.

Asi la representacion homogenea de un objeto en un espacio-n , es un objeto en el espacio n+1.

Un vector en el espacio homogeneo-(n+1) puede ser visto como un vector en el espacio-n al cual se le ha sumado una coordenada al vector y al que se le llama factor de escala.

La representacion del punto en 2D (x,y) , es en general (wx,wy,w) donde w es un escalar w≠0.

El vector esta en 3D y puede ser manejado como tal.

Los objetos en 3D son tratados de manera analoga y la representacion homogenea de (x,y,z) es (wx,wy,wz,w) para w≠0.

Dada la representacion en coordenadas homogeneasde un punto P(x,y,w) , se puede encontrar en coordenadas cartesianas un punto r(x/w,y/w).

Utilizando esta tecnica , puede obtenerse la representacion matricial para la la translacion , el escalamiento y rotacion :

$$T(Dx, Dy) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ Dx & Dy & 1 \end{bmatrix}$$

$$S(Sx, Sy) = \begin{bmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R(\theta) = \begin{bmatrix} \cos\theta & \operatorname{sen}\theta & 0 \\ -\operatorname{sen}\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$P' = P \cdot T(Dx_1, Dy_1)$$

$$P' = P \cdot S(Sx_1, Sy_1)$$

$$P' = P \cdot R(\theta)$$

De esta manera las tres representaciones son multiplicaciones de matrices.

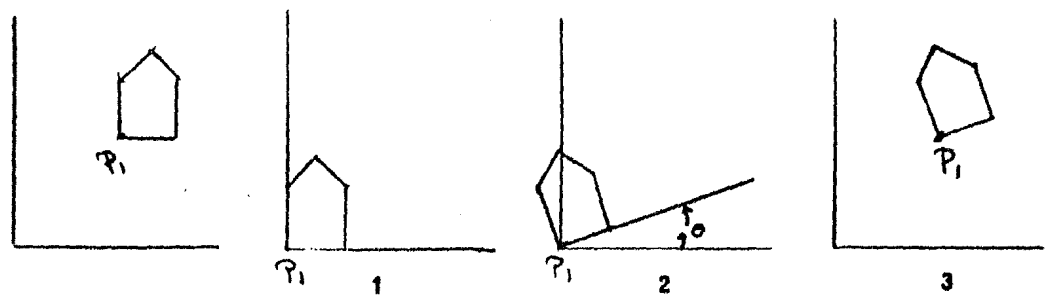
ii)COMPOSICION DE TRANSFORMACIONES EN 2D.

El proposito basico de las composiciones es la de ser lo mas eficiente , es decir, el aplicar una sola transformacion a un punto que aplicar una serie de transformaciones , una despues de otra.

Por ejemplo, si se quiere rotar un objeto con respecto a un punto arbitrario P_1 , que pasos se seguirian.

Ya que solo conocemos como rotar con respecto al origen, hay que descomponer este problema en tres mas sencillos:

1.- Transladar el punto P al origen .
2.- Rotar.
3.- Trasladar del origen al punto P.



Las matricas serian:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -x_1 & -y_1 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & \text{sen}\,\theta & 0 \\ -\text{sen}\,\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x_1 & y_1 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta & \text{sen}\,\theta & 0 \\ -\text{sen}\,\theta & \cos\theta & 0 \\ x_1(1-\cos\theta)+y_1\,\text{sen}\,\theta & y_1(1-\cos\theta)-x_1\,\text{sen}\,\theta & 1 \end{bmatrix}$$

$$RR(x_1, y_1) = (x_1, y_1) \cdot T(-x_1, y_1) \cdot R(\theta) \cdot T(x_1, y_1)$$

3.4 TECNICAS BASICAS DE GRAFICACION.

i)VENTANA Y PUERTO DE VISION.

La transformacion de ventana se llama asi porque hay mucha semejanza con las ventanas reales.

Esta ventana en el espacio del objeto va a contener la informacion que queremos desplegar .

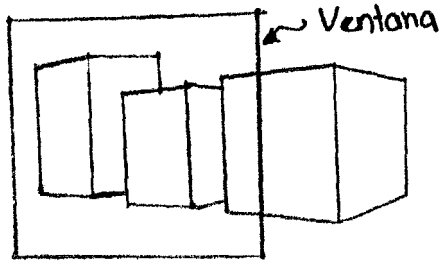
La ventaja de utilizar esta transformacion es que se especifica directamente el area de interes del dibujo definido.

Junto con la ventana ,podemos definir un puerto de vision (view-port), un rectangulo sobre el dispositivo de graficacion donde se va a desplegar el contenido de la ventana.

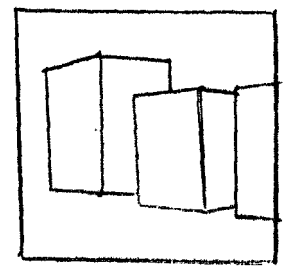
De esta manera, se usa la ventana para definir que se quiere desplegar y se define el puerto de vision para especificar en donde se va a desplegar en el dispositivo de graficacion.

Moviendo la ventana y seleccionando el tamaño de ventanas grandes y pequeñas , se puede crear efectos de alejarse y acercarse al algun objeto.

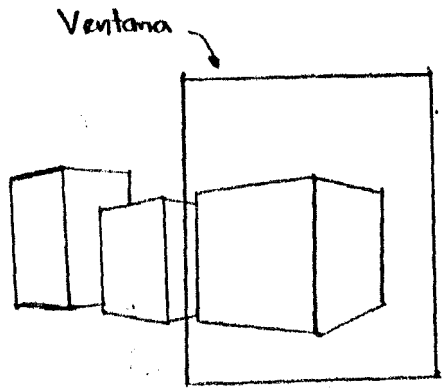
Ejemplo:

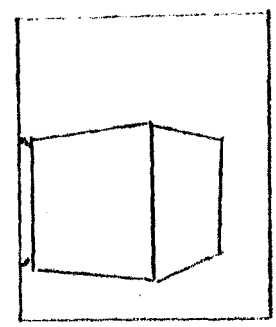


Sistema de Coordenadas del Objeto

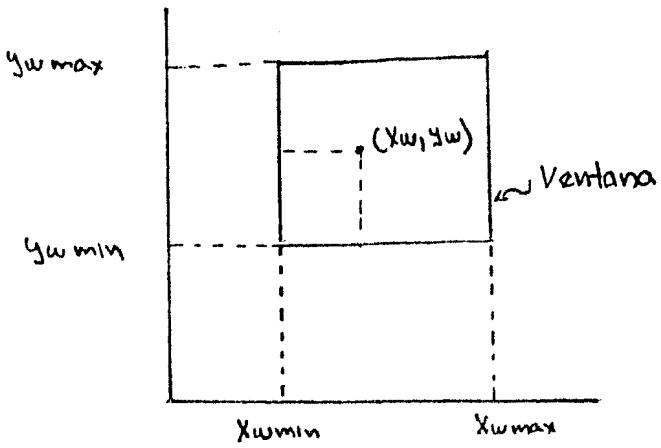


Sistema de Coordenadas del Graficador

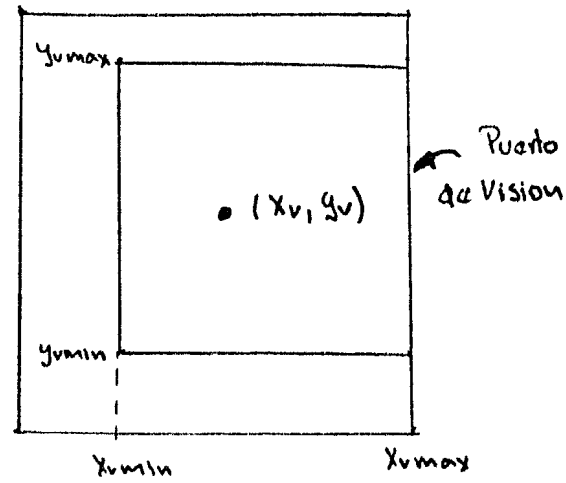




Una vez que se ha definido que es lo que se va ha desplegar, se hace el mapeo sobre el puerto de vision.



Sistema de Coordenadas del Objeto



Sistema de Coordenadas del Graficador

Para obtener el mapeo del punto (X_w, Y_w), al punto (X_v, Y_v) hay que ver que se conserve las proporciones, es decir, su posicion relativa dentro del rectangulo.

Asi que las condiciones requeridas son las siguientes:

$$\frac{X_{wd}}{X_{wl}} = \frac{X_w - X_{wmin}}{X_{wmax} - X_{wmin}} = \frac{X_v - X_{vmin}}{X_{vmax} - X_{vmin}} = \frac{X_{vd}}{X_{vl}}$$

y

$$\frac{Y_{wd}}{Y_{wl}} = \frac{Y_w - Y_{wmin}}{Y_{wmax} - Y_{wmin}} = \frac{Y_v - Y_{vmin}}{Y_{vmax} - Y_{vmin}} = \frac{Y_{vd}}{Y_{vl}}$$

De lo anterior se obtiene las ecuaciones para el mapeo:

$$X_v = X_{vmin} + \frac{X_{vmax} - X_{vmin}}{X_{wmax} - X_{wmin}} (X_w - X_{wmin})$$

$$Y_v = Y_{vmin} + \frac{Y_{vmax} - Y_{vmin}}{Y_{wmax} - Y_{wmin}} (Y_w - Y_{wmin})$$

las cuales tienen la forma

$$X_v = S_x (X_w - X_{wmin}) + X_{vmin}$$

$$Y_v = S_y (Y_w - Y_{wmin}) + Y_{vmin}$$

donde S_x y S_y son factores de escala de la ventana al puerto de vision.

ii)RECORTE.

Ya que los dispositivos de graficacion son de tamaño fijo (casi siempre rectangulares), entonces partes de un objeto que rebase las cotas de esta area puede causar problemas.Para ello se ha creado el algoritmo de recorte ("clipping") , para suprimir las lineas que rebasen estas cotas.

Tambien habra ocasiones en que se quiera dibujar solo las lineas que se encuentren en algun rectangulo dado y se cambie de escala y llene este toda el area fijada y es asi que se necesitara tambien de recortar aquello que este fuera del area.

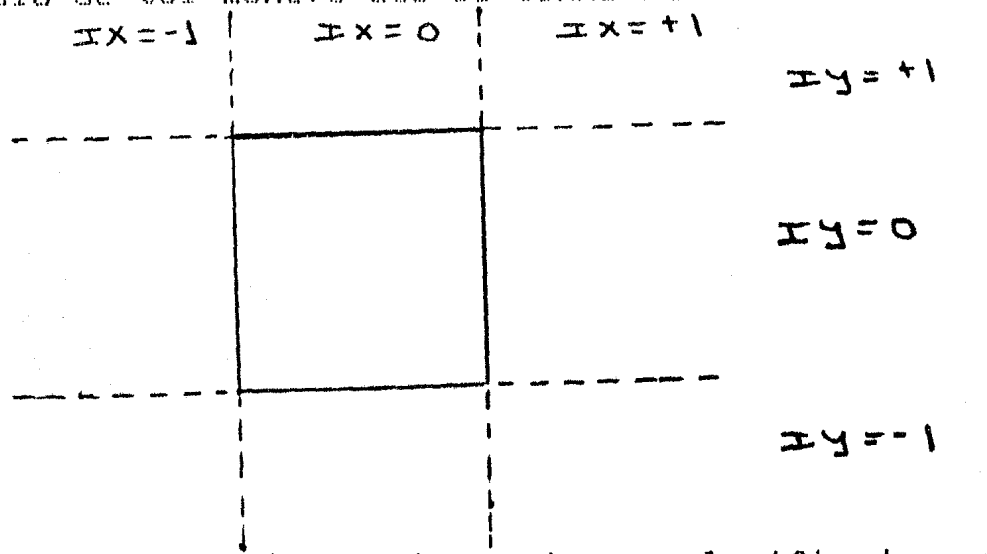
METODO:

El metodo para el recorte de lineas es el siguiente:

Asumir primero que el origen es el centro del rectangulo ,asi, si el rectangulo es de 2DX por 2DY los vertices del rectangulo son (+- DX ,+- DY).

El problema se reduce a ver cuales segmentos de linea , el cual resulta de unir (X_1, Y_1) con (X_2, Y_2) , tienen un subsegmento dentro del rectangulo.

La solucion al problema es extender los lados del rectangulo de tal manera que se tenga nueve sectores.



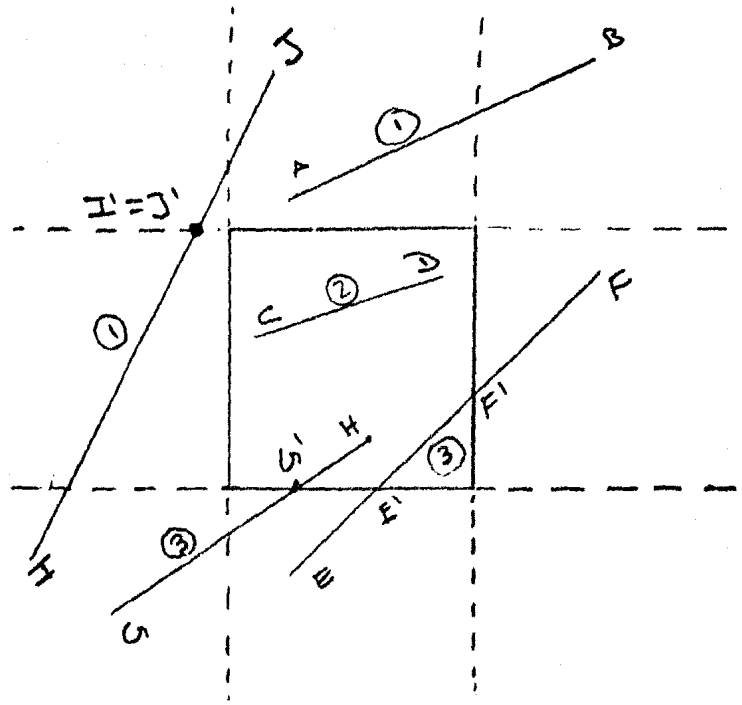
Cada punto en el espacio puede ser clasificado por dos parametros IX , IY donde:

1)IX = -1 o +1 dependiendo en donde se encuentre el valor de de la coordenada X , es decir, si esta a la izquierda , en medio o a la derecha de los lados verticales del rectangulo.

2) IY =-1 o +1 dependiendo en donde se encuentre el valor de la coordenada Y ,abajo o arriba o entre los lados horizontales del rectangulo.

Para encontrar el valor de IX ,IY para cada punto se tiene la rutina BUSCA. Lo que se presunta en esta rutina es si X se sale o no del area, es decir, si

ABS(X) > DX entonces IX puede tomar el valor de +1 o -1
,si el ABS(Y) > DY ,IY puede tomar el valor de +1 o -1
.



1) La rutina termina inmediatamente.

2) Las lineas son dibujadas inmediatamente.

3) La linea es recortada.

Si dos puntos que son los extremos del segmento de linea , por ejemplo (X_1, Y_1) y (X_2, Y_2) tiene los parametros IX1 ,IY1 y IX2 ,IY2 , respectivamente ,hay que considerar las siguientes posibilidades:

1.-Si IX1=IX2≠0 o IY1=IY2≠0 , entonces todo el segmento de linea esta afuera del rectangulo , asi la linea es totalmente ignorada.

2.- Si IX1=IY1=IX2=IY2=0, entonces el segmento de linea esta totalmente adentro del rectangulo ,asi la recta es dibujada completamente.

3.-Los demas casos hay que tratarlos con detalle.

Si IX1≠0 y/o IY1 ≠0 , entonces el punto descansa afuera del rectangulo, y un punto (X_1', Y_1') debe ser calculado.Este punto debe ser un punto cercano a (X1, Y1) , es el punto donde la linea corta al rectangulo.

Si X1=0 y Y1=0 entonces $(X_1', Y_1') = (X_1', Y_1')$, y lo mismo sucede para (X2 ,Y2) para obtener $(X2', Y2')$,

La linea a ser dibujada sera resultado de unir $(X1', Y1')$ con $(X2', Y2')$.

Si la linea degenera en $(X1', Y1') = (X2', Y2')$,este degenera en un punto y entonces es ignorado.

Resumiendo , se tiene dos puntos (X1 ,Y1) y (X2,Y2) que son los extremos de una linea , existen tres posibilidades:

1.- Que la rutina termine inmediatamente.

2.- La linea sea dibujada inmediatamente,

3.- Que la linea sea recortada.

La rutina es utilizada para cada segmento de linea a ser dibujada.

Cuando se desee recortar otra cosa que no sean rectas, por ejemplo curvas, se puede descomponer a estas en segmentos de linea muy cortas y luego recortarlas.

Puede ser un metodo no eficiente , pero es lo que generalmente se hace.

Para otro tipo de elemento grafico se descompone y se aplica cualquier tipo de algoritmo para recortar lineas.

3.5 METODOS DE INTERPOLACION PARA EL DISEÑO DE CURVAS.

i) REPRESENTACION DE CURVAS.

Los objetos son las formas mas complejas que ocurren en 3-D , y se ha estudiado tecnicas para modelarlos de manera apropiada y generar imagenes reales.

Estos objetos pueden ser aproximados por un conjunto de poligonos y luego aplicando determinadas transformaciones llegar al objeto deseado.

Una forma puede tener o no una representacion analitica ,pero generalmente lo que mas nos interesa es cuando no se tiene una representacion analitica ya que nos enfrentamos al problema de diseñar.

Va ha existir un diseñador que va ha querer crear o modificar un modelo.

Entonces hay que estar conscientes de la manera que se va ha expresa las modificaciones que se quiere.

Lo ideal es que el diseñador cambie solo un pequeño numero de parametros del modelo y alcanze el resultado deseado.

Se puede dar una lista de algunas propiedades para el diseño de curvas.

1.-Puntos de Control.

Una manera comun para controlar la forma de una curva es interaccionar con los puntos atravez de los cuales la curva pasa.

Estos puntos son llamados puntos de control y algunas veces llamados nodos(knots).Una curva se dice que interpola los puntos de control si pasa atravez de ellos.

2.- Multiples Valores.

En general una curva no es necesariamente una funcionde una sola variable, es independiente del sistema de coordenadas que se escoja.

3.- Independencia de los ejes de Coordenadas.

La forma del objeto no debe cambiar cuando los puntos de control son medidos en diferentes sistemas de coordenadas.

Por ejemplo , si los puntos de control son rotados 90 grados , la curva rotada no debe cambiar de forma.

4.- Control Global o Local.

Cuando un diseñador maneja los puntos de control , una curva puede cambiar solo de forma en la region cercana al punto de control , o puede cambiar de forma en toda ella.

Este ultimo comportamiento es llamado control

slobal ,puede este ultimo molestar al diseñador y sobre todo cuando se esta haciendo ajustes muy finos a una porcion de la curva.

5.- Propiedad de Variacion Minima.

Algunos metodos matematicos en lugar de obtener una buena aproximacion al polisono se presenta muchas irregularidades.Asi una curva que oscila es indeseable.

6.- Versatilidad.

Cuando la representacion de la curva solo permita una variedad limitada de formas puede frustar al diseñador.

La idea es quitar y añadir puntos de control para que la curva tome un numero grande de formas adicionales.

7.- Orden de Continuidad.

Una forma compleja usualmente no es modelada por una sola curva , pero variar curvas si son utilizadas uniendolas por los extremos.

Cuando se crean uniones , el diseñador querra controlar el orden de continuidad al unirlas.

con este tipo de requerimientos se empezara ha desarrollar tecnicas basicas usadas en la representacion de curvas y superficies.

se produce se aproxima mas al polisono definido.

Cuando k=2 , la curva generada es una serie de lineas rectas las cuales son identicas al polisono definido.

El orden de la curva se ve reflejada en el "knot vector" el cual es usado para generar la curva.

Es necesario especificar valores del "knot-vector" de multiplicidad k al principio y al final del conjunto de valores.

Ejemplo:

Consideremos un polisono con 5 vertices , con vertices no duplicados.

Cuando los vertices no aparecen duplicados, el parametro t varia de 0 a n-k+2 sobre toda la curva.

Por ejemplo, una curva de orden cuatro es continua en la primera y segunda derivada .

La curva B-spline es matematicamente definida como una funcion Spline Polinomial de orden k (grado k-1) ya que satisface las dos condiciones.

1) La funcion P(t) es un polinomio de grado k-1 en cada intervalo $(x_i \leq t \leq x_{i+1})$

2)P(t) y sus derivadas de orden 1,2,...,k-2 son todas continuas sobre toda la curva.

En particular , una curva B-spline de orden cuatro es un Spline cubico.

Parametros para cambiar la curva:

1.- Cambiar el orden de k para $2 \leq k \leq n+1$

2.- Usar vertices repetidos.

3.- Cambiar el numero y/o la posicion de los vertices no repetidos en el polisono definido.

La siguiente figura se muestra como una curva puede ser "jalada: a un vertice especifico por el uso de vertices repetidos.

Ejemplo:

Consideremos cuatro vertices de un polisono.

Sean estos $P_0[1\ 1]$, $P_1[2\ 3]$, $P_2[4\ 3]$

y $P_3[3.\ 1]$

El "knot vector" para una curva de orden 2 es $[0\ 0\ 1\ 2\ 3\ 3]$

donde $x_0 = 0$, $x_1 = 0$, $\cdots$, $x_2 = 1$ $\cdots$, $x_5 = 3$

Se usa la convencion $0/0 = 0$ .Los valores $N_{i,k}$

necesitados son :

$$N_{0,2}(t) = \frac{(t-0)\,N_{0,1}(t)}{x_1 - x_0} + \frac{(x_2 - t)\,N_{1,1}(t)}{x_2 - x_1}$$

$$N_{1,2}(t) = \frac{(t-0)\,N_{1,1}(t)}{x_2 - x_1} + \frac{(x_3 - t)\,N_{2,1}(t)}{x_3 - x_2}$$

$$N_{2,2}(t) = \frac{(t-1)\,N_{2,1}(t)}{x_3 - x_1} + \frac{(x_4 - t)\,N_{3,1}(t)}{x_4 - x_3}$$

$$N_{3,2}(t) = \frac{(t-2)\,N_{3,1}(t)}{x_4 - x_5} + \frac{(x_5 - t)\,N_{4,1}(t)}{x_5 - x_4}$$

Para $t = 0$

$N_{0,2}(0) = 0(1)\,|0 + (1-0)(1)\,|1 = 1.0$

$N_{1,2}(0) = 0(1)\,|1 + (2-0)(0)\,|1 = 0.$

$t = .5$

$N_{1,2}(.5) = 0.5\,(1)\,|1 + (2-.5)(0)\,|1 = 0.5$

$t = 1.0$

$N_{3,2}(1) = 1(0)\,|1 + 2(0)\,|0 = 0.$

Los demas se encuentran de manera similar.
De la ecuacion general

$$P(t) = \sum_{i=0}^{n} P_i \, N_{i,K}(t)$$

$$P(t) = P_0 \, N_{0,2} + P_1 \, N_{1,2} + P_2 \, N_{2,2} + P_3 \, N_{3,3}$$

Para t=0

$$P(0) = P_0(1) + P_1(0) + P_2(0) + P_3(0) = P_0$$

Similarmente

$$P(0.5) = P_0(0.5) + P_1(0.5) + P_2(0) + P_3(0)$$
$$= 0.5\,(P_0 + P_1)$$
$$P(1) = P_0(0) + P_1(1) + P_2(0) + P_3(0) = P_1$$
$$P(1.5) = P_0(0) + P_1(0.5) + P_2(0.5) + P_3(0)$$
$$= 0.5\,(P_1 + P_2)$$

De la misma manera

$$P(2) = P_2$$
$$P(2.5) = 0.5\,(P_2 + P_3)$$
$$P(3) = P_3$$

Si el orden de la curva ya no es dos sino cuatro para los mismos cuatro vertices , una curva de Bezier se obtendra ya que el orden es igual al numero de vertices.
El "knot vector" sera

$t_{max} = 3 - 4 + 2 = 1$ es $[0\,0\,0\,0\;\;1\,1\,1\,1]$

$$N_{0,4}(t) = \frac{(t-0)\,N_{0,3}(t)}{0-0} + \frac{(1-t)\,N_{1,3}(t)}{1-0}$$

$$N_{1,4}(t) = \frac{(t-0)\,N_{1,3}(t)}{1-0} + \frac{(1-t)\,N_{2,3}(t)}{1-0}$$

$$N_{2,4}(t) = \frac{(t-0)\ N_{2,3}(t)}{1-0} + \frac{(1-t)\,N_{3,3}(t)}{1-0}$$

$$N_{3,4}(t) = \frac{(t-0)\ N_{3,3}(t)}{1-0} + \frac{(1-t)\,N_{4,3}(t)}{1-1}$$

Ahora se requiere de calcular los $N_{i,3}$

$$N_{0,3}(t) = \frac{(t-0)\ N_{0,2}(t)}{0-0} + \frac{(1-t)\ N_{1,2}(t)}{0-0}$$

$$N_{1,3}(t) = \frac{(t-0)\,N_{1,2}(t)}{0-0} + \frac{(1-t)\,N_{2,2}(t)}{1-0}$$

$$N_{2,3}(t) = \frac{(t-0)\,N_{2,2}(t)}{1-0} + \frac{(1-t)\,N_{3,2}(t)}{1-0}$$

$$N_{3,3}(t) = \frac{(t-0)\,N_{3,2}(t)}{1-0} + \frac{(1-t)\,N_{4,2}(t)}{1-1}$$

$$N_{4,3}(t) = \frac{(t-1)\,N_{4,2}(t)}{1-1} + \frac{(1-t)\,N_{5,2}(t)}{1-1}$$

Ahora se requiere calcular los $N_{i,2}$

$$N_{0,2}(t) = \frac{(t-0)\ N_{0,1}(t)}{0-0} + \frac{(0-t)\,N_{1,1}(t)}{0-0}$$

$$N_{1,2}(t) = \frac{(t-0)\,N_{1,1}(t)}{0-0} + \frac{(0-t)\,N_{2,1}(t)}{0-0}$$

$$N_{2,2}(t) = \frac{(t-0)\,N_{2,1}(t)}{0-0} + \frac{(0-t)\,N_{2,1}(t)}{0-0}$$

$$N_{3,2}(t) = \frac{(t-0)\,N_{3,1}(t)}{1-0} + \frac{(1-t)\,N_{4,1}(t)}{1-1}$$

$$N_{4,2}(t) = \frac{(t-1)N_{4,1}(t)}{1-1} + \frac{(1-t)N_{5,1}(t)}{1-1}$$

$$N_{5,2}(t) = \frac{(t-1)N_{5,1}(t)}{1-1} + \frac{(1-t)N_{6,1}(t)}{1-1}$$

Para $t = 0.5$

$$N_{0,1} = N_{1,1} = N_{2,1} = N_{4,1} = N_{5,1} = N_{6,1} = 0$$

$$N_{3,1} = 1.0$$

Asi $N_{0,2} = N_{1,2} = N_{4,2} = N_{5,2} = 0$ y

$$N_{2,2}(0.5) = 0 + \frac{(0.5)(1)}{1} = 0.5$$

$$N_{3,2}(0.5) = \frac{(0.5)(1)}{1} + 0 = 0.5$$

Para $N_{0,3} = N_{4,3} = 0$ y

$$N_{1,3} = 0 + \frac{(0.5)(0.5)}{1} = 0.25$$

$$N_{2,3} = \frac{(0.5)(0.5)}{1} + \frac{(0.5)(0.5)}{1} = .5$$

$$N_{3,3} = \frac{(0.5)(0.5)}{1} + 0 = .25$$

Para $N_{0,4} = 0 + \frac{(0.5)(0.25)}{1} = 0.125$

$$N_{1,4} = \frac{(0.5)(0.25)}{1} + \frac{(0.5)(0.5)}{1} = 0.375$$

$$N_{2,4} = \frac{(0.5)(0.5)}{1} + \frac{0.5(0.25)}{1} = .375$$

$$N_{3,4} = \frac{(0.5)(0.25)}{1} + 0 = .125$$

Finalmente en $t = 0.5$ $P(0.5) = P_0 N_{0,4} + P_1 N_{1,4} + P_2 N_{2,4} + P_3 N_{3,4}$

o

$$P(0.5) = [1\ 1]\,0.125 + [2\ 3]\,0.375 + [4\ 3]\,0.375 + [3\ 1]\,0.125 = [2.75 \quad 2.5]$$

II)CURVAS DE BEZIER. INTRODUCCION .- P. Bezier , de la compañia francesa RENAULT , es pionero del uso de la computadora para el modelado de superficies en el diseño de automoviles.

Su sistema UNISURF , ha sido usado por diseñadores desde 1972, aplicando este sistema a varios carros de la marca RENAULT.

METODO.- Bezier define la curva $P(u)$ en terminos de los n+1 puntos de control P_i

$$P(u) = \sum_{i=0}^{n} P_i \, B_{i,n}(u) \qquad \ldots (1)$$

donde $B_{i,n}(u)$ es la funcion blending.

$$B_{i,n}(u) = C(n,i)\, u^i (1-u)^{n-i}$$

y $C(u,i)$ es el coeficiente binomial ,

$$C(u,i) = \frac{n!}{((i)!\,(n-i)!)}$$

La ecuacion (1) puede ser escrita como funciones separadas:

$$x(u) = \sum_{i=0}^{n} x_i \, B_{i,n}(u)$$

$$y(u) = \sum_{i=0}^{n} y_i \, B_{i,n}(u)$$

$$z(u) = \sum_{i=0}^{n} z_i \, B_{i,n}(u)$$

donde los puntos de control P_i son $[x_i \; y_i \; z_i]$

Las funciones blending son la clave del

comportamiento de las curvas de Bezier.

Podemos evaluar a las curvas de Bezier en terminos de la siguiente lista de propiedades.

1.- Puntos de Control.

Lo primero que hay que observar de las curvas de Bezier es su dificultad de uso porque no todos los puntos de control van a estar en la curva. La curva es predecible en relacion a los puntos de control ya que cada vez que se agrega un punto de control da la sensacion de que "jala" a la curva en esta posicion de la curva.

Los puntos de control tambien satisface dos importantes propiedades matematicas: la curva pasa por los extremos o sea por los puntos de control del principio y final de la curva (P0 y Pn) , y la curva es tangente al vector que une a P0 y P1.

2.- Valores Multiples.

La formulacion parametrica de la curva de Bezier permite representar formas con multiples valores. En efecto , si el primer y ultimo punto de control coinciden , la curva es cerrada.

3.- Independencia de Ejes.

Una curva de Bezier es independiente del sistema de coordenadas usado.

4.- Control Global o Local.

Estas curvas no proporcionan control local : moviendo cualquier punto de control cambiara la forma de toda la curva.

5.- Propiedades de Variacion.

Las curvas de Bezier son de variacion minima.

6.- Versatilidad.

La versatilidad de una curva de Bezier esta en el numero de puntos de control usados.

Se puede usar un buen numero de puntos de control para definir formas mas complejas.

7.- Orden de Continuidad.

Las curvas de Bezier de orden muy alto pueden ser unidas para describir curvas mas complejas.En estos casos, las uniones entre las curvas deben ser suaves.

Ejemplo:

Tomemos como puntos de control los siguientes vectores

$P_0 [1\ 1]$, $P_1 [2\ 3]$, $P_2 [4\ 3]$ y $P_3 [3\ 1]$

Mencionemos las ecuaciones utilizadas:

$$P(t) = \sum_{i=0}^{n} P_i \; B_{n,i}(t)$$

donde

$$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

y

$$\binom{n}{i} = \frac{n!}{i!\,(n-i)!}$$

Aqui n=3 , ya que se tiene 4 vertices

asi $$\binom{n}{i} = \frac{6}{i!\,(3-i)!}$$

y

$$B_{3,0}(t) = (1)\,t^0\,(1-t)^3 = (1-t)^3$$

$$B_{3,1}(t) = 3t\,(1-t)^2$$

$$B_{3,2}(t) = 3t^2\,(1-t)$$

$$B_{3,3}(t) = t^3$$

Entonces

$$P(t) = P_0 B_{3,0} + P_1 B_{3,1} + P_2 B_{3,2} + P_3 B_{3,3}$$

Tomando varios valores para t se obtiene la siguiente tabla:

Coeficientes para la Curva de Bezier

t	$B_{3,0}$	$B_{3,1}$	$B_{3,2}$	$B_{3,3}$
0	1.0	0	0	0
0.15	0.614	0.325	0.0574	0.0034
0.35	0.275	0.444	0.239	0.043
0.5	0.125	0.375	0.375	0.125
0.65	0.043	0.239	0.444	0.275
0.85	0.0034	0.0574	0.325	0.614
1.0	0	0	0	1

$P(0) = P_0 = [1 \quad 1]$

$P(0.15) = 0.614\,P_0 + 0.325\,P_1 + 0.0574\,P_2 + 0.0034\,P_3 = [1.5 \quad 1.765]$

$P(0.35) = 0.275\,P_0 + 0.444\,P_1 + 0.239\,P_2 + 0.043\,P_3 = [2.248 \quad 2.367]$

$P(0.5) = 0.125\,P_0 + 0.375\,P_1 + 0.375\,P_2 + 0.125\,P_3 = [2.75 \quad 2.5]$

$P(0.65) = 0.043\,P_0 + 0.239\,P_1 + 0.444\,P_2 + 0.275\,P_3 = [3.122 \quad 2.36]$

$P(0.85) = 0.0034\,P_0 + 0.514\,P_1 + 0.325\,P_2 + 0.614\,P_3 = [3.248 \quad 1.75]$

$P(1) = P_3 = [3 \quad 1]$

iii) CURVAS B-SPLINE.

Desde el punto de vista matematico , una curva la cual es generada usando los vertices de un poligono definido es dependiente de alguna interpolacion o algun esquema de aproximacion para establecer la relacion entre curva y poligono.

Este esquema depende de la forma en que se escoga la base (elemento principal) o tambien llamada funcion de peso o "weighting function".

La base B-spline es generalmente no global.El comportamiento no global de las curvas B-spline es debido al hecho de que cada vertice Pi(del poligono) es asociado con una unica funcion base.

Entonces ,cada vertice afecta la forma de la curva solo sobre un rango que pertenece a los valores del parametro donde la funcion base asociada a estos es distinta de cero.

La base B-spline permite controlar el resultado de la curva sin cambiar el numero de vertices que define al poligono.

Sea $P(t)$ la posicion de los vectores a lo largo de la curva,como una funcion del parametro t , una curva es generada usando la base B-Spline , esta es dada de la
siguiente manera:

$$P(t) = \sum_{i=0}^{n} P_i \, N_{i,k}(t)$$

donde los P_i son los $n+1$ vertices que definen el poligono.
Para la i-esima base B-spline de orden K , la funcion de peso o base $N_{i,k}(t)$ esta definida por formulas de recursion.

$$N_{i,1}(t) = \begin{cases} 1 & \text{si } x_i \le t \le x_{i+1} \\ & i = 0, \dots, n \\ 0 & \text{en otro caso} \end{cases}$$

y

$$N_{i,k}(t) = \frac{(t - x_i) N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t) N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}}$$

Los valores de x_i son elementos que llamamos "knot vector".
El parametro t varia de 0 a t_{max} a lo largo de la curva

Una variable adicional debe ser usada para las curvas B-spline para tener la flexibilidad mencionada.
Esta es alcanzada usando el "knot vector".

Un "knot vector" es simplemente una serie de valores enteros x_i , tales que $x_i \leq x_{i+1} \quad \forall x_i$

Ejemplos de estos vectores son

$$[\,0\ 1\ 2\ 3\ 4\,] \quad y \quad [\,0\ 0\ 0\ 1\ 1\ 2\ 3\ 3\ 3\,]$$

Los valores de x_i son considerados como "parametric knots".
Estos son usados para indicar el rango del parametro t para generar la curva B-spline con $0 \leq t \leq t_{max}$
Por ejemplo , el vector $[0\ 1\ 2\ 3\ 4]$ indica que el parametro t varia de 0 a 4.
El numero de "knot vector" intermedios depende del numero de tramos que
define el polisono.

Los valores duplicados en el "knot vector" indica multiplicidad en ese vertice (tramo de longitud cero).

Es conveniente usar valores con una unidad de separacion entre nodos.

Junto con los valores del "knot vector" , el orden de la curva debe ser especificado.

Si el orden k es igual al numero de vertices del polisono , y no hay vertices multiples , entonces se genera una curva de Bezier.
Para una curva de orden 3 (k=3) con 5 vertices , el valor de $t_{max} = n-k+2 = 4-3+2$
Hay que crear el "knot vector" ,usando multiplicidad 3

al principio y al final.

$$[\,0\ 0\ 0\ 1\ 2\ 3\ 3\ 3\,]$$ multiplicidad 3

Una curva de orden 2 va ha tener multiplicidad 2 al principio y al final del "knot vector"

$$t_{max} = n-k+2 = 4-2+2 = 4$$

$$[\,0\ 0\ 1\ 2\ 3\ 4\ 4\,]$$

Un polisono con siete vertices y de orden tres

$$t_{max} = n-k+2 = 6-3+2 = 5$$

la multiplicidad es 3

$$[\,0\ 0\ 0\ 1\ 2\ 3\ 4\ 5\ 5\ 5\,]$$

Cuando el orden de la curva decrece , la curva que

C A P I T U L O I V

CAPITULO IV

ELEMENTOS MATEMATICOS PARA LA REPRESENTACION DE OBJETOS EN TRES DIMENSIONES EN UN ESPACIO DE DOS DIMENSIONES.

4.1 INTRODUCCION.
4.2 TRANSFORMACIONES GEOMETRICAS EN TRES DIMENSIONES.
  4.2.1 Representacion Matricial de Transformaciones en 3D.
  4.2.2 Composicion de Transformaciones en 3D.
4.3 SOLUCION A LA REPRESENTACION DE OBJETOS EN TRES DIMENSIONES EN UN ESPACIO DE DOS DIMENSIONES.
  4.3.1 Introduccion.
  4.3.2 Clasificacion de las Proyecciones Geometricas Planas.
  4.3.3 Bases Matematicas para las Proyecciones Geometricas Planas.
    4.3.3.1 Introduccion.
    4.3.3.2 Proyeccion Ortografica.
    4.3.3.3 Proyeccion Oblicua.
    4.3.3.4 Proyeccion en Perspectiva.
    4.3.3.5 Proyeccion Estereopar.

4.1 INTRODUCCION.

El proposito de este capitulo es introducir las transformaciones geometricas en 3D usadas en Graficacion por Computadora. La translacion , escalamiento y la rotacion vistas aqui son la parte escencial de muchas aplicaciones en Graficacion.

Las transformaciones son aplicadas tanto en Paquetes de Graficacion como en Programas de Aplicacion. Por ejemplo , la operacion de manejar la imagen en 2D , se usa translacion y escalamiento para mapear la imagen del sistema de coordenadas del objeto al sistema de coordenadas del dispositivo de graficacion.

Igualmente importante es el manejo de transformaciones en 3D. la rotacion y la translacion seran usadas para cambiar de posicion, orientacion y tamaño del objeto en el dibujo.

El proceso del manejo de imagen en 3D es

inherentemente mas complejo que el proceso en 2D . En 2D, se puede simplemente especificar una ventana sobre el sistema de coordenadas del objeto y manejar un puerto de vision sobre una superficie en 2D.

Conceptualmente , los objetos en el sistema de coordenadas del objeto son recortados con respecto a una ventana y entonces son transformados dentro de una ventana del dispositivo de graficacion.

La solucion a la no compatibilidad entre objetos en 3D y los dispositivos de graficacion es resuelto introduciendo las "Proyecciones" , las cuales transforman objetos en 3D sobre el "plano de proyeccion " en 2D.

En este capitulo se presenta el material necesario (transformaciones geometricas en 3D y proyecciones geometricas planas) para llegar a obtener la representacion de las funciones de dos variables.

4.2 TRANSFORMACIONES GEOMETRICAS EN TRES DIMENSIONES.

4.2.1 Representacion Matricial de Transformaciones en 3D.

La representacion en 2D tiene su paralelo en 3D, las cuales son matrices de 4X4.

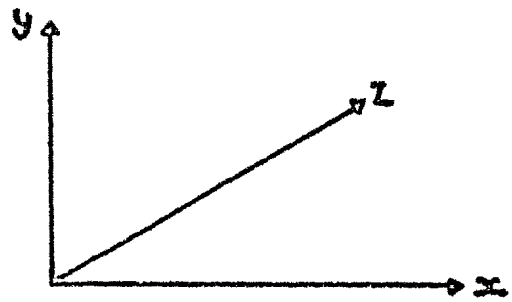
Para esto , un punto (x,y,z) en coordenadas cartesianas sera en coordenadas homogeneas (xw,yw,zw,w) con $w \neq 0$.

Si $w \neq 1$,entonces hay que dividir por w para obtener el valor de (x,y,z) en coordenadas cartesianas.

Usando el sistema de coordenadas derecho,las rotaciones positivas son las que van en contra de las manecillas de reloj.

Si el eje de rotacion es	Direccion de rotacion
x	y a z
y	z a x
z	x a y

En graficacion es mas deseable pensar en un sistema izquierdo ya que va mas con la manera en que se puede manejar el sistema en pantalla o sobre el papel.



4.1 Sistema de Coordenadas de la Pantalla.

Las rotaciones son en direccion a las manecillas del reloj.

Usando coordenadas homogeneas la Translacion se define como :

$$T(Dx, Dy, Dz) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ Dx & Dy & Dz & 1 \end{bmatrix}$$

$$[x\ y\ z\ 1] \times T[Dx, Dy, Dz] = [x+Dx\ \ y+Dy\ \ z+Dz\ \ 1]$$

El Escalamiento como :

$$S(Sx, Sy, Sz) = \begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$[x\ y\ z\ 1] \times S(Sx, Sy, Sz) = [x \cdot Sx\ \ y \cdot Sy\ \ z \cdot Sz\ \ 1]$$

La Rotacion con respecto a los tres ejes:

$$R_z(\theta) = \begin{bmatrix} \cos\theta & \operatorname{sen}\theta & 0 & 0 \\ -\operatorname{sen}\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_x(\theta)=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \operatorname{sen}\theta & 0 \\ 0 & -\operatorname{sen}\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_y(\theta)=\begin{bmatrix} \cos\theta & 0 & -\operatorname{sen}\theta & 0 \\ 0 & -1 & 0 & 0 \\ \operatorname{sen}\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Cada una de estas transformaciones tiene sus inversas.
La inversa de T es negando Dx, Dy, Dz ; para S se

reemplaza Sx, Sy, Sz por sus recíprocos; las rotaciones : negando el ángulo de rotación.

4.2.2 Composicion de Transformaciones en 3D.

Una vez que se han definido las transformaciones basicas en 3D , estas son usadas para resolver problemas como el de rotar un objeto sobre un eje dado , o el de transladar un objeto de un punto a otro, etc.

Como en el caso de dos dimensiones ,se descompone el problema en otros mas sencillos usando una serie de transformaciones que llevaran a la solucion del problema.

Ya que las transformaciones vistas son con respecto a el origen , los tres tipos de transformaciones (translacion, escalamiento y rotacion) , tendran su transformacion inversa que regresara al objeto a su posicion original.

Si A es la matriz de transformacion logicamente su inversa sera A^{-1} , No sera necesario calcular directamente la matriz inversa,en el caso de la translacion , su inversa es negando sus argumentos -Tx,-Ty,-Tz , para el escalamiento sera 1/Sx , 1/Sy , 1/Sz y para la rotacion , sera -Angulo.

Si una transformacion T esta dada por una serie de transformaciones elementales A1,.....,An (esto es , T=A1XA2X....XAn; hay que tener cuidado con el orden) entonces la transformacion inversa sera $T^{-1}= An^{-1}X \ldots X A2^{-1} X A1^{-1}$ ($T X T^{-1}= I$, la matriz identidad).

Hay que recordar que la multiplicacion de matrices no es conmutativa.

ROTACION CON RESPECTO A UN EJE ARBITRARIO.

En general cualquier recta en el espacio puede servir como eje de rotacion. El problema es obtener la matriz de rotacion con un angulo con respecto a una recta.

En este caso, la transformacion puede ser realizada en cuatro etapas.

1.- Mover el eje al origen.

2.- Rotar con respecto a X hasta que el eje de rotacion este en el plano XZ.

3.- Rotar con respecto a Y hasta que el eje Z corresponda al eje de rotacion.

4.- Rotar con respecto a Z.

Finalmente , se aplica las transformaciones inversas!

4.2.2 Composicion de Transformaciones en 3D.

Una vez que se han definido las transformaciones basicas en 3D , estas son usadas para resolver problemas como el de rotar un objeto sobre un eje dado , o el de transladar un objeto de un punto a otro, etc.

Como en el caso de dos dimensiones ,se descompone el problema en otros mas sencillos usando una serie de transformaciones que llevaran a la solucion del problema.

Ya que las transformaciones vistas son con respecto a el origen , los tres tipos de transformaciones (translacion, escalamiento y rotacion) , tendran su transformacion inversa que regresara al objeto a su posicion original.

Si A es la matriz de transformacion logicamente su inversa sera A^{-1} . No sera necesario calcular directamente la matriz inversa,en el caso de la translacion , su inversa es negando sus argumentos -Tx,-Ty,-Tz , para el escalamiento sera 1/Sx , 1/Sy , 1/Sz y para la rotacion , sera -Angulo.

Si una transformacion T esta dada por una serie de transformaciones elementales A1,.....,An (esto es , T=A1XA2X.....XAn, hay que tener cuidado con el orden) entonces la transformacion inversa sera T^{-1}= An^{-1}X ....X $A2^{-1}$X $A1^{-1}$(T X T^{-1}= I , la matriz identidad).

Hay que recordar que la multiplicacion de matrices no es conmutativa.

ROTACION CON RESPECTO A UN EJE ARBITRARIO.

En general cualquier recta en el espacio puede servir como eje de rotacion. El problema es obtener la matriz de rotacion con un angulo con respecto a una recta.

En este caso, la transformacion puede ser realizada en cuatro etapas.

1.- Mover el eje al origen.

2.- Rotar con respecto a X hasta que el eje de rotacion este en el plano XZ.

3.- Rotar con respecto a Y hasta que el eje Z corresponda al eje de rotacion.

4.- Rotar con respecto a Z.

Finalmente , se aplica las transformaciones inversas!

5.- La rotacion inversa para el eje Y.

6.- La rotacion inversa para el eje X.

7.- La translacion para regresar al eje de rotacion y las coordenadas a su posicion original.

Un punto del eje de rotacion con su direccion es suficiente para especificar este eje.

El punto proporciona informacion para la translacion , y la direccion dara los angulos correctos para hacer la rotacion y hacer la alineacion con el eje Z.

La ecuacion parametrica para una recta esta dada por

$$x = Au + x_1$$

$$y = Bu + y_1$$

$$z = Cu + z_1$$

un punto sobre la recta es (x_1, y_1, z_1) y la direccion del vector esta especificada por el vector $[A, B, C]$.

Se puede ahora determinar la matriz de transformacion necesitada para la rotacion general de la recta.

La translacion para mover el punto (x_1, y_1, z_1) que esta sobre la recta , sera

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -x_1 & -y_1 & -z_1 & 1 \end{bmatrix}$$

Tambien se necesitara de su inversa para regresar a su posicion original una vez que se ha completado las rotaciones.

$$T^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ x_1 & y_1 & z_1 & 0 \end{bmatrix}$$

La siguiente estapa es el proceso de rotacion con respecto al eje X. Se quiere rotar hasta que el eje coincida con el plano XZ.

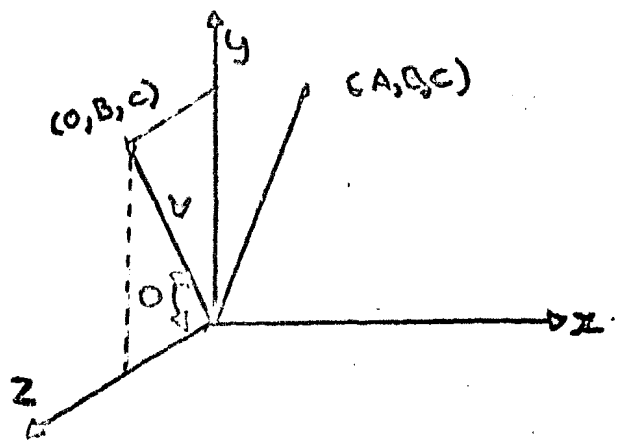
Para determinar al angulo de rotacion, hay que considerar la direccion del vector en el nuevo origen y su proyeccion en el plano YZ.

Consideremos su proyeccion en el plano YZ , el eje ira del (0,0,0) al (0,B,C) .

Si este eje se rota con respecto a X , el segmento de recta descansara sobre el eje Z.

Calculando el angulo a partir de la proyeccion sera: la longitud de la recta proyectada sera:

$$V = \sqrt{B^2 + C^2}$$



4.2 Proyeccion de un segmento de recta sobre el plano YZ.

y por definicion de seno y coseno ,

$$\text{Sen } \theta = B / V$$

$$\cos \theta = C / V$$

la rotacion con respecto a X sera

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \text{sen}\,\theta & 0 \\ 0 & -\text{sen}\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Asi que se tiene

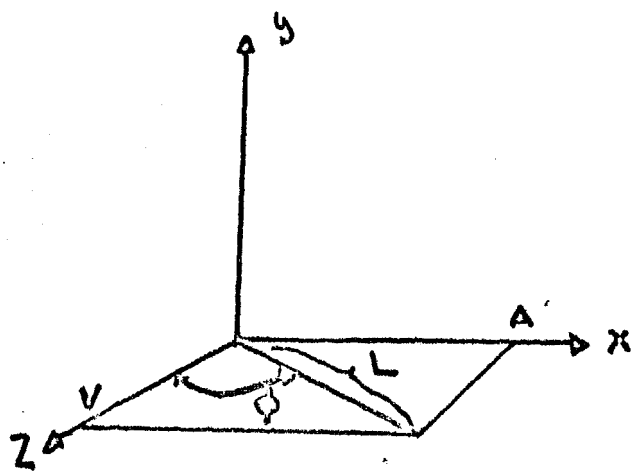
$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c/v & B/v & 0 \\ 0 & -B/v & c/v & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Su matriz inversa es la misma pero con direccion opuesta .

Los cosenos se dejan sin cambio

$$R_x^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c/v & -B/v & 0 \\ 0 & B/v & c/v & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Despues de la rotacion el eje descansa sobre el plano XZ.



4.3 Rotacion del eje sobre el plano XZ.

La rotacion con respecto a X , no alterara la coordenada en x, tambien se conserva la longitud del segmento asi que

$$L = \sqrt{A^2 + B^2 + C^2}$$

La coordenada en z sera:

$$C = \sqrt{(L^2 - A^2)} = (A^2 + B^2 + C^2 - A^2)^{1/2} = (B^2 + C^2)^{1/2}$$
$$= V$$

Asi si se quiere rotar con un angulo ϕ con respecto al eje Y el sesmento de recta se alineara con el eje Z.

$$\operatorname{sen} \phi = A / L$$

$$\cos \phi = V / L$$

La matriz de rotacion sera

$$R_y = \begin{bmatrix} \cos\phi & 0 & \operatorname{sen}\phi & 0 \\ 0 & 1 & 0 & 0 \\ -\operatorname{sen}\phi & 0 & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} V/L & 0 & A/L & 0 \\ 0 & 1 & 0 & 0 \\ -A/L & 0 & V/L & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

La inversa de la transformacion es:

$$R^{-1}_y = \begin{bmatrix} V/L & 0 & -A/L & 0 \\ 0 & 1 & 0 & 0 \\ A/L & 0 & V/L & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Finalmente estara en el lugar adecuado para realizar la rotacion con un angulo ω con respecto

al eje arbitrario.

$$R_2 = \begin{bmatrix} \cos\omega & \operatorname{sen}\omega & 0 & 0 \\ -\operatorname{sen}\omega & \cos\omega & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

De esta manera se ha obtenido la serie de transfoemaciones a partir de las cuales se obtendra una sola matriz de transformacion que se aplicara a cada uno de los puntos que componen la figura:

$$R_\theta = T R_x R_y R_2 R_y^{-1} R_x^{-1} T^{-1}$$

4.3 SOLUCION A LA REPRESENTACION DE OBJETOS EN TRES DIMENSIONES EN UN ESPACIO DE DOS DIMENSIONES.

4.3.1 INTRODUCCION.

Un problema que se presenta en Graficacion es la de dar la solucion a la representacion de objetos en tres dimensiones en graficadores donde su espacio es de dos dimensiones.

La solucion a este problema es resuelto con el uso de las "Proyecciones Planas".

Las proyecciones de este tipo son conocidas como proyecciones geometricas planas porque la proyeccion es sobre un plano y no sobre una superficie curva y se usan rectas mas que proyectores curvos.

Muchas proyecciones cartograficas son no-planas o no-geometricas [RICH 72].

En general , las proyecciones transforman puntos de un sistema de coordenadas de dimension n en un sistema de coordenadas n-1.

En este caso la proyeccion es de 3D a 2D. Para producir una vista en dos dimensiones de un objeto , cada punto del objeto debe ser mapeado sobre un plano (plano de proyeccion).

Dependiendo del tipo de mapeo usado se podra decir que tipo de proyeccion se esta usando y que efectos visuales se espera.

Una proyeccion geometrica plana de un objeto es obtenida pasando lineas llamadas "proyectores" uno atravez de cada punto del objeto, y encontrar la imagen formada por la interseccion de estos proyectores con un plano de proyeccion.

Los proyectores emanan de un solo punto llamado "centro de proyeccion".

Cuando la distancia del centro de proyeccion al plano de proyeccion es finita, una proyeccion en perspectiva es obtenida.

En cambio , cuando el centro de proyeccion es un punto al infinito , los proyectores son paralelos y se obtiene en el plano de proyeccion una imagen la cual es una proyeccion paralela.

De esta manera podemos decir que las proyecciones se clasifican en dos clases: proyeccion en perspectiva y proyecciones paralelas.

Para cualquier tipo de proyeccion existe dos aproximaciones para obtener la vista deseada del objeto.

1) Transformar el objeto y proyectarlo .

2)Escoger nuevos planos de proyeccion.

Para el primero , se usa fijar el centro de proyeccion y el plano de proyeccion y mover el objeto en la posicion deseada.

Para el segundo , se deja fijo el objeto y se escoge el centro de proyeccion y el plano de proyeccion para obtener la vista deseada.

Para escoger el tipo de proyeccion depende de un numero de factores:

1) Para que se quiere la representacion de la imagen.

2) Los efectos visuales que son deseados.

3) La forma del objeto.

Los diferentes tipos de proyecciones en perspectiva y proyecciones paralelas son vistas ampliamente en un articulo escrito por Carlbom y Parciorek [CARL 78].

4.3.2 CLASIFICACION DE LAS PROYECCIONES GEOMETRICAS PLANAS.

PROYECCION EN PERSPECTIVA.

La proyeccion en perspectiva crea un efecto visual similar al sistema fotografico y al sistema visual humano, y este es usado cuando se quiere alcanzar un alto grado de realismo en las figuras.

Esta proyeccion no es util si se desea obtener la forma y las medidas exactas del objeto ya que las distancias no pueden ser tomadas de la proyeccion y los angulos son conservados solo en las caras del objeto que son paralelas al plano de proyeccion y generalmente las lineas paralelas no se proyectan como lineas paralelas.

Las proyecciones en perspectiva de cualquier conjunto de lineas paralelas las cuales no son paralelas al plano de proyeccion conversen en un punto de concurrencia.

Se va a tener a lo mas tres puntos de concurrencia que corresponden al numero de ejes coordenados cortados por el plano de proyeccion.

Por ejemplo, si el plano de proyeccion corta al eje Z (y por lo tanto normal a este) , solo el eje Z tiene un punto de concurrencia porque las lineas paralelas son paralelas al eje X o al eje Y y tambien paralelas al plano de proyeccion y por lo tanto no tiene puntos de concurrencia.

Las proyecciones en perspectiva son clasificadas por el numero de puntos de concurrencia y de eso depende del numero de ejes que corte el plano de proyeccion.

Las proyecciones con dos puntos de concurrencia son muy usadas en arquitectura , insenieria ,diseño industrial y CAD.

Las proyecciones con tres puntos de concurrencia no son muy usadas porque son dificil de construir y ademas se pierde mucho realismo.

PROYECCIONES PARALELAS.

La proyeccion en paralelo es menos real y no da medidas exactas y las lineas lineas paralelas de la figura van a permanecer paralelas.

Los angulos se conservan solo en las caras del objeto que son paralelas al plano de proyeccion .

La clasificacion de las proyecciones paralelas es determinada por el angulo formado entre los proyectores y el plano de proyeccion(Figura 4.12).

Cuando los proyectores son perpendiculares al plano de proyeccion, la proyeccion es ortografica (Figura 4.6), de otra manera se llama obli (Figura 4.7).

Dentro de las proyecciones ortograficas se tiene las proyecciones ortograficas multivisuales y las proyecciones axonometricas.

La clasificacion va a depender de como se maneja el plano de proyeccion y los proyectores.

De esta manera con los proyectores perpendiculares al plano de proyeccion y el plano de proyeccion paralelo a alguno de los ejes coordenados (por ejemplo Z=0) , se va a tener las proyecciones ortograficas multivisuales.Con esta proyeccion se va a tener varias vistas del mismo objeto.

Solo se necesita tres vistas la parte de arriba, la lateral y el frente pero no se obtiene la forma tridimensional del objeto.

En la proyeccion Axonometrica , los proyectores son perpendiculares al plano de proyeccion , y este corta a los ejes coordenados de tal manera que no va a ser normal a ninguno de los ejes coordenados.

Asi la clasificacion de la proyeccion axonometrica va a depender de los angulos formados por el plano de proyeccion y los ejes coordenados.

Si los tres angulos son iguales la proyeccion es isometrica ; si solo dos angulos son iguales la proyeccion es dimetrica. Si todos los angulos son diferentes , la proyeccion es trimetrica.

La proyeccion Oblicua , es la proyeccion que es obtenida con los proyectores no perpendiculares al plano de proyeccion .El plano de proyeccion va a ser normal a uno de los ejes coordenados.

La cara del objeto que queda paralela al plano de proyeccion conserva las medidas y los angulos y tambien se conservan las medidas de todo el objeto pero los demas angulos no se conservaran.

Dentro de la proyeccion Oblicua , se tiene dos muy importantes :la proyeccion Cavalier y la proyeccion Cabinet.

Para la proyeccion Cavalier , se tiene que los proyectores forman un angulo de 45 grados con el plano de proyeccion.

Para las proyecciones Cabinet , el angulo que forman los proyectores con el plano de proyeccion es de arcot(1/2).

Las proyecciones Cabinet son un poco mas reales que las Cavalier.

En la siguiente seccion se desarrollaran los modelos matematicos necesarios para obtener las Proyecciones en Perspectiva y Paralelas(Ortografica Cabinet y Cavalier) para integrarlas al paquete de Graficacion.

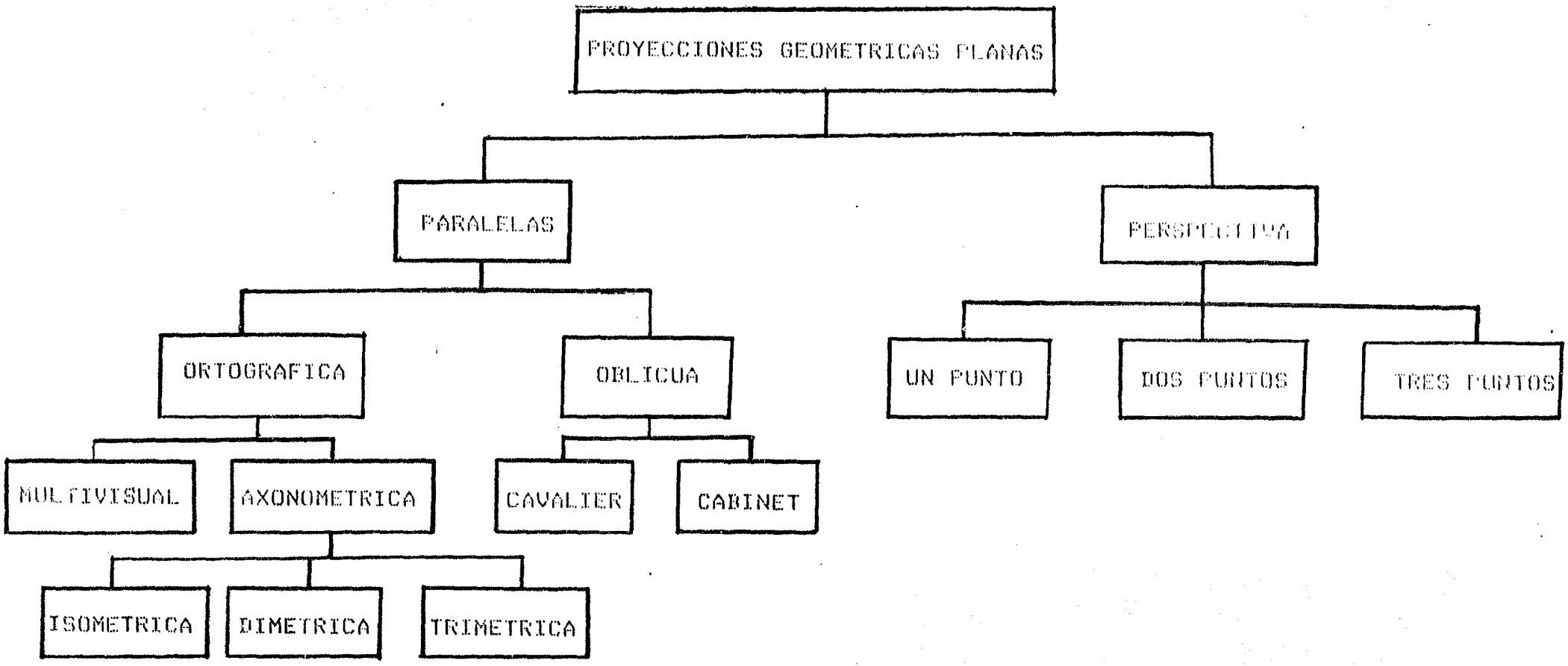


TABLA 4.1 CLASIFICACION DE LAS PROYECCIONES GEOMETRICAS PLANAS

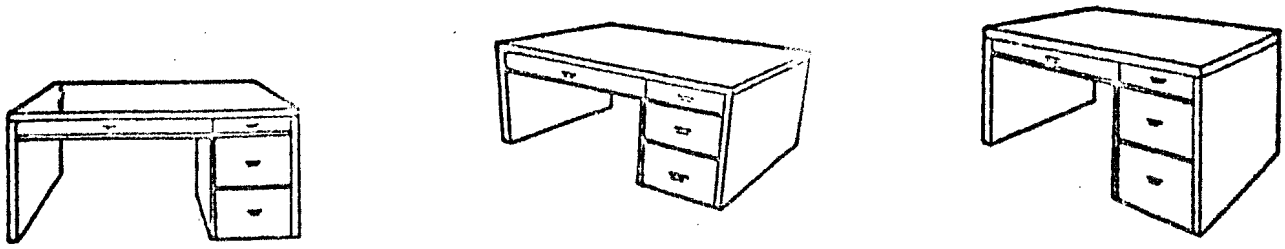


Fig. 4.4 Efectos de la Proyeccion en Perspectiva.

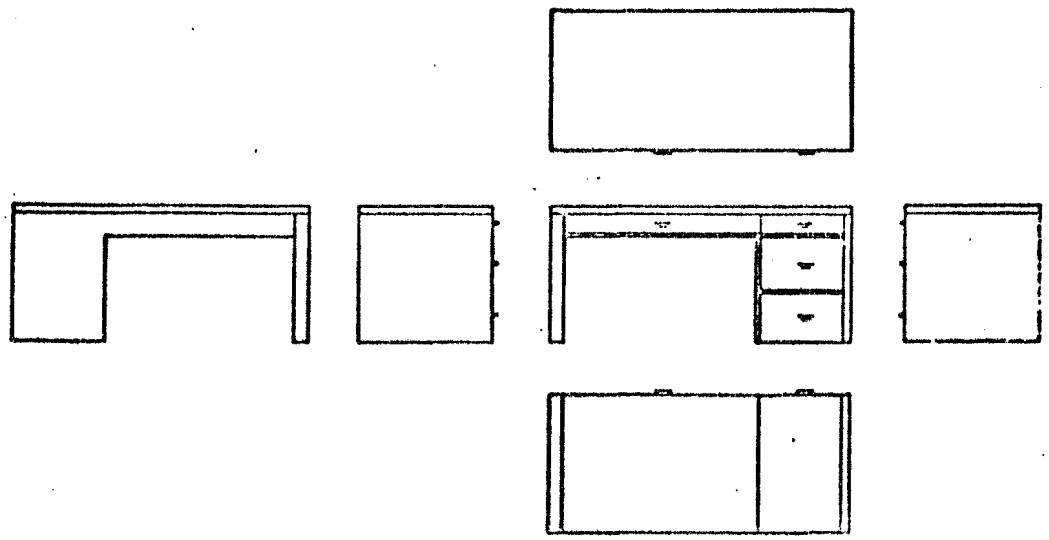
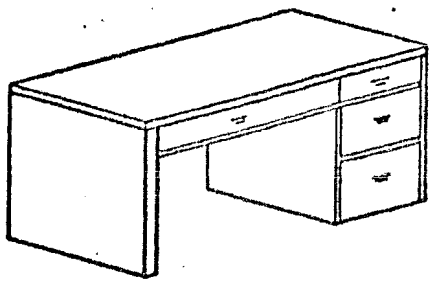
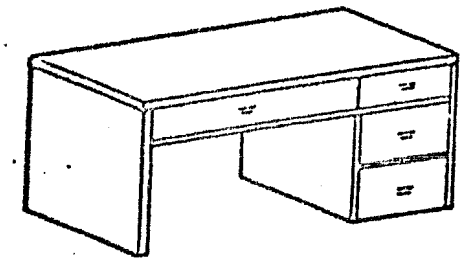


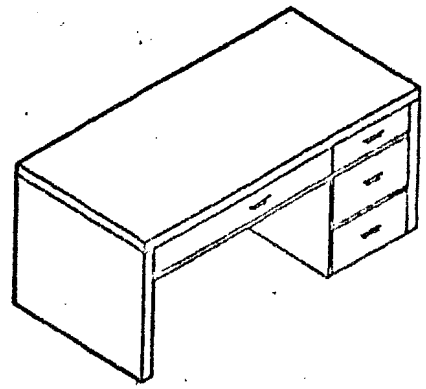
Fig. 4.5 Proyeccion Ortografica Multivisual.



Isometrica

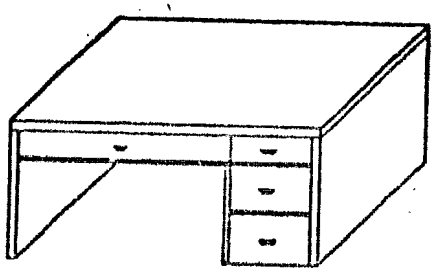


Trimetrica

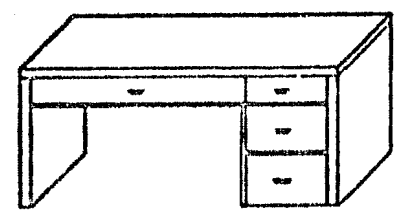


Dimetrica

Fig. 4.6 Efectos de la Proyeccion Ortografica



Cavalier



Cabinet

Fig. 4.7 Efectos de la Proyeccion Oblicua.

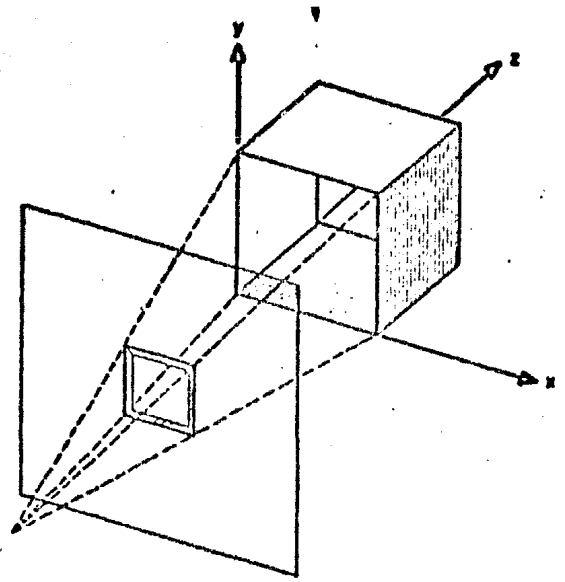


Fig. 4.8 Proyeccion en Perspectiva con un punto de Concurrencia.

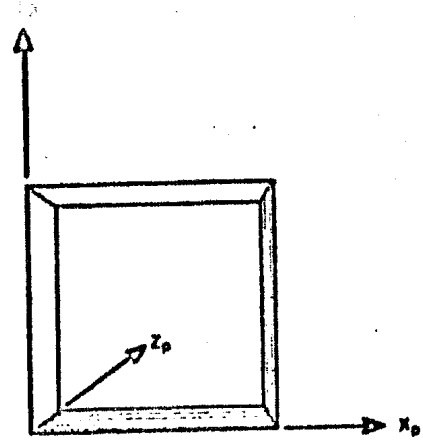


Fig. 4.9 Resultado de la construccion.

Fig. 4.10 Proyección en perspectiva con dos puntos de concurrencia.

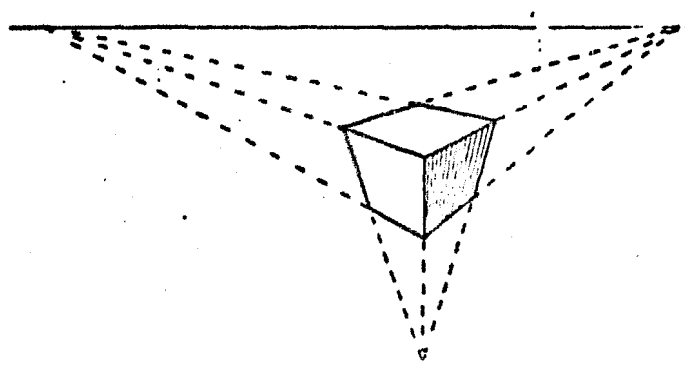


Fig. 4.11 Proyección en Perspectiva con tres puntos de concurrencia.

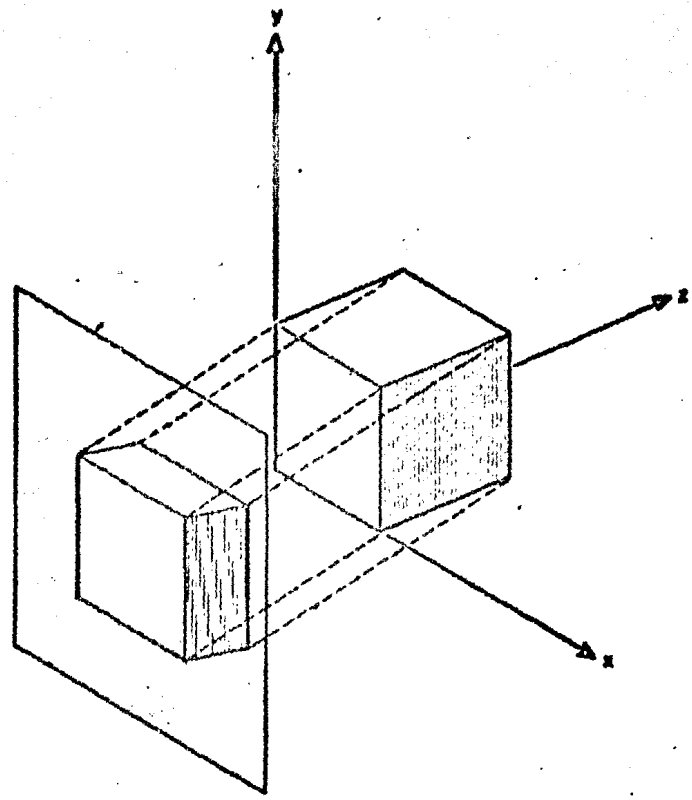


Fig. 4.12 Construccion de una Proyeccion Oblicua.

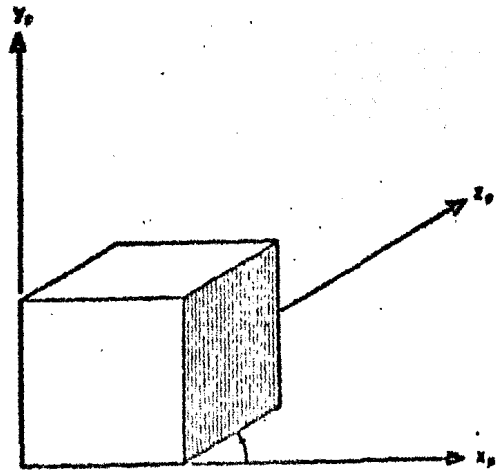


Fig. 4.13 Resultados de la Construccion de una Proyeccion Oblicua.

4.3.4 BASES MATEMATICAS PARA LAS PROYECCIONES GEOMETRICAS PLANAS.

4.3.3.1 Introduccion.

El sistema de coordenadas que se maneja es el sistema izquierdo porque es la manera natural de manejar un sistema de coordenadas en una pantalla de graficacion , con X orientado a la derecha, Y hacia arriba y Z hacia adentro de la pantalla.

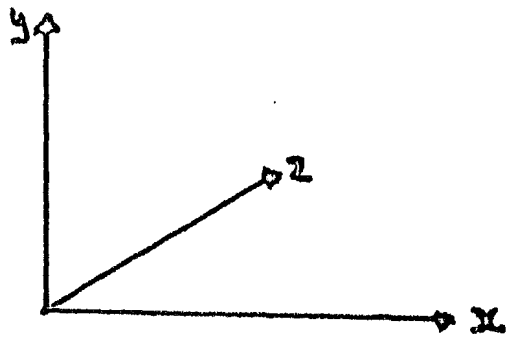


Fig. 4.13 Coordenadas de la Pantalla.

Cada una de las proyecciones pueden ser definidas en una matriz de 4X4 para que coincida con el uso del sistema de coordenadas homogeneas. La conveniencia esta en que esta matriz se puede componer con las matrices de transformacion , haciendose solo dos operaciones (transformar y proyectar) para obtener solo una matriz final.

En esta seccion se empezara con derivar una de las proyecciones mas sencillas , la proyeccion ortografica , desde el punto de vista de transformacion del objeto (ver seccion anterior).

Luego se presentara la proyeccion oblicua , por simplicidad se escogera el plano de proyeccion Z=0.

Enseguida se estudiara la proyeccion en perspectiva escogiendo el plano de proyeccion Z=d y luego se presentara otra manera de obtener una proyeccion en perspectiva con la facilidad de estar cambiando el centro de proyeccion lo cual permite ver la figura desde cualquier lugar que se desee.

La proyeccion estereopar , es una extension de la proyeccion en perspectiva debido a que en la proyeccion en perspectiva se esta considerando la generacion de una sola imagen (para un ojo) . En cambio la proyeccion estereopar se va a generar dos imagenes una para el ojo derecho y la otra imagen para el ojo izquierdo.

De esta forma al concentrarse en cada imagen se

crearа una sensacion de estar viendo al objeto delante de uno.

4.3.3.2 PROYECCION ORTOGRAFICA

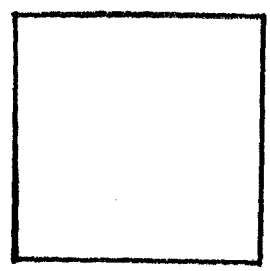
Se habia visto que hay dos maneras de resolver el problema de obtener la vista deseada del objeto : transformar el objeto o escoger planos de proyeccion.

En la proyeccion ortografica se ha escogido la transformacion del objeto.

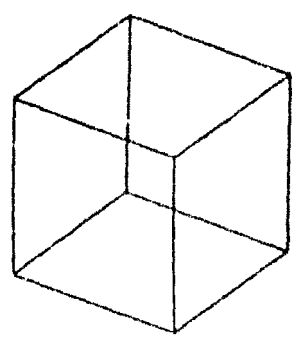
La razon por lo que se escogio esta aproximacion es la necesidad de obtener una imagen que de la sensacion de estar en un espacio tridimensional.

Si no jugamos con la figura ,es decir, si no transformamos al objeto, se pueden obtener figuras sin apariencia de profundidad.

Para ilustrar esto , tomemos un cubo.Si aplicamos proyeccion ortografica , es decir , se hace Z=0 , la imagen que se obtiene es la siguiente:



En cambio si movemos el cubo y aplicamos proyeccion ortografica se obtiene lo siguiente:



La manera mas sencilla de implementar esto es como sigue:

1)Definir el objeto (Estructura de Datos).

2)Transformarlo (translacion , rotacion ,escalamiento).

3)Aplicar la Proyeccion .

1er Paso . Definicion de la Estructura de Datos del Objeto.

Hay muchos objetos que pueden ser definidos de una manera muy sencilla, es suficiente con almacenar los datos que lo definen en arreglos de datos (en FORTRAN por ejemplo, BLOCK DATA).

Esto permite que los datos sean almacenados de una manera muy sencilla.

Cada vertice de la figura esta dado con un unico indice , asi las coordenadas x,y y z del i-esimo vertice esta almacenado en X(I) ,Y(I) y Z(I) respectivamente. Asi se tendra un numero de vetices que va a definir la figura. De esta manera se podra cambiar las figuras sin mucha complicacion.

Tambien hay que tener en cuenta el numero de lineas que forma la figura , la J-esima linea sera almacenada como una pareja de enteros INDICE(1,J) y INDICE(2,J) , los indices de los vertices que forman la i-esima linea.

2o Paso . Definicion de la Transformacion .

Hay que establecer una relacion entre la subrutina que define la figura y otra en la cual se va a definir la transformacion que se desea aplicar a la figura .

Suponsamos que se quiere modificar a la figura con la siguiente transformacion:

a) Rotar los ejes con angulo 0.927295218 con respecto al eje Z.

b)Cambiar el origen al punto (1,0,0).

c) Rotar los ejes un angulo -0.927295218 con respecto al eje Y.

la rutina para calcular la matriz de transformacion seria:

```
      SUBROUTINE TRANSFORMA(P)
      DIMENSION A(4,4),B(4,4),C(4,4),P(4,4)
      PHI=0.927295218
      CALL ROTZ(PHI,A)
      CALL TRANS(1.0,0.,0.,B)
      CALL ROTY(-PHI,C)
      CALL MULT(B,A,R)
      CALL MULT(C,R,P)
      RETURN
      END
```

Cualquier transformacion que se quiera aplicar al objeto hay que definirla antes en la subrutina TRANSFORMA.

3er. Paso. Aplicacion de la Proyeccion.

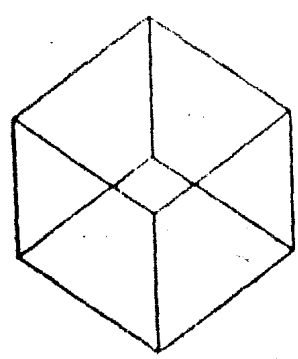
Ya que la coordenada Z es ignorada en una proyeccion ortografica , solo las coordenadas X y Y seran almacenadas en los arreglos XP y YF . ESTos son unidos en el orden correcto , usando la informacion del arreglo INDICE.

Ya que no se necesita calcular la coordenda Z , la transformacion ortografica sera :

```
      DO 1 I=1 ,NOV
      XP(I)=P(1,1)*X(I)+P(1,2)*Y(I)+P(1,3)*Z(I)+P(1,4)
      YP(I)=P(2,1)*X(I)+P(2,2)*Y(I)+P(2,3)*Z(I)+P(2,4)
1     CONTINUE
```

Integrando todo lo anterior en un programa FORTRAN , quedaria:

```
      COMMON /VERTICES/NOV,X(300),Y(300),Z(300),
     1                 XP(300),YP(300)
      COMMON /LINEAS/NOL ,INDICE(2,400)
      CALL DATOS
      CALL TRANSFORMA(P)
      DO 1 I=1,NOV
      XP(I)=P(1,1)*X(I)+P(1,2)*Y(I)+P(1,3)*Z(I)+P(1,4)
      YP(I)=P(2,1)*X(I)+P(2,2)*Y(I)+P(2,3)*Z(I)+P(2,4)
1     CONTINUE
      CALL PLOTS(0,0,1) !PARA CALCOMP
      CALL PLOT(6.5,4.7,-3) !PARA CALCOMP
      DO 2 I=1,NOL
      I1=INDICE(1,I)
      I2=INDICE(2,I)
      CALL PLOT(XP(I1),YP(I1),3)
      CALL PLOT(XP(I2),YP(I2),2)
2     CONTINUE
      CALL EXIT
      END


      SUBROUTINE DATOS
      COMMON /VERTICES/NOV,X(300),Y(300),Z(300),
       1                XP(300),YP(300)
      COMMON /LINEAS/NOL,INDICE(2,400)
      WRITE(5,*)'NUMERO DE VERTICES'
      READ(5,*)'NOV
      WRITE(5,*)NUMERO DE LINEAS'
      READ(5,*)NOL
      RETURN
      END

      BLOCK DATA FIGURA
      COMMON/VERTICES/NOV,X(300),Y(300),Z(300),XP(300),YP(300)
      COMMON/LINEAS/NOL,INDICE(2,400)
C
C     Para el caso de un cubo unitario , se tiene  los
C     siguientes datos
C
      DATA X/1.0,1.0,1.0,1.0,-1.0,-1.0,-1.0,-1.0/
      DATA Y/1.0,1.0,-1.0,-1.0,1.0,1.0,-1.0,-1.0/
      DATA Z/1.0,-1.0,1.0,-1.0,1.0,-1.0,1.0,-1.0/
      DATA INDICE/1,2,2,4,4,3,3,1,5,6,6,8,8,7,7,5
                  1,5,2,6,3,7,4,8/
```



Si se quiere manejar la proyeccion ortografica es una matriz, se obtiene de la siguiente manera:

Para el plano de proyeccion Z=0 , la direccion de la proyeccion es la misma que la normal al plano de proyeccion , es decir, el eje Z en este caso.

Entonces el punto P se proyecta como :

$X_P = X$ $\qquad$ $Y_P = Y$ $\qquad$ $Z_P = 0$

Esta proyeccion es expresada en la matriz como :

$$M_{ort} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.3.3.3 PROYECCION OBLICUA.

La proyeccion oblicua estara escrita en terminos de α y l como se muestra en la figura 4.14 , la cual es un cubo unitario proyectado sobre el plano XY (Z=0) . De la figura puede verse que el punto P(0,0,1) se proyecta sobre P'(lcosα , lsenα ,0) sobre el plano XY.

Por definicion, esto quiere decir que la direccion de la proyeccion es la que pasa a travez de los dos puntos $\overline{PP'}$, como se ve en la figura 4.15 . Esta direccion es P' - P = (l cosα , l senα ,-1).

La direccion de la proyeccion forma un angulo β con el plano XY.

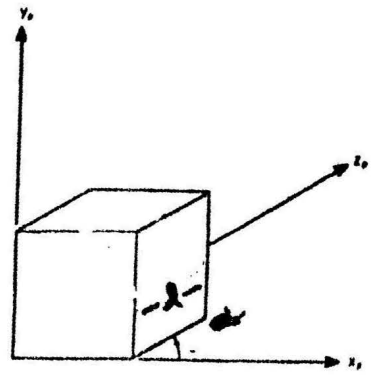


Fig. 4.14 Proyeccion Oblicua del Cubo Unitario.

Hay que considerar ahora un punto x,y,z en general.El objetivo es saber cual es la proyeccion oblicua de (Xp,Yp) sobre el plano XY.

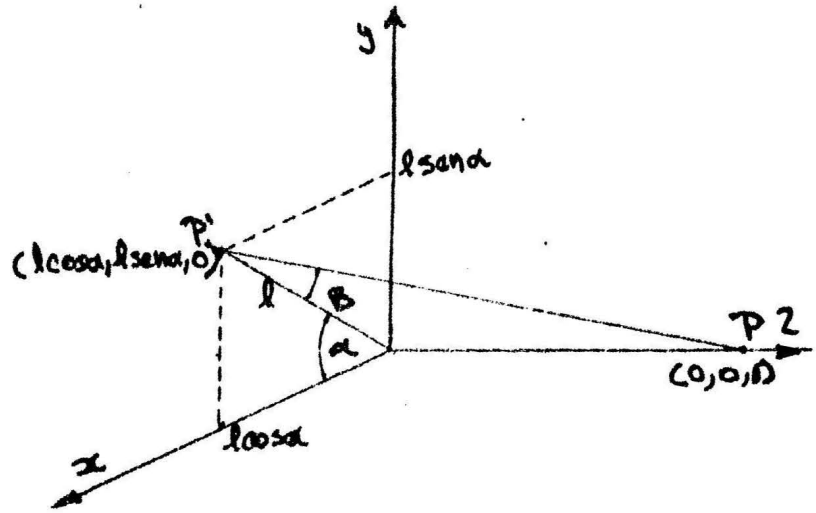


Fig. 4.15

Volviendo a la figura 4.14 se observa lo siguiente:

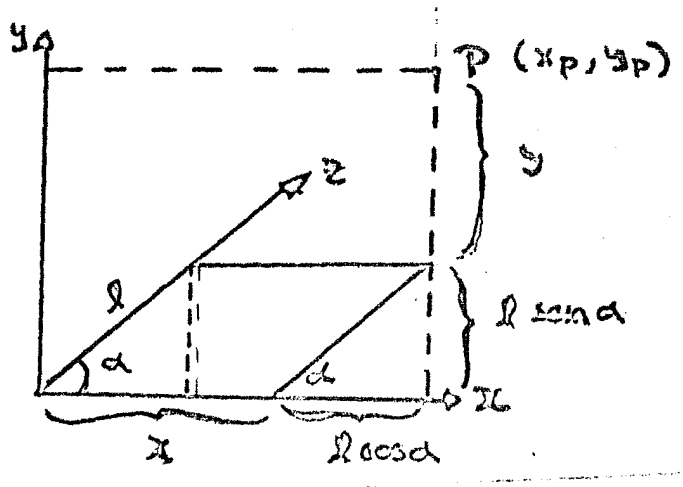


Fig. 4.16

Por trigonometria

$$\cos\alpha = \text{cateto adyacente} / \text{hipotenusa}$$

$$\text{sen}\,\alpha = \text{cateto opuesto} / \text{hipotenusa}$$

$$\therefore \quad \text{cateto adyacente} = z * l\cos\alpha$$

$$\text{cateto opuesto} = z * l\,\text{sen}\,\alpha$$

de esto se deduce que

$$x_p = x + z * l\cos\alpha$$

$$y_p = y + z * l\,\text{sen}\,\alpha$$

La matriz homogenea que realiza estas operaciones y representa la proyeccion oblicua :

$$M_{ob} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ l\cos\alpha & l\,\text{sen}\alpha & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

El efecto de esto es cortar al objeto y proyectarlo : los planos de constante $Z=Z_1$ con transladados en x por $z*l\cos\alpha$ y en y por $z*l\sin\alpha$ y entonces proyectados en Z=0.

Esta transformacion conserva las lineas paralelas y conserva angulos y distancias en los planos paralelos al eJe Z.

Para una proyeccion Cavalier l=1 , asi que el angulo β va a ser de 45 grados .

Para la proyeccion Cabinet l=1/2 y β es arctan(2) , o bien 63.4 grados .

Para una proyeccion ortografica l=0 y β =90 grados.

4.3.3.4 PROYECCION EN PERSPECTIVA (1er. Metodo)

Para obtener una proyeccion en Perspectiva de una manera muy sencilla , asumimos que el plano de proyeccion es normal al eje Z en Z=d.

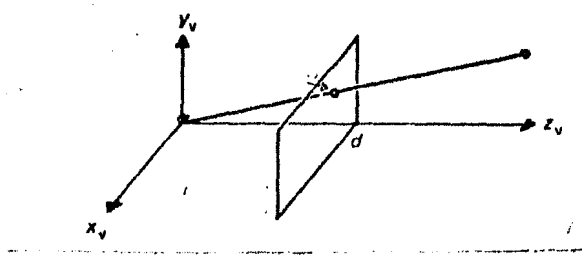


Fig. 4.17 Proyeccion en Perspectiva.

Para calcular la proyeccion P(x,y,z) con coordenadas Xp y Yp , se hace uso de triangulos semejantes (Fig. 18)para obtenerlas siguientes razones:

$$\frac{x_p}{d} = \frac{x}{d} \qquad \frac{y_p}{d} = \frac{y}{2}$$

Multiplicando cada lado por d se tiene:

$$x_p = \frac{x \cdot d}{2} \qquad y_p = \frac{y \cdot d}{2}$$

$$x_p = \frac{x}{2/d} \qquad y_p = \frac{y}{2/d}$$

La distancia d es Justamente el factor de escala aplicado a Xp y Yp.

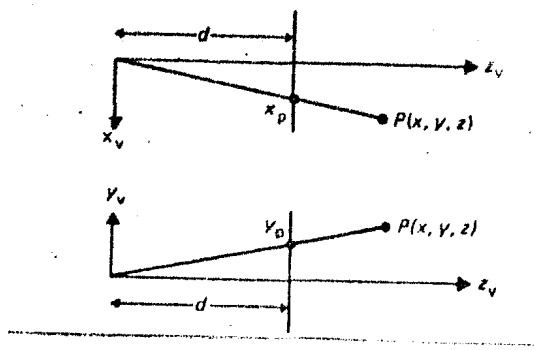


Fig. 4.18 Proyeccion en Perspectiva.

La division por Z hace que la proyeccion en perspectiva de objetos mas distantes se vean mas pequeños que los objetos mas cercanos.

Hay que notar que todos los valores son posibles excepto Z=0.

La matriz homogenea puede ser expresada como:

$$M_{par} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & d & 1/d \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Otra alternativa para la proyeccion en perspectiva es colocar el plano de proyeccion en Z=0 y el centro de proyeccion en Z=-d.

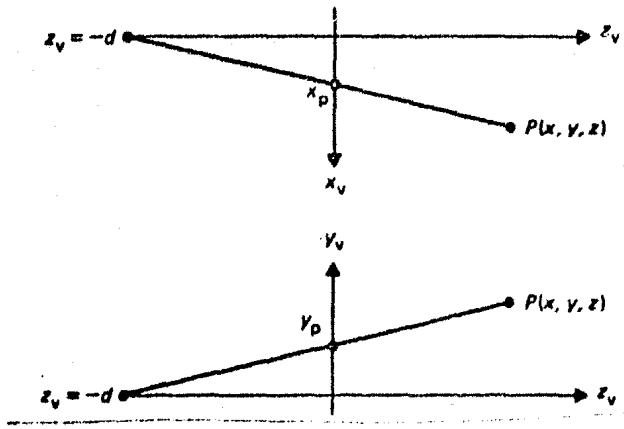


Fig 4.19 Proyeccion en Perspectiva
con otro Plano de Proyeccion.

por triangulos semejantes:

$$\frac{y_p}{d} = \frac{y}{z+d} \quad , \quad \frac{y_p}{d} = \frac{x}{d+z}$$

$$y_p = d \cdot \frac{y}{z+d} \quad , \quad x_p = d \cdot \frac{x}{d+z}$$

$$y_p = \frac{y}{z/d+1} \quad , \quad x_p = \frac{x}{z/d+1}$$

La matriz homogenea sera:

$$M'_{par} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1/d \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

PROYECCION EN PERSPECTIVA (2o. Metodo).

En las proyecciones anteriormente expuestas se ha puesto varias restricciones , el centro de proyeccion , el plano de proyeccion,etc.

Enseguida se presenta un metodo para obtener una proyeccion en Perspectiva con un centro de proyeccion arbitrario , permitiendo de esta manera observar al objeto desde donde se desee.

PROYECCION EN PERSPECTIVA.

La proyeccion plana es la mejor representacion de los objetos reales cuando son vistos atravez de un medio homogeneo en perspectiva.

Este tipo de proyeccion marca un punto arbitrario P en el espacio en un punto P' sobre el plano tal que todas las rectas (proyectores) PP' se intersectan en un punto comun C. El punto C es llamado el centro de la proyeccion y es orientado de tal manera que su normal es paralela a la linea de vision.

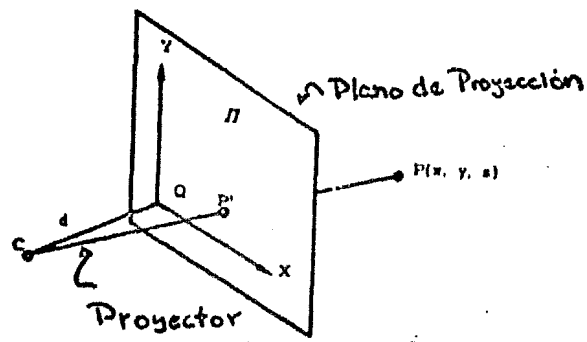


Fig. 4.20 Proyeccion en Perspectiva.

Sea la funcion ha ser graficada, llamemosle $f(x,y)$, dada en terminos de un conjunto de coordenadas rectangulares donde

$$z = f(x,y) \quad \ldots (1)$$

Supongamos que el ojo del observador esta situado en el punto C con coordenadas (C_x, C_y, C_z) referidos al sistema de coordenadas, sun mas, para que la linea de vision hasa angulos α, β, γ con los ejes x, y y z respectivamente.

Sea d, la distancia dada, definimos a $Q(q_x, q_y, q_z)$ como un punto tal que $\overline{CQ}$ es la direccion de vision y $|\overline{CQ}| = d$

El plano de proyeccion Π es construido atravez de Q y normal a $\overline{CQ}$.

La recta que va de C a un punto arbitrario del espacio, intersectara a $P(x,y,z)$ en algun punto $P'(x', y', z')$. P' es la imagen en perspectiva de P en Π con respecto a C.

De la figura 4.20 se tiene

$$q_x = c_x + d(\cos\alpha)$$
$$q_y = c_y + d(\cos\beta) \quad \ldots (2)$$
$$q_z = c_z + d(\cos\gamma)$$

y $$\frac{x' - c_x}{x - c_x} = \frac{y' - c_y}{y - c_y} = \frac{z' - c_z}{z - c_z} = K \quad \ldots (3)$$

Por definicion

$$\overline{CQ} = d(\cos\alpha i + \cos\beta j + \cos\gamma k) \quad \ldots (4)$$

Ya que P' se encuentra en el plano Π debe satisfacer que

$$\overline{CQ} \cdot \overline{QP'} = 0 \quad \ldots (5)$$

ya que son ortogonales

ya que

$$\overline{CQ} \cdot \overline{QP} = (\overline{CQ} + \overline{QP}) = 0$$

$$\overline{CQ} \cdot \overline{QZ} + \overline{CQ} \cdot \overline{CP'} = 0$$

$$\Rightarrow \overline{CQ} \cdot \overline{QZ} = \overline{CQ} \cdot \overline{CP'}$$

de la igualdad (5) se tiene

$$\overline{CQ} \cdot \overline{CP} = \overline{CQ} \cdot \overline{CA}$$

$$d(\cos\alpha, \cos\beta, \cos\gamma)(x'-c_x, y'-c_y, z'-c_z)$$

$$= d(\cos\alpha, \cos\beta, \cos\gamma) \cdot d(\cos\alpha, \cos\beta, \cos\gamma)$$

$$= d^2((x'-c_x)\cos\alpha + (y'-c_y)\cos\beta + (z'-c_z)\cos\gamma)$$

$$= d^2(\cos^2\alpha + \cos^2\beta + \cos^2\gamma)$$

$$(x'-c_x)\cos\alpha + (y'-c_y)\cos\beta + (z'-c_z)\cos\gamma = d \quad \cdots(6)$$

Sustituyendo la ecuacion (3) en la ecuacion (6) se obtiene (para encontrar el valor de K):

$$K = d \,/\, [(x-c_x)\cos\alpha + (y-c_y)\cos\beta + (z-c_z)\cos\gamma] \quad \cdots(7)$$

La ecuacion (3) con la ecuacion (2) nos da la definicion de P'

$$x' = c_x + K(x-c_x) \qquad y' = c_y + K(y-c_y) \qquad z' = c_z + K(z-c_z)$$

Resta expresar a P' en terminos de un sistema

bidimensional en Π.
La ecuacion del plano de proyeccion (ecuacion (5))
puede ser escrita como

$$(x' - q_x)\cos\alpha + (y' - q_y)\cos\beta + (z' - q_z)\cos\gamma = 0 \quad \ldots (9)$$

La recta formada por la interseccion del plano
horizontal $\xi' - q_z = 0$
y Π el plano de proyeccion es la siguiente:

$$\frac{x' - q_x}{\cos\beta} = \frac{y' - q_y}{-\cos\alpha} = \frac{z - q_z}{0} \quad \ldots (10)$$

Uno de los nuevos ejes de coordenados en Π (el eje X)
sera definido a lo larso de esta recta.
El vector unitario, en direccion del eje positivo X,

en terminos de las coordenadas originales , esta dado
por:

$$V_x = S_1 [(\cos\beta)i - (\cos\alpha)j] / \operatorname{sen}\gamma \quad \ldots (11)$$

donde $S_1 = \pm 1$. Para que se pueda definir el otro
eje coordenado en Π (el eje Y) un vector unitario , V_y

en Π se debe encontrar que

$$V_x \cdot V_y = 0 \quad \ldots (12)$$

Sea

$$V_y = ai + bj + ck \quad \ldots (13)$$

La ecuacion (12) requiere que

$$a(\cos\beta) - b(\cos\alpha) = 0 \quad \ldots (14)$$

Como V_y esta en Π , este debe ser normal a $\overline{eq}$.
Asi

$$a(\cos\alpha) + b(\cos\beta) + c(\cos\gamma) = 0 \quad \ldots (15)$$

Aun mas , como U_y es un vector unitario

$$a^2 + b^2 + c^2 = 1 \qquad \ldots (16)$$

La solucion de las ecuaciones simultaneas (14), (15), (16)

$$a = S_2 \cos\alpha \cos\gamma / \operatorname{sen}\gamma$$

$$b = S_2 \cos\beta \cos\gamma / \operatorname{sen}\gamma \qquad \ldots (17)$$

$$c = -S_2 \operatorname{sen}\gamma$$

con $S_2 = \pm 1$

Los signos de S_1 y S_2 dependera de las direcciones d los ejes positivos x y y .
Si el observador esta orientado de tal manera que est colocado sobre el eje $\hat{z}$ y la convecciom usual es seguida (es decir, el eje X a la derecha , el eje Y hacia arriba) U_y debe tener una componente positiva en direccion de $\hat{z}$.
Entonces

$$U_y \cdot \hat{z} > 0 \qquad \ldots (18)$$

Aun mas $U_x \cdot U_y$ debe ser paralelo a la linea de vision y debe señalar el lado donde esta el observador con respecto a Π .

$$U_x \cdot U_y = -\frac{1}{d} \cdot \overline{CQ} \qquad \ldots (19)$$

Sustituyendo las ecuaciones (4), (11), (13) y (17) en las relaciones (18) y (19) obtenemos

$$S_2 \operatorname{sen}\gamma < 0 \qquad S_1 S_2 = -1$$

Como $0 < \gamma < \pi$, $\operatorname{sen}\gamma > 0$ y $S_1 = 1$, $S_2 = -1$

Si el nuevo sistema de coordenadas tiene su origen en Q, las componentes de P' en terminos de los nuevos ejes sera $(\overrightarrow{QP'} \cdot U_x , \overrightarrow{QP'} \cdot U_y)$

Se sigue que

$$x = [-(x' - q_x)\cos\beta - (y' - q_y)\cos\alpha] / \operatorname{sen}\gamma$$

$$y = (z' - q_z) / \operatorname{sen}\gamma \qquad \ldots (20)$$

Resumiendo : Dado el centro de proyeccion , $C(c_x, c_y, c_z)$ la direccion de vision , $(\cos\alpha, \cos\beta, \cos\gamma)$, la distancia del plano de proyeccion d , encontrar la proyeccion de un punto en el espacio , $P(x,y,z)$, se calcula

$$q_x = c_x + d(\cos\alpha) \quad q_y = c_y + d\cos\beta) \quad q_z = c_z + d(\cos\gamma)$$

$$k = d / [(x - c_x)\cos\alpha + (y - c_y)\cos\beta + (z - c_z)\cos\gamma]$$

$$x' = c_x + k(x - c_x) \quad , y' = c_y + k(y - c_y) \quad , z' = c_z + k(z - c_z)$$

$$x = [(x' - q_x)\cos\beta - (y' - q_y)\cos\alpha] / \operatorname{sen}\gamma$$

$$y = (z' - q_z) / \operatorname{sen}\gamma$$

Se requiere que $\operatorname{sen}\gamma \neq 0$,esta singularidad puede ocurrir porque los dos planos cuya interseccion define el eje-X son identicos.
En este caso se requiere que la linea de vision sea vertical, (es decir, $\operatorname{sen}\gamma = 0$) el eje -X puede ser definido como la interseccion de π con el plano vertical $y' - q_y = 0$

asi la transformacion en perspectiva es

$$x = [-(x' - q_x)\cos\gamma + (z' - q_z)\cos\alpha] / \operatorname{sen}\beta$$

$$y = (y - q_y) / \operatorname{sen}\beta$$

para el caso de que $\operatorname{sen}\gamma = 0$:

4.3.3.5 PROYECCION ESTEREOPAR.

La proyeccion estereopar puede producirse automaticamente solo usando la proyeccion en perspectiva.

Dando lo que podria llamarse posicion de la nariz (Xn,Yn,Zn) , y un angulo entre las lineas que conectan el origen y la posicion de los ojos , la posicion deseada de los ojos son :

$$X_{ojo} = X_n \cos(\Theta) \pm y_n \operatorname{sen}(\theta) Q$$

$$Y_{ojo} = y_n \cos(\Theta) \mp X_n \operatorname{sen}(\Theta) Q$$

$$Z_{ojo} = Z_n \cos(\Theta)$$

donde

$$Q = \sqrt{X_n^2 + y_n^2 + Z_n^2} \;\Big/\; \sqrt{X_n^2 + y_n^2}$$

Los signos van a indicar para que ojo se va a formar la imagen. Los signos de arriba son del ojo izquierdo , y los signos de abajo son para el ojo derecho.

C A P I T U L O V

CAPITULO V

ELIMINACION DE LINEAS Y SUPERFICIES OCULTAS.

5.1 INTRODUCCION.
5.2 ESPACIO DEL OBJETO Y ESPACIO DE LA IMAGEN.
5.3 ALGORITMOS DE PROFUNDIDAD DE MEMORIA.
5.4 ANALISIS GEOMETRICO.
5.5 ALGORITMOS DE COHERENCIA DE LINEA DE BARRIDO.
5.6 ALGORITMO DE COHERENCIA DE AEREA.
5.7 ALGORITMO DE PRIORIDAD.
5.8 LA ELECCION DE UN ALGORITMO.
5.9 ALGORITMO DE LINEAS OCULTAS USADO EN EL PAQUETE.

CAPITULO V

ELIMINACION DE LINEAS Y SUPERFICIES OCULTAS.

5.1 INTRODUCCION.

Uno de los problemas mas interesantes dentro de la graficacion por computadora es la de eliminar aquellas partes de los cuerpos solidos que permanecen ocultas al observador.

En la vida real los objetos opacos obstruyen los rayos de luz que vienen de las partes ocultas y nos impiden verlas.En la generacion de una imagen por computadora ,esa eliminacion automatica no tiene lugar cuando los objetos son proyectados en el sistema de coordenadas del dispositivo de graficacion. En vez de esto , todas las partes del objeto aparecen en la pantalla,aun aquellas que deberian ser invisibles.Para eliminar estas partes invisibles y tener asi una imagen mas realista debemos aplicar un algoritmo de "lineas ocultas" o de "superficies ocultas" a todo el conjunto de objetos.

En los años 60's ,cuando el primero de estos algoritmos fue desarrollado, unicamente existian dispositivos de "dibujo de lineas" sobre los cuales se dirigio el esfuerzo de eliminar lineas que deberian estar ocultas.Los primeros de tales algoritmos ,de los cuales el de Robert fue el mas prominente,eran terriblemente lentos. Cuando los dispositivos tipo "raster" hicieron su aparicion , la atencion se dirigio en la direccion de la eliminacion de superficies ocultas y muchas tecnicas, en harduare y software fueron rapidamente desarrolladas.Recientemente el desarrollo de microprocesadores para eliminacion de superficies ocultas permite generar imagenes realistas a una razon de 30 imagenes por segundo.

A pesar de la gran cantidad de algoritmos existentes aun no existe una respuesta simple al problema de la eliminacion de superficies o lineas ocultas , el mejor algoritmo no ha sido creado aun.Muchas de las diferencias entre los algoritmos vienen de los distintos requerimientos :los algoritmos operan en tipos diferentes de modelados de escenas,generan diferentes formas de salida o se las tienen que ver con imagenes de muy diversa complejidad.Un algoritmo diseñado para producir imagenes de tiempo real tiene , o tendra objetivos muy diferentes de un algoritmo diseñado para presentar imagenes muy realistas con superficies sombreadas.

A pesar de la enorme variedad de los detalles de su diseño entre algoritmos comparten muchas caracteristicas.Todos ellos utilizan alguna forma de

ordenamiento geometrico para distinguir las partes del objeto que son visibles de aquellas partes que no lo son.De igual manera que el ordenamiento alfabetico se utiliza para diferenciar palabras que estan al principio del abecedario de aquellas que se encuentran al final,el ordenamiento geometrico separa las partes del objeto que estan mas cerca del observador de las partes mas alejadas. El ordenamiento geometrico es mucho mas complicado que el ordenamiento alfabetico debido a que los objetos mas complejos no siempre tiene un ordenamiento sencillo.

Los algoritmos de lineas ocultas y superficies ocultas utilizan de alguna forma las propiedades de coherencia para reducir el tiempo de calculo requerido para generar una imagen.Las diferentes formas de coherencia estan relacionadas con el orden o la regularidad de la imagen.La coherencia de linea de barrido por ejemplo,se presenta porque el despliegue de una linea de barrido en una imagen de rastreo es muy similar al despliegue de la linea anterior.La coherencia de cuadro , en una secuencia de imagenes diseñadas para aparentar movimiento(animacion) reconoce que cuadros sucesivos de una imagen son muy similares.La coherencia de objeto resulta de las relaciones entre partes distintas de un mismo objeto.

Los algoritmos de superficies ocultas estan diseñadas para explotar una o mas de estas propiedades de coherencia para incrementar su eficiencia.Los algoritmos de superficies ocultas diseñados para producir imagenes desplegandolas barriendo la pantalla de graficacion comparten una gran similitud con los algoritmos de conversion de barrido en dos dimensiones.Ambos algoritmos ponen enfasis en el ordenamiento y en las propiedades de coherencia, aun mas, veremos mas adelante que la conversion de barrido en dos dimensiones tiene un papel muy importante en muchos de los algoritmos tridimensionales.

5.2 ESPACIO DEL OBJETO Y ESPACIO DE LA IMAGEN.

Los algoritmos para la eliminacion de superficies ocultas trabajan ya sea con el espacio del objeto o con el espacio de la imagen. Un algoritmo sobre el espacio del objeto se concentra en las relaciones geometricas entre los objetos de la escena para determinar que parte de que objeto es visible desde el punto de vista del observador ; un algoritmo sobre el espacio de la imagen se concentra en la imagen final y se pregunta que es visible dentro de cada pixel al rastrear la imagen.

Un algoritmo sobre el espacio del objeto (AEO) realiza calculos geometricos con tanta precision como la precision maxima permita el "hardware" de punto flotante de la computadora en que se esta trabajando¦dado que la precision de la solucion es mucho mas alta que la del dispositivo de salida, la imagen puede ser desplegada con una gran amplificacion sin perder exactitud. En contraste, los algoritmos sobre el espacio de la imagen (AEI) realizan calculos con una precision apenas suficiente para ajustar a la resolucion del dispositivo de salida usado para presentar la imagen , estos algoritmos simplemente calculan la intensidad para cada uno de los puntos en el dispositivo de salida .

Esta formas de atacar el problema generalmente dan por resultado caracteristicas de realizacion diferentes, el tiempo de maquina para un AEO tiende a crecer con el numero de objetos en escena , visible o no , mientras que para los AEI tiende a crecer con la complejidad de las partes visibles de la imagen. Generalmente el costo de los AEI crece mas lentamente que el costo de los AEO conforme la complejidad de la escena crece. Muchos de los algoritmos descritos en este capitulo son AEI ya que estos pueden aplicarse tanto a superficies como a lineas ocultas.Los metodos relativos al espacio del objeto de usan principalmente en algoritmos de lineas ocultas y se encuentran explicados en el articulo de Sutherland ,Sproull y Schumacker [SUTH].

5.3 ALGORITMO DE PROFUNDIDAD EN MEMORIA.

De todos los algoritmos del espacio de la Imagen , el mas simple es el algoritmo de "profundidad de memoria" (depth-buffer algorithm) .Para cada pixel en la pantalla , guardamos un registro de la profundidad del objeto dentro del pixel que cae mas cerca del observador , ademas de la profundidad , tambien guardamos la intensidad que deberia ser desplegada para mostrar el objeto.En este sentido la memoria de profundidad es una extension de la memoria de cuadro ("frame buffer").

Este algoritmo , que se explica mas abajo , requiere de dos arreglos: INTENSIDAD y PROFUNDIDAD cada uno de los cuales es indicado por las coordenadas del pixel (X,Y).

El algoritmo , paso por paso:

a)Para todos los pixels de la pantalla hacer PROF(X,Y)=1.0

b)Idem para INT(X,Y)=al valor mas bajo.

c)Para cada polisono de la escena , encontrar todos los pixels(X,Y) que caen dentro de las fronteras del polisono proyectado en la pantalla.

Para cada uno de estos pixels:

1) Calcular la profundidad (Z) del polisono en (X,Y).

2) Si Z<PROF(X,Y) , este polisono esta mas cerca del observador que otros ya checados en este pixel. En este caso poner PROF(X,Y)=Z e INT(X,Y)=al valor correspondiente al sombreado del polisono.Si Z>PROF(X,Y) , algun polisono checado antes esta mas cerca del observador que este nuevo polisono y ninguna accion se lleva a cabo.

Despues que todos los polisonos han sido checados , el arreglo INT contiene la solucion buscada. Este algoritmo muestra algunas de las caracteristicas comunes a los alosoritmos de superficies ocultas:primero , requiere la presentacion de todas las superficies opacas de la escena ,polisonos en este caso.Estos polisonos pueden ser las caras de un poliedro grabado en el modelo de la escena o simplemente puede representar hojas opacas y delgadas en la escena , cualquier modelo que permita la enumeracion de los puntos (X,Y) en las superficies

opacas servira tambien para este algoritmo de profundidad de la memoria.

La segunda caracteristica importante de este algoritmo es su uso del sistema de coordenadas de la pantalla , despues de los paso (a) y (b) todos los poligonos de la escena son transformados al sistema de coordenadas de la pantalla usando una matriz para las transformaciones basicas para modificar el objeto y para la perspectiva.

Los poligonos deben ser recortados tambien para asegurarse que todos ellos caen completamente dentro de la caja de visualizacion ,ademas debe incluirse el recorte por profundidad porque permite escoger representacion de punto fijo para que el arreglo PROF tenga garantizado el acomodamiento de cualquier posible valor.Despues del recorte se hace una transformacion del puerto de vision para mapear (X,Y) dentro del sistema de coordenadas de la pantalla.

Resumiendo , este algoritmo opera sobre los poligonos que han sido convertidos a coordenadas de la pantalla.Dos propiedades importantes de este sistema coordenado son explotadas:Primero ,debido a que la perspectiva de cualquier objeto es una proyeccion ortografica de las coordenadas de pantalla ,enumerando las coordenadas de los pixels (X,Y) que caen dentro de las fronteras del poligono proyectado es precisamente la conversion de barrido (la cual a veces es llamada "tiling").Las coordenadas (X,Y) de pantalla de los vertices del poligono son pasadas directamente a un algoritmo de conversion de barrido para realizar el paso (c) de este algoritmo. Segundo,que los calculos de la profundidad Z de un poligono en un punto arbitrario (X,Y) se ven simplificados debido a que podemos guardar la ecuacion del plano de cada poligono en el sistema de coordenadas de la pantalla.Esta ecuacion puede resolverse para cada (X,Y) encontrado o puede ser evaluada utilizando los metodos incrementales.

Este algoritmo a pesar de ser tan simple no es practico debido a que los arreglos PROF e INT pueden llegar a ser enormes.Para generar una imagen con un rastreo de 500 X 500 pixels se requieren alrededor de 250,000 localidades de memoria para cada arreglo ,aun cuando la memoria del "frame buffer" puede proporcionar la memoria para el arreglo INIT , el arreglo PROF permanece muy grande.Para reducir la cantidad de memoria requerida , la imagen puede ser dividida en pequeñas imagenes , aplicando el algoritmo a cada una por separado. Por ejemplo, el rastreo original de 500 X 500 puede ser dividido en 100 rastreos de 50 X 50 pixels cada uno , el procesamiento de cada uno de estos rastreos requiere unicamente 2,500 localidades de memoria pero esto mismo obliga a que el tiempo de

procesamiento crezca debido a que cada polisono es procesado muchas veces .Alternativamente ,se pueden usar 100 bandas horizontales cada una con 5 lineas de barrido de alto y 500 pixels de ancho.Extendiendo este concepto al limite podemos implementar al algoritmo para cada linea de barrido y procesar entonces 500 lineas de barrido separadamente.Por otra parte subdividir la pantalla no siempre incrementa el tiempo de procesado¿aun mas puede ayudar a reducir el trabajo requerido para generar una imagen. Esta reduccion tiene lugar debido a las propiedades de coherencia entre las pequeñas partes de la pantalla.Si se hacen los calculos de estas pequeñas regiones en un cierto orden , y no aleatoriamente ,los ahorros en tiempo y en espacio , son considerables.

5.4 ANALISIS GEOMETRICO.

Muchas de las propiedades de coherencia (o simplemente coherencias) requieren un analisis geometrico mas elaborado que el que le da el algoritmo de profundidad de memoria para ser explotadas a su maxima eficiencia;muchas tecnicas de superficies ocultas ,por ejemplo necesitan comparar dos polisonos para decidir cual obscurece a cual .

Estos calculos pueden ser consumidores de tiempo a menos que sean hechos con cuidado.

ECUACIONES DEL PLANO.

Nosostros usamos la ecuacion del plano del polisono en el sistema de coordenadas de la pantalla para calcular la profundidad de un polisono a cierto valores (X,Y) (en la pantalla).

Utilizando las conversiones usuales expresamos la ecuacion como

$$ax + by + cz + d = 0$$

,y adoptamos la conversion de que

$$ax + by + cz + d > 0$$

si un punto cae fuera del objeto del cual el plano es una cara. Esta ecuacion del plano en el sistema de coordenadas de la pantalla puede ser encontrada a partir de la ecuacion del plano en las coordenadas del objeto en una forma muy simple.

La ecuacion del plano se usa para identificar las caras traseras , i.e., los polisonos que no es posible ver debido a que caen en el lado del objeto que esta mas lejano al punto de vista del observador.Una cara trasera se identifica cuando se obtiene un valor negativo al sustituir las coordenadas del punto de vision (0 , 0 ,$-\infty$) en la ecuacion del plano (probar que si c>0 es por lo tanto equivalente).La eliminacion de las caras traseras , lo cual es una practica comun utilizando la propiedad de coherencia del objeto, reduce a la mitad el numero de polisonos que hay que considerar durante la generacion de la imagen con superficies ocultas. Hay casos en los que la eliminacion de las caras traseras resuelve el problema de las superficies ocultas , por ejemplo si tenemos un poliedro convexo , eliminando las caras traseras eliminamos igualmente todas las superficies ocultas.Esto es un ejemplo de como la coherencia de objeto hace particularmente faciles los calculos de poliedros convexos.

PRUEBAS DE ENCIMAMIENTO.

Si dos polisonos no se enciman en X y en Y, no pueden obscurecerse uno al otro de ninguna manera ; un metodo rapido para checar el encimamiento reduce grandemente el numero de polisonos que hay que checar con respecto a otros.El mejor ejemplo son las pruebas de tipo minimax; si la coordenada menor en X de un polisono es mayor que la coordenada mayor en X de otro ,entonces es posible que no se encime; el mismo argumento se aplica a Y (y si se esta considerano el encimamiento de profundidad , a Z).La prueba minimax bidimensional XY es llamada usualmente de "caja frontera".

Si la prueba minimax falla en mostrar la separacion aun puede haber la posibilidad de que no se encimen los polisonos considerados; en estos casos, una prueba mas detallada debe ser aplicada para determinar cual (o cuales) de los polisonos intersectan con cual otro y como. Por ejemplo cada lado del polisono puede ser comparado con cada uno de los lados del otro polisono para determinar si hay interseccion y donde.

El uso de las pruebas minimax sobre los lados puede ayudar a apurar este proceso determinando rapidamente cuando dos lados no puden intersectarse. Ademas cuando el poliedro es muy grande la clasificacion puede ser util.

POLIGONOS QUE RODEAN.

Algunos algoritmos de coherencia de aerea dependen de la prueba de si un polisono rodea completamente o no una ventana rectangular de la pantalla ;si las cuatro esquinas de la ventana estan dentro del polisono entonces toda la ventana se encuentra rodeada por el.Podemos determinar de una manera sencilla cuando las coordenadas de la esquina (X,Y) y nos movemos hacia el punto (-^ ,Y) y contamos el numero de veces que el trazo corta al polisono ; si el numero de cortes es par estamos dentro y si es impar estamos fuera .Otra forma de checar si esta adentro o no es sumando los angulos subtendidos por los lados y checar ; si es cero estamos fuera del polisono y si es 2T estamos dentro. Esta prueba es mas facil de hacer ya que los angulos pueden ser calculados con una precision sumamente baja (2 bits) sin perder mucha precision en el algoritmo.

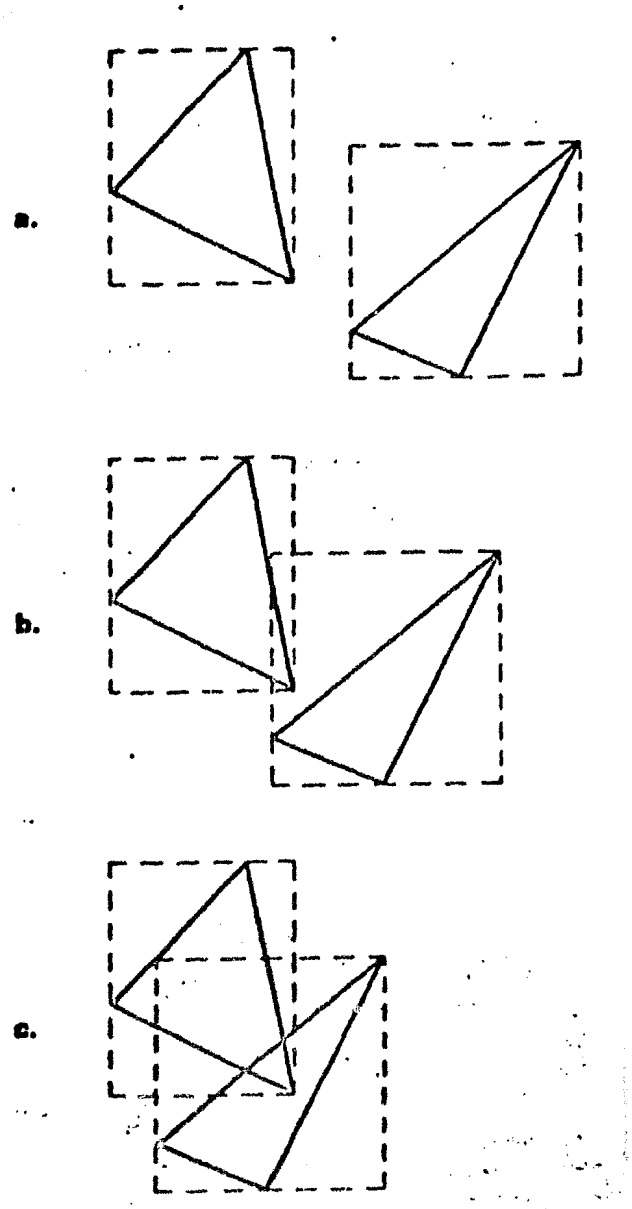


Fig. 5.1 Prueba Minimax.

a) Los polígonos no se enciman.
b) Los lados no pueden intersectarse.
c) Hay encimamiento.

5.5 ALGORITMOS DE COHERENCIA DE LINEA DE BARRIDO.

Los algoritmos de linea de barrido resuelven el problema de las superficies ocultas para cada linea de barrido , usualmente procesando las lineas de arriba para abajo de la pantalla. El algoritmo examina sucesivamente una secuencia de ventanas en la pantalla ;cada ventana mide una linea de barrido de alto y es tan ancha como la pantalla .El mas simple de los algortimos de linea de barrido es la version unidimensional del algoritmo de profundidad de memoria :se requiere nuevamente dos arreglos PROF(X) e INT(X) para guardar los valores de cada linea de barrido.

Descripcion del Algoritmo:

a)Para cada pixel en la linea se barrido hacemos PROF(X)=1.0

b)Para cada pixel hacemos INT(X)=al valor mas bajo.

c)Para cada polisono en la escena encuentra todos los pixels en la linea de barrido Y que caen dentro del polisono ; para cada uno de estos vectores X:

1)Calcula la profundidad Z del polisono en (X,Y).

2)Si Z < PROF(X) tiene PROF(X)=Z e INT(X)=al valor de la intensidad correspondiente al sombreado del polisono.

d)Despues que todos los polisonos han sido considerados , el arreglo INT(X) tiene la solucion al problema.

ALGORITMOS DE COHERENCIA DE SECCION

Los algoritmos de linea de barrido pueden capitalizar una forma unidimensional de coherencia de aerea conocida como coherencia de seccion.Pequeñas secciones, o sucesiones de pixels , en una linea de barrido caeran en un mismo polisono y el alsoritmo busca una seccion por lo tanto , necesitando hacer unicamente una cuantas comparaciones de profundidad para identificar que polisono es visible a lo largo de esa seccion. Se hacen calculos en el plano XZ para determinar las relaciones entre los segmentos de todos los polisonos que intersectan a la linea de barrido y

asi decidir que segmento es visible en que seccion ; estos segmentos son generados de igual manera que el algoritmo de coherencia de linea de barrido. Las coordenadas X y Z de los puntos terminales de un segmento pueden ser calculadas en forma incremental debido a que ambas son funciones lineales de Y , es decir , la ecuacion de un lado del poligono puede ser escrita como :

$$x = a_1 y + b_1$$

$$z = a_2 y + b_2$$

Las secciones de una linea e barrido se identifican clasificando todos los puntos terminales de los segmentos por su valor X y examinando las regiones delimitadas por estos valores X. Cada seccion caera en una de las siguientes tres clases.

1)Ningun segmento aparece dentro de la seccion ;la intensidad de fondo se despliega.

2)Un unico segmento cae dentro de la seccion ; claramente este segmento es visible y por lo tanto el sombreado del poligono se despliega a lo largo de esta seccion.

3)Varios segmentos caen dentro de la misma seccion ; el segmento mas cercano al observador (el que tiene el valor mas pequeño de Z) se encuentra facilmente comparando las profundidades de todos los segmentos para un mismo valor de X. Una vez que el poligono visible es identificado , se pueden determinar las intensidades apropiadas para esta seccion.

El algoritmo saca ventaja de la coherencia para reducir el ordenamiento en X y Z ; una unica lista , la cual contiene el ordenamiento hecho para los valores de X , con las descripciones de los lados de estos poligonos que intersectan la linea de barrido. A medida que se van renovando en forma incremental, de una en una , las lineas de barrido , se van reclasificando.

La identificacion de secciones es una forma simple de obtener los lados de esa lista.

Durante el procesamiento de cada linea de barrido , el cual es de izquierda a derecha , los lados se usan para mantener una lista de poligonos "activos". El

lado que inicia una seccion estara ya sea del lado izquierdo de un Polisono ,haciendo a este activo , o caera del lado derecho inactivandolo y terminando asi el Polisono.

La naturaleza izquierda o derecha de los lados Puede ser distinguida (1) asociado al Polisono un bit Para registrar cuando el Polisono se halla activo y complementando al bit cada vez que un lado del Polisono se encuentra; o (2) si los Polisonos se dibujan siempre de una manera consistente (contra las manecillas del reloj segun se ve desde fuera del objeto P. ej.) dejando que la direccion de los lados determinen la Paridad , en nuestro caso un lado que se Pinta hacia abajo es un lado izquierdo.

Del lado izquierdo de la seccion , los Polisonos activos son revisados Para encontrar aquel con el valor de Z mas Pequeño (y Por lo tanto el mas cercano al observador). Para simplificar la revision , la lista de Polisonos activos Puede ser clasificada de acuerdo con Z , con los Polisonos entrando y saliendo de la lista segun nos movemos de seccion en seccion. Aun cuando los valores raramente alteran el orden en la Profundidad de los Polisonos; de hecho el orden nunca cambia despues de haber insertado un Polisono en una lista de activos si se Prohiben las Penetraciones entre caras, en las cuales una cara de un objeto Penetra a traves de la cara de otro . El manejo de las Penetraciones requiere ademas aumentar los tres casos vistos a lo larso de una seccion.

Alterando el algoritmo que decide que Polisono es visible dentro de una seccion Para manejar mas casos , Podemos Procesar una linea de barrido en (o con) Pocas secciones , Por ejemplo tenemos el algoritmo de Watkins , el cual es capaz de manejar secciones muy grandes y ademas recuerda , de una linea de barrido a otra, las secciones que son lo suficientemente simple de resolver.Los algoritmos de linea de barrido Pueden tomar ventaja tambien de la coherencia de Profundidad. Una Posicion de un segmento en la lista de Polisonos activos (clasificados Por Profundidad) Parece no variar de una linea de barrido a la siguiente ; una estimacion de esta Posicion reduce el numero de calculos de Profundidad que se requiere Para insertar un Polisono en la lista de activos.

Resumiendo, los algoritmos de linea de barrido sacan Provecho de las coherencias entre dos lineas de barrido consecutivas y de la coherencia de una seccion dentro de una linea de barrido . Tambien simplifican los calculos geometricos reduciendo el Problema tridimensional a una comParacion bidimensional de segmentos en el Plano XZ. Esta simplificacion geometrica Puede acarrear Problemas Por que aquellos Polisonos que sean muy Pequeños o muy angostos Pueden

caer entre dos planos de la linea de barrido .

La actuacion de estos algoritmos depende (o esta relacionado fuertemente con) de la complejidad de la imagen visible lo cual fue observado por Watkins primero que nadie.

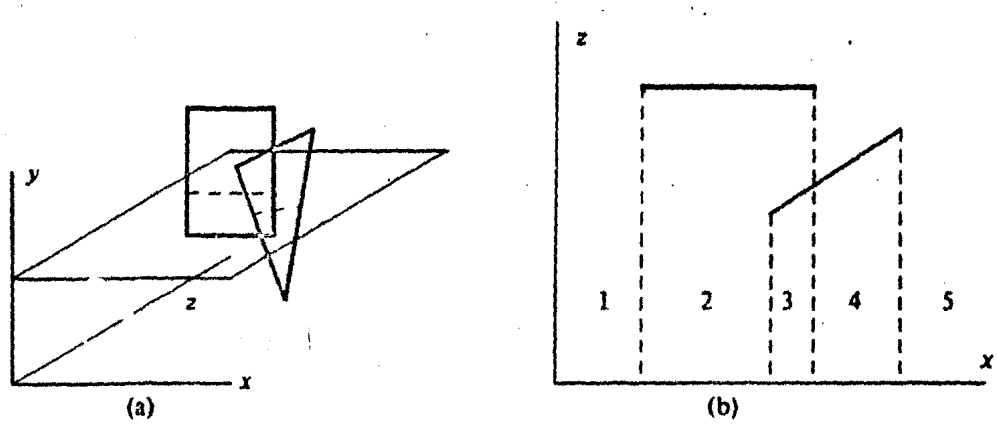


Fig. 5.2 Algoritmo de Linea de Barrido.
a)Sistema de Coordenadas de la Pantalla.
b)Regiones definidas por los valores de x.

5.6 ALGORITMOS DE COHERENCIA DE AREA.

Los algoritmos de coherencia de area tratan de capitalizar la observacion de que la imagen de un poligono tenga igual extension en X que en Y ; los pixels dentro de tal area son coherentes porque muestran una superficie simple. A pesar que el concepto es similar al de coherencia de linea de barrido y al de coherencia de area tratan las direcciones X o Y simetricamente en vez de clasificarlas primero en una direccion y luego en otra. Para tomar mejor ventaja de la coherencia de area , las ventanas probadas por el algoritmo no pueden ser fijadas a priori pero pueden ser seleccionadas de acuerdo a la complejidad de la imagen ; i. e. , las dimensiones de la ventana seleccionada deben ajustarse a las dimensiones a las que la imagen debe ser coherente.

EL ALGORITMO DE WARNOCK.

Warnock desarrollo uno de los primeros algoritmos de coherencia de aerea , el cual selecciona las ventanas de una manera recursiva ; el algoritmo primero trata de "resolver " el problema de las superficies ocultas para una ventana que cubre toda la pantalla , si los poligonos se enciman a la ventana en X y en Y entonces se llama un procedimeinto de decision que trata de analizar las relaciones entre los poligonos y genera un despliegue para esta ventana.

Los casos mas simples , P. ej. un solo poligono en la ventana o mas aun ninguno , son rapida y facilmente resueltos. Si al procedimiento de decision se le hace muy complicado el despliegue en esa ventana entonces el algoritmo divide la ventana en cuatro mas pequeñas , usualmente iguales , y vuelve a intentarlo en cada una. esta tecnica da lugar a una subdivision de arbol en las ventanas ; si una region de la imagen es muy complicada , la recursion forzara al analisis de ventanas mas y mas pequeñas terminando cuando eventualmente se encuentre una ventana que puede ser resuelta o bien cuando el tamaño de la ventana sea tan pequeño como un pixel ; en este ultimo caso la intesidad del pixel se escoge para que represente uno de los poligonos que es visible dentro de ese pixel.

El procedimiento de decision que analiza las ventanas toma como entrada una lista de los poligonos que puedan ser relevantes para esa ventana , clasificando cada poligono dentro de uno de tres grupos:

a) Polisonos disjuntos: ninsun polisono (o parte de alsun polisono) se encima con la ventana en cuestion en la direccion X o en la direccion Y.

b) Polisonos intersectados : los polisonos caen parcial o totalmente dentro de la ventana.

c) Polisonos que rodean : los polisonos rodean completamente a la ventana.

Los calculos de visibilidad usan esta clasificacion para descartar aquellos polisonos que son relevantes para senerar una imasen en la ventana ; los polisonos disjuntos claramente no son relevantes.

El tratamiento de los polisonos que rodean es la clave para la eliminacion de las superficies ocultas ya que si existe un polisono que rodea que esta mas cerca que un polisono intersectado este ultimo es considerado invisible y por lo tanto removido de la lista de polisonos relevantes; la eliminacion de estas caras ocultas es mas eficiente si la lista de los polisonos potencialmente relevantes estan clasificados por la profundidad Zmin del vertice mas cercano al observador . Siempre que el procedimiento de decision encuentra un polisono que rodea , debe recordar al punto mas profundo del polisono en la ventana como Zminmax asi cuando considere a otro polisono en la lista , si su Zmin es mayor que Zminmax claramente es ocultado por el polisono que rodea ; mas aun ya que la lista esta clasificada de acuerdo a Zmin todos los polisonos que se encuentran mas abajo que este que estamos checando tambien son visibles terminando prematuramente la busqueda en la lista de polisonos que rodea y ahorrando asi una buena cantidad de tiempo de procesado.

Cuando los polisonos disjuntos y aquellos que rodean que son relevantes han sido eliminados de la lista debe ser claro ya como senerar la imasen para esa ventana , si ninsun polisono , o solo uno , queda entonces la imasen se puede senerar muy facilmente ; si solo quedan polisonos intersectados y no se encuman en X o en Y la imasen puede ser senerada tambien de una manera facil; si por otra parte no se puede decidir que debe ser mostrado y que no , el alsoritmo subdivide la ventana y vuelve a tratar.

El analisis de las relaciones entre los polisonos y las ventanas aumenta la velocidad del analisis de las subventanas ; por ejemplo, si un polisono rodea a una ventana rodeara a todas las subdivisiones sisuientes de la misma, analosamente si un polisono es disjunto a una ventana , sera disjunto a todas las subdivisiones sisuientes y no debe ser pasado por el alsoritmo de decision a las subsisuientes subdivisiones , asi

conforme la recursion avanza y las ventanas se van haciendo mas pequeñas la lista de polisonos que es relevante a las ventanas se van enconsiendo hasta que eventualmente la lista llesa a ser tan pequeña que al algoritmo ya no analiza sino que despliesa directamente.

El tiempo de maquina requerido por el algoritmo de Warnock es proporcional a la complejidad de la imagen final y no a la complejidad de la escena , la cantidad de calculos puede ser estimada por el numero de subdivisiones requeridas . Las subdivisiones siempre se llevan a cabo si las caracteristicas desplesables caen dentro de la ventana a subdividir , por lo tanto el tiempo de maquina es proporcional a la complejidad visual. El numero exacto de subdivisiones requeridas esta influenciado por los detalles del procedimiento de decision ; entre mas casos que se detectan sean desplesados de estos elementos se pueden hacer grandes ahorros ; una evaluacion de la actuacion de algunos procedimientos de decision ha sido hecha por McCallister, s. y I.E. Sutherland [MCCA].

Aun cuando la subdivision de la pantalla depende de la complejidad de la escena , la geometria cuadrada de las ventanas es una fuente de ineficiencia e ineficacia. Otra forma de hacerlo podria ser crear subventanas que se ajustaran a la forma de los polisonos en la ventana orisinal de tal manera que los polisonos que fueran relevantes en la ventana orisinal deberan ser recortados contra la subventanas para ser procesados. La salida final de tal algoritmo es un nuevo conjunto de polisonos para los cuales se puede sarantizar que no se enciman en la direccion X ni en la direccion Y y ademas retienen sus relaciones orisinales de profundidad ,como por ejemplo el algoritmo de Wiler, K. y P. Atherton [ATHE]. Una diferencia adicional del algoritmo es que los polisonos son calculados con una alta precision en el espacio del objeto lo cual puede ser de mucha utilidad cuando se trata de simular el sombreado.

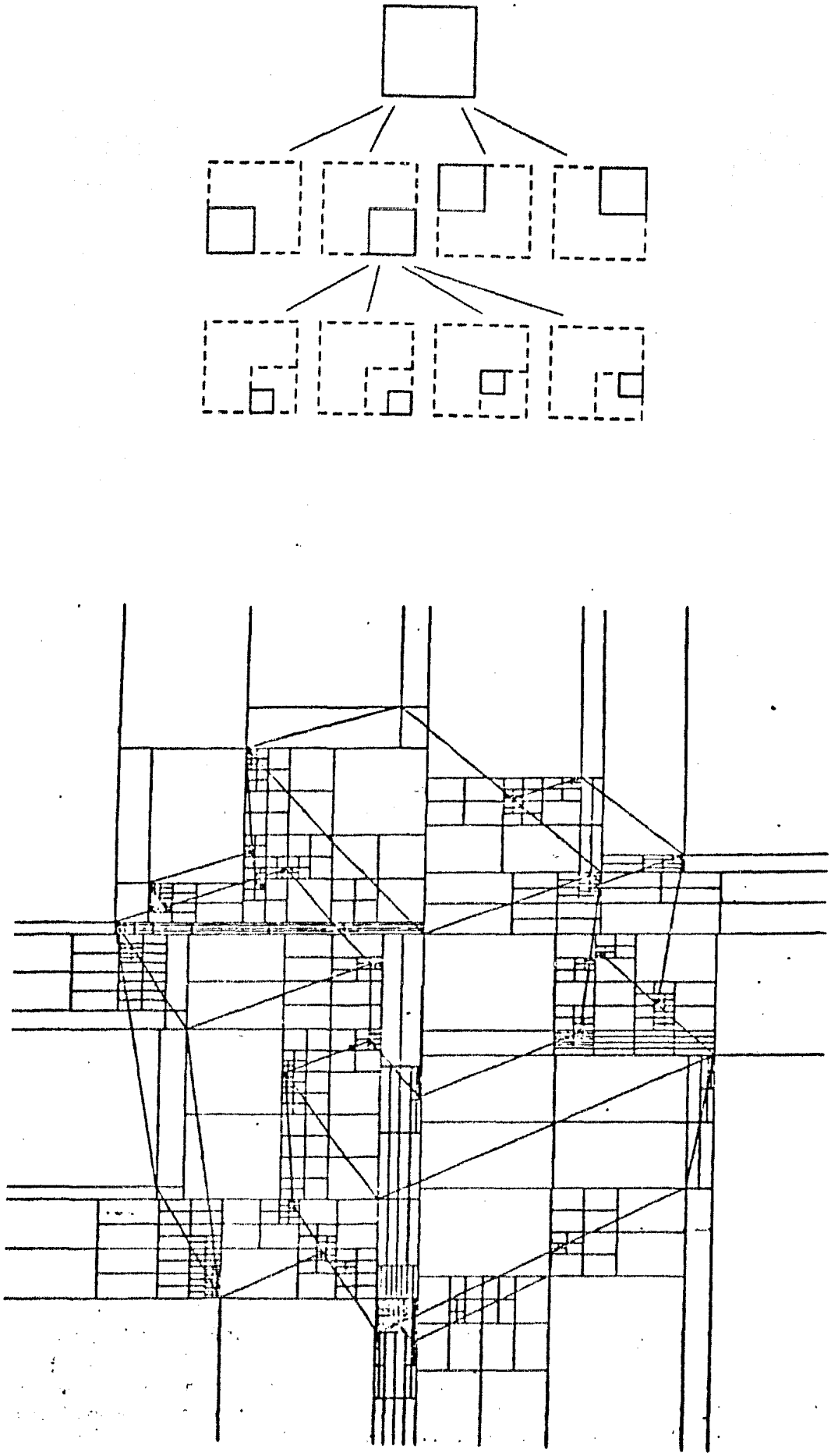


Fig. 5.3 Algoritmo de Warnock.

5.7 ALGORITMOS POR PRIORIDAD.

A diferencia de los algoritmos de linea de barrido y de coherencia de area los cuales tratan de encontrar las relaciones de profundidad y despues de realizar los calculos sobre el plano XY solo hasta que la visibilidad se ha determinado ; los algoritmos de prioridad resuelven el encimamiento cuando estan convirtiendo el barrido a un area bidimensional.

EL ALGORITMO DE NEWELLL , NEWELL Y SANCHA.

La idea del algoritmo es arreglar todos los polisonos en la escena en orden de prioridad basandose en su profundidad ; aquellos polisonos mas cercanos al observador tendran asignada una prioridad mas alta que aquellos que se encuentran mas alejados.despues que la prioridad ha sido determinada ,los polisonos son transformados por barrido, uno a la vez , a la memoria de almacenamiento para luego desplegarla , empezando con los polisonos de mas baja prioridad ; este procedimiento generara una imagen con superficies ocultas adecuada si el orden de prioridad se calcula adecuadamente.

Los calculos de prioridad empiezan por ordenar en una lista a todos los polisonos en funcion de su Zmax,la profundidad de aquel punto del polisono que esta mas alejado del observador ; si despues de la clasificacion ningun polisono de la lista se encima con otro en profundidad (ver la discusion del encimamiento en las secciones anteriores), la lista esta en el orden correcto de prioridades> estos calculos tan simples de prioridad se dan a basto solo para unas cuantas escenas tales como series do planos perpendiculares a la direccion el observador, etc.

Si los polisonos en la lista de prioridad se enciman en profundidad entonces se requieren calculos mas cuidadosos para determinar el orden correcto. Considerese p. ej. el ultimo polisono en la lista , P ; si P no se encima en profundidad con su predecesor , Q , P no se encima con ningun otro polisono en la lista dado que es el polisono de menor prioridad. Por lo tanto, su posicion al final de la lista esta correcta. Lo mas usual es que P si se encima en profundidad con un conjunto de polisonos , (Q) , que le preceden inmediatamente en la lista ; para encontrar este conjunto hay que barrer toda la lista hacia atras buscando cuando P se encima con algun polisono. El siguiente paso es demostrar que P no obscurece a ningun polisono en (Q), i.e. que P esta ya en su lugar

correcto en la lista de prioridad. . Si algun polisono Q , en (a) se ve obscurecido por P entonces postulamos que Q debe tener una prioridad mas baja que P y lo movemos a que este despues de P en la lista de prioridad.

El punto central de la clasificacion por prioridad es la relacion "P obscurece a Q" , P nunca obscurecera a Q si alguna de las siguientes pruebas es verdadera:

1) La prueba minimax de profundidad indica que P y Q se enciman en profundidad y Q esta mas cerca al observador que P. Esta prueba esta ya implementada en la clasificacion inicial por profundidad para todos los polisonos y por la forma en que P y (a) han sido seleccionados.

2)La prueba minimax en XY indica que P y Q no se enciman en X o en Y.

3) Todos los vertices de P estan mas alejados del punto del observador que el plano de Q . Esta prueba es simplementa substituyendo las coordenadas (x,y) de cada vertice en la ecuacion del plano de Q , y resolviendo para la profundidad de Q.

4) Todos los vertices de Q estan mas cercanos al punto del observador que el plano de P.

5) La prueba de encimamiento total indica que P y Q no se enciman en X o en Y.

Estas pruebas se deben aplicar en el orden dado arriba debido a que la ultima es mucho mas compleja que las anteriores.

La relacion "P obscurece a Q" no es enteramente suficiente para clasificar los polisonos por orden de prioridad ; puede haber situaciones complejas donde , por ejemplo, P obscurece a Q en una parte , Q obscurece a R en otra parte y por ultimo R obscurece a P en otra parte , otra situacion critica es en el caso de que tensamos penetracion de una cara por una seccion de un polisono.Una forma de atacar este problema es dividir unos de los polisonos conflicto en secciones y asisnando diferentes prioridades a cada parte.

Este algoritmo de prioridades calcula la visibilidad mediante criterios geometricos en vez de comparaciones pixel por pixel en profundidad(ver algoritmo de profundidad en memoria) y por lo tanto capitaliza la coherencia de los polisonos en profundidad para hacer decisiones de visibilidad para los polisonos como un todo. La clasificacion por prioridad se realiza en el espacio del objeto la lista resultante de prioridades es valida para una unica posicion del observador pero puede ser usada para

generar imagenes a diferentes escalas.

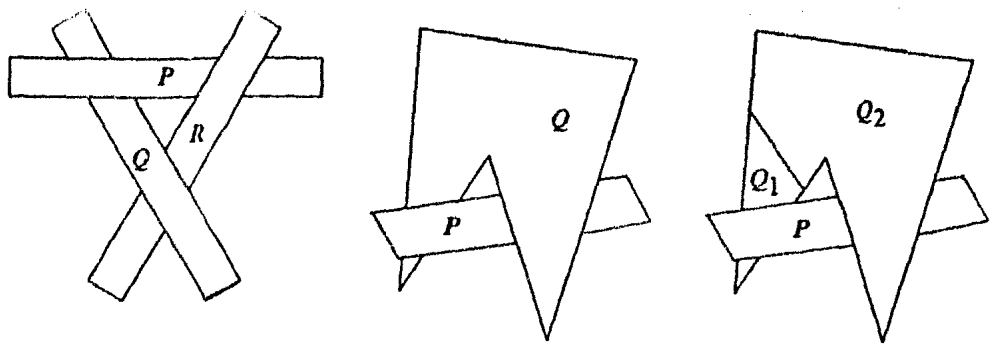


Fig. 5.4 Poligonos donde no existe prioridad.

5.8 LA ELECCION DE UN ALGORITMO.

En las secciones precedentes se ha descrito someramente cinco algoritmos : el de profundidad en memoria,dos de linea de barrido, el metodo de Warnock y el algoritmo de prioridad de Newell, Newell y Sancha , siendo el orden de aparicion el de complejidad; aun cuando el algoritmo de profundidad en memoria es el mas simple de implementar los otros son muy complicados de implementar.

Estos algoritmos pueden ser extendidos considerablemente mas alla de la descripcion hecha aqui , aun cuando eso les daran una mayor complejidad. Cada prueba de visibilidad para cada algoritmo puede ser extendida para manejar relaciones entre poligonos mas complejos (seleccion de secciones de Watkins, etc) ,Pero pueden hacer al algoritmo mas lento.

Las escenas con penetraciones entre superficies requieren de un tratamiento especial dentro de los algortimos de profundidad en memoria y de prioridad , los cuales pueden manejar las penetraciones mas o menos bien mientras que los algoritmos de linea de barrido y el de Warnock deben ser extendidos (o aumentados).

Estas ideas , a pesar de que los algoritmos funcionan sobre modelos poligonales de una escena , pueden ser adoptadas para eliminar partes ocultas en superficies suaves. Catmull ha desarrollado un algoritmo basado en la subdivision de superficies y en la profundidad de memoria.

Los algoritmos de linea de barrido que manejan superficies que han sido diseñados por Blinn , Whitted , y Lane y Carpenter . Muchos dispositivos de superficies curveadas no usan un algoritmo directo si no que operan con aproximaciones poligonales y despues se utilizan metodos de sombreados para que se vean como "suaves".

Generar imagenes en tiempo real requieren de un algoritmo eficiente , ademas de necesitar de cierto equipo especial en "harduare" para implementarlo. Los primeros equipos de esta clase fueron diseñados y construidos por General Electric para la NASA (Depto. de Espacio y Naves Tripuladas) en 1968 y se basan en el algoritmo de prioridad que concurrentemente barren y convierten un gran numero de poligonos , seleccionando aquel que que tiene la prioridad mas alta para desplegarlo.

El Algoritmo de Watkins de linea de barrido es la base para muchos sistemas que operan en tiempo real como el construido por Evans y Sutherland Computer Corp. El algoritmo de Warnock , p. ej. , se puede adaptar facilmente para producir dibujo de lineas ; cuando se encuentra que una ventana es suficiente para

el despliegue directo, los lados de los polígonos visibles se muestran como si fueran líneas. Si el algoritmo genera alguna vez ventanas que sean (o midan) una unidad de pantalla entonces se pinta un punto al menos (por default). Los algoritmos de línea de barrido pueden ser adaptados de igual manera ; en vez de desplegar segmentos sombreados , Lo que se hace es pintar como líneas los lados. En cierta forma cualquier algoritmo de superficies ocultas puede usarse para generar una imagen con líneas ocultas usando la regla de sombreado que le asigna a los lados del polígono (su perímetro) el color negro y al interior el color blanco (o viceversa) ; aun mas , a estas adaptaciones se suman muchos algoritmos diseñados específicamente para eliminar líneas ocultas , estos se concentran en los lados y en la propiedad de coherencia de lado para generar una imagen.

La elección de un algoritmo no se limita a las indicadas arriba ya que una gran cantidad de algoritmos se han desarrollado , algunos especializados en cierto tipo de imágenes. Aun mas las técnicas para los algoritmos existentes pueden ser combinadas de varias maneras para diseñar nuevos algoritmos;por ejemplo al algoritmo de línea de barrido descrito con anterioridad es simplemente una combinación entre el algoritmo de profundidad en memoria y el algoritmo de conversión de barrido en XY , tales combinaciones son en general simples de hacer y permiten explotar varias características y varias propiedades de coherencia o utilizar ciertas estructuras de datos con implementaciones particularmente eficientes en la computadora que se este usando . Otras ideas para combinar algoritmos están dadas por Sutherland , Sproull y Schumaker.[SUTH]

Ambas , la eliminación de líneas ocultas y la eliminación de superficies ocultas , parece ser que requieren de la clasificación geométrica ;todos los algoritmos que se han explorado han mostrado explícitamente la existencia de ciertos pasos de clasificación ;otros algoritmos descritos por Sutherland ,también hacen uso de la clasificación .Las diferencias entre los algoritmos son relativas a las diferentes técnicas de clasificación usadas y al orden en el cual la clasificación por profundidad y la clasificación lateral se hacen.

Los pasos de la clasificación en un algoritmo de superficies ocultas tienen una fuerte influencia en la actuación del algoritmo ; la tabla 5.1 compara las estimaciones del tiempo de ejecución de cuatro de los algoritmos sobre escenas que tiene 100, 2500 y 60000 caras. El crecimiento en los tiempos de ejecución se puede atribuir a:

los algoritmos de clasificacion usados, el numero de cosas a clasificar y la complejidad de la comparacion entre los objetos de la escena ,etc.

TIEMPOS ESTIMADOS DE EJECUCION DE
ALGUNOS ALGORITMOS DE SUPERFICIES OCULTAS

ALGORITMO	NUMERO DE CARAS			RAZON APROXIMADA		
	100	2 500	60 000	1:	25:	600
Profundidad en memoria	7.5	7.5	7.5	1:	1:	1
Coherencia de seccion	0.5	3	64	1:	6:	120
Warnock	1.5	9	43	1:	6:	30
Newell , Newell y Sancha	0.14	1.4	71	1:	10 :	500

TABLA 5.1

>

Muchos de los algoritmos requieren mas tiempo de maquina entre mas compleja es la escena a procesar debio a que el numero de cosas a clasificar aumenta.

Aun cuando el algoritmo de Newell, Newell y Sancha se ve muy atractivo para cuando tenemos un numero pequeño de caras , ya no lo es tanto si el numero de caras aumenta (ver tabla 5.1).

El algoritmo de Warnock y el algoritmo de Coherencia de Secciones , los cuales operan completamente en el espacio de la imagen , parecen tener una mejor actuacion aun cuando las escenas se vuelven sumamente complejas. Unicamente el algoritmo de profundidad en memoria parece tener una actuacion uniforme, y parece que esto es debido al hecho de que a medida que las caras crecen en numero decrecen en tamaño.

Cuando algunos de estos algoritmos se implementan en computadoras estandard, ninguno procesa escenas interesantes lo suficientemente rapido como para que sea usado en modo interactivo ; a pesar de eso ya se

ha construido "hardware" de proposito especifico para que supere las velocidades de interaccion y logre llegar a la generacion de imagenes en tiempo real. Cuando existe el procesamiento en paralelo (o es barato conseguirlo) en circuitos LSI , estos algoritmos pueden hacer uso facilmente del procesamiento multiple para generar imagenes lo suficientemente rapido para ser considerados dentro de un sistema interactivo (Procesador de Arreglos).

Aun hoy estos algoritmos son usados comunmente para obtener imagenes finales de alta calidad en modelos geometricos construidos interactivamente.

5.9 ALGORITMO DE LINEAS OCULTAS USADO EN EL PAQUETE.

Una funcion g de R a R es a menudo presentada como un conjunto de valores sobre los puntos de una malla.

La tarea de generar imagenes de superficies reticuladas puede ser realizada por algoritmos de lineas ocultas muy generales [FRANK] ,[SUTH].

Esta generalidad hace que los algoritmos sean muy lentos , por esa razon se ve la necesidad de crear algoritmos especificamente para superficies reticuladas [AND] , [BUT], [WILL] ,[WRIG].

Butland se restringe a proyecciones paralelas las cuales se hacen sobre el plano XY con un angulo de 45 grados con respecto al eje X , de esta manera no hay mucha versatilidad en el manejo de la imagen pero ayuda a que sea mas sencilla la eliminacion de las lineas ocultas.

El algoritmo de Kubert, Szabo y Giuliori trabaja en el espacio del objeto y con una proyeccion en perspectiva la cual no restringe el plano de proyeccion ni el centro de proyeccion. Este algoritmo se concentra en las relaciones geometricas entre las lineas que forman la superficie para determinar que parte de la superficie es visible desde el punto de vista del observador.El tiempo de maquina que se emplea en este algoritmo tiende a crecer con el numero de lineas que forman la malla , visibles o no.

Una alternativa es trabajar el algoritmo en el espacio de la imagen [WRIG] , de esta manera crece mas lentamente en tiempo de maquina.

La solucion que da Williamson [WILL] , es la de dibujar en una sola direccion con respecto a X o con respecto a Y , pero no en ambas direcciones , de esta ,manera no se obtiene una superficie reticulada.

Analizando todos estos algoritmos se llega a la conclusion que el algoritmo que se necesita es el que trabaja en el espacio de la imagen.

Una vez que se tiene la imagen transformada se tiene que tener un criterio para obtener la eliminacion de las lineas ocultas.

Una vez que se ha calculado la funcion que se desea graficar , esta se almacena en un arreglo rectangular para formar la malla.

Los elementos de la malla con indice (i,j) es el rectangulo en el plano XY acotado por los planos x=xi , x=xi+1 , y=yj y y=yj+1.

La imagen de un elemento de la malla bajo una funcion sera llamada faceta o cara.

Una superficie reticulada es una funcion continua g de R a R.

Cada elemento de la malla es transformado antes de

obtener la proyeccion. Se puede cambiar de escala y rotar la superficie.

Una vez que se ha transformado se obtiene una proyeccion oblicua cavalier para manejar la superficie en el espacio de la imagen.

La tecnica usada para ocultar las lineas en el espacio de la imagen , esta basada en dos premisas:

1)Las lineas que estan mas cercanas al observador son graficadas primero.
2)Una curva o porcion de curva es ocultada si esta dentro de la region acotada por lineas previamente graficadas.

El ocultamiento es obtenido al guardar el maximo y el minimo valor alcanzado por las lineas previamente graficadas.

Tanto el minimo como el maximo son funciones que definen una region donde no deben verse las lineas.

Para cada porcion de curva que caiga dentro de la region acotada por estas dos funciones es considerada oculta.

La primera region es determinada por las lineas que estan mas cercanas al observador.

Se utiliza tres parametros para localizar el punto previo con respecto a la region acotada por las funciones.

Si es 1 , el punto esta arriba de la region, si es 0 esta adentro de los limites de la region y si es -1 , esta abajo de la region.

El area se va actualizando conforme se va graficando las curvas.

Si (IX,IY) es el punto actual, si sucede que IY>MAX , donde MAX es el maximo que define la region , se actualiza el area de de ocultamiento , o bien si IY<MIN se altera el minimo que define la region.

Clasificando de esta manera los puntos a graficar se va a tener informacion del punto previo y del punto inicial. Esta informacion va a ayudar a trazar o no una porcion de curva.

Si el punto previo (IX',IY') esta arriba de la region y el punto actual (IX,IY) tambien lo esta , entonces se traza.

Si el punto previo (IX',IY') esta en la region y el punto (IX,IY) tambien lo esta no se traza.

Si los puntos (IX',IY') y (IX,IY) estan abajo de la region entonces se traza el segmento.

Queda ver cuando (IX,IY) o (IX',IY') estan en la region o fuera de ella alguno de estos puntos.

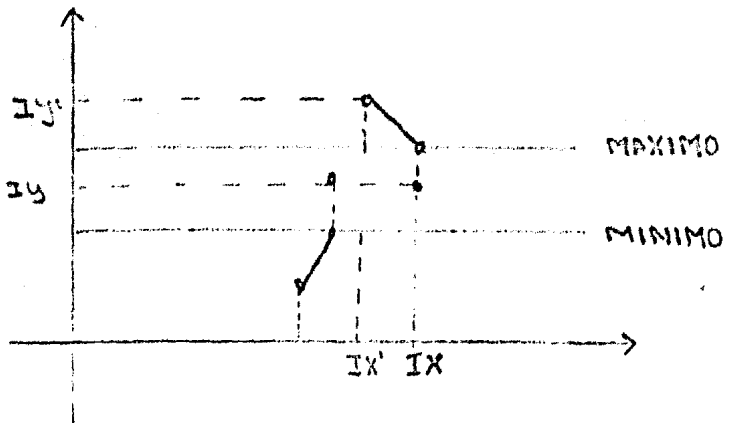


Fig. 5.5

Si el punto previo esta arriba (IX',IY') y el punto (IX,IY) cae en la region , el segmento a graficar es (IX',IY') ,(IX,MAXIMO).

Si el punto esta abajo (IX',IY') y el punto (IX,IY) cae en la region , el segmento a graficar es(IX',IY') ,(IX,MINIMO).

Si ambos puntos caen dentro de la region no se grafica pero se guarda la informacion del punto (IX,IY) para el siguiente punto.

Los ejemplos de este algoritmo pueden verse de las figuras(0-11) del siguiente capitulo.

C A P I T U L O V I

CAPITULO VI

UN PAQUETE DE GRAFICACION PARA EL MODELADO DE CURVAS Y PARA LA REPRESENTACION DE FUNCIONES DE DOS VARIABLES.

6.1 INTRODUCCION.
6.2 DISEÑO DEL PAQUETE.
6.3 USO Y EJEMPLOS DEL PAQUETE DE GRAFICACION.

6.1 INTRODUCCION. El diseño de sistemas de Graficacion es un aspecto muy importante en la Graficacion por Computadora.Sin tales sistemas los programas de aplicacion pueden ser extremadamente dificiles de escribir y el obtenerlos se vuelve muy lento.

Un paquete de Graficacion es un tipo de sistema de Graficacion considerado como el mas basico y esencial de todos ellos.

Este es un conjunto de subrutinas o funciones usados por un programa de aplicacion (en este caso el modelado de curvas y el graficar funciones de dos variables) para generar dibujos sobre un graficador de papel o pantalla de rayos catodicos.

PROGRAMA DE APLICACION.

En muchas areas como en la fisica , insenieria y matematicas se maneja una cantidad abundante de figuras e ilustraciones para investigaciones, reportes tecnicos ,enseñanza ,etc. . Muchas de estas figuras son funciones matematicas de una o dos variables.Las funciones de una variable son facilmente graficadas ya que estan en un espacio de dos dimensiones.Si no se tiene la representacion analitica de la funcion o se quiere obtener alguna representacion grafica , se usa los metodos para el modelado de curvas (interpolacion).

Si se quiere graficar funciones de dos variables se debe utilizar algun artificio.

Los dos metodos mas comunes de representar funciones de dos variables son graficar las curvas de nivel (contornos) o bien usar proyecciones planas.

Se usa las curvas de nivel cuando se requiere

obtener informacion numerica y las proyecciones planas para proporcionar una mejor visualizacion de la fisura.

La utilidad de la descripcion de funciones matematicas no se restringe a libros de texto o reportes tecnicos.Su aplicacion tambien se da con la solucion de problemas en fisica o ingenieria ayudando a cientificos o ingenieros a obtener una vision de la naturaleza de las funciones con las cuales esta trabajando.

6.2 DISEÑO DEL PAQUETE.

La manera en que diseño el paquete de Graficacion se puede observar tres niveles (figura 5.1).

El nivel mas bajo son las rutinas basicas ,estas son un conjunto de rutinas que generan salida al controlador del sistema de graficacion y el usuario no necesita de comunicarse con el "hardware" en su estructura de datos.

En lugar de esto se comunica con el conjunto de subrutinas tales como: mover la pluma,dibujar ejes con cierta notacion,etc.(Anexo).

El segundo nivel ,las rutinas Generales de Graficacion ,compuesto de rutinas muy generales que son creadas independientemente de la aplicacion.

Las rutinas que pertenecen a este nivel son : las rutinas de transformaciones en dos y tres dimensiones , recorte ,ventana ,puerto de vision ,los distintos tipos de proyecciones, algoritmos de lineas ocultas etc.

El tercer nivel ,Programas de Aplicacion, es el nivel mas alto .En este nivel se elaboraron las rutinas que utilizan los dos niveles anteriores ,pero estas ya tendran alguna aplicacion especial.

En este nivel corresponden a las rutinas de modelado de curvas y las rutinas para generar la superficie utilizando las rutinas de proyecciones y lineas ocultas. Como se puede observar son rutinas mas especificas.

Las rutinas en un nivel pueden llamar a rutinas del mismo nivel o bien a los que estan a un nivel mas bajo.

Por ejemplo, las rutinas del segundo nivel no pueden llamar a rutinas del nivel de aplicacion.

La idea de tener el nivel de las rutinas Generales de Graficacion ,es el de reducir la cantidad de nuevo codigo que tiene que hacerse para cualquier aplicacion particular.

Otro proposito es que estas rutinas sean de proposito general para que sea transferible a diferentes ambientes haciendo muy pocos cambios a este nivel y al nivel de aplicacion.

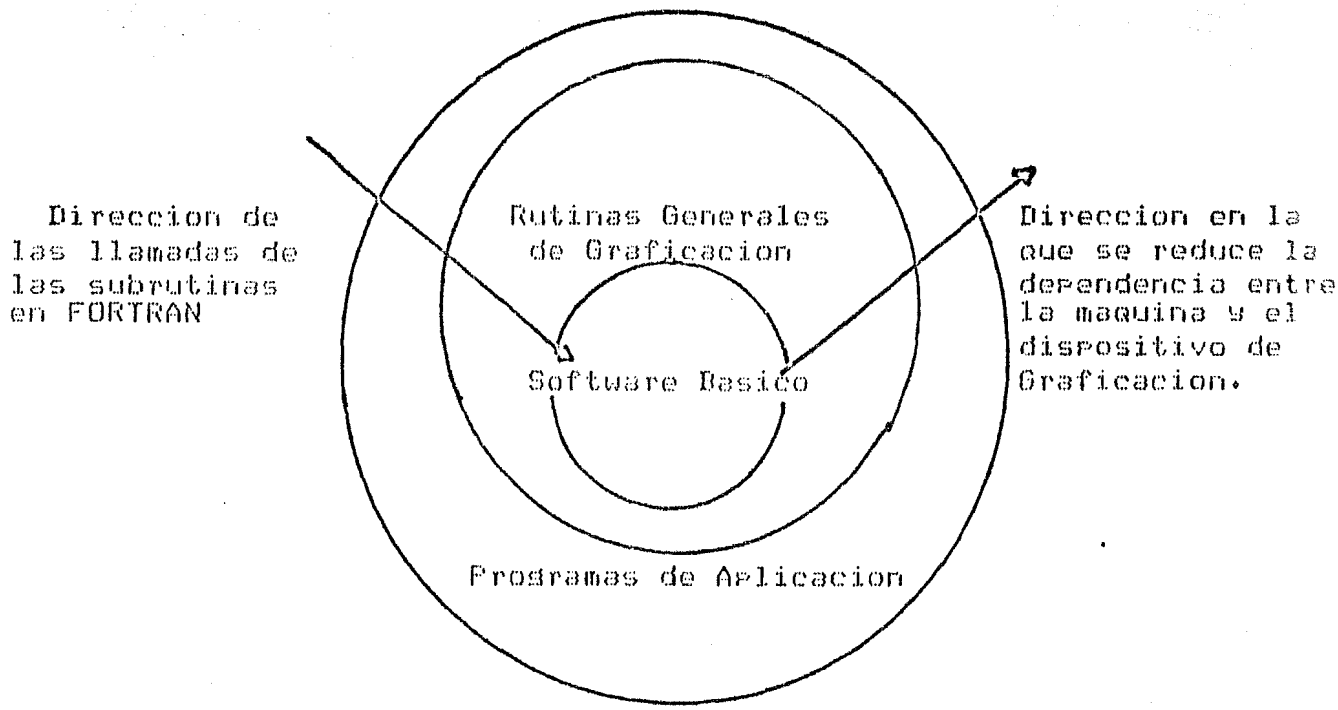


FIG. 6.1 NIVELES DE SOFTWARE.

6.3 USO Y EJEMPLOS DEL PAQUETE DE GRAFICACION.

ETAPAS PARA EL USO DEL PAQUETE PARA EL MODELADO DE CURVAS.

Se llama al paquete DOSD para modelar una curva con los puntos de control

```
>@DK2:[200,200]DOSD
```

De aqui en adelante se escose el metodo que se desea:BEZIER o B-SPLINES.

Para BEZIER se pide lo siguiente :

1.- Numero de puntos de Control.

2.- Las coordenadas de los puntos de Control.

3.- EL numero de puntos para el parametro U.

4.- El tamaño de la ventana.

Para B-SPLINES se pide lo siguiente:

1.- Numero de puntos de Control.

2.- Las coordenadas de los puntos de Control.

3.- El orden del B-spline.

4.- El tamaño de la ventana.

ETAPAS PARA OBTENER LA GRAFICA DE FUNCIONES DE DOS VARIABLES.

Antes de escoger cualquier proyeccion hay que saber cual grafica de que funcion se desea .

Primero, hay que usar el editor para crear el programa en FORTRAN.

Segundo, se llama al paquete TRESD para graficar la superficie.

>@DK2:[200,200]TRESD

De aqui en adelante se va a contestar las preguntas que se necesiten dependiendo de la proyeccion que se desee.

Tercero , cuando aparezca TKB> , hay que dar CONTROL-Z.

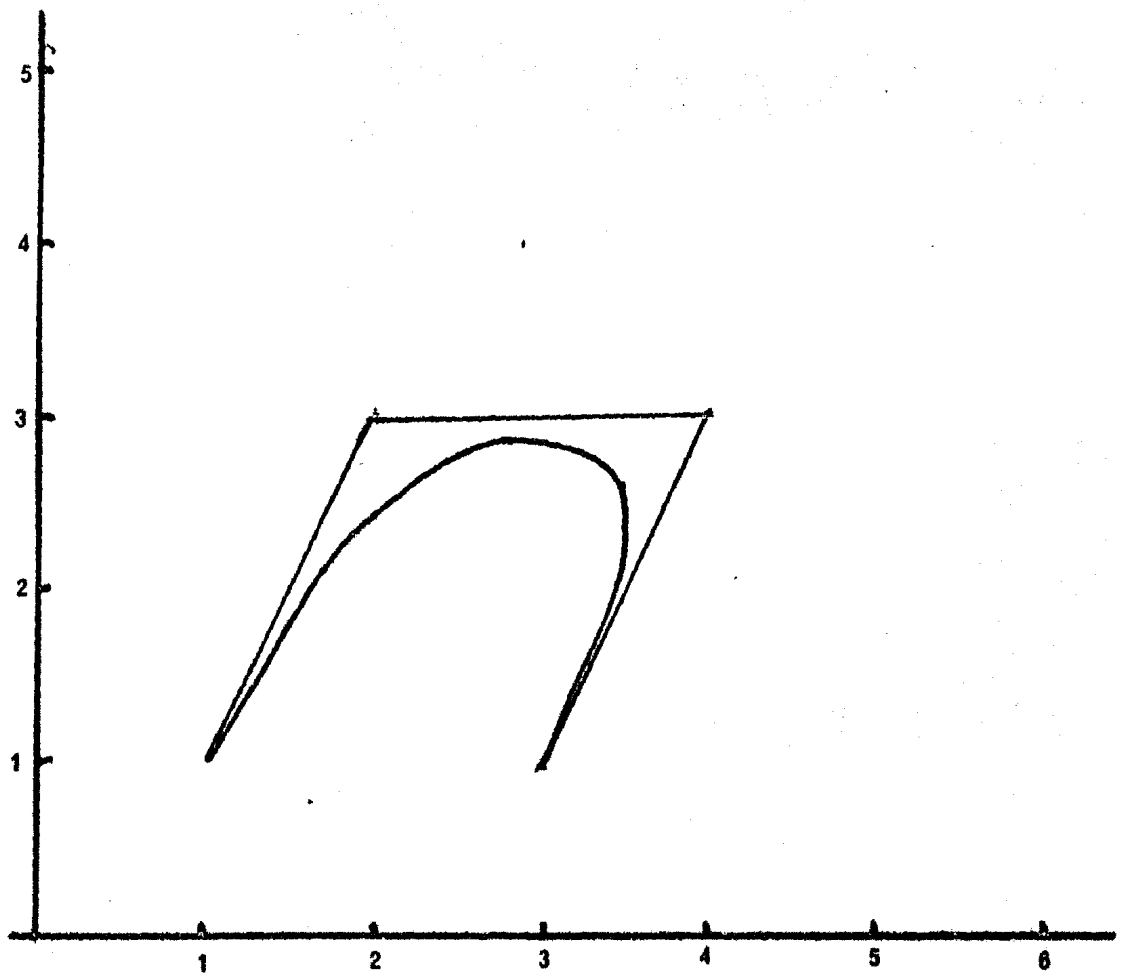
Cuarto , se ejecuta la tarea para obtener la grafica.

M O D E L A D O D E C U R V A S

(CALCOMP)

```
>@DOSD
>;**********************************************************************************
>;
>;
>;       Escoger alguno de los siguientes metodos de interpolacion:
>;
>;       Metodo de Bezier          ....     GENBEZ
>;
>;       Metodo de Splines
>;       Cubicos                   ....     GENCUB
>;
>;       Metodo de Bsplines        ....     GENBSP
>;
>;
>;**********************************************************************************
>* Cual metodo se desea? [S]: GENBEZ
>PIP FUNCION.FTN=GENBEZ.FTN
>FOR FUNCION=FUNCION
.MAIN.
>TKB @TAREA1
>TKB
TKB>^Z

>* Se desea correr el programa?? [Y/N]:Y
>RUN TAREA1
```

METODO DE BEZIER

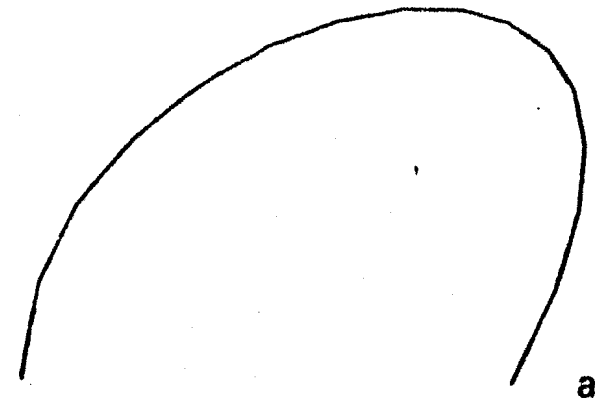


VERTICES: (1,1) ,(2,3) , (4,3) ,(3,1)

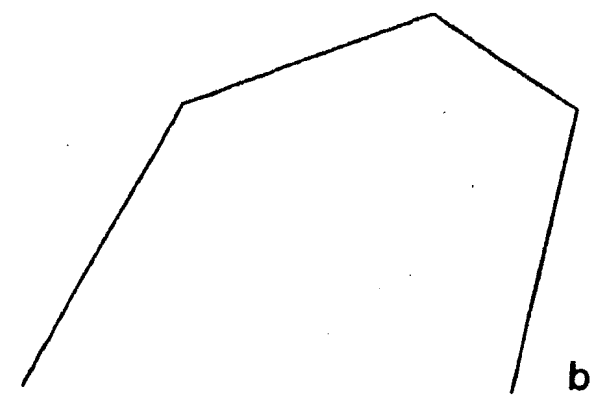
a)u=15

b)u=4

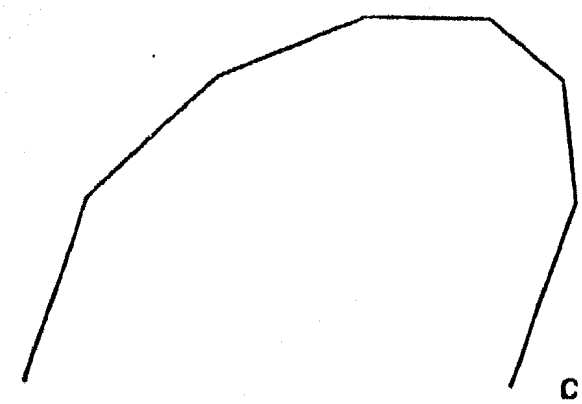
c)u=7



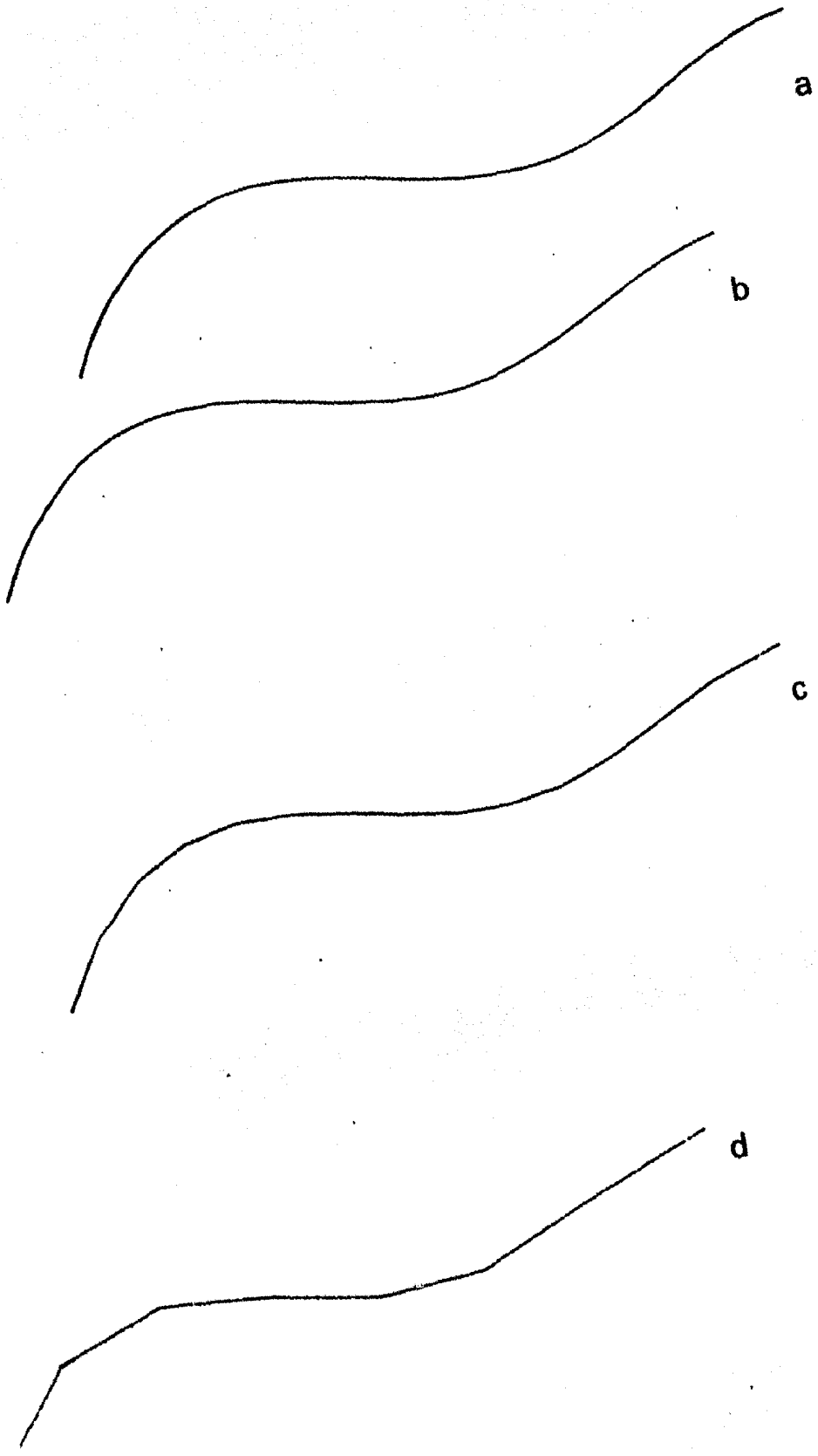
a

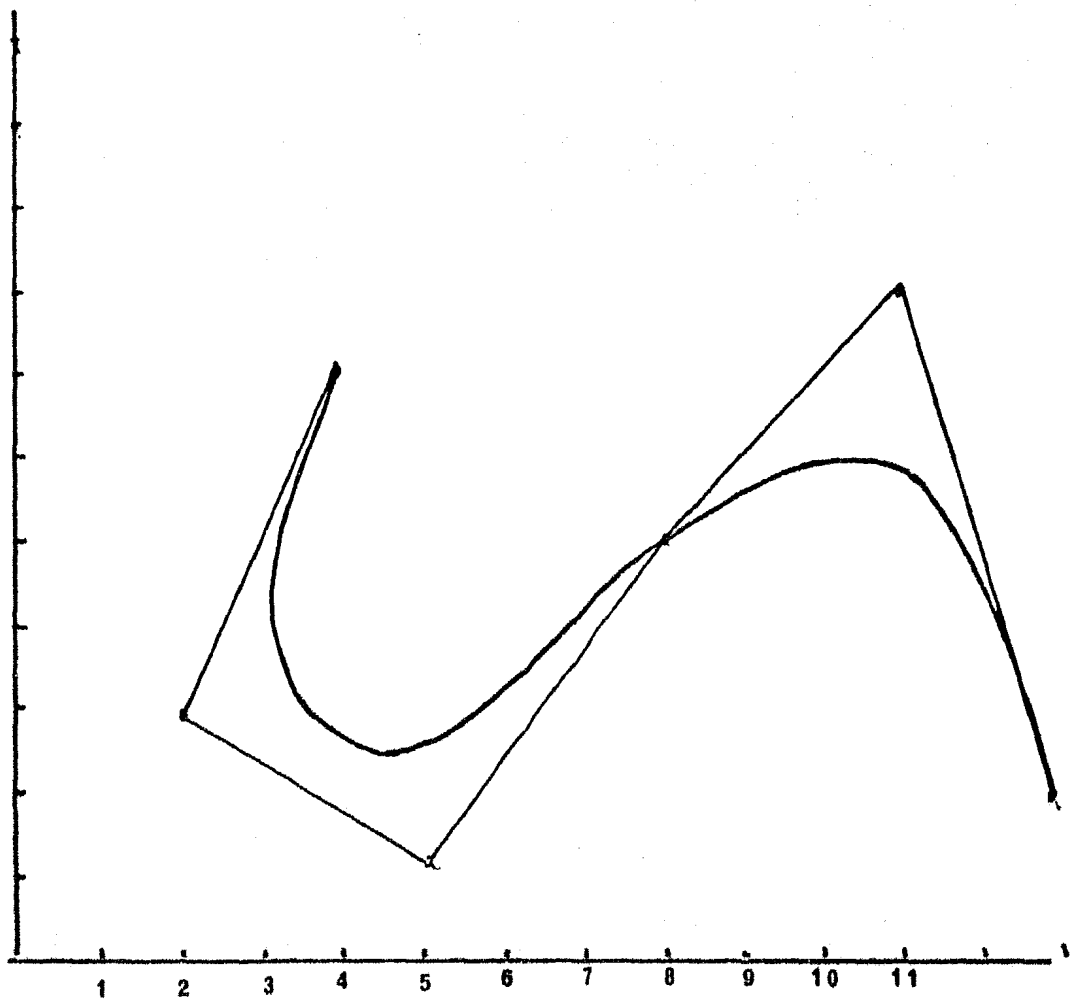


b



c


a
b
c
d



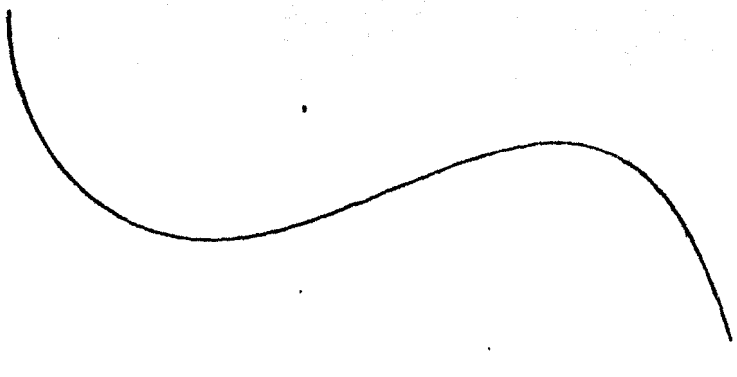
VERTICES: (4,7),(2,3),(5,1),(8,5),(11,8),(13,2)

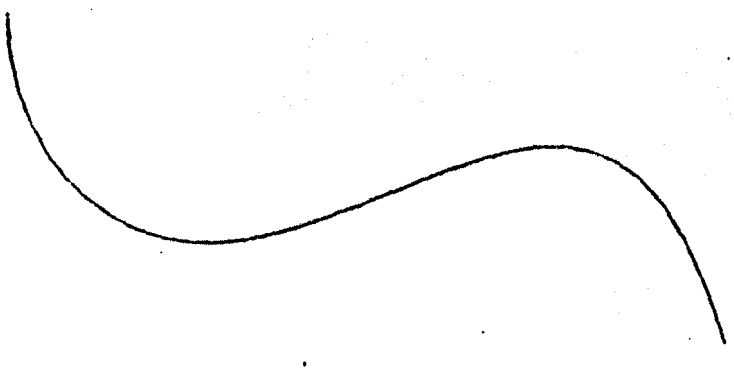
a)u=6

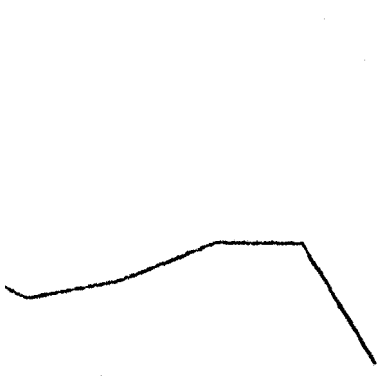
b)u=15

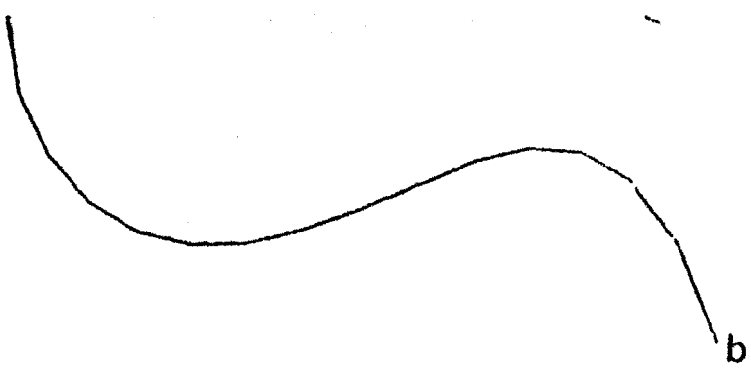
c)u=30

d)u=45

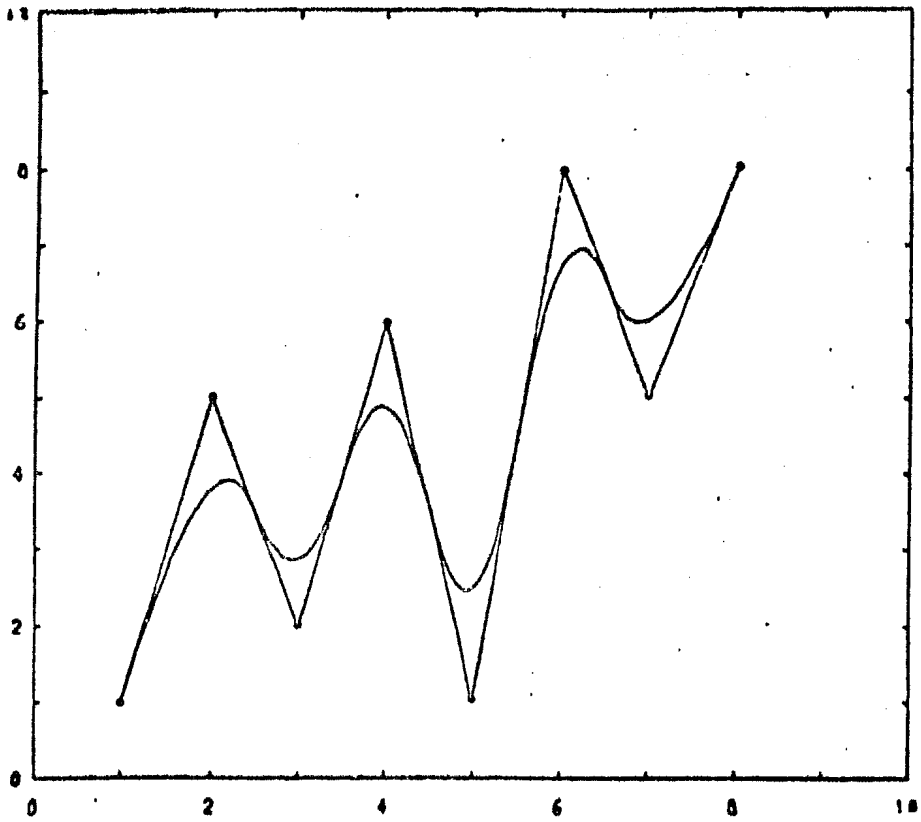

d

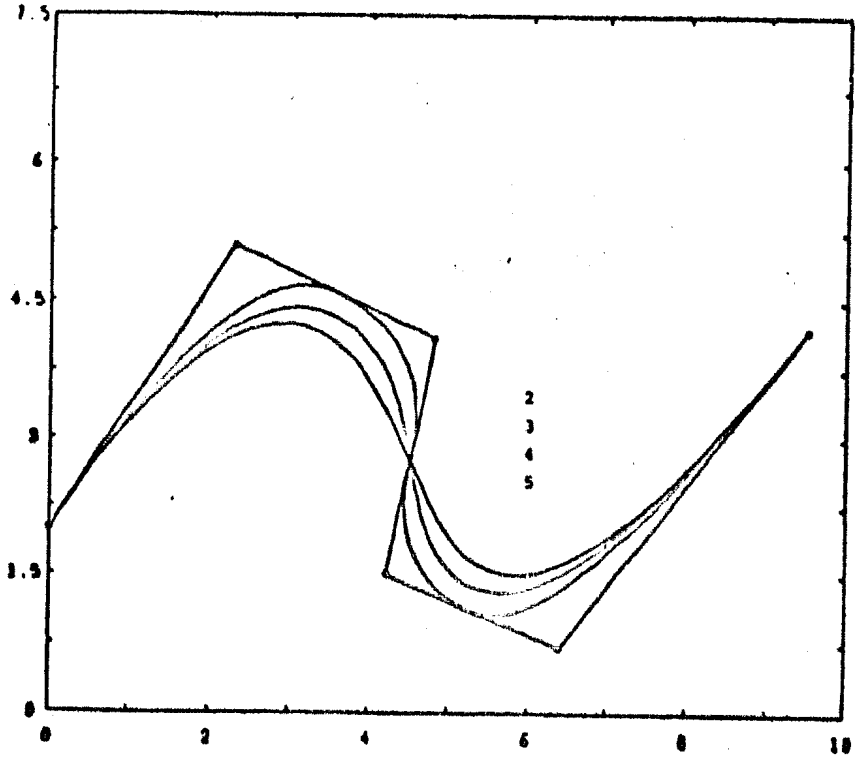

c

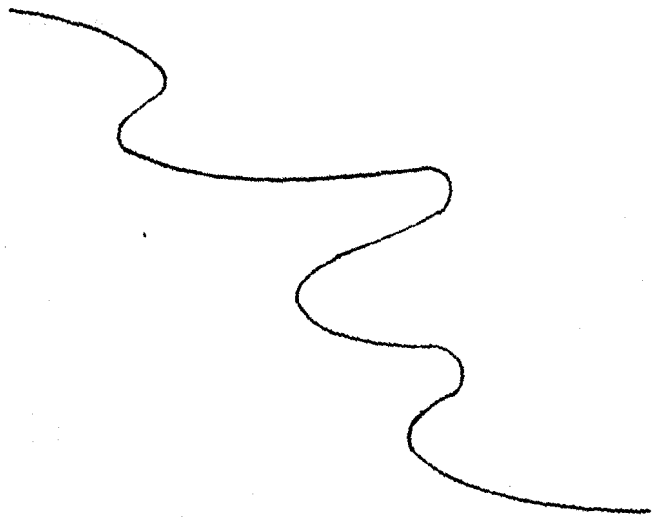

a

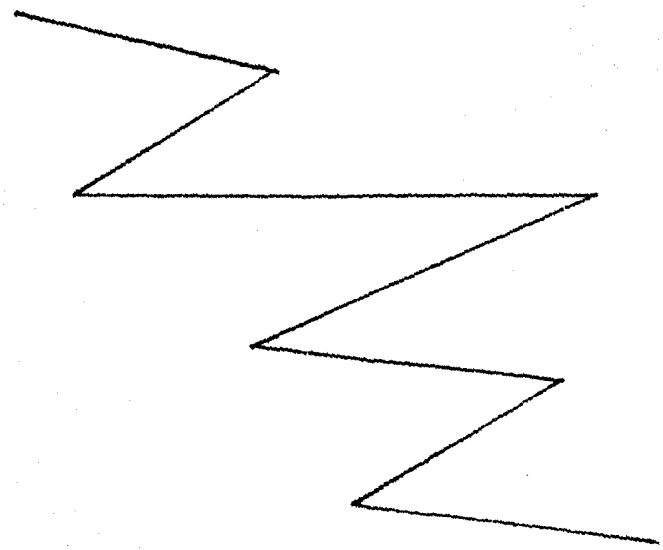

b

METODO DE B-SPLINES


0
2
4
6
8
10


0
1.5
3
4.5
6
0
2
4
6
8
10
2
3
4
5





P R O Y E C C I O N E S
(CALCOMP)

```
>@TRES0
>*  Dar archivo de la funcion: [S]: F0
>PIP F.FTN=F0.FTN
>FOR F=F
F
>* Compilo bien la funcion?? [Y/N]:Y
>*************************************************
>*
>*
>*Escoger alguna de las siguientes proyecciones:
>*
>*Proyeccion en Perspectiva      ......  GENPER
>*Proyeccion Ortografica        ......  GENORT
>*Proyeccion Oblicua            ......  GENOBL
>*Proyeccion Estereopar         ......  GENSTE
>*
>*
>*************************************************
>* Cual proyeccion  desea ? [S]: GENPER
>PIP FUNCION.FTN=GENPER.FTN
>FOR FUNCION=FUNCION
.MAIN.
>TKB @CONSTRUYE


>TKB
TKB>^Z

>* Se desea correr el programa?? [Y/N]:Y
>RUN TAREA
```

```
 HELL
ACCOUNT OR NAME: 310,20
PASSWORD:

        RSX-11M BL22    MULTI-USER SYSTEM

GOOD EVENING
03-APR-84 18:10 LOGGED ON TERMINAL TT7:

BIENVENIDOS AL LABORATORIO DE COMPUTACION  **SISTEMA RSX-11M**
>

 PIP TI:=F5.FTN
        FUNCTION F(X,Z)
        F=3*SIN(SQRT(X*X+Z*Z))
        RETURN
        END
>
>
>@DK2:[200,200]TRESD
>*  Dar archivo de la funcion: [S]: F5
>FOR F=F5
F
>* Compilo bien la funcion?? [Y/N]:Y
>;*************************************************
>;
>;
>;Escoger alguna de las siguientes proyecciones:
>;
>;Proyeccion en Perspectiva       ......  GENPER
>;Proyeccion Ortografica          ......  GENORT
>;Proyeccion Oblicua              ......  GENOBL
>;Proyeccion Estereopar           ......  GENSTE
>;
>;
>;*************************************************
>* Cual proyeccion  desea ? [S]: GENPER
>FOR FUNCION=DK2:[200,200]GENPER
.MAIN.
>TKB @DK2:[200,200]CONSTRUYE


>TKB
TKB>^Z

>* Se desea correr el programa?? [Y/N]:Y
```

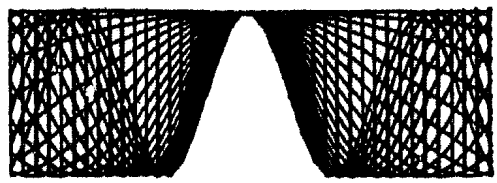
```
>RUN TAREA
  Dar las coordenadas de vision :   45.0,27.,39.
 Taman?"o de la fisura : 30.
 Numero de puntos de X,Z
 deben ser dos valores menores a 200  20,20
 Ranso de X : [XI,XS]  -12.0,12.0
 Ranso de Z :[ZI,ZS]  -12.0,12.0
Numero de pluma = 1
>PIP *.OBJ;*/DE
>@ <EOF>
>



>RUN TAREA
  Dar las coordenadas de vision :   45.0,27.0,39.0
 Taman?"o de la fisura : 20.
 Numero de puntos de X,Z
 deben ser dos valores menores a 200  30,30
 Ranso de X : [XI,XS]  -12.0,12.0
 Ranso de Z :[ZI,ZS]  -12.0,12.0
Numero de pluma =

1
>


>RUN TAREA
  Dar las coordenadas de vision :   45.0,27.0,39.0
 Taman?"o de la fisura : 20.
 Numero de puntos de X,Z
 deben ser dos valores menores a 200  60,60
 Ranso de X : [XI,XS]  -12.0,12.0
 Ranso de Z :[ZI,ZS]  -12.0,12.0
Numero de pluma = 1
>




>BYE
>
HAVE A GOOD EVENING
03-APR-84 18:56 TT7:  LOGGED OFF
>
>
>
```

```
>PIP TI:=F0.FTN
C
C
C       PROYECCION ORTOGRAFICA
C
C
C       NUNMERO DE PUNTOS : 40,40
C       INTERVALOS EN X,Z : -3.0,3.0
C       ANGULO: 45 GRADOS
C       FACTOR DE ESCALA =1.0
C
        FUNCTION F(X,Z)
        F=COS(X*Z)
        RETURN
        END
>
>PIP TI:=F1.FTN
C
C
C       PROYECCION OBLICUA CAVALIER
C
C
C
C       NUMERO DE PUNTOS : 40,40
C       INTERVALOS EN X,Z: -3.0,3.0
C       ANGULO: 30 GRADOS
C       FACTOR DE ESCALA :1.0
        FUNCTION F(X,Z)
        F=COS(X*Z)
        RETURN
        END
>PIP TI:=F2.FTN
C
C
C       PROYECCION OBLICUA CAVALIER
C
C
C       NUMERO DE PUNTOS EN X,Z : 40,40
C       INTERVALOS EN X,Z: -3.0,3.0
C       ANGULO : 45 GRADOS
C       FACTOR DE ESCALA : 1.0
C
        FUNCTION F(X,Z)
        F=COS(X*Z)
        RETURN
        END
>PIP TI:=F3.FTN
C
C
C       PROYECCION OBLICUA CABINET
C
C
C       NUMERO DE PUNTOS EN X,Z: 40,40
C       INTERVALOS EN X,Z: -3.0,3.0
C       ANGULO : 45 GRADOS
C       FACTOR DE ESCALA : 1.0
        FUNCTION F(X,Z)
        F=COS (X*Z)
        RETURN
        END
>
```

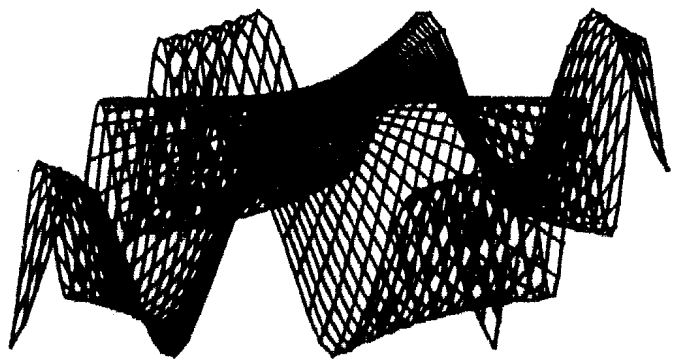
```
>PIP TI:=F4.FTN
C
C
C       PROYECCION EN PERSPECTIVA
C
C
C       NUMERO DE PUNTOS EN X,Z: 50,50
C       COORDENADAS DE VISION: 45.0,45.0,45.0
C       ESCALA: 50.0
C       INTRVALOS EN X,Z: -3.0,3.0
C
        FUNCTION F(X,Z)
        F=COS(X*Z)
        RETURN
        END
>PIP TI:=F5.FTN
C
C
C       PROYECCION EN PERSPECTIVA
C
C       NUMERO DE PUNTOS EN X,Z:50,50
C       COORDENADAS DE VISION: 45.0,0.0,45.0
C       ESCALA: 50.0
C       INTERVALOS: -3.0,3.0
        FUNCTION F(X,Z)
        F=COS(X*Z)
        RETURN
        END
>PIP TI:=F6.FTN
C
C
C       PROYECCION EN PERSPECTIVA
C
C
C       NUMERO DE PUNTOS EN X,Z: 50,50
C       COORDENADAS DE VISION : 45.0 , 25.0,90.
C       ESCALA : 50.0
C       INTERVALOS EN X,Z: -3.0,3.0
        FUNCTION F(X,Z)
        PI=3.1416
        F=Z*SIN(PI*X)
        RETURN
        END
>PIP TI:=F7.FTN
C
C
C       PROYECCION EN PERSPECTIVA
C
C
C       NUMERO DE PUNTOS :50,50
C       COORDENADAS DE VISION :
C       A) 45.0,45.0,45.0
C       B) 45.0,0.0,90.0
C       C) 45.0,25.0,60.0
C       ESCALA 80.0
C       INTERVALOS EN X,Z:-1.0,1.0
C
        FUNCTION F(X,Z)
        F=(X*X-Z*Z)/(X*X+Z*Z)
        RETURN
        END
```

```
>PIP TI:=F8.FTN
C
C
C         PROYECCION EN PERSPECTIVA
C
C
C         NUMERO DE PUNTOS EN X,Z:50,50
C         COORDENADAS DE VISION : 45.0,25.0,60.0
C         ESCALA : 200.0
C         INTERVALOS EN X,Z: -1.0,1.0
C
          FUNCTION F(X,Z)
          F=(X**X-Z*Z)/(X**X+Z*Z)
          RETURN
          END
>PIP TI:=F9.FTN
C
C
C         PROYECCION EN PERSPECTIVA
C
C
C         NUMERO DE PUNTOS EN X,Z: 10,10
C         COORDENADAS DE VISION : 25.0,20.0,12.0
C         ESCALA : 50.0
C         INTERVALOS EN X,Z: -2.0,2.0
C
C
          FUNCTION F(X,Z)
          F=(3*X**X**X**X-4*X**X**X-12*X**X+18)/(12*(1+4*Z*Z))
          RETURN
          END
>PIP TI:=F10.FTN
C
C
C         PROYECCION EN PERSPECTIVA
C
C
C         NUMERO DE PUNTOS EN X,Z:60,60
C         COORDENADAS DE VISION : 25.0,25.0 ,12.0
C         ESCALA : 70.0
C         INTERVALOS EN X,Z: -2.0,2.0
C
          FUNCTION F(X,Z)
          F=(3*X**X**X**X-4*X**X**X-12*X**X=18)/(12*(1+4*Z*Z)
          RETURN
          END
>PIP TI:=F11.FTN
C
C
C         PROYECCION EN PERSPECTIVA
C
C
C         NUMERO DE PUNTOS EN X,Z: 50,50
C         COORDENADAS DE VISION : 45.0,45.0,45.0
C         ESCALA: 200.0
C         INTERVALOS EN X,Z: -3.0,3.0
          FUNCTION F(X,Z)
          F=(X**X-Z*Z)/(X**X+Z*Z)
          RETURN
          END
>
```

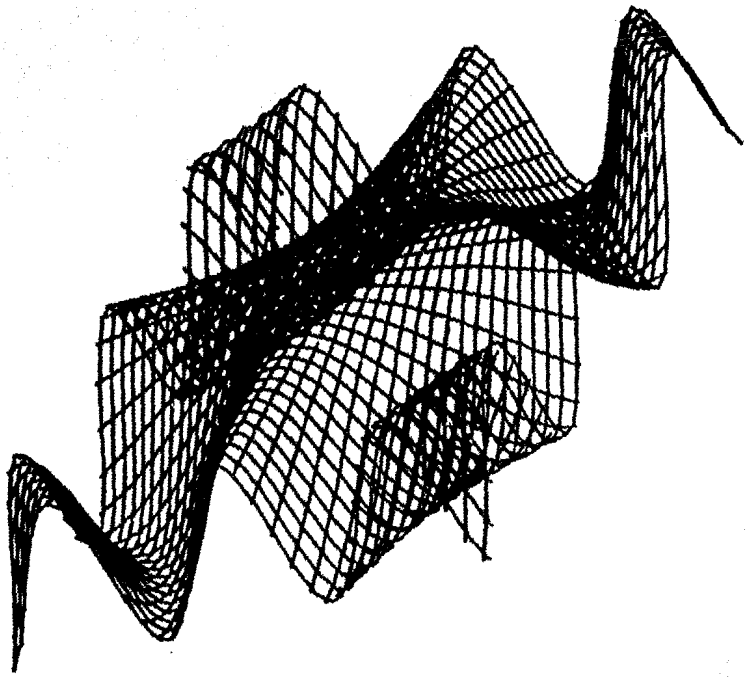
```
 0:  F0.FTN
 1:  F1.FTN
 2:  F2.FTN
 3:  F3.FTN
 4:  F4.FTN
 5:  F5.FTN
 6:  F6.FTN
 7:  F7.FTN
 8:  F8.FTN
 9:  F9.FTN
10:  F10.FTN
11:  F11.FTN
```



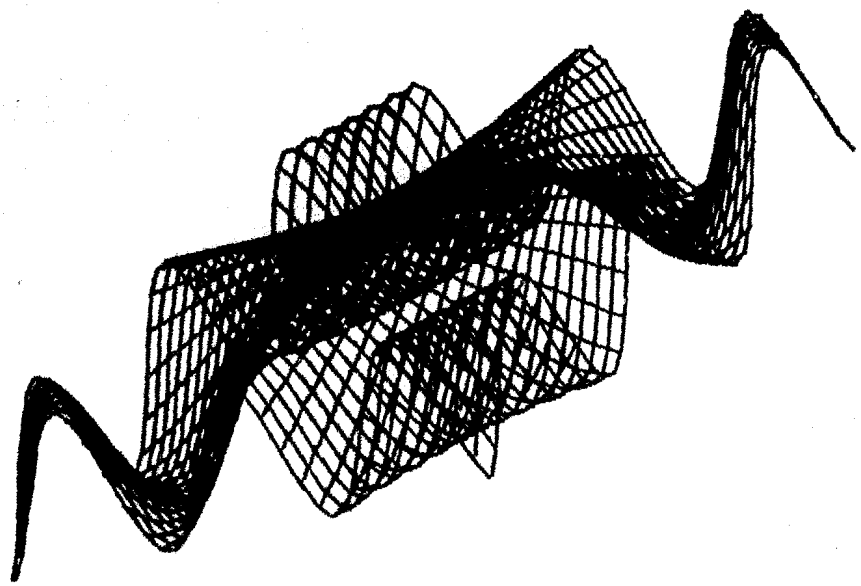
0



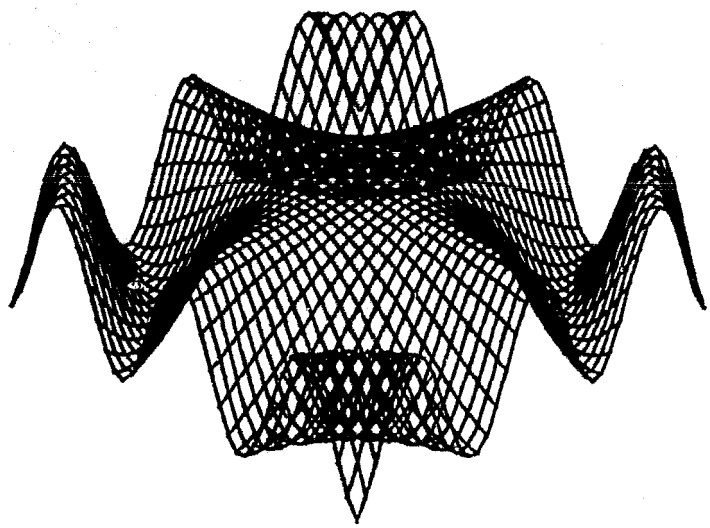
1



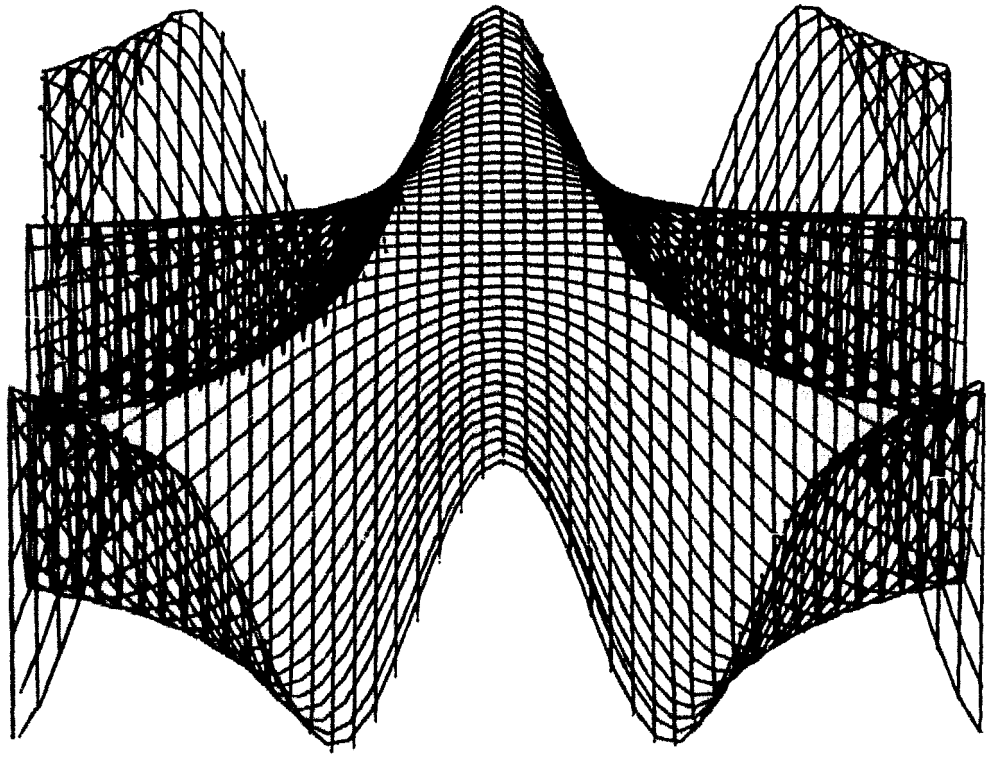
2



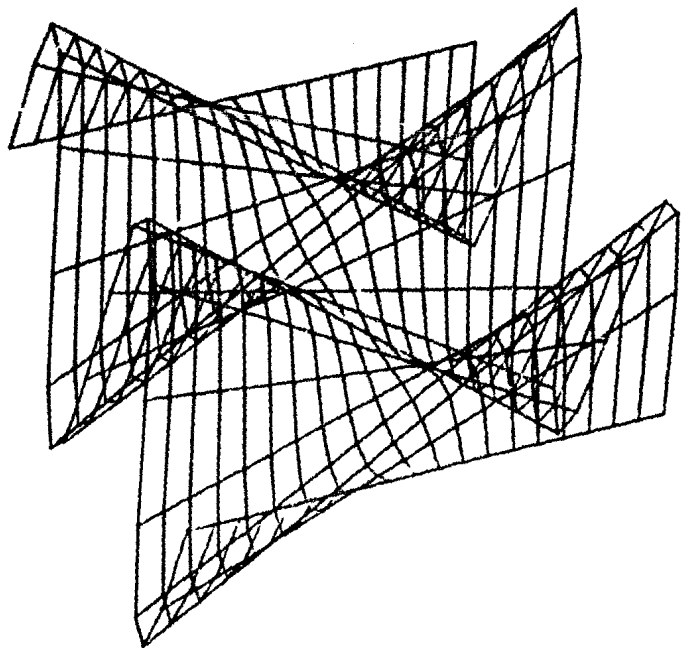
3

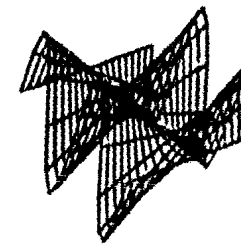


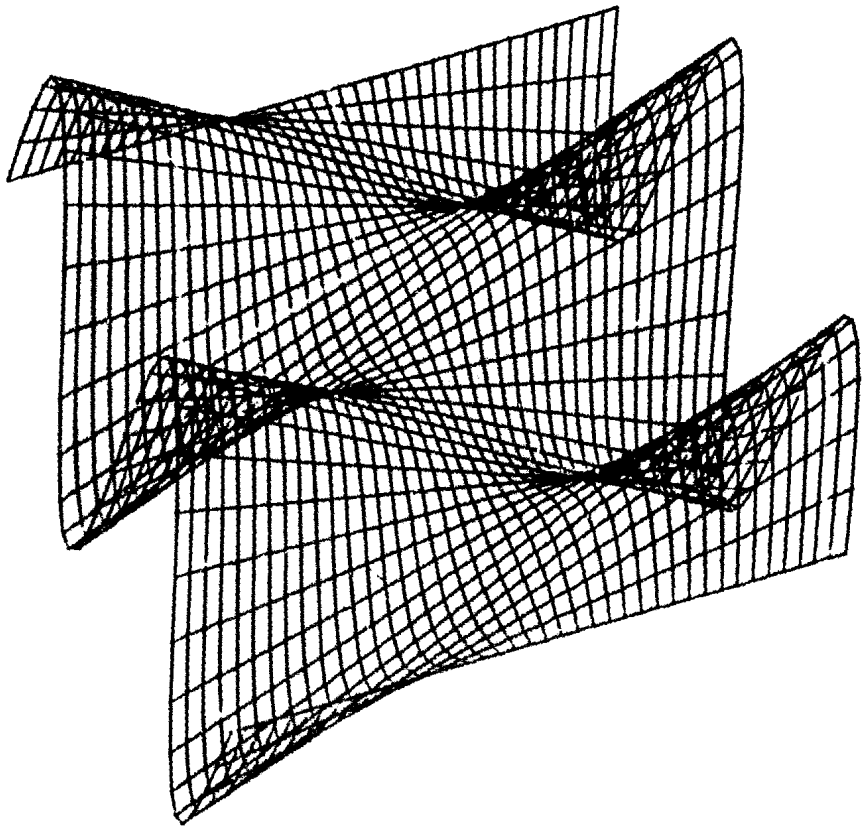
4

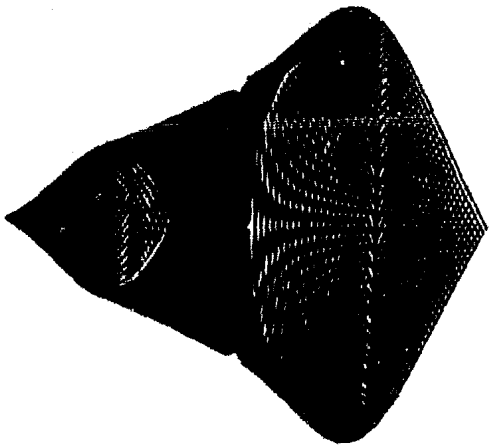


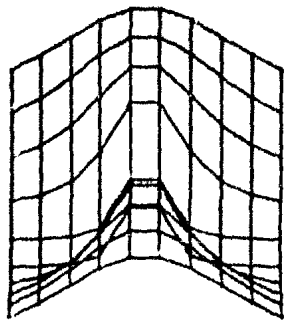
5



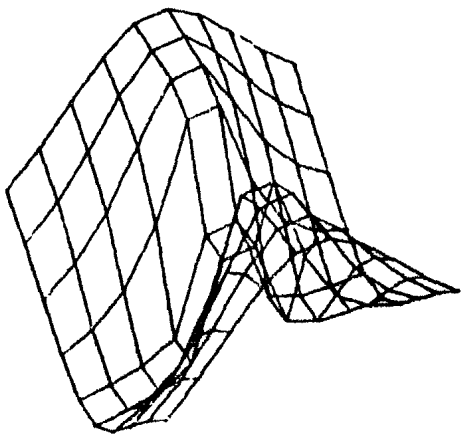


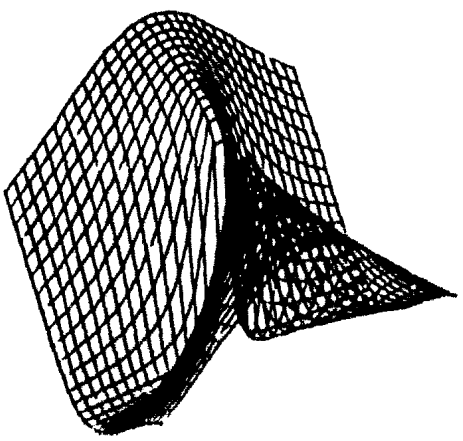


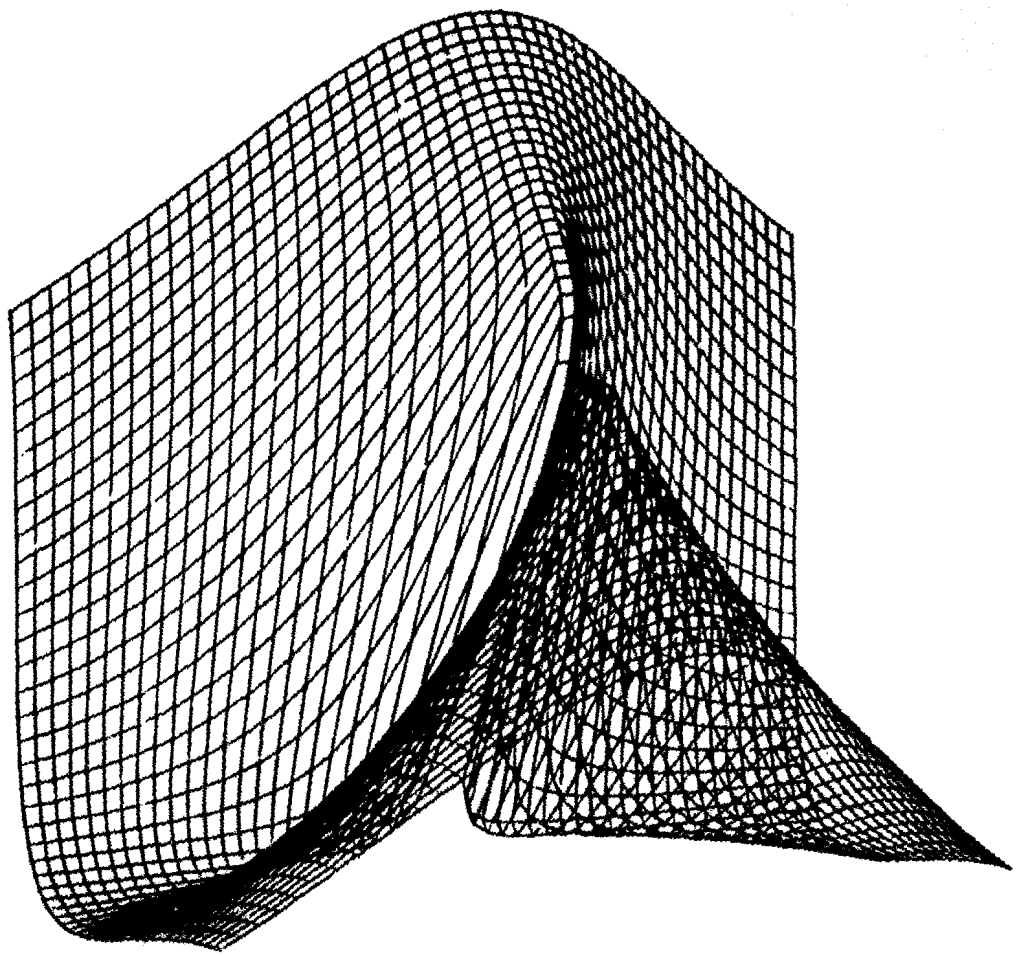


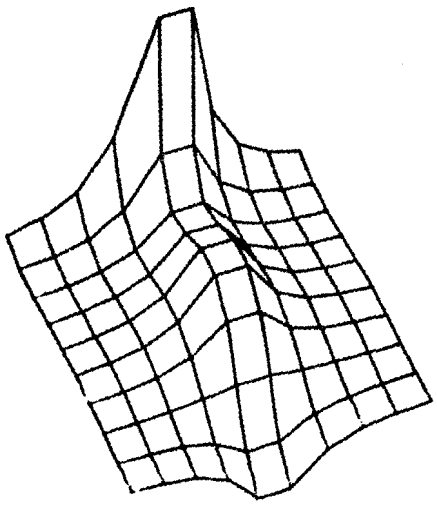


7

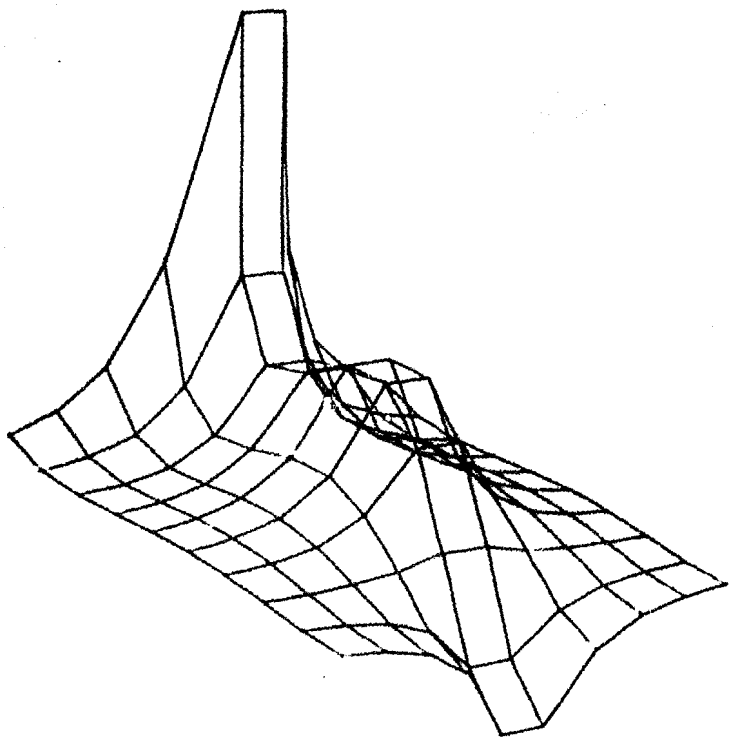


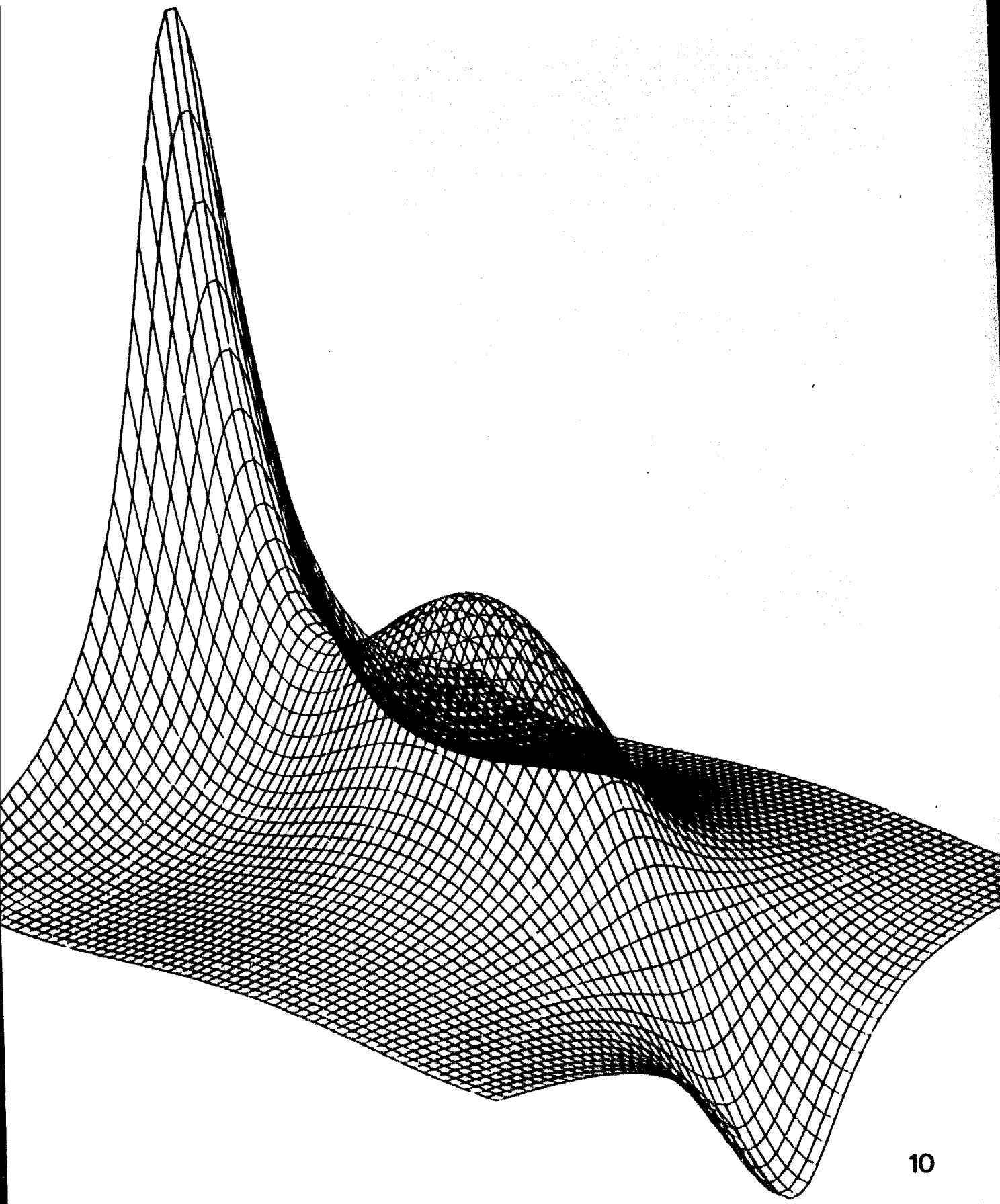


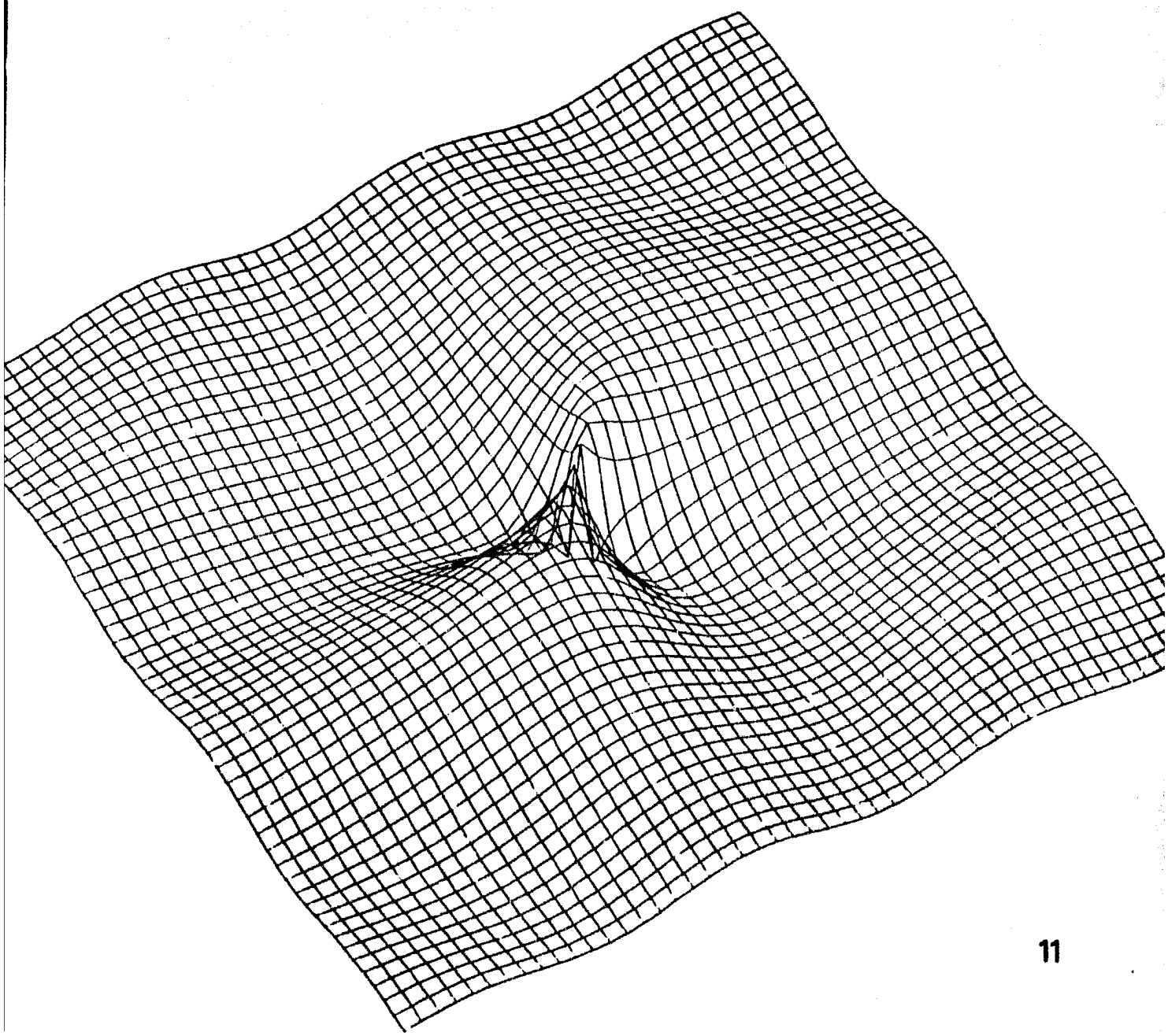




9







S U P E R F I C I E S C O N L I N E A S O C U L T A S

(TEKTRONIX)

```
0:  AS/FUNCION/PANTALLA
1:  AS/FUNCION/1/PANTALLA
2:  AS/FUNCION/2/PANTALLA
3:  AS/FUNCION/3/PANTALLA
4:  AS/FUNCION/4/PANTALLA
5:  AS/FUNCION/5/PANTALLA
6:  AS/FUNCION/6/PANTALLA
7:  AS/FUNCION/7/PANTALLA
8:  AS/FUNCION/8/PANTALLA
9:  AS/FUNCION/9/PANTALLA
```

```
$SET AUTOBIND
$BIND=FROM_OBJECT/AS/BIBLIOTECA/PANTALLA,*SERVICIO/AST/=;
$RESET_FREE
FILE   5=FILE5,UNIT=REMOTE,RECORD=22,IO
FILE   88=ECHO,UNIT=REMOTE,IO
C
C
C      INTERVALOS: -3.0,3.0
C      VENTANA   : -15.0,-15.0,40.,40.
C
C
      DIMENSION V(30,30)
      CALL TRESD(V,30)
      CALL EXIT
      END
      FUNCTION F(X,Z)
      F=COS(X*Z)
      RETURN
      END
```

AS/FUNCION/1/PANTALLA (11/06/84) 6:16 PM WEDNESDAY, NOVEMBER 7, 1984

```
$SET AUTOBIND
$BIND=FROM_OBJECT/AS/BIBLIOTECA/PANTALLA,*SERVICIO/AST/=;
$RESET_FREE
FILE   5=FILE5,UNIT=REMOTE,RECORD=22,IO
FILE   88=ECHO,UNIT=REMOTE,IO
C
C
C      INTERVALOS: -12.0,12.0
C      VENTANA   : -10.0,-10.0,45.0,45.0
C
C
      DIMENSION V(30,30)
      CALL TRESD(V,30)
      CALL EXIT
      END
      FUNCTION F(X,Z)
      F=3*SIN(SQRT(X*X+Z*Z))
      RETURN
      END
```

```
100     $BIND=FROM OBJECT/AS/BIBLIOTECA/PANTALLA,*SERVICIO/AST/=;
200     $RESET FREE
300     FILE  5=FILE5,UNIT=REMOTE,RECORD=22,IO
400     FILE  88=OCHO,UNIT=REMOTE,IO
410     C
420     C
430     C       INTERVALOS: -3.0,3.0
440     C       VENTANA    : -15.0,-15.0,50.,50.
450     C
460     C
470             DIMENSION V(30,30)
500             CALL TREED(V,30)
600             CALL EXIT
700             END
800             FUNCTION F(X,Z)
900             F=(X*Z)/(SQRT(X*X+Z*Z))
1000            RETURN
1100            END
```

AS/FUNCION/5/PANTALLA (11/05/84) 6:18 PM WEDNESDAY, NOVEMBER 7, 1984

```
100     $SET AUTOBIND
200     $BIND=FROM OBJECT/AS/BIBLIOTECA/PANTALLA,*SERVICIO/AST/=;
300     $RESET FREE
400     FILE  5=FILE5,UNIT=REMOTE,RECORD=22,IO
410     FILE  88=OCHO,UNIT=REMOTE,IO
420     C
430     C
440     C       INTERVALOS:-3.0,3.0
450     C       VENTANA    :-15.0,-15.0,40.,40.
460     C
470     C
500             DIMENSION V(30,30)
600             CALL TREED(V,30)
700             CALL EXIT
800             END
900             FUNCTION F(X,Z)
1000            R=SQRT(X*X+Z*Z)
1010            PI=3.1416
1012            F=(SIN(PI*R))/PI*R
1100            RETURN
1200            END
```

```
100     $SET AUTOBIND
200     $BIND=FROM OBJECT/AS/BIBLIOTECA/PANTALLA,*SERVICIO/AST/=;
300     $RESET FREE
400     FILE   5=FILE5,UNIT=REMOTE,RECORD=22,IO
410     FILE   88=ECHO,UNIT=REMOTE,IO
420     C
430     C
440     C       INTERVALOS: -3.0,3.0
450     C       VENTANA   : -15.0,-15.0,40.,40.
460     C
470     C
500             DIMENSION V(30,30)
600             CALL TRESD(V,30)
700             CALL EXIT
800             END
900             FUNCTION F(X,Z)
950             PI=3.1416
1000            F=10*SIN(PI*X)/(1+Z*Z)
1100            RETURN
```

```
00000100
00000200
00000300
00000400
00000410
00000420
00000430
00000440
00000450
00000460
00000470
00000500
00000600
00000700
00000800
00000900
00000950
00001000
00001100
```



```
100     $SET AUTOBIND
200     $BIND=FROM OBJECT/AS/BIBLIOTECA/PANTALLA,*SERVICIO/AST/=;
300     $RESET FREE
400     FILE   5=FILE5,UNIT=REMOTE,RECORD=22,IO
410     FILE   88=ECHO,UNIT=REMOTE,IO
420     C
430     C
440     C       INTERVALOS: -3.0,3.0
450     C       VENTANA   : -15.0,-15.0,40.,40.
460     C
470     C
500             DIMENSION V(30,30)
600             CALL TRESD(V,30)
700             CALL EXIT
800             END
900             FUNCTION F(X,Z)
910             F=(X*Z*(X*X-Z*Z))/(X*X+Z*Z)
1100            RETURN
1200            END
```

```
00000100
00000200
00000300
00000400
00000410
00000420
00000430
00000440
00000450
00000460
00000470
00000500
00000600
00000700
00000800
00000900
00000910
00001100
00001200
```

```
100     $SET AUTOBIND                                                    00000100
200     $BIND=FROM OBJECT/AS/BIBLIOTECA/PANTALLA,*SERVICIO/AST/=;        00000200
300     $RESET FREE                                                      00000300
400     FILE  5=FILES,UNIT=REMOTE,RECORD=22,IO                           00000400
410     FILE  6=OCHO,UNIT=REMOTE,IO                                      00000410
411     C INTERVALOS: -3.<X<3.  -3.<Z<3.                                 00000411
412     C VENTANA: -20.,-20.,50.,50.                                     00000412
500           DIMENSION V(30,30)                                         00000500
600           CALL TRESD(V,30)                                           00000600
700           CALL EXIT                                                  00000700
800           END                                                        00000800
900           FUNCTION F(X,Z)                                            00000900
1000          R=X*X*X-3*X                                                00001000
1010          R1=1+Z*Z                                                   00001010
1012          F=R/R1                                                     00001012
1100          RETURN                                                     00001100
1200          END                                                        00001200
```

6:19 PM WEDNESDAY, NOVEMBER 7, 1984

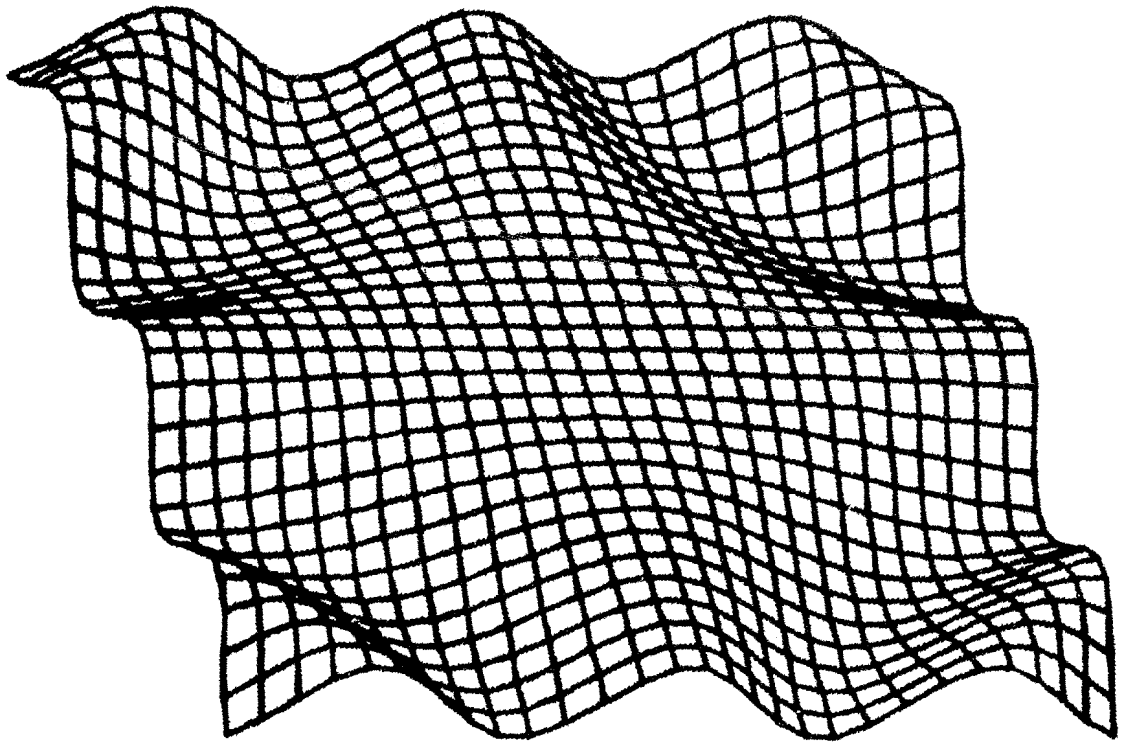
S/FUNCION/7/PANTALLA (1)(05/84)

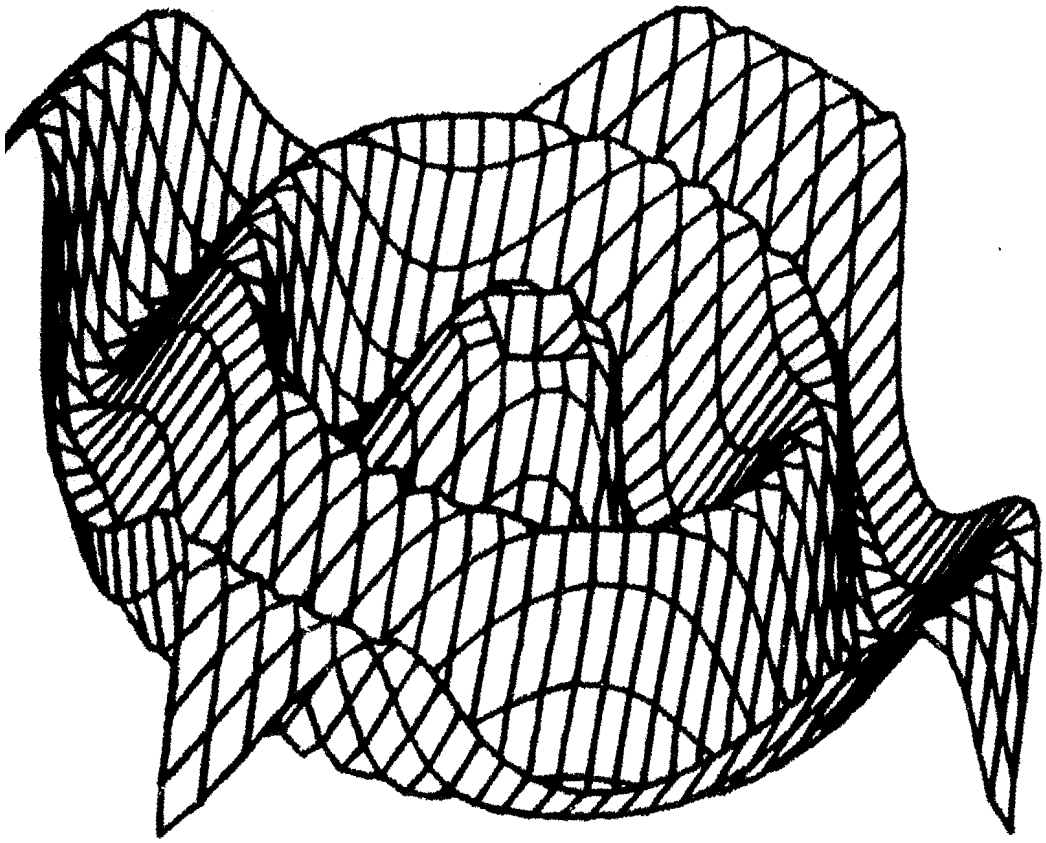
```
100     $SET AUTOBIND                                                    00000100
200     $BIND=FROM OBJECT/AS/BIBLIOTECA/PANTALLA,*SERVICIO/AST/=;        00000200
300     $RESET FREE                                                      00000300
400     FILE  5=FILES,UNIT=REMOTE,RECORD=22,IO                           00000400
410     FILE  6=OCHO,UNIT=REMOTE,IO                                      00000410
420     C                                                                00000420
430     C                                                                00000430
440     C       INTERVALOS: -3.0,3.0                                     00000440
450     C       VENTANA   : -15.0,-15.0,40.,40.                          00000450
460     C                                                                00000460
470     C                                                                00000470
500           DIMENSION V(30,30)                                         00000500
600           CALL TRESD(V,30)                                           00000600
700           CALL EXIT                                                  00000700
800           END                                                        00000800
900           FUNCTION F(X,Z)                                            00000900
1000          F=12*(X*X+Z*Z)*EXP(-X*X-Z*Z)                               00001000
1100          RETURN                                                     00001100
1200          END                                                        00001200
```

```
100     $SET AUTOBIND                                                          00000100
200     $BIND=FROM OBJECT/AS/BIBLIOTECA/PANTALLA,*SERVICIO/AST/=;              00000200
300     $RESET FREE                                                            00000300
400     FILE   5=FILE5,UNIT=REMOTE,RECORD=22,IO                                00000400
410     FILE   88=OCHO,UNIT=REMOTE,IO                                          00000410
420     C                                                                      00000420
430     C                                                                      00000430
440     C       INTERVALOS: -3.0,3.0                                           00000440
450     C       VENTANA   : -20.0,-20.0,40.0,40.0                              00000450
460     C                                                                      00000460
470     C                                                                      00000470
500             DIMENSION V(30,30)                                             00000500
600             CALL TRESD(V,30)                                               00000600
700             CALL EXIT                                                      00000700
800             END                                                            00000800
900             FUNCTION F(X,Z)                                                00000900
1000            F=(SIN(Z*X*X+3*Z*Z))/(X*X+Z*Z)                                 00001000
1100            RETURN                                                         00001100
1200            END                                                            00001200
```

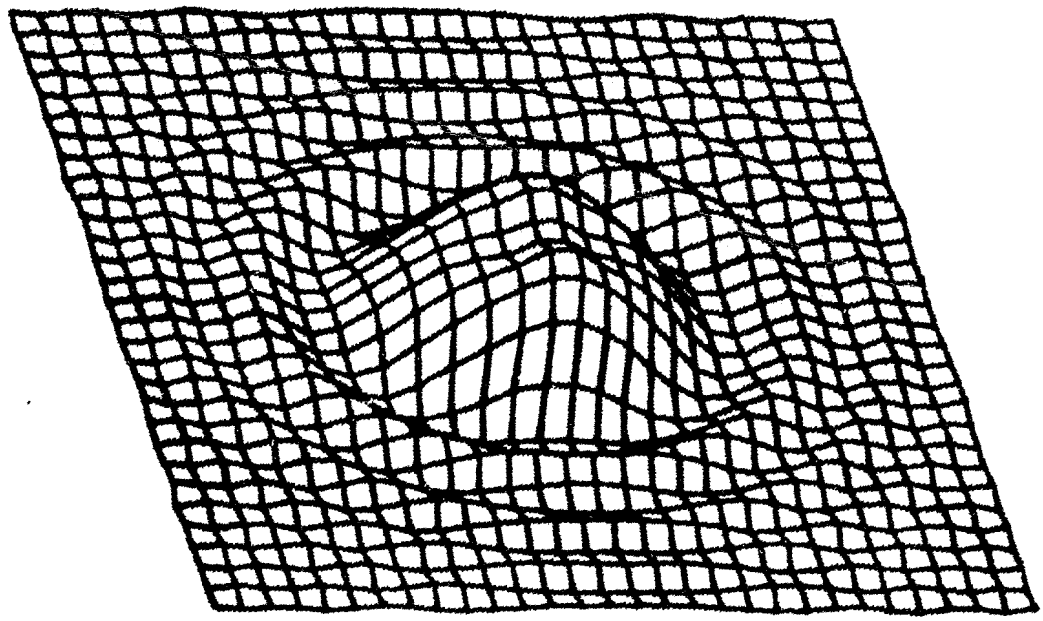

S/FUNCION/3/PANTALLA (1)/02/84)

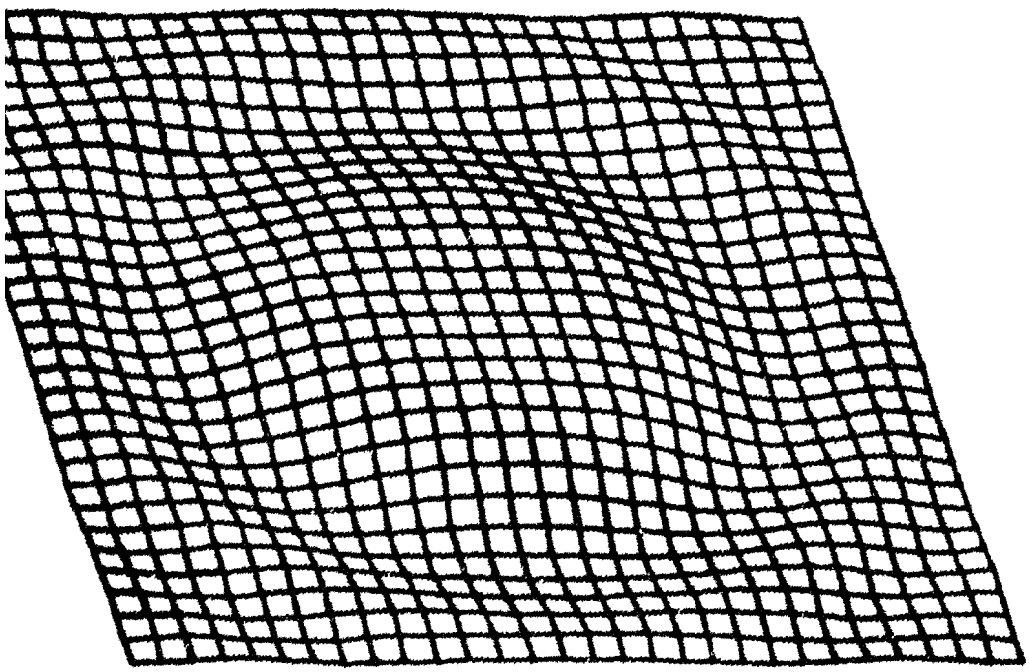
```
100     $SET AUTOBIND                                                          00000100
200     $BIND=FROM OBJECT/AS/BIBLIOTECA/PANTALLA,*SERVICIO/AST/=;              00000200
300     $RESET FREE                                                            00000300
400     FILE   5=FILE5,UNIT=REMOTE,RECORD=22,IO                                00000400
410     FILE   88=OCHO,UNIT=REMOTE,IO                                          00000410
420     C                                                                      00000420
430     C                                                                      00000430
440     C       INTERVALOS: -3.0,3.0                                           00000440
450     C       VENTANA   : -15.0,-15.0,40.,40.                                00000450
460     C                                                                      00000460
470     C                                                                      00000470
500             DIMENSION V(30,30)                                             00000500
600             CALL TRESD(V,30)                                               00000600
700             CALL EXIT                                                      00000700
800             END                                                            00000800
900             FUNCTION F(X,Z)                                                00000900
1000            F=(SIN(X*X+Z*Z))/(X*X+Z*Z)                                     00001000
1100            RETURN                                                         00001100
1200            END                                                            00001200
```

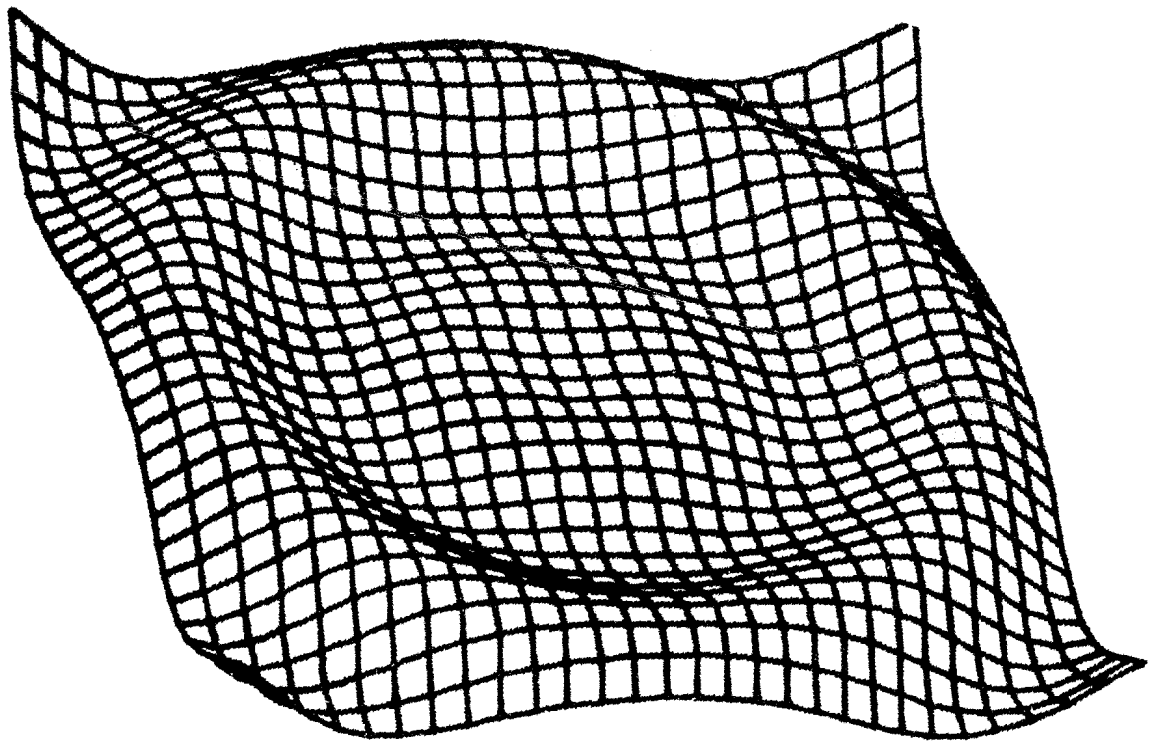


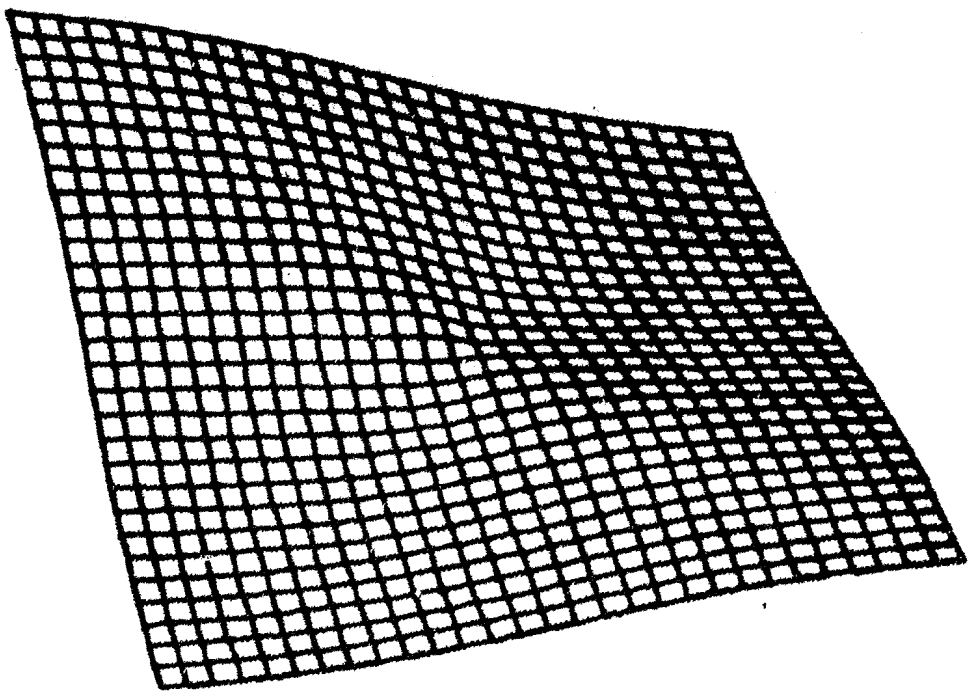


1

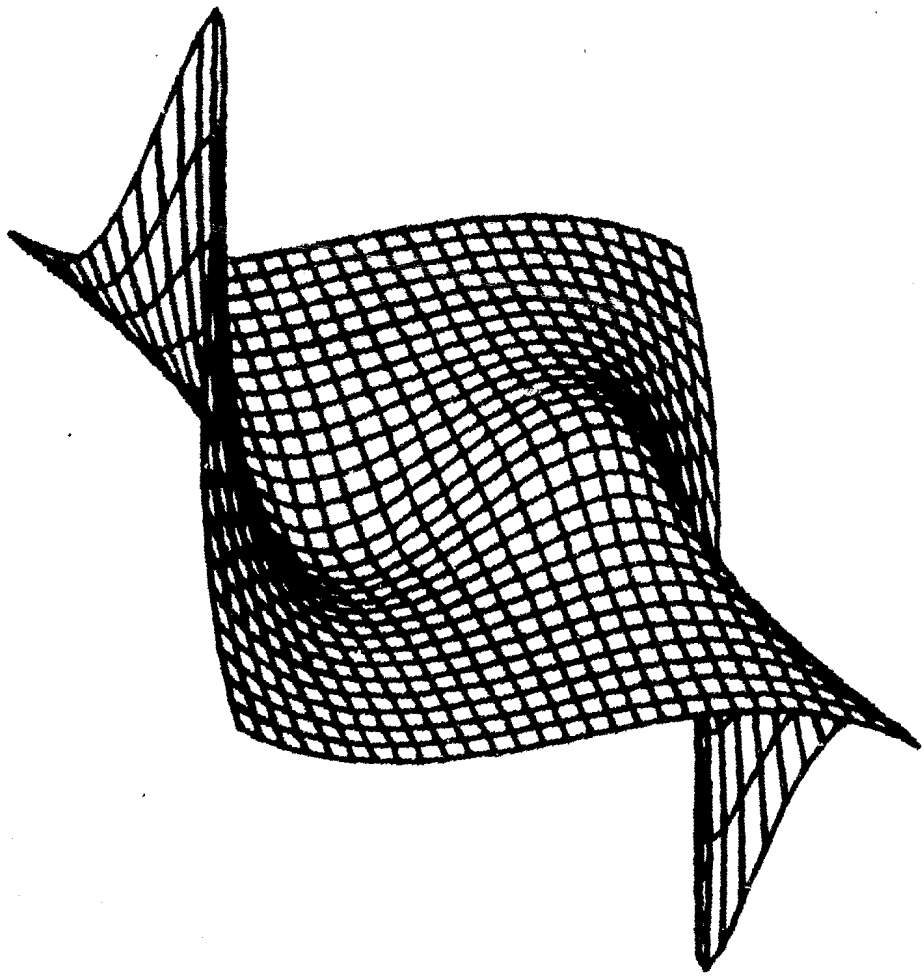




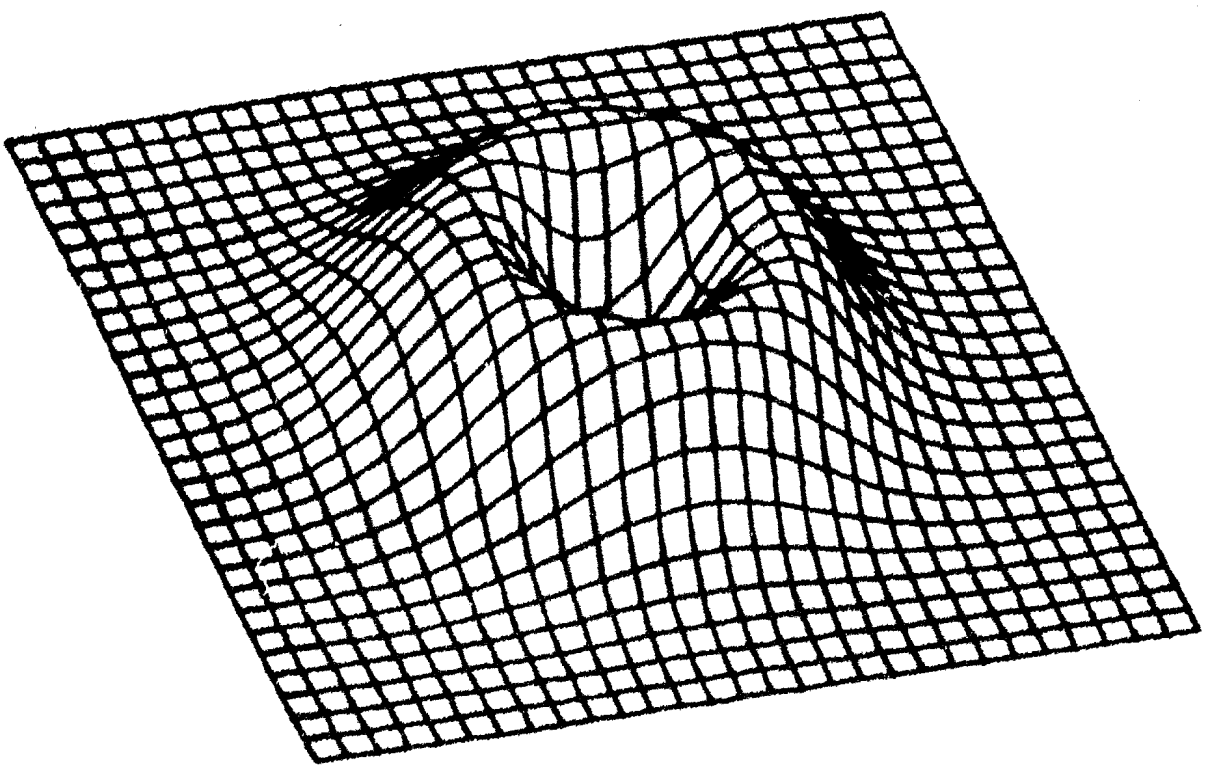




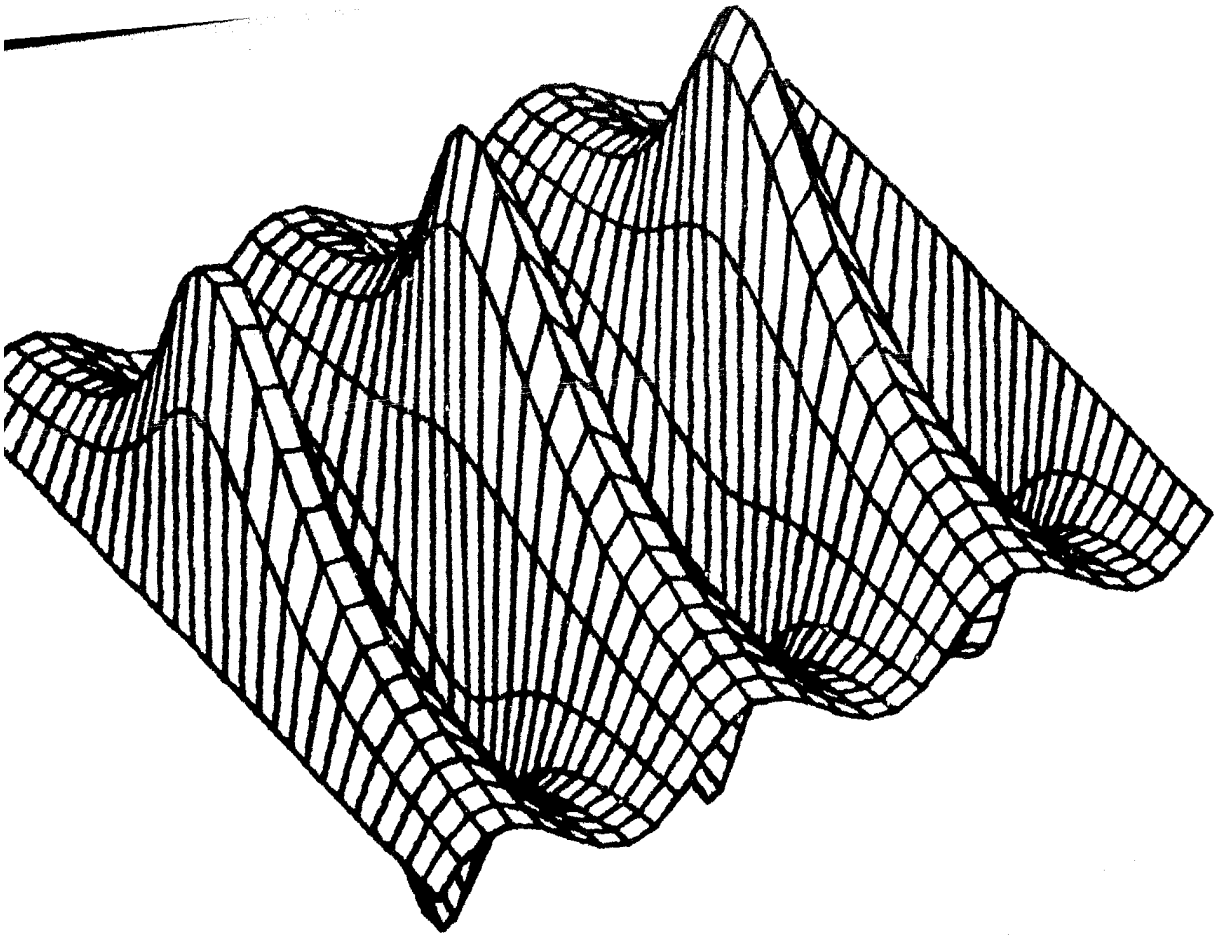
5



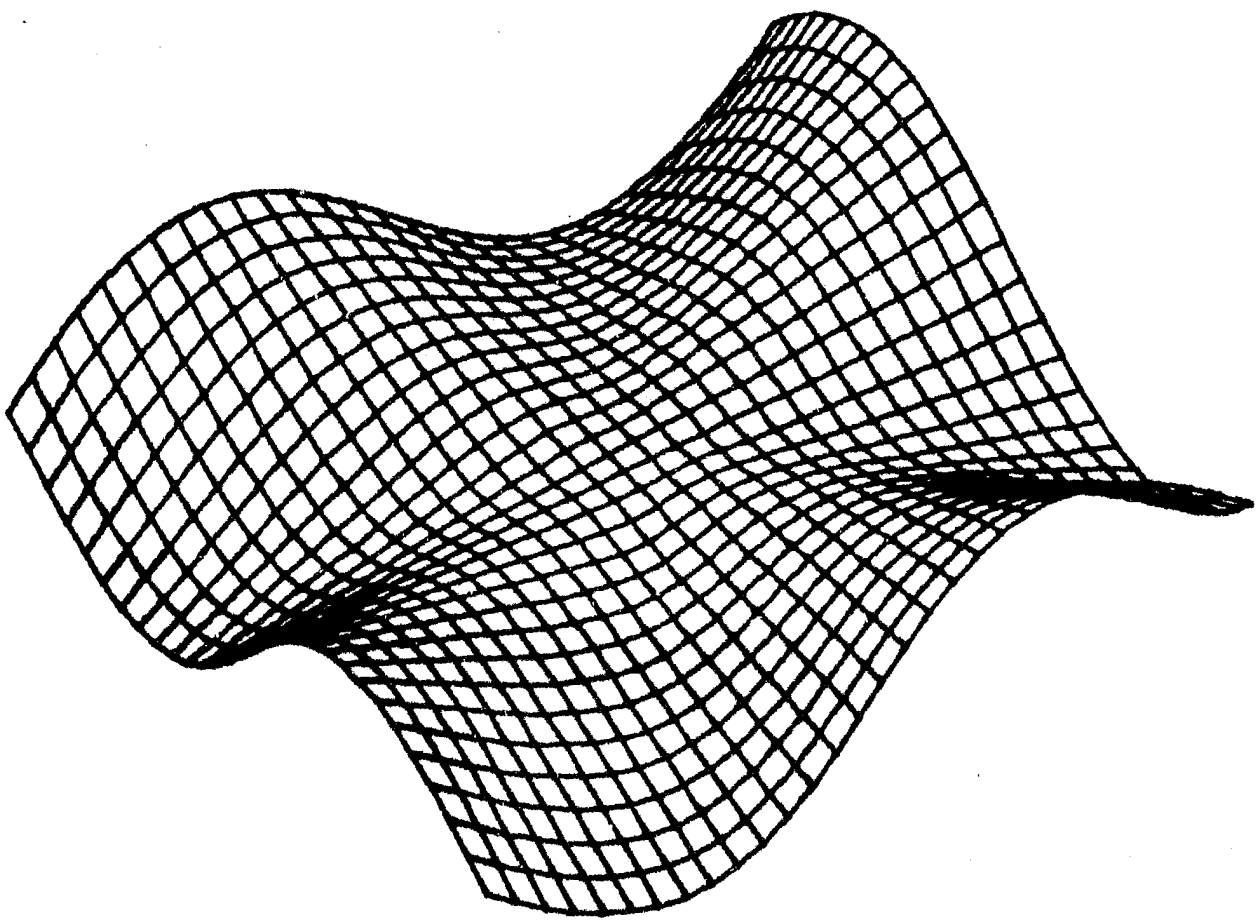
6



7



9



A:DESCRIPCION DEL GRAFICADOR CALCOMP MODELO 1012 Y DEL SOFTWARE BASICO.

Descripcion Funcional del Graficador Calcomp.

El graficador de rodillos Calcomp modelo 1012 , es un dispositivo de salida grafico electromecanico. El graficador 1012 tiene un controlador el cual permite ser conectado directamente a la computadora via una interface RS-232.

Las plumas estan montadas en un carril movible (manejado por un motor). El cilindro sirve para colocar el papel y tiene movimiento para deslizar el papel hacia la derecha o izquierda.

Un ventilador se encuentra colocado bajo el rodillo para mantener el papel firme en el rodillo.Cuando la pluma es bajada y el rodillo es movido , una linea es dibujada sobre el papel.

El carril se mueve en direccion del eje-Y y el rodillo en direccion del eje-X ;el movimiento simultaneo produce lineas diagonales.

El graficador dibuja con los pulsos que recibe la pluma.Cada movimiento de la pluma es el resultado de un comando que fue mandado por un programa o del panel de control del graficador. Una serie de comandos son usados para dibujar segmentos de linea.

Los segmentos de linea son dibujados atravez de las combinaciones de los movimientos del rodillo y del carril donde se encuentran las plumas.

Las combinaciones que se pueden hacer son ocho.Esto se ilustra en la Figura 1.

Todas las lineas y angulos que no sean estas , son una combinacion de estas ocho direcciones.

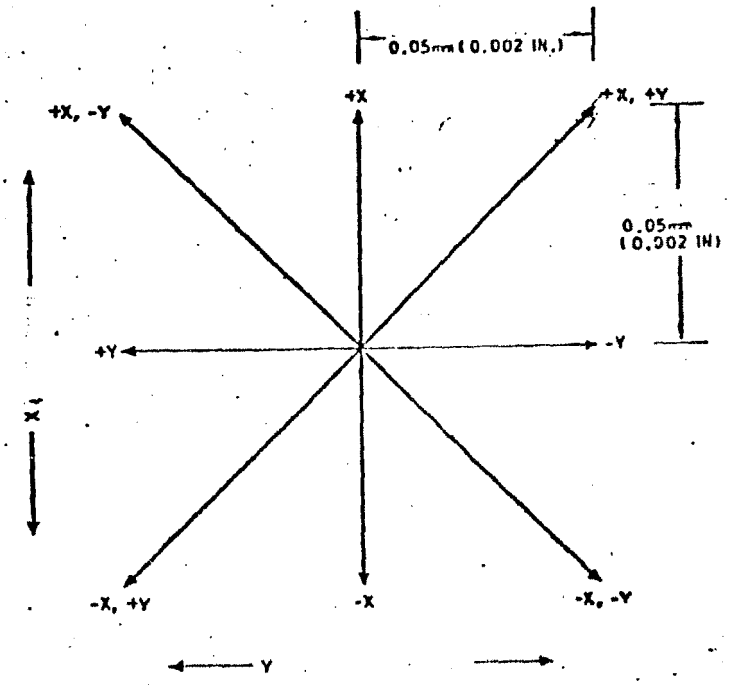
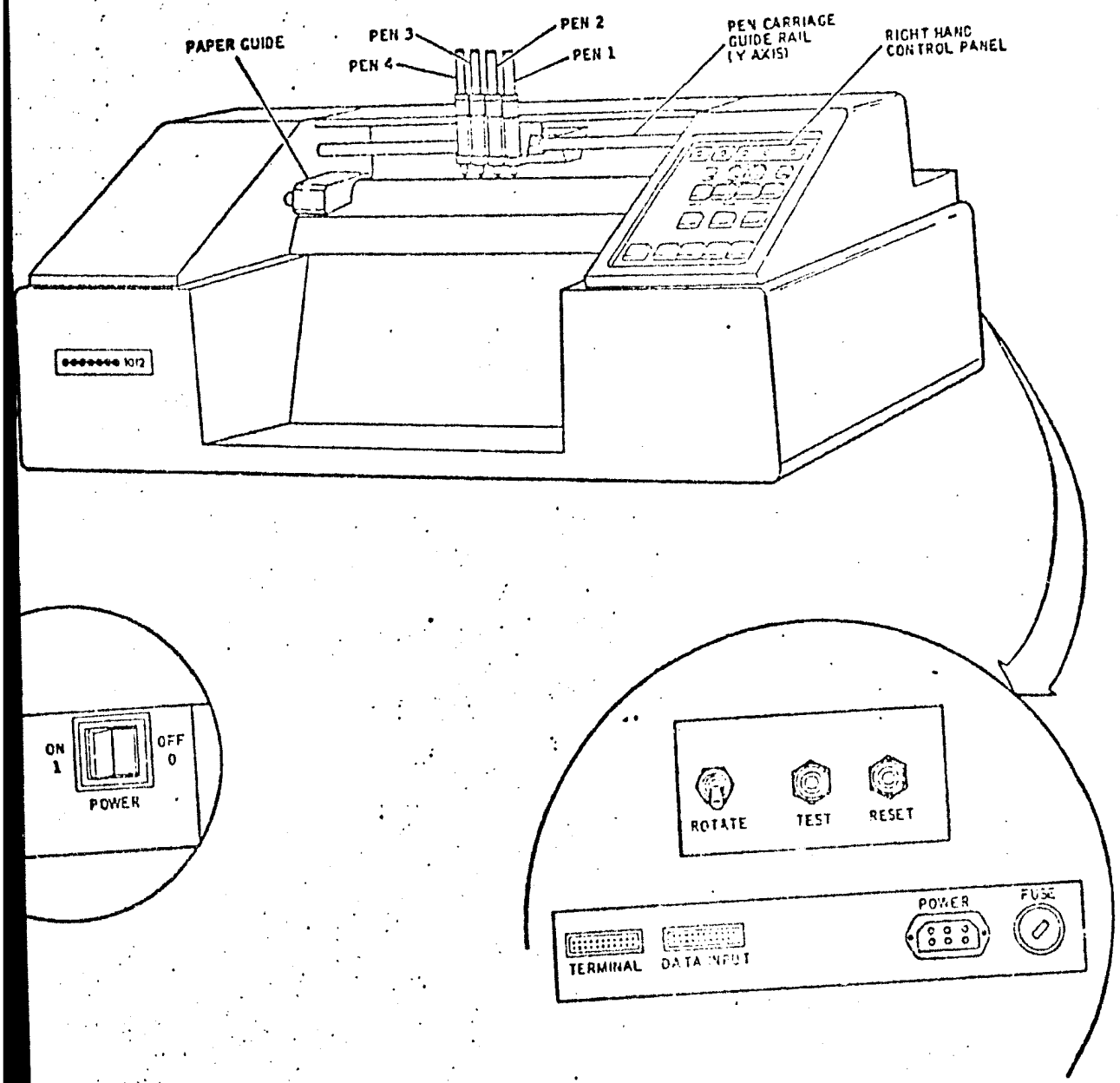


FIGURA 1



G R A F I C A D O R C A L C O M P M O D 1 0 1 2.

1. Caracteristicas Particulares.

*Servo motor DC.
*Motor para el manejo de las plumas.
*Cuatro plumas.
*Dos tipos de plumas:tipo nylon y tipo
 boligrafo.
*Interruptores para el manejo del eje-Y.
*Senal para el suministro de papel.
*Una tecla de RETURN para regresar a
 la ultima posicion que se grafico.
*Un conjunto de 96 caracteres.
*Un buffer de 256 bytes para entrada/salida.
*Un puerto para la terminal.
*Un generador de lineas.
*Un eje de rotacion.
*Un comando de indice.

2. Caracteristicas Opcionales

Interface en paralelo.
RS 232 - 20MA Current loop IEEE-488-1975

Resolucion.
0.05 mm (0.002 in)

Rapidez de Trazado.
Axial 254 mm/sec
Diagonal 359 mm/sec

Aceleracion.
0.75 G

Tipo de Papel.
Serie 200 f
Translucido Z-fold 200 hojas/paquete
11 in x 8.5 in Hojas (1700 in)
279.4 mm x 216 mm

Tamano de Papel.
11 in x 8.5 in
297 mm x 210 mm

Area de Graficacion.
11 in x 1700 in

Senal de Entrada.
RS-232
Baud-rate : 110-9600

OPERACION

El graficador tiene dos modos de operacion , el Manual y el Auto.

El modo Manual es usado para preparar al graficador a ser usado en modo Auto.

El modo Auto es usado para graficar los datos que vienen del dispositivo de entrada , en este caso la computadora.

Antes de encender el graficador hay que colocar el papel, instalar las plumas y hacer los ajustes de las plumas (si es necesario).

Para probar las plumas, que pinten bien y que el papel corra sin atorarse, hay que hacerlo en modo Manual.

Una luz colocada en el panel de control , indicara si se esta en modo Auto o en modo Manual. En modo Manual , la luz estara intermitente.Si la luz esta fija, se estara en modo Auto.

CONTROL DEL PANEL.

En el panel de control se tiene varios botones e indicadores con las siguientes funciones.

PEN SELECT

Se utiliza en modo manual. Sirve para bajar o subir las plumas de manera manual. Tambien puede utilizarse para probar las plumas.

PEN FORCE

Se debe seleccionar el ajuste para el tipo de pluma usada.

Tipo de Pluma	Fuerza Usada
Bolisrafo	HIGH
Nylon o boli-grafo	MED
Nylon	LOW

INDEX

Posiciona a la pluma 1 cerca del borde de -Y ,-X . Presionando +X o -X en combinacion con este boton y sosteniendolo medio segundo INDEX , se adelantara o se regresara respectivamente a la posicion senalada.

RETURN

La pluma regresa a la ultima posicion que tomo la pluma cuando se estaba en modo AUTO , Es solo valido durante el modo MANUAL. La senal RETURN no es valida cuando se mueve +Y o -Y o se interrumpa el modo de graficacion antes de que alcanze la ultima posicion de modo AUTO.

MODE

Es un indicador. Se presiona el boton para seleccionar el modo MANUAL o AUTO. Si la luz esta parpadeando, indica que esta en modo manual, no se perdera los datos, se puede regresar a modo AUTO apretando de nuevo el boton.

MANUAL MOVMENT

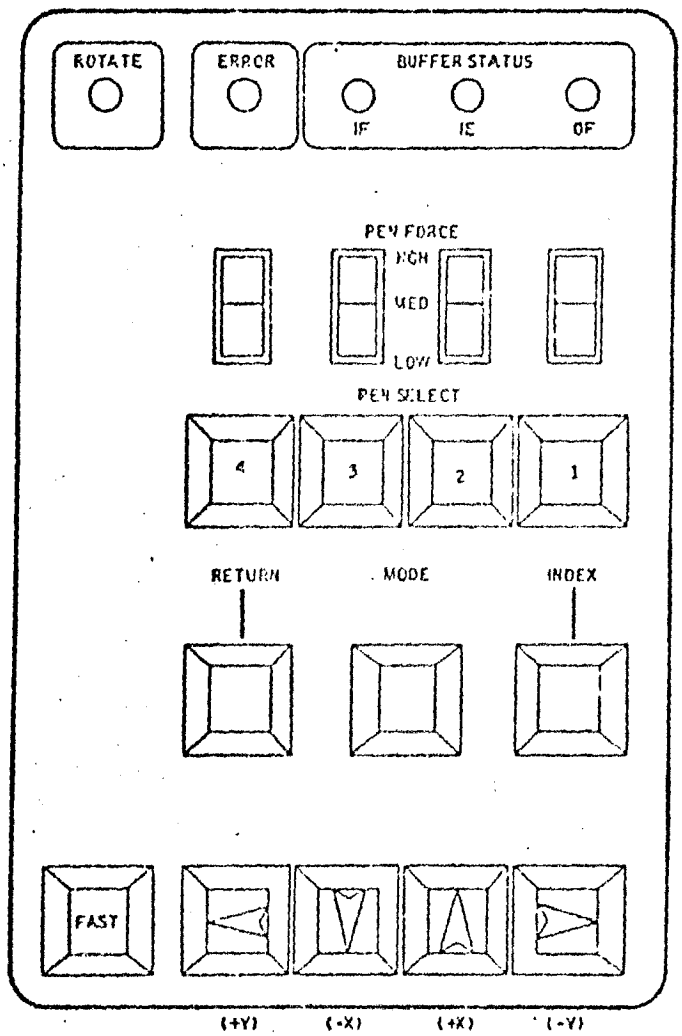
Presionando cualquier boton causara el incremento de movimiento en la direccion indicada (+Y ,-X,+X ,-Y).

Presionando y sosteniendo cualquiera de estos botones causara un movimiento lento en la direccion indicada.
Al presionar el boton FAST se incrementara la velocidad de movimiento.

ROTATE — Indicador.Se prende cuando el eje se ha rotado.

ERROR — Indica una condicion de error para la transmision de datos u otra funcion para el graficador.

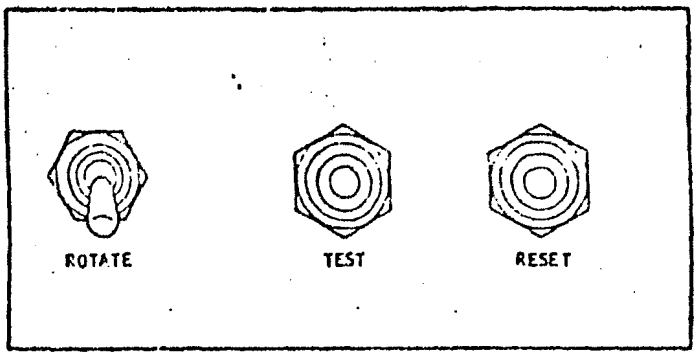
ESTADOS DEL BUFFER	INDICADORES
I F	Buffer de entrada lleno
I E	Buffer de entrada vacio
O F	Buffer de salida lleno



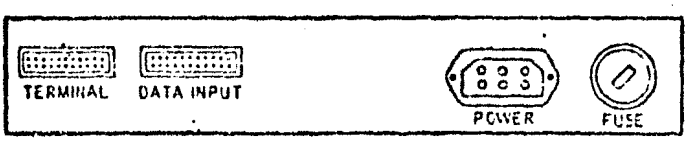
PANEL DE CONTROL DELANTERO

CONTROL TRASERO.

RESET	Al presionar se pasa al modo Manual. Se inicia toda parte electronica del graficador y se limpian los registros para un estado inicial.
TEST	Se da inicio a una serie de programas internos para probar el funcionamiento del graficador.
ROTATE	Se activa la rotacion , se rota 90 grados.
INPUT (DB25P)	Conector. Este conector es usado como un puerto de entrada serial RS232.
TERMINAL	Conector.Este conector es usado para pasar datos en serie como una salida o coneccion a una terminal.



C O N T R O L T R A S E R O



TERMINAL : En este lugar se conecta el cable que viene de la terminal.

DATA INPUT: En este lugar se conecta el cable que viene de la computadora

RUTINAS DE GRAFICACION DE CALCOMP

Introduccion.

El objetivo de esta seccion es presentar las rutinas que proporciona la compania Calcomp para la utilizar el Graficador.

La idea no es ver tecnicas de Graficacion , sino conocer como manejar y elaborar programas en Fortran para utilizar las rutinas que proporciona Calcomp.

Sin embarso, si hay que situarse para ver a que nivel se va a utilizar el Graficador dentro del area de la Graficacion.

Primero hay que mencionar que existe dos maneras de generar salidas graficas. Una seria la Graficacion Pasiva o Graficacion por Computadora ,que normalmente se denota por CG y la otra es la Graficacion Interactiva y se denota por IG.

Como el graficador con el que se cuenta , es un graficador de papel ,entra dentro de la clasificacion de la Graficacion Pasiva.

Sin embarso hay problemas tan generales que se pueden utilizar el mismo algoritmo tanto para la Graficacion Pasiva ,como en la Interactiva. Muchas veces se persigue que los programas hechos en un graficador puedan ser utilizados en otro .

Pero para llegar a elaborar programas de aplicacion, o sea programas hechos por los usuarios orientados a cierta aplicacion para obtener una salida grafica o pictorica, antes hay que conocer el graficador con el que se cuenta.

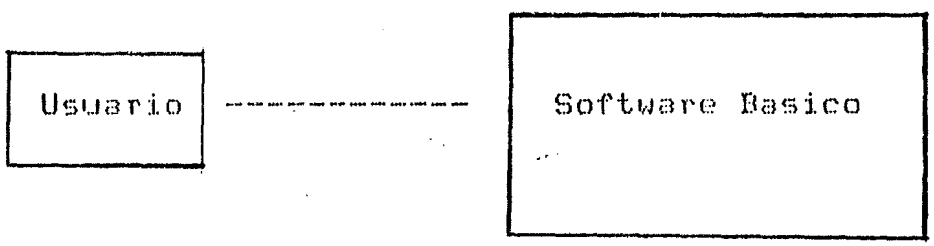
Para ello se puede hacer una clasificacion dependiendo del nivel de programacion en el cual este trabajando el usuario.

Esta clasificacion podria darse como sigue:

1.- Software Basico.

2.- Rutinas Generales de Graficacion.

3.- Programas de Aplicacion.



CLASIFICACION EN EL NIVEL DE PROGRAMACION EN GRAFICACION.

-SOFTWARE BASICO.

El software basico es un conjunto de rutinas que generan salida al controlador del sistema de graficacion.

El usuario no requiere de comunicarse con el hardware en su estructura de datos.

En lugar de esto se comunica con el conjunto de subrutinas tales como:

Mueve la pluma a coordenadas especificas.

Coloca algunos caracteres en cierta localidad en la area que se destino para graficar.

Dibuja los ejes con cierta notacion .

Escala y dibuja una linea atravez de una serie de puntos.

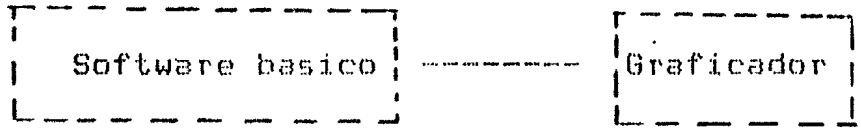
Al conjunto de rutinas que integran el software basico , tambien se le llaman primitivos graficos porque son rutinas elementales y basicas para el graficador como son el manejo de textos , movimiento de la pluma y trazado de lineas.

A este nivel , es en el que se va a conocer al graficador.

El usuario hace uso directo de las rutinas para crear su salida grafica

Asi la comunicacion se reduce al problema de formatear la salida grafica.

Los primitivos graficos estaran residentes en la computadora del usuario.



Manda señales al Graficador

·Rutinas Generales de Graficacion.

Ensesuida se tendria el siguiente nivel en el que se crean rutinas tan senerales que son creadas independientemente de cual sea la aplicacion que se vaya a realizar en el Graficador.

El sesundo nivel de software son rutinas que van a ser utilizadas por varias aplicaciones (Prosramas de aplicacion).

Estas rutinas son senerales , no estan orientadas a ninsuna especializacion . Alsunos ejemplos serian:

Funciones de Ventana.

Funciones de recorte de fisuras.

Metodos como Splines, Bezier,etc.

Perspectiva.

Lineas y Superficies Ocultas.

Transformacion en 2D y 3D.

Prosramas de Aplicacion.

Es el nivel mas alto de software. En este nivel,se elaboran las rutinas que utilizaran los dos niveles anteriores ,pero estas ya tendrian alsuna aplicacion especial.

Los usuarios pueden desarrollar sus propios prosramas de aplicacion. Alsunos ejemplos serian :

Un prosrama de proposito seneral para contornos.

Graficacion en tres dimensiones y vistas en perspectiva.

Graficacion para manejo de informacion.

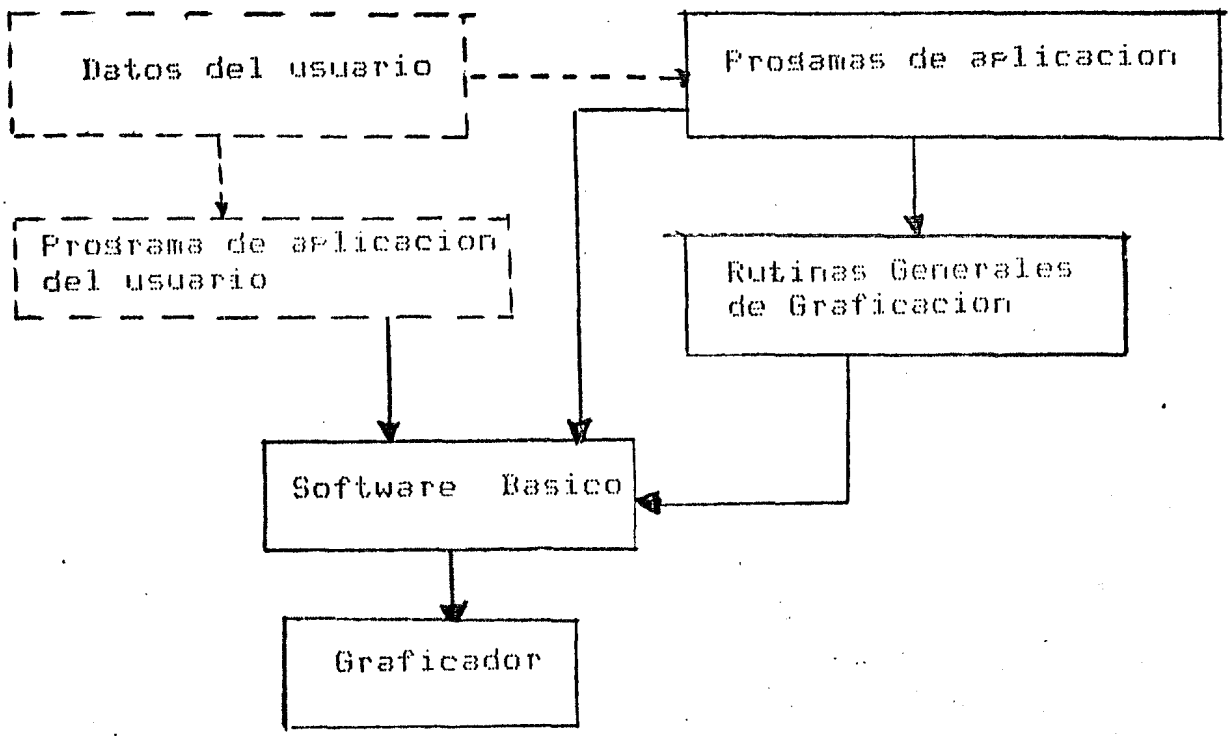
Graficacion de funciones con dos variables.

Histosramas

Manejo de distintos tipos de simbolos.

Animacion.

Ensenanza.



C L A S I F I C A C I O N E N E L N I V E L D E

P R O G R A M A C I O N E N G R A F I C A C I O N

El software basico que proporciona Calcomp consiste de un conjunto de rutinas escritas en FORTRAN , las cuales controlan las operaciones elementales del graficador y ademas agrega otras rutinas (no primitivos graficos), para facilitar la generacion de graficas en el plano XY.

Las subrutinas incluidas en el Software Basico son las siguientes:

PLOT — Grafica una linea entre dos puntos; establece el origen.
Tambien contine cuatro subrutinas auxiliares:

- PLOTS — Se dan las condiciones iniciales al graficador.
- FACTOR — Se da una escala global al dibujo.
- WHERE — Regresa la localidad de la pluma.
- NEWPEN — Selecciona la pluma deseada.

SYMBOL — Grafica textos y simbolos especiales.

NUMBER — Grafica el equivalente decimal de un numero en punto flotante.

SCALE — Determina el valor inicial y la escala para un arreglo de datos a ser graficados.

AXIS — Dibuja un eje con algun texto para una grafica.

LINE — Escala y grafica un conjunto de datos definidos en el plano XY.

OBTENIENDO LA SALIDA GRAFICA.

Etapas para ejecutar un programa utilizando el graficador.

Antes de poder ejecutar un programa usando las rutinas de CALCOMP, hay que seguir una serie de etapas.

Primero hay que usar el editor para crear el programa en FORTRAN.

Segundo, compilarlo con FOR.

Tercero, es la etapa del TKB , es necesario añadir la biblioteca donde se encuetran las rutinas de CALCOMP , y luego , ejecutarlo.

Las rutinas del graficador se encuentran en :

DK2:[200,200]PLOTLIB.OLB.

Los pasos a seguir son los siguientes:

1)
```
EDI FUN.FTN
-------------------
-------------------
-------------------
-------------------
CALL EXIT
  END
*ED
[EXIT]
```

2)
```
> FOR FUN=FUN
```

3)
```
>TKB
TKB>FUN=FUN,DK2:[200,200]PLOTLIB/LB/SS
TKB> /
ENTER OPTIONS
TKB>ASG=TT1:1
TKB>//
```

4)
```
>RUN FUN
```

Planeando la Salida Grafica.

Los dibujos y graficas requieren de algun plan para obtener el formato que se quiere.

Las siguientes sugerencias pueden ayudar a obtener tales resultados.

1.-Familiarizarse con el dispositivo grafico el cual se va a tener acceso.

2.-Tener una formacion matematica es esencial y pensar que cualquier problema en Graficacion por Computadora se reduce a especificar puntos en un espacio de dos dimensiones.Los problemas se reducen en saber como especificar estos puntos.

3.-La posicion inicial de la pluma,cuando la operacion empieza,es tomada como el origen logico (x=0,y=0).

Todo movimiento de la pluma esta definido en base a x y y.

Subsecuentemente ,puede ser establecida otras posiciones y origenes.

4.-El eje-X , alcanza un maximo de 122 pies , que es la longitud del rollo de papel.

5.-El eje-Y , es paralelo a las plumas.

6.-El angulo de rotacion con respecto a cualquier punto, es determinado por un vector.Cuando este vector coincide con el eje de las X's es de 0 grados.

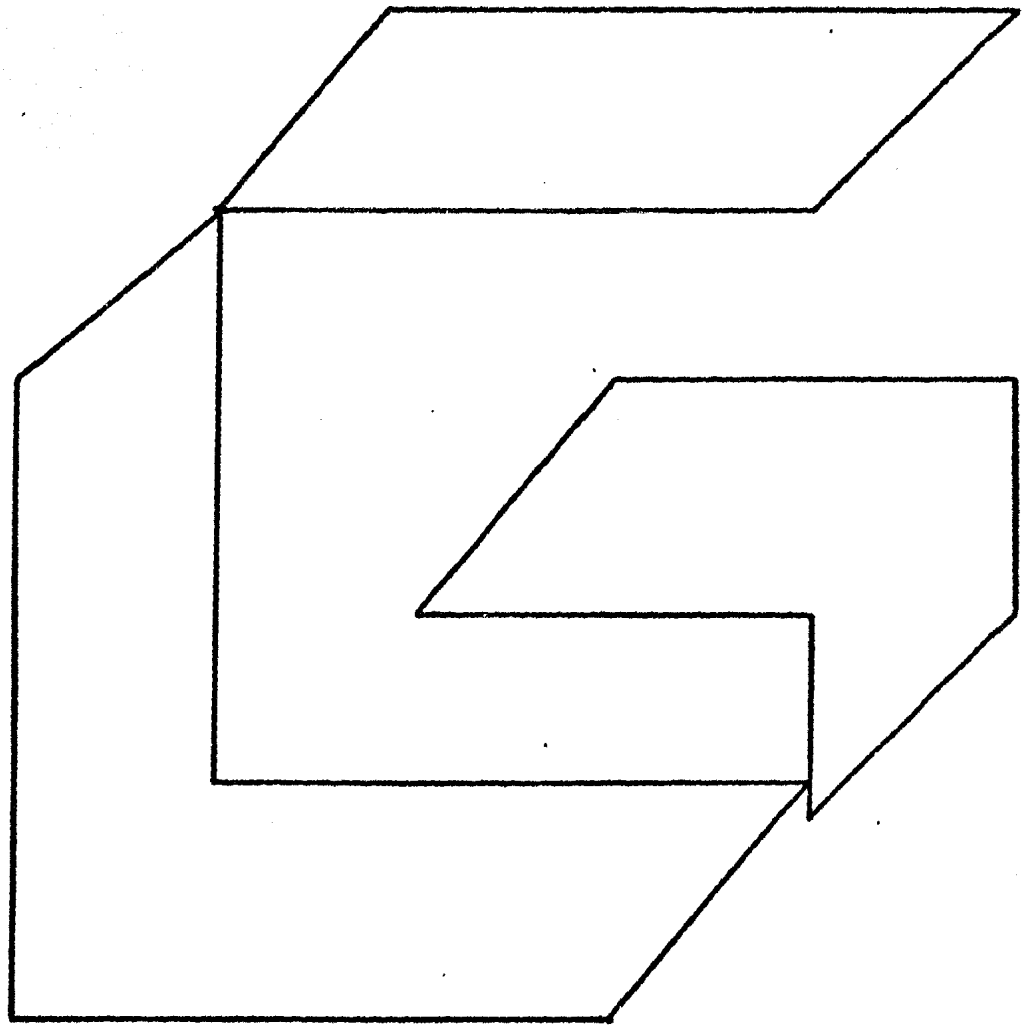
7.- Despues de graficar , la pluma debe ser movida en una posicion en la que sea facilemnte removida del graficador.

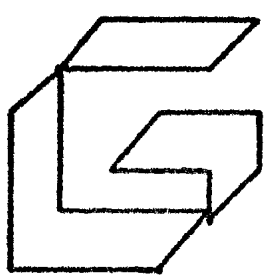
Ejemplo 1 : FIGURA .

Lee las coordenadas de la figura que
se desea dibujar.
Los datos son almacenados en el archivo
de datos FOR002.DAT ,
Se almacena las coordenadas y el estado
de la pluma (2 o 3).

```
 CALL PLOTS(0,0,1)
 WRITE(5,*)' DAR FACTOR DE ESCALA GLOBAL '
 READ(5,*)FAC
 CALL FACTOR(FAC)
 WRITE(5,*)'DAR NUMERO DE PUNTOS DE LA FIG '
 READ(5,*)NPTS
 DO 5 I=1,NPTS
 READ(2,*)X,Y,IPEN
 CALL PLOT(X,Y,IPEN)
 CONTINUE
 CALL PLOT(0.0,0.0,999)
 CALL EXIT
 END
```

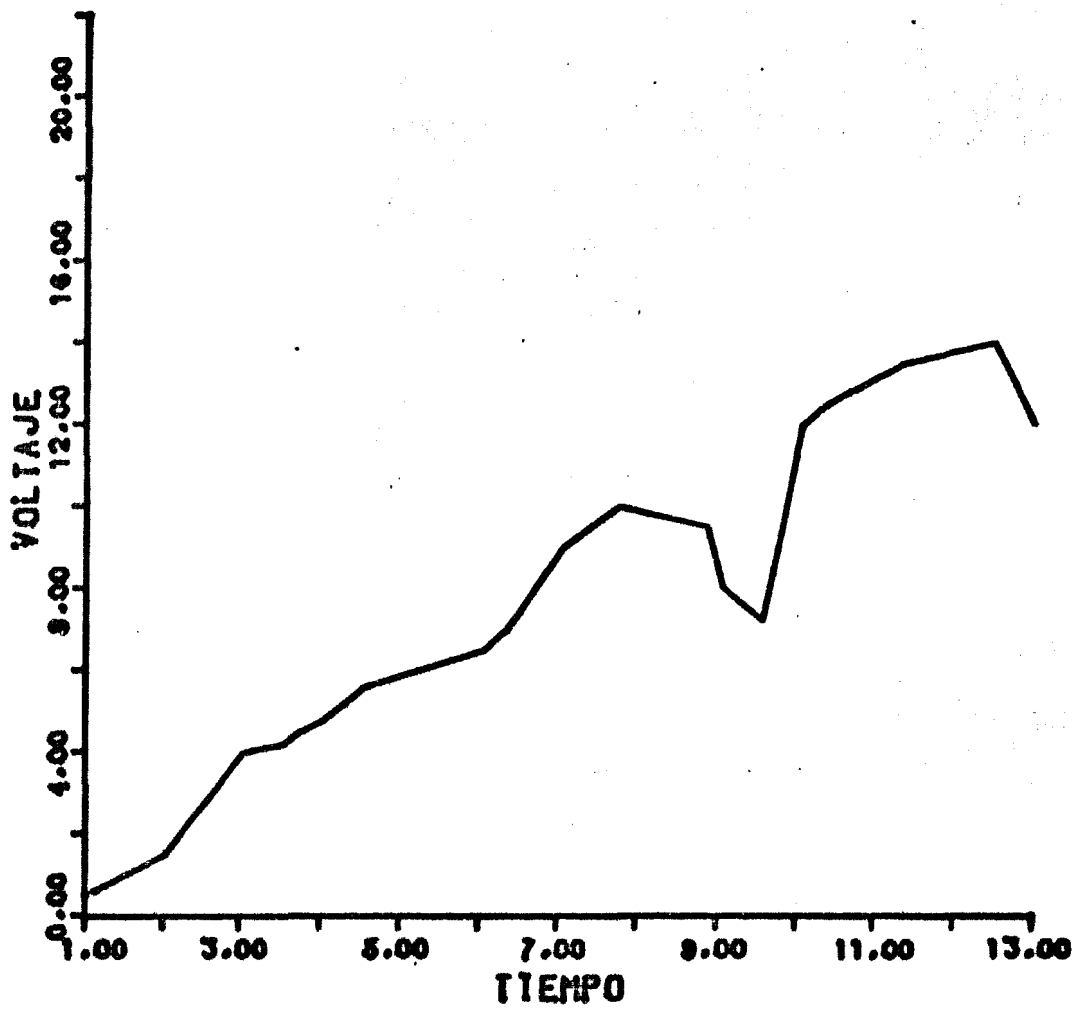
```
TI:=FOR002.DAT
 .7,3
 .7,2
2.5,2
2.5,2
2.5,3
2.9,2
3.5,2
2.5,2
3.5,3
4.0,2
4.0,2
2.5,2
3.5,3
4.0,2
3.3,2
2.5,2
3.0,3
3.0,2
2.3,2
2.3,2
2.0,3
1.0,2
 .8,3
 .8,2
1.2,3
1.9,2
 .8,3
2.2,2
1.3,3
 .9,2
1.9,2
2.2,2
```





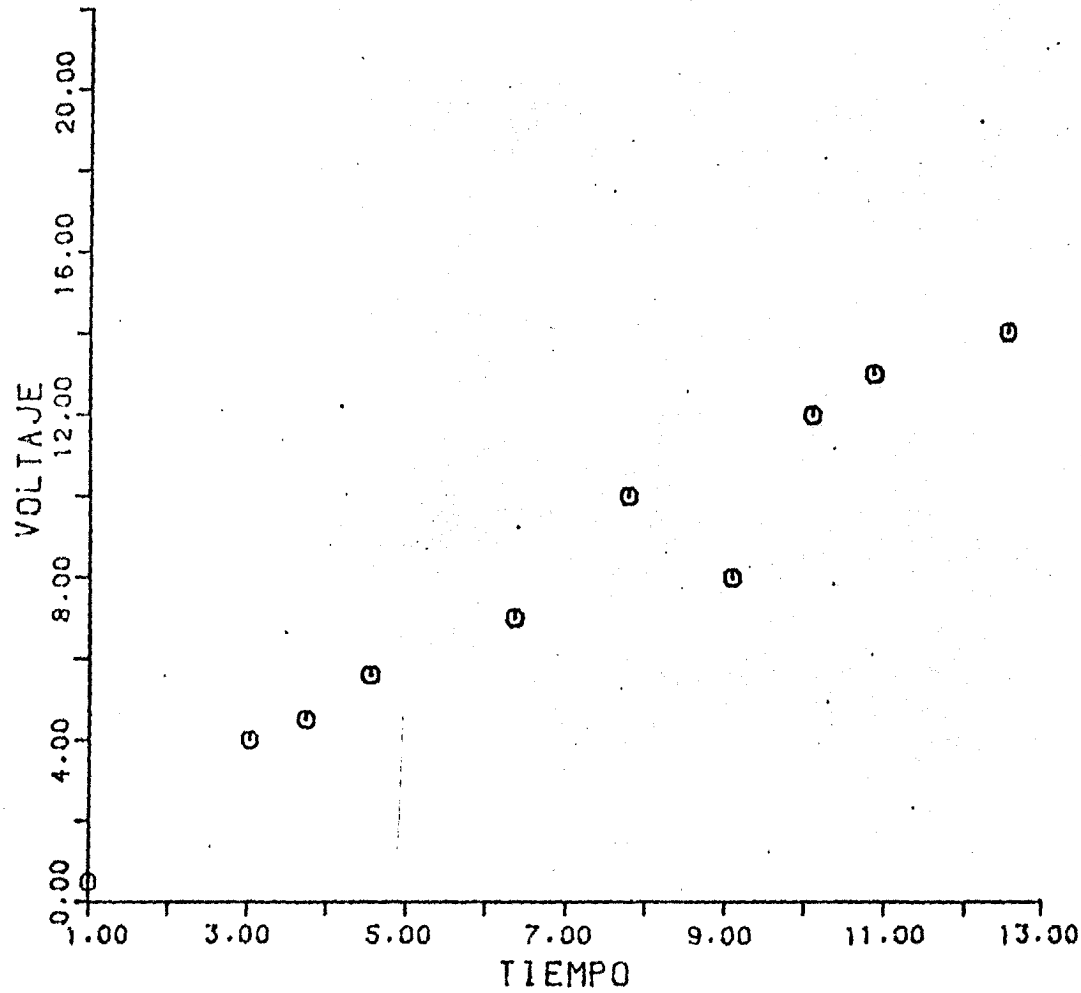
I:=PROGR.FTN

```
  Los datos son leidos de un archivo de datos
cual se le llamo FOR003.DAT.
  Se ilustra como se utilizan las rutinas:
  PLOTS, PLOT,SCALE,AXIS,LINE.
  Para LINE , se muestra las tres opciones
  los datos conectados,solo simbolos usados
  y la combinacion de ambos.

  Primer ejemplo: En LINE , LINTYP=0 ,
  los datos son solo conectados.
  DIMENSION X(52),Y(52)
  CALL PLOTS(0,0,1)
  WRITE(5,*)'Cuantos puntos son ? '
  READ(5,*)NP
  DO 5 I=1,NP
  READ(3,*)X(I),Y(I)
  CONTINUE
  CALL PLOT(3.0,9.0,-3)
  CALL SCALE(X,12.0,NP,1)
  CALL SCALE(Y,11.0,NP,1)
  CALL AXIS(0.0,0.0,'TIEMPO',-6,12.0,0.0,X(NP+1),X(NP+2))
  CALL AXIS(0.0,0.0,'VOLTAJE',7,11.0,90.0,Y(NP+1),Y(NP+2))
  CALL LINE(X,Y,NP,1,0,0)
  CALL PLOT(20.0,0.,999)
  CALL EXIT
  END
```


VOLTAJE
0.00
4.00
8.00
12.00
16.00
20.00
1.00
3.00
5.00
7.00
9.00
11.00
13.00
TIEMPO

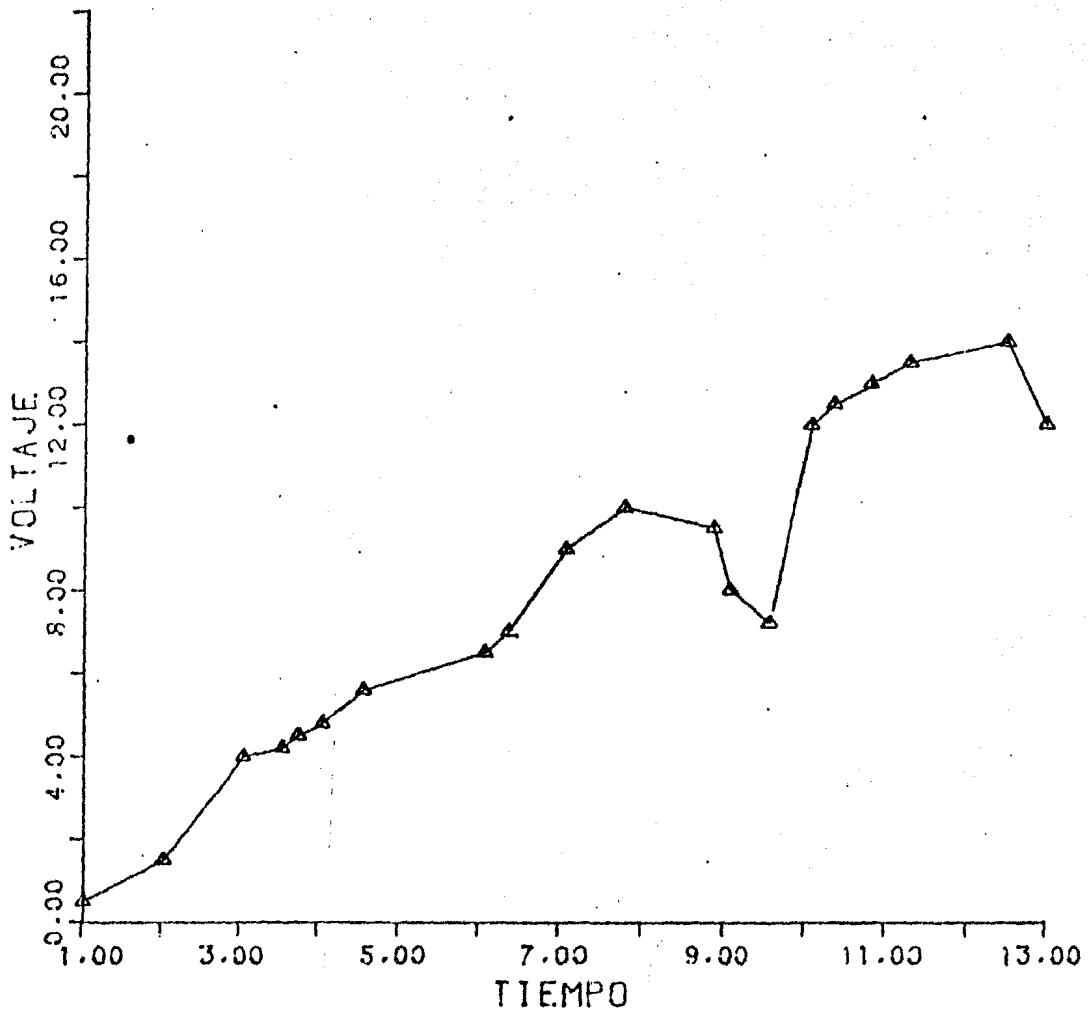
I:=PROGR1.FTN

```
Los datos son leidos de un archivo de datos
al cual se le llamo FOR003.DAT.
Se ilustra como se utilizan las rutinas:
PLOTS, PLOT,SCALE,AXIS,LINE.
Para LINE , se muestra las tres opciones
los datos conectados,solo simbolos usados
y la combinacion de ambos.

Segundo ejemplo: En LINE , LINTYP=-2 ,
los  datos no son  conectados.
Solo el simbolo INTEQ= 1, es
usado.
DIMENSION X(52),Y(52)
CALL PLOTS(0,0,1)
WRITE(5,*)'Cuantos puntos son ? '
READ(5,*)NP
DO 5 I=1,NP
READ(3,*)X(I),Y(I)
CONTINUE
CALL PLOT(3.0,9.0,-3)
CALL SCALE(X,12.0,NP,1)
CALL SCALE(Y,11.0,NP,1)
CALL AXIS(0.0,0.0,'TIEMPO',-6,12.0,0.0,X(NP+1),X(NP+2))
CALL AXIS(0.0,0.0,'VOLTAJE',7,11.0,90.0,Y(NP+1),Y(NP+2))
CALL LINE(X,Y,NP,1,-2,1)
CALL PLOT(20.0,0.,999)
CALL EXIT
END
```


VOLTAJE
0.00
4.00
8.00
12.00
16.00
20.00
1.00
3.00
5.00
7.00
9.00
11.00
13.00
TIEMPO

TI:=PROGR2.FTN

```
Los datos son leidos de un archivo de datos
al cual se le llamo FOR003.DAT.
Se ilustra como se utilizan las rutinas:
PLOTS, PLOT,SCALE,AXIS,LINE.
Para LINE , se muestra las tres opciones
los datos conectados,solo simbolos usados
y la combinacion de ambos.

Tercer ejemplo: En LINE , LINTYP=1 ,
los datos son  conectados,
y el simbolo INTEQ = 2 ,es usado.
DIMENSION X(52),Y(52)
CALL PLOTS(0,0,1)
WRITE(5,*)'Cuantos puntos son ? '
READ(5,*)NP
DO 5 I=1,NP
READ(3,*)X(I),Y(I)
CONTINUE
CALL PLOT(3.0,9.0,-3)
CALL SCALE(X,12.0,NP,1)
CALL SCALE(Y,11.0,NP,1)
CALL AXIS(0.0,0.0,'TIEMPO',-6,12.0,0.0,X(NP+1),X(NP+2))
CALL AXIS(0.0,0.0,'VOLTAJE',7,11.0,90.0,Y(NP+1),Y(NP+2))
CALL LINE(X,Y,NP,1,1,2)
CALL PLOT(20.0,0.,999)
CALL EXIT
END
```


VOLTAJE
0.00
4.00
8.00
12.00
16.00
20.00
1.00
3.00
5.00
7.00
9.00
11.00
13.00
TIEMPO

CONCLUSIONES

El modelado de curvas es una parte muy importante en Graficacion porque tiene una aplicacion muy amplia en muchas areas como el diseño de simbolos(simbolos matematicos , tipografia,etc) , para diseñar figuras en 2D (animacion ,diseño grafico) ,interpolacion para el analisis de datos en 2D , etc.

Todo dispositivo de Graficacion debe contar con algun metodo para el modelado de curvas , el mas comun de encontrar es el metodo de Bezier.

Pero existen mejores metodos como el de B-splines , que es el que se presenta en este Paquete.

Actualmente se estan desarrollado metodos como el de Beta-splines, el cual permite mejor manejo de las curvas usando parametros sin modificar los puntos de control .

Este metodo fue desarrollado por Barsky en 1983 [BARK].

La extension del modelado de curvas es el modelado de superficies , el cual es aplicado al diseño de imagenes tridimensionales , en lugar de tener puntos de control , se va a tener una malla la cual va a definir la figura inicial y se va a poder modificar la malla hasta obtener la figura final.

Esta ultima es una opcion que puede añadirse al Paquete de Graficacion.

Con presentar la proyeccion mas sencilla hasta las mas complicada , se pretende hacerlo asi porque en algun momento se va a querer hacer animacion o quizas otra aplicacion en la que se va a necesitar que se tenga una gran versatilidad para cambiar la imagen y no solamente tener una proyeccion que de la sensacion de estar viendo la figura en 3D.

Hay que crear otro tipo de proyecciones como el de mapear una cierta figura en un cubo , o bien en una botella que no es necesariamente un plano.

El manejar las sombras de las figuras asi como el color y textura es otro aspecto que debe estudiarse.

Con respecto a los algoritmos de lineas y superficies ocultas , como se vio anteriormente, hay algoritmos desde muy sencillos pero muy costosos hasta algunos muy complicados de implementar, pero se pretende obtener el optimo dependiendo de la aplicacion.

Va a ser distinto el crear un algoritmo de lineas o superficies ocultas y el que se emplee para graficar objetos en una escena , o bien para objetos animados en 3D en tiempo real.

Con respecto al Paquete podria sustituirse por un Lenguaje de Graficacion con la aplicacion dada aqui.

Bien puede añadirse al Paquete otro tipo de

estructura como el manejo de sesmentos , el cual permite manejar a cada fisura de manera individual permitiendo añadir las caracteristicas que se quiera como la visibilidad o no visibilidad de la imasen , de esta manera se va a poder crear , borrar o añadir nuevos sesmentos que finalmente van a crear la imasen total.

Tambien añadir un interprete reduciria el tiempo que se emplea en desplesar la imasen , ya que se crearia un archivo en donde se suardaria la imasen a sraficar y el interprete se encarsaria de vaciar la imasen en el dispositivo de Graficacion.

Lo que se cuido al crear el Paquete de Graficacion fue la modularidad y la estandarizacion, esto se comprobo al pasar las rutinas del Paquete a la B-7800 para usar el Graficador de Pantalla Tektronix.

Solamente se vio cuales eran los primitivos sraficos y en base a ellos se implemento el Paquete.

Asi si se tiene otro dispositivo de Graficacion este Paquete va a poder funcionar siempre y cuando se tensa compilador FORTRAN IV y las rutinas basicas o primitivos sraficos.

BIBLIOGRAFIA

[AHL] Ahlberg J. H., Nilson E.N. , Walsh J.L. "Theory of Splines and their Applications" Academic ,New York, 1967.

[AHU] Ahuja, D.V. "An Algorithm for Generating Spline like Curves " , IBM Syst. J. 206 , 1968.

[ARM] Armit A.P. ,Forrest A.R. "Interactive Surface Design" Comput. Graphics 1970.

[AND] Anderson , David " Hidden Elimination in Projected Grid Surfaces " , ACM Transactions on Graphics ,Vol 1 No. 4 Octubre 1982 pag 274-288.

[ATHE] Atherton , P.K. Weiler y Greenberg " Polygon Shadow Generation" Computer Graphics 12(3):275 ,Agosto 1978

[BAR] Barsky B.A. "Description and Evaluation of Various 3-D Models " IEEE Computer Graphics and Applications , Mar. 1984.

[BER] Bergeron R.D. , Bono P.R. ,Foley J.D. "Graphics Programming using the Core System " ACM Computing Surveys Vol. 10 No. 4 Diciembre 1978 389-443 .

[BUT] Butland ,J. "Surface drawing made simple " Computer Aided Design 11 ,1(Enero 1979) 19-22

[CALC] "Calcomp Software Reference Manual" ,Document No 1005, California Computer Products , Inc. Anaheim ,Calif. Feb. 1968.

[CARL 78] Carlbom ,I. y J. Paciorek , "Geometric Projection and Viewing Transformations" , Computing Surveys ,1(4), 1978 , pp 465-502.

[CLA] Clark ,J.H. "3-D Design of Free-Form B-spline Surfaces" Univ. Utah Comput. Sci. Dept. UTEC-CS-74-120 Sep. 75. MTIS A002736/AD/A002736.

[DEB] De Boor C. "On Calculating with B-splines " J. Approx. Theory 6: 50-62 , 1972.

[EVE] Everett R.R. "The Wirlwind I Computer " Joint AIEE-IRE Conf. 1952 Rev. Electron Digital Comp. Febrero 1952 ,p 70.

[FOR] Forrest A.R. "Interactive Interpolation and Approximation by Bezier Polynomials" Comp. J. 15(1) 72, Enero 1972.

[FOL] Foley J.D. Van Dam A. "Fundamentals of Interactive Computer Graphics" Addison - Wesley Systems Programing Series 1982.

[GORD1] Gordon W.J. Riesenfelf R.F. "Bernstein -Bezier Methods for Computer Aided Design of Free -Form Curves and Surfaces" JACM 21(2) 293-310 ,Abril 1974.

[GRIF] Grifths J.G. "A Bibliography of Hidden Line and Hidden Surface Alsorithms" Comp. Aided Design 10(3) 203-206 Mayo 1978.

[HAT] Hatfield L. "Graphics Software -from Techniques to Principles" IEEE Computer Graphics and Applications Enero 1982.

[HARR] Harrington ,S. "Computer Graphics: A Programming Approach", Mc. Graw- Hill 1983.

[JOH] Johnson, T.E. "Sketchpad III: A Computer Program for Drawing in Three Dimensions" AFIPS Conf. Proc. Vol. 23 1963 SJCC pp 347-353.

[KUB] Kubert,B. R. ,Szabo , Giulieri " The perspective representation of functions of two variables" J. ACM 15 2(abril 1968) 193-204.

[MAH] Machover C. "A Brief Personal History of Computer Graphics" IEEE Computer Nov. 1978.

[MCCA] Mc Callister , S. I.E. Sutherland "Final Report on the Area Warnock Hidden-Line Alsoritm" Evans and Sutherland Computer Corp. Salt Lake City , Feb. 1970.

[NEW] Newman W. , Sproull R. "Principles of Interactive Computer Graphics " 2a Edicion Mc.Grau Hill 1979.

[NEWD] Newman W. , Van Dam A. "Recent efforts towards Graphics Standardization" ACM Computing Surveys Vol. 10 No. 4 Diciembre 1970 365-380.

[REQ] Requicha A.A.G. , Voelcker H.B. "Solid Modeling : A Historical Summary and Contemporary Assessment" IEEE Computer Graphics and Applications ,Marzo 1982.

[RICH72] Richardus , P. y R. Adler ,"Map Projections' Elsiver , North-Holland , New York 1972.

[RIES75] Riesenfeld R.F. "Aspects of Modeling in Computer Aided Design" , NCC 1975 p 597.

RIES77] Riesenfeld R.F. "Homogeneous Coordinates and Projective Planes in Computer Graphics: Uni. Utah Comp. Sci. Dept. 1977.

ROGE] Rogers , D.F. , Adams J.A. "Mathematical elements for Computer Graphics" Mc. Graw-Hill, Inc.; New York 1976 , 239pp.

SUTH63] Sutherland I.E. "SKETCHPAD :A Man Machine Graphical Communication System " AFIPS Conf. Proc. Vol. 23 , 1963 SJCC pp 329-346.

SUTH] Sutherland I.E. , Sproull F. , Schumaher " A Characterization of Ten Hidden Surface Algoritms" Computing Surveys , Vol. 6 No. 1 Mar. 1974 pp 1-55.

VAN] Van Dan Bos , Carruthers L.C. , Van Dam "GPGS : A Device - independent General Purpose Graphics System" Comp. Graphics 11(2) 112-115 , Summer 1977.

WILL] Williamson ,H. "Algorithm 420 -Hidden-line plotting program" [J6] Commun. ACM 15,2 (Febrero 1972) 100-103

WRIG] Wright , T.J. "A two space solution to hidden line problem for plotting functions of two variables." IEEE Trans. Comput. C-22 , 1 (Enero 1973) 28-33.