UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

COLEGIO DE CIENCIAS Y HUMANIDADES
UNIDAD ACADEMICA DE LOS CICLOS
PROFESIONALES Y POSGRADO

INSTITUTO DE INVESTIGACIONES
EN MATEMATICAS APLICADAS
Y EN SISTEMAS

SISTEMA GRAFICO AUXILIAR A LA DOCENCIA PARA ILUSTRAR MECANISMOS DE REACCIONES QUIMICAS Y ESTRUCTURAS MOLECULARES

QUE PARA OBTENER EL GRADO DE
MAESTRO EN CIENCIAS DE LA COMPUTACION
PRESENTA
GRACIANO CRUZ ALMANZA

MARZO 1987

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

I N D I C E

CAPITULO I

ANTECEDENTES

El desarrollo de software ha demostrado ser de gran ayuda a la docencia en diversas áreas como la Química, Física, Matemáticas, etc. En particular ayuda en el reforzamiento de conocimientos, consistiendo en sesiones previamente diseñadas que amplien los conocimientos adquiridos en base a "prácticas programadas".

La Facultad de Química de la U. N. A. M. tiene necesidades de:

a) Elaboración de clases por medio de computadora.
b) Elaboración de prácticas por medio de computadora.
c) Graficado de funciones.
d) Graficado de moléculas.

La Facultad de Química ha creado un laboratorio de apoyo a la docencia con computadoras personales. Dicho laboratorio ayuda al reforzamiento de los conocimientos adquiridos en el salón de clase.

Las necesidades de la Facultad de Química en el graficado de moléculas son:

a) Mostrar una molécula en tres dimensiones.
b) Tener un conjunto expandible de moléculas básicas.
c) Rotar moléculas con respecto a los ejes X, Y y Z.
d) Rotar la molécula en cualquier dirección.
e) Amplificar o reducir el tamaño de una molécula.
f) Ver únicamente los átomos o únicamente las ligaduras.
g) Tener hasta 100 átomos por molécula.
h) Atomos de diferentes tamaños.
i) Tener hasta 20 vistas diferentes de cada molécula.
j) Facilitar la enseñanza del concepto de "espejo", el cual consiste en mostrar dos moléculas especificas, de tal manera que el usuario vea que es imposible hacer coincidir las moléculas por más rotaciones que le haga. Esto es equivalente a tratar de hacer coincidir la mano derecha con la mano izquierda.

Para un mejor aprovechamiento del laboratorio se

estudió el software que hay actualmente, sin embargo la producción de software orientado a la Química en México está por debajo de los requerimientos y esto obliga a obtenerlo de procedencia extranjera. Es por esto que el software que se utiliza en la actualidad no corresponde a las necesidades propias ya sea por el idioma o por el contenido.

Existen en la actualidad sistemas orientados al auxilio de la docencia en Química, por ejemplo el sistema INTRODUCCION TO GENERAL CHEMISTRY, diseñado en E. U. A. por Stanley G. Smith de la Universidad de Illinois, Ruth Chaboy de la Universidad de Stanford y Elizabeth Kean de la Universidad de Wisconsin.
Este sistema es para estudiantes sin previos conocimientos de Química, fue diseñado como un complemento en un curso introductorio en Química General. Además esta ampliamente documentado, esto es que las instrucciones para el usuario estan contenidas en el disco como parte del programa. De esta forma los usuarios con poca o nula experiencia con computadoras encuentran los programas fáciles de utilizar. El sistema consta de 10 discos flexibles que contienen sesiones prácticas sobre:

1. Tabla periódica.
2. Nomenclatura inorgánica.
3. Balanceo de ecuaciones.
4. Peso atómico y molecular.
5. Porcentaje de composición.
6. Juegos educativos.
7. Gases.
8. Ph.
9. Sistema métrico.
10. Soluciones.

El inconveniente de este sistema es su poca posibilidad de expansión y el usuario no puede modificar las características del mismo. En términos generales, este es un buen sistema, pero no resuelve las necesidades gráficas de la facultad de Química.

El sistema más usado en la actualidad es MOLECULAR ANIMATOR (Molanim), diseñado en E. U. A. por J. Jeffry Howbert, el cual tiene las siguientes opciones:

a) Grafica estructuras moleculares en tres dimensiones.
b) Rota las moléculas con respecto a los ejes X, Y y Z.
c) Amplifica o reduce el tamaño de la molécula.
d) Permite ver todo el modelo, únicamente los átomos o únicamente las ligaduras.
e) Incluye un conjunto de 21 moléculas.
f) Tiene un editor y se pueden crear moléculas hasta de 60 átomos.
g) Los átomos pueden tener 2 tamaños.

h) La molécula se grafica en un color.

Aunque Molanim cumple algunos de los requerimientos de la Facultad de Química, esto no es suficiente y es por esto que se diseñó un sistema gráfico auxiliar a la docencia para ilustrar mecanismos de reacciones Químicas y estructuras moleculares, cuyo nombre es EDMOL (Editor Molecular), en esta tésis.

El presente trabajo consiste en el diseño y realización de un sistema que resuelva las necesidades de la Facultad de Química planteadas anteriormente, de esta manera EDMOL tiene las siguientes características:

a) Grafica estructuras moleculares en tres dimensiones.
b) Rota la Molécula con respecto a los ejes X, Y y Z.
c) Rota la molécula con respecto a cualquier eje dado por el usuario.
d) Amplifica o reduce el tamaño de la molécula.
e) Acerca o aleja al usuario de la Molécula.
f) Permite ver todo el modelo, únicamente los átomos o únicamente las ligaduras.
g) Incluye un conjunto de 25 moléculas básicas con posibilidad de extenderse de acuerdo a la capacidad del disco.
h) Tiene un editor y se pueden crear moléculas hasta de 100 átomos y 200 ligaduras.
i) Los átomos pueden tener diferentes tamaños, esto es definido por el usuario.
j) La molécula se grafica en colores, esto es definido por el usuario.
k) Para cada molécula se pueden tener almacenadas en disco hasta 99 vistas diferentes de la molécula.
l) Muestra dos moléculas a la vez, para mostrar el concepto de "espejo".
m) Tiene una opción de auxilio para el usuario donde da una descripción de cada una de las opciones permitidas por EDMOL.

Como se puede apreciar EDMOL reforzará a la docencia permitiendo un mejor aprovechamiento. Las características del sistema, todas ellas positivas, permitirán un uso amplio ya que es interactivo, amigable, portatil y fácil de extender. Se tendrá software creado en la universidad el cual será adecuado a las necesidades de la misma universidad y del país, además del ahorro de divisas por la compra de sistemas.

EDMOL apoyará también a la investigación, dado que el usuario podrá ver las moléculas y así estudiar su topología. Además EDMOL tiene las bases gráficas para realizar otros trabajos, es decir, EDMOL contribuye a la creación de otros sistemas que necesiten graficar moléculas.

El capítulo II contiene las bases para el graficado en dos dimensiones, principia con las operaciones básicas de graficado (punto, segmento, círculo), tranformaciones como la traslación, cambio de escala y rotación. Se comenta brevemente el uso de la geometría proyectiva, para argumentar el uso de coordenadas homogéneas en el plano. Se muestran también las ventajas de utilizar coordenadas homogéneas para realizar transformaciones en base a matrices de 3*3. Por último se analizan los conceptos de ventana y recorte para protección de regiones.
El capítulo III contiene los conceptos básicos en el desarrollo de sistemas gráficos en tres dimensiones. Principia con transformaciones (rotación, cambio de escala y traslación) y se comenta brevemente la geometría proyectiva para argumentar el uso de coordenadas homogéneas en el espacio.
Con base en coordenadas homogéneas se describen las transformaciones en tres dimensiones usando notación matricial. Para terminar se comentan las proyecciones paralela y perspectiva.
El capítulo IV contiene la descripción del diseño de EDMOL, principiando con un diseño conceptual, siguiendo con los algoritmos importantes y terminando con una descripción de las estructuras de datos usadas.

CAPITULO II

GRAFICAS EN DOS DIMENSIONES

2.1 OPERACIONES DE GRAFICACION

La unidad mínima de graficado es el punto, también llamado "pixel". Las líneas deben ser dibujadas como una sucesión de puntos (pixels) cercanos. Si se desea dibujar una letra, por ejemplo, se usa una matriz de puntos.

En la actualidad, casi todas las pantallas proveen funciones que permiten pintar una línea o segmento. Este recurso da mayor poder al usuario, ya que no se tiene que preocupar por dibujar un segmento punto por punto. Si fuera necesario dibujar un segmento punto por punto, podrían usarse los algoritmos DDA y el de Bresenham, los cuales son descritos en el apéndice A de este trabajo.

Una forma en que las computadoras facilitan al usuario dibujar segmentos y puntos, es permitiendo el uso de coordenadas cartesianas sobre la pantalla. De esta manera los usuarios usan parejas (X, Y) para hacer referencia a un punto sobre la pantalla.

Dada una pantalla, llamaremos "resolución",al número de puntos visibles distintos que pueden ser mostrados en un segmento dado. Una pantalla típica puede tener una resolución de 100 puntos por pulgada. Hay que tomar en cuenta que la pantalla tiene un número finito de puntos.

En general las pantallas tienen un conjunto básico de operaciones primitivas de graficación, que son:

a) Inicialización de la pantalla.
b) Pintado de un punto.
c) Pintado de un segmento.

Sin embargo, es necesario contar con otras facilidades más, como dibujar triángulos, cuadrados, círculos, etc. Para trazar un triángulo o un cuadrado, simplemente hay que usar varias veces la operación primitiva que pinta segmentos. Pero pintar un círculo no es tan inmediato, aunque se utiliza el mismo principio. Para pintar un círculo en la pantalla, basta dar su posición y su tamaño. La posición puede especificarse por las coordenadas (XC, YC) del centro del círculo, y el tamaño mediante el radio R. También es necesaria la regla para determinar las coordenadas de los puntos que habrán de iluminarse para que se vea el círculo.

Al representar circulos, lo que realmente se pinta son aproximaciones de ellos, dado que la pantalla tiene un número finito de puntos.

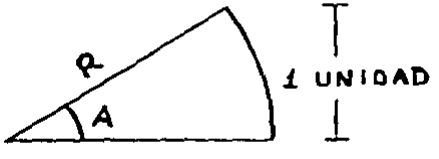
El circulo puede ser dibujado de varias maneras. Una forma conveniente es determinar coordenadas sucesivas (X, Y) de puntos distribuidos uniformemente a lo largo del perimetro [2] [7], de la siguiente forma:

$$X = XC + R*COS(A)$$

$$Y = YC + R*SIN(A)$$

Una vez calculados los puntos, se procede a unirlos con segmentos, obteniéndose un polígono que – si se escoge un ángulo A "pequeño"– dará la idea de circulo al ojo humano.

Cuando se desea aumentar la velocidad de graficado de un circulo se evita el conectar los puntos (X, Y) calculados, para esto es necesario pintar los puntos de tal forma que para el ojo humano aparenten no estar separados.
Se obtiene una buena aproximación a un círculo punto por punto, cuando éstos son adyacentes en la ruta marcada por el circulo. La distancia entre puntos adyacentes es una unidad en dirección tanto horizontal como vertical, por lo que la separación angular (en radianes) de dos puntos en el círculo puede ser aproximada por el inverso del radio, 1/R. Como lo muestra el siguiente dibujo:



Esta aproximación trabaja bien para la mayoria de los casos, de tal forma que para un paso angular pequeño, asegurará que el circulo no tendrá huecos [1]. Esto es deseable porque permite el uso de algunos algoritmos simples para rellenarlo.

Este circulo está centrado en el origen del sistema de coordenadas. Para un circulo centrado en la posición (XC , YC) , se suma XC a todos los valores X y se suma YC a todos los valores Y. Estas sumas mueven (trasladan) el circulo del origen de coordenadas a la posición deseada.

En algunas microcomputadoras existe la instrucción para

dibujar círculos, y sus parámetros son el centro del círculo (XC, YC) y el radio R. Donde XC, YC, R, pueden ser constantes o expresiones aritméticas.

Aunque el círculo es de las curvas más usadas, también es importante la elipse al realizar gráficas por computadora, siendo el círculo un caso particular.
Las ecuaciones para la elipse pueden escribirse en la forma:

$$X = XC + Rx * COS (A)$$

$$Y = YC + Ry * SIN (A)$$

en que Rx y Ry denotan la longitud de los semiejes [2] [7]. Si Rx es igual a Ry tendremos un círculo.

Las curvas elípticas son usadas en muchas áreas de la graficación, por ejemplo las órbitas de los satélites, la visión en tres dimensiones de un cilindro mostrará la base de él en forma de elipse, etc.

2.2 TRANSFORMACIONES EN DOS DIMESIONES

Un sistema de graficado debe permitir al usuario la aplicación de transformaciones a sus objetos graficos [3] [4] [5] [8]. Por ejemplo, agrandar la gráfica, de tal manera que pueda verle detalles, o reducirla, para que una mayor parte del objeto a graficarse sea visible.

Las transformaciones más usadas son traslaciones, cambios de escala y rotaciones.

Cada una de las transformaciones es usada para generar un nuevo punto (X', Y') a partir de las coordenadas de un punto (X, Y) en la gráfica original.
Si el objeto a graficar incluye un segmento, es suficiente con aplicar la transformacion a los puntos finales de él y entonces graficar el segmento entre los dos puntos finales transformados.

Las traslaciones son de la forma:

$$X' = X + Tx$$

$$Y' = Y + Ty$$

De tal manera que a cada punto (X, Y) se calcula un (X',Y').

Las transformaciones de cambio de escala son:

$$X' = X * Sx$$

$$Y' = Y * Sy$$

Asi, a cada punto (X, Y) se le afecta por los factores de escala Sx , Sy para obtener (X', Y').
Si Sx y Sy no son iguales, tienen el efecto de distorsionar la figura, alargándola o achicándola en las direcciones de los ejes coordenados. Las reflexiones de un objeto respecto a los ejes pueden ser generadas usando valores negativos de Sx y Sy.

Por último las rotaciones con centro en el origen y un ángulo A son de la forma:

$$X' = X * COS(A) + Y * SIN(A)$$

$$Y' = -X * SIN(A) + Y * CON(A)$$

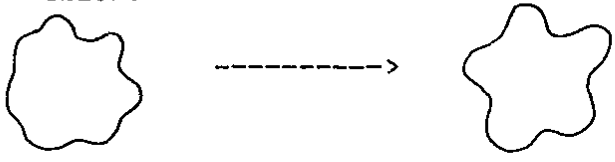
Secuencias de transformaciones anteriores pueden ser combinadas para definir nuevas transformaciones mediante la

composición usual. En general, es necesario efectuar transformaciones más complicadas como sería el rotar con respecto a un punto arbitrario.
La rotación con respecto a un punto arbitrario puede ser efectuada realizando tres transformaciones simples: una traslación, seguida de una rotación y después otra traslación.

El orden en que las transformaciones son calculadas no debe ser modificado, ya que no conmutan.

Componer varias transformaciones tiene varias ventajas, debido a que podemos representarlas de una forma más compacta. Además, es generalmente posible calcular las transformaciones con menos operaciones aritméticas, que si aplicáramos cada una de las transformaciones en secuencia.

Como es bien sabido, las transformaciones lineales del plano en sí mismo pueden ser representadas por matrices de 2 * 2 [6] [13]. Es decir:



TRANSFORMACIONES LINEALES EN DOS DIMENSIONES. MATRICES DE 2 * 2

Y en términos de esta matriz, la transformación (secuencia de rotaciones con centro en el origen y cambios de escala) es expresada así:

$$[\ X',\ Y'\ ] = [\ X,\ Y\ ] \begin{vmatrix} A & D \\ B & E \end{vmatrix}$$

Si vemos las tranformaciones por separado, tenemos que para:

$$\text{ROTACION: } [\ X',\ Y'\ ] = [\ X,\ Y\ ] \begin{vmatrix} COS(A) & -SIN(A) \\ SIN(A) & COS(A) \end{vmatrix}$$

$$\text{CAMBIO DE ESCALA: } [\ X',\ Y'\ ] = [\ X,\ Y\ ] \begin{vmatrix} Sx & 0 \\ 0 & Sy \end{vmatrix}$$

Es fácil verificar que estas fórmulas son equivalentes a las descritas al inicio de esta sección.
Y como la composición de transformaciones lineales corresponde a la multiplicación de matrices [13], computacionalmente conviene realizar la multiplicación para

tener representada la composición de varias rotaciones y cambios de escala mediante una matriz [A].

Algunos casos particulares, correspondientes a ciertas transformaciones importantes, aunque no muy usadas, son:

Reflexión en el eje Y: $[X', Y'] = [X, Y] \begin{vmatrix} -1 & 0 \\ 0 & 1 \end{vmatrix}$

Reflexión en el eje X: $[X', Y'] = [X, Y] \begin{vmatrix} 1 & 0 \\ 0 & -1 \end{vmatrix}$

Reflexión en el origen: $[X', Y'] = [X, Y] \begin{vmatrix} -1 & 0 \\ 0 & -1 \end{vmatrix}$

Reflexión en la linea Y=X: $[X', Y'] = [X, Y] \begin{vmatrix} 0 & 1 \\ 1 & 0 \end{vmatrix}$

Reflexión para Y= -X: $[X', Y'] = [X, Y] \begin{vmatrix} 0 & -1 \\ -1 & 0 \end{vmatrix}$

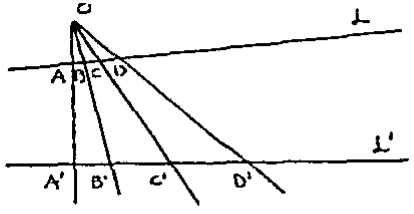
Nótese , sin embargo, que dentro de las representaciones matriciales no hay alguna para la traslación. Esto se debe a que la traslación no es una transformación lineal y no es posible asociarle una matriz de 2 * 2.

Para salvar este obstaculo, se ha hecho uso de una herramienta matemática: las coordenadas cartesianas homogéneas. En las próximas secciones damos una presentación de ellas.

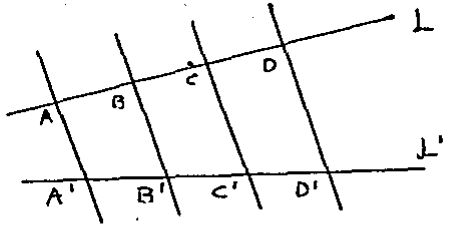
2.3 GEOMETRIA PROYECTIVA

La geometría de Euclides que comúnmente se trabaja está estrechamente relacionada con longitudes, ángulos y áreas. Es una geometría de medida, o geometría métrica [1].
Como es natural, en esta geometría, son de interés todas aquellas transformaciones que respetan las longitudes, ángulos, áreas y los objetos mismos, es decir, su forma. A este tipo de transformaciones se les denomina movimientos rígidos. Ejemplos de ellos son las traslaciones y las rotaciones.

Existen sin embargo, otro tipo de transformaciones, como la transformación de cambio de escala, que en general deforma, y así, por ejemplo un círculo puede ser transformado en una elipse cambiando la escala de una de las coordenadas. Las transformaciones de este tipo están incluidas en las transformaciones llamadas proyecciones, que se resumen en dos tipos: proyección central y proyección paralela.
Por ejemplo, consideremos dos líneas distintas L y L' en el plano y algunos puntos A, B, C, D, sobre l. La proyección central de dichos puntos de l desde O, proyectados sobre L', donde O es un punto que no pertenece ni a l ni a l', son los puntos A' B', C', D', que se muestran en la siguiente figura:

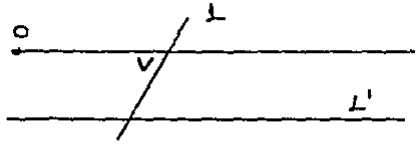


Si en lugar de usar líneas a través de un punto O para efectuar la proyección de L a L', se usaran líneas paralelas con una dirección dada, distinta de las líneas l. y L', como se ve en la figura:

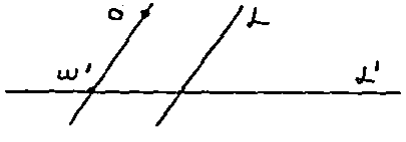


La proyección es llamada proyección paralela. En general, las proyecciones no respetan ángulos, áreas, longitudes ni formas de figuras.

Diremos que una propiedad de una figura es proyectiva si es preservada por cualquier proyección.
El estudio de las propiedades proyectivas y sus relaciones constituye lo que se denomina geometria proyectiva.
Cabe hacer notar que en la proyección central mostrada anteriormente, un punto V sobre L no tendrá proyección sobre L' cuando la recta que pasa por O y V sea paralela a L', esto es:



Hay un solo punto sobre L con esta propiedad.
Similarmente, hay un punto W' sobre L', que no es proyección de ningún punto de L, y que resulta de intersectar a L' con la paralela a L que pasa por O, como se muestra en la figura:



Nuevamente hay un solo punto sobre L' con esta propiedad.

Salvo estos dos puntos, hay una correspondencia uno a uno entre los puntos de L distintos y los punto de L'.
Sin embargo, las excepciones se originan en el hecho de que dos rectas paralelas no se intersectan. Asi que, si cambiamos ésto, removeremos dichas excepciones. Si dos rectas paralelas se intersectan, deberá ser en un nuevo punto distinto a los ya conocidos. Este nuevo punto es denominado punto al infinito. De este modo, por cada familia de rectas paralelas en una dirección dada, aumentaremos al plano un punto al infinito correspondiente a tal dirección. El conjunto de todos los puntos al infinito constituye lo que se denomina la recta al infinito o linea al infinito.
El nuevo plano, obtenido de aumentarle al plano original los puntos al infinito, se denomina plano extendido. Usualmente también se denomina plano finito y puntos finitos al plano original y sus puntos, respectivamente.

La ganancia que se obtiene de esta construcción es que se suprimen muchas excepciones existentes en la geometria

tradicional, y más aún, los resultados de ésta pueden verse como casos particulares de aquélla.
Para utilizar los puntos al infinito y aprovechar las ventajas que ellos ofrecen, conviene introducir un sistema de coordenadas para ellos, las llamadas coordenadas homogéneas.

2.4 COORDENADAS HOMOGENEAS

Las coordenadas que se utilizan para los puntos en el plano extendido, se denominan coordenadas homogéneas [4] [8] y se definen en terminos de las coordenadas cartesianas en R3.
Primero definiremos las coordenadas homogéneas para puntos finitos.

Las coordenadas homogéneas (X1, X2, X3) de un punto finito de coordenadas cartesianas (X, Y) son cualesquiera números X1, X2, X3, tales que X=X1/X3 y Y=X2/X3. Asi, por ejemplo, dado el punto P = (X, Y), tanto (X, Y, 1) como (2X, 2Y, 2) y (-3X, -3Y, -3) son coordenadas homogéneas del punto P.
En general, dado cualquier número r distinto de cero, (rX,rY,r) son coordenadas homogéneas de P. Un punto finito tiene entonces un conjunto infinito de coordenadas homogéneas. Nótese que la tercera coordenada debe ser distinta de cero.
Inversamente, cualesquiera tres números X1, X2, X3, donde X3 <> 0, son coordenadas homogéneas de un punto definido, del punto (X, Y) donde X = X1/X3 y Y = X2/X3. Asi (3, -2, 4) es la representación en coordenadas homogéneas del punto (3/4, -1/2).

Se tiene, por tanto, que los puntos finitos tienen asociadás como coordenadas homogéneas todas las ternas (X1, X2, X3) para las cuales X3 <> 0. Solo restan las ternas de la forma (X1, X2, 0). Estas ternas se asignan como coordenadas homogéneas de los puntos al infinito

Se define un punto al infinito por cada conjunto de rectas paralelas con una dirección dada, esto es , con una pendiente especifica. Las coordenadas homogéneas para tal punto al infinito deben, por tanto, depender sólo de dicha pendiente. Así, cualquier terna (X1, X2, 0) para la cual X2/X1 = LAMDA, es una coordenada homogénea del punto al infinito en la dirección de pendiente LAMDA.

Usualmente se escogen representaciones sencillas, por ejemplo para un punto finito (X, Y) se toman las coordenadas homogéneas (X, Y, 1) y para el punto al infinito en la dirección de pendiente LAMDA se toma (1, LAMDA, 0)

2.5 TRANSFORMACIONES EN DOS DIMENSIONES CON COORDENADAS HOMOGENEAS

En la sección 2.2 se vio la importancia práctica de expresar las transformaciones en el plano mediante producto de matrices, de manera que un punto (X, Y) se transforma en el punto (X', Y') así:

$$(X', Y') = (X, Y) \begin{vmatrix} A & C \\ B & D \end{vmatrix}$$

donde la matriz de 2 * 2 especifica completamente la transformación. Sin embargo, no hay representación matricial de este tipo para las traslaciones. Veamos que sucede, si en lugar de trabajar con coordenadas cartesianas normales, usamos coordenadas homogéneas [3] [4] [5] [8].
En este caso, tendremos que el punto (X, Y, 1) se transforma en el punto (X', Y', 1) y la matriz asociada a la transformación debe ser una matriz de 3 * 3, esto es, se debe tener:

$$(X', Y', 1) = (X, Y, 1) \begin{vmatrix} A & D & G \\ B & E & H \\ C & F & I \end{vmatrix}$$

que se traduce en:

$$X' = A*X + B*Y + C$$

$$Y' = D*X + E*Y + F$$

$$1 = G*X + H*Y + I$$

De esta manera tenemos que las rotaciones, cambios de escala y traslaciones se representan en las formas siguientes:

$$\text{ROTACION: } [X', Y', 1] = [X, Y, 1] \begin{vmatrix} COS(A) & -SIN(A) & 0 \\ SIN(A) & COS(A) & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

2.6 VENTANAS Y RECORTE

Para una aplicación dada, por ejemplo en la arquitectura, el usuario desea tener todo el plano del edificio a construir (objeto), sin embargo, podría ser de su interés el ver solamente un piso del edificio. Es decir que en la aplicación gráfica será necesario ver partes de la gráfica, de tal manera que se puedan apreciar mejor los detalles. Esto da el efecto de ver el objeto a graficar a través de una ventana, más aún, es deseable agrandar esas porciones para obtener las ventajas de utilizar toda el área disponible de la pantalla.

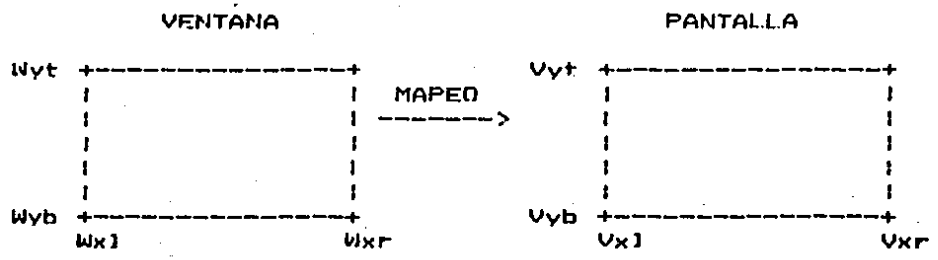
A la región seleccionada para ver una porción del objeto a graficar se le llama "ventana" [3] [4] [5] [8].
Aunado al concepto de ventana, está el de puerto de visión, que es el rectángulo en la pantalla donde deseamos dibujar el contenido de la ventana.

Por ejemplo, si tenemos el dibujo:

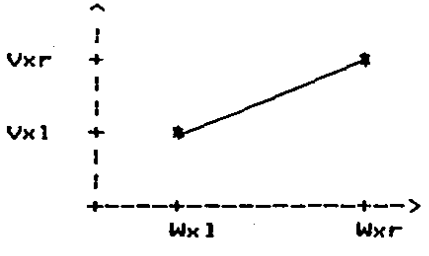
y deseamos ver solamente el sol, definimos la ventana de tal manera que deje ver solamente a éste. Ya que se tiene el área a graficar, hay que definir en que región de la pantalla se desplegará.

Resumiendo, diremos que la ventana define "qué" es lo que queremos ver, y el puerto de visión especifica "dónde" será desplegado.
Estos dos conceptos son de gran ayuda ya que el usuario podrá examinar parte por parte un objeto. Esto se hará definiendo una ventana de tamaño fijo y variando su posición.

Dada una ventana y un puerto de visión, definidos de la siguiente forma:



Deseamos encontrar las transformaciones que nos llevan de la ventana a la pantalla. Esto se hará de la siguiente forma, tenemos el segmento determinado por Wxl y Wxr, y deseamos mapearlo al segmento determinado por Vxl y Vxr.
Esta transformación podemos verla gráficamente como la recta que pasa por los puntos (Wxl, Vxl) y (Wxr, Vxr).



La ecuación que tranforma Xw (X de ventana) en Xs en la pantalla es:

$$Xs = \frac{(Vxr - Vxl)}{(Wxr - Wxl)} * (Xw - Wxl) + Vxl$$

De igual manera, calculamos Ys a partir de Yw y tenemos:

$$Ys = \frac{(Vyt - Vyb)}{(Wyt - Wyb)} * (Yw - Wyb) + Vyb$$

Con las dos ecuaciones anteriores, se transforma un punto (Xw, Yw) del objeto a (Xs, Ys) en coordenadas de la pantalla.

La transformación de la ventana es usada para cada segmento, aplicándola a los dos puntos que lo determinan, sin embargo, al pintar en el puerto de visión podría suceder que el segmento no quede totalmente incluido en el área definida por el puerto de visión.

La forma de resolver el problema es con el concepto de "recorte" de segmentos, el cual consiste en que para cada segmento a dibujarse, se hará un análisis para ver qué parte del segmento cae dentro del puerto de visión. Este análisis permitirá "cortarle" al segmento la parte que no quede dentro del puerto de visión.
En el apéndice B se describe el algoritmo escrito por Dan Cohen e Ivan Suterland [A].

Para concluir, mencionaremos que algunas computadoras en la actualidad realizan el recorte, de tal modo que el usuario no tiene que escribir estos procedimientos.

CAPITULO III

GRAFICAS EN TRES DIMENSIONES

3.1 TRANSFORMACIONES.

Las transformaciones son parte importante en la generación de imágenes en escenas de tres dimensiones. Estas son usadas para expresar posiciones de objetos en relación a otros.

Algunas transformaciones son expresadas por matrices y la composición es calculada mediante la multiplicación de matrices, para obtener una sola matriz [3] [4] [5] [8].

De esta manera, la aplicación sucesiva de varias transformaciones se puede llevar a cabo mediante multiplicación por una sola matriz [13]. Supongamos que dos transformaciones T1 y T2, son aplicadas sucesivamente. El mismo efecto puede obtenerse si aplicamos una matriz T3, la cual es el producto de las matrices T1 y T2. La concatenación de transformaciones puede verse así :
El punto (X, Y, Z) es transformado a (X', Y', Z') por la transformación T1.

$$[\ X',\ Y',\ Z'\ ] = [\ X,\ Y,\ Z\ ]\ \ T1$$

El punto (X", Y", Z") es generado por T2 :

$$[\ X",\ Y",\ Z"\ ] = [\ X',\ Y',\ Z'\ ]\ \ T2$$

De donde obtenemos:

$$[\ X",\ Y",\ Z"\ ] = (\ [\ X,\ Y,\ Z\ ]\ \ T1\ )\ \ T2$$

$$= \ [\ X,\ Y,\ Z\ ]\ (\ T1\ \ T2\ )$$

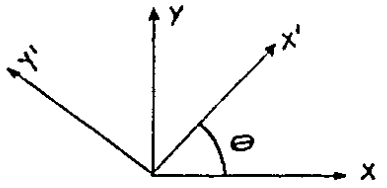
Se debe tener cuidado en preservar el orden de aplicación cuando las matrices se multipliquen, ya que T1*T2 no es necesariamnete igual a T2*T1. Recuérdese que el producto de matrices no es conmutativo [8] [13].

Como en el caso de dos dimensiones, existen cuatro transformaciones básicas : rotación, cambio de escala, reflexión y traslación.

La transformación de rotación es más compleja que en dos

dimensiones. Inicialmente, consideraremos los casos simples, en donde el eje de rotación coincide con uno de los ejes coordenados.

La rotación sobre el eje Z con un ángulo θ se puede ver fácilmente en el siguiente dibujo:



En este dibujo, el eje Z, que es el de rotación, es perpendicular a esta página. De esta forma hemos reducido el problema a una rotación en dos dimensiones, así que el sistema queda como:

$$X' = X*\cos(\theta) + Y*\sin(\theta)$$

$$Y' = -X*\sin(\theta) + Y*\cos(\theta)$$

$$Z' = Z$$

De esta manera, en notación matricial la rotación con respecto al eje Z en dirección de las manecillas del reloj es:

$$[\ X',\ Y',\ Z'\ ] = [\ X,\ Y,\ Z\ ] \begin{vmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

De una manera similar, encontramos la ecuaciones para realizar una rotación sobre el eje X, con un ángulo P.

$$Y' = Y*\cos(P) + Z*\sin(P)$$

$$Z' = -Y*\sin(P) + Z*\cos(P)$$

$$X' = X$$

Y en notación matricial:

$$[\ X',\ Y',\ Z'\ ] = [\ X,\ Y,\ Z\ ] \begin{vmatrix} 1 & 0 & 0 \\ 0 & \cos(P) & -\text{sen}(P) \\ 0 & \text{sen}(P) & \cos(P) \end{vmatrix}$$

Análogamente, daremos las ecuaciones y la matriz de rotación para el eje Y, con un ángulo Q.

$$Z' = Z*\cos(Q) + X*\text{sen}(Q)$$

$$X' = -Z*\text{sen}(Q) + X*\cos(Q)$$

$$Y' = Y$$

Y en notación matricial:

$$[\ X',\ Y',\ Z'\ ] = [\ X,\ Y,\ Z\ ] \begin{vmatrix} \cos(Q) & 0 & \text{sen}(Q) \\ 0 & 1 & 0 \\ -\text{sen}(Q) & 0 & \cos(Q) \end{vmatrix}$$

Finalmente, la rotación en el espacio con respecto a los ejes X, Y, Z (en ese orden) y el origen como pivote, se obtiene por una composición de las tres rotaciones anteriores. Dicha composición se obtiene multiplicando las tres matrices de rotación [13].

Tomando la convención de rotar primero con respecto al eje X, después con respecto al eje Y, y por último con respecto al eje Z, obtenemos que la matriz de transformación será:

$$Rx\ Ry\ Rz = \begin{vmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\text{sen}(\theta) \\ 0 & \text{sen}(\theta) & \cos(\theta) \end{vmatrix} * \begin{vmatrix} \cos(P) & 0 & \text{sen}(P) \\ 0 & 1 & 1 \\ -\text{sen}(P) & 0 & \cos(P) \end{vmatrix}$$

$$* \begin{vmatrix} \cos(Q) & -\text{sen}(Q) & 0 \\ \text{sen}(Q) & \cos(P) & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

$$= \begin{vmatrix} \cos P\cos Q & -\cos P\text{sen}Q & \text{sen}P \\ \cos\theta\text{sen}Q+\text{sen}\theta\text{sen}P\cos Q & -\text{sen}\theta\text{sen}P\cos Q+\cos\theta\cos Q & -\text{sen}\theta\cos P \\ -\cos\theta\text{sen}P\cos Q+\text{sen}\theta\text{sen}Q & \cos\theta\text{sen}P\text{sen}Q+\text{sen}\theta\cos Q & \cos\theta\cos P \end{vmatrix}$$

$$\begin{vmatrix} A & B & C \\ D & E & F \\ G & H & I \end{vmatrix}$$

Donde A, B, C, D, E, F, G, H, I son los respectivos términos de la matriz de arriba.

La transformación de cambio de escala afecta a cada coordenada por separado, tal como ocurre en dos dimensiones, y es:

$$X' = X * Sx$$

$$Y' = Y * Sy$$

$$Z' = Z * Sz$$

Y en notación matricial:

$$[\ X',\ Y',\ Z'\ ] = [\ X,\ Y,\ Z\ ] \begin{vmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & Sz \end{vmatrix}$$

La traslación determinada por (Tx, Ty, Tz) es tal que cualquier punto (X, Y, Z) será transformado en (X + Tx, Y + Ty, Z + Tz).

Al igual que en el caso de dos dimensiones no podemos dar una representación matricial para la traslación en el espacio usando matrices de 3 * 3. Por esta razón usaremos análogamente coordenadas homogéneas para el espacio.

3.2 GEOMETRIA PROYECTIVA EN EL ESPACIO

De las transformaciones analizadas hasta aqui, algunas de ellas mueven un objeto de un lugar a otro sin alterar en forma alguna al objeto en si, esto es, sólo efectúan un cambio de la posición del objeto en el espacio. A estas transformaciones se les llama movimientos rigidos y ejemplos de ellas son las traslaciones y las rotaciones. Sin embargo, hay otras transformaciones que deforman al objeto y son llamadas proyecciones. El estudio de este tipo de transformaciones y sus propiedades [11] es lo que da lugar a la geometria proyectiva en el espacio. No obstante, algunos de los teoremas contienen excepciones dentro de sus enunciados y para disolverlas se hace necesario "ampliar" o extender el espacio.

En el capitulo II se introdujo el concepto de plano extendido, el cual comprende a todos los puntos del plano euclidiano, llamados puntos finitos, más unos nuevos puntos denominados puntos al infinito que satisfacen ciertas propiedades. De esta misma manera, la extensión del espacio euclidiano o espacio finito, se efectúa por medio de la adjunción de puntos al infinito, con las siguientes propiedades [11]:

a) Hay un solo punto al infinito correspondiente a cada dirección en el espacio finito, esto es, al conjunto de todas las lineas en el espacio con una misma dirección le corresponde un solo punto al infinito.

b) Dada una dirección especifica, todos los puntos al infinito en las direcciones perpendiculares a ella forman una linea al infinito. Esto es, dada una dirección, al conjunto de todos los planos perpendiculares a ella le corresponde una linea al infinito.

c) Todos los puntos al infinito constituyen un plano, el plano al infinito.

Nótese que estas propiedades son una generalización de las propiedades de los puntos al infinito que se dieron en la extensión del plano.

Finalmente, al igual que en el plano, no se pueden utilizar coordenadas cartesianas para expresar a todos los puntos del plano extendido, pues todas las ternas (X, Y, Z) se utilizan como coordenadas para representar a los puntos finitos. Se hace necesario, por consiguiente, utilizar otro tipo de coordenadas, las coordenadas homogéneas para el espacio.

3.3 COORDENADAS HOMOGENEAS EN EL ESPACIO

Las coordenads homogéneas de un punto del espacio cartesiano están definidas en términos de las coordenadas cartesianas del mismo. Si (X, Y, Z) son las coordenadas cartesianas de un punto finito P, las coordenadas homogéneas (X1, X2, X3, X4) de P, se definen por las relaciones:

$$X1 / X4 = X$$

$$X2 / X4 = Y$$

$$X3 / X4 = Z$$

$$X4 <> 0$$

Las coordenadas homogéneas de los puntos al infinito son de la forma (X1, X2, X3, 0).

Si L es cualquier línea en la dirección cuyas componentes son l, m, n, y (X0, Y0, Z0) es un punto fijo sobre L, un punto finito arbitrario P sobre L tiene coordenadas cartesianas (X0 + lt, Y0 + mt, Z0 + nt) y coordenadas homogéneas (X0 + lt, Y0 + mt, Z0 + nt, 1) o bien (X0/t + l, Y0/t + m, Z0/t + n, 1/t). Cuando P se aleja indefinidamente sobre L, el parámetro t tiende a infinito y entonces las coordenadas homogéneas tienen como limite a (l, m, n, 0) así que, en general, el punto al infinito que le corresponde a la dirección con componentes X1, X2, X3 es aquel que tiene coordenadas homogéneas (X1, X2, X3, 0).

Por simplificación, usualmente se utiliza para los puntos finitos la representacion homogénea (X, Y, Z, 1).

3.4 TRANSFORMACIONES EN TRES DIMENSIONES CON COORDENADAS HOMOGENEAS

En la sección 3.1 se vio que muchas transformaciones se pueden expresar por medio de matrices [3] [4] [5] [8], la matriz de transformación, y que la concatenación o composición de varias transformaciones se traduce en el producto de sus respectivas matrices de transformación. Así, dado un punto (X, Y, Z) en el espacio y una transformación lineal que mapea (X, Y, Z) en (X', Y', Z'), ésta se puede expresar como:

$$(X', Y', Z') = (X, Y, Z) \begin{vmatrix} A & D & G \\ B & E & H \\ C & F & I \end{vmatrix}$$

donde la matriz 3 * 3 es su matriz de transformación. Recordemos tambien que ésto se podia hacer para las transformaciones lineales pero no para la traslación.

Con el uso de coordenadas homogéneas este problema se resuelve, pues en esta forma los puntos tienen representaciones (X, Y, Z, 1) y las matrices de transformación son matrices de 4 * 4. Así, si el punto (X, Y, Z, 1) se transforma en el punto (X', Y', Z', 1), esto se expresa por medio de:

$$(X', Y', Z', 1) = (X, Y, Z, 1) \begin{vmatrix} A & E & I & M \\ B & F & J & N \\ C & G & K & O \\ D & H & L & P \end{vmatrix}$$

que se traducen:

$$X' = AX + BY + CZ + D$$

$$Y' = EX + FY + GZ + H$$

$$Z' = IX + JY + KZ + L$$

$$1 = MX + NY + OZ + P$$

Haciendo un análisis semejante al de la sección 3.1, obtenemos que las representaciones de las rotaciones, cambios de escala y traslaciones son como sigue:

TRASLACION:

$$(X', Y', Z', 1) = (X, Y, Z, 1) \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ Tx & Ty & Tz & 1 \end{vmatrix}$$

CAMBIOS DE ESCALA:

$$(X', Y', Z', 1) = (X, Y, Z, 1) \begin{vmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

ROTACION EN EL EJE Z:

$$(X', Y', Z', 1) = (X, Y, Z, 1) \begin{vmatrix} \cos\theta & -\text{sen}\theta & 0 & 0 \\ \text{sen}\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

ROTACION EN EL EJE X:

$$(X', Y', Z', 1) = (X, Y, Z, 1) \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos P & -\text{sen}P & 0 \\ 0 & \text{sen}P & \cos P & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

ROTACION EN EL EJE Y:

$$(X', Y', Z', 1) = (X, Y, Z, 1) \begin{vmatrix} \cos Q & 0 & \text{sen}Q & 0 \\ 0 & 1 & 0 & 0 \\ -\text{sen}Q & 0 & \cos Q & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

ROTACION Rxyz:

$$(X', Y', Z', 1) = (X, Y, Z, 1) \begin{vmatrix} A & B & C & 0 \\ D & E & F & 0 \\ G & H & I & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

donde A, B, C, D, E, F, G, H, I son las componentes de la matriz 3 * 3 dada en la sección 3.1.

La rotación y el cambio de escala se pueden combinar multiplicando sus matrices obteniéndose así una transformación con representación matricial:

$$(X', Y', Z', 1) = (X, Y, Z, 1) \begin{vmatrix} ASx & BSy & CSz & 0 \\ DSx & ESy & FSz & 0 \\ GSx & HSY & ISz & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

De manera similar, se pueden combinar las transformaciones de rotación, escala y traslación, obteniéndose una nueva transformación con representación matricial:

$$(X', Y', Z', 1) = (X, Y, Z, 1) \begin{vmatrix} ASx & BSy & CSz & 0 \\ DSx & ESy & FSz & 0 \\ GSx & HSY & ISz & 0 \\ Tx & Ty & Yz & 1 \end{vmatrix}$$

Las anteriores transformaciones rotan con respecto a los ejes coordenados, pero en general, cualquier línea en el espacio sirve como eje de rotación. Este problema de rotación consiste en derivar una matriz de transformación para una rotación en un ángulo θ, alrededor de una línea arbitraria. La matriz de rotación correspondiente será construida en base a las matrices básicas descritas anteriormente.

El problema se resolverá de la siguiente manera: primero se efectuará una traslación para mover el origen a un punto sobre la línea, se harán rotaciones sobre los ejes X y Y para alinear el eje Z con la línea. De esta manera, el rotar con respecto a la línea es simplemente rotar con respecto al eje Z.
Finalmente se aplicarán las transformaciones inversas de las rotaciones sobre Y y X, y para la traslación, dado que hay que restaurar la línea y las coordenadas a su orientación original.

Una representación conveniente para una línea es mediante un punto sobre la línea y la dirección de ella. Esta representacion da ventajas al método ya que el punto provee información para la traslación, y la dirección nos ayudará en los ángulos de rotación para alinear el eje Z con la línea. Es posible encontrar esta forma, a partir de las ecuaciones paramétricas de una línea:

$$X = A*u + X1$$

$$Y = B*u + Y1$$

$$Z = C*u + Z1$$

Un punto sobre la linea es (X1, Y1, Z1) y la dirección es especificada por el vector (A, B, C) <> (0, 0, 0).

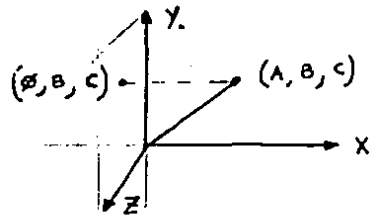
Para determinar la matriz de traslación en el caso de rotación en general, el paso inicial es mover el origen al eje de rotación, de esta forma tenemos:

$$T = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ Tx & Ty & Tz & 1 \end{vmatrix}$$

Después de la traslación, el punto (X1, Y1, Z1) sobre la linea es movido al origen y será necesaria la transformación inversa de esta traslación para restaurarlo. Esto será despues de la rotación.
La matriz de traslación inversa es:

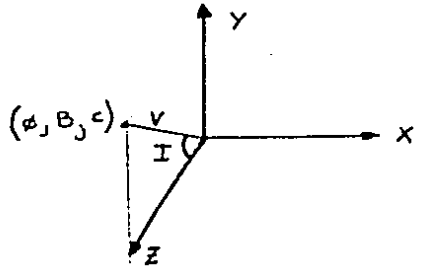
$$T' = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -Tx & -Ty & -Tz & 1 \end{vmatrix}$$

El siguiente paso es rotar con respecto al eje X, esta rotación se hara hasta que el eje general de rotación está en el plano XZ. Para determinar el ángulo de rotación, colocaremos el vector de dirección en el nuevo origen y consideraremos su proyección en el plano YZ. El segmento de linea entre (0, 0, 0) y (A, B, C) estará en la dirección del eje general de rotación y dado que ya habiamos hecho traslación, tenemos que el segmento está a lo largo del eje general de rotación. El segmento se proyecta al plano YZ y tenemos que la proyección es el segmento de (0, 0, 0) a (0, B, C). Vea el siguiente dibujo:



Rotaremos con respecto al eje X hasta que el eje (A, B, C)

esté en el plano XZ, lo cual es equivalente a que el segmento proyectado coincida con el eje Z. Así la rotación será de acuerdo al ángulo I, mostrado en el siguiente dibujo:



La longitud del segmento proyectado es:

$$V = (B*B + C*C)^{1/2} ,$$

Además:

$$\text{sen}(I) = B / V$$

$$\cos(I) = C / V ,$$

de tal forma que la rotación con respecto al eje X es:

$$Rx = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(I) & \text{sen}(I) & 0 \\ 0 & -\text{sen}(I) & \cos(I) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & C/V & B/V & 0 \\ 0 & -B/V & C/V & 0 \\ 0 & 0 & 00 & 1 \end{vmatrix}$$

La transformación inversa es una rotación de igual magnitud y en dirección opuesta. Recordando que la función coseno es par y la funcion seno es impar, tenemos que:

$$\cos(-I) = \cos(I) \quad \text{y}$$

$$\text{sen}(-I) = -\text{sen}(I) .$$

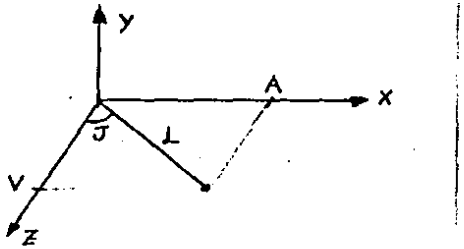
Podemos tener la matriz de la transformación inversa:

$$R'x = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & C/V & -B/V & 0 \\ 0 & B/V & C/V & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Ya que el eje de rotación esta en el plano XZ, y que conocemos la longitud del segmento, que es:

$$L = (A*A + B*B + C*C)^{1/2}$$

Podemos ver el ángulo de rotación J, con respecto al eje Y, para que coincidan el eje Z y el eje general de rotación.



De la definición de seno y coseno tenemos:

$$sen(J) = A / L$$

$$cos(J) = V / L$$

Y la matriz de de rotación del eje Y con un ángulo J es:

$$Ry = \begin{vmatrix} cos(J) & 0 & sen(J) & 0 \\ 0 & 1 & 0 & 0 \\ -sen(J) & 0 & cos(J) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} V/L & 0 & A/L & 0 \\ 0 & 1 & 0 & 0 \\ -A/L & 0 & V/L & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

La inversa para esta transformación es:

$$R'y = \begin{vmatrix} V/l & 0 & -A/L & 0 \\ 0 & 1 & 0 & 0 \\ A/l & 0 & V/l & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Finalmente estamos en posición de rotar con respecto al eje general, en un ángulo θ. Y dado que el eje general coincide con el eje Z, será suficiente rotar con respecto al eje Z y tenemos:

$$Rz = \begin{vmatrix} \cos(\theta) & -\text{sen}(\theta) & 0 & 0 \\ \text{sen}(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Por lo tanto, para tener la transformación de rotación en cualquier eje, en un ángulo θ, se encontrará la transformación producto:

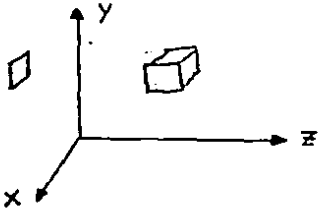
$$R\theta = T \; Rx \; Ry \; Rz \; R'y \; R'x \; T'$$

y Rθ será la matriz que rotará en el eje dado y en el ángulo deseado.

3.5 PROYECCIONES

Se ha comentado sobre la transformación de objetos en tres dimensiones, pero ya que nuestra superficie de visión (pantalla, graficador, etc.) es de dos dimensiones solamente, debemos tener posibilidad de proyectar nuestro objeto tridimensional a un dispositivo de dos dimensiones [3] [5] [8] [12]. El dispositivo que usaremos será la pantalla. La forma más sencilla es descartar la coordenada Z. Este es un caso especial del método de "proyección paralela". Una proyección paralela es formada trazando líneas paralelas desde cada vértice en el objeto hasta el plano de intersección correspondiente a la pantalla. El punto de intersección es la proyección del vértice. Después se conectan en la pantalla los vértices proyectados, mediante segmentos que correspondan a conexiones en el objeto original.

El caso especial de descartar la coordenada Z, es cuando la pantalla o superficie de visión, es paralela al plano XY, y las líneas de proyección son paralelas al eje Z. Así, las coordenadas X y Y permanecen constantes y la coordenada Z es la que cambia.
La forma más sencilla es cuando el centro del objeto es identificado con el centro de la pantalla y el eje Z es perpendicular a la pantalla. Además los ejes X y Y del objeto tienen el mismo sentido que los ejes X y Y de la pantalla, por lo tanto, la imagen proyectada es formada de las coordenadas X y Y del objeto y la coordenada Z es descartada.
Ver el siguiente dibujo:

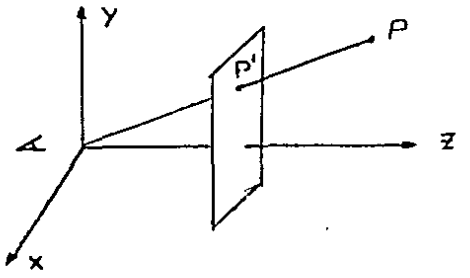


Por ejemplo, la proyección paralela del cubo (+-1, +-1, +-1) aparecerá en la pantalla como el cuadrado (+-1, +-1).

La proyección paralela es una técnica útil para mostrar figuras tridimensionales, sin embargo no toma en cuenta la perspectiva. En el mundo real las líneas paralelas que se alejan dan la impresión de irse juntando.

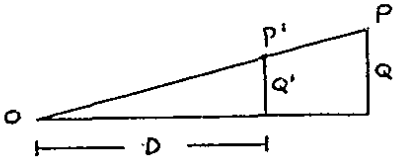
La segunda forma de proyectar objetos tridimensionales a dos dimensiones es la "proyección perspectiva". Como ya

sabemos, el ojo no puede ver todo el espacio, y su visión está limitada a un cono de rayos que llegan a la retina. Para formar la visión perspectiva del espacio, se coloca un "plano perspectivo" perpendicular al eje del "cono de visión" y simplemente proyectamos cada punto del objeto al plano, como se muestra en el siguiente dibujo:



Este plano perpendicular al eje del cono de visión se hace corresponder con el plano del dispositivo de graficación (pantalla, graficador, etc.), de tal manera que las coordenadas del punto proyectado en la pantalla P'=(Xs, Ys) pueden ser calculadas fácilmente a partir de las coordenadas del punto P=(Xe, Ye, Ze) del sistema de coordenadas del usuario (ver posición del ojo en el dibujo anterior).

Considere el plano YeZe, que se muestra a continuación:



Los triángulos OQ'P' y OQP son semejantes, de donde obtenemos la relación:

$$Ys / D = Ye / Ze$$

De una manera similar, en el plano XeZe, obtenemos:

$$Xs / D = Xe / Ze$$

Ya que se tienen los números Xs y Ys, se afectarán por el tamaño de la pantalla. De tal manera tenemos:

$$Xs = (D * Xe) / (S * Ze)$$

$$Ys = (D * Ye) / (S * Ze)$$

Y en notación matricial:

$$(Xs, Ys) = (Xe, Ye) \begin{vmatrix} D / S*Ze & 0 \\ 0 & D / S*Ze \end{vmatrix}$$

Las coordenadas Xs y Ys correspondientes a la pantalla, podrán ser convertidas a coordenadas de pantalla correspondientes a una región dada (puerto de visión).

Las coordenadas correspondientes al puerto de visión son:

$$Xs = ((D * Xe) / (S * Ze)) * Vsx + Vcx$$

$$Ys = ((D * Ye) / (S * Ze)) * Vsy + Vcy$$

donde el centro del puerto de visión es el punto (Vcx, Vcy) y tiene 2*Vsx unidades de ancho y 2*Vsy unidades de alto.

La perspectiva es una transformación fundamentalmente diferente a la rotación, traslación y escala, dado que divide por el valor de la coordenada Ze. Mientras que las otras involucran solamente multiplicaciones y sumas.

La generación de una imagen verdaderamente perspectiva requiere dividir cada punto entre su profundidad. Afortunadamente, la imagen perspectiva de una línea puede ser generada fácilmente, transformando solamente sus puntos extremos, y dibujando la línea entre los dos puntos obtenidos.

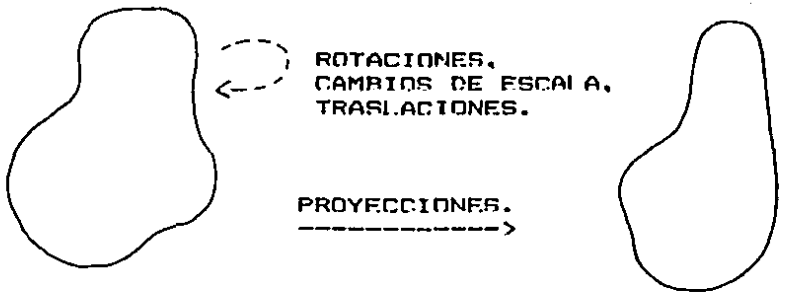
CAPITULO IV

DISEÑO DE EDMOL

4.1 DISEÑO CONCEPTUAL

EDMOL trabaja con un conjunto de (representaciones de) objetos tridimensionales: el conjunto de las moléculas. Dichas moléculas están constituidas por átomos y ligaduras.

Podemos considerar las transformaciones que realiza EDMOL como funciones de moléculas en ellas mismas, o sea:



restringiéndonos (por la aplicación) principalmente a las rotaciones, cambios de escala, traslaciones, proyecciones y composiciones de ellas. La proyección en perspectiva pasa de una representación en tres dimensiones a una representación en dos dimensiones, para poder desplegar en pantalla la molécula.

En el caso de EDMOL, las representaciones son en base a listas de vecinos (parejas de átomos que conecta), permitiendo posteriores modificaciones de representación, como por ejemplo, a matriz de adyacencia [15] [16].

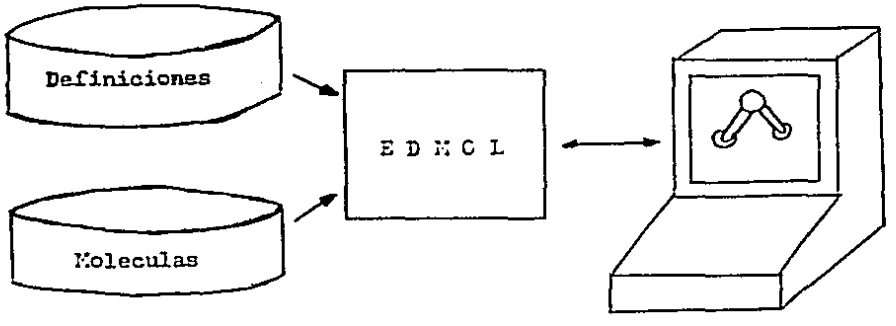
Las transformaciones permiten al usuario (con ayuda de la computadora) estudiar estructuras moleculares dando una mejor visión de su topología, ya que las podrá "observar" desde distintos lados y distancias.

Este trabajo pretende poner las bases para el desarrollo de sistemas gráficos ("hechos en México") de ayuda en Química, Física, Astronomía y otras ciencias.

Teniendo como punto de partida a EDMOL, se puede pensar en complementarlo con algoritmos de simulación, análisis de comportamiento molecular, animación, etc.

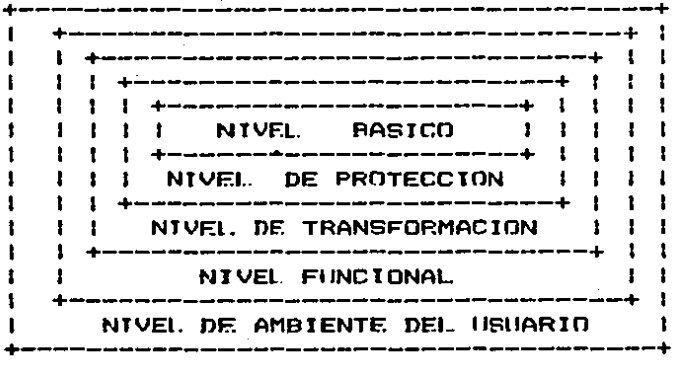
4.2 DESCRIPCION DE EDMOL

EDMOL es un sistema escrito en Pascal [16] [17] para computadoras personales y posibilidades gráficas. Es necesario espacio en disco para el almacenamiento de las moléculas, tal como se muestra en la siguiente figura:



La comunicación del usuario con EDMOL es por medio de menús, que permiten un mejor manejo de las opciones permitidas.

EDMOL tiene una estructura en capas, como se muestra a continuación:



El nivel básico está constituido por procedimientos gráficos primitivos como la inicialización, pintado de un punto, de una línea, de un círculo, etc.

El nivel de protección está constituido por procedimientos que impiden que zonas ajenas a la de trabajo sean invadidas por las figuras a proyectarse.
Ejemplos de este nivel son las definiciones de ventanas y la realización de recorte de los segmentos a dibujar.

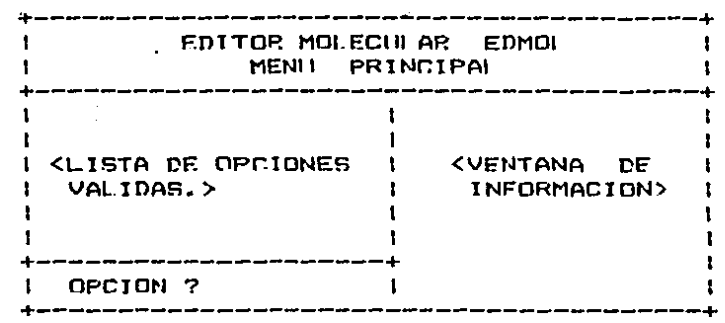
Los dos niveles anteriores están basados en las características dadas por el lenguaje Pascal [16] en una computadora personal (compatible con IBM). Y para efectuar alguna modificación (o instalación en una computadora diferente) se harán modificaciones a estos dos niveles. En el apéndice A existen procedimientos basicos escritos en Pascal.

El nivel de transformación está constituido por procedimientos que afectan ya sea al objeto, como a su vista. Ejemplos de este nivel son las rotaciones, cambios de escala, traslaciones y proyecciones.

El nivel funcional está constituido por los procedimientos que realizan las opciones permitidas por EDMOL. Estos procedimientos son la parte más importante del sistema, dado que están construidos específicamente para la función deseada.
Muestras de estos procedimientos son los que manejan los archivos, el ordenamiento de los datos, la edición de la molécula, tratamiento al problema de líneas ocultas, etc.

El nivel de ambiente del usuario está constituido por los procedimientos que mantienen la comunicación con el usuario, ya sea por menús o por mensajes específicos.

El menú principal de EDMOL, tiene la siguiente forma:



Este menú permite 12 opciones (A, B, C, ... K, ?), que son:

A) LISTA MOLECULAS EXISTENTES.

Muestra en pantalla la lista de los nombres (de 20 caracteres como máximo) de todas las moléculas existentes en el disco. Dichas moléculas estarán numeradas para permitir al usuario una selección mediante el número.

B) SELECCIONA MOLECULA ACTIVA.

Pregunta al usuario por una molécula, la cual será asignada como molécula principal. El usuario dará el número de la molécula. Algunos comandos de EDMOL asumen que el usuario ya escogió la molécula principal, de tal manera que los comandos afectan a la molécula elegida. La forma de dar el número de la molécula es:

<número> <CR>

Cuando se ha dado un número válido, el usuario verá en la parte superior del menú el nombre de la molécula activa.

C) SELECCIONA VISTA ACTIVA.

EDMOL informa al usuario el número de vistas existentes de la molécula activa (comando B) y posteriormente pregunta al usuario el número de vista que desea. Si el usuario da la vista 0, entonces se muestran en la pantalla todas las vistas existentes en forma secuencial, permitiendo seleccionar la que más le convenga. Existe la posibilidad de tener 99 vistas de la misma molécula. Nuevamente esto dependerá de la capacidad del disco. La forma de dar el número de vista es:

<número> <CR>

D) MUESTRA LA MOLECULA ACTIVA.

Muestra en la pantalla la molécula activa (comando B) con la vista seleccionada (comando C). La molécula se verá a color (si se tiene el monitor) y con proyección en perspectiva para una mejor visión. Con este comando se pasa a otro menú, el cual permitirá al usuario, en forma interactiva, rotar la molécula en cualquier dirección y sentido, teniendo en forma inmediata las rotaciones con respecto a los ejes X, Y y Z, con un angulo de 15 grados. Los comandos correspondientes son la letras "X", "x", "Y", "y", "Z", "z". Además de las rotaciones, se tiene la posibilidad de escalar la molécula, pudiendo agrandarla o achicarla según se desee. Los comandos correspondientes son "G" y "P", sin importar mayúsculas o minúsculas.
El nuevo menú permite crear en el disco nuevas vistas con sólo dar el comando "S".
Como última posibilidad del nuevo menú se permite ver la molécula sin ligaduras, es decir, solamente los átomos. Esto es con el comando "B".
Para regresar al menú principal hay que teclear el comando "F" o "f".

E) MUESTRA DOS MOLECULAS.

Pregunta al usuario el nombre de la segunda molécula a graficarse. Esta molécula estará fija mientras que la molécula activa se podra afectar por el menú descrito en el comando D.

F) CREA UNA NUEVA MOLECULA.

Se pregunta por el nombre de la nueva molécula, asignándola como principal y pasando al menú de edición. Con el nuevo menú es posible insertar átomos, insertar ligaduras, borrar átomos, borrar ligaduras, listar todos los átomos y listar todas las ligaduras.
Todos los comandos del menú de edición se dan por medio del número de comando, o sea:

<número> <CR>

Si no se ha dado el <CR> es posible rectificar el número de comando.

G) BORRA MOLECULA ACTIVA.

Borra definitivamente la molécula activa (ver el comando B). Este comando suprime del disco todas las vistas existentes de la molécula activa. Por protección se pregunta

una vez más si es que el usuario está seguro de borrar la molécula.

H) BORRA VISTA ACTIVA.

Pregunta al usuario por el número de vista a borrar, verificando que dicha vista exista. La vista borrada corresponderá a la molécula activa (ver comando B). Un reacomodo de las vistas es hecho, no es conveniente borrar varias vistas a la vez, ya que hay que ver los nuevos valores usando el comando "C".

I) MODIFICA MOLECULA ACTIVA.

Muestra el menú de edición de moléculas para la molécula activa permitiendo insertar y borrar átomos, insertar y borrar ligaduras, así como listar todos los átomos y todas las ligaduras. Este comando borrará todas las vistas de la molécula anterior, ya que no corresponden con la molécula modificada. La nueva molécula tendrá la vista número uno únicamente. Por protección se pregunta al usuario si está seguro de querer modificar la molécula.

J) COPIA LA MOLECULA ACTIVA.

Permite crear nuevas moléculas con otros nombres a partir de una existente. Esto posibilita construir moléculas cuya estructura contiene a una molécula ya definida en EDMOL.

?) DESCRIPCION DE COMANDOS.

Muestra en pantalla una breve descripción de las instrucciones.

4.3 ALGORITMOS IMPORTANTES DE EDMOL

EDMOL es un programa escrito en PASCAL que consta de 1600 líneas aproximadamente, sin embargo, tiene una estructura simple.

Los algoritmos fundamentales de EDMOL son: rotación sobre cualquier eje, proyección en perspectiva de la molécula y solución al problema de líneas ocultas.

4.3.1 ROTACION SOBRE CUALQUIER EJE

La rotación sobre cualquier eje se discutió en la sección 4 del capítulo III, obteniéndose la transformación:

$$R\theta = T\ Rx\ Ry\ Rz\ R'y\ R'x\ T'$$

sin embargo, Rθ contempla una traslación del origen al eje de rotación (con su respectiva inversa), la cual no será necesaria en EDMOL, dado que las rotaciones requeridas por los usuarios son con respecto al origen.
Esto nos lleva a la transformación:

$$R\xi = Rx\ Ry\ Rz\ R'y\ R'x$$

donde:

$$Rx = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(I) & \text{sen}(I) & 0 \\ 0 & -\text{sen}(I) & \cos(I) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & C/V & B/V & 0 \\ 0 & -B/V & C/V & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$R'x = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & C/V & -B/V & 0 \\ 0 & B/V & C/V & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$
Ry = \begin{vmatrix} \cos(J) & 0 & \operatorname{sen}(J) & 0 \\ 0 & 1 & 0 & 0 \\ -\operatorname{sen}(J) & 0 & \cos(J) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} V/L & 0 & A/L & 0 \\ 0 & 1 & 0 & 0 \\ -A/L & 0 & V/L & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}
$$

$$
R'y = \begin{vmatrix} V/L & 0 & -A/L & 0 \\ 0 & 1 & 0 & 0 \\ A/L & 0 & V/L & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}
$$

$$
Rz = \begin{vmatrix} \cos(\bar{\varphi}) & -\operatorname{sen}(\bar{\varphi}) & 0 & 0 \\ \operatorname{sen}(\bar{\varphi}) & \cos(\bar{\varphi}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}
$$

4.3.2 PROYECCION EN PERSPECTIVA

La proyección en perspectiva se discutió en la sección 5 del capítulo III, de donde se obtenían las coordenadas relativas al puerto de visión:

$$Xs = ((D * Xe) / (S * Ze)) * Vsx + Vcx$$

$$Ys = ((D * Ye) / (S * Ze)) * Vsy + Vcy$$

donde el centro del puerto de visión es el punto (Vcx, Vcy) y tiene 2*Vsx unidades de ancho y 2*Vsy unidades de alto. Con S determinando el tamaño de la pantalla y D la distancia del ojo al plano de proyección.

EDMOL utiliza constantes definidas globalmente para determinar el tamaño de la pantalla, la distancia del ojo al plano de proyección y el tamaño del puerto de visión.
La distancia del ojo al objeto se almacena en una variable para dar facilidad al usuario de ver el objeto a diferentes distancias.
De esta manera, la proyección en perspectiva de un punto del objeto cuyas coordenadas son (Objeto.x, Objeto.y, Objeto.z) se calcula de la siguiente manera:

```
Xs = ((OjoAPantalla * Objeto.x) / (AltoPantalla * Objeto.z))
                        * AltoPantalla + XPantalla

Ys = ((OjoAPantalla * Objeto.y) / (AnchoPantalla * Objeto.z))
                        * AnchoPantalla + YPantalla
```

Note que la pantalla y el puerto de visión de EDMOL tienen el mismo tamaño , de donde se pueden cancelar términos.
En EDMOL se dejaron tal cual los términos para posteriores modificaciones.

Ya que se tiene el cálculo de la proyección en perspectiva de un punto, podremos tener la proyección de toda la molécula. Como la molécula está formada por átomos y ligaduras (donde los átomos son esferas y las ligaduras son segmentos que unen a los átomos) es posible calcular la proyección fácilmente.

Para calcular la proyección en perspectiva de un átomo, se calcula la proyección en perspectiva del centro del átomo,

así como su tamaño. Y para las ligaduras es suficiente con calcular las proyecciones de los puntos extremos y pintar el segmento entre los dos puntos obtenidos.

4.3.3 LINEAS OCULTAS EN EDMOL

Para pintar cuerpos opacos, los programas de graficado deben decidir cuáles partes son visibles y cuáles ocultas, de tal manera que las partes ocultas no sean pintadas.

En la vida real, el material opaco de los objetos obstruye los rayos de luz que provienen de las partes ocultas.

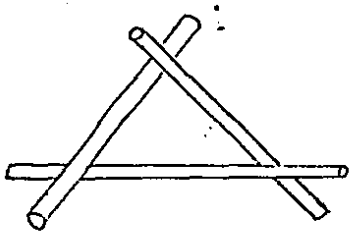
Al generar por computadora una imagen, la eliminación no es automática cuando los objetos son proyectados en la pantalla. La tarea de decidir cuáles partes de un objeto deben mostrarse y cuáles omitirse fue originalmente conocido como el " PROBLEMA DE LAS LINEAS OCULTAS " [3] [5] [8] [14].

El uso de algoritmos para resolver los problemas de líneas ocultas permiten obtener imágenes más realistas.

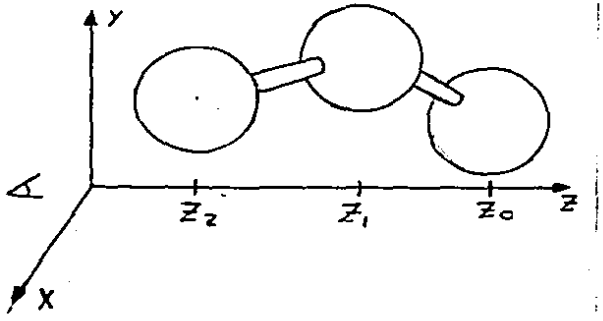
Para resolver el problema de líneas ocultas en EDMOL, se diseñó una variación del " ALGORITMO DEL PINTOR " dado que no se tiene un algoritmo que lo resuelva. Este algoritmo obtiene su nombre de la manera en que se crea una pintura al óleo.
El artista principia con una capa base de algún color, la cual puede llenar toda el área de trabajo. Posteriormente pinta los objetos que estan más alejados y continúa pintando los objetos que están más cercanos a él.
Si el artista determina un orden en el pintado de los objetos, digamos en base a la distancia de los objetos a pintar, tendrá que los objetos más cercanos tapan a los más lejanos. Y como se usa pintura, no es necesario ir borrando.

En el caso de EDMOL, el área de trabajo es la pantalla y los objetos son círculos y líneas, los cuales se van pintando en el orden dado por el algoritmo del pintor. Es decir, se pintan primero los objetos más alejados del observador. Después se pintan los que le siguen en distancia, y asi, sucesivamente, hasta terminar por pintar los más cercanos.

La variación del algoritmo del pintor se debió a que no es posible dar un orden total para pintar los átomos y las ligaduras, donde los átomos se determinan por un punto en tres dimensiones y las ligaduras por dos puntos en tres dimensiones, pues se pueden presentar situaciones como la de la figura siguiente:



Para resolver el problema de orden, se diseñó un algoritmo que crea una partición del espacio y así obtiene un orden parcial, permitiendo pintar los objetos en forma " ordenada".
La partición del espacio se puede ver fácilmente en forma gráfica, la cual describiremos primero y terminaremos con la forma implementada en EDMOL.
En forma gráfica se tendrá un plano por cada átomo de la(s) molécula(s), donde dichos planos serán paralelos al plano XY y estarán colocados en la coordenada Z correspondiente al átomo que lo determinó.
Numeraremos a los planos de la forma Z0, Z1, Z2,...,Zn, tal como se muestra en el dibujo siguiente:



La sucesión de planos dividen el espacio en rebanadas, de tal manera que el algoritmo del pintor será enfocado a las rebanadas de espacio. Es decir, primero se pintará la rebanada más lejana al observador (Z0), después la siguiente (Z1) y así sucesivamente, hasta terminar con todas.

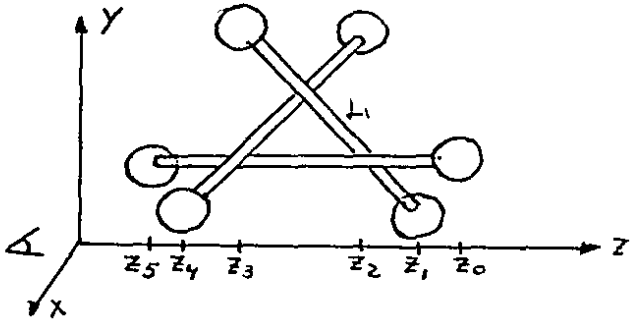
Cuando se tiene determinada la rebanada, es necesario conocer qué parte de la molécula está contenida en ella. Este problema se resuelve efectuando cortes de la molécula con los planos correspondientes.
Una vez que se tiene la parte de la molécula a graficar, hay

que seguir un orden de graficado dentro de la rebanada de espacio. Este orden se obtiene pintando primero las partes más alejadas del eje Z, para despues pintar las más cercanas.

En la realización de EDMOL se utiliza el algoritmo "Quick Sort" [9] [15] para ordenar los planos en base a la coordenada Z de los átomos. De esta forma, se toma un plano y antes de pintar el átomo que lo generó, se pintan todas las ligaduras que están detrás del plano.

A todas la ligaduras o partes de ligaduras que están pendientes para pintarse, se les llamó las "HISTORIAS". Esto es porque una ligadura puede atravesar varios planos, en cuyo caso se irá pintando tramo por tramo, dependiendo de la rebanada que se esté procesando.

Por ejemplo, en el siguiente dibujo:

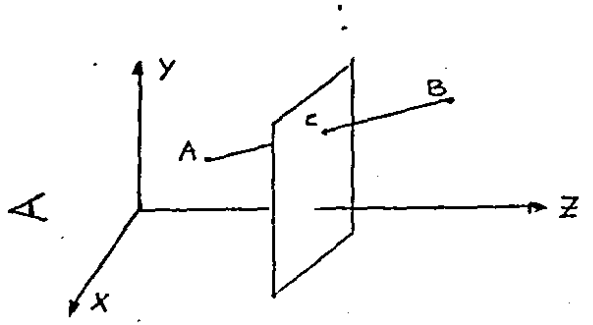


la ligadura L1 se pintará en dos partes, la parte que está en la rebanada Z0, Z1 y despues la parte que esta en la rebanada Z1, Z2 (ver la posición del ojo).
Hay que notar que una ligadura pasará a ser parte de las historias cuando tenga conexión con el átomo que generó el plano de corte, ésto, obviamente, si es que la ligadura no se ha pintado ya.

Existen dos casos al pintar una historia:

a) Si la historia está totalmente contenida en la rebanada entonces se pintará y se suprimirá de la lista de historias.

b) Si la historia NO está totalmente contenida, quiere decir que existe un tramo de dicha historia en rebanadas posteriores. Por esta razón se realizará la intersección de la historia con el plano [2] [7].
Dicha intersección puede verse en el siguiente dibujo:



en que las coordenadas (X', Y', Z') del punto C son:

$$X' = \text{CONSTANTE} * (Xb - Xa) + Xa$$

$$Y' = \text{CONSTANTE} * (Yb - Ya) + Ya$$

$$Z' = \text{VALOR CORRESPONDIENTE AL PLANO.}$$

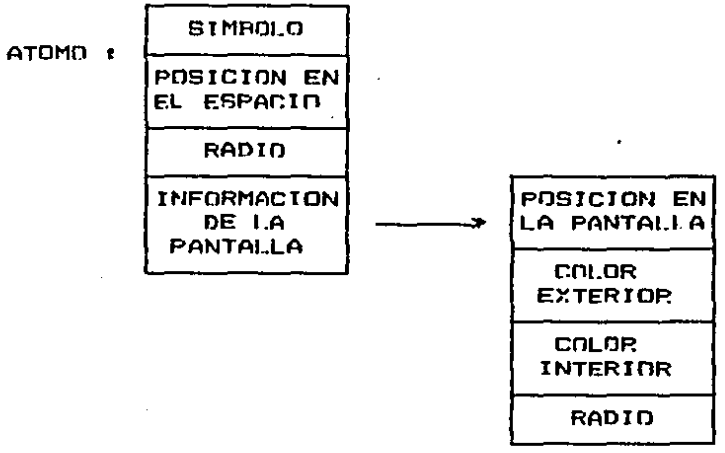
Entonces se pinta el segmento BC. El tramo AC pasa a ser parte de las historias para que en la rebanada siguiente se haga el mismo análisis descrito aquí.

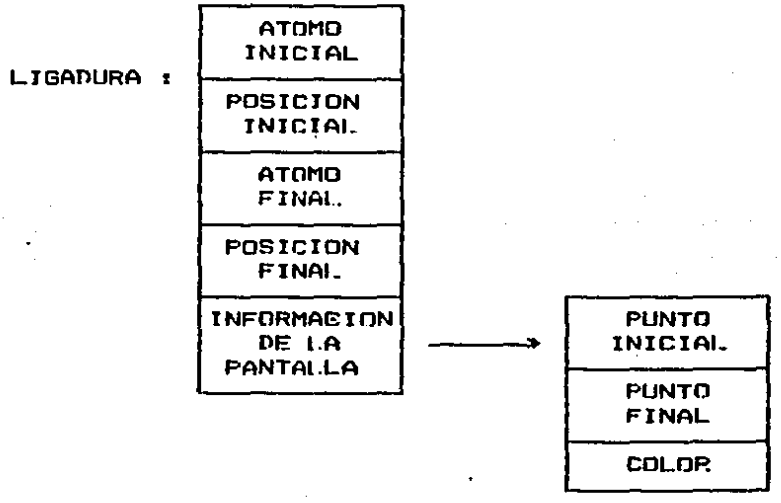
Como el número de historias varía dependiendo del número de ligaduras en la molécula y de la vista que se desee de ella, se usa una lista ligada para almacenar las historias [15] [16] [17].

4.4 ESTRUCTURA DE DATOS DE EDMOL

En esta sección describiremos las estructuras más importantes de EDMOL, principiando con las básicas.

Las estructuras básicas son los átomos y las ligaduras, que tienen la siguiente forma:

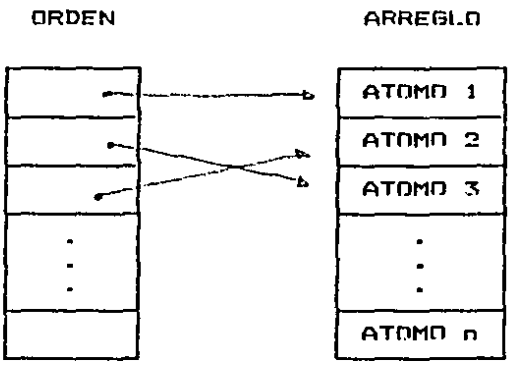




Es decir, las estructuras se llaman igual que los objetos y tienen en sub-campos de sus registros la información

correspondiente a la pantalla. Esta información es calculada cuando se realiza la proyección en perspectiva. Mientras que la información que corresponde a tres dimensiones es actualizada al leer una molécula de disco o insertar nuevos átomos o ligaduras.

En un nivel superior tenemos un arreglo de apuntadores llamado ORDEN que tiene la forma:



Este arreglo facilita el trabajo de ordenar los átomos para ser usados en el algoritmo de líneas ocultas. Esto permite el ordenamiento por intercambio, y para hacer un intercambio basta con cambiar los apuntadores y no la información.

El siguiente nivel lo componen las moléculas, teniendo la siguiente forma:

MOLECULA: :

ARREGLO DE ATOMOS
NUMERO DE ATOMOS
ARREGLO DE LIGADURAS
NUMERO DE LIGADURAS
ARREGLO PARA EL ORDEN
MATRIZ DE ROTACION

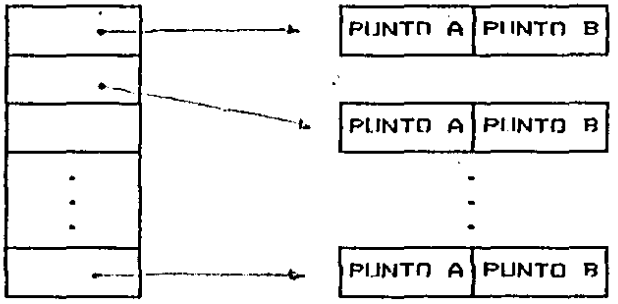
Asi, el último nivel esta formado por un conjunto de moléculas.

CONJUNTO DE MOLECULAS :

ARREGLO DE MOLECULAS
POSICION PRINCIPAL
MOLECULA PRINCIPAL

Estas dos últimas estructuras son manejadas principalmente por los procedimientos que realizan las operaciones descritas en los menús.

Una estructura auxiliar de EDMOL, que es muy importante, es la usada en el algoritmo de lineas ocultas. En la sección anterior se describieron las historias, y como el manejo de ellas requiere un ordenamiento, se utiliza un arreglo de apuntadores, es decir:



Por último, se mostrarán las tablas de información, que son cargadas en el proceso de inicialización de EDMOL.

DIRECTORIO

NOMBRE DE LA MOLECULA	NUMERO DE VISTAS EXISTENTES
.	.
.	.
.	.

COLORES

SIMBOLO	COLOR EXTERIOR	COLOR INTERIOR	RADIO
.	.	.	.
.	.	.	.
.	.	.	.

CONCLUSIONES

EDMOL sienta las bases gráficas para el desarrollo de posteriores sistemas de graficación en Química y otras áreas que involucren la graficación de moléculas.

El editor molecular EDMOL supera a los sistemas que se venden en la actualidad, esta superioridad va desde características elementales como los colores y tamaños de átomos, hasta el hecho de mostrar dos moléculas a la vez o el de tener 99 posibles vistas de las moléculas.

En el futuro, EDMOL tendrá un proceso de depuración en la Facultad de Química de la U. N. A. M. . Dicho proceso permitirá un mejoramiento del sistema en diversos aspectos como el gráfico, pedagógico, etc. En particular, nos permitirá sensibilizarnos en cuanto a un uso de computadoras en la educación.

En cuanto a experiencias con el paquete, mencionaremos que EDMOL fue usado por primera vez en el Instituto de Materiales de la U. N. A. M. con resultados satisfactorios. Esta primer experiencia ayudó mucho para tener nuevas opciones, las cuales estan contenidas en la versión actual de EDMOL.

Como resultado de otras experiencias se consideran pertinentes las siguientes modificaciones a corto plazo:

i) Modificar el editor para tener mayor facilidad en la creación y/o modificación de las moléculas.

ii) Mejorar la representación gráfica de las ligaduras, dibujándolas como cilindros en vez de lineas en el espacio. Esta representación dará mayor realismo a la gráfica.

iii) Incrementar el menú gráfico con opciones como la rotación de partes de la molécula o la inserción de moléculas completas.

Una modificación a mediano plazo será el diseñar un cuantificador de valencias, el cual determinará si el número de ligaduras conectadas a un átomo es correcto.

APENDICE A

ALGORITMOS PARA TRAZADO DE LINEAS.

En este apéndice se muestran dos procedimientos [5] [8] para pintar lineas con base en una secuencia de puntos. Estos procedimientos serán de gran ayuda para los usuarios cuyas computadoras no tengan la posibilidad de pintar lineas, pero si tengan la posibilidad de pintar puntos.

El primer procedimiento es conocido como el algoritmo DDA (Digital Differential Analyzer). Mientras que el segundo es conocido como el algoritmo de Bresenham.

```
PROCEDURE  DDA( x1, y1, x2, y2 : INTEGER );
VAR    length, i : INTEGER;
       x, y,
       xincrement,
       yincrement: REAL;
BEGIN
length := ABS( x2 - x1 );
IF ABS( y2 - y1 ) > length THEN length := ABS( y2 - y1 );
xincrement := ( x2 - x1 ) / length;
yincrement := ( y2 - y1 ) / length;
x := x1 + 0.5;  y := y1 + 0.5 ;
FOR i:=1 TO length DO
       BEGIN
       PLOT( TRUNC(x), TRUNC(y) );
       x := x + xincrement;
       y := y + yincrement;
       END;
END;
```

El algoritmo de Bresenham mostrado a continuación asume que (y2 - y1) < (x2 - x1). Donde y2, y1, x2, x1 son números enteros.

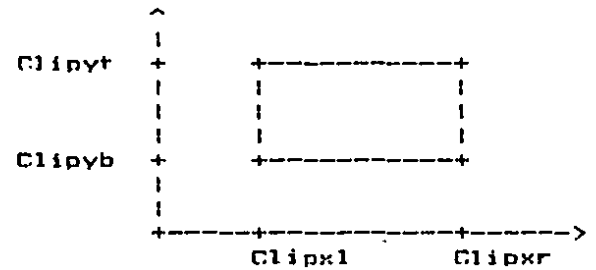
```
PROCEDURE  BRESENHAM( x1, y1, x2, y2 : INTEGER );
VAR    x, y,
       deltax,
       deltay : INTEGER;
       e      : REAL;
BEGIN
x := x1;
y := y1;
deltax := ( x2 - x1 );
deltay := ( y2 - y1 );
e := ( deltay / deltax ) - 0.5;
```

```
FOR i:=1 TO deltax DO
      BEGIN
      PLOT( x, y );
      IF ( e > 0 ) THEN
            BEGIN
            y := y + 1;
            e := e - 1;
            END;
      x := x + 1;
      e := e + ( deltay / deltax );
      END;
END;
```

APENDICE B

ALGORITMO DE RECORTE

El algoritmo de recorte que se describirá fue escrito por Dan Cohen e Ivan Sutherland [8]. El algoritmo determina primero si la línea está totalmente contenida en el puerto de visión o está totalmente fuera del puerto de visión. Si no satisface ninguno de estos dos casos, entonces el segmento es dividido en dos parte y los dos análisis anteriores son aplicados nuevamente. Este algoritmo depende del hecho de que toda la línea esta totalmente dentro del puerto de visión o puede ser dividida de tal forma que una parte pueda ser descartada.
El análisis es implementado extendiendo los bordes de la pantalla de tal forma que el espacio ocupado por el objeto sin recorte este dentro de nueve regiones. La simplicidad del algoritmo es mostrada por la siguiente implementación en PASCAL la cual asume que el recorte se efectuara con:



El código es:

```
VAR Clipxl, Clipxr, Clipyb, Clipyt : REAL;

PROCEDURE Clip( x1, y1, x2, y2 : REAL );
   LABEL return;
   TYPE edge=(left, right, bottom, top );
        outcode = SET OF edge;
   VAR  c, c1, c2 : outcode;
        x, y      : REAL;

PROCEDURE code( x, y : REAL; VAR c : outcode );
BEGIN
c:=[];
```

```
IF x < Clipxl THEN c:=[left] ELSE IF x > Clipxr
                THEN c:=[right];
IF y < Clipyb THEN c:= c + [bottom] ELSE
                IF y > Clipyt THEN c:=c+[top];
END;

BEGIN
code( x1, y1, c1); code( x2, y2, c2 );
WHILE (c1 <> []) OR (c2<>[]) DO
       BEGIN
       IF (c1*c2) <> [] THEN GOTO return;
       c:=c1; IF c=[] THEN c:=c2;
       IF left IN c THEN BEGIN
          y:=y1+(y2-y1)*(Clipxl-x1)/(x2-x1);
          x:=Clipxl
          END ELSE
       IF right IN c THEN BEGIN
          y:=y1+(y2-y1)*(Clipxr-x1)/(x2-x1);
          x:=Clipxr
          END ELSE
       IF bottom IN c THEN BEGIN
          x:=x1+(x2-x1)*(Clipyb-y1)/(y2-y1);
          y:=Clipyb;
          END ELSE
       IF top IN c THEN BEGIN
          x:=x1+(x2-x1)*(Clipyt-y1)/(y2-y1);
          y:=Clipyt
          END;
       IF c=c1 THEN BEGIN
                x1:=x; y1:=y; code(x,y,c1)
                END ELSE BEGIN
                x2:=x; y2:=y; code(x,y,c2)
                END
       END;
showline( x1, y1, x2, y2 );
return: END;
```

APENDICE C

DESCRIPCION DE LOS PROCEDIMIENTOS.

En este apéndice se muestra una descripción de todos los procedimientos de EDMOL. Dichos procedimientos se listan en el mismo orden que tienen dentro del programa.

La descripción consta de comentarios informales acerca de:

A) Significado de los parámetros de los procedimientos.

B) Restricciones en los valores de los parámetros.

C) Breve descripción del procedimiento.

D) Efectos laterales.

E) Valores de los parámetros de salida.

Esta descripción facilitará el estudio de EDMOL para posteriores correcciones a usuarios con poca experiencia en graficación por computadora. Además facilitaría una formalización posterior, tendiente a demostrar propiedades del procedimiento y programas que lo usen.

FUNCION Dist2Dim(P1, P2 : Pto2Dim) : ENTERO;
- A) Recibe como parámetro dos puntos P1 y P2 de la forma (P1.x, P1.y) , (P2.x, P2.y).
- B) Los datos deben ser números enteros.
- C) Calcula la distancia entre dos puntos, en dos dimensiones. Los puntos deben ser distintos.
- D) No hay efectos laterales.
- E) El valor de la función es la distancia entera entre los dos puntos en Dist2Dim.

FUNCION Dist3Dim(P1, P2 : Pto3Dim) : REAL;
- A) Recibe como parámetro dos puntos P1 y P2 de la forma (P1.x, P1.y, P1.z) , (P2.x, P2.y, P2.z).
- B) Los datos deben de ser números reales.
- C) Calcula la distancia entre dos puntos, en tres dimensiones. Los puntos deben tener una distancia distinta de cero.
- D) No hay efectos laterales.
- E) La salida es la distancia real entre los dos puntos en Dist2Dim.

PROCEDIMIENTO Error(Num : ENTERO);
A) Recibe como parámetro el número (entre 1 y 6) correspondiente al error.
B) El dato debe ser un número entero.
C) Selecciona el mensaje correspondiente al error y lo escribe en pantalla, anexándole un par de campanitas.
D) Muestra en pantalla el mensaje correspondiente.
E) No hay valores de salida.

PROCEDIMIENTO LeeEntero(VAR Num : ENTERO);
A) Recibe como parámetro la variable donde se almacenará el valor leído del teclado.
B) La variable debe ser entera.
C) Lectura de un número entero con protección de entrada.
D) No hay efectos laterales.
E) La salida es la variable con el valor leído si no se dió un error.

PROCEDIMIENTO LeeReal(VAR Num : REAL);
A) Recibe como parámetro la variable donde se almacenará el valor leído del teclado.
B) La variable debe ser real.
C) Lectura de un número real con protección de entrada.
D) No hay efectos laterales.
E) La salida es la variable con el valor leído si no se dió un error.

PROCEDIMIENTO Marco;
A) No hay parámetro de entrada.
B) No hay restricción.
C) Limpia la pantalla y pinta el marco del menú principal.
D) Muestra en pantalla un marco.
E) No hay salida.

PROCEDIMIENTO MenuPrincipal;
A) No hay parámetro de entrada.
B) No hay restricción.
C) Limpia la pantalla, pinta el marco y el menú principal con sus opciones.
D) Muestra en pantalla el menú principal.
E) No hay salida.

PROCEDIMIENTO PerpPunto(Objeto: Pto3Dim; VAR Video : Pto2Dim);
A) Recibe como parámetro un punto Objeto en tres dimensiones y un punto Video en dos dimensiones.
B) Los valores deben ser del tipo Pto3Dim y Pto2Dim.
C) Calcula la proyección perspectiva del punto Objeto y lo

guarda en Video.
La pantalla es de 2*AnchoPantalla * 2*AltoPantalla.
D) No hay efectos laterales.
E) Da como salida la variable Video con los valores correspondientes a la posición en la pantalla.

PROCEDIMIENTO PintaAtomo(At : Atomos);
A) Recibe como parámetro la estructura de un Atomo.
B) La estructura debe contener la información correspondiente a la pantalla, es decir su posición y color.
C) Pinta circulos concéntricos decrementando el radio dado en Radio2Dim de la estructura, hasta llenar toda la región.
D) Muestra en pantalla el Átomo pasado como parámetro.
E) No hay salida.

PROCEDIMIENTO PintaLiga(PtoP, PtoQ : Pto3Dim);
A) Recibe como parámetro dos puntos en tres dimensiones.
B) Los valores deben de ser reales.
C) Calcula las proyecciones perspectivas de los dos puntos y después pinta una línea de acuerdo a las posiciones calculadas para la pantalla.
D) Muestra en pantalla la liga pasada como parámetro.
E) No hay salida.

PROCEDIMIENTO Ordena(VAR Mol : Molecula; l, r : ENTERO);
A) Recibe como parámetros una estructura Molécula y dos limites para ordenar.
B) La estructura Molécula debe tener valores y los limites deben ser enteros.
C) Ordena los átomos con respecto a su coordenada Z, se usa el algoritmo de Quick Sort.
D) No hay efectos laterales.
E) La salida es la estructura Molécula, teniendo sus átomos ordenados.

PROCEDIMIENTO ReduceLiga(VAR Mol : Molecula; VAR Liga : Ligadura);
A) Recibe como parámetros una estructura Molécula y una estructura Ligadura.
B) Las estructuras Molécula y Ligadura deben tener sus valores correspondientes.
C) Ajusta el tamaño del segmento definido en la ligadura a su tamaño real de acuerdo al radio de la misma. Esto se hace con ayuda de las posiciones de los átomos que estan almacenados en la estructura Molécula.
D) No hay efectos laterales.
E) Da como salida la estructura Ligadura con los valores

ajustados al tamaño de los átomos.

PROCEDIMIENTO CreaMatRot(VAR Rot : Rotacion);
A) Recibe como parámetro una estructura Rotación que contiene un ángulo y las coordenadas del vector dirección.
B) El vector debe ser distinto del vector cero.
C) Crea la matriz de transformación para poder rotar los puntos un ángulo ANG, en una dirección (X, Y, Z).
D) No hay efectos laterales.
E) Como salida se tiene la matriz R con los valores para poder transformar cualquier punto.

PROCEDIMIENTO RotaPunto(VAR x,y,z : REAL; Rot : Rotacion);
A) Recibe como parámetros un punto en tres dimensiones y la matriz de rotación con la que se transformará dicho punto.
B) La matriz debe contener valores válidos.
C) Rota un punto x,y,z con la matriz de rotación R creada por el procedimiento CreaMatRot.
D) No hay efectos laterales.
E) La salida es el punto transformado.

PROCEDIMIENTO EscalaMolecula(Escala : REAL);
A) Recibe como parámetro de entrada la Escala que afectará a la molécula.
B) El parámetro debe ser real.
C) Afecta la distancia del Ojo al Objeto con el propósito de que se vea la molécula más grande o más pequeña.
D) Cambia la distancia del ojo al objeto.
E) No hay salida.

PROCEDIMIENTO ModifTamano(VAR Mol : Molecula);
A) Recibe como parámetro una estructura Molécula.
B) La estructura debe contener valores válidos.
C) Modifica el tamaño de toda la molécula de acuerdo a un factor dado por el usuario desde la terminal.
D) No hay efectos laterales.
E) La molécula tendrá un tamaño diferente.

PROCEDIMIENTO Perspectiva(VAR Mol : Molecula);
A) Recibe como parámetro una estructura Molécula.
B) La estructura deber tener valores válidos.
C) Calcula la proyección perspectiva de toda la molécula, se auxilia del procedimiento que calcula la proyección de un solo punto.
D) No hay efectos laterales.
E) La estructura tendrá los valores válidos para

desplegarse en pantalla.

PROCEDIMIENTO PintaMolecula(VAR Mol : Molecula);
A) Recibe como parámetro una estructura Molécula.
B) La estructura debe tener valores válidos para la pantalla.
C) Pinta una molécula usando el algoritmo del pintor, modificado para resolver el problema de las líneas ocultas. El algoritmo consiste en construir una partición de la molécula e ir pintando en base a rebanadas del espacio. Para cada rebanada de espacio se crea una lista ligada en forma dinámica, la cual contiene secciones de ligaduras que se van pintando ordenadamente. La lista ligada se llama historia de ligaduras, esta historia se va creando con las nuevas ligaduras y para pintar una ligadura se intersecta con el plano corespondiente.
D) Muestra en pantalla la molécula pasada como parámetro.
E) No hay salida.

PROCEDIMIENTO EditaMolecula(VAR Mol : Molecula);
A) Recibe como parámetro una estructura molécula.
B) Debe tener valores válidos.
C) Edita una molécula, permitiendo insertar un átomo, borrar un átomo, insertar una ligadura, borrar una ligadura listar todos los átomos y listar todas las ligaduras. Cada una de las opciones es un procedimiento interno y el manejo es muy sencillo ya que es en base a un menú. El proceso de edición está protegido contra errores de entrada y salida.
D) No hay efectos laterales.
E) La salida es la molécula editada, es decir, modificada.

PROCEDIMIENTO Inicializa;
A) No hay parámetros de entrada.
B) No hay restricción.
C) Inicialización de la tablas de información para EDMOL. Principia con los valores iniciales para número de átomos y número de ligaduras. Después lee los archivos EDMOL.DIR y EDMOL..COL donde:

EDMOL.DIR Archivo que contiene los nombres de las moléculas existentes, así como el número de vistas.

EDMOL.COL Archivo que contiene las definiciones de color para los átomos.

Para terminar, inicializa las variables correspondientes al tamaño de la pantalla y la

distancia del ojo al objeto (molécula).
De ocurrir un problema con la lectura de los archivos, EDMOL por protección termina.

D) Actualiza las tablas: TabDesAtomos y TabDesMoleculas.
E) No hay salida.

PROCEDIMIENTO DirMoleculas;

A) No Hay parámetros de entrada.
B) No hay restricciones.
C) Muestra en pantalla la lista de todas las moléculas existentes en el directorio de EDMOL. Los nombres están numerados para una fácil identificación en posteriores opciones.
D) No hay efectos laterales.
E) No hay salida.

PROCEDIMIENTO LeeMolecula(NomPri : CUERDA; Vista1:INTEGER; VAR Mol:MOLECULA);

A) Los parámetros de entrada son el nombre de la molécula, en NomPri; el número de vista , en Vista1; y el lugar donde se almacenará la molécula leida, Mol.
B) El archivo debe existir en el disco.
C) Construye el nombre del archivo a partir del nombre de la molécula, esto tomando los ocho primeros caracteres, y el tipo es construido con el número de vista. Toma precaución por si el archivo no existe. Si esto ocurre entonces hay un problema de consistencia en EDMOL. Se sugiere verificar el estado de todos los archivos involucrados en el directorio.
Si no hay problemas lee la molécula del archivo, asignándole los colores y tamaños correspondientes a los átomos.
D) No hay efectos laterales.
E) La molécula es almacenada en la estructura molécula que se pasó como parámetro.

PROCEDIMIENTO MolPrincipal;

A) No hay parámetros de entrada.
B) No hay restricciones.
C) Lee el número de la molécula que será la molécula principal. Si el número es válido, entonces hace uso del procedimiento LeeMolecula para que dicha molécula pase a ser la molécula activa. Se lee la vista 1 por comodidad para el usuario.
D) La molécula activa será almacenada en ConjMolecula[1].
E) No hay salida.

PROCEDIMIENTO VistaPrincipal;

A) No hay parámetros de entrada.

B) No hay restricciones.
C) Se muestra en pantalla el número de vistas existentes para la molécula activa, y se pregunta al usuario el número de vista deseado. Se tiene la opción de ver todas las vistas para elegir mejor, esto se permite si el usuario teclea la vista cero.
Si el usuario dio la vista cero, entonces se efectúa una iteración para leer todas las vistas existentes e irlas mostrando. Si el usuario tecleó una vista válida entonces se lee del disco únicamente la vista deseada. Toda la informacion está protegida para errores de entrada salida, asi como para valores no válidos de vistas existentes.
D) La vista correspondiente a la molécula activa es almacenada en ConjAtomos[1].
E) No hay salida.

PROCEDIMIENTO RotaMolecula(VAR Mol : Molecula; Modo : INTEGER);
A) Recibe como parámetros una Molécula, y un modo que determina si se desea ver dos moléculas o sólo una.
B) La restricción es que el modo sea 1 o 2.
C) Principia colocando los caracteres correspondientes a las opciones permitidas. Define una ventana para el modo gráfico. Pregunta por el parámetro "Modo", para saber si son dos moléculas o una.
Si son dos moléculas, se pide la información correspondiente para leer la segunda molécula y se grafica en un tamaño menor al definido por el usuario dado que se ocupará la mitad de la pantalla únicamente. Posteriormente se pinta la molécula activa y espera un comando del usuario, este comando puede ser alguno de los mostrados en el menú. Si el usuario da un comando no válido , el programa no hará nada. Los comandos válidos permiten:

i) Rotación de la molécula en dirección de los ejes X, Y, Z. En ambos sentidos con un ángulo de 15 grados.

ii) Escalar el tamaño de la molécula.

iii) Mostrar la molécula sin ligaduras, es decir, sólo los átomos.

iv) Rota la molécula en general de tal forma que el usuario de la dirección y el ángulo de rotación.

v) Ver en la pantalla una pequeña descripción de los comandos para auxilio de los usuarios.

vi) Permite salvar en disco la vista que el usuario tenga.

vii) Modifica el tamaño de la molécula por un factor que el usuario da.

D) No hay efectos laterales.
E) Los valores almacenados en la estructura molécula activa se ven afectados de acuerdo a la nueva posición después de haber sido modificados por las opciones.

PROCEDIMIENTO EscMolecula(Mol:Molecula);
A) El parámetro de entrada es una estructura molécula.
B) No hay restricciones.
C) Construye el nombre del archivo donde se va a almacenar la molécula con el nombre de la molécula principal y la vista 1. Guarda en dicho archivo toda la información de la molécula pasada como parámetro.
D) Actualiza el directorio.
E) No hay salida.

PROCEDIMIENTO CreaMolecula;
A) No hay parámetros.
B) No hay restricciones.
C) Pregunta al usuario el nombre de la nueva molécula que desea construir. Llama al editor para que el usuario efectúe la inserción de átomos y ligaduras. Crea una nueva molécula en disco, con la vista 1.
D) Crea un archivo en disco.
E) No hay salida.

PROCEDIMIENTO BorraMolecula(Pregunta : BOOLEAN);
A) Recibe como parámetro un booleano para saber si pregunta antes de borrar la molécula del disco.
B) No hay restricción.
C) Verifica que exista la molécula por consistencia de EDMOL. Si es necesaria la Pregunta entonces manda un mensaje al usuario para que confirme la acción de borrar definitivamente. Para borrar construye el nombre del archivo con la variable Principal y al borrar la molécula se manda un mansaje de "molécula borrada".
Por último, borra la variable Principal, dado que ya no existe la molécula.
Borra todas las vistas de la molécula.
D) Borra archivo(s) en disco.
E) No hay salida.

PROCEDIMIENTO BorraVista;
A) No hay parámetros de entrada.
B) No hay restricciones.
C) Verifica si existe una molécula activa, para poder

borrarle la vista. Pregunta al usuario el número de vista a borrar, y si la vista es válida entonces borra la vista del disco en forma definitiva.
El nombre de la molécula debe estar en Principal.

D) Borra archivo en disco.
E) No hay salida.

PROCEDIMIENTO ModifMolecula;
A) No hay parámetro de entrada.
B) Debe haber una molécula activa para poder modificarse.
C) Verifica que haya una molécula activa.
Llama al editor, para poder modificar la molécula.
Borra la molécula vieja.
Crea la nueva molécula en disco.
Deja la nueva molécula como la molécula activa.
Hay opción de arrepentirse.
D) No hay efectos laterales.
E) No hay salida.

PROCEDIMIENTO CopiaMolecula;
A) No hay parámetros de entrada.
B) No hay restricciones.
C) Verifica que haya una molécula activa.
Pregunta el nombre de la nueva molécula.
Escribe la molécula activa en disco con el nuevo nombre.
D) Crea un archivo en disco.
E) No hay salida.

PROCEDIMIENTO Ayuda;
A) No hay parámetros de entrada.
B) No hay restricciones.
C) Muestra el contenido del archivo EDMOL.HLP , el cual contiene una pequeña descripción de los comandos del menú principal.
Espera a que el usuario presione una tecla para continuar con la descripción, ésto es cuando ya llenó la parte de la pantalla asignada.
D) Muestra en pantalla el texto de ayuda.
E) No hay salida.

{ programa principal }
A) No hay parámetros.
B) No hay restricciones.
C) El programa principal de EDMOL inicia con la lectura del directorio de moléculas. Después con los patrones de color asignados a cada átomo. Esto está en los archivos EDMOL.DIR y EDMOL.COL respectivamente.
Se muestra el menú principal en la pantalla y espera a

que el usuario teclee una opción válida.
Sólo en caso de error será mostrado un mensaje.

D) No hay efectos laterales.

E) Como salida se obtiene la modificación del directorio, el cual está en el archivo EDMOL..DIR, donde dicha modificación reflejará lo que el usuario efectuó en la sesión de EDMOL.

APENDICE D

ARCHIVOS AUXILIARES.

En este apéndice se describe el formato que tienen los archivos auxiliares de EDMOL.
Como se mencionó en la sección 4 del capítulo IV, en la inicialización de EDMOL, se leen los archivos EDMOL.DIR y EDMOL.COL.

El archivo EDMOL.DIR contiene el directorio, es decir, los nombres de las moléculas existentes. Además contiene el número de vistas que han sido creadas.
EL archivo EDMOL.DIR tiene la siguiente forma:

```
<Número de moléculas existentes.>
<Nombre 1>      <número de vistas.>
<Nombre 2>      <número de vistas.>
   .                    .
   .                    .
   .                    .
```

donde el número de moléculas es de tipo entero, los nombres de las moléculas son cadenas de 20 caracteres y el número de vistas es un entero entre 1 y 99.
Por ejemplo, el archivo contiene:

```
7
EJEMPLO              1
BENCENO              1
CICLOPROPANO         7
METANO               1
ETILENO              1
METANOL              1
sol                  1
```

El archivo EDMOL.COL contiene la definición de los colores y el radio para los Atomos.
El archivo EDMOL.COL tiene la siguiente forma:

```
<Número de simbolos existentes.
<Símbolo 1> <ColorExterior> <ColorInterior> <Radio>
<Símbolo 2          .              .            .
    .               .              .            .
    .               .              .            .
    .               .              .            .
```

Por ejemplo, el archivo contiene:

```
9
C 3 3  8
c 3 3  8
H 2 2  6
h 2 2  6
O 2 1 10
o 2 1 10
N 3 1  8
n 3 1  8
Si1 1  4
  15.0
```

Las moléculas en disco tienen la forma:

```
<Número de átomos.>
x       y       z
x       y       z
.       .       .
.       .       .
.       .       .
<Número de ligaduras.>
<AtomoFuente>   <átomoDestino>
      .               .
      .               .
      .               .
      .               .
```

Por ejemplo, el archivo para la molécula Benceno, se vera así:

```
12
  X         Y         Z         T
H  50.00    00.00    00.00     6
H  25.00    42.50    00.00     6
H -25.00    42.50    00.00     6
H -50.00    00.00    00.00     6
H -25.00   -42.50    00.00     6
H  25.00   -42.50    00.00     6
C  30.00    00.00    00.00     8
```

```
C   15.00    25.50   00.00    8
C  -15.00    25.50   00.00    8
C  -30.00    00.00   00.00    8
C  -15.00   -25.50   00.00    8
C   15.00   -25.50   00.00    8
12
```

AT1	AT2
7	8
8	9
9	10
10	11
11	12
12	7
1	7
2	8
3	9
4	10
5	11
6	12

BIBLIOGRAFIA

[1] Angell, Ian O.
A Practical Introduction to Computer Graphics.
McMillan Press, 1981.

[2] Efimov, N.
Curso Breve de Geometria Analitica.
Ed. MIR, 1969.

[3] Foley, J.
Fundamentals of Interactive Computer Graphics.
Addison Wealey.

[4] Giloi, W.
Interactive Computer Graphics: Data Structures, Algorithms, Languages.
Prentice Hall, 1978.

[5] Harrington, Steven
Computer Graphics: A Programming Approach.
McGraw-Hill, 1983

[6] Lang, Serge.
Algebra Lineal.
Addison-Wesley

[7] Lehman,
Geometria Analitica.

[8] Newman, W. M. & Sproull, R. F.
Principles of Interactive Computer Graphics.
McGraw-Hill, 2a. ed., 1979.

[9] Sedgewick, R.
Algorithms.
Addison Wesley, 1984.

[10] Salem, J.
The Organic Chemist's Book of Orbitals.
Academic Press, 1973.

[11] Seidenberg, A.
Elementos de Geometria Proyectiva.
Ed. CECSA, 1965

[12] Standhammer, J. & Khurama, A.
Display of Molecular Models With Interactive Graphics.
IEE Computer Graphics & Applications, January 1986.

[13] Strang, Gilbert.
Linear Algebra and Its Applications.
Academic Press, 1980.

[14] Sutherland, I. E. , Sproull, R. F. & Schumacker, R. A.
A Characterization of Ten Hidden-Surface Algorithms.
Computing Surveys, Vol. 6, No. 1, 1974.

[15] Trembray & Sorenson.
An Introduction to Data Structures With Applications
McGraw-Hill.

[16] Turbo Pascal V. 3.0
Borland International Inc., 1985.

[17] Wirth, N.
Algorithms + Data Structures = Programs.
Prentice Hall, 1976.