1. Principios Básicos del Modelado de Sólidos

En un mundo real, se tienen tres coordenadas para la representación de un lugar en el espacio, tenemos entonces un espacio tridimensional, donde el objeto básico es el sólido. El modelado, es la construcción artificial de un objeto, para facilitar su estudio; los modelos consisten en información almacenada en archivos que permiten visualizar los objetos y son usados para representar formas que se utilizan en una escena. Las formas simples pueden definirse a partir de expresiones analíticas, las formas más complejas requieren de métodos más complejos.

El modelado de sólidos es el procesamiento de la información geométrica y espacial de los objetos en tercera dimensión (3D o 3-D), es parte del modelado geométrico o gráfico. Los modelos de un objeto son los datos necesarios para representar al objeto completo, cuando se toma la parte geométrica como la base descriptiva del mismo se dice que son modelos geométricos. Además existen otros tipos de modelos: los modelos de figura representados por imágenes; los modelos de superficie que detallan la información en una curva de superficie, sin suficiente información geométrica; y los modelos de “alambres”, donde se agrega una tercera coordenada a las figuras de segunda dimensión (2D o 2-D).

1.1 El espacio

El entorno que nos rodea, en el que vivimos, conforma el espacio. Cada elemento indicado sobre dicho marco es un *punto*. Para poder dirigirnos de manera explícita sobre cualquiera de esos puntos, se define un sistema de coordenadas fijo en un origen y con tres ejes. Cada eje está determinado por un vector que marca su dirección y está totalmente definido mediante dos puntos, el origen y el punto que determina el vector coordenado.

Un *modelo* es la representación de algunas características de un objeto o ente.

1.2 Geometría del espacio

La estructura de trabajo se le llama *espacio*, y las piezas a las que nos referimos como puntos, vectores y superficies conforman esa estructura. La pantalla de la computadora es el plano, es decir, es un espacio afín euclídeo de dimensión dos.

Un *espacio afín*, es aquel en el que se cumple que cada par de puntos le corresponde un vector con las propiedades siguientes:

a) ∇ P,Q,R $\in$ X, $\overrightarrow{PR} = \overrightarrow{PQ} + \overrightarrow{QR}$
b) ∇ P $\in$ X, $\vec{x} \in$ E, existe un único Q $\in$ X / $\overrightarrow{PQ} = \vec{x}$

llamado **AXIOMA DE WEYL**[1]

La referencia de la coordenadas tiene tres ejes y un origen. Cada eje es determinado por un vector de dos puntos, con un mismo punto inicial, por lo tanto se requiere un punto para su definición. Con cuatro puntos no coplanarios se define un sistema de coordenadas, llamado *referencia afín***.**

1.3 Representación de 3D en 2D

La representación de cuerpos sólidos de tres dimensiones (3D) en una computadora es como dibujar en dos dimensiones puntos de tres dimensiones. Lo que requiere de establecer un homomorfismo entre $\mathbf{R^3}$ y $\mathbf{R^2}$. Para ello existen las proyecciones en perspectivas, que a continuación se describen:

1.3.1 Perspectiva Caballera

Su característica principal es ser oblicua y cilíndrica, la resultante se obtiene proyectando el objeto sobre uno de los planos coordenados con líneas de visión paralelos y no perpendiculares a dicho plano. El eje saliente suele proyectarse de manera que su proyección forme un ángulo de 135° con el eje horizontal, con un factor de escala de 0.5 o 0.6.

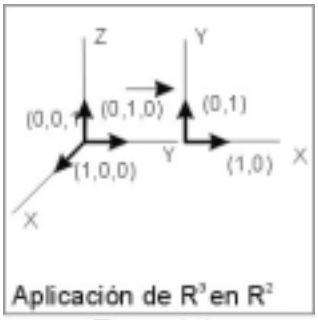


Figura 1.1

Obtener un punto bidimensional a partir de uno tridimensional, utilizando la proyección caballera, requiere la aplicación de una matriz.

Para obtener la matriz que buscamos, debe de transformarse

[1] *Escribano, Manuel. Programación de Gráficos en 3D, p.3.*

cada vector de la base $\mathbf{R^3}$ en una combinación lineal de $\mathbf{R^2}$ Por ejemplo, de la figura 1.1 tenemos:

f(0,0,1) = (0,1); el vector OZ en $\mathbf{R^3}$ se transforma en OY de $\mathbf{R^2}$**.**

El eje en perspectiva de R^3, OX se transforma en una combinación lineal de (1,0) y (0,1), teniendo en cuenta la reducción del factor escala y el ángulo con el eje horizontal, sería:

f(1,0,0) = (d_1,d_2)=(λCosφ, λSenφ). (Figura 1.2)

Proyección sobre R^2 del eje OX de R^3

Figura 1.2

(Medido en sentido contrario a la agujas del reloj, y por lo tanto es negativo).

En conclusión, al punto (x, y, z) le corresponde en la pantalla las coordenadas (x_{2d}, y_{2d}), donde:

$$\begin{bmatrix} X_{2d} \\ Y_{2d} \end{bmatrix} = \begin{bmatrix} \lambda Cos\varphi & 1 & 0 \\ \lambda Cos\varphi & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ 3 \end{bmatrix}$$

Fig. 1.3 Ecuaciones de la perspectiva de Caballera.

1.3.2 Perspectiva axonométrica

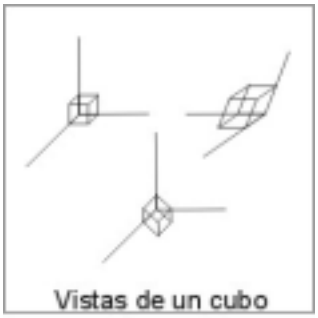

Vistas de un cubo

Figura 1.4

Se caracteriza porque las líneas de visión son ortogonales al plano de proyección, siendo éste un plano cualquiera. Utilizando proyecciones centrales y las ortogonales, resulta sencillo generar diversas vistas del objeto desde diferentes posiciones, variando sólo el ángulo que forman entre sí los ejes proyectados. En efecto, esto equivale a cambiar la posición del observador. Pero si se precisa una determinada vista, es más fácil pensar en dónde está situado el observador que imaginar cómo quedarían los ejes con el dibujo terminado. (Figura 1.4)

Entonces, se requiere saber desde donde queremos ver al objeto, una posición del observador y no las proyecciones de los ejes (Figura 1.5).

Figura 1.5

Se utilizan coordenadas esféricas, a partir de coordenadas cartesianas, mediante la transformación: Tg ϕ = y/x y Senθ = x/r donde $r = x^2 + y^2 + z^2$.

El plano de proyección es perpendicular a la recta que une el observador y al origen, entonces se proyecta cada punto del espacio a este plano. Estas son paralelas y perpendiculares al plano del dibujo, por ello es axonométrica[2]. Entonces da igual que la distancia este más o menos alejada del dibujo ya que son paralelas; de la posición del observador sólo nos interesa la dirección con la que ve el origen, llamada dirección de proyección, sin importar la distancia, los parámetros importantes son ϕ y θ.

Figura 1.6

El proceso es el siguiente, mediante las fórmulas de transformación se hallan las coordenadas polares (x_p, y_p, z_p) de las coordenadas cartesianas (x, y, z); añadimos a los ejes OX y OY de la pantalla un eje OZ, que iría en la dirección de la línea de visión; esta referencia sigue siendo ortonormal, pues OZ es perpendicular al plano de la pantalla; utilizando los procesos de rotación y traslación se pueden obtener las coordenadas (x_p, y_p, z_p), las coordenadas en 2D son (x_p, y_p).

La transformación se puede lograr con una matriz, mediante los siguientes pasos[3]:

Figura 1.7

1º. Realizar un giro respecto de OZ para colocarlos de tal manera que sean coplanares junto con OY y la dirección de visión.

[2] *Pavlidis Theo, Algorithms for Graphics and image processing, p.140.*
[3] *Escribano Manuel, Programación de Gráficos en 3D, p.34.*

Figura 1.8

2º. Se realiza un giro de 90+ϕ respecto a OX_1 para hacer coincidir Oz_1 con la dirección de visión.

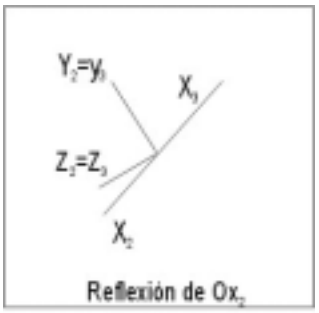


Figura 1.9

3º. Hacer la reflexión del eje OX_2 para que él y OY_2 coincidan con los ejes usuales de R_2.

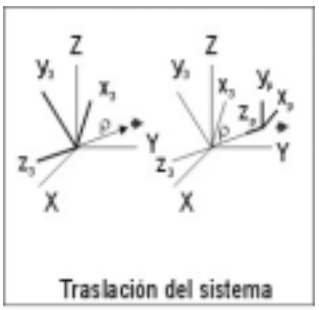


Figura 1.10

4º. Hacer la traslación del sistema hasta el observador, el vector dato T(0,0,ρ), recordemos que es irrelevante la posición exacta del plano de proyección, sólo nos interesa su orientación..

Las ecuaciones resultantes de la perspectiva cónica son:

$$\begin{bmatrix} X_p \\ Y_p \\ Z_p \end{bmatrix} = \begin{bmatrix} X_3 \\ Y_3 \\ Z_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \rho \end{bmatrix}$$

$$\begin{bmatrix} X_p \\ Y_p \\ Z_p \end{bmatrix} = [GIROS]^{-1} \begin{bmatrix} X_3 \\ Y_3 \\ Z_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \rho \end{bmatrix}$$

$$\boxed{\begin{aligned} X_{2d} &= X_p \\ Y_{2d} &= Y_p \end{aligned}}$$

Finalmente, tenemos que para un punto (x, y, z) le corresponde un punto (x_{2d}, y_{2d}) en la pantalla, después de una transformación debido a la ubicación del origen en la pantalla del monitor. De la relación anterior tenemos que:

F(0, 0, 1) = f(0,1)
F(1, 0, 0) = (Cos (90+β), Sen (90+β))
F(0, 1, 0) = (Cos (90-α), Sen(90-α))
Cos (90-α) = Sen α
Sen (90-α) = Cos α
Cos (90+β) = - Sin β
Sin (90+β) = Cos β

$$\begin{bmatrix} X_{2d} \\ Y_{2d} \end{bmatrix} = \begin{bmatrix} -Sen\beta & Sen\alpha & 0 \\ Cos\beta & Cos\alpha & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

1.3.3 Perspectiva cónica

Los dibujos se parecen más a la realidad que en las técnicas anteriores, ya que las distancias se acortan con la profundidad y las líneas de visión ya no son paralelas, sino que parten del ojo del observador[4], por lo que se hace imprescindible conocer su posición exacta. Para determinar las coordenadas (x_{2d}, y_{2d}) de un punto del espacio de (x, y, z) primero se transforma el sistema de coordenadas de (OX, OY, OZ) a (OX_p, OY_p, OZ_p); así las coordenadas (x_{2d}, y_{2d}) de la pantalla han coincidido con (x_p, y_p) más una afectación que implica una escalación en función de la profundidad y distancia D entre el observador y la pantalla.

De la relación que se establece se puede deducir de forma sencilla el valor de las coordenadas de pantalla; debido a los triángulos semejantes OPQ y O´P´Q´.

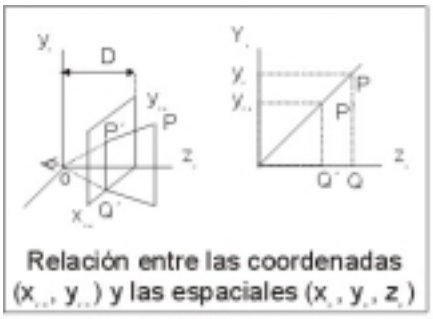


Figura 1.11

$$\frac{OQ}{OQ'} = \frac{PQ}{P'Q'} \rightarrow \frac{Z}{D} = \frac{Y}{Y_p}$$

$$Y_p = \frac{Pq}{Z}$$

Igualmente:

$$X_p = \frac{D \cdot X}{Z}$$

Las proyecciones axonométricas y cónicas son las utilizadas en el uso de gráficos para las computadoras.

[4] *Anand Vera B.,Computer Graphics and Geometric Modeling for Engineers, p.110.*

1.4 Modelado de Superficies

En muchas aplicaciones se desea modelar una superficie en un espacio tridimensional, como en las aplicaciones de CAD. Dos de los métodos más populares para representar superficies son el de Bézier y las splines.

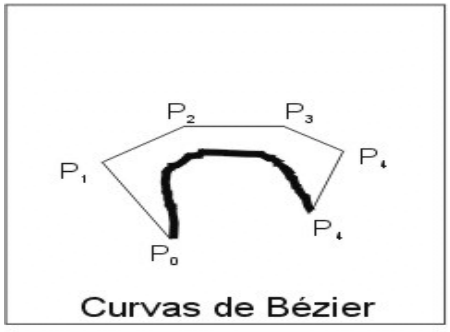
1.4.1 Superficies de Bézier

Dentro del modelado de superficies, Bézier ingeniero de la Reanult, fue el primero en hacer un método por computadora, para el diseño de carrocerías. Primero partimos de las curvas en R^2, y después plantearemos las superficies en R^3.

La curva se calcula en coordenadas paramétricas o polares, debido a que en las coordenadas cartesianas no se puede dibujar por encima y por debajo de un mismo valor de x, las funciones son aplicaciones que a cada elemento del conjunto origen tiene una sola imagen, lo que restringe el tipo de dibujos que se pueden obtener (es imposible obtener una circunferencia); por otro lado, la curva puede tener puntos con tangentes verticales, lo que conlleva a la no existencia de la derivada y crea problemas de continuidad.

El paso a tercera dimensión radica en añadir una ecuación más para z, idéntica a las de la x y la y. La curva se calcula por aproximación, a partir de una serie de puntos dados por el usuario, el método que se use es lo que lo distingue de otros métodos. El cálculo se hace a partir del peso que los puntos de control tienen sobre él, dado por la distancia al punto de control y a una distribución binomial ordinaria. El punto es la suma de todos ellos.

Dado n+1 puntos de control, P_0, P_1, …, P_n, por sus coordenadas cartesianas (x_i, y_i), el polinomio de Bézier es:



Curvas de Bézier

Figura 1.12

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

Donde:

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

En componentes:

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

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

El orden del polinomio es n+1, si aumentamos el nivel de detalle, el orden crecerá. Todos los puntos están afectados por los puntos de control, $B_{i,n}(\theta)$ siempre sea

distinto de cero, excepto para θ=0 y θ=1. Cualquier variación en algún punto alteraría la curva.

Algoritmo de representación de una curva 2D de Bézier:

```
{
Para  θ=0 hasta θ=1 con paso p
Calcular x
Calcular y
Convertir x a entero
Convertir y a entero
Punto (x, y)
                    }
```

1.4.1.1 Curvas de Bézier en el espacio

Una curva de Bézier se generaliza a tres dimensiones añadiendo un juego de ecuaciones para la z:

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

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

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

Su algoritmo sería:

```
{
  Para θ=0 hasta θ=1 con paso p
     Calcular x
     Calcular y
     Calcular z
     Proyectar (x, y, x) a (x2d, y2d)
     Convertir x2d a entero
     Convertir y2d a entero
     Punto (x2d, y2d)
                    }
```

1.4.1.2 Modelado de Superficies de Bézier

Para modelar superficies con el Método Bézier se usan dos conjuntos de curvas de Bézier, se deben introducir (m+1)·(n+1) puntos de control y la superficie se obtiene como el producto cartesiano de las curvas.

La representación en la pantalla de superficies de este tipo es la solución a la función z= f(x,y).

$$P(\theta,\phi) = \sum_{i=0}^{n} \sum_{i=0}^{n} P_i \cdot B_{i,n}(\theta)$$

Figura 1.13

Dado los algoritmos de aproximaciones, el cálculo es muchas veces tardado, obtener 100 puntos a partir de cuatro puntos de control consiste en evaluar 200 veces un polinomio cúbico y realizar 200 conversiones a enteros. Lo que implica consideraciones de eficiencia, que se deben tomar en cuenta. Los que se puede superar utilizando la regla de Horner[5]: hallar cada punto a partir de sumar al anterior una constante, reduciendo en más de 10 el número de operaciones. Otra desventaja es su manera tan global de comportamiento, es decir, si existe un ajuste interviene en todos los puntos; otra más, es que exige aumentar el polinomio aproximador, creando oscilaciones polinomiales, con un mayor error en la aproximación.

1.4.2 Superficies splines

Este es un enfoque alternativo a las aproximaciones utilizadas anteriormente, logrando crear curvas por "trozos". Spline, era una lámina o patrón que se utilizaba en la realización de contornos de figuras. Mediante la vectorización de base es posible crear intervalos para que un punto sea afectado en esa región. Además agregamos ciertas condiciones para que exista continuidad entre los intervalos, haciendo curvas suaves.

La formulación de una curva B-spline con n+1 puntos de control llamados P_0, P_1, …, P_n es:

$$P(\theta) = \sum_{k=0}^{n} P_k \cdot N_{k,t}(\theta)$$

Las splines cúbicas utilizadas son cuando t=4, entonces el grado es (t-1). Para definir $N_{k,t}(\theta)$, se hace que valga 0 para algunos valores de θ. Se introduce un vector de nudos U, que significa los extremos de θ, para $N_{k,t}(\theta)$. El espaciado de los valores n+t+1, proporciona B-splines uniformes o no uniformes.

[5] *Escribano, Manuel. Programación de Gráficos en 3D, p.30*

El parámetro θ varía entre 0 y n-t+2, así considerando todo lo anterior se usa lo siguiente:

$$N_{k,t}(\theta) = \begin{cases} 1 \text{ si } u_k \, \theta < u_{k+1} \\ 0 \text{ en cualquier otro caso} \end{cases}$$

$$N_{k,t}(\theta) = \frac{\theta - u_k}{u_{k+t-1} - u_k} \; N_{k,t-1}(\theta) + \frac{u_{k+t} - \theta}{u_{k+t} - u_{k+1}} \; N_{k+1,t-1}(\theta)$$

Cualquier término que genere un resultado 0/0 se considera como 0. Para saber como funciona el vector U, consideramos lo siguiente, que nos daría splines uniformes:

$$U_j = \begin{cases} 0 & \text{si } j<t \\ j-t+1 & \text{si } t \leqslant j \leqslant n \\ n-t+2 & \text{si } j>n \end{cases}$$

$$j = 0, \ldots, (n+t)$$

La multiplicidad de los puntos de control, acerca la curva al punto; esta flexibilidad en el control es una de las razones de la amplia difusión de las B-splines.

Existen varios tipos de curvas splines:

- De interpolación: aquellas que en lugar de aproximarse se obtienen de la interpolación.
- β-Spline: utilizan dos parámetros más $\beta 1$ y $\beta 2$, que permiten mayor control en la curva que se crea.

Para la creación de curvas B-spline en el espacio, se desdoblan sus componentes, duplicando el juego de ecuaciones a otras componente.

1.4.3 Superficies B-spline

Una superficie B-spline se obtiene del producto cartesiano de dos curvas B-spline, de tal manera que:

$$P(\theta,\phi) = \sum_{k=0}^{n} \sum_{k=0}^{n} P_{j,k} \cdot N_{j,s}(\theta) \cdot N_{k,t}(\theta)$$

1.5 Modelado de sólidos

En el mundo tridimensional (3D) el objeto representativo es el sólido. Existen diversos métodos de representación al nivel de estructura interna de datos para los programas, esos modelos son mostrados a través de diferentes técnicas; sin embargo se necesitan otros aspectos, que debemos tomar en cuenta.

1.5.1 Definición de primitivas

Las primitivas, en un sistema de modelado de sólidos, consisten en el conjunto de objetos base, generalmente incluidas dentro de una librería de primitivas sólidas, es decir, es el grupo de objetos que el usuario no tiene que molestarse en describir al programa; al contrario son ofrecidas por el mismo sistema. Estos objetos están descritos de manera especial, en forma interna, y a partir de las características de este objeto se basan las descripciones generales, el usuario precisa las dimensiones y otras características. Así de las primitivas, los usuarios obtienen objetos más complejos, a partir de descripciones sencillas.

Mediante las operaciones booleanas entre sólidos se obtienen objetos más complejos, este es de los métodos más populares; a través de la unión, la diferencia y la intersección de volúmenes. Esta forma de trabajo es fácil de usar y muy consistente ya que las operaciones descritas son cerradas, y al aplicarlas se obtiene siempre otro sólido, cuando el entorno es regularizado, ya que si la intersección es en un vértice o en una cara, éste resultaría en un punto o en un cuadrado, lo cual no es válido.

- Modelos basados en geometría euclídea o descripción topológica, donde los sólidos y superficies normalmente usados son poliédricos o soportan formulación de más o menos cierta complejidad.

- Modelado de superficies naturales, que incluyen formas variadas, no geométricas, suaves, de contornos muy diversos. Como animales, cuerpos humanos, nubes, montañas o lagos.

1.5.2 Modelos basados en geometría euclídea o descripción topológica

En estos modelos, el sólido consiste en un conjunto de líneas descritas en el sistema, típicamente son las aristas.

1.5.2.1 Modelo de esquemas o alámbrico

En estos modelos la información del sólido esta basada es solo sus aristas, lo que no es suficiente para considerar el modelo como un sólido propiamente, por ello es actualmente obsoleto, tiene además otras desventajas como la falta de un proceso de eliminación de caras ocultas, debido a que sólo tiene aristas sin caras. Así la falta de un conocimiento real del volumen provoca restricciones en el uso de relaciones booleanas, y el desconocimiento de los límites reales, evita el detectar interferencias entre sus componentes. Finalmente, es imposible aplicar los modelos de iluminación y sombreado de la imagen.

1.5.2.2 Modelo de B-rep (representación de fronteras)

En este método se definen los objetos en función de la superficie que los encierra (su frontera), describiendo sus vértices, aristas y caras. En algunos métodos especializados B-rep se restringen las caras a solamente triángulos y polígonos convexos, esta generalización hace que muchos modelos pierdan cierta resolución y aumente en los requerimientos de memoria de almacenamiento de las estructuras de datos.

Normalmente, el usuario hace la descripción del sólido, entonces el sistema tiene que verificar la validez de él, utilizando las propiedades de la fórmula de Euler[6], que describe una relación invariante entre el número de vértices, caras y aristas de un poliedro.

Aunque claro, no es suficiente la verificación de esta relación. Mediante los operadores de Euler, introducidos por Baumgart, podemos dar un paso más en la confirmación del sólido.

El sistema debe ofrecer herramientas de primitivas para formar sólidos válidos, añadiendo y eliminando vértices, aristas y caras al modelo.

La estructura de datos es compleja y se basa en grafos dirigidos con relaciones topológicas del poliedro[7]. Idealmente, la única forma exacta de que el programa conozca completamente cualquier sólido es con su descripción topológica de su frontera, y son los métodos más precisos para la representación interna de un sólido.

Pero su representación en pantalla es difícil y nos es intuitivo su forma de descripción, así como muy compleja su implementación.

Una estructura de representación realmente simple, tal vez la más simple, es una tabla de CARAS, que contiene los vértices que forma cada cara introducidas en el

[6] *Foley, Van Dam, Feiner, Hughes. Computer Graphics: Principles and Practice ,p. 130.*
[7] *Bailey C. Trevor, Gatrell C. Anthony, Interactive Spatial Data Analysis, p. 221.*

sentido de las agujas del reloj, mirando las caras desde afuera del sólido. Al introducir así los vértices se necesita un vector normal a la cara, y es necesario en cálculos posteriores de eliminación de caras ocultas y sombreado. La tabla de *vértices* concentra las cualidades puntuales de cada vértice, es decir el punto que lo forma.

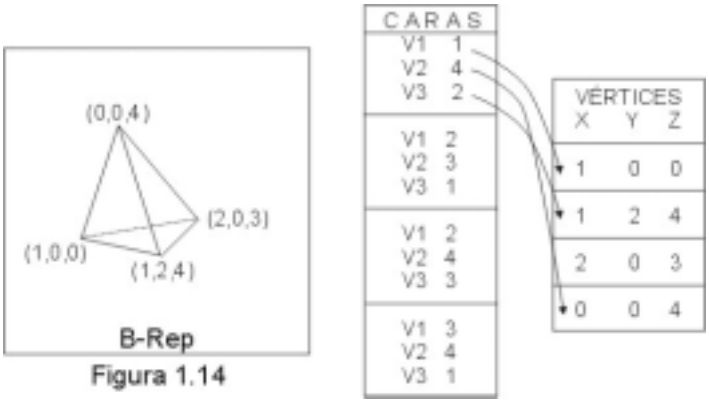


Figura 1.14

1.5.2.3 Modelado C-REP o CSG

Este es el método de modelado más utilizado y popular, con orientación a las operaciones booleanas, haciéndolo fácil y de gran consistencia en su uso, con un cálculo sencillo de las propiedades físicas de los sólidos, donde se pueden manejar superficies curvas y poliédricas, de igual manera, esto último es una gran ventaja.

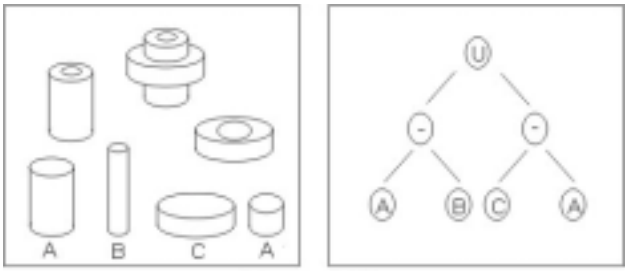


Figura 1.15

C-Rep significa *representación constructiva*, también conocido como *CSG*, *"geometría sólida constructiva"* y surge como la estructura de datos más natural que se pueda aplicar al proceso de diseño originadas de operaciones booleanas.

Este método se basa en ofrecer primitivas sólidas que el usuario combina formando modelos complejos, siendo una forma natural de hacer diseño. La estructura de almacenamiento es un árbol binario, donde los nodos son operaciones booleanas y las hojas son las primitivas o sólidos básicos.

Los árboles CSG son fáciles de presentar en la pantalla, pues sus operaciones no se resuelven entre volúmenes 3D, sino en una dimensión. Además se dispone de la información física de la masa del objeto, es fácil el cálculo de las propiedades físicas restantes.
La solución del árbol se hace cada vez que se requiere el contorno exacto del modelo, esto produce algo de lentitud en los resultados finales.

Entonces lo mejor es mezclar un modelo B-rep con C-rep para aprovechar las ventajas de ambos métodos.

1.5.2.4 Método de Barridos

La principal función utilizada, es barrer la región del espacio con una superficie plana que es la base del modelo; este proceso puede ser traslacional (perpendicular al plano de la superficie o no) también puede ser rotacional (girando en un punto); lo que permite generar modelos que son difíciles en otros métodos. Sin embargo, ciertas superficies que generan sólidos al utilizar barridos, no son permitidas en operaciones booleanas, así se deben almacenar con una descripción de otro tipo.

Figura 1.16

1.5.2.5 Descomposición Espacial

El espacio es dividido en celdillas cúbicas llamadas VOXELS y se va rastreando qué objetos están en cada celda (ocupada o no, no existe el concepto de ocupación parcial); muchas veces la resolución depende del tamaño del voxel, cuando es baja las pendientes altas se ven dentadas y se observan cuadros[8]; VOXEL significa elementos de volumen en referencia similar a pixel; aumentando la resolución se tiene un acercamiento mayor, lo que implica aumentar el número de voxels en la pantalla (y disminuir su tamaño), para superficies curvas siempre hay una pérdida de exactitud.

1.5.2.6 Método de enumeración espacial

[8] *Hearn Donald, Baker M. Pauline, Computer Graphics, p. 219.*

En este método, la escena es formada por un número prefijado de voxels iguales que forma una red regular. La representación de un objeto se hace al ir notando que celdas se ocupan y almacenarlo en una lista asociada a las celdillas, sin embargo se requiere de un excesivo gasto de memoria.

1.5.2.7 Árboles octales

Se basa en un algoritmo DAC "divide y vencerás", primeramente considera a toda la escena como un voxel; si esta totalmente llena o vacía, se acaba; si no se parte en ocho cubos (octantes) y se aplica la misma idea de manera recursiva a cada uno. El proceso termina cuando se alcance un cierto umbral, o el tamaño del voxel es muy pequeño; siendo el último ocupado.

Así se disminuye el gasto excesivo de memoria, sobre todo en escenarios donde hay pocos objetos o muy separados, ya que la red de voxels se asemeja a una matriz 3x3 dispersa. Logrando una descomposición con celdas de distintos tamaños, con el consiguiente una eficaz de memoria. Pero, para escenarios de complejidad normal, la ganancia de memoria no es muy significativa, y los pasos recursivos son muchos.

1.5.2.8 Representación por barras

En este método, cuando se tiene un conjunto de voxels consecutivos se considera como una barra, y un objeto se considera descompuesto en barras de distintas longitudes; al identificar una barra solo hay que almacenar dos voxels que la delimitan.

Las ventajas de la descomposición es la facilidad para implementar operaciones booleanas y su representación en pantalla. Ya que solo basta orientar el cubo de la escena al observador y dibujar los voxels ocupados de adelante hacia atrás. Además cuando un voxel es dibujado, todos los que están detrás de él quedan oscurecidos, y no es preciso seguir con ellos.[9]

Su mayor inconveniente es el gasto de memoria y la pérdida de resolución en figuras curvas, que se disminuye al aumentar la resolución.

Su mayor ventaja es la rapidez cuando se aplican rayos a una escena, generando imágenes realísticas con un proceso muy sencillo.

1.5.2.9 Superficies algebráicas

Consisten en modelos geométricos basados en ecuaciones matemáticas sencillas, como por ejemplo las esferas, toroides, elipsoides, cilindros y conos; éstas figuras se ajustan a ciertas ecuaciones con parámetros que nos ayudan a ajustar sus propiedades; siendo más fácil hacer sus modelos basándose en sus propias ecuaciones[10]. Estos modelos al poligonizarlos pierden cierta resolución y la memoria necesaria para almacenarlos se requiere grandes cantidades de memoria.

[9] *Hoffman Christoph M., Solid Modeling: an Introduction, p. 23.*
[10] *Hearn Donald, Baker M. Pauline, Computer Graphics, p. 116.*

1.5.3 Modelado de superficies naturales

En este tipo de modelos se incluyen formas no geométricas, suaves, de contornos muy diversos, como animales, cuerpos humanos, nubes, montañas o lagos.

1.5.3.1 Metaballs, modelado de sólidos con campos escalares

Su forma de trabajo es comparada con utilizar bolas de arcilla o plastilina por autores de cortos de animación hechos con uno de estos métodos. Sus primitivas de modelado se basan en escalares que se autodeforman en la proximidad de otra primitiva.

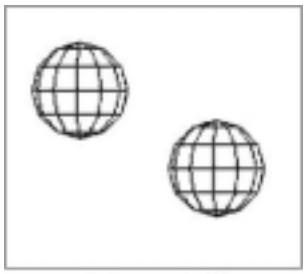


Figura 1.17

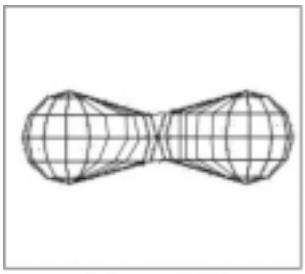


Figura 1.18

La idea es que un objeto 3D se modela como una isosuperficie de un campo escalar que es generado por una serie de primitivas puntuales, así dado que sólo tenemos una primitiva, las isosuperficies serán esferas; las cuales generarán un campo, lo suficientemente alejadas para que ambos campos no se interfieran (fig. 1.17); si se usan varias primitivas, entonces se suma el campo aportado por cada una de ellas, y las isosuperficies pueden adquirir formas complicadas (fig. 1.18). Aquí vemos la reacción, una suma de sus respectivos campos en el espacio entre ellas hace que se aparezca superficie donde un método clásico no lo haría, y se produce una deformación de una esfera sobre la otra; existe una interacción entre las primitivas y se consiguen deformaciones locales.

Este método es muy adecuado para modelar animales, rostros, fluidos y formas similares. Sin embargo, presenta ciertas desventajas como la dificultad de asignar texturas a las primitivas, sobre todo en animaciones cuyas texturas son cambiantes con la luz, no hay manera actual de asignar una textura a estas primitivas. Existen tres métodos, basados en esta representación: el modelo Blobby, los soft-objects y los metaballs; cuya diferencia es la definición del campo generado por la primitiva, llamado V_1.

El método de metaballs es el más famoso y fue presentado en IMAGINA y SIGGRAPH, desarrollado en Japón por Nishimura en 1985. Además existe una

variación llamada SUPERMETABALLS, donde el campo creado por las primitivas es orientado.