ĐỒ ÁN
ĐỒ HỌA MÁY TÍNH

Tên đồ án: Tìm hiểu về đồ họa máy tính 3D (Tạo đối tượng 3D ở mức: Wireframe, Surface, Solid; Tạo hoạt cảnh (Animation))

Giáo viên hướng dẫn: PGS.TS Lý Quốc Ngọc

Nhóm: CG2223

- Lê Bảo Chấn Phát - 19120114
- Lê Nhựt Anh - 20120427
- Nguyễn Anh Tuấn - 2012395

Mục lục

Chương 1: Giới thiệu

1.1. Động lực nghiên cứu

Đồ án này bao gồm nhiều khía cạnh quan trọng trong đồ họa máy tính ba chiều, trong đó có thể kể đến các khối cơ bản, phép biến hình, tô màu, tương tác với chuột, bàn phím và tạo animation. Mục đích của đồ án này nhằm để giúp cho những người đam mê về đồ họa máy tính có thể tiếp cận dễ dàng hơn trong việc xây dựng đồ họa ba chiều trên màn hình máy tính hai chiều.

Đồ án sẽ sử dụng môi trường nghiên cứu là thư viện OpenGL cùng với ngôn ngữ lập trình là C++.

1.2. Các phương diện liên quan

Đây sẽ là một báo cáo nghiên cứu về đồ họa máy tính, do đó phương diện toán học sẽ chiếm trọng số.

- Về toán học, các học viên trước tiên phải hiểu rõ về các khái niệm, công thức cơ bản về hình học không gian hai chiều và hình học không gian ba chiều.
- Về tin học, các học viên cần phải biết thế nào là tọa độ trên màn hình máy tính, pixel, hệ màu, tương tác phần mềm và phần cứng trên màn hình với máy tính. Đồng thời các học viên cũng cần phải biết cách lập trình phần mềm và sử dụng thư viện hỗ trợ xây dựng đồ họa máy tính.

1.3. Đóng góp

Với nội dung đồ án được trích dẫn từ nhiều nguồn tin cậy, bài báo cáo này sẽ là một bài hướng dẫn mang tính khoa học, logic và có lỗi dẫn hướng rõ ràng chưa các học viên mới, góp phần giúp các viên mới có thể tiếp thu và học hỏi được nhiều khía cạnh trong đồ họa máy tính ba chiều.

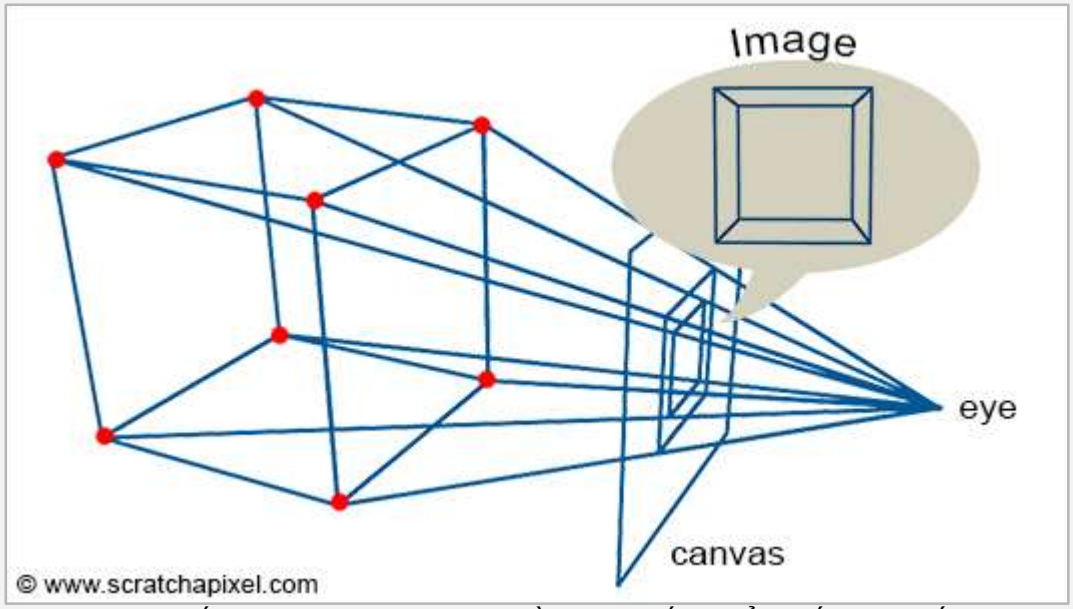
Chương 2: Các công trình nghiên cứu liên quan

2.1. Khái niệm về 3D trong đồ họa máy tính

Đồ họa máy tính 3D, hay **đồ họa 3D** là loại đồ họa sử dụng dữ liệu hình học để mô phỏng không gian ba chiều. Đồ họa 3D hiện nay có rất nhiều công dụng, trong đó có thể kể đến việc tính toán, dự đoán và thiết kế ảnh số.

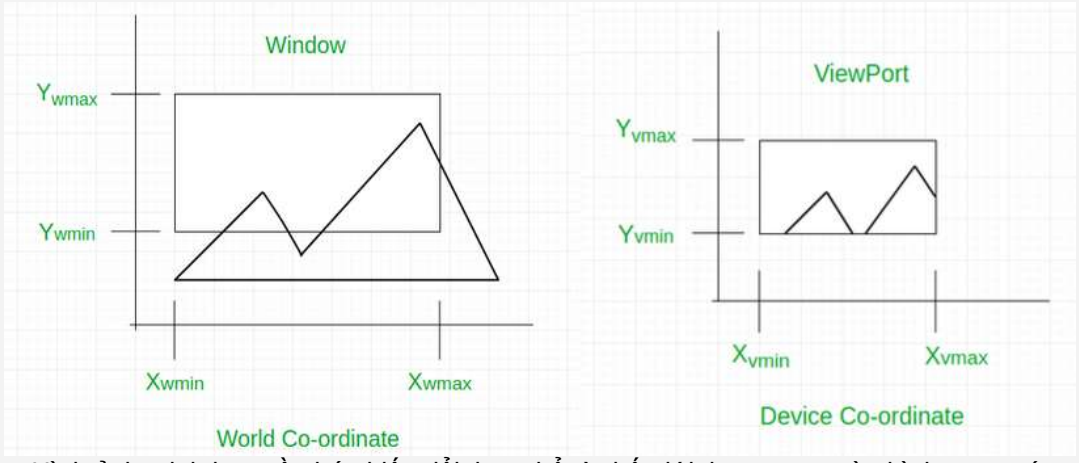
Mặc dù tên gọi là đồ họa 3D, nhưng những gì chúng được áp dụng là hoàn toàn ngược lại. Hầu hết các đồ họa 3D hiện nay đều được chiếu trên mặt phẳng hai chiều, điển hình là màn hình của máy tính. Do đó, hiển thị của đồ họa 3D không có chiều sâu, tuy nhiên thực tế các dữ liệu được lưu trong máy tính là dữ liệu được lưu ở các vector ba chiều.

Những vật thể, hình ảnh được tạo ra trong đồ họa 3D phụ thuộc quan trọng vào các hàm và giải thuật trên môi trường hai chiều. Phép chiếu là một ví dụ điển hình trong việc chuyển 3D thành 2D, cho phép mắt người chúng ta có thể tưởng tượng ra được hình ảnh ba chiều giống như ngoài đời thực dù chỉ trên màn hình hai chiều.



Hình ảnh của khối lập phương được tạo ra bằng cách nối từ điểm mắt người đến các đỉnh của khối lập phương. Hình ảnh in ra trên màn chính là hình ảnh được mô phỏng 3D mà chúng ta thấy trên màn hình máy tính. *Xem thêm ở phần Phép chiếu*.

Ngoài ra, các thực thể bên ngoài thực tế đa phần có kích thước lớn hơn rất nhiều so với màn hình máy tính, do đó các dữ liệu đo đạc được khi đã được sử dụng bởi máy tính còn phải thông qua các bước hiển thị sao cho vừa đủ với kích thước của màn hình. Vì vậy, ngoài các phép biến đổi hình ảnh sao cho mắt người có thể tưởng tượng ra 3D, thì các phần mềm đồ họa cũng cần phải có các thuật toán biến hình sao cho từ các kích thước, tọa độ ngoài đời thực chuyển về kích thước, tọa độ tương ứng trên màn hình.



Hình ảnh minh họa về phép biến đổi thực thể từ thế giới thực sang màn hình quan sát (viewport) của máy tính. *Xem thêm ở phần Phép biến hình*.

2.2. Tọa độ trong đồ họa 3D

Đồ họa 3D trong máy tính sử dụng hệ tọa độ Descartes với ba trục lần lượt là Ox, Oy, Oz tương ứng với chiều dài, chiều rộng và chiều cao. Do đó, một điểm được định nghĩa bởi ba thông số (x, y, z) và đây cũng là cách mà đồ họa máy tính sử dụng ở tầng 1 (tầng thấp nhất trong đồ họa máy tính, chuyên nghiên cứu về nền tảng và xây dựng các thuật toán hỗ trợ cho các tầng cao hơn dựa vào dữ liệu thô từ đời thực).

Về mặt lý thuyết, chúng ta có thể làm việc và tính toán dễ dàng trong môi trường ba chiều, thậm chí cả trên máy tính. Tuy nhiên, để chiếu chúng lên một màn hình hai chiều thì đây có thể sẽ là một thử thách.

Màn hình máy tính cũng sử dụng hệ tọa độ Descartes, tuy nhiên chỉ có hai trục lần lượt là Ox và Oy, tương ứng với chiều ngang và chiều dọc của máy tính. Đối với hệ tọa độ 3 chiều, điểm O(0, 0, 0) sẽ là điểm trung tâm, còn đối với hệ tọa độ 2 chiều trên màn hình máy tính thì điểm O(0, 0) lại nằm ở góc trên cùng phía bên trái. Do đó, để hiển thị tốt nhất các vật thể 3D trên màn hình 2D, chúng ta phải cần các phép biến đổi từ ba chiều thành hai chiều.

Để khai thác tối ưu nhất dữ liệu tọa độ trong không gian ba chiều, thì các lập trình viên thường sử dụng cấu trúc dữ liệu được khai báo dưới ba kiểu dữ liệu cơ bản là:

```
struct Point {
    int x,
    int y,
    int z
}
```

Và cũng từ cấu trúc dữ liệu đã được định nghĩa, họ sẽ mở rộng và xây dựng các vật thể 3D khác dựa trên Point này, ví dụ như đường thẳng Line sẽ được cấu thành hai điểm Point, tam giác Plane sẽ được cấu thành từ ba điểm Point, vân vân…

```
struct Line {
    Point P0,
    Point P1
}
```

```
struct Plane {
    Point P0,
    Point P1,
    Point P2
```

```
}
```

2.3. Phân loại mô hình ba chiều

Mô hình ba chiều được phân thành ba loại mô hình: **wireframe**, **surface**, và **solid**. Mỗi loại đều được tạo thành một cách khác nhau và chúng có những đặc điểm riêng biệt trong quá trình chỉnh sửa.

Wireframe

Mô hình khung dây wireframe là mô hình đặc tả khung xương của vật thể ba chiều. Trong loại mô hình này không tồn tại mặt phẳng hay các hình khối nào cả, và chỉ có điểm, đoạn thẳng, và các đường cong. Chúng đặc tả các cạnh của một vật thể ba chiều và cũng từ đó định nghĩa nên một vật thể ba chiều.

Chúng ta có thể tạo mô hình ba chiều bằng cách đặt tùy ý mặt phẳng 2D vào trong không gian 3D. Do mỗi đối tượng của mô hình wireframe phải được vẽ và đặt một cách độc lập, loại mô hình này được xem là tốn thời gian nhất.

Biểu diễn các đối tượng bằng các đường cong cạnh(edge curves) và đỉnh(vertices) trên bề mặt(surface) của đối tượng, bao gồm các phương trình hình học.

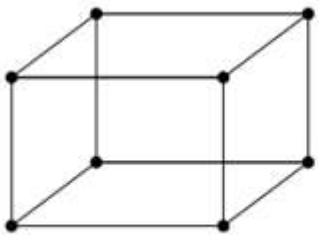
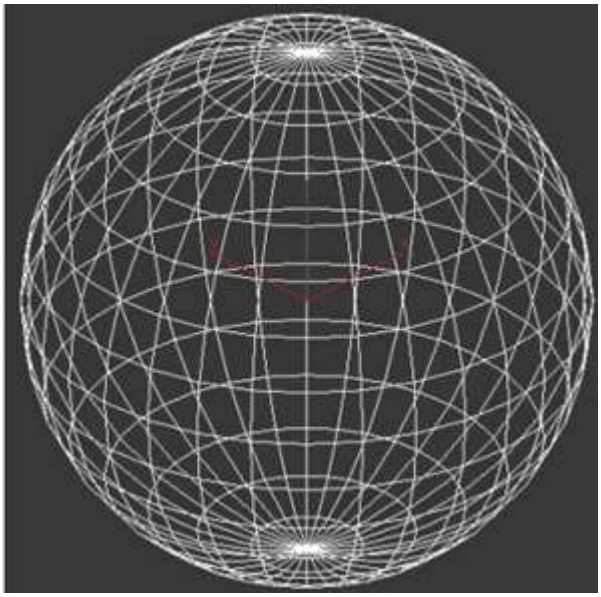
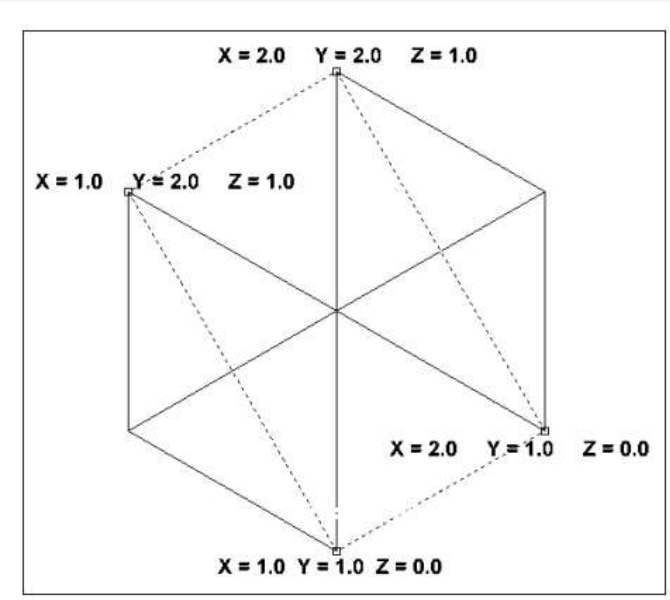


Figure 1.1: Wire frame model of a cube.



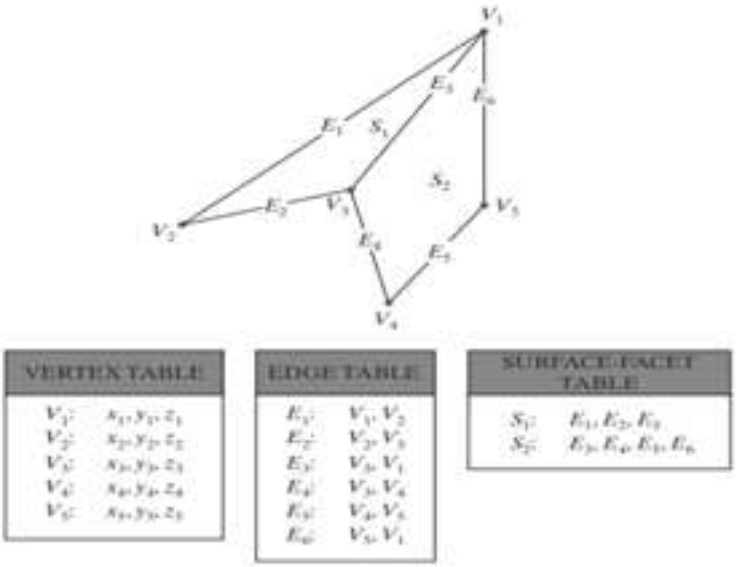


Hình ảnh mô hình khung dây wireframe của một khối lập phương.

Cấu trúc dữ liệu

Một mô hình wireframe bao gồm, bảng đỉnh (Vertex table) và bảng cạnh (Edge table), bảng mặt phẳng (Polygon-surface table).

Mỗi mục (index) của bảng đỉnh (Vertex table) ghi lại giá trị tọa độ của đỉnh đó, mỗi mục (index) của bảng cạnh ((Edge table)) có hai thành phần cho hai đỉnh của cạnh(Ei) đó. Bảng mặt phẳng(Polygon-surface table) ghi nhận lại các cạnh tạo nên mặt phảng Si.



Struct VT[] là tọa độ trong không gian 3 chiều với 3 thành phần x,y,z.

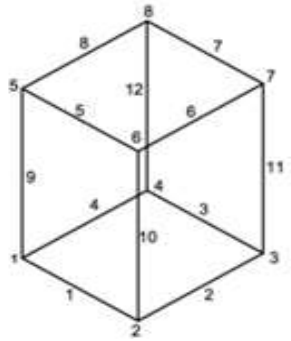
```
struct VT {
    int x, y, z;
};
```

Struct ET[] 2 đỉnh của cạnh Ei.

```
struct ET {
    struct VT* p1, p2;
};
```

Struct PT[] 3 cạnh của mặt phẳng Si.

```
struct PT {
    struct ET* p1, * p2, * p3;
};
```



Để biểu diễn một hình lập phương được xác định bởi 8 đỉnh và 12 cạnh, người ta cần có các bảng sau:

Vertex Table			
Vertex #	x	y	z
1	1	1	1
2	1	-1	1
3	-1	-1	1
4	-1	1	1
5	1	1	-1
6	1	-1	-1
7	-1	-1	-1
8	-1	1	-1

Vertex Table			
Vertex #	x	y	z
1	1	1	1
2	1	-1	1
3	-1	-1	1
4	-1	1	1
5	1	1	-1
6	1	-1	-1
7	-1	-1	-1
8	-1	1	-1

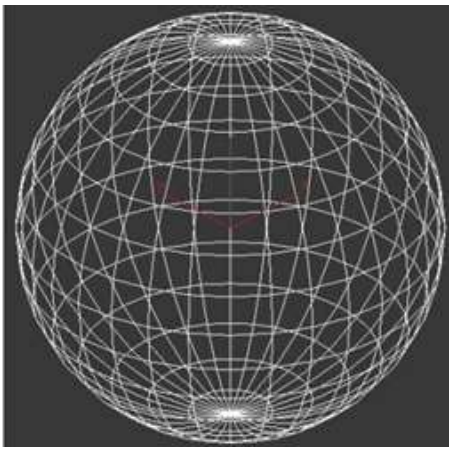
Polygon-surface Table	
Polygon-surface#	List_Edge
1	S1: 4,8,9
2	S2: 3,11,7
3	S3: 2,10,6
4	S4: 1,5,9
5	S5: 6,7,8
6	S6: 1,2,3

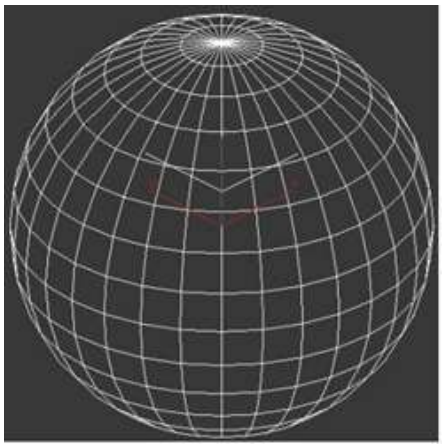
Hidden lines

Khó khăn với mô hình khung dây là các đường ẩn(hidden lines)không bị xóa, đối với các thực thể phức tạp(complex items), kết quả có thể là một mớ hỗn độn các đường không thể xác định được.

Loại bỏ đường ẩn (Hidden line removal) là một phần mở rộng của kết xuất mô hình khung dây trong đó các đường (hoặc các đoạn) được bao phủ bởi bề mặt không được vẽ.

Loại bỏ đường ẩn là phương pháp tính toán các cạnh không bị ẩn bởi các mặt của các bộ phận đối với chế độ xem được chỉ định và hiển thị các bộ phận trong hình chiếu của mô hình thành mặt phẳng 2D.

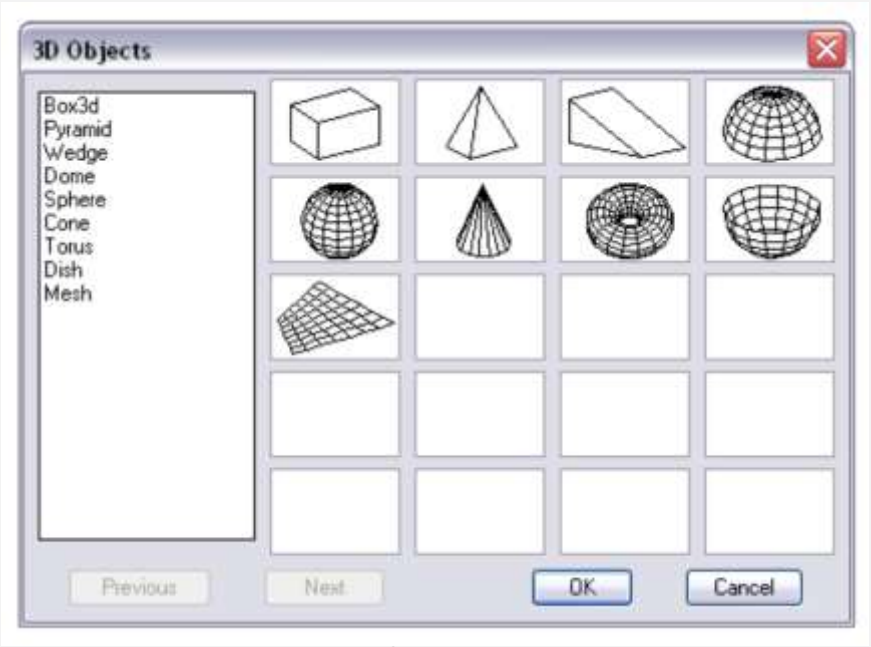




Surface

Mô hình bề mặt surface là một phiên bản tinh vi hơn so với mô hình khung dây. Không những đặc tả ra các cạnh của vật thể ba chiều, mà nó còn đặc tả ra các bề mặt của nó. Các bề mặt này có thể là những mặt phẳng (tạo từ những đường thẳng), hay là những mặt cong (tạo từ đường cong). Những mặt cong có thể được thể hiện dưới dạng nhiều đa giác để người dùng có thể dễ dàng hình dung ra hơn. Ví dụ khối cầu nếu không diễn tả dưới dạng đa giác thì chúng ta chỉ thấy một mặt cong duy nhất, và khi chiếu nó lên màn hình 2D thì trông nó sẽ giống như một hình tròn.

Các bề mặt có thể được xây dựng từ các đoạn thẳng và sử dụng các thủ thuật như tabulated surface, ruled surface, revolve surface, edge surface.

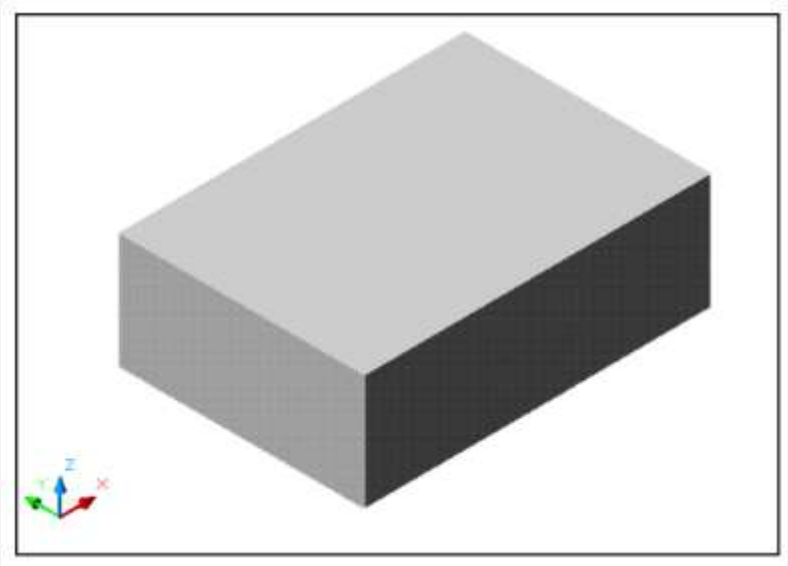


Các mô hình mặt phẳng của một số khối cơ bản.

Solid

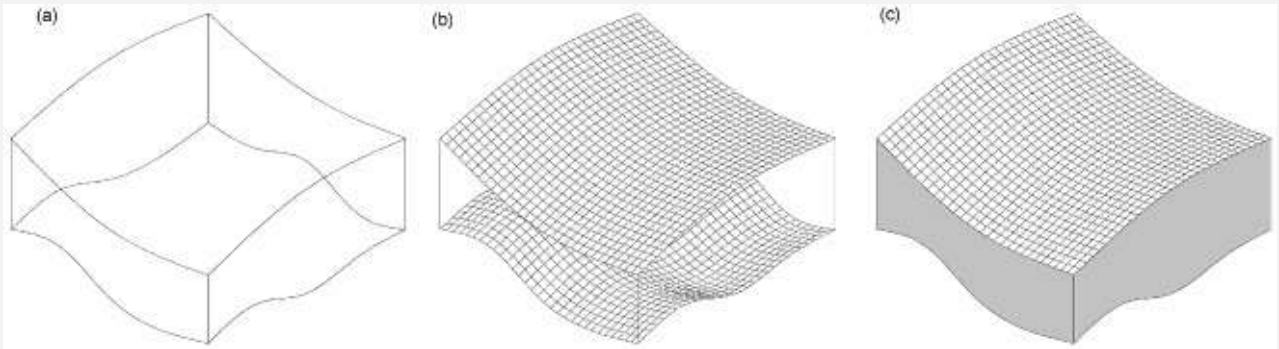
Mô hình khối solid là mô hình 3D dễ sử dụng nhất. Với loại mô hình này, chúng ta có thể xây dựng hầu hết những vật thể 3D mà mình mong muốn từ các khối 3D cơ bản: khối lập phương, hình nón, hình trụ, khối cầu, khối tam giác, và torus (giống bánh donut). Bằng cách hợp chúng lại từ các mối chúng ta sẽ tạo thành một khối hoàn chỉnh.

Mô hình khối solid còn có thể được tạo bằng cách cho xoay (revolve) một bề mặt surface quanh một trục cố định hay là ép đẩy vào (extrude) một bề mặt surface với độ sâu cho trước.

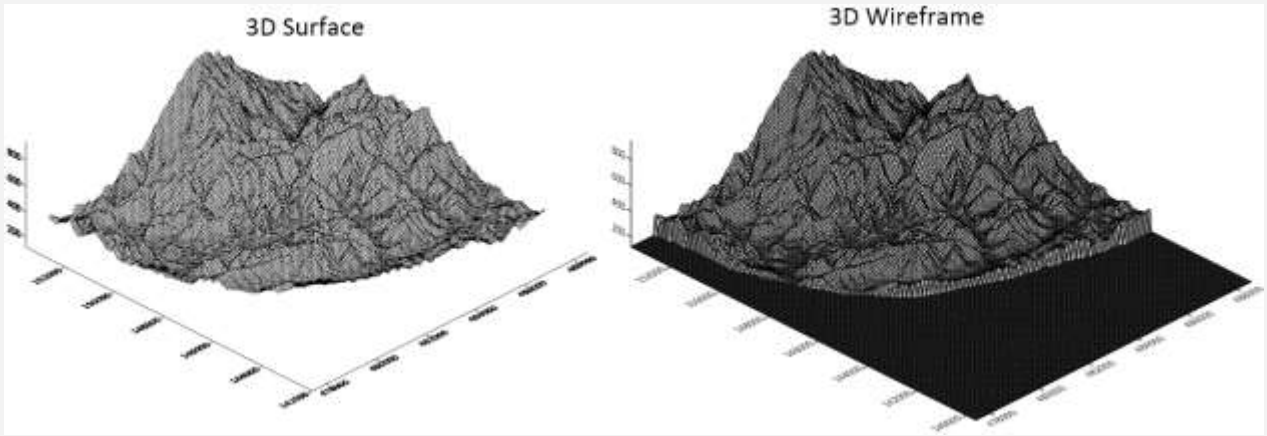


Khối hình hộp chữ nhật được trình bày dưới dạng mô hình khối.

Dưới đây là minh họa về phân loại ba loại mô hình ba chiều:



Hình ảnh minh họa về khối vật thể 3D được xây dựng từ các đường thẳng và đường cong dưới dạng trình bày của ba loại mô hình: Wireframe, Surface, Solid (từ trái sang phải)



Mô phỏng địa hình dưới dạng mô hình surface và wireframe.

2.4. Các hình khối cơ bản

Hình trụ

Phương trình:

$$p(s,t) = x(s,t), y(s,t), z(s,t)) = (R.cos(s), R.sin(s), t)$$

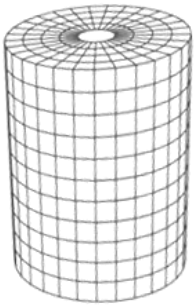
R: bán kính đáy

s: góc quay trong đoạn [0,2pi]

t: tham số chiều cao trong đoạn [0,H]

H: là chiều cao của hình trụ

Hình minh họa:



Hình cầu

Phương trình:

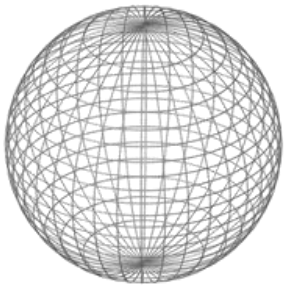
$$p(s,t) = (x(s,t), y(s,t), z(s,t)) = (R.cos(t).cos(s), R.sin(t).cos(s), R.sin(s))$$

R: bán kính hình cầu

s: góc quay trong đoạn [0, π]

t: góc quay trong đoạn [0, 2π]

Hình minh họa:



Hình nón

Phương trình:

$$p(s,t) = (x(s,t), y(s,t), z(s,t)) = (R.(1 - t/H).cos(s), R.(1 - t/H).sin(s), t)$$

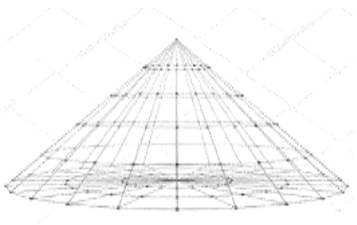
R: bán kính đáy

s: góc quay trong đoạn [0,2π]

t: tham số chiều cao trong đoạn [0,H]

H: là chiều cao của hình nón

Hình minh họa:



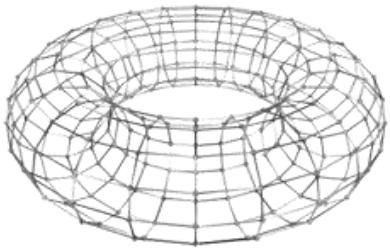
Hình xuyến

Phương trình:

$$p(s,t) = (x(s,t), y(s,t), z(s,t)) = (cos(s).(R + r.sin(t)), sin(s).(R + r.sin(t)), r.sin(s))$$

R: bán kính đường tròn lớn
r: bán kính đường tròn nhỏ
s: góc quay trong đoạn [0,2π]
t: góc quay trong đoạn [0,2π]

Hình minh họa



2.5. Phép biến hình

Phép biến hình, hay còn có tên gọi tổng quát hơn là phép biến đổi hình học, trong không gian ba chiều tức là các thuật toán thay đổi cấu trúc, vị trí của thực thể ba chiều.

Phép tịnh tiến

Giả sử ta có vector T(T_x, T_y, T_z) là vector tịnh tiến thì ta được công thức phép tịnh tiến như sau:

$$\begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x + T_x \\ y + T_y \\ z + T_z \\ 1 \end{pmatrix}$$

Ta có hàm OpenGL sau hỗ trợ phép tịnh tiến: `glTranslatef( tx, ty, tz );`

Phép xoay

Xoay quanh Ox:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ \cos\theta \cdot y - \sin\theta \cdot z \\ \sin\theta \cdot y + \cos\theta \cdot z \\ 1 \end{pmatrix}$$

Xoay quanh Oy:

$$\begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta \cdot x + \sin\theta \cdot z \\ y \\ -\sin\theta \cdot x + \cos\theta \cdot z \\ 1 \end{pmatrix}$$

Xoay quanh Oz:

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta \cdot x - \sin\theta \cdot y \\ \sin\theta \cdot x + \cos\theta \cdot y \\ z \\ 1 \end{pmatrix}$$

Sử dụng công thức đã biến đổi trên 3 loại trục tọa độ, ta có thể suy ra được công thức cho phép quay tại một trục bất kỳ trong không gian ba chiều. Ma trận được sử dụng với bộ biến xoay (R_x, R_y, R_z) với R_x, R_y, R_z lần lượt là tọa độ cách trục Ox, Oy, Oz của điểm C và OC là trục xoay.

$$\begin{bmatrix} \cos\theta + {R_x}^2(1-\cos\theta) & R_xR_y(1-\cos\theta) - R_z\sin\theta & R_xR_z(1-\cos\theta) + R_y\sin\theta & 0 \\ R_yR_x(1-\cos\theta) + R_z\sin\theta & \cos\theta + {R_y}^2(1-\cos\theta) & R_yR_z(1-\cos\theta) - R_x\sin\theta & 0 \\ R_zR_x(1-\cos\theta) - R_y\sin\theta & R_zR_y(1-\cos\theta) + R_x\sin\theta & \cos\theta + {R_z}^2(1-\cos\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Ta có hàm OpenGL hỗ trợ cho phép xoay: `glRotatef (θ, x, y, z)`

- Ví dụ `glRotatef(30.0, 0.0, 1.0, 0.0)` thì sẽ quay quanh y một góc 30°. Lưu ý các tham số sử dụng trong phép quay phải viết dưới dạng 360°.

Phép co giãn

Giả sử ta có bộ biến số co giãn (S1, S2, S3) thì ta có công thức phép co giãn như sau:

$$\begin{bmatrix} S_1 & 0 & 0 & 0 \\ 0 & S_2 & 0 & 0 \\ 0 & 0 & S_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} S_1 \cdot x \\ S_2 \cdot y \\ S_3 \cdot z \\ 1 \end{pmatrix}$$

Ta có hàm OpenGL hỗ trợ cho phép co giãn: `glScalef(0.5, 0.5, 0.5);`

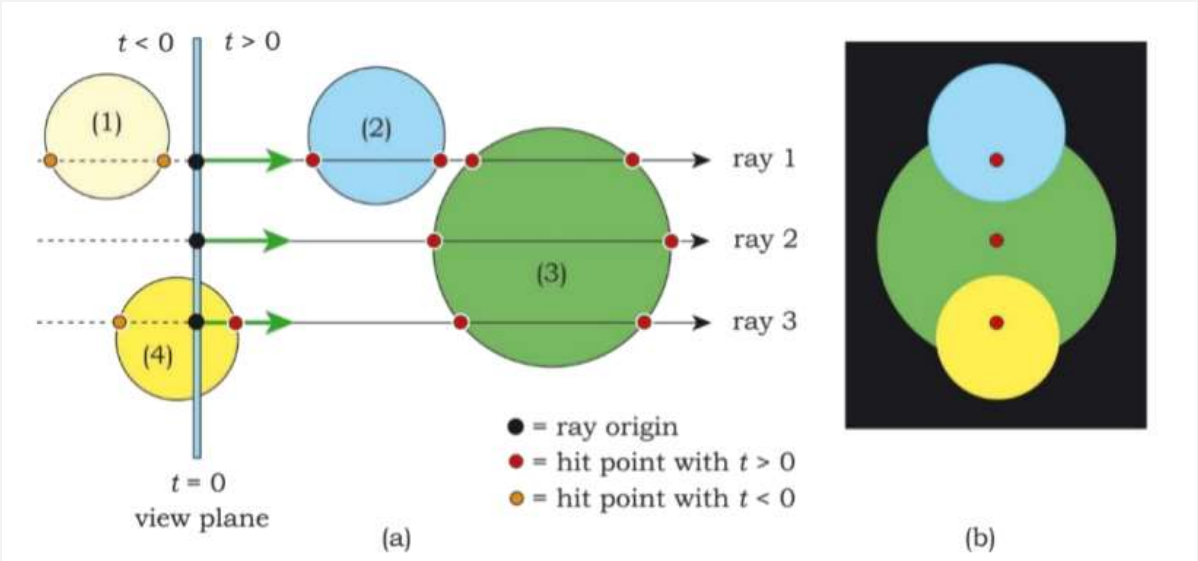
2.6. Biến đổi 3D thành 2D

Ray Casting và Ray Tracing

Ray Casting là một thuật ngữ được sử dụng rộng rãi trong đồ họa máy tính ba chiều để chỉ việc bắn một tia ra từ một vị trí trong không gian 3D và di chuyển theo một hướng cụ thể. Chùm điểm mà tập tia bắn ra tiếp xúc được sẽ là ảnh xuất hiện lên màn hình chiếu. Ray Casting sẽ không tạo độ sáng cho vật thể, vì các tia bắn ra đều gặp tại vật thể đó và số lượng tia nhận lên màn hình là không bị biến đổi.

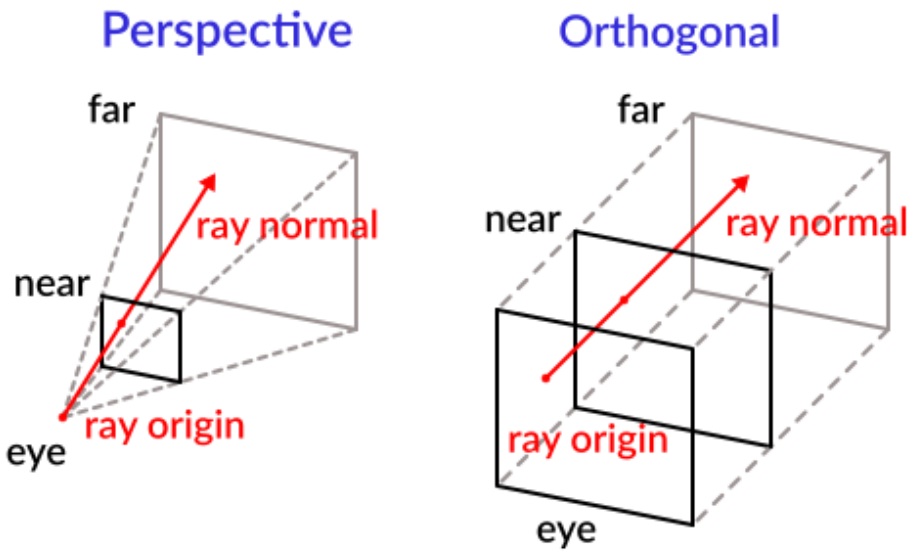
Ray Tracing cũng được định nghĩa tương tự, tuy nhiên các tia sau khi tiếp xúc với vật thể có thể bị đổi hướng (do vật liệu, góc phản chiếu). Do đó màn hình chiếu sẽ không nhận lại toàn bộ các tia ban đầu mà là các tia đã bị biến đổi.

Hình ảnh nhận về mắt người khác nhau khi ray tracing (bên trái) so với ray casting (bên phải)

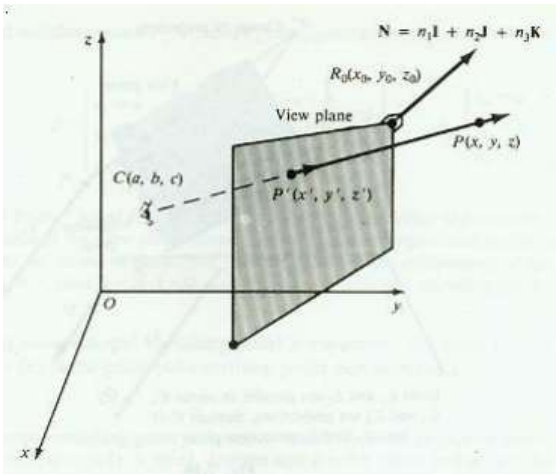


Phép chia ba quả cầu lên màn hình, quả cầu nào có hit point nhỏ hơn thì xuất hiện đầu tiên.

Phép chiếu



Phép chiếu phối cảnh:



Tâm chiếu $C(c_x, c_y, c_z)$

Mặt phẳng chiếu $(R_0, \vec{N}), R_0(x_0, y_0, z_0), \vec{N}(n_1, n_2, n_3)$

$$P' = Per(C, (R_0, \vec{N})).P$$

$$\vec{P'C} = \alpha.\vec{PC},$$

$$\begin{cases} x' = \alpha.(x - c_x) + c_x \\ y' = \alpha.(y - c_y) + c_y \\ z' = \alpha.(z - c_z) + c_z \end{cases}$$

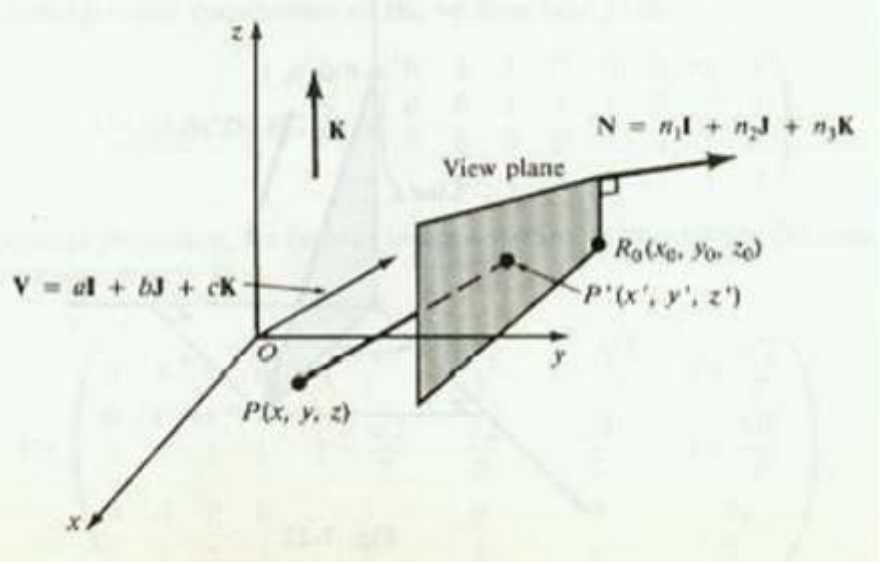
Xác định ma trận biến đổi trong hệ tọa độ thuầ

$$Per(C,(R_0,\vec{N})) = T^{-1}(\vec{CO}).Per(O,(R_0,\vec{N})).T(\vec{CO}$$

$$P' = Per(C,(R_0,\vec{N})).P$$

$$\begin{bmatrix} x_H \\ y_H \\ z_H \\ H \end{bmatrix} = \begin{bmatrix} d+c_x.n_1 & c_x.n_2 & c_x.n_3 & -c_x.d_0 \\ c_y.n_1 & d+c_y.n_2 & c_y.n_3 & -c_y.d_0 \\ c_z.n_1 & c_z.n_2 & d+c_z.n_3 & -c_z.d_0 \\ n_1 & n_2 & n_3 & -d_1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Phép chiếu song song



Hướng chiếu $\vec{V}(a,b,c)$

Mặt phẳng chiếu $(R_0,\vec{N})$

$$P' = Par(\vec{V},(R_0,\vec{N})).P$$

$$Par(\vec{V},(R_0,\vec{N})) = T(\vec{OR_0}).A^{-1}(\vec{N}).Par(\vec{V},(O,Oz).A(\vec{N}).T(\vec{R_0O})$$

Xác định ma trận chiếu

$$T(\vec{R_0}O)=\begin{bmatrix}1&0&0&-x_0\\0&1&0&-y_0\\0&0&1&-z_0\\0&0&0&1\end{bmatrix}\quad Par(\vec{V},(O,Oz))=\begin{bmatrix}1&0&-\frac{a}{c}&0\\0&1&-\frac{b}{c}&0\\0&0&0&0\\0&0&0&1\end{bmatrix}$$

$$A(\vec{N})=\begin{bmatrix}\frac{\lambda}{|\vec{N}|}&\frac{-n_1n_2}{\lambda|\vec{N}|}&\frac{-n_1n_3}{\lambda|\vec{N}|}&0\\0&\frac{n_3}{\lambda}&-\frac{n_2}{\lambda}&0\\\frac{n_1}{|\vec{N}|}&\frac{n_2}{|\vec{N}|}&\frac{n_3}{|\vec{N}|}&0\\0&0&0&1\end{bmatrix},\lambda=\sqrt{n_2^2+n_3^2}$$

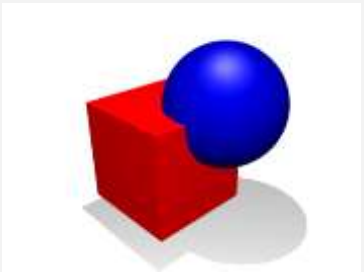
2.7. Ứng dụng trong việc tạo các vật thể 3D khác

Hiện nay, hầu hết các vật thể ba chiều đều được vẽ dựa trên các vật thể, hình khối cơ bản thông qua các phép biến đổi hình học. Các dạng hiển thị wireframe, surface, solid vẫn được sử dụng từ các câu lệnh đã được nhắc trong xây dựng các hình khối cơ bản, từ đó có thể cho phép người dùng dễ dàng mường tượng hơn.

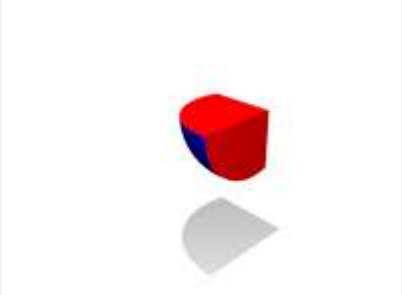
Kỹ thuật kết hợp

Kỹ thuật kết hợp hay tên gọi đầy đủ là **kỹ thuật xây dựng hình học khối rắn** (constructive solid geometry) là một kỹ thuật tạo nên một vật thể mới dựa trên hai hay nhiều hơn vật thể khác nhau trong không gian ba chiều. Các vật thể này phải ít nhất giao nhau, hoặc cắt nhau và phải là các vật thể dưới dạng mô hình rắn solid.

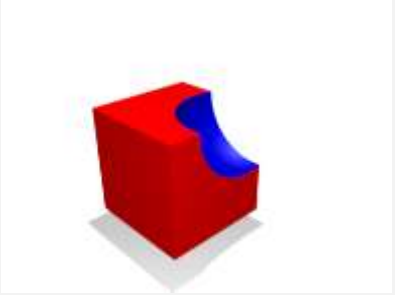
Kỹ thuật này được xây dựng bằng phép toán boolean, do đó chúng ta có thể biểu diễn dưới dạng cây nhị phân. Có ba kỹ thuật kết hợp chính:



Phép hợp: Kết hợp hai vật thể với nhau thành một khối hoặc của vật thể này hoặc của vật thể kia

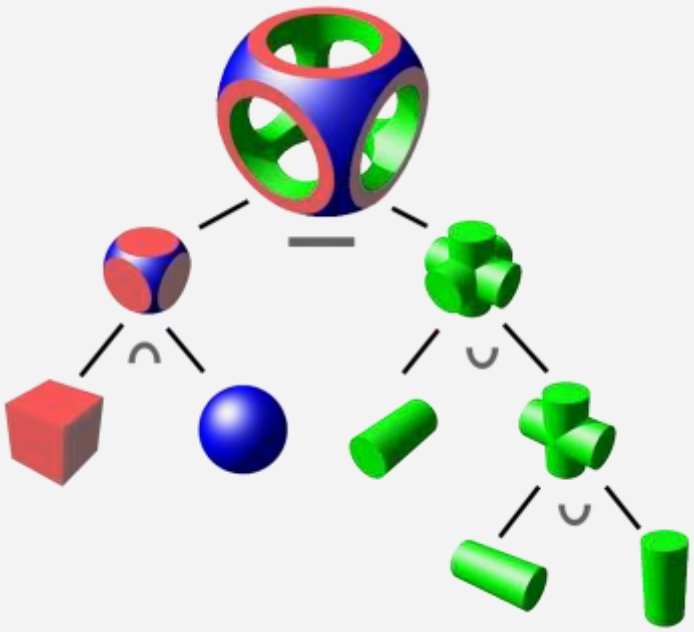


Phép giao: Kết hợp hai vật thể với nhau thành một khối của cả hai vật thể trước đó



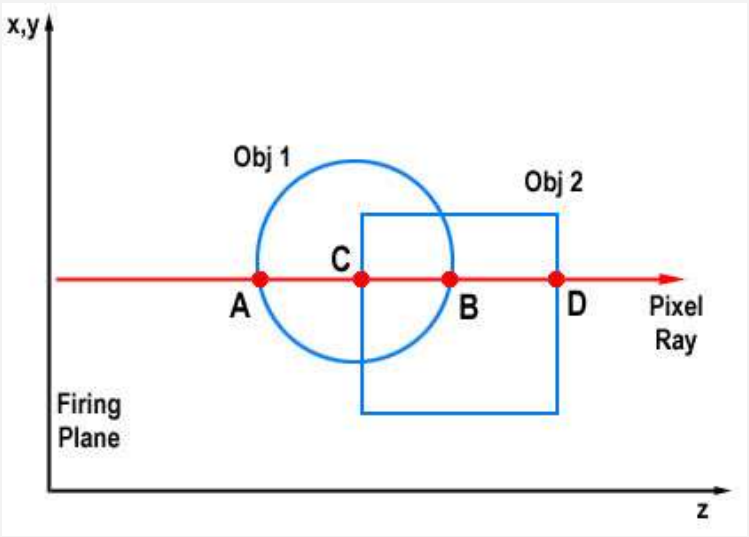
Phép trừ: Kết hợp hai vật thể với nhau thành một khối nhưng chỉ lấy phần của vật thể trừ

Đây là ví dụ của ba loại kỹ thuật kết hợp trên:



Một vật thể có thể được biểu diễn dưới dạng cây nhị phân các phương pháp kết hợp hình học. Ở mô hình này, các nút được đánh dấu ∩ là phép giao, ∪ là phép hội, và — là phép trừ.

Để có thể xây dựng phép kết hợp thì người ta dựa vào thuật toán Ray casting đã nêu ở phần trước. Ray casting sẽ bắn ra chùm tia tiếp xúc với vật thể (thông qua các cạnh hoặc điểm), đồng thời lưu lại các hit points và tính giá trị hit points theo giá trị lần tiếp xúc với vật thể.



Giả sử màn hình chiếu là mặt phẳng chứa trục Oz, thì ta được 4 hit points là tại 4 điểm A, C, B, D lần lượt được gán giá trị tăng dần.

Dựa theo mô hình, thì ta có thể suy ra được các loại phép kết hợp được xây dựng như sau:

Phép kết hợp (Obj1 với Obj2)	**Giới hạn bề mặt**
Phép hợp	A, D
Phép giao	C, B
Phép trừ	A, C

Từ đó ta có công thức chung:

Phép kết hợp (Obj1 với Obj2)	**Giới hạn bề mặt**
Phép hợp	min(Obj1 OR Obj2) → max(Obj1 OR Obj2)
Phép giao	min(Obj1 AND Obj2) → max(Obj1 AND Obj2)
Phép trừ	min(Obj2 AND !Obj1) → max(Obj2 AND !Obj1)

Mô hình hóa bề mặt

Mô hình hóa bề mặt (surface modeling) là một kỹ thuật mô hình hóa vượt trội hơn cả wireframe vì nó không chỉ giúp định nghĩa các cạnh của vật thể ba chiều mà còn trên những bề mặt của nó. Mô hình hóa bề mặt cho phép các nhà thiết kế, kỹ sư không gian 3D dễ kiểm soát hơn, linh động hơn trong việc xây dựng các vật thể 3D thực tế.

Mô hình hóa bề mặt được áp dụng cho hai loại thực thể bề mặt sau:

- Bề mặt giải tích (analytic surface): mặt phẳng (plane surface), bề mặt hợp viền (ruled surface), bề mặt xoay (surface of revolution), bề mặt hợp lưới (tabulated surface)
- Bề mặt nhân tạo (synthetic surface): bề mặt tham số bậc 3, bề mặt bezier, bề mặt B-Spline…

Hiện nay, người ta áp dụng mô hình hóa bề mặt trong việc tính toán khối lượng vật thể, kiểm tra va chạm giữa các chi tiết máy, tạo lập vật thể bị cắt…

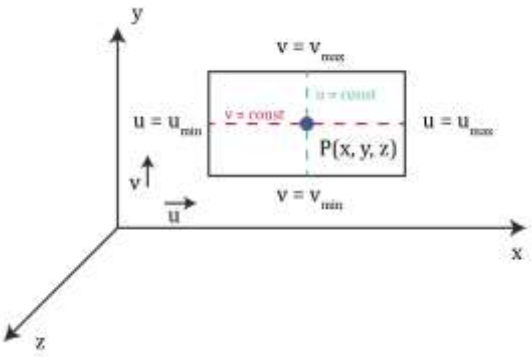
Bề mặt giải tích

Bề mặt giải tích có thể được biểu diễn thông qua phương trình tường minh và phương trình ẩn số.

- Phương trình ẩn số: $F(x, y, z) = 0$
- Phương trình tường minh: $V = [x, y, z]^T = [x, y, f(x, y)]^T$, với V là vectơ vị trí của điểm trên bề mặt. Trong phương trình này, hai biến x, y được định nghĩa trực tiếp, còn z phải được định nghĩa thông qua phương trình $f(x, y)$.

- Phương trình tham số: $V(s,t) = [x,y,z]^T = [X(s,t), Y(s,t), Z(s,t)]^T$, với X, Y, Z lần lượt là phương trình được định nghĩa theo hai biến s, t.

Tiếp theo ta cần định nghĩa:



- $P(u,v) = [x \quad y \quad z]^T$
- $P(u,v) = [x(u,v) \quad y(u,v) \quad z(u,v)]$
- $u_{min} \le u \le u_{max}, v_{min} \le v \le v_{max}$

Biểu diễn bề mặt giải tích bằng phương trình tham số

Mặt phẳng:

Phương trình tham số của mặt phẳng được định nghĩa bởi ba điểm P0, P1 và P2:

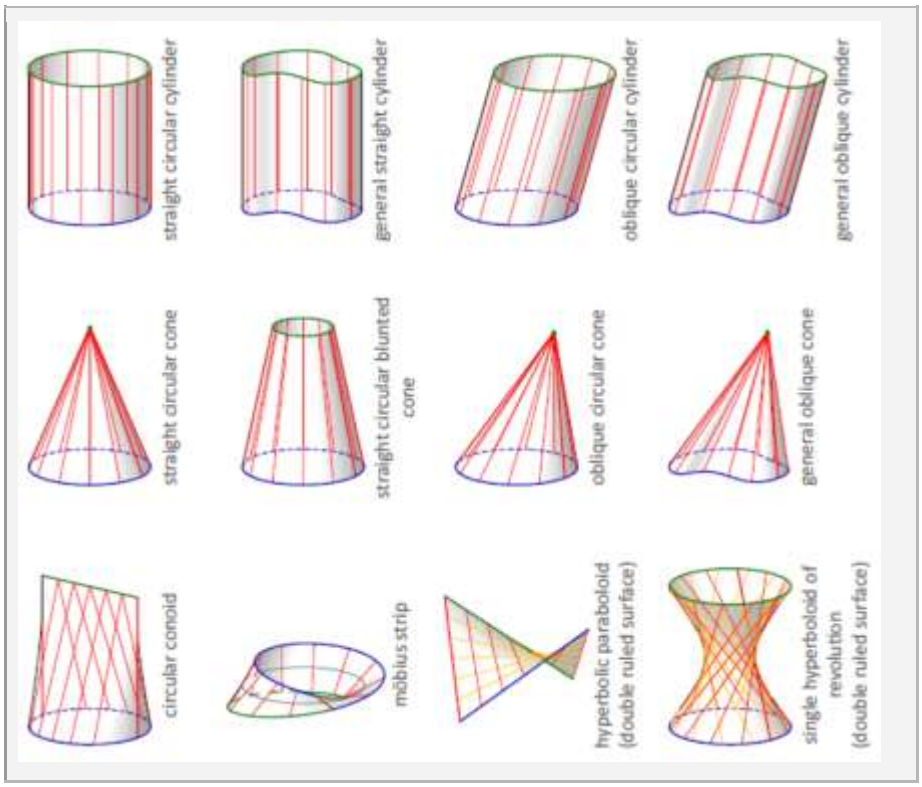
$$P(u,v) = P_0 + u(P_1 - P_0) + v(P_2 - P_0)$$
$$0 \le u \le 1, 0 \le v \le 1$$

Bề mặt hợp viền:

Bề mặt hợp viền được tạo lập bằng cách kết nối giữa hai đường cong G(u) và Q(u) thông qua tập các đoạn thẳng. Phương trình tham số của bề mặt hợp viền là:

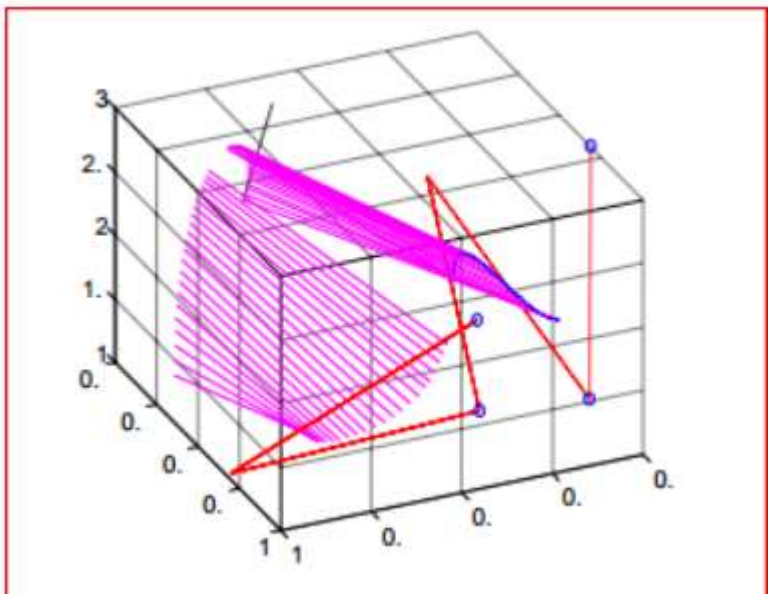
$$P(u,v) = (1-v)G(u) + vQ(u)$$
$$0 \le u \le 1, 0 \le v \le 1$$

Cho u chạy vòng lặp từ 0 đến 1 thì sẽ tạo ra tập các viền đoạn thẳng theo hướng v của bề mặt, ngược lại nếu cho v chạy thì sẽ tạo ra tập các viền đoạn thẳng theo hướng u của bề mặt.



Một số ví dụ từ kỹ thuật bề mặt hợp viền

Bề mặt được xây dựng bằng cách cho trượt một đoạn thẳng trên đường cong. Các mặt hợp viền nhận được bằng phép nội suy tuyến tính từ 2 đường cong biên cho trước tương ứng với hai biên đối diện của mặt kẻ P1(u) và P2(u).

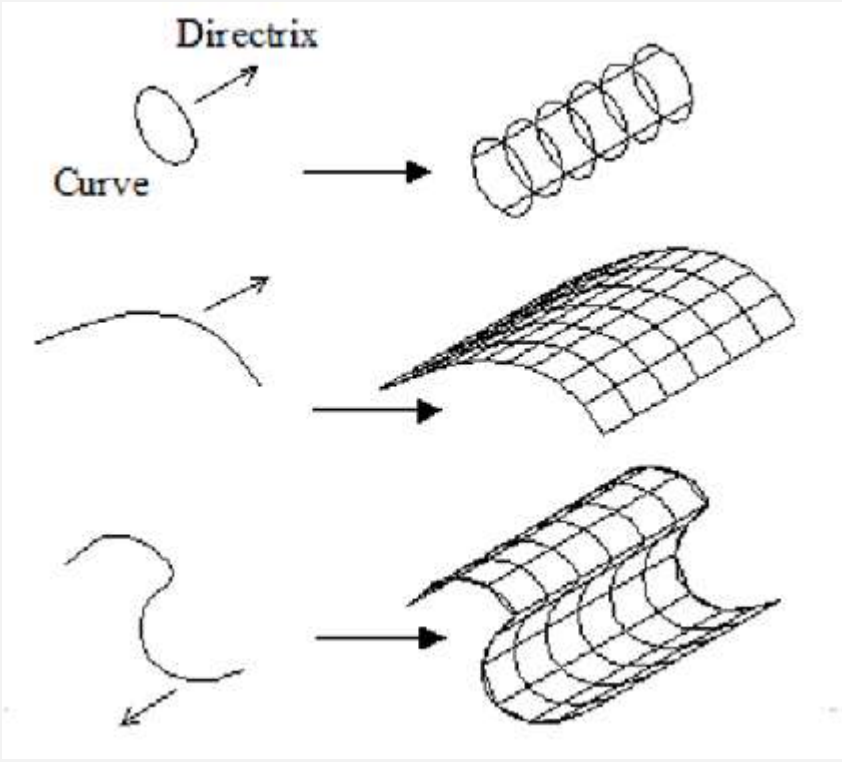


Bề mặt hợp lưới:

Bề mặt hợp lưới (hay còn được gọi là trụ lưới **tabulated cylinder**) được tạo lập bằng cách cho kéo giãn một đường cong G(u) theo hướng đi của một vector. Phương trình tham số của bề mặt hợp lưới là:

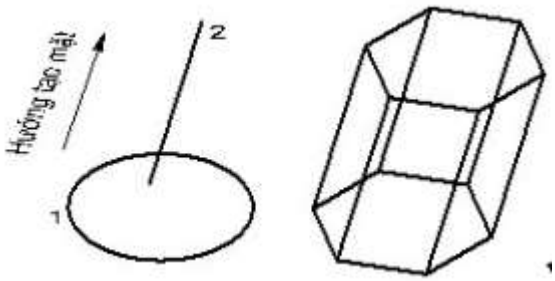
$$P(u,v) = G(u) + v\vec{n}$$

$0 \le u \le u_{max}, 0 \le v \le v_{max}$



Một số ví dụ từ kỹ thuật bề mặt hợp lưới

Tabulated cylinder dùng để tạo mặt lưới trụ theo hình dạng của đường chuẩn (path curve) quét dọc theo vectơ định hướng (direction vector).



Bề mặt xoay:

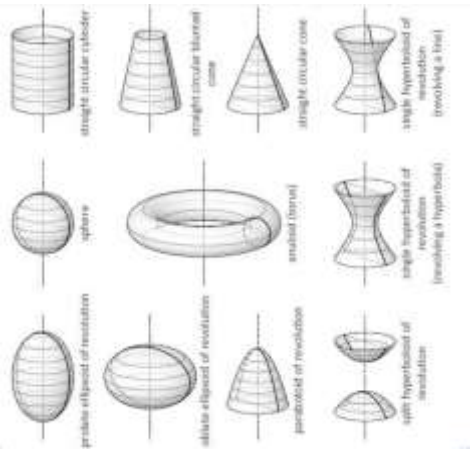
Bề mặt xoay được tạo lập bằng cách cho một đường cong xoay quanh một trục cố định. Kết quả của kỹ thuật xoay thường là một khối vật thể rắn ba chiều mang tính thẩm mỹ cao.

Phương trình tham số của bề mặt xoay là:

$$P(u,v) = G(u) + v\big(Q(u) - G(u)\big)$$

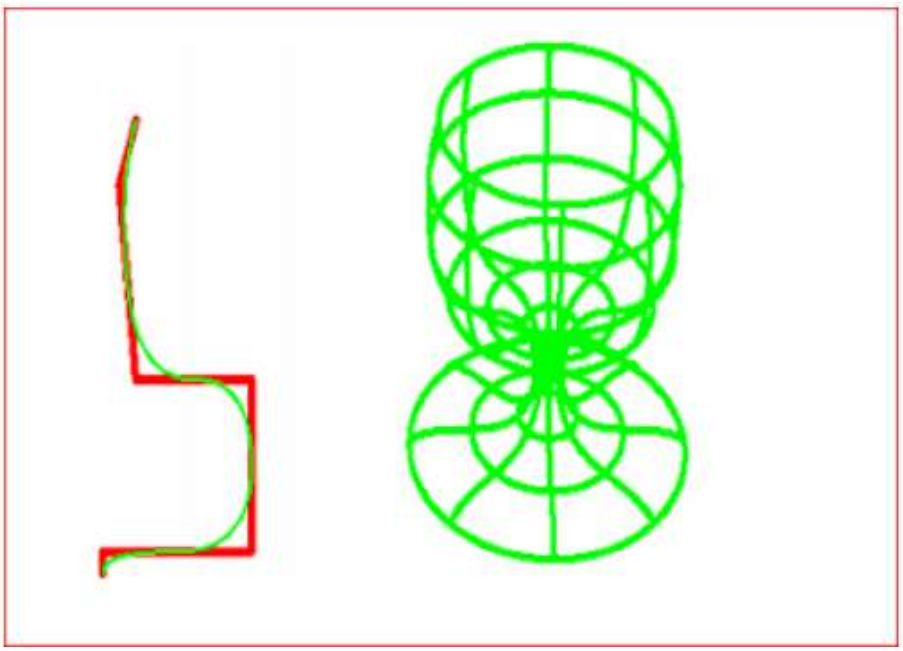
$0 \le u \le u_{max}, 0 \le v \le v_{max}$

với G(u) là phương trình đường cong và Q(u) là phương trình của trục xoay.



Mặt được xây dựng bởi đường thẳng hay một đường cong phẳng, quanh 1 trục trong không gian.
Quay quanh trục x một thực thể nằm trong mặt phẳng Oxy, phương trình bề mặt có dạng:

$$Q(t, f) = [\, x(t)\ y(t)\ \cos fz(t)\ \sin f \,]$$



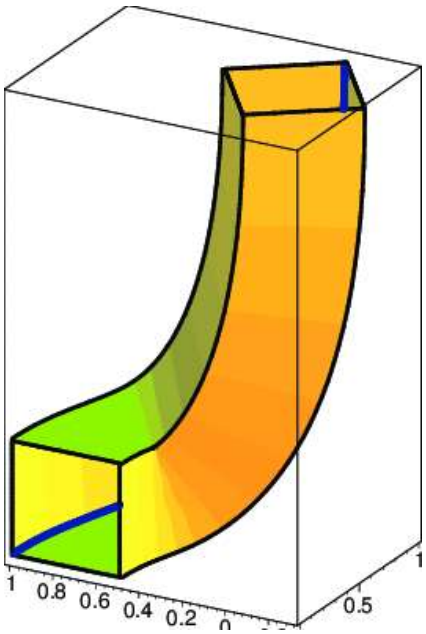
Bề mặt quét:

Bề mặt quét **swept surface** được tạo lập bằng cách cho một mặt đường quét theo chiều của một đường cong cố định khác.

Phương trình tham số của bề mặt quét là:

$$P(u, v) = G(u) + Q(v)$$

$0 \le u \le u_{max}, 0 \le v \le v_{max}$



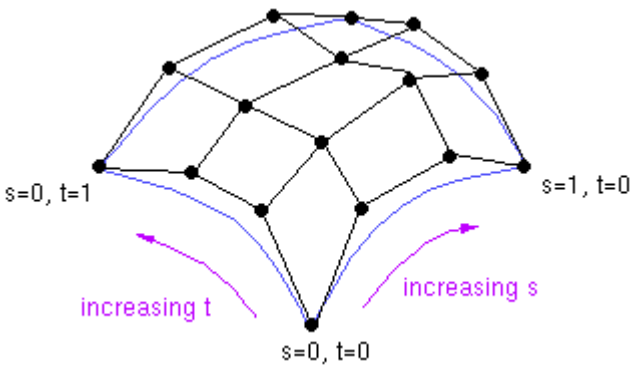
Biểu diễn bề mặt nhân tạo

Bề mặt cong Hermite

Bề mặt cong Hermite có thể được đặc tả bằng phương trình bậc 3 tương ứng:

$$r = V(s,t) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} s^i t^j$$

$0 \le s \le 1, 0 \le t \le 1$



Biểu diễn các đường cong tham biến (Parametric representation): x = f3(u), y = f3(u), z = f3(u) với u thuộc [0,1]

Theo Lagrange:

$$x = a_1 + b_1 u + c_1 u_2 + d_1 u_3$$

$$y = a_2 + b_2 u + c_2 u_2 + d_2 u_3$$

$$z = a_3 + b_3 u + c_3 u_2 + d_3 u_3$$

Với 3 phương trình 12 ẩn số, với 4 điểm p0, p1, p2, p3 phương trình xác định. Mỗi 1 điểm cho cặp 3 giá trị:

$$P_0 = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} \quad P_1 = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \quad P_2 = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} \quad P_3 = \begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix}$$

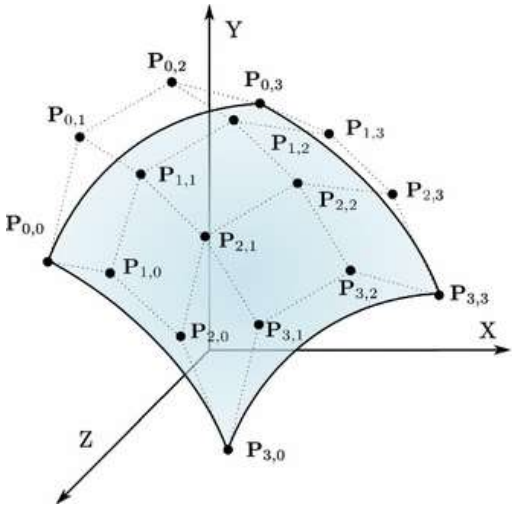
Bề mặt cong Bezier

Bề mặt cong Bezier có thể được đặc tả bằng phương trình:

$$r = V(s,t) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} BEZ_{i,3}(s) BEZ_{j,3}(t)$$

$$0 \le s \le 1, 0 \le t \le 1$$

với: $BEZ_{i,3}(s) = C_3^i s^i (1-s)^{3-i}$, và $BEZ_{j,3}(t) = C_3^j t^j (1-t)^{3-j}$



Ta có các phương trình:

$$\mathbf{p}(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} B_{m,i}(u) B_{n,j}(v) \mathbf{p}_{ij}$$

$$\mathbf{p}(u,v) = \sum_{i=0}^{m} B_{m,i}(u) \left(\sum_{j=0}^{n} B_{n,j}(v) \mathbf{p}_{ij} \right)$$

$$\mathbf{q}_i(v) = \sum_{j=0}^{n} B_{n,j}(v) \mathbf{p}_{ij}$$

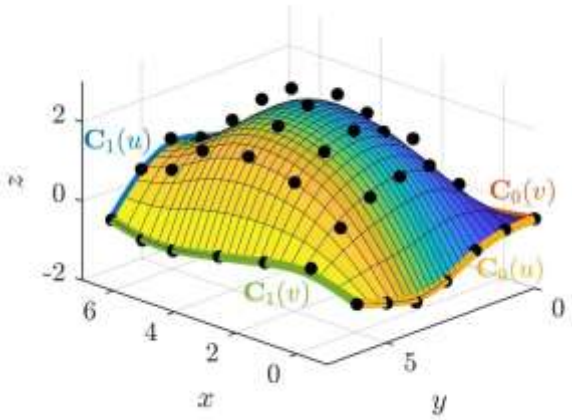
$$\mathbf{p}(u,v) = \sum_{i=0}^{m} B_{m,i}(u) \mathbf{q}_i(v)$$

Bề mặt cong đồng dạng B-Spline bậc 3

Bề mặt cong B-Spline bậc 3 là bề mặt cong phổ biến nhất trong các dạng của B-Spline, vì độ phức tạp tính toán thấp. Bề mặt cong B-Spline bậc 3 có thể biểu diễn dưới dạng ma trận như sau:

$$\mathbf{S}_i(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_{i-1} \\ \mathbf{p}_i \\ \mathbf{p}_{i+1} \\ \mathbf{p}_{i+2} \end{bmatrix}$$

$$0 \leq t \leq 1$$



Ta có các phương trình:

$$\mathbf{p}(u,v) = \sum_{i=0}^{m} N_{i,p}(u) \left(\sum_{j=0}^{n} N_{j,q}(v) \mathbf{p}_{i,j} \right)$$

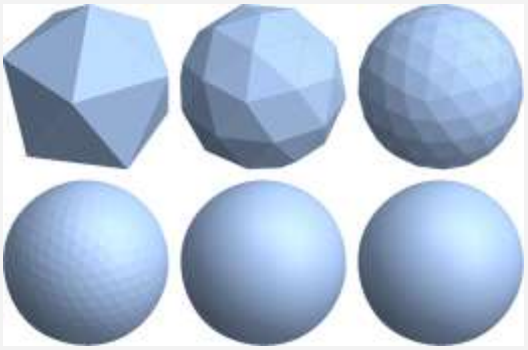
$$\mathbf{q}_i(v) = \sum_{j=0}^{n} N_{j,q}(v) \mathbf{p}_{i,j}$$

$$\mathbf{p}(u,v) = \sum_{i=0}^{m} N_{i,p}(u) \mathbf{q}_i(v)$$

Kỹ thuật *chia vùng*

Bề mặt chia vùng (subdivision surface) là một bề mặt sử dụng kỹ thuật chia trên vật thể thành những bề mặt con là trung bình của các bề mặt xung quanh. Kỹ thuật này có ứng dụng chính là làm mượt mà các góc cạnh, nơi giao giữa các mặt phẳng.

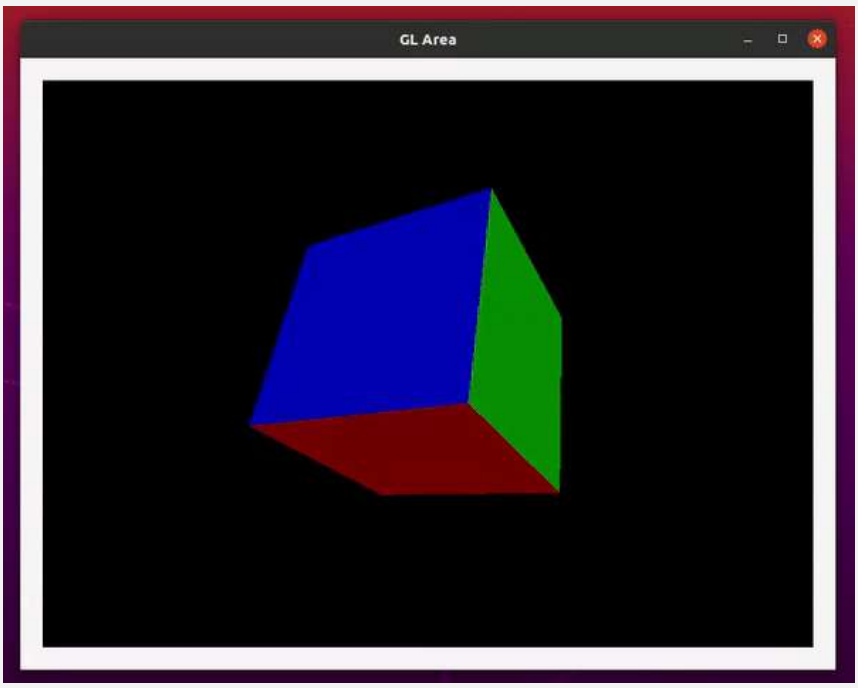
Bề mặt Doo-Sabin áp dụng theo phương pháp chia vùng



Khối cầu được tạo ra bằng cách làm mượt các bề mặt

2.8. Tạo animation từ vật thể 3D

Các vật thể 3D hiện nay được các ứng dụng cho phép tạo hoạt ảnh animation thông qua việc sử dụng các phép biến hình và liên kết với các khoảng thời gian cố định.



Minh họa hoạt ảnh animation của một khối cầu xoay quay trục z=x.

Animation trong 3D đóng nhiều vai trò quan trọng trong ứng dụng công nghệ hiện đại bây giờ, ví dụ như phát triển game, dựng hình ảnh phim CGI, mô phỏng khoa học…

Trong thư viện OpenGL, họ có cho chúng ta một hàm để gắn từng frame theo thời gian cho trước: `glutTimerFunc(unsigned int time, void(*callback)(int), int value)`

Ta có giải thuật cơ bản cho animation như sau:

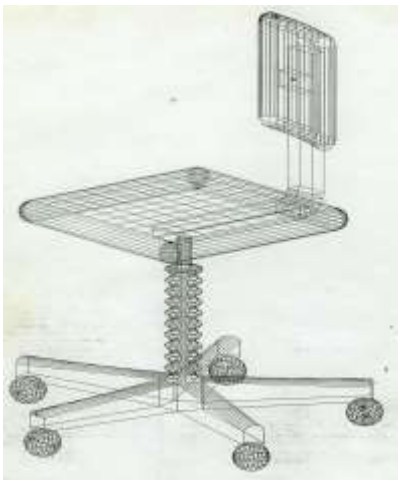
```
void timer(int t)
{
    glutPostRedisplay();
    glutTimerFunc(1000 / 60, timer, 0);

    // Ví dụ
    angle += 0.01;
    if (angle > 360)
        angle -= 360;
}

int main(int argc, char** argv) {
    ...
    glutTimerFunc(0, timer, 0);
    ...
}
```

Chương 3: Phương pháp

3.1. Ví dụ xây dựng mô hình chiếc ghế



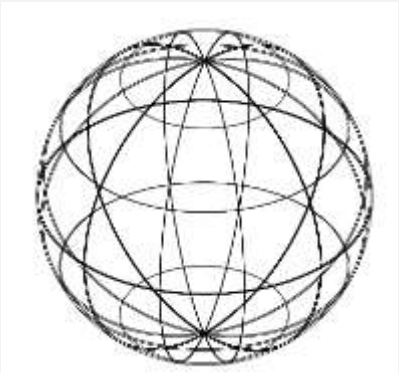
Ví dụ ta có mô hình chiếc ghế như hình bên phải, ta có thể xây dựng nó từ các vật thể cơ bản và sử dụng một trong các ứng dụng tạo dựng hình học vật thể ba chiều.

Ta tách chiếc ghế ra thành từng thành phần nhỏ, rồi sau đó có thể phân tích ra như sau:

- Bánh xe
- Chân xe
- Trục
- Chỗ ngồi
- Chỗ dựa
- Đồ hỗ trợ giữa chỗ ngồi và chỗ dựa

Bánh xe

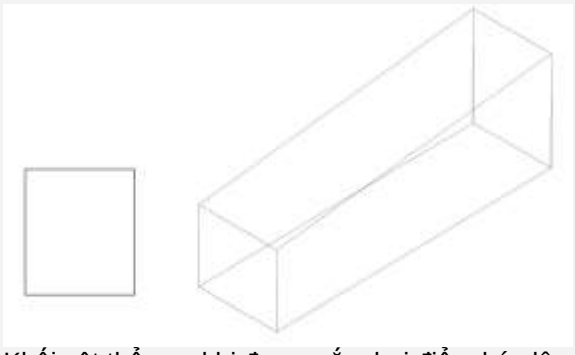
Bánh xe có thể được xây dựng bằng cách cho xây dựng một khối cầu cơ bản với bán kính nhỏ.



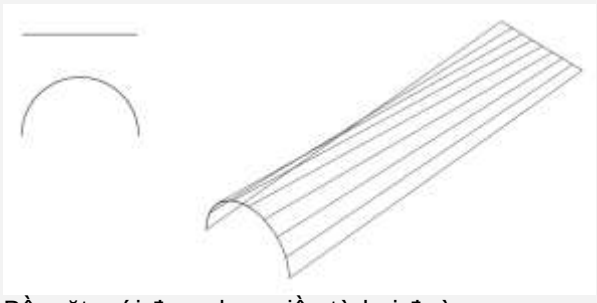
Khối cầu được sử dụng cho bánh xe

Chân xe

Phần chân xe đầu tiên chúng ta xây dựng một hình hộp chữ nhật trước, sau đó ta nắm và kéo hai điểm ở trên lên một khoảng.

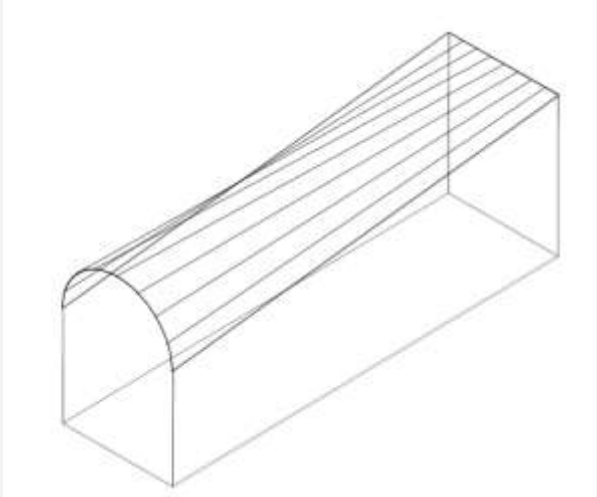


Khối vật thể sau khi được nắm hai điểm kéo lên (phần phía sau cao hơn)



Bề mặt mới được hợp viền từ hai đường cong

Sau đó, ta tiếp tục áp dụng kỹ thuật kết hợp hai vật thể lại, ta được một phần chân xe như đã được thiết kế trong hình:



Phần chân xe được xây dựng từ kỹ thuật kết hợp

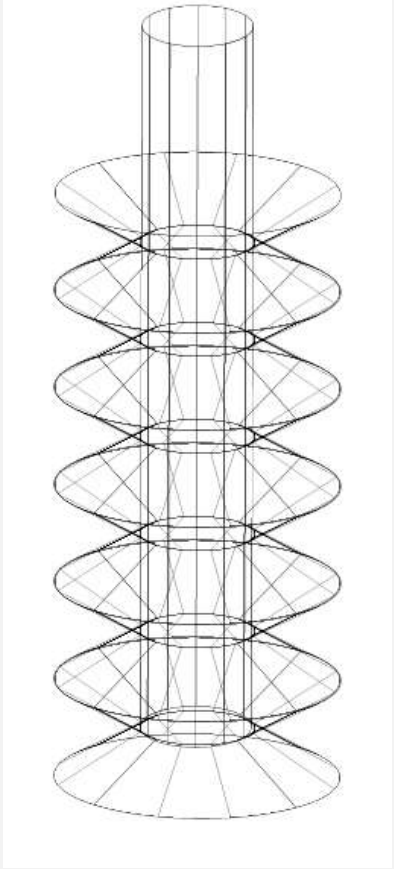
Cuối cùng, ta sử dụng phép quay 360/5 độ để tạo ra 5 chân xe.

Trục

Đầu tiên ta có thể dễ dàng nhìn thấy trục chiếc ghế được nâng bởi hình trụ cơ bản, do đó ta sẽ xây dựng hình trụ đầu tiên.

Tiếp theo ta vẽ một bề mặt được tạo từ đoạn thẳng IJ với hai điểm I, J cho trước, sau đó sử dụng kỹ thuật bề mặt quay 360 độ.
Lấy đối xứng bề mặt trên qua trục y, ta được hình hoàn chỉnh như bên dưới.

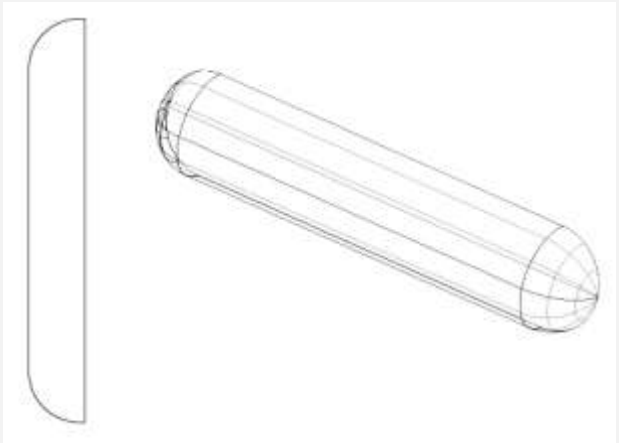
Nhân bản ra nhiều hình vừa tạo dọc theo trục khối trụ. Cuối cùng, sử dụng kỹ thuật kết hợp, ta được trục ghế xe.



Trục xe được xây dựng từ kỹ thuật kết hợp

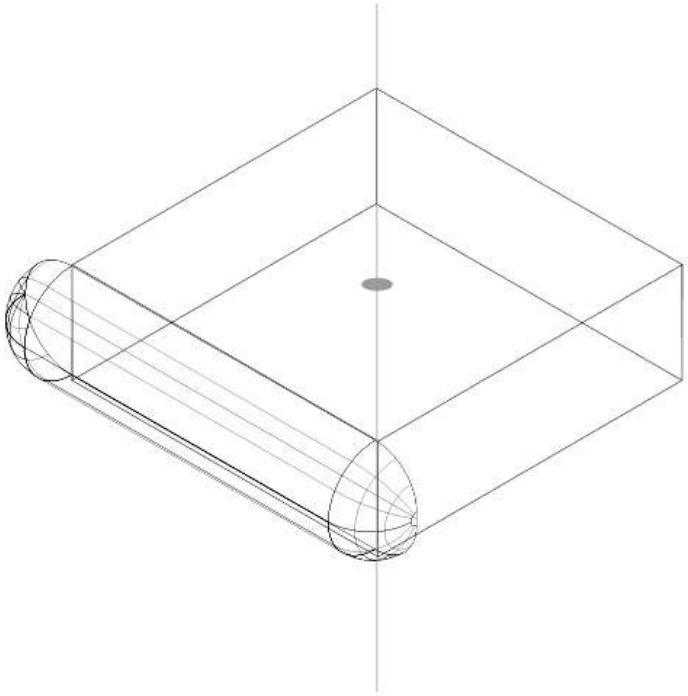
Chỗ ngồi

Đầu tiên ta vẽ hình chữ nhật, sau đó ta bo tròn hai góc của nó để trở thành một hình như sau. Sau đó ta sử dụng kỹ thuật bề mặt xoay 180 độ.



Kỹ thuật xoay 180 độ lấy cạnh bên phải làm trục cho ra hình bên phải

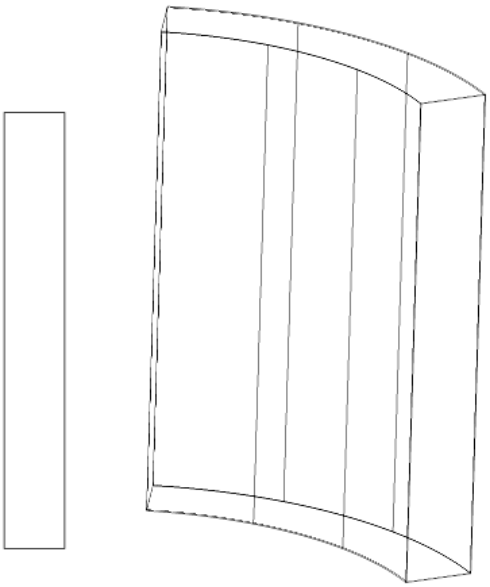
Tiếp theo ta sử dụng phép quay 90 độ cho vật thể vừa được tạo quanh trục chính giữa của hình hộp chữ nhật, ta được phần ngồi của chiếc ghế.



Chỗ dựa

Đối với chỗ dựa từ sử dụng lại vật thể đã xây dựng ở thành phần trước (vật thể từ phép xoay hình chữ nhật đã bo tròn).
Tiếp theo ta vẽ hình chữ nhật, rồi sử dụng kỹ thuật bề mặt xoay cho trục cách xa hình chữ nhật một đoạn R.

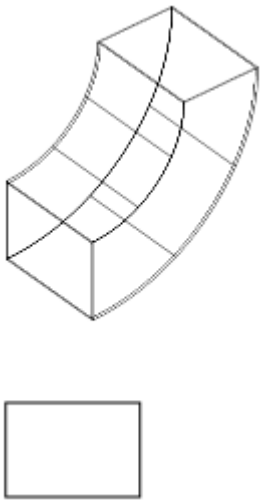
Sau khi sử dụng kỹ thuật bề mặt xoay ta được vật thể:



Cuối cùng ta ghép hai vật thể ở hai đầu vật thể vừa tạo ta được phần tựa của ghế ngồi.

Đồ hỗ trợ

Ở thành phần cuối cùng, ta có thể sử dụng lại các kỹ thuật vừa nêu. Sau đó ghép vào hai khối hình hộp chữ nhật.



Chương 5: Kết luận và hướng phát triển

Thông qua đồ án, chúng em tìm hiểu được về đồ họa máy tính 3D: wireframe , surface, solid. Biết thêm về các phép biến đổi, các phép chiếu, biểu diễn các bề mặt surface, biến đổi 3D thành 2D,... Nhóm xây dựng được các mô hình minh họa cho đồ án như mô phỏng chiếc ghế thông qua việc phân tích thành các thành phần nhỏ để giải quyết. Ngoài ra, đồ án còn giúp chúng em trong việc làm việc nhóm với nhau, sự cộng tác, trao đổi giữa các thành viên.

Nhận thấy rằng hướng nghiên cứu và phương pháp của đồ án đã là vấn đề mà các nhà khoa học đã đạt được từ nhiều thập kỷ trước mà không những thế nó còn để lại nhiều sự thành công và dấu ấn lớn trong nền khoa học vi tính thời hiện đại bây giờ. Do đó, bản thân nhóm chúng em đối với đồ án sẽ có thể không có một hướng phát triển mới nào cả, chỉ phát triển dựa trên những nghiên cứu đã có để cải tiến thêm.