LỜI CẢM ƠN

Trước hết em xin chân thành PGS.TS Đỗ Năng Toàn là giáo viên hướng dẫn em trong quá trình làm đồ án. Thầy đã giúp em rất nhiều và đã cung cấp cho em nhiều tài liệu quan trọng phục vụ cho quá trình tìm hiểu về đề tài "Tìm hiểu kỹ thuật đánh bóng Gauss trong đồ họa 3D".

Thứ hai, em xin chân thành cảm ơn các thầy, cô trong bộ môn công nghệ thông tin đã chỉ bảo bảo em trong quá trình học và rèn luyện trong 4 năm học vừa qua. Đồng thời em cảm ơn các bạn sinh viên lớp CT1201 đã gắn bó với em trong quá trình rèn luyện tại trường.

Cuối cùng em xin chân thành cảm ơn ban giám hiệu trường Đại Học Dân Lập Hải Phòng đã tạo điều kiện cho em có kiến thức, thư viện của trường là nơi mà sinh viên trong trường có thể thu thập tài liệu trợ giúp cho bài giảng trên lớp. Đồng thời các thầy cô trong trường giảng dạy cho sinh viên kinh nghiệm cuộc sống. Với kiến thức và kinh nghiệm đó sẽ giúp em cho công việc và cuộc sống sau này.

Em xin chân thành cảm ơn!

Hải Phòng, ngày tháng năm

Sinh viên

Đặng Minh Thắng

MỤC LỤC

PHẦN MỞ ĐẦU

Đồ họa máy tính là một lĩnh vực phát triển nhanh nhất trong tin học. Nó được áp dụng rộng rãi trong nhiều lĩnh vực khác nhau thuộc về khoa học, kỹ nghệ, y khoa, kiến trúc và giải trí.

Năm 1966, Sutherland ở Học viện Công nghệ Massachusetts là người đầu tiên đặt nền bóng cho đồ họa 3D bằng việc phát minh ra thiết bị hiển thị trùm đầu (head-amounted display) được điều khiển bởi máy tính đầu tiên. Nó cho phép người nhìn có thể thấy được hình ảnh dưới dạng lập thể 3D. Từ đó đến nay đồ họa 3D trở thành một trong những lĩnh vực phát triển rực rỡ nhất của đồ họa máy tính.

Nó được ứng dụng rộng rãi trong hầu hết tất cả các lĩnh vực như Điện ảnh, Hoạt hình, kiến trúc và các ứng dụng xây dựng các mô hình thực tại ảo…..Và không thể không nhắc đến vai trò tối quan trọng của đồ họa 3D trong việc tạo ra các game sử dụng đồ họa hiện nay …. Việc sử dụng đồ họa 3D trong game làm cho người chơi thích thú và có cảm giác như đang sống trong một thế giới thực. Có thể nói đồ họa 3D đã đang và sẽ tạo nên một nền công nghiệp game phát triển mạnh mẽ.

Mục đích chính của đồ họa 3D là tạo ra và mô tả các đối tượng, các mô hình trong thế giới thật bằng máy tính sao cho càng giống với thật càng tốt. Việc nghiên cứu các phương pháp các kỹ thuật khác nhau của đồ họa 3D cũng chỉ hướng đến một mục tiêu duy nhất đó là sao cho các nhân vật, các đối tượng, các mô hình được tạo ra trong máy tính giống thật nhất. Và một trong các phương pháp đó là tạo bóng cho đối tượng.

Xuất phát từ vấn đề này đồ án của em xây dựng gồm 3 chương:

CHƯƠNG 1:CÁC KIẾN THỨC CƠ BẢN ĐỒ HỌA 3D VÀ TẠO BÓNG

Chương này nói về các kiến thức cơ bản về ánh sáng, về hiển thị 3D và về các bộ đệm, và khái quát các kỹ thuật tạo bóng.

CHƯƠNG 2:KỸ THUẬT TẠO BÓNG GOURAUD

Chương này đi vào chi tiết kỹ thuật để tạo bóng Gouraud.

CHƯƠNG 3:CHƯƠNG TRÌNH THỬ NGHIỆM

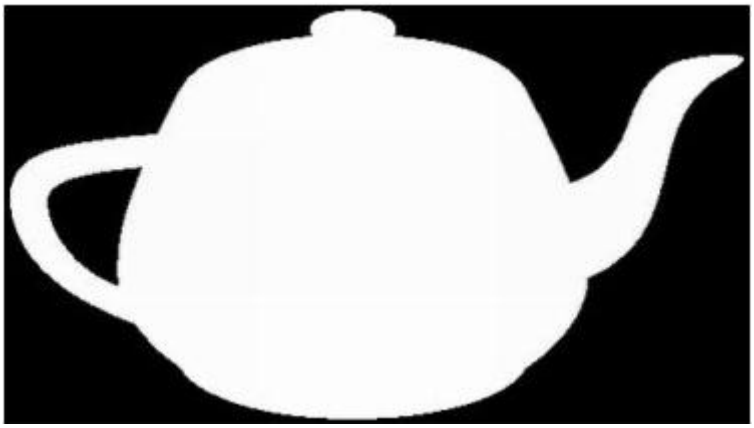
CHƯƠNG 1:CÁC KIẾN THỨC CƠ BẢN CỦA ĐỒ HỌA 3D VÀ TẠO BÓNG

CÁC KIẾN THỨC CƠ BẢN CỦA ĐỒ HỌA 3D

1.1. ÁNH SÁNG (LIGHTING)

Ánh sáng trong đồ họa 3D đóng vai trò khá quan trọng. Và đặc biệt nó là thành phần không thể thiếu để tạo ra bóng. Có nguồn sáng chỉ chiếu theo một hướng nhất đinh (giống ánh sáng mặt trời), có nguồn sáng chiếu ra toàn khung cảnh….Trong một khung cảnh có thể có nhiều nguồn sáng. Các nguồn sáng này có thể được tắt bật từng cái giống như ta tắt đèn bằng công tắc vậy. Theo mô hình ánh sáng của OpenGl thì ánh sáng gồm có 4 thành phần chính: Emissive Light, Ambient Light, Diffuse Light, Specular Light. Các thành phần này có thể được tính toán độc lập với nhau, và cuối cùng được kết hợp lại với nhau.

Ambient Light là ánh sáng bị phân rã bởi môi trường và không thể xác định hướng của chúng. Nếu trong một khung cảnh ta không xác định nguồn sáng thì kết quả đưa ra cũng giống như khi chúng ta sử dụng Ambient Light.



Hình 1.1: Chiếc ấm được chiếu bằng Ambient Light.

Diffuse Light (ánh sáng khuếch tán) là ánh sáng chiếu theo một hướng nhất, tuy nhiên khi nó gặp một bề mặt nó sẽ bị phân rã bằng nhau về mọi hướng, Vì thế nó sáng bằng nhau cho dù có đặt mắt nhìn ở đâu chăng nữa. Mọi nguồn sáng đến từ một điểm hay từ một hướng nhất định đều có thành phần Diffuse Light.

Hình 1.2: Ấm chè được chiều bằng Diffuse Light

Specular Light là ánh sáng phản xạ. Khi gặp một bề mặt nó sẽ phản xạ lại đúng theo quy luật phản xạ. Nó có thể được nhìn thấy trên những bề mặt cong.

Hình 1.3. Ấm chè được chiếu bằng Specular Light

1.2. HIỂN THỊ 3D (3D VIEWING)

1.2.1. Biểu diễn điểm và các phép biến đổi

Sự chuyển đổi từ tọa độ thế giới sang tọa độ của thiết bị là một chuỗi của các phép biến đổi affine và các phép chiếu. trong không gian Decarts 3 chiều.

Các phép biến đổi affine và các phép chiếu trong không gian Decarts 3 chiều có thể được biểu diễn tốt nhất bởi các ma trận 4x4 tương ứng với các tọa độ đồng nhất (Homogeneous coordinates) (x,y,z,w). Điểm 3D với tọa độ đồng nhất (x,y,z,w) sẽ có tọa độ affine là (x/w,y/w,z/w).

Mối quan hệ giữa tọa độ affine và tọa độ đồng nhất không phải là quan hệ 1-1. Cách đơn giản nhất để chuyển từ tọa độ affine (x,y,z) của một điểm sang tọa độ đồng nhất là đặt w=1: (x,y,z,1). Chúng ta thừa nhận rằng tất cả các tọa độ thế giới được biểu diễn bằng cách này.

Ta sẽ biểu diễn các phép biến đổi affine (như là co giãn (scaling transformations), phép quay (rotations), và phép tịnh tiến (translations)) bằng các ma trận mà sẽ không làm thay đổi thành phần w (w=1).

● Tịnh tiến bởi véc tơ $\vec{T} = (T_x, T_y, T_z)$:

$$\mathbf{M_t}(\vec{T}) = \begin{pmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \mathbf{M_t}(\vec{T}) \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x + T_x \\ y + T_y \\ z + T_z \\ 1 \end{pmatrix}$$

● Phép co giãn theo các nhân tố $\vec{S} = (S_x, S_y, S_z)$

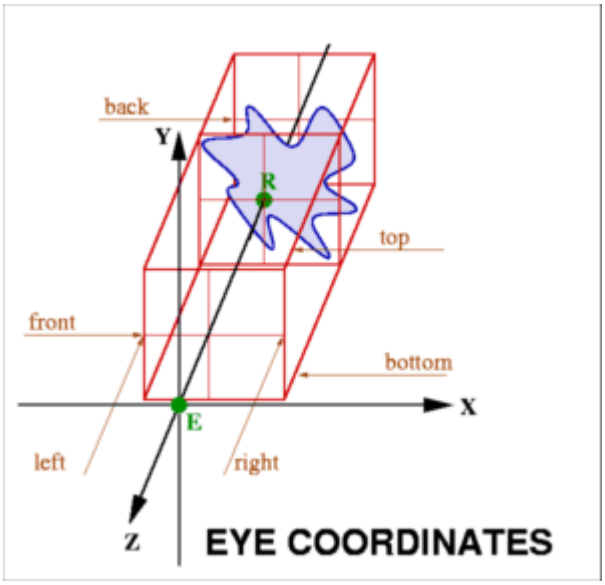
$$\mathbf{M_s}(\vec{S}) = \begin{pmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \mathbf{M_s}(\vec{S}) \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} S_x \cdot x \\ S_y \cdot y \\ S_z \cdot z \\ 1 \end{pmatrix}$$

● Phép quay quanh gốc tọa độ mà theo đó tập các véc tơ chuẩn tắc là $\{\vec{u}, \vec{v}, \vec{n}\}$, trực giao từng đôi một, sẽ được chuyển về $\{\vec{X}, \vec{Y}, \vec{Z}\}$.

$$\mathbf{M_r}(\vec{u}, \vec{v}, \vec{n}) = \begin{pmatrix} u_x & u_y & u_z & 0 \\ v_x & v_y & v_z & 0 \\ n_x & n_y & n_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

1.2.2. Phép chiếu trực giao (Orthographic Projection)

Trong trường hợp phép chiếu trực giao, vùng không gian hiển thị là một ống song song trong hệ tọa độ mắt. Các mặt của ống song song này song song với các mặt của hệ tọa độ mắt. Kích thước và vị trí của vùng không gian hiển thị được xác định bởi tọa độ mắt xleft, xright, ybottom, ytop, zfront và zback . (xleft, ybottom) và (xright, ytop) xác định một cửa sổ trong mặt phẳng chiếu (hoặc là bất kỳ mặt nào song song với mặt XY) mà vùng không gian hiển thị sẽ được hiển thị trên đó. Cửa sổ này phải được đưa về dạng hình vuông [-1,+1]2. zfront và zback định nghĩa 2 mặt phẳng cắt trước và cắt sau. Tọa độ của tất cả các điểm trong không gian (hoặc ít nhất là những điểm ta muốn nhìn) phải thỏa mãn zback $\leq$ z $\leq$ zfront . Khoảng giá trị của z phải được đưa về các giá trị chiều sâu (depth value) nằm trong đoạn [-1,+1]. Các điểm gần mắt hơn sẽ có giá trị chiều sâu nhỏ hơn.



Hình 1.4 : Vùng không gian hiển thị của phép chiếu trực giao.

Phép chiếu trực giao thu được bằng cách thực hiện các phép biến đổi sau theo thứ tự:

● Phép tịnh tiến $M_t(-\vec{M})$ sẽ đưa tâm của vùng không gian hiển thị về gốc tọa độ của hệ tọa độ mắt.

$$\vec{M} = \left(\frac{x_{\text{right}} + x_{\text{left}}}{2}, \frac{y_{\text{top}} + y_{\text{bottom}}}{2}, \frac{z_{\text{front}} + z_{\text{back}}}{2} \right)$$

● Một phép co giãn để đưa kích thước của vùng hiển thị về 2 đơn vị mỗi chiều.

● Một phép đối xứng qua mặt XY để các điểm nằm gần hơn sẽ nhận giá trị z nhỏ hơn.

Phép co giãn và phép đối xứng ở trên có thể thu được chỉ bằng một phép biến đổi đơn: $M_s(\vec{S})$ với:

$$\vec{S} = \left(\frac{2}{x_{\text{right}} - x_{\text{left}}}, \frac{2}{y_{\text{top}} - y_{\text{bottom}}}, \frac{-2}{z_{\text{front}} - z_{\text{back}}} \right)$$

Như vậy ma trận của phép chiếu trực giao sẽ là:

$$\mathbf{M_s}(\vec{S}) \cdot \mathbf{M_t}(-\vec{M}) = \begin{pmatrix} \frac{2}{x_{\text{right}} - x_{\text{left}}} & 0 & 0 & -\frac{x_{\text{right}} + x_{\text{left}}}{x_{\text{right}} - x_{\text{left}}} \\ 0 & \frac{2}{y_{\text{top}} - y_{\text{bottom}}} & 0 & -\frac{y_{\text{top}} + y_{\text{bottom}}}{y_{\text{top}} - y_{\text{bottom}}} \\ 0 & 0 & \frac{-2}{z_{\text{front}} - z_{\text{back}}} & \frac{z_{\text{front}} + z_{\text{back}}}{z_{\text{front}} - z_{\text{back}}} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Thành phần z không thay đổi, bởi vì phép chiếu trực giao là một phép biến đổi affine. Phép chiếu này được sử dụng trong các ứng dụng cần đến các quan hệ hình học (các tỉ số khoảng cách) như là trong CAD.

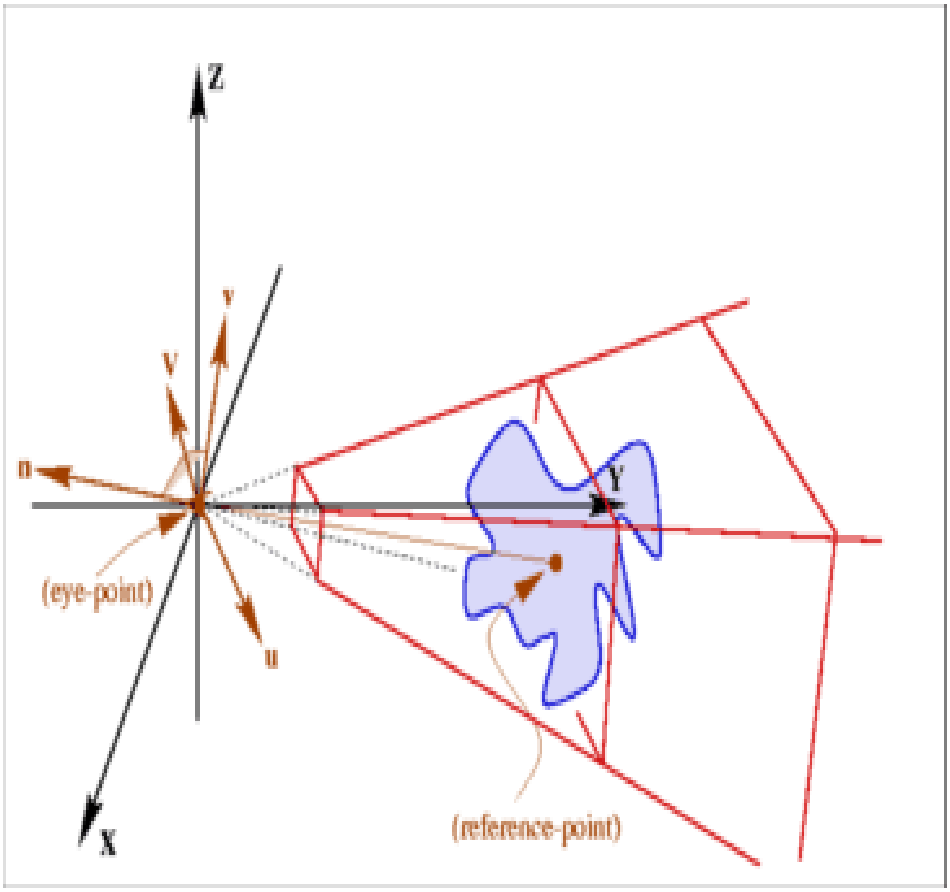
1.2.3. Phép biến đổi hiển thị (Viewing Transformation)

Phép biến đổi hiển thị sẽ đưa một camera ảo được cho tùy ý về một camera với điểm nhìn trùng với gốc tọa độ và hướng nhìn dọc theo chiều âm của trục Z (xem hình 2.1) Trục Y sau phép biến đổi tương ứng sẽ chỉ lên phía trên của màn hình. Trục X sẽ chỉ về phía phải.

Một cách thuận tiện để xác định vị trí của camera ảo là cho sẵn vị trí của điểm nhìn $\vec{E}$, Một điểm trong khung nhìn $\vec{R}$ (điểm tham chiếu) và một hướng $\vec{V}$ sẽ chỉ lên phía trên trong màn hình.

Phép biến đổi hiển thị sẽ gồm 2 bước:

- Một phép tịnh tiến sẽ đưa điểm nhìn $\vec{E}$ về gốc tọa độ. Ma trận biến đổi tương ứng sẽ là $M_t(-\vec{E})$. Kết quả sẽ như sau:



Hình 1.5: Vùng không gian hiển thị của phép biến đổi hiển thị

- Một phép quay sẽ chuyển hướng nhìn ngược về trục Z, quay vectơ $\vec{V}$ về mặt phẳng YZ. Vector $\vec{V}$ sẽ chỉ được quay về trùng với trục Y nếu $\vec{V}$ vuông góc với hướng nhìn. Trước hết ta sẽ xây dựng tập các véc tơ chuẩn tắc phù hợp trong tọa độ thế giới.

$\vec{n} = \frac{\vec{E} - \vec{R}}{\left\|\vec{E} - \vec{R}\right\|}$ Ngược với hướng nhìn $\rightarrow \vec{Z}$ $(\overrightarrow{Oz})$

$\vec{u} = \frac{\vec{V} \times \vec{n}}{\left\|\vec{V} \times \vec{n}\right\|}$ Chỉ về phía phải, vuông góc với $\vec{n} \rightarrow \overrightarrow{X}$

$\vec{v} = \vec{n} \times \vec{u}$ Chỉ lên giống $\vec{V}$, nhưng vuông góc với $\vec{n}$ và $\vec{u} \rightarrow \vec{Y}$

Như vậy ma trận của phép quay sẽ là: $M_r(\vec{u},\vec{v},\vec{n})$

Và do đó ma trận của phép biến đổi sẽ là:

$$\mathbf{M_r}(\vec{u},\vec{v},\vec{n}) \cdot \mathbf{M_t}(-\vec{E}) = \begin{pmatrix} u_x & u_y & u_z & 0 \\ v_x & v_y & v_z & 0 \\ n_x & n_y & n_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & -E_x \\ 0 & 1 & 0 & -E_y \\ 0 & 0 & 1 & -E_z \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} u_x & u_y & u_z & -\vec{u} \cdot \vec{E} \\ v_x & v_y & v_z & -\vec{v} \cdot \vec{E} \\ n_x & n_y & n_z & -\vec{n} \cdot \vec{E} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Trong đó $\vec{u},\vec{v}$ và $\vec{v}$ được tính từ $\vec{E}$, $\vec{R}$ và $\vec{V}$

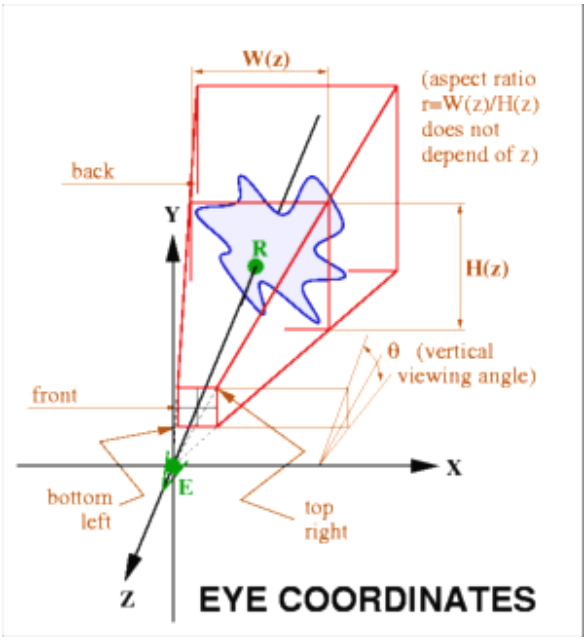
1.2.4. Phép chiếu phối cảnh (Perspective Projection)

Phép chiếu phối cảnh phù hợp và gần hơn với quan sát của con người (bằng một mắt) trong thế giới 3D. Tất cả các điểm trên một đường thẳng đi qua điểm nhìn sẽ được ánh xạ lên cùng một điểm trong màn hình 2D. Điểm ảnh này được xác định bởi tọa độ thiết bị chuẩn hóa x và y. Nếu 2 điểm được ánh xạ vào cùng một điểm trên màn hình, ta cần phải xác định điểm nào sẽ được hiển thị bằng thuật toán Z-buffer, nghĩa là so sánh chiều sâu của chúng. Vì lý do này chúng ta cần định nghĩa một thành phần tọa độ khác của thiết bị chuẩn hóa là z sao cho nó là một hàm tăng đơn điệu của khoảng cách từ điểm đó đến mặt phẳng mắt XY. Khoảng cách từ một điểm trong không gian đến mặt phẳng XY không bằng với khoảng cách từ điểm đó đến điểm nhìn (được đặt ở gốc tọa độ), nhưng nó sẽ được tính toán đơn giản hơn và cũng đủ để xác định được các mặt sẽ được hiển thị.

Như vậy, phép chiếu trực giao sẽ đưa một điểm (với tọa độ đồng nhất) trong hệ tọa độ mắt (x,y,z,1) về một điểm (tọa độ đồng nhất) trong hệ tọa độ cắt (x',y',z',w'). Sau đó các tọa độ của thiết bị chuẩn hóa (affine) (x",y",z") sẽ thu được bằng cách chia x',y',z' cho w' (Phép chia phối cảnh):

$$(x'', y'', z'') = \left(\frac{x'}{w'}, \frac{y'}{w'}, \frac{z'}{w'}\right)$$

Với phép chiếu phối cảnh, vùng không gian hiển thị là một hình tháp cụt với đầu mút là gốc tọa độ.



Hình 1.6: Vùng không gian hiển thị của phép chiếu phối cảnh cân xứng (Symmetrical Perspective Projection)

1.2.5. Phép biến đổi cổng nhìn (Viewport Transformation)

Phép biến đổi cổng nhìn chỉ gồm một phép tịnh tiến và một phép thay đổi tỉ lệ để:

● Tọa độ thiết bị chuẩn hóa (x, y) với $-1 \le x \le 1, -1 \le y \le 1$ được chuyển qua tọa độ pixel.

$$left \le x_w \le left + width$$

$$bottom \le y_w \le bottom + height$$

● Thành phần z với $-1 \le z \le 1$ được co lại trong đoạn $0 \le z_w \le 1$.

Giá trị z_w này sẽ được sử dụng để loại bỏ những bề mặt bị ẩn. Những điểm có giá trị z_w nhỏ sẽ nằm trước những điểm có giá trị z_w lớn hơn.

Xây dựng ma trận biến đổi là công việc đơn giản. Tuy nhiên sẽ hiệu quả hơn nếu ta thực hiện phép biến đổi một cách trực tiếp:

$$x_w = left + width \cdot \frac{x+1}{2}$$

$$y_w = bottom + height \cdot \frac{y+1}{2}$$

$$z_w = \frac{z+1}{2}$$

1.3. BỘ ĐỆM VÀ CÁC PHÉP KIỂM TRA

Một mục đích quan trọng của hầu hết các chương trình đồ họa là vẽ được các bức tranh ra màn hình. Màn hình là một mảng hình vuông của các pixel. Mỗi pixel đó có thể hiển thị được 1 màu nhất định. Sau các quá trình quét (bao gồm Texturing và fog…), dữ liệu chưa trở thành pixel, nó vẫn chỉ là các "mảnh" (Fragments). Mỗi mảnh này chứa dữ liệu chung cho mỗi pixel bên trong nó như là màu sắc là giá trị chiều sâu. Các mảnh này sau đó sẽ qua một loạt các phép kiểm tra và các thao tác khác trước khi được vẽ ra màn hình.

Nếu mảnh đó qua được các phép kiểm tra (test pass) thì nó sẽ trở thành các pixel. Để vẽ các pixel này, ta cần phải biết được màu sắc của chúng là gì, và thông tin về màu sắc của mỗi pixel được lưu trong bộ đệm màu (Color Buffer).

Nơi lưu trữ dữ liệu cho từng pixel xuất hiện trên màn hình được gọi là bộ đệm (Buffer). Các bộ đệm khác nhau sẽ chưa một loại dữ liệu khác nhau cho pixel và bộ nhớ cho mỗi pixel có thể sẽ khác nhau giữa các bộ đệm. Nhưng trong một bộ đệm thì 2 pixel bất kỳ sẽ được cấp cùng một lượng bộ nhớ giống nhau. Một bộ đệm mà lưu trữ một bít thông tin cho mỗi pixel được gọi là một bitplane. Có các bộ đệm phổ biến như Color Buffer, Depth Buffer, Stencil Buffer, Accumulation Buffer.

1.3.1. Bộ đệm chiều sâu (Z-Buffer)

1.3.1.1. Khái niệm:

Là bộ đệm lưu trữ giá trị chiều sâu cho từng Pixel. Nó được dùng trong việc loại bỏ các bề mặt ẩn. Giả sử 2 điểm sau các phép chiếu được ánh xạ vào cùng một pixel trên màn hình. Như vậy điểm nào có giá trị chiều sâu (z) nhỏ hơn sẽ được viết đè lên điểm có giá trị chiều sâu lớn hơn. Chính vì vậy nên ta gọi bộ đệm này là Z-buffer.

1.3.1.2. Depth test: Với mỗi pixel trên màn hình, bộ đệm chiều sâu lưu khoảng cách vuông góc từ điểm nhìn đến pixel đó. Nên nếu giá trị chiều sâu của một điểm được ánh xạ vào pixel đó nhỏ hơn giá trị được lưu trong bộ đệm chiều sâu thì điểm này được coi là qua Depth test (depth test pass) và giá trị chiều sâu của nó được thay thế cho giá trị lưu trong bộ đệm. Nếu giá trị chiều sâu của điểm đó lớn hơn giá trị lưu trong Depth Buffer thì điểm đó "trượt" phép kiểm tra chiều sâu. (Depth test Fail)

1.3.2. Bộ đệm khuôn (Stencil Buffer)

1.3.2.1. Khái niệm: Bộ đệm khuôn dùng để giới hạn một vùng nhất định nào đó trong khung cảnh. Hay nói cách khác nó đánh dấu một vùng nào đó trên màn

hình. Bộ đệm này được sử dụng để tạo ra bóng hoặc để tạo ra ảnh phản xạ của một vật thể qua gương…

1.3.2.2. Stencil Test: Phép kiểm tra Stencil chỉ được thực hiện khi có bộ đệm khuôn. (Nếu không có bộ đệm khuôn thì phép kiểm tra Stencil được coi là luôn pass). Phép kiểm tra Stencil sẽ so sánh giá trị lưu trong Stencil Buffer tại một Pixel với một giá trị tham chiếu theo một hàm so sánh cho trước nào đó. OpenGL cung cấp các hàm như là GL_NEVER, GL_ALWAYS, GL_LESS, GL_LEQUAL, GL_EQUAL, GL_GEQUAL, GL_GREATER hay là GL_NOTEQUAL. Giả sử hàm so sánh là GL_LESS, một "mảnh" (Fragments) được coi là qua phép kiểm tra (pass) nếu như giá trị tham chiếu nhỏ hơn giá trị lưu trong Stencil Buffer.

Ngoài ra OpenGL còn hỗ trợ một hàm là

glStencilOp(GLenum fail, GLenum zfail, GLenum zpass);

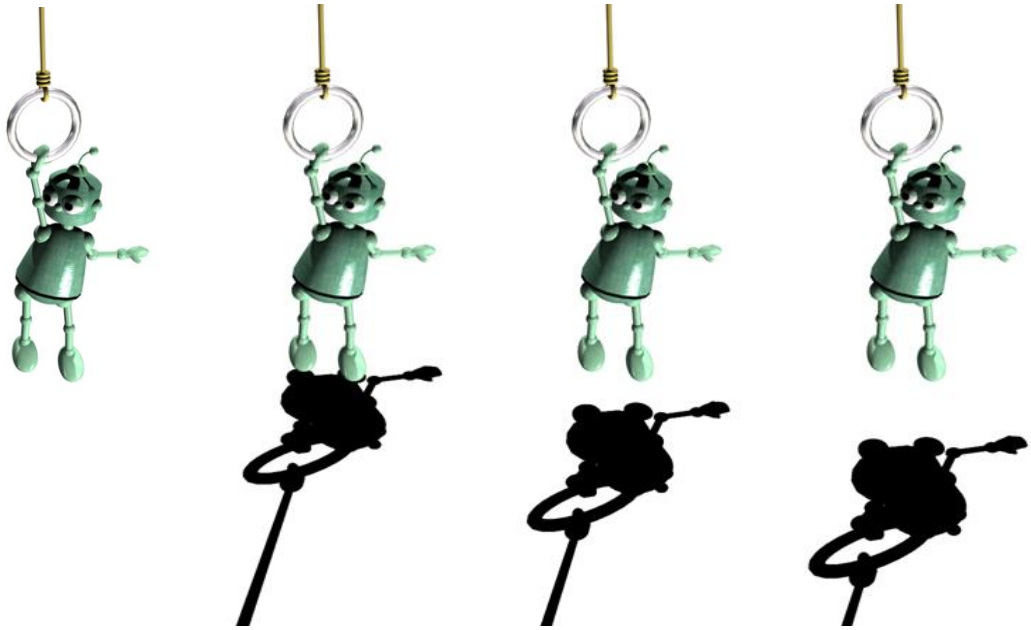
Hàm này xác định dữ liệu trong stencil Buffer sẽ thay đổi thế nào nếu như một "mảnh" pass hay fail phép kiểm tra stencil. 3 hàm fail, zfail và zpass có thể là GL_KEEP, GL_ZERO, GL_REPLACE, GL_INCR, GL_DECR …Chúng tương ứng với giữ nguyên giá trị hiện tại, thay thế nó với 0, thay thế nó bởi một giá trị tham chiếu, tăng và giảm giá trị lưu trong stencil buffer. Hàm fail sẽ được sử dụng nếu như "mảnh" đó fail stencil test. Nếu nó pass thì hàm zfail sẽ được dùng nếu Depth test fail và tương tự, zpass được dùng nếu như Depth test pass hoặc nếu không có phép kiểm tra độ sâu nào được thực hiện. Mặc định cả 3 tham số này là GL_KEEP.

1.4. TẠO BÓNG

1.4.1. Khái niệm bóng:

"Bóng (Shadow) là một vùng tối nằm giữa một vùng được chiếu sáng, xuất hiện khi một vật thể được chiếu sáng toàn bộ hoặc một phần"

Bóng là một trong những yếu tố quan trọng nhất của tri giác con người về việc nhận biết các vật thể trong thế giới 3 chiều. Bóng giúp cho ta nhận biết được vị trí tương đối của vật đổ bóng (occluder) với mặt nhận bóng (receiver), nhận biết được kích thước và dạng hình học của cả vật đổ bóng và mặt nhận bóng.



Hình 1.7: Bóng cung cấp thông tin về vị trí tương đối của vật thể. Với ảnh ở bên trái ta không thể biết được vị trí của con rối. Nhưng với lần lượt 3 ảnh ở bên phải ta thấy vị khoảng cách của chúng so với mặt đất xa dần.

Hình 1.8: Bóng cung cấp thông tin về dạng hình học của mặt tiếp nhận. Hình bên trái ta không thể biết được dạng hình học của mặt tiếp nhận, còn mặt bên phải thì dễ dàng thấy được.

1.4.2 Các phương pháp chính của tạo bóng.

1.4.2.1. Tạo bóng cứng

Các tính toán tạo bóng thực chất là việc xác định xem một điểm trong khung nhìn có nằm trong vùng bóng không. Một cách cơ bản nó là một phép kiểm tra tính hiển thị của một điểm. Các thuật toán tạo bóng cứng phổ biến là:

+ Tạo bóng giả (Fakes Shadow)

Các thuật toán tạo bóng giả bao gồm các trường hợp đặc biệt tạo bóng không đúng đắn bằng các phương pháp toán học. Những kỹ thuật này chỉ được sử dụng trong những trường hợp đặc biệt (Ví dụ như bóng chỉ được vẽ cho những đối

tượng đặc biệt, hoặc bóng chỉ được vẽ lên một mặt phẳng. Tuy nhiên các phương pháp này cũng tạo ra bóng làm cho ta có cảm giác khá thật.

+ Bóng khối (Shadow Volume)

Bóng khối là một kỹ thuật tạo bóng cần đến cấu trúc hình học của vật đổ bóng. Vật đổ bóng phải được tạo bởi các khối đa giác. Theo đó ta sẽ tìm những đỉnh và cạnh viền, là những cạnh đóng vai trò tạo nên bóng khối. Một tia sáng chiếu tới vật thể sẽ tiếp xúc với vật thể tại điểm hoặc cạnh viền đó và đi cắt mặt phẳng nhận bóng. Những cạnh viền, và đỉnh viền này sẽ tạo ra các mặt bên đa giác của bóng khối. Từ đó dựa vào các phép kiểm tra ta sẽ kiểm tra được một điểm trong khung cảnh có thuộc bóng khối hay không. Việc xác định các cạnh viền và kiểm tra ta sẽ nghiên cứu ở phần dưới.

+ Dùng bản đồ bóng (Shadow Mapping)

Đây là thuật toán dùng đến bộ đệm chiều sâu (Depth Buffer). Ý tưởng chủ yếu là sử dụng bản đồ chiều sâu (hay còn gọi là bản đồ bóng) để lưu trữ các giá trị chiều sâu khi tạo ảnh từ vị trí của ánh sáng rồi sau đó sử dụng các giá trị này để xác định pixel nào được chiếu sáng hay là nằm trong bóng.

+ Lần theo tia sáng (Ray Tracing)

Thuật toán này sử dụng kỹ thuật Ray Tracing:

Với mỗi tia sáng đi ra từ mắt ta vào một không gian là một đường thẳng sẽ cắt vào cửa sổ (màn hình) và chạm vào vật thể trong không gian (gần nhất từ mắt). Tại điểm chạm vào vật thể đó thì tuỳ ở mỗi điểm chạm của vật thể đó có tính chất như thế nào mà ta chia ra các tia sáng tiếp theo.

Nếu điểm chạm đó có tính khúc xạ, phản xạ thì ta lại lần theo tia sáng đó theo từng tia phản xạ, khúc xạ...

Nếu tại điểm chạm đó vật thể có tính xuyến thấu, phản xạ tức là 1 phần của tia sáng đi qua vật thể đó, một phần tia sáng đó được phản xạ ta lại xét từng tia....tiếp tục mỗi tia lại chạm vào vật thể khác lại chia ra từng tia khúc xạ phản xạ riêng ở mỗi điểm chạm.

Sau khi cắt mọi vật thể có thể trong không gian ta tính màu tại tia từ mắt cắt ở cửa sổ và đặt ở đó 1 giá trị màu. Tương ứng quét tất cả các tia từ mắt đến màn hình...

Bóng tạo bởi kỹ thuật này trông rất thật. Nhưng chi phí để thực hiện nó quá đắt vì phải thực hiện quá nhiều phép tính. Chính vì vậy kỹ thuật này ít được sử dụng trong các ứng dụng thời gian thực.

1.4.2.2. Tạo bóng mềm.

Các kỹ thuật tạo bóng mềm sẽ cho bóng sinh ra trông thật hơn rất nhiều so với bóng được sinh ra bởi các thuật toán tạo bóng cứng. Tính thật của nó được biểu hiện bởi cả vùng nửa bóng và vùng thuần bóng. Hình dạng của bóng sinh ra bởi các thuật toán tạo bóng mềm sẽ phụ thuộc vào hình dạng, kích thước của vật thể tạo bóng , nguồn sáng và cả vị trí tương đối giữa nguồn sáng và vật thể.

Các kỹ thuật tạo bóng mềm phổ biến có thể kể đến là:

+ Thuật toán bộ đệm khung (Frame Buffer Algorithms)

Được đề xuất bởi Brotman và Badler dựa trên việc sinh ra các đa giác thuần bóng trong suốt quá trình tiền xử lý. Bộ đệm chiều sâu 2D mà được sử dụng để xác định mặt được hiển thị sẽ được mở rộng để lưu bộ đếm nắm giữ các thông tin để xác định xem một pixel bất kỳ là nằm trong vùng nửa bóng hay vùng thuần bóng.

+ Dõi quang tia 2 chiều và phân bố (Distributed and Bidirectional Ray Tracing)

Rất nhiều mở rộng của thuật toán Ray-Tracing được sử dụng để tạo bóng mềm. Dõi quang tia phân bố cung cấp một kỹ thuật tạo bóng láng, mờ và chuyển động mờ trong khi Dõi quang tia 2 chiều cung cấp một phương pháp tạo bóng mềm rất nhanh.

+ Ánh sáng nâng cao (Radiosity)

Radiosity là một kỹ thuật tạo bóng mềm bằng cách tính toán tất cả các phản xạ, khuếch tán ánh sáng giữa các mặt khác nhau của tất cả các vật thể trong khung cảnh. Nó hầu như chỉ được sử dụng cho các mặt đa giác bởi vì chi phí tính toán của phương pháp này rất lớn.

1.4.2.3. Tạo bóng Phong.

Phương pháp tô bóng Phong là thuật ngữ dùng để chỉ một tập hợp các kỹ thuật trong Đồ họa 3D bao gồm: một mô hình phản xạ ánh sáng từ các bề mặt và một phương pháp dùng để ước lượng màu sắc của điểm ảnh bằng cách nội suy véc-tơ trực giao bề mặt.

Mô hình phản xạ thường được biết tới với tên gọi Mô hình Phản xạ Phong (Phong reflection model), Mô hình Chiếu sáng Phong (Phong illumination) hay là Mô hình thắp sáng Phong (Phong lighting).

Phương pháp nội suy thường được gọi là Nội suy Phong (Phong interpolation). Phương pháp này còn được biết đến là phương pháp thắp sáng mỗi điểm ảnh (per-pixel lighting).

1.4.2.4. Tạo bóng Gouraud

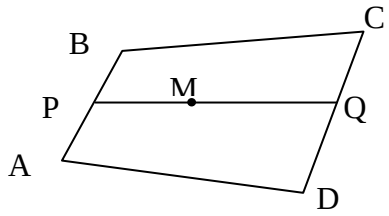
Gouraud shading được đặt theo tên của Henri Gouraud , là một phương pháp nội suy được sử dụng trong đồ hoạ máy tính để liên tục đổ bóng bề mặt đại diện bởi các lưới đa giác.

Trong thực tế, tô bóng Gouraud được sử dụng thường xuyên nhất để đạt được ánh sáng liên tục trên bề mặt hình tam giác bằng cách tính toán chiếu sáng ở các góc của mỗi tam giác và nội suy tuyến tính màu sắc cho mỗi điểm ảnh bao phủ bởi tam giác.

Ý tưởng của phương pháp này là tính toán cường độ ánh sáng tại một số điểm và sau đó dùng phương pháp nội suy để suy ra cường độ sáng của các điểm khác trên mặt. Cụ thể hơn ở đây ta tính cường độ ánh sáng tại các nút mạng lưới dựa vào các véc tơ pháp tuyến đã tính được. Sau đó sử dụng phương pháp nội suy tuyến tính để suy ra cường độ của các đường biên của đa giác. Và cũng bằng phương pháp nội suy tuyến tính tính ra cường độ của các điểm nằm bên trong đa

giác. Và cũng bằng phương pháp nội suy tuyến tính tính ra cường độ của các điểm nằm bên trong đa giác.

Ví dụ: một mặt trong mạng lưới ABCD, véc tơ pháp tuyến tại A, B, C, D đã tính được (không phải làm nội suy) và dựa vào góc tới của tia sáng sẽ tính được độ sáng tại các điểm đó, xét tỉ lệ của P chia AB thì ta dùng phương pháp nội suy tuyến tính tính được độ sáng tại P. Giả sử PA/PB =k thì



Hình 1.9: Mạng lưới ABCD

IP= (IA+k*IB)/(k+1)

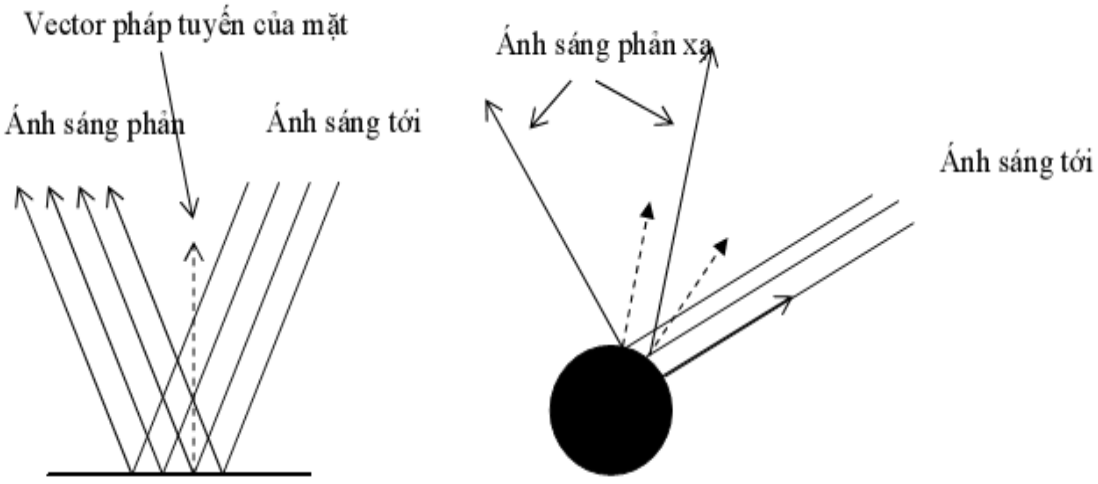
Tương tự ta có thể tính độ sáng tại Q nếu ta biết nó chia CD theo tỷ lệ nào, còn một điểm trong đa giác nằm giữa PQ cũng được nội suy như thế. Có điều chú ý ở đây là ta xét các dòng PQ theo độ phân giải của màn hình, là dòng pixel, tức là xét từ trên xuống dưới bắt đầu từ dùng vị trí thấp đến vị trí cao của toạ độ màn hình.

CHƯƠNG 2: KỸ THUẬT TẠO BÓNG GOURAUD

2.1. CÁC LOẠI NGUỒN SÁNG.

2.1.1. NGUỒN SÁNG XUNG QUANH

Ánh sáng xung quanh là mức sáng trung bình, tồn tại trong một vùng không gian. Một không gian lý tưởng là không gian mà tại đó mọi vật đều được cung cấp một lượng ánh sáng lên bề mặt là như nhau, từ mọi phía ở mọi nơi. Thông thường ánh sáng xung quanh được xác định với một mức cụ thể gọi là mức sáng xung quanh của vùng không gian mà vật thể đó cư ngụ, sau đó ta cộng với cường độ sáng có được từ các nguồn sáng khác để có được cường độ sáng cuối cùng lên một điểm hay một mặt của vật thể

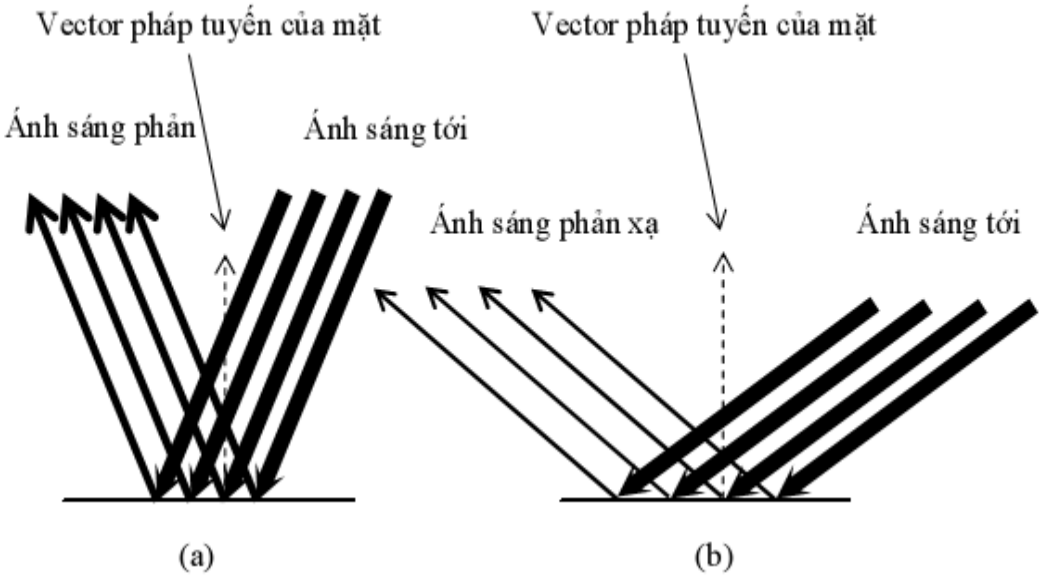


Hình 2.1: Sự phản xạ ánh sáng

2.1.2. NGUỒN SÁNG ĐỊNH HƯỚNG

Nguồn sáng định hướng giống như những gì mà mặt trời cung cấp cho chúng ta. Nó bao gồm một tập các tia sáng song song, bất kể cường độ của chúng có giống nhau hay không. Có hai loại kết quả của ánh sáng định hướng khi chúng chiếu đến bề mặt là: khuyếch tán và phản chiếu. Nếu bề mặt phản xạ toàn bộ (giống như mặt gương) thì các tia phản xạ sẽ có hướng ngược với hướng của góc tới. Trong trường hợp ngược lại, nếu bề mặt là không phản xạ toàn phần (có độ nhám, xù xì) thì một phần các tia sáng sẽ bị toả đi các hướng khác hay bị hấp thụ, phần còn lại thì phản xạ lại, và lượng ánh sáng phản xạ lại này tỷ lệ với góc tới. Ở đây chúng ta sẽ quan tâm đến hiện tượng phản xạ không toàn phần vì đây là hiện

tượng phổ biến (vì chỉ có những đối tượng được cấu tạo từ những mặt như mặt gương mới xảy ra hiện tượng phản xạ toàn phần), và đồng thời tìm cách tính cường độ của ánh sáng phản xạ trên bề mặt.



Hình 2.2: Sự phản xạ không toàn phần của ánh sáng

Trong *hình 2.2* thể hiện sự phản xạ ánh sáng không toàn phần. Độ đậm nét của các tia ánh sáng tới thể hiện cường độ sáng cao, độ mảnh của các tia phản xạ thể hiện cường độ sáng thấp. Nói chung, khi bề mặt là không phản xạ toàn phần thì cường độ của ánh sáng phản xạ (hay tạm gọi là tia phản xạ) luôn bé hơn so với cường độ của ánh sáng tới (hay gọi là tia tới), và cường độ của tia phản xạ còn tỷ lệ với góc giữa tia tới với vector pháp tuyến của bề mặt, nếu góc này càng nhỏ thì cường độ phản xạ càng cao,nếu góc này lớn thì cường độ phản xạ rất thấp. Ở đây ta chỉ quan tâm đến thành phần ánh sáng khuyếch tán và tạm bỏ qua hiện tượng phản xạ toàn phần. Để cho tiện trong việc tính toán ta tạm đổi hướng của tia tới thực sự, vậy bây giờ hướng của tia tới được xem là hướng ngược lại của tia sáng tới.

Nếu gọi θ là góc giữa tia tới với vector pháp tuyến của bề mặt thì Cos(θ) phụ thuộc vào tia tới a và vector pháp tuyến của mặt n theo công thức:

$$Cos(\theta) = \frac{\vec{a}.\vec{n}}{|\vec{a}|.|\vec{n}|}$$

Trong công thức trên Cos(θ) bằng tích vô hướng của a và n chia cho tích độ lớn của chúng. Nếu ta đã chuẩn hóa độ lớn của các vector a và n về 1 từ trước thì ta có thể tính giá trị trên một cách nhanh chóng như sau:

$$\textbf{Cos(}\theta\textbf{) = tích vô hướng của } \vec{a} \textbf{ và } \vec{n} =$$

a.x*n.x+a.y*n.y+a.z*n.z

Vì Cos(θ) có giá trị từ +1 đến -1 nên ta có thể suy ra công thức tính cường độ của ánh sáng phản xạ là:

Cường độ ánh sáng phản xạ = Cường độ của ánh sáng định hướng * [(Cos(θ)+1)/2]

Trong đó [(Cos(θ)+1)/2] có giá trị trong khoảng từ 0 đến 1. Vậy qua công chúng ta có thể tính được cường độ của ánh sáng phản xạ trên bề mặt khi biết được cường độ của ánh sáng định hướng cũng như các vector pháp tuyến của mặt và tia tới.

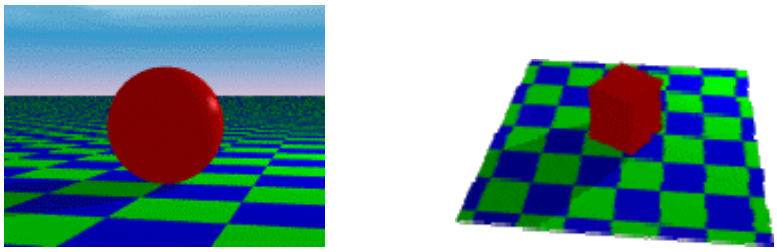
2.1.3. NGUỒN SÁNG ĐIỂM

Nguồn sáng định hướng là tương đương với nguồn sáng điểm đặt ở vô tận. Nhưng khi nguồn sáng điểm được mang đến gần đối tượng thì các tia sáng từ nó phát ra không còn song song nữa mà được toả ra theo mọi hướng theo dạng hình cầu. Vì thế, các tia sáng sẽ rơi xuống các điểm trên bề mặt dưới các góc khác nhau. Giả sử vector pháp tuyến của mặt là n=(xn, yn, zn), điểm đang xét có toạ độ là (x0, y0, z0) và nguồn sáng điểm có tọa độ là (plx, ply, plz) thì ánh sáng sẽ rọi đến điểm đang sét theo vector (x0- plx, y0-ply, z0-plz), hay tia tới:

a = (plx - x0, ply - y0,plz - z0).

Từ đó cường độ sáng tại điểm đang xét sẽ phụ thuộc vào Cos(θ) giữa n và a như đã trình bày trong phần nguồn sáng định hướng.

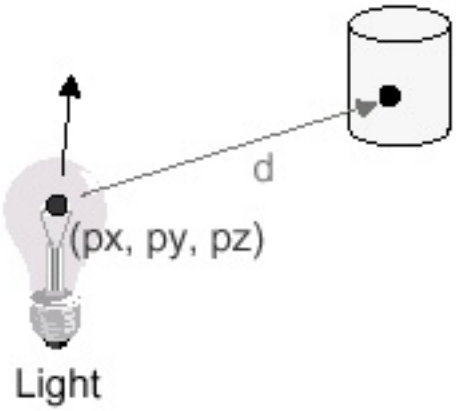
Vậy với nguồn sáng định hướng, chúng ta cần tính tia tới cho mọi điểm trên mặt, từ đó kết hợp với vector pháp tuyến của mặt để tính được cường độ sáng tại điểm đó, nếu tính toán trực tiếp thì có thể mất khá nhiều thời gian do phải tính vector a và tính Cos(θ) thông qua công thức với tất cả các điểm trên mặt. Nên nhớ rằng trong tình hướng nguồn sáng điểm thì chúng ta buộc lòng phải tính Cos(θ) thông qua công thức vì vector a sẽ thay đổi khi mặt hay nguồn sáng thay đổi (trừ khi mặt tĩnh, song nếu mặt tĩnh và nguồn sáng cố định thì suy ra chúng ta chỉ cần tính cường độ sáng một lần).



Hình 2.3: Ví dụ về nguồn sáng điểm

Hướng của các tia sáng sẽ thay đổi với các điểm khác nhau trên bề mặt. Như vậy, ta phải tính vector chỉ phương cho mỗi điểm:

$$\vec{d} = \frac{\vec{p} - \vec{l}}{\|\vec{p} - \vec{l}\|}$$



$$I_L = \frac{I_0}{k_c + k_l d + k_q d^2}$$

Trong đó k_c, k_l, k_q là hệ số suy giảm theo khoảng cách d.

2.2. ĐẶC TRƯNG CỦA TẠO BÓNG GOURAUD

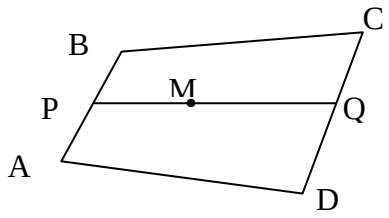
Gouraud shading, được đặt theo tên của Henri Gouraud, là một phương

pháp nội suy được sử dụng trong đồ hoạ máy tính để liên tục đổ bóng bề mặt đại diện bởi các lưới đa giác.

Trong thực tế, tô bóng Gouraud được sử dụng thường xuyên nhất để đạt được ánh sáng liên tục trên bề mặt hình tam giác bằng cách tính toán chiếu sáng ở các góc của mỗi tam giác và nội suy tuyến tính màu sắc cho mỗi điểm ảnh bao phủ bởi tam giác.

Ý tưởng của phương pháp này là tính toán cường độ ánh sáng tại một số điểm và sau đó dùng phương pháp nội suy để suy ra cường độ sáng của các điểm khác trên mặt. Cụ thể hơn ở đây ta tính cường độ ánh sáng tại các nút mạng lưới dựa vào các véc tơ pháp tuyến đã tính được. Sau đó sử dụng phương pháp nội suy tuyến tính để suy ra cường độ của các đường biên của đa giác. Và cũng bằng phương pháp nội suy tuyến tính tính ra cường độ của các điểm nằm bên trong đa giác. Và cũng bằng phương pháp nội suy tuyến tính tính ra cường độ của các điểm nằm bên trong đa giác.

Ví dụ: một mặt trong mạng lưới ABCD, véc tơ pháp tuyến tại A, B, C, D đã tính được (không phải làm nội suy) và dựa vào góc tới của tia sáng sẽ tính được độ sáng tại các điểm đó, xét tỉ lệ của P chia AB thì ta dùng phương pháp nội suy tuyến tính tính được độ sáng tại P. Giả sử PA/PB =k thì



$I_P = (I_A + k*I_B)/(k+1)$ *Hình 2.4: Mạng lưới ABCD*

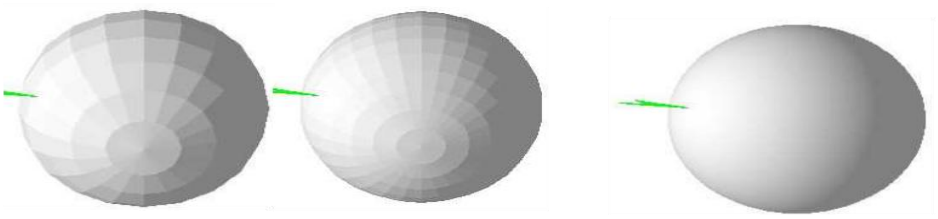
Tương tự ta có thể tính độ sáng tại Q nếu ta biết nó chia CD theo tỷ lệ nào, còn một điểm trong đa giác nằm giữa PQ cũng được nội suy như thế. Có điều chú ý ở đây là ta xét các dòng PQ theo độ phân giải của màn hình, là dòng pixel, tức là xét từ trên xuống dưới bắt đầu từ dùng vị trí thấp đến vị trí cao của toạ độ màn hình.

2.3. KỸ THUẬT TẠO BÓNG GOURAUD TRONG ĐỒ HOẠ 3D

Kỹ thuật tạo bóng Gouraud là một phương pháp vẽ bóng, tạo cho đối tượng 3D có hình dáng cong có một cái nhìn có tính thực hơn. Phương pháp này đặt cơ sở trên thực tế sau: đối với các đối tượng 3D có bề mặt cong thì người ta

thường xấp sỉ bề mặt cong của đối tượng bằng nhiều mặt đa giác phẳng, ví dụ như một mặt cầu có thể sấp sỉ bởi một tập các mặt đa giác phẳng có kích thước nhỏ sắp xếp lại, khi số đa giác xấp xỉ tăng lên (có nghĩa là diện tích mặt đa giác nhỏ lại) thì tính thực của mặt cầu sẽ tăng, sẽ cho ta cảm giác mặt cầu trông tròn trịa hơn, mịn và cong hơn. Tuy nhiên, khi số đa giác xấp xỉ một mặt cong tăng thì khối lượng tính toán và lưu trữ cũng tăng theo tỷ lệ thuận theo số mặt, điều đó dẫn đến tốc độ thực hiện sẽ trở nên chậm chạp hơn. Chúng ta hãy thử với một ví dụ sau: Để mô phỏng một mặt cầu người ta xấp xỉ nó bởi 200 mặt thì cho ta một cảm giác hơi gồ ghề, nhưng với 450 mặt thì ta thấy nó mịn và tròn trịa hơn, song khi số mặt là 16200 thì cho ta cảm giác hình cầu rất tròn và mịn. Tuy hình ảnh mặt cầu với 16200 mặt đa giác thì mịn hơn so với 200 mặt, song lượng tính toán phải thực hiện trên mỗi đa giác cũng tăng lên gấp 16200/200= 81 lần.

Song vấn đề vẫn còn nảy sinh một khi ta phóng lớn hay thu nhỏ vật thể. Nếu ta phóng lớn thì rõ ràng là các đa giác cũng được phóng lớn theo cùng tỷ lệ, dẫn đến hình ảnh về các mặt đa giác lại hiện rõ và gây ra cảm giác không được trơn mịn. Ngược lại, khi ta thu nhỏ thì nếu số đa giác xấp xỉ lớn thì sẽ dẫn đến tình trạng các đa giác quá nhỏ, chồng chất lên nhau không cần thiết.



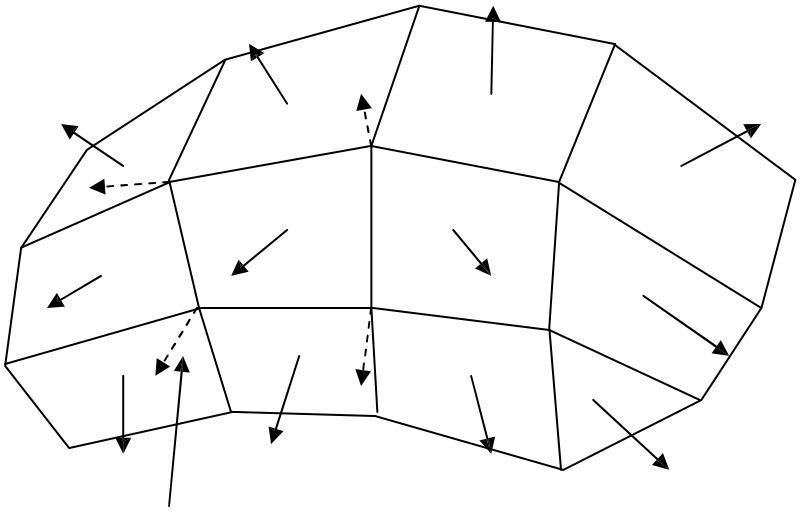
200 mặt 450 mặt 16200 mặt

Hình 2.5: Tạo bóng hình tròn

Để giải quyết vấn đề trên, chúng ta có thể tiến hành theo phương pháp tô bóng Gouraud. Mô hình bóng Gouraud tạo cho đối tượng một cái nhìn giống như là nó có nhiều mặt đa giác bằng cách vẽ mỗi mặt không chỉ với một cường độ sáng mà vẽ với nhiều cường độ sáng khác nhau trên các vùng khác nhau, làm cho mặt phẳng nom như bị cong. Bởi thực chất ta cảm nhận được độ cong của các mặt cong do hiệu ứng ánh sáng nên sẽ đón nhận và phản xạ ánh sáng khác nhau, từ đó chúng ta sẽ cảm nhận được các độ sáng khác nhau trên cùng một mặt cong.

Thường thì mỗi mặt đa giác có một vector pháp tuyến, và như phần trên đã trình bày, vector pháp tuyến đó được dùng để tính cường độ của ánh sáng phản

xạ trên bề mặt của đa giác từ đó suy ra cường độ sáng của mặt. Tuy nhiên mô hình Gouraud lại xem một đa giác không chỉ có một vector pháp tuyến, mà mỗi đỉnh của mặt đa giác lại có một vector pháp tuyến khác nhau, và từ vector pháp tuyến của các đỉnh chúng ta sẽ nội suy ra được vector pháp tuyến của từng điểm trên mặt đa giác, từ đó tính được cường độ sáng của điểm. Như thế, các điểm trên cùng một mặt của đa giác sẽ có cường độ sáng khác nhau và cho ta cảm giác mặt đa giác không phải là mặt phẳng mà là mặt cong.

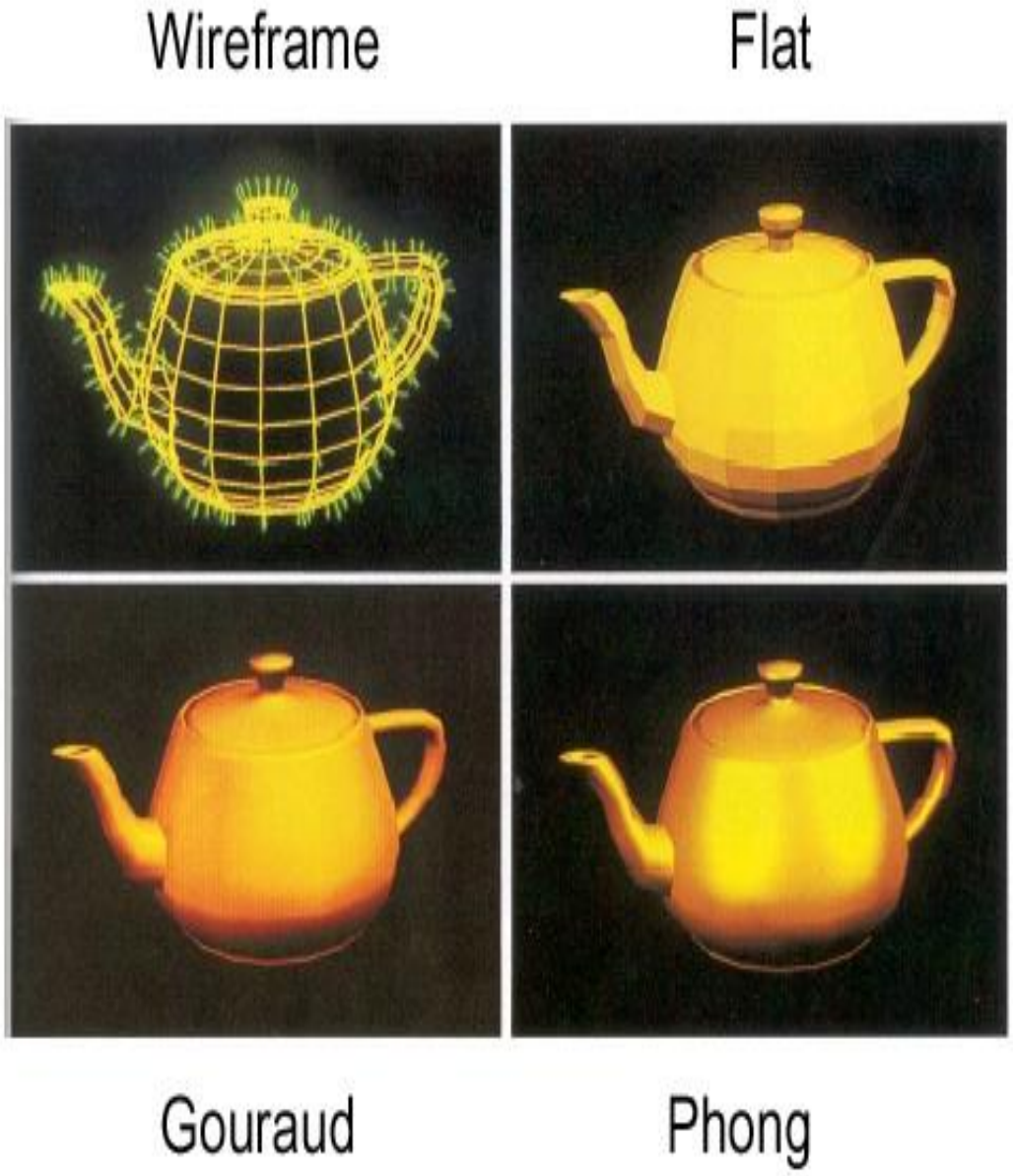


Vector trung bình cộng bằng trung bình cộng của các vector pháp tuyến lận cận

Hình 2.6. Biểu diễn đa giác nhiều mặt

Thực chất thì mỗi mặt đa giác chỉ có một vector pháp tuyến, song phương pháp Gouraud tính toán vector trung bình tại mỗi đỉnh của đa giác bằng cách: lấy trung bình cộng các vector pháp tuyến của các đa giác có chứa đỉnh đang xét.

Việc nội suy vector pháp tuyến của từng điểm trên mặt đa giác được thực hiện tương tự như việc nội suy độ sâu trong giải thuật "vùng đệm độ sâu".



Hình 2.7. So sánh với các loại tạo bóng khác

CHƯƠNG 3. CHƯƠNG TRÌNH THỬ NGHIỆM

3.1. Bài toán.

Do tính chất công việc cũng như thẩm mỹ của con người luôn luôn thay đổi để phù hợp với thực tiễn mà những đòi hỏi, yêu cầu đặt ra cho xử lý ảnh ngày càng cao, đa dạng. Theo xu hướng đó, xử lý ảnh phát triển không ngừng hướng tới quy trình xử lý ảnh hoàn thiện. Tạo bóng Gouraud là một trong những khâu quan trọng của quy trình xử lý ảnh. Việc đạt đến một công cụ toàn năng, có thể nâng cao chất lượng ảnh cũng như mọi cấu trúc ảnh vẫn là một mục tiêu xa vời. Chính vì vậy, trong lĩnh vực này vẫn còn rất nhiều cơ hội và thách thức.

Với bài toán tạo bóng Gouraud, em cài đặt chương trình thử nghiệm với Kỹ thuật tạo bóng Gouraud.

Đầu vào : Một vật thể 3D có định dạng .3D.

Đầu ra : Vật thể được tạo bóng theo phương pháp Gouraud.

3.2. Phân tích, thiết kế.

Hoạt động của chương trình :

Bước 1: Đưa một ảnh vật thể 3D có định dạng .3D để tạo bóng

Bước 2: Xử lý bằng cách thao tác.

- Cho vật thể chuyển động theo chiều kim đồng hồ.
- Cho vật thể tạm dừng chuyển động.
- Tăng độ cao của nguồn sáng.
- Giảm độ cao của nguồn sáng.
- Phóng to vật thể.
- Thu nhỏ vật thể.

Bước 3 : Áp dụng phương pháp Gouraud shading với các mô phỏng thuộc tính khác nhau

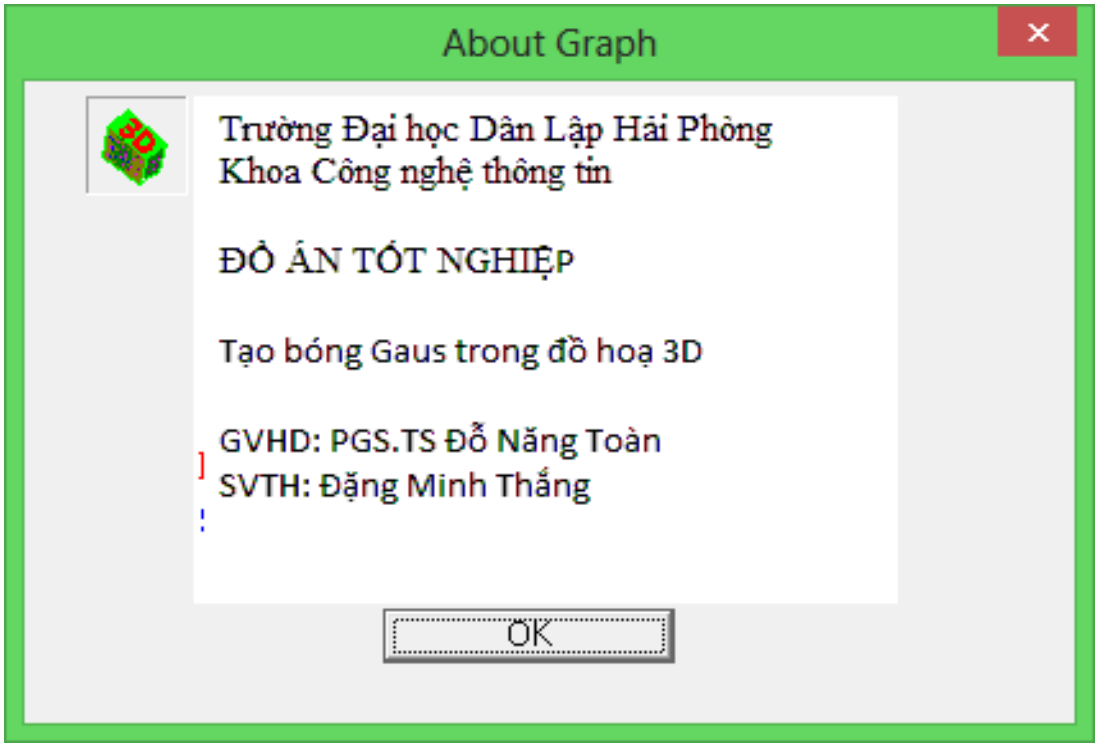
Bước 4 : Hiển thị vật thể sau khi đã được đánh bóng qua các góc nhìn khác nhau bằng phương pháp Gouraud.

3.3. Một số kết quả chương trình.

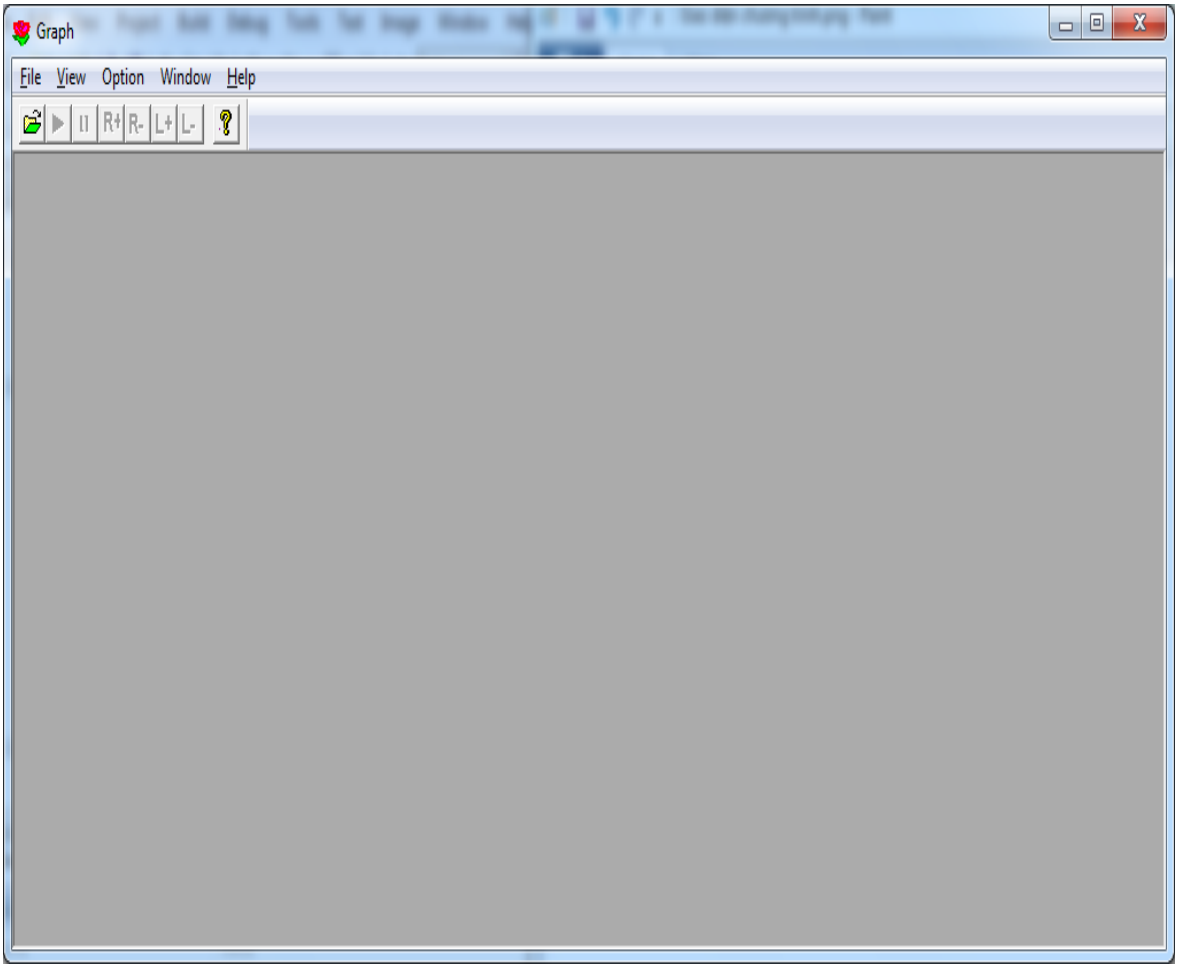
Chương trình được xây dưng bằng bộ công cụ Visual studio 2008. Chương trình thử nghiệm cài đặt kỹ thuật đánh bóng Gouraud. Một số modul chính của chương trình:

- Chọn ảnh đầu vào ảnh 3D
- Xử lý ảnh bằng kỹ thuật đánh bóng Gouraud.

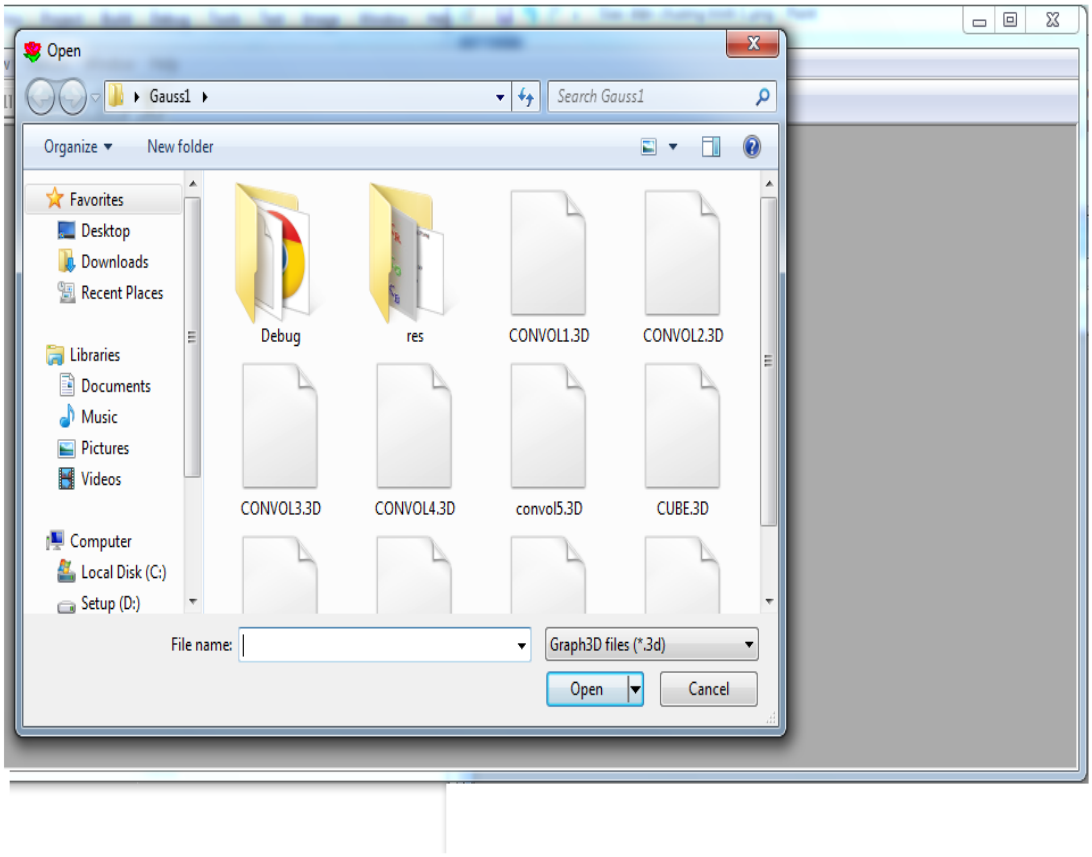
- Hiển thị ảnh kết quả.



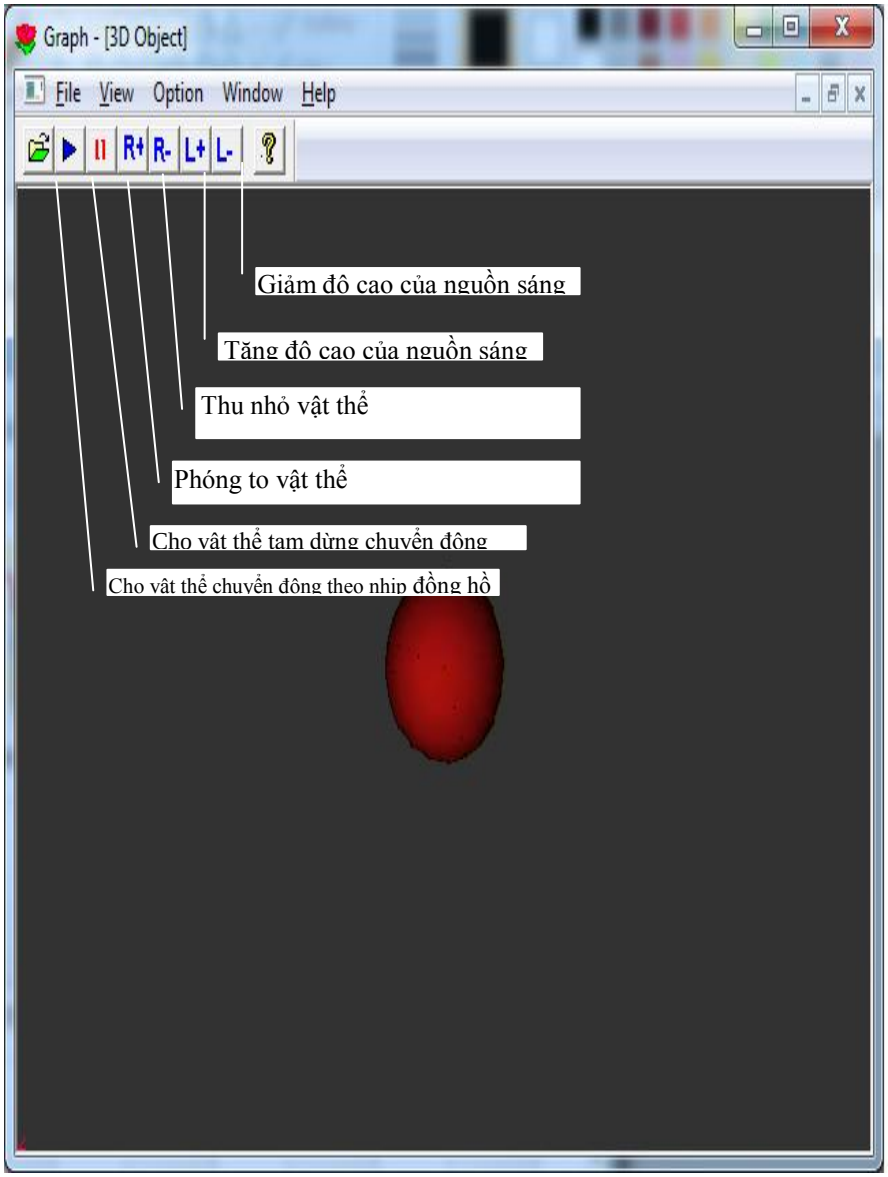
Hình 3.1. Giới thiệu chương trình



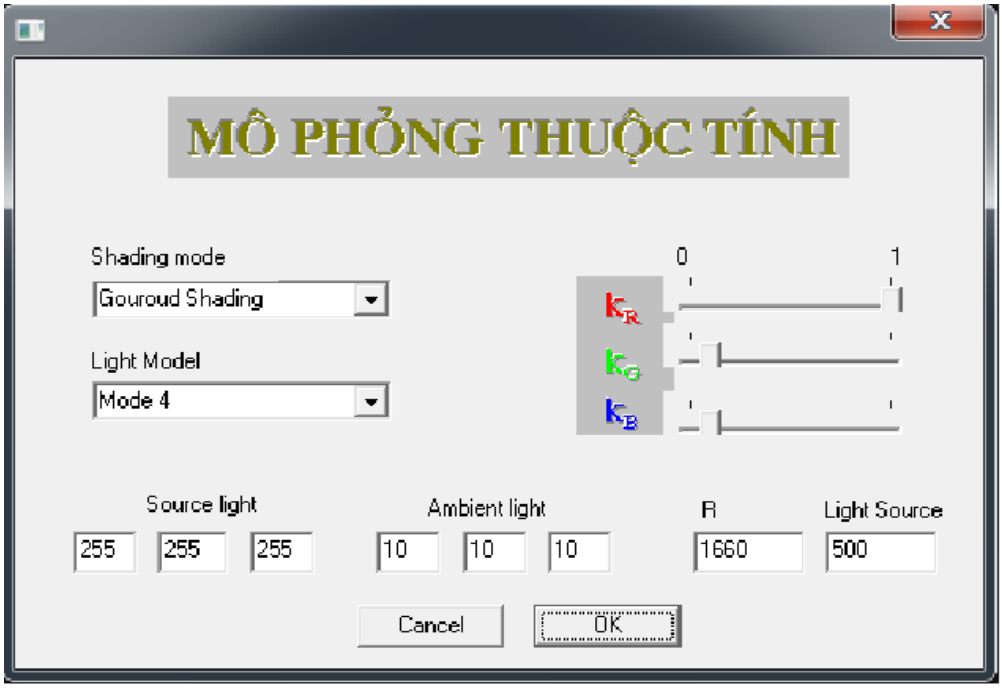
Hình 3.2. Giao diện chương trình



Hình 3.3. Các hình ảnh đầu vào



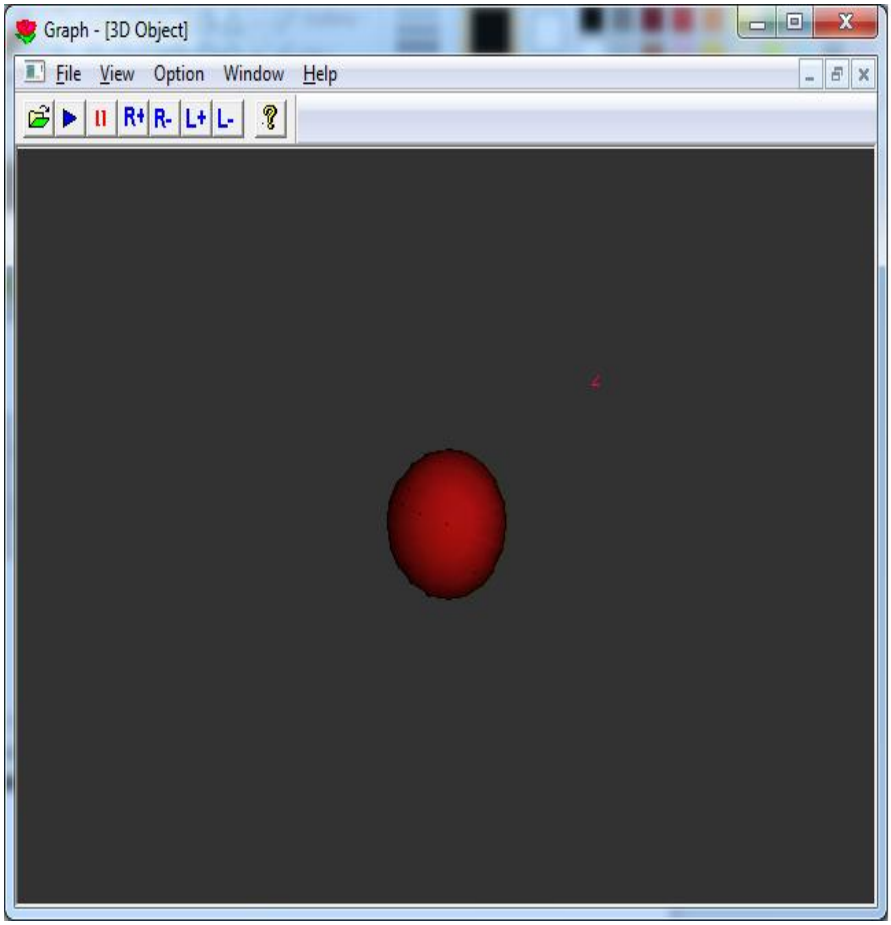
Hình 3.4. Vật thể chưa áp dụng Gouraud



Hình 3.5. Các thuộc tính mô phỏng

Dùng để thay đổi

- Cách tô bóng theo các mô hình chọn: shading mode.
- Chọn cách tính cường độ ánh sáng.
- Hệ số phản xạ đơn sắc của bề mặt vật thể.
- Cường độ ánh sáng của nguồn và của môi trường.
- Khoảng cách chiếu phối cảnh, khoảng cách từ nguồn sáng cho đến mặt phẳng ảnh.



Hình 3.5. Kết quả khi đã tạo bóng Gouraud với góc nhìn từ phía trên bên phải

PHẦN KẾT LUẬN

Trong quá trình nghiên cứu và thực hiện đồ án dưới sự định hướng dẫn của thầy hướng dẫn, đồ án đã tìm hiểu được một cách tổng quan về xử lý ảnh và bài toán tạo bóng Gouraud, và một số kỹ thuật tạo bóng Gouraud. Dựa vào những tài liệu tìm được em đã tiến hành cài đặt chương trình thử nghiệm, xây dựng thuật toán tạo bóng Gouraud. Tuy nhiên kết quả vẫn còn thiếu sót và em cảm thấy khi ứng dụng vào thực tế sẽ không đạt hiệu quả như mong muốn.

Mặc dù đã hoàn thành được mục tiêu chính của đồ án nhưng do điều kiện về thời gian có hạn mà lĩnh vực cần tìm hiểu cũng tương đối rộng nên những gì tìm hiểu được trong đồ án sẽ khó tránh khỏi những thiếu sót. Chương trình thử nghiệm cũng chưa thực sự hoàn thiện nhưng đó cũng là một kết quả khả quan. Trong thời gian tới nếu có điều kiện em sẽ tìm hiểu thêm và có thể sẽ xây dựng một chương trình thử nghiệm về các thuật toán tạo bóng hoàn chỉnh hơn.

TÀI LIỆU THAM KHẢO

Tài liệu tiếng Việt

- [1]. Dương Anh Đức, Lê Đình Huy – Chiếu sáng và tạo bóng.
- [2]. Đỗ Năng Toàn, Phạm Việt Bình (2007), *Giáo trình xử lý ảnh*, Nhà xuất bản Đại học Thái Nguyên.
- [3]. Phạm Anh Phương, Nguyễn Hữu Tài, "*Giáo trình Lý thuyết Đồ họa*", 15-09-2006

Tài liệu tiếng Anh

- [4]. Andrew V.Nealen, *"Shadow Volume and Shadow Mapping, Recent Development.*
- [5]. Ikrima Elhassan, *"Shadow Algorithms"* , 20-02-2007.