**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC THỦY LỢI**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN**

**MÔN: Đồ họa máy tính**

**Đề tài: Mô phỏng chuyển động của cánh tay ROBOT**

**Giảng viên: Th.S Trần Thị Minh Hoàn**

**Lớp: 58TH1**

**Họ và tên: Lê Xuân Huynh**

**Mã SV: 1651061098**

Hà Nội, Ngày ... tháng ... năm ...

**LỜI NÓI ĐẦU**

Hiện nay đồ họa máy tính là một trong những chương trình thông dụng nhất, nó đóng góp phần quan trọng làm cho giao tiếp con người và máy tính trở nên thân thiện. Thật vậy, giao diện kiểu văn bản đã được thay thế hoàn toàn bằng giao diện đồ họa, cùng với công nghệ đa phương tiện đã đưa ngành Công Nghệ Thông Tin sang phiên bản mới.

Và ngày nay robot đã được áp dụng rộng rãi trong hầu hết mọi lĩnh vực như sản xuất, công nghiệp, công nghệ cao như vũ trụ, hàng không, giao thông vận tải, ... Các nước phát triển đã xuất hiện xu hướng tạo ra những dây chuyền và thiết bị tự động có tính linh hoạt cao và xu hướng này đã đạt được khá nhiều kết quả khả quan. Chính vì thế việc nghiên cứu, ứng dụng và phát triển các hệ thống sản xuất tự động linh hoạt ngày càng trở lên cấp thiết, nhất là các loại robot.

Sau đây nhóm em xin “trình bày tóm tắt các kỹ thuật biến đổi hình” gồm có phép biến đổi hình 2D, phép biến đổi hình 3D, biến đổi hệ trục tọa độ của cánh tay robot đơn giản,... Cuối cùng nhóm em có cho chương trình để mô phỏng các kĩ thuật biến đổi hình.

Trong quá trình làm có nhiều thiếu sót, rất mong cô và các bạn đóng góp để giúp nhóm em cài thiện hơn nữa.

Xin chân thành cảm ơn !!

**Nội dung**

**1. Mô tả chương trình:**

*Yêu cầu của chương trình:*

* Mô phỏng cánh tay robot gồm có cánh tay, khuỷu tay, cổ tay và ngón tay.
* Mô phỏng chuyển động của cánh tay robot.

*Mô tả chức năng của chương trình:*

* Sau khi chạy chương trình, click chuột để bắt đầu việc mô phỏng chuyển động của cánh tay robot. Sẽ có nhiều bộ phận được chuyển động độc lập, và mỗi bộ phận tương ứng với một nút chuột khác nhau.
* Dùng 3 nút chuột để làm cho cánh tay robot chuyển động.
  + Nút chuột trái để chuyển động cả cánh tay.
  + Nút chuột phải để chuyển động khuỷu tay.
  + Nút chuột giữa để chuyển động các ngón tay.

*Phương pháp nghiên cứu:*

* Mục đích nghiên cứu: Thực hành vẽ các đối tượng đồ họa 3D đã học trên lớp và một số kiến thức thu được trong quá trình học tập và nghiên cứu.
* Môi trường và ngôn ngữ lập trình: Chương trình được chạy trên môi trường Windows, ngôn ngữ lập trình là C++ và thư viện OpenGL.

*Mô tả chi tiết:*

Bước 1: Khởi tạo

* Tạo khung nhìn Form gồm : Chế độ nhập, Chế độ nhìn, Chế độ màu, Form size, Vị trí Form , Tiêu đề Form.
* Các biến là các góc quay tạo các khớp của cánh tay và ngón tay.
* Khởi tạo Chế độ nhập Init().

Bước 2: Chức năng xử lý

* Dùng hàm Display() để xử lý vẽ hình, tô màu, quay hình.
* Khởi tạo màu nền và chế độ buffers.
* Xử lí xoay hình theo bàn phím key.
* Dùng hàm Reshape() để xử lý thay đổi kích thước khung Form.
* Dùng hàm Keyboard() để xử lý bàn phím máy tính.
* Dùng hàm Mouse() để xử lý chuột máy tính.
* Dùng hàm MainLoop() để dừng màn hình hiển thị.

**2. Mô tả cài đặt các hàm chính của chương trình:**

* Hàm **Draw**:
* Các phương thức chính:
  + ***glPushMatrix:*** Dùng để tạo ma trận và đẩy lên trên cùng.
  + ***glPopMatrix:*** Dùng để xóa ma trận.
  + ***glRotatef:*** Dùng để quay cánh tay một góc angle ngược chiều kim đồng hồ xung quanh tia nối từ gốc tọa độ đến điểm (x,y,z).
  + ***glTranslatef:*** Dùng để tịnh tiến đối tượng di chuyển theo (x,y,z) tương ứng lần lượt với ba trục.
  + ***glScalef:*** Dùng để biến đổi tỉ lệ mô hình với hệ số tỉ lệ tương ứng với ba trục Ox, Oy, Oz lần lượt là x,y,z.
  + ***glutWireCube:*** Dùng để vẽ hình lập phương khung (ở đây là cánh tay).
* Hàm **Mouse**:
* Các phương thức chính:
* ***switch/case:*** Dùng để kiểm tra các giá trị. Mỗi giá trị là một **case - trường hợp**.
* Hàm **Keyboard**:
  + Các phương thức chính:
    - ***if:*** Dùng để thực hiện các lệnh nếu điều kiện được kiểm tra là **True**.
* Hàm **Display**:
  + Các phương thức chính:
    - ***glClear:*** Dùng để xóa pixel.
    - ***glColor3f:*** Dùng để thiết lập màu vẽ.
    - ***glLoadIdentity:*** Dùng để reset camera.
    - ***gluLookAt:*** Định vị camera tại vị trí chỉ định.
    - ***glFlush:*** Thực hiện tất cả các hàm chưa được chạy trước tới khi kết thúc.
* Hàm **Reshape**:
  + Các phương thức chính:
    - ***glViewport:*** Thiết lập chế độ xem.
    - ***glMatrixMode:*** Chỉ định ma trận nào là ma trận hiện tại.
    - ***gluPerspective:*** Thiết lập ma trận chiếu phối cảnh.
* Hàm **init**:
  + Các phương thức chính:
    - ***glClearColor:*** Chọn màu nền.
    - ***glShadeModel:*** Chọn bóng phẳng hoặc mịn.
* Hàm **main**:
  + Các phương thức chính:
    - ***glutInit:*** Khởi tạo thư viện.
    - ***glutInitDisplayMode:*** Thiết lập chế độ hiển thị ban đầu.
    - ***glutInitWindowSize:*** Kích thước cửa sổ
    - ***glutInitWindowPosition:*** Vị trí cửa sổ
    - ***glutCreateWindow:*** Tạo cửa sổ
    - ***glutDisplayFunc:*** Gọi hàm Display
    - ***glutReshapeFunc:*** Gọi hàm Reshape
    - ***glutKeyboardFunc:*** Kích hoạt sự kiện bàn phím
    - ***glutMouseFunc:*** Kích hoạt sự kiện chuột
    - ***glutMainLoop:*** Hiển thị

**3.Cấu trúc dữ liệu và các hàm trong chương trình:**

**3.1 Cấu trúc dữ liệu:**

Các hoạt động cơ bản trên ngăn xếp có thể liên quan tới việc khởi tạo ngăn xếp, sử dụng nó và sau đó xóa nó. Ngoài các hoạt động này, một ngăn xếp có hai hoạt động nguyên sơ liên quan tới khái niệm, đó là:

Hoạt động **Push():** lưu trữ một phần tử trên ngăn xếp.

Hoạt động **Pop():** xóa một phân tử từ ngăn xếp.

**3.2 Các hàm trong chương trình:**

**3.2.1 Hàm vẽ cánh tay: void** Draw()

**-** *Chức năng của hàm là*: vẽ các khối lập phương tương ứng với cánh tay và các ngón tay.

**-** *Phân tích*:Đầu tiên, chúng ta sẽ vẽ khối lập phương thứ nhất. Tạo ma trận 1 trên cùng bằng glPushMatrix() sau đó áp dụng các phép biến đổi đối tượng cho ma trận 1. Rồi dùng hàm glWireCube() để vẽ khối lập phương với tham số truyền vào là kích thước của khối lập phương. Kết thúc khối lập phương thứ nhất ta nhân bản ma trận 1 để tạo ma trận 2 và đẩy nó lên trên cùng. Sử dụng các hàm biến đổi cho ma trận 2 ( không ảnh hưởng đến ma trận 1). Sau đó dùng glWireCube để vẽ khối lập phương thứ 2. Tiếp theo là xóa ma trận 2, ma trận 1 sẽ trở về trên cùng bằng hàm glPopMatrix(). Rồi xóa tiếp ma trận 1. Áp dụng phân tích này để vẽ cánh tay và các ngón tay bằng các khối lập phương.

**3.2.2 Hàm xử lí sự kiện chuột: void** Mouse(**int** button, **int** state, **int** x, **int** y)

- *Chức năng của hàm là*: kiểm tra thao tác xử lí chuột.

- *Phân tích:*

+ Khi nhấn chuột trái: quay khối lập phương thứ nhất (cánh tay). Dùng “*GLUT\_LEFT\_BUTTON*” để bắt sự kiện chuột trái, nếu điều kiện ‘*state == GLUT\_DOWN’* là **True** thì thực hiện câu lệnh bên trong.

+ Khi nhấn chuột giữa: quay các ngón tay. Dùng “*GLUT\_MIDDLE\_BUTTON*” để bắt sự kiện chuột giữa, nếu điều kiện ‘*state == GLUT\_DOWN’* là **True** thì thực hiện câu lệnh bên trong.

+ Khi nhấn chuột phải: quay khối lập phương thứ hai (khuỷu tay). Dùng “*GLUT\_RIGHT\_BUTTON*” để bắt sự kiện chuột phải, nếu điều kiện ‘*state == GLUT\_DOWN’* là **True** thì thực hiện câu lệnh bên trong.

**3.2.3 Hàm xứ lí sự kiện bàn phím:** **void** Keyboard(**unsigned** **char** key, **int** x, **int** y)

**-** *Chức năng của hàm là:* kiểm tra thao tác xử lí bàn phím.

**-** *Phân tích*: Khi nhấn phím ‘*s*’ thì sẽ quay ngón tay của robot. Dùng “*if (key == ‘s’)*” để bắt sự kiện bàn phím, nếu điều kiện là **True** thì thực hiện câu lệnh bên trong.

**3.2.4 Hàm hiển thị: void** Display (**void**)

- *Chức năng cùa hàm là:* Hàm thực hiện các thao tác vẽ theo yêu cầu của chương trình.

- *Phân tích:* Thiết lập các thuộc tính của thao tác vẽ như thiết lập màu vẽ, xóa pixel, thiết lập điểm nhìn, gọi vào hàm Draw() để vẽ và cuối cùng là thực hiện quá trình đẩy ra buffer.

**3.2.5 Hàm đáp ứng của cửa sổ:** **void** reshape (**int** w, **int** h)

- *Chức năng của hàm là*: Thực hiện các thao tác cần làm khi cửa sổ bị thay đổi kích thước.

- *Phân tích:* Hàm này sẽ được gọi bất cứ khi nào kích thước hoặc hình dạng của cửa sổ ứng dụng bị thay đổi. Hàm *reshape* lấy hai tham số, đó là chiều cao và chiều rộng của cửa sổ. Những tham số này chủ yếu được sử dụng để thiết lập viewport mới.

**3.2.6 Hàm thực hiện các khởi tạo:** **void** init (**void**)

- *Chức năng của hàm là:* Hàm khởi tạo các giá trị của các đối tượng và biến.

- *Phân tích:* Khởi tạo màu nền cho cửa sổ bằng *glClearColor()*, tô bóng phẳng hoặc mịn bằng *glShadeModel()* (có 2 giá trị là GL\_SMOOTH và GL\_FLAT).

**3.2.7 Hàm main của chương trình**: **int** main(**int** argc, **char**\*\* argv)

- *Chức năng của hàm là:* Dùng để chạy chương trình và gọi các hàm có trong chương trình.

- *Phân tích:*

*+* Hàm main là nơi chứa những đoạn code sẽ được chạy đầu tiên, khi biên dịch chương trình sẽ chạy hàm glutInit để khởi tạo thư viện GLUT, tạo cửa sổ, vị trí cửa sổ, gọi vào các hàm bên trên và cuối cùng là trả về giá trị return.

*+* Hai tham số của hàm được sử dụng với glutInit, nó sẽ khởi tạo thư viện GLUT, trong tiến trình này glutInit có thể chấm dứt chương trình GLUT nếu có 1 lỗi thông báo cho người dùng rằng GLUT không thể khởi tạo đúng cách. Ví dụ như lỗi thất bại trong việc kết nối với hệ thống cửa sổ, hệ thống không hỗ trợ cho OPENGL hoặc các câu lệnh không hợp lệ.

3.2.8 Hàm thay đổi màu: **void** SetColor(GLfloat mat\_diffuse[4])

- Chức năng của hàm là: Tạo màu cho khối lập phương.

- Phân tích:

+ Thiết lập màu mặc định là màu đen (ambient) và diffuse, biến color truyền vào là specular.

+ GL\_FRONT\_AND\_BACK có tác dụng gán màu cho mặt trước và mặt sau của khối lập phương.

+ Khởi tạo màu vàng *Glfloat yellow* ở hàm Draw() và gán màu cho các khối lập phương bằng *setColor(yellow).*

**3.3 Thuật toán:**

Trong mã nguồn có áp dụng một số thuật toán mà trong đó thuật toán chính là ma trận ngăn xếp. Một ngăn xếp được gọi là một stack. Thêm một ma trận vào ngăn xếp (glPushMatrix), bật một ma trận ra khỏi ngăn xếp (glPopMatrix). Ma trận được dùng kết hợp với nhau để vẽ các khối lập phương tạo hình cho cánh tay.

*Ví dụ:*

glPushMatrix();

glTranslatef(1.5, 0.0, 0.0);

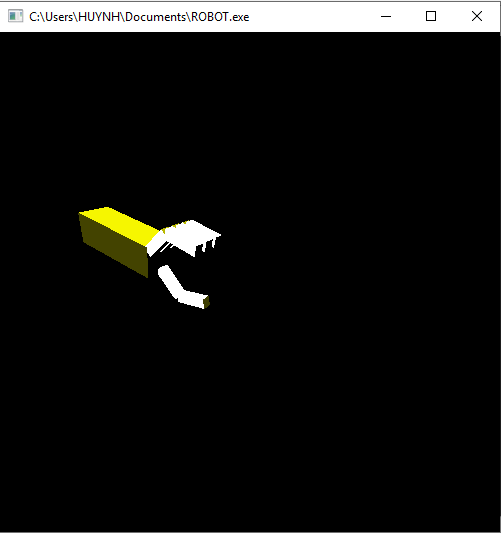
glScalef(3.0, 1.0, 1.0);

glutWireCube (1.0);

glPopMatrix();

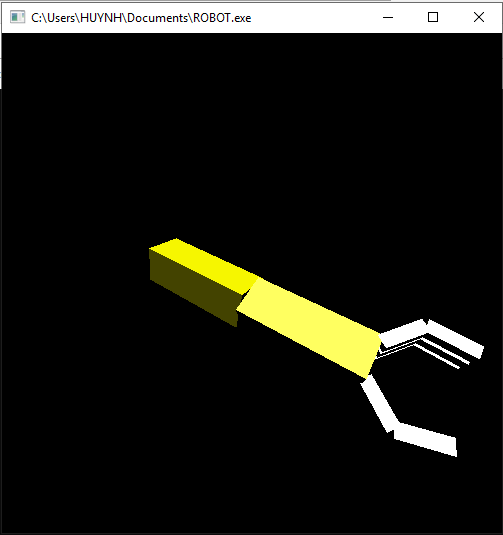
* Bước 1: Lưu ma trận hiện hành.
* Bước 2: Tịnh tiến đến vị trí chỉ định.
* Bước 3: Biến đổi tỉ lệ mô hình.
* Bước 4: Vẽ khối lập phương.
* Bước 5: Phục hồi lại ma trận hiện hành cũ.

**4. Demo chương trình:**

****

**Hình 1.** Cánh tay Robot mặc định

Khi chạy chương trình sẽ có mô hình cánh tay xuất hiện, ta có thể dùng các nút chuột và bàn phím để thay đổi chuyển động của cánh tay và các ngón tay.Nút chuột trái dùng để quay cả cánh tay theo chiều ngược kim đồng hồ. Nút chuột phải dùng để quay khuỷu tay theo chiều ngược kim đồng hồ. Nút chuột giữa và bàn phím dùng để quay các ngón tay theo chiều ngược kim đồng hồ. Kết quả ta được:



****



***Hình 2, 3, 4.*** Cánh tay Robot chuyển động

**5. Kết luận:**

Qua việc thực hiện nghiên cứu đề tài “Mô phỏng chuyển động cánh tay Robot ở dạng 3D”. Nhóm em đã được biết thêm rất nhiều về môn đồ họa máy tính. Về các câu lệnh, các thuật toán, cấu trúc dữ liệu, các thư viện OpenGL. Bên cạnh đó rèn luyện tư duy và kiến thức để làm các dự án từ nhỏ đến lớn khác.

Trong quá trình thực hiện đề tài có rất nhiều ý tưởng hay, độc đáo. Nhưng do kiến thức còn hạn hẹp và thời gian không cho phép nên em chưa thể thực hiện được ý tưởng đó. Tuy nhiên em đã cố gắng để xây dựng một chương trình hoàn chỉnh nhất và đẹp nhất dựa theo năng lực của mình. Trong quá trình xây dựng chương trình khó tránh khỏi những sai xót. Vì vậy rất mong rằng thầy và các bạn cùng góp ý để nhóm em có thể hoàn thành sản phẩm một cách hoàn chỉnh nhất.

Em xin chân thành cảm ơn !