ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN CUỐI K Ỳ

Môn: Đồ họa máy tính

Đề tài:
XÂY DỰNG ỨNG DỤNG 3D TĨNH DỰA VÀO OPENGL TRÊN WINDOWS

Nhóm thực hiện
Graphics4.0

Giảng viên hướng dẫn
PGS.TS Lý Quốc Ngọc

BÁO CÁO ĐỒ ÁN CUỐI K Ỳ

Môn: Đồ họa máy tính

Đề tài:
XÂY DỰNG ỨNG DỤNG 3D TĨNH DỰA VÀO OPENGL TRÊN WINDOWS

DANH SÁCH THÀNH VIÊN

Họ và tên	MSSV
Nguyễn Gia Thuận	1712174
Nguyễn Phục Dược	1712372
Đào Khánh Duy	1712380
Nguyễn Quí Em	1712399
Nguyễn Việt Hoàng	1712459

Ngày 25 tháng 12 năm 2019

MỤC LỤC

NỘI DUNG

1. Giới thiệu

1.1. Sơ lược về Đồ họa máy tính

Đồ họa máy tính là một lĩnh vực của khoa học máy tính nghiên cứu về cơ sở toán học, các thuật toán cũng như các kĩ thuật để cho phép tạo, hiển thị và điều khiển hình ảnh trên màn hình máy tính. Đồ họa máy tính có liên quan ít nhiều đến một số lĩnh vực như đại số, hình học giải tích, hình học họa hình, quang học,... và kĩ thuật máy tính, đặc biệt là chế tạo phần cứng (các loại màn hình, các thiết bị xuất, nhập, các vỉ mạch đồ họa...).[1]

1.2. Đặt vấn đề

- Mô phỏng thế giới thực ngày càng trở nên quan trọng hơn trong thời đại mà cuộc cách mạng công nghiệp 4.0 đang bùng nổ mạnh mẽ, trong mọi lĩnh vực từ công nghiệp đến giải trí.

- Cần có một công cụ đủ mạnh và tiện lợi giúp người lập trình tạo ra các ứng dụng đồ họa.

1.3. Mục tiêu

- Tạo một ứng dụng mô phỏng các hình 3D cơ bản như hình hộp chữ nhật, hình lập phương, hình chóp, hình lăng trụ,…

- Có thể sử dụng để minh họa cho việc giảng dạy môn hình học, giúp học sinh dễ dàng nhận biết, hình dung các đối tượng vốn khó tưởng tượng nếu chỉ nhìn qua giấy phẳng.

1.4. Công cụ sử dụng và môi trường lập trình

- Sử dụng thư viện SharpGL, một thư viện được tạo ra dựa trên OpenGL nhưng thiết kế riêng cho ngôn ngữ C# giúp tiện lợi hơn cho việc lập trình giao diện.

- Môi trường lập trình: Chương trình được viết bằng Visual Studio 2017 và chạy thực thi trên môi trường Windows.

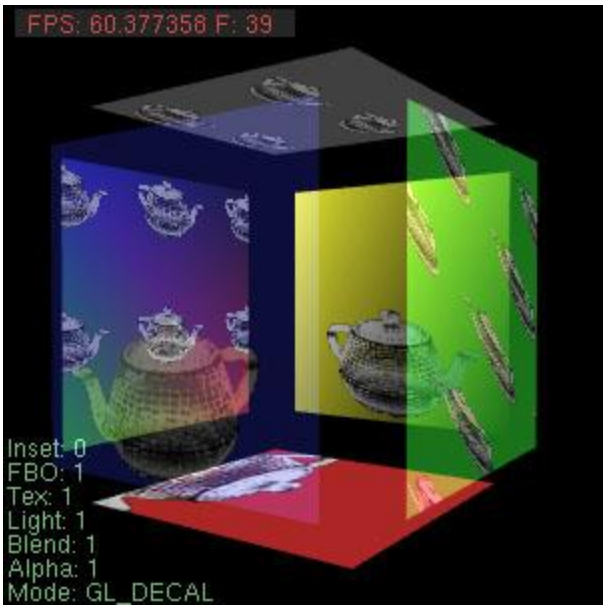
2. Phương pháp thực hiện

2.1. Giai đoạn tìm hiểu

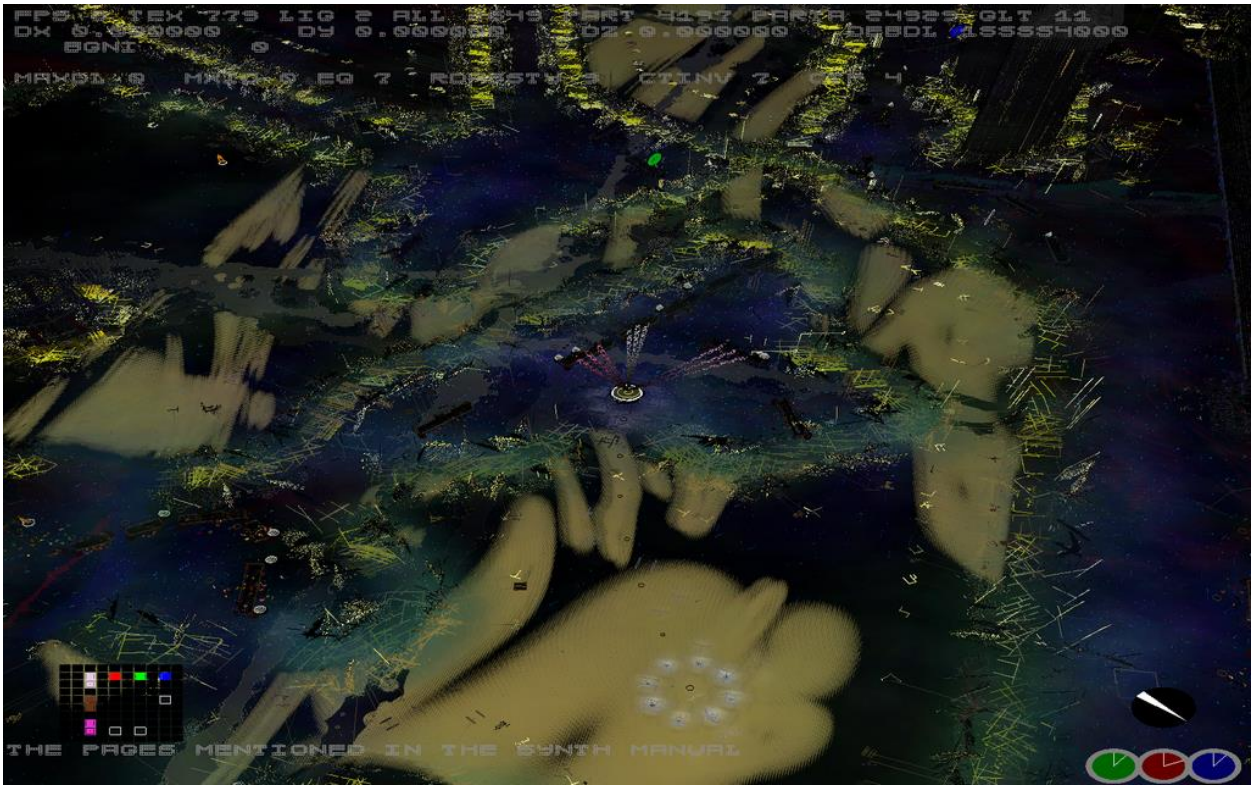
- Tìm hiểu tổng quan về lập trình 3D, các bộ thư viện nổi tiếng được sử dụng nhiều trong giới lập trình như: OpenGL, DirectX,… Nhóm quyết định chọn SharpGL (xây dựng từ OpenGL) là thư viện được sử dụng trong đồ án.

- Tìm hiểu qua các ứng dụng thực tế được viết từ OpenGL, SharpGL về mặt tính năng mà nó làm được: Mô phỏng hình ảnh khúc xạ, mô phỏng kết cấu di động, thiết kế game engine, render 3D,…

Khúc xạ dùng OpenGL [2]



Kết cấu di động [3]



Game [4]

- Tìm một số mã nguồn từ cộng đồng github liên quan đến các tác vụ 3D.

2.2. Giai đoạn thực hiện

- Nghiên cứu các đối tượng hình học 3D cơ bản: Hình hộp chữ nhật, hình lập phương, hình chóp, hình lăng trụ,… về các tính chất toán học của nó.

- Tìm hiểu, nghiên cứu về các phép biến đổi affine cơ bản lên các đối tượng hình học: Translate, Rotate, Scale.

- Tìm hiểu về cách biểu diễn trục tọa độ, gốc đặt đối tượng, khung nhìn,…

- Tìm hiểu cách cài đặt và sử dụng cơ bản SharpGL trong Visual Studio.

3. Chi tiết đồ án

3.1. Tổng quan về OpenGL và SharpGL

- OpenGL:

Là thư viện gồm các hàm API cơ bản dùng để biểu diễn các đối tượng đồ họa 2D, 3D. OpenGL mang tính độc lập đối với ngôn ngữ lập trình và sử dụng được trên nhiều hệ thống.

OpenGL có tính năng che giấu được sự tương tác phức tạp trong không gian 3D bằng cách đưa ra một giao diện lập trình thống nhất, che giấu sự khác biệt giữa các phần cứng 3D bằng cách bắt buộc các phần cứng tương thích OpenGL phải hỗ trợ tất cả các chức năng giao diện mà nó cung cấp.

OpenGL được tạo từ nền tảng toán học từ cơ bản đến nâng cao. Nó hoạt động như một hệ thống state machine. Ta có thể hình dùng cơ chế hoạt động của nó thông qua một bến cảng container trong đó các container là các đối tượng trong OpenGL, các kiện hàng trong container là những thứ chúng ta tạo ra trong ứng dụng và lúc này OpenGL API chính là các cần trục giúp chúng ta "giao tiếp" với các thùng hàng này.

Một số hàm và kiểu dữ liệu trong OpenGL [5]
glBegin, glEnd: Bắt đầu và kết thúc một tập các lệnh nào đó ở giữa 2 lệnh này)
glClear: Xóa bộ đệm,…
glCopyPixel: Sao chép pixel.
....
GLbyte, GLshort, GLint, GLfloat, GLdouble, GLboolean.

3.2. Ứng dụng vẽ hình 3D cơ bản [6]

Tổ chức mã nguồn theo hướng đối tượng với các lớp thực hiện một hoặc nhiều chức năng nào đó cho công tác chính là vẽ 3D. Trong đó quan trọng nhất là lớp Object với các lớp con kế thừa từ lớp Object là: Cube, Cylinder, Pyramid. Mỗi

lớp này đều có các trường để lưu các thông số cần thiết như: Danh sách đỉnh, màu, chiều dài cạnh, tọa độ tâm, tên, hệ số cho các phép affine,…

Chi tiết một số lớp được cài đặt:

- **Lớp Object**: Là abstract class, gồm 1 list các vertex để lưu lại các hình đã vẽ.

- **Lớp Draw**: Gồm 1 list các object, đầu tiên vẽ thì tạo 1 menu. Mỗi lần nhấn vào 1 hình (cube = 1, pyramid = 2, cylinder = 3) thì biến chooseIcon sẽ kiểm tra nó có giá trị bao nhiêu để vẽ ra đúng hình đó.

+ Hàm DrawShape để vẽ mỗi đối tượng ra màn hình.

- **Lớp Cube**:

- Phương thức khởi tạo Cube(): Thiết lập các thông số ban đầu về độ dày nét vẽ, màu đối tượng, độ chênh lệch tọa độ tâm để dịch các đỉnh, danh sách các đỉnh, các hệ số của affine,..
- Phương thức Save(): Lưu các đỉnh vào list.
- Phương thức Draw(): Thực hiện vẽ. Đầu tiên gọi hàm Save() để lấy danh sách đỉnh, gọi hàm gl.PushMatrix() để push ma trận nhằm thực hiện các phép affine, gán màu thông qua hàm gl.Color(), gọi hàm DrawRaw() (sẽ trình bày phía dưới) và DrawBorder() để vẽ viền, pop ma trận và hiển thị các dữ liệu bằng gl.Flush().
- Phương thức DrawRaw(): Thực hiện vẽ lưới mặt phẳng.
- Phương thức drawBorder(): Kiểm tra biến solid (biến này là thành phần của mỗi class, dùng để kiểm tra xem đối tượng có đang được chọn hay không) nếu là true thì đổi màu viền thành màu cam đậm, ngược lại viền đen nhạt. Sau đó vẽ các cạnh theo các cặp đỉnh nhờ vào tham số OpenGL.GL_LINES và hàm gl.Vertex().

- **Lớp Cylinder**:

- Các phương thức tương tự như lớp Cube nhưng có sự khác biệt trong cài đặt vì mỗi hình có mỗi tính chất toán học khác nhau.
- Có thêm các trường R_bot để tính bán kính, alpha để tính góc ở đỉnh và center để xác định tâm.

- **Lớp Pyramid**: Có các phương như lớp Cube, khác biệt ở cách cài đặt cho phù hợp.

- **Lớp Vertex**:

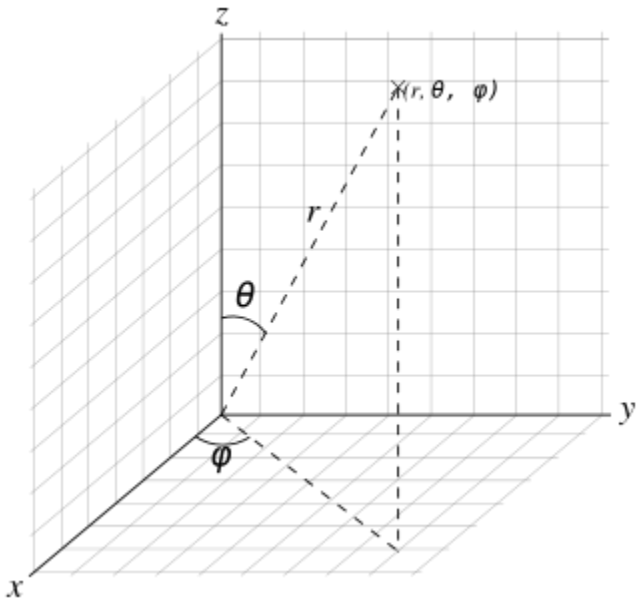
+ Các trường x, y, z dùng để lưu tọa độ đỉnh.

+ Các phương thức:

- Vertex(): Khởi tạo giá trị 3 tọa đồ đều bằng 0.
- Vertex(a, _length, c): Cập nhật đỉnh khi người dùng thay đổi chiều rộng đáy.
- Vertex(Vertex A): Tạo lớp đỉnh từ giá trị của lớp đỉnh khác.

Lưu ý: Trong source code tụi em đã đổi vị trí của trục Oz và Oy lại nên công thức trong code sẽ bị thay đổi các giá trị góc.

- **Lớp Camera**: Lớp dùng tạo các góc nhìn khác nhau cho người xem, có thể nhìn gần, xa, xoay trái, phải, xoay phía trên và xuống dưới.



Gồm: eyeX, eyeY, eyeZ là các điểm nhìn ta muốn xem.

lookX,lookY,lookZ là điểm nhìn hiện tại.

Góc Phi, theta như hình vẽ trên.

Ta coi góc nhìn camera là 1 mặt cầu ngoại tiếp với hình ở trên mặt phẳng 3 chiều.

Hàm ComputePhi và ComputeTheta để tính số đo góc Phi và Theta hiện tại để phục vụ cho các thao tác xoay phải. Cụ thể:

Góc phi là góc của Ox và điểm chiếu của điểm nhìn lên mặt Oxy.

Góc Theta là góc của bán kính từ gốc đến điểm nhìn và trục Oz.

Khi ta muốn zoom in hoặc zoom out tức là ta đang tăng hoặc giảm bán kính khối cầu.

+ Muốn xoay phải (trái) tức tăng (giảm) giá trị góc phi và ta tính lại giá trị của eyeX, eyeY theo lượng giác (do không thay đổi lên xuống nên giá trị eyeZ giữ nguyên).

+ Muốn di chuyển lên (xuống) tức giảm (tăng) giá trị góc theta và ta tính lại giá trị eyeX, eyeY, eyeZ theo lượng giác.

- **Lớp Viewport**: Dùng phép chiếu phối cảnh, ở đây là vẽ ra 1 mặt phẳng lưới 3D:

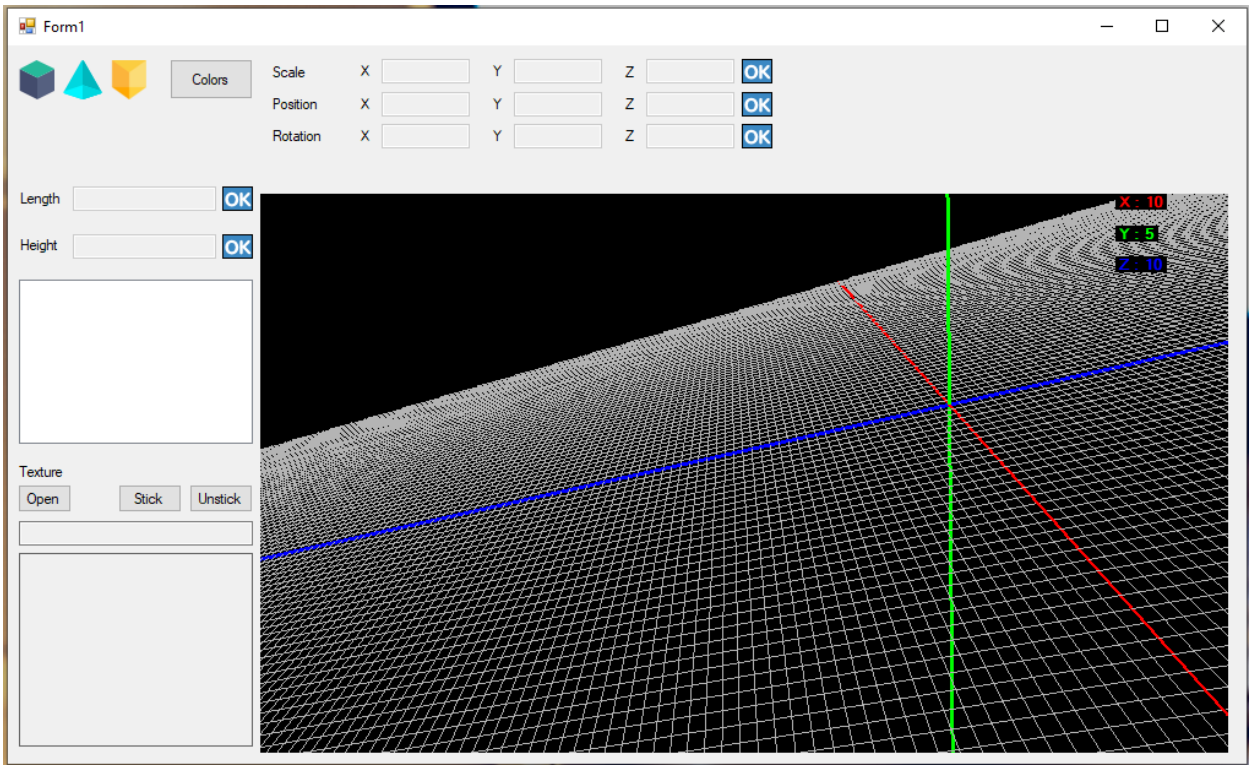
+ Đầu tiên khởi tạo gốc tọa độ O, tô màu các trục Ox, Oy, Oz.

+ Sau đó dùng vòng loop để khởi tạo các điểm kề đối xứng với tọa độ là -size, 0, i và size, 0, i đối xứng qua Oz cùng với i, 0, -2*size và i, 0, -2*size đối xứng nhau qua Ox.

4. Các kết quả đạt được

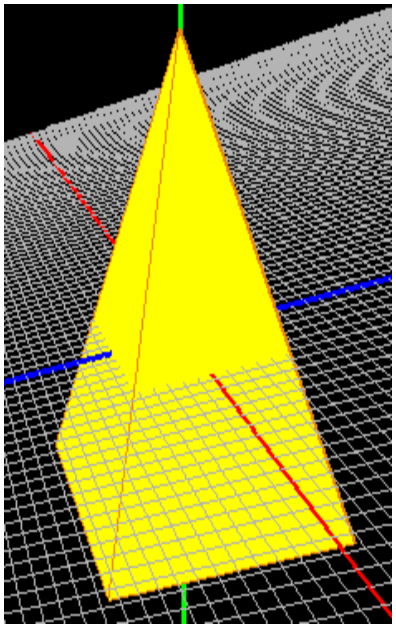
Tạo được chương trình vẽ các hình 3D cơ bản hỗ trợ giao diện người dùng:

- Có các nút chọn 1 trong 3 hình để vẽ.
- Có nút chọn màu để thay đổi màu của đối tượng.
- Khung nhập giá trị x, y, z cho các phép affine.
- Khung nhập thay đổi chiều cao, chiều rộng của đối tượng.
- Khung hiển thị tên đối tượng sau khi tạo, cho phép chọn đối tượng để thao tác.
- Hiển trị các trục tọa độ và lưới mô tả mặt phẳng giúp dễ dàng hình dung.

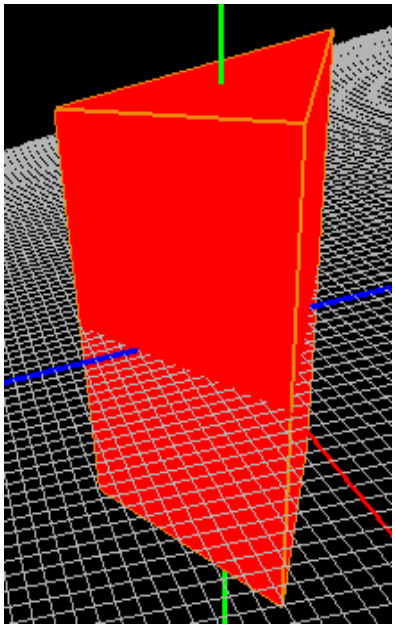


Giao diện chương trình

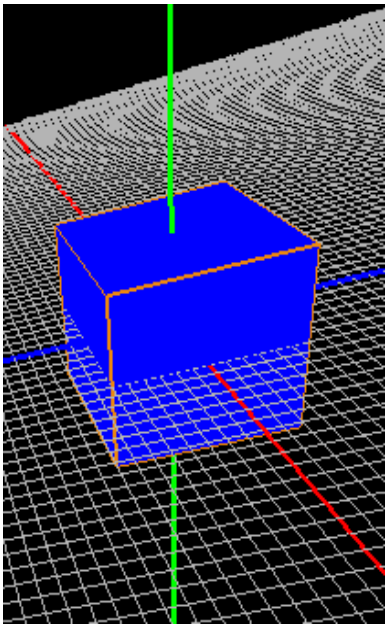
Các khối cơ bản (đã được scale)



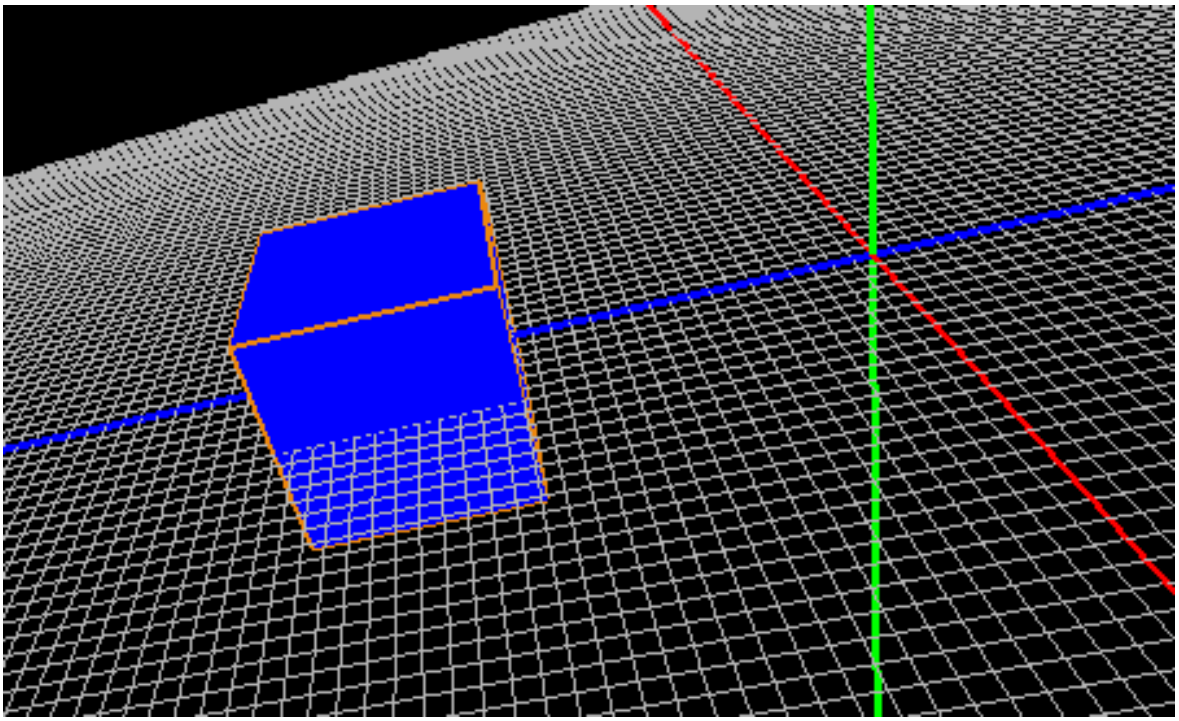
Khối chóp



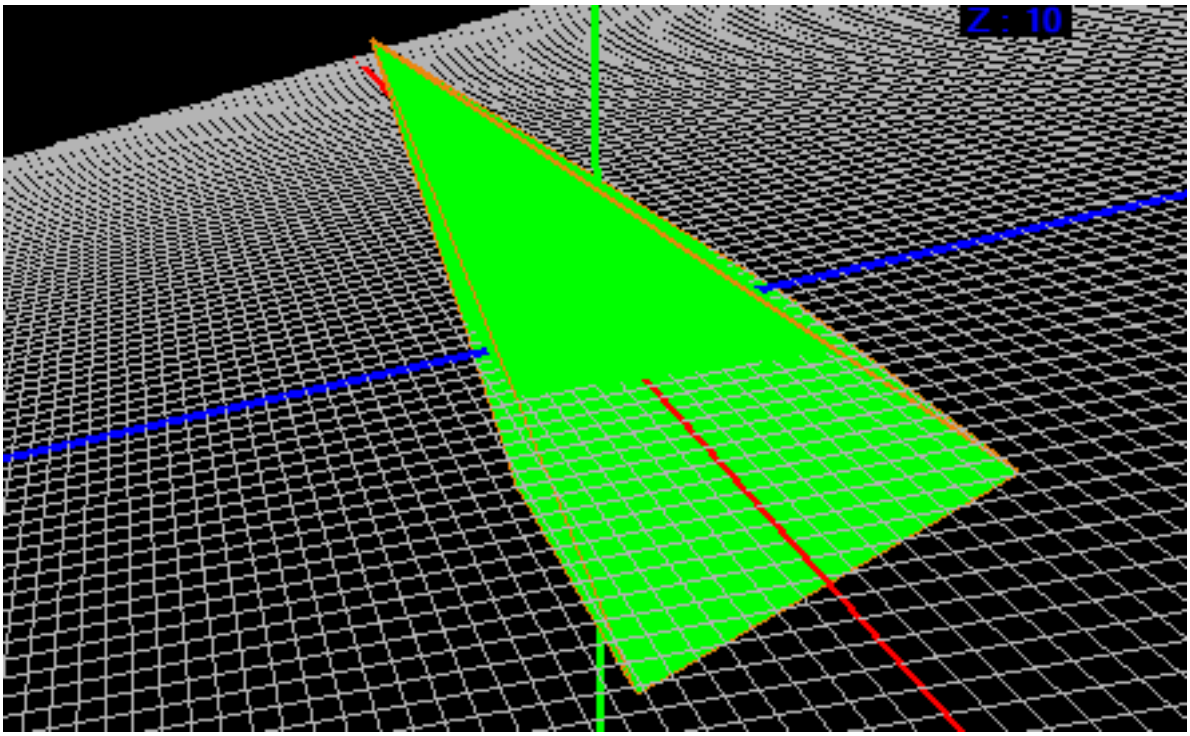
Khối lăng trụ



Khối lập phương



Translate



Rotate

Video demo: https://youtu.be/-IKjy6hARq4

5. Đánh giá đồ án

- Ưu điểm:

+ Nắm được các kiến thức cơ bản về OpenGL.

+ Sử dụng được thư viện SharpGL để lập trình.

+ Tạo được ứng dụng ở mức cơ bản, với các thao tác như vẽ các khối 3D đơn giản, thay đổi màu đối tượng, thực hiện các phép biến đổi affine lên đối tượng, hiển thị lưới tọa độ & trục tọa độ giúp mô phỏng không gian, có các nút chọn hình, ô nhập tọa độ,…

- Hạn chế:

+ Việc tương tác với đối tượng còn nhiều khó khăn do phải thao tác với khung nhập chức năng chứ chưa thể thao tác trực tiếp bằng chuột lên đối tượng.

+ Các hình vẽ được chưa đa dạng, chỉ là các khối cơ bản.

+ Chưa làm được chức năng texture.

- Cải tiến trong tương lai:

    + Cho phép người dùng tương tác trực tiếp với đối tượng bằng chuột.

    + Thêm một số khối 3D như khối cầu,…

    + Bổ sung tính năng texture.

6. Tài liệu tham khảo

[1] Đồ họa máy tính – Wikipedia
https://vi.wikipedia.org/wiki/%C4%90%E1%BB%93_h%E1%BB%8Da_m%C3%A1y_t%C3%ADnh

[2] Khúc xạ dùng OpenGL – Wikipedia
https://vi.wikipedia.org/wiki/OpenGL#/media/T%E1%BA%ADp_tin:JOGL_Refrection_Demo_Screenshot.png

[3] Kết cấu di động dùng OpenGL – Wikipedia
https://vi.wikipedia.org/wiki/OpenGL#/media/T%E1%BA%ADp_tin:Perl_OpenGL_fob2.jpg

[4] Trò chơi dùng OpenGL -Wikipedia
https://vi.wikipedia.org/wiki/OpenGL#/media/T%E1%BA%ADp_tin:Synth_video_game_screenshot_C.png

[5] Computer Graphics with OpenGL – Hearn Baker Carithers.

[6] Draw some basic 3D objects with OpenGL (SharpGL) – Github
https://github.com/hthoai/3d-objects-opengl?fbclid=IwAR3oJ9fdYGxXM9V_R3Cj4UJA1bNuZNSmnmd_EQXpV6YKNXnslZHt5J-6i3c