Ứng dụng MFC (Visual C++) trong mô phỏng Robot và hệ Cơ điện tử

Bài 6:
Các thao tác vẽ cơ bản với OpenGL

PHẠM MINH QUÂN

mquan.ph@gmail.com

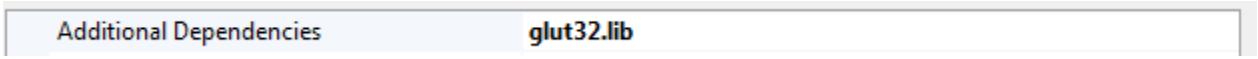
Nội dung

1. Cài đặt ánh sáng, màu sắc trong OpenGL
2. Các lệnh vẽ hình khối của thư viện GLUT
3. Biến đổi hệ tọa độ trong OpenGL

Nhắc lại về cài đặt OpenGL

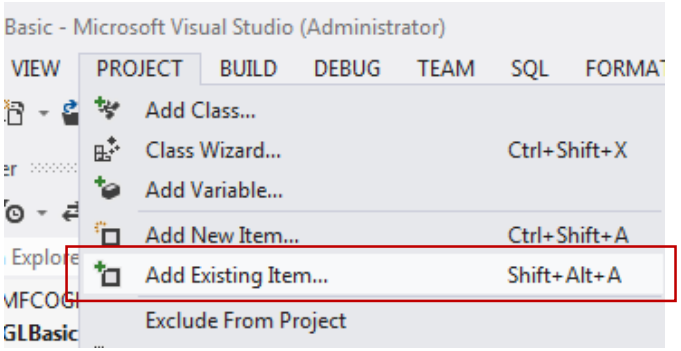
❑Tạo project mới và add thêm lớp COpenGLControl đã xây dựng từ Bài 5

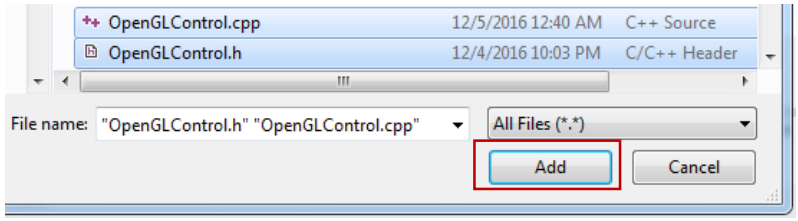
- Tạo project mới và thêm thông tin thư viện “glut32.lib” vào
  Project -> Properties -> Linker -> Input -> Additional Dependencies



- Copy các file OpenGLControl.h và OpenGLControl.cpp vào thư mục chứa các file .h và .cpp khác của project

Thêm các file này vào project bằng cách dùng menu Project -> Add Existing Items…

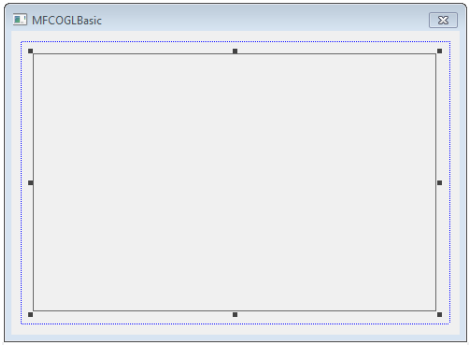




Nhắc lại về cài đặt OpenGL

❑Thêm điều khiển vào dialog và khai báo để sử dụng lớp COpenGLControl

- Thêm điều khiển vào dialog và thay đổi Properties: ID=IDC_OPENGL và Visible=False



Visible	False
Misc	
(Name)	IDC_OPENGL2 (Picture Cont
Group	False
ID	IDC_OPENGL

- Khai báo và sử dụng đối tượng của lớp COpenGLControl

File ...Dlg.h

```
#include "OpenGLControl.h"

    COpenGLControl m_oglWindow;
```

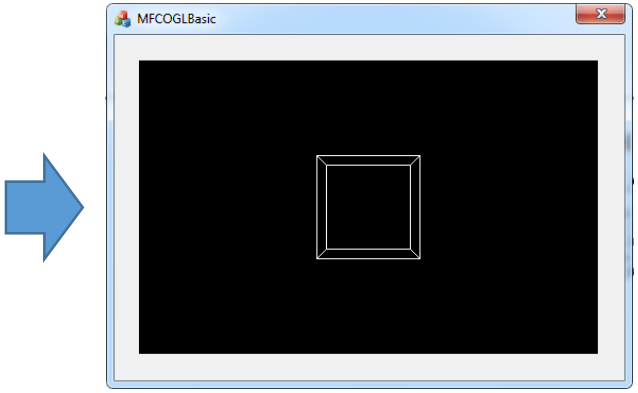
File ...Dlg.cpp

// Hàm OnInitDialog()

```
    CRect rect;
    GetDlgItem(IDC_OPENGL)->GetWindowRect(rect);
    ScreenToClient(rect);
    m_oglWindow.oglCreate(rect, this);
```

// Hàm OnPaint()

```
    else
    {
        m_oglWindow.oglDrawScene();
        CDialogEx::OnPaint();
    }
```



1. Cài đặt ánh sáng, màu sắc trong OpenGL

❑Thay đổi hình vẽ để dễ quan sát ảnh hưởng của ánh sáng, màu sắc

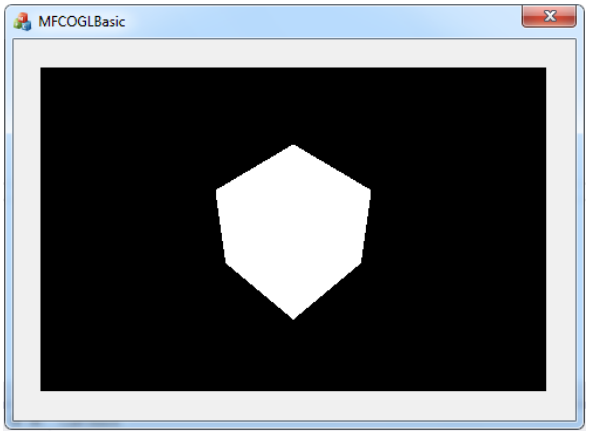
➢ Thay đổi nội dung hàm oglDrawScene() của lớp COpenGLControl như sau

```
void COpenGLControl::oglDrawScene(void)
{
    // Clear color and depth buffer bits
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    glutSolidCube (2.0);    // Vẽ khối lập phương, kích thước 2.0

    // Swap buffers
    SwapBuffers(hdc);
}
```

➢ Hiệu chỉnh lại góc nhìn:
thay đổi độ zoom, rotate ban đầu cho phù hợp

```
COpenGLControl::COpenGLControl(void)
{
   m_fPosX = 0.0f;     // X position of model in camera view
   m_fPosY = 0.0f;     // Y position of model in camera view
   m_fZoom = 10.0f;    // Zoom on model in camera view
   m_fRotX = 45.0f;     // Rotation on model in camera view
   m_fRotY = -45.0f;     // Rotation on model in camera view
}
```

Kết quả chạy khi chưa cài đặt ánh sáng, màu sắc:



1. Cài đặt ánh sáng, màu sắc trong OpenGL

❑Cài đặt nguồn sáng

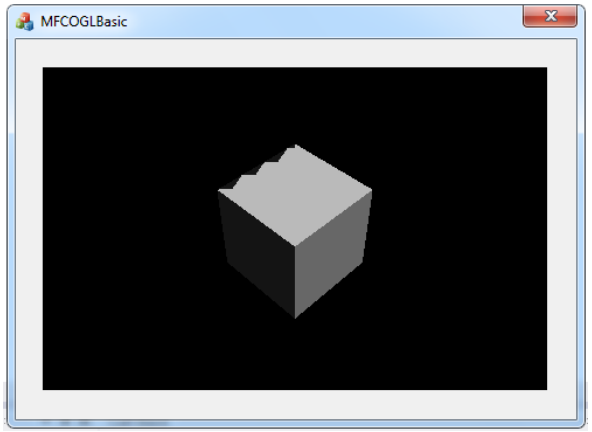
➢ Thêm đoạn code sau vào hàm *oglInitialize()* của lớp COpenGLControl

```
...
// Turn on depth testing
glEnable(GL_DEPTH_TEST);
glDepthFunc(GL_LEQUAL);

// Setup Light
GLfloat ambient[] = { 0.2f, 0.2f, 0.2f};
GLfloat diffuse[] = { 1.0f, 1.0f, 1.0f};
GLfloat specular[] = { 1.0f, 1.0f, 1.0f};
GLfloat position0[] = { 1.0f, 1.0f, 1.0f, 0.0};

glLightfv(GL_LIGHT0, GL_AMBIENT, ambient);
glLightfv(GL_LIGHT0, GL_DIFFUSE, diffuse);
glLightfv(GL_LIGHT0, GL_SPECULAR, specular);
glLightfv(GL_LIGHT0, GL_POSITION, position0);

glEnable(GL_LIGHTING);
glEnable(GL_LIGHT0);

// Send draw request
OnDraw(NULL);
}
```

// Màu sắc của ánh sáng

// Vị trí/hướng nguồn sáng

Kết quả chạy khi đã cài đặt ánh sáng trắng:



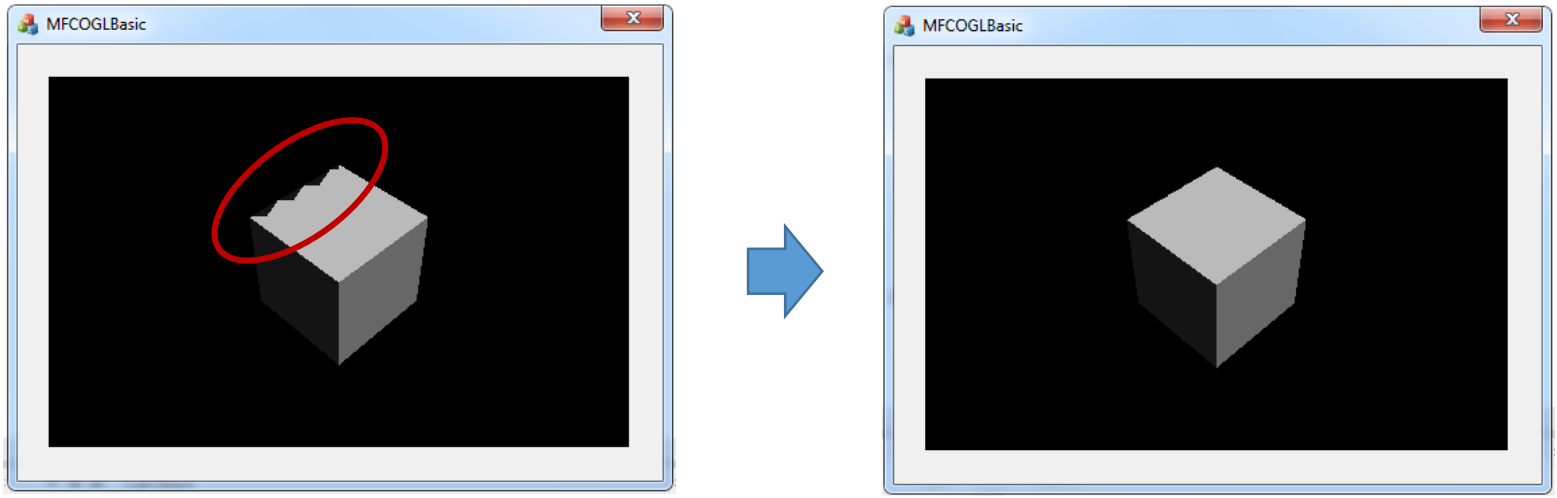
1. Cài đặt ánh sáng, màu sắc trong OpenGL

❑Cài đặt nguồn sáng

- Chỉnh lại thông số ***znear***, ***zfar*** của gluPerspective trong hàm *OnSize()* cho phù hợp với độ zoom

```
...
// Set our current view perspective
gluPerspective(35.0f, (float)cx / (float)cy, 0.1f, 2000.0f);
...
```

znear zfar



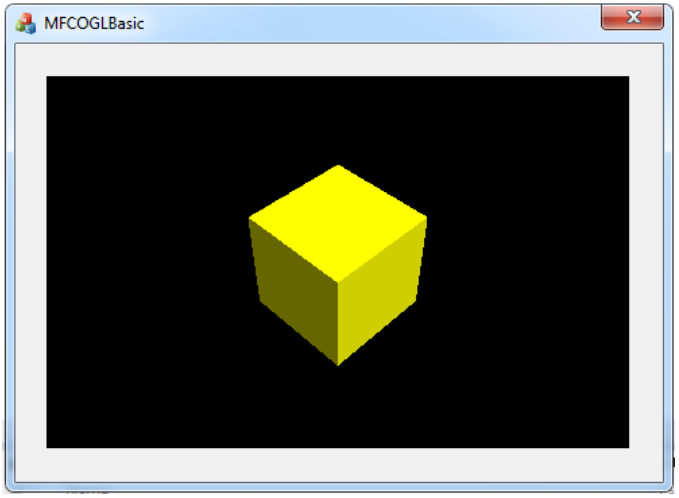
1. Cài đặt ánh sáng, màu sắc trong OpenGL

❑Cài đặt màu sắc cho vật thể

➢ Thêm đoạn code vào hàm *oglDrawScene()*

```
void COpenGLControl::oglDrawScene(void)
{
    // Clear color and depth buffer bits
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    GLfloat mat_color[] = { 1.0, 1.0, 0.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0 };

    glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE, mat_color);
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialf(GL_FRONT, GL_SHININESS, 30);

    glutSolidCube (2.0);

    // Swap buffers
    SwapBuffers(hdc);
}
```

Kết quả chạy sau khi cài đặt màu vàng cho vật thể:



❑Thay đổi màu nền

➢ Có thể thay màu nền bằng cách thay đổi tham số của glClearColor trong hàm *oglInitialize()*

```
// Set color to use when clearing the background.
glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
```

// màu nền đen

2. Các lệnh vẽ hình khối của thư viện GLUT

❑Vẽ các hình khối

Hình lập phương

```
glutSolidCube (1.0);
```

Hình cầu

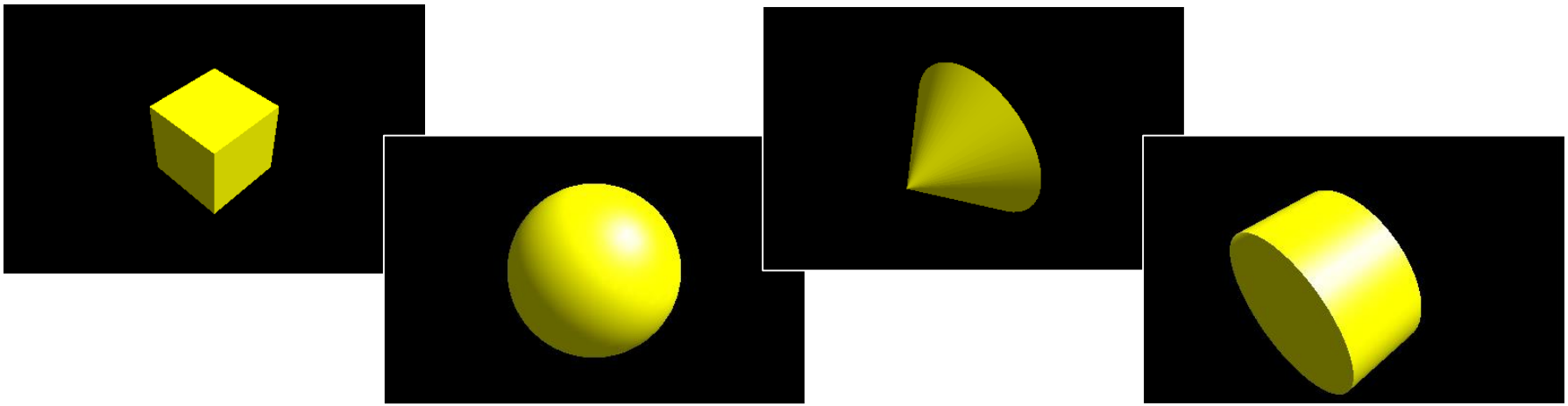
```
glutSolidSphere(1.0, 50, 50);
```

Mặt nón

```
glutSolidCone(1.0, 1.0, 50, 1);
```

Mặt trụ hoặc nón cụt

```
GLUquadric *quadric = gluNewQuadric();
gluQuadricDrawStyle(quadric, GLU_FILL);

gluCylinder(quadric, 1.0, 1.0, 1.0, 50, 10);
```

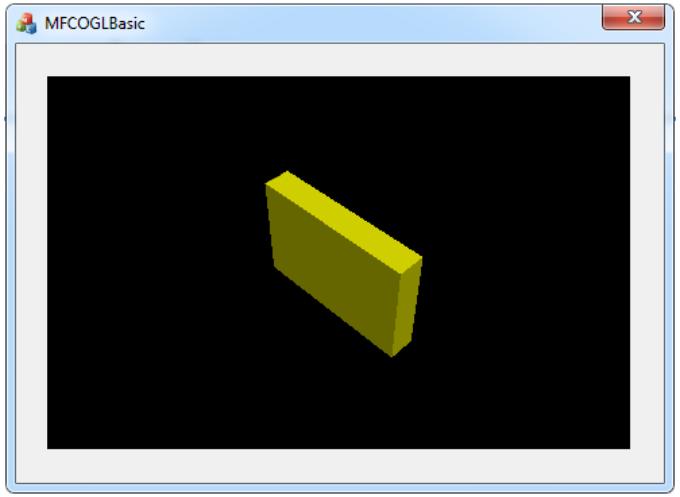


3. Biến đổi hệ tọa độ trong OpenGL

- Phép tỉ lệ *glScalef()*

Ví dụ: Vẽ khối hình hộp chữ nhật kích thước 3x2x0.5

```
glScalef (3.0, 2.0, 0.5);
glutSolidCube (1.0);
```

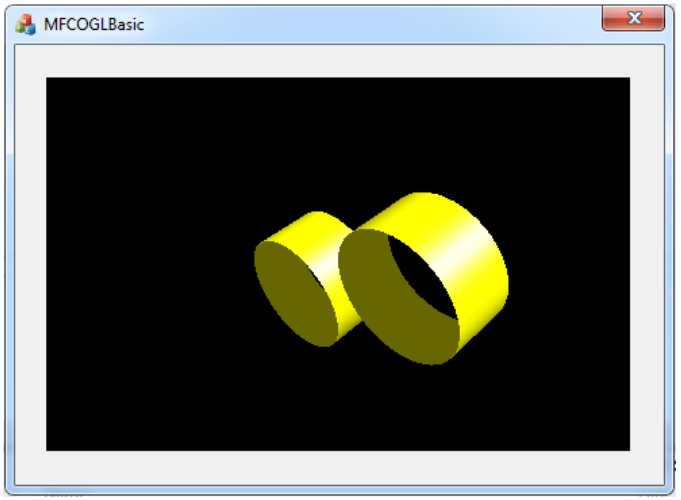


3. Biến đổi hệ tọa độ trong OpenGL

- Phép tịnh tiến *glTranslatef()*

Ví dụ: Vẽ hình trụ thứ 2 cạnh hình trụ thứ nhất cách theo trục X là 2.0, trục Y là 1.5

```
GLUquadric *quadric = gluNewQuadric();
gluQuadricDrawStyle(quadric, GLU_FILL);

gluCylinder(quadric, 1.0, 1.0, 1.0, 50, 10);
glTranslatef(2.0, 1.5, 0.0);
gluCylinder(quadric, 1.0, 1.0, 1.0, 50, 10);
```

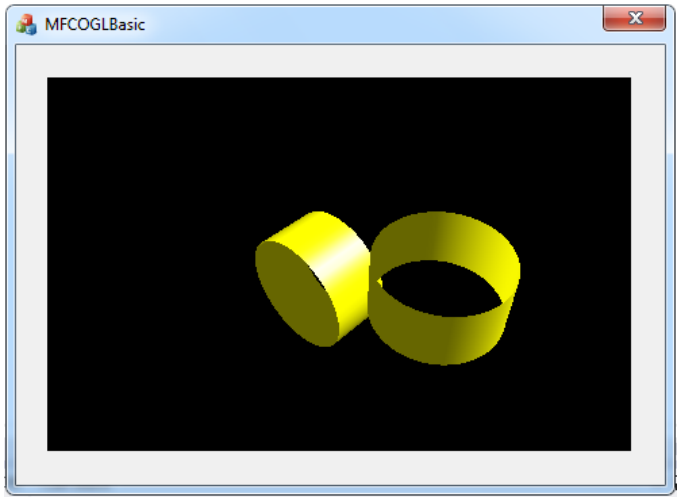


3. Biến đổi hệ tọa độ trong OpenGL

- Phép xoay *glRotatef()*

Ví dụ: Vẽ hình trụ thứ 2 cạnh hình trụ thứ nhất và xoay đi 1 góc 90 độ theo trục X

```
GLUquadric *quadric = gluNewQuadric();
gluQuadricDrawStyle(quadric, GLU_FILL);

gluCylinder(quadric, 1.0, 1.0, 1.0, 50, 10);
glTranslatef(2.0, 1.5, 0.0);
glRotatef(90, 1.0, 0.0, 0.0);
gluCylinder(quadric, 1.0, 1.0, 1.0, 50, 10);
```



3. Biến đổi hệ tọa độ trong OpenGL

❑Phạm vi tác dụng của các phép biến đổi

- Để giới hạn phạm vi tác dụng của phép biến đổi
  -> đặt code giữa glPushMatrix() & glPopMatrix()

```
glPushMatrix();
...
glPopMatrix();
```

- Có thể sử dụng nhiều tầng glPushMatrix() và glPopMatrix(), giống như các dấu { và }

```
glPushMatrix();
    glPushMatrix();
    ...
    glPopMatrix();
    glPushMatrix();
    ...
    glPopMatrix();
glPopMatrix();
```

**Lưu ý: bất cứ khi nào sử dụng đến các lệnh biến đổi tọa độ thì đều nên đặt trong phạm vi của ít nhất 1 tầng glPushMatrix() & glPopMatrix()*

Hết Bài 6