

OpenGL & GLUT

321190

2012년 봄학기

3/12/2012

박경신

OpenGL & GLUT

- ▣ OpenGL
 - <http://www.opengl.org/>
 - <http://www.sgi.com/software/opengl>
 - Windows95 이후 OpenGL이 표준으로 들어가 있음.
 - <ftp://ftp.microsoft.com/softlib/MSLFILES/opengl95.exe>
- ▣ GLUT for Win32
 - <http://www.xmission.com/~nate/glut.html>
 - [glut-3.7.6-bin.zip](http://www.xmission.com/~nate/glut.html) 내려받기
- ▣ GLUT for LINUX
 - <http://www.opengl.org/resources/libraries/glut/>
 - <http://rpmfind.net/linux/rpm2html/search.php?query=glut>
 - [glut-3.7-9.i386.rpm](http://rpmfind.net/linux/rpm2html/search.php?query=glut) 내려받기
 - *rpm -rebuild glut-3.7-9.i386.rpm (as root)*

2

Installing OpenGL & GLUT

- ▣ Libraries를 Visual C++의 <C:\Program Files\Microsoft Visual Studio 10.0\VC\lib>에 설치
 - Opengl32.lib
 - Glu32.lib
 - Glut32.lib
- ▣ Include files를 Visual C++의 <C:\Program Files\Microsoft Visual Studio 10.0\VC\include\GL>에 설치
 - Gl.h
 - Glu.h
 - Glut.h
- ▣ Dynamically-linked libraries를 <C:\WINDOWS\system32> (Windows XP)에 설치
 - Opengl32.dll, Glu32.dll, Glut32.dll

3

Compiling OpenGL/GLUT Programs

- ▣ Visual Studio 2010 VC++ 실행
- ▣ 프로젝트 새로 만들기
 - 메뉴에서 File->New->Projects
 - Win32 Console Application 선택
 - 프로젝트 이름 지정
- ▣ Linker에 library files을 지정
 - 메뉴에서 Project->Settings->link->Object/library modules (Project->Properties->Linker->Input->Additional dependencies)
 - opengl32.lib glu32.lib glut32.lib를 넣는다
- ▣ 프로젝트에 파일 새로 만들기
 - 메뉴에서 Project->Add to Project-> Files
 - 파일 이름 지정
- ▣ 빌드 (build) (F7)와 실행 (execute) (F5)

4

Compiling OpenGL/GLUT Programs

- ❑ LINUX 사용자는 cc 컴파일하거나 makefile을 이용할 것
- ❑ cc 컴파일
 - %cc -I/usr/X11R6/include program.c -o program -L/usr/X11R6/lib -lglut -lGLU -lGL -lXm -lXi -lXext -lX11 -lpthread
 - %./program
- ❑ Makefile을 아래와 같이 만들고, %make

```
CFLAGS = -I/usr/X11R6/include
LIBS = -L/usr/X11R6/lib -lglut -lGLU -lGL -lXm -lXi -lXext -lX11
       -lpthread
program: program.c
$(CC) ($CFLAGS) program.c -o program $(LIBS)
```

5

Windows System

- ❑ 윈도우 시스템
 - Microsoft Windows
 - X Window systems
- ❑ 윈도우 시스템과 OpenGL 시스템은 모두 래스터 그래픽스 시스템임
- ❑ OpenGL 프로그래밍을 하기 위해서는
 - 사용 윈도우 시스템에서 제공하는 래스터 시스템을 기반으로 윈도우 프로그래밍 수행
 - 윈도우 프로그래밍 문맥에서 추상적인 래스터 시스템인 OpenGL 시스템을 윈도우 시스템에 연결
 - OpenGL에서 제공하는 함수들을 사용하여 3차원 그래픽스 프로그래밍을 수행
 - 원하는 OpenGL 작업이 실제로 하드웨어를 제어하고 있는 사용 윈도우 시스템이 효율적으로 이해할 수 있는 형태로 전환

OpenGL

- ❑ 실리콘 그래픽스사 (SGI)가 개발한 3차원 그래픽스 라이브러리 API
- ❑ 2차원 그래픽스는 (z축의 값을 0으로 처리한) 3차원의 특수 경우로 봄
- ❑ OpenGL 그래픽스 함수는 프로그래밍 언어에 독립적인 기능으로 지정되어 있음
- ❑ C/C++, Java, Fortran, Python 등 다수 언어와 사용가능
- ❑ OpenGL은 하드웨어에 종립적임
 - No I/O library
 - No specific model loading mechanism
 - No hardware specific functions (but available as extensions)

7

OpenGL and Windowing Toolkits

- ❑ OpenGL은 하드웨어에 종립적
 - Problems displaying OpenGL scenes in a specific windowing environment
 - Different platforms have different ways to integrate OpenGL with their windowing environment
 - X Window System (GLX)
 - Apple (AGL)
 - Windows (WGL)
 - IBM OS/2 (PGL)

8

OpenGL/GLUT Programming

▣ 단순히 윈도우를 여는 프로그램 예제

```
#include <GL/glut.h>
void display (void)
{
}
int main(int argc, char *argv[])
{
    glutInit(&argc, argv);
    glutCreateWindow(argv[0]);
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}
```

9

OpenGL/GLUT Programming

▣ void glutInit([int *argc, char **argv])

- GLUT와 OpenGL 환경을 초기화. 인수에는 main의 인수를 그대로 건네줌.

▣ int glutCreateWindow(char *name)

- 윈도우를 여는 함수. 인수 name은 그 윈도우의 이름이 타이틀 바에 표시됨.

▣ void glutDisplayFunc(void (*func)(void))

- 인수 func는 열린 윈도우 내에 디스플레이하는(즉, 그림을 그리는) callback 함수 포인터. 윈도우가 열리거나 다른 윈도우에 의해 숨겨진 윈도우가 다시 디스플레이될 때 이 함수가 실행

▣ void glutMainLoop(void)

- GLUT 루프. 이 함수의 호출로 프로그램은 이벤트를 기다리는 상태임.

10

OpenGL/GLUT Programming

▣ 윈도우를 전부 파란색으로 칠하는 프로그램 예제

```
#include <GL/glut.h>
void display (void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glFlush();
}
void init (void)
{
    glClearColor(0.0, 0.0, 1.0, 1.0);
}
int main(int argc, char *argv[])
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_RGBA);
    glutCreateWindow(argv[0]);
    glutDisplayFunc(display);
    init();
    glutMainLoop();
    return 0;
}
```

11

OpenGL/GLUT Programming

▣ void glutInitDisplayMode(unsigned int mode)

- 디스플레이의 표시모드를 설정. Mode에 GLUT_RGBA를 지정했을 경우는 색의 지정을 RGB로 사용함을 지정. 그 밖에 인덱스 칼라모드 (GLUT_INDEX)를 지정하면 효율을 향상시킬 수 있음.

▣ void glClearColor(GLclampf R, GLclampf G, GLclampf B, GLclampf A)

- 윈도우를 전부 칠할 때의 색을 지정. R, G, B, A는 0~1 사이의 값을 가짐. (0, 0, 0, 1)을 지정하면 백색의 블루명을 그림.

▣ void glClear(GLbitfield mask)

- 윈도우를 전부 칠함. Mask에는 전부 칠하는 버퍼를 지정한다. OpenGL이 관리하는 화면상의 버퍼(메모리)에는 color buffer, depth buffer, stencil buffer, overlay buffer, 등이 겹쳐서 존재함. GL_COLOR_BUFFER를 지정했을 때는 컬러버퍼만 전부 칠해짐.

▣ void glFlush(void)

- 이 함수는 아직 실생되지 않은 OpenGL명령을 전부 실행.

12

OpenGL/GLUT Program 작성 예

▣ 윈도우 내에 선을 그리는 프로그램 예제

```
#include <GL/glut.h>
void display (void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_LINE_LOOP);
    glVertex2d(-0.9, -0.9);
    glVertex2d(0.9, -0.9);
    glVertex2d(0.9, 0.9);
    glVertex2d(-0.9, 0.9);
    glEnd();
    glFlush();
}
void init (void)
{
    /* 변경없음 */
}
int main(int argc, char *argv[])
{
    /* 변경없음 */
}
```

13

OpenGL/GLUT Programming

▣ void glBegin(GLenum mode)

▣ void glEnd(void)

- 도형을 그리려면, glBegin()과 glEnd() 사이에 그 도형의 각 정점의 좌표치를 설정하는 함수를 둠. Mode에 GL_POINTS, GL_LINES, GL_POLYGON, 등등 도형의 타입을 지정.

▣ void glVertex2d(GLdouble x, GLdouble y)

- 이 함수는 2차원의 좌표치를 설정하는 사용. 인수의 형태는 GLdouble임. Float 형태는 glVertex2f(..)를 int 형태는 glVertex2i(..)를 사용함.

14

OpenGL/GLUT Programming

▣ 윈도우 내에 도형을 전부 칠하는 프로그램 예제

```
#include <GL/glut.h>
void display (void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3d(1.0, 0.0, 0.0);
    glBegin(GL_POLYGON);
    glVertex2d(-0.9, -0.9);
    glVertex2d(0.9, -0.9);
    glVertex2d(0.9, 0.9);
    glVertex2d(-0.9, 0.9);
    glEnd();
    glFlush();
}
```

15

OpenGL/GLUT Programming

▣ void glColor3d(GLdouble r, GLdouble g, GLdouble b)

- 이 함수는 그림의 색을 지정. 인수는 GLdouble 형태로, r, g, b에는 각각 0~1의 범위에서 지정함. 인수가 Float 형태는 glColor3f(..)를 int 형태는 glColor3i(..)를 사용함.

16

OpenGL/GLUT Programming

▣ 도형의 색을 정점마다 지정하는 프로그램 예제

```
#include <GL/glut.h>
void display (void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POLYGON);
    glColor3d(1.0, 0.0, 0.0); // red
    glVertex2d(-0.9, -0.9);
    glColor3d(0.0, 1.0, 0.0); // green
    glVertex2d(0.9, -0.9);
    glColor3d(0.0, 0.0, 1.0); // blue
    glVertex2d(0.9, 0.9);
    glColor3d(1.0, 1.0, 0.0); // yellow
    glVertex2d(-0.9, 0.9);
    glEnd();
    glFlush();
}
```