CGL
COMPUTER GRAPHICS LABORATORY

컴퓨터 그래픽스
[02]

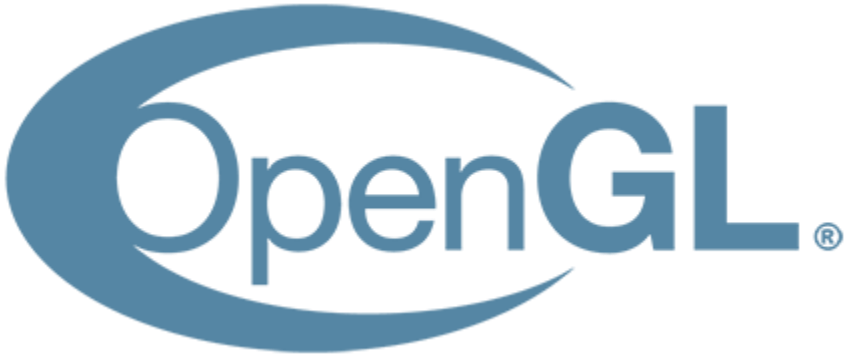
2023학년도 1학기

담당교수 : 마 준

그래픽 렌더링 라이브러리

- Direct X
  - Microsoft사에서 배포한 그래픽 라이브러리
  - Windows에 종속적인 특징
- Vulkan
  - OpenGL NG, glNext 등으로 불리던 그래픽 라이브러리
  - 크로스 플랫폼 2D/3D 그래픽 라이브러리
  - 정보가 적은 편

- 범용적 그래픽 렌더링 API
  - 그래픽 소프트웨어 라이브러리
  - 그래픽 하드웨어에 접근하기 위한 소프트웨어
  - 1980년대 실리콘 그래픽스(SGI)에 의해서 개발됨
  - 2006년 크로노스 그룹에 의해 관리받는 것으로 이전됨



OpenGL의 장점

- Windows system에 독립적
- Operating system에 독립적
- Graphics programming의 표준
- 많은 platforms에서 사용 가능
- 빠르고 이식성이 좋은 open library
- 성능이 좋은 (hardware-accelerated) 그래픽 API

OpenGL as a renderer

- 기하 기본요소 (Geometric primitives)
  - Points, lines, polygons
- 이미지 기본요소 (Image primitives)
  - Images and bitmaps
  - Separate pipeline for images and geometry
  - Texture mapping
- State dependent rendering
  - Color, materials, light sources etc.

관련 API

- GLU (OpenGL Utility Library)
  - 많은 모델링을 제공: quadric surfaces, NURBS, tessellators
- GLUT (OpenGL Utility Toolkit)
  - 윈도우와 user interface API
  - 편리한 멋진 형태의 primitives 제공 (torus, teapot, cube)
- GLEW (OpenGL Extension Wrangler Library)
  - OpenGL 확장 라이브러리로 쉐이더 프로그램을 작성할 때 사용
- GLFW (OpenGL Library for Windows)
  - Windows 프로그램을 생성하기 위해 사용하는 라이브러리 (GLUT와 유사)

실습을 위한 OpenGL Setup

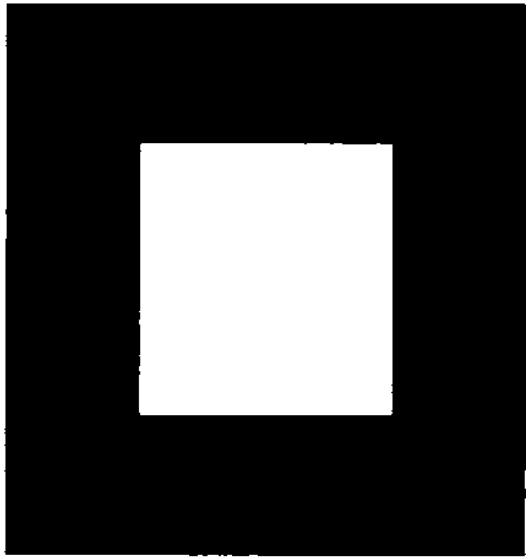
CGL
COMPUTER GRAPHICS LABORATORY

- Visual Studio 2010
- GLUT을 과목 사이트에서 다운 받을것 (학내가상강좌사이트)
- 다운로드 받은 파일을 위치로 복사
  - **glut32.dll을 C:₩WINDOWS₩system32에 복사**
  - **glut32.lib을 C:₩Program Files₩Microsoft Visual Studio 9.0₩VC98₩Lib에 복사**
  - **glut.h을 C:₩Program Files₩Microsoft Visual Studio 9.0₩VC98₩Include₩GL에 복사**

OpenGL 참고 사이트

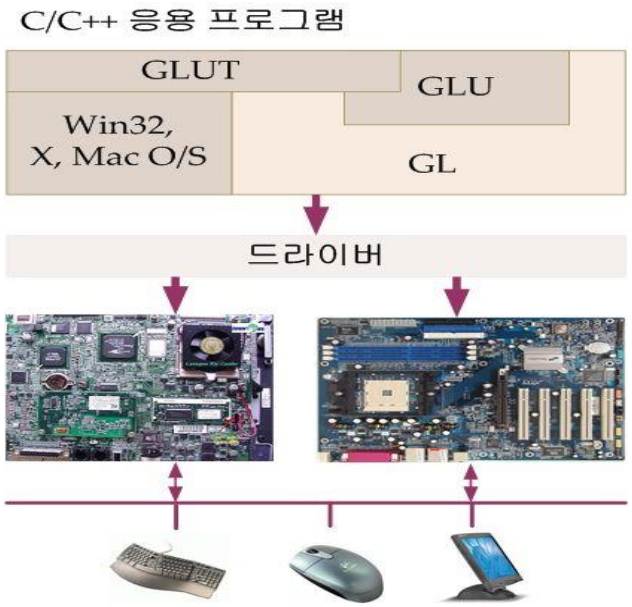
- OpenGL의 공식 사이트(영문)
  - http://www.opengl.org
- OpenGL API Tutorial을 제공하는 nehe.gamedev.net(영문)
  - http://nehe.gamedev.net
- 현직 개발자 개인 사이트, OpenGL에 대한 자료와 nehe.gamedev.net의 번역자료도 보유하고 있음
  - http://www.gisdeveloper.co.kr/category/Programming/OpenGL
- 델파이를 통한 OpenGL – OpenGL의 기본 지식에 대해 자세히 설명되어있음
  - http://www.delphinara.com.ne.kr/open_r.htm
- 소엔 랩
  - http://soen.kr/

- 화면에 간단한 사각형 그리기
  - OpenGL을 이용한 사각형 출력



GL & GLU : OpenGL Utility Library

- GL: 렌더링 기능을 제공하는 함수 라이브러리
- GLU: GL 라이브러리의 도우미 역할을 하는 라이브러리
  - 약 50여개의 함수로 구성
  - 다각형 분할(Tessellation), 투상(Projection), 2차원 곡면 (Quadratic Surface), 너브스(NURBS) 등 고급기능을 제공



GLUT : OpenGL Utility Toolkit

- 운영체제와 시스템 수준의 입출력을 가능하게 만드는 OpenGL 프로그램용 유틸리티 라이브러리로 사용자 입력을 받아들이거나 화면 윈도우를 제어하기 위한 함수
  - 운영체제에 대한 깊이 있는 지식이 없더라도 쉽게 프로그램을 작성 가능
  - 창의 크기와 형태를 정의하고 제어
  - 키보드와 마우스 입력을 감지하는 기능
  - 정육각형, 구, 찻잔과 같은 기하학적인 기본객체 (geometric primitives) 제공
  - 팝업 메뉴를 생성하는 기능 제공
  - glut라는 접두사로 시작: glutPostRedisplay, glutCreateWindow

실습 Code: 01_01 사각형 그리기

CGL
COMPUTER GRAPHICS LABORATORY

```c
#include <GL/glut.h>
#include <stdlib.h>


void display(void)
{
    glColor3f(1.0, 1.0, 1.0); // 흰색으로 설정

    // 사각형의 좌표 입력
    glBegin(GL_POLYGON);
        glVertex3f(0.25, 0.25, 0.0);
        glVertex3f(0.75, 0.25, 0.0);
        glVertex3f(0.75, 0.75, 0.0);
        glVertex3f(0.25, 0.75, 0.0);
    glEnd();

    glFlush(); // 사각형을 화면에 그림
}
```

```c
void init(void)
{
    glClearColor (0.0, 0.0, 0.0, 0.0);  // 윈도우 배경을 검은색으로 설정
    glClear(GL_COLOR_BUFFER_BIT); // 색상 버퍼를 지움
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity(); // 단위 행렬

    // 직교 투영: left, right, bottom, top, near, far
    glOrtho(0.0, 1.0, 0.0, 1.0, -1.0, 1.0);
}


int main(int argc, char** argv)
{

    glutInit(&argc, argv);   // GLUT 상태를 초기화
    // 디스플레이모드형식 선택: Single buffer & RGBA color 모드 선택
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);

    // 창의 크기와 위치 설정
    glutInitWindowSize(250, 250);
    glutInitWindowPosition(100, 100);

    glutCreateWindow ("01 사각형 그리기"); // 윈도우 생성
    init();

    // 필요한 콜백함수 등록
    glutDisplayFunc(display);
    glutMainLoop();  // 이벤트 처리엔진 시작
    return 0;
}
```

주요 함수 설명

- void glutInit(int *argc, char **argv);
  - GLUT 라이브러리를 초기화하고 기반 플랫폼의 윈도우 시스템과 연결하여 하나의 세션 형성
  - 윈도우 시스템과 연결할 수 없다거나 해당 운영체제가 그래픽 인터페이스를 제공하지 않는다면 에러 메시지를 출력하고 프로그램을 강제 종료
  - 다른 어떤 GLUT 함수보다 먼저 실행 되어야 함

주요 함수 설명

- void glutInitDisplayMode(unsigned int mode);
  - 그리기 표면의 주요 특징들을 결정
  - 한번 윈도우를 생성한 후에는 변경할 수 없음
  - 여러 개의 모드를 OR 연산자로 묶어서 지정(상호 배타적인 속성의 플래그들은 하나만 지정 가능)

mode	설 명	mode	설 명
GLUT_RGBA	RGBA의 창 모드를 지원	GLUT_ALPHA	색상버퍼에 알파값 사용
GLUT_RGB	GLUT_RGBA와 동일	GLUT_DEPTH	깊이버퍼 사용
GLUT_INDEX	인덱스를 가진 색상만 지원	GLUT_STENCIL	stencil buffer 사용
GLUT_SINGLE	단일버퍼 사용	GLUT_MULTISAMPLE	다중맵플링 지원
GLUT_DOUBLE	이중버퍼 사용	GLUT_STEREO	입체창(stereo window) 지원
GLUT_ACCUM	누적버퍼 사용	GLUT_LUMINANCE	휘도를 가진 색상모델 지원

주요 함수 설명

- void glutInitWindowSize(int width, int height);
  - 그래픽을 출력할 윈도우의 크기(너비와 높이)를 지정
- void glutInitWindowPosition(int x, int y);
  - 화면 좌상단을 원점으로 하여 x, y 위치에 윈도우의 좌상단을 위치 시킴
- int glutCreateWindow(char *name);
  - 윈도우를 생성
  - 윈도우 상단의 제목을 문자열을 인수로 지정
  - 새로운 윈도우에 대한 ID가 리턴됨
  - glutMainLoop()이 호출되기 전까지는 윈도우가 디스플레이되지 않음

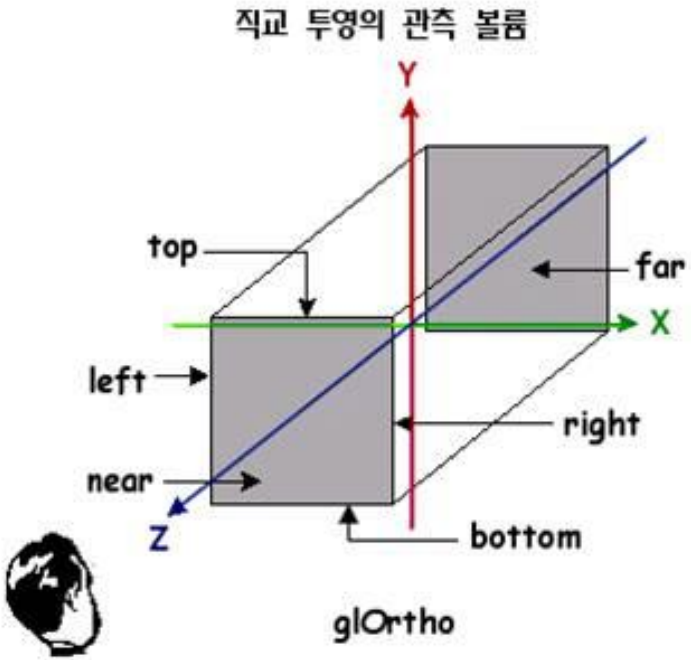
주요 함수 설명

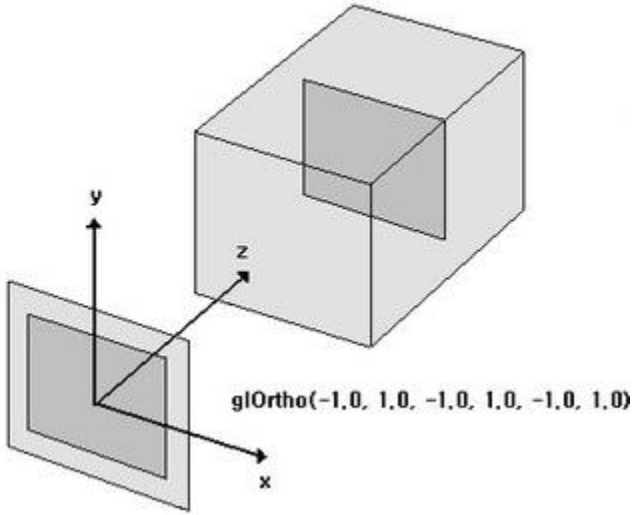
- void glClearColor (GLclampf red, GLclampf green, GLclampf blue, GLclampf alpha);
  - 윈도우의 배경을 지울때 사용될 색상을 지정
  - 4개의 인수는 RGBA 각 색상 요소의 강도를 지정 (0 ~ 1)
- void glMatrixMode(GLenum mode);
  - 사용할 행렬 모드를 지정

mode	설 명
GL_MODELVIEW	이후의 행렬들을 Modelview 행렬 스택에 적용
GL_PROJECTION	이후의 행렬들을 Projection 행렬 스택에 적용
GL_TEXTURE	이후의 행렬들을 Texture 행렬 스택에 적용
GL_COLOR	이후의 행렬들을 Color 행렬 스택에 적용

- void glLoadIdentity();
  - 행렬을 단위행렬로 초기화

- void glOrtho(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble nearVal, GLdouble farVal);
  - 직교 투영을 위한 범위 설정
  - 멀고 가까움은 표현 불가





주요 함수 설명

CGL
COMPUTER GRAPHICS LABORATORY

- void glutDisplayFunc(void (*func)(void));
  - 디스플레이 콜백 함수 호출
- Display 콜백 함수가 호출 경우
  - 처음 윈도우를 열 때
  - 윈도우 위치를 옮길 때
  - 윈도우 크기를 조절할 때
  - 앞 윈도우에 가려져 안 보이던 뒤 윈도우가 활성화 되어 앞으로 드러날 때
  - glutPostRedisplay() 함수에 의해 이벤트 큐에 flag가 게시될 때
- 콜백 함수
  - 필요한 콜백 함수를 등록하고 해당 콜백함수에 실행될 내용을 입력
  - 콜백 함수에 대한 호출은 GLUT가 알아서 처리함

주요 함수 설명

- void glClear(GLbitfield mask);
  - 인수로 어떤 버퍼를 지울 것인가를 지정
  - OR 연산자로 연결하여 두 개 이상의 버퍼를 한꺼번에 지울 수 있음

mask	설 명
GL_COLOR_BUFFER_BIT	색상 버퍼를 지움
GL_DEPTH_BUFFER_BIT	깊이 버퍼를 지움
GL_STENCIL_BUFFER_BIT	스텐실 버퍼를 지움
GL_ACCUM_BUFFER_BIT	누적 버퍼를 지움

- void glColor3f(GLfloat red, GLfloat green, GLfloat blue);
  - Red, Green, Blue를 이용한 색상 지정 (0.0 ~ 1.0)

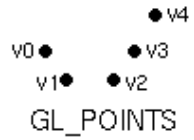
Suffix	Data Type	Typical Corresponding C-Language Type	OpenGL Type Definition
b	8-bit integer	signed char	GLbyte
s	16-bit integer	short	GLshort
i	32-bit integer	int or long	GLint, GLsizei
f	32-bit floating-point	float	GLfloat, GLclampf
d	64-bit floating-point	double	GLdouble, GLclampd
ub	8-bit unsigned integer	unsigned char	GLubyte, GLboolean
us	16-bit unsigned integer	unsigned short	GLushort
ui	32-bit unsigned integer	unsigned int or unsigned long	GLuint, GLenum, GLbitfield

Table 1-1 Command Suffixes and Argument Data Types

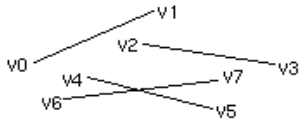
주요 함수 설명

- void glBegin(GLenum mode); void glEnd(void);
  - 선, 점, 면 등을 그리기 위해 사용
  - glBegin()과 glEnd()의 사이에 정점 정보를 넣음

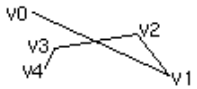
Value	Meaning
GL_POINTS	Treats each vertex as a single point. Vertex *n* defines point *n*. *N* points are drawn.
GL_LINES	Treats each pair of vertices as an independent line segment. Vertices *2n - 1* and *2n* define line *n*. *N/2* lines are drawn.
GL_LINE_STRIP	Draws a connected group of line segments from the first vertex to the last. Vertices *n* and *n+1* define line *n*. *N - 1* lines are drawn.
GL_LINE_LOOP	Draws a connected group of line segments from the first vertex to the last, then back to the first. Vertices *n* and *n + 1* define line *n*. The last line, however, is defined by vertices *N* and *1*. *N* lines are drawn.
GL_TRIANGLES	Treats each triplet of vertices as an independent triangle. Vertices *3n - 2*, *3n - 1*, and *3n* define triangle *n*. *N/3* triangles are drawn.



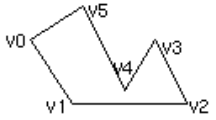
GL_POINTS



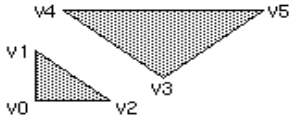
GL_LINES



GL_LINE_STRIP

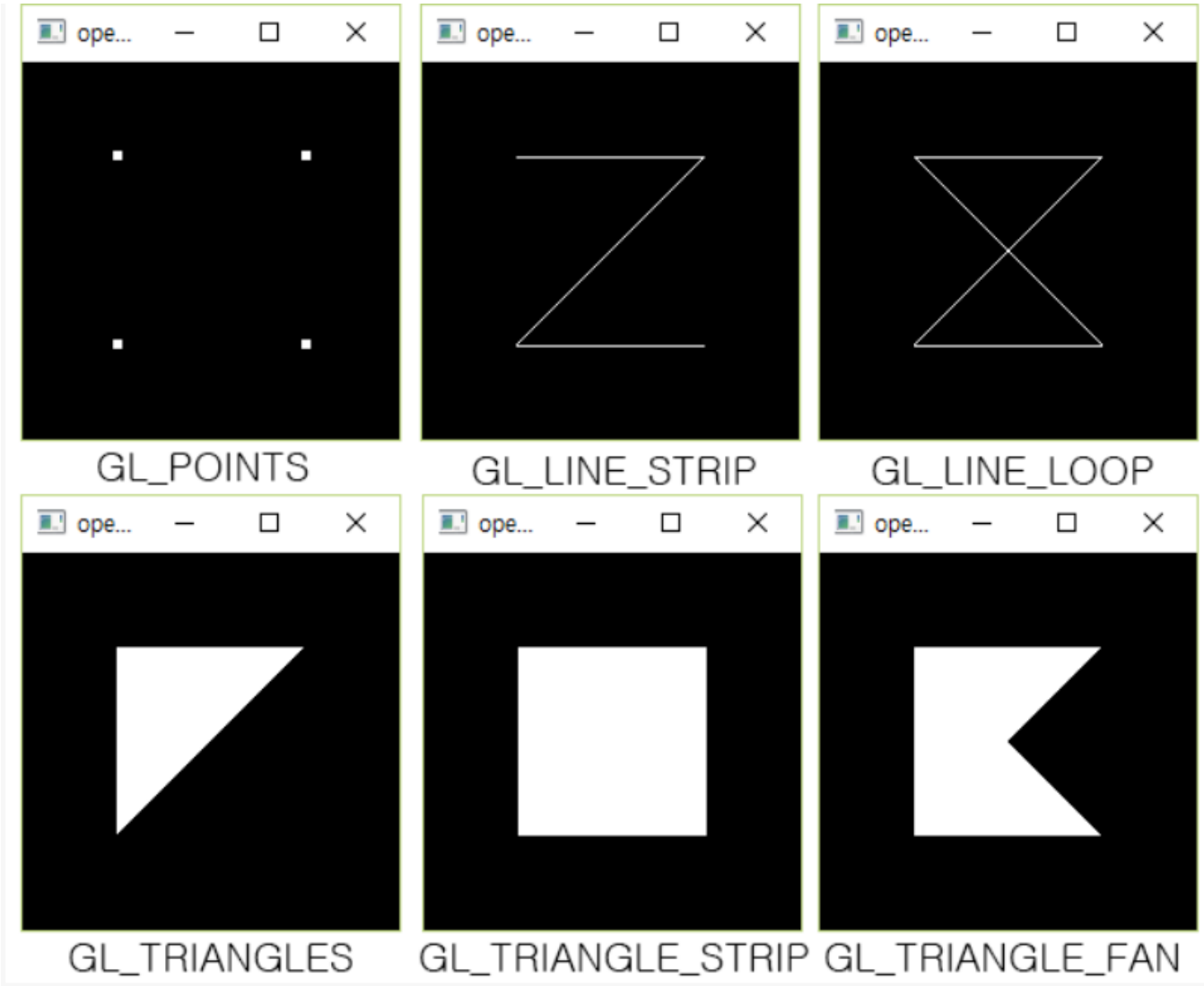


GL_LINE_LOOP



GL_TRIANGLES

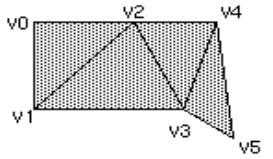
주요 함수 설명


ope...
GL_POINTS
GL_LINE_STRIP
GL_LINE_LOOP
GL_TRIANGLES
GL_TRIANGLE_STRIP
GL_TRIANGLE_FAN

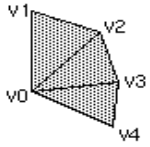
주요 함수 설명

CGL
COMPUTER GRAPHICS LABORATORY

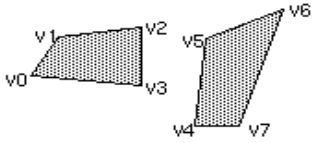
GL_TRIANGLE_STRIP	Draws a connected group of triangles. One triangle is defined for each vertex presented after the first two vertices. For odd *n*, vertices *n*, *n* + *1*, and *n* + *2* define triangle *n*. For even *n*, vertices *n* + *1*, *n*, and *n* + *2* define triangle *n*. *N* - *2* triangles are drawn.
GL_TRIANGLE_FAN	Draws a connected group of triangles. one triangle is defined for each vertex presented after the first two vertices. Vertices *1*, *n* + *1*, *n* + *2* define triangle *n*. *N* - *2* triangles are drawn.
GL_QUADS	Treats each group of four vertices as an independent quadrilateral. Vertices *4n* - *3*, *4n* - *2*, *4n* - *1*, and *4n* define quadrilateral *n*. *N/4* quadrilaterals are drawn.
GL_QUAD_STRIP	Draws a connected group of quadrilaterals. One quadrilateral is defined for each pair of vertices presented after the first pair. Vertices *2n* - *1*, *2n*, *2n* + *2*, and *2n* + *1* define quadrilateral *n*. *N/2* - *1* quadrilaterals are drawn. Note that the order in which vertices are used to construct a quadrilateral from strip data is different from that used with independent data.
GL_POLYGON	Draws a single, convex polygon. Vertices *1* through *N* define this polygon.



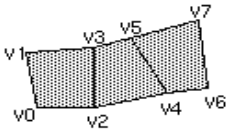
GL_TRIANGLE_STRIP



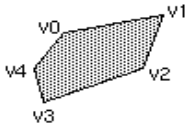
GL_TRIANGLE_FAN



GL_QUADS



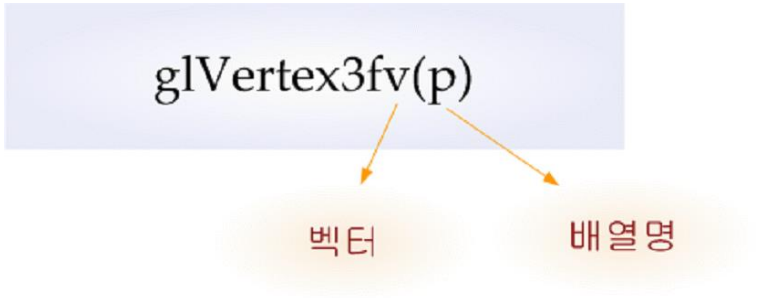
GL_QUAD_STRIP



GL_POLYGON

주요 함수 설명

- void glVertex3f(GLfloat x, GLfloat y, GLfloat z);
  - 정점의 x, y, z 좌표 설정




```
float p[3] = {10.0, 20.0, 30.0};
glVertex3fv(p);
```

- void glFlush(void);
  - 버퍼에서 기다리던 명령들을 실행

주요 함수 설명

- void glutMainLoop(void);
  - 무한 이벤트 처리 루프
  - 이벤트가 발생하면 일치하는 콜백 함수가 등록되어 있을 경우, 등록된 함수가 자동으로 호출

실습 Code: 01_02 랜덤사각형 그리기

CGL
COMPUTER GRAPHICS LABORATORY

```
#include <stdlib.h>
#include <time.h>
#include <gl/glut.h>

static int delay = 10;

void init()
{
    srand(time(0));
    glClearColor( 1.0, 1.0, 1.0, 1.0 );
    glColor3f   ( 1.0, 0.0, 0.0 );
    gluOrtho2D  ( 0.0, 50.0, 0.0, 50.0 );
    glClear     ( GL_COLOR_BUFFER_BIT );
}

void display()
{
    int x1, y1, x2, y2, r, g, b;

    x1=rand()%50;  y1=rand()%50;  x2=rand()%50; y2=rand()%50;
    r =rand()%256; g =rand()%256; b =rand()%256;

    glColor3f( (GLfloat)r/255, (GLfloat)g/255, (GLfloat)b/255 );

    glBegin( GL_POLYGON );
    glVertex2f( x1, y1 ); glVertex2f( x1, y2 );
    glVertex2f( x2, y2 );   glVertex2f( x2, y1 );
    glEnd();

    glFlush();
}
```

```
void timer(int t)
{
    glutPostRedisplay();
    glutTimerFunc( delay, timer, t );
}

int main(int argc, char* argv[])
{
    glutInit( &argc, (char**)argv );
    glutInitWindowSize( 500, 500 );
    glutCreateWindow("Prog02: Random boxes");
    glutDisplayFunc(display);
    glutTimerFunc( delay, timer, 0 );

    init();
    glutMainLoop();

    return 0;
}
```

CGL
COMPUTER GRAPHICS LABORATORY

THANK YOU

마 준 | majun@sch.ac.kr