

KAIST

OPENGL tutorial 2

Joo Ho Lee

VISUAL COMPUTING Lab

KAIST

Announcement

- Please buy this book.

FOUNDATIONS OF
3D COMPUTER GRAPHICS
Steven J. Gortler

Steven J. Gortler, MIT PRESS, 2012

VISUAL COMPUTING Lab

KAIST

Computer Structure

- A computer has two processing units
 - Central processing unit: CPU
 - Graphics processing unit: GPU

CPU

GPU

VISUAL COMPUTING Lab

KAIST

OpenGL graphics system

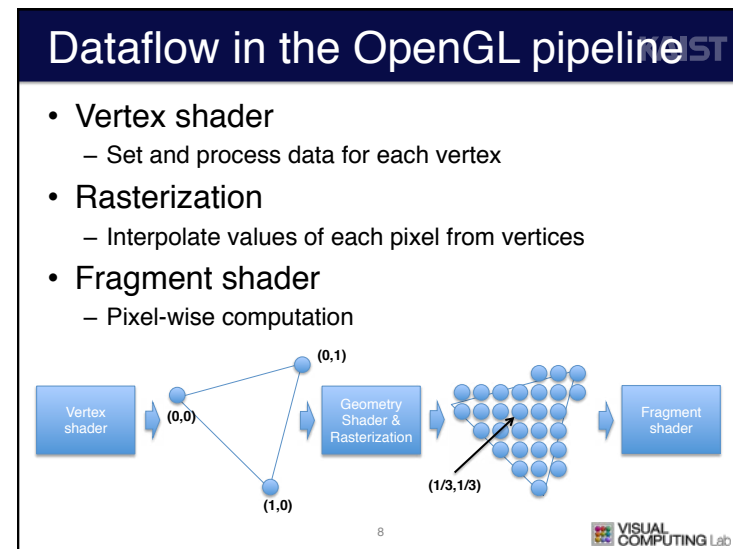
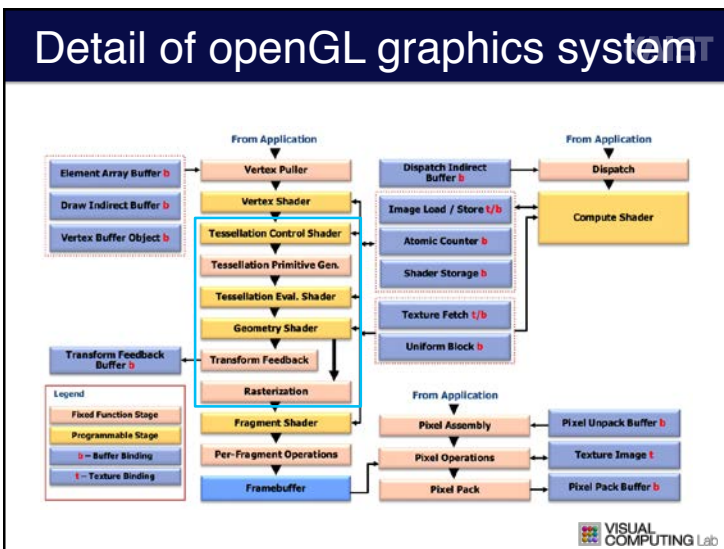
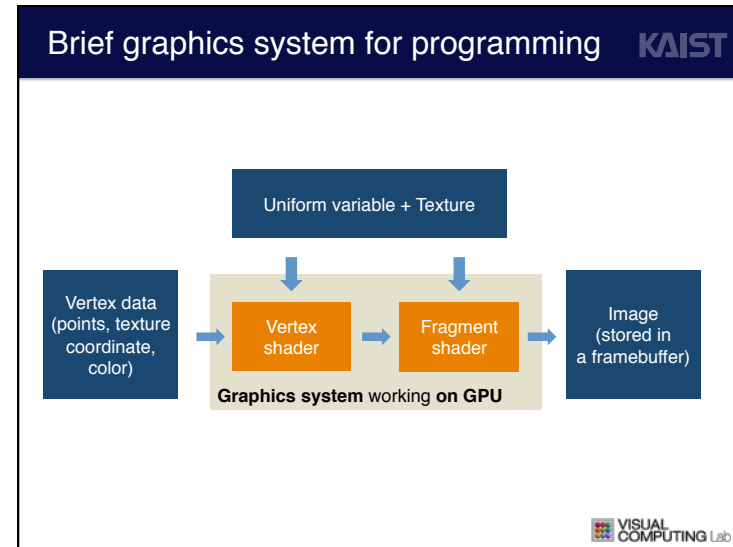
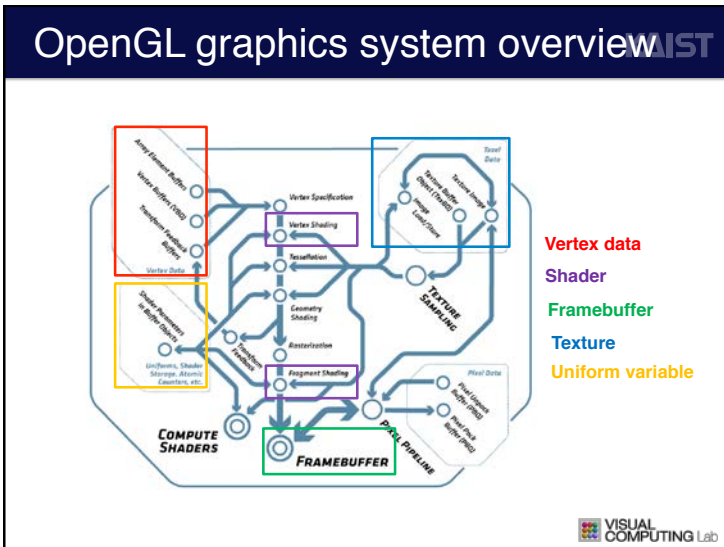
- The system working on **GPU** to produce **high-quality graphical images, specifically color images of three-dimensional objects.**

3D scene information
(e.g. 3D object, light,
color, camera)

OpenGL Graphics System

Graphical images

VISUAL COMPUTING Lab

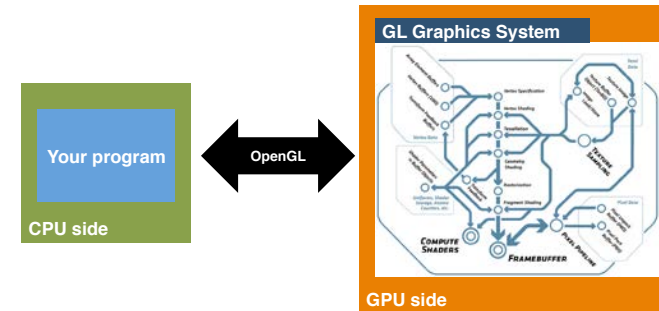


Why we need OpenGL? KAIST

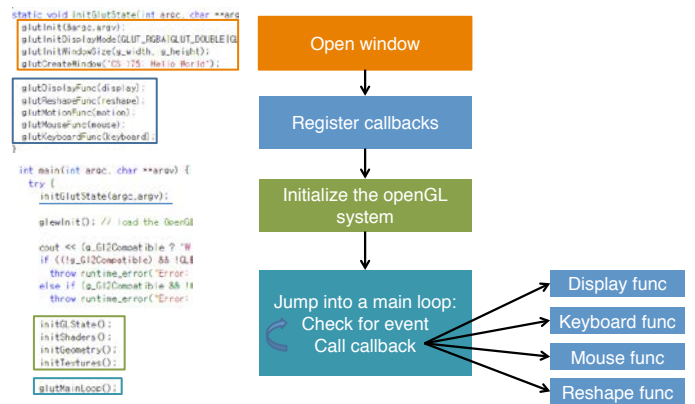
- How we pass 3D scene data to the OpenGL graphics system?
- How we change the mode (state) of the graphics system?
- How we launch the graphics system to render the image?

Overview of your openGL program KAIST

- Using openGL, your program control and interact with the OpenGL graphics system.

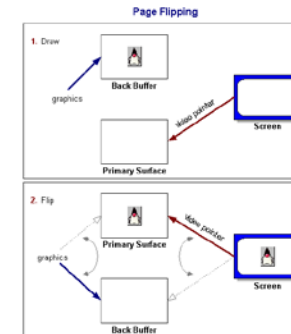


openGL Program workflow KAIST



Double buffering KAIST

- `glutInitDisplayMode(GLUT_DOUBLE)`
- Page Flipping: `glutSwapBuffer()`



Passing data to the OpenGL graphics system – Uniform variable

- First, get a location of a uniform variable
 - glGetUniformLocation
- Second, specify the value of a uniform variable
 - glUniformXX

```

ShaderState(const char* vert, const char* frag) {
    readShaderSource(ShaderProgram, vert, frag);

    const GLuint n = program; // start here

    // Set some handles to uniform variables
    GLuint texID = glGetUniformLocation(program, "textureID");
    GLuint uID = glGetUniformLocation(program, "uTexID");

    // Set some handles to vertex attributes
    GLuint pos = glGetAttribLocation(program, "position");
    GLuint tex = glGetAttribLocation(program, "texCoord");
    GLuint coord = glGetAttribLocation(program, "texCoord");
    GLuint coord = glGetAttribLocation(program, "texCoord");

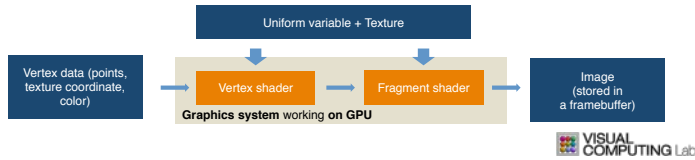
    state = new ShaderState();
    state->program = program;
    state->texID = texID;
    state->uTexID = uID;
    state->pos = pos;
    state->tex = tex;
    state->coord = coord;

    state->drawFunc = [this] {
        glUseProgram(program);

        glUniform1i(texID, textureID);
        glUniform1f(uTexID, 1.0f);

        glBindVertexArray(vao);

        glDrawElements(GL_TRIANGLES, 6, GL_UNSIGNED_INT, 0);
    };
}
    
```



Passing data to the OpenGL graphics system – vertex data

```

void draw(const ShaderState& state) {
    int count = 6;
    state->bindVertexBuffers(count, state->pos, state->tex, state->coord);
    state->drawFunc();
}

void draw(const ShaderState& state) {
    int count = 6;
    state->bindVertexBuffers(count, state->pos, state->tex, state->coord);
    state->drawFunc();
}

void draw(const ShaderState& state) {
    int count = 6;
    state->bindVertexBuffers(count, state->pos, state->tex, state->coord);
    state->drawFunc();
}
    
```

Where is a start button for rendering?

```

void draw(const ShaderState& state) {
    int count = 6;
    state->bindVertexBuffers(count, state->pos, state->tex, state->coord);
    state->drawFunc();
}

void draw(const ShaderState& state) {
    int count = 6;
    state->bindVertexBuffers(count, state->pos, state->tex, state->coord);
    state->drawFunc();
}

void draw(const ShaderState& state) {
    int count = 6;
    state->bindVertexBuffers(count, state->pos, state->tex, state->coord);
    state->drawFunc();
}
    
```

OpenGL graphics system starts to operate!

Homework 1

- Geometric overview

Reshaping KAIST

Detect changing window size

Call reshape function

```

                /// Whenever a window is resized, a "resize" event is
                /// generated and glut is told to call this reshape
                /// callback function to handle it appropriately.

                static void reshape(int w, int h) {
                    g_width = w;
                    g_height = h;
                    glViewport(0, 0, w, h);
                    glutPostRedisplay();
                }
            
```

- glViewport sets the width and height of the viewing box.
- GlutPostRedisplay() calls the display callback function.

VISUAL COMPUTING Lab

Texture mapping KAIST

- texture(uTexUnitX, vTexCoordX)

VISUAL COMPUTING Lab

Texture mapping KAIST

- In a vertex shader
 - Set a texture coordinate for each vertex
- In a fragment shader
 - As an input of the fragment shader, we get an interpolated texture coordinate for each pixel.

19 VISUAL COMPUTING Lab

Tip KAIST

- The openGL functions are well specified in OpenGL documentation.
- Just google it or read a text book.

20 VISUAL COMPUTING Lab