Computer Graphics

유주한

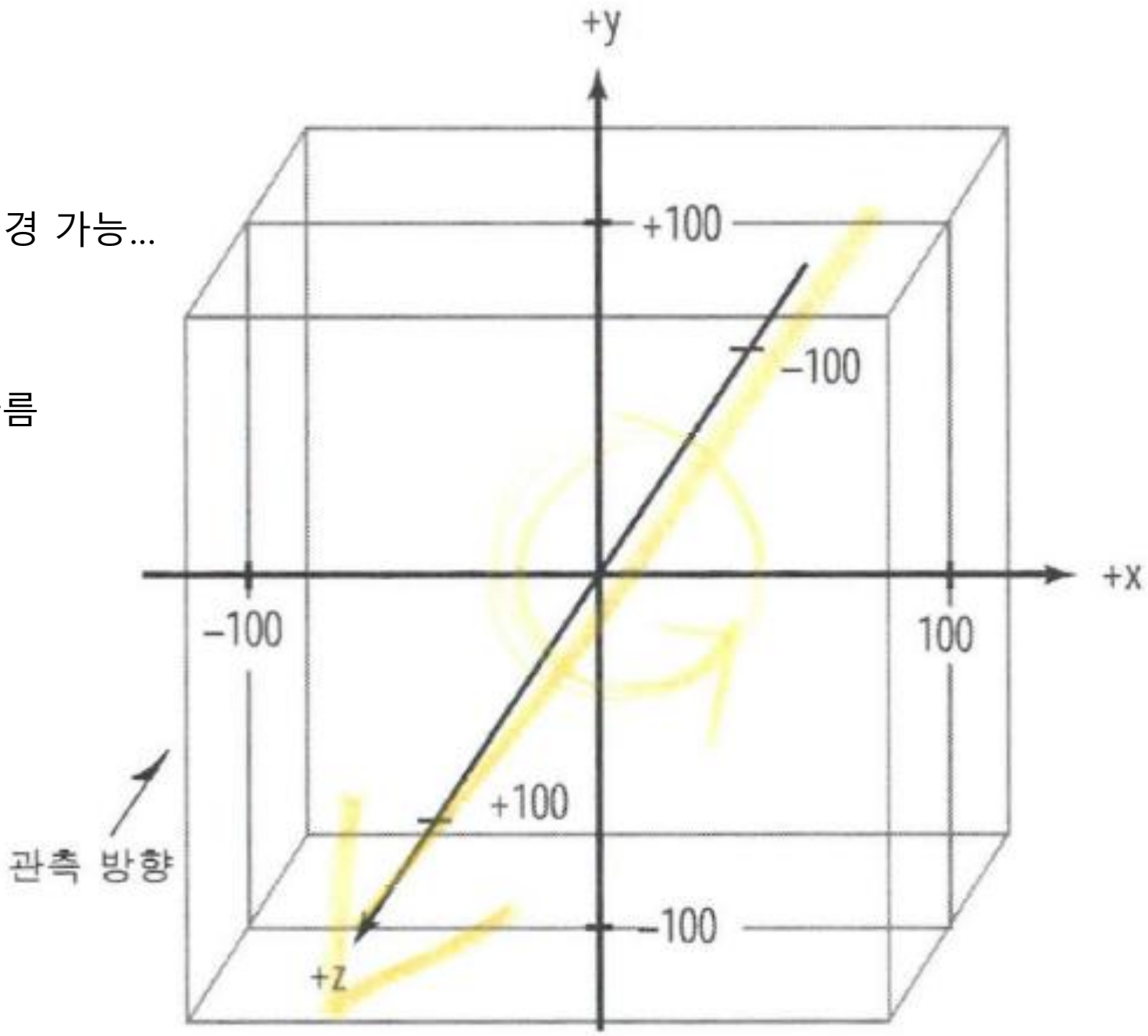
컴퓨터AI학부
동아대학교
2025년 01학기

점 그리기(3_1)

- 3D 캔버스의 설정

GL_PROJECTION 모드에 따라 변경 가능...
- glOrtho()
- gluPerspective()

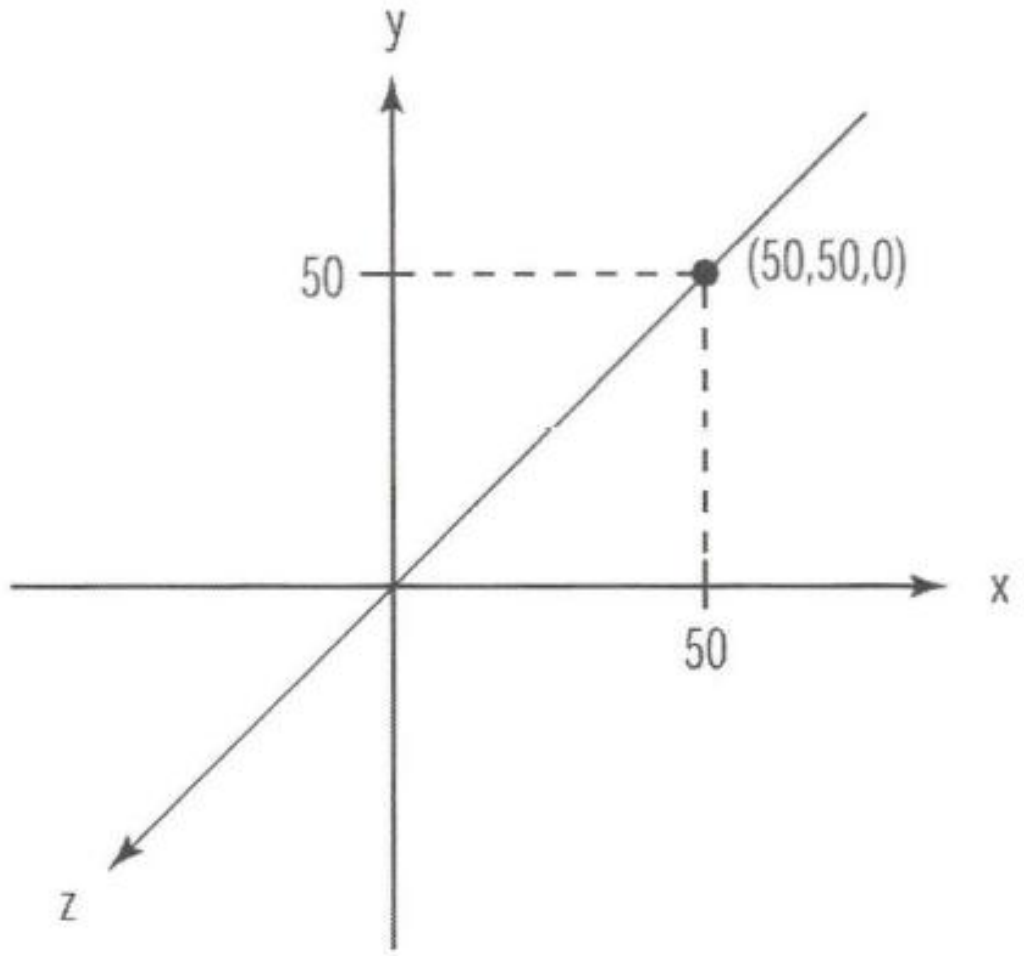
일반적으로는 왼쪽 그림 구정 따름



점 그리기(3_1)

- 3D 캔버스 내에서 무언가를 그릴 지점을 선택하기 위해서는 OpenGL 함수인 glVertex를 사용
- x축으로 50만큼, y축으로 50만큼 이동한 지점에 점을 찍는 코드

```
glVertex3f(50.0f, 50.0f, 0.0f);
```



- 2개 인자
  - glVertex2f(50.0f, 50.0f)
  - Z좌표는 0.0으로 고정

- 4개 인자 : 동차좌표계(homogeneous coordinate system)
  - glVertex4f()
  - 4번째 값은 크기 조정

점 그리기(3_1)

- glBegin 의 인자인 GL_POINT는 OpenGL로 하여금 이어지는 버텍스가 점으로 해석되도록 함
- 따라서 이어지는 두 좌표는 점으로 해석되어, 화면에 두 점이 그려짐

```
glBegin(GL_POINTS);                          // 기본 모델로 점을 선택한다.
    glVertex3f(0.0f, 0.0f, 0.0f);            // 점을 지정한다.
    glVertex3f(50.0f, 50.0f, 50.0f);         // 또 다른 점을 지정한다.
glEnd();                                     // 점 그리기 완료
```

- 다음과 같은 방법은 코드의 낭비이며, 실행 속도도 상대적으로 느림

```
glBegin(GL_POINTS);                  // 그려질 점을 지정한다.
    glVertex3f(0.0f, 0.0f, 0.0f);
glEnd();

glBegin(GL_POINTS);                  // 다른 점을 지정한다.
    glVertex3f(50.0f, 50.0f, 50.0f);
glEnd()
```

점 그리기(3_1)

- code

- 노란색 원 안에 점(빨간색)이 있는 것임!
  - 눈 좋은 사람~
  - glVertex3f(0.0f, 0.0f, 0.0f);
  - glVertex3f(50.0f, 50.0f, 50.0f);

점 그리기(3_1_1)

- 다음과 같이
  직선을 따라 점들을 찍어 보시오!
- 직선의 범위는(0,0,0)~(50,50,50)

- 노란색 원 안에 점(빨간색)

점 그리기(3_1_1)

- 잘 그렸는데... 점이 잘 안보임...

- 점의 크기 변경

```
void glPointSize(GLfloat 크기);
```

- 지원되는 점의 크기 범위와 단계를 얻어야...어느 범위내에서 크기를 설정해야 하는지 알아야...
  - 시스템에서 지원되는 점 크기의 범위(size), 점 크기 사이의 최소치 (단계 값, step)

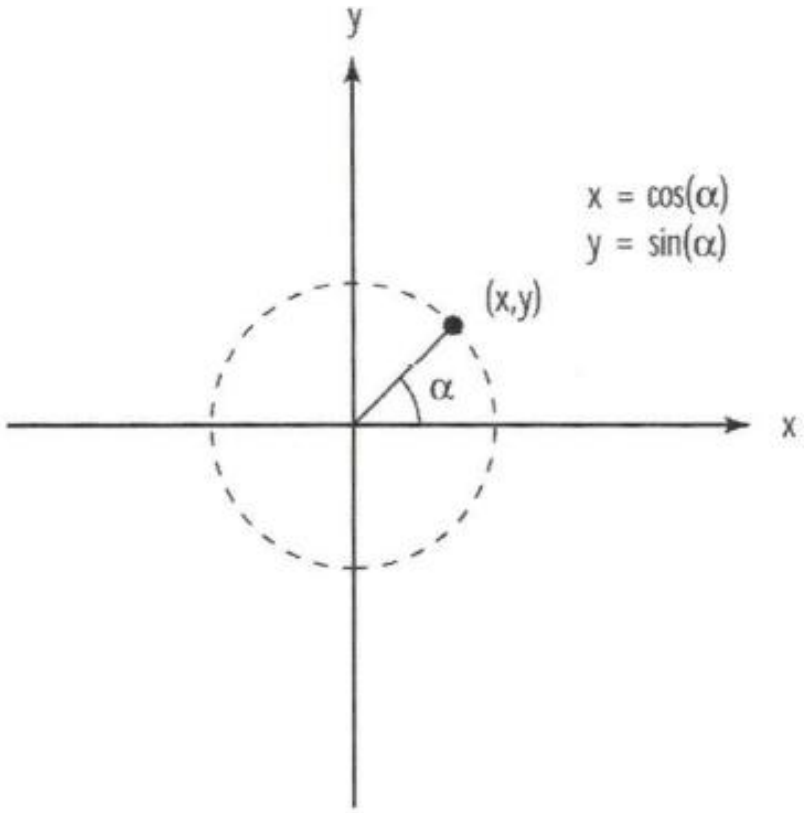
```
GLfloat sizes[2];   // 지원되는 점 크기의 범위를 저장한다.
GLfloat step;       // 지원되는 점 크기의 간격을 저장한다.
glGetFloatv(GL_POINT_SIZE_RANGE,sizes);
glGetFloatv(GL_POINT_SIZE_GRANULARITY,&step);
```

  - ✓ sizes 배열은 각각 지원되는 점의 최소치와 최대치 저장
  - ✓ step 변수는 크기 사이의 간격을 저장

점 그리기(3_1_1_a)

Try

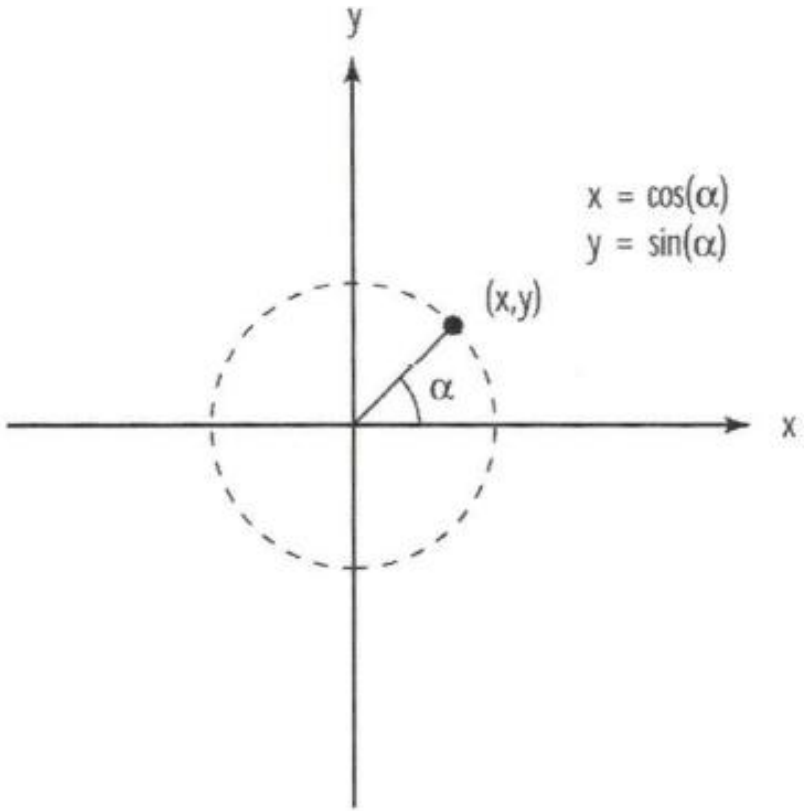
- 삼각함수를 이용하여 원 모양으로 점을 그리시오



점 그리기(3_1_1_a)

- 삼각함수를 이용하여 원 모양으로 점을 그리시오

```
#include <GL/glut.h>
#include<iostream>

#define GL_PI 3.1415f

//장면 렌더링
void RenderScene(void)
{
    //std::cout << "RenderScene" << std::endl;
    GLfloat x, y, z, angle;

    //현재 색상을 사용하여 화면을 지운다 .
    glClear(GL_COLOR_BUFFER_BIT);

    glColor3f(1.0f, 0.0f, 0.0f);

    glBegin(GL_POINTS);
    for (angle = 0.0f; angle <= (2.0f * GL_PI); angle += 0.1f)
    {
        x = 50.0f * cos(angle);
        y = 50.0f * sin(angle);
        //점의 위치를 지정하고 다음 위치 설정을 위해 Z 값을 약간 증가시칸다 .
        glVertex3f(x, y, 0);
    }
    glEnd();

    glFlush();
}
```

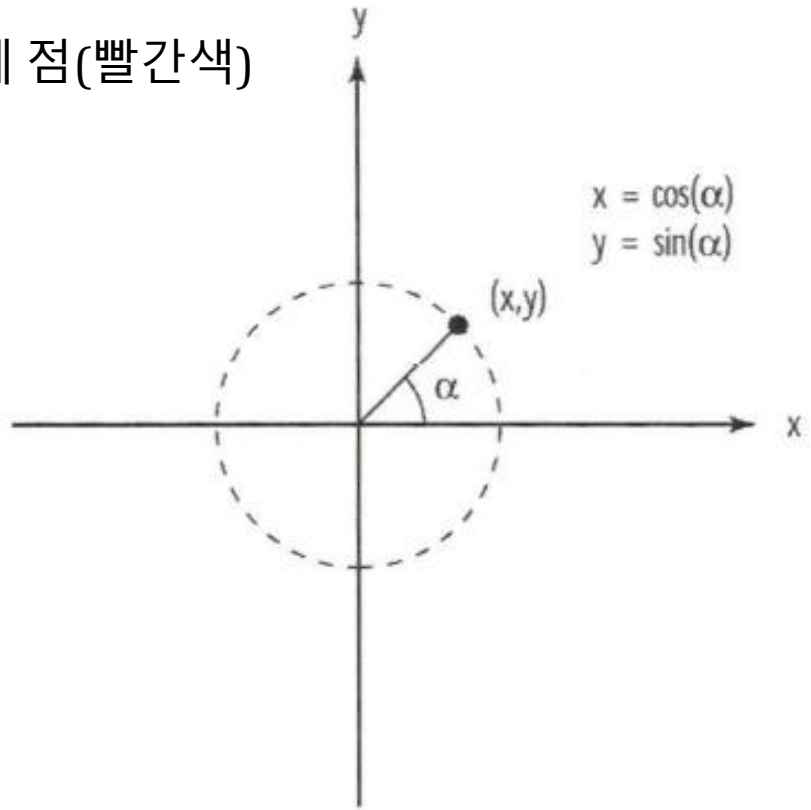


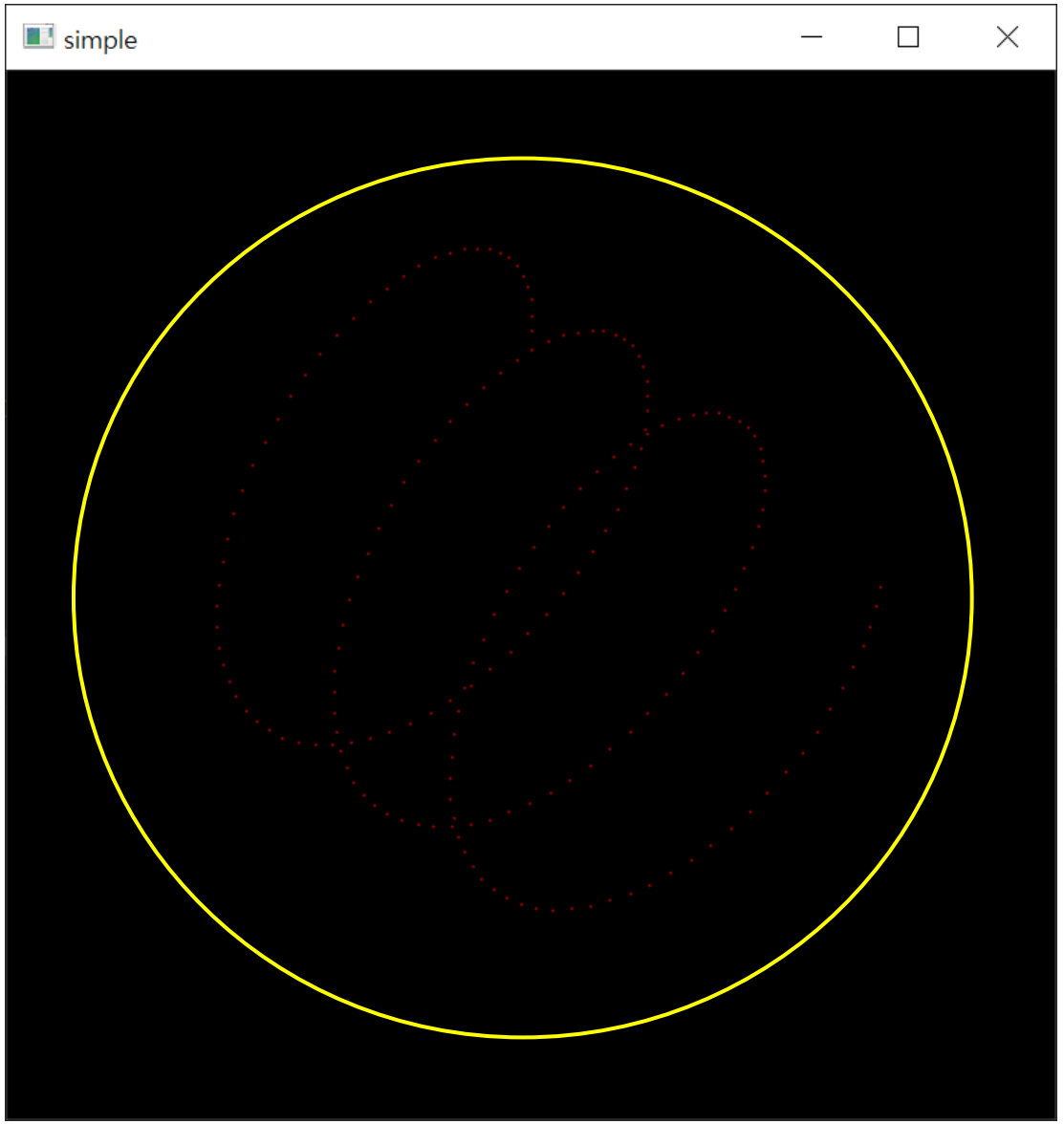
점 그리기(3_1_2)

Try

- z축을 기준으로 나선 모양으로 회전하는 경로로 여러 점들을 찍는 예제

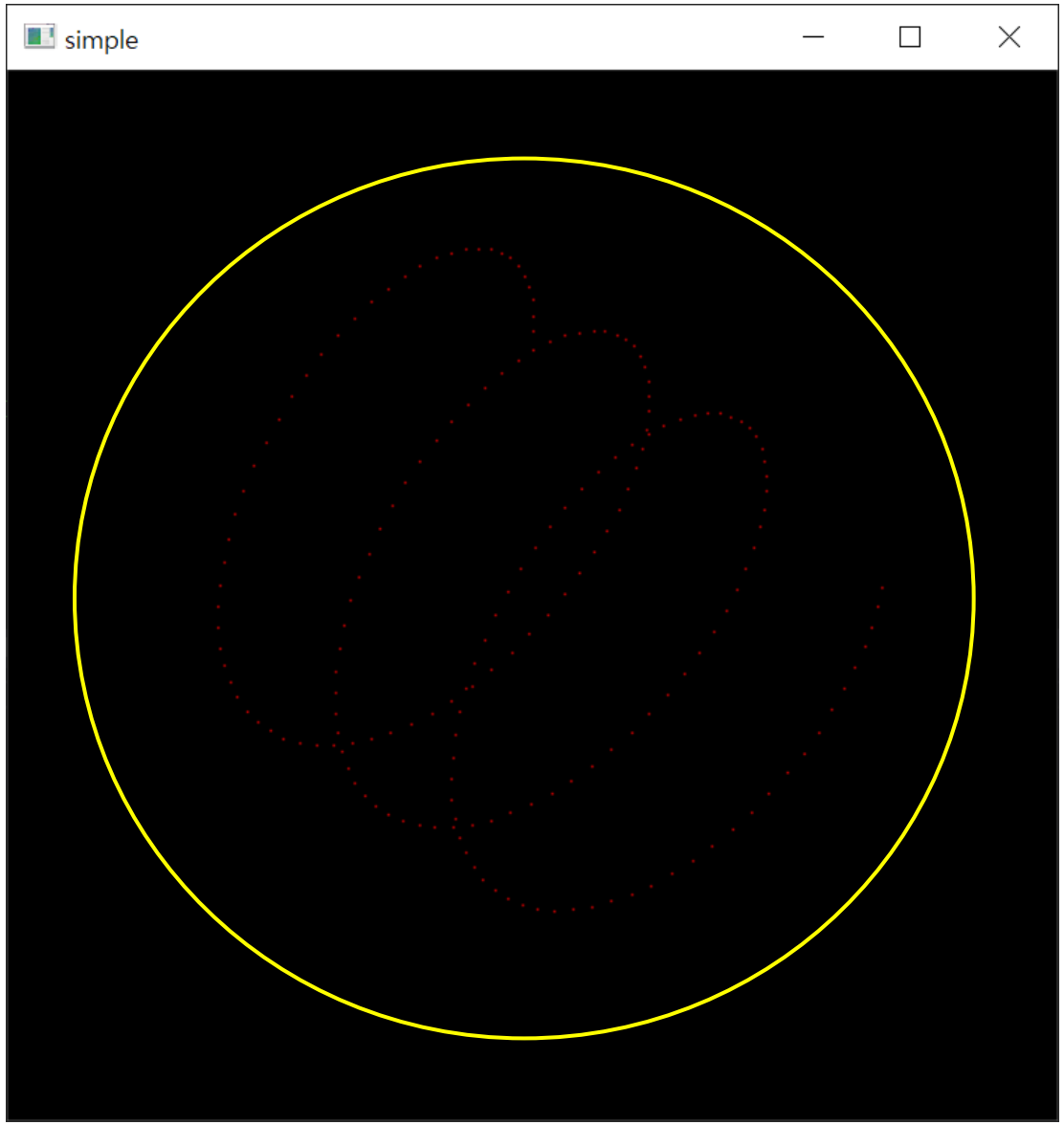
- code
  - push / pop matrix 관련 나중에... : (3_1_2_r)
  - 3번 회전
  - 노란색 원 안에 점(빨간색)





점 그리기(3_1_2)

- 이 code에서 **시작점**은 어디일까?



점 그리기(3_1_3)

Try

- 3_1_2에서 점의 크기를 기존보다 조금 크게 해서 출력하기!

점 그리기(3_1_4)

Try

- 점의 크기가 점점 커지는 나선형 출력하기!
  - 점의 크기 변경

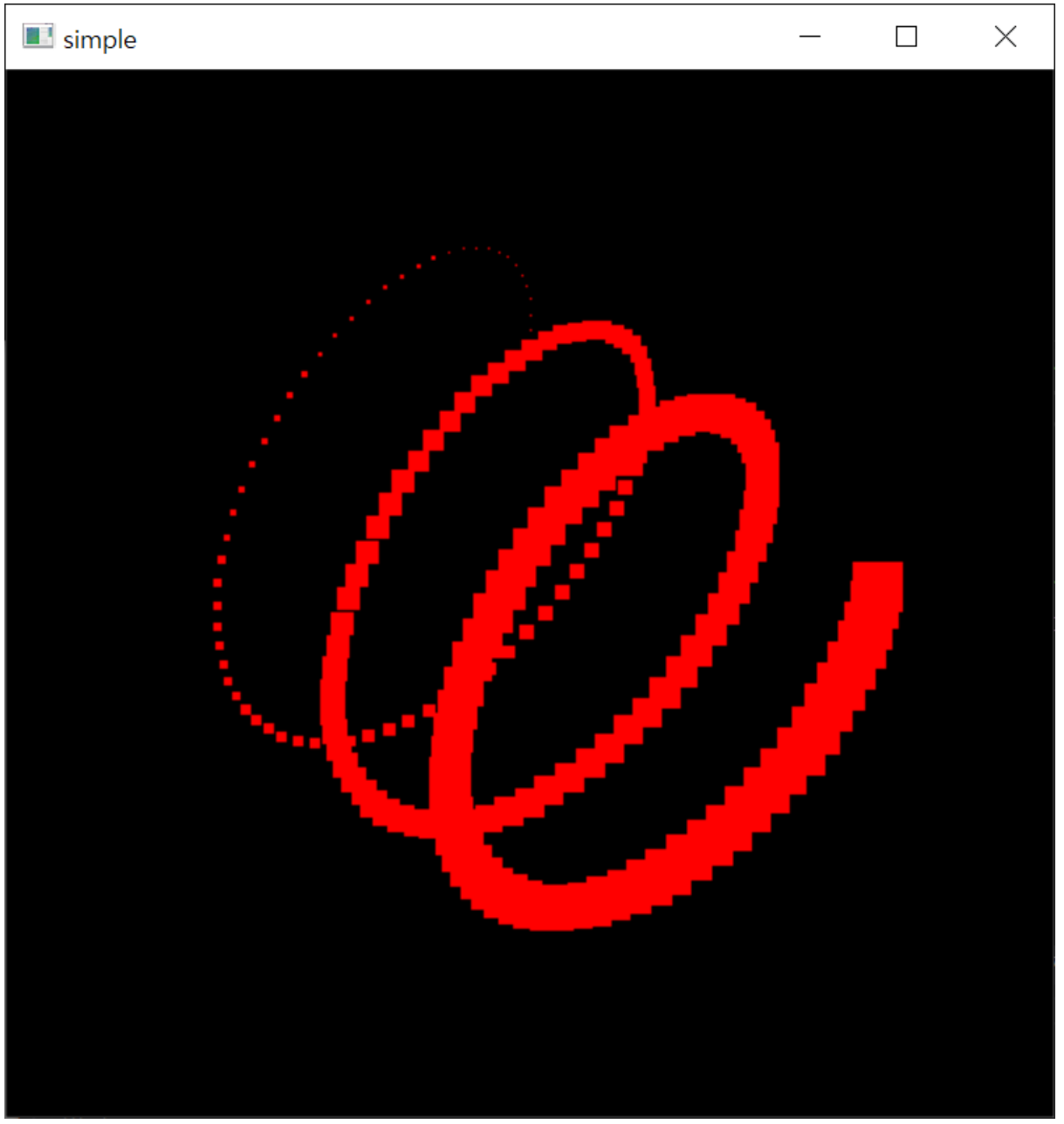
```
void glPointSize(GLfloat 크기);
```

  - 지원되는 점의 크기 범위와 단계를 얻어야...
    어느 범위내에서 크기를 설정해야 하는지 알아야...
  - 시스템에서 지원되는 점 크기의 범위(size),
    점 크기 사이의 최소치 (단계 값, step)

```
GLfloat sizes[2];    // 지원되는 점 크기의 범위를 저장한다.
GLfloat step;        // 지원되는 점 크기의 간격을 저장한다.
glGetFloatv(GL_POINT_SIZE_RANGE,sizes);
glGetFloatv(GL_POINT_SIZE_GRANULARITY,&step);
```

✓ sizes 배열은 각각 지원되는 점의
최소치[0]와 최대치[1] 저장

✓ step 변수는 크기 사이의 간격을 저장



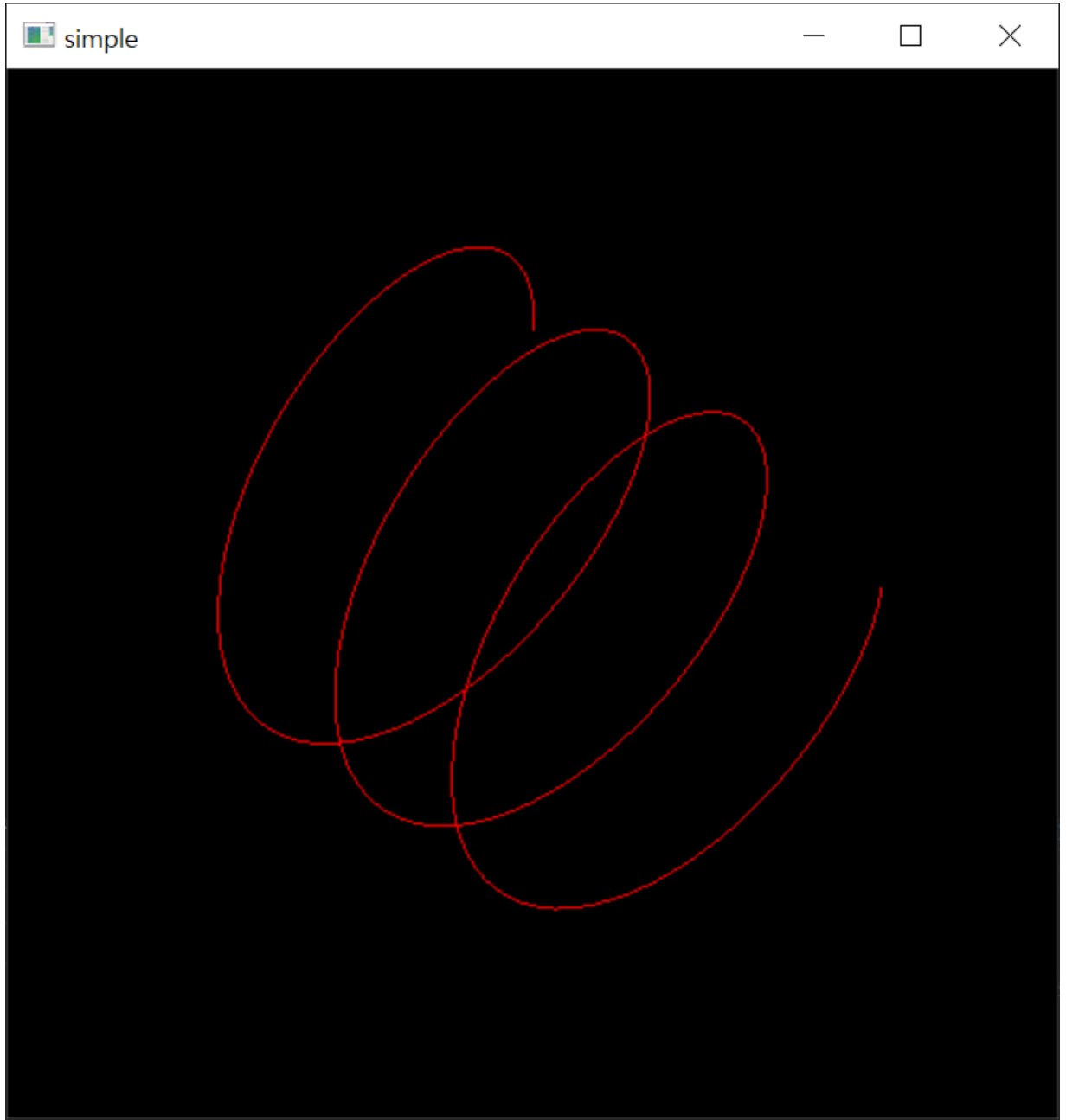
점 그리기(3_1_2a)

- 앞 예제에서 각 point를 중심으로 하는 정사각형들을 그리 시오
  - 지금까지 배운 것만 이용

선 그리기(3_4)

Try

- **GL_LINE_STRIP** 을 이용함
  - glBegin(GL_LINE_STRIP)
  - 점들을 서로 연결해 줌
- 해보기!
  - hint) 3_1_2 예제와 아~주 유사함



선 그리기(3_4_1)

- 일반적인 직선
  - glBegin(**GL_LINES**)

- 선 두께 설정

```
void glLineWidth(GLfloat 두께);
```

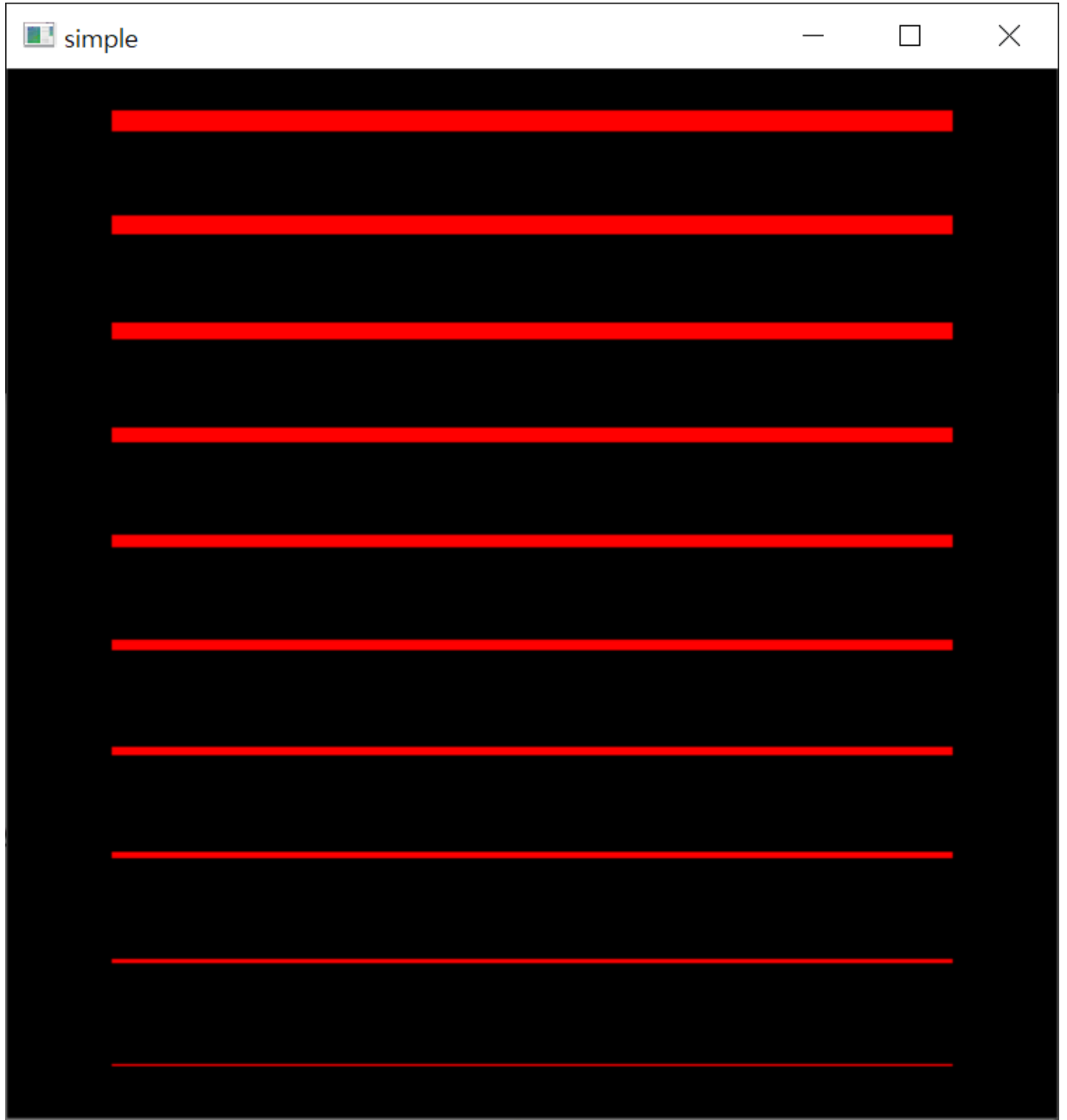
- 지원되는 선의 두께 범위와 단계를 얻어야...어느 범위내에서 두께를 설정해야 하는지 알아야...
  - 시스템에서 지원되는 선 두께의 범위(size), 선 두께의 최소 증가치(단계 값, step)

```
GLfloat sizes[2];   // 지원되는 선의 두께 범위를 저장
GLfloat step;       // 지원되는 선 두께 단계를 저장

// 지원되는 선 두께와 범위를 얻는다.
glGetFloatv(GL_LINE_WIDTH_RANGE,sizes);
glGetFloatv(GL_LINE_WIDTH_GRANULARITY,&step);
```

선 그리기(3_4_1)

- code

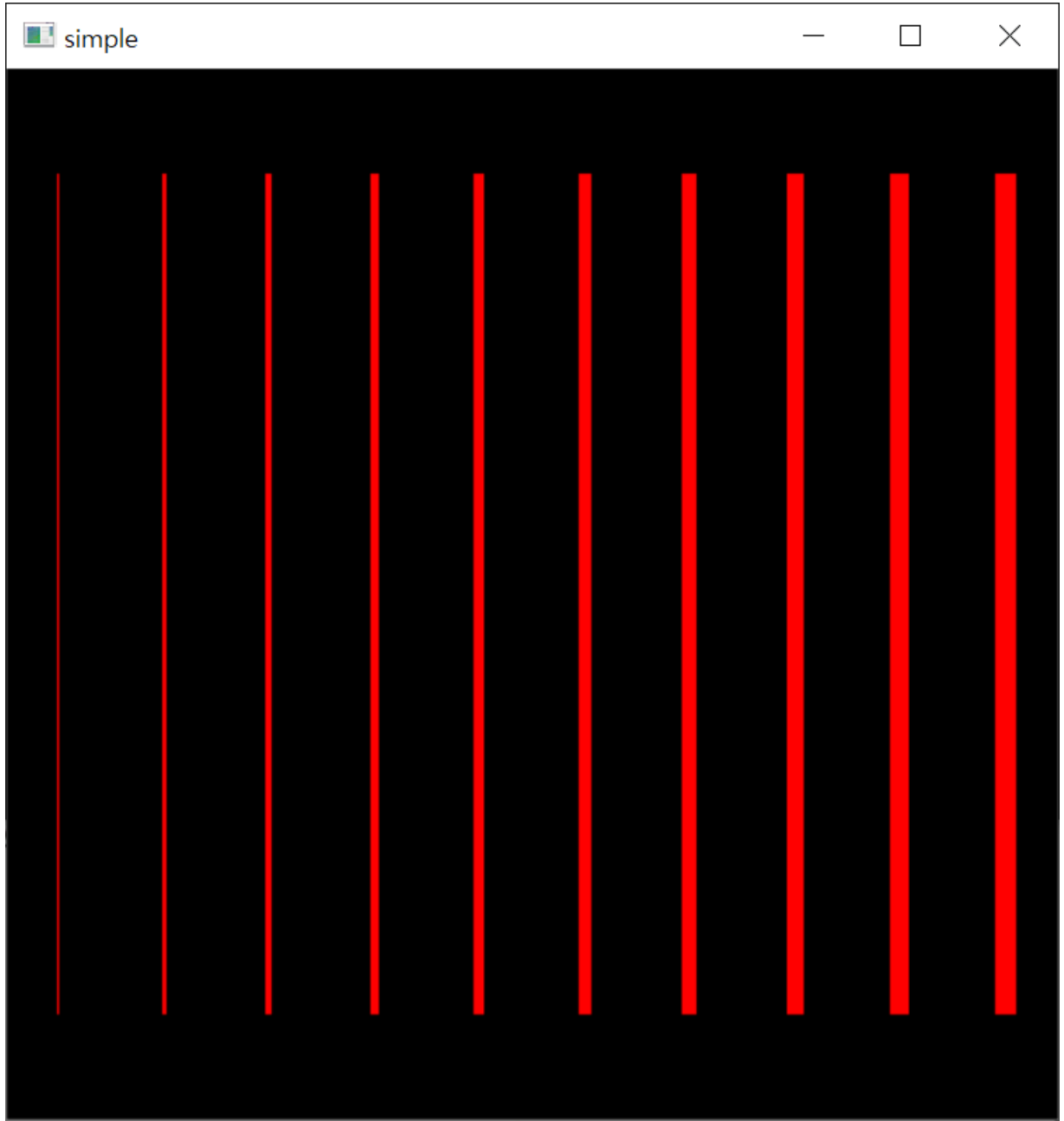
- 2D 고려



선 그리기(3_4_2)

Try

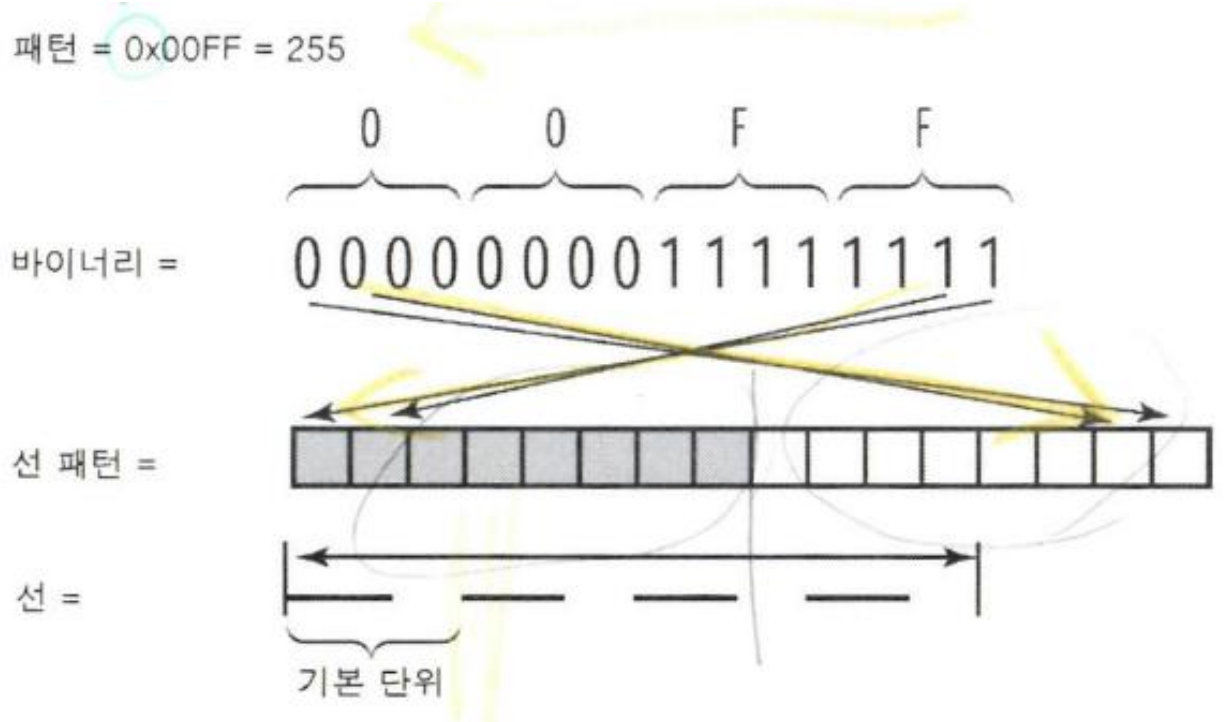
- 세로로 그려보자!



점선 그리기(3_5)

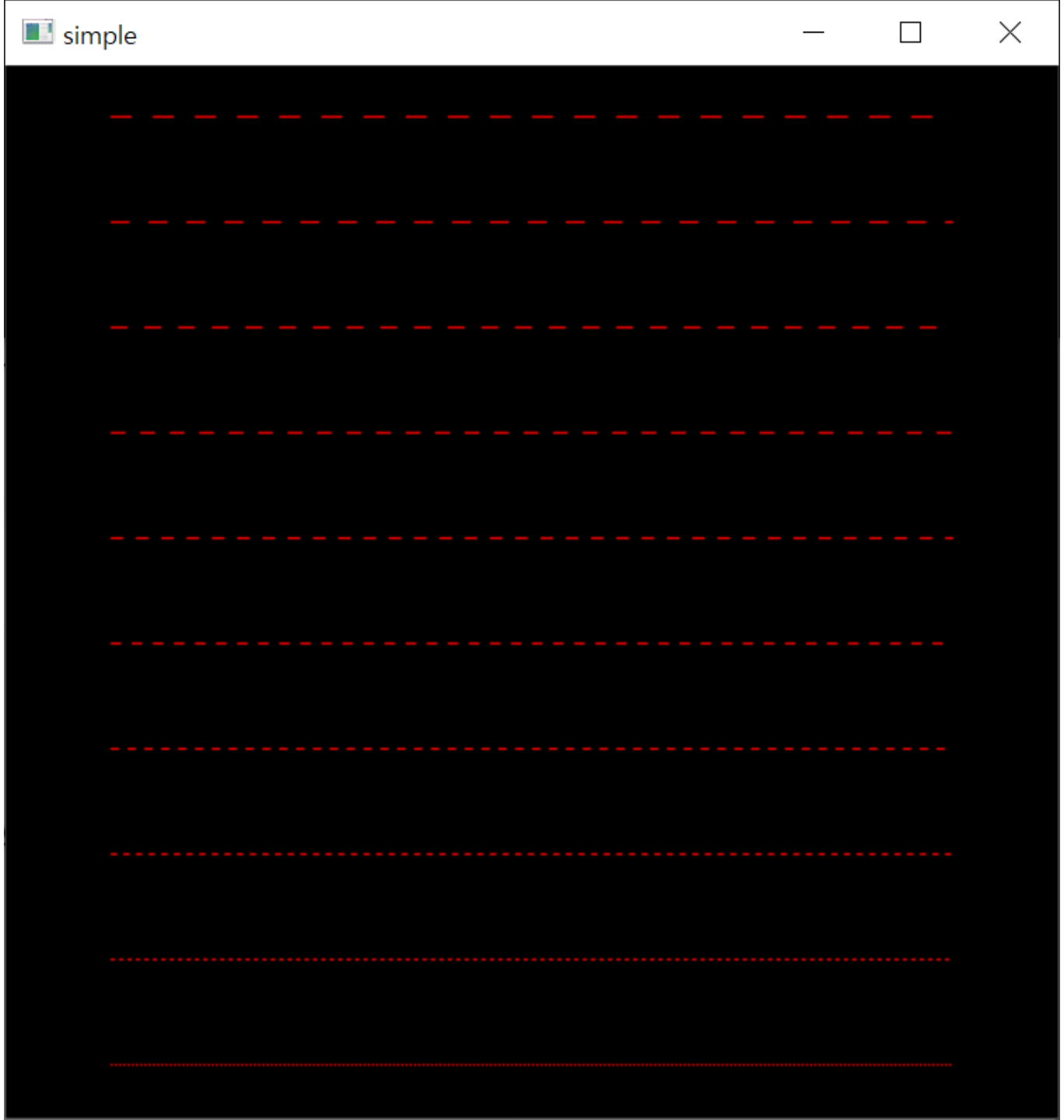
- glEnable (**GL_LINE_STI PPLE**) 를 호출하여 드로잉에 사용할 선의 패턴을 지정
  - glEnable로 사용 가능하게 설정한 기능은 glDisable로 해제 할 수 있음

- glLineStipple(Glint 팩터(factor), Glushort 패턴(pattern))
  - pattern의 1 bit가 그려지는 픽셀 수
    - ✓ factor 픽셀이 그려지고 안 그려지고
  - 패턴
    - ✓ 16bit
    - ✓ 각 비트는 선의 특정 부위가 그려질지 않을지 결정
    - ✓ 패턴의 너비를 조절
    - ✓ bit와 대응되는 선이 반대
      - OpenGL 내부적으로 bit를 해석하는 구조 차이 때문



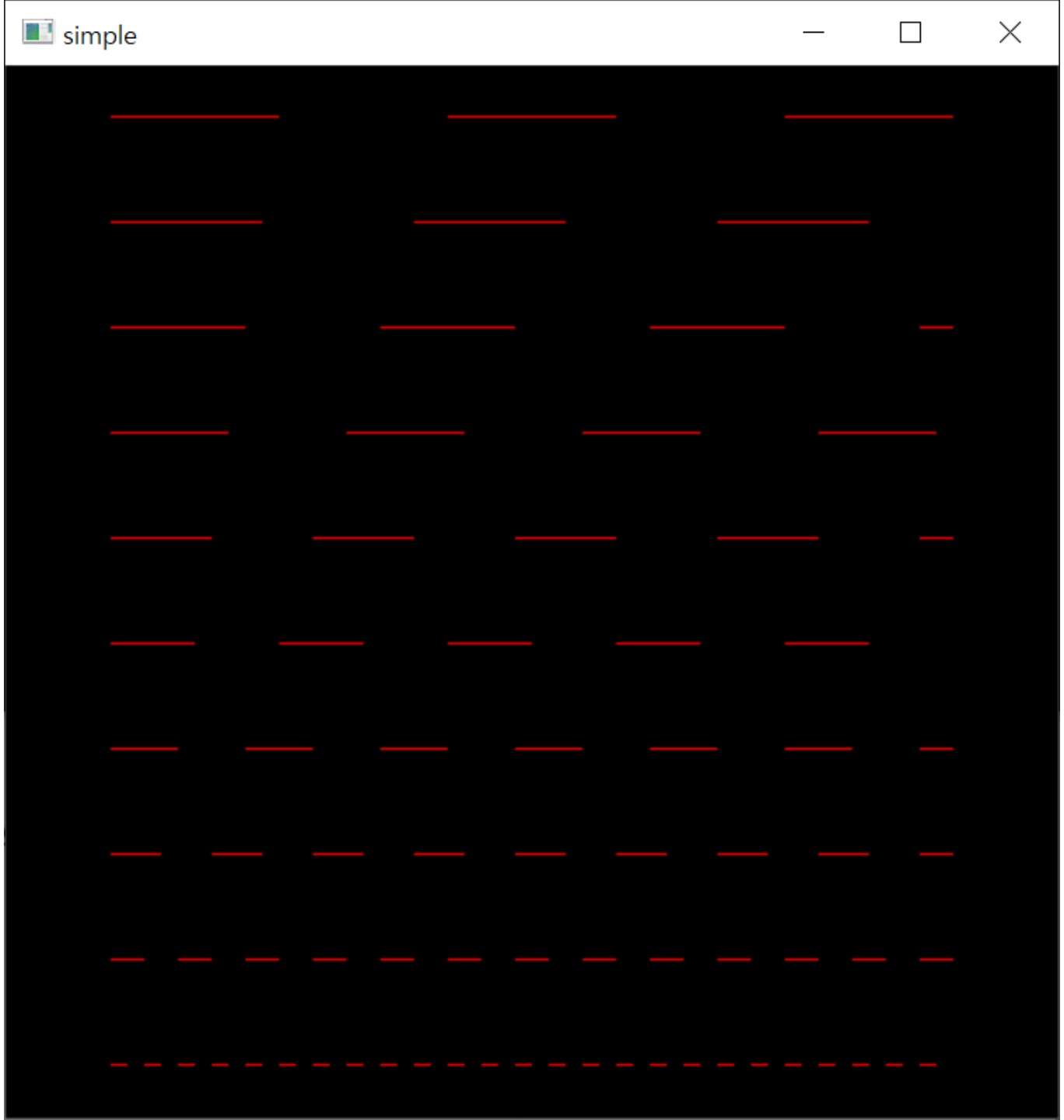
점선 그리기(3_5)

- code



점선 그리기(3_5_1)

- 점선을 더 길게 해보자!



Try

점선 그리기(3_5_1)

- 점선을 더 길게 해보자!
  - `GLushort pattern = 0x00ff;`