기초 컴퓨터 그래픽스

HW1 README

20191559 강상원

1. [환경 명세]

- Windows 10 64bit, Intel® Core™ i5-8257U CPU, Intel® Iris® Plus Graphics 645, Visual Studio Community 2022 Release x64

2. [요구사항]

(a) 윈도우 화면

- 구현 여부: 예

- 작동 확인 방법: 프로그램을 실행하면 윈도우가 화면에 뜬다.

- 구현 방법 : `void main()` 함수에서 `glutInitWindowSize(750, 750);`으로 가로, 세로 각각 750 픽셀의 윈도우를 정의하고 `glutInitWindowPosition(500, 200);`을 통해 (500, 200) 위치에 윈도우를 띄운다는 사실을 알 수 있다.
최종적으로 `glutCreateWindow(program_name);` 함수로 프로그램 제목을 작업 표시줄에 띄운 창이 생성된다.

(b) 선분 그리기

- 구현 여부: 예

- 작동 확인 방법: 프로그램 실행 시 뜬 화면에 한 끝점은 파란색, 한 끝점은 흰색인 빨간색 선분이 그려져 있다.

- 구현 방법: `void draw_line(float px, float py, float qx, float qy)` 함수에서 `glVertex2f()`으로 선분의 양 끝 점 좌표를 정의한다. 좌표값은 각각 `(px, py)`, `(qx, qy)` 형태로 변수로 선언되어 있다. 이후 `display()` 함수에서 `draw_line()`이 호출되어 그려진다.

(c) 비대칭 다각형 그리기

- 구현 여부: 예

- 작동 확인 방법: 프로그램 실행 시 무게중심점은 파란색, 다른 꼭짓점은 흰색으로 그려진 비대칭 다각형 (6각형)이 우측 하단에 그려져 있다.

- 구현 방법: `void draw_line(void)` 함수에서 `glVertex2f()` 함수로 `object[6][2]`에 있는 좌표값대로 선을 그린다. 무게중심은 각 꼭짓점의 `x, y` 좌표값을 평균 내어 구하였다. 이후 `display()` 함수에서 `draw_object()`가 호출되어 그려진다.

(d) 선분 Rotation 변환 기능

- 구현 여부: 예

- 작동 확인 방법: 마우스 스크롤 휠을 스크롤 하는 방향에 따라 선분이 파란색 꼭지점을 중심으로 각각 반시계 방향, 시계 방향으로 회전한다.

- 구현 방법: `void mousewheel(int wheel, int direction, int x, int y)` 함수에서 스크롤 direction에 따라 `(qx, qy)`의 좌표를 아핀 변환을 통해 이동한다. (px, py)를 중심으로 회전하기 때문에 먼저 `(px, py)`만큼 Translation 변환을 통해 중심을 옮긴 다음, `cos()`, `sin()` 함수를 활용해 Rotation 변환을 기술하였다. 이후 다시 `(px, py)`만큼 이동하여 스크롤 할 때 파란색 꼭짓점을 중심으로 선분이 회전하게 구현하였다.

(e) 파란색 꼭지점 이동 (Picking 기능)

- 구현 여부: 예

- 작동 확인 방법: SHIFT 키를 누른 상태에서 파란색 꼭지점을 왼쪽 마우스 버튼을 클릭하여 이 점을 다른 곳으로 이동시킬 수 있음을 확인할 수 있다.

- 구현 방법: 먼저 마우스 좌표계와 화면 좌표계에 차이가 있기 때문에(y축 방향, 원점 위치, 거리) 이를 변환시켜주는 `xCoord(float x), yCoord(float y)` 함수를 작성하였다. `mousepress(int button, int state, int x, int y);` 함수에서 마우스 왼쪽 버튼이 눌린 상태이며, 그 좌표가 파란색 꼭지점임을 확인한다. (정확하게 파란색 꼭지점을 클릭하기 힘드므로 적당히 작은 좌표 범위를 설정해 주었다. `mousemove(int x, int y);` 함수에서 앞서 확인한 왼쪽 마우스 조건과, SHIFT 키가 눌린 조건을 확인하여 둘 다 조건을 충족시킬 때, 마우스를 움직여 마우스 좌표와 `(qx, qy)` 좌표가 연동되게 한다. SHIFT 키를 누르지 않은 상태에서는 해당 기능이 작동하지 않는다.

(f) 다각형 마우스 이동 (Translation 변환 기능)

- 구현 여부: 예

- 작동 확인 방법: ALT 키를 누른 상태에서 오른쪽 마우스 버튼을 클릭하여 움직이면 다각형이 마우스의 움직임과 같은 속도와 방향으로 이동한다. (커서가 다각형 영역에 있지 않더라도 작동)

- 구현 방법: `mousemove(int x, int y);` 함수에서 마우스 오른쪽 클릭 상태와, ALT 키 상태를 확인하여 조건을 동시 만족할 때 마우스의 움직임과 같은 속도와 방향으로 이동하게 하였다. 이 때, 마우스 클릭 지점으로 다각형이 순간이동하지 않고, 단지 같은 속도와 방향으로 이동하게 하기 위해서는 미리 선언한 `float prev_mx, prev_my` 변수에 원래 다각형의 무게중심 좌표를 기록하여야 한다. 마우스가 이동한 `dx, dy` 값을 다각형 좌표에 더해줌으로써 다각형이 이동하는 기능을 구현하였다. 앞서 구현한 대로 ALT 키를 누르지 않은 상태에서는 해당 기능이 작동하지 않는다.

(g) 다각형 Scaling 변환 기능

- 구현 여부: 예

- 작동 확인 방법: CTRL 키를 누른 상태에서 오른쪽 마우스 버튼을 클릭하여 왼쪽으로 움직이면 커서가 윈도우에서 움직인 양에 따라 다각형이 자신의 무게 중심점을 기준으로 크기가 작아진다. 반대로 오른쪽으로 움직이면 그 크기가 커진다.

- 구현 방법: 마찬가지로 `mousemove(int x, int y);` 함수에서 마우스 오른쪽 클릭 상태와, CTRL 키 상태를 확인하여 조건을 동시 만족할 때 마우스의 좌우 움직임에 따라 다각형의 크기가 변화하도록 하였다. 그냥 Scaling 변환을 하면 원점을 중심으로 작아지고, 커지기 때문에 먼저 무게중심점을 Translation 변환을 통해 원점으로 이동시키고, 마우스 최초 클릭 지점의 x 좌표와 현재 마우스 x 좌표의 차인 `dx`만큼 Scaling 변환을 하였다. 앞서 구현한 대로 CTRL 키를 누르지 않은 상태에서는 해당 기능이 작동하지 않는다.

(h) 추가 구현

- 구현 여부: 예

- 구현 내용: 귀여운 모양의 피카츄를 화면 좌측 하단에 도시되어 있다.
SHIFT 키를 누른 상태에서 임의의 좌표에 오른쪽 마우스 클릭을 한 채로 움직이면 피카츄가 해당 위치로 순간이동한 후, 마우스를 따라 움직인다. (전광석화; Translation 변환), ALT 키를

누른 상태에서 마우스 휠을 회전하면 스크롤 양만큼 Shearing 변환과 Scaling 변환이 이루어져 피카츄가 뒤틀리며 커지고 작아진다.

- 작동 확인 방법: 화면 좌측 하단에 피카츄 모양이 그려져 있다.
SHIFT 키를 누른 상태에서 임의의 좌표에 오른쪽 마우스 클릭을 한 채로 움직이면 피카츄가 해당 위치로 순간이동한 후, 마우스를 따라 움직인다.
ALT 키를 누른 상태에서 마우스 휠을 회전하면 스크롤 양만큼 피카츄가 뒤틀리며 커지고 작아진다.

- 구현 방법: `float pikachu[147][2]` 배열을 선언하여 피카츄 모양의 각 147개 꼭지점 좌표들을 집어넣었다. `pikachu_center_x, pikachu_center_y` 값을 각 좌표값에 더해 피카츄 중심점의 위치를 나타내었고, `pikachu_size`를 곱해 Scaling 변환이 가능케 하였다. 마지막으로 float `pikachu_sh_x` 변수를 통해 x-shearing의 sh_x 값을 나타내었다.
`void draw_pikachu()` 함수에서 void `draw_pikachu_line(int st, int n, float r, float g, float b, int mode=GL_LINE_STRIP);` 함수를 호출하여 피카츄 각 몸통, 눈, 볼, 귀 모양을 그려주었고 색깔 값과 `mode` 값 변화로 안의 색칠 여부와 선의 색깔을 결정하였다. `draw_pikachu_line()` 함수에서는 `glVertex2f()` 함수로 피카츄의 선분들을 그려주었고, 여기서 Translation, Shearing, Scaling 변환이 구현되었다.