|  |  |  |
| --- | --- | --- |
| 이름 | 백성빈 | 손주은 |
| 학번 | 20170680 | 20170702 |
| 학과 | 컴퓨터공학과 | 컴퓨터공학과 |
| ID | sungbin0515 | jeson |

CSED 451 Computer Graphics—Assn2

**Assn 2: 피카츄 배구**

ASSN 1에서 만들었던 2D 피카츄 배구 게임을 개선하는 것을 목표로 한다.

# 개발 환경:

**IDE:** Visual Studio 2017 community

**Compile mode (VS 2017):** Release x86

**Dependency**: OpenGL / GLEW 2.1.0 / Freeglut 3.0.0

# 요구 사항

## Ball

* 공 가장자리 안쪽을 따라 전기가 흐르는 것을 표현한다. (전기는 공의 내부에서 이동한다.)
* 아래 예시를 따르거나 다른 2레벨 이상의 계층 구조를 사용한다. (선을 이용하여 그린다.)
* 전류가 자연스럽게 흐르는 것처럼 표현하지 않아도 된다.

## Cloud

* 좌우로 이동하며 모양이 변화하는 구름을 만든다.
* 2개 이상의 구름을 만든다.
* (예시) 큰 적색 원을 기준으로 실선 하늘색 원들은 고정되어 있으며, 작은 적색 원을 기준으로 점선 하늘색 원들이 주어진 각도 내에서 회전할 수 있다.
* Cloud를 구성하는 Component가 계층구조를 가지도록 구현해야 한다.

## Character

* 귀와 꼬리를 표현한다. (꼬리는 2단계 이상일 것)
* 좌우로 움직일 땐 귀와 꼬리가 고정되어 본체의 움직임을 따라간다.
* 공과 충돌 시 귀와 꼬리 각 축에 왕복 회전을 주어 떨림을 표현한다.
* 꼬리와 귀는 플레이어의 위치(왼쪽)에만 구현해도 무방하다.
* 충돌 검사는 Assn1과 마찬가지로 꼬리와 귀를 무시한 경계 박스를 이용해 구현한다.

## Restart

* 게임 도중에 R 버튼(혹은 r버튼)을 누르면 점수와 공의 위치를 리셋 하는 재시작 기능을 추가.

# **프로그램 설계**

Assign 1번에 구현한 내용은 작성하지 않았으며, 새로운 기능 혹은 개선된 사항만을 작성한다.

1. **Anchor**

물체의 회전을 나타낼 때, 계층 구조를 활용해서 나타내기 위해 새로 작성한 Field를 의미한다. 물체의 상대 회전축을 나타낸다.

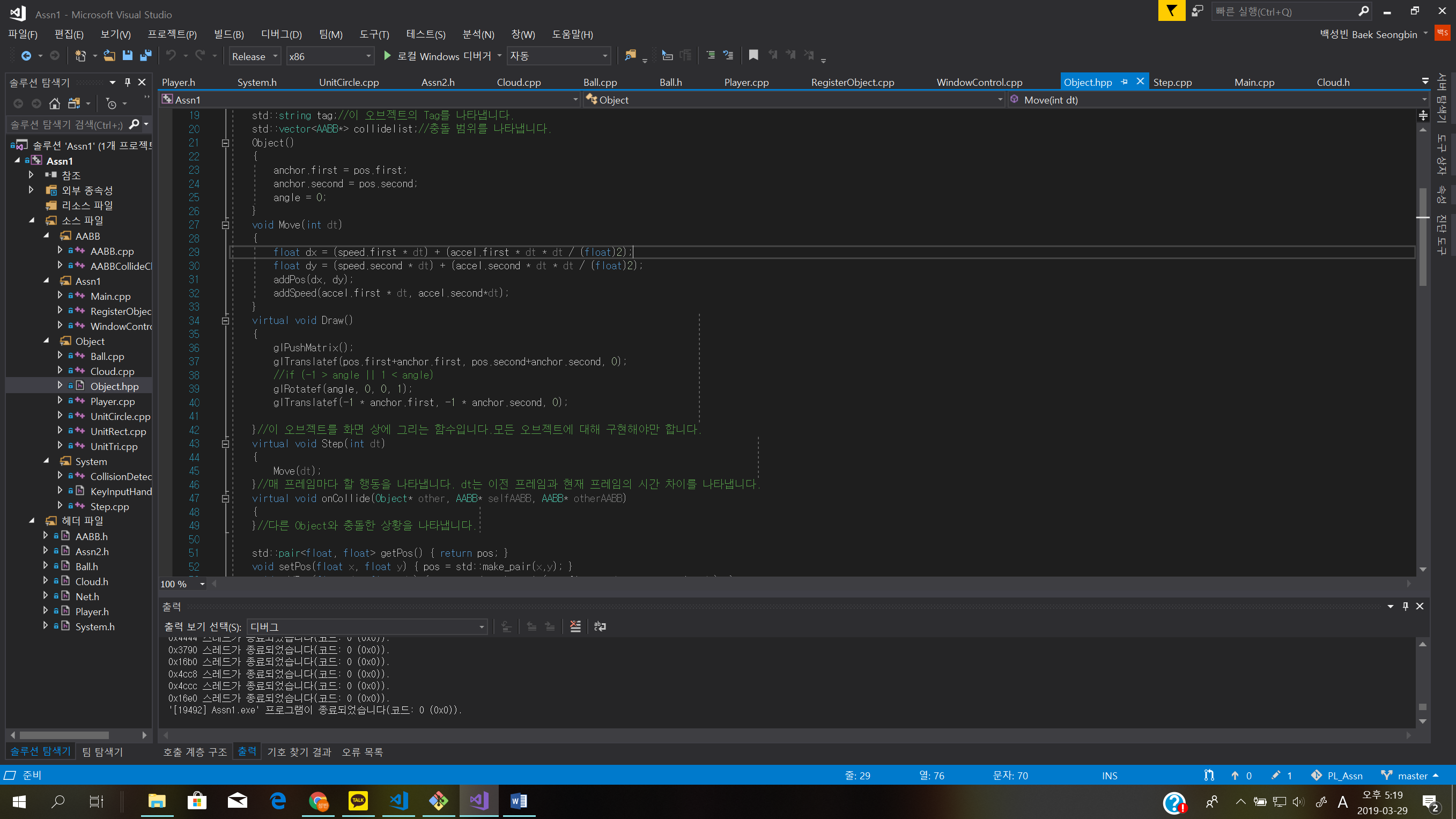


Figure 1 각 Object를 그리는 Method의 Base가 되는 Method

구현 방법은 다음과 같다: 현재까지의 Transform matrix를 저장하고, anchor의 절대 위치로 좌표축을 이동한다. 그리고 anchor에서 좌표축을 angle만큼 회전시키고, 물체의 절대위치로 좌표축을 이동한다.

1. **ChildList(Hierarchical structure)**

계층 구조를 편리하게 나타내기 위한 data field이다. Object를 상속한 Class를 담을 수 있는 Container로, GameManager에 Object를 등록하고 오브젝트의 setParent / addChild를 호출해서 상속 관계(부모/자식 관계)를 형성할 수 있다.

부모 Object의 Step(int dt) / Draw() 함수가 호출되면 각자 자신의 ChildList 를 순회하면서 자식 Object의 Step(dt), Draw() 함수를 호출한다. 이를 통해 계층 구조에 따른 이동, 회전 등을 수월하게 나타낼 수 있다.

1. **Unit shape class 추가**

구름이나 꼬리와 같이 복잡한 도형을 그릴 때 하나 이상의 도형이 필요한 경우에 반복적인 일을 최소화할 수 있도록 UnitTri, UnitCircle 등의 unit 도형 class를 추가하였다.

* 1. UnitTri: position과 size에 부합하는 직각삼각형을 그리는 클래스이다.
  2. UnitCircle: position과 radius에 따라 원을 그리는 클래스이다.

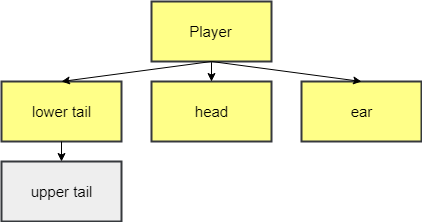
1. **공의 전기**
   1. 공은 기존의 움직임을 유지하고 그와 별개로 원의 중심을 기준으로 회전하는 전기가 있어야 한다. 공의 일부를 따라 이동하는 지그재그 모양의 노란 선을 정의하고, 이 선이 공의 중심 위치를 기준으로 회전하도록 구현했다.
   2. 다른 오브젝트와 달리 Class 간의 계층 구조는 없지만, 공의 위치를 기준으로 상대적인 위치를 파악해 그 위치에 전기를 그리도록 구현했으므로 전기와 공 사이에는 계층 구조가 있다고 볼 수 있다.
2. **구름**
   1. 각 구름은 좌우로 천천히 움직이고, 그와 동시에 구름을 이루는 Component의 움직임 또한 있어야 한다.
   2. Assn 요구조건에서 첨부된 예시의 구름과 최대한 비슷하게 표현했다. Cloud class내에서는 정삼각형의 세 꼭지점을 중심으로 하는 원의 위치는 상대적으로 고정되어 있고 나머지 양 옆의 ‘날개’ 구름들은 매 frame마다 직선 경로를 따로 왕복 translation을 하게 된다.
3. **꼬리와 귀**
   1. 꼬리와 귀 모두 한 개 또는 두개의 UnitTri Object를 이용해 나타낼 수 있다. UnitTri Class는 자신의 위치를 중심으로 하는 직각삼각형을 그린다. 공과 충돌할 때 이 직각삼각형의 세 점을 단순 직선 이동의 연속으로 구현해서는 안된다. 따라서 우리는 기준점을 중심으로 좌표축을 회전시켜 정말 기준점을 축으로 회전하는 애니메이션을 구현했다.
   2. 꼬리를 2단 계층 구조로 나타내기 위해 아래와 같이 lower tail(노란색)을 player의 child list에 추가하고, upper tail(검정색)을 lower tail의 child list에 추가했다.
4. **재시작 기능**

Figure 2 Player 의 계층 구조

‘R’ 또는 ‘r’을 누르면 게임 점수와 공의 위치 등이 초기화됨으로써 게임이 다시 시작된다.

# **추가기능**

1. **중력**

* 공 Object에게 아래 방향으로 가속도를 지속적으로 부여해서 마치 중력의 영향을 받아 아래 방향으로 가속하는 것 같은 움직임을 구현하였다. 이를 통해 핑퐁처럼 직선 운동을 하는 배구에서 벗어나 진짜 공을 가지고 하는 배구와 유사한 사용자 경험을 제공할 수 있다.

# **시연**

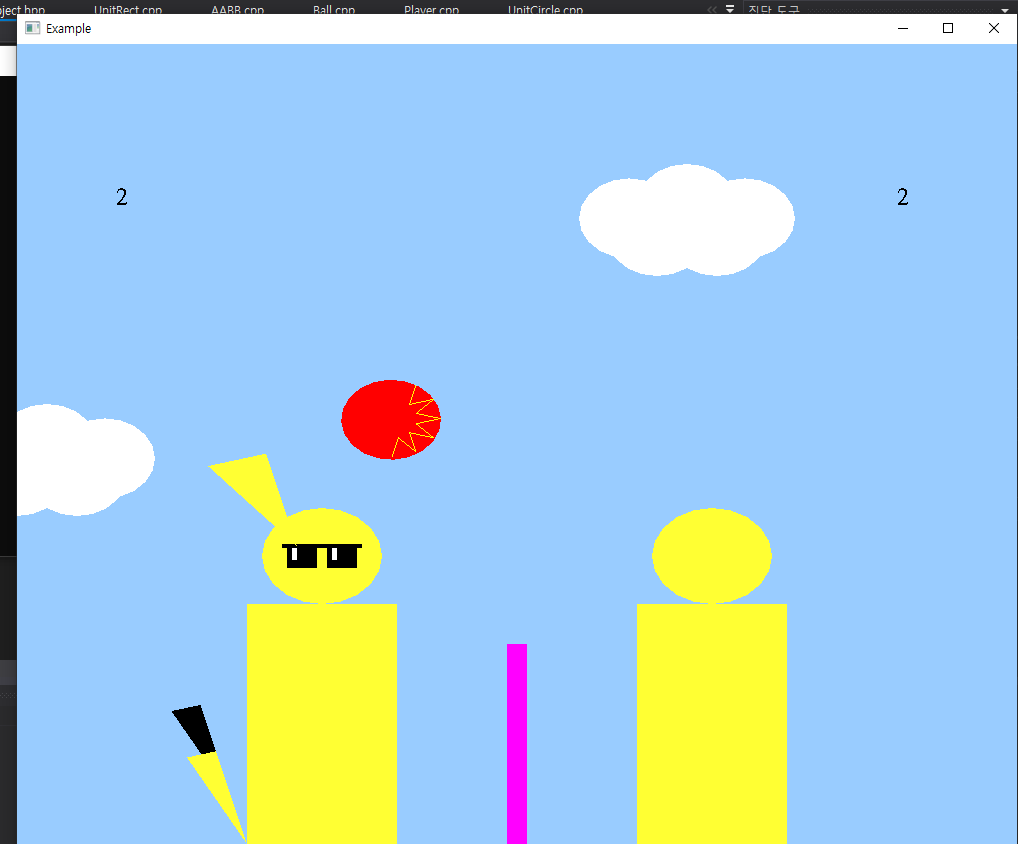


Figure 2 게임 화면이다. 구름, 꼬리, 귀, (공에 흐르는) 전기, 선글라스 등이 추가된 것을 확인할 수 있다.

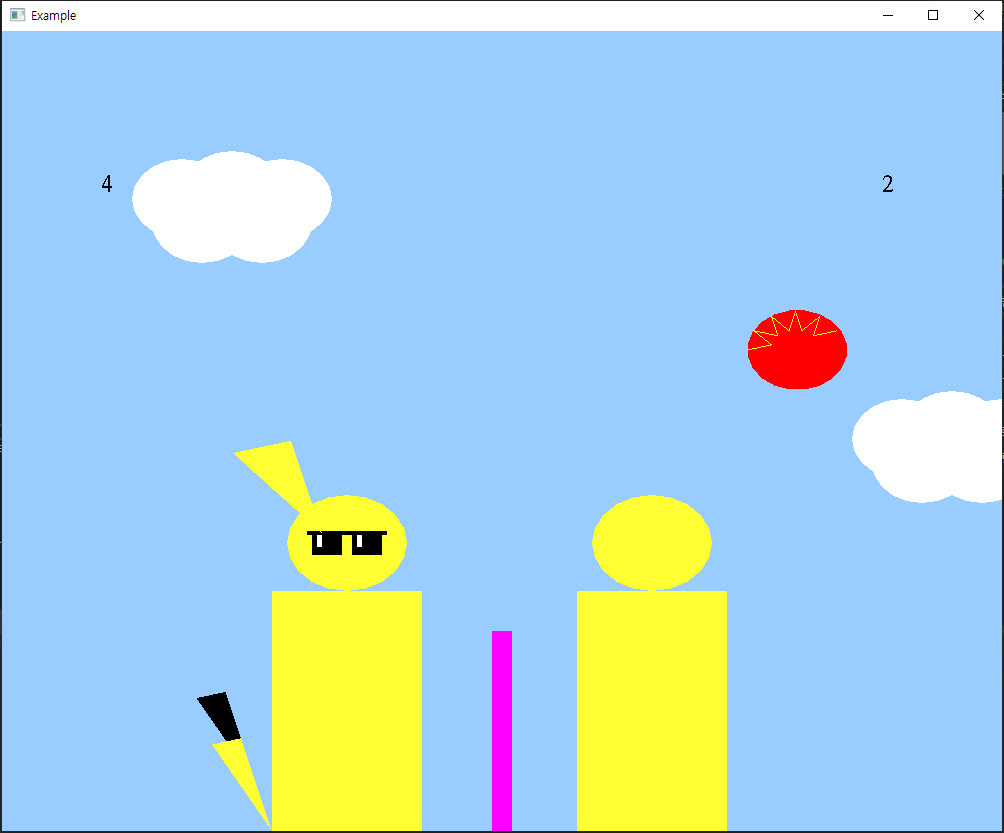


Figure 3 시간에 따라 구름과 전기의 위치와 모양이 변하는 것을 확인 할 수 있다.

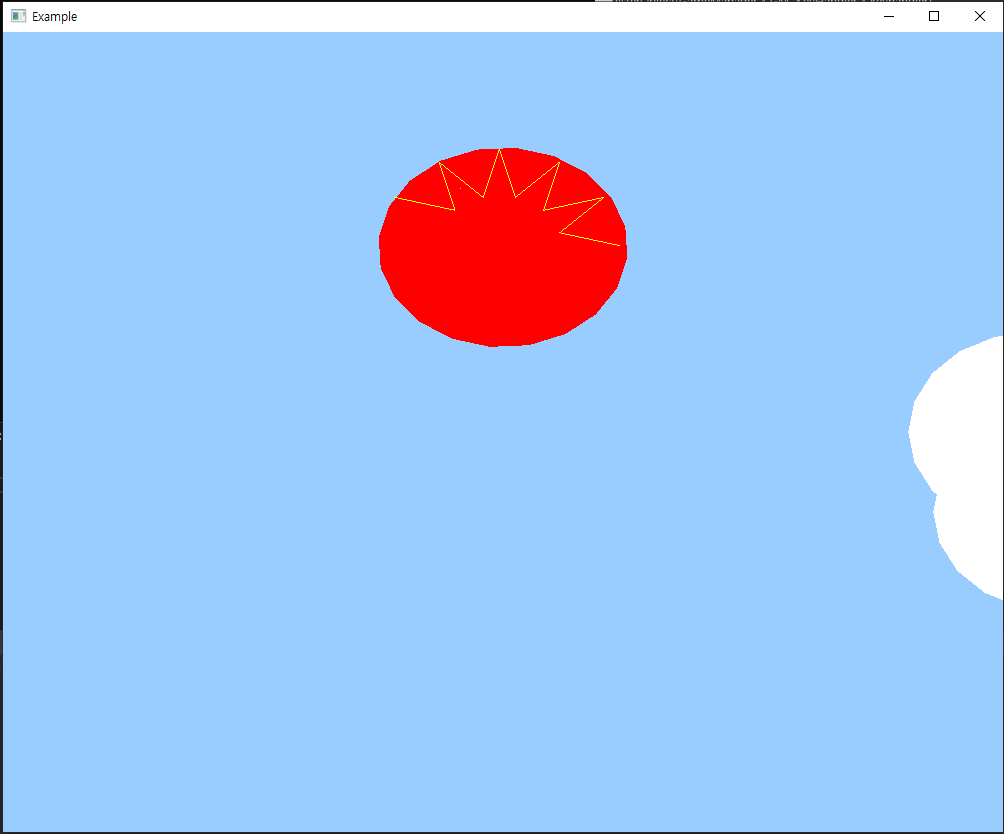


Figure 4 Zoom mode(‘x’키 눌렀을 때)에서 공의 중심을 기준으로 회전하는 전기의 모습이다.

# **토론**

처음에는 귀와 꼬리 등의 회전 운동을 구현할 때 object마다 개별적으로 draw 함수 내에서 rotate을 했으나 원하는 대로의 회전을 구현하기 힘들었다. 하지만 Structure를 정리해서 회전하는 각도를 정해주는 부분과 draw하는 부분을 따로 나눔으로써 계층 구조를 좀 더 잘 부각시킬 수 있었고, Assn에서 원하는 형태의 축 회전을 구현할 수 있었다.

# **결론**

이번 과제에서는 기존의 Game Object에 계층 구조를 추가함으로써 전보다 더 동적인 2D 애니메이션을 구현할 수 있었다. 물체 자체의 움직임 뿐만 아니라 상위 object를 기준으로 한 하위 object의 transformation을 나타낼 수 있었다. (Hierarchical structure를 효과적으로 구현할 수 있었다)

# **개선 방향**

현재는 피카츄를 10개도 안되는 매우 적은 수의 도형만을 사용해서 구현해 저급한 그래픽만을 그렸지만 조금 더 많은 도형을 사용해서 조금 더 피카츄 같은 캐릭터를 만들 수 있을 것이다.

또한 현재는 전기 흐르는 모습이 그냥 공이 돌아가는 모습처럼 보이지만, 정말 전기가 공을 따라 회전하는 것 처럼 개선할 수 있을 것이다.