CSED232 객체지향프로그래밍 2018S

Assignment 2

Inheritance: Circle Renderer

Objective

본 과제에서는 Frame 의 pixel 에 RGBA color space 로 여러 형태의 Circle 을 rendering 하는 프로그램을 구현한다. Frame class, Circle class 와 그의 subclasses 들로 Inheritance 에 대해 익힐 수 있다.

Background – Pixel, subpixel, RGB and RGBA

Figure 1. High and low resolution image
Photo by Wexor Tmg on Unsplash

Pixel(Picture element)은 화면을 표현하는 물리적인 단위로 디스플레이에서 접근 가능한 가장 작은 점과 같다. 여러 픽셀들이 모여서 한 이미지 혹은

화면이 이루어진다. 해상도는 얼마나 많은 픽셀이 한 화면을 구성하는데에 필요한지 그 정도를 나타낸다. Figure 1 에서 위의 사진이 아래의 사진보다 훨씬 높은 해상도를 가진다. 디스플레이의 경우 HD(High-definition), UHD, QHD 등으로 해상도를 표현하며 이미지 단위에서는 ppi(pixels per inch)로 해상도를 나타내고 비교할 수 있다.

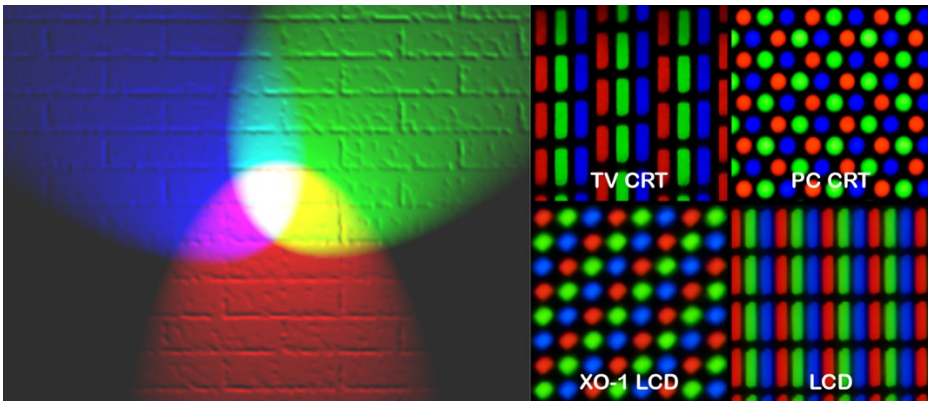


Figure 2. RGB color and various subpixels

빛의 3 원색은 Red, Green, Blue 로 이들의 가산 혼합을 통해 여러 가지 색을 표현할 수 있다. 디스플레이의 기술적 특징으로 인해 여러 color channel 을 한 번에 표현하기 어렵기 때문에 각 pixel 들은 여러 색의 subpixel 로 이루어져있다. Subpixel 들의 조합은 해당 디스플레이의 품질을 결정한다. Pixel 의 모양이 정사각형으로 정해져있지 않듯이 subpixel 들의 순서나 개수, 그 조합도 다양하다.

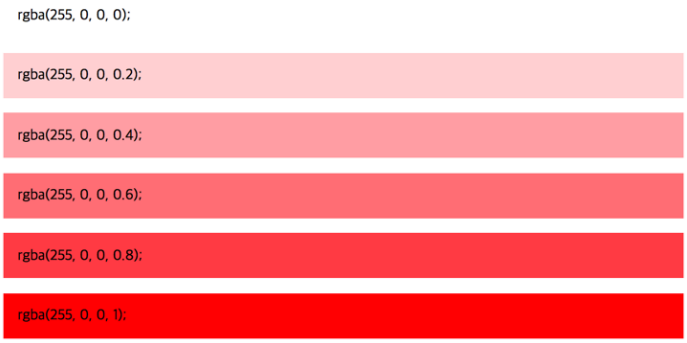
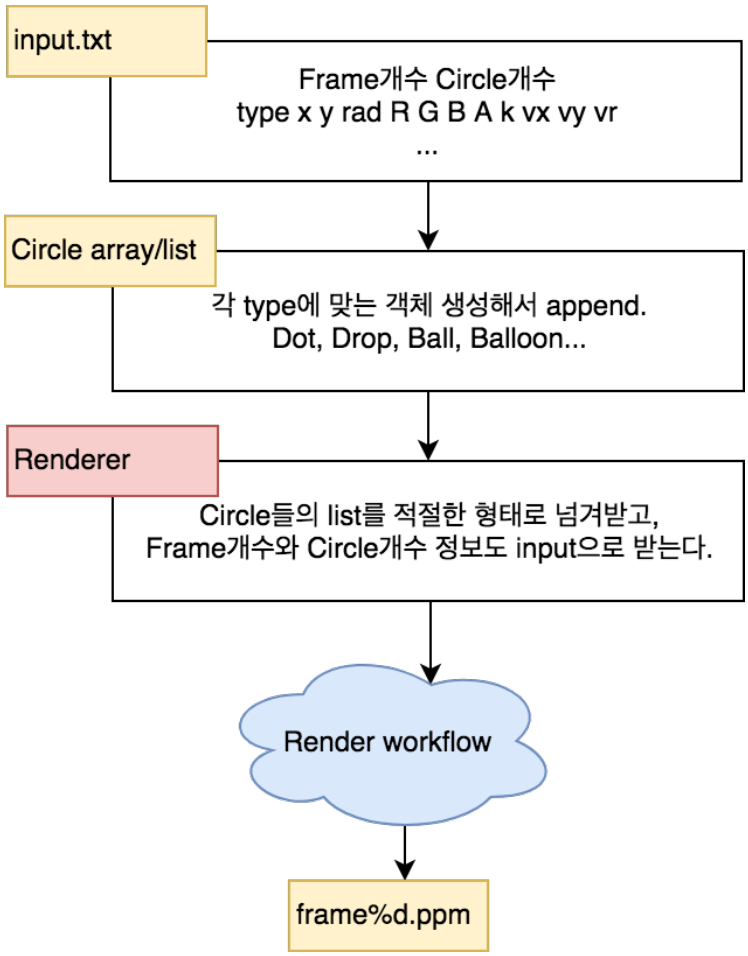


Figure 3. RGBA color example

컴퓨터 그래픽스가 발전하면서 보다 다양한 표현을 위해 기존의 R, G, B, channel 에 Alpha channel 을 더해 투명도를 조절할 수 있게 되었다.

Explanation (1) – Overview



Explanation (2) – Circle Renderer workflow and frame structure

<Renderer>

Circle array, Number of circles, Number of frames 를 입력받는다.
Frame array 는 frame 개수에 맞게 renderer 내에서 생성해서 render 한다.
불필요한 memory 는 해제해준다. 각 frame 내에서의 행동은 아래 설명을 참고한다.

work flow:

(For initial frame)

Circle 들의 순서대로 각각의 Circle 에 대해 다음을 수행한다.

a. Frame 에서 해당 circle 이 기여하는 부분에 rgba data 를 update 한다.
b. 모든 frame 이 update 된 후 ppm image 를 출력한다.

(For the other frames)

Circle 들의 순서대로 각각의 Circle 에 대해 다음을 수행한다.

a. Frame 경계와 겹치는 부분이 있는지 확인
b. a 의 결과에 따라 circle 을 bounce 혹은 pop 처리한다.(뒤 설명 참조)
c. 해당 circle 의 update method 실행
d. Frame 에서 해당 circle 이 기여하는 부분에 rgba data 를 update 한다.
e. 모든 frame 이 update 된 후 ppm image 를 출력한다.

Renderer 프로그램의 설계는 자유롭게 할 수 있다. 다만 input 이나 output 형식이 과제에 명시된 것과 같아야하며 work flow 를 지켜야한다. 또한 그 내용을 보고서에 자세히 기술하여야 한다. Frame struct 를 ppm image 로 변환하는 부분의 skeleton code 는 주어진다. 해당 코드에서 빈 부분을 구현해서 이용하면 되며 나머지 부분은 수정하지 않는다.

〔Frame〕 **(struct, not class)**

- width, height, 그리고 4*width*height 크기의 float data array 를 member 변수로 가지며 이 때 data array 를 픽셀이 보유한 RGBA 값으로 보면 된다. color 의 초기값은 white, 즉 rgba(255, 255, 255, 1)이다. width, height 은 각각 320, 480 으로 초기화한다.

Explanation (3) – How the circles behave

공통 사항: 각 circle 들에 저장된 정보는 initial frame 에서의 정보이므로 그 다음 frame 들에서 render 될 때마다 update method 로 position 이나 기타 정보들을 갱신해주어야한다. 입력된 x, y, radius, speed 의 초기값들은 모두 양수로 처리된다. update method 에서는 position, radius, speed 를 조건에 맞게 갱신해준다.

주의점: Circle array/list/vector 들에게 Dot, Drop, Ball, Balloon 등을 저장하면 Circle array 에서 access 했을 때 해당 class 의 method 를 바로 사용할 수 없다. 이 경우 type 을 변환하여 사용한다. (Down casting)

<Circle>

- center x, y 좌표와 radius, rgba color 정보를 담고 있다.
- 이를 상속하는 DynamicCircle 과 Dot 이 있다.

<Dot>

- 그 위치나 반지름이 변하지 않는 정적인 Circle 이다.
- erase 멤버 변수가 true 면 eraser 로 작동하며 원래 Dot 을 초기화했던 Color 정보를 무시하고 해당 pixel 의 color 를 검은색으로 (rgba(0, 0, 0, 1) 만든다.

<DynamicCircle>

- DynamicCircle 은 x, y, r 방향에 대한 speed 정보를 가지고 있다. x, y 방향의 speed 는 time 이 변할 때 Circle 이동하는 정도이며 r 방향의 speed 는 Circle 의 크기가 커지는(radius 가 증가) 정도를 의미한다.
- DynamicCircle 엔 세 sub-class Drop, Ball, Balloon 이 있다.

<Drop>

- Drop 은 gravity 라는 data member 를 가진다.
- y 방향 speed 만 유효하다.

<Ball>

- Ball 은 x, y 방향 speed 를 가지고 가속도는 없다.
- Circle 의 어떤 부분이 frame 을 벗어나면 bounce method 를 실행한다.
- Bounce method 에서는 튕겨나온 후의 x, y speed 를 계산해서 set 한다. 탄성계수를 나타내는 coeff 변수와 뒤에 주어질 식을 이용한다.

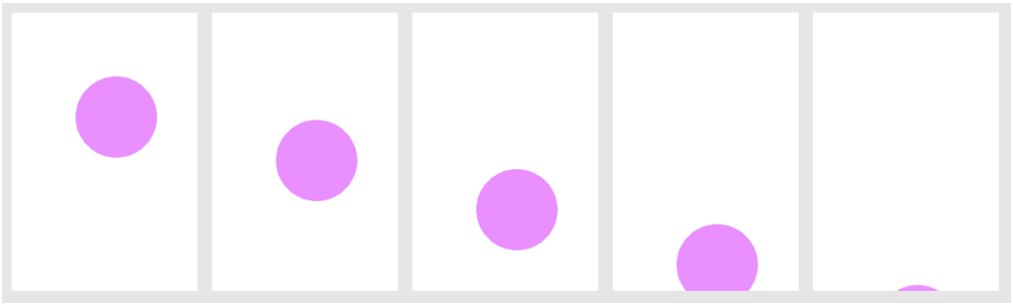
<Balloon>

- Balloon 은 r 방향의 speed 만 가지고 있다.
- popping 이라는 bool variable 을 가지는데 true 면 Circle 의 한 부분이 frame 경계를 침범했을 때 pop method 를 호출한다.
- Pop method 에서는 radius, speed_r 를 0 으로 set 한다.

(case 1; Three Dots)

왼쪽 그림과 오른쪽 그림은 같은 Dot 들을 render 하지만 render 순서가 다르면 결과가 다를 수 있다.

(case 2; Drop)



각각 Time 0, 1, 2, 3, 4 일 때의 모습이다. 시간이 갈 수록 y 방향으로만 움직이며 그 speed 가 gravity 에 의해 증가한다.

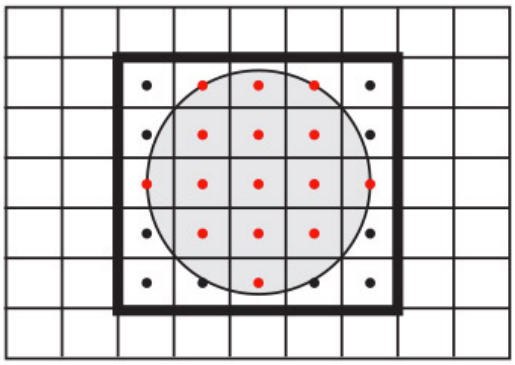
(case 3; Ball)

각각 Time 0, 1, 2, 3, 4 일 때의 모습이다. Ball 은 항상 bounce 하므로 Time 2 에서 Ball 이 Frame 경계로 나오면 그 다음 frame 에서 update 전에 bounce method 가 먼저 실행된다.

(case 4; Balloon)

각각 Time 0, 1, 2, 3, 4 일 때의 모습이다. 위의 경우는 pop 을 하지 않고 아래의 경우는 pop 을 한 경우이다. Time 2 에서 Balloon 이 render 되면 Time 3 에서 Balloon 이 frame 경계에 닿은 것이 감지된다. pop 을 하면 radius 가 0 이 되므로 frame 에서 보이지 않는다. 그리고 r 방향 speed 도 0 으로 초기화되기 때문에 그 다음 frame 에도 나타나지 않는다. Pop 하지 않는 경우 Balloon 은 계속 자라며 frame 을 나가지 않는 선에서 render 된다.

Explanation (3) – Formulas and constraints



Render:
한 Circle 이 어떤 pixel 에 기여하는지 아래와 같이 계산할 수 있다.

(pixelCX – circleCX)^2 + (pixelCY – circleCY)^2 < radius^2

pixel 의 center 는 frame 이 int grid 형식으로 되어있는데, grid[0][0]에 위치하는 경우 (0.5, 0.5)를 pixel 의 center 로 볼 수 있다(그림의 점들이 pixel center).

새로운 r, g, b, a 값의 계산은 다음과 같이 할 수 있다.

*newColor = inputAlpha * inputColor + (1- inputAlpha) * existingColor*

newAlpha = existingAlpha + inputAlpha

inputColor, inputAlpha, existingColor, existingAlpha 는 각각 새로 render 되는 RGB, A 값, 기존에 pixel 에 저장되어있던 RGB, A 값을 의미하며 newColor, newAlpha 는 최종적으로 pixel 에 저장될 RGB, A 값이다. 편의를 위해 통합하여 나타냈는데 r, g, b 값에 대해 각각 따로 계산을 해 주어야 한다.

R, G, B values should be between 0~255
A value should be between 0~1

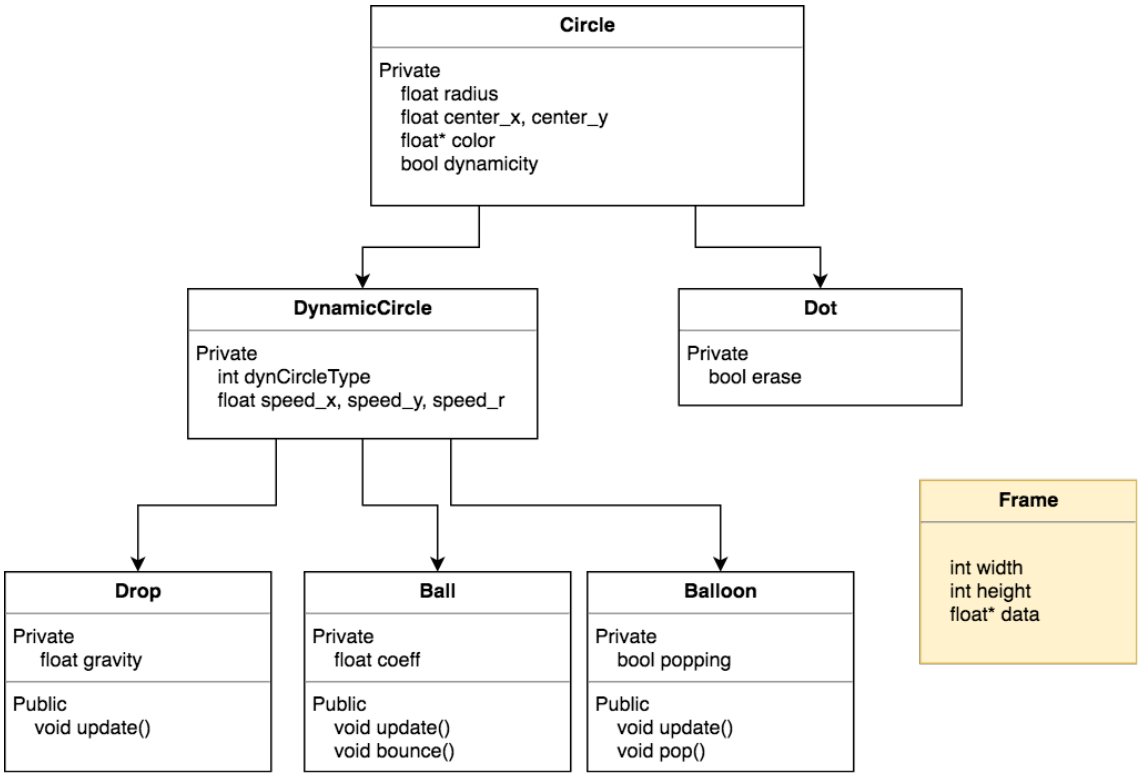
checkConflict:

circle 가장자리가 정확히 frame 경계에 위치하는 경우 conflict 로 보지 않는다. 예를 들어 어떤 circle 이 (10, 10) 위치에 반지름 10 으로 있으면 conflict 이 아니지만 반지름이 10.1 로 벗어나게되면 conflict 으로 간주한다.

Bounce: // Sorry physics lovers

*speed_after = (-1)*coeff*speed_before (for both x, y direction)*

Requirements – Classes and methods



이해를 돕기 위해 Class 들 간의 상속관계와 최소한도로 필요한 variable, mathod 를 표시한 그림을 첨부한다. 그림에 나와있는 것들은 모두 구현되어야한다. **기본적으로 getter, setter method 나 필요한 constructor, destructor 는 구현해야 본 어싸인을 완료할 수 있다.** 그 이외에 기능상 필요한 method 를 구현하거나 data 변수를 추가할 시 보고서에 상세히 기술한다.

Explanation (4) – Input and Output

Input: text file
첫 줄에 frame 수와 circle 수, frame width, frame height 가 주어지며 이어지는 줄들은 각각
(Type, 중심 x, 중심 y, radius, R, G, B, A, k, v_x, v_y, v_r)을 의미한다.
해당 circle type에 관계없는 정보는 무시한다.
예) Type이 Balloon인데 x, y 방향 speed인 v_x, v_y가 입력된 경우 무시.
type – **Dot(0), Drop(1), Ball(2), Balloon(3)**
Output: ppm image (skeleton of the code will be given)

image 상의 x, y 좌표는 왼쪽 아래가 (0, 0) 이며 오른쪽, 위로 갈 수록 값이 증가한다. 즉 width 320, height 480 인 frame 이면 오른쪽 위의 좌표는 (320, 480)이다. ppm image 는 alpha channel 을 무시하고 RGB 정보만 저장하게 된다. 헤더파일과 구현파일이 제공되나 cpp 구현 파일에서 //〔TO DO〕로 주석 처리되어있는 부분을 추가로 구현해서 사용하여야 한다.

Restrictions

input 형식 : input.txt
첫 줄에 frame 개수, circle 개수를 받고 이어지는 줄들에서 circle 개수만큼의 정보가 있다. (type x y rad R G B A k vx vy vr) k 는 Dot, Drop, Ball, Balloon 의 필수 data member 가 된다. 각각 순서대로 circletype, 중심의 x 좌표, 중심의 y 좌표, 반지름, R, G, B, A 값, data 변수, x 방향 speed, y 방향 speed, r 방향 speed 를 의미한다. 각각 circle type 에 맞게 필요한 값들만 constructor 에 넣어 객체를 생성한다.

output 형식 : frame%d.ppm (frame0.ppm, frame1.ppm, …)
i/o 형식이 다르면 채점 상 불이익이 있을 수 있다.
한 frame set 는 모두 같은 width 와 height 를 가져야 하며 각각 320, 480 로 초기화한다. Ppm file 이 조회되지 않는 os 의 경우 online 에 ppm viewer website 를 이용한다.

Polymorphism, Operator overloading 을 사용할 수 없다.

Submissions

AssignmentReadme 를 참조하여 제출 기한에 맞게 제출한다.

단, report 에 아래 질문에 대한 답을 추가로 기재한다.

- writeup question : Downcasting 이 권장되지 않는 이유를 서술하고 이를 안전하게 하려면 본 과제에서 어떻게 코드를 설계해야하는지 논하시오.

채점 기준

linux g++ 4.8 이상에서 compile 할 예정이다.

1. Test case 들에 대해 ppm image 가 일치하는지를 점수로 산정
2. Renderer 프로그램 설계 및 구현 내용
3. Writeup question, 제출 형식, 주석, format

+)
STL 이나 Boost 등의 라이브러리는 사용하지 않는다. (사용할 경우 0 점)
서버에서 작업하는 경우 output file 을 로컬로 가져와서 확인한다.
Report 는 pdf 형식으로 제출