10주차 결과보고서

전공: 수학과,컴퓨터공학과 학년: 3학년 학번: 20181256 이름: 김도현

1.

결과보고서의 중점 내용이 물이 흐르는 알고리즘을 기술하는 것이다. 그래서 코드에 대한 대부분의 설명은 주석 처리를 하여 코드에 나타냈기 때문에 일부 설명은 생략할 수도 있다.

우선 ofApp.h 파일의 헤더파일이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

여기서는 앞으로 물이 떨어지게 할 알고리즘에 사용할 각종 Flag 들과 변수들, struct 배열 구조체가 들어있다. Line\_water struct 구조체는 물이 line을 타고 흐르기 때문에 9주차에서는 선분의 시작점의 x,y 좌표와 끝 점의 x,y 좌표만 저장했지만 여기서는 slope라는 line의 기울기 까지 함께 저장해야 한다. 앞으로 해당 struct들은 water\_line, water\_dot으로 배열을 설정하여 이를 이용할 것이다. 그리고 dot\_water는 물이 흐르기 시작하는 점이기 때문에 그대로 그 점의 x,y 좌표만 저장하면 된다. Num\_of\_line 은 Line의 개수, num\_of\_dot 은 점의 개수이다.

다음은 ofApp.cpp 파일의 setup() 함수이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

우선 초당 프레임을 의미하는 ofSetFrameRate는 기존의 15를 그대로 유지했다. 그리고 기존에 없었던 코드인 ofSetBackgroundAuto를 false로 하여 물이 떨어지는 것을 표현하기 위해 새롭게 도형이 업데이트 되지 않고 기존의 그림에서 계속 덧대어 그릴 수 있게 바꿔주었다. 이후 배경색은 흰색으로 하고 첫 설정이기 때문에 draw\_flag와 load\_flag는 0으로 설정하여 설정 키인 ‘d’,’e’,’s’ 키등이 처음에 눌리지 않도록 하였다. Dot\_diameter 는 지름으로 실제 원을 그릴 때는 반지름 값을 넣어 그려야 하기 때문에 나중에 2로 나누어 나눈다.

물이 떨어지는 것을 구현하는데 핵심 역할을 하는 update() 함수이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Update 함수는 일정 시간에 이 함수가 1번 실행되어 계속해서 screen의 그림이 바뀔 수 있도록 해준다. 처음에 water\_flag가 1인지 아닌지 판단한다. 만약 1이라면 물이 떨어질 수 있는 모든 준비가 되었다는 의미이다. 즉 ‘I’ 키 를 통해 input.txt 파일을 불러왔고 또 ‘d’ 키를 통해 screen에 dot 과 line을 그렸다는 것이다. 그 다음은 water\_y2 즉 물이 떨어지는 끝 점의 y 좌표가 ofGetHeight()-40 을 넘었는지 넘지 않았는지 판단한다. 만약 넘었다면 더 이상 물줄기를 그릴 필요가 없기 때문에 water\_flag를 0으로 하여 더이상 update 함수에서 물줄기를 계산하지 않게 해준다. 다음은 물이 떨어지는 첫 시작점인지 아닌지 판단한다. 이 판단은 water\_x1과 water\_y1이 0으로 초기화 되어 있는지 아닌지 확인할 수 있다. 만약 둘 다 초기화 값인 0 이라면 물이 점에서 처음 떨어져야 한다. 즉 수직으로 물이 떨어져야 하기 때문에 vertical\_flag를 1로 해주고 선분의 양 끝 점의 x,y 좌표를 water\_dot[check]의 x, y좌표로 설정해준다. 이후 for문을 통해 모든 Line들을 탐색하며 물이 어느 line을 타고 흘러야 하는지 찾는다. 우선 water\_y1의 값 즉 물줄기의 첫 시작 점이 water\_line[i]의 시작점의 y좌표, 끝점의 y좌표 보다 크다는 의미는 line이 screen 상에서 더 위에 있다는 것으로 이 line들은 물이 흐를 수가 없다. 그래서 바로 다음 Line으로 탐색하도록 continue를 써주었다. 만약 line이 더 물줄기의 시작 y좌표보다 아래 있다면 다음은 기울기를 통해 실제 흘러야 하는 물줄기를 찾는다. 이때 line의 직선의 방정식을 세워 이를 water\_y2에서 뺀다. (y=(slope)(x-x1)+y1) 즉 (water,x2, water\_y2) 가 line의 직선의 방정식의 해와 거의 유사한지 판단하는 것으로 이게 3보다 작다면 그 line에서 물줄기가 흐르도록 설정하였다. 하지만 실제 line은 직선이 아니라 선분이다. 그래서 line의 양 끝 y좌표 사이에 water\_y2가 있는지 없는지를 통해 line이 선분임을 표현하였다. 만약 이 조건문이 없다면 line 선분 밖의 직선의 방정식, 즉 line이 그려지지 않는 곳에서도 물이 흐를 것이다. 이때 물줄기가 흘러야 하기 때문에 물이 더이상 수직으로 떨어지면 안되어 vertical\_flag를 0으로 설정하였다. 만약 모든 line에서 (water,x2, water\_y2)가 line의 선분 위에 있지 않다면 물이 수직을 떨어져야 하므로 vertical\_flag를 1로 설정하였다.

실험자는 물줄기를 그릴 때 물방울로 표현하지 않고 선분으로 물줄기를 표현하였다. 즉 선분은 시작 좌표와 끝 좌표가 있어야 한다. 때문에 물줄기의 기울기가 변할 때마다 시작 좌표를 바꿔주어야 한다. 이 문제는 start\_flag를 두어 해결하였다. vertical\_flag 과 start\_flag 두개가 같은지 판단하여 만약 다르다면 그 지점에서 물줄기의 기울기가 바뀌어 시작 좌표 역시 바뀌어야 한다. 이때 시작 좌표는 기존의 끝 좌표로 설정하면 된다. 이후 다시 start\_flag를 vertical\_flag로 설정하여 다음 Update() 함수 호출 시에는 물줄기의 시작점이 이미 바뀐 직선의 시작점임을 알린다. (가장 처음 start\_flag를 초기화 할 때는 물이 수직으로 떨어질 vertical\_flag와 같은 1으로 초기화해준다.)

그리고 vertical\_flag가 1이라면 물이 수직으로 떨어져야 하기 때문에 water\_y2를 3만큼 늘려준다. 여기서 3으로 설정한 이유는 적당한 속도로 물이 떨어지기를 원해서이다. (1로 설정 시 너무 느리게 떨어졌다.) 만약 vertical\_flag가 0 이라면 물줄기가 line을 타고 흘러야 한다. 이때 해당 line의 slope를 판단하여 slope 가 0보다 크다면 water\_x2를 오른쪽으로 3만큼 이동시킨다. (우리가 아는 직교좌표와 y좌표가 반대로 되어 있어 기울기가 0보다 크다는 의미는 왼쪽 위에서 오른쪽 아래로 물이 흘러야 한다.) slope가 0보다 작거나 같으면 water\_x2를 왼쪽으로 3만큼 이동시킨다. 그리고 거기에 따른 water\_y2는 위에서 언급했던 line의 선분(직선)의 방정식에 water\_x2를 대입하여 나온 결과값을 water\_y2에 넣는다. 이미 물줄기가 line 위의 흘러 물줄기의 끝 점이 선분위의 점이기 때문에 직선의 방정식으로 계산하여도 충분하다.

이 같은 과정이 update()함수를 통해 계속해서 반복된다면 우리가 원하는 line을 타고 흐르는 물줄기가 생길 것이다.

지금까지 update함수를 통해 표현한 물줄기 알고리즘을 실제로 screen 상에 그리도록 해주는 draw() 함수이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

우선 brown색으로 screen의 위 아래에 ofDrawRectangle를 이용하여 직사각형을 그려 border를 표현한다. 그리고 draw\_flag가 1인지 아닌지를 판단한다. 1이라면 load를 통해 input.txt 파일을 읽고 ‘d’키를 눌렀다는 의미이므로 dot,line을 그려준다. 둘 다 반복문을 통해 num\_of\_dot, num\_of\_line과 배열의 index가 같아지기 전까지 ofDrawCircle, ofDrawLine을 통해 해당 index에 맞는 배열구조체를 가져와서 도형을 그린다. (자세한 코드 설명은 주석에 있다.) 이때 dot은 check의 위치에서는 물줄기가 떨어져야 되므로 빨간색으로 표현해야 되기 때문에 ofSetColor(255,0,0)으로 설정한다.

그리고 water\_line이 1이라는 의미는 ‘s’키를 눌러 이제 물이 떨어지기 시작했다는 것으로 ofSetLineWidth(5)로 하여 line과 두께가 같은 물줄기를 ofsetColor(0,0,255) 파란색으로 ofDrawLine을 통해 그린다. 이때 시작점은 (water\_x1,water\_y1), 끝 점은 (water\_x2,water\_y2) 이다. ofSetBackgroundAuto가 false이기 때문에 계속해서 겹쳐지도록 그려지고 물줄기 line의 시작점이 바뀌어도 이전의 물줄기 line은 변하지 않는다.

해당 키를 눌렀을 때마다 초기화 및 동적배열 해제, flag들을 재설정해주는 keyPressed(int key) 이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

우선 ‘q’ 키를 누르면 draw\_flag를 0으로 재설정해주고 동적 배열을 할당 받은 water\_line, water\_dot의 메모리를 delete[]를 통해 해제한다. 그리고 \_Exit(0) 으로 프로그램을 종료시킨다.

만약 ‘d’ 키를 누르면 그 전에 ‘L’키가 눌렀는지 판단한다. 이는 load\_flag를 통해 알 수 있다. 만약 눌러지지 않았다면 아무 일도 일어나지 않는다. 아니라면 draw\_flag를 1로 설정하고 다시 ‘l’키를 통해 재 load할 수 있도록 load\_flag를 0으로 바꾼다. 다음 물이 떨어지는 물줄기의 시작 점, 끝 점을 의미하는 water\_x1, water\_x2, water\_y1, water\_y2 좌표를 0으로 모두 초기화 시켜주어 이제 dot, line이 그려졌고 물줄기는 시작하지 않았음을 나타낸다.

‘s’ 키를 누르면 이제 물줄기가 흘러야 한다. 그 이전에 draw\_flag가 1인지 아닌지 판단한다. 만약 1이 아니라면 아직 screen에 dot, line이 그려지지 않아 물줄기가 흘러서는 안된다. 아니라면 start\_flag를 1로 설정하여 vertical\_flag와 같게 초기화 해준다. Draw\_flag는 다시 0으로 설정하는 것이 중요한데 이를 통해 물줄기가 흐르는 동안에는 방향키를 통해 dot을 움직이지 못하도록 해준다. 그리고 ‘e’ 키를 눌러 물줄기를 멈추게 할 수 있는데 물줄기를 멈출 때 ofSetBackgroundAuto를 true로 하여 물줄기를 멈춘다. 그래서 다시 ‘s’ 키를 누를 경우에 대비하여 ofSetBackgroundAuto를 false로 다시 재설정하는 것을 추가했다.

‘e’ 키를 누르면 물이 흐르는 것을 멈추어야 하는데 ‘e’ 키가 작동하는 전제조건이 draw\_flag가 1 이거나 water\_flag가 1 일 때만 가능하다. 만약 1이라면 water\_flag를 0으로 설정해 update() 함수에서 물줄기 계산 알고리즘을 더이상 못 돌아가게 하고 방향키를 통해 다시 물줄기를 흐르게 하는 dot을 찾을 수 있게 해준다. 물줄기의 시작 좌표, 끝 좌표를 다시 초기화 하고 ofSetBackgroundAuto(true)를 통해 그려졌던 물줄기를 다시 screen에서 다시 삭제한다. 이때 draw\_flag 역시 1로 변경되었기 때문에 기존에 그려져 있던 line, dot 들은 draw() 를 통해 다시 그려지게 된다.

여기서는 input.txt 파일을 가져오기 및 방향키를 조절해 물이 떨어질 dot을 찾는 keyReleased() 이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

만약 ‘l’키를 눌렀다면 input.txt 파일을 불러오도록 ofFileDialogResult로 불러온다. 그리고 만약 파일 불러오는 것을 성공했다면 load\_flag를 1로 하여 ‘l’을 이외의 키들도 누를 수 있게 해주고 여기서는 아직 물줄기가 떨어지면 안되기 때문에 water\_flag는 0으로 초기화 해준다. 또한 물줄기가떨어질 dot의 index를 의미하는 check 역시 0으로 초기화 해주고 ‘l’ 을 누를 때마다 screen을 모두 흰 바탕으로 초기화 해주었다.

그리고 dot중에서 물줄기가 떨어질 dot을 찾아야 되는데 우선 water\_flag가 0일때와 num\_of\_dot 즉 dot의 개수가 0이 아닐 때 만 방향키로 움직이게 하였다. 오른쪽 방향키를 눌렀다면 check를 1만큼 늘려주고 만약 check가 num\_of\_dot과 같다면 해당 dot은 가장 왼쪽 dot과 같기 때문에 check를 0으로 설정한다.

왼쪽 방향키를 눌렀다면 check를 1빼주고 check가 -1이 되었을 때는 해당 dot이 가장 오른쪽의 dot과 같기 때문에 check를 num\_of\_dot-1 로 설정해준다.

Input.txt를 불러와 해당 정보들을 num\_of\_line, num\_of\_dot, water\_dot, water\_line 구조체 배열에 저장하는 processOpenFileSelection() 이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

파일의 존재유무를 check하고 해당 존재유무에 따라 cout<<endl을 통해 정보들을 출력해 준다. Input\_type을 통해 해당 줄이 line에 대한 것인지 dot에 대한 것인지 판단한다. Input\_type이 0이라면 line에 관련한 정보, 1이라면 dot에 관련한 정보이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

For문을 통해 더이상 정보가 없을 때까지 계속해서 정보들을 저장한다.

만약 word.size가 1이라는 것은 dot 이나 line의 개수를 의미한다. 그래서 여기서는 Input\_type이 0이라면 num\_of\_line에 line 개수를 받아들이고 water\_line 구조체 배열을 new를 통해 num\_of\_line만큼 동적 메모리를 할당 받는다.

Input\_type이 0이라면 마찬가지로 num\_of\_dot에 dot 개수를 받아들이고 water\_dot 구조체 배열에 new를 통해 num\_of\_dot 만큼 동적 메모리를 할당 받는다.

만약 word.size가 1보다 크다면 좌표를 받아야 한다. 그리고 i는 index를 의미하기 때문에 input\_type이 0이라면 num\_of\_line의 각 양 끝점의 x,y 좌표를 받아 water\_line[i] 구조체 배열에 저장한다. 그리고 추가적으로 slope을 double(y2-y1)/double(x2-x1)으로 기울기를 구하여 water\_line[i].slope 에 저장한다. Index인 i를 1씩 증가시켜 계속해서 이 과정을 시행한다. 그래서 i+1이 num\_of\_line과 같아진다면 input\_type을 1로 설정하고 i 역시 0으로 재설정해준다.

만약 inputy\_type이 1이라면 dot의 x,y 좌표를 받아 water\_dot[i]에 넣어주고 i값을 1씩 증가시켜 마지막 for문까지 돌면 모든 정보가 다 받아들여 진다.

1-2.

물이 떨어지는 알고리즘을 표현한 update 함수의 flow 차트 이다.

Update 함수 시작

Water\_flag 가 1인지

아닌지 판단

Update 함수 종료

false

true

물줄기의 끝 y좌표(water\_y2)가 ofGetHeight-40보다 작은지 판단

Water\_flag 0으로 설정

false

true

Water\_x1==0&& water\_y1==0(초기화 상태라면)

Vertical flag=1

Water\_x1=water\_dot[check].X

Water\_y1=water\_dot[check].Y

Water\_x2=water\_dot[check].X

Water\_y2=water\_dot[check].Y

true

false

For문을 num\_of\_line만큼 돈다

Line의 양 끝의 y좌표가 물줄기의 water\_y1보다 작은지 판단

true

false

Line 직선 방정식에 water\_x2를 넣은 값과 water\_y2의 절대값의 차가 3보다 작으면

false

true

Water\_y2가 line의 양 끝 y 좌표 사이에 있다면

false

Vertical flag=1

For문이 끝났으면

true

Vertical flag=0

Vertical flag와 start flag가 다르면(물줄기 시작점을 바꿈)

(시작점을 이전의 끝 점으로 재설정)

Water\_x1=water\_x2

Water\_y1=water\_y2

Start flag=vertical flag

true

Vertical flag가 1이라면(물줄기 수직으로 떨어져야함)

Water\_y2+=3

물줄기 y2좌표에 가중치를 더해줌

true

false

Water\_line[i]의 기울기가 0보다 크다면

false

true

Water\_x2-=3

물줄기 x2좌표에 가중치를 빼줌

Water\_x2+=3

물줄기 x2좌표에 가중치를 더해줌

Water\_y2는 해당 line의 직선의 방정식에 water\_x2값을 넣은 값을 설정

Update 함수 종료

도형을 그리는 Draw 함수의 flow 차트이다.

Draw 함수 시작

water\_flag가 1인지 확인(떨어지는 물 그림)

false

Draw\_flag가 1인지 확인(그림 가능)

true

두께를 5로 설정하고 색깔을 blue로 ofDrawLine을 이용하여 물떨어지는 선분 그림

true

Index i 가 num\_of\_dot이랑 다를때까지만 반복문을 돈다

false

그렇지 않으면 점을 검은색으로 표시

i==check 랑 같다면

false

true

OfDrawCircle로 원을 그려주고 i 를 1 증가시킴

점을 빨간색으로 표시하게 한다.

i 를 0으로 다시 초기화

Index i 가 num\_of\_line이랑 다를때까지만 반복문을 돈다

Draw 함수 종료

색깔을 brown 으로 하여

Ofdrawline으로 선분그리고 i를 1 증가시킴

i 를 0으로 다시 초기화

Input 파일을 입력받는 processOpenFileSelection 함수 flow 차트 이다.

Input type이 0인가?

ProcessOpenFileSelection 함수 시작

true

파일이 존재하는가 확인

false

Index i+1이 Num\_of\_line인가?

true

true

false

false

Water\_line 구조체 배열에 한줄 씩 x1,y1,x2,y2 좌표와 이로 계산한 기울기를 받아 저장함

Input\_type을 0으로 두어 line, dot 정보 구분

Index i 초기화

For문으로 input 파일 끝까 line마다 받아옴

Input\_type을 1로 바꿔 다음 부터는 dot에 관련한 정보를 받아오게 함 index i초기화

Line이 1글자인가?

Water\_dot 구조체 배열에 한줄 씩 x1,y1 좌표를 받아 저장함

true

Input type이 0인가?

Num\_of\_dot에 그 정보 저장하고

그 크기만큼 Water\_dot 동적할당배열 만듬

false

true

Num\_of\_line에 그 정보 저장하고

그 크기만큼 Water\_line 동적할당배열 만듬

함수 종료

방향키를 눌렀을 때 물이 떨어질 지점을 결정하는 코드의 flow 차트이다.

Water\_flag가 0이고 num\_of\_dot이 0이 아닐때

false

true

왼쪽 키를 눌렀다면

오른쪽 키를 눌렀다면

false

false

true

true

Check를 1 빼주어 왼쪽 점으로 이동한다.

Check를 1 더해주어 오른쪽 점으로 이동한다.

만약 check가 num\_of\_dot과 같다면

만약 check가 -1과 같다면

false

true

true

Check를 num\_of\_dot-1 로 재설정(가장 오른쪽 점 의미)

Check를 다시 0으로 초기화 시킴

함수 종료