#include<GL\glut.h>

#include<time.h>

#include<vector>

#include<iostream>

class Circle

{

public:

float x, y, r, alpha;

Circle(float \_x = 0.0, float \_y = 0.0, float \_r = 10.0) :x(\_x), y(\_y), r(\_r), alpha(1.0) {};

void Draw();

};

using namespace std;

void Circle::Draw()

{

glBegin(GL\_LINE\_LOOP);

for (float i = 0.0; i < 2 \* 3.14; i += 3.14 / 18)

{

glVertex2f(this->x + this->r\*sin(i), this->y + this->r\*cos(i));

}

glEnd();

}

vector <Circle> circ;

float WinWid = 400.0, WinHei = 400.0;

void Draw()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

vector<Circle>::iterator i = circ.begin();

while (i != circ.end())

{

i->Draw();

if (i->alpha <= 0.1)

i = circ.erase(i);

else

++i;

}

glutSwapBuffers(); // for GLUT\_DOUBLE

}

void Initialize()

{

glClearColor(0.0, 0.0, 0.0, 1.0);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(-WinWid / 2, WinWid / 2, -WinHei / 2, WinHei / 2, -200.0, 200);

glMatrixMode(GL\_MODELVIEW);

}

void Timer(int value)

{

for (vector<Circle>::iterator i = circ.begin(); i != circ.end(); i++)

{

i->r++;

i->alpha /= 1.05;

}

glutPostRedisplay();

glutTimerFunc(50, Timer, rand() % 2);

}

void Timer2(int)

{

Circle c(rand() % int(WinWid) - WinWid / 2, rand() % int(WinHei) - WinHei / 2, rand() % 20 + 2);

circ.push\_back(c);

glutPostRedisplay();

glutTimerFunc(1000, Timer2, 0);

}

int main(int argc, char\*\* argv)

{

srand(time(0));

for (int i = 2; i < 2 + rand() % 5; i++)

{

Circle c(rand() % int(WinWid) - WinWid / 2, rand() % int(WinHei) - WinHei / 2, rand() % 20+2);

circ.push\_back(c);

}

//Initialization

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);

glutInitWindowSize(WinWid, WinHei);

glutInitWindowPosition(100, 200);

glutCreateWindow("Fuck me in my ass");

//Registration

glutDisplayFunc(Draw);//Drawing

glutTimerFunc(500, Timer, 0); // animation func registration

glutTimerFunc(100, Timer2, 0);

Initialize();

glutMainLoop();

return 0;

}