Лабораторная работа № 6

**Приемы синтеза динамических образов на плоскости**

**1 Цель работы**

Практическое освоение типовых приемов синтеза динамических графических образов на примере создания программ управления движением набора простых графических объектов.

**2 Зрительное восприятие быстрой смены образов**

Синтез динамических эффектов на экране монитора проводится чередованием набора образов, в той или иной степени моделирующим моменты какого-либо реального физического процесса. При этом для достижения приемлемых результатов следует учитывать психофизические особенности восприятия глазом быстрой смены образов, а также принципы и особенности формирования образов на экране монитора.

Приводимые ниже две программы создания эффектов мерцания позволяют понять некоторые из этих принципов.

***Uses*** *Сrt, Graph;*

***Var*** *Gx,Gy, Gd,Gm, x,y, x1,y1 ,i :integer;*

***begin***

*Gd:=detect; InitGraph(Gd,Gm,'c:\tp7\bgi');*

*Gx:=GetMaxX; Gy:=GetMaxY; randomize;*

*{------------------------------------------------------------------------------------*

*"Небо в алмазах" -*

*генератор точек с гашением в их окрестности набора точек*

*--------------------------------------------------------------------------------------}*

***Repeat*** *i:=random(7)+9; x:=random(Gx); y:=random(Gy);*

*Putpixel(x,y,i);*

*for i:=0 to random(900) do*

*Putpixel(x-15+random(31),y-15+random(31),0)*

***Until*** *KeyPressed;*

*ReadKey; Cleardevice;*

*{---------------------------------------------------------------------------------------*

*"Мерцание" -*

*последовательно высвечиваются и гасятся две случайные точки, из-за инерционности зрения возникает эффект мерцающего набора точек*

*-------------------------------------------------------------------------------------------------------}*

***Repeat*** *i:=random(7)+9; x1:=x; x:=random(Gx);*

*y1:=y; y:=random(Gy);*

*Putpixel(x,y,i);*

*delay(100); Putpixel(x1,y1,0)*

***Until*** *KeyPressed;*

*CloseGraph*

***end****.*

**3 Практическое задание**

3.1 Модифицировать алгоритм "Небо в алмазах"

3.1.1 Выводить точки в определенную область экрана, например

3.1.2 Выводить точки с сортировкой по цвету в зависимости от ее положения на экране

яркость

3.2 Модифицировать алгоритм "Мерцание"

3.2.1 Увеличить "заполненность" экрана, оперируя одновременно с несколькими точками

3.2.2 Организовать "мерцающую волну" с направлением по вариантам п. 3.1.2

**4 Синтез динамических эффектов с анализом возможности**

**выполнения процедур**

Одним из узловых моментов моделирования большинства динамических процессов является установка набора ограничительных условий на выполнение тех или иных элементов этих процессов. Причем эти условия обычно являются весьма точным отображением реальных физических ограничений. Рассмотрим это на примере моделирования некоторых физических процессов.

В раздел описаний предыдущих программ добавим

***const*** *N=500; { число объектов }*

***var*** *X,Y, { массивы координат объектов }*

*vX,vY { массивы скоростей объектов }*

*P: { цвет объектов } array[1..N] of integer;*

*{--------------------------------------------------------------------------------------*

*"Молекулы газа в поле тяготения" -*

*динамический процесс с анализом его допустимости.*

*---------------------------------------------------------------------------------------}*

***for*** *i:=1* ***to*** *N do* ***begin***

*X[i]:=random(Gx); { начальные координаты "молекул"}*

*Y[i]:=random(Gy) end;*

***REPEAT***

***for*** *i:=1* ***to*** *N* ***do******begin***

*PutPixel(X[i],Y[i],0); { стирание старого изображения }*

*{ Генерируем случайные составляющие скорости "молекул" }*

*dX:=-20+random(41); { -20...20 - равновероятно влево и вправо }*

*dY:=-10+random(41); { -10...30 - с тяготением вниз }*

*{ Если при выбранной скорости "молекула" не выйдет за границу,*

*то проводится смена ее координат и перерисовка }*

***if****( X[i]+dX> 0) and ( X[i]+dX< Gx)* ***then*** *inc(X[i],dX);*

***if****( Y[i]+dY> 0) and( Y[i]+dY< Gy)* ***then*** *inc(Y[i],dY);*

*PutPixel(X[i],Y[i], 14)*

***end***

***until*** *KeyPressed;*

*ReadKey; ClearDevice;*

**5 Практическое задание по модификации алгоритма "Молекулы газа в поле тяготения"**

5.1 Точки ярких цветов направить яркие

к одной из границ экрана, прочие -

к противоположной границе. темные

5.2 Смоделировать поведение

молекул газа в цилиндре под

движущимся поршнем.

5.3 Смоделировать процесс

испарения смеси трех жидкостей

с разной температурой кипения.

**6 Анализ взаимодействия и свойств объектов**

Следующие фрагменты программы иллюстрируют принцип выполнения графических процедур с анализом области, где моделируется динамический процесс.

*{--------------------------------------------------------------------------------------*

*"Снег" - перемещение малых объектов с анализом "пути следования"*

*---------------------------------------------------------------------------------------}*

***for*** *i:=1* ***to*** *N* ***do******b***

***egin***

*X[i]:=random(Gx); { начальное положение "снежинок"}*

*Y[i]:=random(Gy); PutPixel(X[i],Y[i],15) end;*

*Rectangle(100,Gy-100, Gx-100,Gy); { преграда }*

***REPEAT***

***for*** *i:=1* ***to*** *N* ***do******begin***

***if****(Y[i]<Gy) and( GetPixel(X[i],Y[i]+1)=0)*

*{ если нет преграды по ходу движения }* ***then***

***begin***

*PutPixel(X[i],Y[i],0); inc(Y[i]); {смещение "снежинок" вниз}*

*PutPixel(X[i],Y[i],15)* ***end***

***else*** *{ при достижении преграды генерируются }*

***begin*** *{ новые координаты "снежинки" }*

*Y[i]:=random(Gy div 10);*

*X[i]:=random(Gx); PutPixel(X[i],Y[i],15) end*

***end***

***until*** *KeyPressed;*

*ReadKey; ClearDevice;*

*{----------------------------------------------------------------------------------------*

*"Взрыв" -*

*перемещение малых объектов с остановкой при достижении границы*

*-----------------------------------------------------------------------------------------}*

***for*** *i:=1* ***to*** *N* ***do******begin***

*X[i]:= 300 +i mod 25; { начальные координаты осколков }*

*Y[i]:= 200 +i div 25;*

*PutPixel(X[i],Y[i],15);*

*vX[i]:= -10+random(21); { составляющие скоростей осколков }*

*vY[i]:= -10+random(21)*

***end****; ReadKey;*

*{ звук взрыва }* ***for*** *i:=1* ***to*** *300* ***do******begin***

*sound(random(i)); delay(1); nosound*

***end****;*

***REPEAT*** *{ процесс разлета осколков }*

***for*** *i:=1* ***to*** *N* ***do******begin***

***if****( X[i]+vX[i]> 0) and( X[i]+vX[i]< Gx)and*

*( Y[i]+vY[i]> 0) and( Y[i]+vY[i]< Gy)*

***then******begin*** *{ если осколок еще не долетел до края }*

*PutPixel(X[i],Y[i], 0); inc(X[i],vX[i]);*

*inc(Y[i],vY[i]);*

*PutPixel(X[i],Y[i],12)*

***end*** *{ перерисовка в новом положении }*

***else begin*** *{ пиксель долетел до края - обнуляем его скорость*

*и генерируем звук при "ударе" о край }*

*vX[i]:=0; vY[i]:=0;*

*sound(200); delay(2); nosound*

***end***

***end***

***until*** *KeyPressed*

***END****.*

**7 Практическое задание**

7.1 Модифицировать алгоритм "Снег"

7.1.1 Смоделировать снег с ветром переменного направления.

7.1.2 Смоделировать образование "снежных хлопьев" при достижении движущимися пикселями нижней части экрана.

7.1.3 Смоделировать град, соскальзывающий на наклонных преградах.

7.2 Модифицировать алгоритм "Взрыв"

7.2.1 Смоделировать взрыв объекта со снижением скорости осколков в процессе их разлета.

7.2.2 Смоделировать ракету, взрывающуюся при столкновении с препятствием.

7.2.3 Смоделировать две бомбы: осколки от взрыва первой подрывают вторую.

7.2.4 Смоделировать разлет осколков с учетом действия сопротивления воздуха и силы тяжести.

7.2.5 Смоделировать фейерверк (или салют) с ограничением "времени жизни" светящихся осколков.