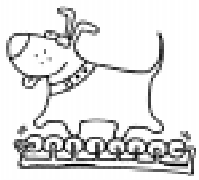
Паньгина Нина Николаевна

ЗАНЯТИЕ 5. МОДЕЛИРОВАНИЕ В VISUAL BASIC

Одной из наиболее интересных тем в курсе VB является тема «Моделирование». Компьютерные модели дают широкие возможности представить изучаемые материалы (или процессы) наглядно с различных сторон, а также способствуют углубленному их пониманию. Но самым замечательным является то, что, кроме создания простых моделей, адекватных некоторым сложным процессам, можно еще и управлять моделируемыми процессами, изменяя соответствующие параметры модели.

УНИВЕРСАЛЬНАЯ МОДЕЛЬ ДВИЖЕНИЯ



Рассмотрим моделирование некоторых видов движения, которые хорошо знакомы из уроков физики или из повседневности.

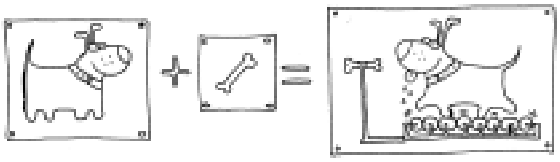
Модели движения относятся к динамическим моделям, то есть моделям, которые учитывают зависимость состояния модели от времени. Состояние модели определяется точкой в координатном пространстве. Точки, соответствующие различным временным состояниям модели, составляют некоторую траекторию.

Чтобы построить динамическую модель, нужно задать систему координат и описать в ней траекторию движения тела.

Модели движения согласно [1] подразделяются на аналитические, дифференциальные и разностные (различие представлений демонстрируется в таблице 1 для прямолинейного движения).

На уроках физики в средней школе даются аналитические модели движения (дифференциальные модели движения – это прерогатива высшей школы), а вот для компьютерной реализации наиболее удобными являются разностные модели, то есть рассматривается положение точки через некоторые заданные промежутки времени, и непрерывное движение заменяется последовательностью отдельных прямолинейных шагов.

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ УНИВЕРСАЛЬНОЙ МОДЕЛИ ДВИЖЕНИЯ



Основная идея построения универсальной модели движения проста и основывается на известном соотношении, свя-

ТИПЫ ДВИЖЕНИЯ	ВИДЫ МОДЕЛЕЙ		
	Аналитическая	Дифференциальная	Разностная
Равномерное	$X = X_0 + Vt$	$\frac{dX}{dt} = V$	$\begin{cases} X(t+dt) = X(t) + Vdt \\ X(0) = X_0 \\ t = 0,\ dt,\ 2dt,\ \ldots \end{cases}$
Равноускоренное	$X = X_0 + Vt + at^2/2$	$\frac{dX}{dt} = V,$ $\frac{dV}{dt} = a$	$\begin{cases} X(t+dt) = X(t) + V(t)dt \\ V(t+dt) = V(t) + adt \\ X(0) = X_0 \\ V(0) = V_0 \\ t = 0,\ dt,\ 2dt,\ \ldots \end{cases}$

Таблица 1

зывающем вектор перемещения **ΔS** материальной точки (рисунок 1) с вектором ее средней скорости $\mathbf{V}_{ср}$.

$$\mathbf{\Delta S} = \mathbf{V}_{ср} \cdot \Delta t; \qquad (1)$$

где $\mathbf{V}_{ср} = (\mathbf{V}_0 + \mathbf{V}_1)/2$;

Δt – промежуток времени между двумя последовательными рассматриваемыми состояниями материальной точки, в которых известны векторы скорости $\mathbf{V}_0(t)$ и $\mathbf{V}_1(t + \Delta t)$.

Основная формула (1) получается из известных формул равноускоренного движения (полагается, что **a** – вектор ускорения изменяется незначительно за время Δt):

$$\mathbf{V}_1 = \mathbf{V}_0 + \mathbf{a} \cdot \Delta t; \qquad (2)$$

$$\mathbf{\Delta S} = \mathbf{V}_0 \cdot \Delta t + \mathbf{a} \cdot \Delta t^2/2.$$

Ниже представлена детальная разработка Windows-приложения в среде Visual Basic для моделирования и демонстрации различных видов движения: и таких простых, как равномерное прямолинейное, и более сложных, как движение тела, брошенного под углом к горизонту, и таких очень сложных, как движение бумажного самолетика или движение спутника Земли. «Универсальность» модели состоит в том, что будет использоваться одна и та же формула (1) для различных видов движения. Усложнение модели приводит лишь к учету дополнительных факторов в определении вектора скорости в формуле (2).

Общий алгоритм для моделирования отдельного вида движения можно записать в следующем виде:

Начало
Ввод начальных данных (для момента времени $t = t_0$);
Повторять
Вычислить новые значения скорости и координат в момент времени $t + \Delta t$;
Нарисовать отрезок траектории;
Переприсвоить значения скорости и координат для нового момента времени t;
Конец повтора;
Конец.

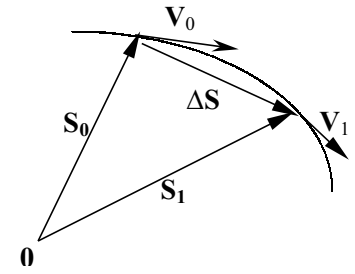


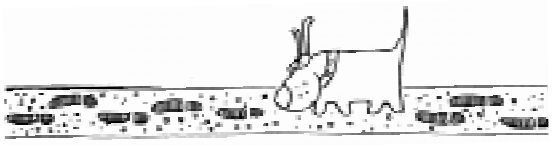
Рисунок 1

Рисунок 2

1.1. РАВНОМЕРНОЕ ДВИЖЕНИЕ

Равномерное движение характеризуется тем, что в формуле (2) вектор ускорения – нулевой и потому отсутствует соответствующее слагаемое.

Модель 1.
Равномерное прямолинейное движение в одном измерении.



Рассмотрим движение в плоскости вдоль оси абсцисс (рисунок 2).

Скорость постоянна:

$\mathbf{V}_1 = \mathbf{V}_0$;

$\mathbf{S}_1 = \mathbf{S}_0 + \mathbf{\Delta S}$.

Следовательно,

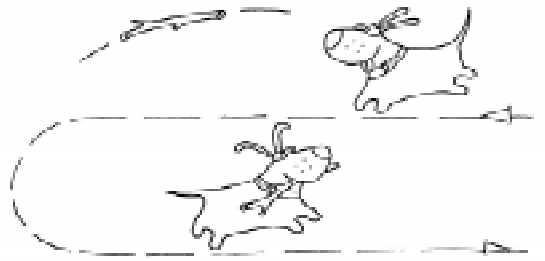
$\mathbf{S}_1 = \mathbf{S}_0 + (\mathbf{V}_0 + \mathbf{V}_1)/2 \cdot \Delta t$, или

$X_1 = X_0 + V_0 \cdot \Delta t$,

то есть траектория в данном случае – это прямая линия.

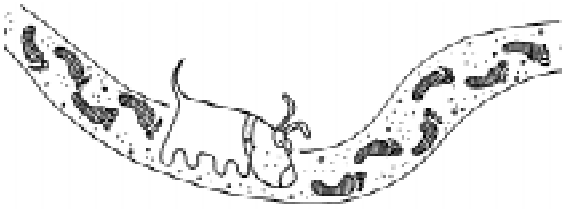
Модель 2.
Равномерное прямолинейное движение с отражением.

Таким же образом можно промоделировать движение шара по прямой до стенки и обратно. В алгоритм вводится про-



верка достижения шаром стенки и замена вектора скорости на противоположный по направлению (абсолютно упругий удар):
Если (стенка достигнута) **то** $\mathbf{V}_0 = -\mathbf{V}_0$;

Модель 3.
Равномерное прямолинейное движение в двух измерениях.



Движение в плоскости рассматривается как составное движение на прямой соответственно по оси X (абсцисс) и оси Y (ординат).

Скорость постоянна

$\mathbf{V}_1 = \mathbf{V}_0$,

и направление вектора скорости составляет угол α с осью абсцисс. Вектор скорости раскладывается на две составляющие $\mathbf{V}_{0x}$ и $\mathbf{V}_{0y}$ или в координатах:

$V_{0x} = V_0 \cdot \cos(\alpha)$;
$V_{0y} = V_0 \cdot \sin(\alpha)$;

Следовательно, по формуле (1),

$\mathbf{S}_1 = \mathbf{S}_0 + \mathbf{V}_{cp} \cdot \Delta t$, или в координатах

$X_1 = X_0 + V_{0x} \cdot \Delta t$;
$Y_1 = Y_0 + V_{0y} \cdot \Delta t$;

то есть траектория в данном случае – это прямая линия, направленная под углом α к оси абсцисс.

Модель 4.
Движение бильярдного шара (для самостоятельного выполнения).

1. Сформулируйте задачу о движении бильярдного шара по столу с отражением от стенок, приведите соответствующие формулы.

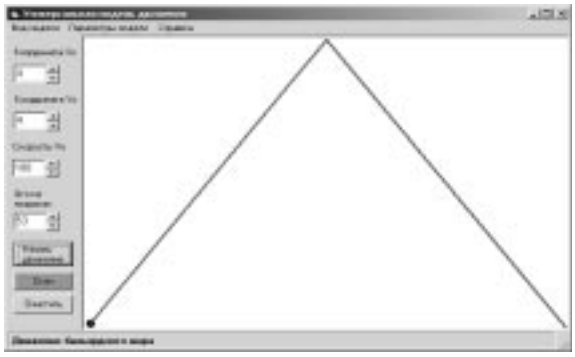
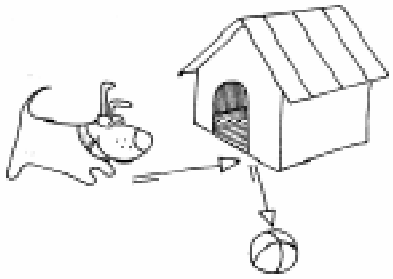
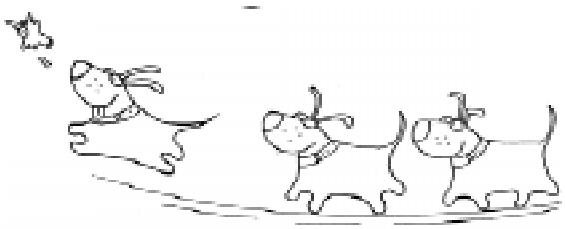


Рисунок 3

2. В готовом нижеприведенном проекте поработайте с моделью: попадите шаром из заданного положения в лузу (или другой шар) с однократным отражением от борта (рисунок 3).



1.2. РАВНОУСКОРЕННОЕ ДВИЖЕНИЕ



Модель 5.
Равноускоренное движение тел в поле силы тяжести.

Какова же будет траектория движения тела, брошенного под углом к горизонту, при отсутствии сопротивления воздуха, но с учетом силы тяжести? В формуле (2) вектор ускорения **a** есть вектор ускорения свободного падения **g**, то есть получаем векторную формулу

$\mathbf{V}_1 = \mathbf{V}_0 + \mathbf{g} \cdot \Delta t$.

При таком движении (рисунок 4) составляющая вектора скорости $\mathbf{V}_x$ по оси

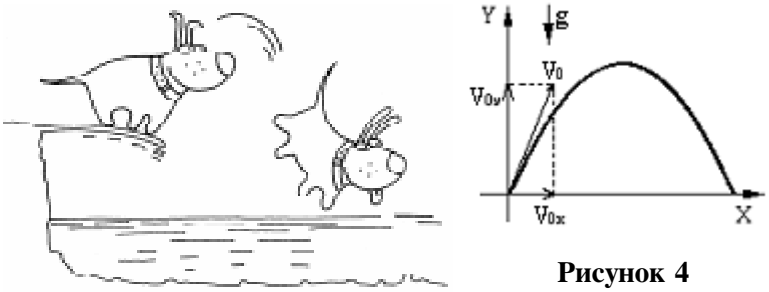


Рисунок 4

X не меняется, а составляющая вектора скорости $\mathbf{V}_y$ по оси *Y* будет уменьшаться, так как вектор ускорения направлен противоположно выбранному положительному направлению.

$V_{1x} = V_{0x}$;

$V_{1y} = V_{0y} - g \cdot \Delta t$.

Из формулы (1) следует, что точка траектории во время $t + \Delta t$ имеет координаты

$X_1 = X_0 + (V_{0x} + V_{1x})/2 \cdot \Delta t$,

$Y_1 = Y_0 + (V_{0y} + V_{1y})/2 \cdot \Delta t$.

Очевидно, что приведенные выше формулы определения координат справедливы и для ранее рассмотренных моделей, и, в силу их общности, далее не приводятся.

По такому незначительно видоизмененному алгоритму можно составить не просто программу, моделирующую движение тела, брошенного под углом к горизонту, а программу, решающую реальные физические задачи, такие как:

– вычислить дальность полета,
– вычислить высоту полета,
– определить время подъема тела до самой высокой точки траектории,
– определить общее время полета тела.

Проявив долю фантазии, можно смоделировать, например, полет резинового мяча с упругим отражением от горизонтальной или вертикальной поверхности, а затем приступить к разработке игр на этой основе (попасть в цель из орудия или мячом в баскетбольную корзину).

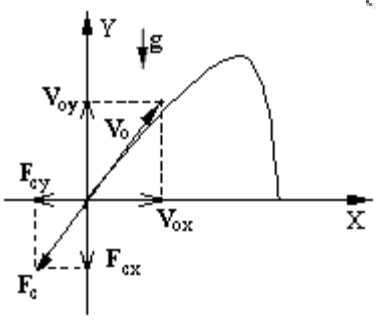


Рисунок 5

Модель 6.
Движение тел в поле силы тяжести при сопротивлении воздуха.

При относительно малых скоростях движения сила сопротивления воздуха пропорциональна скорости движения тела и направлена в противоположную сторону:

$\mathbf{F}_c = - c \cdot \mathbf{V}$,

где *c* – коэффициент, который зависит от формы и размера тела, плотности воздуха (наглядно данный закон проявляется при езде на велосипеде).

Следовательно, ускорение $\mathbf{a}_c$ можно выразить следующим образом (рисунок 5):

$\mathbf{a}_c = -\alpha_c \cdot \mathbf{V}$,

где $\alpha_c = c / m$ (*m* – масса тела).

По формуле (2) векторная форма:

$\mathbf{V}_1 = \mathbf{V}_0 + (\mathbf{g} + \mathbf{a}_c) \cdot \Delta t$.

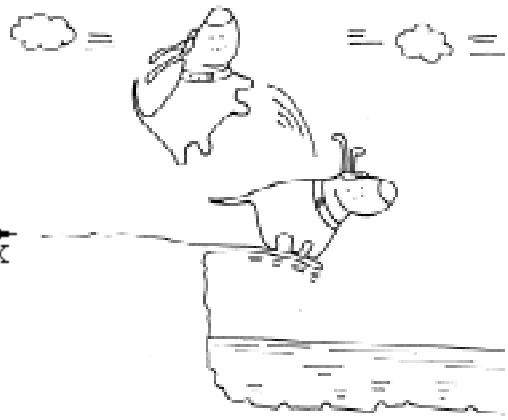
В проекциях:

$V_{1x} = V_{0x} - \alpha_c \cdot V_{0x} \cdot \Delta t$;

$V_{1y} = V_{0y} - g \cdot \Delta t - \alpha_c \cdot V_{0y} \cdot \Delta t$.

Особенность данной траектории состоит в том, что, в отличие от параболы (Модель 5), она представляет собой несимметричную кривую относительно вертикали, проведенной через наивысшую точку полета, то есть нисходящая ветвь круче восходящей.

В спортивном бадминтоне этим пользуются очень широко. Высокую подачу, например, подают настолько высоко, насколько позволяют высота потолка и силы игрока, при этом на задней линии



поля противника волан падает почти вертикально, а по отвесно падающему волану нельзя нанести ответно сильного удара, так как мешает оперение.

Модель 7.
Полет бумажного самолетика.

Эксперименты с самолетиком под разными углами и с разными начальными скоростями показывают, что его траектория гораздо сложнее, чем брошенный камень или волан. Угадать траекторию практически невозможно, однако это вовсе не означает, что поведение кувыркающегося самолетика не поддается анализу [2]. Эту задачу поставил и решил основоположник аэродинамики Н.Е. Жуковский (1847–1921). Жуковский назвал эту задачу «задачей о планере».

На бумажный самолетик действуют три силы (рисунок 6):

$\mathbf{F}_Т$ – сила тяжести,

$\mathbf{F}_c$ – сила сопротивления воздуха,

$\mathbf{F}_П$ – подъемная сила, перпендикулярная вектору скорости $\mathbf{V}_0$.

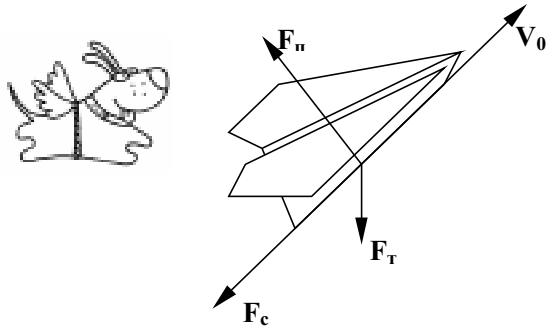


Рисунок 6

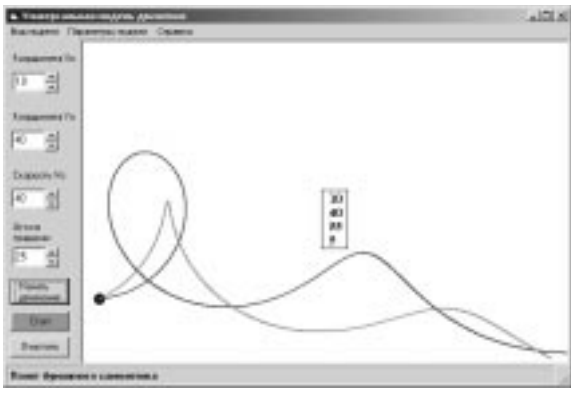


Рисунок 7

По так называемой теореме Жуковского величина подъемной силы пропорциональна квадрату скорости

$F_П = \beta \cdot V^2$.

Рассуждения, аналогичные предыдущему случаю, приводят к следующим формулам:

$V_{1x} = V_{0x} - \alpha_c \cdot V_{0x} \cdot \Delta t - \alpha_p \cdot V \cdot V_{0y} \cdot \Delta t$;

$V_{1y} = V_{0y} - g \cdot \Delta t - \alpha_c \cdot V_{0y} \cdot \Delta t + \alpha_p \cdot V \cdot V_{0x} \cdot \Delta t$.

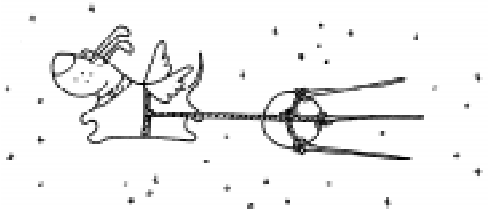
Здесь α_p – коэффициент, учитывающий подъемную силу, а $V = \sqrt{V_{0x}^2 + V_{0y}^2}$.

На рисунке 7 показаны траектории полета бумажного самолетика при различных параметрах.

При моделировании данного вида движения, наряду с компьютерными экспериментами, можно проводить настоящие физические эксперименты, сделав бумажные самолетики. Варьируя такими исходными данными, как начальная скорость и угол наклона, можно добиться различных результатов и получить разнообразные траектории движения.

Интересно заметить, что если при компьютерном моделировании принять коэффициент сопротивления воздуха равным нулю, то получится траектория движения, аналогичная профилю океанской волны. И это не случайно, поскольку сходными являются формулы, описывающие эти движения [3].

Модель 8.
Движение спутника Земли.



Вполне естественным после стольких рассмотренных видов движения представляется желание смоделировать Вселенную или хотя бы Солнечную систему. Можно начать с более простого – смоделировать движение спутника Земли [4].

На спутник действует единственная сила **F** – сила притяжения Земли. Ее ве-

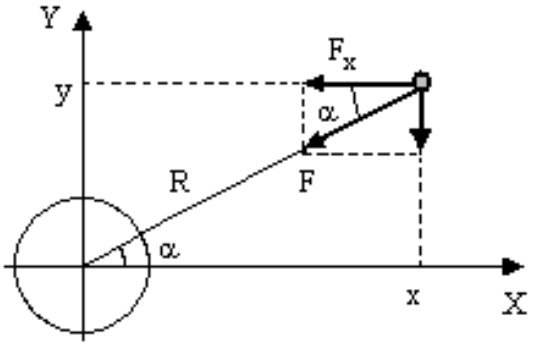


Рисунок 8

личину можно вычислить по формуле закона всемирного тяготения:

$$F = G\frac{M_3 \cdot m_c}{R^2},$$

где G – гравитационная постоянная, она равна $6.67 \cdot 10^{-11}$ м^3/(кг $\cdot$ с^2);
M_3 – масса Земли, равная $5.98 \cdot 10^{24}$ кг;
m_c – масса спутника.

Если известны координаты X, Y спутника, то расстояние от центра Земли до спутника:

$$R = \sqrt{X^2 + Y^2}.$$

Из рисунка 8 видно, что
$\mathbf{F}_x = \mathbf{F}\cos(\alpha)$;
$\mathbf{F}_y = \mathbf{F}\sin(\alpha)$.

Если на спутник действует только притяжение Земли, то второй закон Ньютона можно записать в виде:

$$m_c \cdot a_x = -G\frac{M_3 \cdot m_c}{R^2}\frac{x}{R};$$

$$m_c \cdot a_y = -G\frac{M_3 \cdot m_c}{R^2}\frac{y}{R}.$$

Сократив на m_c и подставив значение ускорения в известные нам формулы (1) и (2), получим формулы для вычисления новых значений составляющих скорости через время Δt:

$$V_{1x} = V_{0x} - G \cdot M_3 \cdot X_0 \cdot \Delta t / R^3;$$
$$V_{1y} = V_{0y} - G \cdot M_3 \cdot Y_0 \cdot \Delta t / R^3.$$

Существуют специальные методы расчета необходимого шага по времени, но программно можно уменьшать шаг в два раза и находить новую траекторию. Если решение для нового Δt неотличимо с заданной точностью от предыдущего, то будем считать, что выбор шага удовлетворителен.

На рисунке 9 приведен результат работы программы. В задаче заданы реальные физические данные для полета космического корабля с космонавтом Юрием Гагариным, который облетел Землю за 108 минут. В задаче получено хорошее приближение – 102 минуты. На рисунке приведены также орбиты спутников при различных начальных скоростях.

Можно самостоятельно изменить и дополнить программу. Первая часть касается эстетического оформления, например, через несколько циклов выводить на экран текущие значения скорости и расстояния до Земли. Можно найти время, за которое спутник совершит полный оборот, да и сам спутник можно рисовать не точкой, а в виде фигурки.

Помимо художественных дополнений, можно изменить и саму модель. Так, если в формулах для V_x и V_y заменить минусы на плюсы, то закон всемирного тяготения превратится в закон всемирного отталкивания. Тем самым модель спутника превратится в модель взаимодействия одноименно заряженных частиц. Сила в законе тяготения обратно пропорциональна квадрату расстояния. А если учесть кривизну пространства вблизи притягивающих масс? Достаточно в формулах заменить R^3 на $R^{2,9}$, и мы сможем «смоделировать» орбиту ближайшей к Солнцу планеты Меркурий, испытывающей влияние искривленности околосолнечного про-

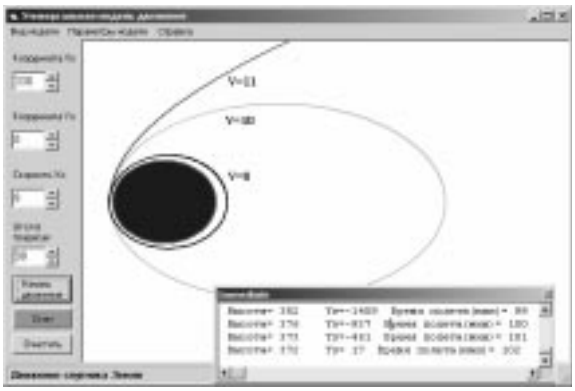


Рисунок 9

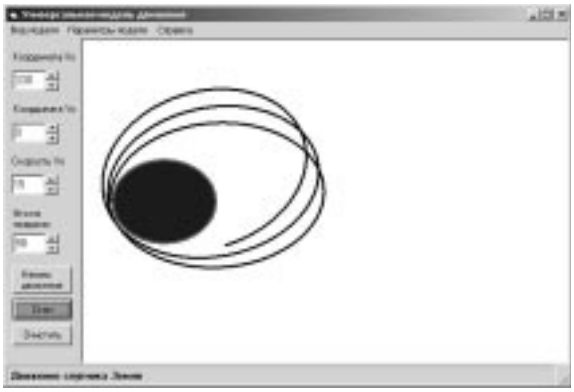
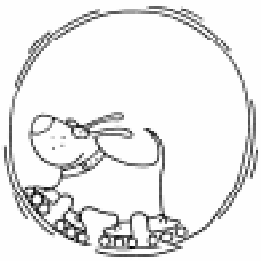


Рисунок 10

странства (рисунок 10) [5]. А можно смоделировать несуществующую Вселенную. Заменив в формулах R^3 на $R^{2,5}$, выбрав скорость 100 и сместив модель Земли к центру окна изображения, получите красивый рисунок траектории спутника.

ПОСТРОЕНИЕ УНИВЕРСАЛЬНОЙ МОДЕЛИ ДВИЖЕНИЯ



Создадим программу в среде Visual Basic 6.0 для демонстрации универсальной модели движения.

Первая часть – *визуальное программирование*.

1. Поместим в Toolbox – стандартный блок инструментов VB – дополнительные элементы управления и контроля. Для этого выберем опцию Components в меню Project. В появившемся списке вкладки Controls поставим галочку против элементов Microsoft Windows Common Controls 6.0 (содержит, в частности, объект StatusBar – Строка состояния), Microsoft Windows Common Controls-2 6.0 (содержит объект UpDown – Счетчик), Microsoft Common Dialog Control 6.0 (для отображения в программе стандартного окна выбора цвета – Color).

2. Разместим слева на форме (Form) массив из четырех Текстовых окон (TextBox) для задания начальных параметров модели: координаты X_0, координаты Y_0, скорости V_0, начального угла в градусах. Для обозначения каждого Текстового окна используется соответствующий массив Меток (Label).

3. К каждому текстовому полю присоединим свой элемент Счетчик (UpDown). Соответствие указывается в свойстве BuddyControl или выбирается автоматически при AutoBuddy = True. Если установить BuddyProperty = Default, то элемент UpDown будет автоматически обновлять стандартное свойство Text связанного с ним элемента управления TextBox. Определите минимальное (Min), максимальное (Max) и величину изменения (Increment) числовых значений, определяемых каждым объектом UpDown.

4. Разместим по нижнему краю формы объект Строка состояния (StatusBar) для отображения необходимой информации в процессе работы программы (о выбранной демонстрационной модели). Щелкнув правой кнопкой мыши на объекте StatusBar, выберем вкладку Панели (Panels) диалогового окна Properties Pages и установим свойство AutoSize = sbrSpring для первой панели, в результате чего она заполнит все пространство строки состояния. Выберем жирный шрифт (Bold) на вкладке Font.

5. Поместим на форму объект Диалоговое окно (CommonDialog), присвоим ему имя cdlColor.

6. Нанесем на форму три Командные кнопки, изменив их свойство Caption на «Начать движение», «Стоп» и «Очистить», соответственно.

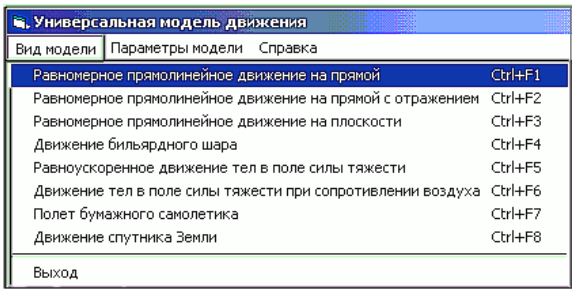


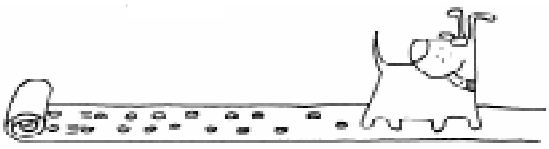
Рисунок 11

7. Создадим меню из трех главных пунктов «Вид модели», «Параметры модели» и «Справка». В первом пункте создадим массив из восьми подпунктов, обозначающих вид конкретной модели, присвоив каждому элементу массива еще и комбинацию горячих клавиш (рисунок 11), там же, после разделительной черты, поместим и подпункт «Выход». В пункте меню «Параметры модели» создадим два подпункта «Движение с траекторией» и «Цвет траектории». В подпункте «Движение с траекторией» используем свойство Checked (рядом с названием будет ставиться галочка при активизации данного подпункта). В пункте «Справка» можно поместить краткую информацию о данной программе.

8. Нанесем на форму объект Таймер, присвоив свойству Interval значение 5 (это означает, что каждые 5 миллисекунд будет срабатывать событие Timer).

9. И, наконец, нанесем на форму основной элемент управления Окно рисунка (Picture Box), присвоив свойству Align значение 4 – Align Right (выравнивание по правому краю формы), а свойству AutoRedraw – значение True. Для свойства FillColor выберем синий цвет, а для FillStyle – значение Solid.

Вторая часть – *написание кода программы*.



1. В данном приложении будут использоваться следующие константы: ускорение свободного падения (**g**), число Пи (**pi**), гравитационная постоянная (**Gr**), масса Земли (**M**), радиус Земли (**R3**), максимальные значения координат по оси абсцисс (**Xmax**) и по оси ординат (**Ymax**) для масштабирования Окна рисунка. Также понадобятся следующие глобальные переменные: координаты начального положения модели (**x0**, **y0**); начальная скорость и ее составляющие по осям координат (**V0**, **V0x**, **V0y**); начальный угол (**alf**); переменная, обозначающая вид модели (**vid**); коэффициент сопротивления воздуха (**ac**); коэффициент подъемной силы (**ap**); расстояние от центра Земли до спутника (**Rs**); шаг по времени (**dt**); время обращения вокруг Земли (**T**); радиус модели (**r**); величина **GrM**, равная произведению **Gr** на **M**; номер цвета для траектории (**col**). Описываем их в общем разделе описаний General Declarations.

```
Const pi = 3.1415926, g = 9.8 '[м/сек2]
Const Gr = 6.67E-20  '[км3/(кг*сек2)]
Const M = 6E+24      '[кг]
Const R3 = 6370      '[км]
Const Xmax = 300, Ymax = 200
Dim x0, y0, v0, v0x, v0y, alf, _
vid,  ac, ap, Rs, dt, T, r, GrM, col
```

Примечание. Поскольку ширина страницы книги не бесконечна, некоторые операторы программы разбиты на несколько строк. В конце таких строк используется символ подчеркивания (_) с предваряющим пробелом, что является стандартным символом продолжения языка Visual Basic. При вводе текста в окно редактора вы вполне можете опустить символы продолжения и записать оператор в одной строке. Обе записи правомерны, и компилятор правильно их воспримет.

2. Первое событие после загрузки Формы – выбор модели движения в пункте меню «Вид модели». В процедуре обработки данного события в статусную строку помещаем название выбранной модели, в переменной **vid** запоминаем номер этой модели, равный индексу соответствующего подпункта меню, и вызываем общую процедуру ввода данных **Input_Data**.

```
Private Sub mnu_TypeOfMotion_Click _
(Index As Integer)
    StatusBar1.Panels(1).Text = _
    mnu_TypeOfMotion(Index).Caption
    vid = Index
    CmdStop.Enabled = False
    Input_Data
End Sub
```

3. Описание процедуры **Input_Data**:

· с помощью свойства Scale масштабируем Окно рисунка;

· присваиваем начальные значения следующим переменным: шагу по времени,

координатам центра и радиусу модели, начальной скорости и углу;

· в зависимости от вида модели в операторе выбора либо меняем некоторые параметры, либо вводим через Окно ввода (InputBox) новые значения таких параметров, как коэффициент сопротивления воздуха, либо коэффициент подъемной силы;

· при выборе модели 8 (спутник Земли) задаются специфические параметры и новое масштабирование;

· в Текстовых окнах выводятся соответствующие параметры модели, а в Окне рисунка появляется модель в виде небольшого шарика.

```
Sub Input_Data()
  Picture1.Scale (0, Ymax)-(Xmax, 0)
  dt = 0.01
  r = 2
  v0 = 50: alf = 0
  x0 = 5: y0 = 10
  Select Case vid
   Case 3 To 7
    alf = 25
    ac = 0.1: ap = 0.02
    If vid >= 6 Then ac = InputBox _
    ("Коэф. сопротивления воздуха", _
     "Окно ввода", ac)
    If vid = 7 Then ap = InputBox _
    ("Коэф. подъемной силы", _
     "Окно ввода", ap)
   Case 8
    GrM = Gr * M
    dt = 2
    v0 = 8: alf = 90
    x0 = 330 '[км] - высота спутника
    y0 = 0
   Picture1.Scale (-10000, 25000) _
   -(50000, -25000)
   End Select
  Text1(1).Text = x0
  Text1(2).Text = y0
  Text1(3).Text = v0
  Text1(4).Text = alf
End Sub
```

4. В процедуре обработки события Щелчок на пункт меню «Движение с траекторией» необходимо выполнить следующие действия:

· сделать отмеченным галочкой (либо снять пометку) данный пункт меню,

· при выборе движения с траекторией задать режим рисования обычный (поверх), а без траектории – режим с наложением цветов.

```
Private Sub mnu_Track_Click()
  mnu_Track.Checked = _
  Not mnu_Track.Checked
  If mnu_Track.Checked Then
    Picture1.DrawMode = vbCopyPen
  Else
   Picture1.DrawMode = vbNotXorPen
  End If
End Sub
```

5. В процедуре обработки события Щелчок на пункт меню «Цвет траектории» необходимо отобразить диалоговое окно выбора цвета. В его свойстве Color будет размещаться числовой код выбранного цвета, который необходимо запомнить в переменной **col**.

```
Private Sub mnu_Color_Click()
   cdlColor.ShowColor
   col = cdlColor.Color
End Sub
```

6. Следующее событие – щелчок на Командную кнопку «Начать движение». В процедуре обработки данного события считываются из текстовых полей начальные параметры движения, которые могут быть предварительно изменены «исследователем». Для модели со спутником (**vid = 8**) рисуется Земля (голубая планета) и выбирается система отсчета относительно центра Земли. Дается зеленый цвет «светофора» кнопке «Стоп», а свойству Enabled объекта Timer присваивается значение True, чтобы «включить» демонстрацию движения для выбранной модели.

```
Private Sub CmdStart_Click()
  x0 = Text1(1).Text
  y0 = Text1(2).Text
  v0 = Text1(3).Text
  alf = Text1(4).Text * pi / 180
  If vid = 8 Then
    T = 0
    x0 = -(R3 + x0)
    Picture1.Circle (0, 0), R3, _
    QBColor(3), , , 0.8
  End If
  Picture1.Circle (x0, y0), r
  v0x = v0 * Cos(alf)
```

```
  v0y = v0 * Sin(alf)
  CmdStop.Enabled = True
  CmdStop.BackColor = &HFF00&
  Timer1.Enabled = True
End Sub
```

7. По событию от таймера перерисовывается объект модели (круг) в состоянии спустя промежуток времени **dt**, а для этого производится расчет изменения компонент скорости. Затем определяются координаты нового положения и рисуется (согласно меню) либо сам объект, либо его траектория выбранным цветом. Учитывается: для модели 2 и 4 отражение от границ, для модели 5, 6, 7 – достижение нижней границы, для модели 8 в окне отладки Immediate печатается дополнительная информация о положении спутника и времени полета (таким образом, можно определять, например, апогей и период обращения). В конце процедуры переприсваиваются значения скорости в точке и ее координаты.

```
Private Sub Timer1_Timer()
  Select Case vid
  Case 1 To 4
    v1x = v0x
    v1y = v0y
  Case 5
    v1x = v0x
    v1y = v0y - g * dt
  Case 6
    v1x = v0x - (ac * v0x) * dt
    v1y = v0y - (g + ac * v0y) * dt
  Case 7
    V = Sqr(v0x ^ 2 + v0y ^ 2)
    v1x = v0x - (ac * v0x + ap * _
    V *  v0y) * dt
    v1y = v0y - (g + ac * v0y - _
    ap * V * v0x) * dt
  Case 8
    Rs = Sqr(x0 ^ 2 + y0 ^ 2)
    v1x = v0x - (GrM * x0 / _
    Rs ^ 3)* dt
    v1y = v0y - (GrM * y0 / _
    Rs ^ 3)* dt
  End Select
    x = x0 + (v0x + v1x) * 0.5 * dt
    y = y0 + (v0y + v1y) * 0.5 * dt

  If mnu_Track.Checked Then
   Picture1.Line (x0, y0)-(x, y), col
  Else
   Picture1.Circle (x0, y0), r
   Picture1.Circle (x, y), r
  End If
  '-------------------------------
  If vid = 4 Or vid = 2 Then
      If x > Xmax - r Or x < r _
      Then v1x = -v1x
      If y > Ymax - r Or y < r _
      Then v1y = -v1y
  End If

  If vid >= 5 And vid <= 7 Then
  If y < r Then Timer1.Enabled = False
  End If

  If vid = 8 Then
   T = T + dt
   If CLng(T) Mod 60 = 0 Then _
   Debug.Print "Высота="; _
   CLng(Rs - R3), " Yз="; CLng(y); _
   "Время полета(мин)="; CLng(T / 60)
   End If
  '-------------------------------
  v0x = v1x: v0y = v1y
  x0 = x: y0 = y
End Sub
```

8. Следующее событие – щелчок на Командную кнопку «Стоп». В процедуре обработки данного события Таймер либо «выключается», то есть приостанавливается движение модели, либо «включается», то есть продолжается движение модели. Дается либо зеленый цвет «светофора» кнопке «Стоп», либо красный.

```
Private Sub CmdStop_Click()
    Timer1.Enabled = Not Timer1.Enabled
    If Timer1.Enabled Then
     CmdStop.BackColor = &HFF00&
     Else
     CmdStop.BackColor = &HFF&
     End If
End Sub
```

9. В процедуре обработки события Щелчок на Командную кнопку «Очистить» очищается Окно рисунка.

```
Private Sub CmdCls_Click()
   Picture1.Cls
End
```

10. Последнее событие – щелчок на пункт меню «Выход». В процедуре

обработки этого события при подтверждении выхода завершается работа программы.

```
Private Sub mnu_Exit_Click()
   If MsgBox("Вы действительно _
   желаете выйти?", vbYesNo, _
   "Выход") = vbYes Then End
End Sub
```

11. Запустить программу на выполнение, протестировать ее и завершить.

Работая с данной программой, в одном и том же Окне рисунка можно получать траектории движения модели при разных значениях параметров и даже траектории движения различных моделей. Таким образом, можно исследовать влияние на модель начальной скорости, угла наклона, коэффициентов сопротивления воздуха и подъемной силы, проводить сравнительный анализ разных моделей между собой.

Задание 1. «Моделирование электрического поля двух зарядов» (для самостоятельного выполнения).

Согласно закону Кулона, на пробный заряд q, помещенный на расстоянии r от заряда Q, действует сила

$$F = \frac{q \cdot Q}{r^2}$$

(формула аналогична гравитационной силе).

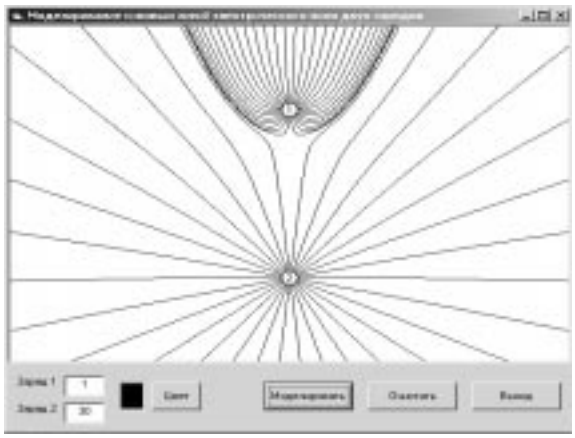


Рисунок 12

Для зарядов одного знака это – сила отталкивания, для противоположных – сила притяжения. Напряженность поля E в точке, где находится заряд q, определяется как $E = F / q$. Для случая нескольких зарядов напряженность E будет представлять алгебраическую сумму (или просто сумму в проекциях по осям координат) напряженности полей, создаваемых каждым зарядом в отдельности. Касательная в каждой точке силовой линии определяет направление результирующего вектора E.

Создать приложение для графического представления силовых линий электрического поля двух зарядов (диполя) с оформлением, подобным оформлению на рисунке 12, и выполняющее следующие функции:

1) Возможность задания величины зарядов через Текстовые окна.

2) Возможность задания цвета для изображения силовых линий с помощью стандартного диалогового окна.

3) Возможность очистки Окна рисунка.

4) Начало процесса моделирования и завершение программы осуществляются соответствующими Командными кнопками.

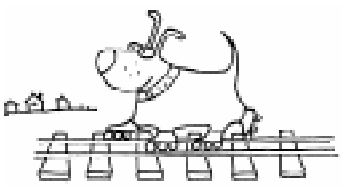
Указания к выполнению задания 1.

Для каждого заданного заряда поместить пробный единичный заряд того же знака на небольшом удалении от исходного и далее переместить (прорисовывая) на малое расстояние Δr в направлении вектора E.

Принять следующую точку за исходную и продолжить процесс до попадания пробного заряда за пределы окна просмотра или в ε-окрестность второго заряда диполя. Таким образом, получится приближение одной силовой линии.

Для изображения нескольких силовых линий использовать цикл по углу от 0 до 2π. Аналогично построение электрического поля нескольких зарядов.

Задание 2. Выбор места строительства железнодорожной станции (для самостоятельного выполнения).



Рассмотрим известную задачу «О выборе места строительства железнодорожной станции» [6]. Вот ее постановка.

В районе имеется несколько населенных пунктов. По территории района проходит железная дорога. Планируется построить железнодорожную станцию и связать ее с населенными пунктами грунтовыми дорогами. Требуется определить место наилучшего расположения железнодорожной станции, исходя из следующих принципов.

1) С точки зрения экономии средств на прокладку дорог наилучшим будет то расположение станции, при котором суммарная длина дорог будет наименьшей. Такой принцип носит название *принципа экономической целесообразности.*

2) С точки зрения социальных служб необходимо, чтобы дорога от самого удаленного пункта до станции стала наиболее короткой. В этом состоит *принцип социальной справедливости*.

3) Задав степень компромисса удовлетворения вышеуказанных принципов, можно получить новое расположение станции. Таким образом, будет выполнен *принцип компромисса*.

Разрабатывая модель к данной задаче, можно высказать ряд *предположений*, например, о рельефе местности, железной дороге и населенных пунктах. Предположим, что местность – это декартова плоскость, населенные пункты – точки на плоскости, железная дорога проходит вдоль оси абсцисс, а дороги – прямолинейные отрезки, соединяющие точки-пункты с точкой-станцией.

Исходные данные – это количество населенных пунктов N и их координаты $\{X_i,\ Y_i\}_{i=1,\ ...,\ N}$.

Результат – координата X_c железнодорожной станции (по предположению, координата $Y_c = 0$).

Для окончательного построения модели осталось задать *связи* между исходными данными и результатом. Для разных принципов они будут разными. Пусть S_i – расстояние от i-го пункта до станции, равное $\sqrt{(X_i - X_c)^2 + (Y_i - Y_c)^2}$, тогда для всевозможных значений X_c:

1) по принципу экономической целесообразности, необходимо найти min функции

$$f_1(\mathrm{X_c}) = \sum_i S_i ,$$

2) по принципу социальной справедливости, необходимо найти min функции

$$f_2(X_c) = \max_i(S_i) .$$

3) по принципу компромисса, необходимо найти min функции

$$f_3(X_c) = k \cdot f_1(X_c) + (1 - k) \cdot f_2(X_c),$$

где k – степень компромисса (коэффициент от 0 до 1).

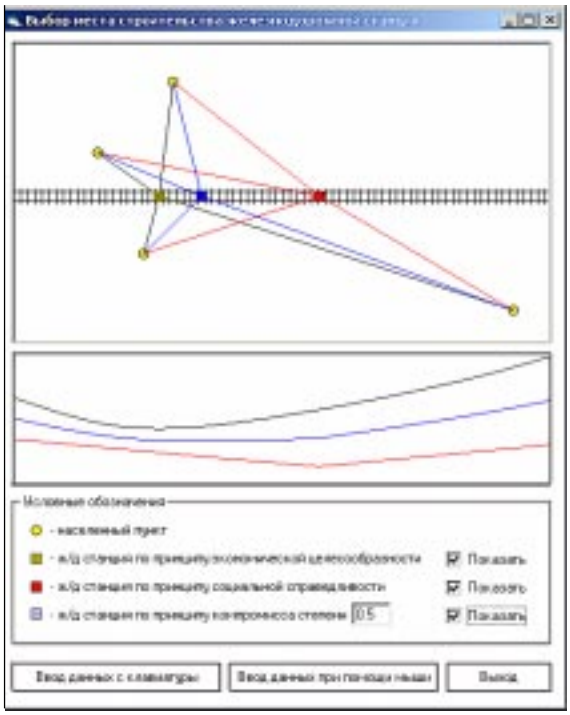


Рисунок 13

Создать приложение, решающее задачу о выборе места строительства железнодорожной станции с оформлением, например, как на рисунке 13, и выполняющее следующие функции:

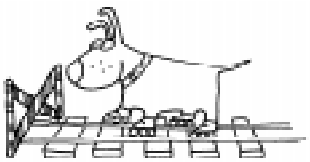
1) Ввод числа населенных пунктов с указанием их координат должен осуществляться как с клавиатуры через Окно ввода, так и непосредственно с помощью манипулятора «мышь».

2) Для наглядности моделирования необходимо использовать:

· Графическое окно для изображения пунктов и положения станции, согласно выбранному принципу. Пункты изображаются в графическом окне относительно его центра как начала координат.

· Графическое окно для построения графиков, соответствующих выбранным принципам (моделям). Ордината минимума графика определяет искомое место расположения станции для исследуемой модели.

3) Осуществить возможность независимого выбора с помощью Флажков



принципов наилучшего расположения железнодорожной станции с графическим показом соответствующего расположения различным цветом, согласно условным обозначениям.

4) Предусмотреть вывод приближенного значения координаты *X* станции при задании данных с клавиатуры.

Указания к выполнению задания 2.

Для построения функции выбранной модели привести координаты пунктов в соответствие с координатами графического окна и найти оптимальное расположение станции путем перебора ее координаты X_c от 0 с шагом 1 до значения ширины графического окна в пикселах.

Литература.

1. Каймин В.А., Питеркин В.М., Уртминцев А.Г. Информатика. Пособие по углубленному изучению информатики для учащихся средних школ и поступающих в технические университеты. М.: БРИДЖ, 1994.
2. Хилькевич С.С., Зайцева О.А. Как построить траекторию. Векторные уравнения в кинематике. «Квант», № 7, 1987.
3. И. Воробьев. Океанская зыбь. «Квант», № 9, 1992.
4. Носков Н.Н., Столбоушкин С.К. Спутник на дисплее. «Квант», № 10, 1989.
5. Виленкин Н.Я. О кривизне. «Квант», № 4, 1992.
6. А.Г. Гейн и др. Основы информатики и вычислительной техники, X–XI классы. М: Просвещение, 1996.

Паньгина Нина Николаевна,
преподаватель ОИиВТ
школы-лицея № 8, г. Сосновый Бор.