Федеральное агентство образования

Димитровградский институт технологий,

управления и дизайна

Курсовая работа
по курсу: “Компьютерное моделирование”
на тему: “Моделирование движения тела, брошенного под углом к горизонту вблизи поверхности земли”

Выполнил студент гр. ВТ-31:
Потеренко А.Г.
Проверил преподаватель:
Солдатиков А.А.
Дата сдачи курсовой работы:

_____

Оценка:

_____

Димитровград 2006г.

Содержание

1. Задание

Разработать программу, моделирующую движение тела, брошенного под углом к горизонту. Угол лежит в пределах от 0 до 90°. Должны выводиться все сведения о движении материального объекта: его траектория по горизонтали и вертикали, максимальное и минимальное положение тела. В некотором пределе по обеим осям координат должно выводится на экран само движение тела. Движение тела также должно зависеть от скорости ветра - пользователь сам задает вид ветра: попутный или встречный, а также скорость его и коэффициент сопротивления ветра (он в свою очередь зависит от силы ветра, приходящийся на единицу длины).

2. Введение

Физика в наше время играет большую роль в жизни каждого. Использование компьютеров при решении задач физики, несомненно, является стандартом де-факто. Например, моделирование движения объекта позволяет визуально просмотреть, как движется объект, его характеристики в текущий момент времени. Механика является одним из разделов высшей физики. Механика, как предмет, сформировалась в начале 18 века, когда Ньютон впервые сформулировал общие законы движения тела вблизи поверхности земли. Он же разработал общие положения этой науки. Только благодаря им, автор смог разработать данную программу моделирования движение тела, брошенного под углом к горизонту.

Механика также является одним из любимых разделов физики автора программы. При решении задач физики можно не учитывать внешние факторы, такие как сопротивление среды. Но в данной программе этот фактор учитывается.

Решение задач движения тела вблизи поверхности земли играют большую роль в промышленности, военной технике.

3. Назначение и область применения

Данная программа предназначена для решения задач физики, в частности механики, и моделирования движения тела вблизи поверхности земли. Цель программы – показать пользователю как именно может двигаться объект, брошенный под углом к горизонту.

Данная программа может применяться студентами физических специальностей при решении задач механики и проверки правильности решения.

Область применения: институты моделирования физического взаимодействия тел, институты Министерства образования.

4. Выбор состава технических и программных средств

Аппаратная платформа тестируемой программы:

Тип ЦП	Intel Pentium 4, 2400 MHz (18 x 133)
Системная плата	Gigabyte GA-8PEMT4(-C) (3 PCI, 1 AGP, 2 DDR DIMM, Audio, LAN)
Чипсет системной платы	Intel Brookdale i845PE
Системная память	256 Мб (PC2700 DDR SDRAM)
Видеоадаптер	NVIDIA GeForce FX 5200 (128 Мб)
3D-акселератор	nVIDIA GeForce FX 5200
Монитор	LG Flatron L1510P (Analog) [15" LCD] (1303140602)
Дисковый накопитель	Maxtor 6Y060L0 (60 Гб, 7200 RPM, Ultra-ATA/133)

Минимальные аппаратные требования:

Тип ЦП	Intel Pentium 386
Системная память	128 Мб
Видеоадаптер	AGP
Внешняя память	3 Мб

Программные средства разработки:

Операционная система	Microsoft Windows XP Professional
Пакет обновления ОС	Service Pack 1
DirectX	4.09.00.0904 (DirectX 9.0c)
Среда разработки	Borland C++ Builder 6.0

Программный продукт Borland C++ Builder 6.0 был выбран как наиболее перспективный для создания графических приложений. При использовании определенных опций можно сделать довольно качественную прорисовку выводимых изображений. Также, используя данный программный продукт, можно создать свое приложение очень быстро. В частности, интерфейс пользователя разрабатывается очень быстро благодаря визуальному проектированию программы. Нет необходимости “писать руками” обработчики событий от интерфейсных компонентов. Уделяется внимание только разработке прикладной части программы.

5. Организация входных и выходных данных

Входные данные: начальная скорость тела, масса тела, угол к вертикали, коэффициент сопротивления телу по горизонтали, скорость ветра (встречный, попутный).

Выходные данные: максимальная высота, на которую поднимается тело, максимальная дальность полета, время полета.

Также программа позволяет находить требуемые характеристики по различному критерию, т.е. пользователь может сам выбирать, что ему нужно найти, исходя из известных величин.

Входные данные могут быть введены пользователем сразу же после запуска программы. После ввода входных данных начинается процесс расчета. Когда программа завершает процесс вычислений, пользователю в текстовых окнах выдаются запрашиваемые данные. Затем пользователь может снова и снова вводить данные для просчета. Время, затраченное на вычисления характеристик тела, зависит от начальных данных. Возможны ситуации, когда пользователь будет ждать ответа очень долго т.к. полет тела происходит в режиме реального времени (при желании можно отключить анимацию и получить ответ “мгновенно”).

6. Разработка программы

“Жизненный цикл программы”:

- Постановка задачи.
- Решить задачу механики.
- Непосредственное программирование задачи.
- Тестирование программы.

Решение задачи механики. Используя второй закон Ньютона[1], можно найти ускорение тела вдоль оси OX. Ускорение тела вдоль этой оси сообщает ему ветер. Это ускорение[1] равно:

$$a = \left| \frac{k_c \cdot \upsilon_B}{m} \right|$$

Знак ускорения: “+” – если попутный ветер, то есть тело ускоряется и “-” – если встречный ветер.

k_c - коэффициент сопротивления ветра, m – масса тела, υ_B - скорость ветра.

Само уравнение движения тела описывается уравнениями[1]:

$$x = x_0 + \upsilon_0 \cos(\alpha) t \pm \frac{at^2}{2}$$

$$y = y_0 + \upsilon_0 \sin(\alpha) t - \frac{gt^2}{2}$$

$$\upsilon_x(t) = \upsilon_0 \cos(\alpha) \pm at$$

$$\upsilon_y(t) = \upsilon_0 \sin(\alpha) - gt$$

υ_0 - начальная скорость тела, α - угол к вертикали.

Из этих уравнений можно найти время движения:

$$t_{дв} = \frac{2\upsilon_0 \sin(\alpha)}{g}$$

---

1. См. литературу п.2,п.3

Переменная t принимает значение от 0 до $t_{дв}$. Пробегая все эти значения и подставляя в уравнения выше, можно найти требуемые характеристики.

Непосредственное программирование задачи[2]. При нажатии на кнопке “Выстрел” вызывается процедура решения задачи:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
```

В которой проверяется корректный ввод данных пользователем.

```
if (StrToFloat(Edit2->Text)<=0)
  {
   ShowMessage("Тело не имеет массу! Система не имеет решений!");
   return;
  }
if (StrToFloat(Edit3->Text)<=0)
  {
   ShowMessage("Тело не имеет начальной скорости! Система не имеет решений!");
   return;
  }
```

Если данные прошли проверку, то вычисляется время движения. Так как по вертикали сопротивление отсутствует, то скорость ветра не влияет на время движения. Учитывается начальная скорость, начальный угол движения тела. Затем пользователю выводится время движения объекта. То есть время заранее известно.

```
T=vo*sin(a*M_PI/180)/g+sqrt(vo*vo*sin(a*M_PI/180)*sin(a*M_PI/180)+2*g*yo)/g;

Edit13->Text=VarToStr(T);
```

После этого либо запускается таймер для анимации движения тела или показываем готовый результат :

```
if (Form1->CheckBox1->Checked==true)
  {
    bool flirt=false;
    while (flirt!=true)
    {
    int C1=520;
    int C2=150;
    y=yo+vo*sin(a*M_PI/180)*t-(g*t*t)/2;
    float A=kc*vv/m;
    if (VETER==2)
     x=xo+vo*cos(a*M_PI/180)*t+A*t*t/2;
    if (VETER==1)
     x=xo+vo*cos(a*M_PI/180)*t-A*t*t/2;
    if (VETER==0)
```

---

2. См. литературу п.1

```
    x=xo+vo*cos(a*M_PI/180)*t;
    Edit11->Text=VarToStr(y);
    if (y>StrToFloat(Edit7->Text)) Edit7->Text=VarToStr(y);
    if (x>StrToFloat(Edit8->Text)) Edit8->Text=VarToStr(-40+x);
    Edit10->Text=VarToStr(-40+x);
    Edit12->Text=VarToStr(t);
    t=t+0.1;
    if (T<t)
     {
      Button1->Enabled=true;
      Edit11->Text=VarToStr(0);
      Edit10->Text=Edit8->Text;
      Edit12->Text=VarToStr(ceil(T)-1);
      flirt=true;
     }
    }
   }
  else
    Timer2->Enabled=true;
```

В приведенном ниже фрагменте программы запускается цикл подсчета необходимых характеристик. Если время истекло, то цикл заканчивается и пользователю выводится готовый результат в отдельных окнах типа Edit.

Если анимация включена, то в таймере происходит задержка по времени и тело “движется”. Движение тела реализуется движением компонента Image14 на форме. Координаты этого компонента совпадают с координатами самого тела на плоскости.

```
void __fastcall TForm1::Timer2Timer(TObject *Sender)
{
  int C1=520;
  int C2=150;
  y=yo+vo*sin(a*M_PI/180)*t-(g*t*t)/2;
  float A=kc*vv/m;
  if (VETER==2)
    x=xo+vo*cos(a*M_PI/180)*t+A*t*t/2;
  if (VETER==1)
    x=xo+vo*cos(a*M_PI/180)*t-A*t*t/2;
  if (VETER==0)
    x=xo+vo*cos(a*M_PI/180)*t;
  Edit11->Text=VarToStr(y);
  if (y>StrToFloat(Edit7->Text)) Edit7->Text=VarToStr(y);
  if (x>StrToFloat(Edit8->Text)) Edit8->Text=VarToStr(-40+x);
  Edit10->Text=VarToStr(-40+x);
  Edit12->Text=VarToStr(t);
  Image14->Visible=true;
  Image14->Top=C1-ceil(y);
  Image14->Left=C2+ceil(x);
  t=t+0.1;
  if (T<t)
   {
    Timer2->Enabled = false;
    Button1->Enabled=true;
    Edit11->Text=VarToStr(0);
    Edit10->Text=Edit8->Text;
```

```
    Edit12->Text=VarToStr(ceil(T)-1);
    ////////////////////Создаем яму/////////////////////////////////////////////
    if (Image14->Left<125 || Image14->Left>253)
    {
      TRect SOURCE,DEST;
      DEST.Left=0+Image14->Left-15;DEST.Top=0+Image14->Top-123;
      DEST.Right=Image15->Width+Image14->Left-15;
      DEST.Bottom=Image15->Top+Image14->Top-123;
      SOURCE.Left=0;SOURCE.Top=0;SOURCE.Right=Image15->Width;
      SOURCE.Bottom=Image15->Top;
      Image1->Canvas->CopyMode=cmSrcAnd;
      Image1->Canvas->CopyRect(DEST,Image15->Canvas,SOURCE);}
   else
    { Image16->Visible=true;Image14->Visible=false;
      Image4->Visible=false;Image5->Visible=false;Image6->Visible=false;
      Image7->Visible=false;Image8->Visible=false;Image9->Visible=false;
      Image10->Visible=false;Image11->Visible=false;Image12->Visible=false;
      Image13->Visible=false;
      ShowMessage("Вы повредили орудие! Начните игру заново!");
      Application->Terminate();}
   }
}
```

В этом обработчике таймера происходит все движение компонента Image14. Если время движения тела закончилось, то таймер завершает свою работу. Здесь же происходит подсчет координаты по горизонтали и по вертикали. В зависимости от направления ветра происходит подсчет координаты по горизонтали. Ускорение тела по горизонтали подсчитывается как:

```
float A=kc*vv/m;
```

На рис.1 изображена среда разработки и процесс программирования задачи.

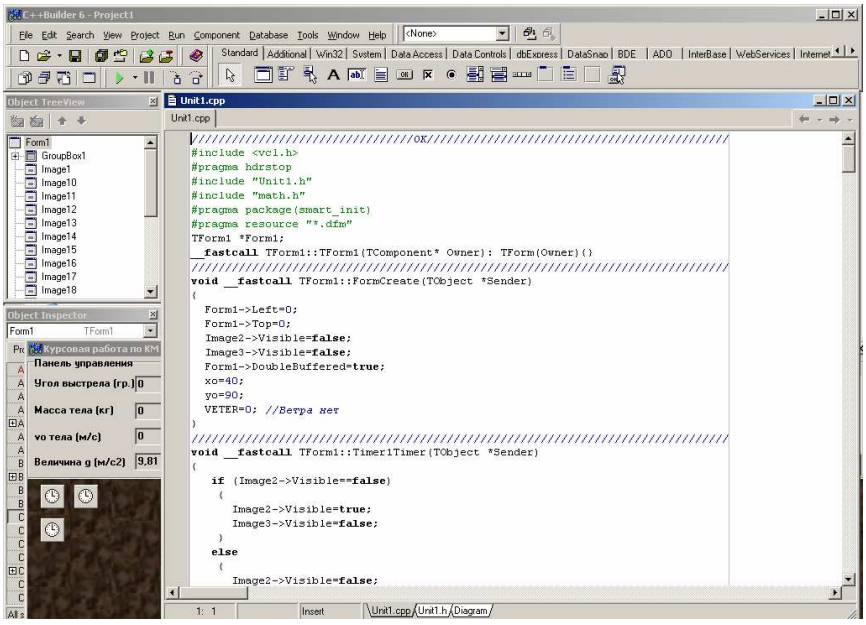

Рис. 1. Разработка программы

Далее показано, какие элементы формы используются пользователем и придают форме графический фон:

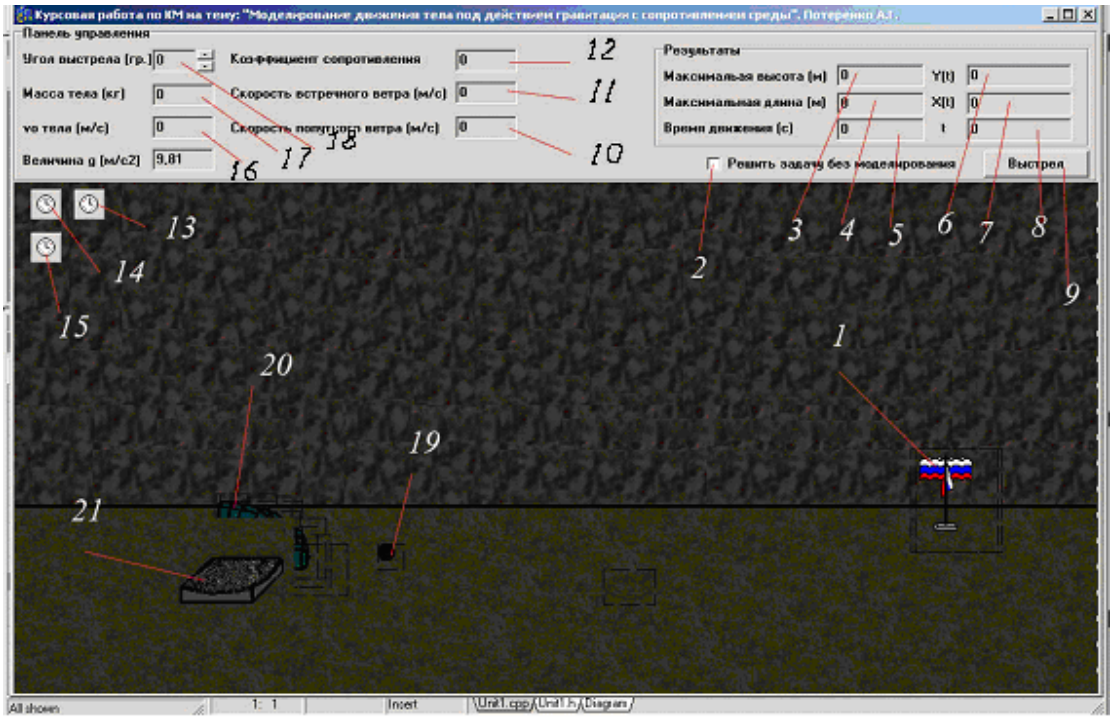

Рис. 2. Компоненты формы

1,19,20,21 - TImage компоненты. Используются для отображения формы флагов, отображения вида ствола орудия, самого движущегося тела.

13,14,15 – TTimer компоненты. Используются для анимации изменения формы флага, движения компонента тела.

2 – TCheckbox компонент. Позволяет делать выбор между анимацией движения тела и выводом решения без анимации.

3,4,5,6,7,8 – TEdit компоненты. Окна вывода, позволяющие выводить решения пользователю.

10,11,12,16,17,18 – TEdit компоненты. Окна ввода, с помощью которых пользователь может вводить данные в программу для решения поставленной задачи.

9 – TButton компонент. Кнопка запуска процесса для решения поставленной задачи.

Тестирование программы. Рассмотрим некорректные действия пользователя: пуск без задания начальных параметров. При этом выводятся соответствующие сообщения:

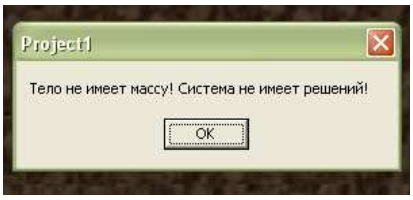

Рис. 3. Некорректный ввод пользователем

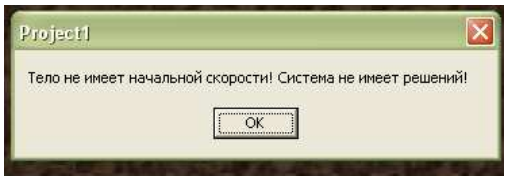

Рис. 4. Некорректный ввод пользователем

Если пользователь ввел некорректные значения, то:

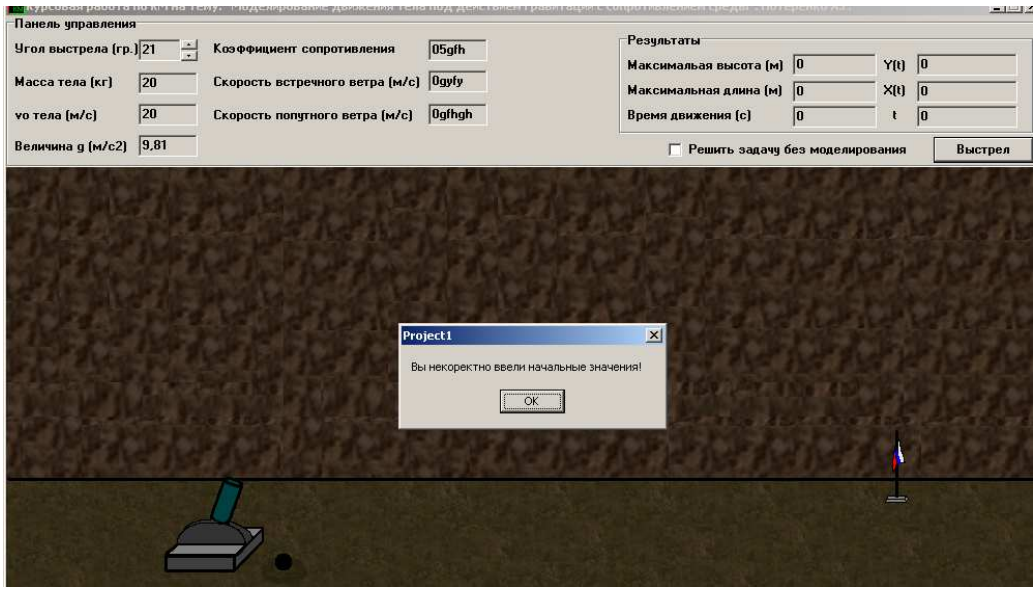

Рис. 5. Некорректный ввод пользователем

выводится соответствующее сообщение.

Если пользователь ввел правильные начальные значения:

Начальная скорость – 100 м/с

Угол – 30 градусов

Масса – 100

Коэффициент сопротивление – 70

Скорость встречного ветра – 0 м/с

Скорость попутного ветра - 80 м/с

В этом случае программа выдаст решение программы:

Максимальная высота – 478 м

Максимальная длина – 10769 м

Время движения - 18 с

При выборе опции анимации получится следующее:

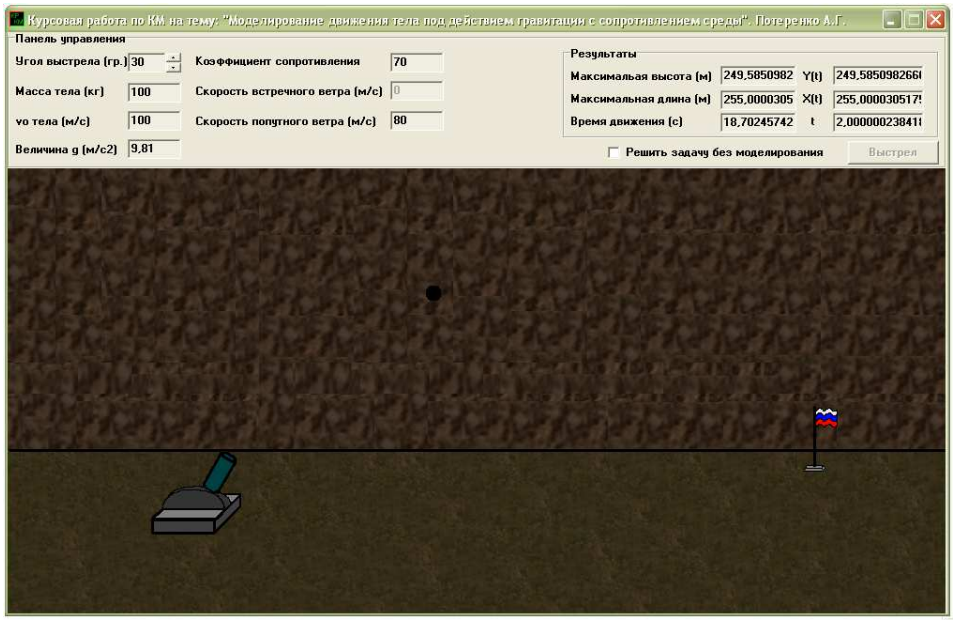

Рис. 6. Правильная работа программы

7. Спецификация программы

Данная программа включает файлы:

1. Project1.bpr – главный файл проекта
2. Project1.cpp – исходный код проекта – используется для создания главного окна Windows и запуска потока программы.
3. Unit1.h – заголовочный файл проекта, здесь описываются все визуальные объекты формы.
4. Unit1.cpp – главный модуль с исходным кодом программы.
5. Project1.exe – исполняемый модуль программы.

8. Заключение

Механика необходима ученым, студентам, аспирантам. Понимание данного предмета является главным для каждого студента, обучающегося на физических специальностях. Моделирование движения тела – способ совместить знание по физике и программирование. Данная программа должна помочь тем, кто решает задачи по механике в институте и не только. Программа окажет неоценимую помощь тем, кто составляет задачи по механике.

Автор выражает надежду, что программа произвела впечатление на студентов, которые отрицательно относятся к физике как к науке.

Для дальнейшего продолжения развития программы можно сделать программу в 3D. Это, несомненно, является большим недостатком, который в будущем автор программы попытается устранить.

9. Литература

1. Вэрсон Э.К. Руководство по C++Builder 6.0. Официальное руководство BCB. Лондон. 2004г.
2. Гаврилов Г.В. “Высшая физика”. Москва. Просвещение, 2001г.
3. Дедлов И.М. “Теоретическая механика”. Москва. Просвещение, 2001г.