Язык программирования Си++

Иванов А.П., Князева О.С.

Семинар 5. Введение в программирование графики и математическое моделирование.

1. Постановка задачи

В типовом задании требуется построить траекторию движения тела массы m в поле силы тяжести (ускорение св. падения g), если это тело вбрасывается на левой границе поля зрения на высоте h с горизонтальной начальной скоростью V0.

Тело может отскакивать от нижнего края экрана (упруго, или неупруго – с потерей f % энергии при каждом ударе о поверхность).

В момент выхода тела за правый кран экрана (ширину экрана будем считать равной L) или его остановки (при неупругом ударе), выполнение задачи заканчивается.

2. Программирование графики

Последовательность создания проекта:

1. Завести проект Win32 Console Application.
2. Выбрать пункт меню Project/Add New Item
3. В карточке выбрать C++ Source file, назвать его как-нибудь.
4. Повторить пункт 3.
5. В карточке выбрать Resource Script, тоже как-то назвать файл.
6. По щелчку правой кнопки мыши выбрать Add Resource.
7. В карточке выбора типа ресурса выбрать Dialog и нажать кнопку New.
8. Удалить с диалога кнопку Cancel.
9. В заголовке диалога написать (можно по-русски) название задачи.
10. В окне редактирования свойств диалога (правая кнопка мыши) надо поставить галочки Set Foreground и Visible.
11. Закрыть окно редактирования ресурсов.
12. Писать программу.

Рекомендуется сначала добиться того, чтобы программа хоть что-то нарисовала в графическом окне и лишь по достижении этого результата переходить к обсчету и рисованию траектории для конкретной физической задачи.

```
#include <iostream>
#include <windows.h>
#include "resource.h"
using namespace std;

/* Блок глобальных переменных (массивы заранее рассчитанных координат)
   в двух системах: физической (вещественные координаты) и пиксельной,
   связанной с текущим окном, в котором производится отрисовка траектории.
 */
int x1, y1, x2, y2;

int WINAPI DlgProc( HWND hDlg, WORD wMsg, WORD wParam, DWORD )
{
    PAINTSTRUCT ps;
```

```
    if( wMsg == WM_CLOSE || wMsg == WM_COMMAND && wParam == IDOK ) {
        EndDialog(hDlg,0);
    } else
    if( wMsg == WM_INITDIALOG ) {
/* Узнаем размер окна (ось OY направлена вниз): */
        RECT rc;
        GetClientRect(hDlg,&rc);
        int dx = rc.right - rc.left;
        int dy = rc.bottom- rc.top;
/* После этого нужно отмасштабировать физические координаты траектории,
   чтобы они помещались в поле зрения окна. То есть, нужно рассчитать
   для каждой точки траектории пиксельные координаты в системе координат
   данного окна. Само окно пользователь еще не видит.
 */
    } else
    if( wMsg == WM_PAINT ) {
        BeginPaint(hDlg,&ps);
/* Зададим цвет линий: */
        HPEN hPen = (HPEN)CreatePen(PS_SOLID,1,RGB(0,0,255));
        HPEN hOldPen = (HPEN)SelectObject(ps.hdc,hPen);
/*
  Вот что вообще можно здесь рисовать:
    MoveToEx(ps.hdc,int x,int y,NULL);
        ( последний параметр – адрес структуры POINT
          для возврата координат предыдущей позиции )
    LineTo(ps.hdc,int x,int y); (с первого пиксела, но без последнего)
    TextOut(ps.hdc,int x,int y,char* szText,lstrlen(szText));
    Rectangle(ps.hdc,int left,int top,int right,int bottom);
    Ellipse(ps.hdc,int left,int top,int right,int bottom);
    Polygon(ps.hdc,const POINT * lp,int nPoints);
    Polyline(ps.hdc,const POINT * lp,int nPoints);
    SetPixel(ps.hdc,int x,int y, RGB(red,green,blue));
 */

/* Здесь – производится отрисовка расчитанных пиксельных координат
   траектории в виде ломаной, в цикле по общему количеству точек.
 */
        POINT ptOld;
        MoveToEx(ps.hdc,x1,y1,&ptOld);
        LineTo(ps.hdc,x2,y2);

/* Перо нам больше не требуется, уничтожим его: */
        SelectObject(ps.hdc,hOldPen);
        DeleteObject(hPen);
        EndPaint(hDlg,&ps);
    }
    return 0;
}

void main()
{

/* Ввод параметров задачи: */
    cout << "Please, enter 4 coords:\n" << flush;

    cin >> x1 >> y1 >> x2 >> y2;
    cout << "x1 = " << x1 << "\ny1 = " << y1
         << "\nx2 = " << x2 << "\ny2 = " << y2 << "\n" << flush;
```

```
    /*
       Здесь, перед показом, нужно расчитать координаты
       всех точек траектории в физической системе координат.
       Массивы x-y координат должны быть доступны глобально -
       в DlgProc и в функции main.
    */
    DialogBox(NULL,MAKEINTRESOURCE(IDD_DIALOG1),NULL,(DLGPROC)DlgProc);
}
```

3. Метод решения

Отвлечемся от процесса визуализации и сосредоточимся на расчете координат тела в последовательные моменты времени.

Уравнение движения тела под действием силы тяжести записывается в виде:

$m\frac{d^2\vec{r}}{dt^2} = mg$ – это дифференциальное уравнение второго порядка. Сделаем замену переменной $\frac{d\vec{r}}{dt} = \vec{v}$, тогда получим систему дифференциальных уравнений первого порядка, которую будем решать методом конечных разностей (методом Эйлера).

$$\begin{cases} d\vec{v}/dt = \vec{g} \\ d\vec{r}/dt = \vec{v} \end{cases}$$

Так как мы заранее не знаем, какая будет скорость и как скоро тело вылетит за пределы поля зрения, то будем действовать следующим образом: зададим систему координат с центром в точке вброса тела, осью Y, направленной вертикально вниз и осью X, направленной вправо. При этом, начальное положение тела (0,0), а поверхность земли проходит по линии h оси Y.

В любой момент времени (между ударами), горизонтальная составляющая скорости сохраняется постоянной, а вертикальная изменяется по закону:

$V_y = gt$

Зададим некоторый дискретный шаг отсчета времени Δt.

Состояние тела в любой момент будет описываться четырьмя параметрами:

X, Y, V_x, V_y

Изменения этого состояния можно вычислять рекуррентно:

$X[i] = X[i-1] + V_x\Delta t$

$Y[i] = Y[i-1] + V_y[i-1]\Delta t$

$V_y[i] = V_y[i-1] + g\Delta t$

При этом, надо будет обнаружить два события:

- выход тела за нижнюю границу экрана (когда выполнится условие $Y[i] > h$)
- выход тела за правую границу экрана (когда выполнится условие $X[i] > L$)

В первом случае надо сменить знак скорости на противоположный (неупругий удар далее не рассматривается), а во втором – закончить просчет траектории и перейти к ее выводу, после чего можно будет завершить задачу.

После удара о поверхность, который поменяет знак вертикальной компоненты скорости на отрицательный (при неупругом ударе – еще уменьшится абсолютное значение скорости), формула изменения скорости приобретет вид:

$V_y = V_{y1} + gt$

где V_{y1} будет отрицательной величиной.

Описанный рекуррентный метод решения известен как метод Эйлера и применим, в том числе и к задачам, где внешняя сила меняется с течением времени. Однако, следует помнить, что точность этого метода – невысока, кроме того, он характерен накоплением ошибок вычислений с течением времени.

Для более точного решения поставленной задачи следует в выражении:

$$y(t_1) = y(t_0) + \int_{t_0}^{t_1} y'(t)dt$$

заменить определенный интеграл не по формуле прямоугольников (как это делается в методе Эйлера), а по формуле трапеций:

$$y(t_1) = y(t_0) + \Delta t\,(y'(t_0) + y'(t_1)) / 2$$

1. Вариант

Построить траекторию падения тела в поле силы тяжести, если тело начало движение в точке (x_0,y_0), с начальной скоростью (V_{x0},V_{y0}). Начальные условия задает пользователь.
Удар о землю абсолютно неупругий. Следовательно, траектория должна заканчиваться либо в точке удара тела о землю, либо в точке вылета тела за границы экрана.

2. Вариант

Построить траекторию падения тела в поле силы тяжести, если тело начало движение в точке (x_0,y_0), с начальной скоростью (V_{x0},V_{y0}). Начальные условия задает пользователь.
Удар о землю абсолютно упругий. Построение траектории должно заканчиваться в момент вылета тела за границы экрана.

3. Вариант

Построить траекторию падения тела в поле силы тяжести, если тело начало движение в точке (x_0,y_0), с начальной скоростью (V_{x0},V_{y0}). Начальные условия задает пользователь.
При ударе о землю тело теряет 10% своей кинетической энергии. Построение траектории должно заканчиваться в момент вылета тела за границы экрана.

4. Вариант

Построить траекторию движения заряженной частицы в электрическом поле с напряженностью $\vec{E}$. Частица начинает движение в точке (x_0,y_0), с начальной скоростью (V_{x0},V_{y0}). Построение траектории должно заканчиваться в момент вылета частицы за границы экрана.

Уравнение движения: $m\frac{d\vec{V}}{dt} = q\vec{E}$.

5. Вариант

Построить траекторию движения заряженной частицы в однородном магнитном поле **B**=(0,0,B). Частица начинает движение в точке (x_0,y_0), с начальной скоростью (V_{x0},V_{y0}). Построение траектории должно заканчиваться в момент вылета частицы за границы экрана.

Уравнение движения: $m\frac{d\vec{V}}{dt} = q[\vec{V} \times \vec{B}]$.

6. Вариант

Построить траекторию движения заряженной частицы в поле электрического диполя. Частица начинает движение в точке (x_0,y_0), с начальной скоростью (V_{x0},V_{y0}). Построение траектории должно заканчиваться в момент вылета частицы за границы экрана.

7. Вариант

Построить траекторию движения заряженной частицы в поле электрического квадруполя. Частица начинает движение в точке (x_0,y_0), с начальной скоростью (V_{x0},V_{y0}). Построение траектории должно заканчиваться в момент вылета частицы за границы экрана.

8. Вариант

Построить траекторию движения тела брошенного под углом к горизонту с учетом сопротивления воздуха ($\vec{F}_s = -\xi\vec{V}$). Тело начало движение в точке (0,0), с начальной

скоростью (V_{x0},V_{y0}). Траектория должна заканчиваться либо в точке удара тела о землю, либо в точке вылета тела за границы экрана.

9. Вариант

Построить траекторию движения тела брошенного под углом к горизонту, с учетом сопротивления воздуха, пропорционального квадрату скорости тела относительно воздуха. Тело начало движение в точке (x_0,y_0), с начальной скоростью (V_{x0},V_{y0}). Траектория должна заканчиваться либо в точке удара тела о землю, либо в точке вылета тела за границы экрана.

10. Вариант

Построить траекторию движения 2 заряженных частиц, соединенных пружинкой, жесткостью k, в однородном электрическом поле с напряженностью E. Расчет заканчивается, когда частицы вылетают за границы экрана.

11. Вариант

Построить траекторию движения частицы в стакане. Частица начинает движение в точке (x_0,y_0), с начальной скоростью (V_{x0},V_{y0}). Отражение от стенок стакана абсолютно упругое, внешних полей, действующих на частицу нет. Вычисление должно заканчиваться либо при вылете частицы из стакана, либо через определенное пользователем время T.

12. Вариант

Построить траекторию движения частицы в стакане, состоящем из дна и бесконечных вертикальных стенок. Частица начинает движение в точке (x_0,y_0), с начальной скоростью (V_{x0},V_{y0}). Отражение от стенок стакана неупругое с потерей 10% кинетической энергии, на частицу действует сила тяжести. Вычисление должно заканчиваться через определенное пользователем время T.

13. Вариант

Задача двух тел. Построить траекторию движения спутника массы m_1 вокруг планеты m_2.

Уравнение движения: $m_2 \frac{d^2\vec{r}}{dt^2} = -\gamma \frac{m_1 m_2}{|\vec{r}|^3}\vec{r}$

γ – гравитационная постоянная, r – радиус вектор от центра планеты к спутнику.

14. Вариант

Построить графики зависимости смещения из положения равновесия от времени для двух масс, сцепленных пружинкой и прикрепленных пружинками к боковым стенкам с учетом трения о нижнюю поверхность

15. Вариант

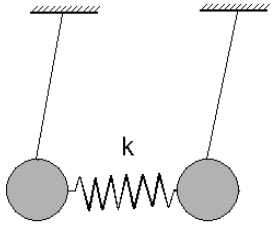
Задача трех тел. Построить траектории движения трех гравитирующих масс m_1, m_2, m_3. Уравнение движения – система дифференциальных уравнений:

$$m_j \frac{d^2\vec{r}_j}{dt^2} = -\gamma \sum_{k\neq j}^{3} \frac{m_k m_j}{\left|\vec{r}_k - \vec{r}_j\right|^3}(\vec{r}_k - \vec{r}_j),\ j=1,2,3.$$

Для задачи трех тел нужно сложить действующие на тело силы, рассчитав их независимо друг от друга.

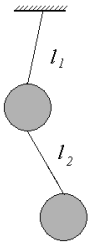
16. Вариант

Построить графики зависимости смещений из положения равновесия от времени для двух маятников, подвешенных рядом и соединенных невесомой пружиной.



17. Вариант

Построить графики зависимости смещений из положения равновесия от времени для двух маятников на жестком подвесе, подвешенных один к другому.

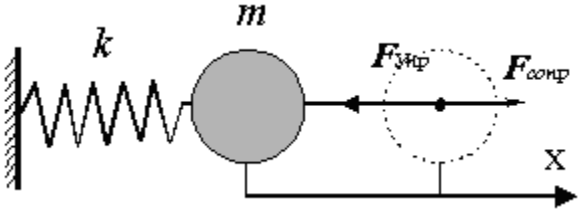


18. Вариант

Построить график зависимости смещения из положения равновесия от времени для груза массы m, подвешенного на пружине (пружинный маятник).

19. Вариант

Построить график зависимости смещения из положения равновесия от времени для шарика массы m, прикрепленного к стенке пружиной, жесткостью k (см. рис.), с учетом силы трения $\vec{F}_{\text{сопр}} = -\xi\vec{V}$.



если в начале его отклонили от равновесного положения на L.

20. Вариант

Построить график зависимости координат от времени для шарика под действием силы тяжести, помещенного в гладкую сферическую полость радиусом R.

21. Вариант

Построить график зависимости координаты от времени для положительно заряженного шарика (заряд Q, масса M), помещенного между двумя неподвижными точечными зарядами (+q). Шарик может двигаться только вдоль прямой, соединяющей эти два заряда.

22. Вариант

Построить траекторию броуновского движения частицы в вязкой среде ($\vec{F}_s = -\xi\vec{V}$), т.е. с учетом силы трения и случайной силы.

23. Вариант

Построить траекторию движения двух заряженных частиц (+q и –q) и массами m1 и m2, начинающих движение на расстояние L друг от друга, с начальными скоростями $\vec{V}_1 = (0, Vy_0)$, $\vec{V}_2 = (0, Vy_0)$, в поле силы тяжести.

24. Вариант

Задача Циолковского. Ракета стартует под углом к горизонту, в единицу времени Δt она теряет часть своей начальной массы Δm, которая истекает против движения ракеты со скоростью V относительно ракеты. В силу этого, ракета за каждый интервал Δt приобретает дополнительный импульс, действующий на остающуюся массу. Построить траекторию движения такой ракеты до момента, когда она израсходует 80% своей первоначальной массы.

25. Вариант

По плоскому полю равномерно распределено несколько липких шаров радиуса R. Центральный шар начинает двигаться с начальной скоростью (V_{x0},V_{y0}). Построить траекторию движения этого шара, считая удары с любым другим шаром абсолютно неупругими, а с боковыми стенками – абсолютно упругими.