$$y(t) = v_{\text{верт}} \cdot t = v_0 \cdot \sin(\alpha) \cdot t - g \cdot t^2 \quad (16.2)$$

Выразив время через координату x (на основании формулы

$$t = \frac{x}{v_0 \cdot \cos(\alpha)} \quad (16.3)$$

и подставив выражение для времени в формулу для координаты y, получим уравнение траектории $y(x)$:

$$y(x) = x \cdot \text{tg}(\alpha) - x^2 \cdot \frac{g}{2 \cdot v_0^2 \cdot \cos^2(\alpha)} \quad (16.4)$$

Поскольку сопротивление при движении тела отсутствует, горизонтальная составляющая скорости изменяться не будет, а изменение вертикальной составляющей определяется влиянием ускорения свободного падения.

$$v_{\text{гор}}(t) = v_0 \cdot \cos(\alpha) \quad (16.5)$$

$$v_{\text{верт}}(t) = v_0 \cdot \sin(\alpha) - g \cdot t \quad (16.6)$$

Время t_0, через которое будет достигнута наивысшая точка траектории, найдём из условия $v_{\text{верт}} = 0$.

$$t_0 = \frac{v_0 \cdot \sin(\alpha)}{g} \quad (16.7)$$

Максимальную высоту подъёма H найдём из уравнения вертикального движения (формула 16.2) в момент времени t_0.

$$H = y(t_0) = \frac{v_0^2 \cdot \text{sin}}{2 \cdot} \quad (16.8)$$

Полное время полёта T очевидно, равно $2t_0$, поэтому дальность полёта S определим как

$$S = v_{rop} \cdot T = v_0 \cdot \cos(\alpha) \cdot 2 \cdot t_0 = \frac{v_0^2 \cdot \sin^2(}{g} \quad (16.9)$$

Все эти формулы понадобятся для вычисления координат точек траектории и параметров траектории при моделировании.

Оборудование и материалы.

Персональный компьютер, среда разработки Python.

Указания по технике безопасности:

Соответствуют технике безопасности по работе с компьютерной техникой.

Задания

Текст программы, моделирующей траекторию полёта тела с пользовательскими подрограммами приведён ниже.

```
# -*- coding: utf-8 -*
```

```
# Моделирование задачи о теле
# брошенном под углом к горизонту
#-

 import Tkinter

 import math

#

 def plot_x_axe(x0, y0, x1):

 x_axe=[]
 xx=(x0, y0)
 x_axe.append(xx)
 xx=(x1, y0)
 x_axe.append(xx)
 canvas.create_line(x_axe, fill="black", width=2)
#

 def plot_y_axe(x0, y0, y1):

 y_axe=[]
 yy=(x0, y1)
 y_axe.append(yy)
 yy=(x0, y0)
 y_axe.append(yy)
 canvas.create_line(y_axe, fill="black", width=2)
#

 def DrawGraph():

# Получаем и пересчитываем параметры
 dta=sc.get()
 alpha=dta_math.pi/180
 dtlbl=clist.get()
# Очищаем область для текста
 canvas.create_rectangle(x1i-90, y1i-50,
 x1i +50, y1i +10, fill="#eeeeff")
# Считаем g=10, v0 подбираем, чтобывсёвлезалов canvas
 g=10.0
 v0=63
#
 S=int((v0**2)*math.sin(2*alpha)/g)
 H=int(((v0**2)*(math.sin(alpha))**2)/(2*g))
#
 points=[]

         for x in range(x0i, x1i):

        xx=(x-x0)
```

```
        y=(xx *math.tan(alpha))-((xx--2)-g/ \
              (2*(v0**2)*(math.cos(alpha)**2)))
#
```

if y > 0 :

```
        yy=int(y0-y)
```

else:

```
        yy=y0i
  #
        pp=(x, yy)
        points.append(pp)
# Собственнографик
  canvas.create_line(points, fill=dtlbl, smooth=1)
  plot_x_axe(x0i, y0i, x1i)
# Параметры графика
  dtext="Дальность: "+str(S)
  vtext="Высота: "+str(H)
  dalnost=canvas.create_text(x1i-70, y1i-30, text=dtext,
        fill=dtlbl, anchor="w")
  vysota=canvas.create_text(x1i-70, y1i-10, text=vtext,
        fill=dtlbl, anchor="w")
#
# Основная часть
tk=Tkinter.Tk()
tk.title("Моделирование полёта")
# Верхняя часть окна со списком и кнопками
menuframe=Tkinter.Frame(tk)
menuframe.pack({"side":"top", "fill":"x"})
# Надписьдлясписка
lbl=Tkinter.Label(menuframe)
lbl["text"]="Выборцвета:"
lbl.pack({"side":"left"})
# Инициализация и формирование списка
clist=Tkinter.StringVar(tk)
clist.set('black')
#
cspis=Tkinter.OptionMenu(menuframe, clist,
  'red',
  'green',
  'blue',
  'cyan',
  'magenta',
  'purple',
  'black')
cspis.pack({"side":"left"})
# Кнопкауправлениярисованием
btnOk=Tkinter.Button(menuframe)
btnOk["text"]="Нарисовать"
btnOk["command"]=DrawGraph
```

```
btnOk.pack({"side":"left"})
# Кнопказакрытияприложения
button=Tkinter.Button(menuframe)
button[" text "]="Закрыть"
button["command"]=tk.quit
button.pack({"side":"right"})
#
# Надписьдляшкалыуглов
lbl2=Tkinter.Label(tk)
lbl2["text"]="Угол, градусы:"
lbl2.pack({"side":"top"})
# Шкалауглов
sc=Tkinter.Scale(tk, from_=0, to=90, orient="horizontal")
sc.pack({"side":"top", "fill":"x"})
#
# Областьрисования (холст)
canvas=Tkinter.Canvas(tk)
canvas["height"]=360
canvas["width"]=480
canvas["background"]="#eeeeff"
canvas["borderwidth"]=2
canvas.pack({"side":"bottom"})
#
# Установки осей координат
x0=50.0
y0=300.0
x1=450.0
y1=50.0
#
x0i=int(x0)
x1i=int(x1)
y0i=int(y0)
y1i=int(y1)
# Осикоординат
plot_x_axe(x0i, y0i, x1i)
plot_y_axe(x0i, y0i, y1i)
#
tk.mainloop()
```

Результат работы с моделью показан на рис. 16.1.

Реализация модели имеет ряд особенностей. Во-первых, величина ускорения свободного падения g принята как 10. Во-вторых, модуль начальной скорости выбран так, чтобы при любых значениях угла вся траектория попадала в область графика. Не совсем правильно с точки зрения принципа разделения программ и данных установка значений для g и v_0 прямо в коде, но такое решение значительно упрощает работу с моделью.

"Ползунок" на шкале установки углов показывает значения в градусах, а для правильных вычислений в тригонометрических функциях эти значения нужно перевести в радианы.

Высота и дальность полёта пишутся для каждой траектории соответствующим цветом в прямоугольнике в верхнем правом углу. Для каждой следующей траектории этот прямоугольник рисуется заново и текст переписывается.

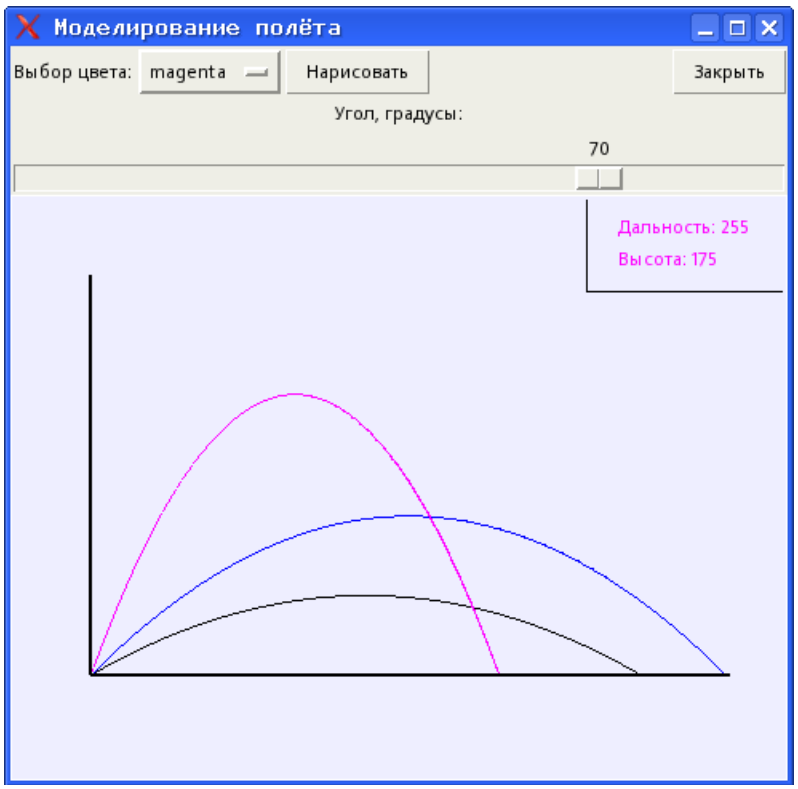

Рис. 16.1. Поиск угла для достижения максимальной дальности на модели

В этой лабораторной работе рассмотрены лишь некоторые базовые возможности модуля Tkinter и использования Python для создания моделей. Если возникнет желание более подробно познакомиться с применением объектом и методов этого модуля, можно изучить оригинальную документацию и другие примеры, используя ресурсы Интернет.

Задачи и упражнения

1. Для примера, показанного на рис. 15.1, нанесите на оси метки и проставьте значения в масштабе графика.
2. Напишите подпрограмму формирования строки со значениями координат для примера, показанного на рис. 14.2.
3. Напишите подпрограммы для нанесения меток и вывода значений по горизонтальной и вертикальной осям для примера моделирования математических функций.
4. В модели тела, брошенного под углом у горизонту, напишите подпрограммы вывода метки "точки падения" и метки максимальной высоты для каждой траектории.
5. Модифицируйте код для моделирования полёта так, чтобы можно было изменять начальную скорость, а график автоматически масштабировался в области рисования.

Контрольные вопросы

1. Применение физического моделирования для выявления закономерностей изучаемого явления.
2. Проверка правильности и границ применимости эксперимента.
3. Модели тела, брошенного под углом у горизонту.
4. Поиск угла для достижения максимальной дальности на модели.
5. Верификация моделей.