

Занятие 13. Сказка о зайце и геометрической оптике

«— Ату его, — слышался в это время протяжный крик одного из остановившихся борзятников. Он стоял на полубугре жнивья, подняв арапник, и ещё раз повторил протяжно: — А-ту — его!»

Лев Толстой «Война и мир»

13.3. Заяц фотон

А на кого кричал борзятник (псарь, смотрящий за борзыми собаками) из эпиграфа? Он кричал и показывал арапником (плеткой для собак) на зайца, который, услышав этот крик, побежал во всю прыть, спасаясь от преследующих его собак.

Герои романа Л.Н. Толстого болели за собак – поймут ли они или не поймут зайца. А давайте мы поболеем за зайца – убежит ли он или не убежит от собак. И поможем ему. В наше время охота, будь она псовая («Война и мир») или ружейная («Анна Каренина»), связанная не с добычей пропитания, не с защитой людей и домашнего скота, а с некой забавой (хобби), превратилась в довольно сомнительное, если не сказать в довольно подлое дело. Мы не только поможем зайцу убежать от собак, но и на его примере поговорим о геометрической оптике.

Мы поможем зайцу, а нам поможет математика и компьютер с установленной на ней программой SMath.

Случай первый – простейший

Зайцу нужно как можно быстрее пробежать из точки 0 до точки 2 через поле (см. второй график на рис. 13.1). Поле разделено на две части – одна часть (верхняя на рис. 13.1) с нескошенной травой, а другая (нижняя) – со скошенной.

Кратчайшее расстояние между точками – это прямая. Но для нашего зайца важно не расстояние, а время. Максимальная скорость зайца на нескошенном и скошенном поле разная. Поэтому заяц должен бежать не по прямой линии, а по линии с изломом, что и отображено в расчёте на рис. 13.1.

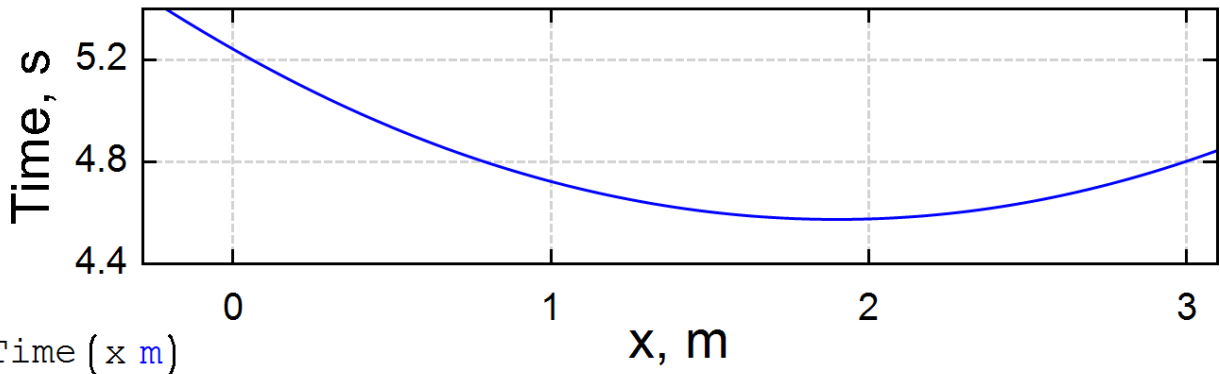
В первых трех строках расчёта вводятся исходные данные: координаты точек 0 и 1, скорости бега зайца по двум участкам поля и ордината границы скошенного и нескошенного участков.

Далее вводится функция пользователя с именем *Time* (время бега зайца от точки 0 до точки 2 через точку 1) с аргументом x (абсцисса точки 1). Строится график функции *Time*, на котором четко виден минимум. Его абсцисса рассчитывается через численный поиск корня уравнения: производная гладкой функции в точке минимума равна нулю.

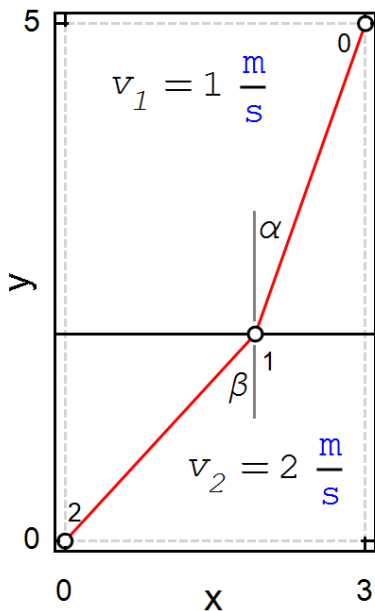
Занятие 13

$$\begin{aligned}
 x_0 &:= 3 \text{ m} & y_0 &:= 5 \text{ m} & v_1 &:= 1 \frac{\text{m}}{\text{s}} \\
 y_1 &:= 2 \text{ m} \\
 x_2 &:= 0 \text{ m} & y_2 &:= 0 \text{ m} & v_2 &:= 2 \frac{\text{m}}{\text{s}}
 \end{aligned}$$

$$\text{Time}(x) := \frac{\sqrt{(x_0 - x)^2 + (y_0 - y_1)^2}}{v_1} + \frac{\sqrt{(x - x_2)^2 + (y_1 - y_2)^2}}{v_2}$$



$$x := \text{solve} \left(\frac{d}{d x} \text{Time}(x \text{ m}) = 0, x, 1 \text{ m}, 3 \text{ m} \right) \text{ m} = 1.8997 \text{ m}$$



\sin Shift+Ctrl+k пробел Shift+Ctrl+k альфа

$$\sin \alpha := \frac{x_0 - x}{\sqrt{(x_0 - x)^2 + (y_0 - y_1)^2}} = 0.3443$$

$$\alpha := \arcsin(\sin \alpha) = 20.14^\circ$$

$$\sin \beta := \frac{x - x_2}{\sqrt{(x - x_2)^2 + (y_1 - y_2)^2}} = 0.6887$$

$$\beta := \arcsin(\sin \beta) = 43.53^\circ$$

$$\left\{ \begin{array}{l} y_1 \\ \begin{bmatrix} x_0 & y_0 \\ x & y_1 \\ x_2 & y_2 \end{bmatrix} \end{array} \right.$$

$$\frac{\sin \alpha}{\sin \beta} = 0.5 \quad \frac{v_1}{v_2} = 0.5$$

Рис. 13.1. Заяц бежит через поле из точки 0 в точку 2

Во второй части расчёта (после нахождения абсциссы точки минимума) строится уже упомянутый нами график бега зайца с изломом в точке 1 и рассчитываются значения синусов двух углов – угла α «падения» зайца на границу раздела поля на два участка и угла β – угла «преломления» зайца на этой границе. Да-да! Наш заяц – это фактически луч света (фотон света!), который, согласно оптическому принципу Ферма, перемещается из одной точки в другую за минимальное время (таутохронизм). Принцип Ферма (принцип наименьшего времени Ферма) – это постулат в геометрической оптике, согласно которому свет выбирает из множества путей между двумя точками тот путь, который потребует наименьшего времени.

Подсчет отношения синусов углов падения и преломления, равный отношению скоростей зайца, пардон, света в разных средах (операторы в рамочке на рис. 13.1) позволяет нам вспомнить о законе геометрической оптики – о законе преломления Снелла. Если углы падения и преломления луча света достаточно малы (меньше семи градусов), то от синусов можно отказаться и работать с самими углами. Это существенно упрощает расчёты в области геометрической оптики. Но переход от ручных к компьютерным расчётам делает в ряде случаев такое упрощение излишним [1].

Примечание.

Функция *Time* называется *целевой функцией*. Цель нашего расчёта – это нахождение минимума этой функции. Но есть расчёты, где ищется не минимум, а максимум. В общем случае говорят об оптимизации.

Случай второй – более сложный

Зайцу нужно как можно быстрее спрятаться в лесу от собак. Заяц начинает свой бег со скоростью v из точки 0 (см. схему на рис. 13.2) по нескошенному полю, в середине которого есть круглый скошенный участок¹ радиусом r , где заяц может бежать быстрее – со скоростью v_r . За полем есть спасительный лес (линия с ординатой точки 3), расстояние до которого по прямой от точки старта зайца равно Δ . Давайте подскажем зайцу, что в некоторых ситуациях бежать нужно не прямо к лесу, а наискосок к круглому участку (точка 1), находящемуся посреди поля. Заяц в точке 1 должен изменить направление бега, пересечь круг по хорде, выбежать из него с ещё одной сменой направления (точка 2) и по кратчайшему пути финишировать в лесу (точка 3). Давайте рассчитаем траекторию бега зайца – найдем координаты точек 1 и 2, при которых время бега будет минимальным! Точка начала координат в нашей новой задаче находится в центре круга.

На рисунке 13.2 показано начало решения этой задачи.

На первой строке расчёта принимается, что метр и секунда – это не единицы расстояния и времени, а просто единицы. Это сделано для того, чтобы инструменты поиска минимума работали без ошибки, которую разработчик SMath планируют в будущем исправить. Переменные m и s будут у нас играть роль простых комментариев. Но это довольно

¹ В интернете можно найти сделанные с высоты фотографии полей со скошенными причудливым образом участками (особый вид изобразительного искусства).

Занятие 13

рискованный прием. Если в формулах будет допущена ошибка – проставлена, например, неверная степень, то расчёт не прервется сообщением об ошибке, связанной с неправильным сочетанием единиц измерения, а выдаст неверный ответ. Кроме того, в ответах не будут автоматически проставляться метры и секунды – это нужно будет делать вручную. Смиримся с такими временными неудобствами ради спасения бедного зайца. В первой задаче о беге зайца (рис. 13.1) единицы расстояния (метр) и времени (секунды) не были отключены. Поэтому единицу длины метр приходилось вставлять и в аргумент графика, и в аргумент функции `solve`. Мы этот прием уже использовали в задаче о трамвайных рельсах – см. рис. 1.6-1.8 в главе 1.

На второй строке расчёта вводятся исходные данные: ширина поля (Δ), по которому бежит заяц, радиус круглой площадки в середине поля (r) и две скорости зайца – по нескошенному (v) и скошенному полю (v_r). На третьей строке рассчитываются ординаты нулевой (y_0) и третьей (y_3) точек, а также время бега зайца к лесу по прямой поперек поля (T). Далее задается абсцисса нулевой точки x_0 (точка старта зайца) и в расчёт вводится целевая функция с именем `Time` с двумя аргументами – с абсциссами первой и второй точек. Ординаты этих точек рассчитываются из уравнения окружности $x^2 + y^2 = r^2$, которое также используется при рисовании этой окружности на графике рисунка 13.2. Затем рассчитывается искомое время бега зайца по ломаной.

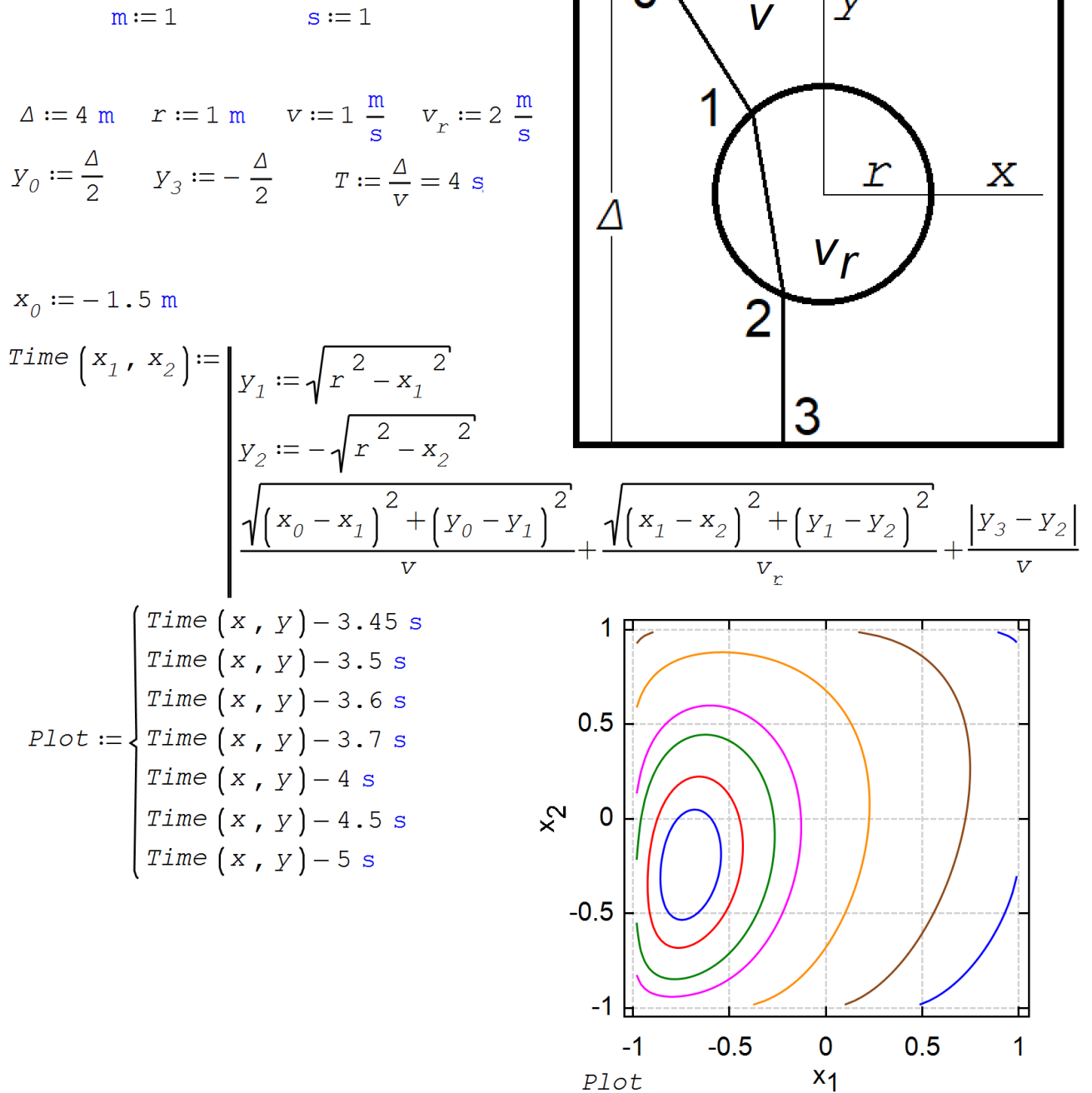


Рис. 13.2. Начало решения задачи о беге зайца по полю с круглым скошенным участком посередине

При решении первой задачи для того, чтобы убедиться в существовании точки минимума, мы строили «провисающую» кривую – см. рис. 13.1. В новой задаче просто одной кривой не обойтись – было построено *семейство кривых*. Внизу рисунка 13.2 можно видеть график – линии одного уровня нашей целевой функции. Из него видно, что эта функция имеет точку экстремума. Исходя из сути задачи несложно понять, что это точка минимума, а не максимума. Вблизи этого места графика выбрана точка, от которой ведется поиск решения – значения переменных x_1 и x_2 .

Занятие 13

У гладкой функции двух аргументов в точке минимума частные производные равны нулю. Это свойство мы и использовали для решения задачи при значениях x_0 , равных -2.44 м (рис. 13.3 и -2.45 м (рис. 13.4).

$$x_0 := -2.44 \text{ m} \quad x_1 := -0.8 \text{ m} \quad x_2 := -0.3 \text{ m}$$

$$\text{Time}(x_1, x_2) = 4.0186 \text{ s}$$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} := \text{roots} \left(\begin{bmatrix} \frac{d}{d x_1} \text{Time}(x_1, x_2) \\ \frac{d}{d x_2} \text{Time}(x_1, x_2) \end{bmatrix}, \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right) = \begin{bmatrix} -0.8992 \\ -0.3329 \end{bmatrix} \text{ m}$$

$$t_{\min} := (\text{Time}(x_1, x_2)) = 3.9975 \text{ s}$$

$$y_1 := \sqrt{r^2 - x_1^2} = 0.4376 \text{ m}$$

$$y_2 := -\sqrt{r^2 - x_2^2} = -0.943 \text{ m}$$

$$x_0 = -2.44 \text{ m} \quad t_{\min} < T = 1$$

$$\text{Plot} := \begin{cases} \begin{bmatrix} x^2 + y^2 - r^2 \\ \text{if } t_{\min} < T \\ \begin{bmatrix} x_0 & x_1 & x_2 & x_2 \\ y_0 & y_1 & y_2 & -\frac{\Delta}{2} \end{bmatrix}^T \\ \text{else} \\ \begin{bmatrix} x_0 & \frac{\Delta}{2} \\ x_0 & -\frac{\Delta}{2} \end{bmatrix} \end{cases}$$

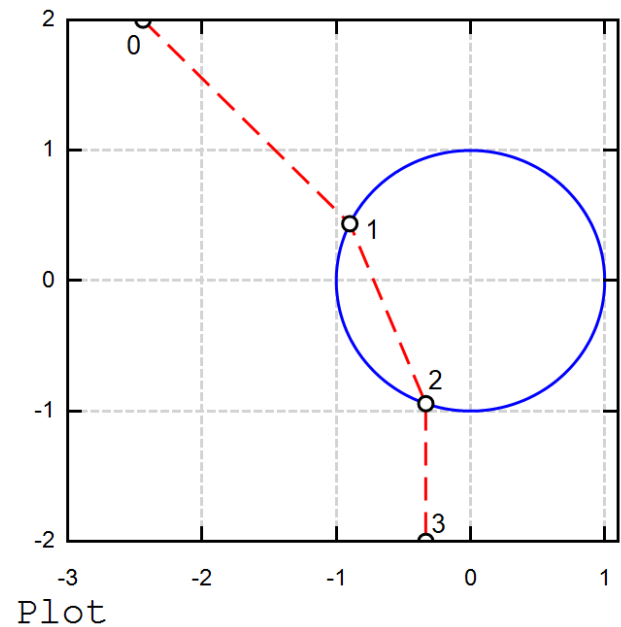


Рис. 13.3. Заяц бежит по круглому участку поля со скошенной травой

$$x_0 := -2.45 \text{ m} \quad x_1 := -0.8 \text{ m} \quad x_2 := -0.3 \text{ m}$$

$$\text{Time}(x_1, x_2) = 4.0262 \text{ s}$$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} := \text{roots} \left(\begin{bmatrix} \frac{d}{d x_1} \text{Time}(x_1, x_2) \\ \frac{d}{d x_2} \text{Time}(x_1, x_2) \end{bmatrix}, \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right) = \begin{bmatrix} -0.9004 \\ -0.3336 \end{bmatrix} \text{ m}$$

$$t_{min} := (\text{Time}(x_1, x_2)) = 4.0045 \text{ s}$$

$$y_1 := \sqrt{r^2 - x_1^2} = 0.4351 \text{ m}$$

$$y_2 := -\sqrt{r^2 - x_2^2} = -0.9427 \text{ m}$$

$$x_0 = -2.45 \text{ m} \quad t_{min} < T = 0$$

$$\text{Plot} := \begin{cases} \begin{bmatrix} x^2 + y^2 - r^2 \\ \text{if } t_{min} < T \\ \begin{bmatrix} x_0 & x_1 & x_2 & x_2 \\ y_0 & y_1 & y_2 & -\frac{\Delta}{2} \end{bmatrix}^T \\ \text{else} \\ \begin{bmatrix} x_0 & \frac{\Delta}{2} \\ x_0 & -\frac{\Delta}{2} \end{bmatrix} \end{cases}$$

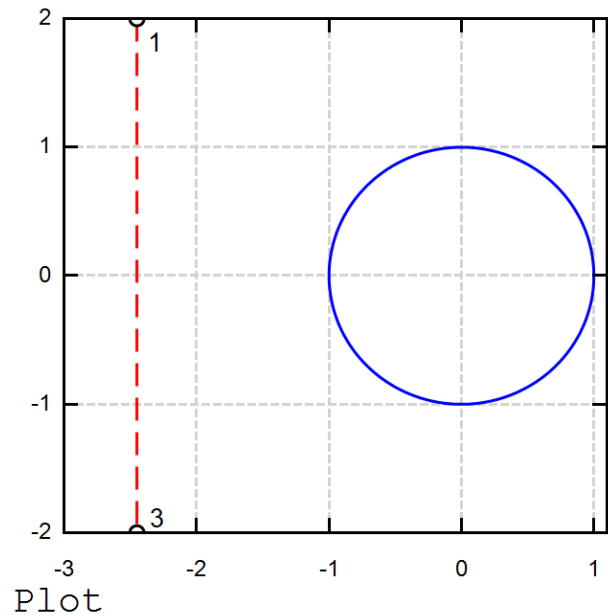


Рис. 13.4. Заяц бежит по прямой к лесу

Из рисунка 13.3 видно, что при $x_0 = -2.44 \text{ m}$ (стартовое значение) и $x_1 = -0.8 \text{ m}$ и $x_2 = -0.3 \text{ m}$ (первое приближение к решению задачи) время бега зайца к лесу по ломаной составит примерно 4.02 секунды, что больше времени бега по прямой (4 секунды). При оптимальных же найденных значениях $x_1 = -0.8992 \text{ m}$ и $x_2 = -0.3329 \text{ m}$ время бега зайца к лесу составит менее 4 секунд. Это время будет уменьшаться до 3 секунд по мере приближения значения стартовой переменной x_0 к нулю. Если же мы будем уменьшать стартовое значение переменной x_0 , делать его равным -2.45 m , то круглая скошенная площадка нашему зайцу не поможет – ему нужно будет бежать в лес от собак по кратчайшему прямому пути без изломов. Это показано на графике рисунка 13.4.

Несложно подсчитать значения углов падения и преломления луча света, пронизывающего окружность на рис. 13.3, и убедиться в том, что синусы этих углов относятся друг к другу, как значения скоростей света снаружи и внутри окружности (закон Снелла – см. выше).

Описанные две задачи имеют реальный физический аналог. Первая задача (рис. 13.1): луч света пронизывает границу двух сред – стекла и воздуха, например. Вторую задачу (рис. 13.3) можно описать такой ситуацией. Мы отлили стеклянную пластину для будущей линзы (см. ниже третью задачу), внутри которой оказался дефект – пузырек воздуха. Как он будет преломлять свет, идущий сквозь стекло? Скорость света в стекле, как известно, ниже скорости света в воздухе.

Задачу о беге зайца по полю с круглой скошенной площадкой можно усложнить – поместить, например, в центре этой площадки новую бетонированную круглую площадку для вертолета сельхозавиации. В этом случае целевая функция *Time* будет иметь не два (см. рис. 13.2), а четыре аргумента. Поиск оптимальных координат точек излома нужно будет вести через решение системы четырех, а не двух уравнений. С такой новой задачей встроенная в *SMath* функция `roots` не справится.

Точку минимума функции с разным количеством аргументов можно найти и иным путем.

На рисунке 13.5 показана программа (функция с именем *MinTwoStep*), реализующая алгоритм поиска точки минимума авторским методом «Два шага». А сейчас, как мы уже отметили в главе 2, наблюдается некий ренессанс простейших и наглядных методов численного решения задач на компьютере, позволяющих не только решить задачу, но и понять, как это делается.

Из начальной заданной точки делаются шаги «вперед и назад» заданной длины D (цикл `for` с параметром Δx) во всех направлениях (цикл `for` с параметром i). Запоминается точка с минимальным значением целевой функции – меньшем, чем в исходной точке. В эту точку делается переход для поиска новой подобной точки. Если такая точка не обнаружена (везде наблюдается рост значения целевой функции), то шаг поиска уменьшается вдвое, а поиск продолжается до тех пор, пока (цикл `while`) шаг станет меньше заданного значения D_{min} . Как говорил один киноперсонаж «Это не эстетично, зато дешево, надежно и практично!». Шутки шутками, но в настоящее время, повторяем, наблюдается возрождение простых и понятных численных методов решения задач. Многие сложные алгоритмы, в которых в настоящее время не разберутся даже сами разработчики, создавались во времена тихоходных электронных вычислительных машин. Сейчас скорость ЭВМ значительно возросла, что позволяет без особых проблем работать на них по простейшим понятным алгоритмам.

Наша самодельная функция «Два шага» помимо прочего позволяет видеть трассировку поиска точки минимума, выдавая матрицу промежуточных значений – см. рис. 13.5. Последняя строка этой матрицы – это решение задачи, которое практически совпало с решением, показанным на рис. 13.3. Всего же сделано пятьдесят итераций.

Занятие 13

$$\Delta := 4 \text{ m} \quad r := 1 \text{ m} \quad v := 1 \frac{\text{m}}{\text{s}} \quad v_r := 2 \frac{\text{m}}{\text{s}} \quad x_0 := -2.44 \text{ m} \quad y_0 := \frac{\Delta}{2} \quad y_3 := -\frac{\Delta}{2}$$

$$Time(x_1, x_2) := \left[\begin{array}{l} y_1 := \sqrt{r^2 - x_1^2} \quad y_2 := -\sqrt{r^2 - x_2^2} \\ \frac{\sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}}{v} + \frac{\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}}{v_r} + \frac{|y_3 - y_2|}{v} \end{array} \right]$$

```
MinTwoStep(F, x, D, D_min) := M := x^T
[ L := length(x)  f_min := F(x) ]
while D > D_min
  p := 1
  while p ≠ 0
    p := 0
    for i ∈ [1..L]
      for Δx ∈ [-D D]
        [ x_i := x_i + Δx  f := F(x) ]
        if f < f_min
          [ p := Δx  j := i  f_min := f ]
          x_i := x_i - Δx
        x_j := x_j + p
      M := stack(M, x^T)
    D := D/2
  M
```

$$F(x) := Time(x_1, x_2) \quad M := MinTwoStep\left(F, \begin{bmatrix} 0.9 \text{ m} \\ 0.9 \text{ m} \end{bmatrix}, 0.1 \text{ m}, 0.0001 \text{ m}\right) \quad n := \text{rows}(M) = 50$$

$$x_1 := M_{n1} = -0.8992 \text{ m} \quad x_2 := M_{n2} = -0.3330 \text{ m} \quad Time(x_1, x_2) = 3.9974981 \text{ s}$$

```
Plot := {
  Time(x m, y m) - 4 s
  Time(x m, y m) - 4.1 s
  Time(x m, y m) - 4.2 s
  Time(x m, y m) - 4.3 s
  Time(x m, y m) - 4.4 s
  Time(x m, y m) - 5 s
  Time(x m, y m) - 5.5 s
}
M
```

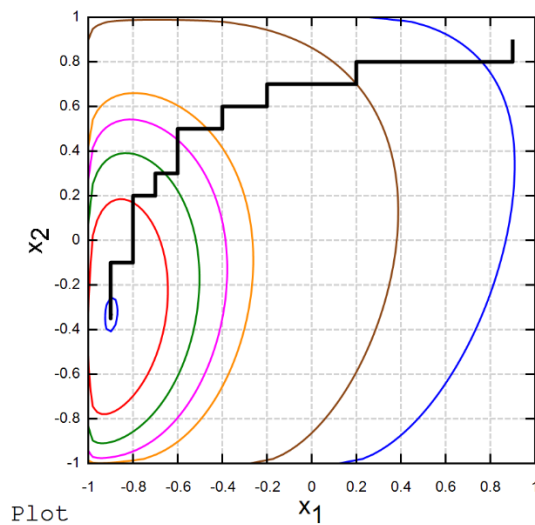


Рис. 13.5. Поиск минимума методом «Два шага»

13.2. Проектируем линзу

На рисунке 13.6 показана оптическая схема простейшего микроскопа, через линзы которого лучи света проходят также за минимальное время. Вот мы и добрались до *линз* – главного объекта оптики.

А давайте рассчитаем форму линзы. Многие считают, что её поверхность должна быть сферической, но это не совсем так.

Увеличение микроскопа

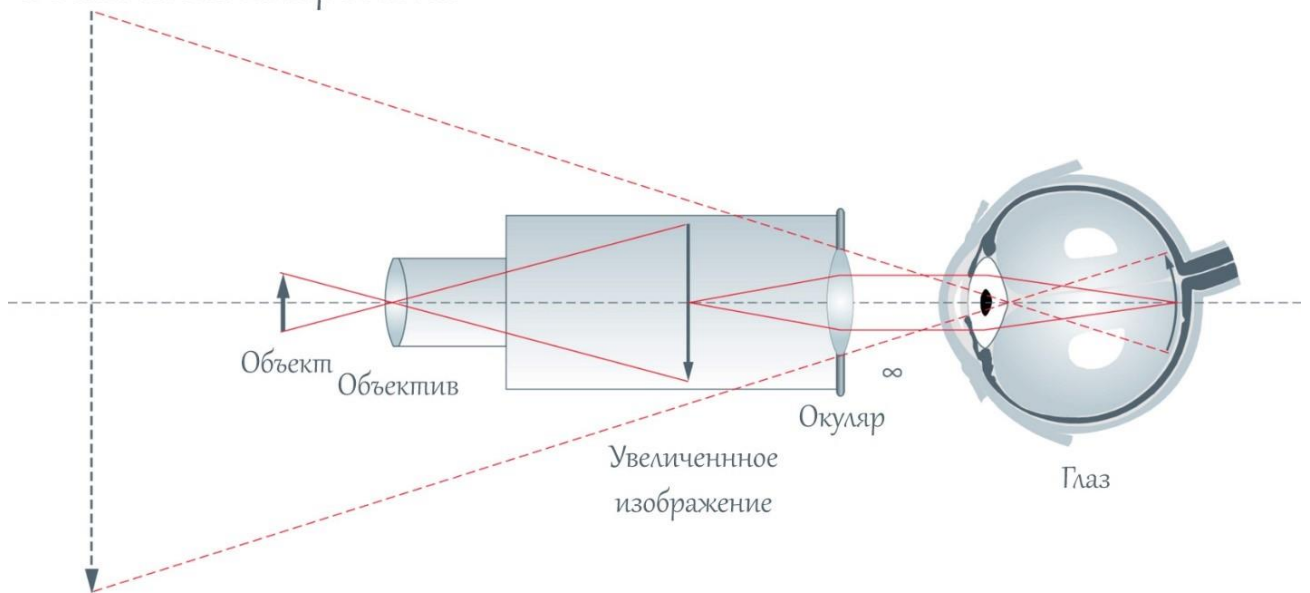


Рис. 13.6. Оптическая схема микроскопа

Читатель, с каким запахом у тебя связано наступление весны?! Для многих это запах цветущей черемухи или терпкий запах распускающихся почек на березе или тополе. Для автора же этой книги запах весны – это запах... жженого дерева. Почему?

Самую раннюю весеннюю пору называют Весной Света. Февраль, снег ещё и не думает таять, никаких цветов и листьев пока нет, но наконец-то после долгой зимы ярко засветило солнце. Во времена детства автора многие пацаны в это время выбегали на улицу с лупами и выжигали картинки или слова на всем, что сделано из дерева² – см. рис. 13.7. А лупы (увеличительные стекла) в те времена были практически в каждой семье. Приходит старость, глаза слабеют, нужны не только очки, но и лупа, если хочется почитать книгу, газету или

² В рассказе чешского писателя Карела Чапека (1890–1938) «Истории о взломщике и поджигателе» описывается загадочный пожар. В ясный солнечный день сидит человек дома со своим семейством и празднует именины. Но тут стучат ему в окно и кричат, то у него горит крыша... На следующий день сыщик, выполняя функции пожарного дознавателя, замечает на обгоревшей балке чердака сгоревшего дома какой-то блестящий предмет. Это оказалась лупа, которую, как выяснилось в результате расследования, задолго до пожара установил на чердаке бывший ученик хозяина дома. Он был зол на патрона за то, что тот не позволял ему делать опыты с лупой и лупил его за это (невольный каламбур!). Решив отомстить хозяину, мальчик рассчитал, где будет стоять солнце в полдень в день именин хозяина, и укрепил на чердаке лупу под таким углом, чтобы загорелась солома, а он к тому времени уберется подальше...

Занятие 13

журнал... Это сейчас тексты можно читать не только без лупы, но даже и без очков. Достаточно пальцами или мышкой увеличить текст на компьютере, электронной книге или смартфоне...



Рис. 13.7. Выжигание на дереве с помощью лупы

А какую форму должна иметь лупа, чтобы параллельный пучок света, пройдя через лупу (линзу), сошелся в фокусе.

На рисунке 13.8 показана схема задачи о плоско-выпуклой линзе, выполненной из материала с показателем преломления n . Спрашивается, какой формы должна быть нижняя поверхность линзы (верхняя её поверхность плоская), чтобы параллельный пучок света сошелся в фокусе, отстоящем от начала координат (от нижнего края линзы) на расстоянии F (фокусное

расстояние).

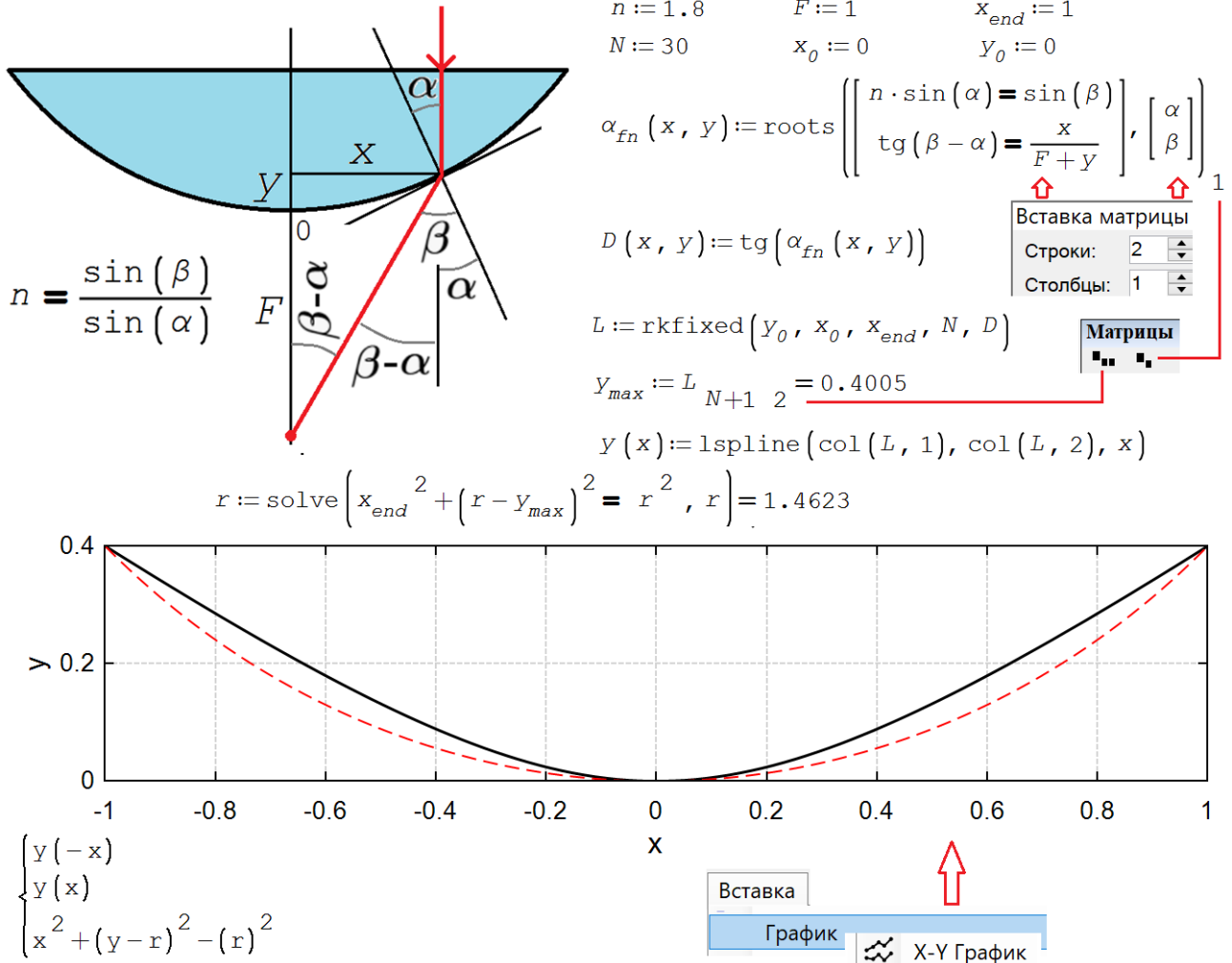


Рис. 13.8. Численное решение задачи о линзе

Мы намеренно повернули луч света вниз, чтобы упростить задачу. И линзу для этого же перевернули выпуклой стороной вниз, а не вверх (к солнцу!). Но обычно все оптические расчёты иллюстрируются схемами с горизонтальной оптической осью (рис. 13.6).

Решение задачи о линзе показано на рисунке 13.8. Она сводится к решению двух алгебраических уравнений и одного дифференциального уравнения (производная равна тангенсу угла наклона касательной). Первое алгебраическое уравнение – это математическая запись закона Снелла с показателем преломления n . Второе уравнение выражает тангенс угла через отношение длины противолежащего катета x к длине прилежащего катета $F + y$ — см. треугольник на схеме задачи с одним из острых углов, равным $\beta - \alpha$. При решении системы двух алгебраических уравнений используется встроенная в SMath функция `roots`, которая генерирует функцию пользователя с именем α_{fn} и с двумя аргументами x и y . Функция `roots` возвращает вектор с двумя элементами, из которого изымается первый элемент. Далее формируется дифференциальное уравнение, правая часть которого – это тангенс значения функции пользователя α_{fn} . Левая же часть – это производная искомой функции. При решении используется встроенная в SMath функция `rkfixed`, возвращающая матрицу с

Занятие 13

двумя столбцами и $N + 1$ строками. Первый столбец матрицы – это дискретные значения x , а второй – y . Первая строка этой матрицы была задана – это x_0 и y_0 . По столбцам матрицы сплайн-интерполяцией (см. главу 6) генерируется функция $y(x)$, по которой строится искомый график – профиль нижней части линзы. Он дополнен пунктирной дугой окружности, проходящей через три точки: начало координат и два края линзы. Радиус этой окружности r рассчитывается с помощью встроенной функции `solve`.

Вывод: линза, которая должна сфокусировать параллельный пучок света в точке, не имеет сферической формы. Но линзы сферической формы проще изготавливать, в частности, полировать на полировальном круге сферической формы. При этом приходится мириться с так называемой сферической aberrацией – с размазыванием фокуса (https://ru.wikipedia.org/wiki/Сферическая_абберация).

Решение задачи о форме линзы через использование функции `Roots` было подсказано автору на форуме пользователей SMath https://en.smath.com/forum/yaf_postst25209_How-to-solve-this-ODE-in-SMath.aspx

В главе 5 было показано, как с помощью замены синуса малого угла на само значение угла было аналитически решено дифференциальное уравнение колебания маятника – см. рис. 5.2 и 5.3. Такую операцию можно провести и для трех уравнений линзы – двух алгебраических и одного дифференциального (см. рис. 13.9), приняв во внимание, что при малых углах можно приравнять к значению самого угла не только синус, но и тангенс.

В расчёте на рис. 13.9 функция `dsolve`, входящая в приложение `maple` пакета SMath (см. рис. В.4 во Введении), выдала аналитическое решение. Это уравнение можно, конечно, решить и без компьютера, введя вспомогательное уравнение $z(x) = y(x) + F$, разделив переменные и получив несложным интегрированием более простое решение, но мы удовлетворимся тем, что выдала функция `dsolve`.

Занятие 13

$$\left[\begin{array}{l} n \cdot \sin(\alpha) = \sin(\beta) \\ \operatorname{tg}(\beta - \alpha) = \frac{x}{F + y} \end{array} \right] \rightarrow \left[\begin{array}{l} n \cdot \alpha = \beta \\ \beta - \alpha = \frac{x}{F + y} \end{array} \right] \rightarrow \operatorname{maple} \left(\operatorname{solve} \left(n \cdot \alpha - \alpha = \frac{x}{F + y}, \alpha \right) \right) = \frac{x}{(F + y) \cdot (-1 + n)}$$

$$\operatorname{maple} \left(\operatorname{dsolve} \left(\left\{ \begin{array}{l} \frac{d}{dx} y(x) = \frac{x}{(F + y(x)) \cdot (n - 1)} \\ y(0) = 0 \end{array} \right\} \right) \right) = \left[\begin{array}{l} y(x) = \frac{F \cdot (1 - n) + \sqrt{F^2 \cdot (1 + n \cdot (-2 + n)) + x^2 \cdot (-1 + n)}}{-1 + n} \\ y(x) = \frac{F \cdot (1 - n) - \sqrt{F^2 \cdot (1 + n \cdot (-2 + n)) + x^2 \cdot (-1 + n)}}{-1 + n} \end{array} \right]$$

$$F := 1 \text{ m} \quad n := 1.8 \quad x_{\text{end}} := 1 \text{ m} \quad y(x) := \frac{F \cdot (1 - n) + \sqrt{F^2 \cdot (1 + n \cdot (n - 2)) + x^2 \cdot (n - 1)}}{n - 1} \quad y_{\text{max}} := y(x_{\text{end}}) = 0.5 \text{ m}$$

$$r := \operatorname{solve} \left(x_{\text{end}}^2 + (r \text{ m} - y_{\text{max}})^2 = (r \text{ m})^2, r \right) \text{ m} = 1.25 \text{ m}$$

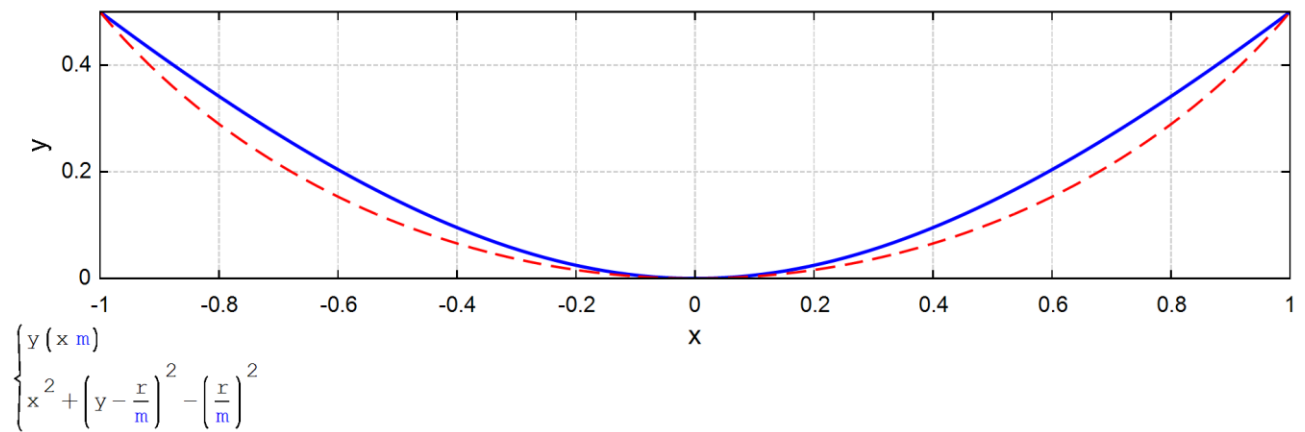


Рис. 13.9. Аналитическое решение линеаризованной задачи о линзе

Если увеличивать значение фокусного расстояния F (проводить своеобразный вычислительно-физический эксперимент – см. рис. 13.10), то график численного решения задачи и график линеаризованного решения будут постепенно сливаться в один. И это понятно – углы α и β будут уменьшаться, а значения синуса и тангенса будут приближаться к значению самих углов. Линза будет становиться все тоньше и тоньше, а её поверхность будет стремиться к сферической. Но с тонкой линзой особо не повыжигаете на весеннем солнышке... Такие линзы годятся разве что для очков³.

³ Прапрадед автора этой книги первым надел очки в той деревне, где он жил – так зародилась наша фамилия. А до этого он был Степановым – как и половина жителей этой местности.

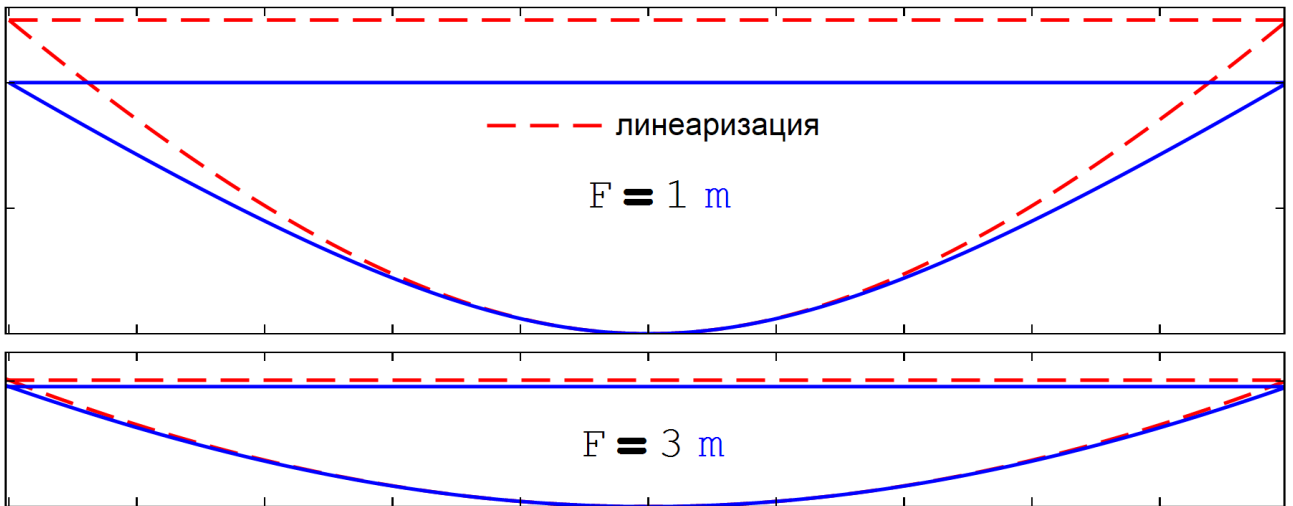


Рис. 13.10. Профили двух линз с разным фокусным расстоянием

Но линзу можно сделать тоньше и другим путем: функцию $y(x)$ достаточно сделать ступенчатой – см. рис. 13.11. Получится так называемая линза Френеля.

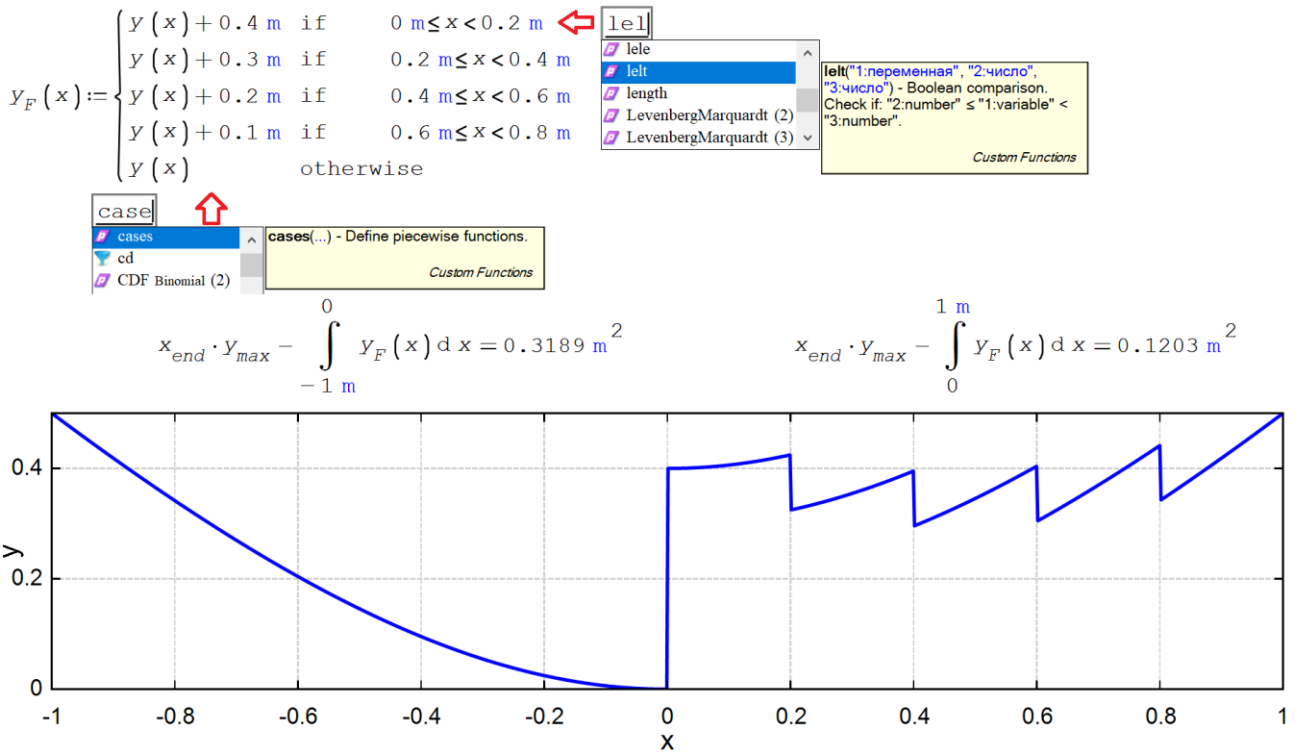


Рис. 13.11. Линза Френеля

По определённым интегралам, показанным, на рис. 13.11, можно оценить, насколько линза Френеля легче обычной линзы. По интегралам рассчитаны площади над двумя частями графика – кривой и ломаной (рис. 13.11). Там же показано, как в расчёт вводятся функции (операторы) cases (случаи) и lelt (меньше или равно и меньше), которых нет на панелях инструментов среды SMath.

Задача о форме линзы оказалась занимательной не только с позиций оптики, но и с позиций информатики.

Судите сами.

1. В решении, показанном на рис. 13.8, число шагов разбиения участка численного решения дифференциального уравнения N было принято за 30. Если это число увеличить до 32 и больше, то в пакете *SMath* решение выдано не будет, нельзя будет дождаться результата вычислений. Придется нажимать клавишу *Esc* и прерывать расчёт.
2. Замена встроенной функции *rkfixed* на пользовательскую функцию *Euler* (она описана в главе 10 – см. рис. 10.3) позволяет без зависания программы увеличить значение N до 50, но не более.
3. Попытка решить с использованием функции *rkfixed* линеаризованный вариант задачи, показанной на рис. 13.8, приводит к тому, что описано в п.1. Вернее, дело происходит так. Если убрать два синуса и тангенс из системы двух алгебраических уравнений, то функция *rkfixed* будет выдавать решение. А если убрать тангенс и из дифференциального уравнения, то... см. пункт 1.
4. Аналитическое решение линеаризованной задачи дает два ответа – см. рис. 13.9. Второй ответ со знаком минус перед квадратным корнем соответствует не плоско-выпуклой, а плоско-вогнутой линзе. Как это коррелируется с физикой (оптикой) задачи?

13.2. Задача о погоне

Мы начали эту главу с задачи о беге зайца через поле – спроектировали траекторию его бега – см. рис. 13.1, 13.4 и 13.4. А давайте рассчитаем траекторию бега того, кто гонится за зайцем – собаки.

Рассмотрим такую задачу. На круглой площадке с помощью вращающейся консоли с постоянной скоростью волочат по земле по кругу тряпичного зайца. В какой-то момент из любого места площадки выпускают настоящую собаку и натравливают её на этого зайца – см. эпиграф. Как побежит собака, если вектор ее движения направлен строго на зайца, а скорость постоянна и может быть выше или ниже скорости зайца.

На рисунке 13.12 показан расчёт в среде *SMath*, по которому генерируются точки траекторий движения зайца (*hare*) и бегущей за ним собаки (*dog*). Затем эти точки (траектории) отображаются на графике (рис. 13.13–13.14) и в анимации (рис. 13.15–13.17).

На первой строке расчёта (рис. 13.12) задается радиус окружности, по которой бежит заяц – 700 метров. Единица длины метр (синяя, а не черная буква *m*) вводится через кнопку ввода единиц измерения (кнопка с изображением воронки на верхней панели программы *SMath*). Далее задаются скорости зайца и собаки (переменные с текстовыми индексами, которые вводятся через точку) с единицами скорости. Числа, естественно, условные. Единицы измерения помогут избежать ошибок типа сложения метров с килограммами. Переменная Δt – это длительность скачка собаки и зайца во время бега. А делают они n скачков. Для повышения точности расчета можно уменьшать значение Δt и увеличивать значение n .

Занятие 13

На второй строке расчета определяется значение переменной $\Delta\alpha$ – угла разворота зайца относительно центра круга при каждом скачке (отношения пройденного пути $v_{hare} \Delta t$ к длине окружности R). Затем переменной i (номер очередного скачка зайца и собаки) присваиваются значения 1, 2, 3 и т.д. до n . Это делается с помощью кнопки ввода диапазона значений с единичным шагом в панели инструментов **Матрицы**. Фрагмент этой и других панелей инструментов показан на рис. 13.12.

На третьей строке расчета заполняются векторы x_{hare} и y_{hare} , хранящие координаты зайца при каждом его скачке, и первые элементы векторов x_{dog} и y_{dog} – начальные координаты собаки (собака, напоминаем, стартует из центра круга, который является центром координат). Но можно менять стартовые координаты собаки и строить новые траектории её бега за зайцем.

Занятие 13

$$R := 700 \text{ m} \quad v_{hare} := 5 \frac{\text{m}}{\text{s}} \quad v_{dog} := 5.35 \frac{\text{m}}{\text{s}} \quad \Delta t := 1 \text{ s} \quad n := 50 \quad (1)$$

$$\Delta\alpha := \frac{v_{hare} \cdot \Delta t}{R} \quad i := [1..n] \quad (2)$$

v.hare[1]

$$x_{hare\ i} := R \cdot \sin(\Delta\alpha \cdot i) \quad Y_{hare\ i} := R \cdot \cos(\Delta\alpha \cdot i) \quad x_{dog\ 1} := 0 \quad Y_{dog\ 1} := 0 \quad (3)$$

$$\varphi_1 := \left[\arctg \left(\frac{Y_{hare\ 1} - Y_{dog\ 1}}{m}, \frac{x_{hare\ 1} - x_{dog\ 1}}{m} \right) \right] \quad (4) \quad \arctg(-1) = -45^\circ \quad \arctg(1, -1) = 135^\circ$$

for i ∈ [1..n]

$$\begin{bmatrix} x_{dog\ i+1} \\ Y_{dog\ i+1} \\ \varphi_{i+1} \end{bmatrix} := \begin{bmatrix} x_{dog\ i} + v_{dog} \cdot \Delta t \cdot \cos(\varphi_i) \\ Y_{dog\ i} + v_{dog} \cdot \Delta t \cdot \sin(\varphi_i) \\ \arctg \left(\frac{Y_{hare\ i} - Y_{dog\ i}}{m}, \frac{x_{hare\ i} - x_{dog\ i}}{m} \right) \end{bmatrix} \quad (5)$$

$$XY_{hare} := \text{augment}(x_{hare}, Y_{hare}, ".", 5, "gray") \quad (6)$$

$$XY_{hareN} := \text{augment}(x_{hare\ n}, Y_{hare\ n}, ".", 15, "gray") \quad (7)$$

$$XY_{dog} := \text{augment}(x_{dog}, Y_{dog}, ".", 5, "red") \quad (8)$$

$$XY_{dogN} := \text{augment}(x_{dog\ n}, Y_{dog\ n}, ".", 15, "red") \quad (9)$$

$$S := \sqrt{(x_{hare\ n} - x_{dog\ n})^2 + (Y_{hare\ n} - Y_{dog\ n})^2} = 445.0844 \text{ m} \quad (10)$$

$$P := \begin{bmatrix} x_{hare\ n} & Y_{hare\ n} \\ x_{dog\ n} & Y_{dog\ n} \end{bmatrix} \quad \text{plot} := \begin{cases} XY_{hare} \\ XY_{hareN} \\ XY_{dog} \\ XY_{dogN} \\ P \end{cases} \quad (11)$$

Вставка третьего, четвертого и т.д. элемента через нажатие клавиши "запятая"!

Рис. 13.12. Расчет траектории погони собаки за зайцем.

Угол φ – это угол направления бега собаки к зайцу относительно горизонтальной линии (см. рис. 13.13 справа сверху). Такой угол имеет название азимута – угла, отсчитываемого между заранее выбранным направлением (у нас это восточное направление – см. рис. 13.13) и направлением на заданный предмет. Начальное значение этого азимута определяется оператором на четвертой строке расчета. Используется функция арктангенса, но не с одним аргументом, а с двумя аргументами. Отличия в этих вариантах функции арктангенса проявляются при отрицательных значениях аргументов – см. два примера вызова этих функций на рис. 13.12 (пункт 4). Функция арктангенса с двумя аргументами не работает с размерными величинами (недоработка программы SMath). Поэтому эти аргументы через деление на метр лишаются размерности.

Занятие 13

Пятый оператор расчета формирует три вектора x_{dog} , y_{dog} и φ . Эти векторы нельзя заполнить отдельными операторами, как это сделано для векторов x_{hare} и y_{hare} , (см. строку 3 на рис. 13.12). Будет возникать сообщение об ошибке. Векторы x_{dog} , y_{dog} и φ нужно заполнять одновременно, меняя при этом значение переменной i . Поэтому операторы для их заполнения вставлены в тело цикла с параметром `for`. Зная положение собаки перед очередным скачком в точке i и азимут в этой точке (угол φ), можно рассчитать положение собаки после очередного скачка движения в точке $i + 1$. И все!

Операторы 6-11 предназначены для построения графиков по рассчитанным данным – траекторий бега зайца и собаки (операторы 6 и 8), изображения кружочками размером 15 единиц самого зайца и собаки (операторы 7 и 9). Заяц рисуется серым цветом, а собака красным.

Переменная S хранит расстояние от собаки до зайца, которое будет отображаться на графике вместе с другими переменными. Соответствующие операторы нужно создать отдельно, а потом наложить их на график. Матрица P хранит текущие координаты зайца и собаки, что позволяет построить прямую линию, соединяющую собаку с зайцем (линия азимута).

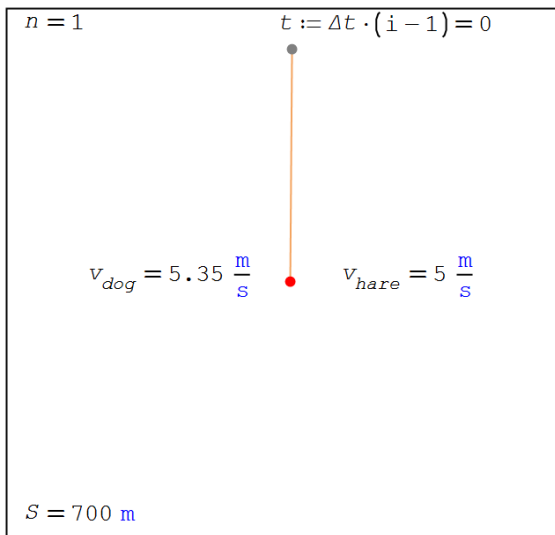
Графики на рисунках 13.13 и 13.14 – это штатные графики пакета SMath, которые вставляются в расчет командами меню Вставить / График / Двухмерный (2D). После вставки в расчет графика (тут можно воспользоваться и горячей клавишей $@$) и его аргумента кривых не будет видно, так как область графика ограничена значениями от -10 до 10 по горизонтали и вертикали. Чтобы изменить область графика, необходимо активировать график щелчком мыши по нему и поработать колесиком мышки (zoom). При этом если прижимать клавиши Shift или Ctrl, это будет приводить к изменению области охвата по горизонтали или вертикали. На рис. 13.13 показаны четыре кадра анимации погони собаки за зайцем. На 518 секунде собака поймает зайца, когда расстояние между ними сократится до 4 мм.

Нижний пятый кадр – это увеличение момента приближения собаки к зайцу. Из этой картинке можно сделать вывод, что на самом деле максимальное сближение собаки с зайцем произошло намного раньше. После этого момента собака обгонит зайца, потом отстанет от него – будет как бы прыгать вокруг зайца, облаивая его, не хватая зубами зайца и не заканчивая погоню. Сразу вспоминается знаменитый парадокс Зенона об Ахиллесе и черепахе. Ахиллес бежит в десять раз быстрее, чем черепаха, и находится позади неё на расстоянии в тысячу шагов. За время, которое Ахиллес пробежит это расстояние, черепаха в ту же сторону проползёт сто шагов. Когда Ахиллес пробежит сто шагов, черепаха проползёт ещё десять шагов, и так далее. Процесс будет продолжаться до бесконечности, Ахиллес так никогда и не догонит черепаху.

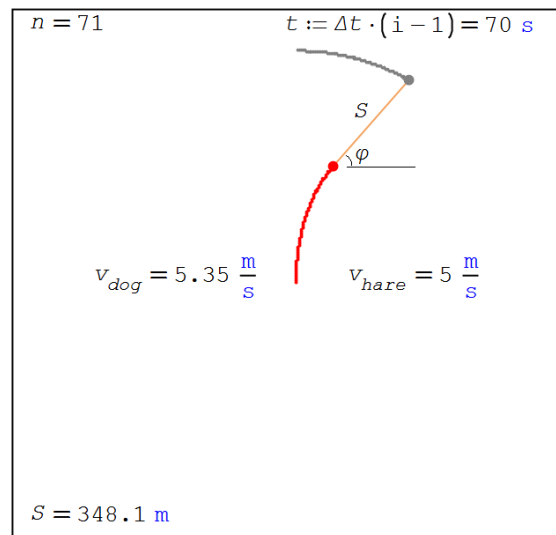
Этот парадокс в математике решен через теорию дифференциального исчисления, когда приращение аргумента превращается в дифференциал. При решении задачи о погоне можно также перейти от приращения Δt к дифференциалу dt , что позволит составить дифференциальное уравнение, а потом решить его. Но беда в том, что дифференциальные уравнения погони аналитически решаются только для простых случаев бега жертвы (зайца). В сложных случаях дифференциальные уравнения приходится решать численно, т. е. отходить от дифференциала назад к приращению. А можно сразу решать задачу о погоне без составления дифференциального уравнения. Что мы и сделали.

Методы численного решения дифференциальных уравнений могут быть с постоянным шагом и с переменным шагом. В нашей задаче о погоне шаг тоже можно сделать переменным. Когда заяц далеко, шаг (скачок) собаки большой. При приближении к зайцу шаг может быть уменьшен.

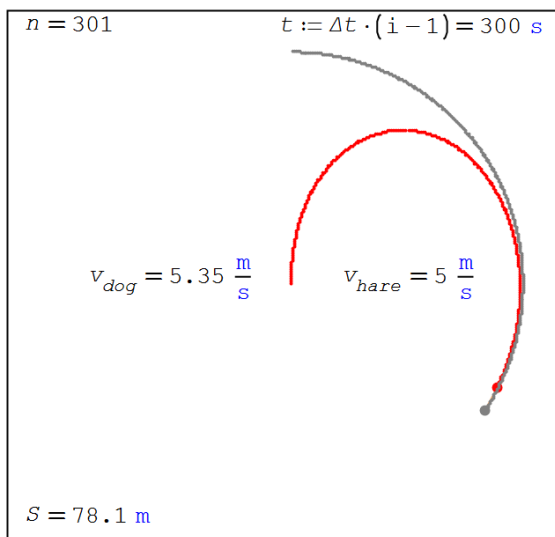
Занятие 13



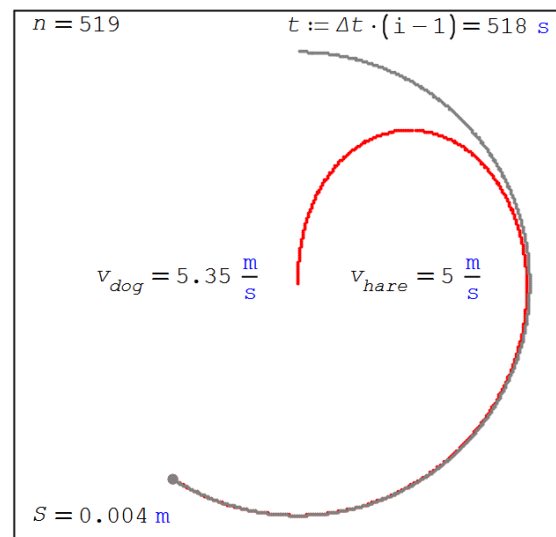
Plot



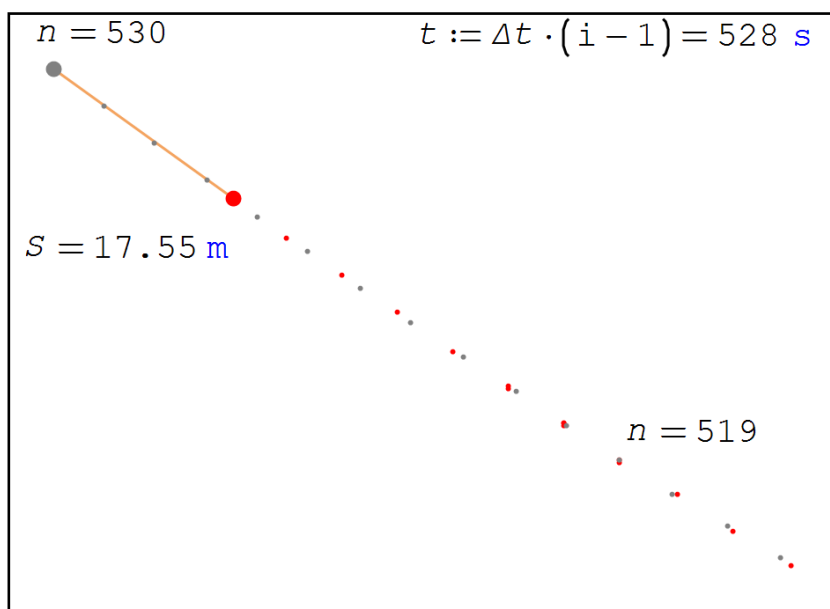
Plot



Plot



Plot

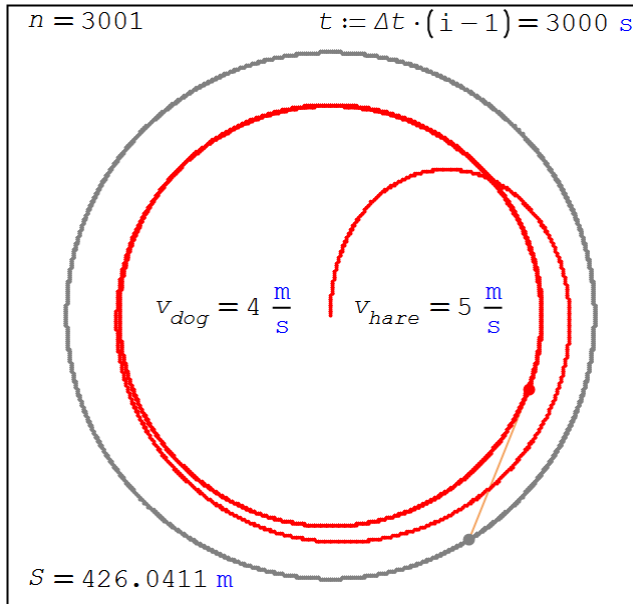


Plot

Занятие 13

Рис. 13.13. Кадры анимации погони собаки за зайцем

Если скорость собаки будет меньше скорости зайца, то собака, сделав некий крюк, выйдет также на стабильное круговое движение – см. рис. 13.14. Радиус окружности, по которой будет бегать собака сзади и сбоку от зайца (пеленг), определить несложно, учитывая тот факт, что у зайца и собаки при разных линейных скоростях будут одинаковые угловые скорости.



Plot

Рис. 13.14. Собака бежит тоже по кругу

Погоню собаки за зайцем лучше наблюдать в анимации.

На рисунке 13.15 показан SMath-документ, позволяющий построить такую анимацию. В нем после ввода исходных данных (без ввода значения n) и расчета значений трех переменных, не зависящих от значения n , создается функция пользователя с именем `frame` (кадр анимации) и с аргументом n . Остальные операторы такие же, как в расчете на рис. 13.12.

Занятие 13

$$R := 700 \text{ m} \quad v_{hare} := 5 \frac{\text{m}}{\text{s}} \quad v_{dog} := 5 \frac{\text{m}}{\text{s}} \quad \Delta t := 1 \text{ s}$$

$$\Delta\alpha := \frac{v_{hare} \cdot \Delta t}{R} \quad x_{dog_1} := -R \quad y_{dog_1} := 0 \text{ m}$$

$$\begin{aligned}
 \text{frame}(n) := & \left[\begin{array}{l}
 i := [1..n] \\
 x_{hare_i} := R \cdot \sin(\Delta\alpha \cdot i) \\
 y_{hare_i} := R \cdot \cos(\Delta\alpha \cdot i) \\
 \varphi_1 := \left(\arctg \left(\frac{y_{hare_1} - y_{dog_1}}{m}, \frac{x_{hare_1} - x_{dog_1}}{m} \right) \right) \\
 \text{for } i \in [1..n] \\
 \left[\begin{array}{l}
 x_{dog_{i+1}} \\
 y_{dog_{i+1}} \\
 \varphi_{i+1}
 \end{array} \right] := \left[\begin{array}{l}
 x_{dog_i} + v_{dog} \cdot \Delta t \cdot \cos(\varphi_i) \\
 y_{dog_i} + v_{dog} \cdot \Delta t \cdot \sin(\varphi_i) \\
 \arctg \left(\frac{y_{hare_i} - y_{dog_i}}{m}, \frac{x_{hare_i} - x_{dog_i}}{m} \right)
 \end{array} \right] \\
 \left. \begin{array}{l}
 \text{augment}(x_{hare}, y_{hare}, ".", 5, \text{"gray"}) \\
 \text{augment}(x_{hare_n}, y_{hare_n}, ".", 15, \text{"gray"}) \\
 \text{augment}(x_{dog}, y_{dog}, ".", 5, \text{"red"}) \\
 \text{augment}(x_{dog_n}, y_{dog_n}, ".", 15, \text{"red"})
 \end{array} \right\} \\
 \left[\begin{array}{ll}
 x_{hare_n} & y_{hare_n} \\
 x_{dog_n} & y_{dog_n}
 \end{array} \right]
 \end{array} \right]
 \end{aligned}$$

Рис. 13.15. Программа анимации погони

Для создания анимации в расчет вставляется график (рис. 13.16 и 13.17), в область данных которого вставлена функция *frame* с аргументом *t*, который автоматически связан с переменной *tFrame* (рис. 13.16), заданной в расчете.

Частота кадров анимации и другие ее параметры задаются через диалоговое окно, показанное на рис. 13.17.

Занятие 13

$tFrame := [1, 10 \dots 1000]$

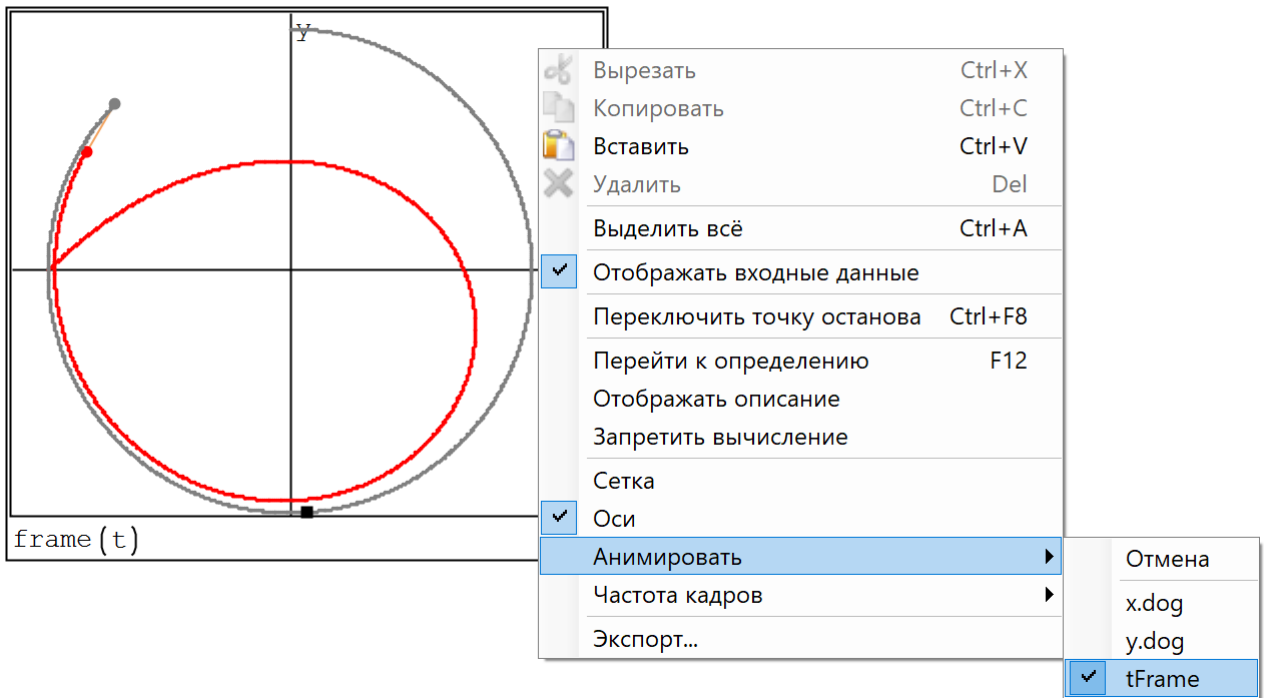


Рис. 13.16. Подготовка к созданию анимации (см. также рис. 1.17 в первой главе)

$tFrame := [1, 10 \dots 1000]$

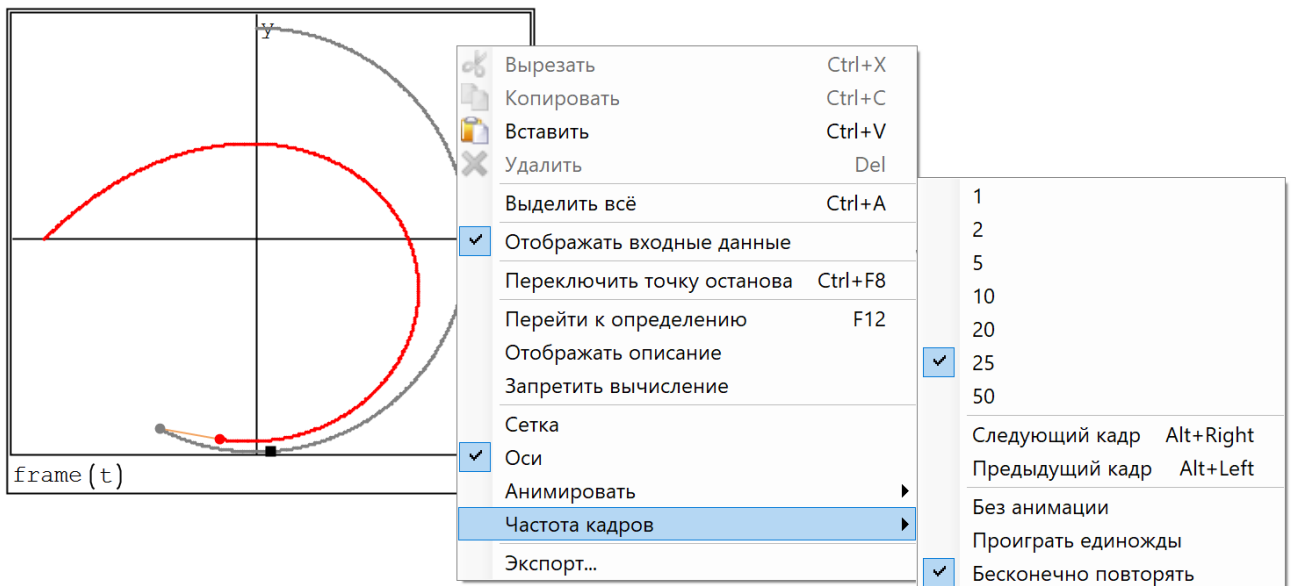


Рис. 13.17. Установка параметров анимации (см. также рис. 1.18 в первой главе)

Команда Экспорт позволяет созданную анимацию экспортировать в gif-файл и использовать её вне пакета SMath. Например, для вставки в документы Word.

Мы уже писали о том, что анимацию нельзя вставлять в электронную версию этой книги.

Занятие 13

На сайте <https://community.ptc.com/t5/Mathcad/Is-it-my-own-or-Mathcad-15-error/td-p/576164/page/2> можно увидеть интересную авторскую анимацию разных случаев погони собаки (волка) за зайцем. А на сайтах <https://community.ptc.com/t5/Mathcad/Mice-problem/m-p/575570> и <https://community.ptc.com/t5/Mathcad/Is-it-my-own-or-Mathcad-15-error/td-p/576164/page/2> размещена анимация частного случая задачи о погоне – задача о летучих мышах, которые попарно гоняются друг за другом.

Автор вместе с И.Е. Васильевой (она упоминается в Предисловии) внесли новое в решении задачи о погоне [4]. Раньше составляли дифференциальные уравнения и пытались решить их численно посредством неких разностных схем. Авторский же подход состоял в том, чтобы сразу решать разностное уравнение, минуя этап составления дифференциального уравнения.

Задания читателям:

1. Определите траекторию бега зайца, если скорость его бега по круглой площадке меньше скорости бега по полю вне круглой площадки. Эта задача на порядок сложнее описанной выше. Некоторые подходы к решению даны здесь <https://community.ptc.com/t5/Mathcad/The-problem-of-cockroach-races/m-p/484692>.
2. Решите обе задачу о беге зайца по полю не через работу с частными производными целевой функции, а через использование закона Снелла.
3. Создайте такую анимацию по рисункам 13.3 и 13.4. Точка x_0 движется слева направо, изменяя траекторию бега зайца. Сначала он бежит по прямой поперек поля, а затем траектория бега зайца становится ломаной линией.
4. Решите задачу для случаев, когда внутренняя площадка со скошенной травой не круглая, а, к примеру, квадратная или овальная.
5. Разбейте область с кругом посередине на две зоны. Точки первой зоны – это стартовое положение зайца, когда он должен бежать к лесу по ломаной через круг. Точки второй зоны – это стартовое положение зайца, когда он должен бежать к лесу по прямой, минуя круг.
6. Попробуйте решить задачу о линзе, перевернув её.
7. Попробуйте решить задачу о двояковыпуклой линзе.

Литература:

1. <https://mathus.ru/phys/geometricaloptics.pdf>
2. Очков В.Ф. Сказка о зайце и геометрической оптике // Физика в школе. № 4. 2022 (<http://tw.t.mpei.ac.ru/ochkov/Optic-Hare-Ochkov.pdf>)
3. В.Ф. Очков, А.В. Соколов, С.Д. Федорович, Л. Мекес. Путешествие от дома в школу по маршруту Ферма, или Второе оптическое свойство гиперболы // Cloud of Science. 2016. Т. 3. № 4. С. 494-517 (<http://www.tw.t.mpei.ac.ru/ochkov/Optic-Ochkov.pdf>)
4. В. Ф. Очков, И. Е. Васильева. Применение разностных схем к решению задачи о погоне // Труды СПИИРАН. Выпуск 18 (6), 2019. С. (<http://proceedings.spiiras.nw.ru/index.php/sp/article/view/4041>)