

Лабораторная работа №2

Задача о погоне

Крутова Екатерина Дмитриевна, НПИбд-01-21

Содержание

Цель работы	5
Задание	6
Теоретическое введение	7
Выполнение лабораторной работы	8
Математическая модель	8
Решение с помощью программ	11
Выводы	16
Список литературы	17

Список иллюстраций

1	Выбор варианта	6
1	Поиск времени и расстояния, после которого катер начнёт раскручиваться по спирали	9
2	Поиск времени и расстояния, после которого катер начнёт раскручиваться по спирали	9
3	тангенциальная скорость	10
4	Система решения	10
5	Решение в полярных координатах	11
6	Установка	12
7	Решение 1	14
8	Решение 2	15

Список таблиц

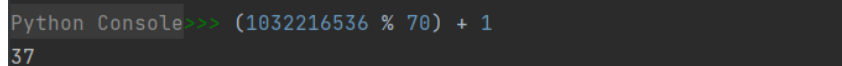
Цель работы

Изучить основы языков программирования Julia и OpenModelica. Освоить библиотеки этих языков, которые используются для построения графиков и решения дифференциальных уравнений. Решить задачу о погоне.

Задание

1. Запишите уравнение, описывающее движение катера, с начальными условиями для двух случаев (в зависимости от расположения катера относительно лодки в начальный момент времени).
2. Постройте траекторию движения катера и лодки для двух случаев.
3. Найдите точку пересечения траектории катера и лодки.

В соответствии с формулой $(S_n \bmod N) + 1$, где S_n — номер студбилета, N — количество заданий, я взяла вариант 37 (рис. [-@fig:001]).



```
Python Console>>> (1032216536 % 70) + 1
37
```

Рис. 1: Выбор варианта

Теоретическое введение

Julia — высокоуровневый высокопроизводительный свободный язык программирования с динамической типизацией, созданный для математических вычислений. Эффективен также и для написания программ общего назначения. Синтаксис языка схож с синтаксисом других математических языков (например, MATLAB и Octave), однако имеет некоторые существенные отличия. Julia написан на Си, C++ и Scheme. Имеет встроенную поддержку многопоточности и распределённых вычислений, реализованные в том числе в стандартных конструкциях.

Физические термины:

- Тангенциальная скорость - составляющая вектора скорости, перпендикулярная линии, соединяющей источник и наблюдателя. Измеряется собственному движению - угловому перемещению источника.
- Радиальная скорость — проекция скорости точки на прямую, соединяющую её с выбранным началом координат.
- Полярная система координат — двумерная система координат, в которой каждая точка на плоскости определяется двумя числами — полярным углом и полярным радиусом.

Выполнение лабораторной работы

Математическая модель

1. Примем за момент отсчета времени момент первого рассеивания тумана. Введем полярные координаты с центром в точке нахождения браконьеров и осью, проходящей через катер береговой охраны. Тогда начальные координаты катера $(14.1; 0)$. Обозначим скорость лодки v .
2. Траектория катера должна быть такой, чтобы и катер, и лодка все время были на одном расстоянии от полюса. Только в этом случае траектория катера пересечется с траекторией лодки. Поэтому для начала катер береговой охраны должен двигаться некоторое время прямолинейно, пока не окажется на том же расстоянии от полюса, что и лодка браконьеров. После этого катер береговой охраны должен двигаться вокруг полюса удаляясь от него с той же скоростью, что и лодка браконьеров.
3. Чтобы найти расстояние x (расстояние после которого катер начнет двигаться вокруг полюса), необходимо составить следующие уравнение. Пусть через время t катер и лодка окажутся на одном расстоянии x от полюса. За это время лодка пройдет x , а катер $14.1+x$ (или $14.1-x$, в зависимости от начального положения катера относительно полюса). Время, за которое они пройдут это расстояние, вычисляется как рис. [-@fig:002].

$$\frac{x}{v} = \frac{14.1 - x}{3.9v}$$

Решить

Найдите v

$v \neq 0, x = \frac{141}{49}$

Найдите x

$x = \frac{141}{49}, v \neq 0$

Рис. 1: Поиск времени и расстояния, после которого катер начнёт раскручиваться по спирали

Или как рис. [-@fig:003].

$$\frac{x}{v} = \frac{x + 14.1}{3.9v}$$

Решить

Найдите v

$v \neq 0, x = \frac{141}{29}$

Найдите x

$x = \frac{141}{29}, v \neq 0$

Рис. 2: Поиск времени и расстояния, после которого катер начнёт раскручиваться по спирали

Так как время должно быть одинаковым, эти величины тоже будут друг другу равны. Из этого получаем объединение из двух уравнений (двух из-за двух разных изначальных позиций катера относительно полюса).

Задачу будем решать для двух случаев. После того, как катер береговой охраны окажется на одном расстоянии от полюса, что и лодка, он должен сменить прямолинейную траекторию и начать двигаться вокруг полюса удаляясь от него со скоростью лодки v . Для этого скорость катера раскладываем на две составляющие: радиальная скорость и тангенциальная скорость (рис. [-@fig:004]).

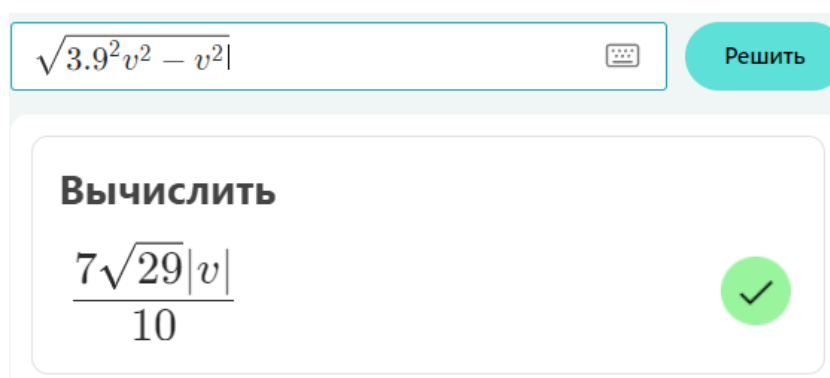


Рис. 3: тангенциальная скорость

Решение исходной задачи (рис. [-@fig:005] - [-@fig:006]):

$$\begin{cases} \frac{dr}{dt} = v \\ r \frac{d\theta}{dt} = (7\sqrt{29}) \cdot 0.5 / 10 \end{cases} \text{ с начальными условиями } \begin{cases} \theta_0 = 0 \\ r_0 = x_1 \end{cases} \text{ или } \begin{cases} \theta_0 = -\pi \\ r_0 = x_2 \end{cases}$$

Рис. 4: Система решения

$$\frac{dr}{d\theta} = \frac{r}{\sqrt{29}}$$

Рис. 5: Решение в полярных координатах

Решение с помощью программ

OpenModelica

К сожалению, OpenModelica не адаптирована к использованию полярных координат, поэтому адекватное отображение результатов данной задачи там невозможно. [1]

Julia

Решить дифференциальное уравнение, расписанное в постановке задачи лабораторной работы, поможет библиотека DifferentialEquations. Итоговые изображения в полярных координатах будут строиться через библиотеку Plots. [2]

Для начала нужно установить Julia (рис. [-@fig:007])



Рис. 6: Установка

Код программы:

```
using Plots
using DifferentialEquations

const a = 14.1
const n = 3.9

const r0 = a/(n + 1)
const r0_2 = a/(n - 1)

const T = (0, 2*pi)
const T_2 = (-pi, pi)
```

```

function F(u, p, t)
    return u / sqrt(n*n - 1)
end

problem = ODEProblem(F, r0, T)

result = solve(problem, abstol=1e-8, reltol=1e-8)
@show result.u
@show result.t

dxR = rand(1:size(result.t)[1])
rAngles = [result.t[dxR] for i in 1:size(result.t)[1]]

plt = plot(proj=:polar, aspect_ratio=:equal, dpi = 1000, legend=true, bg=:white)

plot!(plt, xlabel="theta", ylabel="r(t)", title="Задача о погоне - случай 1", legend=:none)
plot!(plt, [rAngles[1], rAngles[2]], [0.0, result.u[size(result.u)[1]]], label="Путь катера")
scatter!(plt, rAngles, result.u, label="", mc=:blue, ms=0.0005)
plot!(plt, result.t, result.u, xlabel="theta", ylabel="r(t)", label="Путь катера")
scatter!(plt, result.t, result.u, label="", mc=:green, ms=0.0005)

savefig(plt, "lab02_01.png")

problem = ODEProblem(F, r0_2, T_2)
result = solve(problem, abstol=1e-8, reltol=1e-8)
dxR = rand(1:size(result.t)[1])
rAngles = [result.t[dxR] for i in 1:size(result.t)[1]]

plt1 = plot(proj=:polar, aspect_ratio=:equal, dpi = 1000, legend=true, bg=:white)

```

```

plot!(plt1, xlabel="theta", ylabel="r(t)", title="Задача о погоне - случай 2", label="")
plot!(plt1, [rAngles[1], rAngles[2]], [0.0, result.u[size(result.u)[1]]], label="")
scatter!(plt1, rAngles, result.u, label="", mc=:blue, ms=0.0005)
plot!(plt1, result.t, result.u, xlabel="theta", ylabel="r(t)", label="Путь катера")
scatter!(plt1, result.t, result.u, label="", mc=:green, ms=0.0005)

savefig(plt1, "lab02_02.png")

```

Итоговые графики траектории движения катера и лодки для случая обоих случаев.:
(рис. [-@fig:008] - [-@fig:009])

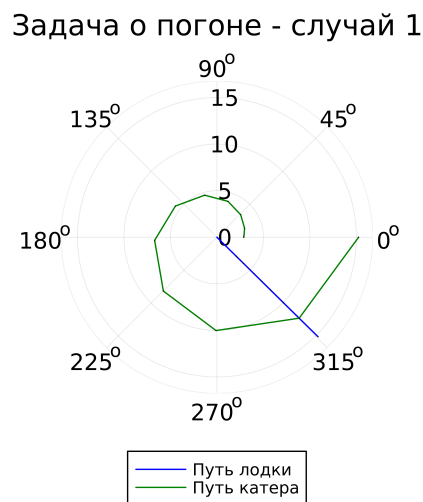


Рис. 7: Решение 1

Задача о погоне - случай 2

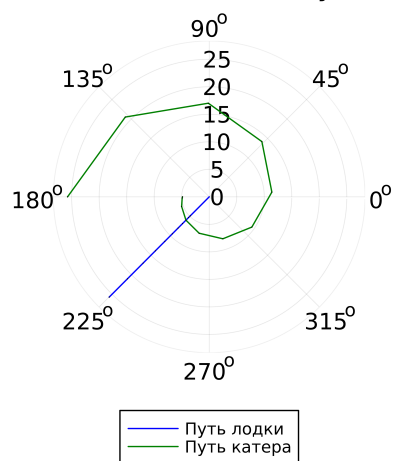


Рис. 8: Решение 2

Выводы

Были изучены основы языков программирования Julia и OpenModelica. Освоены библиотеки этих языков, которые используются для построения графиков и решения дифференциальных уравнений. Поскольку OpenModelica не работает с полярными координатами, она пока что не была использована в данной лабораторной работе.

Список литературы

- [1] Документация по OpenModelica: <https://openmodelica.org/>
- [2] Документация по Julia: <https://docs.julialang.org/en/v1/>