

# **Лабораторная работа №2**

**Задача о погоне. Вариант 39**

Абдуллина Ляйсан Раисовна, НПИбд-01-21

# Содержание

<b>Цель работы</b>	<b>4</b>
<b>Задачи</b>	<b>5</b>
<b>Теоретическое введение</b>	<b>6</b>
Julia . . . . .	6
OpenModelica . . . . .	6
<b>Выполнение лабораторной работы</b>	<b>7</b>
Произведение расчетов . . . . .	7
Моделирование . . . . .	12
Результаты работы . . . . .	13
<b>Выводы</b>	<b>15</b>

## Список иллюстраций

1	Вычисление варианта . . . . .	7
2	Вычисления . . . . .	9
3	Вычисления . . . . .	9
4	Разложение скорости катера на тангенциальную и радиальную составляющие . . . . .	10
5	Вычисления . . . . .	11
6	Установка Julia . . . . .	12
7	Результат запуска программы - график №1 . . . . .	14
8	Результат запуска программы - график №2 . . . . .	14

## Цель работы

Решить задачу о погоне. Изучить основы языков программирования OpenModelica и Julia.

## Задачи

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

# Теоретическое введение

## Julia

Julia — высокоуровневый 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.

## OpenModelica

OpenModelica — свободное открытое программное обеспечение для моделирования, симуляции, оптимизации и анализа сложных динамических систем. Основано на языке Modelica. OpenModelica используется в академической среде и на производстве. В промышленности используется в области оптимизации энергоснабжения, автомобилестроении и водоочистке.

# Выполнение лабораторной работы

Выбор варианта вычислялся из остатка деления студенческого билета на количество вариантов, плюс один. Таким образом Получили 39 вариант (Рис.1).

$$1032216538 \div 70 = 14745950 \text{ целых } 38 \text{ в остатке} = 14745950 \frac{38}{70}$$

$$\begin{array}{r} 1032216538 \overline{)70} \\ \underline{-70} \phantom{00000000} \\ 332 \phantom{00000000} \\ \underline{-280} \phantom{00000000} \\ 522 \phantom{00000000} \\ \underline{-490} \phantom{00000000} \\ 321 \phantom{00000000} \\ \underline{-280} \phantom{00000000} \\ 416 \phantom{00000000} \\ \underline{-350} \phantom{00000000} \\ 665 \phantom{00000000} \\ \underline{-630} \phantom{00000000} \\ 353 \phantom{00000000} \\ \underline{-350} \phantom{00000000} \\ 38 + 1 = 39 \end{array}$$

Рис. 1: Вычисление варианта

## Произведение расчетов

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 (21; 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  $21 + x$  (или  $21 - x$ , в зависимости от начального положения катера относительно полюса). Время, за которое они пройдут это расстояние, вычисляется как  $x/v$  или  $(21 - x)/5,5v$ ,  $((21 + x)/5,5v)$ . 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):

$$\begin{cases} x/v = (21 - x)/5,5v \\ x/v = (21 + x)/5,5v \end{cases}$$

Из данных уравнений можно найти расстояние, после которого катер начнет раскручиваться по спирали. Для данных уравнений решения будут следующими (Рис.2-3):  $x_1 = 42/13$ ,  $x_2 = 14/3$ .



Calculator

$$\frac{x}{v} = \frac{21-x}{5.5v}$$


---


$$x = \frac{42}{13}, v \neq 0$$

Alternative Form

$$x \approx 3.23077, v \neq 0$$

Рис. 2: Вычисления

Calculator

$$\frac{x}{v} = \frac{x+21}{5.5v}$$


---


$$x = \frac{14}{3}, v \neq 0$$

Alternative Form

$$x \approx 4.66667, v \neq 0$$

Рис. 3: Вычисления

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  $v$ . (Рис.4)

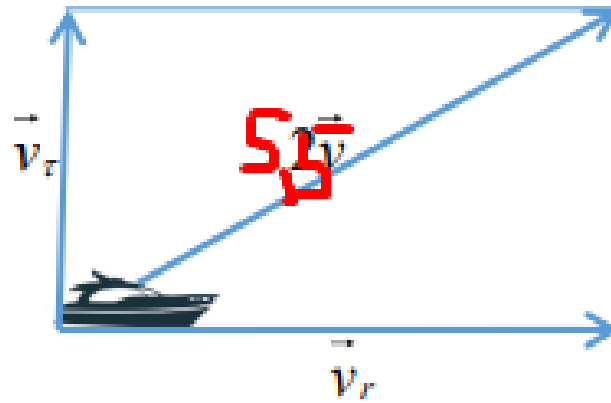


Рис. 4: Разложение скорости катера на тангенциальную и радиальную составляющие

Для этого скорость катера раскладываем на две составляющие:  $v_r = dr/dt = v$  - радиальная скорость и  $v_\tau = r d\theta/dt$  - тангенциальная скорость (Рис.5).

$$v_\tau = 3\sqrt{13}v/2$$

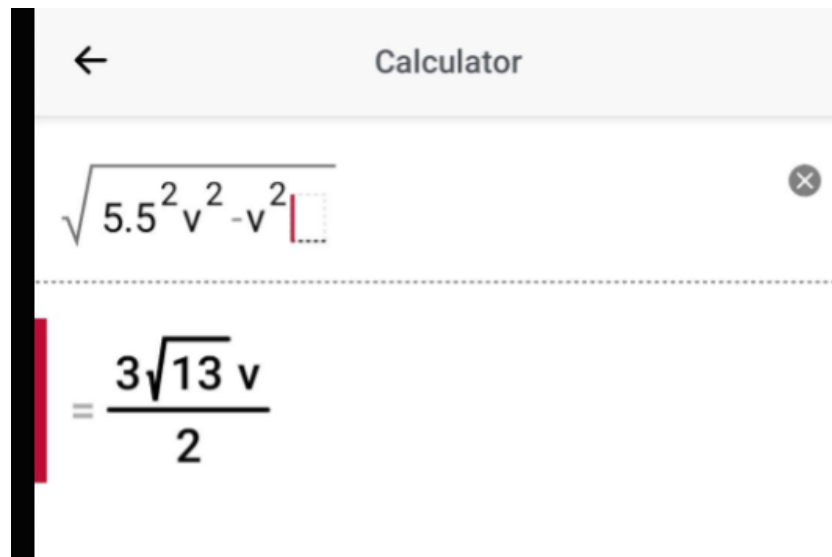


Рис. 5: Вычисления

4. Решение исходной задачи сводится к решению системы из двух дифференциальных уравнений:

$$\begin{cases} dr/dt = v \\ r d\theta/dt = 3\sqrt{13}v/2 \end{cases}$$

с начальными условиями

$$\begin{cases} \theta_0 = 0 \\ r_0 = x_1 = 42/13 \end{cases}$$

или

$$\begin{cases} \theta_0 = -\pi \\ r_0 = x_2 = 14/3 \end{cases}$$

Исключая из полученной системы производную по  $t$ , можно перейти к следующему уравнению (с неизменными начальными условиями):

$$dr/d\theta = 2r/3\sqrt{13}$$

Решением этого уравнения с заданными начальными условиями и будет являться траектория движения катера в полярных координатах.

## Моделирование

OpenModelica не может быть использована для этой задачи, так как здесь используются полярные координаты.

Установка Julia и необходимых для нее пакетов (Рис.6).

```

C:\WINDOWS\system32> winget install julia -s msstore
Перед использованием источника "msstore" необходимо просмотреть следующие соглашения.
Terms of Transaction: https://aka.ms/microsoft-store-terms-of-transaction
Для правильной работы источника требуется отправить во внутреннюю службу двухбуквенный код текущего региона компьютера (например, "RU").

Вы согласны со всеми условиями исходных соглашений?
[Y] Да [N] Нет: y
Найдено Julia [9N3JMM8PVKMN] Версия Unknown
Этот пакет предоставляется через Microsoft Store. Программе winget может потребоваться получить пакет в Microsoft Store от имени текущего пользователя.
Соглашения для Julia [9N3JMM8PVKMN] Версия Unknown
Версия: Unknown
Издатель: Julia Computing, Inc.
URL-адрес издателя: https://juliai.org/
Описание: Julia is a high-level, high-performance, dynamic, open-source programming language.
Лицензия: ms-windows-store://pdp/?ProductId=9N3JMM8PVKMN
URL-адрес заявления о конфиденциальности: https://juliacomputing.com/privacy/
Соглашения:
Category: Developer tools
Pricing: Free
Free Trial: No
Terms of Transaction: https://aka.ms/microsoft-store-terms-of-transaction
Seizure Warning: https://aka.ms/microsoft-store-seizure-warning
Store License Terms: https://aka.ms/microsoft-store-license

Издатель требует, чтобы вы просмотрели указанную выше информацию и приняли соглашения перед установкой.
Вы согласны с условиями?
[Y] Да [N] Нет: y
Запуск установки пакета...
  
```

Рис. 6: Установка Julia

Исходный код программы:

using Plots using DifferentialEquations

#расстояние от лодки до катера const a = 21 const n = 5.5

#расстояние начала закругления-спирали const r0 = a/(n + 1) const r0\_2 = a/(n - 1)

#интервал const T = (0, 2\*pi) const T\_2 = (-pi, pi)

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

#задача ОДУ problem = ODEProblem(F, r0, T)

#решение 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]]

#холст plt = plot(proj=:polar, aspect\_ratio=:equal, dpi = 1000, legend=true, bg=:white)

#параметры для холста plot!(plt, xlabel="theta", ylabel="r(t)", title="Абдуллина. Вap

39. Задача о погоне. Случай 1", legend=:outerbottom) plot!(plt, [rAngles[1], rAngles[2]],

```
[0.0, result.u[size(result.u)[1]]], label="Путь лодки", color=:blue, lw=1) 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", color=:green, lw=1) scatter!(plt, result.t, result.u, label="",
mc=:green, ms=0.0005)

savefig(plt, "lab2_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]]

#холст2 plt1 = plot(proj=:polar, aspect_ratio=:equal, dpi = 1000, legend=true, bg=:white)
#параметры для холста plot!(plt1, xlabel="theta", ylabel="r(t)", title="Абдуллина. Вар
39. Задача о погоне. Случай 2", legend=:outerbottom) plot!(plt1, [rAngles[1], rAngles[2]],
[0.0, result.u[size(result.u)[1]]], label="Путь лодки", color=:blue, lw=1) 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", color=:green, lw=1) scatter!(plt1, result.t, result.u,
label="", mc=:green, ms=0.0005)

savefig(plt1, "lab2_02.png")
```

## Результаты работы

Запуск программы и получение результатов (Рис.7-9):

```
C:\Users\oalil\work\study\2023-2024\Математическое моделирование\lastmod\lab2\report> julia lab2.jl
result.u = [3.238769238769231, 3.269356684887744, 3.4336196622498657, 3.620872859383838, 4.363516665632019, 5.056952374845729, 5.969153836832315, 7.08672278054]
result.t = [0.0, 0.049308882029284666, 0.32933855682380794, 0.3872935688605543, 1.6235161688816658, 2.423167036596137, 3.320888612464826, 4.28836095928]
C:\Users\oalil\work\study\2023-2024\Математическое моделирование\lastmod\lab2\report>
```

Абдуллина. Вар 39. Задача о погоне. Случай 1

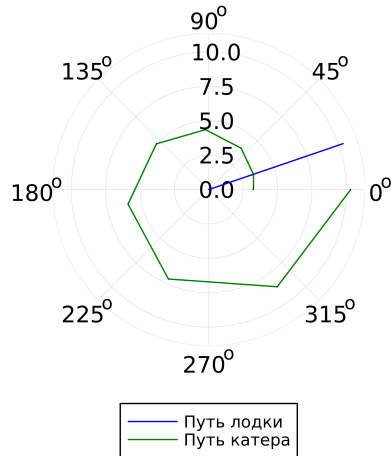


Рис. 7: Результат запуска программы - график №1

Абдуллина. Вар 39. Задача о погоне. Случай 2

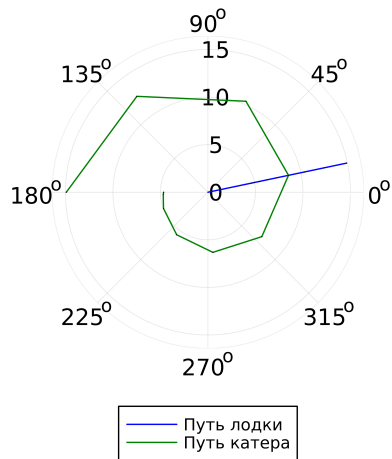


Рис. 8: Результат запуска программы - график №2

## Выводы

Мы смогли решить задачу о погоне, изучить основы языков программирования Julia, а также выполнили все поставленные задачи: были построены графики для обоих случаев, где получилось отрисовать траекторию катера, траекторию лодки и получилось наглядно найти их точки пересечения.