Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО «АЛТАЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт цифровых технологий, электроники и физики

Кафедра вычислительной техники и электроники (ВТиЭ)

Лабораторная работа №2

**Познакомиться с программой-симулятором робототехники Webots.**

|  |
| --- |
| Выполнил: студент гр. 5.306М:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лаптев А. В.  Проверил: доц. каф. ВТиЭ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Соловьев А. А.  Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

**Цель работы:** познакомиться с программой-симулятором робототехники Webots.

**Задача:** для модели 4-х колесного робота реализовать ПИД-регулятордля стабилизации скорости при движении по пересеченной местности.

**Ход работы:**

1. После создания проекта в дерево проекта было необходимо добавить карту, которая имитирует пересеченную местность. Для реализации этого шага был добавлен и настроен новый node из списка PROTO для Webots Project - Uneven Terrain. Это шаблон, на котором можно случайным образом сгенерировать неровности. Параметры неровности задаются значением randomSeed. Я выбрал значение 21, чтобы препятствия не получились слишком высокими.
2. Затем необходимо было выбрать 4-х колесного робота, для которого и будет реализован ПИД-регулятор. Для этого в дерево проекта через то же самое меню был добавлен робот Pioneer 3-AT. У него есть двигатель на каждое из 4 колес и максимальная скорость 6.4.
3. Для того, чтобы робот мог двигаться для него нужно было создать новый контроллер, в котором реализован ПИД-регулятор.

Алгоритм работы контроллера, реализующего ПИД-регулятор:

Начало

1. Создание экземпляра класса робота.
2. Получение доступа к мотором робота по их названию и установка начальных параметров для каждого колеса.
3. Установка коэффициентов ПИД-регулятора (Kp, Ki, Kd) и желаемой скорости.
4. Создание экземпляра ПИД-реулятора.
5. Задание временного промежутка движения робота (значение шага).

Цикл

1. Если робот остановился, то завершение работы программы.
2. Получение текущей скорости каждого колеса робота.
3. Вычисление средней скорости всех колес робота и вывод этого значения в консоль.
4. Получение корректирующего значения от ПИД-регулятора.
5. Установка скорректированного значения в качестве параметра скорости для каждого колеса.

Конец

Класс ПИД-регулятора:

Начало

1. Инициализация класса ПИД-регулятора с инициализацией начальных параметров и методов.

Метод для корректировки

1. Расчет значения ошибки.
2. Корректировка интегратора на значение ошибки.
3. Расчет производной как разности текущей и предыдущей ошибок.
4. Получение выходного значения сигнала по формуле ПИД-регулятора.
5. Сохранение текущего значения ошибки для дальнейшего использование.
6. Возврат выходного значения сигнала.

Конец

Программный код:

from controller import Robot, Motor

class PIDController:

def \_\_init\_\_(self, Kp, Ki, Kd, setpoint):

self.Kp = Kp

self.Ki = Ki

self.Kd = Kd

self.setpoint = setpoint

self.prev\_error = 0

self.integral = 0

def update(self, feedback\_value):

# Функция корректировки

error = self.setpoint - feedback\_value

self.integral += error

derivative = error - self.prev\_error

output = self.Kp \* error + self.Ki \* self.integral + self.Kd \* derivative

self.prev\_error = error

return output

robot = Robot()

# Получение доступа к моторам робота

wheels = []

wheel\_names = ["front left wheel", "front right wheel", "back left wheel", "back right wheel"]

for name in wheel\_names:

# Установка начальных параметров колес

wheel = robot.getDevice(name)

wheel.setPosition(float('inf'))

wheel.setVelocity(0)

wheels.append(wheel)

# Параметры ПИД-регулятора

Kp = 0.5

Ki = 0.2

Kd = 0.1

setpoint\_speed = 6 # Желаемая скорость

# Создание экземпляра ПИД-регулятора

pid\_controller = PIDController(Kp, Ki, Kd, setpoint\_speed)

# Основной цикл управления

timestep = int(robot.getBasicTimeStep())

while robot.step(timestep) != -1:

# Получение текущей скорости робота

velocities = []

for wheel in wheels:

velocities.append(wheel.getVelocity())

# Вычисление средней скорости всех колес

wheels\_speed = sum(velocities) / len(wheels)

print(wheels\_speed)

# Получение корректирующего сигнала от ПИД-регулятора

control\_signal = pid\_controller.update(wheels\_speed)

# Применение корректировки к моторам

for wheel in wheels:

wheel.setVelocity(control\_signal)