Тема 3 · Урок 3.2

П-регулятор: плавно и быстро

Робот поворачивает тем сильнее, чем дальше ушла линия, и проходит дуги без рывков.

2 × 45 мин | 1–2 ученика на робота | Практика

РЕЗУЛЬТАТ УРОКА

Робот плавно едет по линии с П-регулятором и проходит дугу своего уровня. Коэффициент и скорость подобраны и записаны в журнал.

Ты научишься

- Перевести показания двух датчиков в общую шкалу от 0 до 100.
- Посчитать ошибку как разницу показаний и объяснить её знак.
- Объяснить, как поправка меняет скорости колёс.
- Подобрать коэффициент и скорость по наблюдениям на дорожке.

Нужно знать

- 3.1 Датчики линии и первое следование
- 2.1 Моторы: едем и останавливаемся

Понадобится

Датчик линии (аналоговый, Trema) × 2 | USB-кабель для Arduino

Зачем этот урок

В уроке 3.1 робот ехал по линии рывками: колесо либо крутилось, либо стояло. На крутых дугах итоговой задачи так не проехать. Сегодня робот научится поворачивать **чуть-чуть**, когда линия ушла немного, и **сильно**, когда ушла далеко. Так работают автопилоты и роботы-курьеры.

РАЗМИНКА · ВСПОМНИ

1. Как программа из урока 3.1 решала, что под датчиком линия?
2. Почему робот в уроке 3.1 ехал рывками?
3. Что делает функция `drive()`?

ЗАНЯТИЕ 1 · 45 МИН

1 Ошибка вместо «да или нет»

1 Плавно — это как?

8 мин

Что должен делать робот. Роботу нужно двигаться по линии плавнее: при маленьком отклонении чуть поправлять курс, а при большом — сильнее. Используем те же датчики отражения и моторы, добавив П-регулятор — правило расчёта поправки.

Что используем и как это работает. Сначала переводим измерения каждого датчика в общую шкалу по его белому и чёрному. Разность уровней называется ошибкой. Коэффициент kp умножает ошибку: чем больше отклонение, тем больше поправка. Одному колесу прибавляем её к базовой команде, у другого вычитаем. Новое измерение позволяет снова скорректировать движение — это обратная связь.

Как связать измерение с действием. Слишком малая поправка не успевает вернуть робота, слишком большая вызывает колебания. Меняем один параметр за раз и сравниваем траекторию и показания на одинаковой трассе. Нулевая ошибка означает равные показания, но не всегда правильное положение: оба датчика могут потерять линию.

Представь, что ты ведёшь велосипед по линии: чуть ушла — чуть поворачиваешь руль, сильно ушла — сильнее. Программа из урока 3.1 так не умеет: она знает только «линия есть» или «нет».

1. Запусти программу из урока 3.1 на дуге 350 мм и посмотри, как едет робот.
2. Запиши в журнал одной фразой: чего не хватает программе, чтобы поворачивать плавно?

✓ **ДОЛЖНО ПОЛУЧИТЬСЯ**

Ты можешь объяснить: сила поворота должна зависеть от того, насколько далеко ушла линия.

2 Общая шкала для двух датчиков

15 мин

У датчиков разные числа белого и чёрного, поэтому сравнивать их напрямую нельзя. Нужна общая шкала: **чёрное — 0, белое — 100**. Показание на шкале — это «сколько процентов пути от чёрного до белого» оно прошло.

1. Создай скетч `L32_Error` по шаблону, перенеси измерения из журнала.
2. Напиши функцию `level` сам. Ошибку в `loop()` пока не трогай.

Шаблон: шкала и ошибка L32_Error_Gaps.ino

```
// Наблюдение в MyRobotica включается сборщиком; обычный Arduino работает без
#include <math.h>
#if defined(ROBOT_TELEMETRY_BUILD_ID)
#include <RobotTelemetry.h>
robot_telemetry::TelemetryStream courseTelemetry;
#endif

// Шаблон среднего уровня. Напиши код в разделе «Твоя часть» по целям в комме

const int LEFT_SENSOR_PIN = A4;
const int RIGHT_SENSOR_PIN = A5;

// Измерения белого и чёрного из журнала (урок 3.1).
const int LEFT_WHITE = 0;
const int LEFT_BLACK = 0;
const int RIGHT_WHITE = 0;
const int RIGHT_BLACK = 0;

// ===== Твоя часть: напиши сам =====

int level(int value, int white, int black) {
  // Шкала: чёрное — 0, белое — 100, между ними — по прямой.
  // Работает и когда у датчика чёрное — большое число, и когда маленькое; не
  return 0;
}

void loop() {
  int leftRaw = analogRead(LEFT_SENSOR_PIN);
  int rightRaw = analogRead(RIGHT_SENSOR_PIN);
  int leftLevel = level(leftRaw, LEFT_WHITE, LEFT_BLACK);
  int rightLevel = level(rightRaw, RIGHT_WHITE, RIGHT_BLACK);

  // Ошибка: насколько и в какую сторону линия ушла от середины; линия посере
  float error = NAN; // Пока ошибка не вычислена, график показывает «нет данн

  courseTelemetryReport(leftRaw, rightRaw, error, NAN);
  Serial.print("L=");
  Serial.print(leftLevel);
  Serial.print("  R=");
  Serial.print(rightLevel);
  Serial.print("  ошибка=");
```

```
  Serial.println(error);
  courseWait(200);
}

// ===== Готовое: настройка и защита — не меняй =====

void setup() {
  courseTelemetryBegin();
  if (LEFT_WHITE == LEFT_BLACK || RIGHT_WHITE == RIGHT_BLACK) {
    Serial.println("Впиши измерения белого и чёрного");
    while (true) {
      courseTelemetryPoll();
    }
  }
}

// ===== Готовое: передача наблюдений — не меняй =====
void courseTelemetryBegin() {
#if defined(ROBOT_TELEMETRY_BUILD_ID)
  using namespace robot_telemetry::wire;
  courseTelemetry.begin((1UL << Line0) | (1UL << Line1) |
                        (1UL << LineError) | (1UL << ControlOutput));
#else
  Serial.begin(9600);
#endif
}

void courseTelemetryPoll() {
#if defined(ROBOT_TELEMETRY_BUILD_ID)
  courseTelemetry.update();
#endif
}

// NAN означает «ещё не вычислено», а не нулевую ошибку или поправку.
void courseTelemetryReport(int leftRaw, int rightRaw, float error, float outp
#if defined(ROBOT_TELEMETRY_BUILD_ID)
  using namespace robot_telemetry::wire;
  courseTelemetry.setUnsigned(Line0, leftRaw);
  courseTelemetry.setUnsigned(Line1, rightRaw);
  if (isnan(error)) courseTelemetry.setMissing(LineError);
  else courseTelemetry.setFloat(LineError, error);
  if (isnan(output)) courseTelemetry.setMissing(ControlOutput);
  else courseTelemetry.setFloat(ControlOutput, output);
```

```
  courseTelemetry.update();
#endif
}

void courseWait(unsigned long durationMs) {
  unsigned long started = millis();
  while (millis() - started < durationMs) {
    courseTelemetryPoll();
    delay(1);
  }
}
```

ПРЕДСКАЖИ

У датчика белое — 120, чёрное — 780. Какое число шкалы получится для показания 450? Посчитай до запуска.

3. Загрузи и проверь на белом, чёрном и краю линии.
4. Подними робота на 3–4 см над столом: датчику почти нечего отразить. Какое «сырое» число получается, что показывает шкала и какая часть функции за это отвечает?
5. Ответь в журнале: какого типа число ты взял для расчёта и почему не обычный `int`?

Подсказка: С чего начать формулу

Путь от чёрного до белого — `white - black`. Пройденная часть — `value - black`. Осталось превратить их отношение в проценты.

Подсказка: Шкала скачет у белого

На Arduino Uno `int` хранит числа только до 32 767. Посчитай, сколько выйдет `(value - black) * 100` для показания 50 и чёрного 780.

Подсказка: Шкала уходит за 0…100

`constrain(x, a, b)` возвращает `x`, но не меньше `a` и не больше `b`.

ДОЛЖНО ПОЛУЧИТЬСЯ

На белом около 100, на чёрном около 0, для показания 450 из примера — 50. Шкала не уходит за 0…100, и ты знаешь почему. В журнале — ответ про тип числа.

3 Считаем ошибку

15 мин

Ошибка показывает, насколько и в какую сторону линия ушла от середины. Считаем её как разницу: левое показание минус правое.

1. Напиши расчёт ошибки в `loop()` сам и загрузи скетч.

ПРЕДСКАЖИ

Линия ушла под левый датчик. Какой знак будет у ошибки? Почему?

2. Проверь прогноз: подвинь линию под левый датчик, потом под правый. Запиши в журнал.

Где линия	leftLevel	rightLevel	Ошибка
Посередине			
Под левым			
Под правым			

3. Ответь в журнале: в какую сторону должен повернуть робот, если ошибка отрицательная?

Подсказка: Знак ошибки

Чёрная линия — 0 на шкале. Если линия под левым датчиком, какое из двух чисел меньше?

ДОЛЖНО ПОЛУЧИТЬСЯ

Знаки в таблице совпали с прогнозом, и ты можешь объяснить, куда поворачивать при каждом знаке.

2

ЗАНЯТИЕ 2 · 45 МИН

П-регулятор на дорожке

4 От ошибки к скоростям

10 мин

Правило П-регулятора: **поправка пропорциональна ошибке**. Чем дальше ушла линия, тем сильнее одно колесо ускоряется, а другое замедляется. Насколько сильно — задаёт `kp`.

1. Создай скетч `L32_P_Line` по шаблону, перенеси измерения и свою функцию `level`.
2. Напиши `loop()` сам: ошибка, поправка, скорости колёс в пределах 0…255 и команда моторам.

Шаблон: П-регулятор L32_P_Line_Gaps.ino

```
// Наблюдение в MyRobotica включается сборщиком; обычный Arduino работает без
#include <math.h>
#if defined(ROBOT_TELEMETRY_BUILD_ID)
#include <RobotTelemetry.h>
robot_telemetry::TelemetryStream courseTelemetry;
#endif

// Шаблон среднего уровня. Напиши код в разделе «Твоя часть» по целям в комме

const int LEFT_SENSOR_PIN = A4;
const int RIGHT_SENSOR_PIN = A5;

// Измерения белого и чёрного из журнала (урок 3.1).
const int LEFT_WHITE = 0;
const int LEFT_BLACK = 0;
const int RIGHT_WHITE = 0;
const int RIGHT_BLACK = 0;
const int LEFT_DIR_PIN = 7;
const int LEFT_SPEED_PIN = 6;
const int RIGHT_DIR_PIN = 4;
const int RIGHT_SPEED_PIN = 5;
const int FORWARD = HIGH;
float kp = 0.5;     // сила поворота: подбираем на дорожке
int baseSpeed = 100; // скорость на прямой

// ===== Твоя часть: напиши сам =====

int level(int value, int white, int black) {
  // Перенеси свою функцию level из L32_Error.
  return 0;
}

void loop() {
  float observedError = NAN;
  float observedOutput = NAN;
  int leftRaw = analogRead(LEFT_SENSOR_PIN);
  int rightRaw = analogRead(RIGHT_SENSOR_PIN);
  int leftLevel = level(leftRaw, LEFT_WHITE, LEFT_BLACK);
  int rightLevel = level(rightRaw, RIGHT_WHITE, RIGHT_BLACK);

  // Ошибка — как в L32_Error.
```

```
  // Поправка пропорциональна ошибке, коэффициент kp.
  // Робот поворачивает к линии тем сильнее, чем больше ошибка; скорость кажд
  // Отправь скорости на моторы.
  // Сохрани вычисленные ошибку и поправку в observedError и observedOutput д
  courseTelemetryReport(leftRaw, rightRaw, observedError, observedOutput);
}

// ===== Готовое: моторы, настройка и защита — не меняй =====

void drive(int leftDir, int leftSpeed, int rightDir, int rightSpeed) {
  digitalWrite(LEFT_DIR_PIN, leftDir);
  digitalWrite(RIGHT_DIR_PIN, rightDir);
  analogWrite(LEFT_SPEED_PIN, leftSpeed);
  analogWrite(RIGHT_SPEED_PIN, rightSpeed);
}

void setup() {
  courseTelemetryBegin();
  pinMode(LEFT_DIR_PIN, OUTPUT);
  pinMode(LEFT_SPEED_PIN, OUTPUT);
  pinMode(RIGHT_DIR_PIN, OUTPUT);
  pinMode(RIGHT_SPEED_PIN, OUTPUT);
  drive(FORWARD, 0, FORWARD, 0);

  // Без измерений робот не поедет.
  if (LEFT_WHITE == LEFT_BLACK || RIGHT_WHITE == RIGHT_BLACK) {
    while (true) {
      courseTelemetryPoll();
    }
  }
  courseWait(2000);
}

// ===== Готовое: передача наблюдений — не меняй =====
void courseTelemetryBegin() {
#if defined(ROBOT_TELEMETRY_BUILD_ID)
  using namespace robot_telemetry::wire;
  courseTelemetry.begin((1UL << Line0) | (1UL << Line1) |
                        (1UL << LineError) | (1UL << ControlOutput));
#else
  Serial.begin(9600);
#endif
}
```

```
85
86  void courseTelemetryPoll() {
87  #if defined(ROBOT_TELEMETRY_BUILD_ID)
88    courseTelemetry.update();
89  #endif
90  }
91
92  // NAN означает «ещё не вычислено», а не нулевую ошибку или поправку.
93  void courseTelemetryReport(int leftRaw, int rightRaw, float error, float outp
94  #if defined(ROBOT_TELEMETRY_BUILD_ID)
95    using namespace robot_telemetry::wire;
96    courseTelemetry.setUnsigned(Line0, leftRaw);
97    courseTelemetry.setUnsigned(Line1, rightRaw);
98    if (isnan(error)) courseTelemetry.setMissing(LineError);
99    else courseTelemetry.setFloat(LineError, error);
100   if (isnan(output)) courseTelemetry.setMissing(ControlOutput);
101   else courseTelemetry.setFloat(ControlOutput, output);
102   courseTelemetry.update();
103 #endif
104 }
105
106 void courseWait(unsigned long durationMs) {
107   unsigned long started = millis();
108   while (millis() - started < durationMs) {
109     courseTelemetryPoll();
110     delay(1);
111   }
112 }
```

3. Посчитай на бумаге: `baseSpeed = 100`, `kp = 0.5`, ошибка −40. Какие скорости получат колёса и куда повернёт робот? На подставке в следующем шаге проверишь это на колёсах.

ПРЕДСКАЖИ

Что будет делать робот, если поставить `kp = 0`? А если сделать `kp` отрицательным?

Подсказка: Какое колесо ускорять

Линия слева — ошибка отрицательная. Какое колесо должно ехать быстрее, чтобы робот повернул к линии? Согласуй знак поправки с этим.

Подсказка: Границы скорости

Мотор понимает только 0…255. Какая команда из урока не даёт числу выйти за границы?

ДОЛЖНО ПОЛУЧИТЬСЯ

`loop()` написан. Для примера: поправка −20, левое колесо 80, правое 120 — робот поворачивает налево, к линии. Прогноз про `kp = 0` записан — проверишь на подставке.

5 Проверь на подставке

7 мин

1. Поставь робота на подставку, загрузи программу, включи питание моторов.
2. Подведи линию под левый датчик, потом под правый, потом посередине. Запиши, какое колесо ускоряется и какое замедляется.
3. Проверь свой прогноз про `kp = 0`: поставь `kp = 0`, загрузи и подвигай линию. Что делают колёса? Верни свой `kp`.

Где линия	Левое колесо	Правое колесо	Совпало с прогнозом?
Под левым			
Под правым			
Посередине			

ДОЛЖНО ПОЛУЧИТЬСЯ

Колёса меняют скорость плавно. Таблица заполнена, прогноз про `kp = 0` проверен.

ЕСЛИ НЕ ПОЛУЧИЛОСЬ

Что видишь	Что сделать
Колёса не крутятся	Написан ли `loop()` с командой моторам? Вписаны ли измерения?
Ускоряется не то колесо	Проверь знак ошибки по монитору порта и формулы скоростей

6 Подбери коэффициент

18 мин

Подбери `kp` сам. Правило одно: **меняй одно число за раз**.

1. Начни с `kp = 0.5` при `baseSpeed = 100`. Затем попробуй одно явно меньшее и одно явно большее значение и сузь диапазон.
2. Запиши в журнал не меньше четырёх попыток.

`kp`	`baseSpeed`	Что видно	Что поменять дальше и почему
0.5	100		
	100		
	100		
	100		

3. С лучшим `kp` подними `baseSpeed`. Что пришлось изменить, чтобы робот снова держал дугу?

Подсказка: Как понять, куда менять kp

Робот уходит с дуги наружу — поправка слабая. Робот виляет даже на прямой — поправка сильная.

ДОЛЖНО ПОЛУЧИТЬСЯ

В журнале четыре попытки с объяснением каждого шага и пара `kp` – `baseSpeed`, на которой робот проходит дугу.

7 Задания по уровням

10 мин

★ **СРЕДНИЙ УРОВЕНЬ · ДУГА НА СКОРОСТИ**

Пройти дугу радиусом 250 мм, как на среднем поле, на скорости не ниже той, что подобрана для прямой. Если не получается — что важнее увеличить: `kp` или уменьшить скорость?

ПРОВЕРЬ СЕБЯ

- [ ] Я могу объяснить знак ошибки, когда линия слева и справа.
- [ ] Я могу посчитать скорости колёс по ошибке, kp и baseSpeed.
- [ ] Я подобрал kp по наблюдениям, меняя одно число за раз.

Проверь себя

★ Средний уровень

1. На дуге радиусом 250 мм робот на скорости съезжает наружу. Что попробовать?

- ○ Немного увеличить kp или уменьшить скорость
- ○ Уменьшить kp и увеличить скорость
- ○ Поменять местами датчики

2. Почему на дуге меньшего радиуса нужна более сильная поправка?

- ○ Линия уходит в сторону быстрее, и ошибка растёт быстрее
- ○ На дуге датчики хуже видят линию
- ○ На дуге колёса крутятся медленнее

3. Линия ушла под левый датчик, ошибка error = leftLevel - rightLevel отрицательная. Какое колесо должно ускориться, чтобы робот вернулся к линии?

- ○ Правое: робот повернёт налево, к линии
- ○ Левое: робот повернёт налево, к линии
- ○ Оба одинаково: так робот быстрее догонит линию

4. Почему в функции level результат считают в long, а не в int?

- ○ Произведение на 100 может не поместиться в int Arduino Uno
- ○ С long программа работает быстрее
- ○ analogRead возвращает только long

5. Команда поменяла сразу kp и baseSpeed, и робот стал проходить дугу. Что не так с этим опытом?

- ○ Непонятно, какое из двух изменений помогло
- ○ Всё так: чем больше изменений, тем быстрее подбор
- ○ kp нельзя менять вместе с программой

Отвечай на вопросы своего уровня. Нужно верных ответов: Начальный — 4 из 5; Средний — 4 из 5; Продвинутый — 4 из 5. Ответы проверит преподаватель или программа курса.

Итог

Робот поворачивает пропорционально ошибке — это П-регулятор. Он плавнее правила «стоп — едет» и проходит крутые дуги. Дальше робот научится находить перекрёстки на линии.

Как подтвердить результат

Для каждого опыта сохраняй условия, версию программы, наблюдение и вывод. Сравнивая попытки, меняй одну причину за раз; неудачные попытки тоже остаются в журнале. Отметка результата относится к уже выполненным действиям, повторять их ради отметки не нужно. Если результат не получен, запиши расхождение и следующую проверяемую гипотезу. Модель, измерения настоящего робота и объяснение — разные виды доказательств.

Словарь урока

ошибка	Насколько линия ушла от середины: разница показаний левого и правого датчиков.
поправка	На сколько изменить скорости колёс, чтобы вернуться к линии.
коэффициент	Число kp: во сколько раз поправка больше или меньше ошибки.
П-регулятор	Правило «поправка пропорциональна ошибке»: больше ушли — сильнее поворачиваем.
constrain	Команда, которая не даёт числу выйти за границы, например за 0…255.