Обучающий кейс «Робот почтальон»

Предпочтительное время проведения активности: 8 часов

Место занятия в структуре программы: изучение основных методов программирования и управления базовой моделью мобильного робота. Целесообразно предлагать для реализации после знакомства с платформой Arduino, базовыми электронными компонентами и основами схемотехники и изучения основ программирования в Arduino IDE.

Проблемная ситуация:

Почтальон заболел, и нужно доставить посылку в указанный дом, но послать больше некого. Поэтому на помощь приходят роботы! У вас есть базовая модель робота и поле с нарисованным путём в виде линий и перекрестков от места «Почты» до нескольких «Домов». Необходимо запрограммировать робота так, чтобы он смог самостоятельно доехать от «Почты» до выбранного «Дома».

Цель кейса: практическое закрепление обучающимися полученных знаний в электронике, программировании при работе с мобильным роботом.

Задачи кейса:

закрепить навыки сборки мобильного робота;

научить обучающихся применять навыки программирования в среде Arduino IDE для решения комплексных задач;

развивать умение работать с датчиками линии.

Задание для обучающихся: запрограммировать стандартного мобильного робота для перевозки посылки из «Почты» до выбранного «Дома», следуя нарисованному маршруту.

Планируемы результаты кейса

Образовательные: в результате выполнения кейса ученик научится решать комплексные задачи управления мобильным роботом, разделять комплексную задачу на более простые и последовательно решать их.

Продуктовые: Программный алгоритм для мобильного работа базовой конструкции для выполнения задачи автономного перемещения по заранее известным ориентирам (нарисованным линиям) от точки до точки.

Связь с учебными предметами: Информатика, физика, математика.

*В зависимости от уровня и погруженности обучающихся в тематику, схема может гибко варьироваться педагогом/наставником.

Учебный план

№ п/п	Тема занятия	Формы работы	Количество акад. часов		
			Всего	Теория	Практика
1.	Знакомство с мобильным роботом.	Практическая работа	1	0	1
2.	Разбор алгоритма движения по линии.	Практическая работа	2	1	1
3.	Разбор фиксации и подсчёта перекрестков во время движения по линии.	Практическая работа	2	1	1
4.	Программирование заданной траектории движения робота.	Практическая работа	3	0	3
	Итого		**8**	**2**	**6**

Занятие 1.

Тема: Знакомство с мобильным роботом.

Практика: Считывание значений датчиков линии в Arduino. Анализ значений с датчиков на линии и фоне. Управление моторами с платы Arduino.

Занятие 2.

Тема: Разбор алгоритма движения по линии. Тестирование на поле.

Теория: Основы движения по линии с несколькими датчиками. Релейный регулятор на двух датчиках линии.

Практика: Программирование, тестирование и настройка движения мобильного робота по линии.

Примечание: В зависимости от успеваемости обучающихся возможен дополнительный разбор других регуляторов движения по линии (например, пропорциональный).

Занятие 3.

Тема: Разбор фиксации и подсчёта перекрестков во время движения по линии.

Теория: Обработка значений датчиков на перекрёстках. Движение робота на перекрёстке: проезд, поворот. Подсчёт перекрёстков в программе.

Практика: Программирование, тестирование и настройка движения мобильного робота по линии с обработкой, проездом и подсчётом перекрёстков.

Занятие 4.

Тема: Программирование заданной траектории движения робота.

Практика: Написание программы для мобильного робота, позволяющую проехать заданную траекторию. Тестирование и отладка на поле.

Примечание: В зависимости от имеющегося поля и успеваемости обучающихся, траектория может иметь разную сложность, но должна содержать минимум один перекрёсток, на котором необходимо сделать поворот.

Необходимые материалы и оборудование

Аппаратное и техническое обеспечение:

Поле для тестирования движения мобильного робота, содержащее траекторию с контрастными линией (например, черная линия на белом фоне) и перекрёстками. Пример поля представлен на рисунке 1.

Рабочее место обучающегося:

1.Ноутбук, мышь;

2. Собранный базовый мобильный робот, включающий в себя:

- плату Arduino;
- 2-мотора постоянного тока (мотор-редуктора) с колёсами;
- драйвер для управления моторами;
- 2-х датчика линии;
- автономный источник питания (аккумулятор, батарея);
- корпус и соединительные элементы (провода, макетная плата или др.).

*Примеры подходящего мобильного робота представлены на рисунке 2.

Программное обеспечение: Arduino IDE.

Кадровое обеспечение: Педагог дополнительного образования, который компетентен в следующих областях: электроника, схемотехника, программирование микроконтроллеров Arduino, знание основных алгоритмов движения мобильного робота.

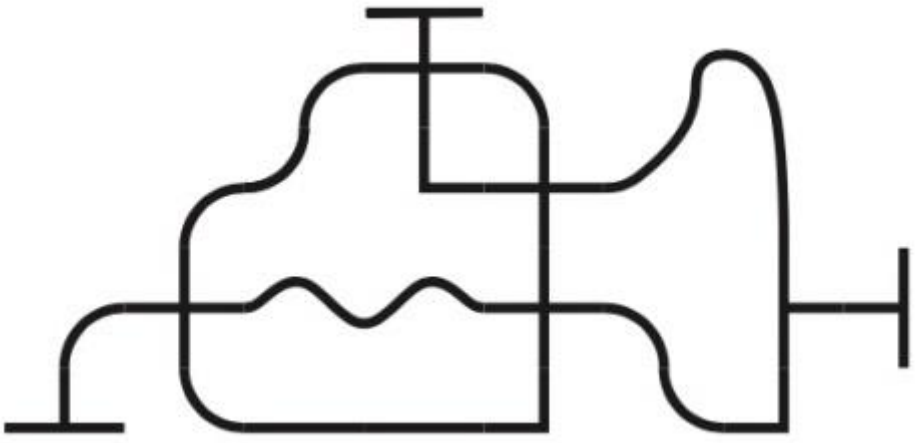
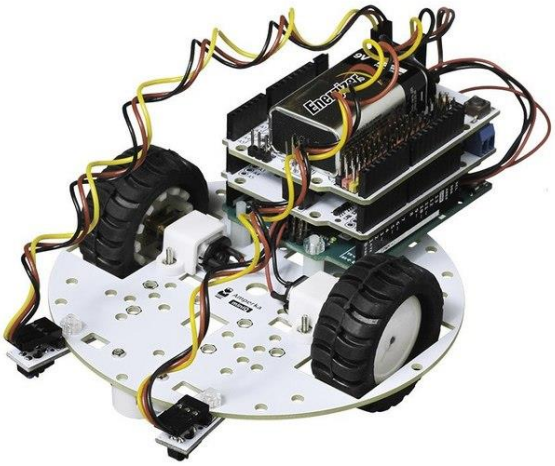


Рисунок 1. Пример поля, подходящего для данного кейса



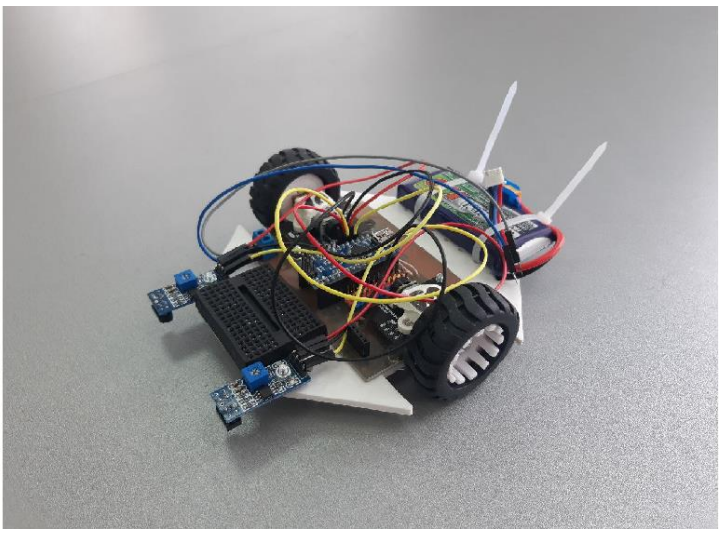


Рисунок 2. Примеры мобильных роботов, подходящих для данного кейса

Контрольное задание кейса – успешное прохождение мобильным роботом траектории движения.

Критерии оценивания:

1) Прохождение траектории во время контрольного заезда: Менее 40% - 0 баллов 40%<1 балл <70% 70%<2 балла <100% 100% - 3 балла	**2) Количество затраченных попыток:** 1 попытка = 3балла 2 попытки = 2 балла 3 попытки = 1 балл Более 3 попыток – 0 баллов. Максимальное количество попыток устанавливает наставник

Соответствие итоговой оценки уровню освоение модуля:
6-5 баллов – высокий
4-3 балла – средний
1-2 балла – низкий

Приложение 1.

Пример кода

```
// Данный пример кода написан для прохождения траектории мобильным роботом
// Основные компоненты робота, на основе которых написан данный код:
// Arduino UNO;
// - Motor Shield производства Amperka, управляющий двумя моторами постоянного тока
//
http://wiki.amperka.ru/%D0%BF%D1%80%D0%BE%D0%B4%D1%83%D0%BA%D1%82%D1%8B:arduino-motor-shield;
// - 2 аналоговых датчика линии
//
http://wiki.amperka.ru/%D0%BF%D1%80%D0%BE%D0%B4%D1%83%D0%BA%D1%82%D1%8B:analog-line-sensor;
// Правый мотор подключен к клемме: M1+ и M1-
// Левый мотор подключен к клемме: M2+ и M2-
// Motor shield использует четыре контакта 4, 5, 6, 7 для управления моторами
// 4 и 7 — для направления, 5 и 6 — для скорости

#define SPEED_R 5
#define DIR_R 4
#define SPEED_L 6
#define DIR_L 7

// Подключение аналоговых датчиков линии:
// Левый в A0, Правый в A1
// Робот будет ехать по чёрной линии на белом фоне
// Примем следующие значения, выдаваемые датчиками (необходимо проверить индивидуально для каждого робота):
// Чистое значение белого - 50
// Чистое значение чёрного - 750
// Граничное значение между белым и чёрным (серое) - 400

#define LEFT_SENSOR_PIN A0
#define RIGHT_SENSOR_PIN A1

int num_cross = 0; //переменная для подсчёта перекрёстков

void setup() {
  // Настраиваем порты управления MotorShield в режим выхода сигнала
  pinMode(SPEED_R, OUTPUT);
  pinMode(DIR_R, OUTPUT);
  pinMode(SPEED_L, OUTPUT);
  pinMode(DIR_L, OUTPUT);
}
```

```
void loop() {
  // Считываем и запоминаем значения с датчиков линии
  int lSens = analogRead(LEFT_SENSOR_PIN);
  int rSens = analogRead(RIGHT_SENSOR_PIN);

  if ((lSens < 400) && (rSens < 400)) {
    //если оба датчика видят белое (светлее серого), т.е. линия между датчиками,
    // то едем вперёд
    digitalWrite(DIR_R, 1); //Запускаем оба мотора вперёд с одинаковой скоростью
    analogWrite(SPEED_R, 100);
    digitalWrite(DIR_L, 1);
    analogWrite(SPEED_L, 100);
  }
  else if ((lSens > 400) && (rSens < 400)) {
    //иначе, если левый на чёрном (темнее серого), а правый на белом (светлее
серого), т.е. линия слева,
    // то едем вперёд с подворотом налево
    digitalWrite(DIR_R, 1); //Запускаем правый мотор быстрее вперёд
    analogWrite(SPEED_R, 120);
    digitalWrite(DIR_L, 1); //Запускаем левый мотор медленнее вперёд
    analogWrite(SPEED_L, 80);
  }
  else if ((lSens < 400) && (rSens > 400)) {
    //иначе, если левый на белом (светлее серого), а правый на чёрном (темнее
серого), т.е. линия справа,
    // то едем вперёд с подворотом направо
    digitalWrite(DIR_R, 1); //Запускаем правый мотор медленнее вперёд
    analogWrite(SPEED_R, 80);
    digitalWrite(DIR_L, 1); //Запускаем левый мотор быстрее вперёд
    analogWrite(SPEED_L, 120);
  }
  else if ((lSens > 400) && (rSens > 400)) {
    //иначе, если оба датчика видят чёрное (темнее серого), т.е. наехали на
перекрёсток,
    // то считаем данный перекрёсток
    num_cross = num_cross + 1; //и проверяем посчитанный номер

    if (num_cross == 1) {
      //если перекрёсток первый, то нужно повернуть на нём налево
      //проезжаем немного вперёд, чтобы колёсами встать ровно на перекрёсток (время
задержки подбирается экспериментально)
      digitalWrite(DIR_R, 1);
      analogWrite(SPEED_R, 100);
      digitalWrite(DIR_L, 1);
```

```
      analogWrite(SPEED_L, 100);
      delay(1000);
      //затем поворачиваем налево вокруг своей оси, пока робот не довернётся до линии продолжения (время задержки подбирается экспериментально)
      digitalWrite(DIR_R, 1);
      analogWrite(SPEED_R, 100);
      digitalWrite(DIR_L, 0);
      analogWrite(SPEED_L, 100);
      delay(1000);
    }
    else if (num_cross == 2) {
      //иначе, если перекрёсток второй, то снова нужно повернуть на нём налево
      //проезжаем немного вперёд, чтобы колёсами встать ровно на перекрёсток (время задержки подбирается экспериментально)
      digitalWrite(DIR_R, 1);
      analogWrite(SPEED_R, 100);
      digitalWrite(DIR_L, 1);
      analogWrite(SPEED_L, 100);
      delay(1000);
      //затем поворачиваем налево вокруг своей оси, пока робот не довернётся до линии продолжения (время задержки подбирается экспериментально)
      digitalWrite(DIR_R, 1);
      analogWrite(SPEED_R, 100);
      digitalWrite(DIR_L, 0);
      analogWrite(SPEED_L, 100);
      delay(1000);
    }
    else if (num_cross == 3) {
      //иначе, если перекрёсток третий (финишный), то на нём надо остановиться
      //Ставим скорости обоих моторов равные нулю и тормозим программу здесь на долгое время
      digitalWrite(DIR_R, 1);
      analogWrite(SPEED_R, 0);
      digitalWrite(DIR_L, 1);
      analogWrite(SPEED_L, 0);
      delay(10000000000);
    }
  }
}
```