СВЕТОДИОДНЫЙ МАЯЧОК ДЛЯ РОБОТА

Кулаков Владимир Геннадьевич
SPIN РИНЦ: 2111-7702
Контакт с автором: kulakovvlge@gmail.com

В данной статье применительно к игрушечным роботам рассматривается одна из классических задач робототехники: движение робота в направлении источника света, реализованного в виде светодиодного маяка, установленного на подставке. Для реализации автоматического управления роботом-тележкой используются персональный компьютер и два контроллера Arduino, а программа управления написана на языке Python. Связь компьютера с роботом осуществляется с помощью двух беспроводных модулей типа NRF24L01. Передача изображения с видеокамеры на компьютер осуществляется по технологии Wi-Fi.

Современные игрушечные роботы до сих пор лишены одной очень важной возможности – возможности использования компьютерного зрения, так как встроенные микроконтроллеры подобных роботов являются недостаточно производительными и имеют слишком маленький объем оперативной памяти для того чтобы обрабатывать поступающие с видеокамеры кадры изображения в реальном времени [1, 2, 10].

Между тем, следует отметить, что игрушки с видеокамерами давно выпускаются, но сигнал от беспроводной камеры обычно передается человеку-оператору: полученное изображение выводится на экран планшета или смартфона, и за выбор направления движения для робота отвечает оператор. Таким образом, в игрушках в настоящее время реализован режим **телеуправления**, а не автоматический режим.

Однако во второй половине прошлого века инженеры уже сталкивались с проблемой низкой производительности встроенных контроллеров промышленных роботов и успешно ее решали следующим образом: контроллер применялся только для работы с датчиками и исполнительными механизмами, а обработкой изображения с камеры и принятием решений о необходимых действиях робота и выборе маршрута его движения занимался внешний по отношению к роботу мощный компьютер. Обмен информацией между роботом и компьютером при этом производился по кабелю или по радиоканалу, а видеокамера либо устанавливалась на корпусе робота, либо размещалась на потолке помещения, по которому передвигался робот.

Упрощенная схема взаимодействия компьютера и робота показана на рисунке 1. Микроконтроллер собирает данные с датчиков и передает их по низкоскоростному радиоканалу (каналу 1) на компьютер, а с компьютера по тому же каналу принимает управляющие сигналы и отправляет их на исполнительные механизмы робота. Видеокамера использует собственный высокоскоростной радиоканал (канал 2) для передачи изображения на компьютер. Роль компьютера, управляющего действиями игрушечного робота,

может исполнять любое высокопроизводительное вычислительное устройство: настольный компьютер, ноутбук, планшет или смартфон.

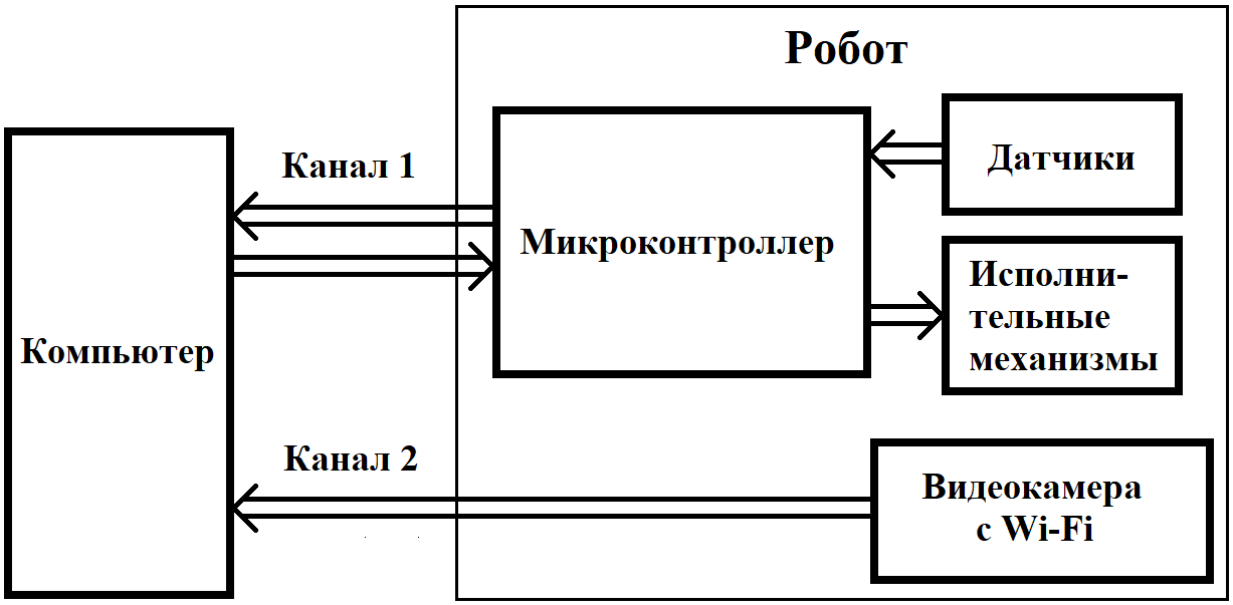

Рисунок 1. Упрощенная схема взаимодействия компьютера и робота

В качестве контроллера для самодельных роботов чаще всего используют какое-либо устройство из линейки Arduino, а в качестве приемопередатчиков для обмена информацией между роботом и компьютером применяют стандартные модули, предназначенные для Arduino. В таком случае на стороне компьютера также приходится использовать контроллер Arduino, который обеспечивает взаимодействие между компьютером и беспроводным приемопередатчиком.

Структурная схема канала управления роботом, в котором реализуется подобный технический прием, показана на рисунке 2. В качестве приемопередатчиков в данном примере используются модули **NRF24L01** [13], а для управления электродвигателями робота-тележки применяется **ЛМ2-130** [4], драйвер моторов для робототехники, построенный на базе микросхемы L293D.

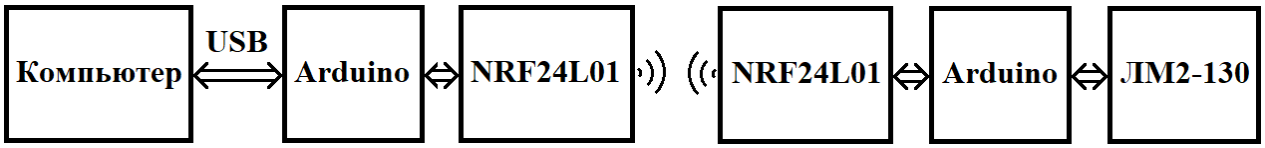

Рисунок 2. Структурная схема канала управления роботом

Принципиальная схема приемопередатчика, подключаемого к компьютеру, показана на рисунке 3. В качестве контроллера в данной схеме

можно использовать Arduino Nano или Arduino Uno. Электропитание и управляющие сигналы приемопередатчик получает от компьютера по шине USB.

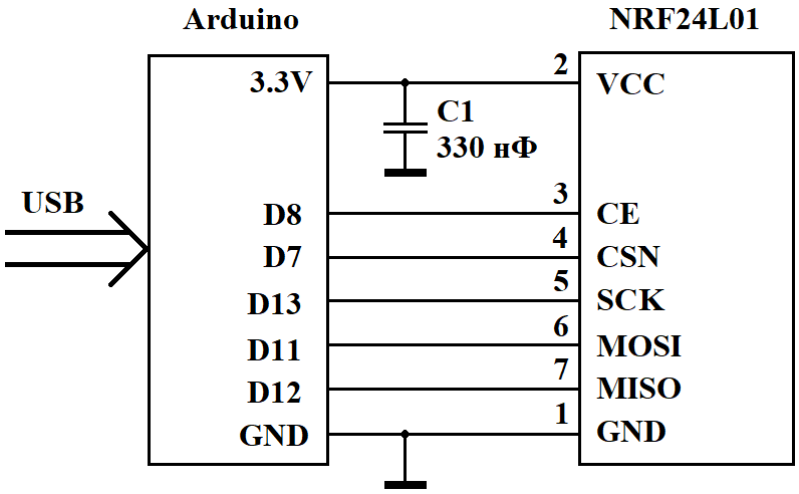

Рисунок 3. Принципиальная схема приемопередатчика, подключаемого к компьютеру

В контроллер Arduino, установленный на стороне компьютера, нужно загрузить управляющую программу (скетч), приведенную в листинге 1.

Листинг 1. Управляющая программа для контроллера Arduino, установленного на стороне компьютера

```
// Подключаем библиотеки программ
#include <SPI.h>
#include "Mirf.h"
#include "nRF24L01.h"
#include "MirfHardwareSpiDriver.h"
// Задаем размер пакета данных в байтах
#define MAX_BUFF 1
// Однобайтный массив, содержащий код управления моторами:
byte mot[1];

void setup(){
  // Настраиваем скорость работы последовательного порта
  Serial.begin(9600);
  // Задаем параметры работы беспроводного передатчика
  Mirf.spi = &MirfHardwareSpi;
  Mirf.init();
  Mirf.setRADDR((byte *)"master"); // Адрес отправителя
  Mirf.setTADDR((byte *)"robot");  // Адрес получателя
  Mirf.payload = MAX_BUFF;         // Размер буфера данных
  Mirf.channel = 10;               // Канал приёма-передачи
  Mirf.config();
}
```

```
void loop(){
  // Если в буфере последовательного порта есть команда,
  // то считываем ее
  if (Serial.available() > 0)
  {
    mot[0] = Serial.read();
    // Передаем команду роботу-тележке
    Mirf.send((uint8_t *)mot);
    // Цикл ожидания завершения передачи
    while(Mirf.isSending()){};
  }
}
```

Принципиальная схема устройства, предназначенного для управления роботом, приведена на рисунке 4.

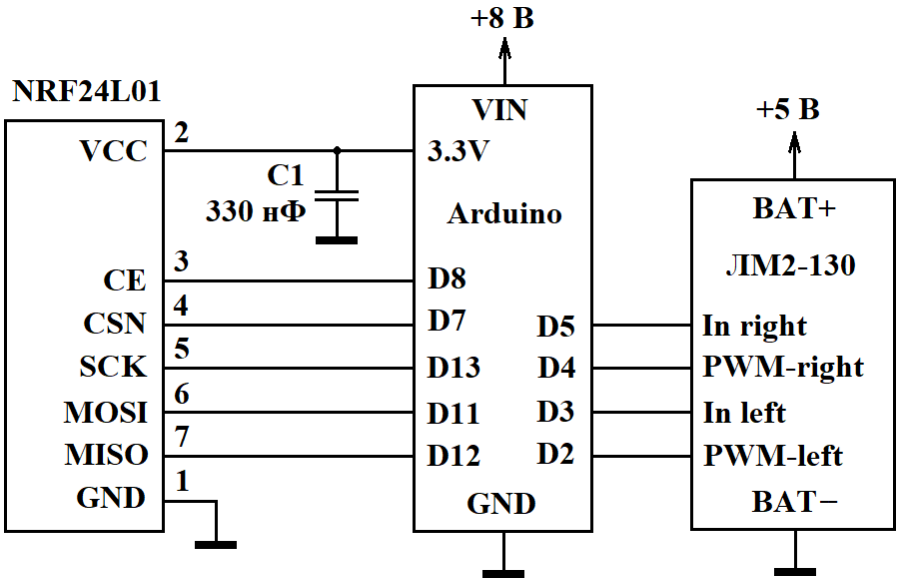

Рисунок 4. Принципиальная схема устройства управления роботом-тележкой

Робот-тележка построен на основе 4-х моторного шасси конструктора **ЛАРТ** [11]. Драйвер ЛМ2-130, используемый для управления моторами робота, имеет следующие входы:

- In right – управление направлением вращения правых электромоторов,
- PWM-right – управление скоростью вращения правых электромоторов,
- In left – управление направлением вращения левых электромоторов,
- PWM-left – управление скоростью вращения левых электромоторов.

Если не использовать широтно-импульсную модуляцию (PWM), а только включать либо отключать моторы, то процесс управления тележкой существенно упрощается, а размер команды уменьшается до одного байта.

Программа для контроллера Arduino, управляющего работой моторов робота, приведена в листинге 2.

Листинг 2. Программа для контроллера Arduino, управляющего моторами робота

```
#include <SPI.h>
#include "Mirf.h"
#include "nRF24L01.h"
#include "MirfHardwareSpiDriver.h"
// Задаем размер пакета данных в байтах
#define MAX_BUFF 1
// Глобальные переменные:
// Текущее системное время в мс
unsigned long currenttime;
// Время поступления последней команды в мс
unsigned long lastrectime;
// Массив команд (однобайтный)
byte mot[1];
// Инициализируем контроллер
void setup(){
  // Указываем выходы, используемые управления двигателями
  pinMode(2, OUTPUT); pinMode(3, OUTPUT);
  pinMode(4, OUTPUT); pinMode(5, OUTPUT);
  // Инициализируем приемник
  Mirf.spi = &MirfHardwareSpi;
  Mirf.init();
  Mirf.setRADDR((byte *)"robot");  // Адрес получателя
  Mirf.setTADDR((byte *)"master"); // Адрес отправителя
  Mirf.payload = MAX_BUFF;         // Объем буфера
  Mirf.channel = 10;               // Номер радиоканала
  Mirf.config();
  // Записываем текущее системное время
  lastrectime = millis();
  // Заносим в массив команду отключения моторов
  mot[0] = 0;
}

// Рабочий цикл контроллера
void loop(){
  // Прием команды
  if(Mirf.dataReady())
  {
    Mirf.getData((uint8_t *)mot); // Читаем данные
    lastrectime = millis();
  }
  else
  {
    currenttime = millis();
    // Останов, если новые команды не поступали 300 мс
    if(currenttime >= lastrectime + 300l) mot[0] = 0;
  }
```

```
  // Анализируем текущую команду:
  // Остановиться
  if(mot[0] == 0)
  {
    digitalWrite(2, LOW); digitalWrite(3, LOW);
    digitalWrite(4, LOW); digitalWrite(5, LOW);
  }
  // Двигаться вперед
  if(mot[0] == 1)
  {
    digitalWrite(2, HIGH); digitalWrite(3, LOW);
    digitalWrite(4, HIGH); digitalWrite(5, LOW);
  }
  // Двигаться назад
  if(mot[0] == 2)
  {
    digitalWrite(2, HIGH); digitalWrite(3, HIGH);
    digitalWrite(4, HIGH); digitalWrite(5, HIGH);
  }
  // Повернуть налево
  if(mot[0] == 3)
  {
    digitalWrite(2, HIGH); digitalWrite(3, HIGH);
    digitalWrite(4, HIGH); digitalWrite(5, LOW);
    // Прекратить поворот через 25 мс
    delay(25); mot[0] = 0;
  }
  // Повернуть направо
  if(mot[0] == 4)
  {
    digitalWrite(2, HIGH); digitalWrite(3, LOW);
    digitalWrite(4, HIGH); digitalWrite(5, HIGH);
    // Прекратить поворот через 25 мс
    delay(25); mot[0] = 0;
  }
}
```

В приведенном примере управляющей программы реализованы следующие команды:

- 0 – прекратить движение;
- 1 – двигаться вперед;
- 2 – двигаться назад;
- 3 – повернуть налево;
- 4 – повернуть направо.

Команды поворота отрабатываются роботом следующим образом: поворот осуществляется на месте и на небольшой угол (несколько градусов).

При использовании внешнего управляющего компьютера необходимо иметь какой-то защитный механизм на случай **потери связи** между роботом и

компьютером, например, из-за выхода робота за пределы зоны надежной связи или внезапного отключения компьютера. Необходимо также учитывать особенности протоколов, по которым осуществляется беспроводная передача [9]: они не обеспечивают гарантированную доставку пакетов данных от отправителя к получателю и могут создавать существенную с точки зрения робота задержку при передаче данных.

Обычно для решения подобных проблем применяется тот же способ, который когда-то использовался в Луноходах: если в течение заданного небольшого периода времени (в приведенном выше примере программы – **300 мс**) не поступает новый управляющий сигнал, то робот замирает на месте.

Принципиальная схема системы питания робота показана на рисунке 5. Для обеспечения питания используются два включенных последовательно аккумулятора типа Li18650 емкостью 2600 мА/час с номинальным выходным напряжением 3,7 В (реальное напряжение обычно несколько выше номинального, поэтому напряжение на выходе батареи из двух аккумуляторов составляет около 8 В). Питание от аккумуляторов получают контроллер Arduino, драйвер двигателей и видеокамера, причем на драйвер и камеру подается стабилизированное напряжение, сниженное до 5 В.

Микросхемы стабилизаторов питания 7805 должны быть снабжены радиаторами воздушного охлаждения площадью не менее 5см^2, так как через них будет протекать сильный ток: видеокамера потребляет ток около 300 мА, а четыре одновременно работающих двигателя – до 600 мА.

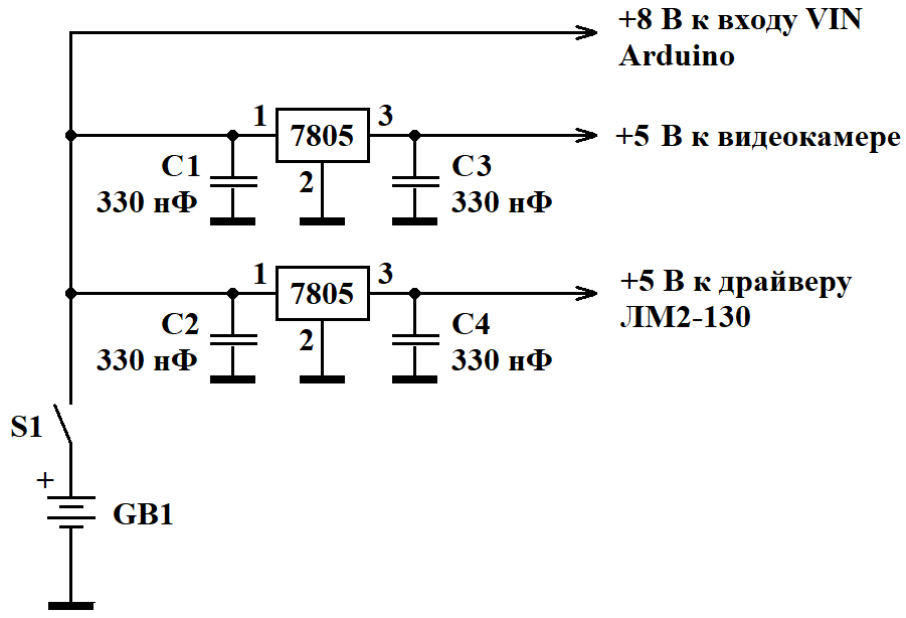

Рисунок 5. Принципиальная схема системы питания робота

Рассмотрим теперь работу с видеокамерой, установленной на тележке. Как видно из приведенных выше рисунков, камера работает совершенно автономно, передавая изображение компьютеру напрямую, а от тележки

получат только электропитание. В данном примере используется камера **Ai-Ball** [5], специально приспособленная для работы с малогабаритными движущимися роботами: она имеет пониженное энергопотребление, напряжение питания +5 В, небольшие размеры и вес.

Задача ориентации в пространстве при помощи источника яркого света является для робототехники классической, однако следует отметить, что робот должен иметь возможность быстро определить не только направление на маяк, но и расстояние до него (по величине угла α между оптической осью видеокамеры и линией, направленной от камеры на светодиод). Следовательно, если оптическая ось камеры расположена горизонтально (строго параллельно полу), то светодиодный маячок нужно расположить либо ниже этой оси, например, на полу, либо выше – на специальной подставке.

Второй вариант, изображенный на рисунке 6, является предпочтительным, так как в этом случае меньше вероятность того, что какое-либо препятствие перекроет световой сигнал, идущий от маяка. Кроме того, роботу не будут мешать ориентироваться световые блики на поверхности пола и находящихся на нем различных предметов.

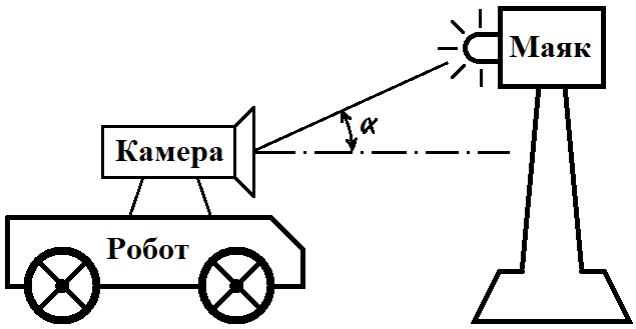

Рисунок 6. Использование светодиодного маяка для обеспечения ориентации робота в окружающем пространстве

Освещенность пола и стен помещения, а также поверхностей всех находящихся в помещении предметов должна быть умеренной, чтобы светодиод очевидно выделялся на их фоне своей яркостью. Желательно также, чтобы поверхности пола, стен и предметов были матовыми.

Вообще говоря, для ориентации в пространстве по методу триангуляции необходимо использовать два маяка, но для начала неплохо было бы научить робота самой простой операции: движению на маяк.

Допустим, что в качестве маяка используется **яркий** диод с **красным** цветом свечения.

Программа на языке **Python** [3, 7, 8], предназначенная для решения данной задачи, приведена в листинге 3. Эта программа работает под управлением операционной системы Windows. Она определяет точку в верхней части полученного с видеокамеры кадра, имеющую самое большое значение

компоненты, соответствующей красному цвету, а затем заставляет робота-тележку двигаться в направлении данной точки.

Листинг 3. Программа для управления движением робота

```
mport serial
import cv2

# Ищем самую яркую красную точку в верхней
# половине кадра размером 320x200 точек
def posit(fr):
    yp = 0
    xp = 0
    bri = 0
    for row in range(0,120,1):
        for col in range(0,320,1):
            b = fr[row,col,0];
            g = fr[row,col,1];
            r = fr[row,col,2];
            if (r>bri) and (r>b) and (r>g):
                yp = row
                xp = col
                bri = r
            if bri == 255: break
        if bri == 255: break
    return yp,xp,bri

# Основная программа
arduino = serial.Serial(port='COM3',baudrate=9600)
cap = cv2.VideoCapture('http://192.168.2.1/?action=stream')
while(True):
    # Поочередный захват кадров видеопотока
    ret, frame = cap.read()
    # Ищем самую яркую красную точку
    y, x, br = posit(frame)
    # Сформируем и передадим команду роботу-тележке
    if br<200:  arduino.write([0]) #останов
    elif y>115: arduino.write([0]) #останов
    elif x<150: arduino.write([3]) #налево
    elif x>170: arduino.write([4]) #направо
    elif y<10:  arduino.write([0]) #останов
    else:       arduino.write([1]) #вперед
```

В качестве значения параметра port в третьей строке программы нужно указать наименование виртуального последовательного порта, к которому подключен контроллер Arduino с беспроводным передающим модулем (в приведенном примере использовался порт COM3).

Для получения доступа к потоку данных в данном примере (в четвертой строке программы) указан IP-адрес, заданный для камеры Ai-Ball на заводе-изготовителе.

Здесь следует также отметить, что в управляющей программе необходимо создать некоторую зону нечувствительности, чтобы робот не вибрировал вблизи положения равновесия, бесконечно поворачиваясь рывками то налево, то направо.

Кроме того, в программе задан также останов двигателей робота вблизи маяка, чтобы тележка не наезжала на него: в качестве признака останова используется смещение самой яркой точки в верхние строки кадра.

Захват текущего видеокадра выполняется при помощи библиотеки **OpenCV** [12, 14, 15]. Полученная информация представлена в виде трехмерного массива (строки, столбцы, цветовые компоненты), содержащего изображение в формате **BGR** (Blue, Green, Red), но в программе анализируется только компонента с индексом 2, соответствующая красному цвету (счет индексов в языке Python ведется с нуля).

Видеокамера становится доступной для подключения по Wi-Fi через 10 с после включения питания робота-тележки. После того, как подключение произведено, необходимо в любом браузере открыть страничку настройки параметров видеопотока, по умолчанию (по заводским настройкам) имеющую IP-адрес **192.168.2.1**, и **уменьшить** используемое **разрешение** до значения **320×240** точек. Следует отметить, что максимальное разрешение камеры Ai-Ball достигает 640×480 точек, но размер распакованного кадра изображения в этом случае составляет 640×480×3 = 921600 байт (почти мегабайт) данных, а передача осуществляется камерой со скоростью 30 кадров в секунду. Управляющий компьютер же должен успевать просматривать в поисках самой яркой красной точки всю верхнюю половину кадра и формировать команду для тележки также 30 раз в секунду. Поэтому используемое разрешение и было уменьшено, благодаря чему объем данных, соответствующий одному кадру снижается в 4 раза: 320×240×3 = 230400 байт.

Далее нужно **включить** светодиодный **маяк** и убедиться, что он виден в окошке, отображающем поступающие с камеры кадры на экране компьютерного монитора. Желательно также **повернуть корпус** робота таким образом, чтобы изображение светодиода было смещено к левому или правому краю окошка. Если яркость изображения слишком велика (то есть центр светодиода выглядит белым, а не красным), ее требуется **уменьшить**, а затем закрыть окно настройки параметров видеопотока.

После этого нужно **подключить** к компьютеру **приемопередатчик** с контроллером Arduino, а затем **запустить** на выполнение **программу** на языке Python, приведенную в листинге 3: робот должен сориентироваться на маяк и начать **двигаться к нему**, **корректируя траекторию** своего движения по мере необходимости, а затем **остановиться** возле маяка.

По окончании эксперимента нужно **закрыть среду Python** и **выключить питание** робота.

Список использованной литературы

1. Белов А. В. ARDUINO: от азов программирования до создания практических устройств. – СПб.: Наука и техника, 2018. – 480 с.: ил.
2. Блум Д. Изучаем Arduino: инструменты и методы технического волшебства: Пер. с англ. – СПб.: БХВ-Петербург, 2018. – 336 с.: ил.
3. Гуриков С.Р. Основы алгоритмизации и программирования на Python: учебное пособие / С.Р. Гуриков. – М: ФОРУМ: ИНФРА-М, 2020. – 443 с.
4. Драйвер электромотора ЛМ2-130 – М: Ларт плюс, 2015. – 2 с.
5. Инструкция по эксплуатации Wi-Fi мини видеокамеры Ai-Ball – TREK 2000 International LTD, 2018. – 10 с.
6. Кулаков В.Г. Компьютерное зрение для игрушечных роботов. [Электронный ресурс]. URL: http://new-idea.kulichki.net/pubfiles/200325125933.pdf (дата обращения: 25.03.2020).
7. Мэтиз Э. Изучаем Python: программирование игр, визуализация данных, веб-приложения. 3-е изд. – СПб.: Питер, 2020. – 512 с.: ил.
8. Мюллер Д.П. Python для чайников, 2-е изд. : Пер. с англ. - СПб. : ООО "Диалектика", 2019. – 416 с.: ил.
9. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 4-е изд. – СПб. : Питер, 2010. – 944 с. : ил.
10. Петин В. А. Проекты с использованием контроллера Arduino. – 2-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2016. – 464 с.: ил.
11. Шасси 4-х моторное II с доп. плитой – М: Ларт плюс, 2018. – 2 с.
12. Alberto Fernández Villán. Mastering OpenCV 4 with Python. A practical guide covering topics from image processing, augmented reality to deep learning with OpenCV 4 and Python 3.7. – Packt Publishing Ltd. 2019. – 691 с.
13. Single chip 2.4 GHz Transceiver nRF24L01 – Nordic Semiconductor, 2006. – 43 с.
14. Widodo Budiharto. Modern Robotics with OpenCV. – Science Publishing Group, 2014. – 232 с.
15. Özkaya Ö, Yıllıkçı G. Arduino Computer Vision Programming. – Packt Publishing Ltd. 2015. – 194 с.

© В.Г. Кулаков, 2021