УДК 519.7

АЛГОРИТМ ПРОГРАММНОГО УПРАВЛЕНИЯ РОБОТА С ОБХОДОМ ПРЕПЯТСТВИЙ

Высоцкий Е.В., Лозовенко И.С.

МФ МГТУ им. Н.Э. Баумана

E-mail: overlord0794@yandex.ru

Представляется алгоритм программного движения робота на плоскости из точки А в точку В при наличии между ними заранее известных и, возможно, непредвиденных препятствий. Авторами разработаны алгоритмы коррекции курса, обхода препятствий и метод вычисления нового азимута. Так же представлена блок-схема программы робота.

Ключевые слова: робот, коррекция курса, прокладка курса, обход препятствий.

ALGORITHM FOR PROGRAM CONTROL OF A ROBOT WITH AN OBSTACLE BYPASS

Vysotsky E.V., Lozovenko I.S.

An algorithm for the programmed motion of a robot on a plane from point A to point B is presented, with pre-existing and possibly unforeseen obstacles between them. The authors presented algorithms for course correction, obstacle clearance and the method for calculating a new azimuth. A block diagram of the robot program is also presented.

Keywords: robot, course correction, course laying, avoiding obstacles.

Роботы перемещаются в автоматическом режиме. Обычно в современных роботах для этой цели используется навигационная система, которая определяет собственные координаты робота, планирует траекторию в текущий момент времени и управляет его движением. Так как реальная среда, в которой находится робот, обычно содержит подвижные препятствия, движение в ней по предварительно высчитанной траектории практически невозможно [1].

Постановка задачи.

В данной статье рассматривается способ автоматического управления движением робота к заданной точке при наличии непредвиденных и статических объектов (препятствий) при условии потери управляющего сигнала от оператора при телеоператорном управлении.

Принятые ограничения:

1) Внешнее управление отсутствует;
2) Поиск путей обхода препятствий с использованием трёх ультразвуковых датчиков;

Описание робота.

Основой системы управления данного робота является контроллер Arduino Mega 2560.[5] Робот приводится в движение электродвигателями постоянного тока, управление которыми осуществляется микроконтроллером с помощью микросхемы L298N DC, выполняющей функцию усилителя мощности.[4] Повороты робота на месте производятся разнонаправленным вращением колёс по левому и правому борту. Повороты в движении осуществляются заданием разной скорости вращения колёс по левому и правому борту.

Во время движения робота пространство перед роботом проверяется на наличие препятствий с помощью трёх ультразвуковых датчиков HC-SR04.[3] При обнаружении препятствия микроконтроллер производит коррекцию курса, используя для этого гироскопический датчик MPU6050.[2]

Алгоритм работы робота.

Блок-схема программы робота представлена на рисунке 1.

Решение задачи прокладки курса и его коррекции.

Алгоритм коррекции курса.

Угол отклонения — разность между курсом робота и пеленгом конечной точки маршрута в любой момент времени от старта и до достижения конечной точки.

Угол отклонения α. Он может изменяться от 0 до ±180°.

Для управления электродвигателями с целью повышения к.п.д. привода используется широтно-импульсная модуляция управляющего сигнала (ШИМ). Микроконтроллер Arduino Mega 2560 формирует ШИМ дискретно, всего 256 значений — от 0 до 255.

Средний ток через обмотку двигателя и, следовательно, его крутящий момент определяется величиной напряжения на обмотке двигателя и длительностью импульсов. Напряжение на обмотке будем считать величиной постоянной, а управляющее воздействие на двигатель будем выражать через ширину импульсов ШИМ. Для удобства объяснения алгоритма ширину импульса ШИМ будем выражать не в единицах времени, а числом от 0 до 255.

Нам нужно получить отклонение по скорости, а не саму скорость, то есть значение от 0 до 50. В итоге, у нас есть отклонение по углу, которое может принимать значения от 0 до 180 и отклонение по скорости моторов, оно лежит в промежутке от 0 до 50. Идея заключается в том, чтобы привести число из одного диапазона к соответствующему ему числу из другого диапазона. То есть, если α = 90°, то ему будет соответствовать число, равное 25-ти единицам скорости моторов. Но это слишком маленькое значение для такого большого отклонения, поэтому в дальнейшем оно умножается на специальный коэффициент (коэффициент отклонения). Он находиться

экспериментально в зависимости от количества и мощности моторов. Для приведения одного числа к масштабу другого используется функция

$$m(x, a, b, a_{нов.}, b_{нов.}) = \frac{(x - a) * (b_{нов.} - a_{нов.})}{b - a} + a_{нов.}$$

где a – нижняя граница числа,

b – верхняя граница числа,

$a_{нов.}$ – новая нижняя граница числа,

$b_{нов.}$ – новая верхняя граница числа.

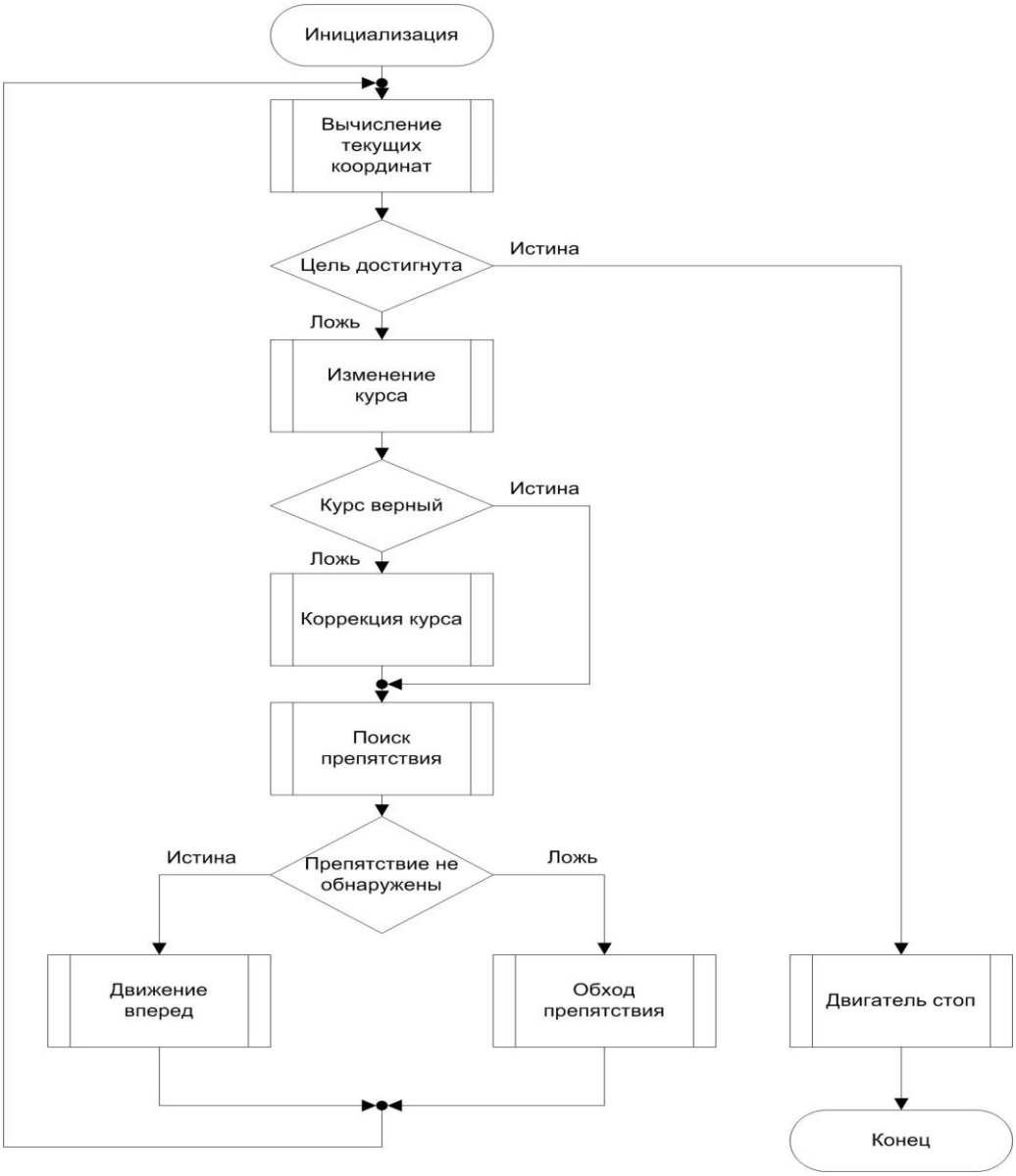

Рисунок 1 – Блок-схема программы робота

Так как скорость мотора – это целое значение, то найденное значение нужно округлить, для этого будет использоваться обозначение:

$$[x]$$

В итоге получаем формулу:

$$\Delta v = [k * m([|\alpha|], 0{,}180{,}0, v_{max} - v_{min})],$$

где Δv –отклонение по скорости; k – коэффициент отклонения; α – угол отклонения (курсовой угол); v_{max}– максимально допустимое значение скорости; v_{min} – минимально допустимое значение скорости.

Для получения конечного результата, а именно значений скорости для каждого из моторов, к значению базовой скорости правого мотора прибавляется отклонение по скорости, взятое со знаком угла отклонения, а в случае левого мотора, наоборот, отнимается тоже самое значение. Знак угла отклонения от курса находится через математическую функцию

$$\operatorname{sgn} x = \begin{cases} 1, x > 0, \\ 0, x = 0, \\ -1, x < 0 \end{cases}$$

Она возвращает 1 или -1 соответственно знаку указанного числа. После чего, найденные скорости ограничиваются минимальным и максимальным значением. Для этого используется функция

$$c(x, a, b) = \begin{cases} a, x < a, \\ x, a \le x \le b, \\ b, x > b \end{cases}$$

где a – нижняя граница числа, b – верхняя граница числа.

Она заменяет число его соответствующим граничным значением, при условии, что оно выходит за указанные границы.

В результате получаем формулы скоростей моторов:

$$v_1 = c(v_{base} - (\operatorname{sgn} \alpha) * \Delta v, v_{min}, v_{max}),$$

$$v_2 = c(v_{base} + (\operatorname{sgn} \alpha) * \Delta v, v_{min}, v_{max}),$$

где v_1- скорость левого мотора; v_2- скорость правого мотора; v_{base}- базовая скорость моторов.

Часть исходного кода программы, где вычисляются описанные выше функции:

```
int speedRatio = lround(SPEED_COEFFICIENT * map(lround(abs(courseAngle * RADIANS_TO_DEGREES)), 0, 180, 0, BASE_MOTOR_SPEED_MAX - BASE_MOTOR_SPEED_MIN));
```

uint8_t speed1 = constrain(BASE_MOTOR_SPEED - sign(courseAngle) * speedRatio, BASE_MOTOR_SPEED_MIN, BASE_MOTOR_SPEED_MAX);

uint8_t speed2 = constrain(BASE_MOTOR_SPEED + sign(courseAngle) * speedRatio, BASE_MOTOR_SPEED_MIN, BASE_MOTOR_SPEED_MAX);

Метод коррекции курса: вычисление нового азимута.

В данном методе сначала находится гипотенуза между координатой x робота и расстоянием по оси OY от текущей позиции до точки назначения. Найденное значение считается новым оставшимся расстоянием:

$$s_{ост.\ нов.} = \sqrt{x^2 + (s_{ост.} - y)^2}$$

После чего находится новый азимут, путем прибавления к его старому значению значения угла между найденной гипотенузой и старым направлением на конечную точку (в дальнейшем $\Delta\theta$).

$$\theta_{нов.} = \theta + \Delta\theta$$

Причем есть два граничных значения для оставшегося расстояния. Если оставшееся расстояние больше первого значения, то $\Delta\theta$ вычисляется так, как описано выше:

$$\Delta\theta = \arcsin\frac{x}{s_{ост.}}$$

Если оставшееся расстояние находится между первым и вторым значениями, то $\Delta\theta$ делится на 2, обеспечивая более плавное завершение маршрута.

$$\Delta\theta = \frac{\arcsin\frac{x}{s_{ост.}}}{2}$$

Если же оставшееся расстояние меньше второго значения, то переход к новому маршруту не осуществляется, что предотвращает возможное «кружение» робота вокруг конечной точки.

$$\Delta\theta = 0$$

Также, в первых двух случаях обнуляются координаты, так как движение теперь происходит относительно нового азимута.

Алгоритм обхода препятствий.

На роботе установлено три ультразвуковых сенсора, направленных вперёд по ходу движения: один посередине, направленный вдоль продольной оси робота и два по бокам, направленные под углами –45 и +45 градусов по отношению к продольной оси робота.

Режимы работы робота:

показания одного или нескольких датчиков говорят о том, что препятствие слишком близко – двигаемся назад;

если оба боковых датчика говорят о том, что препятствие довольно близко, а передний датчик вообще не видит препятствие или оно еще далеко, то это говорит о том, впереди узкий туннель – поворачиваемся в ту сторону, где больше расстояние;

передний датчик видит, что препятствие довольно близко - поворачиваемся в ту сторону, где больше расстояние;

средний датчик видит препятствие, но оно еще довольно далеко – замедляем движение;

показания всех датчиков больше граничных расстояний – двигаемся вперед (препятствия не обнаружены).

режим проезда ворот, когда два боковых датчика обнаруживают препятствие «довольно близко», а центральный датчик или совсем не находит препятствие, либо расстояние до препятствия «довольно большое». Реализация данного режима представляет собой отдельную тему для исследования и выходит за рамки данной статьи.

Движение вперед.

1) скорость движения робота постоянна;
2) гироскоп является следящей системой за направлением робота при его движении

Поворот.

В этом режиме робот осуществляет поворот в сторону противоположную курсовому углу, пока не будет достигнуто допустимое отклонение по курсовому углу. Переход в состояние поворота осуществляется при превышении курсовым углом его допустимого значения, то есть, когда угол становится слишком большим. Это состояние используется для быстрого возвращения курса робота к заданному азимуту (например, при завершении обхода препятствия или при воздействии извне, повлекшим за собой сильное отклонение от заданного направления). Алгоритм корректировки в этом случае будет малоэффективен из-за длительного времени возвращения к курсу.

Дистанция пройдена.

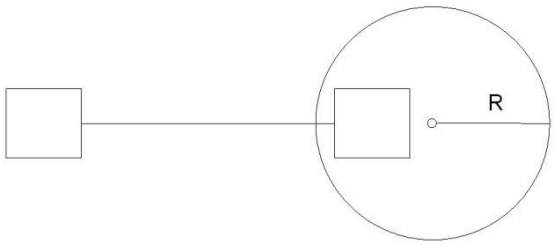

Рисунок 2 – Пример прохождения дистанции

Это состояние, говорящее о том, что робот прошёл по заданному маршруту и достиг точки назначения. Дистанция считается достигнутой, когда координата y стала больше или равна оставшейся дистанции.

Примеры обхода препятствий.

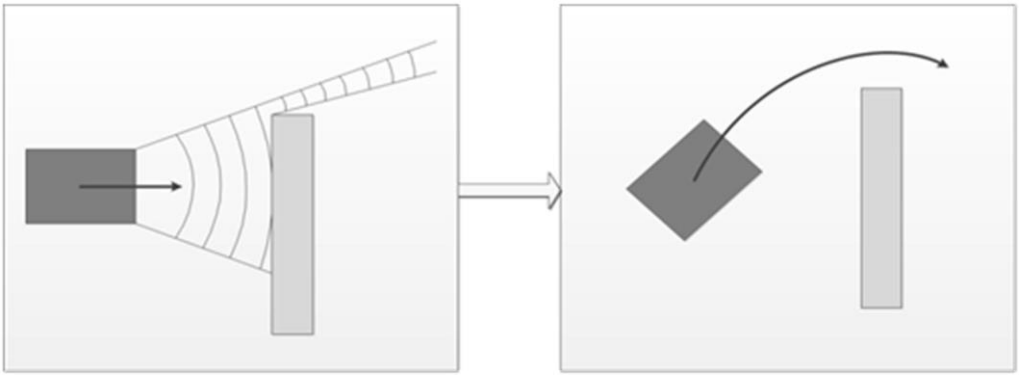

Рисунок 3 – Обход сплошного препятствия

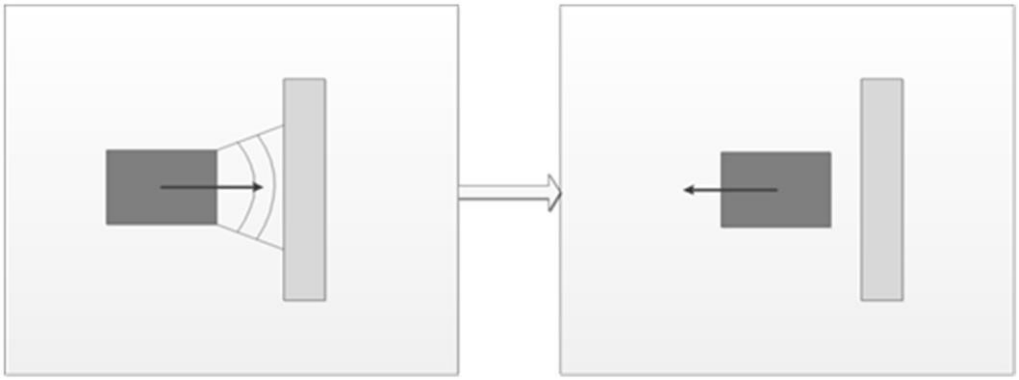

Рисунок 4 – Случай с близко расположенным препятствием

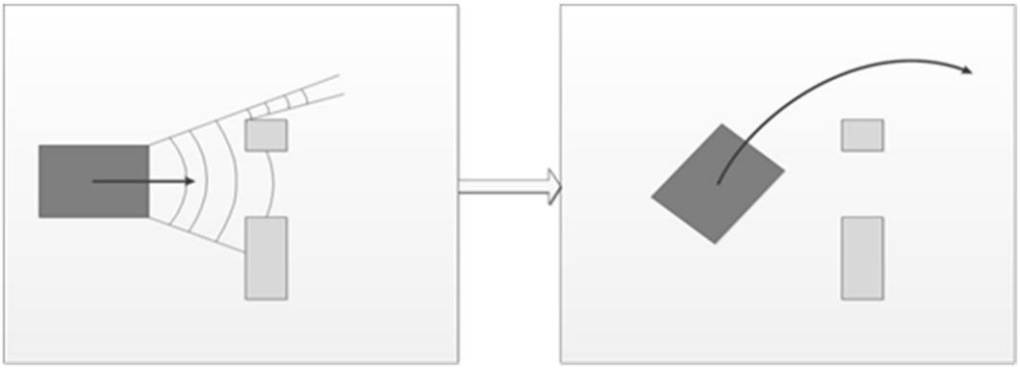

Рисунок 5 – обход узкого коридора

Вывод.

Данный способ решения задачи доказал свою работоспособность на экспериментальной модели робота.

Список литературы

1. В.Н. Герасимов, Б.Б. Михайлов Решение задачи управления движением мобильного робота при наличии динамических препятствий // ISSN 0236-3933. Вестник МГТУ им. Н.Э. Баумана. Сер. “Приборостроение”. 2012. С. 83-92

2. https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf – техническая спецификация на MPU6050

3. http://www.micropik.com/PDF/HCSR04.pdf – техническая спецификация на HC-SR04

4. https://github.com/adafruit/Adafruit-Motor-Shield-library– библиотека для работы с моторами

5. http://arduino.ru/Hardware/ArduinoBoardMega2560 – техническая спецификация на Arduino Mega 2560