УДК 004.056
DOI: 10.25686/978-5-8158-2474-4-2025-693-703

Автоматизация движения мобильного робота за стеной Turtlebro с помощью ПИД-регулятора ROS

Зеар Аунг[1], Шидловский Станислав Викторович[2], Ну Ну Вар[3]
[1,2]Томский государственный университет, Томск, Россия, zayaraung53@gmail.com,shidlovskiysv@mail.ru
[3]Университет корпоративного управления, Сагаинг (UCMS)
Сагаинг, Мьянма, dr.nuwar@gmail.com

Аннотация. В статье представлена реализация траектории алгоритмы отслеживания и обхода препятствий в Turtlebro3 Burger, использующие операционную систему ROS 2. Для отслеживания лемнискатных и круговых траекторий используется пропорциональное управление с компенсацией. Кроме того, для обхода препятствий в динамичных условиях используется алгоритм потенциального искусственного поля (APF). При таком подходе робот моделируется как частица, погруженная в потенциальное поле, которая использует притягивающие функции для достижения пункта назначения и отталкивающие функции для обхода препятствий. Для проверки работоспособности алгоритмов были проведены имитационные тесты в программном обеспечении Gazebo и экспериментальные испытания. Результаты показали, что пройденный маршрут соответствует требованиям задачи, подтверждая эффективность предложенного подхода к планированию маршрута.

Ключевые слова: ROS 2, искусственное потенциальное поле, Turtlebro, планирование пути, объезд препятствий.

Automated wall following of Turtlebro using PID control ROS

Zayar Aung[1], Stanislav Viktorovich Shidlovsky[2], Nu Nu War[3]
[1,2]Tomsk State University, Tomsk, Russian Federation, zayaraung53@gmail.com,shidlovskiysv@mail.ru
[3]University of Co-operative and Management of Sagaing (UCMS), Sagaing, Myanmar, dr.nuwar@gmail.com

Abstract. The paper presents the implementation of trajectory tracking and obstacle avoidance algorithms on a Turtlebro Burger using the ROS 2 operating system. A proportional control with compensation is used for lemniscate and circle trajectory tracking. In addition, a potential artificial field (APF) algorithm is incorporated for obstacle avoidance in dynamic environments. In this approach, the robot is modeled as a particle immersed in a potential field that uses attractive functions to reach a destination point and repulsive functions to go around obstacles. Simulated tests in Gazebo software and experimental tests are performed to validate the performance of the algorithms. The results showed that the route followed meets the requirements of the problem, confirming that the proposed route planning approach is efficient.

Keywords: ROS 2, Artificial potential field, Turtlebro, path planning, obstacle avoidance.

Введение

Одной из главных проблем мобильной робототехники является автономная навигация, при которой робот должен двигаться к цели и одновременно избегать столкновений с препятствиями окружающей среды [1]. Эта задача известна как планирование и отслеживание пути, она учитывает распределение препятствий, присутствующих в окружающей среде статически и динамически, чтобы рассчитать непрерывный маршрут и достичь целевой точки из исходного положения с эффективным и быстрым обходом препятствий [2].

Алгоритмы планирования траектории в мобильных роботах должны соответствовать требованиям безопасной маршрутизации, скорости вычислений и надежности. Алгоритм искусственного потенциального поля (APF) – это эвристический подход, который обеспечивает простое и эффективное планирование траектории, что может гарантировать обработку в режиме реального времени. Этот подход использует потенциальные поля отталкивания вокруг препятствий, которые отталкивают робота, и потенциальное поле притяжения в целевой точке, которое притягивает робота к ней [3]. Сумма двух потенциалов заставляет робота испытывать обобщенную силу, равную отрицательному градиенту общего потенциала. Эта сила поддерживает направление и величину, которые перемещают робота в поисках точки с наименьшим потреблением энергии. Эффективная управляемость этого алгоритма в режиме реального времени позволяет обрабатывать однородные траектории, поэтому он широко используется в автономной навигации. Например, способ управления формированием двух

мобильных роботов, обеспечивающий эффективное преодоление препятствий с использованием алгоритма APF, представлен в [4]. Авторы работы [5] используют алгоритм APF, чтобы заставить робота достичь желаемого положения, избегая препятствий, добавляя локальные аттракторы, чтобы избежать проблемы локальных минимумов. Исходя из этого, в работе [6] предложен усовершенствованный метод APF для обнаружения и обхода фиксированных препятствий, который позволяет роботу достигать цели по оптимальной траектории, решая проблему, связанную с тем, что робот может попасть в ловушку локального минимума. Новое применение этой методики представлено в работе [7], в которой разрабатывается динамическое планирование траектории для отслеживания движущихся целей с использованием алгоритма APF, применяемого к беспилотному летательному аппарату. Недавно в [8] был представлен алгоритм навигации для мобильных роботов на базе операционной системы ROS, который позволяет осуществлять автономную навигацию в любой среде, используя Turtlebro Burger в качестве испытательного стенда, выполняя обработку данных с датчиков, установленных на роботе, для обработки алгоритмом APF и планирования маршрута на основе сенсорной информации.

В этой статье предлагается пропорциональный контроллер для отслеживания траектории движения и обхода препятствий мобильными роботами с дифференциальным приводом. Внутренняя одометрия генерируется с использованием алгоритма искусственного потенциального поля, обработки данных в режиме реального времени, лидара и датчиков абсолютного кодирования. Зависимость от удаленного компьютера устранена благодаря реализации алгоритмов навигации на Raspberry pi4 с ROS 2. Все эксперименты проводились на роботе Turtlebro Burger Robot. Результаты позволяют следовать установленным маршрутом с минимальными ошибками, что демонстрирует эффективность метода.

Статья организована следующим образом: сначала представлен анализ кинематической модели дифференциального робота с учетом смещенной контрольной точки, а также модели одометрии, основанной на данных датчика, для оценки его относительного положения в плоскости. В конце этого раздела разработан алгоритм отслеживания траектории с использованием пропорционального закона управления с компенсацией. Затем алгоритм обхода препятствий разработан в соответствии с подходом APF для определения наиболее оптимального пути достижения роботом цели. Далее, наконец, сгенерированные алгоритмы тестируются с помощью моделирования и эксперимента.

Предварительные приготовления

A. Кинематическая модель со смещенной точкой

Кинематическая модель описывала скорости и взаимосвязь между управляющими входами и поведением системы, учитывая представление системы в пространстве состояний. Предполагается, что транспортное средство движется в горизонтальной плоскости (X,Y) относительно смещенной точки P(x,y), как показано на рисунке 1 [9]. Параметры приведены в таблице 1. Рассматриваются две системы отсчета: локальная система {P}, которая размещается в контрольной точке и перемещается вместе с роботом; глобальная система {G} представляет собой фиксированную систему координат, в которой робот описывает траекторию. Поза робота может быть вычислена следующим образом:

$$P = \begin{pmatrix} x_r(t) + a\cos(\theta_r(t)) \\ y_r(t) + a\sin(\theta_r(t)) \end{pmatrix}. \quad (1)$$

Исходя из этих положений, уравнения линейной и угловой скоростей выглядят следующим образом:

$$\dot{P} = \begin{pmatrix} \nu(t)\cos(\theta_r(t)) - a\omega\cos(\theta_r(t)) \\ \nu(t)\sin(\theta_r(t)) + a\omega\sin(\theta_r(t)) \end{pmatrix}. \quad (2)$$

Переписав (2) в матричной форме, кинематическая модель в перемещенной точке P задаем формулой

$$\begin{bmatrix} \dot{P}_x \\ \dot{P}_y \end{bmatrix} = \begin{bmatrix} \cos(\theta_r) & -a\sin(\theta_r) \\ \sin(\theta_r) & a\cos(\theta_r) \end{bmatrix} \begin{bmatrix} \nu \\ \omega \end{bmatrix} = B(\theta_r)U. \quad (3)$$

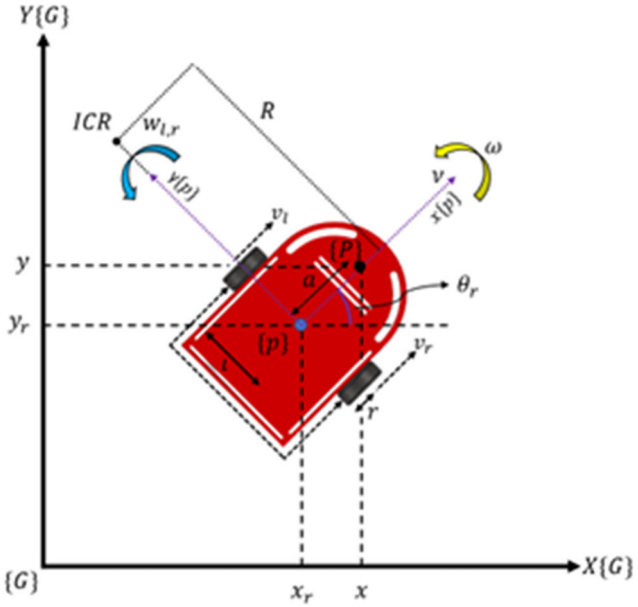


Рис. 1. DCL со смещенной точкой

Таблица 1. **Номенклатура для дифференциального анализа роботов**

Номенклатура
G – Глобальная справочная система
P – Локальная справочная система
P – Центральная точка робота
ICR – Мгновенный радиус поворота
R – Расстояние от центральной точки до ICR
x_r, y_r – Положение робота в системе отсчета G
θ_r – Ориентация робота относительно оси X_G
x, y – Положение смещенной контрольной точки
a – Расстояние от центра робота до смещенной точки
v, v_r, v_l – Линейные скорости робота
ω – Угловая скорость робота
l – Расстояние от центра робота до колес
r – Радиус колеса

B. Одометрия

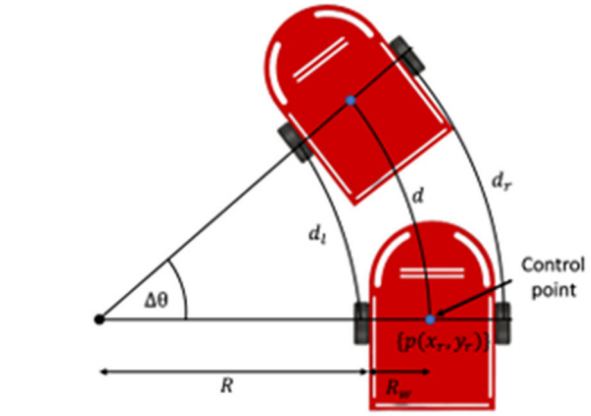
Turtlebot3burger оснащен динамическим микшером XL430-W250-Tm с абсолютным датчиком, который необходим для расчета одометрии. Расстояние, пройденное роботом в плоскости, определяется импульсами или отсчетами, генерируемыми датчиками, общим количеством импульсов за оборот (PPR) и диаметром колес.

1) ***Пройденное расстояние.*** Расстояние, пройденное каждым из колес d_i, с учетом информации от датчиков определяется по формуле

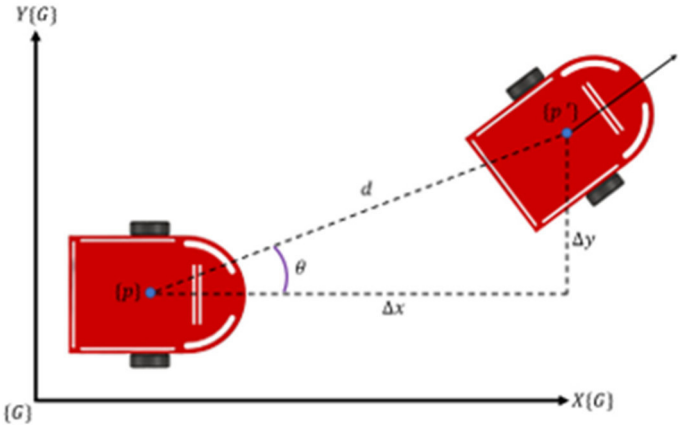
$$d_l = \frac{(S_l t - S_{l(t-1)})}{q} C = \frac{\Delta S_l}{q} C, \tag{4}$$

$$d_r = \frac{(S_r t - S_{r(t-1)})}{q} C = \frac{\Delta S_r}{q} C. \tag{5}$$

2) ***Одометрическая модель.*** На рисунке 2*a* показана диаграмма, соответствующая перемещению и ориентации робота относительно контрольной точки. В таблице 2 используются соответствующие параметры.



a)



b)

Рис. 2. Диаграммы перемещения и вращения робота

Как видно из рисунка 2*a*, где приведены формулы для вычисления дуги окружности, d_l, d_r и d определяются с помощью следующих уравнений:

$$d_l = R\,\Delta\,\theta; \tag{6}$$

$$d_r = (R\ +\ 2R_\omega)\,\Delta\,\theta; \tag{7}$$

$$d = (R\ +\ R_\omega)\,\Delta\,\theta, \tag{8}$$

где d_l и d_r – известные значения, поскольку они могут быть определены с помощью уравнений (4) и (5). Тогда θ и d – неизвестные значения, соответствующие изменению ориентации и общему пройденному расстоянию. Полное смещение и изменение ориентации определяется алгебраическими манипуляциями с уравнениями (6), (7) и (8) следующим образом:

$$\Delta\theta = \frac{d_r - R\Delta\theta}{2R_\omega} = \frac{d_r - d_l}{2R_\omega}, \tag{9}$$

$$d = (\mathrm{R} + R_\omega)\Delta\theta = d_l + \frac{d_r - d_l}{2} = \frac{d_l + d_r}{2}. \tag{10}$$

Таблица 2. **Номенклатура, используемая для расчета с помощью термометра**

Номенклатура
d_l,d_r – Пройденные расстояния на каждом колесе
D – Расстояние, пройденное от центральной точки
P – Центральная точка робота
$\Delta\,\theta$ – Увеличенный угол поворота
R – Расстояние от центральной точки до центра левого края
R_W – Расстояние от центральной точки до центра ободьев
Δ_x, Δ_y – Составляющие пройденного расстояния

С помощью уравнения (10) вычисляется изменение глобальных координат, как показано на рисунке 2b, где расстояние d разложено на составляющие x и представлено в виде приращений, поскольку расчет производится для каждого момента времени, например,

$$\Delta x(t) = d\cos(\theta_r(t));$$
$$\Delta y(t) = d\sin(\theta_r(t)). \quad (11)$$

3) ***Одноплатный компьютер (SBC).*** Для выполнения расчетов одометрии на ROS 2 необходимо понимать взаимодействие между Raspberry Pi 4 (RP4) и компонентами, составляющими TurtleBot в ROS.RP4 работает под управлением операционной системы ROS2 Foxy Fitzroy. Узлы связи, показанные на рисунке 3, инициализируют узел, отвечающий за взаимодействие компонентов робота. Эта сеть обеспечивает поток данных через темы и сообщения.

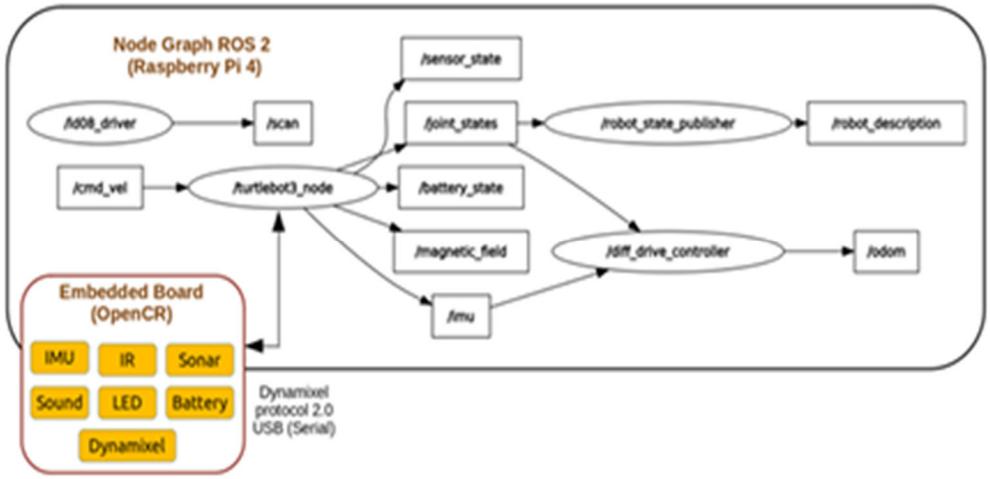


Рис. 3. Узловая сеть и поток данных

На рисунке 3 овалы обозначают узлы, а прямоугольники – темы. Стрелки указывают на передачу данных. Начиная с узла, направление стрелки указывает на то, что узел публикует информацию по одной или нескольким темам. И наоборот, когда стрелка указывает на узел, это означает, что узел подписан на одну или несколько тем. Ниже мы опишем каждую тему, которая содержит конкретную информацию о компоненте робота.

/состояние датчика: в этом разделе представлена информация от датчиков в виде счетчика и проверяется, включен ли крутящий момент на двигателях.

/состояние батареи: в этом разделе представлена вся информация, относящаяся к батарее, такая как напряжение, процент заряда и напряжение в каждом элементе.

/imu: содержит информацию о линейном и угловом ускорении, относящемся к локальной системе отсчета.

/магнитное поле: обеспечивает ориентацию робота и силу магнитного поля в навигационной среде.

/состояние соединения: предоставляет информацию об угловом положении каждого колеса и его угловых скоростях.

/сканирование: содержит информацию о расстоянии до объектов, обнаруживаемых вокруг робота каждым лучом лидарного датчика.

/odom: в этом разделе указывается положение относительно исходной точки, заданное при инициализации anode.

/cmdvel: записывает команды линейной и угловой скорости, отправляемые роботу в качестве управляющих воздействий, для расчета одометрии.

/состояние датчика: раздел используется для отображения количества тиков, генерируемых каждым двигателем при движении робота, с использованием сообщений левого и правого датчиков.

Алгоритм отслеживания траектории

Учитывая, что $B\,\theta_r$ не является сингулярным, разрабатывается контроллер обратной связи по состоянию, такой как

$$U = B\,(\theta_r)^{-1}u, \tag{12}$$

где u представляет собой вспомогательный управляющий вход. Подставляя (12) в (3), получаем линеаризованную модель дифференциального робота в виде

$$\dot{p} = u. \tag{13}$$

Затем, как показано в работе [10], предлагается пропорциональный регулятор с компенсацией, который регулирует линейную и угловую скорости в соответствии с генерируемой погрешностью.

Что касается предыдущих зависимостей, то мы получили

$$\mathrm{u} = \begin{bmatrix}\dot{\mathrm{x}}_{\mathrm{dr}}\\ \dot{\mathrm{y}}_{\mathrm{dr}}\end{bmatrix} + \mathrm{K_p e(t)}, \tag{14}$$

где $k_p \in \mathbb{R}^+$. Ошибка отслеживания e(t) определяется как разница между желаемой траекторией (x_{dr},y_{dr}) и расчетной (p_x,p_y) относительно глобальной системы отсчета {G}, что означает

$$e(t) = \begin{bmatrix}e_x(t)\\ e_y(t)\end{bmatrix} = \begin{bmatrix}x_{dr} - P_x\\ y_{dr} - P_y\end{bmatrix}. \tag{15}$$

Подставляя (13) в (14), получим

$$\begin{bmatrix}\dot{\mathrm{P}}_{\mathrm{x}}\\ \dot{\mathrm{P}}_{\mathrm{y}}\end{bmatrix} = \begin{bmatrix}\dot{\mathrm{x}}_{\mathrm{dr}}\\ \dot{\mathrm{y}}_{\mathrm{dr}}\end{bmatrix} + \mathrm{K_p e(t)} = \begin{bmatrix}\dot{x}_{dr} - \dot{P}_x\\ \dot{y}_{dr} - \dot{P}_y\end{bmatrix} + \mathrm{K_p e(t)} = 0. \tag{16}$$

Следовательно,

$$\dot{\mathrm{e}}(\mathrm{t}) + \mathrm{K_p e(t)} = 0. \tag{17}$$

Таким образом, управляющие входы задаются с помощью:

$$U = \begin{bmatrix}\cos(\theta_r) & -a\sin(\theta_r)\\ \sin(\theta_r) & -a\cos(\theta_r)\end{bmatrix}^{-1}\begin{bmatrix}\dot{x}_{dr}\\ \dot{y}_{dr}\end{bmatrix} + \mathrm{K_p e(t)}. \tag{18}$$

Обнаружение препятствий и их объезд с помощью APF

С помощью алгоритма APF робот-черепаха моделируется как частица, находящаяся под воздействием двух потенциальных полей: притягивающего и отталкивающего [3]. Потенциальное поле определяется функциями притяжения и отталкивания, как показано на рисунке 4. Оба они спроектированы так, чтобы робот притягивался к месту назначения и удалялся от препятствий, как частица, стремящаяся к точке с минимальной энергией [11]:

$$U(q, q_d, q_{obs}) = U_{attr}(q, q_d) + U_{rep}(q, q_{obs}), \tag{19}$$

поскольку q – относительные расстояния робота, q_d – желаемые расстояния, а q_{obs} – расстояния до препятствий.

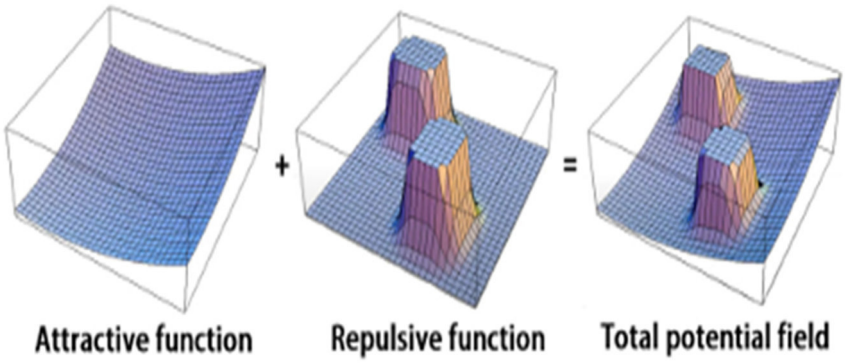


Рис. 4. Представление алгоритма APF

1) Функция притяжения. Считается, что параболическая функция конического притяжения определяет пороговое значение Q_d, соответствующее расстоянию, на котором функция изменяется с параболической на коническую, и постоянную параболической функции Q_d, получая:

$$U_{attr}(q, q_d) = \begin{cases} \frac{1}{2}\epsilon_q d^2 & si\ d\ <\ Q_d \\ Q_d\epsilon_q d - \epsilon_d \frac{Q_d^2}{2} & si\ d\ \geq\ Q_d \end{cases} \quad (20)$$

2) Отталкивающая функция. Гиперболическая отталкивающая функция определяется как

$$U_{rep}(q, q_{obs}) = \begin{cases} 0 & si\ \ Q^*\ \leq\ d \\ \epsilon_r(\frac{1}{d} - \frac{1}{Q_*}) & si\ Q_* \leq d \leq\ Q^* \\ \epsilon_r(\frac{1}{Q^*} - \frac{1}{Q_*}) & si\ d \leq Q_* \end{cases}, \quad (21)$$

где $Q^* = Q_{inf}$ и $Q_* = Q_{\min}$, где $Q_{\min}\ Q_{inf}$. Q_{inf} представляет собой расстояние, на которое препятствие не оказывает влияния, а $Q_{\min}$ – минимальное безопасное расстояние, которое должно быть окружено. Параметр ϵ_r – это показатель, который показывает, насколько функция сможет отразить удар робота.

3) Общее потенциальное поле. После вычисления функций притяжения и отталкивания общее потенциальное поле с учетом *n* препятствий может быть вычислено следующим образом:

$$U_{(q,q_d,q_{obs})} = U_{attr}(q, q_d) + \sum_{i=1}^{n} U_{rep}(q, q_{obs}^i). \quad (22)$$

Чтобы спланировать траекторию, необходимо следовать направлению, противоположному градиенту, который представляет собой вектор сил, определяемый суммой градиентов функций притяжения и отталкивания, величина и направление которых будут зависеть от конфигурации робота, пункта назначения и препятствий. Предполагается, что отрицательный градиент будет указывать на направление с минимальной энергией, то есть направление, которое минимизирует потенциальное воздействие на робота. Отрицательный градиент силы притяжения определяется формулой

$$\nabla U_{attr}(q, q_d) = \begin{cases} \begin{bmatrix} \epsilon_q d_d(x - x_d) \\ \epsilon_q d_d(y - y_d) \end{bmatrix} & si\ \ d_d\ \ \leq Q_d \\ \begin{bmatrix} \frac{Q_d\epsilon_q(x - x_d)}{d_d} \\ \frac{Q_d\epsilon_d(y - y_d)}{d_d} \end{bmatrix} & si\ d_d\ \ \geq Q_d \end{cases}, \quad (23)$$

где $d(d, d_q) = \sqrt{(x - x_d)^2 + (y - y_d)^2}$, (x, y) представляют относительное положение робота и (x_d, y_d) желаемое положение. Аналогично сила отталкивания, полученная в результате суммарного отталкивающего воздействия всех препятствий при подходе APF, представляет собой отрицательный градиент, заданный формулой

$$\nabla U_{rep}(l) = \sum_i \begin{cases} \begin{bmatrix} \frac{\epsilon_r l_{x,i}}{d_i^3} \\ \frac{\epsilon_r l_{x,i}}{d_i^3} \end{bmatrix} & si\ \ \ Q_* \leq d \leq Q^* \\ [0 \quad 0]^T & Otherwise \end{cases}, \quad (24)$$

где (l_x, l_y) – компоненты расстояния каждого датчика до препятствия. $d_i = \sqrt{l_{x,i}^2 + l_{y,i}^2}$ и ϵ_γ являются параметрами отталкивания. Сумма значений указывает на то, что при расчете учитываются все расстояния, определяемые каждым лучом лидарного датчика. Эта информация служит основой для закона управления, позволяющего безопасно доставить робота TurtleBro к желаемому месту назначения.

4) Закон управления. Исходя из диаграммы, показанной на рисунке 1, мы знаем относительное положение робота в смещенной контрольной точке, кроме того, матрица B(θ_r) из уравнения (3) обратима. Следовательно, мы получаем следующее уравнение, умножая обе части на обратную матрицу, которую мы будем называть J, и очищая линейную и угловую скорости

$$\begin{bmatrix}\omega\\ v\end{bmatrix} = \begin{bmatrix}\cos(\theta_r) & \sin(\theta_r)\\ -a\sin(\theta_r) & a\cos(\theta_r)\end{bmatrix}\begin{bmatrix}\dot{P}_x\\ \dot{P}_y\end{bmatrix}, \quad (25)$$

где v и ω – линейная и угловая скорости. Следовательно, с учетом формулы равномерного кругового движения угловые и линейные скорости, относящиеся к угловым скоростям, вычисляются по формуле

$$\omega(t) = \frac{r(\omega_r - \omega_l)}{2l}, \quad (26)$$

$$v(t) = \frac{r(\omega_l + \omega_r)}{2}. \quad (27)$$

Подставляя уравнения (26) и (27) в (2), получим

$$\begin{aligned}\dot{P}_x &= \frac{r}{2}(\omega_l + \omega_r)\cos(\theta_r) - \frac{r}{2l}(\omega_r - \omega_l)a\sin(\theta_r);\\ \dot{P}_y &= \frac{r}{2}(\omega_l + \omega_r)\sin(\theta_r) + \frac{r}{2l}(\omega_r - \omega_l)a\cos(\theta_r).\end{aligned} \quad (28)$$

Поскольку представляет интерес модулировать угловые скорости на каждом колесе, то после алгебраических манипуляций их можно переписать в виде выражений

$$\begin{aligned}\dot{P}_x &= \frac{\omega_r r}{2}(\cos(\theta_r) - \frac{a}{l}\sin(\theta_r) + \frac{\omega_l r}{2}(\cos(\theta_r) + \frac{a}{l}\sin(\theta_r)\\ \dot{P}_y &= \frac{\omega_r r}{2}(\sin(\theta_r) - \frac{a}{l}\cos(\theta_r) + \frac{\omega_l r}{2}(\sin(\theta_r) - \frac{a}{l}\cos(\theta_r)\end{aligned}, \quad (29)$$

которые могут быть перестроены следующим образом:

$$\begin{bmatrix}\dot{P}_x\\ \dot{P}_y\end{bmatrix} = \frac{r}{2}\begin{bmatrix}(\cos(\theta_r) + \frac{a}{l}\sin(\theta_r) & (\cos(\theta_r) - \frac{a}{l}\sin(\theta_r)\\ (\sin(\theta_r) - \frac{a}{l}\cos(\theta_r) & (\sin(\theta_r) + \frac{a}{l}\cos(\theta_r)\end{bmatrix}\begin{bmatrix}\omega_l\\ \omega_r\end{bmatrix} = j(q)h, \quad (30)$$

где $q = [x, y, \theta_r]^T$ и $h = [\omega_l, \omega_r]^T$. Таким образом, умножая обе части на обратную матрицу J, получаем следующее:

$$h = j^{-1}(q)\left[\dot{P}_x, \dot{P}_y\right]^T \quad (31)$$

$$j^{-1}q = \frac{1}{r}\begin{bmatrix}\cos(\theta_r) + \frac{l}{q}\sin(\theta_r) & \sin(\theta_r) - \frac{a}{l}\cos(\theta_r)\\ \cos(\theta_r) - \frac{q}{a}\sin(\theta_r) & \sin(\theta_r) + \frac{q}{a}\cos(\theta_r)\end{bmatrix}.$$

Основываясь на предыдущем, предлагаем следующий закон регулирования:

$$h = j^{-1}(q)\dot{\eta}^{ref}(q, q_d, l), \quad (32)$$

где η^{ref} – произведение линейной опорной скорости на суммарные силы, рассчитанные по отрицательному градиенту суммарного потенциального поля как $\eta^{ref} = v_{ref}\begin{bmatrix}F_x\\ F_y\end{bmatrix} = v_{ref}(\begin{bmatrix}\cos(\theta_r) & -\sin(\theta_r)\\ \sin(\theta_r) & \cos(\theta_r)\end{bmatrix})F_{rep}$.

Результаты

В этом разделе представлены результаты моделирования и экспериментов по отслеживанию траектории полета Turtlebro burger и схеме обхода препятствий.

A. Следование по пути

Для отслеживания траектории был реализован контроллер, указанный в уравнении (14). Были протестированы два типа траекторий: лемниската 1 и круговая траектория. Мы описали эти траектории, используя следующие уравнения:

$$\text{лемниската:}[x_d, y_d] = [x_0 + a\sin(\omega t) \quad y_0 + b\sin(2\omega t)];$$

$$\text{круг: } [x_d, y_d] = [r\cos(\omega t) \quad r\sin(\omega t)\ ].$$

Параметры, используемые для характеристики алгоритма, приведены в таблице 3.

Таблица 3. **Предлагаемые параметры траекторий**

Параметр	Переменная	Ценность
Размер (по оси Y)	a	0.6 m
Размер (ось X)	b	0.48 m
Центр лемнискаты	(x_0, y_0)	(0,0) m
Угловая частота (лемниската)	ω	0.157 rad/s
Радиус окружности	r	0.4 m
Угловая частота (окружность)	ω_c	0.1 rad/s

Эффективность отслеживания траектории была эвристически скорректирована путем проведения многочисленных имитационных и экспериментальных испытаний. Следующие значения были признаны оптимальными для каждого случая: ksim = 4, kexp = 1,6. На рисунке 5 показаны результаты моделирования, полученные при заданной траектории, по которой бургер Turtlebro следует непрерывной линией, а также ошибка отслеживания, полученная при траектории в форме лемнискаты с использованием Gazebo, которая обеспечивает среда 3D-моделирования, имитирующая реальный мир. Кроме того, можно импортировать модель робота и протестировать его работу в сложных имитируемых условиях. На графике показана максимальная погрешность в пределах ±0,3 см, соответствующая отклонению желаемого маршрута от реального, что является хорошим приближением. В то же время результаты экспериментов показаны пунктирной линией с учетом тех же условий эксплуатации. Мы получаем, что погрешность отслеживания находится в диапазоне от ±2,5 см по оси y до ±2 см по оси x.

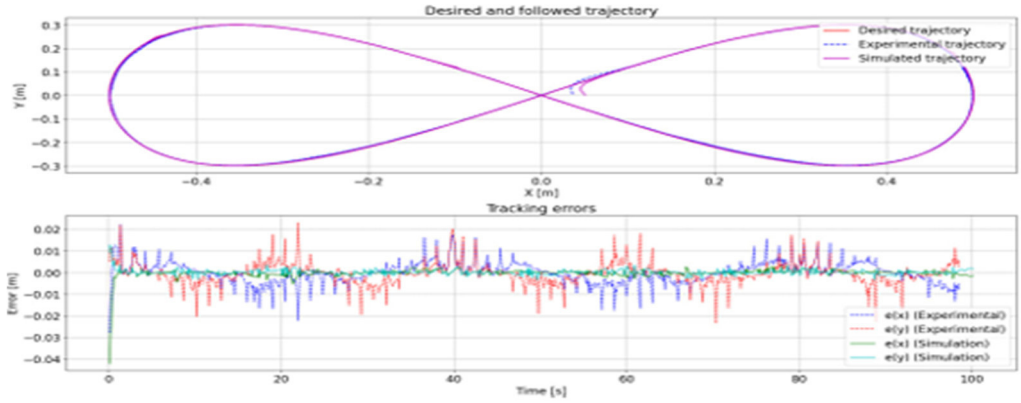


Рис. 5. Следующая траектория в форме лемнискаты

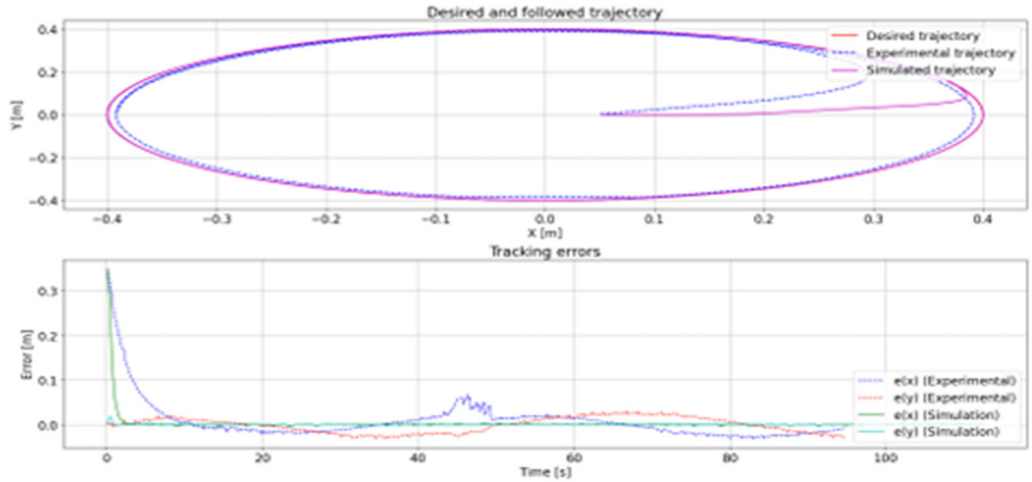


Рис. 6. Следующая траектория в форме круга

Впоследствии были проведены тесты с использованием круговой траектории для наблюдения за работой контроллера. Результаты, полученные по желаемой и пройденной траекториям, а также

ошибки отслеживания при моделировании показаны на рисунке 6 непрерывной линией, где наблюдается практически незначительная ошибка, которая асимптотически быстро сводится к нулю. Кроме того, в экспериментальной части пунктирными линиями показана погрешность отслеживания, составляющая приблизительно ±5 см по обеим осям, что отражено на графике траектории.

B. Избегание препятствий и следование по пути

В следующем разделе показаны результаты реализации алгоритма APF с использованием закона управления, описанного в уравнении (32). В моделируемой и экспериментальной среде тестирования одни и те же препятствия расположены в следующих позициях в метрах и ориентированы в радианах: [P_{Obs1} , O_{Obs1}] = [0.45, -0.6, 0], [P_{Obs2} , O_{Obs2}] = [1,4, -0,6, -0,3952] и [P_{Obs3} , O_{Obs3}] = [1, -1.4, 0]. Параметрами, использованными для моделирования и экспериментальных испытаний, были радиус колеса r = 0,033 м, расстояние l = 0,08 м, расстояние от центра до смещенной точки a = 0,04 м, скорость v = 0,15 м/с, которая обеспечивала наилучшие результаты при отслеживании и объезде препятствий, минимальное расстояние Q_{inf}. = 0,3 м, безопасное расстояние Q_{safe}= 0,16 м, ϵ_q = 2,2 и ϵ_r = 0,000845, причем последний параметр является наиболее важным для оценки.

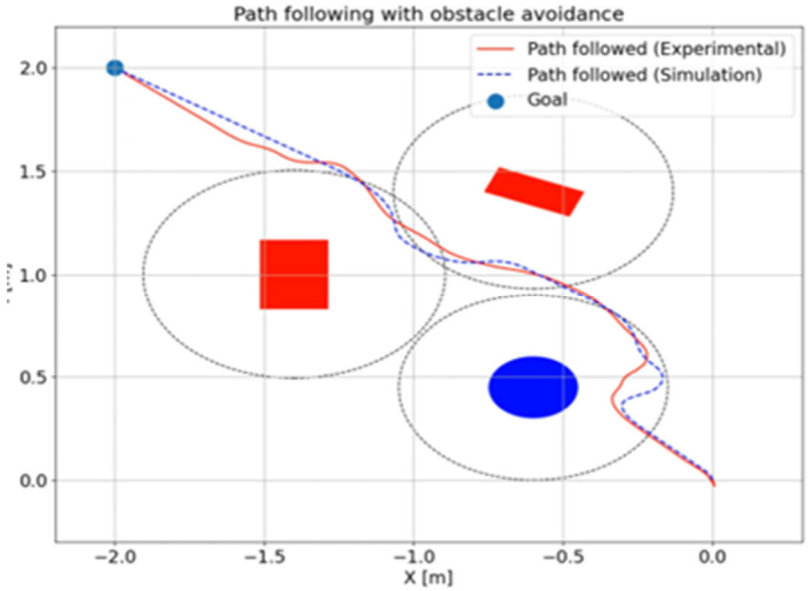


Рис. 7. Обход препятствия с помощью APF

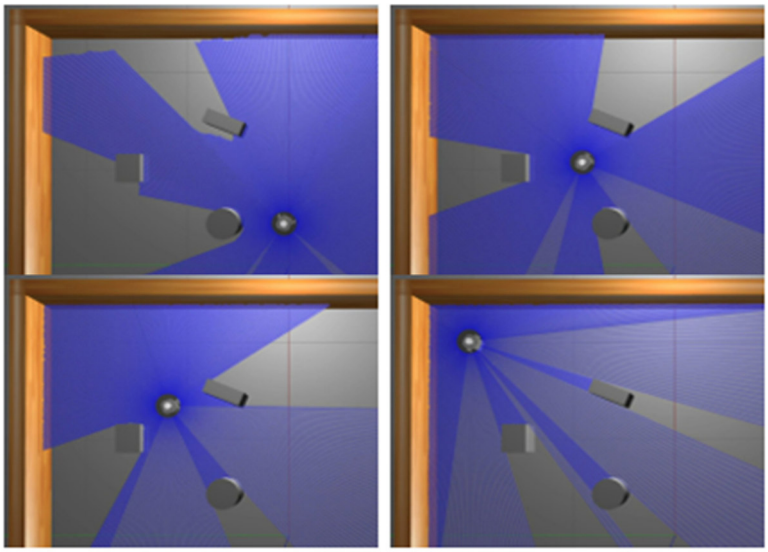


Рис. 8. Последовательность траекторий, пройденных в Gazebo

На рисунке 7 показаны результаты, полученные при моделировании Gazebo, при соблюдении траектории, описанной красным цветом, для достижения цели обхода препятствий в навигационном

пространстве. Результаты эксперимента показаны зеленым цветом. Для сравнения эффективности метода с теми же параметрами была создана точная копия имитируемой среды, которая показана на рисунке 8 в виде последовательности кадров, идущих к цели. Препятствия окружены по окружности пунктирной линией, которая указывает минимальную область, где робот обнаружит наличие блока. Как видно из обоих случаев, имитационного и экспериментального, результаты демонстрируют эффективность предложенного метода. Цель достигается за 26 секунд при моделировании и за 28 секунд при реальном применении.

Заключение

В данной статье рассмотрена проблема автономной навигации в мобильной робототехнике путем реализации алгоритмов управления, соответствующих отслеживанию траектории и обходу препятствий, для дифференциального робота Turtlebro burger, запрограммированного в ROS 2. Представлен подход, основанный на алгоритме APF, для расчета непрерывного маршрута, который позволяет роботу эффективно достигать пункта назначения, избегая препятствий. Особое внимание уделяется использованию одометрии с абсолютными датчиками и лидаром для обработки сенсорных данных в режиме реального времени. Эффективность этих методов подтверждается результатами, которые демонстрируют способность робота следовать установленным маршрутам с минимальными ошибками. Кроме того, предлагаемый подход обеспечивает гибкость, предназначен для практического применения при планировании маршрутов в статических и динамических средах, что подчеркивает его способность обрабатывать данные в режиме реального времени и устранять внешние зависимости. В дальнейшем мы планируем повысить точность отслеживания маршрута, используя подход model predictive control (MPC), который оценивает будущую переменную u для оптимизации будущего поведения выходных данных установки y [12-14], алгоритмы для определения наиболее быстрого маршрута в динамических навигационных средах, таких как как D*, а также интеграцию дополнительных датчиков для получения более точных результатов за счет повышения эффективности вычислений.

Список литературы (References)

1. R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, Introduction to autonomous mobile robots. MIT press, 2011.
2. J. A. Oroko and G. N. Nyakoe, “Obstacle avoidance and path planning schemes for autonomous navigation of a mobile robot: A review.”
3. J. A. Oroko and G. Nyakoe, “Obstacle avoidance and path planning schemes for autonomous navigation of a mobile robot: a review,” in: Proceedings of the Sustainable Research and Innovation Conference, 2022, pp. 314–318.
4. J. Lagunas-Avila, R. Castro-Linares, and J. Alvarez-Gallegos, “Obstacle avoidance in leader-follower formation using artificial potential field algorithm,” 2021, pp. 1–6.
5. M. Melchiorre, L. S. Scimmi, L. Salamina, S. Mauro, S. Pastorelli et al., “Robot collision avoidance based on artificial potential field with local attractors,” 2022.
6. S. M. H. Rostami, A. K. Sangaiah, J. Wang, and X. Liu, “Obstacle avoidance of mobile robots using modified artificial potential f ield algorithm,” EURASIP Journal on Wireless Communications and Networking, vol. 2019, p. 70, 2019. Available: https://doi.org/10.1186/s13638-019-1396-2
7. H. M. Jayaweera and S. Hanoun, “A dynamic artificial potential field (d apf) uav path planning technique for following ground moving targets,” IEEE Access, vol. 8, pp. 192760–192776, 2020.
8. M. Panagoda, M. Lokuliyanage, A. Senarath, N. K. V. M. N. Nisansala, R. W. A. D. U. Rajapakshe, U. U. S. Rajapaksha, and C. Jayawardena, “Moving robots in unknown environments using potential field graphs,” 2022, pp. 96–101.
9. F. I. Heredia-Moreno, M. E. J´ ativa-Brito, C. G. Merino-S´anchez, and A. V. Mac´ıas-Espinales, “Simulaci´ on del modelo matem´atico de un robot m´ovil diferencial con control de posici´ on sin orientaci´ on basado en ley de lyapunov,” Polo del Conocimiento, vol. 5, no. 1, pp. 880–899, 2020.
10. A. Escamilla and C. Mauricio, “Esquema de formaci´ on de robots m´oviles con evasi´on de obst´ aculos est´ aticos y din´ amicos,” 2022.
11. O. Khatib, “Real-time obstacle avoidance system for manipulators and mobile robots,” in Proceedings of the 1985 IEEE International Conference on Robotics and Automation, St. Louis, MO, USA, 1985, pp. 25–28.
12. G. Valencia-Palomo and J. A. Rossiter, “Auto-tuned predictive con trol based on minimal plant information,” IFAC Proceedings Volumes, vol. 42, no. 11, pp. 554–559, 2009.
13. G. Valencia-Palomo, K. R. Hilton, and J. A. Rossiter, “Predictive control implementation in a plc using the iec 1131.3 programming standard,” in 2009 European Control Conference (ECC). IEEE, 2009, pp. 1317 1322.
14. S. Chitraganti, S. Aberkane, C. Aubrun, G. Valencia-Palomo, and V. Dragan, “On control of discrete-time state-dependent jump linear systems with probabilistic constraints: A receding horizon approach,” Systems & Control Letters, vol. 74, pp. 81–89, 2014.