Indian Journal of Science and Technology, *Vol 8(29), DOI: 10.17485/ijst/2015/v8i29/86866, November 2015*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Control Algorithms of a Mobile Robot with Castor Wheels in Field with Barriers

Ivan Anatolievich Vasiliev and Alexander Vladimirovich Popov*

Central Research and Development Institute of Robotics and Cybernetics, Department 32 Saint Petersburg, Russian Federation; s.nikitin@rtc.ru

Abstract

There are different kinematic schemes of wheeled mobile robots. This article presents full four - wheeled drive with two steer wheels. Specific algorithms of the mobile robot's movements in field with barriers have been developed. All the algorithms have been implemented in practice and tested on a work-performing robot. Scientific novelty of the article consists in the usage of only one simple indicator of a working area. Besides, any sensor, including a contact one, can be used as such an indicator. The application of these algorithms is evident - they may be primarily used in control systems of self - driving cars.

Keywords: Algorithms, Robotics, Robot

1. Introduction

This article presents control system of a four - wheeled mobile robot with two castor yaw wheels (Figure 1).

The target point position in the coordinate system, known to the robot[1,2], is suggested as a motor task. Let us assume that the motion path of the robot consists of straight ways as longest as possible, lying on straight lines, passing through the point, where the target is.

2. Concept of the Research Conducting

Based on the above, the robot motion looks as follows: the robot starts moving from the home position to the position, from which it can go in the direction of the target. If it meets any barriers, then, having passed them, the robot should again take up the position, from which it goes in the direction of the target[2].

Firstly, we examine the middle control level of the robot, i.e. what control should be attributed to the wheels for the desirable path tracking.

All four wheels turn evenly for rectilinear motion. For arch motion of R radius velocities are calculated as follows (Figure 2).

$$\dot{q}_{ss} = \dot{q}_{target}\frac{R-\frac{rut}{2}}{R}\dot{q}_{sl} = \dot{q}_{target}\frac{R+\frac{rut}{2}}{R}\dot{q}_{cs} = \frac{\dot{q}_{target}}{R}r_1 \quad (1)$$

$$\dot{q}_{cl} = \frac{\dot{q}_{target}}{R}r_2 r_1 = \sqrt{\left(R+\frac{rut}{2}\right)^2 + B^2} \quad (2)$$

$$r_2 = \sqrt{\left(R=\frac{rut}{2}\right)^2 + B^2} \quad (3)$$

where ss - static wheel, going in a small radius (inside);
sl - static wheel, going in a large radius (outside);
cs - castor wheel, going in a small radius;
cl - castor wheel, going in a large radius;
rut - wheel tread;
$\dot{q}_{target}$ - target velocity, i.e. radial velocity of the robot's central part (midpoint between static wheels);
B - wheel base (distance between wheel axis).

*Author for correspondence

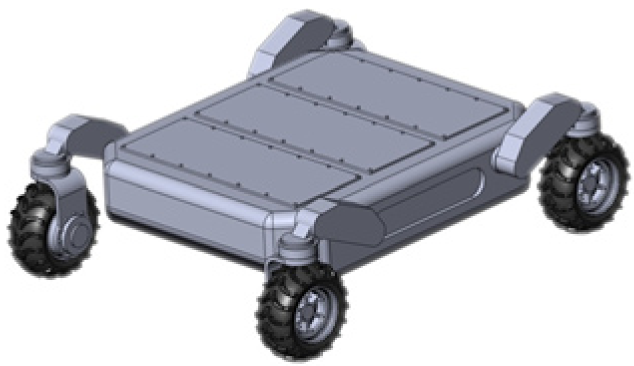

Figure 1. Image of a platform.

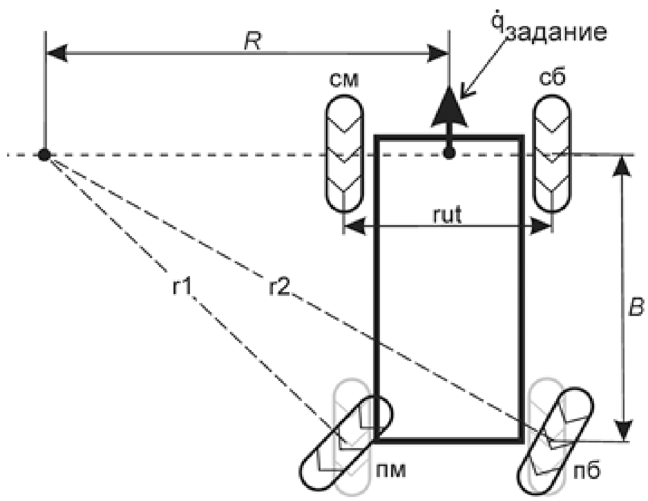

Figure 2. Robot plan during arch motion.

2.1 Algorithms of Moving to the Target

There are three algorithms, as three scenarios are possible:

1. The target point is located far away and can be reached by a normal turn;
2. The target point is located nearby and can be reached by performing one turn in stages;
3. When moving directly to the target, robot meets a barrier.

Obviously, all three scenarios are implemented conjointly in the robot control system, but it is reasonable to separate them for the sake of simplicity of presentation.

2.2 Mutual Position of the Robot and Target

The target position can be "far from" and "close to" the current position of the robot.

It is necessary to explain, what "far" and "close" mean (Figure 3).

The current position of the robot is marked by point*0*in the figure, and orientation - by the vector going from it. The interval (*0, target*) coincides with the direction from the current position of the robot to the target position. We can construct an arch of R_{min} radius, so that the arch has the robot orientation vector, tangent in the point of the current position of robot. Consequently, if distance *L* to the target is larger than the chord value *Chord*, then we consider the target to be "far". The required calculation is below:

$$H = R(1 - Cos(\alpha))\xi = \arccos\left(1 - \frac{H}{2R}\right)\delta = \arccos\left(1 - \frac{H}{R}\right) \quad (4)$$

$$\alpha_{turn} = \delta + \xi \quad (5)$$

$$Chord = 2R\sin(\delta), \quad (6)$$

where all the symbols are implied from Figure 3. As low as practicable R_{min} radius can be used here as a turning radius *R*.

If $L \geq Chord$, then we consider the target to be "far". In other words, the robot will come to the target or to the target direction by turning around R_{min} radius. If $L < Chord$, the robot will not reach the target by a normal turn, and the target is considered to be "close".

2.3 Target is Far

Motion algorithm here is the following:

1. If α_{turn} equals zero, go directly to the target. This algorithm is completed and it is required to proceed directly to the target motion algorithm.
2. Wheel turn in the direction of α_{turn} sign;
3. Go radially, checking, whether you have reached the target:
   - At the target - stop.
   - Toward the target - proceed directly to the target motion algorithm.

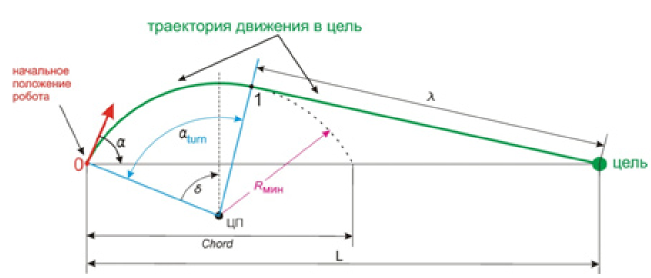

Figure 3. Plan of motion to the target point.

- Angularly to α_{turn} - calculate new parameters and proceed to point 1.
- To insurmountable barrier - proceed to point 4.

4. Wheel turn in the neutral position;
5. Go straightly rearward at a safe distance (for instance, 0.5 meters);
6. Proceed to point 2.

The idea of the algorithm of barrier avoidance (pp. 4–6) (Figure 4).

Traffic ability, i.e. absence of barriers in a spaced position *s*, as well as output to direction toward the target is being checked, while moving radially. If the robot meets a barrier, when it stays in point 1, then the robot drives rearwards to the point 2 in a spaced position *delta*.

After that the robot once again drives radially, checking for barriers and for direction to the target. This recursive algorithm allows running into the target without bumping into the barrier.

If the position of a barrier makes direction to the target impossible, then the turn to initially calculated α_{turn} angle is tested, and if it is the case, then the algorithm startup is initiated once again[3].

2.4 Target is Close

The target is located too close to be reached in one turn around a minimal radius. The following algorithm is applicable here:

1. Wheel turn opposite to α_{turn} angle;
2. Rearward movement radially at a certain angle (for instance, for 0.5 rad) with checking the direction to the target;after having reached the direction to the target, proceeddirectly to the target motion algorithm.
3. Wheel turn according to α_{turn};
4. Forward movement radially at a certain angle (for instance, for 0.5 rad).
5. Checking of:
   - Achieving the direction to the target - proceed directly to the target motion algorithm.
   - Reaching the target - stop.
   - aMeeting an insurmountable barrier - proceed to point 1.
6. Proceeding to point 1.

The idea of this algorithm is shown in Figure 5.

The robot stays at the point 0 in the beginning. Then the robot goes radially to point 1, leaving *s* sized safety area to the barrier. After that the robot goes rearward to point 2 radially at a certain angle. From point 2 the robot goes forward radially and at the point 3 it comes towards the target.

The following question is arising here: why in the "target is far" mode while avoiding a barrier, should the robot go rearward straightway, and in the "target is close" mode should it go in an arch? It is made on security grounds. In the first mode the most important is to avoid the barrier. The target is far and we will anyway enter its direction. That is why it is possible to move rearward even straightway (because there are no indicators at the back of the robot). Whereas, in the second mode we have a goal, the robot should turn to the target. Avoiding the barrier is a secondary task. Moreover, "star - formed" turn maneuver will be realized in this mode even if there are no barriers on the way.

2.5 Direct to the Target Motion

As robot cannot make a turn in place, it is necessary to not only bypass a barrier, but also to move on for a certain distance[4] during the barrier avoidance, so that it will not be necessary to bypass this barrier later on, which can cause algorithm circularity ("the robot is lost").

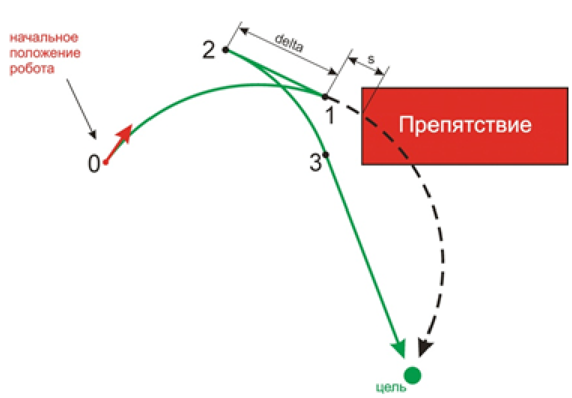

Figure 4. Barrier avoidance while moving radially.

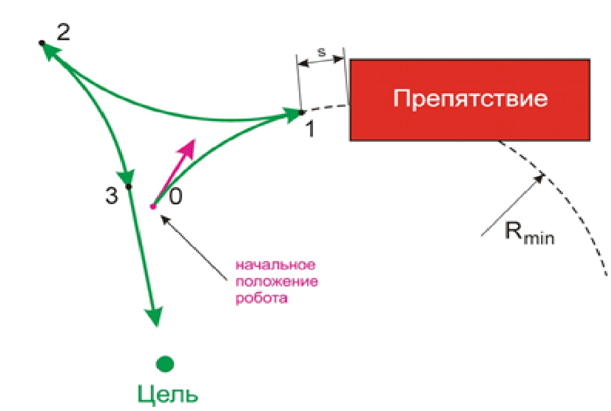

Figure 5. Motion to the closest point.

Algorithm in this case is quite simple:

1. Wheel turn to a neutral position;
2. Moving straightway with checking:
   - Reaching the target - stop.
   - Meeting a barrier on the way - proceed to point 3.
3. Wheel turn opposite to α_{turn} angle;
4. Rearward movement radially at a certain angle (for instance, for 0.5 rad)
5. Wheel turn according to α_{turn};
6. Moving straightway in an arch with checking:
7. If there is a barrier on the way – proceed to point 3.
8. If there is certain space for forward motion –proceed to point 7.
9. Wheel turn to a neutral position;
10. Moving straightway until minimal distance, measured by a distance meter, becomes much bigger;
11. Go forward in a certain distance (for instance, ½ of the robot's length);
12. Proceed to the calculation of current position and target parameters point.

The algorithm is shown in Figure 6.

In the beginning the robot is in point 0 and is directed to the target. The robot goes directly to the target till point 1, where distance to the insurmountable barrier becomes less than *s*. After that the robot goes rearward in an arch to point 2 at a certain angle. Then the robot is moving in an arch forward, simultaneously checking the distance to the barrier - if this distance is more than a specified value, as in point 3, the robot goes straightway. This motion is realized until the barrier has been fully bypassed (as in point 4). After that, the robot is moving straightway in a certain distance to the point 5. Then the algorithm of motion to the target should restart.

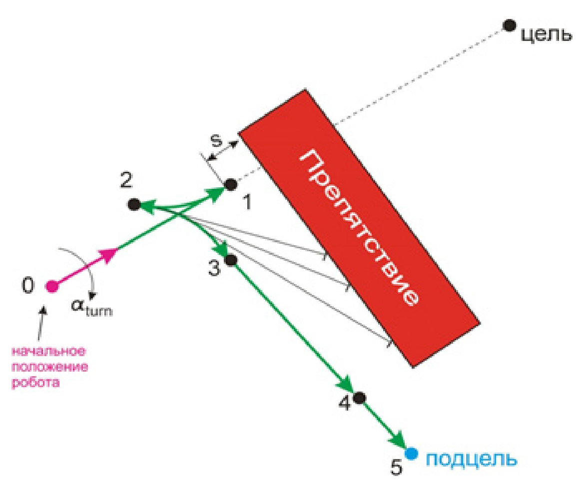

Figure 6. Barrier avoidance in straightway.

3. Discussion

The list of algorithms given above assures that the robot reaches the target position with the required accuracy. But these algorithms are not suited for working in labyrinths, i.e. in the zones where distances between barriers are equal to the specific robot size. That is why this article is concerned with working in fields with more or less infrequent barriers. Although the algorithms have been tested in a quite non-spacious room.

4. Conclusion

The information about robot localization (i.e. definition of current coordinates) is omitted in this article on purpose. Due to working in open country, localization by a laser scanning distance meter has not proved its relevance. In such cases it is necessary to have other localization indicators: GPS/GLONASS - navigation, electronic compasses and inertial navigation systems. During workout on solid surface wheel indicator localization has shown its reasonably good performance.

5. Acknowledgments

The article has been written with financial support from the State represented by the Ministry of Education and Science in the course of work, according to Agreement № 14.578.21.0047 RFMEFI57814X0047 about granting for the purposes of Federal Target Program "Research and development in priority orientations of science and technology complex development in Russia for 2014–2020".

6. References

1. Siciliano B, Khatib O. (Eds). Springer Handbook of Robotics. Berlin: Springer - Verlag Berlin; Heidelberg: 2008.
2. Bourdakov SF, Miroschnik IV, Stelmakov RE. Motion control system of wheeled robots. SPb: "Nauka"; 2001.
3. Polovko SA, Smirnova EY, Yurevich EI. Qualitative guidance of wheeled robots. Scientific and technical journal "Robotics and technical cybernetics". 2014; (3-4):30–3.
4. Vasilyev IA, Lyashin AM. Control System of Mobile Vehicle Developed for Cross - Country Motion. Proceedings of "International Symposium on Industrial Electronics 2005". Dubrovnic - Croatia. P. 173–5.