Trajectory Tracking Control and Obstacle Avoidance for a Differentially Driven Mobile Robot *

Wojcich Kowalczyk ∗ Maciej Michalek ∗∗
Krzysztof Kozłowski ∗∗∗
Chair of Control and Systems Engineering, Poznan University of Technology, Piotrowo 3A, 60-965 Poznań, Poland

∗ e-mail: wojciech.kowalczyk@put.poznan.pl
∗∗ e-mail: maciej.michalek@put.poznan.pl
∗∗∗ e-mail: krzysztof.kozlowski@put.poznan.pl

Abstract: In this paper a Vector Field Orientation control algorithm for a differentially-driven mobile platform in an environment with obstacles is presented. The goal is to track the desired trajectory. Collision avoidance is assured using the local artificial potential function (APF) method. A proof of stability and convergence for the algorithm is presented. Effectiveness of the algorithm is illustrated with experiments on a real robot. The paper also presents how the parameters of the control algorithm influence effectiveness of the method in practice.

1. INTRODUCTION

Many different control strategies for wheeled mobile robots were proposed during the last two decades of robotics research Kolmanovsky [1995], Morin [2008]. A common approach to the motion control solution for mobile robots is to assume that the movement takes place in an obstacle-free environment. With this assumption in the background the convergence analysis for the control laws does not take into account possible collisions which may happen between a robot and obstacles during the motion transient stage. It is also commonly assumed that the reference trajectory or the reference stabilized point are collision-free, which means that they were properly planned or selected before robot movement. However, in real motion conditions such prerequisites seem to be quite stringent, not only due to the obvious reasons but also because the collision-free motion planning task in a cluttered environment may be computationally very demanding and time consuming Minguez [2008]. Therefore, the practical utilization of the proposed control laws requires their further extensions which would improve reliability of the original solutions to the unexpected collision events.

On the other hand a number of avoidance strategies have been proposed using different approaches. Usually obstacles are surrounded by a local artificial potential function (APF) causing that the robot to be repelled when it is too close to the obstacles’ boundaries. This approach can be used to avoid collisions with static objects but also with other robots Do [2008], Kowalczyk [2008], Mastellone [2008].

Another solution is the navigation function approach Rimon [1992] which gives control without local minima. The control problem can be solved for very complex environments Rimon [1991], however, that method requires large computational effort. Because of that, navigation function methods are rarely used.

This paper describes a control algorithm which merges together the two important control problems, namely trajectory tracking and obstacle collision avoidance for a differentially-driven mobile robot as a single motion control algorithm. There is no need to replan the trajectory as the collision avoidance problem is solved in the motion control layer. The proposed solution is a geometrically motivated extension of the original (VFO) control approach presented in details in Michalek [2008].

The paper is organized as follows. In Section 2 a model of the system, a collision avoidance module and a trajectory tracking algorithm are presented. In Section 3 a stability proof for the algorithm is discussed. In Section 4 experimental results are given. Section 5 includes concluding remarks.

2. ARTIFICIAL POTENTIAL FUNCTION

2.1 Model of the system

The kinematic model of a robot is given by the equation:

\[
\begin{bmatrix}
\dot{\theta} \\
\dot{x} \\
\dot{y}
\end{bmatrix} =
\begin{bmatrix}
1 & 0 \\
0 & \cos \theta \\
0 & \sin \theta
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2
\end{bmatrix},
\]

where \(x, y, \) and \(\theta\) are position and orientation coordinates of the robot, and \(u = [u_1 u_2]^T\) is the control vector, which includes \(u_1\) - angular velocity control and \(u_2\) - linear velocity control.

Before the control values are applied to the motion controllers they are transformed to the wheel velocities:

\[
\begin{align*}
\omega_R &= \frac{u_2 + \frac{1}{2}bu_1}{r_w}, \\
\omega_L &= \frac{u_2 - \frac{1}{2}bu_1}{r_w},
\end{align*}
\]

where \(r_w\) is the radius of the robot wheels and \(b\) is a half of distance between robot wheels.
Practical realization of the control algorithm requires control input consideration. In the case of a differentially-driven wheeled vehicle the kinematic limitation is imposed on the maximum angular velocity \( \omega_{\text{max}} > 0 \) which can be realized by the vehicle wheel. In order to take into account this limit the following control scaling procedure is proposed. Denoting by \( \omega = [\omega_R \ \omega_L]^T \) the computed and unlimited control input vector from (2), the scaled and physically realizable input \( \omega_d = [\omega_{Rd} \ \omega_{Ld}]^T \) can be obtained as follows:

\[
\omega_d(\tau) = \frac{\omega(\tau)}{s(\tau)},
\]

where

\[
s(\tau) = \max \left\{ 1, \frac{[\omega_R(\tau)]}{\omega_{\text{max}}}, \frac{[\omega_L(\tau)]}{\omega_{\text{max}}} \right\} \geq 1.
\]

The above scaling procedure keeps the direction of the scaled (limited) control vector \( \omega_d \) equal to the previously computed control vector \( \omega \) preserving in turn the instantaneous vehicle motion curvature \( \kappa = \omega/v \).

2.2 Obstacles and artificial potential fields

Each obstacle located at position \( p_i \) is surrounded by the artificial potential field which exerts a repelling force on robot entering it. In this paper it is assumed that all obstacles can be modeled as circle-shaped objects (Fig. 2). Let us define the set of coordinates for the collision area of the \( i \)-th obstacle:

\[
\Delta_i = \{ p \in \mathbb{R}^2, l_i \leq r_i \},
\]

where \( p = [x \ y]^T \) and \( l_i = \|p - p_{o_i}\| \), the set of coordinates for the repel area:

\[
\Gamma_i = \{ p \notin \Delta_i, r_i < l_i < R_i \},
\]

and \( D_i = \Delta_i \cup \Gamma_i \) - the set that includes both areas, where \( r_i \) is the radius of the least circle that covers the obstacle, and \( R_i \) is the radius of the area where the repel force caused by APF acts.

The artificial potential function (APF) is given by the following equation Kowalczk [2008]:

\[
B_{ai}(l_i) = \begin{cases} 
  \text{not defined for } & l_i < r_i \\
  \frac{e^{r_i - l_i}}{r_i - l_i} & \text{for } \ r_i \leq l_i < R_i \\
  0 & \text{for } \ l_i \geq R_i
\end{cases},
\]

where \( r_i > 0, R_i > 0 \) fulfill inequality \( R_i > r_i \), and \( l_i \) is the distance to the obstacle.

The collision avoidance task requires the APF to infinity as the distance to the boundary of the colliding object decreases to zero. To fulfill this condition Eq. (7) is mapped to \( (0, \infty) \) using the following transformation:

\[
V_{ai}(l_i) = \frac{B_{ai}(l_i)}{1 - B_{ai}(l_i)}.
\]

In Fig. 1 an example of the APF for \( r = 1 \) and \( R = 2 \) is presented. The function is smooth in the whole used range: \( l \in (r, \infty) \). Other APF’s can be found in literature Mastellone [2008], Do [2008]. The former is non-smooth but applicable to the kinematic algorithm presented here, the latter is smooth but it is integral-based and requires many iterations to be computed precisely.

2.3 Tracking control with collision avoidance

Definition 1. (Control problem). Let us define the admissible reference trajectory \( q_d(t) = [q_x(t) \ q_y(t)]^T \in \mathbb{R}^2 \), \( p_d(t) = [x_d(t) \ y_d(t)]^T \in \mathbb{R}^2 \) which fulfills kinematics (1) for some bounded reference input \( u_d(t) = [u_{1d}(t) \ u_{2d}(t)]^T \in \mathbb{R}^2 \) with the following persistent excitation condition: \( \forall t \geq 0 : u_{2d}(t) \neq 0 \).

Assuming that:

A1. \( \forall t \geq 0 \ p_{d}(t) \notin D = \bigcup_i D_i \)

A2. \( p(t) \in \Gamma = \bigcup_i \Gamma_i \Rightarrow q_d(t) = q_d(t^{-}) \); \( q_d(t) = q_d(t) \equiv 0 \)

A3. \( D_i \cap D_j = \emptyset, \quad i \neq j \)

the aim is to find the bounded feedback control law \( u = u(q_d, q_{\cdot}) \) which guarantees that the tracking errors

\[
e_{\theta} = \theta_d - \theta, \quad e(t) = p_d(t) - p(t) \quad (9)
\]

asymptotically converge to zero if \( \forall t \geq 0 \ p(t) \notin \Delta = \bigcup_i \Delta_i \). In above equation \( f_\theta : \mathbb{R} \mapsto S^1 \).

Convergence of the system is possible due to assumption A1. In the case the robot is out of the APF the task for the robot is pure trajectory tracking. When it gets near the obstacle the tracking task is modified to avoid collisions. As the robot approaches the boundary of the obstacle the tracking task become less important and convergence is more disturbed. The solution to this is planning trajectories according to assumption A1.

Undisturbed collision avoidance is possible due to assumption A2. As collision avoidance is a higher priority problem, because a collision could lead to system damage, the trajectory tracking is temporarily suspended to facilitate bypassing the obstacle. After the robot leaves the collision region the desired trajectory is updated.

Assumption A3 guarantees that there are no local minima in the task space. Local minima can lead to a deadlock. The tracking error is defined as follows:

\[
e = [e_x \ e_y]^T = p_d - p. \quad (10)
\]

Now the Vector Field Orientation control algorithm (VFO) Michalek [2008] will be described. Let us propose the convergence vector:

\[
h = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} = \begin{bmatrix} k_1 e_{\alpha} + \delta_{\alpha} \\ k_p E_x + \delta_{\alpha} \\ k_p E_y + \delta_{\alpha} \end{bmatrix},
\]

where \( k_1, k_p > 0 \) are orientation and position control gains, respectively. Modified tracking error \( E \) is computed as a difference between the tracking error and the sum of gradients of the APFs associated with the obstacles:

\[
E = [E_x \ E_y]^T = e - w, \quad (12)
\]

where \( w \) is defined as follows:

\[
w = \sum_{i=1}^{M} \frac{\partial V_{ai}(l_i)}{\partial p} \right]^T. \quad (13)
\]
In the above equation $V_{\alpha l}(l_i)$ is the APF of the $i$-th obstacle, $l_i = \|p - p_i\|$ is the distance between the robot and the $i$-th obstacle for $i = 1, \ldots, M$, $M$ - the number of obstacles. A graphical interpretation of the modified position error $E$ is computed as a difference between the tracking error and the gradient of the APF.

Fig. 2. Robot is in the repel area of obstacle $O_1$. Modified position error $E$ is computed as a difference between the tracking error and the gradient of the APF.

The auxiliary orientation variable $\theta_a$ has the following form:

$$\theta_a = \text{Atan2c}(\text{sgn}(u_2d) \dot{y}_d, \text{sgn}(u_2d) \dot{x}_d),$$

which is equivalent to the auxiliary orientation variable (14) takes the form:

$$\theta_a = \text{atan2c}(\text{sgn}(u_2d) \dot{y}_d, \text{sgn}(u_2d) \dot{x}_d),$$

where $\text{sgn}(\cdot)$ denotes the signum function and $\text{Atan2c}(\cdot, \cdot)$ is a continuous version of $\text{Atan2}(\cdot, \cdot)$. The definition of $\text{Atan2c}(\cdot, \cdot)$ can be found in Michalek [2008]. The term $\text{sgn}(u_{2d})$ in (14) determines the direction of motion of the platform during the trajectory tracking and as it can be taken constant it does not affect the time derivatives of $\theta_a$ needed to compute control.

The control vector $u = [u_1 \ u_2]^T$ is given by the following equation:

$$u_1 = h_1, \ u_2 = h_2 \cos \theta + h_3 \sin \theta$$

Remark 2.1. When $e_\alpha \in (\frac{\pi}{2} + \pi d - \delta, \frac{\pi}{2} + \pi d + \delta)$, where $\delta$ is a small positive value, $d = 0, \pm 1, \pm 2, \ldots$, then the auxiliary orientation variable $\theta_a$ is replaced by $\theta_a = \theta_0 + \text{sgn}(e_\alpha - (\frac{\pi}{2} + \pi d)) \varepsilon$, where $\varepsilon$ is a small value that fulfills the condition $\varepsilon > 0$.

When the desired trajectory is continuously along the axis of the robot wheels, violation of nonholonomic constraints occurs. The procedure described in Remark 2.1 lets the robot reach this state.

Remark 2.2. When the robot reaches a saddle point, the reference trajectory is disturbed to drive the robot out of a local equilibrium point. This technique was discussed in detail in Urakubo [2004].

Remark 2.3. In some configurations the following condition may occur: $\|h^*\| = 0$, where $h^* = [h_2 \ h_3]^T$. In this case $\theta_a(t)$ cannot be computed by Eq. (14) and the previous value of the auxiliary orientation variable is taken: $\theta_a(t) = \theta_a(t^-)$. As shown in Rimon [1992] there is one saddle point associated with each circle-shaped obstacle. The situation described in Remark 2.3 occurs at a saddle point. Equality $\|h^*\| = 0$, may also occur outside $\Gamma_i$, however, this state is non-attracting and the same strategy as shown in remark 2.3 is applicable.

3. STABILITY ANALYSIS

In this section a proof of stability is presented. According to assumption A3 the indexes denoting the number of the obstacles are omitted in this section without loss of generality. The proof consists of three steps: 1. $\lim_{t \to \infty} (\theta(t) - \theta_0(t)) = 0$ - proof of convergence of the orientation to the auxiliary orientation variable, 2. $\lim_{t \to \infty} (p(t) - p_0(t)) = 0$ - proof of stability and asymptotic convergence of the robot position to the reference position, 3. $\lim_{t \to \infty} e_\theta = 0$ - proof of convergence of the auxiliary orientation variable to the desired orientation.

Substituting the first row of (11) into the first equation of (15) and using the first row of Eq. (1) one obtains: $e_\alpha = -k_1 e_\alpha$, which guarantees that an auxiliary orientation error decreases exponentially to zero:

$$\lim_{t \to \infty} e_\alpha = 0,$$

and the robot orientation converges exponentially to the auxiliary orientation.

The Lyapunov function candidate is given by the following equation:

$$V_l = \frac{1}{2} e^T e + V_a(l) = \frac{1}{2} (e_x^2 + e_y^2) + V_a(l),$$

where $V_a$ is a potential given by (8). Two cases are investigated separately:

C1. $p \in D^c$, $D^c = \mathbb{R}^2 \setminus D$,

C2. $p \in \Gamma$.

Calculation steps are omitted due to the lack of the space. Details can be found in Kozłowski [2009]. In case C1 the time derivative of the Lyapunov function fulfills the following equation:

$$\frac{dV_l}{dt} \leq -e^T Q e + \|e\| \|\dot{p}_d\| [\sin^2(e_\alpha)]$$

where $Q = \text{diag}\{k_2 \cos^2(e_\alpha), k_p \cos^2(e_\alpha)\}$ and $\alpha = \angle(e, \dot{p}_d)$. Using (18) one can easily show that function $W$ is positive definite if:

$$\|e\| > \frac{1}{\lambda_{\text{min}}(Q)} [\|\dot{p}_d\| \sin e_\alpha (\sin e_\alpha - \sin \alpha \cos e_\alpha)],$$

where $\lambda_{\text{min}}(Q)$ is the smallest eigenvalue of matrix $Q$. Taking into account that $e_\alpha$ converges exponentially to zero (16) one can conclude that

$$\lim_{t \to \infty} e(t) = 0.$$
reference orientation generated according to model (1). The above result together with (16) leads to:
\[
\lim_{e \to 0} (\theta_a(e) - \theta_d) = 0 \quad \Rightarrow \quad \lim_{t \to \infty} e_\theta(t) = 0.
\]

4. EXPERIMENTS

4.1 Experimental set-up

Experimental tests were carried out using the MTV3 Jedwabny [2004] differentially-driven mobile robot presented in Fig. 3. Three LED markers were mounted on the vehicle top for use with the vision localization system employed in the control scheme as illustrated in Fig. 3. All the control blocks of the proposed algorithm were implemented on an external PC computer (platform control level). The desired angular wheel velocities \( \omega_d \) computed on the platform control level were transmitted through the radio link to the PI regulatory loops working on the vehicle board (drive control level). The feedback rate of the platform control level with vision feedback was equal to 45 Hz; the drive control level worked at the rate of 512 Hz.

4.2 Experiment 1 - collision avoidance with a single obstacle

In Figs. 4-7 experimental results for robot in an environment with a single obstacle are shown. The values of the control parameters are as follows: \( k_1 = 2, \ k_p = 1, \)
\( \omega_{w\text{max}} = 10 \text{rad/s} \). An obstacle with radius \( r = 0.1 \text{m} \) is placed at position \( P_{O1} = (0.48, -0.3) \) and the radius of the APF is \( R = 0.3 \text{m} \). In Fig. 4 time graphs of the position error components, real and auxiliary orientation errors are presented. The discontinuity at \( t \geq 5 \text{s} \) is caused by switching on the desired trajectory when the robot leaves the APF area of the obstacle. In Fig. 5 time evolution of controls for the platform is shown. Fig. 6 presents the desired and real paths for the robot. It can be seen that after the robot leaves the APF, the desired point on the trajectory jumps some distance. In Fig. 7 time evolution of \( w \) is shown. It is increased as the robot approaches the obstacle. The dashed line represents a logical variable that indicates whether the robot is in \( \Gamma \): it is 0 when the robot is in the area of the obstacle’s APF and 1 otherwise.

To make the presented case difficult the initial position of the robot was chosen near the saddle point. The chattering observed in the transient state is caused by the delays in the control loop. Oscillations of this kind were not observed in simulations. The most time consuming operations are: acquisition of data from the vision system, computation of the robot’s position and sending the controls to the low level motion controllers through the serial link.

4.3 Experiment 2 - collision avoidance with a single obstacle without freezing the desired trajectory

In this subsection the results for the same environment as in the previous case are shown. The control parameters are also the same, however, assumption A1 from definition 1 is removed. There is no stability proof for this case but the presented experiments show that the system works well. In Fig. 8 the position and orientation errors are shown. Compared to the previous case they are smoother. The graphs of control are similar (Fig. 9), but robot leaves set \( \Gamma \) of the obstacle 2 seconds earlier.

4.4 Experiment 3 - collision avoidance with two obstacles

In Figs. 12-13 the results for a more complex environment are presented. The desired trajectory is frozen as in experiment 1. In Fig. 12 the time graphs of position errors are shown. A discontinuity caused by switching on the desired trajectory can be easily observed at \( t \geq 4 \text{s} \). The robot gets to the APF area of both obstacles. Figure 13 presents the position on the \((x, y)\) plane.

4.5 Experiment 4 - collision avoidance with two obstacles without freezing the desired trajectory

In Figs. 14-15 the results for an environment with two obstacles are shown. The desired trajectory is not frozen in case of a collision. In this case a second collision caused multiple changes of the motion direction (Fig. 15). Oscillation can be observed on the position error time graph (Fig. 14). Finally leaving the repel area of the obstacle takes more time as compared to previous case.

4.6 Experiment 5 - collision avoidance with two obstacles - high value of \( \omega_{w\text{max}} \) and \( \omega_{L\text{max}} \)

In this case it is presented how a high value of \( \omega_{w\text{max}} \) parameter influences task performance. In the experiments s limit of 40 rad/s was set. This parameter can be used to tune the control algorithm according to practical abilities of the robot. In the present case the robot’s path approaches closely the boundary of the obstacle (Fig. 17). The value of the repulsion component of the control is increased. The controls (Fig. 16) and \( w \) (Fig. 18) reach high values.

5. CONCLUSIONS

In this paper the VFO control method for environments with obstacles was presented. Proof of stability and convergence for the algorithm was given. The effectiveness of the method was illustrated by a set of experiments. In the near future results for the proposed approach extended to many robots will be published. The authors are working on experimental verification for this case.

REFERENCES


K. Kozłowski, W. Kowalczyk Motion Control for Formation of Mobile Robots in Environment With Obstacles
Fig. 12. Experiment 3: Graph of $e_x$, $e_y$ as a function of time

Fig. 13. Experiment 3: Position on the $(x, y)$ plane

Fig. 14. Experiment 4: Graph of $e_x$, $e_y$ as a function of time

Fig. 15. Experiment 4: Position on the $(x, y)$ plane

Fig. 16. Experiment 5: Linear and angular controls

Fig. 17. Experiment 5: Position on the $(x, y)$ plane

Fig. 18. Experiment 5: Collision avoidance component of control ($w$)


