Journal of Engineering Science and Technology
Vol. 10, No. 1 (2015) 57 - 71
© School of Engineering, Taylor's University

A RECEDING HORIZON CONTROLLER FOR PATH FOLLOWING OF A MOBILE ROBOT WITH ACTUATOR CONSTRAINTS

KIATTISIN KANJANAWANISHKUL

Mechatronics Research Laboratory, Faculty of Engineering, Mahasarakham University,
Kamriang, Kantarawichai, Mahasarakham, 44150, Thailand
Email: kiattisin.k@msu.ac.th

Abstract

In path following control, the forward velocity can be seen as an additional degree of freedom. However, many path following control laws assume that the forward velocity is a constant and actuators are ideal, i.e., no actuator constraints. Unfortunately, these controllers cannot be realized in real-world implementation. Thus, we propose an extension to an existing path following controller in order that the forward velocity can be varied according to the robot's posture errors and the desired forward velocity. This can be done by adopting a receding horizon scheme, where the optimization problem is solved online along some look-ahead time intervals. Furthermore, we treat boundaries of wheel speed and acceleration as constraints in the optimization problem. As a result, a mobile robot can slow down when making a sharp turn and it can speed up when being behind a reference point. Simulation tests have been conducted in order to illustrate the effectiveness of our proposed scheme.

Keywords: Path following, Mobile robots, Receding horizon control, Actuator constraints.

1. Introduction

In this paper, we focus on the path following problem, where a mobile robot is required to converge to and follow a path-parameterized reference. It is unlike the trajectory tracking problem, where a mobile robot is required to track a time-parameterized reference. Nevertheless, both are basic motion control problems of mobile robots in the literature.

In general, trajectory tracking problems are solved by determining control laws that make the robots track predetermined feasible trajectories, i.e., trajectories that specify the time evolution of the position, orientation (spatial dimension), as well as the linear and angular velocities (temporal dimension) [1]. However, the drawback of this method is that the robots' dynamics usually contain complex nonlinear terms and involve parameter uncertainties, making the task of finding a feasible trajectory difficult. In addition, in the presence of large tracking errors, the controller may produce too large control signals to catch up with a time-parameterized reference point. This may lead to poor closed-loop performance. One approach to eliminate such problems is to use a path following controller instead of a trajectory tracking controller, as seen in our simulation results.

The basic idea of the path following controller is that the robot's forward velocity tracks a desired velocity profile, while the controller determines the robot's heading direction to drive it to the desired path without any consideration in temporal specifications. Typically this controller eliminates the aggressiveness of the trajectory tracking controller by forcing convergence to the path in a smooth fashion and control signals are less likely pushed to saturation when compared to trajectory tracking [2]. Although the path following problem has been well studied, it is still the subject of numerous research studies. In particular, it has recently been used to replace the trajectory tracking as it is more suitable for certain applications [1]. Pioneering work in this area can be found in [3, 4].

For the path following problem, assigning a velocity profile to the robot can be an additional task, in which the forward velocity is used as an extra degree of freedom. However, many path-following control laws act only on the angular velocity and let the forward velocity be a constant. In this work, we incorporate a receding horizon (RH) scheme with bounded inputs into the work of Soeanto et al. [5] so that the mobile robot can travel safely and smoothly. We extend it in such a way that the forward velocity is varied according to the robot's posture errors and the desired forward velocity along some look-ahead time intervals. The remarkable advantage of this approach is that future information is utilized to improve system performance because the reference path is known beforehand.

This paper is organized as follows: Section 2 describes the problem of path following and models a path following control law for a mobile robot. An RH algorithm for the forward velocity with wheel input constraints is presented in Section 3. In Section 4, our proposed strategy is validated by simulation. Finally, our conclusions and future work are given in Section 5.

2. Path Following Control

In path following control [1], control laws are designed to steer an object (robot arm, mobile robot, aircraft, etc.) to reach and to follow a geometric path, i.e., a manifold parameterized by a continuous scalar s. This setting is more general than the trajectory tracking problem, in which the path variable s is left as an extra degree of freedom. To determine the path variable s, Micaelli and Samson [3] used a numerical projection from the current state onto the path. This point plays the role of a virtual vehicle that should be tracked by the real vehicle. The main problem of a numerical projection is that singularities occur when the distance to the path is not well-defined. This problem can be solved by explicitly controlling the timing law for s to be tracked along the path.

Usually, the choice of the timing law for the path variable s has the following desired behaviour: When the path following error is large, the virtual vehicle will wait for the real one, when the path error is small, the virtual vehicle will move along the path at the speed close to the desired speed assignment. This behaviour is suitable in practice because it avoids the use of large control signals for large path errors. Diaz del Rio et al. [6] proposed a method called error adaptive tracking, in which the tracking adapts to the errors. They defined the function of $\dot{s}$ as $\dot{s} = g(e)$, where e is the distance error. They also proposed $\dot{s} = g(t, e)$ in order to preserve time determinism of trajectory tracking. Soeanto et al. [5] controlled $\dot{s}$ explicitly by modelling the kinematic equations of motion with respect to the Frenet frame. A virtual vehicle concept was also employed by Egerstedt et al. [7], whose control law ensures global stability by determining the motion of the virtual vehicle on the desired path via a differential equation containing error feedback. Kanjanawanishkul [8] proposed another possibility to obtain optimal motion of the virtual vehicle by using model predictive control (MPC). Recently, Ostafew et al. [9] presented a feedforward, proportional iterative learning control algorithm with a feedback-linearized path-tracking controller to reduce path-tracking errors over repeated traverses along a reference path. However, the forward velocity was adjusted by velocity scheduling rules.

The differential-drive mobile robot addressed in this paper has two identical parallel wheels, which are controlled by two independent motors, on the same axle at a distance $2b$ one from the other as depicted in Fig. 1. Since a receding horizon algorithm uses an explicit model of the plant, which is used to predict the future output behaviour, we consider the following kinematic model (see Fig. 1):

$$\begin{bmatrix} \dot{x}_m \\ \dot{y}_m \\ \dot{\theta}_m \end{bmatrix} = \begin{bmatrix} v_m \cos\theta_m \\ v_m \sin\theta_m \\ \omega_m \end{bmatrix} \tag{1}$$

where $\mathbf{x}_m(t) = [x_m, y_m, \theta_m]^T$ is the state vector in the world frame. v_m and ω_m stand for the forward and angular velocities, respectively.

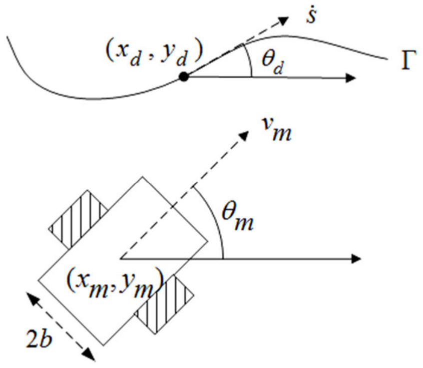


Fig. 1. A graphical representation of a differential-drive mobile robot and a reference path.

In general, we wish to find the control laws of $\dot{s}$ and ω_m such that the robot follows a virtual vehicle with position $\mathbf{x}_d = [x_d, y_d, \theta_d]^T$. The kinematic model of a mobile robot can be formulated with respect to the Frenet frame moving along the reference path. This frame plays the role of the body frame of a virtual vehicle that must be followed by the real robot as depicted in Fig. 1. In the path following problem, we let the forward velocity v_m track a desired velocity profile v_d, while $\dot{s}$ converges to v_m. The error state vector $\mathbf{x}_e$ between the robot state vector $\mathbf{x}_m$ and a virtual vehicle's state vector $\mathbf{x}_d$ can be expressed in the frame of the path coordinate as follows:

$$\begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} = \begin{bmatrix} \cos\theta_d & \sin\theta_d & 0 \\ -\sin\theta_d & \cos\theta_d & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_m - x_d \\ y_m - y_d \\ \theta_m - \theta_d \end{bmatrix} \tag{2}$$

Using Eqs. (1) and (2), the error state dynamic model chosen in a rotated coordinate frame becomes

$$\begin{aligned} \dot{x}_e &= y_e \dot{s}\kappa - \dot{s} + v_m \cos\theta_e \\ \dot{y}_e &= -x_e \dot{s}\kappa + v_m \sin\theta_e \\ \dot{\theta}_e &= \omega_m - \dot{s}\kappa \end{aligned} \tag{3}$$

where κ is the path curvature and $\dot{s}$ is the velocity of a virtual vehicle.

Following [5], we consider the Lyapunov candidate function as follows:

$$V = \frac{1}{2}\left(x_e^2 + y_e^2\right) + \frac{1}{2\gamma}\left(\theta_e - \delta(y_e, v_m)\right)^2 \tag{4}$$

where γ is a positive constant. Note that the function $\delta(y_e, v_m)$ introduced in Eq. (4) has the purpose of shaping the transient convergence of the state of zero and should be treated as an extra degree of freedom. The assumptions for this function are that

(i) $\lim_{t\to\infty} v_m(t) \neq 0$,

(ii) $\delta(0, v_m) = 0$,

(iii) $y_e v_m \sin\delta(y_e, v_m) \leq 0$, $\forall y_e, \forall v_m$.

A possible choice, as suggested in [5], is $-\text{sign}(v_m)\theta_a \tanh y_e$ for some angle θ_a. As a result, $\delta(y_e, v_m)$ does not depend on $|v_m|$, namely that δ is invariant with respect to the absolute value of the robot's forward velocity.

The time derivative of V can be made globally semi-negative definite by using the following control laws (the reader is referred to [5] for more details):

$$\dot{s} = v_m \cos\theta_e + k_1 x_e \tag{5a}$$

$$\omega_m = \dot{\delta} - \gamma y_e v_m \frac{\sin\theta_e - \sin\delta}{\theta_e - \delta} + \kappa\dot{s} - k_2(\theta_e - \delta) \tag{5b}$$

where k_1 and k_2 are positive constants. This results in

$$\dot{V} = -k_1 x_e^2 - \frac{k_2}{\gamma}(\theta_e - \delta)^2 + y_e v_m \sin\delta \leq 0. \tag{6}$$

Closed loop asymptotic global stability of the equilibrium (x_e, y_e, θ_e) = (0, 0, 0) can be shown by invoking LaSalle's Invariant Set Theorem or Barbalat's Lemma [10] to the above Lyapunov candidate function and to the closed loop state equations.

Substituting Eqs. (5a) and (5b) into Eq. (3) yields the following equations of the closed-loop system:

$$\begin{aligned}\dot{x}_e &= -k_1 x_e + k_1 \kappa x_e y_e + y_e \kappa v_m \cos\theta_e \\ \dot{y}_e &= -\kappa x_e v_m \cos\theta_e - k_1 \kappa x_e^2 + v_m \sin\theta_e \\ \dot{\theta}_e &= \dot{\delta} - \gamma y_e v_m \frac{\sin\theta_e - \sin\delta}{\theta_e - \delta} - k_2(\theta_e - \delta)\end{aligned} \tag{7}$$

We incorporate the receding horizon scheme into the closed-loop equations (7) to obtain the forward velocity and to take into account the wheel velocity and acceleration.

The current approach is based on a kinematic model since it is simpler than the dynamic one. In particular, it does not involve numerous parameters associated with the robot and its actuators (geometry of bodies, masses, moments of inertia, etc.) [11]. For many applications, all these terms are not known precisely. Moreover, most commercial available robots allow only the specification of translation and rotation, which we treat as control inputs. However, a control law based on a dynamic model clearly shows higher performance and is more realistic because it reflects the physics of the system. This would form the basis of future work to improve this result.

3. Receding Horizon Control

The general path following problem is characterized by the forward velocity not being part of the control problem opposed to the trajectory tracking problem where a virtual reference vehicle is tracked and both the forward and angular velocities are controlled. Hence, the forward velocity, seen as an extra degree of freedom in this paper, is determined according to the following desired behaviours:

(i) Since, during normal operation, the motors of a mobile robot can, in general, deliver larger velocities than desirable, saturation in the actual actuators (the motors) are not of concern, except when fast heading or turning is required. Thus, to maximize feasible velocities, we have to consider wheel velocities explicitly. Let v_r and v_l denote the right and left wheel velocities, respectively. Discarding wheel slippage and uneven floors, the forward and angular velocities relate to the left and right wheel velocities in the following way

$$v_m = \frac{1}{2}(v_r + v_l) \tag{8a}$$

$$\omega_m = \frac{1}{2b}(v_r - v_l) \tag{8b}$$

Let $v_{\max}^w$ and $v_{\min}^w$ be the maximum and minimum allowable velocities for the left and right wheels (we assume equal constraints on the wheels), i.e.,

$$v_{\min}^w \le v_r \le v_{\max}^w \qquad v_{\min}^w \le v_l \le v_{\max}^w \tag{9}$$

To prevent slippage, we impose the following constraints:

$$a^w_{\min} \le a_r \le a^w_{\max} \qquad a^w_{\min} \le a_l \le a^w_{\max} \tag{10}$$

where the left- and right-hand wheel accelerations are computed as

$$a_r = \frac{v_r(k) - v_r(k-1)}{\Delta} \tag{11}$$

and

$$a_l = \frac{v_l(k) - v_l(k-1)}{\Delta} \tag{12}$$

and $a^w_{\max}$ and $a^w_{\min}$ are the maximum and minimum allowable accelerations, and Δ is the sampling period.

(ii) For sharp turning, the vehicle velocities, forward and angular, must be restricted so that turning is appropriately restrained and smooth. A large forward velocity together with a large angular velocity will jeopardize stability and safety of the robot, for example, for $v_m = 0$, the constraints on the wheels alone may allow an undesirable large angular velocity. Hence, we include the following velocity constraints:

$$v_{\min} \le v_m \le v_{\max} \qquad \omega_{\min} \le \omega_m \le \omega_{\max} \tag{13}$$

The constraints given in Eq. (13) can either be self-imposed due to desired behaviour or safety concerns. Figure 2 illustrates how the constraints (9) and (13) relate and it is assumed that zero belongs to the set of valid velocities. This relation can maximize feasible forward velocity as long as boundaries of wheel speed and acceleration are not violated, resulting in less conservative bounds. Furthermore, the angular velocity is considered a higher priority than the forward velocity, i.e., when the larger angular velocity is required, the forward velocity is forced to be less than the desired velocity.

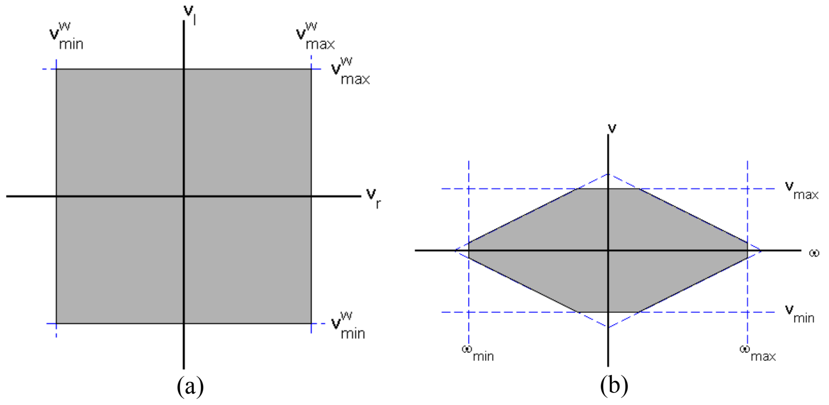


Fig. 2. The velocity constraints: (a) left and right wheel velocities, (b) forward and angular velocities corresponding to (a).

To realize such behaviours mentioned above, we adopt a receding horizon (RH) approach, where the forward velocity is based on predictions of the robot's posture and the reference path.

The conceptual structure of RH control is illustrated in Fig. 3. RH control in the literature is also known as model predictive control (MPC), where an explicit model of the plant is used to predict the future output behaviour. This prediction capability allows computing a sequence of manipulated variable adjustments in order to solve optimal control problems online, where the future behaviour of a plant is optimized over a future horizon, possibly subject to constraints on the manipulated inputs and outputs [12, 13]. The result of the optimization is applied according to a receding horizon philosophy: At time t only the first input of the optimal command sequence is actually applied to the plant. The remaining optimal inputs are discarded, and a new optimal control problem is solved at time $t + \Delta$, where Δ is the sampling period. As new measurements are collected from the plant at each time t, the receding horizon mechanism provides the controller with the desired feedback characteristics.

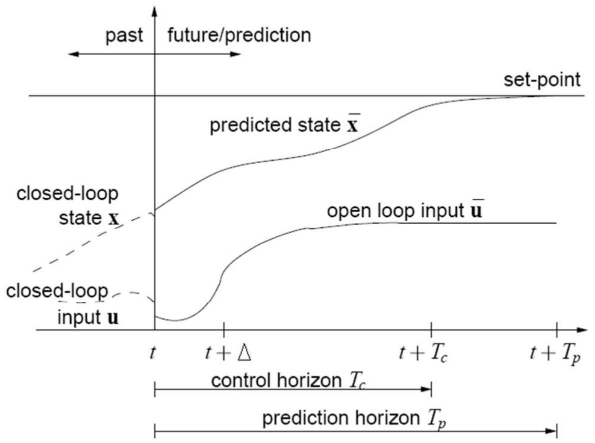

past
future/prediction
set-point
predicted state $\bar{\mathbf{x}}$
closed-loop state $\mathbf{x}$
open loop input $\bar{\mathbf{u}}$
closed-loop input $\mathbf{u}$
t
$t+\Delta$
$t+T_c$
$t+T_p$
control horizon T_c
prediction horizon T_p

Fig. 3. Principle of model predictive control [12].

Since many systems are inherently nonlinear and linear RH is not suitable for such highly nonlinear systems, nonlinear models must be used [12]. However, the major concern in the use of a nonlinear RH algorithm is whether such an open-loop control can guarantee system stability. It is shown that an infinite predictive control horizon can guarantee stability of a system, but the infinite predictive horizon may not be feasible for a nonlinear system in practice [12]. Mayne et al. [13] presented essential principles for the stability of RH control of constrained dynamical systems. Different approaches to attain closed-loop stability using finite horizon lengths exist. We intentionally do not review published contributions because of a large number of publications. We refer the reader to [12, 13].

Although RH control is apparently not a new control method, works dealing with an RH algorithm of path following problems are rare. A comprehensive survey paper related to RH control for a mobile robot can be found in [14].

A nonlinear system is normally described by the following nonlinear differential equation:

$$\dot{\mathrm{x}}(t) = f(\mathrm{x}(t), \mathrm{u}(t)) \tag{14}$$

subject to: $\mathrm{x}(t) \in X$, $\mathrm{u}(t) \in U$, $\forall t \geq 0$

where $\mathrm{x}(t) \in R^n$, $\mathrm{u}(t) \in R^m$ are the n dimensional state vector and the m dimensional input vector of the system, respectively. $X \subseteq R^n$ and $U \subseteq R^m$ denote the set of feasible states and inputs of the system, respectively. The input applied to the system is given by the solution of the following finite horizon open-loop optimal control problem, which is solved at each time instant:

$$\min_{\mathrm{u}(\cdot)} \int_t^{t+T_p} F(\bar{\mathrm{x}}(t), \bar{\mathrm{u}}(t)) d\tau + V(\bar{\mathrm{x}}(t+T_p)) \tag{15}$$

subject to:

$$\dot{\bar{\mathrm{x}}}(\tau) = \mathrm{f}(\bar{\mathrm{x}}(\tau), \bar{\mathrm{u}}(\tau)) \tag{16a}$$

$$\bar{\mathrm{u}}(\tau) \in U \, , \; \forall t \in [t, t+T_c] \tag{16b}$$

$$\bar{\mathrm{x}}(\tau) \in X \, , \; \forall t \in [t, t+T_p] \tag{16c}$$

$$\bar{\mathrm{x}}(t+T_p) \in \Omega \tag{16d}$$

The bar denotes an internal controller variable. T_p represents the length of the prediction horizon or output horizon, and T_c denotes the length of the control horizon or input horizon ($T_c \leq T_p$). When $T_p = \infty$, we refer to this as the infinite horizon problem, and similarly, when T_p is finite, as a finite horizon problem. $V(\bar{\mathrm{x}}(t+T_p))$ is the terminal penalty and Ω is the terminal region.

The robot's input constraints in Eq. (16b) include Eqs. (9), (10), and (13). In this paper, the desired velocity must primarily satisfy high-level tasks. However, with our proposed RH scheme, the robot will be forced to slow down at sharp turns and to speed up if it is behind the reference point.

The cost function for Eq. (15) is, in general, as follows:

$$F(\bar{\mathrm{x}}, \bar{\mathrm{u}}) = \bar{\mathrm{x}}^T Q \bar{\mathrm{x}} + \bar{\mathrm{u}}^T R \bar{\mathrm{u}} \tag{17}$$

where the deviation from the desired values is weighted by the positive definite matrices Q, and R. To achieve our objective, we select the following cost function:

$$F(\mathrm{x}, \mathrm{u}) = q_{11} x_e^2 + q_{22} y_e^2 + q_{33} (\theta_e - \delta)^2 + q_{44} \eta_e^2 \tag{18}$$

where $\eta_e = v_m - v_d$, matrix $Q = \mathrm{diag}(q_{11}, q_{22}, q_{33}, q_{44})$, and let matrix R be equal to 0.

To guarantee the control stability, the following Lyapunov function, similar to [15], can be selected as the terminal penalty in Eq. (13) (note that the subscript T denotes the terminal state)

$$V(\mathrm{x}(t+T_p)) = \frac{1}{2} x_{eT}^2 + \frac{1}{2} y_{eT}^2 + \frac{1}{2\gamma} (\theta_{eT} - \delta)^2 + \frac{1}{2} \eta_{eT}^2 \tag{19}$$

under the terminal-state controller: $a_m = \dot{v}_m = -\alpha\eta_{eT}$, where $\alpha > 0$, such that the following condition is satisfied:

$$\dot{V}(\mathrm{x}(t)) + F\big(t, \mathrm{x}_e(t), \mathrm{u}_e(t)\big) \le 0, \qquad \forall \mathrm{x}(t) \in \Omega \tag{20}$$

All weight parameters have to be selected such that Eq. (20) is satisfied, i.e., the stability condition becomes

$$\begin{aligned} \dot{V} + F &= x_{eT}\dot{x}_{eT} + y_{eT}\dot{y}_{eT} + \eta_{eT}\dot{\eta}_{eT} + (\theta_{eT} - \delta)(\dot{\theta}_{eT} - \dot{\delta}) + F \\ &= -k_1 x_{eT}^2 - \frac{k_2}{\gamma}(\theta_{eT} - \delta)^2 + y_{eT} v_m \sin\delta - \alpha\eta_{eT}^2 + q_{11} x_{eT}^2 \\ &\quad + q_{22} y_{eT}^2 + q_{33}(\theta_{eT} - \delta)^2 + q_{44}\eta_{eT}^2 \le 0 \end{aligned} \tag{21}$$

for all $\mathrm{x} \in \Omega$.

Similar to [15], the following conditions for the weight parameters are required to have a negative derivative of the value function:

$$-k_1 + q_{11} + q_{22} \le 0 \tag{22a}$$

$$-\frac{k_2}{\gamma} + q_{33} \le 0 \tag{22b}$$

$$-\alpha + q_{44} \le 0 \tag{22c}$$

and the terminal-state region is defined as follows:

$$|x_{eT}| \ge |y_{eT}| \tag{23}$$

However the hard constraint of the terminal-state region (23) may lead to the infeasible solution, the slack variable ε is introduced to soften the constraint. Note that, unlike [15], where the control laws were designed for trajectory tracking, our proposed controller is used for a path following task.

4. Simulation Results

Our RH scheme incorporated into the path following controller has been evaluated through the MATLAB simulation. The controller algorithm has been implemented as follows:

```
Step 1. Initialize all system parameters
Step 2. Update robot states and path states
Step 3. Use fmincon function to solve the optimization problem (15) with the
        following slack variable ($\varepsilon$):
            $\min_{\mathrm{u}(\cdot)}\left(\int_t^{t+T_p} F(\bar{\mathrm{x}}(t), \bar{\mathrm{u}}(t))\,d\tau + V(\bar{\mathrm{x}}(t+T_p)) + \varepsilon^2\right)$
        subject to Eqs. (7), (9)-(13), and (23) with the following slack variable ($\varepsilon$):
            $|y_{eT}| - |x_{eT}| - \varepsilon < 0$
            $\varepsilon > 0$
        to obtain $v_m$
Step 4. Use Eqs. (5a) and (5b) to get $\dot{s}$ and $\omega_m$
Step 5. Go to Step 2, unless the path following task is completed
```

Note that fmincon is a function which finds minimum of a constrained nonlinear multivariable function, also known as constrained nonlinear optimization.

The following eight-shaped curve is selected as a reference path because its geometrical symmetry and sharp changes in curvature make the test challenging:

$$x_d(t) = 1.8\sin(0.1t) \tag{24a}$$

$$y_d(t) = 1.2\sin(0.2t) \tag{24b}$$

where t is time in case of trajectory tracking, while this reference is numerically parameterized by the path variable s in case of the path following problem. All the parameters of our framework are set as follows:

Q = diag(0.1, 1, 0.1, 1), $T_c = T_p = 0.15$ s, $\Delta = 0.05$ s,

$v_d = 0.2$ m/s, $s(0) = 0$ m, $\theta_a = \pi/4$ rad,

$k_1 = 1.1$, $k_2 = 1$, $\gamma = 10$, $\alpha = 1$, $b = 0.15$ m,

$v_{\max} = 0.25$ m/s, $v_{\min} = -0.25$ m/s, $\omega_{\max} = 0.5$ rad/s, $\omega_{\min} = -0.5$ rad/s,

$v^w_{\max} = 0.25$ m/s, $v^w_{\min} = -0.25$ m/s, $a^w_{\max} = 1$ m/s^2, $a^w_{\min} = -1$ m/s^2.

The chosen values in Q result in a relatively large penalty in the states y_e and η_e. This means that if y_e or η_e is large, the large values in Q will amplify the effect of y_e and η_e in the optimization problem. For time horizon, the longer T_c and T_p improve performance but it leads to an increase of online computation. Thus, T_c and T_p must be chosen to compromise between performance and computation. v_d is our desired forward velocity, while b is a distance from one wheel to the centre of the robot. θ_a, k_1, k_2, γ, and α are chosen to ensure system stability (22) and system performance. The rest of the parameters are the actuator constraints of the robot. Note that for visualization the circles in all figures below are snapshots of robot location at every 2.5 s and the robot trajectories are shown as dashed lines.

Several path following controllers in the literature act only on the angular velocity, while the forward velocity is a constant. This situation leads to two problems. One is the control signals become saturated at the sharp turn, resulting in large lateral errors. The other is the problem of slipping off the path. Thus, in the first simulation test, we show that our RH scheme can generate a sequence of forward velocities according to the desired behaviours without constraint violation.

We compare our approach to the work of Soeanto et al. [5]. Although they developed a nonlinear adaptive control law dealing with vehicle dynamics, most commercial available robots do not allow the direct control of forces or torques. Thus, the controller (5a) and (5b) with the kinematic model is implemented for a comparison purpose. It is important to note that control signals of this controller are not bounded, leading to impractical implementation. One possibility is to apply the technique of bounded low-level velocity constraints [16]. A saturation of the command velocities that preserve the current curvature $\kappa = \omega_m/v_m$ is performed as

$$\sigma = \max\{|v_m|/v_{\max}, |\omega_m|/\omega_{\max}, 1\} \tag{25}$$

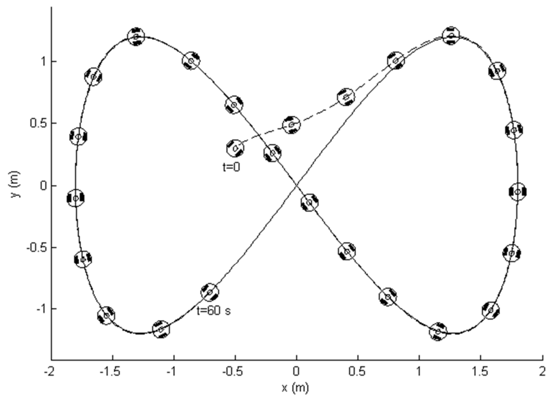
where the actual command velocity v_c and ω_c stand for

$$v_c = \text{sign}(v_m)v_{\max},\ \omega_c = \omega_m/\sigma; \quad \text{if} \quad \sigma = |v_m|/v_{\max}$$

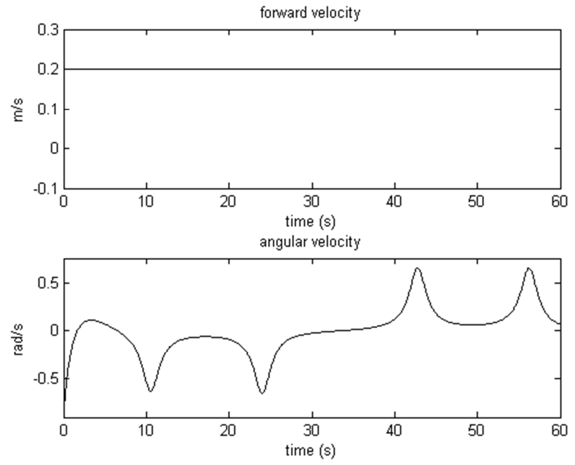
$$v_c = v_m/\sigma,\ \omega_c = \text{sign}(\omega_m)\omega_{\max};\quad \text{if}\quad \sigma = |\omega_m|/\omega_{\max}$$

$$v_c = v_m,\ \omega_c = \omega_m;\quad \text{if}\quad \sigma = 1.$$

As seen in Fig. 4(a), even though the path deviations are negligible, this controller cannot be realized since control signals exceed the input boundaries ($\omega_{\max}$ = 0.5 rad/s), as shown in Fig. 4(b). However, when bounded low-level velocity techniques are applied, the path deviations are not evitable (see Figs. 5 and 9). As obviously seen in Figs. 6 and 9, our RH scheme can reduce the posture errors because the robot moves smoothly at lower velocity (see Fig. 7) without wheel speed and acceleration violation (see Fig. 8) while it is making a sharp turning. However, it is important to note that an initial posture of the robot should be close to the reference path in order to avoid infeasible solutions.



(a)



(b)

Fig. 4. The path following results using the controller proposed by Soeanto et al. [5]: (a) with unbounded control signals, (b) the velocity profiles corresponding to (a).

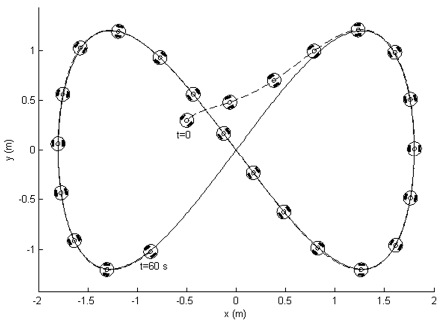


Fig. 5. Path following results using the controller proposed by Soeanto et al. [5] with bounded low-level velocity techniques.

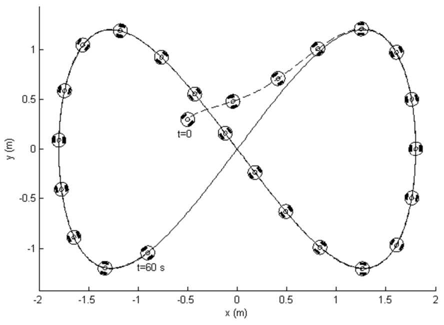


Fig. 6. Path following results with $v_d = 0{:}2$ m/s and $T_p = 0.15$ s using our RH strategy.

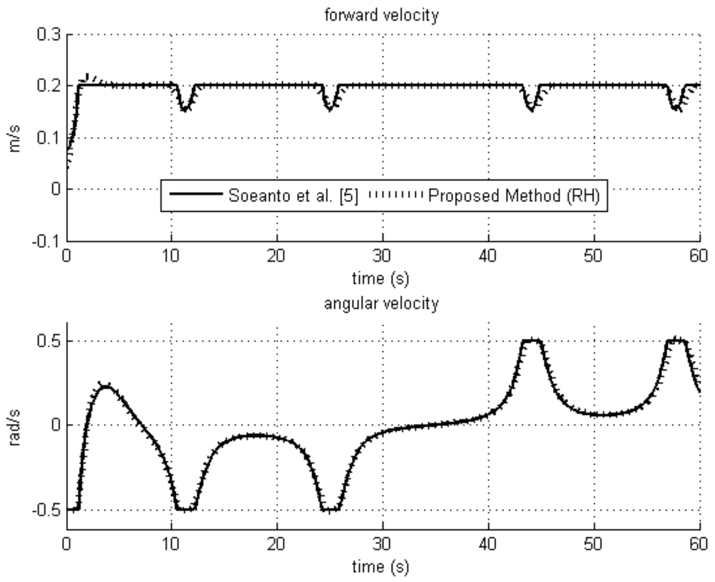


Fig. 7. Comparison results of velocity profiles corresponding to Figs. 5 and 6.

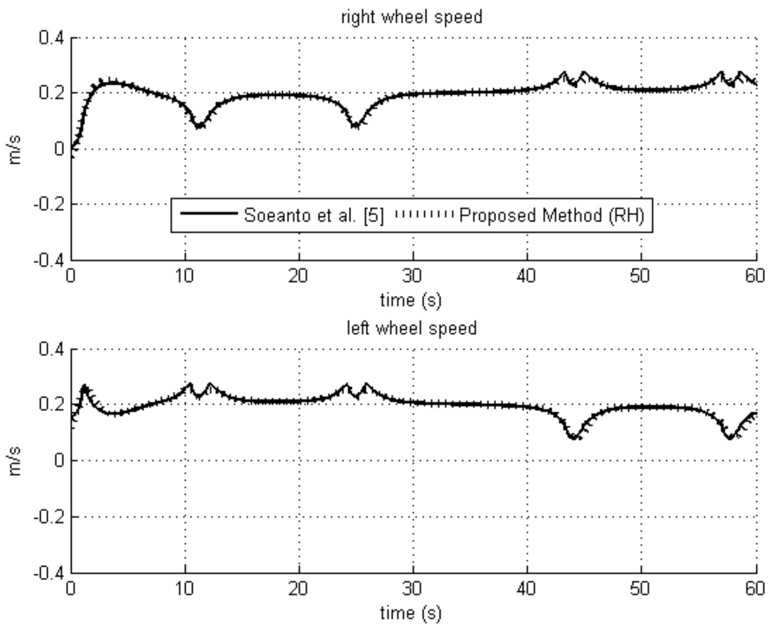


Fig. 8. Comparison results of left and right wheel speed corresponding to Figs. 5 and 6.

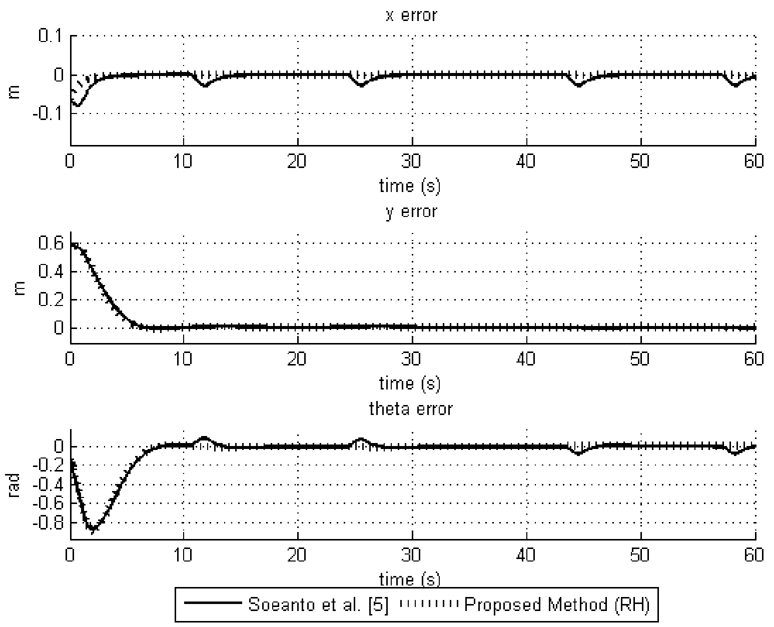


Fig. 9. Comparison results of robot's posture errors corresponding to Figs. 5 and 6.

5. Conclusions and Future Work

In this paper, we explored the path following problem of a mobile robot, where the objective is to be on the path rather than at a certain point at a specific time. Even though this problem draws less attention in the literature, it offers some remarkable advantages over trajectory tracking in some cases.

We applied a receding horizon (RH) scheme to the path following control proposed by Soeanto et al. [5] to satisfy the following objectives: (i) path following control with optimal forward velocity and (ii) bounded wheel control signals. The RH algorithm is used to produce a sequence of forward velocity along some look-ahead time intervals by taking into account input boundaries and stability constraints. It predicts the posture of the robot together with knowledge of an upcoming turning to compensate the control signal, appropriately. As a result, the mobile robot can turn at sharp corners smoothly. Likewise, the robot can travel safely at high speed with the full range of wheel speed.

Currently, we are building a real mobile robot which can be used to validate our control law in real-world environments. In addition, we will extend our controller to accomplish path following tasks in dynamic environments with static and moving obstacles.

References

1. Aguiar, A.P.; Dacic, D.B.; Hespanha, J.P.; and Kokotovic, P. (2004). Path-following or reference-tracking? An answer relaxing the limits to performance. *Proceedings of the IFAC/EURON Symposium on Intelligent Autonomous Vehicles*, Lisbon, Portugal.
2. Al-Hiddabi, S.A.; and McClamroch, N.H. (2002). Tracking and maneuver regulation control for nonlinear non-minimum phase systems: application to flight control. *IEEE Transactions on Control Systems Technology*, 10(6), 780-792.
3. Micaelli, A.; and Samson, C. (1993). Trajectory-tracking for unicycle-type and two-steering-wheels mobile robots, INRIA Sophia-Antipolis, Tech. Rep. 2097.
4. de Wit, C.C.; Samson, C.; Khennouf, H.; and Sordalen, O.J. (1993). Nonlinear Control Design for Mobile Robots, Zheng, Y.F. (Ed.), *Recent Trends in Mobile Robots*, World Scientific Publishing, Singapore.
5. Soeanto, D.; Lapierre, L.; and Pascoal, A. (2003). Adaptive non-singular path-following, control of dynamic wheeled robots. *Proceedings of the International Conference on Advanced Robotics*, Coimbra, Portugal, 1387-1392.
6. Diaz del Rio, F.; Moreno, G.J.; Ramos, J.L.S.; Rodriguez, C.A.A.; and Balcells, A.A.C. (2002). A new method for tracking memorized paths: application to unicycle robots. *Proceedings of the 10th IEEE Mediterranean Conference on Control and Automation*, Lisbon, Portugal.
7. Egerstedt, M.; Hu, X.; and Stotsky, A. (2001). Control of mobile platforms using a virtual vehicle approach. *IEEE Transactions on Automatic Control*, 46(11), 1777-1782.

8. Kanjanawanishkul, K. (2013). Path following control of a mobile robot using contractive model predictive control. *Applied Mechanics and Materials*, 397 -400(2013), 1366-1372.
9. Ostafew, C.; Schoellig, A.; and Barfoot, T. (2013). Visual teach and repeat, repeat, repeat: iterative learning control to improve mobile robot path tracking in challenging outdoor environments. *Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Tokyo, Japan, 176-181.
10. Khalil, H. K. (2002). *Nonlinear systems*. Prentice Hall, New Jersey.
11. Morin, P.; and Samson, C. (2008). *Motion control of wheeled mobile robot,* Siciliano, B. and Khatib, O. (Eds.), *Springer Handbook of Robotics*, Springer Berlin Heidelberg.
12. Allgower, F.; Findeisen, R.; and Nagy, Z.K. (2004). Nonlinear model predictive control: from theory to application. *Journal of Chinese Institute of Chemical Engineers*, 35(3), 299-315.
13. Mayne, D.Q.; Rawlings, J.B.; Rao, C.V.; and Scokaert, P.O.M. (2000). Constrained model predictive control: stability and optimality. *Automatica*, 36(6), 789–814.
14. Kanjanawanishkul, K. (2012). Motion control of a wheeled mobile robot using model predictive control: A survey. *KKU Research Journal*, 17(5), 811-837.
15. Gu, D.; and Hu, H. (2006). Receding horizon tracking control of wheeled mobile robots. *IEEE Transactions on Control Systems Technology*, 14(4), 743-749.
16. Oriolo, G.; Luca, A.D.; and Vendittelli, M. (2002). WMR control via dynamic feedback linearization: design, implementation and experimental validation. *IEEE Transactions on Control Systems Technology*, 10(6), 835-852.