CrossMark

Tracking Error Learning Control for Precise Mobile Robot Path Tracking in Outdoor Environment

Erkan Kayacan[1] (iD) · Girish Chowdhary[2]

Abstract

This paper presents a Tracking-Error Learning Control (TELC) algorithm for precise mobile robot path tracking in off-road terrain. In traditional tracking error-based control approaches, feedback and feedforward controllers are designed based on the nominal model which cannot capture the uncertainties, disturbances and changing working conditions so that they cannot ensure precise path tracking performance in the outdoor environment. In TELC algorithm, the feedforward control actions are updated by using the tracking error dynamics and the plant-model mismatch problem is thus discarded. Therefore, the feedforward controller gradually eliminates the feedback controller from the control of the system once the mobile robot has been on-track. In addition to the proof of the stability, it is proven that the cost functions do not have local minima so that the coefficients in TELC algorithm guarantee that the global minimum is reached. The experimental results show that the TELC algorithm results in better path tracking performance than the traditional tracking error-based control method. The mobile robot controlled by TELC algorithm can track a target path precisely with less than 10 cm error in off-road terrain.

Keywords Learning control · Mobile robot · Path tracking · Tracking error

1 Introduction

Mobile robots in off-road terrain are increasingly used in wide range of nonindustrial applications such as agriculture, search and rescue, military, forestry, mining and security [12, 17]. However, guidance, navigation, and control of mobile robots require advanced control methods to alleviate the effects of unmodeled surface and soil conditions (e.g., snow, sand, grass), terrain topography (e.g., side-slopes, inclines), and complex robot dynamics. In the outdoor environment, it is sometimes arduous and/or almost impossible to obtain a priori model for such effects (i) modeling of robot-terrain interactions is challenging, (ii) the soil condition is often not known ahead of time, and (iii) the identification of system parameters is a cumbersome process and must be re-carried out for different terrains [3, 8, 10, 15, 22].

The initial studies on autonomous mobile robots used traditional controllers, e.g., proportional-integral-derivative, optimal, and model predictive controllers (MPCs) [1, 5, 6, 21]. Proportional-integral-derivative controllers are convenient only for single-input-single-output systems; however, mobile robots are multi-input-multi-output systems. As alternative methods, linear quadratic regulators and linear MPCs, which can be designed readily for multi-input-multi-output systems, were proposed for autonomous navigation of mobile robots in literatur [18, 23, 26]. Since these methods require a linear system model to be designed, tracking error-based control algorithms were developed [16, 24] in which the system model is linearized around the target path, and the total control input is obtained by the sum of feedback and feedforward control inputs [2]. The feedback control law is designed based on a nominal model, which is a priori model and cannot capture all the effects of uncertainties that are summarized in the previous paragraph. Moreover, the feedforward control law is derived

✉ Erkan Kayacan
erkank@mit.edu

[1] Senseable City Laboratory and Computer Science & Artificial Intelligence Laboratory Massachusetts Institute of Technology, Cambridge, MA 02139, USA

[2] Coordinated Science Laboratory and Distributed Autonomous Systems Laboratory, University of Illinois at Urbana-Champaign, Champaign, IL, USA

Springer

taking the target path and nominal model into account so that it also cannot contain the effects of uncertainties. Since tracking error-based models might not represent real-time systems behavior accurately, the traditional tracking error-based control methods cannot ensure precise path tracking performance in the outdoor environment. Therefore, it can be concluded that a prerequisite for accurate tracking performance of model-based controllers is the achievement of a precise mathematical model of the system to be controlled [13]. This shows that traditional approaches are not always inherently robust. Moreover, an amplitude-saturated output feedback control approach was proposed in [25]. In this approach, the output of the control input is limited by upper and lower bounds; however, the input rate cannot be limited. Furthermore, this approach requires to know the direct relation between the input and output, which is known in tracking error-based control methods. In this paper, since traditional tracking error-based controllers use an MPC as a feedback controller, we also use an MPC controller for a fair comparison with the previous works in literature.

To cope up with restrictions on traditional controllers, adaptive control approaches and MPC scheme with friction compensation were respectively proposed in [19, 20], and successful results were reported for indoor applications. However, there is no evidence that these methods work well for outdoor applications where the uncertainty is so high, and soil conditions are changing. Robust trajectory tracking error model-based predictive controller was designed for unmanned ground vehicles to overcome the limitations of the tracking error-based controllers [14]. Although it exhibited robust control performance, it could not ensure precise path tracking performance, e.g., the tracking error was more than 20 cm. Moreover, learning-based nonlinear MPC algorithms were proposed in which online parameters estimators were used to update the system parameters in the system model for an articulated unmanned ground vehicle [7, 9, 11]. Although precise path tracking performance was reported in these studies, the required computation times is large, especially for embedded applications. Therefore, the purpose of this paper is to develop a high precision control algorithm for mobile robots, which must be computationally efficient and can learn the mobile robot dynamics by utilizing longitudinal and lateral error dynamics. Thus, the feedforward control action will be in charge of the overall control of the mobile robot in steady-state behavior.

The main contribution of this study beyond state of the art is that a novel Tracking Error Learning Control (TELC) scheme is developed and implemented in real-time for the first time in literature. The first contribution of this paper is that the cost functions consisting of tracking error dynamics of the mobile robot are used to update the coefficients in the feedforward control law. Hence, TELC can learn mobile robot dynamics, and the feedback control action is removed from the overall control signal when the robot is on-track. Therefore, the model-plant mismatch problem for outdoor applications cannot deteriorate the path tracking performance in the TELC scheme. The second contribution is that the stability of the TELC algorithm is proven that the TELC algorithm is asymptotically stable. The stability analysis shows the longitudinal and lateral error dynamics converge to zero if the learning coefficients are large enough. The third contribution is that it is proven that the TELC algorithm does not have any local minima so that it can reach the global minima. Moreover, it is shown that the feedforward control actions are bounded for a finite value for the coefficients in steady-state. Along with the theoretical results, this paper also presents path tracking-test results of the presented TELC algorithm on a mobile robot. TELC algorithm results in precise mobile robot path tracking performance when compared to the traditional tracking error-based control method.

This paper is organized as follows: The tracking error-based model is derived in Section 2. The traditional tracking error-based control method is given in Section 3. The TELC algorithm is formulated in Section 4 while the update rules for the linear and angular velocity references are respectively derived in Sections 4.1 and 4.2, and the stability analysis is given in Section 4.3. Experimental results on a mobile robot are presented in Section 5. Finally, a brief conclusion of the study is given in Section 6.

2 Tracking Error-Based System Model

In this paper, the mobile robot is illustrated in Fig. 1. The velocities of two driven wheels result in linear velocity $v = (v_l + v_r)/2$ and angular velocity $\omega = (v_l - v_r)/L$ with the distance between wheels L, which are two control inputs of the mobile robot, $\mathbf{u} = [v, \omega]$. The traditional unicycle model is written for a mobile robot as follows:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v \cos(\theta) \\ v \sin(\theta) \\ \omega \end{bmatrix} \tag{1}$$

where x and y are the position, θ is the heading angle, v is the linear velocity, ω is the angular velocity of the mobile robot.

The target path with respect to the inertial reference frame fixed to the motion ground is defined by a reference state vector $\mathbf{q}_r = (x_r, y_r, \theta_r)^T$ and a reference control vector $\mathbf{u}_r = (v_r, \omega_r)^T$. The error state $\mathbf{e} = [e_1, e_2, e_3]^T$ expressed in the frames on the mobile robot is written as:

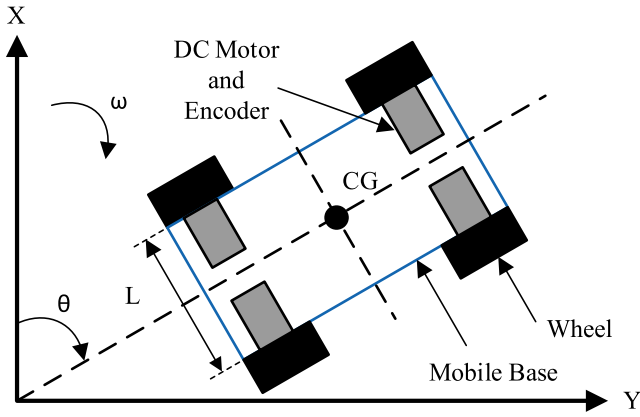$$\mathbf{e} = \mathbf{T}(\theta) \times [\mathbf{q}_r - \mathbf{q}] \tag{2}$$

Fig. 1 Schematic illustration of the mobile robot

where $\mathbf{T}(\theta)$ is the transformation matrix formulated as below:

$$\mathbf{T}(\theta) = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The tracking error-based model is derived by taking the time-derivative of the error state in (2) and the unicycle model in (1) into account as follows:

$$\begin{aligned} \dot{e}_1 &= \omega e_2 - v + v_r \cos(e_3) \\ \dot{e}_2 &= -\omega e_1 + v_r \sin(e_3) \\ \dot{e}_3 &= \omega_r - \omega \end{aligned} \tag{3}$$

where e_1 is the longitudinal error, e_2 is the lateral error and e_3 is the heading angle error.

The tracking error-based model (3) is linearized around the target path ($e_1 = e_2 = e_3 = 0$) as follows:

$$\begin{aligned} \dot{e}_1 &= \omega_r e_2 - v + v_r \\ \dot{e}_2 &= -\omega_r e_1 + v_r e_3 \\ \dot{e}_3 &= \omega_r - \omega \end{aligned} \tag{4}$$

Finally, it can be written in the state-space form as follows:

$$\begin{aligned} \dot{\mathbf{e}} &= \mathbf{A}\mathbf{e} + \mathbf{B}\mathbf{u}_e \\ \dot{\mathbf{e}} &= \begin{bmatrix} 0 & \omega_r & 0 \\ -\omega_r & 0 & v_r \\ 0 & 0 & 0 \end{bmatrix} \mathbf{e} + \begin{bmatrix} -1 & 0 \\ 0 & 0 \\ 0 & -1 \end{bmatrix} \mathbf{u}_e \end{aligned} \tag{5}$$

where the state and control vectors are written as

$$\mathbf{e} = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}^T \tag{6}$$

$$\mathbf{u}_e = \begin{bmatrix} v_e & \omega_e \end{bmatrix}^T \tag{7}$$

where $v_e = v - v_r$ and $\omega_e = \omega - \omega_r$.

Remark 1 The tracking error-based system model is fully controllable when either the linear velocity reference v_r or the angular velocity reference ω_r is nonzero, which is a sufficient condition.

3 Traditional Tracking Error-Based Control

The traditional tracking-error-based control algorithm for a mobile robot is formulated in this section. The total control input applied to the mobile robot is calculated as the summation of the feedback control action $\mathbf{u}_b$ and the feedforward control action $\mathbf{u}_f$ as follows:

$$\mathbf{u} = \mathbf{u}_b + \mathbf{u}_f \tag{8}$$

The feedback controller generates the differences between the actual and reference control inputs while the feedforward control actions are the reference control inputs. The traditional feedback and feedforward control actions are formulated in following subsections.

3.1 Feedback Control Action: Model Predictive Control

A mobile robot can be described by a linear continuous-time model:

$$\dot{\mathbf{e}}(t) = \mathbf{A}\mathbf{e}(t) + \mathbf{B}\mathbf{u}_e(t) \tag{9}$$

where $\mathbf{e} \in \mathbb{R}^3$ is the state vector and $\mathbf{u}_e(k) \in \mathbb{R}^2$ is the control input vector. The matrices $\mathbf{A}$ and $\mathbf{B}$ are found from the tracking error-based system model in (5).

The input constraints for the mobile robot are defined for all $t \geq 0$ as follows:

$$-0.1 \text{ m/s} \leq v_e(t) \leq 0.1 \text{ m/s} \tag{10}$$

$$-0.1 \text{ rad/s} \leq \omega_e(t) \leq 0.1 \text{ rad/s} \tag{11}$$

The cost function is written as follows:

$$J(\Delta\mathbf{U}, \mathbf{e}) = \frac{1}{2}\left\{ \sum_{i=k+1}^{k+N_p} \|\mathbf{e}(t_i)\|_Q^2 + \sum_{i=k+1}^{k+N_c} \|\Delta\mathbf{u}_e(t_i)\|_R^2 \right\} \tag{12}$$

where $N_p = 20$ and $N_c = 5$ represent the prediction and control horizons, $\Delta\mathbf{u}_e$ is the input change, and $\Delta\mathbf{U} = [\Delta\mathbf{u}_e^T(t_k), ..., \Delta\mathbf{u}_e^T(t_{k+N_c})]^T$ is the matrix of the input vectors from sampling instant t_k to sampling instant t_{k+N_c}. Since the sampling time of the experiments is equal to 200 millisecond, the prediction and control horizons are respectively equal to 4 second and 1 second. The positive-definite weighting matrices $Q^{3\times3}$ and $R^{2\times2}$ are defined as follows:

$$Q = diag(1, 1, 1) \quad \text{and} \quad R = diag(1, 1) \tag{13}$$

The following plant objective function is solved at each sampling time for the MPC:

$$\min_{\mathbf{e}(.),\mathbf{u}_e(.)} \quad \frac{1}{2}\left\{ \sum_{i=k+1}^{k+N_p} \|\mathbf{e}(t_i)\|_Q^2 + \sum_{i=k+1}^{k+N_c} \|\Delta\mathbf{u}_e(t_i)\|_R^2 \right\}$$

$$\text{subject to} \quad \mathbf{e}(t_k) = \hat{\mathbf{e}}(t_k)$$
$$\dot{\mathbf{e}}(t) = \mathbf{A}\mathbf{e}(t) + \mathbf{B}\mathbf{u}_e(t) \quad t \in [t_{k+1}, t_{k+N_p}]$$
$$-0.1 \leq v_e(t) \leq 0.1 \quad t \in [t_{k+1}, t_{k+N_c}]$$
$$-0.1 \leq \omega_e(t) \leq 0.1 \quad t \in [t_{k+1}, t_{k+N_c}]$$
$$(14)$$

The convex optimization problem in (14) is solved for the current error state information $\hat{\mathbf{e}}(t_k)$ in a receding horizon fashion. The steps for the implementation of the MPC algorithm are summarized as below:

1. Measure or estimate the current error states $\hat{\mathbf{e}}(t)$.
2. Obtain $\Delta\mathbf{U}^* = [\Delta\mathbf{u}_e^*(t_{k+1}), \ldots, \Delta\mathbf{u}_e^*(t_{k+N_c})]^T$ by solving the optimization problem in (14).
3. Calculate the feedback control action $\mathbf{u}_e^*(t_{k+1}) = \Delta\mathbf{u}_e^*(t_{k+1}) + \mathbf{u}_e^*(t_k)$

The MPC problem is thereafter solved for the next sampling instant over shifted prediction and control horizons. The control input generated by the MPC $\mathbf{u}_e^*$ is the feedback control action $\mathbf{u}_b$:

$$\mathbf{u}_b = \mathbf{u}_e^* \tag{15}$$

In tracking error control scheme, the designed MPC minimizes the tracking error between the target path and the measured position of the mobile robot, and finds the differences between the actual and reference control inputs. Therefore, the feedback control inputs generated by the MPC are not the actual control inputs, which are sent to the mobile robot. We will formulate traditional feedforward control actions in the next Section 3.2.

3.2 Feedforward Control Action

In open-loop control, the feedforward control laws can only drive a mobile robot on a target path if there exist no uncertainties, and initial state errors. Feedforward control inputs v_r and ω_r are derived for a given target path (x_r, y_r) by using the unicycle model (1).

The linear velocity reference, i.e., v_r, for a mobile robot is derived for a target path (x_r, y_r) defined in a time interval $t \in [0, T]$ as follows:

$$v_r = \pm\sqrt{(\dot{x}_r)^2 + (\dot{y}_r)^2} \tag{16}$$

where the sign $\pm$ is the desired driving direction of the mobile robot (+ for forward, − for reverse).

The heading angle reference is derived from (1) as follows:

$$\theta_r = \arctan 2(y_r, x_r) + \gamma\pi \tag{17}$$

where $\gamma = 0, 1$ is the desired driving direction of the mobile robot (0 for forward, 1 for reverse) and the function $\arctan 2$ is a four-quadrant inverse tangent function. The angular velocity reference, ω_r, for a mobile robot is derived by taking the time-derivative of (17) for a given target path (x_r, y_r) defined in a time interval $t \in [0, T]$ as follows:

$$\omega_r = \frac{\dot{x}_r\ddot{y}_r - \dot{y}_r\ddot{x}_r}{(\dot{x}_r)^2 + (\dot{y}_r)^2} \tag{18}$$

Remark 2 The necessary condition in the path generation is that the target path must be twice-differentiable, and the linear velocity reference must be nonzero, i.e., $v_r \neq 0$.

4 Tracking Error Learning Control

The tracking error-based model is linearized around a target path by assuming that the longitudinal, lateral and heading angle errors are around zero. Therefore, the mismatch between the tracking error-based model and real system might result in unsatisfactory control performance when the mobile robot is not on-track. Moreover, there always exist uncertainties and unmodeled dynamics, which cannot be modeled, in outdoor applications.

In the TELC, the longitudinal and lateral error dynamics are used to train the learning algorithm, which is required to learn the uncertainties and unmodelled dynamics and to keep the system on track. The learning algorithm generates the references for the control inputs. In other words, it learns the dynamics of the real system.

Like (8), the control inputs consisting of the feedback and feedforward control actions are written as follows:

$$v = v_b + v_f \tag{19}$$
$$\omega = \omega_b + \omega_f \tag{20}$$

where v_b and ω_b are the traditional feedback control actions formulated in Section 3.1, while v_f and ω_f are new feedforward control actions that are updated by the tracking error learning algorithm. The new feedforward control actions are formulated as follows:

$$v_f = v_r k_{v,1} + k_{v,0} \tag{21}$$
$$\omega_f = \omega_r k_{\omega,1} + k_{\omega,0} \tag{22}$$

where v_r and ω_r are the traditional feedforward control actions formulated in (16) and (18), respectively. $k_{v,1}$ and $k_{v,0}$ are the coefficients to update the linear velocity reference and $k_{\omega,1}$ and $k_{\omega,0}$ are the coefficients to update the angular velocity reference. TELC structure for a mobile
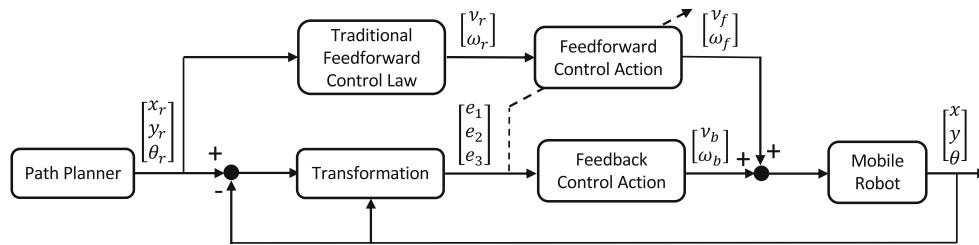
Fig. 2 Tracking error learning control structure for a mobile robot

robot is shown in Fig. 2. In the next subsection, we will formulate the update rules for these coefficients.

4.1 Linear Velocity

The requirement for the Lyapunov stability, i.e., $e_1 = 0$, is satisfied for the selection of the coefficients for the linear velocity in (21). As can be seen from (5), the linear velocity appears in the channel of the longitudinal error e_1. Therefore, we use the following squared first-order longitudinal error dynamics as the cost function:

$$E_v = \frac{1}{2}(\dot{e}_1 + \lambda_v e_1)^2 \qquad (23)$$

where λ_v is a positive constant, i.e., $\lambda_v > 0$. It is implied that if the cost function converges to zero, i.e., $E_v = 0$, then the robust control performance condition $\dot{e}_1 + \lambda_v e_1 = 0$ is satisfied so that the longitudinal error converges to zero.

Gradient descent, which is a first-order iterative optimization algorithm for finding the minimum of a function, is used to minimize the cost function E_v. In this approach, steps are taken proportional to the negative of the gradient of the closed-loop error function, i.e., E_v, to find the minimum of the cost function. The cost function is minimized to decide the coefficients in (21) as follows:

$$\dot{k}_{v,1} = -\alpha_v \frac{\partial E_v}{\partial k_{v,1}} \qquad (24)$$

where α_v is the learning coefficient and positive, i.e., $\alpha_v > 0$. It is re-written by using the chain rule

$$\dot{k}_{v,1} = -\alpha_v \frac{\partial E_v}{\partial v} \frac{\partial v}{\partial k_{v,1}} \qquad (25)$$

Equation 23 is inserted into the equation above, it is then obtained as

$$\dot{k}_{v,1} = -\alpha_v (\dot{e}_1 + \lambda_v e_1) \frac{\partial(\dot{e}_1 + \lambda_v e_1)}{\partial v} \frac{\partial v}{\partial k_{v,1}} \qquad (26)$$

Considering (4), $\frac{\partial(\dot{e}_1 + \lambda_v e_1)}{\partial v} = -1$ is obtained and inserted into the equation above, it is obtained as

$$\dot{k}_{v,1} = \alpha_v (\dot{e}_1 + \lambda_v e_1) \frac{\partial v}{\partial k_{v,1}} \qquad (27)$$

If total control input for the linear velocity applied to the mobile robot (19) and the feedforward control action for the

linear velocity (21) are inserted into (27), the adaptation of the coefficient for the linear velocity is written as follows:

$$\dot{k}_{v,1} = \alpha_v (\dot{e}_1 + \lambda_v e_1) \underbrace{\frac{\partial(v_b + v_r k_{v,1} + k_{v,0})}{\partial k_{v,1}}}_{v_r}$$

$$\dot{k}_{v,1} = \alpha_v v_r (\dot{e}_1 + \lambda_v e_1) \qquad (28)$$

The bias term $k_{v,0}$ for the linear velocity is computed by using the same procedure and found as:

$$\dot{k}_{v,0} = \alpha_v (\dot{e}_1 + \lambda_v e_1) \qquad (29)$$

4.2 Angular Velocity

First we take the time-derivative of the lateral error e_2 so that the angular velocity ω can appear in the same channel with the lateral error e_2. It is obtained considering (4) as follows:

$$\ddot{e}_2 = -(\omega_r)^2 e_2 + \omega_r v - v_r \omega \qquad (30)$$

The requirement for the Lyapunov stability, i.e., $e_2 = 0$, is satisfied for the selection of the coefficients for the angular velocity in (22). Therefore, we use the following squared second-order lateral error dynamics as the same cost function as follows:

$$E_\omega = \frac{1}{2}(\ddot{e}_2 + 2\lambda_\omega \dot{e}_2 + \lambda_\omega^2 e_2)^2 \qquad (31)$$

where λ_ω is a positive constant, i.e., $\lambda_\omega > 0$. It is implied that if the cost function converges to zero, i.e., $E_\omega = 0$, then the robust control performance condition $\ddot{e}_2 + 2\lambda_\omega \dot{e}_2 + \lambda_\omega^2 e_2 = 0$ is satisfied so that the lateral error converges to zero.

As explained in Section 4.1, the gradient descent method is used to find the minimum of the cost function E_ω and to decide the coefficients in (22) as follows:

$$\dot{k}_{\omega,1} = -\alpha_\omega \frac{\partial E_\omega}{\partial k_{\omega,1}} \qquad (32)$$

where α_ω is the learning coefficient and positive, i.e., $\alpha_\omega > 0$. It is re-written by using the chain rule

$$\dot{k}_{\omega,1} = -\alpha_\omega \frac{\partial E_\omega}{\partial \omega} \frac{\partial \omega}{\partial k_{\omega,1}} \qquad (33)$$

Eq. 31 is inserted into the equation above, it is then obtained as

$$\dot{k}_{\omega,1} = -\alpha_\omega (\ddot{e}_2 + 2\lambda_\omega \dot{e}_2 + \lambda_\omega^2 e_2) \frac{\partial(\ddot{e}_2 + 2\lambda_\omega \dot{e}_2 + \lambda_\omega^2 e_2)}{\partial \omega} \frac{\partial \omega}{\partial k_{\omega,1}} \tag{34}$$

Considering (4) and (30), $\frac{\partial(\ddot{e}_2 + 2\lambda_\omega \dot{e}_2 + \lambda_\omega^2 e_2)}{\partial \omega} = -\nu_r$ is obtained and inserted into the equation above, it is obtained as

$$\dot{k}_{\omega,1} = \alpha_\omega \nu_r (\ddot{e}_2 + 2\lambda_\omega \dot{e}_2 + \lambda_\omega^2 e_2) \frac{\partial \omega}{\partial k_{\omega,1}} \tag{35}$$

If total control input for the angular velocity applied to the mobile robot (20) is inserted into (35) considering the feedforward control action for the angular velocity (22), the adaptation of the coefficient for the angular velocity is written as follows:

$$\dot{k}_{\omega,1} = \alpha_\omega \nu_r (\ddot{e}_2 + 2\lambda_\omega \dot{e}_2 + \lambda_\omega^2 e_2) \underbrace{\frac{\partial(\omega_b + \omega_r k_{\omega,1} + k_{\omega,0})}{\partial k_{\omega,1}}}_{\omega_r}$$

$$\dot{k}_{\omega,1} = \alpha_\omega \nu_r \omega_r (\ddot{e}_2 + 2\lambda_\omega \dot{e}_2 + \lambda_\omega^2 e_2) \tag{36}$$

The bias term $\dot{k}_{\omega,0}$ for the angular velocity is computed by using the same procedure and found as:

$$\dot{k}_{\omega,0} = \alpha_\omega \nu_r (\ddot{e}_2 + 2\lambda \dot{e}_2 + \lambda^2 e_2) \tag{37}$$

4.3 Stability Analysis

The summation of the cost functions used for the linear and angular velocities is used to formulate the Lyapunov function as follows:

$$V = E_\nu + E_\omega \tag{38}$$

The Lyapunov function is positive semi-definite, i.e., $V \geq 0$. To check the stability of the tracking-error learning algorithm, the time-derivative of the Lyapunov function above is taken as follows:

$$\dot{V} = \frac{\partial E_\nu}{\partial t} + \frac{\partial E_\omega}{\partial t} \tag{39}$$

It is re-written by using the chain rule as follows:

$$\dot{V} = \frac{\partial E_\nu}{\partial k_{\nu,1}} \frac{\partial k_{\nu,1}}{\partial t} + \frac{\partial E_\nu}{\partial k_{\nu,0}} \frac{\partial k_{\nu,0}}{\partial t}$$
$$+ \frac{\partial E_\omega}{\partial k_{\omega,1}} \frac{\partial k_{\omega,1}}{\partial t} + \frac{\partial E_\omega}{\partial k_{\omega,0}} \frac{\partial k_{\omega,0}}{\partial t} + g(\gamma) \tag{40}$$

where, $g(\gamma)$ represents the derivative of the Lyapunov function V with respect to the variables other than the coefficients in the formulations of the linear and angular velocities.

The time-derivatives of the coefficienting terms in (24), (29), (32) and (37) are inserted into the aforementioned

equation, then the time-derivative of the Lyapunov function is obtained as follows:

$$\dot{V} = -\alpha_\nu \left[\underbrace{(\frac{\partial E_\nu}{\partial k_{\nu,1}})^2}_{\geq 0} + \underbrace{(\frac{\partial E_\nu}{\partial k_{\nu,0}})^2}_{\geq 0} \right]$$
$$-\alpha_w \left[\underbrace{(\frac{\partial E_\omega}{\partial k_{\omega,1}})^2}_{\geq 0} + \underbrace{(\frac{\partial E_\omega}{\partial k_{\omega,0}})^2}_{\geq 0} \right] + g(\gamma) \tag{41}$$

If the learning coefficients for the linear and angular velocity references are large enough, the time-derivative of the Lyapunov function is negative, i.e., $\dot{V} < 0$. This implies asymptotically stability of the learning algorithm.

4.4 Global Minima

The most important concern in the tracking error learning algorithm is that the system might reach some local minima and stay in these local minima. In this section, we will show that there are no local minima for the formulation of the tracking-error learning algorithm. If the second derivatives of the cost functions with respect to variables have the same sign, then the cost functions do not have a change in the curvature sign through the variables. This implies that the cost functions do not have local minima through these variables.

4.4.1 Linear Velocity

If we take the second derivative of the cost function E_ν for the linear velocity with respect to $k_{\nu,1}$, it is obtained as follows:

$$\frac{\partial^2 E_\nu}{\partial k_{\nu,1}^2} = -\alpha_\nu \nu_r \frac{\partial(\dot{e}_1 + \lambda_\nu e_1)}{\partial k_{\nu,1}} \tag{42}$$

The chain rule is applied to the equation above

$$\frac{\partial^2 E_\nu}{\partial k_{\nu,1}^2} = -\alpha_\nu \nu_r \frac{\partial(\dot{e}_1 + \lambda_\nu e_1)}{\partial \nu} \frac{\partial \nu}{\partial k_{\nu,1}} \tag{43}$$

First $\frac{\partial(\dot{e}_1 + \lambda_\nu e_1)}{\partial \nu} = -1$ is obtained considering (4) and inserted into the equation above. Also, total control input for the linear velocity applied to the mobile robot (19) and the feedforward control action for the linear velocity (21) are inserted into the equation above. Then, (43) is obtained as follows:

$$\frac{\partial^2 E_\nu}{\partial k_{\nu,1}^2} = \alpha_\nu \nu_r \underbrace{\frac{\partial(\nu_b + \nu_r k_{\nu,1} + k_{\nu,0})}{\partial k_{\nu,1}}}_{\nu_r}$$

$$\frac{\partial^2 E_\nu}{\partial k_{\nu,1}^2} = \alpha_\nu (\nu_r)^2 \tag{44}$$

The second derivative of the cost function with respect to the bias coefficient $k_{v,0}$ is computed by using the same procedure as follows:

$$\frac{\partial^2 E_v}{\partial k_{v,0}^2} = \alpha_v \tag{45}$$

Equations 44 and (45) show that the sign of the curvature of the cost function for the linear velocity (23) is always positive; therefore, there are no local minima, which indicate that the system reaches to the global minimum. After reaching the global minimum, since α_v is a constant and v_r is bounded, the coefficient update algorithms (28) and (29) show that coefficients converge to a finite value. A finite value for the coefficients in steady-state results in a bounded feedforward control action (21).

4.4.2 Angular Velocity

If we take the second derivative of the cost function E_ω for the angular velocity with respect to $k_{\omega,1}$, it is obtained as follows:

$$\frac{\partial^2 E_\omega}{\partial k_{\omega,1}^2} = -\alpha_\omega v_r \omega_r \frac{\partial(\ddot{e}_2 + 2\lambda_\omega \dot{e}_2 + \lambda_\omega^2 e_2)}{\partial k_{\omega,1}} \tag{46}$$

The chain rule is applied to the equation above

$$\frac{\partial^2 E_\omega}{\partial k_{\omega,1}^2} = -\alpha_\omega v_r \omega_r \frac{\partial(\ddot{e}_2 + 2\lambda_\omega \dot{e}_2 + \lambda_\omega^2 e_2)}{\partial \omega} \frac{\partial \omega}{\partial k_{\omega,1}} \tag{47}$$

First $\frac{\partial(\ddot{e}_2 + 2\lambda_\omega \dot{e}_2 + \lambda_\omega^2 e_2)}{\partial \omega} = -v_r$ is obtained considering (4) and (30) and inserted into the equation above. Then, total control input for the angular velocity applied to the mobile robot (20) and the feedforward control action for the angular velocity (22) are inserted into the equation above. Then, (47) is obtained as follows:

$$\frac{\partial^2 E_\omega}{\partial k_{\omega,1}^2} = \alpha_\omega (v_r)^2 \omega_r \underbrace{\frac{\partial(\omega_b + \omega_r k_{\omega,1} + k_{\omega,0})}{\partial k_{\omega,1}}}_{\omega_r}$$
$$= \alpha_\omega (v_r)^2 (\omega_r)^2 \tag{48}$$

The second derivative of the cost function E_ω with respect to the bias coefficient $k_{\omega,0}$ is computed by using the same procedure as follows:

$$\frac{\partial^2 E_\omega}{\partial k_{\omega,0}^2} = \alpha_\omega (v_r)^2 \tag{49}$$

Equations 48 and (49) show that the sign of the curvature of the cost function for the angular velocity (31) is always positive; therefore, there are no local minima, which indicate that the system reaches to the global minimum. After reaching the global minimum, since α_ω is a constant, v_r and ω_r are bounded, the coefficient update algorithms (36) and (37) show that coefficients converge to a finite value. A finite value for the coefficients in steady-state results in a bounded feedforward control action (22).

5 Experimental Validation

5.1 Mobile Robot

The mobile robot, termed TerraSentia, is constructed entirely out of 3D printed construction as shown in Figs. 3 and 4, which leads to an extremely light-weight robot (14.5 lbs) and has been proven to be structurally resilient to field conditions during an entire season of heavy operation in Corn, Sorghum, and Soybean farms in Illinois [15]. We posit that this robot is an example of the potential of additive manufacturing (3D printing) in creating a new class of agricultural equipment that works in teams to replace, minimize, or augment traditional heavy farm equipment. In addition, lightweight equipment has several benefits, it is easier to manage, has better endurance, safer to operate in general, and leads to lower ownership cost. On the other hand, the ultralight robot described here mitigates many of these challenges, yet, it leads to a uniquely challenging control problem in uneven and soil terrain in crop fields. The difficulties in control arise from complex and unknown wheel-terrain interaction and wheel slip.

The placement of hardware is illustrated in Fig. 5. One Global Navigation Satellite System (GNSS) antenna has been mounted straight up the center of TerraSentia,

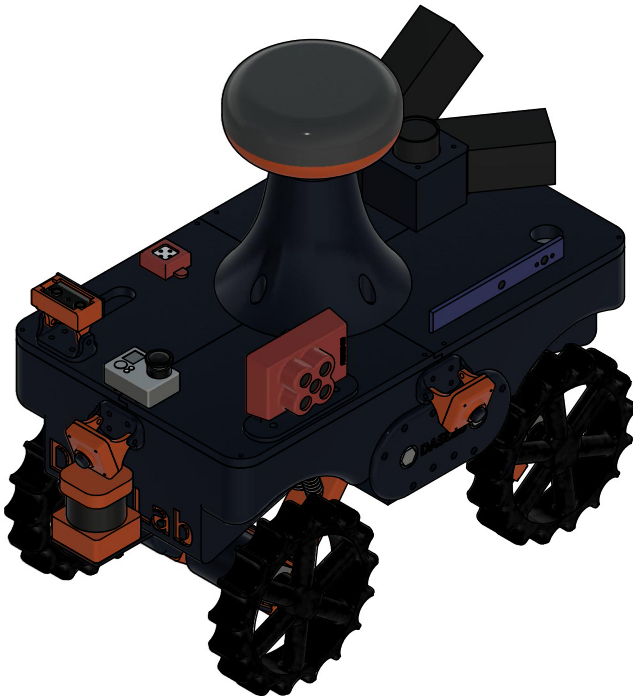**Fig. 3** The mobile robot, termed TerraSentia in off-road terrain

Fig. 4 CAD drawing of the ultra-compact 3D printed robot with a suite of sensors

and the dual-frequency GPS-capable real-time kinematic differential GNSS module (Piksi Multi, Swift Navigation, USA) has been used to acquire centimeter-level accurate positional information at a rate of 5 Hz. Another antenna and module have been used as a portable base station and has transmitted differential corrections. There are four brushed 12V DC motors with a 131.25:1 metal gearbox (Pololu Corporation, USA), which are capable of driving an attached wheel at 80 revolutions per minute. A two-channel Hall-effect encoder (Pololu Corporation, USA) for each DC motor is attached to measure velocities of the wheels. The
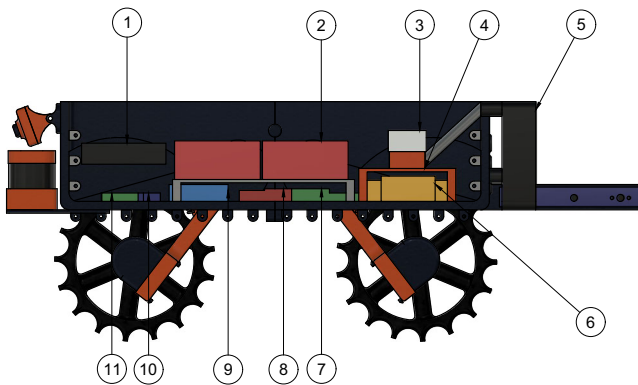
Fig. 5 Interior of the ultra-compact 3D printed robot. 1. Raspberry Pi, 2. Lithium Ion Batteries, 3. Tegra, 4. Heat sink, 5. Cooling Fan, 6. Kangaroo/Sabertooth, 7. Regulator, 8. 3-axis gyroscope, 9. Breadboard, 10. Raspberry Pi, 11. 3-axis accelerometer

Sabertooth motor driver (Dimension engineering, USA) is a two-channel motor driver that uses digital control signals to drive two motors per channel (left and right channel) and has a nominal supply current of 12 A per channel. An onboard computer (1.2GHz, 64bit, quad-core Raspberry Pi 3 Model B CPU) acquires measurements from all available sensors and sends control signals to the Sabertooth motor driver in the form of two Pulse-width modulation signals.

All available measurements from all its onboard sensors (GNSS and encoders) are fed to a state estimator, i.e., extended Kalman filter, to estimate heading angle of the mobile robot. In every time instant, estimates are fed to calculate the errors with respect to the inertial reference frame fixed to the motion ground and the estimated heading angle is fed to the transformation matrix to calculate the tracking errors in the frame on the mobile robot. Then, tracking errors are fed to the TELC algorithm, which generates the linear and angular velocity references to track the target path. Then, the linear and angular velocity references are controlled by a proportional-integral-derivative type motion controller - in other words, the robot's low-level controller - by using feedback from encoders attached to the motors to determine the required control signals in the form of the Pulse-width modulation signal. Thus, the tracking of given reference command signals ensures that the robot's desired velocities are maintained. The motion controller outputs the modified command signals to the Sabertooth Motor Driver which correlates the given control signals to the necessary output voltages needed by the DC motors.

5.2 State Estimation

An Extended Kalman Filter (EKF) is used for the state estimation in real-time because one GNSS antenna is not enough to obtain the heading angle of the mobile robot. We require heading angle information to calculate the heading angle error and also the rotation matrix. The inputs of the EKF are position information coming from the GNSS, and linear and angular velocities coming from the encoders and gyro. The outputs of the EKF are the position of the mobile robot on x- and y-coordinate system and the heading angle of the mobile robot.

The discrete-time unicycle model used for the implementation of the EKF is written as follows:

$$x_{k+1} = x_k + T_s v_k \cos \theta_k$$
$$y_{k+1} = y_k + T_s v_k \sin \theta_k$$
$$\theta_{k+1} = \theta_k + T_s \omega_k \tag{50}$$

where T_s is the sampling interval. The general form of the estimated system model is written:

$$\widehat{\mathbf{q}}_{k+1} = f(\widehat{\mathbf{q}}_k, \mathbf{u}_k) + \mathbf{w}_k$$
$$\widehat{\mathbf{z}}_{k+1} = h(\widehat{\mathbf{q}}_k) + \mathbf{v}_k \tag{51}$$

where $f(\widehat{\mathbf{q}}_k, \mathbf{u}_k)$ is the system model (50), $h(\widehat{\mathbf{q}}_k)$ is the measurement function and $\mathbf{z}_k = [x_k, y_k, v_k, \omega_k]^T$ is the measurements. The difference between the system model and real-time system is the process noise $\mathbf{w}_k$ and observation noise $\mathbf{v}_k$ in the measurement model. These noises are assumed to be independent and zero mean multivariate Gaussian noises with covariance matrices $\mathbf{W}_k$ and $\mathbf{V}_k$, respectively [4]:

$$\mathbf{w}_k \backsim N(0, \mathbf{W}_k)$$
$$\mathbf{v}_k \backsim N(0, \mathbf{V}_k) \tag{52}$$

where the weighting matrices are defined ad follows:

$$\mathbf{W}_k = diag(0.1, 0.1, 0.1) \tag{53}$$
$$\mathbf{V}_k = diag(0.03, 0.03, 0.01745) \tag{54}$$

5.3 Experimental Results

The experiment was carried out in off-road terrain and the GNSS is used as the ground truth measurements. The learning rate for the linear velocity α_v is set to 0.15 and 0.05 for the coefficients $k_{v,1}$ and $k_{v,0}$, respectively, while the learning rate for the angular velocity α_ω is set to 0.1 and 0.05 for the coefficients $k_{\omega,1}$ and $k_{\omega,0}$, respectively. The positive constants λ_v and λ_ω are set to 3. The sampling time of the experiment is equal to 200 milliseconds.

An 8-shaped path is used as the reference path for the mobile robot to evaluate the path tracking performance of the TELC algorithm. The 8-shaped path consists of two straight lines and two smooth curves as illustrated in Fig. 6. The linear velocity reference is set to 0.3 m/s throughout
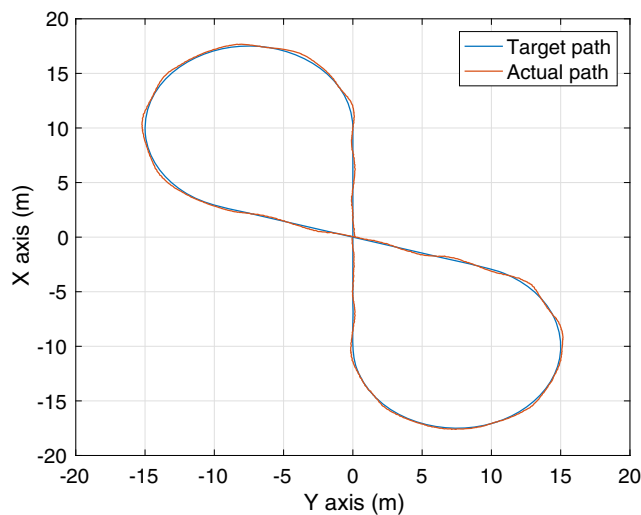

Fig. 6 Target and actual paths. Since an 8-shaped path consisting of two straight lines and two smooth curves was used to evaluate the path tracking performance of tracking-error model-based controllers in literature [2, 14, 16, 24], we also use a similar 8-shaped path. The Euclidean error is plotted in Fig. 7
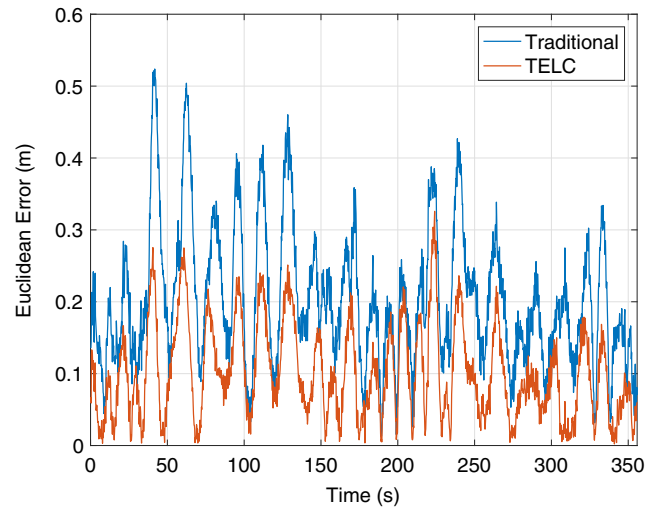

Fig. 7 Euclidean error calculated using raw GNSS data. The mean values of Euclidean errors for the traditonal tracking error-based controller formulated in Section 3 and TELC algorithm are respectively around 20.31 cm and 9.11 cm. This shows significant reduction on tracking error

the path generation. The angular velocity reference is zero for straight lines while it is set to ± 0.05 rad/s for curved lines. Thus, the desired reference trajectory x_r, y_r and θ_r is generated by considering the unicycle model in (1) in the paper. The target and actual paths are shown in Fig. 6. The mobile robot controlled by the developed TELC algorithm can track the target path precisely.

The Euclidean errors for the traditional tracking error control and TELC algorithms are shown in Fig. 7. The mean values of the Euclidean errors for the traditional tracking error control formulated in Section 3 and TELC algorithms are respectively 20.31 cm and 9.11 cm. The TELC results in more precise mobile robot path tracking performance
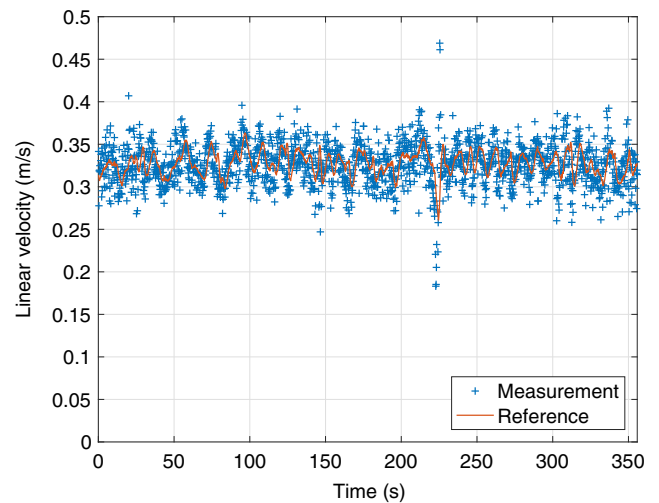

Fig. 8 Reference and measured linear velocities. The low-level controller provides good tracking performance
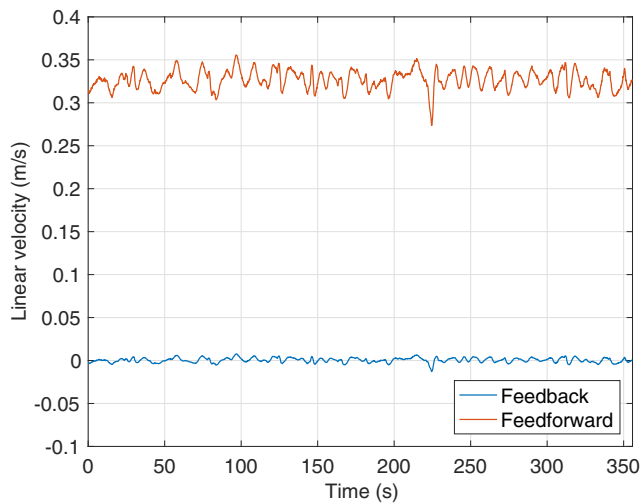
Fig. 9 Control signals for the linear velocity. TELC algorithm learns the mobile robot dynamics so that the feedback control action is around zero while feedforward control action is updated and different than zero throughout the experiment.

Fig. 11 Reference and measured angular velocities. The low-level controller provides good tracking performance

when compared to the traditional tracking error control method. This demonstrates the learning capability of the TELC algorithm in the outdoor environment.

The total linear velocity applied to the mobile robot is shown in Fig. 8. It is observed from the reference and measurements for the linear velocity that the low-level controller provides accurate tracking of the linear velocity reference despite the high noisy measurements. Moreover, the feedback control action generated by the MPC and the feedforward control action generated by the TELC algorithm are shown in Fig. 9. The feedback control action for the linear velocity is around zero as expected and the feedforward control action for the linear velocity takes the
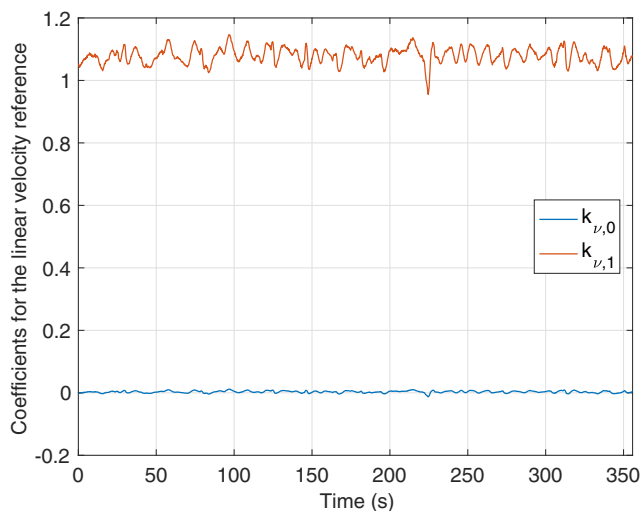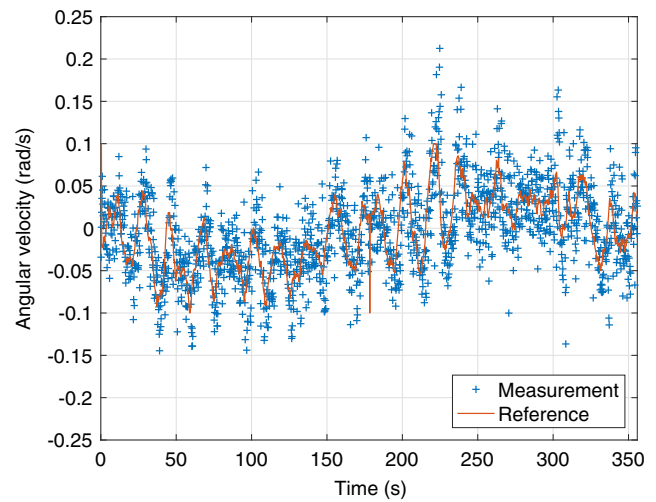
overall control action of the linear velocity of the mobile robot. Furthermore, the coefficients in the feedforward control action for the linear velocity are shown in Fig. 10. The nominal values for the coefficients $k_{v,1}$ and $k_{v,0}$ must be respectively 1 and 0 in the absence of uncertainties. These coefficients are different than the nominal values throughout experiments and varying due to the varying soil conditions in the outdoor environment.

The total angular velocity applied to the mobile robot is shown in Fig. 11. It is observed from the reference and measurements for the angular velocity that the low-level controller provides accurate tracking of the angular velocity reference despite the high noisy measurements.
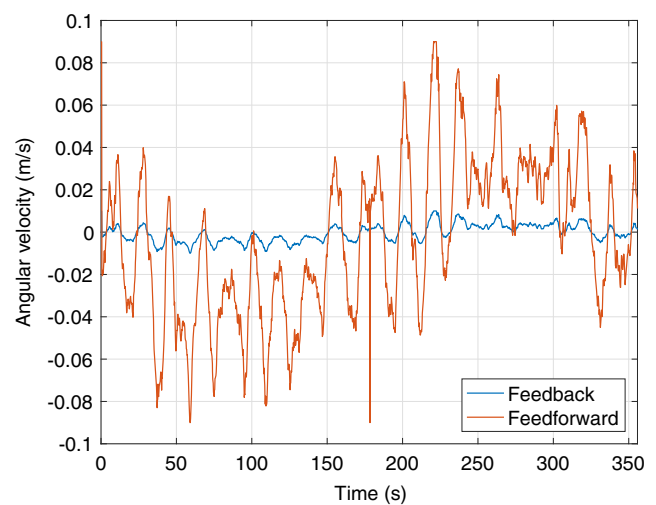

Fig. 10 Adaptations of the coefficients for the linear velocity

Fig. 12 Control signals for the angular velocity. TELC algorithm learns the mobile robot dynamics so that the feedback control action is around zero while feedforward control action is updated and different than zero throughout the experiment
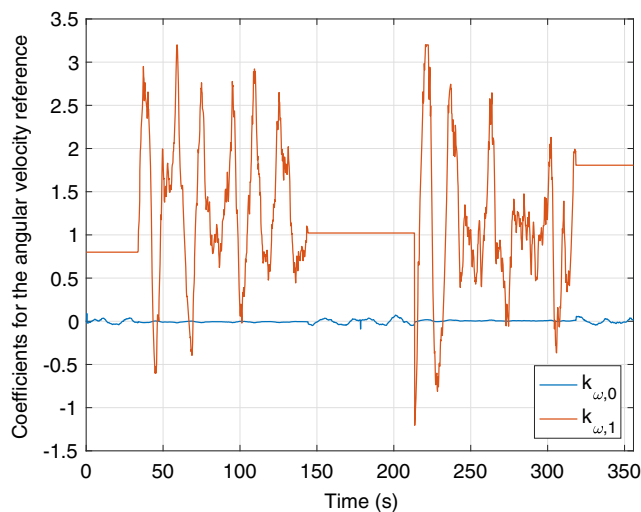
Fig. 13 Adaptations of the coefficients for the angular velocity

Moreover, the feedback control action generated by the MPC and the feedforward control action generated by the TELC algorithm are shown in Fig. 12. The feedback control action for the angular velocity is around zero as expected and the feedforward control action for the angular velocity takes the overall control action of the angular velocity of the mobile robot. Furthermore, the coefficients in the feedforward control action for the angular velocity are shown in Fig. 13. The nominal values for the coefficients $k_{\omega,1}$ and $k_{\omega,0}$ must be respectively 1 and 0 in the absence of uncertainties. These coefficients are different than the nominal values throughout experiments and varying due to the varying soil conditions in the outdoor environment. The angular velocity reference is equal to zero while the mobile robot is tracking straight lines. Therefore, the coefficient $k_{\omega,1}$ is constant, which can be seen from the update rule in (36). As a result, the only coefficient $k_{\omega,0}$ is updated to decrease the unmodeled effects in real-time.

6 Conclusions

In this paper, a novel TELC algorithm has been developed for precise path tracking and experimentally validated on a mobile robot in the outdoor environment. In case of the plant-model mismatch, the TELC algorithm learns mobile robot dynamics by using the tracking error dynamics and updates the feedforward control actions. Therefore, the feedforward controller gradually eliminates the feedback controller from the control of the system once the mobile robot has been on-track. The experimental results on the mobile robot show that the TELC algorithm ensures precise path tracking performance as compared to the traditional tracking error-based control algorithm. The mean value of

the Euclidean errors for TELC is 9 cm approximately in off-road terrain.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

1. Amer, N.H., Zamzuri, H., Hudha, K., Kadir, Z.A.: Modelling and control strategies in path tracking control for autonomous ground vehicles: a review of state of the art and challenges. J. Intell. Robot. Syst. **86**(2), 225–254 (2017). https://doi.org/10.1007/s10846-016-0442-0

2. Blazic, S.: A novel trajectory-tracking control law for wheeled mobile robots. Robot. Auton. Syst. **59**(11), 1001–1007 (2011)

3. Cui, M., Liu, H., Liu, W., Qin, Y.: An adaptive unscented kalman filter-based controller for simultaneous obstacle avoidance and tracking of wheeled mobile robots with unknown slipping parameters. Journal of Intelligent & Robotic Systems. https://doi.org/10.1007/s10846-017-0761-9 (2017)

4. Goodarzi, F.A., Lee, T.: Global formulation of an extended kalman filter on se(3) for geometric control of a quadrotor uav. J. Intell. Robot. Syst. **88**(2), 395–413 (2017). https://doi.org/10.1007/s10846-017-0525-6

5. Huynh, H.N., Verlinden, O., Vande Wouwer, A.: Comparative application of model predictive control strategies to a wheeled mobile robot. J. Intell. Robot. Syst. **87**(1), 81–95 (2017)

6. Kanjanawanishkul, K., Zell, A.: Path following for an omnidirectional mobile robot based on model predictive control. In: 2009 IEEE International Conference on Robotics and Automation, pp. 3341–3346 (2009)

7. Kayacan, E., Kayacan, E., Chen, I.M., Ramon, H., Saeys, W.: On the comparison of model-based and model-free controllers in guidance, navigation and control of agricultural vehicles, pp. 49–73. Springer International Publishing, Cham (2018)

8. Kayacan, E., Kayacan, E., Ramon, H., Saeys, W.: Modeling and identification of the yaw dynamics of an autonomous tractor. In: 2013 9Th Asian Control Conference (ASCC), pp. 1–6 (2013). https://doi.org/10.1109/ASCC.2013.6606388

9. Kayacan, E., Kayacan, E., Ramon, H., Saeys, W.: Distributed nonlinear model predictive control of an autonomous tractor–trailer system. Mechatronics **24**(8), 926–933 (2014)

10. Kayacan, E., Kayacan, E., Ramon, H., Saeys, W.: Nonlinear modeling and identification of an autonomous tractor–trailer system. Comput. Electron. Agric. **106**, 1–10 (2014)

11. Kayacan, E., Kayacan, E., Ramon, H., Saeys, W.: Learning in centralized nonlinear model predictive control: application to an autonomous tractor-trailer system. IEEE Trans. Control Syst. Technol. **23**(1), 197–205 (2015)

12. Kayacan, E., Kayacan, E., Ramon, H., Saeys, W.: Robust tube-based decentralized nonlinear model predictive control of an autonomous tractor-trailer system. IIEEE/ASME Trans. Mechatron. **20**(1), 447–456 (2015)

13. Kayacan, E., Kayacan, E., Ramon, H., Saeys, W.: Towards agrobots: identification of the yaw dynamics and trajectory tracking of an autonomous tractor. Comput. Electron. Agric. **115**, 78–87 (2015)

14. Kayacan, E., Ramon, H., Saeys, W.: Robust trajectory tracking error model-based predictive control for unmanned ground vehicles. IEEE/ASME Trans. Mechatron. **21**(2), 806–814 (2016)

15. Kayacan, E., Zhang, Z., Chowdhary, G.: Embedded high precision control and corn stand counting algorithms for an ultra-compact 3d

printed field robot. In: Proceedings of Robotics: Science and Systems. Pittsburgh, Pennsylvania (2018). https://doi.org/10.15607/RSS.2018.XIV.036

16. Klancar, G., Skrjanc, I.: Tracking-error model-based predictive control for mobile robots in real time. Robot. Auton. Syst. **55**(6), 460–469 (2007)

17. Li, M., Imou, K., Wakabayashi, K., Yokoyama, S.: Review of research on agricultural vehiclev autonomous guidance. International Journal of Agricultural & Biological Engineering **2**(3), 1–16 (2009)

18. Liao, Y., Ou, Y., Meng, S.: Wheeled mobile robot based on adaptive linear quadratic gaussian control. In: 2017 29Th Chinese Control and Decision Conference (CCDC), pp. 5768–5775 (2017). https://doi.org/10.1109/CCDC.2017.7978197

19. Lins Barreto, J.C., Scolari Conceicao, A.G., Dorea, C.E.T., Martinez, L., De Pieri, E.R.: Design and implementation of model-predictive control with friction compensation on an omnidirectional mobile robot. IEEE/ASME Trans. Mechatron. **19**(2), 467–476 (2014)

20. Martins, F.N., Celeste, W.C., Carelli, R., Sarcinelli-Filho, M., Bastos-Filho, T.F.: An adaptive dynamic controller for autonomous mobile robot trajectory tracking. Control. Eng. Pract. **16**(11), 1354–1363 (2008)

21. Normey-Rico, J.E., Alcal, I., Gmez-Ortega, J., Camacho, E.F.: Mobile robot path tracking using a robust pid controller. Control. Eng. Pract. **9**(11), 1209–1214 (2001)

22. Pan, Y., Cheng, C.A., Saigol, K., Lee, K., Yan, X., Theodorou, E., Boots, B.: Agile autonomous driving using end-to-end deep imitation learning. In: Proceedings of Robotics: Science and Systems. Pittsburgh, Pennsylvania (2018). https://doi.org/10.15607/RSS.2018.XIV.056

23. Seder, M., Baotić, M., Petrović, I.: Receding horizon control for convergent navigation of a differential drive mobile robot. IEEE Trans. Control Syst. Technol. **25**(2), 653–660 (2017). https://doi.org/10.1109/TCST.2016.2558479

24. Skrjanc, I., Klancar, G.: A comparison of continuous and discrete tracking-error model-based predictive control for mobile robots. Robot. Auton. Syst. **87**, 177–187 (2017)

25. Sun, N., Fang, Y., Chen, H., Lu, B.: Amplitude-saturated nonlinear output feedback antiswing control for underactuated cranes with double-pendulum cargo dynamics. IEEE Trans. Ind. Electron. **64**(3), 2135–2146 (2017)

26. Xiao, H., Li, Z., Yang, C., Zhang, L., Yuan, P., Ding, L., Wang, T.: Robust stabilization of a wheeled mobile robot using model predictive control based on neurodynamics optimization. IEEE Trans. Ind. Electron. **64**(1), 505–516 (2017)

Erkan Kayacan received the B.Sc. degree in mechanical engineering and the M.Sc. degree in system dynamics and control from Istanbul Technical University, Istanbul, Turkey in 2008 and 2010, respectively. He received the Ph.D. degree in Mechatronics, Biostatistics and Sensors from University of Leuven (KU Leuven), Leuven, Belgium in 2014.

He is currently a Postdoctoral Associate with Senseable City Laboratory and Computer Science & Artificial Intelligence Laboratory, Massachusetts Institute of Technology, MA, USA. Prior to MIT, he was a Postdoctoral Researcher with Delft Center for Systems and Control (DCSC), Delft University of Technology, The Netherlands and Distributed Autonomous Systems lab, University of Illinois at Urbana-Champaign, IL, USA. His research interests include real- time optimization-based control and estimation methods, nonlinear control theory, learning algorithms and machine learning with a heavy emphasis on applications to autonomous systems and field robotics.

Dr. Kayacan is a recipient of the Best Systems Paper Award at Robotics: Science and Systems (RSS) in 2018.

Girish Chowdhary received the Ph.D. degree from the Georgia Institute of Technology (Georgia Tech), Atlanta, GA, USA, in 2010.

He was a Research Engineer with the Institute for Flight Systems Technology, German Aerospace Center, Braunschweig, Germany. He was with Oklahoma State University–Stillwater, Stillwater, OK, USA, until 2016. He has post-doctoral experience with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA, USA, and the School of Aerospace Engineering, Georgia Tech. He is currently an Assistant Professor with the Agriculture and Bioengineering School and the Aerospace Engineering School, University of Illinois at Urbana–Champaign, Champaign, IL, USA. His current research interests include theoretical foundations and practical algorithms for autonomous decision making and machine learning for complex cyber- physical systems that vary with space and time.

Dr. Chowdhary is a member of the American Institute of Aeronautics and Astronautics, the IEEE Control Systems Society, and the Association for Unmanned Vehicle Systems International.