Path Following Controller for Differentially Driven Planar Robots with Limited Torques and Uncertain and Changing Dynamics

V. Pitkänen, V. Halonen, A. Kemppainen, J. Röning

Abstract— **This paper presents a path following controller that is suitable for asymmetrical planar robots with significant mass and limited motor torques. The controller is resistant against environmental forces, and inaccurate estimates of robot's inertia, by estimating their effects with Unscented Kalman Filter. The controller outputs wheel torque commands which take in account the motor torque limits and given relative priority of internal control elements. The method presented is thoroughly explained and the simulation results demonstrate the performance of the controller.**

I. Introduction

Differentially driven nonholonomic wheeled robots are the most common type of mobile robots. Their applications are widely spread across many industries such as service industry, agriculture, and defense. Although they somewhat lack in mobility when compared, for example, to pseudo-omnidirectional robots such as [1-4], differentially driven nonholonomic robots are in most cases more robust and affordable. In practice, most of these systems are provided with velocity control interfaces, such as the ROS differential drive controller interface [5] or those in the commercially available Pioneer and Khepera robots. Their performance, however, can be very restricted if full system dynamics, such as asymmetrical inertias at high-speed movements and external environmental factors (mud, rocks, etc.), are not appropriately considered.

Vast volumes have been written about the control of nonholonomic wheeled mobile robots (for a recent review of path tracking control, see, for example, [6] and references therein). In early studies, most of the proposed control methods are on the kinematic level with limited regard for deeper dynamics. These kinematic controllers are based on the assumption that there is an ideal dynamic controller attached to the robot giving all the necessary velocities. In practice, it is hard to accomplish this kind of ideal dynamic controller and thus kinematic controllers are applicable only on the cases where the robot has low mass and its actuators are relatively high-powered. Of those controllers that involve deeper dynamics, many rely on the assumptions of centered mass, symmetrical inertia, known environmental forces and/or unchanging maximum torque the actuators can produce.

During recent years, there has been a considerable interest in improving the trajectory tracking control of wheeled mobile robots [7-14]. A wide variety of methods have been proposed, including back stepping approach [9, 11], model predictive control [12], disturbance observer-based control [13, 14], neural network-based control [15], prescribed performance bound-based control [10] and two time-scale filtering technique [9], just to name only a few. Although there has been a substantial progress on this subject, it can still be argued that in the presence of uncertainties, the precise tracking control of a mobile robot is an open question. This is mainly due to the fact that the dynamics of a mobile robot can be highly nonlinear. To gain a practical insight into the problem, one could consider such real world scenarios where the robot's total mass and its distribution are only approximately known and in some cases time varying, for example, because of load and unload processes or intentional or unintentional movements of the load.

To improve the trajectory-tracking performance, Martins et al. [16] proposed a velocity-based dynamical model that enables to compensate for system dynamics in velocity-controlled systems. Their approach also allows adaptation if physical parameters (i.e. center of mass, moment of inertia, torque of motors, etc.) are initially unknown or they change. Their model is not, however, able to account for asymmetrical weight distribution or torque limits of the motors. For torque-controlled systems, Ko [17] proposed bounded controllers that are able to account for torque limits and yet provide asymptotic convergence. However, the dynamical uncertainties are not considered. In [18], Onat and Ozkan proposed a multiple model approach to account for unknown and changing system dynamics. The benefit compared to the single adaptive dynamical model approaches, such as in [6], follows from the faster parameter convergence. This is due to the vast number of initial parameter candidates that cover the parameter space. Our approach, in the present study, is quite similar. However, we relied on a Bayesian approach, i.e., we used Unscented Kalman Filter (UKF) [19] techniques together with a more simplified dynamical model whose possible inaccuracies in the mass and its distribution are compensated for by the UKF as it simply "sees" their effect as additional external forces that affect the robot.

In this paper, we propose a conceptually simple path following controller that can be used to control differentially driven mobile robots whose force to mass ratio is too low for kinematic controllers. In addition, as shown in the simulations, the controller is resistant against inaccurately modeled and changing inertia, changing environmental forces, and changing maximum wheel motor torques. Desired acceleration components can be given relative priorities and the controller outputs are wheel torque commands that take in

V. Pitkänen is with the Biomimetics and Intelligent Systems Group, University of Oulu, Finland. (ville.pitkanen@oulu.fi)

V. Halonen is with the Biomimetics and Intelligent Systems Group, University of Oulu, Finland. (veikko.halonen@oulu.fi)

A. Kemppainen is with the Biomimetics and Intelligent Systems Group, University of Oulu, Finland. (anssi.kemppainen@oulu.fi)

J. Röning is with the Biomimetics and Intelligent Systems Group, University of Oulu, Finland. (juha.roning@oulu.fi)

account these priorities and the limited maximum motor torques. The paper is organized as follows: Section II introduces the used robot model. In Section III, the controller is thoroughly described. Section IV shows the simulations and the conclusions are given in Section V.

II. Robot Model and Mathematical Notation

Figure 1 shows the planar robot model used in the simulations. The environment is planar and has an inertial coordinate frame **{G}** attached to it. The robot body is rigid and non-deformable, has two identically oriented coordinate frames, **{B}** and **{C}**, attached to it. **{B}** is located between the wheels and **{C}** is located distance ***L*** away along **{B}**'s x-axis. The wheels are rigid, non-deformable, fixed, and their contact areas to the ground are approximated as points. Powered wheels are located on **{B}**'s y-axis and there is an unpowered castor wheel located at the rear. **{T}** is the target traversing on the given parametric path. It is oriented tangentially to the current path point and has an attached frame **{H}** located distance ***L*** away along its x-axis. For any given time-instant, and for each frame, there exists an inertial coordinate frame that has the same location and orientation, e.g., for any given time instant there is an inertial frame **{BI}** that coincides with **{B}**.

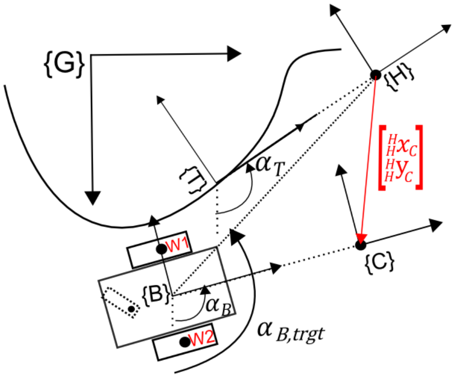

Figure 1. Model of the robot.

Unless explicitly noted otherwise, the notation used in this paper is of the form ${}^C_B r_{A,T}$, where r is the scalar or vector representing the property, A is the object with this property, B is the frame the property is relative to, C is the frame in which the property is expressed in and T is additional info. For the sake of readability, B is omitted if the property is relative to **{G}**, C is omitted if the property is expressed in **{G}** and T is only used when additional clarification is needed. Path distance-wise (s-wise) derivatives are marked with prime (e.g., α') and time-wise derivatives are marked with dot (e.g., $\dot{\phi}_{Wi}$). All the angles are in the range (-π, π]. **{T}**'s path velocity v_T is always non-negative.

The velocities, forces and masses are expressed using planar spatial vector algebra (SVA), as described by Featherstone in [20] and [21]. A spatial motion vector (SMV) describing the total momentary velocity of the robot in **{BI}** is

$$\hat{\boldsymbol{v}}_B = \begin{bmatrix} \dot{\alpha}_B \\ v_{Bx} \\ v_{By} \end{bmatrix} = \begin{bmatrix} \dot{\alpha}_B \\ {}^{BI}\dot{x}_A + {}^{BI}y_A\dot{\alpha}_B \\ {}^{BI}\dot{y}_A - {}^{BI}x_A\dot{\alpha}_B \end{bmatrix} = \begin{bmatrix} \dot{\alpha}_B \\ v_B \\ 0 \end{bmatrix}, \quad (1)$$

where $\dot{\alpha}_B$ is the rotational velocity, $({}^{BI}\dot{x}_A, {}^{BI}\dot{y}_A)$ is a known velocity of some body-fixed point and $({}^{BI}x_A, {}^{BI}y_A)$ is the location of that point in **{B}**. Robot's acceleration SMV is simply the element-wise derivative

$$\hat{\boldsymbol{a}}_B = \begin{bmatrix} \ddot{\alpha}_B \\ a_{Bx} \\ a_{By} \end{bmatrix} = \begin{bmatrix} \ddot{\alpha}_B \\ {}^{BI}\ddot{x}_A + {}^{BI}\dot{y}_A\dot{\alpha}_B + {}^{BI}y_A\ddot{\alpha}_B \\ {}^{BI}\ddot{y}_A - {}^{BI}\dot{x}_A\dot{\alpha}_B - {}^{BI}x_A\ddot{\alpha}_B \end{bmatrix} = \begin{bmatrix} \ddot{\alpha}_B \\ a_B \\ 0 \end{bmatrix}. \quad (2)$$

With a differentially driven platform that has zero lateral velocity, the elements of v_{By}, and the elements a_{By}, cancel out and the SMVs simplify to their right side forms. A spatial force vector (SFV) representing a wrench experienced by the robot as expressed in **{B}** is

$$\hat{\boldsymbol{f}}_B = \begin{bmatrix} \tau_B \\ f_{Bx} \\ f_{By} \end{bmatrix} = \begin{bmatrix} \tau + x_A f_y - y_A f_x \\ f_x \\ f_y \end{bmatrix}, \quad (3)$$

where τ is a free force couple and (f_x, f_y) is a linear force that goes through a point (x_A, y_A). Spatial inertia matrix of the robot expressed at **{B}** is

$$\hat{\boldsymbol{I}}_B = \begin{bmatrix} I_o + m(x_o^2 + y_o^2) & -my_o & mx_o \\ -my_o & m & 0 \\ mx_o & 0 & m \end{bmatrix}, \quad (4)$$

where I_o is the robot's rotational inertia at center of mass (COM), m is the mass and (x_o, y_o) is the location of COM as measured in **{B}**. The 3x3 planar inertia matrix is analytically invertible to

$$\hat{\boldsymbol{\Phi}}_B = \frac{1}{I_o - 2mx_oy_o} \begin{bmatrix} 1 & y_o & -x_o \\ y_o & \frac{I_o}{m} - x_o^2 & -x_oy_o \\ -x_o & -x_oy_o & \frac{I_o}{m} - y_o^2 \end{bmatrix}. \quad (5)$$

The force equation of the robot as expressed in SVA is

$$\hat{\boldsymbol{f}}_B = \hat{\boldsymbol{I}}_B \hat{\boldsymbol{a}}_B + \hat{\boldsymbol{v}}_B \times^* \hat{\boldsymbol{I}}_B \hat{\boldsymbol{v}}_B , \quad (6)$$

where symbol $\times^*$ is the cross-product operator between a SMV and a SFV, which in the planar case results in

$$\hat{\boldsymbol{v}}_B \times^* = \begin{bmatrix} 0 & -v_{By} & v_{Bx} \\ 0 & 0 & \dot{\alpha}_B \\ 0 & \dot{\alpha}_B & 0 \end{bmatrix}. \quad (7)$$

III. Path Controller

The goal of the controller is to match the states of **{B}** and **{T}** using limited torque commands. The steps to do this are the following. Step1 (III.A); update position of **{T}** and path parameters. Step2 (III.B); adjust velocity of **{T}**. Step 3 (III.D); determine low priority acceleration SMV. Step 4 (III.E); determine high priority acceleration SMV. Step 5 (III.F); using an estimate of environmental SFV, calculate the total SFV needed for the robot to have the high priority acceleration SMV of step four. Step 6 (III.G); calculate how much there is left in the limited motor force pool after the realization of the SFV of step five. Step 7 (III.G); calculate how the low priority SMV of step three needs to be scaled for it to be feasible with what is left of the force pool. Step 8 (III.G); give final torque commands to motors. Step 9 (III.H); use UKF to estimate the current environmental forces, which includes the effects of the robot's non-perfect mass estimates, for the next control cycle.

A. Target's Path and Orientation

The following method for calculating **{T}**'s momentary path is the one introduced in [22] and used in authors' previous work [23].

1) Offline Preparation

The path is described by a parametric Beziér spline $\boldsymbol{B}(u) = [x_T(u) \quad y_T(u)]^T, u \in [0,1]$. Its coordinate values, their *u*-wise derivatives, its curvatures ($c_T{=}\alpha'_T$) that can derived from them, and the path distance *s* are pre-calculated and stored at increasing values of *u*, thus mapping these values of traversed path-distance *s*. As the relationship between *u* and *s* is highly non-linear, this mapping must be done using numerical integration methods, such as Legendre-Gauss. In this paper, 30 values of *u* per 1 meter of *s* were used.

2) Run-time Calculations

During the run-time, the stored values are interpolated with the assumption that they change nearly linearly with respect to the change in *s* which is simply the time-wise integral of v_T. The interpolated *u*-wise derivatives are then transformed to distance-wise derivatives:

$$x'_T = \frac{dx_T}{ds} = \frac{dx_T}{du}\frac{du}{ds},\ x''_T = \frac{d^2x_T}{ds^2} = \frac{d^2x_T}{du^2}\left(\frac{du}{ds}\right)^2 + \frac{dx_T}{du}\frac{d^2u}{ds^2},$$

$$y'_T = \frac{dy_T}{ds} = \frac{dy_T}{du}\frac{du}{ds},\ y''_T = \frac{d^2y_T}{ds^2} = \frac{d^2y_T}{du^2}\left(\frac{du}{ds}\right)^2 + \frac{dy_T}{du}\frac{d^2u}{ds^2},$$

$$c'_T = \frac{dc_T}{ds} = \frac{dc_T}{du}\frac{du}{ds} = \alpha''_T,\ \frac{du}{ds} = \left(\left(\frac{dx}{du}\right)^2 + \left(\frac{dy}{du}\right)^2\right)^{-0.5}. \quad (8)$$

B. Target's Path Velocity

Path velocity of the virtual target point **{T}** is determined by the x-distance of **{C}** as "seen" by **{H}**

$$v_T = \begin{cases} 0,\ (v_B < 0)\ or\ (k_{T,v}\,{}^{H}_{H}x_C < -1) \\ v_B(1 + k_{T,v}\,{}^{H}_{H}x_C),\ else \end{cases}, \quad (9)$$

where $v_B$is the robot's measured forward velocity, $k_{T,v}$ is a positive tuning parameters and

$${}^{HI}_{H}x_C = (x_C - x_H)cos(\alpha_T) + (y_C - y_H)sin(\alpha_T), \quad (10)$$

$$x_H = x_T + Lcos(\alpha_T),\ y_H = y_T + Lsin(\alpha_T), \quad (11)$$

$$x_C = x_B + Lcos(\alpha_B),\ y_C = y_B + Lsin(\alpha_B)\ . \quad (12)$$

Effectively, the higher the positive value of ${}^{HI}_{H}x_C$ is, the faster **{T}** moves on its path to "catch up". Likewise, the higher the absolute value of negative ${}^{HI}_{H}x_C$ is, the slower **{T}** moves, or even stops, to "wait". This way, small position errors are compensated by the virtual **{T}**, not by the actual robot.

C. Acceleration Priorities

The choices how to divide the acceleration commands between lower and higher priority ones, and the exact formulation of these commands, presented in this paper are only examples that can be modified according to the scenario. However, the following idea should be retained when a fixed path is used. If the poses of **{B}** and **{T}** are identical, and $\dot{\alpha}_B = \alpha'_T v_B$, then the rotational acceleration the robot needs to stay on the path with the current forward velocity and acceleration is

$$\ddot{\alpha}_B = \alpha''_T v_B^2 + \alpha'_T\, a_B. \quad (13)$$

As v_B can not be instantaneously changed, the element $\alpha''_T v_B^2$ should given priority, at least during non-emergency operation, as not actuating it would cause the robot to drift from a path whose curvature is currently changing. Element $\alpha'_T\, a_B$, whose analogue is the rotational acceleration which happens when a car accelerates on a circular track, should be kept as low priority as a_B can be controlled with proper torque commands.

D. Low Priority Accelerations

In the simulations, the following sigmoid function was used to calculate the robots desired forward acceleration

$$a_{B,vc} = 2a_{B,max}\left(-\frac{1}{2} + \left(1 + e^{-k_{B,v}(v_{B,cl} - v_B)}\right)^{-1}\right), \quad (14)$$

where $a_{B,max}$ is the robot's maximum allowed forward acceleration, $v_{B,cl}$ is the robot's desired forward velocity, $v_B$is the robots measured forward velocity, and $k_{B,v}$ a tuning parameter. This equation could incorporate more info such as knowledge of upcoming path curvature or obstacle proximity, but such additions are outside the scope of this paper. Forward acceleration command $a_{B,vc}$ would cause $\alpha'_T\, a_{B,vc}$ angular acceleration and so the final angular acceleration command caused by $a_{B,vc}$ is

$$\ddot{\alpha}_{B,vc} = \frac{\alpha'_T a_{B,vc}}{1+\sqrt{(x_B - x_T)^2 + (y_B - y_T)^2}}, \quad (15)$$

where the function in the denominator is used to suppress the effect of the path on the robot when the robot is away from the target. The final low priority acceleration is now

$$\hat{\boldsymbol{a}}_{B,lp} = \begin{bmatrix} \ddot{\alpha}_{B,lp} \\ a_{Bx,lp} \\ a_{By,lp} \end{bmatrix} = \begin{bmatrix} \ddot{\alpha}_{B,vc} \\ a_{B,vc} \\ 0 \end{bmatrix}. \quad (16)$$

As $\hat{\boldsymbol{a}}_{B,lp}$ is of a lower priority, it will be scaled if so required by the limited wheel torques. How this happens is detailed in the later sections.

E. High Priority Accelerations

1) Caused by Path

The angular acceleration needed to keep the robot on a path of changing curvature is $\alpha''_T v_B^2$, so the priority acceleration command based on the changing curvature, and error in the location, is

$$\ddot{\alpha}_{B,cc} = \frac{\alpha''_T v_B^2}{1+\sqrt{(x_B - x_T)^2 + (y_B - y_T)^2}}. \quad (17)$$

2) Other Priority Accelerations

Depending on the scenario, various acceleration commands can be given priority. In the simulations of this paper, orientation correction, $\ddot{\alpha}_{B,ec}$, is given priority as correct orientation is critical for path following. The target orientation of the robot is facing toward **{H}**, i.e.,

$$\alpha_{B,trgt} = \text{atan}\left(\frac{y_H - y_B}{x_H - x_B}\right) + \frac{\pi}{2} - \frac{\pi}{2}\ \frac{x_H - x_B}{|x_H - x_B|} \quad (18)$$

and it’s rate of change is

$$\dot{\alpha}_{B,trgt} = \frac{(\dot{y}_H - \dot{y}_B)(x_H - x_B) - (\dot{x}_H - \dot{x}_B)(y_H - y_B)}{(x_H - x_B)^2 + (y_H - y_B)^2}, \quad (19)$$

$$\dot{x}_H = x'_H v_T = (x'_T - c_T L sin(\alpha_T))v_T, \quad (20)$$

$$\dot{y}_H = y'_H v_T = (y'_T + c_T L cos(\alpha_T))v_T. \quad (21)$$

The desired angular acceleration for minimizing both $\alpha_{B,trgt}$ and $\dot{\alpha}_{B,trgt}$is produced by a critically damped virtual spring

$$\ddot{\alpha}_{B,ec} = -\frac{K_c(\dot{\alpha}_B - \dot{\alpha}_{B,trgt}) + K_K(\alpha_B - \alpha_{B,trgt})}{K_M}, \quad (22)$$

i.e., a PD-controller whose parameters are selected so that ${K_c}^2 = 4K_M K_K$. This orientation correction method is primarily indented for small scale pose errors but works also for larger scale errors. The total high priority acceleration SMV is therefore now

$$\hat{\boldsymbol{a}}_{B,hp} = \begin{bmatrix} \ddot{\alpha}_{B,hp} \\ a_{Bx,hp} \\ a_{By,hp} \end{bmatrix} = \begin{bmatrix} \ddot{\alpha}_{B,cc} + \ddot{\alpha}_{B,ec} \\ 0 \\ 0 \end{bmatrix}. \quad (23)$$

Other likely candidates for priority accelerations, among others, are those needed for emergency stops and immediate obstacle avoidance.

F. *High Priority Force Allocation*

For the robot to retain its current $\hat{\boldsymbol{v}}_B$, the $\hat{\boldsymbol{v}}_B \times^* \hat{\boldsymbol{I}}_B \hat{\boldsymbol{v}}_B$ portion of equation (6) and environmental forces $\hat{\boldsymbol{f}}_{B,env}$ (whose formation is described later) need to be compensated for. In addition, force needs to be spent to produce high priority acceleration $\hat{\boldsymbol{a}}_{B,hp}$. The total SFV of these high priority forces is therefore

$$\hat{\boldsymbol{f}}_{B,hp} = \begin{bmatrix} \tau_{B,hp} \\ f_{Bx,hp} \\ f_{By,hp} \end{bmatrix} = -\hat{\boldsymbol{f}}_{B,env} + \hat{\boldsymbol{v}}_B \times^* \hat{\boldsymbol{I}}_B \hat{\boldsymbol{v}}_B + \hat{\boldsymbol{I}}_B \hat{\boldsymbol{a}}_{B,hp}. \quad (24)$$

With differentially driven platforms, $f_{By,hp}$ is produced by the lateral ground-wheel friction and is therefore disregarded. Production of $\hat{\boldsymbol{f}}_{B,hp}$ requires the following forces from the wheels

$$f_{W1,hp} = \frac{\tau_{B,hp} + f_{Bx,hp}\, {}^B_B y_{W2}}{-{}^B_B y_{W1} + {}^B_B y_{W2}}, \; f_{W2,hp} = \frac{\tau_{B,hp} + f_{Bx,hp}\, {}^B_B y_{W1}}{{}^B_B y_{W1} - {}^B_B y_{W2}}. (25)$$

$\hat{\boldsymbol{f}}_{B,hp}$ effectively shifts the force window that is available for $\hat{\boldsymbol{a}}_{B,lp}$ (Figure 2, left). For example, an environmental force (e.g., gravity when going uphill, steep bump on the ground) pushing against the robot allows it to have a higher negative value for $a_{Bx,lp}$ but also limits the highest possible positive value it can get.

G. *Acceleration Feasibility Estimation*

The wheel forces needed for $\hat{\boldsymbol{a}}_{B,lp}$, are calculated similarly as in the previous section

$$\hat{\boldsymbol{f}}_{B,lp} = \begin{bmatrix} \tau_{B,lp} \\ f_{Bx,lp} \\ f_{By,lp} \end{bmatrix} = \hat{\boldsymbol{I}}_B \hat{\boldsymbol{a}}_{B,lp} \quad (26)$$

$$f_{W1,lp} = \frac{\tau_{B,lp} + f_{Bx,lp}\, {}^B_B y_{W2}}{-{}^B_B y_{W1} + {}^B_B y_{W2}}, \; f_{W2,lp} = \frac{\tau_{B,lp} + f_{Bx,lp}\, {}^B_B y_{W1}}{{}^B_B y_{W1} - {}^B_B y_{W2}}. \quad (27)$$

If the point$(f_{W1,lp}, f_{W2,lp})$ is inside the area of shifted available forces (Figure 2, left), then no additional steps are needed as the desired acceleration is fully feasible. If that is not the case, additional steps need to be taken to calculate how $\hat{\boldsymbol{a}}_{B,lp}$ needs to be scaled to be feasible. If $\hat{\boldsymbol{a}}_{B,lp}$ is constructed as in section III.D, an analogy for this feasibility check is how a driver estimates how much acceleration is possible, or is controlled braking necessary, while driving on a circular track. From above, we get the following wheel force relation

$$f_{W2,lp} = -\frac{(\tau_{B,lp} + f_{Bx,lp}\, {}^B_B y_{W1})}{(\tau_{B,lp} + f_{Bx,lp}\, {}^B_B y_{W2})} f_{W1,lp}. \quad (28)$$

If this relation is maintained, then the proportion between $a_{Bx,lp}$ and $\ddot{\alpha}_{B,lp}$ is also maintained. The wheel forces are calculated simply by finding the intersections between the force line and the sifted force window (Figure 2). Finding intersections between a line and rectangle is a simple and well understood problem, the details of which are therefore omitted from this paper. With two intersections, the one closer to the ideal point $(f_{W1,lp}, f_{W2,lp})$ is selected. This results in a scaling and/or sign change of $\hat{\boldsymbol{a}}_{B,lp}$, but as the linear relation between $a_{Bx,lp}$ and $\ddot{\alpha}_{B,lp}$ is maintained, the robot will not deviate from the path. This case will result in the saturation of one wheel’s force command. With no intersections, the current situation heavily exceeds the capabilities of the wheels, for example because of large environmental forces or too high cornering speed. In this case, the closest corner to the line is selected and the force commands will be saturated for both wheels, i.e., there is likely to be unwanted deviation in the acceleration but only as little as possible given the current situation. One intersection at the exact corner is only theoretically possible and would result in the same saturation of both force commands as above.

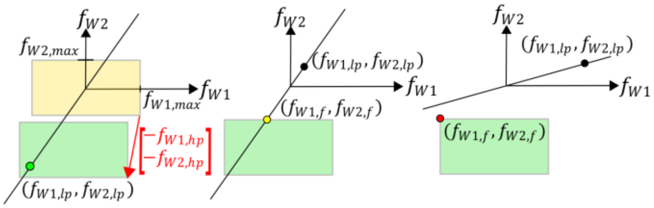

Figure 2. Directly feasible $\hat{\boldsymbol{a}}_{B,lp}$, portionally feasible $\hat{\boldsymbol{a}}_{B,lp}$, unfeasible $\hat{\boldsymbol{a}}_{B,lp}$.

The final force commands to the wheels are therefore either

$$f_{W1,cl} = f_{W1,hp} + f_{W1,lp}, \; f_{W2,cl} = f_{W2,hp} + f_{W2,lp} \quad (29)$$

if $f_{W1,lp}$ and $f_{W2,lp}$ were fully feasible and

$$f_{W1,cl} = f_{W1,hp} + f_{W1,f}, \; f_{W2,cl} = f_{W2,hp} + f_{W2,f} \quad (30)$$

if they were not. These forces result in the following SFV

$$\hat{\boldsymbol{f}}_{B,cl} = \begin{bmatrix} \tau_{B,cl} \\ f_{Bx,cl} \\ f_{By,cl} \end{bmatrix} = \begin{bmatrix} -f_{W1,cl}\, {}^B_B y_{W1} - f_{W2,cl}\, {}^B_B y_{W2} \\ f_{W1,cl} + f_{W2,cl} \\ 0 \end{bmatrix}. \quad (31)$$

H. *Environmental Force Estimation for Next Control Cycle*

The environmental forces are estimated using a direct implementation of UKF, as described in [19] (equations

7.80-7.90, using option 2) with the following values.; $\widehat{\boldsymbol{w}}_0 = [0 \quad 0]^T$, $\widehat{\boldsymbol{w}}_{k-1} = [\tau_{B,env} \quad f_{Bx,env}]^T$, $\boldsymbol{P}_{w_o} = \begin{bmatrix} P_{init} & 0 \\ 0 & P_{init} \end{bmatrix}$, $\boldsymbol{R}^{\boldsymbol{r}}_{k-1} = \begin{bmatrix} PNC & 0 \\ 0 & PNC \end{bmatrix}$, $\boldsymbol{R}^{\boldsymbol{e}}_k = \begin{bmatrix} MNC & 0 \\ 0 & MNC \end{bmatrix}$, $\boldsymbol{x}_k = \begin{bmatrix} \tau_{B,cl} \\ f_{Bx,cl} \end{bmatrix}$, $\widehat{\boldsymbol{b}} = \ \widehat{\boldsymbol{v}}_B \times^* \widehat{\boldsymbol{I}}_B \, \widehat{\boldsymbol{v}}_B$, and $\boldsymbol{G}(\boldsymbol{x}_k, \widehat{\boldsymbol{w}}_k^-) = [\widehat{\boldsymbol{\Phi}}_{\boldsymbol{B}}(\widehat{\boldsymbol{f}}_{B,cl} + \widehat{\boldsymbol{f}}_{B,env} - \widehat{\boldsymbol{b}})]_{2top}$, with “2top” subscript indicating that only the top 2 values of the vector are used. The estimated environmental SFV

$$\widehat{\boldsymbol{f}}_{B,env} = \begin{bmatrix} \widehat{\boldsymbol{w}}_k \\ 0 \end{bmatrix} = \begin{bmatrix} \tau_{B,env} \\ f_{Bx,env} \\ f_{By,env} \end{bmatrix} \tag{32}$$

is used during the next control cycle. The UKF also effectively compensates for model inaccuracies, such as incorrect robot inertia estimates, by simply estimating their effect as part of the $\widehat{\boldsymbol{f}}_{B,env}$.

IV. Simulations

A. Simulation Setup

The simulations shown in this section were done with V-REP simulator [24] using the Vortex [25] physics-engine plugin. The control cycle was 10ms and the internal time-step of the dynamics engine was the default 5ms. The robot was composed of two 16kg wheel modules attached to 10kg rigid cuboid frame and it carried a non-centrally positioned 28kg fixed payload. The robot also had a free-turning caster wheel whose dynamics were not considered in the controller, and thus its effects, along with the effects of the wheels' rotational inertia and exact wheel-ground contact interaction, on the robot's dynamics were intentionally left as modeling errors for the UKF to estimate as additional environmental forces. The parameters used in the simulations are summarized in Table 1.

TABLE I. Simulation Parameters

$x_{o,real} = -0.17m$ $y_{o,real} = 0.10m$	$v_{B,cl} = 1 + 0.5\cos(t)$ or $v_{B,cl} = 1.5 + 0.5\cos(t)$	$L = 0.5$ $k_{T,v} = 1$ $k_{B,v} = 2$
$m_{real} = 70kg$ $I_{o,real} = 21kg\,m^2$	$f_{W1,stall} = \ 300N \ or \ 150N$ $f_{W2,stall} = 300N \ or \ 150N$	$v_{W1,max} = 5m/s$ $v_{W2,max} = 5m/s$
$MNC = 0.50$ $PNC = 100$ $P_{init} = 50$	$\alpha_{UKF} = 0.1$ $\beta_{UKF} = 2$ $\kappa_{UKF} = 0$	$K_c = 1.2$ $K_M = 0.25$ $K_K = 1.44$

For each time-instant, the maximum available force of each wheel is related to its stall force $f_{Wi,stall}$, no-load speed $v_{Wi,max}$ and current speed v_{wi} in the following way

$$f_{Wi,max} = f_{Wi,stall}\left(1 - \frac{v_{Wi}}{v_{Wi,max}}\right). \tag{33}$$

In practice, $f_{Wi,max}$ could also have additional limitations such as the estimated ground-wheel contact quality.

To demonstrate how the controller compensates for changing payloads and inaccurate robot inertia estimates, $\widehat{\boldsymbol{I}}_B$ was manually corrupted such that initially $= 0.25m_{real}$, $I_o = 0.25I_{o,real}$, $x_o = 3x_{o,real}$, and $y_o = 3y_{o,real}$. After each lap on the test path, these values were updated such that $m_{k+1} = 2m_k$, $I_{o,k+1} = 2I_{o,k}$, $x_{o,k+1} = (3-k)x_{o,real}$, and $y_{o,k+1} = (3-k)y_{o,real}$.

B. Simulation Results

Figure 3 shows the robot's start position and five rounds of traversing (black line), a path similar to 5m wide and 10m long figure eight (orange line). Two setups with different force and velocity parameters were tested. In setup A $f_{Wi,stall} = 300N, v_{B,cl} = 1 + 0.5\cos(t)$. In setup B $f_{Wi,stall} = 150N, v_{B,cl} = 1.5 + 0.5\cos(t)$. With setup A the lap time was approximately 29s and with setup B 22s. In setup A the robot had enough available force to maneuver with the given velocity without saturating the force output of either wheel. This resulted in low pose error (Figure 4) that increased significantly only in laps four (time 88-117s) and five (time 118-146s) where the controller was given very corrupted inertia estimates.

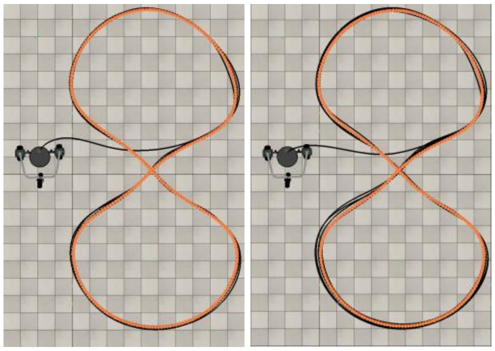

Figure 3. Robot's path of five rounds around figure eight test path. Left, setup A. Right setup B.

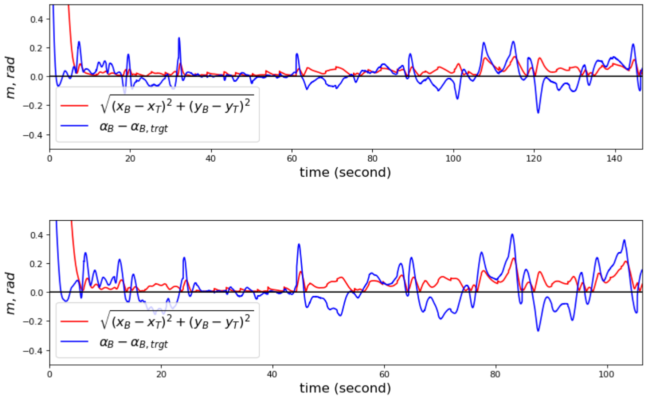

Figure 4. Pose error. Up, setup A. Down setup B.

In contrast, with setup B the robot had, intentionally for demonstration purposes, too little available force for the given velocity demands, resulting in the frequent wheel force output saturations as seen in Figure 5. Despite this, the pose error did not grow excessively. The reason for this is partly visible in Figure 6's differences between $v_{B,cl}$ and v_B. They were caused by the acceleration feasibility estimation of section III.G which scaled $\widehat{\boldsymbol{a}}_{B,lp}$ with respect to the available force in order to keep the robot on the path.

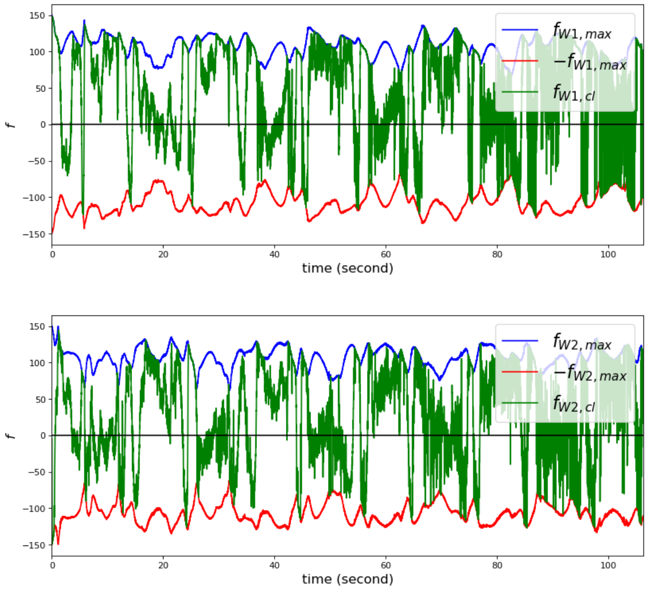

Figure 5. Maximum and control forces of wheels with setup B. Up, left wheel. Down, right wheel.

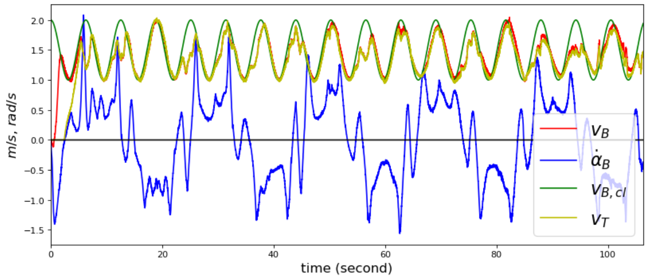

Figure 6. Velocities with setup B.

Figure 7 shows the estimated $\hat{\boldsymbol{f}}_{B,env}$ with setup B case. It can be divided into three segments. During 0-46s, the inertia estimates were too small, resulting in visible oscillation, especially in $f_{Bx,env}$. During 44-66s, the inertia estimates were correct, which can be seen as fairly constant $f_{Bx,env}$. Up until this point, the magnitude of $\hat{\boldsymbol{f}}_{B,env}$ remained almost constant as the other dynamics that were not modeled, especially those of the castor wheel, dominated over the faulty estimate of $\hat{\boldsymbol{I}}_B$. After 66s, the increasingly more manually corrupted $\hat{\boldsymbol{I}}_B$ began to have a more visible effect on $\hat{\boldsymbol{f}}_{B,env}$, especially during the fifth lap starting at 88s, at which point the mass and rotational inertia estimates were four times their actual size and the COM location estimate outside the robot.

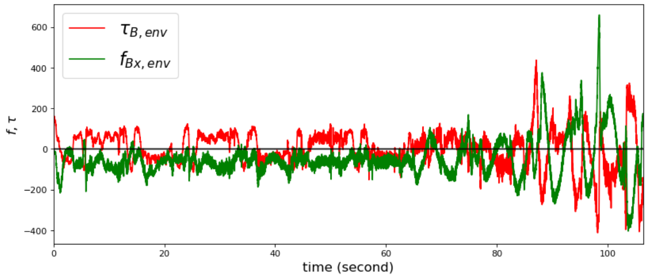

Figure 7. $\hat{\boldsymbol{f}}_{B,env}$ estimated by the UKF with setup B.

As can be seen by comparing Figures 4 and 7, the performance of the controller does decrease with inaccurate estimate of $\hat{\boldsymbol{I}}_B$, but not excessively so because its effect simply becomes part of the total environmental force that is estimated and compensated by the controller. An additional 5 lap test with fixed 10cm high cuboid obstacles is shown on the left side of Figure 8. Apart from the obstacles the setup was identical to setup A except that $v_{B,cl} = 1.5$. The obstacle height and shape were selected so that the robot would not be able to go over them without some momentum, and the higher $v_{B,cl}$ was used give the robot that necessary momentum.

In summary, Figures 3-8 demonstrate how the controller can produce force commands that do realize desired accelerations when feasible, and mitigate the occurring pose error when the situation exceeds the force output of the motors. To underline the importance of sections III.G and III.H, the middle of Figure 8 shows one round of the test with setup B, except that the environmental force was *not* estimated (i.e., $\hat{\boldsymbol{f}}_{B,env} = [0,0,0]^T$), and with $\hat{\boldsymbol{I}}_B$ estimation as in the second lap, while the right side of Figure 8 shows the original test with setup B, except that the feasibility estimation of section III.G. was *not* performed (i.e., always $f_{W1,cl} = f_{W1,hp} + f_{W1,lp}$, $f_{W2,cl} = f_{W2,hp} + f_{W2,lp}$ and commands exceeding wheel force limits were simply set to the properly signed $f_{W1,max}$ or $f_{W2,max}$).

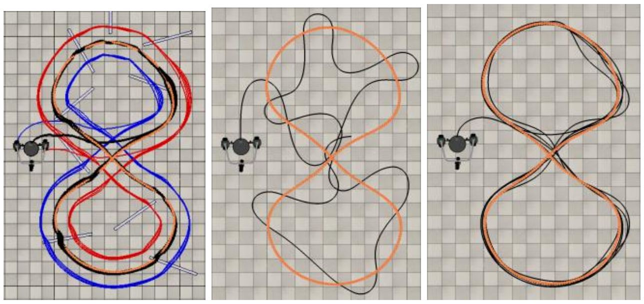

Figure 8. Left, low obstacles (wheel paths also visible). Middle, no environmental force estimation. Right, no feasibility estimation.

V. Conclusion

This paper introduced a conceptually simple controller for differentially steered mobile robots of significant asymmetrical mass and limited motor forces. The proposed controller performed well in the simulations, even when the inertia estimates were corrupted from 25% to 400% of their actual values. While the proposed controller does not remove the need for proper trajectory planning, it is an additional safeguard against modeling errors and unpredictable environmental forces when the robot is used, intentionally or not, at the limits of its performance. Future work for the proposed controller includes field testing and various refinements, such as incorporating the effect of the wheel's rotational inertia, extension of the controller to function with steerable wheels, and refining the used UKF.

Acknowledgments

The authors would like to thank the Finnish Cultural Foundation, University of Oulu Graduate School (UniOGs) and Infotech Oulu for making this research possible.

REFERENCES

[1] Y. Mori, E. Nakano and T. Takahahsi, “Mechanics, control and design methodology of the non-holonomic quasi-omnidirectional vehicle ODV9,” *The International Journal of Robotic Research*, vol. 21, May. 2002, pp. 511-522.

[2] G. Ishigami, K. Iagnemma, J. Overholt and G. Hudas, Design, Development, and Mobility Evaluation of an Omnidirectional Mobile Robot for Rough Terrain,” *Journal of Field Robotics*, Nov. 2014.

[3] M. Lauria, I. Nadeau, P. Lepage, Y. Morin, P. Giguere, F. Gagnon, D. Letourneau, and F. Michaud, “Kinematical analysis of a four steered wheeled mobile robot,” *Proceedings IEEE International Conference on Industrial Electronics*, vol. 4, 2006, pp. 3090–3095.

[4] R. Oftadeh, R. Ghabcheloo, and J. Mattila, "A novel time optimal path following controller with bounded velocities for mobile robots with independently steerable wheels," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Japan, 2013.

[5] http://wiki.ros.org/diff_drive_controller

[6] N. H. Amer, H. Zamzuri, K. Hudha, and Z. A. Kadir, “Modelling and control strategies in path tracking control for autonomous ground vehicles: a review of state of the art and challenges,” *Journal of Intelligent & Robotic Systems*, vol. 86(2), pp. 225-254, 2017.

[7] A. Arab and M. M. Mohammad, "An uncertainty compensator for robust control of wheeled mobile robots," *Advanced Robotics,* vol. 29.20, pp. 1303-1313, 2015.

[8] H. Chen, “Robust stabilization for a class of dynamic feedback uncertain nonholonomic mobile robots with input saturation,” *International Journal of Control, Automation and Systems*, vol. *12*(6), pp. 1216-1224, 2014.

[9] W. Sun, S. Tang, H. Gao, and J. Zhao, “Two time-scale tracking control of nonholonomic wheeled mobile robots. *IEEE Transactions on Control Systems Technology*, vol. 24(6), pp. 2059-2069, 2016.

[10] W. Wang, J. Huang, and C. Wen, “Prescribed performance bound-based adaptive path-following control of uncertain nonholonomic mobile robots,” *International Journal of Adaptive Control and Signal Processing*, vol. 31(5), pp. 805-822, 2017.

[11] W. Dong and Y. Guo, “Dynamic tracking control of uncertain nonholonomic mobile robots,” in *Intelligent Robots and Systems (IROS), 2005 IEEE/RSJ International Conference on,* pp. 2774-2779.

[12] I. Škrjanc and G. Klančar, “A comparison of continuous and discrete tracking-error model-based predictive control for mobile robots,” *Robotics and autonomous systems*, vol. 87, pp. 177-187, 2017.

[13] D. Huang, J. Zhai, W. Ai, and S. Fei, “Disturbance observer-based robust control for trajectory tracking of wheeled mobile robots,” *Neurocomputing*, vol. 198, pp. 74-79, 2016.

[14] L. Xin, Q. Wang, J. She, and Y. Li, (2016). “Robust adaptive tracking control of wheeled mobile robot,” *Robotics and Autonomous Systems*, vol. 78, pp. 36-48, 2016.

[15] M. Boukens and A. Boukabou, "Design of an intelligent optimal neural network-based tracking controller for nonholonomic mobile robot systems," *Neurocomputing,* vol. 226, pp. 46-57, 2017.

[16] F. N. Martins, M. Sarcinelli-Filho, R. Carelli, “A Velocity-Based Dynamic Model and Its Properties for Differential Drive Mobile Robots, Jouran of Intelligent & Robotics Systems, Vol. 85, Issue 2, pp. 277-292, 2017.

[17] K. Do. “Bounded controllers for global path tracking control of unicycle-type mobile robots.” Robotics and Autonomous Systems, Vol. **61**, pp. 775-784, 2013.

[18] A. Onat and M. Ozkan, “Dynamic adaptive trajectory tracking control of nonholonomic mobile robots using multiple models approach.” Advanced Robotics, Volume **29,** 2015.

[19] E. A. Wan, and R. van der Merwe, (2001) The Unscented Kalman Filter, in Kalman Filtering and Neural Networks (ed S. Haykin), John Wiley & Sons, Inc., New York, USA. doi: 10.1002/0471221546.ch7

[20] R. Featherstone, *Rigid Body Dynamics Algorithms*, Springer, New York, 2008, pp. 7-35.

[21] R. Featherstone, "A beginner's guide to 6-d vectors (part 1*)," IEEE Robotics Automation Magazine*, vol. 17, 2010, pp. 83-94.

[22] B. Lau, C. Sprunk, and W. Burgard, "Kinodynamic motion planning for mobile robots using splines," in IEEE Intl. Conf. on Intelligent Robots and Systems (IROS), St. Louis, MO, USA, 2009.

[23] V. Pitkänen, A. Kemppainen, A. Tikanmäki and J. Röning “Path following controller for planar robots with articulated, actuated and independently steerable velocity-limited wheels.” *Robotics and Automation (ICRA), 2017 IEEE International Conference on,* Singapore, 2017.

[24] http://www.coppeliarobotics.com/

[25] https://www.cm-labs.com/