Simulation-Based Engineering Lab
University of Wisconsin-Madison
Technical Report TR–2023–06

Simplified 4DOF Bicycle Model for Robotics Applications

Harry Zhang, Huzaifa Unjhawala, Stefan Caldararu,
Ishaan Mahajan, Luning Bakke, Radu Serban, Dan Negrut

Department of Mechanical Engineering, University of Wisconsin – Madison

June 19, 2023

Contents

1 Introduction

In many robotics applications, people need a state transition model as part of the algorithms and computation. A popular simplified model for ground vehicle [1] is given by:

$$\dot{\mathbf{q}} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \mathbf{f}(\mathbf{q}, \mathbf{u}) = \begin{pmatrix} cos(\theta) \cdot v \\ sin(\theta) \cdot v \\ \frac{v \cdot tan(\rho)}{l} \end{pmatrix} \tag{1}$$

The state $\mathbf{q} = [x, y, \theta]^T$, and commands $\mathbf{u} = [v, \rho]^T$, where x, y are the Cartesian coordinates of the vehicle, θ is the heading angle (yaw angle) of the vehicle with respect to the positive x-axis, v is the longitudinal velocity, ρ is the steering angle of the front wheel, and l is the wheelbase of the vehicle (length between center of the front wheel and rear wheel). This simplified model has many limitations, such as the assumption of zero vehicle lateral velocity, the zero slip of the vehicle's wheel, etc., which leaves an sizable improvement space for accuracy purpose. However, due to its smaller state and input dimension, it is simple to represent and fast to solve compared to other more complex vehicle models. In the following sections, we thus capitalize on the attractive features of the 3DOF model and augment it to develop a new 4DOF simplified vehicle model.

2 Model Description

The state transition model described in section 1 uses the vehicle's longitudinal velocity and vehicle's front wheel steering angle as the vehicle's inputs. However, in reality, most vehicles do not take a desired velocity as command input, but rather have a normalized throttle position as input. In other words, one can physically press a throttle to speed up a vehicle, but cannot physically do something to set a velocity value; the latter is a consequence of how much the throttle is pressed. For instance, a 0 to 1 throttle input adjusts the throttle position from not pressing the paddle to completely pressing the throttle pedal; a -1 to 1 steering inputs indicates from full steering to the left to steering fully to the right. The vehicle and the digital twin of the vehicle in the high-fidelity simulation engine–Autonomy Research Testbed (ART)–both use this above pair of control inputs [2], which are the throttle $[0, 1]$ and steering $[-1, 1]$.

The purpose of this work is to generate a 4DOF state transition model for the vehicle to use in ART, using throttle and steering instead of velocity and steering angle as the model's inputs. The bicycle 3DOF model is extended to a 4-DOF model where the state variables and control inputs are $\mathbf{q} = [x, y, \theta, v]^T$ and $\mathbf{u} = [\alpha, \delta]^T$, respectively. Note that $\mathbf{q}$ contains one more state variable, i.e., the vehicle's longitudinal velocity, v, compared with bicycle model in Eq. (1). By the same token, $\mathbf{u}$ consists of the throttle and steering inputs to the vehicle. The 4DOF vehicle state transition model is described as

$$\dot{\mathbf{q}} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \end{pmatrix} = \mathbf{f}(\mathbf{q}, \mathbf{u}) = \begin{cases} cos(\theta) \cdot v, & \text{(2a)} \\ sin(\theta) \cdot v, & \text{(2b)} \\ \dfrac{v \cdot tan(\beta\delta)}{l}, & \text{(2c)} \\ \dot{\omega} \cdot R_{wheel} = \dfrac{T(\alpha, v) \cdot \gamma}{I_{wheel}} \cdot R_{wheel}. & \text{(2d)} \end{cases}$$

Equation (2c) maps the steering command δ to the wheel steering angle ρ through a coefficient β. Assuming the wheel and the ground has zero slip, Eq. (2d) describes the relation between throttle command α and the longitudinal acceleration of the vehicle, $\dot{v}$, from the motor torque $T(\alpha, v)$, gear ratio, γ, and the inertia and radius of the wheel, I_{wheel} and R_{wheel}, respectively. Herein, the motor torque T is modeled as

$$T(\alpha, v) = T'(\alpha, \omega_m) = \alpha f_1(\omega_m) - c_1\omega_m - c_0, \tag{3a}$$

$$f_1(\omega_m) = -\frac{\tau_0 \omega_m}{\omega_0} + \tau_0, \tag{3b}$$

$$\omega_m = \frac{v}{R_{wheel}\gamma}. \tag{3c}$$

Equation (3a) models the total torque as the difference between the scaled motor torque, $\alpha f_1(\omega)$, and motor resistance torque, $c_1\omega_m + c_0$. For a brushless DC motor, Eq. (3b) can approximate the load at different motor speeds, given the stalling torque τ_0 and maximum no-load speed ω_0, see Fig. 1 as an example. Note that both τ_0 and ω_0 are specifications provided by the manufacturer. Equation (3c) relates the angular velocity of the motor ω_m with that of the wheel, v/R_{wheel}, through the gear ratio γ.

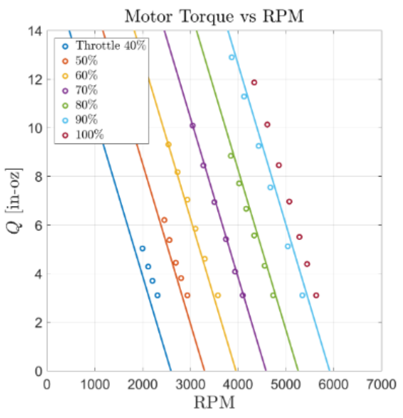


Figure 1: Torque vs. motor speed map of a brushless DC motor [3], where each line corresponds to a different throttle value α, and the slope of the lines is τ_0/ω_0.

3 Numerical Experiments

To validate this 4DOF model, we can utilize this model in different robotics applications, such as control, state estimation, etc. The platform for conducting experiments is ART [2], which uses digital twins produced in Chrono [4–6].

3.1 Implementation in Control Policy Design

One potential use of a dynamics model for robot control might be in conjunction with Model Predictive Control, see more detailed explanation for a way-points tracking problem in previous work [7]. The 4DOF model comes into play when formulating an optimal control problem.

$$J_t^*(\mathbf{e}_t) = \min_{\mathbf{u}_k} \quad \mathbf{e}_N^T \mathbf{Q} \mathbf{e}_N + \sum_{k=0}^{N-1} \mathbf{e}_k^T \mathbf{Q} \mathbf{e}_k + (\mathbf{u}_k - \mathbf{u}_r)^T \mathbf{R} (\mathbf{u}_k - \mathbf{u}_r)$$

$$\mathbf{e}_{k+1} = \mathbf{A}_k \cdot \mathbf{e}_k + \mathbf{B}_k \cdot \mathbf{u}_k \tag{4a}$$

$$\mathbf{e}_k \in \mathbf{E}, \mathbf{u}_k \in \mathbf{U}, k = 0, .., N-1 \tag{4b}$$

$$\mathbf{e}_0 = \mathbf{e}_0 \tag{4c}$$

From a high vantage point, Eq. (4) poses a program that seeks to minimize an error vector, $\mathbf{e}$, which describes how much off the current state is from the reference trajectory. As part of the optimization problem's constraint, $\mathbf{e}_{k+1} = \mathbf{A}_k \cdot \mathbf{e}_k + \mathbf{B}_k \cdot \mathbf{u}_k$ is called dynamics constraint, indicating how one should calculate the system's state at the next step given the current state and command. Here, the error vector $\mathbf{e}$ is defined as

$$e = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \\ v_r - v \end{pmatrix} . \tag{5}$$

Note that x_r, y_r, θ_r and v_r represent pre-defined reference trajectory information. Since the 4DOF model provides the derivative of the state in Eq. (2), it is possible to obtain the derivative of the error state $\mathbf{e}$ and then further discretize like in Eq. 4. Specifically,

$$\dot{e} = \begin{pmatrix} \frac{v \cdot \tan\delta \cdot e_2}{l} + v_r \cdot \cos e_3 - v \\ -\frac{v \cdot \tan\delta \cdot e_1}{l} + v_r \cdot \sin e_3 \\ \frac{v_r \cdot \tan\delta_r - v \cdot \tan\delta}{l} \\ \frac{\tau_0 R_{wheel} \gamma}{I_w} (\alpha_r - \alpha) - e_4 \frac{c_1 \omega_0 + \tau_0}{I_w \omega_0} \end{pmatrix} = g(e, u) \tag{6}$$

$$\dot{e} = \frac{\partial f}{\partial e} \cdot e + \frac{\partial f}{\partial u} \cdot u = \begin{pmatrix} 0 & \frac{v \cdot \tan\delta}{l} & -v_r \cdot \sin e_3 & 0 \\ -\frac{v \cdot \tan\delta}{l} & 0 & v_r \cdot \cos e_3 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{c_1 \omega_0 + \tau_0}{I_w \omega_0} \end{pmatrix} \cdot e + \begin{pmatrix} 0 & \frac{v \cdot e_2}{l \cdot \cos^2\delta} \\ 0 & -\frac{v \cdot e_1}{l \cdot \cos^2\delta} \\ 0 & -\frac{v}{l \cdot \cos^2\delta} \\ -\frac{\tau_0 R_{wheel} \gamma}{I_w} & 0 \end{pmatrix} \cdot u$$

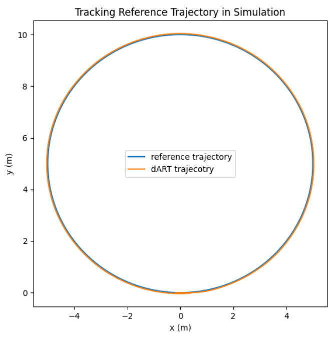
$$e_{t+1} = A_t \cdot e_t + B_t \cdot u_t \tag{7}$$

$$A_t = \begin{pmatrix} 0 & \frac{v \cdot \tan\delta}{l} & -v_r \cdot \sin e_3 & 0 \\ -\frac{v \cdot \tan\delta}{l} & 0 & v_r \cdot \cos e_3 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{c_1 \omega_0 + \tau_0}{I_w \omega_0} \end{pmatrix} \cdot \Delta t + \mathbb{I}_{4\times 4} \qquad B_t = \begin{pmatrix} 0 & \frac{v \cdot e_2}{l \cdot \cos^2\delta} \\ 0 & -\frac{v \cdot e_1}{l \cdot \cos^2\delta} \\ 0 & -\frac{v}{l \cdot \cos^2\delta} \\ -\frac{\tau_0 R_{wheel} \gamma}{I_w} & 0 \end{pmatrix} \cdot \Delta t$$

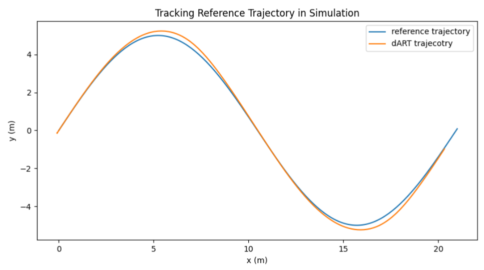
This completely sets up the MPC problem. To test the performance of the MPC controller that draws on the 4DOF model, we tested different reference trajectories in both simulation and reality. The following results illustrate the performance of the controller.

In simulation, for sensing purposes, the vehicle directly receives privileged position information (same as ground truth) to perform a tracking task. Then, we use the same controller in reality. To get accurate position outdoor, RTK-GPS has been used to ensure less than 5 centimeters error [8].

Figure 2 and 3 show good performance of the MPC controller in both simulation and reality, which demonstrates that the 4DOF works decently well for the autonomous operation of this vehicle.



(a) Circle Reference Trajectory



(b) Sinusoidal Reference Trajectory

Figure 2: MPC Tracking Performance in Simulation

3.2 Implementation in Localization Algorithm

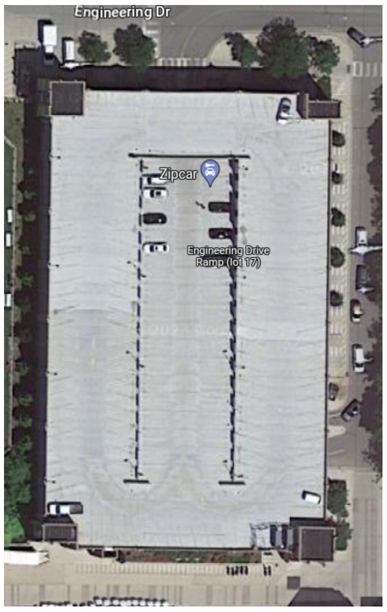
Another practical use of the 4DOF model is in State Estimation, via algorithms such as the Extended Kalman Filter (EKF) and Particle Filter (PF). Shortly, a state estimation algorithm requires two steps: prediction and correction. The 4DOF model will make use of Eq. 2 to firstly provide a prediction of the state at next time step with the known information about state and control input at current time step; then, the measurement comes into play and to correct the prediction made by motion model with some weighted method between the two of them. The more accurate the prediction obtained from the motion model, the more likely it is to provide good state estimation. So, it is worth conducting a Dead Reckoning (DR) experiment using the 4DOF model – this amounts to giving a command $\mathbf{u}$ to the 4DOF model at every time step, and then using Forward Euler method to obtain vehicle state.

In Fig. 4a, the constant throttle and steering command has been executed for both the Chrono dART and 4DOF models. As shown in the results in Fig. 4b, the 4DOF's DR results are close to the dART trajectory. Since constant throttle and steering inputs might be insufficient in practical cases, we also conduct dead reckoning tests using recorded commands generated by the MPC controller as shown in Figs. 4c and 4d. Throttle and steering inputs are more noisy in this case, which leads to more offed trajectory compared with the trajectory generated from constant command inputs.

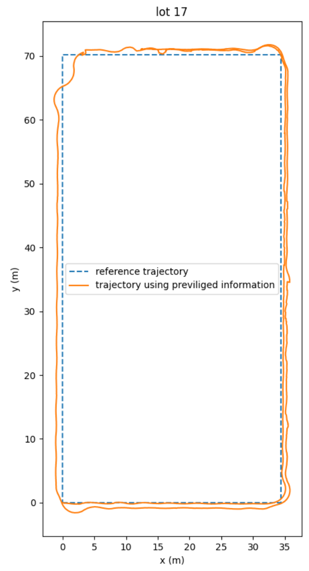
4 Conclusion

In general, the 4DOF model performs well in the control and state estimation exercises considered. For model based control task, the 4DOF model is a good choice, because it leads to fast solving speed owing to only introducing four degrees of freedom into the optimization problem. Since the model is linearized at each time step and because we employ here the feedback control, the 4DOF turns out to be sufficiently accurate to be useful. However, for state estimation, 4DOF might not be precise enough to yield a good “prediction” step. To enhance the performance of state estimation, it is useful to improve the fidelity of the model, for example using a more complicated and more realistic model, to better predict the state of the vehicle based on the commands.

It remains to see if the 4DOF model continues to perform well when the vehicle is driven at high speed and starts showing traces of lateral speeds. Also, one should test the usefulness of the



(a) Testing Field View from Google Map



(b) RTK-GPS Measured Vehicle Trajectory

Figure 3: MPC Tracking Performance in Reality

4DOF vehicle in more challenging applications, when the sensor data is not presented as privileged information but the vehicle is subject to more noisy sensor data produced by sensor models [9,10].

References

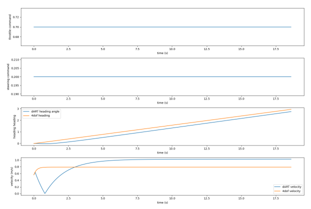
[1] H. Guo, D. Cao, H. Chen, C. Lv, H. Wang, and S. Yang, "Vehicle dynamic state estimation: State of the art schemes and perspectives," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 2, pp. 418–431, 2018.

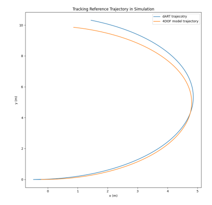
[2] A. Elmquist, A. Young, I. Mahajan, K. Fahey, A. Dashora, S. Ashokkumar, S. Caldararu, V. Freire, X. Xu, R. Serban, and D. Negrut, "A software toolkit and hardware platform for investigating and comparing robot autonomy algorithms in simulation and reality," *arXiv preprint arXiv:2206.06537*, 2022.

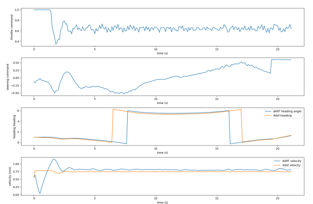
[3] R. H. Brown, *Characterization and Modeling of Brushless DC Motors and Electronic Speed Controllers with a Dynamometer.* PhD thesis, Ann Arbor, 2019. M.S.

[4] R. Serban, M. Taylor, D. Negrut, and A. Tasora, "Chrono::Vehicle template-based ground vehicle modeling and simulation," *Intl. J. Veh. Performance*, vol. 5, no. 1, pp. 18–39, 2019.

[5] H. Mazhar, T. Heyn, A. Pazouki, D. Melanz, A. Seidl, A. Bartholomew, A. Tasora, and D. Negrut, "Chrono: a parallel multi-physics library for rigid-body, flexible-body, and fluid dynamics," *Mechanical Sciences*, vol. 4, no. 1, pp. 49–64, 2013.


(a) Control Inputs, Heading Angle, and Velocity Profile


(b) Vehicle Trajectory Comparision


(c) Control Inputs, Heading Angle, and Velocity Profile

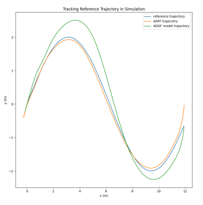

(d) Vehicle Trajectory Comparision

Figure 4: Dead Reckoning Using 4DOF model

[6] A. Tasora, R. Serban, H. Mazhar, A. Pazouki, D. Melanz, J. Fleischmann, M. Taylor, H. Sugiyama, and D. Negrut, "Chrono: An open source multi-physics dynamics engine," in *High Performance Computing in Science and Engineering – Lecture Notes in Computer Science* (T. Kozubek, ed.), pp. 19–49, Springer International Publishing, 2016.

[7] H. Zhang, S. Chatterjee, T. Hansen, S. Caldararu, I. Mahajan, N. Batagoda, L. Fang, R. Serban, and D. Negrut, "Formulating model predictive control (mpc) strategies in conjunction with error dynamics based waypoint-seeking to model robust vehicle control," tech. rep., Simulation-Based Engineering Laboratory, University of Wisconsin-Madison, 2023. `https://sbel.wisc.edu/wp-content/uploads/sites/569/2023/03/TR-2023-01.pdf`.

[8] Y. Feng, J. Wang, *et al.*, "Gps rtk performance characteristics and analysis," *Positioning*, vol. 1, no. 13, 2008.

[9] A. Elmquist and D. Negrut, "Methods and models for simulating autonomous vehicle sensors," *IEEE Transactions on Intelligent Vehicles*, vol. 5, pp. 684–692, 2020.

[10] A. Elmquist, R. Serban, and D. Negrut, "A sensor simulation framework for training and testing robots and autonomous vehicles," *Journal of Autonomous Vehicles and Systems*, vol. 1, no. 2, p. 021001, 2021.