Can the Webots Robot Simulator be Used For Self-Driving Motorcycle Controller Design?

Wenjia Li[1], Paris Francis [1], Benjamin Arky [2] and Alexander A. Brown[2], *Member, IEEE*

***Abstract*—While several commercial software packages for simulating the dynamics of robot or human-ridden Single-Track Vehicles (STVs) such as bicycles, motorcycles, or other Powered Two-Wheelers (PTWs) are common in the literature, open-source options for high-fidelity simulations of STV dynamics are limited. This paper explores whether the open-source Webots robot simulator is capable of representing the dynamics of STVs with sufficient fidelity to allow for robotic rider design. Data from a small-scale self-balancing PTW are compared with both a linear dynamic model and with simulation results from a nonlinear, multi-body model of the vehicle custom-built in Webots. Results indicate that Webots's physics engine provides sufficient dynamic fidelity to accurately represent the behavior of the vehicle in both simple step response tests and in assisted teleoperation with automatic starting and stopping via a motorized kickstand.**

***Index Terms*—Safety Verification and Validation Techniques; Vulnerable Road User Protection Strategies; Real-Time Control; Intelligent Transportation Systems; Single-Track Vehicles; Powered Two-Wheelers**

I. Introduction

Motorcycles, electric scooters, and other powered two-wheelers (PTWs) are increasing in popularity, but pose serious challenges to traffic safety [1] [2]. Users of both bicycles and PTWs have to contend with many safety challenges even without the threat of collision with other vehicles, not the least of which is the fact that Single-Track Vehicles (STVs), including bicycles and PTWs, are often open-loop unstable [3]. Active safety systems for PTWs in particular represent an area for growth, and technologies such as Automatic Emergency Braking are subjects of ongoing study [4]. However, development of active safety systems for PTWs is challenging due to their inherent dynamic complexity, especially under acceleration and deceleration, and when friction is scarce. Human-ridden testbeds for vetting emerging safety systems are an option, but pose serious risks to test riders. Self-stabilizing and/or self-driving test vehicles mitigate this risk, and could therefore play an important role in testing new active safety technologies, infrastructure-based safety improvements, and/or safety gear for PTWs. For both PTWs and human-powered STVs such as bicycles and push scooters, robotically ridden test vehicles can also be used in reliability testing to quantify the performance of detection algorithms in *other* vehicles like autonomous cars and trucks, as Persson et. al point out in [5].

[1]Integrative Engineering Program, Lafayette College, Easton, Pennsylvania, United States

[2]Department of Mechanical Engineering, Lafayette College, Easton, Pennsylvania, United States

However, developing, improving, and validating the behavior of robot-ridden STVs is difficult. In many scenarios, the design and/or validation of robotic STVs could benefit from high-fidelity, nonlinear multi-body simulations either in lieu of or in addition to physical experiments. Simulations allow for large batteries of tests to be conducted quickly, with zero risk, and at relatively low cost. This makes simulations valuable in many contexts relevant to self-driving STV design, from their interactions with road features [6] to system identification [7]. Simulations are also central to some modern types of controller design, e.g. reinforcement learning [8], [9]. While using low-order models for simulations is sufficient in some contexts, having an open-source, nonlinear, multi-body option allows researchers to access high-fidelity dynamic simulations as they develop self-driving motorcycles in efforts to improve rider and roadway safety. Open-source options, in particular, dramatically improve both accessibility and replicability of this type of work.

Efforts to model STV dynamics and control using high-fidelity multi-body simulations are not new, with plentiful examples that use custom simulation source code, e.g. [10], or commercial multi-body software [11]–[15]. However, the use of *open-source* multi-body simulation environments in the design process for a self-driving STV is relatively scarce. A head-to-head comparison between experimental data from a self-driving STV and simulation results from a mature open-source multi-body simulation package appears to be totally absent from the literature.

To address this gap, this paper explores whether the open source Webots robot physics simulator developed in [16] and maintained in [17] is capable of accurately representing the dynamics of a self-driving PTW. In a prior study [6], we investigated whether Webots produces results consistent with canonical models of motorcycle behavior. For the most part it does, which is encouraging for researchers who either prefer to use or *must use* open-source software in their work. However, this initial study did not explore how Webots's predictions compare with experimental data. Therefore, after reviewing related work in Section II, we present a small-scale test vehicle, a low-order model used for controller design, the design of a controller to stabilize the vehicle, and a multi-body Webots representation of the vehicle in Section III. Section IV presents experimental results from a step response in commanded roll angle for the vehicle, along with a more complex teleoperation experiment including automated starting and stopping using a servo-actuated kickstand. We then compare the experimental

results from those tests with simulation results from Webots, finding that Webots not only predicts the stability of the closed-loop system, but also that its predictions are consistent with the performance of the actual vehicle in experimental testing.

II. Related Work

In studies spanning the last three decades and using a host of different control techniques and models, researchers have designed controllers for self-driving STVs and validated their work primarily in simulation [18]–[26]. A substantially fewer number of studies used simulations as an intermediate validation step, but also included results of real experiments with self-driving STVs [27]–[32]. To accomplish self-stabilization and/or path following, some researchers' vehicles relied only on steering action (torque and/or angle) to balance the vehicle, capturing the primary way in which a human rider stabilizes a STV, such as in [18], [19], [21], [23], [28], [30], [32]. However, reflecting the fact that riders can also produce corrective torque with their bodies, some autonomous STVs were equipped with a secondary balancing system like a reaction wheel [20], [24], [25], [27], [29] or an inverted pendulum "balancer" [33].

In simulation, in experiment, or both, prior studies have tested their controllers' performance using several different types of tests, including initial condition or homogeneous responses, e.g. [19], [21], [23], [24], [32], step responses in desired roll angle, e.g. [25], [30], and even vehicle path tracking in [20], [26], [27], [31]. In the studies surveyed, most of the simulations used to predict controller behavior were based on (relatively) low-order models of vehicle dynamics [18], [19], [21]–[23], [27]–[30], [32], with far fewer using multi-body simulations in a controller validation context [11]–[15], [20].

In contrast to studies dealing specifically with the design of self-driving STVs, multi-body simulations are much more common in studies that simulate human-ridden STVs. These studies often explore advanced rider assist technology [11], [12]. While the BikeSim commercial software package is common for this purpose, other studies used Adams, another powerful, custom muti-body dynamics simulation software [13]–[15], [34].

Of the literature surveyed dealing with self-driving STVs, only one study used an open-source multi-body simulation package, Gazebo [20], in the design of an autonomous STV. This indicates that although there is utility in and precedent for using nonlinear, multi-body simulations for studying STV dynamics, its use in self-driving STV design is uncommon. Low-order models, which are most common in the self-driving STV literature, are certainly suitable for the validation of some controller types. However, nonlinear, multi-body simulations can provide more realistic tests of controller performance. This means that increasing and improving access to multi-body simulation software could help speed up the development of self-driving STVs. If an open-source software package like Webots is suitable for this task, it provides a no-cost way for researchers to test their vehicles and controllers in a realistic environment before implementation.

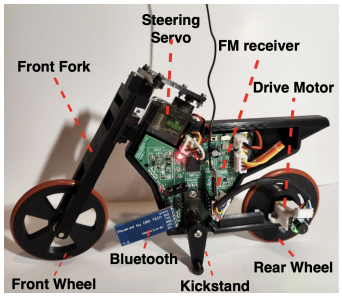

Fig. 1. 1:10th scale test vehicle

III. Methodology

A. Test Vehicle

In order to evaluate Webots's suitability for realistically simulating the dynamics of single-track vehicles under closed loop control, we designed and built a small test vehicle for this study. It consists of a chassis, front frame, and wheels constructed from Polylactic Acid (PLA) plastic using a Fused-Deposition Modeling (FDM) 3D printer.

The vehicle's tires are 2-inch diameter rubber O-rings. Its front frame and front wheel ride on two sealed ball bearings each. The vehicle's rear wheel is driven by a 6V-capable micro gearmotor with a built-in encoder mounted to the chassis. Steering action is achieved using a parallel linkage from the steered frame to a MG90s micro servo modified to allow position feedback to be read by the vehicle's microcontroller. A Second MG90s servo is used to actuate a kickstand to allow the vehicle to start and stop on its own as it receives goal roll angle and start/stop commands from a 433 MHz radio transmitter-receiver pair. Data are transmitted to a laptop computer via an HC-05 Bluetooth module.

We designed a custom Printed Circuit Board (PCB) for the vehicle with a DRV8838 bidirectional motor driver for the rear wheel, a 5V power regulation system receiving input voltage from a 3.7V, 600 mAh lithium battery, a Bosch BNO055 Inertial measurement unit, and an Atmel AT32U4 8-bit processor running at 16MHz. The firmware we developed for the AT32U4 Microcontroller allows for a control loop time of roughly 0.01 seconds.

The vehicle was built to be an approximate 1:10 scale dimensional replica of a 450cc racing motorcycle. Using a small-scale vehicle for this study both reduces cost and increases the demand on the Webots software when considering simulation fidelity, since smaller vehicles have faster open-loop unstable eigenvalues according to the model in Section III-B, and because small-scale test vehicles almost always have lower quality sensing and actuating equipment than more expensive, full-size vehicles. Additionally, the utility of using small-scaled vehicles in the development of intelligent vehicle technology is well known (e.g. [35]), with specific application examples using scaled vehicles in localization technology [36],

TABLE I
VEHICLE PARAMETERS

Symbol	Parameter	Value
b	Wheelbase	0.16 m
c	Trail	0.008 m
λ	Steer axis inclination	1.04 rad
h	Height of total mass center	0.060 m
a	x-coordinate of total mass center	0.0676 m
m_t	Total mass	0.161 kg

human-vehicle interaction [37] and dynamic maneuvers like lane changes [38] providing further support for the value of small vehicles for technology evaluation and development. The parameters of our test vehicle relevant to controller design and Webots model implementation are shown in Table I.

B. Low-Order Vehicle Model

To facilitate a controller design for the test vehicle described in Section III-A, a linear, second-order vehicle model was used. This model is essentially consistent with the "inverted pendulum" type models such as those used in [5], [39], and [40]. Our model, which we use for controller design only, assumes that only the rear frame of the vehicle has significant mass, that the tires maintain a no-slip contact point with the ground and have infinitely thin "knife" edges, that the vehicle is moving at a constant forward speed U, that it only deviates slightly from the "straight running" condition, and that all of the vehicle's mass is concentrated so that it is well-represented by a point mass as shown in Figure 2. This model is limited in its predictive capability, but if high-fidelity multi-body simulations in the Webots robot simulation software can justify its use, controllers based on this model can be suitable for implementation on the test vehicle in physical experiments, as we show in Section IV.

Examining the coordinate system for the vehicle model shown in Figure 2, the model's equation of motion about the vehicle model's only true energetic degree of freedom is in the roll direction, $\hat{\imath}$. While the vehicle is permitted to yaw with a rate $\dot{\psi}$ about the $\hat{k}$ axis, the yaw angular velocity of the vehicle is fully described (constrained) by its steering motion about the $\hat{e}_\lambda$ direction. The vehicle's forward velocity U contributes to its kinetic energy as well, but its speed is assumed constant for the purposes of this model. Ignoring small contributions to yaw rate from steering rate $\dot{\delta}$ and considering only contributions from steer angle δ, yaw rate can be approximated as a function of the steering angle δ, along with its steer axis inclination, wheelbase, and forward speed, as shown in Equation 1. This equation is a consequence of the vehicle's kinematics while turning and relies on the no-slip assumption for contact between the ground and the vehicle's tires.

$$\dot{\psi} = \frac{U \sin\lambda}{b}\delta \tag{1}$$

Using LaGrange's method to derive the system's equation of motion under the assumptions outlined above, we can use the roll angle ϕ as the system's single generalized coordinate. Writing the vehicle mass center's velocity with respect to the origin of the body-fixed coordinate system defined in Figure 2 yields Equation 2.

$$\vec{v} = U\hat{\imath} + \left(a\dot{\psi} - h\dot{\phi}\right)\hat{\jmath} \tag{2}$$

Assuming that changes in the height of the vehicle's center of mass due to trail c and steer angle δ are small enough to be neglected, the vehicle's potential energy is defined fully by its mass and the roll angle ϕ. This means that the vehicle's Lagrangian $L \equiv T - V$ is given by Equation 3.

$$L = \frac{1}{2}m\left(U^2 + a^2\dot{\psi}^2 + h^2\dot{\phi}^2 - 2ah\dot{\psi}\dot{\phi}\right) - mgh\cos\phi \tag{3}$$

Assuming small δ and ϕ angles and applying the approximate relationship between $\dot{\psi}$ and δ given in Equation 1 allows for the computation of the approximate Euler-LaGrange equation about the ϕ degree of freedom for the system. Applying this linearization after computing the relevant derivatives of L yields Equation 4.

$$\begin{aligned} \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\phi}}\right) - \left(\frac{\partial L}{\partial \phi}\right) &= \quad \tau_\phi \\ \ddot{\phi}\left(mh^2 - \frac{U\sin\lambda}{b}mah\dot{\delta}\right) - mgh\phi &= \quad \tau_\phi \end{aligned} \tag{4}$$

In Equation 4, the two non-conservative torques τ_ϕ we consider in our model are both functions of steer angle input δ. The first is due to the D'Alembert inertial reaction at the vehicle's mass center from the vehicle's centripetal acceleration, and the second is due to the lateral shift in the vehicle's mass center when it is steered. These torques, linearized for small angles δ, ϕ are given in Equation 5. More detailed treatments of this model and these specific torques can be found elsewhere in the literature, such as in [40].

$$\tau_\phi = \frac{mhU^2\sin\lambda}{b}\delta - \frac{mgac\sin\lambda}{b}\delta \tag{5}$$

Substituting the non-conservative torques from Equation 5 into Equation 4 yields the final linearized equation of motion for the system model. Converted to transfer function form and representing the linearized open-loop dynamics of the vehicle from steer input δ to roll angle ϕ, the final model is shown in Equation 6. Note that the vehicle mass m can be canceled from the model entirely since the vehicle's mass is assumed to be concentrated at a point.

$$P(s) = \frac{\phi(s)}{\delta(s)} = \frac{\frac{\sin\lambda}{b}\left(\frac{aU}{h}s + \frac{U^2}{h} - \frac{gac}{h^2}\right)}{s^2 - \frac{g}{h}} \tag{6}$$

This model is one of the simplest possible that describes a STV's behavior as it is steered, and is only stable under active control. However, as we will show in Section IV, it is sufficient for the controller design and experiments in this study. The success of this model in stabilizing the test vehicle also offers

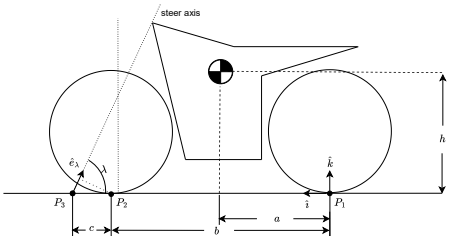

Fig. 2. Low-Order Model Dimensions and Configuration

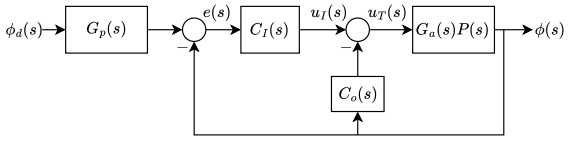

Fig. 3. Modified PID control structure

a second independent evaluation of the Webots multi-body simulation software's predictions by facilitating comparisons of the Webots simulation's predictions with the closed loop transfer function of the controlled vehicle.

C. Feedback Controller Design

This study's goal is to determine whether the multi-body Webots robot simulation software captures the key physical processes involved in single-track vehicle stabilization well enough to be used for controller design and evaluation. To ensure that the results of the validation experiments described in Section III-E are repeatable and accessible to a variety of audiences, a classical control structure was chosen to stabilize the vehicle as modeled by Equation 6 for the test vehicle's top speed, which is approximately $U = 0.9m/s$. Any number of control algorithms from the literature summarized in Section II could be used in its place, but using a classical, linear controller that relies on a low-order model offers a challenge for both physical implementation and for simulating vehicle dynamics in the Webots software. This is mainly because linear, classical control is a relatively "fragile," less robust approach to stabilization of a nonlinear, open-loop unstable system when compared with an approach that uses learning (e.g. reinforcement learning or an artificial neural network), a robust linear or nonlinear control approach, or even with a linear controller that has access to full state feedback.

We chose to use a modified Proportional-Integral-Derivative (PID) control scheme, shown graphically in Figure 3. This controller uses system output for the proportional and derivative portion of the controller, $C_o(s)$, and error for the integral portion $C_I(s)$. This approach eliminates excessive overshoot that often results from "true" PID control's two control zeros in the forward path of a system's closed-loop transfer function. The control scheme also includes a "pre-filter" transfer function $G_p(s)$ applied to the desired vehicle roll angle to smooth out abrupt transitions in the system setpoint.

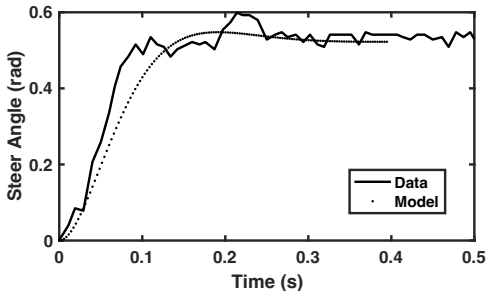

Fig. 4. Steering actuator model vs. experimental data for a 30°step in desired steer angle

This controller configuration has a transfer function from reference input $\phi_d(s)$ to roll angle $\phi(s)$ given by Equation 7, where $P(s)$ is the open-loop transfer function of the vehicle from steer angle to roll angle, given in Equation 6.

$$\frac{\phi(s)}{\phi_d(s)} = \frac{G_p(s)C_I(s)G_a(s)P(s)}{1 + (C_I(s) + C_o(s))\,G_a(s)P(s)} \tag{7}$$

A simple first order transfer function with a time constant $\tau = 0.25s$ and a steady-state gain of 1 was used for $G_p(s)$, which is shown in Equation 8.

$$G_p(s) = \frac{1}{\tau s + 1} \tag{8}$$

The plant transfer function of the vehicle $P(s)$ shown in Figure 3 and Equation 7 is outlined in Equation 6. In addition to the vehicle's nominal open-loop behavior, actuator dynamics representing the MG90s servo on the test vehicle were included in the control loop's forward path. The actuator was modeled as a second-order transfer function in standard form, shown in Equation 9.

$$G_a(s) = \frac{\omega_a^2}{s^2 + 2\zeta_a\omega_a s + \omega_a^2} \tag{9}$$

The values of $\zeta_a = 0.7$ and $\omega_a = 23rad/s$ were fit to data from a 30° step response for the MG90s servo that actuates the vehicle's steering axis while the vehicle's tires were resting on the ground. The results of this test, along with the model fit for $G_a(s)$ are shown in Figure 4. The control transfer functions $C_I(s) = \frac{K_I}{s}$ and $C_o(s) = K_p + K_d s$ represent the I (integral) and PD (proportional-derivative) portions of the standard PID control law, respectively. These are shown in Equation 10.

$$\begin{aligned} C_I(s) &= \frac{K_I}{s} \\ C_o(s) &= K_p + K_d s \end{aligned} \tag{10}$$

Note that in Equation 7, the closed-loop characteristic equation, defined by the transfer function's denominator, shows the sum of $C_I(s)$ and $C_o(s)$ multiplied by the vehicle's plant and the actuator dynamics. Defining $C_T(s) = C_I(s) + C_o(s)$ allows for the recovery of the "standard" PID control law, which means that the derivative gain K_d can be factored out of the controller transfer function $C_T(s)$. This allows the system's closed-loop characteristic equation to be written in

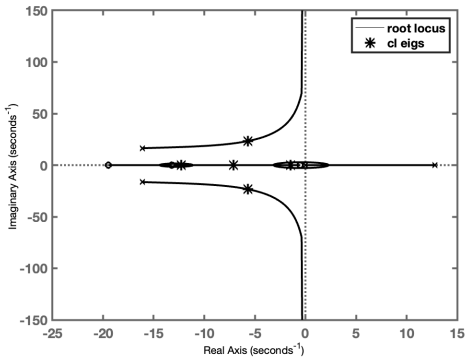

Fig. 5. Root Locus of Vehicle-Actuator System under PID Control

terms of two control zeros z_1 and z_2 as shown in Equation 11.

$$\frac{-1}{K_d} = \frac{(s+z_1)(s+z_2)}{s} G_a(s)P(s) \tag{11}$$

In Equation 11, $z_1 z_2 = \frac{K_i}{K_d}$ and $z_1 + z_2 = \frac{K_p}{K_d}$. This formulation of the system's characteristic equation facilitates the construction of a root locus plot for the system given knowledge of the two control zero locations, and allows for K_d to be treated as the system's root locus gain. Placing the control zeros at $z_1 = 0.5$ and $z_2 = 19.5$ ensures that the root locus branches from the steering actuator $G_a(s)$ do not go unstable, so that the closed loop system has a theoretically infinite increased-gain margin once the gain is large enough to bring the system's open-loop unstable eigenvalues into the left-half of the complex plane. Figure 5 shows the root locus for the system under the direction of the modified PID controller designed above.

Using Figure 5, a root locus gain $K_d = 0.2$ was selected, yielding gains of $K_p = 4$ and $K_I = 2$ to complete the controller design. The closed-loop transfer function's slowest eigenvalue is real, and is located at $s = -1.4$ rad/s with the chosen controller, predicting a 2% settling time of roughly 2.86 seconds.

D. Multi-Body Webots Model

The Webots open-source robot simulation software [16] allows users to construct "robots" from rigid bodies connected by standard joints. The software's physics engine solves constraints defined by the joints and forces due to contact between bodies (e.g. tires and ground) numerically as it simulates the physics of the system. It also allows users to define a "PROTO," which is a re-usable robot configuration with adjustable parameters. In a prior study [6], we built such a PROTO to simulate STV dynamics across varying parameter values, and showed that Webots's dynamic predictions agree well with canonical models of motorcycle dynamics. The configuration of this adjustable simulation element is shown in Figure 6. For the present study, the prismatic joint P_1 representing front suspension and the revolute joint R_2 representing rear suspension were disabled because the test vehicle

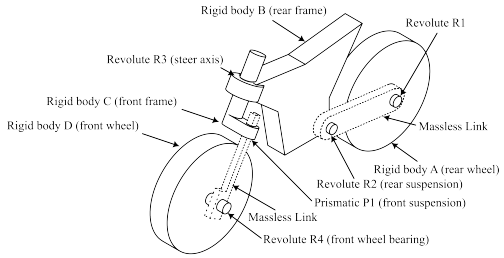

Fig. 6. Rigid body and joint configuration of test vehicle in Webots

described in Section III-A has no suspension. All other inertial and dimensional parameters in the PROTO were matched to measurements from the test vehicle for simulation in Webots. In addition to the bodies and joints shown in Figure 6, the vehicle was equipped with an automatic kickstand, built as a child of Rigid Body B in the vehicle's definition. The drive motor powering Rigid Body A (rear wheel) through Revolute R1 and the steering motor powering Rigid Body C through Revolute R3 both had dynamic parameters fit to match those of the drive motor on the physical test vehicle. The vehicle's representation in the Webots software is shown in Figure 7.

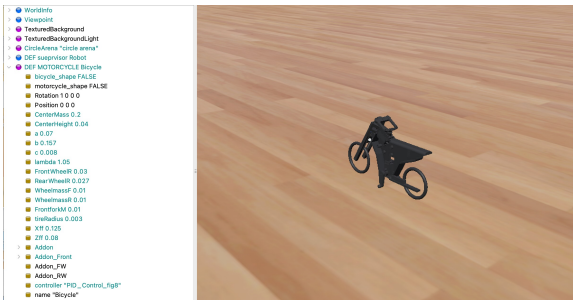

Fig. 7. Webots representation of test vehicle

For the simulations in this study, the Coulomb friction coefficient between tires and ground was set to 1.2 to match typical friction between rubber tires and a hardwood floor. The Coulomb friction coefficient between the vehicle's kickstand and ground was set to 0.2 to match the friction of our smooth plastic kickstand on a hardwood floor. The simulations described in Section III-E all used a simulation timestep of $1ms$ and a controller refresh rate of $100Hz$ to match the control loop time of the test vehicle described in Section III-A.

E. Experimental Design

Two experiments with accompanying simulations were conducted to evaluate the fidelity of the multi-body Webots simulation model described in Section III-D.

First, to compare Webots's ability to predict vehicle roll dynamics with both experimental data and with the predictions of the closed loop linear model developed in Section III-C, a 0.15 radian step response in desired roll angle was conducted at the vehicle's maximum forward speed of $0.9m/s$.

However, this simulation alone doesn't highlight Webots's ability to predict single-track vehicle dynamics outside the

approximately linear region of the vehicle's behavior. To compare Webots's predictions with experimental data in a more complex dynamic task that involves nonlinear effects like sliding friction along with vehicle longitudinal acceleration, the test vehicle was equipped with an automatic kickstand and a Finite State Machine (FSM). This FSM is summarized in Figure 8.

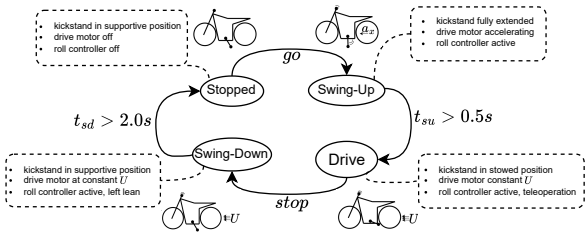

Fig. 8. Finite State Machine for automatic starting and stopping

In Figure 8, elapsed times t_{su}, t_{sd} represent the time in seconds that the FSM has been in states "Swing-Up" and "Swing-Down" respectively. The FSM's actions combine the linear controller of Section III-C with acceleration and deceleration from the rear drive motor along with the actuation of a motorized kickstand. This results in behavior that *can not* be modeled using a linear transfer function such as Equation 7. The FSM achieves "swing-up" by using a kickstand-actuated servo to thrust the vehicle upward in roll from rest as it accelerates to top speed when a "go" command is received from the vehicle's remote control. During a test run, when in the "Drive" state, the vehicle receives ϕ_d commands from the remote control as it drives and stabilizes itself. When a "stop" command is received from the remote control, the vehicle transitions into the "Swing-Down" state, in which the control system receives a -5° goal roll command, and deploys the kickstand into a supportive position so that the vehicle lands back on its kickstand safely at the completion of the experiment. Using this automatic kickstand and its accompanying FSM, a second experiment consisting of controller-assisted teleoperation was conducted with the same time-series of commands and controller signals applied to the Webots simulation to allow for comparison of Webots's predictions with experimental data. That experiment both began and concluded with the test vehicle in the "stopped" state.

IV. Results

The results of the step response test described in Section III-E are shown in Figure 9. Visual analysis of Figure 9 shows that the rise time, settling time, percent overshoot, and steady state value of the vehicle roll angle predicted by the multi-body Webots simulation match well with the data. The linear model is similarly predictive, with a slightly lower steady-state value. The linear model predicts perfect steady-state tracking of the 0.15 radian goal roll angle, while the multi-body simulation and experimental data show slight overshoot of the goal at steady state, most likely due to nonlinearities associated with violations of the knife-edged tire assumption and/or the no-slip assumption in the linear model. Because of the linear model's prediction of a 2.86-second settling time in closed loop, and because of the noise in IMU data during the experiment, we conservatively used the mean of the last 2 seconds of roll measurements to estimate steady state roll from both the Webots simulation and the step response experiment. These estimates were 0.1719 and 0.1604 radians for Webots and experiment, respectively. This means that Webots predicted the vehicle's experimental steady state roll angle within 7%. Further, the Root-Mean-Squared Error (RMSE) between Webots's roll angle predictions and the experimental data was 0.023 radians, or roughly 1.31°, which appears to be mostly due to a combination of disturbances and sensor noise present in the experimental data that are (for the present study) unmodeled in Webots.

The steering angle data in Figure 9 also show good visual agreement between experiment and Webots. However, on the test vehicle, steering angle was measured by soldering a lead onto the steering servo's integral potentiometer, which was then read by the AT32U4 processor's 10-bit analog-to-digital converter. Therefore, the steering measurement is quite noisy and not especially precise. In general, the steering data has a much worse Signal-To-Noise Ratio (SNR) than the IMU's roll angle measurement, and is also subject to disturbances during the experiment from floor surface irregularities and unmodeled effects like mechanism free play that are difficult to replicate in Webots. This explains the steering angle's far less impressive RMSE value of 0.153 radians. Still, the agreement in shape and steady state value are strong, indicating that Webots is capable of reasonably representing control effort on the self-driving STV.

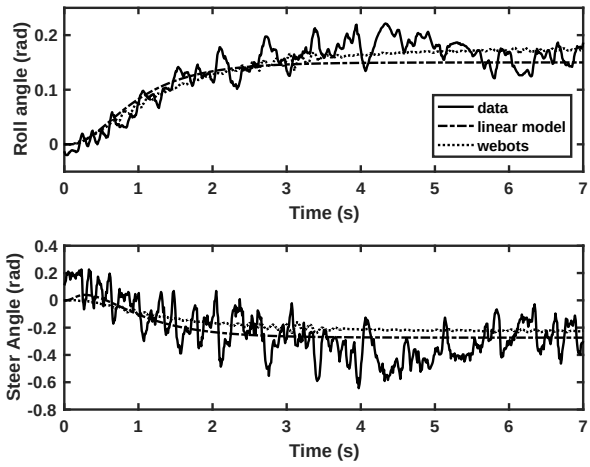

Fig. 9. Closed loop vehicle behavior for a 0.15 radian step in desired roll angle for linear model, Webots, and experiment

The results of the "free driving" tele-operation test with automatic swing-up and swing-down to and from the vehicle's kickstand, described in Section III-E, are shown in Figure 10. A visual analysis of the roll angle panel of Figure 10 shows that the multi-body webots simulation matches well

with experimental data, even during the periods of relatively high acceleration and deceleration at the beginning and end of the test. The RMSE in roll between Webots and experiment was 0.0271 radians for the 32-second test. The experimental data show a more pronounced initial oscillation in the vehicle's roll angle during the swing-up acceleration phase, which could be due in part to tire slip, which was not modeled in Webots. While not included for this study, a higher fidelity tire slip model is possible to add to the Webots simulation after a thorough characterization of the test vehicle's tires.

Although Figure 10 also shows good visual agreement between Webots and experiment for yaw rate and steer angle, both signals suffer from a poor SNR, yielding less impressive RMSE values of 0.62 radians per second and 0.15 radians, respectively. Note that no filtering was applied to these measurements in the controller's code, so none was applied to the data shown in Figure 10 either. The vehicle's speed during the experiment was predicted well by Webots with an RMSE value of 0.09 meters per second.

In summary, visual agreement between Webots's predictions for both the step response test and the teleoperation experiment are strong, even in the face of relatively noisy, low-cost sensors on our test vehicle, and even in the face of the fact that a small-scale STV's eigenvalues are faster (and therefore more difficult to replicate accurately) than those of a full-size vehicle. This suggests that this agreement is likely to generalize to larger vehicles. Numerical comparisons between Webots and experiment for vehicle roll angle are also strong in both tests. While numerical error metrics for steering and yaw rate are less impressive, this can be explained by poor SNR and sensor precision for these signals rather than a shortcoming of Webots's predictions.

V. Conclusions and Future Work

This paper explores whether the nonlinear multi-body Webots robot physics simulator is suitable for the prediction of the closed-loop behavior of PTWs and other STVs. Using a small-scale experimental PTW and a linear controller for roll stabilization, this paper showed that the multi-body Webots simulation served as a suitable intermediate controller validation step between the low-order model used for controller design and data from the physical test vehicle. Our results indicate that Webots has potential as a free, open-source alternative to commercial multi-body software as a predictor of the closed-loop performance of robotic STVs.

These results are encouraging, but they are not without limitations that warrant future investigation. We have shown that Webots can predict the behavior of one vehicle configuration under closed loop control. However, evaluations of Webots's predictions across a variety of vehicle parameter ranges and under more aggressive maneuvers are necessary before Webots could be considered a complete, equivalent replacement for commercial multi-body software. Although comparative analyses between Webots and other multi-body simulation packages would be helpful for contextualizing

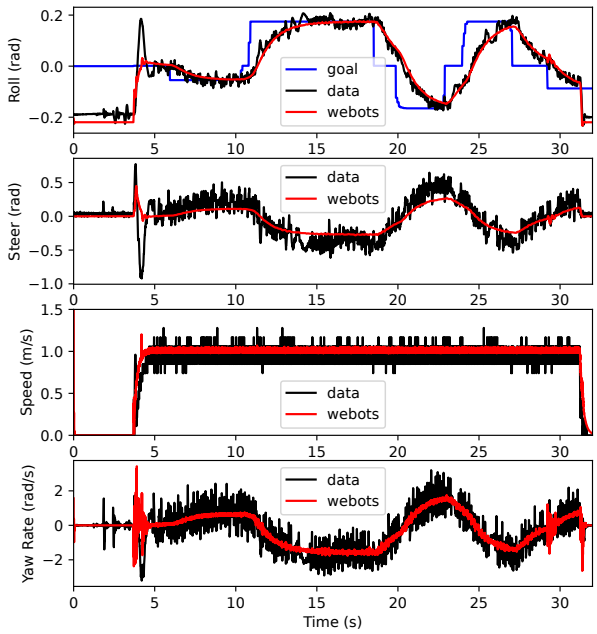

Fig. 10. Closed loop vehicle behavior during swing-up, free driving via teleoperation, and swing-down for both multi-body Webots simulation and experiment

Webots's relative accuracy, such comparisons are out of scope for the present study, and are planned for future work.

Even in light of the limitations of the experiments and simulations presented, our work shows that Webots could have been used to predict our test vehicle's closed-loop stability and dynamic responses in lieu of physical experiments. This provides support for our hypothesis that Webots offers researchers a free, open-source option for designing self-driving STVs and related STV technology in the service of improving road safety for STV pilots and the other road users who interact with them.

References

[1] A. Ragnoli, M. V. Corazza, and P. Di Mascio, "Safety ranking definition for infrastructures with high ptw flow," *Journal of traffic and transportation engineering (English edition)*, vol. 5, no. 5, pp. 406–416, 2018.

[2] T. Koshizen, F. Sato, R. Oishi, and K. Yamakawa, "Predicting motorcycle riding behavior using vehicle density variation," in *2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops)*, 2021, pp. 114–121.

[3] J. Meijaard, J. M. Papadopoulos, A. Ruina, and A. Schwab, "Linearized dynamics equations for the balance and steer of a bicycle: a benchmark and review," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 463, no. 2084, pp. 1955–1982, Jun. 2007, publisher: Royal Society. [Online]. Available: https://royalsocietypublishing.org/doi/10.1098/rspa.2007.1857

[4] G. Savino, M. Pierini, M. Rizzi, and R. Frampton, "Evaluation of an autonomous braking system in real-world ptw crashes," *Traffic injury prevention*, vol. 14, no. 5, pp. 532–543, 2013.

[5] N. Persson, T. Andersson, A. Fattouh, M. C. Ekström, and A. V. Papadopoulos, "A Comparative Analysis and Design of Controllers for Autonomous Bicycles," in *2021 European Control Conference (ECC)*, 2021, pp. 1570–1576.

[6] W. Li, P. Francis, R. McClosky, and A. Brown, "Webikes: a configurable, open-source add-on for simulating single-track vehicle dynamics in the webots robot simulation software," *Advances in Transportation Studies*, vol. Special Vol. 4, pp. 73–88, 2024.

[7] M. Fouka, L. Nehaoua, H. Arioui, and S. Mammar, "Motorcycle inertial parameters identification via algorithmic computation of state and design sensitivities," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 3926–3929.

[8] S. Vadlamudi, K. V. Lakshmi, A. Yaramala, C. Uppalapati, and P. K. Kolluri, "Self balancing motorcycle using reinforcement learning," in *2024 International Conference on Emerging Systems and Intelligent Computing (ESIC)*. IEEE, 2024, pp. 691–696.

[9] K. V. Lakshmi and M. Manimozhi, "Implementation of ddpg based reinforcement learning control for self-balancing motorcycle," *IEEE Access*, 2024.

[10] D. Limebeer, R. Sharp, and S. Evangelou, "The stability of motorcycles under acceleration and braking," *Proceedings of The Institution of Mechanical Engineers Part C-journal of Mechanical Engineering Science - PROC INST MECH ENG C-J MECH E*, vol. 215, pp. 1095–1109, 09 2001.

[11] M. Pryde, O. Alrazouk, L. Nehaoua, H. Hadj–Abdelkader, and H. Arioui, "Visual-inertial lateral velocity estimation for motorcycles using inverse perspective mapping," in *2022 17th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2022, pp. 217–222.

[12] P.-M. Damon, H. Hadj-Abdelkader, H. Arioui, and K. Youcef-Toumi, "Powered two-wheeled vehicles steering behavior study: Vision-based approach," in *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2018, pp. 355–360.

[13] M. R. Garcia, D. A. Mántaras, J. C. Alvarez, and D. Blanco, "Stabilizing an urban semi-autonomous bicycle," *IEEE Access*, vol. 6, pp. 5236–5246, 2018.

[14] V. Del Rosso, A. Andreucci, S. Boria, M. L. Corradini, R. Giambò, and A. Ranalli, "Modelling and control of a self-balancing electric motorcycle: Preliminary results," in *2018 26th Mediterranean Conference on Control and Automation (MED)*, 2018, pp. 867–872.

[15] N. Persson, M. C. Ekström, M. Ekström, and A. V. Papadopoulos, "Trajectory tracking and stabilisation of a riderless bicycle," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021, pp. 1859–1866.

[16] O. Michel, "Webots: Professional mobile robot simulation," *Journal of Advanced Robotics Systems*, vol. 1, no. 1, pp. 39–42, 2004. [Online]. Available: http://www.ars-journal.com/International-Journal-of- Advanced-Robotic-Systems/Volume-1/39-42.pdf

[17] Webots, "http://www.cyberbotics.com," open-source Mobile Robot Simulation Software. [Online]. Available: http://www.cyberbotics.com

[18] B. Xing, L. Guo, S. Wei, and Y. Song, "Dynamic modeling and robust controller design for circular motion of a front-wheel drive bicycle robot," in *2016 IEEE International Conference on Mechatronics and Automation*, Aug. 2016, pp. 1369–1373, iSSN: 2152-744X. [Online]. Available: https://ieeexplore.ieee.org/document/7558762

[19] Y. Sun, M. Zhao, B. Wang, X. Zheng, and B. Liang, "Polynomial Controller for Bicycle Robot based on Nonlinear Descriptor System," in *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society*, Oct. 2020, pp. 2792–2797, iSSN: 2577-1647. [Online]. Available: https://ieeexplore.ieee.org/document/9254572

[20] X. Zhu, X. Zheng, Q. Zhang, Z. Chen, Y. Liu, and B. Liang, "Natural Residual Reinforcement Learning for Bicycle Robot Control," in *2021 IEEE International Conference on Mechatronics and Automation (ICMA)*, Aug. 2021, pp. 1201–1206, iSSN: 2152-744X. [Online]. Available: https://ieeexplore.ieee.org/document/9512587

[21] S. Vatanashevanopakorn and M. Parnichkun, "Steering control based balancing of a bicycle robot," in *2011 IEEE International Conference on Robotics and Biomimetics*, Dec. 2011, pp. 2169–2174. [Online]. Available: https://ieeexplore.ieee.org/document/6181613

[22] G. Lei, H. Kai, S. Yuan, and X. Bin, "Robust controller design for 90 degrees stand motion of a front-wheel drive bicycle robot," in *2016 IEEE International Conference on Mechatronics and Automation*, Aug. 2016, pp. 1356–1362, iSSN: 2152-744X. [Online]. Available: https://ieeexplore.ieee.org/document/7558760

[23] H. D. Sharma and N. UmaShankar, "A Fuzzy Controller Design for an Autonomous Bicycle System," in *2006 IEEE International Conference on Engineering of Intelligent Systems*, Apr. 2006, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/1703218

[24] D. A. Bravo M., C. F. Rengifo R., and J. F. Díaz O., "Comparative Analysis between Computed Torque Control, LQR Control and PID Control for a Robotic Bicycle," in *2019 IEEE 4th Colombian Conference on Automatic Control (CCAC)*, Oct. 2019, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/8920870

[25] A. Suebsomran, "Balancing control of bicycle robot," in *2012 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, May 2012, pp. 69–73. [Online]. Available: https://ieeexplore.ieee.org/document/6392529

[26] N. Getz and J. Marsden, "Control for an autonomous bicycle," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 2, 1995, pp. 1397–1402 vol.2.

[27] N. Aphiratsakun and K. Techakittiroj, "Autonomous AU Bicycle: Self-Balancing and tracking control (AUSB2)," in *2013 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec. 2013, pp. 480–485. [Online]. Available: https://ieeexplore.ieee.org/document/6739505

[28] C. Xiong, Z. Huang, W. Gu, Q. Pan, Y. Liu, X. Li, and E. X. Wang, "Static Balancing of Robotic Bicycle through Nonlinear Modeling and Control," in *2018 3rd International Conference on Robotics and Automation Engineering (ICRAE)*, Nov. 2018, pp. 24–28. [Online]. Available: https://ieeexplore.ieee.org/document/8586765

[29] N. He, Q. Hu, J. Chen, and K. Zhang, "Self-Balancing Bicycle Based on Control Moment Gyroscope," in *2023 42nd Chinese Control Conference (CCC)*, Jul. 2023, pp. 4493–4498, iSSN: 1934-1768. [Online]. Available: https://ieeexplore.ieee.org/document/10240993

[30] L. Guo, Q. Liao, S. Wei, and Y. Zhuang, "Design of linear quadratic optimal controller for bicycle robot," in *2009 IEEE International Conference on Automation and Logistics*, Aug. 2009, pp. 1968–1972, iSSN: 2161-816X. [Online]. Available: https://ieeexplore.ieee.org/document/5262628

[31] F. Li, "Research on self-balancing unmanned bicycle based on cascade PID control," in *2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA)*, vol. 2, Dec. 2021, pp. 442–446. [Online]. Available: https://ieeexplore.ieee.org/document/9687999

[32] L. Guo, Q. Liao, and S. Wei, "Design of Fuzzy Sliding-mode Controller for Bicycle Robot Nonlinear System," in *2006 IEEE International Conference on Robotics and Biomimetics*, Dec. 2006, pp. 176–180. [Online]. Available: https://ieeexplore.ieee.org/document/4141860

[33] L. Keo and M. Yamakita, "Controller design of an autonomous bicycle with both steering and balancer controls," in *2009 IEEE Control Applications, (CCA) & Intelligent Control, (ISIC)*, Jul. 2009, pp. 1294–1299, iSSN: 1085-1992. [Online]. Available: https://ieeexplore.ieee.org/document/5281010

[34] Y. Feng, R. Du, and Y. Xu, "Steering angle balance control method for rider-less bicycle based on adams," in *Proceedings of the Second International Conference on Intelligent Transportation*. Springer, 2017, pp. 15–31.

[35] G. Özbilgin, A. Kurt, and Özgüner, "Using scaled down testing to improve full scale intelligent transportation," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, 2014, pp. 655–660.

[36] M. S. Hamza, O. M. Shehata, E. I. Morgan, and C. M. Elias, "Vision-based indoor positioning system for connected vehicles in small-scale testbed environments," in *2024 IEEE Intelligent Vehicles Symposium (IV)*, 2024, pp. 144–148.

[37] P. Scheffe and B. Alrifaee, "A scaled experiment platform to study interactions between humans and cavs," in *2023 IEEE Intelligent Vehicles Symposium (IV)*, 2023, pp. 1–6.

[38] W. He, X. Wang, G. Chen, M. Guo, T. Zhang, P. Han, and R. Zhang, "Monocular based lane-change on scaled-down autonomous vehicles," in *2011 IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 144–149.

[39] S. Vatanashevanopakorn and M. Parnichkun, "Steering control based balancing of a bicycle robot," in *2011 IEEE International Conference on Robotics and Biomimetics*. IEEE, 2011, pp. 2169–2174.

[40] K. J. Astrom, R. E. Klein, and A. Lennartsson, "Bicycle dynamics and control: adapted bicycles for education and research," *IEEE Control Systems Magazine*, vol. 25, no. 4, pp. 26–47, 2005.