Kinematic and dynamic vehicle models for autonomous driving control design

Conference Paper · June 2015

DOI: 10.1109/IVS.2015.7225830

4 authors, including:

Jason Kong
University of California, Berkeley
10 PUBLICATIONS **948** CITATIONS
SEE PROFILE

Mark Pfeiffer
ETH Zurich
18 PUBLICATIONS **1,170** CITATIONS
SEE PROFILE

Georg Schildbach
Elektronische Fahrwerksysteme GmbH / Audi AG
27 PUBLICATIONS **1,681** CITATIONS
SEE PROFILE

2015 IEEE Intelligent Vehicles Symposium (IV)
June 28 - July 1, 2015. COEX, Seoul, Korea

Kinematic and Dynamic Vehicle Models for Autonomous Driving Control Design

Jason Kong[1], Mark Pfeiffer[2], Georg Schildbach[1], Francesco Borrelli[1]

Abstract— **We study the use of kinematic and dynamic vehicle models for model-based control design used in autonomous driving. In particular, we analyze the statistics of the forecast error of these two models by using experimental data. In addition, we study the effect of discretization on forecast error.**

We use the results of the first part to motivate the design of a controller for an autonomous vehicle using model predictive control (MPC) and a simple kinematic bicycle model. The proposed approach is less computationally expensive than existing methods which use vehicle tire models. Moreover it can be implemented at low vehicle speeds where tire models become singular. Experimental results show the effectiveness of the proposed approach at various speeds on windy roads.

I. Introduction

Work in autonomous vehicles has grown dramatically in the past several years due to advances in computing and sensing technologies. However, the idea of an autonomous vehicle has been around as early as the 1920s [1]. Recent competitions [2]–[4] have accelerated the research in this area and helped advance sensors and algorithms needed to design an autonomous vehicle.

The main components of a modern autonomous vehicle are localization, perception, and control. This paper will discuss the control of the vehicle's acceleration, brake, and steering using Model Predictive Control (MPC). In MPC [5] at each sampling time step, starting at the current state, an open-loop optimal control problem is solved over a finite horizon. The optimal command signal is applied to the process only during the following sampling interval. At the next time step a new optimal control problem based on new measurements is solved over a shifted horizon. The optimal solution relies on a dynamic model of the process, respects input and output constraints, and minimizes a performance index.

MPC has been successful in semi-autonomous and autonomous driving [6]–[10]. In fact, MPC is suited for this class of problems since it can handle Multi-Input Multi-Output systems with input and state constraints while taking into account the nonlinear dynamics of the vehicle. Moreover, the design task of path-generation and path-following is simplified: the reference path fed to the MPC controller does not need to be physically realizable, i.e respect dynamic constraints.

Published MPC schemes use dynamic vehicle models [11, p. 15-46] combined with linear [10], Pacejka [8], and Fiala [7] tire models. This approach has two disadvantages: it is computationally expensive and any tire model becomes singular at low vehicle speeds. Tire models use a tire slip angle estimation term which has the vehicle velocity in the denominator. This prohibits the use of the same control design for stop-and-go scenarios, common in urban driving.

In this paper, we propose to address both disadvantages by using a kinematic bicycle model. The paper has two contributions: In the first part, a comparison between a kinematic and dynamic bicycle model. The dynamic model utilizes a linear tire model to describe the wheel/ground interaction. We examine how well the two models are able to predict a vehicle's future states compared to measured states of an experimental test. We also study the effect of sampling time on forecast error. Our unintuitive results show that the error statistics are comparable for both models especially when the kinematic model is sampled at a slower rate. We present data and provide an explanation of the findings.

In the second part, we use the results of the first part to motivate the design of a controller for an autonomous vehicle using model predictive control (MPC) and a kinematic bicycle model. The proposed approach is less computationally expensive than existing methods which use vehicle tire models. Moreover it can be implemented at low vehicle speeds where tire models become singular. We present experimental results which show that the proposed controller provides satisfactory control performance in a simulation and a wide range of experiments.

The paper is organized as follows. In Section II, an overview of the kinematic bicycle model and dynamic bicycle model are discussed. These two models are compared in Section III. Section IV presents our proposed MPC formulation. Section V discusses the experimental setup. The experimental results of the controller are presented in Section VI. We conclude in Section VII.

II. Vehicle Models

A. Kinematic Bicycle Model

The nonlinear continuous time equations that describe a kinematic bicycle model [11, p. 26] (see Figure 1) in an

*This material is based upon work partially supported by the National Science Foundation under Grant No. 1239323. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The work was also supported by the Hyundai center of Excellence at UC Berkeley.

[1]Department of Mechanical Engineering, University of California, Berkeley, {jasonjkong, schildbach, fborrelli}@berkeley.edu

[2]Institute for Dynamic Systems and Control, ETH Zurich, mark.pfeiffer@mavt.ethz.ch

inertial frame are

$$\dot{x} = v\cos(\psi+\beta) \quad (1a)$$
$$\dot{y} = v\sin(\psi+\beta) \quad (1b)$$
$$\dot{\psi} = \frac{v}{l_r}\sin(\beta) \quad (1c)$$
$$\dot{v} = a \quad (1d)$$
$$\beta = \tan^{-1}\left(\frac{l_r}{l_f+l_r}\tan(\delta_f)\right) \quad (1e)$$

where x and y are the coordinates of the center of mass in an inertial frame (X, Y). ψ is the inertial heading and v is the speed of the vehicle. l_f and l_r represent the distance from the center of the mass of the vehicle to the front and rear axles, respectively. β is the angle of the current velocity of the center of mass with respect to the longitudinal axis of the car. a is the acceleration of the center of mass in the same direction as the velocity. The control inputs are the front and rear steering angles δ_f, and a. Since in most vehicles the rear wheels cannot be steered, we assume $\delta_r = 0$.

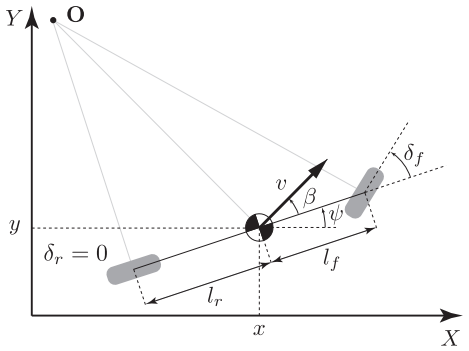

Fig. 1: Kinematic Bicycle Model

Compared to higher fidelity vehicle models, the system identification on the kinematic bicycle model is easier because there are only two parameters to identify, l_f and l_r. This makes it simpler to port the same controller or path planner to other vehicles with differently sized wheelbases.

B. Dynamic Bicycle Model

The inertial position coordinates and heading angle in the dynamic bicycle model [11, p. 27] are defined in the same manner as those in the kinematic bicycle model. The differential equations in this case are given by,

$$\ddot{x} = \dot{\psi}\dot{y} + a_x \quad (2a)$$
$$\ddot{y} = -\dot{\psi}\dot{x} + \frac{2}{m}\left(F_{c,f}\cos\delta_f + F_{c,r}\right) \quad (2b)$$
$$\ddot{\psi} = \frac{2}{I_z}\left(l_f F_{c,f} - l_r F_{c,r}\right) \quad (2c)$$
$$\dot{X} = \dot{x}\cos\psi - \dot{y}\sin\psi \quad (2d)$$
$$\dot{Y} = \dot{x}\sin\psi + \dot{y}\cos\psi, \quad (2e)$$

where $\dot{x}$ and $\dot{y}$ denote the longitudinal and lateral speeds in the body frame, respectively and $\dot{\psi}$ denotes the yaw rate. m and I_z denote the vehicle's mass and yaw inertia, respectively. $F_{c,f}$ and $F_{c,r}$ denote the lateral tire forces at the front and rear wheels, respectively, in coordinate frames aligned with the wheels.

For the linear tire model, $F_{c,i}$ is defined as

$$F_{c,i} = -C_{\alpha_i}\alpha_i, \quad (3)$$

where $i \in \{f, r\}$, α_i is the tire slip angle and C_{α_i} is the tire cornering stiffness.

III. Comparison between Dynamic and Kinematic Models

We performed an experiment where the vehicle states are measured during driving the winding track at Hyundai's California Proving Grounds (CPG) in California City, California. We also perform a large number of of N-step simulations (as many as the size of our dataset). In each simulation, a vehicle model (kinematic or dynamic) is initialized with a measured state and a N-step measured input sequence is input to the model. The models are discretized using Euler methods and are sampled at t_d =100 ms or at t_d =200 ms.

The simulated states are compared to the measured state of the vehicle to examine the accuracy of both the kinematic and dynamic model. This will be referred to as the *open-loop prediction error*.

A. Test setup

The test vehicle used was a Hyundai Azera (total wheelbase of 2.843 m, l_f =1.105 m and l_r = 1.738 m), which is front wheel driven. The localization of the car is done by using a GPS base station. An OTS (Oxford Technical Solutions) RT2002 sensing system is used which measures the position and heading of the vehicle in the inertial frame. The OTS RT2002 system is comprised of a differential GPS, an IMU (inertial measurement unit) and a DSP (digital signal processor).

The positioning errors of the OTS system vary depending on the current modes of the GPS base station. The current mode depends on the number of available satellites and the communication between the vehicle and the base station. The possible modes in our tests are RTK (real-time kinematic) integer precision ($\sigma = 0.02$ m) or differential GPS precision ($\sigma = 0.4$ m). In both modes, the precision of the measured heading angle is the same ($\sigma = 0.1$ deg).

For our experiment, the test vehicle was driven on the winding track at CPG. The winding track is a varied road course with many turns and straight line paths. This makes it an ideal testbed for our experiments to compare different models.

The average speed on this drive was 13.8 m/s, with a standard deviation of 3 m/s. The maximum driven speed was 17.1 m/s.

B. Model Comparison

Figure 2 shows distributions of open-loop prediction errors for the kinematic and dynamic bicycle model discretized at 100 ms and reported at multiples of 200 ms steps. These plots

TABLE I: Open-Loop Errors at Multiples of 200 ms (one step = 200 ms)

Model and Discretization	Mean/Std Dev Errors at 200 ms steps 1-Step	2-Step	3-Step	4-Step
Kinematic - 100 ms	0.08±0.04 m	0.16±0.06 m	0.27±0.11 m	0.40±0.19 m
Kinematic - 200 ms	0.07±0.04 m	0.14±0.05 m	0.22±0.08 m	0.33±0.13 m
Linear Tire - 100 ms	0.07±0.04 m	0.13±0.04 m	0.20±0.05 m	0.27±0.06 m
Linear Tire - 200 ms	0.08±0.04 m	0.15±0.05 m	0.23±0.06 m	0.31±0.08 m

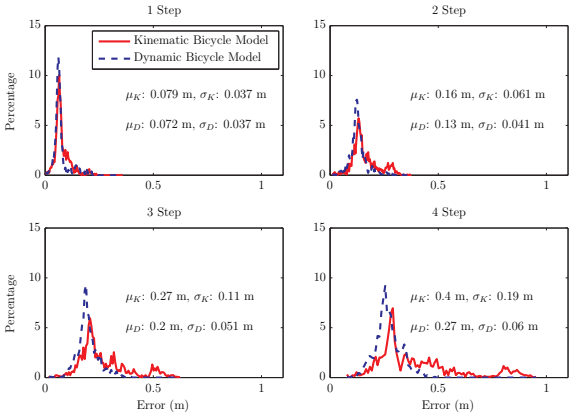

Fig. 2: Distribution of distance errors between the predictions and the actual location of the vehicle at multiple of 200 ms on the winding track (one step=200 ms). These models are discretized at 100 ms.

also show the average error and standard deviation for both models. The error distributions are almost identical for the first two steps. The kinematic model's open-loop prediction mean error at the third and fourth steps are greater than the dynamic model's open loop prediction error by 30% and 48%, with a larger standard deviation for both steps.

These results are summarized in Table I. This table also shows the mean and standard deviation at different discretization times. We notice that a higher discretization time helps the kinematic model and produces smaller errors and standard deviations. This is explained next.

C. Discretization Effects

Larger discretization times allow the model to predict a longer horizon whereas shorter discretization times tend to produce more accurate predictions. Thus, there is often a tradeoff between a longer horizon and accuracy. However, this is not the case for the kinematic model as shown in our study of the effect of discretization. It is observed that the accuracy of the kinematic bicycle can be improved with a longer discretization time.

Figure 3 shows a comparison of a continuous kinematic model, Euler discretized kinematic model at 200 ms, and a continuous dynamic model. In this figure, a constant velocity and steering angle is used to forward propagate these three models. It is observed that the gap between a discretized kinematic model and a continuous dynamic model is smaller than the gap between a continuous kinematic model and a continuous dynamic model. Up to a point, if the discretization is further increased, this gap will be further decreased.

The dynamic model considers slip and slip at the front axle (understeering) generates wider turns. Since the truncation errors from a discretized kinematic model cause a wider turn, this allows the discretized kinematic model to generate predictions more similar to a dynamic model.

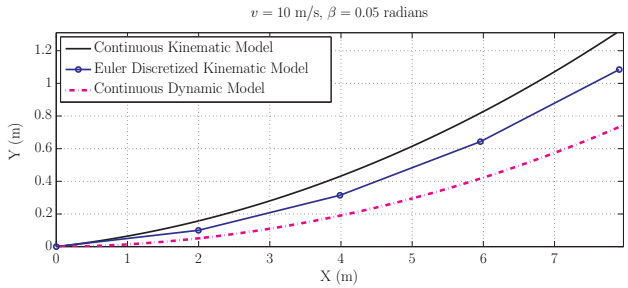

Fig. 3: Comparison of a continuous kinematic and dynamic bicycle models against a kinematic model discretized at 200 ms

Discretization effects on the open-loop prediction error on the winding track experiment are shown in Table I. It is observed that the kinematic bicycle model discretized at 200 ms has smaller open-loop prediction errors compared to discretizing at 100 ms. The dynamic model with a linear tire model is also examined, and here the errors do not benefit from a longer discretization time. At the fourth step, 800 ms, the kinematic model discretized at 200 ms has an improvement of 17% compared being discretized at 100 ms.

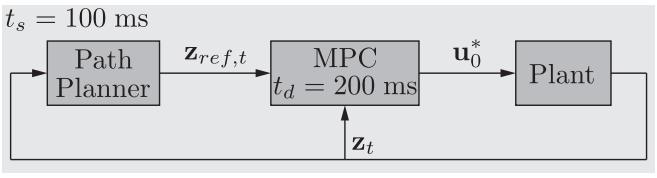

Fig. 4: Simplified control architecture. The controller and the path planner are run at every 100 ms. The discretization time of the model used in the MPC controller is 200 ms.

IV. Model Predictive Controller

A. Control Design

The model comparisons done in Section III motivate the design of an MPC controller using a kinematic bicycle model. The diagram in Figure 4 shows a simplified architecture with a high level algorithm that generates a path which is followed by a low level MPC controller.

At each sampling time step, $t_s = 100\,\text{ms}$, the MPC controller solves the following constrained finite-time optimal control problem:

$$\min_{\mathbf{U}} \quad \sum_{i=0}^{H_p} (\mathbf{z}_i - \mathbf{z}_{ref,i})^T Q (\mathbf{z}_i - \mathbf{z}_{ref,i}) + \sum_{i=0}^{H_p-1} \left[(\mathbf{u}_i - \mathbf{u}_{i-1})^T \bar{R} (\mathbf{u}_i - \mathbf{u}_{i-1}) + \mathbf{u}_i^T R \mathbf{u}_i \right] \tag{4a}$$

$$\text{s.t.} \quad \mathbf{z}_0 = \mathbf{z}(t), \mathbf{u}_{-1} = \mathbf{u}(t - t_s) \tag{4b}$$

$$\mathbf{z}_{i+1} = f(\mathbf{z}_i, \mathbf{u}_i),\ i = 0, \ldots, H_p - 1 \tag{4c}$$

$$\mathbf{u}_{min,i} \leq \mathbf{u}_i \leq \mathbf{u}_{max,i},\ \forall i \tag{4d}$$

$$\dot{\mathbf{u}}_{min,i} \leq \tfrac{\mathbf{u}_i - \mathbf{u}_{i-1}}{\mathrm{t}_d} \leq \dot{\mathbf{u}}_{max,i},\ \forall i \backslash 0 \tag{4e}$$

$$\dot{\mathbf{u}}_{min,i} \leq \tfrac{\mathbf{u}_i - \mathbf{u}_{i-1}}{\mathrm{t}_s} \leq \dot{\mathbf{u}}_{max,i},\ i = 0 \tag{4f}$$

where $Q \in \mathbb{R}_+^{4\times4}$, $R \in \mathbb{R}_+^{2\times2}$, $\bar{R} \in \mathbb{R}_+^{2\times2}$, H_p is the prediction horizon. Q, R and $\bar{R}$ are diagonal matrices comprising the cost weights. $\mathbf{u}_{-1}$ is the input applied at the previous sampling step, $\mathbf{u}(t - t_s)$. t_s is the sampling time of the controller, and t_d = 200 ms is the discretization time of the model used in the MPC controller. A mismatch between t_s and t_d was introduced in order to achieve a longer prediction time for the same prediction horizon in the controller while preserving a higher state-feedback update rate. $f(\mathbf{z}_i, \mathbf{u}_i)$ represents the dynamics update with the kinematic bicycle model through forward Euler.

The variables $\mathbf{u}_{min}, \dot{\mathbf{u}}_{min}$ and $\mathbf{u}_{max}, \dot{\mathbf{u}}_{max}$ represent lower and upper bounds of the input and input rate constraints, respectively. The vector $\mathbf{z}_{ref}$ collects the reference states of the planned path. The MPC controller implements

$$\mathbf{u}(t) = \mathbf{u}_0^*$$

at each iteration t.

A path planner creates a reference path which gives $\mathbf{z}_i$ for the entire horizon. There will be a short discussion on how our path planner creates these reference paths in the following section.

Remark: Although we presented a nominal MPC design here, a robust MPC design based on the quantified uncertainty presented in Section III could be employed and is the subject of ongoing research [12].

V. Controller Setup

The MPC controller, localization, and low level controllers are all run using a dSpace MicroAutobox with an IBM PowerPC processor running at 900 MHz. The controller uses a general purpose nonlinear solver NPSOL [13] to solve the MPC formulation.

The input and prediction horizon of the controller is 8 steps, thus looking ahead 1.6 s. The sampling time and horizon was chosen with regards to the computational complexity of the optimization problem and performance capabilities of the MicroAutobox. The values for $\mathbf{u}_{min}, \dot{\mathbf{u}}_{min}$ and $\mathbf{u}_{max}, \dot{\mathbf{u}}_{max}$ are shown in Table II.

TABLE II: Numerical values for input (rate) constraints

	β	a	$\dot{\beta}$	$\dot{a}$
minimum	-37 °	$-1.5\,\text{m/s}^2$	$-10\,°/\text{s}$	$-3\,\text{m/s}^3$
maximum	37 °	$1\,\text{m/s}^2$	$10\,°/\text{s}$	$1.5\,\text{m/s}^3$

In order to create the reference path ($\mathbf{z}_{ref,i}$), the car's current position is projected onto the reference track/centerline of the of the street, and the reference path is generated from this position to follow the track. The reference path is placed ahead of the vehicle. The distance between reference points is a function of the desired velocity.

VI. Results

Three types of experiments are performed. First, we test the low-speed performance of the controller at a closed street. Next, the tracking of a sinusoidal trajectory and finally the path following on a winding track are analyzed. The second and third experiment are performed at the CPG.

A. Low-Speed Tracking

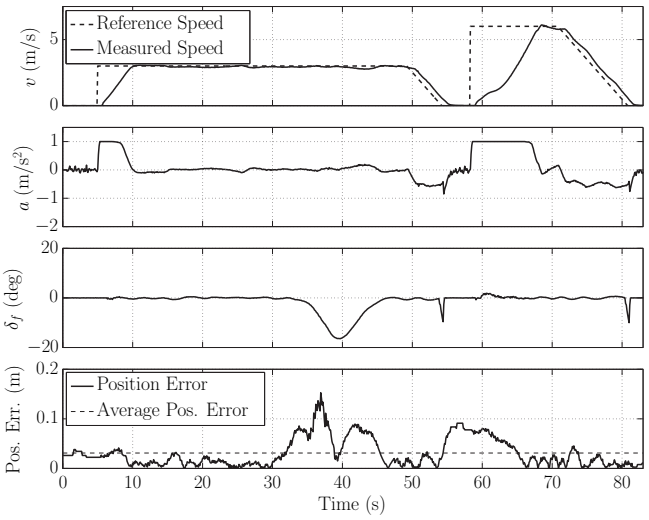

Fig. 5: Experimental results from a low-speed tracking of a right turn. The reference speed varies from 0 to 6 m/s.

We show that the proposed controller is able to operate in a stop and go scenario. We conducted an experiment where the reference trajectory is a right turn and the reference velocity ranges from 0 to 6 m/s.

The results are shown in Figure 5. The controller tracks the reference velocities well and is able to handle the stop and go situation. The mean and standard deviation of the

distance error relative from the reference path is $\mu = 0.03\,\text{m}$ and $\sigma = 0.03\,\text{m}$, respectively. A maximum error of $0.15\,\text{m}$, which occurs during the turn.

B. Sinusoidal Path Following Results

We present the tracking results using a sinusoidal reference with an amplitude of $4\,\text{m}$ and a wavelength of $100\,\text{m}$. Two experiments are presented. In the first experiment, the reference speed along the X-axis is v_x = $10\,\text{m/s}$. In the second experiment, the reference speed along the X-axis is v_x = $15\,\text{m/s}$. The reference speed along the longitudinal axis of the vehicle is

$$v_{ref} = v_x \cdot \sqrt{1 + (A \cdot \omega)^2 \cdot \cos^2(\omega x)} \tag{5}$$

where $\omega = \frac{2\pi}{D}$ and x is the vehicle's position on the X-axis. D is the wavelength and A is the amplitude of the sine, both in meters.

The origin of the inertial coordinate frame is set at the vehicle's starting position. A reference trajectory is created at each time step and is based on the current position of the car relative to the origin. This reference trajectory generates the $\mathbf{z}_{ref}$ according to the velocity reference and the location of the car on the sinusoidal path.

In our experiments, the vehicle starts at a standstill. After a few cycles, it reaches the desired velocity and evolves with periodic dynamics. In Figure 6, the steady state results at both speeds are reported. This figure also highlights the difference between the MPC open-loop predictions and the closed-loop behavior. A finite horizon, model mismatch, sensor noise and external disturbances can affect the mismatch.

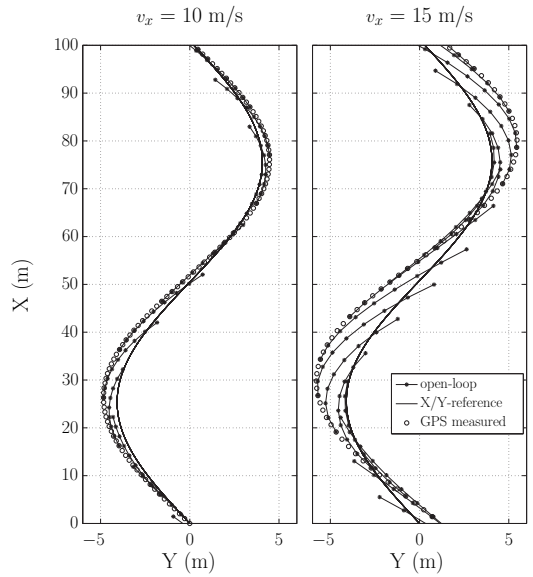

Fig. 6: Steady state test results for following a sinusoidal trajectory $Y = 4 \cdot \sin(2\pi/100 \cdot X)$. On the left, the car follows a speed reference of v_x = 10 m/s. On the right, the car follows a speed reference of v_x = 15 m/s. For clarity, open loop trajectories are only shown every 5 time steps.

In Figure 6, one can observe that the deviation from the reference path is larger for higher speeds. This is due to the higher lateral forces and therefore higher side slip angle of the tires which then negatively affects the accuracy of model forecast. Figure 6 also emphasizes one of the drawbacks of using a kinematic bicycle model in an MPC controller. Even a well-tuned controller may have trouble following a trajectory with high lateral acceleration because the model mismatch becomes more pronounced. This is clearly displayed in the open-loop trajectories predicted by the controller in the $v_x = 15\,\text{m/s}$ case, where the open-loop trajectories no longer match the closed-loop path of the car. Since the vehicle is understeering, the turns driven by the car are always wider than assumed by the controller.

In the experiment where $v_x = 10\,\text{m/s}$, the average error and standard deviation from the reference trajectory is $0.41\,\text{m}$ and $0.25\,\text{m}$, respectively. When $v_x = 15\,\text{m/s}$, the average error and standard deviation from the reference trajectory is $1.08\,\text{m}$ and $0.68\,\text{m}$, respectively.

C. Winding Track Results

The winding track was used as a testbed for the experiments in this section. A map of the winding track was created along with a speed reference at different points along the map. The speed reference on the map, or a speed map, was obtained by recording the driving data of a human driver on this track. The first experiment follows the human driven speed reference. In a subsequent experiment, this speed reference is scaled by a factor of 0.7. Table III provides information on the speed range used for the autonomous drive for both tests.

TABLE III: Velocity information about the winding track

Scaling	Avg. Speed	Max. Speed	Min. Speed	Std. Deviation
0.7	10.8 m/s	15.56 m/s	5 m/s	2.14 m/s
1	15.4 m/s	22.23 m/s	7.2 m/s	3 m/s

Our results show good performance of the proposed MPC controller to track the reference path. In Figure 7, the distributions of reference trajectory errors are shown for different scaled speeds. The average centerline error at a scaled speed of 0.7 and 1 is 0.26 m and 0.58 m, respectively.

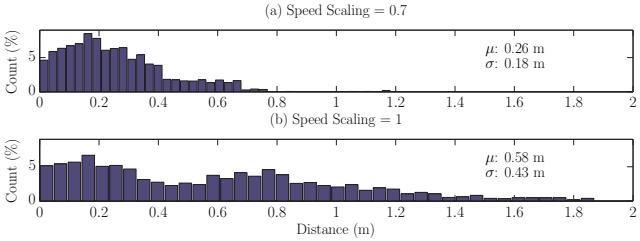

Fig. 7: Distribution of tracking errors relative to the reference trajectory for two reference speeds.

Similar to the sinusoidal reference, Figure 8 displays the open-loop trajectories versus the closed-loop measured GPS positions on the winding track. On the straight line path, the kinematic model accurately models the predicted paths at both scaling factors. However, as is seen in Figure 8, the reference tracking at higher speeds is not as good due to the larger lateral acceleration, which causes more slip. When the lateral acceleration grows larger, it becomes more critical to use a model which takes into consideration slip, such as a dynamic bicycle model.

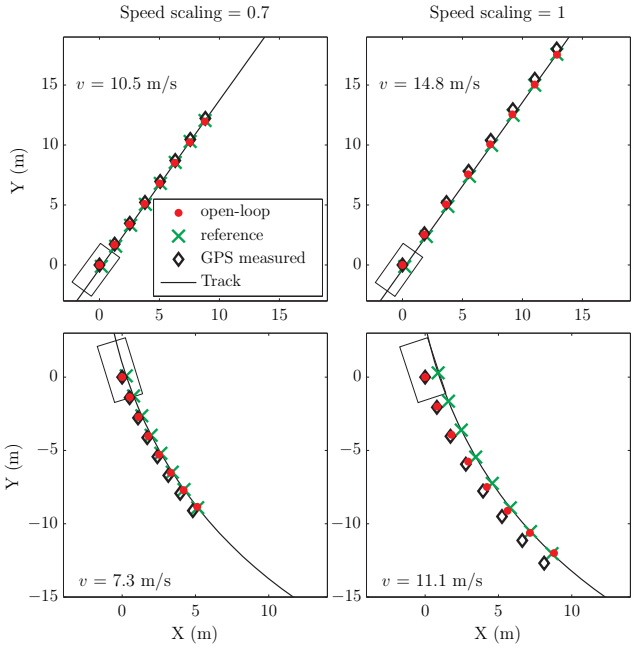

Fig. 8: The open loop error for four different scenarios. In the left column, test results for a straight road and a curvy road are shown for an experiment at a scaled speed of 0.7. On the right side are the same positions on the track at a scaled speed of 1.

VII. Conclusion

Comparisons between the kinematic bicycle model and dynamic bicycle model were done in order to motivate an MPC controller design. Our study shows that the kinematic model discretized at 200 ms is able to perform similarly well to a dynamic model discretized at 100 ms. In addition, our analysis shows that the kinematic model has better forecast errors when discretized at 200 ms compared to 100 ms.

These comparison results motivate a design of an autonomous vehicle controller by using model predictive control (MPC) framework and a kinematic bicycle model. Existing MPC approaches for vehicle control use higher fidelity models that include the tire and road interaction. The proposed approach requires less computational power and can be implemented at a wide range of vehicle speeds including zero speed.

A variety of experimental results show the effectiveness of the proposed approach at various speeds. We showed that our controller is able to control the vehicle in a stop-and-go scenario. For the winding track and sinusoidal experimental tests, the proposed controller is able to track varied reference trajectories well at lower speeds. On the higher speed sinusoidal and winding track tests, the reference errors grow significantly and a tire model based controller would be more appropriate.

The predictive capability of the kinematic bicycle model at different lateral accelerations can be further studied together with a with an optimization of the discretization time. To increase performance, the path planner can also limit the reference velocity such that the lateral acceleration cannot surpass a certain value.

Acknowledgments

The authors would like to thank Ashwin Carvalho and Tammo Zobel for their guidance and help with the vehicle.

References

[1] (1925, August) Science: Radio auto. [Online]. Available: http://content.time.com/time/magazine/article/0,9171,720720,00.html

[2] M. Walton. (2004, May) Robots fail to complete grand challenge. [Online]. Available: http://www.cnn.com/2004/TECH/ptech/03/14/darpa.race/index.html

[3] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y.-W. Seo, S. Singh, J. Snider, A. Stentz, W. . Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson, "Autonomous driving in urban environments: Boss and the urban challenge," Journal of Field Robotics, vol. 25, no. 8, pp. 425–466, 2008. [Online]. Available: http://dx.doi.org/10.1002/rob.20255

[4] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, et al., "Stanley: The robot that won the darpa grand challenge," Journal of field Robotics, vol. 23, no. 9, pp. 661–692, 2006.

[5] F. Borrelli, A. Bemporad, and M. Morari, "Predictive control for linear and hybrid systems," 2014, http://www.mpc.berkeley.edu/mpc-course-material.

[6] P. Falcone, "Nonlinear model predictive control for autonomous vehicles," Ph.D. dissertation, University of Sannio, Benevento, June 2007.

[7] A. Carvalho, Y. Gao, A. Gray, H. Tseng, and F. Borrelli, "Predictive control of an autonomous ground vehicle using an iterative linearization approach," in Intelligent Transportation Systems - 2013, Oct 2013, pp. 2335–2340.

[8] Y. Gao, T. Lin, F. Borrelli, E. Tseng, and D. Hrovat, "Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads," in ASME 2010 Dynamic Systems and Control Conference. American Society of Mechanical Engineers, 2010, pp. 265–272.

[9] C. Beal and J. Gerdes, "Model predictive control for vehicle stabilization at the limits of handling," Control Systems Technology, IEEE Transactions on, vol. 21, no. 4, pp. 1258–1269, 2013.

[10] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Kolter, D. Langer, O. Pink, V. Pratt, et al., "Towards fully autonomous driving: Systems and algorithms," in Intelligent Vehicles Symposium (IV), 2011 IEEE. IEEE, 2011, pp. 163–168.

[11] R. Rajamani, Vehicle Dynamics and Control, ser. Mechanical Engineering Series. Springer, 2011.

[12] A. Carvalho, Y. Gao, S. Lefèvre, and F. Borrelli, "Stochastic predictive control of autonomous vehicles in uncertain environments," in 12th International Symposium on Advanced Vehicle Control (AVEC), 2014.

[13] P. Gill, W. Murray, M. Saunders, and M. Wright, "User's guide for NPSOL (version 4.0)," Systems Optimization Laboratory, Stanford University, Stanford, CA, Tech. Rep. SOL-86-2.