2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

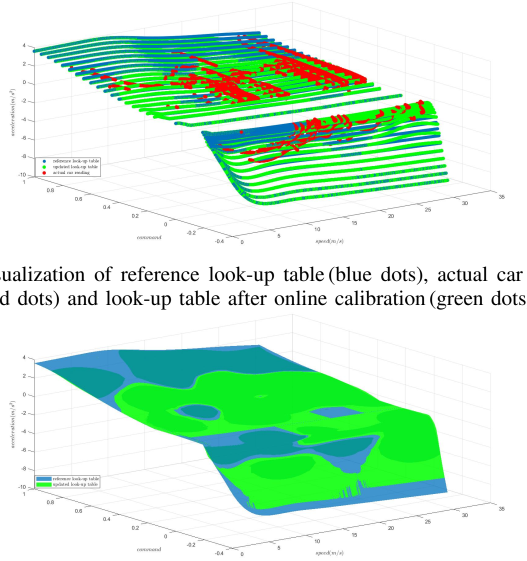
Efficient Online Calibration for Autonomous Vehicle's Longitudinal Dynamical System: A Gaussian Model Approach

Shihao Wang*, Canqiang Deng†, and Qingjie Qi†

Abstract— **In this paper, we present an efficient online calibration system for longitudinal vehicle dynamics of driverless cars. Instead of modeling vehicle's longitudinal dynamical system analytically, we employ a data-driven method to generate an "end-to-end" numerical model with a look-up table which saves vehicle's velocity, control command, and acceleration. This reference table should be calibrated to account for variations of vehicle's hardware status over time. To reduce the expensive labor in calibration process, we propose an effective algorithm to update this reference look-up table with a Gaussian model approach. We introduce a 2-D Gaussian distribution to model acceleration error between interpolated one from look-up table and actual one from vehicle sensors. We estimate model's standard deviations with a "three-sigma rule" heuristic and calculate its height with a backtracking method such that monotonicity constraint between acceleration and control command is strictly satisfied in the updated table. The effectiveness of our proposed system is verified in real-world road tests with Lincoln MKZ.**

I. INTRODUCTION

Autonomous vehicles have received ever-increasing attention in recent years because of their potentials to decrease traffic congestion and fatal accidents, improve productivity and efficiency of the driving time, and serve as transportation tools accessible to anyone at anytime [1]. These expectations correspond to level 4 or higher driving automation from Society of Automotive Engineers (SAE) and many companies and research institutes have been studying and developing strategies to elevate levels of autonomy of existing vehicles [2]. A fully autonomous vehicle should typically be equipped with two main sub-systems: (1). perception system for state estimation, obstacle detection and traffic-sign recognition (2). decision-making system for motion planning, path selection and vehicle control in lateral and longitudinal directions [3], [4]. These intelligent systems enable self-driving cars to navigate themselves in complex environments with static and moving obstacles, and reason in a human-level reliability to deal with unpredictable traffic situations. Though promising results have been demonstrated in recent works [5], [6], [7], [8], the ability to generalize to complex traffic scenarios with many neighboring vehicles are yet to be achieved. Among all the challenges remained to be solved, this paper contributes an effective algorithm to vehicle's decision-making system by increasing the low-level control accuracy of vehicle's longitudinal dynamical system (Fig. 1).



(a) Visualization of reference look-up table (blue dots), actual car readings (red dots) and look-up table after online calibration (green dots).

(b) Visualization of reference look-up table (blue surface) and updated look-up table (green surface).

Fig. 1: A representative comparison between reference look-up table and updated look-up table after calibration. Here, x, y axes are speed and command, respectively and z axis is acceleration. Given vehicle's actual readings in red dots, our algorithm adjusts table's "shape" to reduce acceleration discrepancies, thus increasing vehicle's longitudinal control accuracy.

Given a standard hierarchical planning and control framework, a reference state trajectory $x_{\text{ref}}(t)$ is calculated after planning for route, path, and vehicle motion. This reference denotes vehicle's desired state in time horizon and stabilizing controller will be synthesized for trajectory tracking. Existing approaches have used feedback linearization [9], model predictive control [10], [11], nonlinear control [12], feedback-feedforward control [13] to regulate vehicle's motion to its reference path. However, the gap between the kinematic or dynamic model used in these methods and vehicle's actual dynamical system makes it difficult to accurately convert high-level control signal into low-level control command (throttle/brake). Furthermore, the fact that conditions in both vehicle and road vary over time requires frequent system identification and calibration to be conducted where each task takes considerable manual labour and time [14], [15]. To facilitate the conversion from high-level control policy to low-level throttle/brake command and maintain an efficient lifelong model calibration without valuable human labor, we first employ an offline data-driven approach to implicitly model vehicle's longitudinal dynamical system with a

*: Shihao Wang is with the Department of Mechanical Engineering and Materials Science, Duke University, Durham, NC 27708, USA shihao.wang@duke.edu

†: Cangqiang Deng, and Qingjie Qi are with Deeproute.ai Ltd, Fremont, CA, 94538, USA {canqiangdeng, qingjieqi, tongyicao}@deeproute.ai

978-1-7281-9077-8/21/$31.00 ©2021 IEEE

look-up table. This table connects vehicle's velocity, control command to its corresponding acceleration and enables a bijective mapping between control command and acceleration under fixed speed. Then an online calibration algorithm is proposed to update this reference table according to vehicle's actual information at runtime. We model as a 2-D Gaussian model acceleration error between interpolated one from look-up table and actual one from vehicle's on-board sensors. Model's standard deviations are estimated with a "three-sigma rule" and its height is calculated with a backtracking method such that monotonicity constraint between control command and acceleration remains satisfied in the updated look-up table. By incorporating this calibration algorithm on-the-fly, autonomous vehicle is able to adjust its reference look-up table using its actual dynamical information and experiments in real-world road-tests demonstrate that our proposed online calibration module effectively decreases vehicle's longitudinal position error by 40% after few laps on test road.

II. Related Work

Our proposed system focuses vehicle system identification and calibration. Many related strategies have been proposed to address these topics.

Vehicle model plays an essential role in decision-making system of self-driving cars and model accuracy dramatically influences the performance of vehicle's control system. While kinematic model is generally adopted for vehicle in relatively low-speed scenarios, dynamic model should be considered to describe vehicle motions at high-speed [16], [17]. Depending on the principles used to identify vehicle's dynamical system, existing approaches can be categorized into two classes : *physics-based modeling* and *data-driven-based modeling*. The first class of modeling methods embraces classical mechanics as their fundamental rules and derives explicit equations of motion of vehicle's translational and rotational dynamics with Newton–Euler equations [18], [19], [20]. This type of models captures vehicle's dynamical behaviors with deterministic parameters and provides convenience to model-based controller design and vehicle simulation due to their closed-form expressions. Even though these modeling techniques have obvious advantages resulting from their simplicity and conciseness, they suffer from the difficulty in determining values of physical parameters. A full dynamics model requires a number of parameters to be predetermined to take into consideration of tire-road contact, vehicle powertrain, aerodynamics, engine dynamics, etc. However, some of them are not directly measurable and the estimation of their values needs introducing additional nonlinear regression models, which makes the symbolic dynamics expression no longer simple [21], [22], [23]. To circumvent the above limitations, researchers have proposed data-driven-based approaches for system modeling and identification of automated vehicles [24]. Strategies such as extended Kalman-filter method [25], prediction error method [26], predictor-based subspace identification [27], feed-forward neural networks [28], and deep-learning based on Koopman operator method [29] have demonstrated their powerful abilities in approximating highly nonlinear system dynamics. However, their poor interpretability, difficulty in generalization and unknown model sensitivity make them challenging to be employed as industry-level approaches. To address these issues, we propose a specialized data-driven method to generate an "end-to-end" vehicle model with an interpretable look-up table. This table saves vehicle's velocity, control command, and acceleration and can be visualized as a set of 3-D points which can be straightforwardly explained.

The non-deterministic characteristics of vehicle's parameters suggest model calibration to be conducted for error correction. To decrease the expensive human labor in vehicle calibration process, online model adjustment techniques have been proposed to address several relevant aspects. Tafner presents a robust parameter estimation method for a simplified roll dynamics with sliding mode concepts [30]. Seegmiller proposes an automatic calibration method to model the perturbative vehicle behaviors on arbitrary terrains [31]. The most relevant work to our online calibration method is Baidu Apollo's auto-calibration system where an offline look-up table is initially generated with a three-layer feedforward neural network and then this whole reference table is adjusted with a customized damping approach [32]. Though sharing a similar system structure, our technique has the following advantages:

- **calibration region**: Instead of a "global" calibration, our method divides the reference look-up table into two subregions to separate accelerations effected by throttle and brake, and adaptively calibrates a "local" area of the subregion based on a customized percentage parameter. This transition from "global" to "local" significantly reduces the computational time and makes our approach sufficient for real-time implementation.
- **updated rule**: Apollo introduces a customized similarity evaluation function to calculate the updated value of every point in look-up table where a number of parameters should be carefully tuned. However, our method adopts a 2-D Gaussian model to describe error distribution of acceleration with only few parameters to be determined.
- **constraint satisfaction**: Our method adopts a backtracking strategy to iteratively search feasible solutions for constraint satisfaction while Apollo's approach lacks the ability to deal with constraint violation.

III. Method

A. *Offline Look-up Table Generation*

We adopt a data-driven-based approach to build an "end-to-end" numerical model of vehicle's longitudinal dynamical system. This numerical model is constructed into a 3-D look-up table whose dimensionalities are vehicle's control command, speed, and acceleration. The reason behind this choice is twofold: (1). Among all the variables that influence vehicle's acceleration, control command and velocity are no doubt the dominant ones [18]. (2). By limiting the number of features to be 2, a mapping function $f(c, v) \rightarrow a$ will be constructed where c, v and a denote vehicle's longitudinal control command, speed, and acceleration, respectively. Enumerations along c and v generate a 3-D surface whose shape is intuitive and interpretable (Please refer to Fig. 1b).

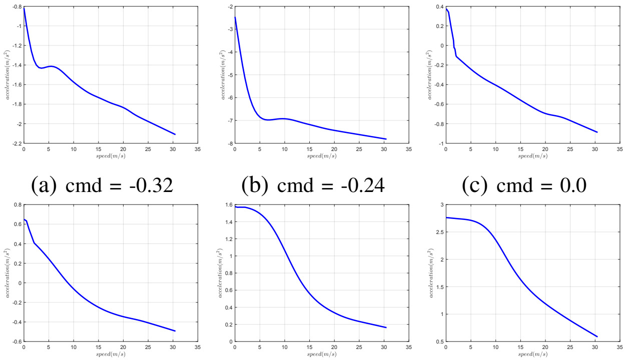
Procedures for look-up table construction are as follows:

1) Feasible range estimation of control command and speed
   - Command range is estimated with two steps: Firstly, vehicle's control command range is normalized to be $c \in (-1, 1)$ where $c > 0$ and $c < 0$ stand for throttle and brake, respectively. Secondly, throttle and brake's deadzones are measured by human engineers and command's feasible range is then divided to be $c \in (-1, c_{\text{brake}}) \cup (c_{\text{throttle}}, 1)$ where vehicle is non-responsive to applied pedal if $c \in [c_{\text{brake}}, c_{\text{throttle}}]$.
   - Speed range is estimated according to vehicle's speed limit from its usual work zone $v \in [0, v_{\max}]$.
2) Data collection of vehicle's speed and acceleration

   After a uniform-grid discretization of command range to be of size N_{brake} and N_{throttle} separately, we collect data of vehicle's velocity and acceleration trajectories under constant control command:
   - For sampled $c \in (-1, c_{\text{brake}})$, we first accelerate vehicle's speed to be $v_{\max}$ and then switch control command to be c to record v and a until vehicle stops.
   - For sampled $c \in (c_{\text{throttle}}, 1)$, vehicle is initially set to be static and then accelerates with c until $v \geq v_{\max}$.
3) Acceleration correction with pitch angle

   Due to the existence of unevenness of the ground, vehicle's pitch angle θ will vary during the data collection process. This pitch angle adds ground projection of gravitational acceleration to longitudinal acceleration and its effect should be cancelled from the measured acceleration: $a = a^* - g\sin(\theta)$ where a^* is the measured acceleration and $g = 9.81m/s^2$. Illustrative $v - a$ trajectories are presented in Fig. 2.
4) Reference table generation with slicing velocities

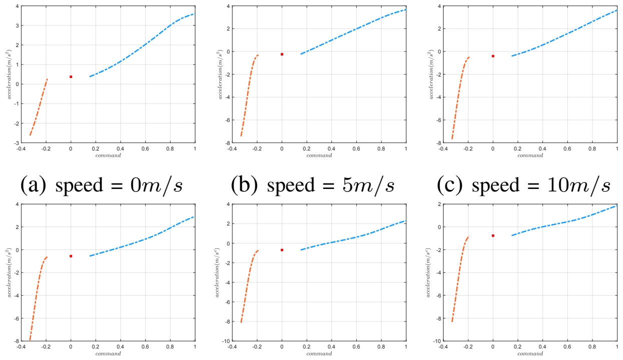
   After collecting vehicle's velocity and corrected acceleration trajectories under fixed control commands, we evenly discretize speed range to be N_{speed} points and generate reference look-up table by slicing trajectories along speed grid. Fig. 3 illustrates several velocity slices from look-up table where a strictly monotonic relationship is satisfied between control command and acceleration.



(a) cmd = -0.32 (b) cmd = -0.24 (c) cmd = 0.0

(d) cmd = 0.25 (e) cmd = 0.5 (f) cmd = 0.75

Fig. 2: 2-D slices of 3-D look-up table along **command** dimension where x axis is speed and y axis is acceleration. Negative and positive cmd denote brake and throttle, respectively and zero cmd (Fig. 2c) is the behavior of vehicle's natural dynamics.



(a) speed = $0m/s$ (b) speed = $5m/s$ (c) speed = $10m/s$

(d) speed = $15m/s$ (e) speed = $20m/s$ (f) speed = $25m/s$

Fig. 3: 2-D slices of 3-D look-up table along **speed** dimension where x and y denote control command and acceleration, respectively. Red curve is brake (cmd < 0) and blue curve (cmd > 0) is throttle. The gap between these two curves is the effect from deadzones whose acceleration is marked with red dot (cmd $= 0$).

This monotonic relationship between c and a allows a bijective mapping between low-level control command and acceleration and this property makes our look-up table approach suitable for both offline simulation $f(c, v) \rightarrow a$ and online control $f^{-1}(a, v) \rightarrow c$.

B. Online Look-up Table Calibration

The reference look-up table describes vehicle's dynamical relationships between its speed, control command and acceleration and should be calibrated to account for variations of vehicle conditions, such as changes in car engine's performance or different vehicles with similar specifications. With actual measurements available from car's on-board sensors, this reference table should be adjusted to minimize acceleration error between table's interpolated value and actual value from sensor reading. For this purpose, we introduce a 2-D Gaussian model to describe distribution of acceleration error on $c \times v$ plane and adopt a backtracking method to calculate model's height A such that acceleration error is reduced and the monotonic constraint is satisfied in the updated table. Note that we employ a "three-sigma-rule" to calculate model's standard deviations σ_c and σ_v in a local region whose size is chosen by area percentage $\gamma \in (0, 1]$.

We define a list of notations to facilitate the following discussion of the proposed online calibration algorithm:

- round to nearest integer operator $[\cdot] : \mathbb{R} \rightarrow \mathbb{Z}$
- $c(\cdot) : \mathbb{Z} \rightarrow \mathbb{R}$ and $v(\cdot) : \mathbb{Z} \rightarrow \mathbb{R}$ evaluate control command and speed at certain grid index, respectively.
- $f(\cdot, \cdot)$: look-up table struct whose members include c_{brake}, c_{throttle}, $v_{\max}$, N_{brake}, N_{throttle} and N_{speed}, and methods contain $c(\cdot)$ and $v(\cdot)$.

Algorithm 1: Online Calibration Algorithm

```
Input : Reference look-up table $f(\cdot,\cdot)$, actual
        measurement $\boldsymbol{x}^* = (c^*, v^*, a^*)$, area
        percentage $\gamma$, learning rate $\eta$
Output: Updated look-up table $f^*(\cdot,\cdot)$
1  if $c^* < 0$ then
2      $n_{\text{cmd}} \leftarrow [\gamma N_{\text{brake}}]$, $c$'s range is $(-1, c_{\text{brake}})$
3  else
4      $n_{\text{cmd}} \leftarrow [\gamma N_{\text{throttle}}]$, $c$'s range is $(c_{\text{throttle}}, 1)$
5  end
6  $n_{\text{speed}} \leftarrow [\gamma N_{\text{speed}}]$, $v$'s range is $[0, v_{\max}]$
7  $f_{\text{local}}(\cdot,\cdot)$, $\mathbb{C}$,
   $\mathbb{V} \leftarrow$ updated_region_sel($n_{\text{cmd}}, n_{\text{speed}}, c^*, v^*, f(\cdot,\cdot)$)
8  $A \leftarrow \eta \cdot (a^* - f(c^*, v^*))$
9  $\sigma_c, \sigma_v \leftarrow$ standard_deviation_cal($\mathbb{C}, \mathbb{V}, c^*, v^*, f(\cdot,\cdot)$)
10 $A^* \leftarrow$ backtracking($A, \sigma_c, \sigma_v, \mathbb{C}, \mathbb{V}, f_{\text{local}}(\cdot,\cdot)$)
11 if $A^* \neq 0$ then
12     $f^*(\cdot,\cdot) \leftarrow$ table_update($A^*, \sigma_c, \sigma_v, \mathbb{C}, \mathbb{V}, f(\cdot,\cdot)$)
13     return $f^*(\cdot,\cdot)$
14 end
15 return $f(\cdot,\cdot)$
```

Algorithm 2: Standard Deviation Calculation

```
Input : Sets of command index $\mathbb{C}$ and speed index
        $\mathbb{V}$, actual command $c^*$ and speed $v^*$, and
        reference look-up table $f(\cdot,\cdot)$
Output: Standard deviations $\sigma_c$ and $\sigma_v$
1  $d_c \leftarrow 0, d_v \leftarrow 0$
2  for $i \in \mathbb{C}$ and $j \in \mathbb{V}$ do
3      $c \leftarrow c(i)$, and $d_c \leftarrow |c - c^*|$ if $d_c \leq |c - c^*|$
4      $v \leftarrow v(j)$, and $d_v \leftarrow |v - v^*|$ if $d_v \leq |v - v^*|$
5  end
6  $\sigma_c \leftarrow \frac{d_c}{3}, \sigma_v \leftarrow \frac{d_v}{3}$
7  return $\sigma_c$, $\sigma_v$
```

- Within $c \times v$ region to be updated, set of control command indices is denoted as $\mathbb{C}$ and set of speed indices set is denoted as $\mathbb{V}$.

Alg. 1 illustrates the details of calibration algoirthm:

- Lines 1–6 distinguishes control command from actual measurement to be **throttle** or **brake** and computes sizes of command grid and speed grid to be updated according to area percentage parameter γ. The introduction of γ allows a local region of trust to be customized where a bigger γ updates a larger area which takes longer computational time while a smaller γ takes less elements for value adjustment but can cause table's overall geometric shape to be not smooth resulting from local "bump"s.
- Line 7 adaptively selects the region to be updated from actual measurement and grid number n_{cmd} and n_{speed}. The center of this rectangle is located at (c^*, v^*) and its width and length in index level should be n_{cmd} and n_{speed}. We denote this local table as $f_{\text{local}}(\cdot,\cdot)$ and it occupies a closed rectangular space on $c \times v$ plane. This rectangular area will not always be in the interior of feasible ranges of control command and speed, and its width or length must be truncated to match the overlapped region if it is noninclusive to feasible ranges of command and speed.
- Line 8 initializes Gaussian model's height A with a scaled acceleration difference between actual acceleration a^* and interpolated acceleration $f(c^*, v^*)$.
- Line 9 calculates standard deviations used for 2-D Gaussian model with "three-sigma-rule" heuristic [33]. Details of this subroutine are illustrated in Alg. 2.
- Line 10 conducts backtracking to calculate 2-D Gaussian model's height A^* given its initial value A and standard deviations σ_c and σ_v. We denote Gaussian function's dependence on height in superscript and express its formula to be
$$G^A(c, v) = A \exp(-(\frac{(c - c^*)^2}{2\sigma_c^2} + \frac{(v - v^*)^2}{2\sigma_v^2})) \qquad (1)$$
where $c = c(i), v = v(j), \forall i \in \mathbb{C}$ and $\forall j \in \mathbb{V}$. To assert that acceleration remains a monotonic function of control command after adding this Gaussian offset function to $f_{\text{local}}(\cdot,\cdot)$, an iterative procedure will be conducted to backtrack A's value by multiplying a scale coefficient $s \in (0, 1)$. With the increment of iterations, Gaussian model's height will decrease exponentially. Since original $f_{\text{local}}(\cdot,\cdot)$ is monotonic between control command and acceleration, our algorithm is guaranteed to produce a feasible updated table (Alg. 3).
- Line 11-14 calibrates original look-up table by adding this Gaussian distribution to the local area of $f(\cdot,\cdot)$ where **table_update**'s detail is similar to Alg. 3's Line 4-10.

Algorithm 3: Backtracking Algorithm

```
Input : Initial height $A$, standard deviations $\sigma_c$ and
        $\sigma_v$, sets of command index $\mathbb{C}$ and speed
        index $\mathbb{V}$ and local look-up table $f_{\text{local}}(\cdot,\cdot)$
Output: Calibrated model height $A^*$
1  Iter $\leftarrow 0, A^* \leftarrow A$
2  while Iter < IterMax do
3      $f_{\text{local}}^{\text{temp}}(\cdot,\cdot) \leftarrow f_{\text{local}}(\cdot,\cdot)$
4      for $i \in \mathbb{C}$ do
5          $c \leftarrow c(i)$
6          for $j \in \mathbb{V}$ do
7              $v \leftarrow v(j)$
8              $f_{\text{local}}^{\text{temp}}(c, v)$ += $G^{A^*}(c, v)$ (1)
9          end
10     end
11     if $f_{\text{local}}^{\text{temp}}(\cdot, j)$ is entry-wise monotonic $\forall j \in \mathbb{V}$ then
12         return $A^*$
13     else
14         $A^* \leftarrow s \cdot A^*$
15         Iter += 1
16     end
17 end
18 return 0
```

C. Sensor Data Processing

Our calibration algorithm (Alg. 1) takes vehicle's runtime measurement $\boldsymbol{x}^* = (c^*, v^*, a^*)$ as nominal value and adapts reference look-up table in compliance with differences between a^* and $f(c^*, v^*)$. $\boldsymbol{x}^*$ has 3 elements and their values can be easily measured from car sensors. However, these raw sensor readings are not directly applicable for real-time calibration because of sensor noise and signal's time delay. As a result, sensor data processing is applied for data refinement before using x^* for table calibration.

1) Noise Filtering: Control command c^*, speed v^* and longitudinal acceleration a^* are measured from controller area network (CAN bus), global navigation satellite system (GNSS), and inertial measurement unit (IMU), respectively. To filter the high frequency noise from these measurements, low-pass butterworth filter is implemented for signal smoothing. We denote filter's order with $\boldsymbol{n}$ and cut-off frequency with $\boldsymbol{f}$, and introduce filters for throttle, brake, speed and acceleration whose specifications are ($\boldsymbol{n}_{\text{throttle}}$, $\boldsymbol{f}_{\text{throttle}}$), ($\boldsymbol{n}_{\text{brake}}$, $\boldsymbol{f}_{\text{brake}}$) , ($\boldsymbol{n}_{\text{speed}}$, $\boldsymbol{f}_{\text{speed}}$), ($\boldsymbol{n}_{\text{acc}}$, $\boldsymbol{f}_{\text{acc}}$) and ($\boldsymbol{n}_{\text{pitch}}$, $\boldsymbol{f}_{\text{pitch}}$). One existing issue of signal processing with non-predictive filters is "phase shift" where a fixed period of time is needed to determine what filtered signal should be. A practical strategy to resolve this issue is to first store these trajectories for a certain amount of time and then conduct filtering in both forward and backward directions to result in zero-phase shift trajectories.

2) Time Shifting: We define response delay δ_{throttle} and δ_{brake} to be the time that vehicle takes to accelerate or decelerate in response to driver's control command in throttle or brake [34]. This delay causes misalignment between control command and acceleration because acceleration trajectory should be shifted forward in time to match its corresponding control command. To calculate this response delay, a cross-correlation problem is formulated. Take throttle for example:

$$\begin{gathered} \max_{\delta_{\text{throttle}}} ||a(t + \delta_{\text{throttle}}) \cdot c_{\text{throttle}}(t)||_2^2 \\ 0 \leq \delta_{\text{throttle}} \leq \Delta \end{gathered} \tag{2}$$

where $a(t)$ and $c_{\text{throttle}}(t)$ are prerecorded trajectories and Δ is an upper bound of time delay. Since response delay is vehicle's inherent characteristic and can be considered to be time-invariant, this allows us to pre-compute it offline and then use it for online time shifting. By discretizing δ_{throttle}'s range into a number of points, a brute force method is used to find the argument of the maximum. Visualizations of time shifting in acceleration trajectories for throttle and brake are illustrated in Fig. 4.

TABLE I: Experimental Parameters

Algorithm Parameters				Filter Coeffs			
c_{throttle}	0.15	N_{throttle}	18	$\boldsymbol{n}_{\text{throttle}}$	2	$\boldsymbol{f}_{\text{throttle}}$	10 Hz
c_{brake}	-0.19	N_{brake}	15	$\boldsymbol{n}_{\text{brake}}$	3	$\boldsymbol{f}_{\text{brake}}$	10 Hz
v_{max}	30.5 m/s	N_{speed}	306	$\boldsymbol{n}_{\text{speed}}$	2	$\boldsymbol{f}_{\text{speed}}$	25 Hz
γ	0.5	η	0.001	$\boldsymbol{n}_{\text{acc}}$	2	$\boldsymbol{f}_{\text{acc}}$	25 Hz
δ_{throttle}	0.35 s	δ_{brake}	0.15 s	$\boldsymbol{n}_{\text{pitch}}$	2	$\boldsymbol{f}_{\text{pitch}}$	10 Hz
IterMax	25	s	0.1				

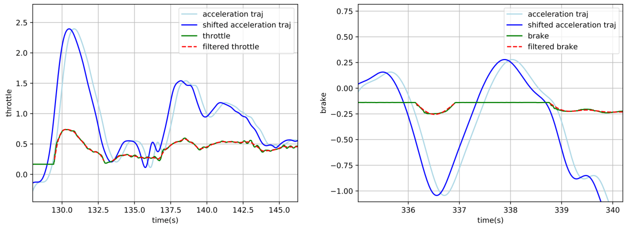
IV. Experimental Evaluation

We evaluate the effectiveness and robustness of the proposed method with two main sets of experiments. The first set of experimentation shows that our method effectively increases longitudinal control accuracy with real-world road tests and the second sets of experiment demonstrates that our calibration algorithm robustly reduces acceleration discrepancies even though intentionally incorrect reference tables are given. We collect vehicle data with Lincoln MKZ vehicle and implement our approach on a 64-bit Intel 12-Core i7 2.60GHz laptop with 16GB RAM. Experimental parameters are listed in Tab. I and online calibration procedure takes less than $0.5\,ms$.

A. Longitudinal Control Accuracy Evaluation

This subsection demonstrates the capability of our proposed method to effectively improve vehicle's longitudinal control accuracy. Given road test path as shown in Fig. 5 and a reference look-up table whose calibration was conducted several months ago, we evaluate the improvement of control accuracy with analyzing vehicle's longitudinal position error. This position error $e_p(t) = p^*(t) - p(t)$ is measured from a comparison between vehicle's planned position $p^*(t)$ with its current position $p(t)$.

Firstly, original position error trajectory is collected along the whole path with the given reference look-up table. Then, we activate the online calibration module and save position error trajectories for a total of 5 laps around this path.



(a) (b)

Fig. 4: Plots of response delay in throttle (a) and brake (b) where acceleration trajectories are shifted forward to align with command trajectories. It is obvious that this time adjustment improves cross-correlation between acceleration and command trajectories.

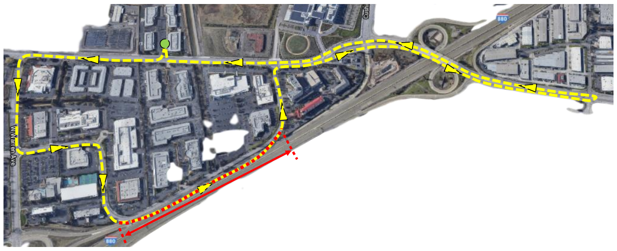


Fig. 5: Representative road-test path which includes traffic lights, stop signs, sharp turns, U-turn, unprotected turn, and highway. Red line denotes the road where position error comparison is conducted. Picture source: Map data ©2020 Google

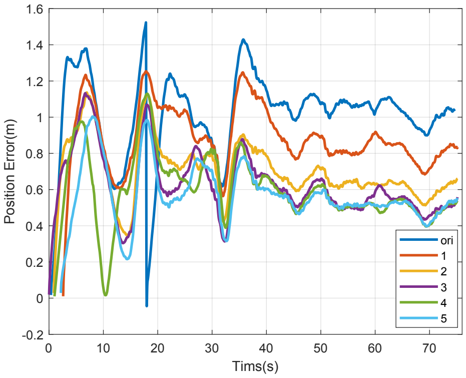


Fig. 6: Trajectories of longitudinal position errors from road tests where their origins are shifted manually to ensure the alignment of geometric shapes.

We compare position error along a segment (red line in Fig. 5 whose traffic situations remain steady during these 6 continuous tests. Fig. 6 illustrates trajectories of longitudinal position errors from our road tests and its performance analysis in terms of average position error (APE) and error percentage decrease (EPD) is shown in Tab. II.

TABLE II: Longitudinal Position Error Results

	original	1	2	3	4	5
APE(m)	1.013	0.891	0.703	0.624	0.598	0.581
EPD(%)		12.0	30.6	38.4	40.9	42.6

It is obvious that vehicle's longitudinal error decreases consistently until it reaches a lower bound with our online calibration module and a more than 40% accuracy improvement is achieved after 5 laps' calibration.

B. Reference Table Acceleration Error Evaluation

In addition to online adjustment, our method can also be adopted in an "offline" fashion. Given vehicle's data collected from actual road tests, our proposed algorithm robustly minimizes acceleration errors even though intentionally incorrect reference tables are given

Given 10 road test data and 3 reference look-up tables, we compare results of without/with calibration algorithm. Each road test data is collected along paths similar to Fig. 5. Among these 3 tables, Table 1 is a standard reference look-up table and we then intentionally add $+/-1.0\,m/s^2$ to Table 1's acceleration dimension to generated offsetted Table 2 and Table 3.

TABLE III: Reference Table Acceleration Error Results

Case	Table1	Table1*	Table2	Table2*	Table3	Table3*
1	0.112	0.086	0.933	0.610	1.085	0.608
2	0.127	0.084	0.925	0.495	1.119	0.807
3	0.107	0.091	0.932	0.477	1.063	0.698
4	0.114	0.093	0.960	0.611	1.056	0.586
5	0.133	0.080	0.922	0.546	1.096	0.643
6	0.123	0.112	0.911	0.701	1.095	0.698
7	0.117	0.087	0.953	0.610	1.081	0.564
8	0.120	0.075	0.945	0.572	1.052	0.869
9	0.112	0.098	0.968	0.641	1.025	0.736
10	0.144	0.132	0.966	0.584	1.000	0.635
Avg	0.119	0.093	0.942	0.585	1.067	0.684
EPD(%)		21.8		37.9		35.9

For each road-test data, we randomly allocate 75% for calibration and 25% for test, and quantitative results using mean absolute error (MAE) are illustrated in Tab. III where the superscript $*$ denotes the calibrated table and value's unit is m/s^2. The result shows that for all 10 test cases, our method is capable of minimizing acceleration errors given different reference tables and these 2 intentionally offsetted reference table are robustly adjusted with $> 35\%$ improvement in matching actual accelerations.

V. CONCLUSION

This paper presents an efficient online calibration system for longitudinal vehicle dynamics of autonomous vehicles. We employ a data-driven approach to generate an "end-to-end" numerical vehicle model with a look-up table which saves vehicle's velocity, control command, and acceleration. This reference table helps connect high-level control objective to vehicle's low-level throttle or brake command and should be calibrated to account for variations of vehicle's hardware status over time. For the sake of reducing expensive labor in manual calibration process, we propose an effective online calibration algorithm to update this reference look-up table with a Gaussian model approach. We introduce a 2-D Gaussian distribution to model acceleration errors between interpolated one from look-up table and actual vehicle acceleration. This model's standard deviations are estimated with a "three-sigma rule" heuristic and its height is calculated with a backtracking method to guarantee monotonicity constraint between acceleration and control command. We demonstrate the effectiveness and robustness of our proposed method with two sets of experiments where sufficient performance improvement is achieved using vehicle data collected from Lincoln MKZ.

During the implementation of the proposed algorithm for real-world road tests, we have observed that response delay plays a significant role in calibrating vehicle's acceleration to its corresponding control command. Currently, we employ an offline approach to compute vehicle's delay time (described in Sec. III-C.2). We plan in the near future to implement an online algorithm to calculate these delays from throttle and brake.

References

[1] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and decision-making for autonomous vehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 187–210, 2018.

[2] SAE(2018), "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," *Technical Report SAE International*, 2018.

[3] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[4] C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, L. Jesus, R. Berriel, T. Paixão, F. Mutz, L. Veronese, T. Oliveira-Santos, and A. F. D. Souza, "Self-driving cars: A survey," 2019.

[5] P. Furgale, U. Schwesinger, M. Rufli, and et al., "Toward automated driving in cities using close-to-market sensors: An overview of the v-charge project," in *2013 IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 809–816.

[6] O. S. Tas, F. Kuhnt, J. M. Zöllner, and C. Stiller, "Functional system architectures towards fully automated driving," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, 2016, pp. 304–309.

[7] S. Ulbrich, A. Reschka, J. Rieken, S. Ernst, G. Bagschik, F. Dierkes, M. Nolte, and M. Maurer, "Towards a functional system architecture for automated vehicles," 2017.

[8] L. E. R. Fernandes, V. Custodio, G. V. Alves, and M. Fisher, "A rational agent controlling an autonomous vehicle: Implementation and formal verification." in *FVAV@iFM*, ser. EPTCS, L. Bulwahn, M. Kamali, and S. Linker, Eds., vol. 257, 2017, pp. 35–42.

[9] A. De Luca, G. Oriolo, and C. Samson, *Feedback control of a nonholonomic car-like robot*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 171–253.

[10] V. L. Bageshwar, W. L. Garrard, and R. Rajamani, "Model predictive control of transitional maneuvers for adaptive cruise control vehicles," *IEEE Transactions on Vehicular Technology*, vol. 53, no. 5, pp. 1573–1585, 2004.

[11] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 3, pp. 566–580, 2007.

[12] G. M. Hoffmann, C. J. Tomlin, M. Montemerlo, and S. Thrun, "Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing," in *2007 American Control Conference*, 2007, pp. 2296–2301.

[13] N. R. Kapania and J. C. Gerdes, "Design of a feedback-feedforward steering controller for accurate path tracking and stability at the limits of handling," *Vehicle System Dynamics*, vol. 53, no. 12, pp. 1687–1704, 2015.

[14] N. Seegmiller, F. Rogers-Marcovitz, G. Miller, and A. Kelly, "Vehicle model identification by integrated prediction error minimization," *The International Journal of Robotics Research*, vol. 32, no. 8, pp. 912–931, 2013.

[15] J. E. A. Dias, G. A. S. Pereira, and R. M. Palhares, "Longitudinal model identification and velocity control of an autonomous car," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 776–786, 2015.

[16] J. Yi, H. Wang, J. Zhang, D. Song, S. Jayasuriya, and J. Liu, "Kinematic modeling and analysis of skid-steered mobile robots with applications to low-cost inertial-measurement-unit-based motion estimation," *IEEE Transactions on Robotics*, vol. 25, no. 5, pp. 1087–1097, 2009.

[17] S. Yang, Y. Lu, and S. Li, "An overview on vehicle dynamics," *International Journal of Dynamics and Control*, vol. 1, p. 385–395, 2013.

[18] R. Rajamani, *Vehicle Dynamics and Control*, 2006.

[19] R. N. Jazar, *Vehicle Dynamics: Theory and Application*. Boston, MA: Springer, 2009.

[20] H. B. Pacejka, *Tire and Vehicle Dynamics*. Oxford: Butterworth-Heinemann, 2012.

[21] L. R. Ray, "Nonlinear state and tire force estimation for advanced vehicle control," *IEEE Transactions on Control Systems Technology*, vol. 3, no. 1, pp. 117–124, 1995.

[22] Jin-Oh Hahn, R. Rajamani, and L. Alexander, "Gps-based real-time identification of tire-road friction coefficient," *IEEE Transactions on Control Systems Technology*, vol. 10, no. 3, pp. 331–343, 2002.

[23] M. C. Best, "Identifying tyre models directly from vehicle test data using an extended kalman filter," *Vehicle System Dynamics*, vol. 48, no. 2, pp. 171–187, 2010.

[24] D. Solomatine, L. See, and R. Abrahart, *Data-Driven Modelling: Concepts, Approaches and Experiences*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 17–30.

[25] C. B. Low and D. Wang, "Integrated estimation for wheeled mobile robot posture, velocities, and wheel skidding perturbations," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 2007, pp. 2355–2360.

[26] K. B. Arikan, Y. S. Unlusoy, I. Korkmaz, and O. Celebi, "Identification of linear handling models for road vehicles," *Vehicle System Dynamics - VEH SYST DYN*, vol. 46, pp. 621–645, 07 2008.

[27] X. Guan, T. Ba, and J. Zhang, "Vehicle handling dynamics modelling by a data-driven identification method," *Vehicle System Dynamics*, vol. 53, no. 11, pp. 1580–1598, 2015.

[28] N. Spielberg, M. Brown, N. Kapania, J. Kegelman, and J. Gerdes, "Neural network vehicle models for high-performance automated driving," *Science Robotics*, vol. 4, 03 2019.

[29] Y. Xiao, X. Zhang, X. Xu, X. Liu, and J. Liu, "A deep learning framework based on koopman operator for data-driven modeling of vehicle dynamics," 2020.

[30] R. Tafner, M. Reichhartinger, and M. Horn, "Robust online roll dynamics identification of a vehicle using sliding mode concepts," *Control Engineering Practice*, vol. 29, pp. 235 – 246, 2014.

[31] N. Seegmiller, F. Rogers-Marcovitz, G. Miller, and A. Kelly, *A Unified Perturbative Dynamics Approach to Online Vehicle Model Identification*, 01 2017, vol. 100, pp. 585–601.

[32] F. Zhu, L. Ma, X. Xu, D. Guo, X. Cui, and Q. Kong, "Baidu apollo auto-calibration system - an industry-level data-driven and learning based vehicle longitude dynamic calibrating algorithm," 2018.

[33] F. Pukelsheim, "The three sigma rule," *The American Statistician*, vol. 48, no. 2, pp. 88–91, 1994.

[34] T. Aono and T. Kowatari, "Throttle-control algorithm for improving engine response based on air-intake model and throttle-response model," *IEEE Transactions on Industrial Electronics*, vol. 53, no. 3, pp. 915–921, 2006.