IEEE Open Journal of
Intelligent Transportation
Systems

Received 29 January 2020; revised 18 May 2020; accepted 19 July 2020. Date of publication 22 July 2020; date of current version 12 August 2020.
The review of this article was arranged by Editor P. Khayyer.

Digital Object Identifier 10.1109/OJITS.2020.3011193

Measurement Correction for Electric Vehicles Based on Compressed Sensing

AHMED AYADI[1] AND JAKOB PFEIFFER[1,2]

[1]Department of Electrical and Computer Engineering, Technical University of Munich, 80333 Munich, Germany

[2]Function and Software Development E-Powertrain, BMW Group, 80788 Munich, Germany

CORRESPONDING AUTHOR: J. PFEIFFER (e-mail: jakob.j.pfeiffer@bmw.de)

This work was supported by the BMW Group.

ABSTRACT Deviations between system current measurements and real values in the power train of Electric Vehicles (EVs) can cause severe problems. Among others, these are restricted performance and cruising range. In this work, we propose a fleet-based framework to correct such deviations. We assume that the real value is the mean of all identically constructed EVs' measurements for the same input. Under this assumption, we decide for each vehicle whether it displays hardware errors with the help of a binary classifier. Depending on the classification, if no hardware errors are detected, we recover the parameters of an assumed measurement error model via Linear Regression. Otherwise, we combine the regression with a convex optimization problem and sparsity constraints. We achieve an overall recovery rate of up to 90%, allowing the full automation of the measurement correction procedure with no need to add more sensors, or computational units on-board of the EV.

INDEX TERMS Measurement correction, electric vehicles, machine learning, compressed sensing, sparsity constraints, intelligent sensors.

I. INTRODUCTION

MEASUREMENTS of High Voltage (HV) currents play a crucial role in the power train of EVs. For example, when the HV system operates close to its physical limits, measurements serve as base for the power limitation. Harming the limit would result in restricted battery life time and threaten safe vehicle operation [1]. The EV's safety is usually ensured with the help of battery protection offsets. The magnitude of the offsets depends on the measurement accuracy to make sure that it is sufficient even under the worst measurement conditions (compare Figure 1). Thus, inaccurate measurements lead to high offsets. High offsets are problematic for several reasons. One reason is that during acceleration the offset leads to a restriction of the power even if the measurement is more accurate than the worst expected and the power-train indeed would be able to provide the requested power. In this case, the EV's performance is restricted unnecessarily. An even worse problem occurs in the opposite situation. During recuperation a too conservatively chosen offset leads to a restriction of the amount of power which is charged into the High Voltage Battery (HVB) although the HVB would be capable to store it. This decreases the EV's cruising range.

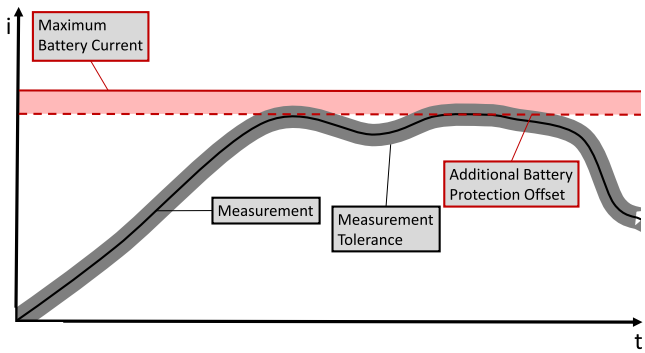

FIGURE 1. A schematic representation of power limitation due to battery protection during the acceleration of an EV. The measurement (black) might differ from the real value by some measurement tolerance (grey). To guarantee that the real value never exceeds the maximum battery current (red, solid), an additional offset (red, dashed) serves as prevention. The same principle is used analogously with negative currents during recuperation [2].

Kirchhoff's current law states that the sum of all currents at an electric node is equal to 0 A. As our HV system consists

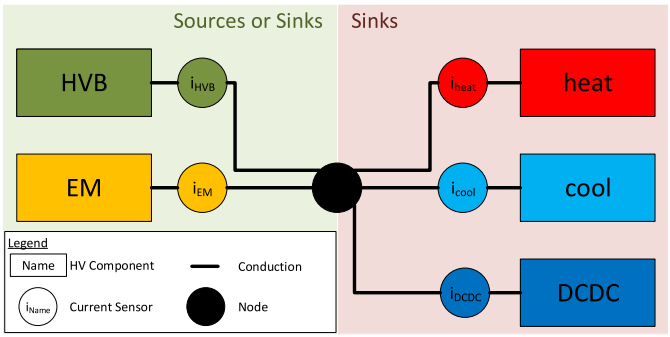

FIGURE 2. A highly simplified view of the HV power train of the EVs which we use for our studies. It consists of only a single node. Thus, the sum of all HV currents must be equal to 0 according to *Kirchhoff's current law.*

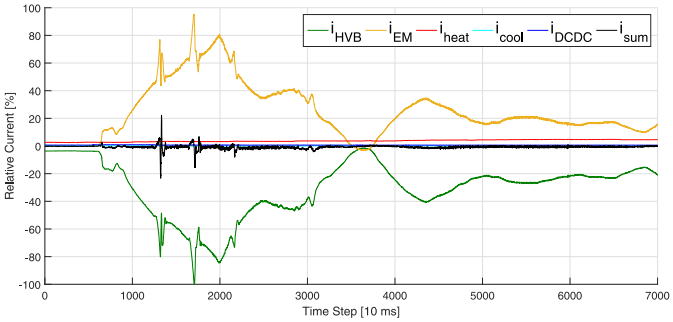

FIGURE 3. Currents of all HV components in an EV on a test drive. The sum of all currents i_{sum} is plotted in black. According to *Kirchhoff's current law*, it should be constantly 0 %. But the measurements show that the deviation i_{sum} is higher than the current of the DCDC converter i_{DCDC}.

of only a single node (compare Figure 2), the sum of all HV currents must be equal to 0 A according to Kirchhoff's law. However, considering measurement signals of EVs with distributed sensor systems, the sum of all currents can differ by up to 25 % of the maximum current (see Figure 3). If we look closer at the Root Mean Square Error (RMSE) of the sum of currents $\text{RMSE}(i_{\text{sum}}) = 0.67\%$, we realize that it has on average the same value as the current of the DCDC converter which is $\mu_{i_{\text{DCDC}}} = 0.67\%$.

Our aim is to increase the measurement accuracy to enable a minimization of the battery protection offsets. To this end, we develop and evaluate a measurement correction system based on the measurement fault detection we propose in [3]. The correction uses compressed sensing and sparsity constraints. It works based on the power train data alone and does not require further expert knowledge for manual calibration.

We explain related work and our contribution to the state of the art in Section II. In Section III, we explain the theory behind our work before we describe the practical experiments in Section IV. The results of our experiments are stated in Section V. In Section VI, we discuss the advantages and drawbacks of the proposed concepts. Finally, we draw our conclusions in Section VII.

II. STATE OF THE ART

In this work, our contribution is the development of an automated measurement correction system based on compressed sensing. We enhance the measurement deviation detection presented in [3] in such a way that it is not only able to detect, but also to recursively correct measurement faults. The proposed system is able to minimize deviations between measurements and real values. The self-reliant correction uses methods from the field of Machine Learning and does not require any expert knowledge or high calibration effort for its execution. We develop the measurement system close to its field of application. Thus, it works with only the data already available in modern series EVs without additional sensors. Another advantage of our system is that it does not increase the computation load or memory consumption of the Electronic Control Units (ECUs) of EVs. We evaluate our approach with simulation data based on previously recorded real power train data of series EVs on public roads. To the best of the authors' knowledge, this is the first time that a measurement correction system is proposed for the HV power trains of close-to-production EVs based on compressed sensing without redundant sensor systems.

Within the scope of our work, we differentiate between two different kinds of faults: *measurement* and *hardware faults*. Hardware faults describe sensors measuring correctly wrong behavior, e.g., in the case of broken actuators. Speaking of measurement faults, we mean faulty measurement data caused by, e.g., corrupted sensors. Besides the two kinds of faults, we distinguish between two groups of Machine Learning approaches for measurement correction: *off-board* and *on-board* approaches.

Off-board approaches train a model outside of its common environment. The training is based on previously recorded data or simulation. After training, the model is executed on-board and detects deviations from the previously learned behavior.

Malakar *et al.* [4] increase the quality of their measurements with an off-board approach based on Neural Networks. They detect and neglect input signals which lead to a bias in the measurement output to improve the measurement quality. However, even corrupted signals might contain parts with valuable information [5]. Neglecting the whole signal means to drop also the valid parts of the information. Thus, we prefer to correct corrupted signals instead of dropping them. Malakar's measurement environment consists only of the sun and the air. There are no further actuators. In contrast to our work, deviations in their data cannot be caused by hardware faults and are always provoked by measurement faults.

Hardware and measurement faults are distinguished by Zhao *et al.* [6]. The authors construct a simulation model of an aero-engine. Their model contains sub-models based on physical principles. The sub-models represent all components including actuators and sensors. Zhao *et al.* detect deviations between expected and measured values with the help of

Principal Component Analysis (PCA) and diagnosis models. They are able to recognize sensor faults and the affected system components due to the component-wise modeling. This modeling technique demands expert knowledge about the physical principles influencing the measurement environment. If the physical modeling is not carried out accurately enough, it can lead to a problem which is known as the *reality gap* in evolutionary robotics [7]. The reality gap denotes the phenomenon that models perform well during simulation, but fail once they are executed in the real world [5]. A reason for this failure is that the simulation data used for training is often only available in certain working conditions, whereas the environment of the real system varies across a broad range during execution [8].

The reality gap is a general problem of off-board approaches. It is their main drawback that situations might occur during execution which the algorithm did not experience during training. As a result, these algorithms are not able to adapt to new circumstances and thus perform suboptimally in certain situations [8]. Another drawback is that the training of Machine Learning methods usually is quite time-consuming [5]. Off-board approaches separate this expensive training from low-performance execution platforms. Thus, they allow cheaper hardware which is an advantage in cost-efficient industries like the automotive domain. Another advantage is that the separated training environment allows the use of higher computation and memory resources. This enables a broad range of algorithms to be considered for solving the requested problem. Additionally, the training is not necessarily required to be executed in real-time.

The major difference of on-board approaches is that the models are directly trained on the execution platform and then updated continuously.

An example for an on-board approach is the DC current calibration presented by Ren *et al.* with an high-precision current adder [9]. The adder is an additional hardware component which is able to correct faulty measurements during execution. The authors apply the adder successfully in the electrolysis industry. However, the automotive industry has different requirements. Due to the restricted available installation space and the cost efficiency resulting from mass production, we want to avoid additional hardware. Our algorithms are supposed to run with the hardware and the measurement data available in modern series vehicles.

The sensor set of a series production engine is sufficient for Lu *et al.*'s inspiring approach [8]. They introduce an on-board approach for sensor fault detection with an Extreme Learning Machine and apply it to the control system of an aero-engine. Although their approach is capable to detect bias and drift faults, it is not able to distinguish between hardware and measurement faults. Thus, if a hardware fault occurs, it must be detected separately. Nevertheless, their approach can correct measurement faults by providing compensation data.

Like Lu *et al.*, Kobayashi and Simon focus on the detection of faults in an aero-engine [10]. They propose an on-board approach which is capable to differentiate between hardware and measurement faults. Instead of an Extreme Learning Machine, Kobayashi and Simon use a bank of Kalman Filters. Each filter monitors a sensor signal separately. Kobayashi and Simon create an additional signal to detect hardware faults. They isolate corrupted signals with the help of a decision matrix. However, particularly for problems with many signals, a filter for each signal leads to a high number of filters and thus to high computational costs.

On the one hand, on-board approaches have the advantage of continuously updating their model. Thus, these algorithms are capable to adapt to never before experienced situations. On the other hand, they suffer from three main disadvantages. First, the training is required to be executed in real-time for many use cases. Second, cost effective design prevents to add additional performance and memory resources to ECUs. Especially in the automotive domain, this is an issue of high interest and restricts the capability of learning on-board of EVs. As a result, many algorithms become infeasible for automotive ECUs. Third, on-board approaches are only able to detect deviating behavior of otherwise working sensors. If the sensor returns biased measurements from the initial execution, the data is mistakenly assumed to be correct.

In our previous work, we develop a fleet-based approach with an on-board trained measurement model and an off-board fault classifier for close-to-production EVs [3]. With that hybrid approach, we combine the ability to handle unseen situations with detecting ab initio corrupted sensors. The on-board measurement model minimizes the data transferred over the air. This enables us to use the resources of a back end. In the back end, the classifier is capable to differentiate between hardware and two kinds of measurement faults. The measurement deviation detection proposed there serves as basis for our work in this paper. Here, we develop the classifier further. Additionally, we append the still missing correction of deviations between measurements and real values.

In their inspiring paper, Ohlsson *et al.* extend classical compressive sensing to quadratic relations and second order Taylor expansions [11]. They give examples for different types of measurements. Their paper serves as theoretical basis for our work described in this article. We want to extend their approach to our measurements to correct measurement faults.

Besides measurement faults and sensor uncertainties, the divergence between measurements and real values can be caused by time delays [12]. Time delays are not the focus of this work. For this work, we assume that all data is synchronized correctly and potential time delays have been detected and corrected previously. We treat with the detection of time delays in our other previous work [2], [12].

III. CONCEPTS

In the HV power train of an EV, we consider an electric system of K currents $i_1, i_2, \ldots, i_K$. To each current i_k, we dedicate one sensor providing measurements of i_k

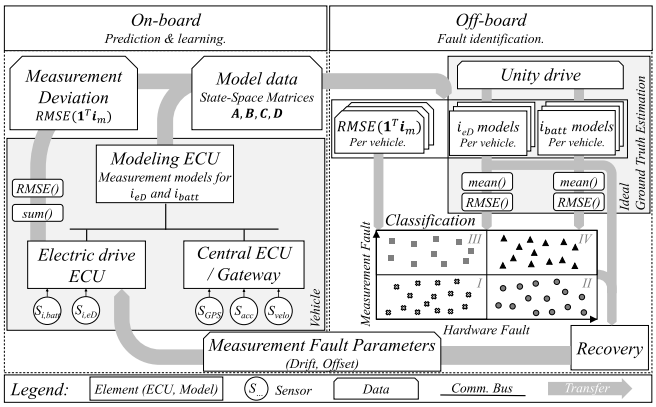

FIGURE 4. A graphical illustration of our approach. For the sake of simplicity, our approach is illustrated here only for the electric machine and the HVB. The reduction to two HV component means that the current vectors $\mathbf{i} = (i_{\mathrm{eD}}, i_{\mathrm{batt}})$ and $\mathbf{i}_m = (i_{\mathrm{eD},m}, i_{\mathrm{batt},m})$ are of dimension $K = 2$ in this case.

in real-time (without time delays), which we henceforth denote by $i_{k,m}$. We further assume that all K currents flow into the same node, so that we have their sum $\sum_{k=1}^{K} i_k$ equal to 0 by *Kirchhoff's current law* (because of measurement errors, this needs not to be true for the sum $\sum_{k=1}^{K} i_{k,m}$). To simplify notation, we denote from now on by $\mathbf{i}$ and $\mathbf{i}_m$ the K-dimensional vectors $(i_1, i_2, \ldots, i_K)^T$ and $(i_{1,m}, i_{2,m}, \ldots, i_{K,m})^T$, respectively.

Our approach is illustrated in Figure 4. We presume an EV collecting measurement data $\{\mathbf{i}_m(t)\}_{1\le t\le T}$ during driving. Based on these data, we state our problem as that of learning a correction mapping $\mathbf{i}_m \mapsto \mathbf{i}$. Our aim is to subsequently use this mapping to correct the measurement signals in real-time. The main challenge thereby is that only $\mathbf{i}_m(t)$ is available, but not the corresponding ground truth $\mathbf{i}(t)$. The lack of the ground truth makes it impossible to learn the desired mapping a priori in a supervised manner.

Thus, we want to estimate an *ideal* ground truth $\mathbf{i}_s$ first (see Section III-A). Then, a pre-trained classifier decides whether the considered EV displays hardware faults (see Section III-B). Finally, depending on the classification result, a recovery algorithm is executed to learn the desired mapping (see Section III-C).

A. IDEAL GROUND TRUTH ESTIMATION

In the sequel, we suppose the existence of a *perfect* vehicle from the investigated model, i.e., a vehicle S featuring a perfect behavior with respect to the model specifications, and equipped with perfect sensors. Let $i_{k,s}$ be the k-th current in S, and $\mathbf{i}_s = (i_{1,s}, i_{2,s}, \ldots, i_{K,s})^T$. Then, we can think of $\mathbf{i}_s$ as the should-be value of $\mathbf{i}_m$ resulting in $\mathbf{i}_s = \mathbf{i}_m$. Thus, for any *imperfect* vehicle we retrieve $\mathbf{i}_m \neq \mathbf{i}_s$, whereby this deviation can be caused by hardware faults, measurement faults or a combination of both.

On the one hand, because of potential hardware faults (e.g., a flat tire), the vehicle might display a dynamical behavior different from that of S, which then changes $\mathbf{i}_s$ into $\mathbf{i}$, but without changing the sum of currents which remains 0, i.e., $\mathbf{1}^T\mathbf{i} = \mathbf{1}^T\mathbf{i}_s = \mathbf{0}$.

On the other hand, because of measurement errors (e.g., a sensor offset/drift), a second deviation might be observed between $\mathbf{i}_m$ and $\mathbf{i}$. In contrast to hardware errors, this deviation does most likely change the sum of currents, so that in general one has $\mathbf{1}^T\mathbf{i}_m \neq \mathbf{0}$. Together, we can write

$$\underbrace{\mathbf{i}_m - \mathbf{i}_s}_{\text{total deviation}} = \underbrace{\mathbf{i}_m - \mathbf{i}}_{\text{measurement error}} + \underbrace{\mathbf{i} - \mathbf{i}_s}_{\text{hardware error}}. \tag{1}$$

We assume the existence of a fleet of N (imperfect) vehicles from the investigated model, and denote the vector of measured currents in the j-th vehicle by $\mathbf{i}_m^j$. For each j, we train a discrete *State-Space Model* consisting of the system matrices $\mathbf{A}^j$, $\mathbf{B}^j$, $\mathbf{C}^j$ and $\mathbf{D}^j$ with state $\mathbf{x}$, input $\mathbf{u}$ and output $\mathbf{i}_m^j$ based on a set $\{\mathbf{i}_m^j(t)\}_{1\le t\le T^j}$ of measurement data as described in [3]. In the next step, we estimate the currents $\{\mathbf{i}_m^j\}_{1\le j\le N}$ that would be measured across the fleet, if all vehicles drove under the exactly same conditions [3]. Simulated on these *unity drive* conditions, $\{\mathbf{i}_{sim}^j(t)\}_{1\le j\le N}$ denotes the outputs of the trained measurement models at time t according to

$$\begin{aligned} \mathbf{x}(t+1) &= \mathbf{A}^j\mathbf{x}(t) + \mathbf{B}^j\mathbf{u}_{ud}(t) \\ \mathbf{i}_{sim}^j(t) &= \mathbf{C}^j\mathbf{x}(t) + \mathbf{D}^j\mathbf{u}_{ud}(t) \end{aligned} \tag{2}$$

with $\mathbf{x}_0 = \mathbf{x}(0)$ and $\mathbf{u}(t) = \mathbf{u}_{ud}(t)$, $t \in \{1, 2, \ldots, T_{ud}\}$.

Finally, by denoting the current vector $\mathbf{i}_s(t)$ of the perfect vehicle S at time t under the unity drive conditions, we come to the following assumption:

Assumption 1: For a sufficiently large N, and at each time step t, $\mathbf{i}_s(t)$ can be accurately approximated by the average of the simulated currents $\mathbf{i}_{sim}^j(t)$ across the N vehicles in the fleet, i.e., we have

$$\mathbf{i}_s(t) \approx \tilde{\mathbf{i}}_s(t) := \frac{1}{N}\sum_{j=1}^{N} \mathbf{i}_{sim}^j(t). \tag{3}$$

This assumption is based on the observation

$$\begin{aligned} \frac{1}{N}\sum_{j=1}^{N} \mathbf{i}_{sim}^j &= \frac{1}{N}\sum_{j=1}^{N}\left(\mathbf{i}_s + \mathbf{i}_m^j - \mathbf{i}^j + \mathbf{i}^j - \mathbf{i}_s + \mathbf{i}_{sim}^j - \mathbf{i}_m^j\right) \\ &= i_s + \frac{1}{N}\sum_{j=1}^{N}\left(\mathbf{i}_m^j - \mathbf{i}^j\right) + \frac{1}{N}\sum_{j=1}^{N}\left(\mathbf{i}^j - \mathbf{i}_s\right) \\ &\quad + \frac{1}{N}\sum_{j=1}^{N}\left(\mathbf{i}_{sim}^j - \mathbf{i}_m^j\right) \end{aligned} \tag{4}$$

and the assumption that for large N, measurement, hardware and simulation errors at a certain time step average to 0 across the fleet, so that the three averages in (4) converge to 0.

B. CLASSIFICATION

Having obtained an estimation for $\mathbf{i}_m^j$ and $\mathbf{i}_s$ at each time step $t \in \{1, 2, \ldots, T_{ud}\}$, namely $\mathbf{i}_m^j(t) \approx \mathbf{i}_{sim}^j(t)$ and $\mathbf{i}_s(t) \approx$

$\tilde{\mathbf{i}}_s(t)$ as in (3), we would have almost solved the problem for the j-th vehicle, if we could exclude the possibility of hardware errors. In fact, if that was the case, we would have $\mathbf{i}^j(t) = \mathbf{i}_s(t) \approx \tilde{\mathbf{i}}_s(t)$, which we could use together with $\mathbf{i}^j_{sim}(t)$ to learn the desired mapping $\mathbf{i}^j_m \mapsto \mathbf{i}^j$ (i.e., the mapping is learned by means of the tuples $\{(\mathbf{i}^j_{sim}(t), \tilde{\mathbf{i}}_s(t))\}_{1\le t\le T_{ud}}$).

In order to reduce the complexity of our approach, we propose to first predict by means of a classifier whether any given EV displays hardware errors. If the prediction is negative (i.e., no hardware errors are detected), we proceed as described above. Otherwise, we utilize a more complex algorithm to recover the measurement error (see Section III-C). In this work, we propose and compare two different classification rules, both based on a set of $K+1$ features $f_1, f_2, \ldots, f_{K+1}$.

On the one hand, it seems reasonable that the absolute total deviation between $\mathbf{i}^j_m$ and $\mathbf{i}_s$ would in average be larger, if the j-th EV suffers not only from measurement faults but also from significant hardware faults (see (1)). Accordingly, we estimate by means of $\mathbf{i}^j_{sim}(t)$ and $\tilde{\mathbf{i}}_s(t)$ for each vehicle j and each current k the average quadratic deviation between $i^j_{k,m}$ and $i_{k,s}$, and define that as the k-th feature for the j-th vehicle

$$f^j_k = \frac{1}{T_{ud}} \sum_{t=1}^{T_{ud}} \left(i^j_{k,sim}(t) - \tilde{i}_{k,s}(t) \right)^2 \approx \mathbb{E}\left(\left(i^j_{k,m} - i_{k,s} \right)^2 \right) \quad (5)$$

for $1 \le k \le K$.

On the other hand, we define the $(K+1)$-th feature as a measure of the significance of measurement faults in the j-th vehicle. Since these faults result in $\mathbf{i}^j_m$ almost surely not summing to 0, we quantify the magnitude of the measurement faults by means of the average square of $\mathbf{1}^T\mathbf{i}^j_m$, i.e.,

$$f^j_{K+1} = \frac{1}{T^j} \sum_{t=1}^{T^j} \left(\left(\mathbf{1}^T \mathbf{i}^j_m(t) \right)^2 \right) \approx \mathbb{E}\left(\left(\mathbf{1}^T \mathbf{i}^j_m \right)^2 \right), \quad (6)$$

where a set $\{\mathbf{i}^j_m(t)\}_{1\le t\le T^j}$ of measurement data is required for the estimation.

Let $\mathbf{f}^j = (f^j_1, f^j_2, \ldots, f^j_{K+1})^T$. Based on $\mathbf{f}^j$, we decide whether the j-th vehicle displays significant hardware faults. We thereby compare between two classification rules:

- *Simple Thresholding Classifier:* The classifier decides **for** the existence of hardware errors, if

$$\eta^j := \sum_{i=1}^{K} f^j_i - \frac{1}{K} f^j_{K+1} > \delta_h \quad (7)$$

for some threshold δ_h. For an intuitive explanation, we rewrite the above criterion as

$$\eta^j \approx \sum_{k=1}^{K} \mathbb{E}\left(\left(i^j_{k,m} - i_{k,s} \right)^2 \right) - \frac{1}{K} \mathbb{E}\left(\left(\mathbf{1}^T \mathbf{i}^j_m \right)^2 \right)$$

$$= \mathbb{E}\left(\sum_{k=1}^{K} \left(i^j_{k,m} - i_{k,s} \right)^2 - \left(\frac{\mathbf{1}^T \mathbf{i}^j_m}{\sqrt{K}} \right)^2 \right)$$

$$= \mathbb{E}\left(\left\| \mathbf{i}^j_m - \mathbf{i}_s \right\|_2^2 - \left\| \mathbf{i}^j_m - \Pi_{\mathcal{H}}\left(\mathbf{i}^j_m \right) \right\|_2^2 \right) > \delta_h \quad (8)$$

by inserting (5). Here, $\Pi_{\mathcal{H}}$ is the projection operator on the hyperplane $\mathcal{H}$ given by $\mathcal{H} = \{\mathbf{x} \in \mathbb{R}^K \mid \mathbf{1}^T\mathbf{x} = 0\}$. Since $\mathbf{i}_s$ satisfies *Kirchhoff's law*, we have $\mathbf{i}_s \in \mathcal{H}$, and thus by *Pythagoras* (see Figure 5)

$$\left\| \mathbf{i}^j_m - \mathbf{i}_s \right\|_2^2 - \left\| \mathbf{i}^j_m - \Pi_{\mathcal{H}}\left(\mathbf{i}^j_m \right) \right\|_2^2 = \left\| \Pi_{\mathcal{H}}\left(\mathbf{i}^j_m \right) - \mathbf{i}_s \right\|_2^2. \quad (9)$$

Knowing that $\mathbf{i}^j$ also satisfies *Kirchhoff's law* and thus $\mathbf{i}^j \in \mathcal{H}$, we can think of $\Pi_{\mathcal{H}}(\mathbf{i}^j_m)$ as the best possible approximation of $\mathbf{i}^j$ in the absence of any further information. Thus, the above criterion is an approximation of the average squared distance $||\mathbf{i}^j - \mathbf{i}_s||_2^2$, i.e., of the average squared deviation caused by hardware faults (see (1)).

- *Decision Tree Classifier:* The classifier decides whether or not the j-th vehicle displays hardware faults based on previously induced comparison rules. The rules are learned during a training phase with an artificially created fleet of EVs. The artificial creation is based on data from real drives and has the advantage for our work that we know about the (non)-existence of hardware faults in advance. Thus, we are able to evaluate the correction.

Based on the utilized classifier, we distinguish between four different cases:

- *Case 1:* Neither hardware nor measurement faults are detected. In this case, we have a nearly *perfect* vehicle, for which no further steps are necessary.

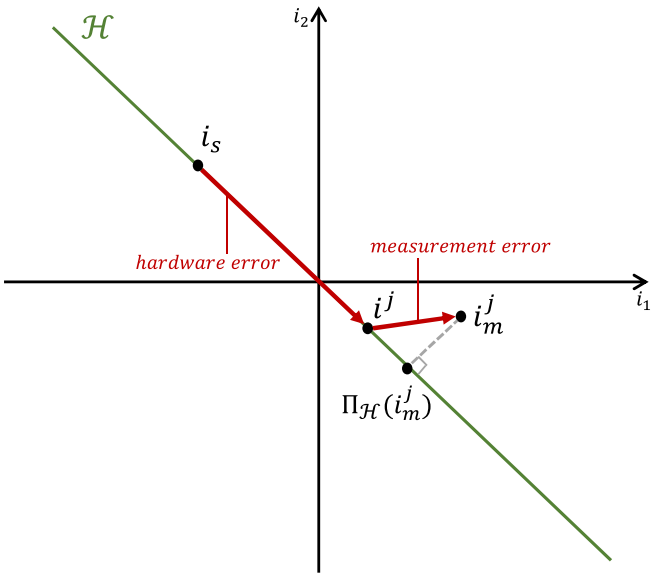

FIGURE 5. Graphical representation for $K = 2$. Here, $\mathcal{H}$ is the line described by $i_1 + i_2 = 0$. $\mathbf{i}_s$ is the should-be current and part of $\mathcal{H}$ as it satisfies *Kirchhoff's law*. Hardware errors in the j-th vehicle move $\mathbf{i}_s$ to another point on $\mathcal{H}$, here $\mathbf{i}^j$. Then, measurement errors move $\mathbf{i}^j$ outside of $\mathcal{H}$ to $\mathbf{i}^j_m$. When projected back on $\mathcal{H}$, $\mathbf{i}^j_m$ returns $\Pi_{\mathcal{H}}(\mathbf{i}^j_m)$. In (8), we use *Pythagoras theorem* on the triangle spanned by the points $\mathbf{i}_s$, $\mathbf{i}^j_m$ and $\Pi_{\mathcal{H}}(\mathbf{i}^j_m)$. The triangle is right-angled at $\Pi_{\mathcal{H}}(\mathbf{i}^j_m)$.

- *Case 2:* Only hardware faults are detected. No further steps are necessary in this case, too, as our goal is to correct measurement faults only.
- *Case 3:* Only measurement faults are detected. Here, only one further step is necessary, namely to use the tuples $\{(\mathbf{i}^j_{sim}(t), \mathbf{i}_s(t))\}_t$ to learn the correction mapping $\mathbf{i}^j_m \mapsto \mathbf{i}^j$.
- *Case 4:* Both types of faults are detected, i.e., the deviation between $\mathbf{i}^j_m$ and $\mathbf{i}_s$ is to be decomposed into its two components as in (1). Without further assumptions, the problem is however ill-posed, as there is an infinite number of such decompositions. In order to provide a unique solution, we propose in the sequel a set of assumptions which we find to be both reasonable and sufficient to make the problem well-posed again.

C. ERROR MODEL AND RECOVERY

In this section, we first present the error models we assume for measurement and hardware faults. Since hardware faults do not alter the validity of physical laws, here in particular *Kirchhoff's law*, one can in general model them using a continuous mapping $\mathbf{f}_h : \mathbf{i}_s \mapsto \mathbf{i}$, such that it holds

$$\mathbf{i}_s \in \mathcal{H} \implies \mathbf{f}_h(\mathbf{i}_s) \in \mathcal{H}. \tag{10}$$

In this work, we choose a simple linear model for $\mathbf{f}_h$ satisfying (10), namely:

$$\mathbf{f}_h(\mathbf{i}_s) = (1 + d_h)\mathbf{i}_s + \mathbf{o}_h, \tag{11}$$

where $d_h \in \mathbb{R}$ is a drift scalar and $\mathbf{o}_h \in \mathcal{H}$ is an offset vector. We retrieve

$$\begin{aligned} \mathbf{i}_s, \mathbf{o}_h \quad &\in \quad \mathcal{H} \\ &\implies \mathbf{1}^T\mathbf{i}_s = \mathbf{1}^T\mathbf{o}_h = \mathbf{0} \\ &\implies (1 + d_h)\mathbf{1}^T\mathbf{i}_s + \mathbf{1}^T\mathbf{o}_h = \mathbf{0} \\ &\implies \mathbf{1}^T\mathbf{f}_h(\mathbf{i}_s) = \mathbf{0} \\ &\implies \mathbf{f}_h(\mathbf{i}_s) \in \mathcal{H}. \end{aligned} \tag{12}$$

Similarly, we model measurement faults as a linear mapping $\mathbf{f}_m : \mathbf{i} \mapsto \mathbf{i}_m$. However, we model the mapping without the constraints from (10), since the noisy vector $\mathbf{i}_m$ does not need to satisfy *Kirchhoff's law*. This results in

$$\mathbf{f}_m(\mathbf{i}) = (\mathbf{I} + \mathbf{D}_m)\mathbf{i} + \mathbf{o}_m, \tag{13}$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{D}_m \in \mathbb{R}^{K\times K}$ is a diagonal drift matrix, and $\mathbf{o}_m \in \mathbb{R}^K$ is an offset vector. Note that in these terms, the goal of this work is to learn the inverse mapping $\mathbf{f}_m^{-1}(\mathbf{i}_m)$. As $\mathbf{i}$ is unknown, we instead aim to learn the parameters $\mathbf{D}_m$ and $\mathbf{o}_m$ with the help of the tuples $(\mathbf{i}_m, \mathbf{i}_s)$.

From (11) and (13), we retrieve

$$\begin{aligned} \mathbf{i}_m &= \mathbf{f}_m \circ \mathbf{f}_h(\mathbf{i}_s) = (\mathbf{I} + \mathbf{D}_m)((1 + d_h)\mathbf{i}_s + \mathbf{o}_h) + \mathbf{o}_m \\ &= (\mathbf{I} + \mathbf{D}_m)(1 + d_h)\mathbf{i}_s + (\mathbf{I} + \mathbf{D}_m)\mathbf{o}_h + \mathbf{o}_m. \end{aligned} \tag{14}$$

However, we cannot generally recover the parameters (in particular $\mathbf{D}_m$ and $\mathbf{o}_m$) from tuples in the form $(\mathbf{i}_m, \mathbf{i}_s)$ (and even less $(\mathbf{i}_{sim}, \tilde{\mathbf{i}}_s)$). To solve this problem, we consider the classification result and differentiate between two cases.

1) RECOVERY WITH NO HARDWARE ERRORS

This case is equivalent to setting both d_h and $\mathbf{o}_h$ to 0. We retrieve $\mathbf{i}_m = (\mathbf{I}+\mathbf{D}_m)\mathbf{i}_s+\mathbf{o}_m$ from (14). We learn the parameters $\mathbf{D}_m$ and $\mathbf{o}_m$ straightforward by linearly regressing $\mathbf{i}_{sim}(t)$ on $\tilde{\mathbf{i}}_s(t)$. More precisely, we execute in total K regressions where we regress $i_{sim,k}(t)$ on $i_{m,k}(t)$ for each k.

2) RECOVERY WITH HARDWARE ERRORS

Here, the problem is ill-posed without additional knowledge. For example, if $(d_h, \mathbf{o}_h, \mathbf{D}_m, \mathbf{o}_m)$ satisfies (14), then $(d_h, \mathbf{o}_h + \mathbf{e}, \mathbf{D}_m, \mathbf{o}_m - (\mathbf{I} + \mathbf{D}_m)\mathbf{e})$ does so, too, for any $\mathbf{e} \in \mathcal{H}$. In this work, we avoid such ill-posedness with the following assumption.

Assumption 2: The vector $\mathbf{x}$ containing all parameters $(d_h, \mathbf{o}_h, \mathbf{D}_m, \mathbf{o}_m)$ is a sparse vector, meaning that only few of the therein included parameters are non-zero.

This assumption bases upon the following reasoning: in one vehicle, it is very unlikely that all, or at least many, fault sources exist concurrently. All faults at the same time would mean that all K involved sensors suffer from offset or drift faults, or even both. Additionally, K currents are affected by hardware faults without exception. Thus, it is reasonable to assume that only relatively few sources of faults exist in the same vehicle at the same time.

Accordingly, we recover $\mathbf{x}$ as follows. First, we estimate the expressions $(\mathbf{I}+\mathbf{D}_m)(1+d_h)$ and $(\mathbf{I}+\mathbf{D}_m)\mathbf{o}_h+\mathbf{o}_m$ based on the tuples $(\mathbf{i}_{sim}(t), \tilde{\mathbf{i}}_s(t))$. We do this estimation by linearly regressing $\mathbf{i}_{sim}(t)$ on $\tilde{\mathbf{i}}_s(t)$. More precisely, we define the k-th diagonal element of $\mathbf{D}_m$ as $d_{m,k}$, and the k-th elements of $\mathbf{o}_h$ and $\mathbf{o}_m$ as $o_{h,k}$ and $o_{m,k}$, respectively. Then, we retrieve for each $k \in \{1, 2, \ldots, K\}$

$$\begin{aligned} \big(1 + d_{m,k}\big)(1 + d_h) &\approx a_k \\ \big(1 + d_{m,k}\big)o_{h,k} + o_{m,k} &\approx b_k \end{aligned} \tag{15}$$

where a_k and b_k are the slope and intercept estimates obtained from regressing $(i_{sim,k}(1), \ldots, i_{sim,k}(T_{ud}))^T$ on $(\tilde{i}_{s,k}(1), \ldots, \tilde{i}_{s,k}(T_{ud}))^T$. Next, we define $\mathbf{y} = (a_1, \ldots, a_K, b_1, \ldots, b_K)^T$ and $\tilde{\mathbf{x}} = (d_h, o_{h,1}, \ldots, o_{h,K}, d_{m,1}, \ldots, d_{m,K}, o_{m,1}, \ldots, o_{m,K}, 1)^T \in \mathbb{R}^{3K+2}$. We then rewrite each of the estimated terms in a quadratic form $\tilde{\mathbf{x}}^T\mathbf{Q}\tilde{\mathbf{x}}$ of $\tilde{\mathbf{x}}$, for example

$$\begin{aligned} \big(1 + d_{m,k}\big)(1 + d_h) &= \left(\tilde{\mathbf{x}}^T\mathbf{e}_{3K+2}\right)^2 + \tilde{\mathbf{x}}^T\mathbf{e}_{3K+2}\mathbf{e}^T_{K+1+k}\tilde{\mathbf{x}} \\ &\quad + \tilde{\mathbf{x}}^T\mathbf{e}_{3K+2}\mathbf{e}_1^T\tilde{\mathbf{x}} + \tilde{\mathbf{x}}^T\mathbf{e}_{K+1+k}\mathbf{e}_1^T\tilde{\mathbf{x}} \\ &= \tilde{\mathbf{x}}^T\Big(\mathbf{e}_{3K+2}\mathbf{e}^T_{3K+2} + \mathbf{e}_{3K+2}\mathbf{e}^T_{K+1+k} \\ &\quad + \mathbf{e}_{3K+2}\mathbf{e}_1^T + \mathbf{e}_{K+1+k}\mathbf{e}_1^T\Big)\tilde{\mathbf{x}} \\ &:= \tilde{\mathbf{x}}^T\mathbf{Q}_k\tilde{\mathbf{x}}, \end{aligned} \tag{16}$$

where the notation $\mathbf{e}_{3K+2}$ denotes the canonical vector with respect to $3K + 2$. Similarly, we obtain $(1+d_{m,k})o_{h,k}+o_{m,k} = \tilde{\mathbf{x}}^T\mathbf{Q}_{K+k}\tilde{\mathbf{x}}$ and rewrite (15) as

$$\forall i \in \{1, 2, \ldots, 2K\} \quad \tilde{\mathbf{x}}^T\mathbf{Q}_i\tilde{\mathbf{x}} \approx \mathbf{y}^T\mathbf{e}_i. \tag{17}$$

Finally, we recover $\mathbf{x}$ (resp. $\tilde{\mathbf{x}}$) by solving the optimization problem

$$\begin{aligned} \min_{\tilde{\mathbf{x}}} \ \|\tilde{\mathbf{x}}\|_1 \ \text{ s.t. } \ & \sum_{i=1}^{2K} \left(\tilde{\mathbf{x}}^T \mathbf{Q}_i \tilde{\mathbf{x}} - \mathbf{y}^T \mathbf{e}_i\right)^2 \leq \epsilon, \\ & \tilde{\mathbf{x}}^T \mathbf{e}_{3K+2} = 1, \\ & \tilde{\mathbf{x}}^T \left(\sum_{i=1}^{K} \mathbf{e}_{1+i}\right) = 0. \end{aligned} \tag{18}$$

Thereby, ϵ is an error threshold. The constraint $\tilde{\mathbf{x}}^T \mathbf{e}_{3K+2} = 1$ forces the last element of $\tilde{\mathbf{x}}$ to be equal to 1, and the constraint $\tilde{\mathbf{x}}^T (\sum_{i=1}^{K} \mathbf{e}_{1+i}) = 0$ makes sure that the hardware offsets $o_{h_1}, \ldots, o_{h,K}$ sum up to 0 to satisfy (10). To solve the optimization problem from (18), we use the Quadratic Basis Pursuit (QBP) algorithm suggested in [11]. This algorithm uses a lifting technique to convexify the above problem and thus makes it computationally tractable.

In the end of this section, we want to emphasize that in theory, we could bypass the classification step from Section III-B. Our proposed recovery does not require the existence of hardware errors. It could theoretically recover the parameters successfully even when $d_h = 0$ and $\mathbf{o}_h = \mathbf{0}$. Thus, we could directly execute the optimization in (18) for all EVs, irrespective of whether or not they are affected by hardware errors. We would only need to run the optimization procedure in (18) when necessary. This would reduce the complexity without negatively affecting the performance. However, in the presence of classification errors, the recovery' success rate might be negatively affected. On the one hand, in case of false positives, the algorithm would try to recover the parameters using the optimization problem from (18). Instead of the simple and highly reliable regression from Section III-C1, the optimization problem might fail to recover the parameters. On the other hand, in case of false negatives, the recovery algorithm would assume wrongly that there are no hardware errors. The wrong assumption would lead to a systematic estimation error. From these observations, we conclude that false negatives are more harmful than false positives in our case. We will take this fact into account for the evaluation of the classification step in Section V-B.

IV. EXPERIMENTAL SETUP

In this section, we describe the experiments we conduct in order to evaluate the suggested method.

A. ARTIFICIAL GROUND TRUTH

Each experiment is based on an artificially created data set $\mathcal{D} = \{\mathbf{i}_s(t)\}_{1 \leq t \leq T}$ of ground truth current signals. To make sure that the data set is realistic, we start with real measurement data $\mathcal{D}' = \{\mathbf{i}_m(t)\}_{1 \leq t \leq T}$ of HV current signals recorded during driving. The data set is modified in such a way that at each time step t, the condition $\mathbf{i}_s(t)^T \mathbf{1} = \mathbf{0}$ is satisfied. We guarantee the satisfaction of the condition by setting

$$i_{k,s}(t) = \begin{cases} i_{k,m}(t) & \text{for } k \in \{1, \ldots, K-1\} \\ -\sum_{j=1}^{K-1} i_{j,m}(t) & \text{for } k = K \end{cases}. \tag{19}$$

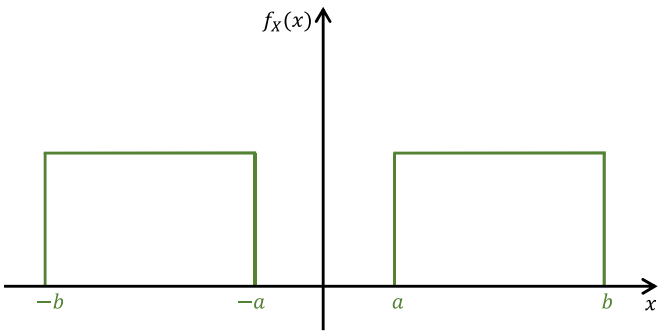

FIGURE 6. The density function.

B. ARTIFICIAL RANDOM FLEET

Besides the artificial ground truth, we create a random fleet of N EVs based on $\mathcal{D}$ and the model given in (14). We do this by randomly drawing the parameters $(d_h, \mathbf{o}_h, \mathbf{D}_m, \mathbf{o}_m)$ for each EV, so that

- the constraint $\mathbf{o}_h^T \mathbf{1} = \mathbf{0}$ is satisfied,[1]
- for a given sparsity level S, only $(100 - S)\%$ of the parameters (up to rounding) are nonzero. (For example, if $K = 3$ and $S = 60$, we have in total $3K + 1 = 10$ parameters. Only 4 of these are set to be nonzero. The support of the parameter vector is thereby chosen at random.),
- each nonzero parameter is set as the realization of some random variable $X = (2B - 1)U$ where

$$B \sim Bernoulli\left(\frac{1}{2}\right) \ \text{ and } \ U \sim \mathcal{U}(a, b) \tag{20}$$

for some hyper-parameters a and b. The corresponding density function is depicted in Fig. 6. The rationale behind this choice of the distribution is to prevent a previously nonzero parameter to take very small values (and thus become approximately zero).

Once we finished the sampling, we define the measurement signal $\mathbf{i}_m^j(t)$ in the j-th vehicle as

$$\mathbf{i}_m^j(t) = \left(\mathbf{I} + \mathbf{D}_m^j\right)\left(1 + d_h^j\right)\mathbf{i}_s(t) + \left(\mathbf{I} + \mathbf{D}_m^j\right)\mathbf{o}_h^j + \mathbf{o}_m^j, \tag{21}$$

where d_h^j, $\mathbf{o}_h^j$, $\mathbf{D}_m^j$ and $\mathbf{o}_m^j$ are the sampled parameters.

On top of the above listed sampling conditions on the individual vehicle level, we make sure that on the fleet-level, the following two conditions are satisfied:

- The fleet is **balanced**, i.e., the ratio of EVs affected by hardware faults to those which are not is around 1:1. Thus, we force $\mathbf{o}_h$ and $\mathbf{d}_h$ to be zero for half of the vehicles in the artificial fleet. We use this trick because for a larger K, and a smaller sparsity level S, almost all EVs would be affected by hardware faults.

1. In our implementation, we utilize a slightly different parametrization, namely by only preserving the first $K - 1$ components of $\mathbf{o}_h$ as unknown parameters, and setting $o_{h,K} = -\sum_{j=1}^{K-1} o_{h,j}$. This constraint is then satisfied automatically.

According to our experiences, this would be very unrealistic. Especially, if the fleet contains EVs of young age with a low amount of vehicle miles traveled.

- The fleet is **symmetric**, i.e., we have for all $t \in \{1, \dots, T\}$

$$\mathbf{i}_s(t) = \frac{1}{N} \sum_{j=1}^{N} \mathbf{i}_m^j(t). \tag{22}$$

This condition makes sure that Assumption 1 is nearly satisfied.[2] It is enforced by creating for each vehicle in the fleet with parameters $(d_h, \mathbf{o}_h, \mathbf{D}_m, \mathbf{o}_m)$, three other vehicles with parameters $(-d_h, -\mathbf{o}_h, -\mathbf{D}_m, -\mathbf{o}_m)$, $(d_h, \mathbf{o}_h, -\mathbf{D}_m, \mathbf{o}_m)$, and $(-d_h, -\mathbf{o}_h, \mathbf{D}_m, -\mathbf{o}_m)$, respectively. This means that we assume N to be divisible by 4.

C. SIMULATION MODELS

In the previous two subsections, we define the ground truth signal $\mathbf{i}_s(t)$ and the measurement signals $\mathbf{i}_m^j(t)$ for all $j \in \{1, \dots, N\}$ and $t \in \{1, \dots, T\}$. The only step left is to define the simulation signals $\mathbf{i}_{sim}^j(t)$ for all $j \in \{1, \dots, N\}$. To do so, we differentiate between two methods:

- *State-Space Simulation:* For each j, we split the available data into two parts. We define $T_1 = \lfloor 0.8T \rfloor$ and use the first part $\{\mathbf{i}_m^j(t)\}_{1 \le t \le T_1}$ to train and validate a State-Space Model which is able to simulate the dynamics of the j-th vehicle. By simulating the model on the remaining 20% of the available data,[3] we obtain $\{\mathbf{i}_{sim}^j(t)\}_{T_1+1 \le t \le T}$.
- *Artificial Simulation:* Alternatively, we assume that the error $\mathbf{e}_{sim}^j = \mathbf{i}_{sim}^j - \mathbf{i}_m^j$ is distributed as $\mathcal{N}(\mathbf{0}, \sigma_{sim}^2 \mathbf{I})$ for some variance σ_{sim}^2 and define for $j \in \{1, \dots, N\}$ and $t \in \{T_1 + 1, \dots, T\}$

$$\mathbf{i}_{sim}^j(t) = \mathbf{i}_m^j(t) + \mathbf{e}_{sim}^j(t) \quad \text{where} \quad \mathbf{e}_{sim}^j(t) \sim \mathcal{N}\left(\mathbf{0}, \sigma_{sim}^2 \mathbf{I}\right). \tag{23}$$

While our method originally relies on the state-space simulation as described in [3], the second method allows us to provide more general results by investigating the effect of σ_{sim}^2 on the overall performance of our method. Besides that, it makes it easier to conduct experiments, especially for larger values of N.

D. EXPERIMENTS

Following the previous sections, each experiment consists of the following steps:

1) As described in Section IV-A, generate for all K currents and time steps $t \in \{1, \dots, T\}$ an artificial ground truth $\mathbf{i}_s(t)$.
2) Depending on $\mathbf{i}_s(t)$, $j \in \{1, \dots, N\}$ and $t \in \{1, \dots, T\}$, create a balanced artificial random fleet of N vehicles as described in Section IV-B with sparsity level S to obtain $\mathbf{i}_m^j(t)$.
3) Choose one of the simulation methods described in Section IV-C to obtain the simulated currents $\{\mathbf{i}_{sim}^j(t)\}_{T_1+1 \le t \le T}$.
4) Based on $\{\mathbf{i}_{sim}^j(t)\}_{T_1+1 \le t \le T}$, estimate the ground truth as described in Section III-A according to

$$\tilde{\mathbf{i}}_s(t) = \frac{1}{N} \sum_{j=1}^{N} \mathbf{i}_{sim}^j(t). \tag{24}$$

5) Based on $\{\tilde{\mathbf{i}}_s(t)\}_{T_1+1 \le t \le T}$ and $\{\mathbf{i}_{sim}^j(t)\}_{T_1+1 \le t \le T}$, **classify** whether the j-th vehicle is affected by **measurement** and/or **hardware faults** as described in Section III-B. Depending on the experiment, we either utilize the simple thresholding classifier with parameter δ_h, or a previously trained decision tree classifier. In the latter case, the classifier is trained on another data set $\mathcal{D}_{train}$. This data set is created artificially in the same way (i.e., using the same sampling scheme and parameters) as $\mathcal{D}$.
6) For each vehicle in the fleet, follow the descriptions in Section III-C to recover the parameters $(d_h, \mathbf{o}_h, _m, \mathbf{o}_m)$.
7) Calculate the recovery rate, i.e., the ratio of cases for which $\mathbf{D}_m$ and $\mathbf{o}_m$ have been recovered with a relative error below 10%.[4] The reason we only focus on $\mathbf{D}_m$ and $\mathbf{o}_m$ is that we do not need the other parameters to be able to correct the current via the inverse mapping

$$\mathbf{i}_m \mapsto (\mathbf{I} + \mathbf{D}_m)^{-1}(\mathbf{i}_m - \mathbf{o}_m). \tag{25}$$

V. RESULTS

In this section, we evaluate our approach in an end-to-end fashion. Therefor, we run the experiment from Section IV-D multiple times with different parameter combinations (i.e., values of the number of currents K, the fleet size N, the sparsity level S, etc.). By fixing all parameters but a few, we investigate in a number of experiment series the effect of the variable parameters on the overall achieved performance as measured by the recovery rate defined above. Due to the large number of parameters involved, we will thereby restrict our analysis to the following effects.

A. EFFECT OF THE SIMULATION NOISE VARIANCE

All the steps of the proposed approach rely on simulated signals, so that a too large simulation error is likely to cause the recovery to fail. To avoid such a failure, it is important to quantify how *good* the simulation should be for the algorithm to achieve a sufficiently high recovery rate. In other words, we investigate the stability of our approach against simulation noise. For this investigation, we fix $N = 1000$, $K = 2$,

2. Note that Assumption 1 uses the simulated currents and not the measured currents.

3. For both training and simulation, we also use the corresponding input vector $\{\mathbf{u}(t)\}_{1 \le t \le T}$ which is available from the original data set used to build the artificial ground truth (see Section IV-A).

4. For our purposes, it is sufficient to assume cases with a relative error below 10% as recovered. Of course, the reader is free to choose a value depending on the individual use case.

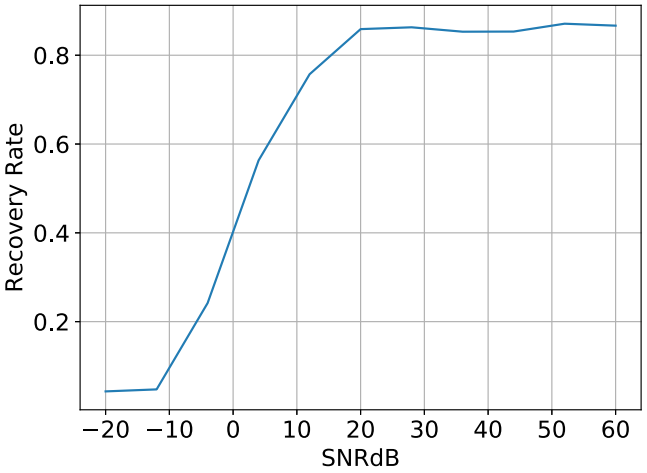

FIGURE 7. The recovery rate is plotted against the signal-to-noise ratio *SNRdB*.

and $S = 60$. For our experiments, we use the signal-to-noise ratio *SNRdB* to measure the goodness of a simulation. We run for the experiment 10 times for several signal-to-noise ratios in the interval $[-20, 60]$ with

$$\sigma_{sim}^2 = 10^{-SNRdB/10} \cdot P, \tag{26}$$

where P is the average power of $i_{s,1}(t)$,[5] i.e., $P = T^{-1} \sum_{t=1}^{T} i_{s,1}(t)^2$. Note that in order to isolate the investigated effect from other noise sources, we enforce a symmetric fleet and a perfect classifier. A non-symmetric fleet would introduce an additional noise in the ground truth estimation (see (4)), and false negatives in the classification would introduce a systematic error in the parameter estimation.[6]

In Figure 7, we plot the average recovery rate (across the 10 repetitions) as a function of *SNRdB*. We see that an SNR above around 15 dB is necessary for the algorithm to have a recovery rate above 80%, which we consider to be a sufficiently mild requirement on the simulation model. In fact, the State-Space Models introduced in our previous work [3] largely satisfy this requirement.

B. EFFECT OF THE CLASSIFIER

As discussed in Section III-C, classification errors in general, and false negatives in particular, are harmful to the recovery rate. To investigate which of the two suggested classifiers (simple thresholding rule vs. decision tree) produces better results, we conduct the following experiment. We set $N = 1000, K = 2$ and $S = 60$ and run the experiment from Section IV-D 10 times for each classifier with $SNRdB = 20$ in (26). Thereby, on the one hand, we set for the thresholding classifier $\delta_h = \tilde{\delta}_h \sigma_\eta$, where $\tilde{\delta}_h$ is empirically set to $1/3$, and

5. Because $K = 2$, we have $i_{s,1}(t) = i_{s,2}(t)$, so that P is also the average power of $i_{s,2}(t)$.

6. For false negatives, the classifier wrongly decides there are no hardware faults. All parameters relating to hardware faults, i.e., d_h and $\mathbf{o}_h$, are wrongly set to 0, which leads to systematic errors in the estimation of $\mathbf{d}_m$ and $\mathbf{o}_m$.

TABLE 1. Results of the second experiment for simple thresholding, the decision tree classifier and the perfect classifier.

	Simple thresholding	Decision tree	Perfect
Mean f_2 score	0.68	**0.84**	1.00
Mean recovery rate	0.84	**0.86**	0.91
Mean complexity	0.93	**0.41**	0.40

σ_η is the standard deviation of the decision criterion across the fleet (see (7)), i.e.,

$$\sigma_\eta = \sqrt{\frac{1}{N-1} \sum_{j=1}^{N} (\eta_j - \bar{\eta}_j)} \quad \text{with} \quad \bar{\eta}_j = \sum_{j=1}^{N} \eta_j. \tag{27}$$

On the other hand, we train the decision tree on an artificial fleet of $3,000$ vehicles. The fleet is created using the same hyper-parameters as the ones utilized to create the primary fleet (i.e., with the same sparsity level S and parameter distributions a and b). We further restrict the tree depth to 6 to prevent overfitting.

On top of the two suggested classifiers, we include the results obtained using a perfect classifier as benchmark. For the evaluation, we use the following three metrics:

- The f_2 score as a performance measure of each classifier. We chose this score since we consider false negatives to be more harmful than false positives, and thus consider recall to be more important than precision.[7]
- The recovery rate as a measure of the end-to-end performance.
- The percentage of cases for which the optimization procedure in (18) is executed (i.e., the percentage of cases classified as displaying hardware errors) as a complexity measure.

Table 1 summarizes the results of this experiment. We see that the decision tree classifier outperforms the simple thresholding rule on all evaluation criteria. In particular, thanks to its better f_2 score, the decision tree classifier is able to reduce the complexity score to only 0.41, so that the classification step is fulfilling its originally conceived target. Note that this gain in computational resources does compensate the resources needed to train the tree in the first place.

C. EFFECT OF THE SPARSITY LEVEL

The sparsity constraint is paramount to make the originally ill-posed problem well-posed again. We therefore want to evaluate our method for various values of S, and find out *how much sparsity* is actually required to obtain a high recovery rate. To do this, we run our algorithm in an end-to-end fashion using the same setting as the last experiment, only this time fixing the classifier to be a decision tree, and letting S vary in $\{20, 40, 60, 80\}$. The results are summarized in Figure 8.

7. This corresponds to the particular choice of $\beta = 2$ in the more general f_β score defined as $f_\beta = (1 + \beta^2) pr / (\beta^2 p + r)$, which is a weighted harmonic mean of recall and precision, where the weight for recall (r) is β^2 times that of precision (p).

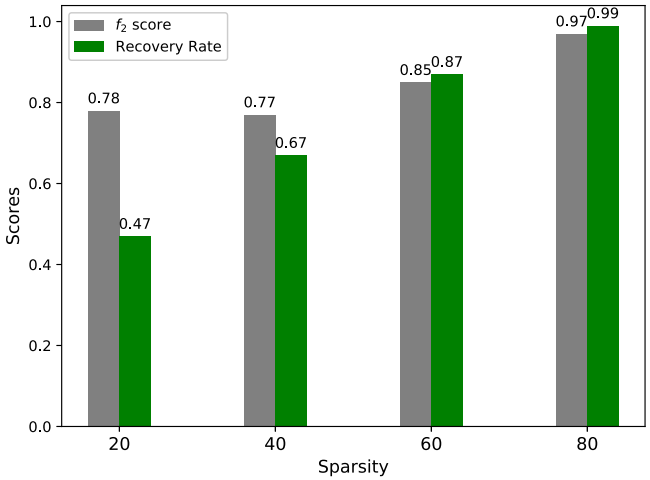

FIGURE 8. Classification and recovery rates for different sparsity levels.

TABLE 2. Results of the end-to-end evaluation using state-space models.

Sparsity	S=20	S=40	S=60	S=80
Mean f_2 score	0.76	0.75	0.91	0.91
Recovery rate ($< 10\%$)	0.45	0.49	0.75	0.96
Recovery rate ($< 25\%$)	0.63	0.71	0.85	0.98

D. OVERALL EVALUATION

As last experiment, we evaluate our method in an end-to-end fashion using the state-space approach suggested in [3] as concrete simulation model. We thereby set $K = 2$ and $N = 500$, enforce the fleets to be symmetric and balanced, and use pre-trained decision trees for the classification step. Each tree is trained on an artificial data set created using the artificial simulation method with $SNRdB = 20$. When executing this for various values of S, we obtain the results in Table 2.

VI. DISCUSSION

Our approach for the correction of measurement faults in the power train of EVs consists of 3 steps. First, we estimate the ideal ground truth. Second, we classify hardware and measurement faults. Finally, the measurement faults are recovered based on the classification result.

To evaluate our approach, we execute 4 experiments. In the first experiment, we show that the recovery rate increases for lower simulation errors. Our previously implemented State-Space Models satisfy the experimental requirement of 15 dB needed to achieve a high recovery rate ($> 80\%$, see Figure 7). It is interesting that the recovery rate improves just marginally at higher dB levels. This means that we need a relatively accurate simulation model but do not require highest accuracy. Thus, our approach is capable to handle simulation errors to a certain extent.

Our second experiment shows that decision trees are superior to the thresholding rule. There are several reasons for the decision trees' advantages. First, they are trained on the data before they are deployed in contrast to rules which are set manually. Second, besides simple thresholding, they are able to learn complex decisions rules and third, they take all defined features as input. However, decision trees also have some drawbacks. They require training which in turn requires the creation of an artificial fleet. For the fleet's creation, we need to state distributional assumptions about the fleet. These assumptions might add uncertainty to our data and lead to wrong decisions. Overall, even if the improvement of decision trees to the recovery rate is relatively small (0.86 vs. 0.84 as can be seen in Table 1), we still find them better as they achieve a better classification rate (0.84 vs. 0.68 as can be seen in Table 1), which, as discussed in the end of Section III, reduces the required computational power.

The main insight of our third experiment is that the sparsity assumption is the crucial influencing factor on the recovery rate. The recovery rate can be decreased to 47% by low sparsity levels. This behavior is comprehensible since the recovery algorithm is conceived to work with sparse vectors and thus to minimize the 1-norm. The goal of sparsity constraints is to deal with the ill-posedness of problems. Assuming low sparsity means to deal with highly ill-posed problems and low recovery.

From the fourth experiment, we learn that the recovery rate is lower (e.g., 75% instead of 86% for $S = 60$) for State-Space models instead of artificial simulation (see Table 1 and Table 2). This is surprising for us because State-Space Models also fulfill the dB requirement of the first experiment. A possible reason for the lower recovery rate might be that the errors induced by the State-Space Models satisfy the dB requirement for the standard deviation but are not normally distributed. This might lead to some correlations and decrease the accuracy of the regression in the first recovery step. Further research into the exact reasons for the lower recovery rate of State-Space Models is up to our future work.

Regarding the whole work described in this article, we realize that Assumption 2 is crucial for the results. Lower sparsities induce worse results. This means that our algorithm is sensitive to the previously stated assumption. We expect this drawback to be solved by stabilizing the algorithm with respect to simulation errors. Overall, if the assumptions are fulfilled, the results look very promising. The good results with high recovery rates serve as basis for our future work. Furthermore, we want to replace the simple linear error model by more complex non-linear models.

VII. CONCLUSION

This article presents an advanced fleet-based framework to correct measurement faults in the power trains of EVs. Through a comparison with the mean behavior of the fleet, we are able to classify whether a certain vehicle suffers from significant hardware errors. Then, based on the classification result, we use a combination of linear regression and convex sparse optimization to recover the parameters defining measurement errors. Using relatively mild and realistic assumptions, we thereby achieve a high recovery rate of up to 90%. Overall, our framework is able to correct measurement

faults in a completely automated way, and without additional sensors or computational power on-board of the EV.

ABBREVIATIONS

ECU	Electronic Control Unit
EV	Electric Vehicle
HV	High Voltage
HVB	High Voltage Battery
PCA	Principal Component Analysis
QBP	Quadratic Basis Pursuit
RMSE	Root Mean Square Error

REFERENCES

[1] P. Komarnicki, J. Haubrock, and Z. A. Styczynski, "Elektrische Komponenten des E-Kfz," in *Elektromobilität und Sektorenkopplung*. Heidelberg, Germany: Springer, 2018, pp. 61–109. [Online]. Available: https://link.springer.com/chapter/10.1007%2F978-3-662-56249-9_3

[2] J. Pfeiffer, X. Wu, and A. Ayadi, "Evaluation of three different approaches for automated time delay estimation for distributed sensor systems of electric vehicles," *Sensors*, vol. 20, no. 2, p. 351, Jan. 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/2/351

[3] J. Pfeiffer, P. Wolf, and R. Pereira, "A fleet-based machine learning approach for automatic detection of deviations between measurements and reality," in *Proc. IEEE Intell. Veh. Symp. (IV)* Paris, France, Jun. 2019, pp. 2086–2092. [Online]. Available: https://ieeexplore.ieee.org/document/8813858/

[4] N. K. Malakar *et al.*, "Estimation and bias correction of aerosol abundance using data-driven machine learning and remote sensing," in *Proc. IEEE Conf. Intell. Data Understand. (CIDU)*, Boulder, CO, USA, 2012, pp. 24–30.

[5] S. Koos, J.-B. Mouret, and S. Doncieux, "The transferability approach: Crossing the reality gap in evolutionary robotics," *IEEE Trans. Evol. Comput.*, vol. 17, no. 1, pp. 122–145, Feb. 2013.

[6] Z. Zhao, Y.-G. Sun, and J. Zhang, "Fault detection and diagnosis for sensor in an aero-engine system," in *Proc. IEEE Chin. Control Decis. Conf. (CCDC)*, Yinchuan, China, 2016, pp. 2977–2982.

[7] N. Jakobi, P. Husbands, and I. Harvey, "Noise and the reality gap: The use of simulation in evolutionary robotics," in *Proc. 3rd Eur. Conf. Artif. Life*, vol. 929, 1995, pp. 704–720.

[8] J. Lu, J. Huang, and F. Lu, "Sensor fault diagnosis for aero engine based on online sequential extreme learning machine with memory principle," *Energies*, vol. 10, no. 1, p. 39, 2017.

[9] S. Ren, Q. Liu, and X. Liu, "Heavy DC current calibration method based on high-precision current adder," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 12, pp. 3039–3044, Dec. 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6867367/

[10] T. Kobayashi and D. L. Simon, "Evaluation of an enhanced bank of Kalman filters for in-flight aircraft engine sensor fault diagnostics," *J. Eng. Gas Turb. Power*, vol. 127, no. 3, p. 497, 2005.

[11] H. Ohlsson, A. Y. Yang, R. Dong, M. Verhaegen, and S. S. Sastry, "Quadratic basis pursuit," Jan. 2013. [Online]. Available: http://arxiv.org/abs/1301.7002.

[12] J. Pfeiffer and X. Wu, "Automated time delay estimation for distributed sensor systems of electric vehicles," in *Proc. 10th IEEE Int. Conf. Intell. Data Acquisition Adv. Comput. Syst. Technol. Appl. (IDAACS)*, Metz, France, Sep. 2019, pp. 609–614. [Online]. Available: https://ieeexplore.ieee.org/document/8924330/

AHMED AYADI received the B.Sc. degree in electrical engineering and information technology and the M.Sc. degree in management from the Technical University of Munich, Munich, Germany, in 2016 and 2018, respectively, where he is currently pursuing the M.Sc. degree in electrical engineering and information technology. His current focus is on machine learning methods and theory.

JAKOB PFEIFFER received the B.Sc. degree in informatics and the M.Sc. degree in robotics, cognition, intelligence from the Technical University of Munich, Munich, Germany, where he is currently pursuing the Ph.D. degree with the Chair for Data Processing, Department of Electrical and Computer Engineering and participating in the ProMotion program of the BMW Group, Munich. In his current work in the field of machine learning, he is contributing to minimizing the differences between measured and real values in the power trains of EVs.