Calibration of the Nonlinear Wheel Odometry Model with an Improved Genetic Algorithm Architecture

Máté Fazekas, Balázs Németh and Péter Gáspár
Institute for Computer Science and Control, SZTAKI-ELKH, Kende u. 13-17., Budapest, Hungary

Keywords: Parameter Estimation, Nonlinear Model, Genetic Algorithm.

Abstract: To guarantee the required motion estimation accuracy for an autonomous vehicle, the integration of the wheel encoder measurements is an adequate choice besides the generally applied GNSS, inertial and visual-odometry methods. Wheel odometry is a robust and cost-effective technique, but the required calibration of the nonlinear odometry model in the presence of noise remains an open problem in the context of autonomous vehicles. The core problem is that due to the nonlinear behavior of the model, the identified parameters will be biased even with Gaussian-type measurement noises. The presented method operates with genetic algorithms and utilizes two novel improvements: compensation of the state initialization of the model inside the estimation process, and equilibration of the parameter estimation by an adaptive weighting technique. With these innovations the distortion effects are mitigated and unbiased model calibration can be obtained even when several local minimums exist. The performance of the developed algorithm and the accuracy of parameter estimation are demonstrated with detailed validation and test with a real vehicle.

1 INTRODUCTION

The state estimation has a critical responsibility in the self-driving software because the trajectory planning and motion control are based on its results. The aim is to determine the motion signals, such as velocities and pose (position and orientation) as accurately as possible, and the robust estimation is also a required capacity in parallel. However, cost-efficiency is important in the automotive industry as well, thus the applied automotive-grade type sensors are from the low-cost ones. The disadvantages of the GNSS (Global Navigation Satellite System), IMU (Inertial Measurement Unit), or vision-based methods can be improved with the integration of the wheel encoder measurements (Funk et al., 2017). Nevertheless, the model containing the encoder measurements suffers from parameter uncertainty. Therefore, this paper focuses on the calibration of the odometry model which is equivalent to the parameter estimation of a nonlinear dynamic system. This type of optimization task behind the calibration process is not solved yet in general, see e.g. (Schoukens and Ljung, 2019), and the problem is more difficult if noisy measurements are applied, such as IMU, GNSS, and wheel-encoder signals.

Most of the related works in the field of odometry calibration deal with mobile robots. A widely applied method is the Augmented Kalman-filter (Martinelli and Siegwart, 2006; Brunker et al., 2017), in which the parameters are estimated simultaneously with the state filtering. Although the solution is simple, observability issue, convergence difficulty, and unstable behavior can appear, see (Antonelli and Chiaverini, 2007), (Censi et al., 2013).

The other way is to estimate the parameters in a regression task. However, due to the nonlinear model, the optimization leads to a non-convex problem that is difficult to solve. Separation, or double linearization are applied in some works (Antonelli et al., 2005; Censi et al., 2013; Seegmiller et al., 2013), to simplify the nonlinear calibration task. Nevertheless, these can be only applied with simple and separable models, but for the proper modeling of the behavior of a real-sized car an improved odometry model is necessary (Fazekas et al., 2020).

From the identification scope issues, which even not appear at all in the case of linear system identification (Ljung, 1987), are revealed from the theoretical point of view (Schoukens and Ljung, 2019). Since the model is nonlinear, linearization and a recursive estimation are necessary (Tangirala, 2015). In this case, the initial states of the dynamic model have to be initialized with measurements containing noise, which introduces a new problem. The noise is affected by

Fazekas, M., Németh, B. and Gáspár, P.
Calibration of the Nonlinear Wheel Odometry Model with an Improved Genetic Algorithm Architecture.
DOI: 10.5220/0011275700003271
In *Proceedings of the 19th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2022)*, pages 640-648
ISBN: 978-989-758-585-2; ISSN: 2184-2809

the nonlinearity, thus its effect on the measured output is no longer familiar with the Gaussian framework, which is applied in almost all methods, such as Kalman-filtering, least squares etc (Ljung, 2010). Thus, the parameter estimation will be certainly biased. Other methods, such as genetic algorithm, that apply random sampling and heuristic based improvements for the convergence instead the gradient, can be used as well, but the distortion effect due to the initialization noise also appears.

The available measurements for the model calibration and the compensation of the noises are always crucial questions. In several papers, a restrictive requirement is applied, e.g. special pre-defined paths for the vehicle (Jung et al., 2016), unbiased estimation of the initial pose (Antonelli et al., 2005), or measurements with expensive sensors (Censi et al., 2013). To the best of our knowledge, (Maye et al., 2016) is the only paper that deals with the distortion effect of the noises in the odometry parameter estimation, but with a special focus on the observability. If more measurements are available, the optimization can be performed in a batch mode to reduce the impact of the initialization noise, however, a weighting is recommended (Seegmiller et al., 2013).

This work focuses on the compensation of the initial state noises, and the determination of the relative weights between segments forming a batch. Measurements only from cost-effective automotive-grade type sensors, such as GNSS, IMU, compass and wheel encoders are utilized, and the signals are saved on real streets. In the proposed algorithm the initial states are handled as parameters, and the estimation is performed with genetic algorithms in a complex architecture. The main contribution is that the proposed method transforms the fully explored parameter space by the genetic algorithm into an adaptive weighting procedure. The efficiency of the proposed algorithm is validated with experimental tests of a real-sized vehicle, which demonstrates that the mentioned issues of the noises can be mitigated, and the estimation accuracy can be significantly improved.

The remainder of the paper is organized as follows. In Section 2, the applied odometry model is presented which includes dynamic effects in order to ensure a proper model for the calibration. Next, the Section 3 introduces the issues of the nonlinear model calibration and summarizes the estimation with a genetic algorithm. The proposed architecture for the calibration can be found in Section 4. The measurement scenarios applied for the estimation are presented in Section 5.1. The validity of our approach is demonstrated via vehicle test experiments in Section 5, and finally, the paper is concluded in Section 6.

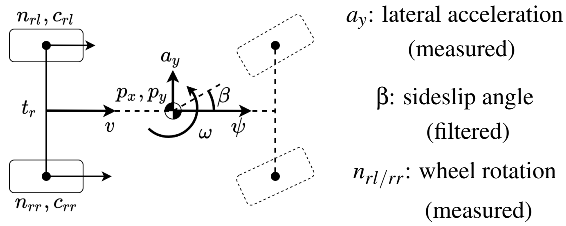

Figure 1: Odometry model.

2 VEHICLE MODEL

The navigation with wheel odometry is based on a model, where the state vector x_t contains the pose, the longitudinal and lateral positions of the center of gravity $p_{x,t}, p_{y,t}$, and the ψ_t orientation of the vehicle. The change of the pose is based on the longitudinal v_t and angular ω_t velocities, thus the planar motion of the vehicle in t discrete time steps is calculated as,

$$\begin{bmatrix} p_{x,t+1} \\ p_{y,t+1} \\ \psi_{t+1} \end{bmatrix} = \begin{bmatrix} p_{x,t} + v_t \cdot T_s \cdot \cos(\psi_t + \omega_t/2 \cdot T_s + \beta_t) \\ p_{y,t} + v_t \cdot T_s \cdot \sin(\psi_t + \omega_t/2 \cdot T_s + \beta_t) \\ \psi_t + \omega_t \cdot T_s \end{bmatrix}, \tag{1}$$

where β_t is the filtered sideslip angle. The velocities are computed utilizing the n wheel rotations,

$$v_t = (n_{rl,t} \cdot c_{rl,t} + n_{rr,t} \cdot c_{rr,t})/2, \tag{2a}$$

$$\omega_t = (n_{rr,t} \cdot c_{rr,t} - n_{rl,t} \cdot c_{rl,t})/t_r, \tag{2b}$$

where $T_s = 0.025\ s$ is the sampling time, $c_{rl,t}, c_{rr,t}$ are the actual wheel circumferences, and t_r is the rear track width. In the literature, the slight change of the wheel radius due to the effect of vertical dynamic is generally neglected, because the odometry-based localization is widely used in low-speed circumstances, e.g., automated parking. However, our method is developed for general driving, where the dynamics is significant (Fazekas et al., 2020). Therefore, the slight change due to the vertical load transfer is considered,

$$c_{rl,t} = c_e + c_d/2 + d \cdot a_{y,t}, \tag{3a}$$

$$c_{rr,t} = c_e - c_d/2 - d \cdot a_{y,t}, \tag{3b}$$

where the c_e is the effective wheel circumference, c_d is the difference between the effective values, $a_{y,t}$ is the lateral acceleration measured by the IMU, and the dynamic component d takes into account the effect of vertical dynamics.

In the calibration process, every state variable is measured, thus the system model is,

$$x_{t+1} = f(x_t, u_t, \theta), \quad x_t = [p_{x,t}, p_{y,t}, \psi_t]^T, \quad y_t = x_t, \tag{4a}$$

$$u_t = [n_{rl,t}, n_{rr,t}, \beta_t, a_{y,t}]^T, \quad \theta = [c_e, c_d, t_r, d], \tag{4b}$$

where $f(\cdot)$ contains (1), and θ is the parameter vector. The values are illustrated in the paper with the units of $m, mm, m, mm \cdot s^2/m$ for the c_e, c_d, t_r, d parameters, respectively.

The nominal values of c_e and t_r can be found in the vehicle datasheet, but with these values the position error is already in the range of 10 m after a few hundred meters. Moreover, the c_d and d parameters are unknown, thus the calibration is essential.

3 CALIBRATION OF NONLINEAR MODELS

3.1 General Methods

The calibration problem of a physical model contains the estimation of system parameters Θ, which leads to an optimization task. Generally, this is formulated as a minimization problem of the squared errors of the output, such as

$$\hat{\Theta}_{opt} = arg\,\underset{\Theta}{min}\, V_K(\Theta) = arg\,\underset{\Theta}{min} \sum_{k=1}^{K} ||\widetilde{y}_k - \hat{y}_k(\Theta)||^2, \tag{5}$$

where the $\hat{y}_k(\Theta)$ is the predictor of the model,

$$\hat{y}_k(\Theta) = h(x_k) \quad x_{k+1} = f(x_k, \widetilde{u}_k, \Theta), \tag{6}$$

and $\widetilde{u}_k - \widetilde{y}_k$ are measured input-output signals. The methods to solve (5) can be grouped according to the usage of gradient techniques.

In the one group, the optimization is solved with the well-known least squares (LS) method which applies the gradient directly. However, in the case of nonlinear models, the dynamic system can not be ordered into the basic least squares form as $\hat{y}_k(\Theta) = \phi_k^T(\widetilde{u}_k)\Theta$, see (Ljung, 1994). Therefore numerical search is required (Tangirala, 2015), where the Gauss-Newton approach is an adequate choice. In this case, the nonlinear least squares problem is handled with the Taylor-approximation of the predictor forming a locally linear estimation and an iterative solution.

The other group contains methods that do not use any gradient to solve the minimization, e.g., particle filter or genetic algorithm. Instead, random sampling of the parameter space, evaluation with the measured input-output signals, and heuristic-based improvements of the possible solutions are applied to reach the optimum.

3.2 Core Problem of Nonlinear Dynamic Model Calibration

Regardless of the applied methods, the calibration of a nonlinear dynamic model has a core problem. The outputs of system model (6) have to be computed with the actual parameter values which requires the initialization of the states x_0 at the beginning of the estimation window. In the odometry model, the output is equal with the state ($y_k = x_k$), thus the measured $\widetilde{y}_{k=0}$ can be utilized for the initialization,

$$\begin{bmatrix} \hat{y}_1(\Theta) \\ \vdots \\ \hat{y}_K(\Theta) \end{bmatrix} = \begin{bmatrix} x_1 \\ \vdots \\ x_K \end{bmatrix} = \begin{bmatrix} f(x_0, \widetilde{u}_0, \Theta) \\ \vdots \\ f(x_{K-1}, \widetilde{u}_{K-1}, \Theta) \end{bmatrix} \mid x_0 = \widetilde{y}_0. \tag{7}$$

Due to the appeared noise on it, the integrated model diverges from the correct path even with the true parameter values. Thus, without proper initialization the parameter estimation is biased, because the optimum of (5) differs from the true value of Θ.

3.3 Estimation in Batch Formulation

The distortion effect of state initialization can be handled, if more K long measurement segments are applied at once, such as

$$\hat{\Theta}_{opt} = arg\,\underset{\Theta}{min} \sum_{n=1}^{N} \sum_{k=1}^{K} ||\widetilde{y}_{n,k} - \hat{y}_{n,k}(\Theta)||^2, \tag{8}$$

where N denotes the batch size. For example, in the case of Gauss-Newton method the Jacobian matrices of the segments are stacked into a huge matrix and the parameters are identified in the same iterative way, or in the sampling based methods the actual parameter values are evaluated for all of the N segments.

Since the model fitting is performed to every segment simultaneously, the bias resulted by the uncertain state initialization is mitigated. The method can be interpreted as a cross-validation technique inside the estimation process, but it will be illustrated that the optimum of the batch is not necessarily identical with the true value of the parameters. Furthermore, a new issue arises with the batch formulation. The segments are differs in shape of the path, velocity, or the appearing noise of the initialization etc. Therefore, the segment with the highest error has more influence on the optimum, which is unfavourable. Thus, relative weighting between the segments are requested to be introduced to equilibrate the optimization to obtain the true value of the parameters.

3.4 Initial State as Parameter

The distortion effect of the inaccurate state initialization can be eliminated, if the poses at the beginning of the segments are estimated as well (Fazekas et al., 2021). In our algorithm the initial state, when the model outputs are calculated in (7), is included as a parameter to be estimated, such as

$$x_1 = f(\vartheta, \widetilde{u}_0, \Theta), \quad \vartheta = x_0 = [p_{x,0}, p_{y,0}, \psi_0]^T. \quad (9)$$

In the paper, the θ contains the vehicle parameters (4b), ϑ is the state initialization parameters (9), and Θ denotes the parameters to be estimated, specifying at the actual method how to include θ and ϑ.

3.5 Parameter Estimation with Genetic Algorithms

When the optimization problem of (5) has several local optimums, the genetic algorithm (GA) is a proper choice, because this procedure explores the entire parameter space. The estimation process is illustrated in Figure 2. The method starts with the initialization of E pieces of entities (model settings) by random sampling, but only from a given range around the nominal values of the estimated parameters. Next, the population is tested, where the outputs of the model parameterized with the entities are calculated. The entities are ranked by the loss, and if the stopping condition does not occur, a new population is generated. The main idea of the GA is to apply genetic operators or functions inspired by the process of natural selection to obtain convergence to the global optimum.

We use the implemented version of GA in MATLAB (Mathworks, 2021a), which is based on (Goldberg, 1989). The *Selection* function chooses the entities as parents to form the new generation of the population. The mathematical method is the stochastic universal sampling (Baker, 1987). These are paired ($\widehat{\Theta}_a$, $\widehat{\Theta}_b$), and the new entities are calculated with the *Crossover* function, such as

$$\widehat{\Theta}_{new} = \widehat{\Theta}_\alpha + \xi \cdot (\widehat{\Theta}_\beta - \widehat{\Theta}_\alpha), \quad (10)$$

where ξ is a vector containing random numbers from the range $[0,1]$. The *Mutation* function enables the GA to search further away from the actual entities. With a low probability (5%), when the crossing is executed a random vector is added to the ξ, which helps to avoid sticking into a local optimum. The final step of the formulation of the new population is the *Reinsertion*. The stable convergence is ensured with the so-called "elite strategy", where the best entities (5% in our case) in the current generation are guaranteed to survive to the next generation. The other entities are chosen based on the loss values, the best 95% of the entities from the current population and the newly generated $0.8E$ ones with the crossover and mutation are collected. This iterative evolution of the population continues until the stopping condition, e.g. maximum generation or lower improvement of the best entity than a given limit, is reached.

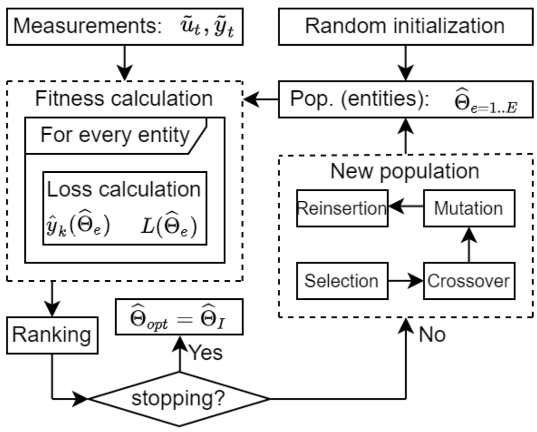

Figure 2: Process of the GA based estimation.

4 PROPOSED ARCHITECTURE

4.1 Parameter Estimation with Initialization Compensation

The θ vehicle parameters are identified in batch mode with N segments, where the initial state of every segment are also estimated. Since a ϑ_n initialization value is independent from the other segments in the batch, several local minimums are exist. Therefore, in our method, the GA is applied for the optimization, because this procedure uses random sampling, making it a suitable methodology for solving problems with several local minimums.

The parameter vector and the cost function is the following,

$$\widehat{\Theta} = [\widehat{\theta}, \widehat{\vartheta}_1, ...\widehat{\vartheta}_N]^T, \quad (11a)$$

$$\widehat{\Theta}_{opt} = \underset{\widehat{\Theta}}{arg\ min} \sum_{n=1}^{N} \underbrace{\sum_{k=1}^{K} l(\widetilde{y}_{n,k}, \widehat{y}_{n,k}(\widehat{\theta}, \widehat{\vartheta}_n), W_n)}_{L_n}, \quad (11b)$$

$$l(\cdot) = \left[\widetilde{y}_{n,k} - \widehat{y}_{n,k}(\widehat{\theta}, \widehat{\vartheta}_n)\right]^T W_n \left[\widetilde{y}_{n,k} - \widehat{y}_{n,k}(\widehat{\theta}, \widehat{\vartheta}_n)\right], \quad (11c)$$

where L_n is the loss of a segment, and W_n is a weight matrix which is formulated, such as

$$W_n = w_{r,n} \cdot diag([w_{p,x}, w_{p,y}, w_\psi]^T_{1x3}). \quad (12a)$$

The $w_{p,x}, w_{p,y}, w_\psi$ are the weights of the system equations (4) to compensate the different magnitude of the position error in meter and orientation error in radian, and thus these are the same for every segment in the batch. The values are determined with an experimental tuning, the optimal setting is resulted as

$$[w_{p,x}, w_{p,y}, w_\psi] = [1, 1, 40], \quad (13)$$

because only the ratio of the weights matters. In contrast, the $w_{r,n}$ is the relative weight between the segments to prevent that some segments has higher influence than others on the estimated vehicle parameters.

4.2 Architecture with Relative Weighting

Our proposed method integrates a cross-validation based knowledge to determine the proper relative weight of the segments. The main idea is that a segment whose optimal model setting (which minimizes the L_n individual segment error) has a high error on the other segment of the batch should not have significant weight in the parameter identification. This idea is implemented in the architecture presented in Figure 3, where there are 3 steps to determine the proper $w_{r,n}$ weights, before the model is calibrated with the presented method in the previous section.

Step 1: In the first step, GA based estimation is performed on the segments separately (raw mode). In this step, $i_{max} = 7$ generations are applied with a population size of $E = 2000$. Because most of the odometry parameters are geometry ones, the search space can be constrained with the following lower- and upper bounds

$$\theta_{lb} = [1.93, -5, 1.4, -1]^T,\ \theta_{ub} = [1.97, 5, 1.7, 3]^T. \tag{14}$$

The initial state parameters are also bounded around the corresponding $\widetilde{y}_{n,k=0}$ filtered pose value

$$\vartheta_{lb} = \widetilde{y}_{n,k=0} - [3, 3, 0.2]^T,\ \vartheta_{ub} = \widetilde{y}_{n,k=0} + [3, 3, 0.2]^T. \tag{15}$$

Since the entire parameter space is explored during the convergence, k-means clustering is performed (Mathworks, 2021b). The parameter space of every segment is represented with $C = 200$ clusters ($\widehat{\theta}_{n,1..200}$ sorted by the L_n loss) formed from the last 5 generations of entities.

Step 2: In the second step, the clustered models are tested on the other segments of the batch, and the average error is computed,

$$\overline{L}_{n,1..200} = \frac{1}{N-1} \sum_{m=1}^{N} L_m(\widehat{\theta}_{n,1..200}, \widehat{\vartheta}_m), \quad (m \neq n) \tag{16}$$

where $L_m(\widehat{\theta}_{n,c}, \widehat{\vartheta}_m)$ is the loss on the m segment with the $c-th$ clustered model of the n segment.

Step 3: The mentioned cross-validation based knowledge is transformed to $w_{r,n}$ relative weights in this step. Straight line is fitted to the $\overline{L}_{n,1..200}$ errors of the segments, and the weights are calculated in the following way,

$$w_{r,n} = 1 + 2 \cdot \frac{p_{1,n} - \underline{p}}{\overline{p} - \underline{p}}, \tag{17a}$$

$$\underline{p} = min(p_{1,n=1..N})), \quad \overline{p} = max(p_{1,n=1..N})), \tag{17b}$$

where $p_{1,n}$ is the steepness of the fitted straight of segment n. This scheme determines the $w_{r,n}$ relative weights between 1 and 3, and means that the weight of a segment, whose individual optimal settings (which have increasing error on the actual segment) have decreasing error on the other ones, will be low. In contrast, the segments, whose sorted clustered models have similarly increasing error on the other segments, will get higher weights. Thus, the impact of segments that results in possible incorrect calibration can be mitigated, and in parallel, the estimation bias should be lower.

Step 4: Finally, the odometry model is calibrated with the GA in batch mode utilizing the pre-calculated $w_{r,n}$ weights. The bounds for the θ and ϑ parameters are the same as in Step 1, but since in the batch mode the number of parameters is 34 (4 vehicle, and $N = 10$ times 3 initial state parameters), $i_{max} = 10$ generations with a population size of $E = 10000$ is utilized to reach the global optimum.

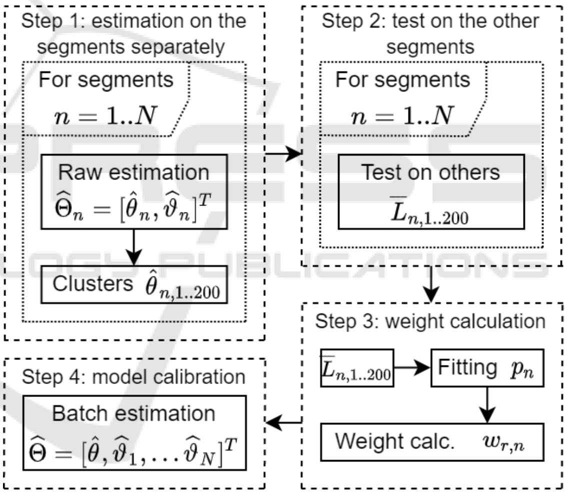

Figure 3: Calibration architecture of the proposed method.

5 EXPERIMENTAL RESULTS

5.1 Test Measurement Scenario

The test vehicle is a Nissan Leaf electric car which is equipped with automotive-grade GNSS, compass, and IMU sensors, and from the vehicle CAN bus the wheel encoder signals are also saved. The sampling frequency is 40 Hz. The test track is a 24 km long route in suburb and city driving under normal driving conditions. The track contains various bends, two roundabouts, and lots of crossroads.

The signals of the GNSS, compass, and IMU sensor are utilized for the model calibration. The pose can be measured directly with the first two, although these signals are assumed to be noisy but unbiased. In contrast, the pose computation from the acceleration and angular velocity measurements with the IMU is generally biased, but the noise is lower. The filtering problem is well-explored, the implemented method is similar to (Caron et al., 2006) to obtain reference output measurements ($\widetilde{y}_t$) for the calibration. The sideslip is also estimated with an IMU-based method (Bevly et al., 2006) in the bends, which are detected using the vehicle trajectory.

The 24 km long route is divided into 300 m long segments ($K = 1350$) with step size of 2.5 s, which results in 2037 segments. Since the c_d, t_r and d parameters can be observed properly only with the yaw rate equations (2b), the 1127 segments with higher absolute angular velocity than 0.15 rad/s are selected for the calibration.

5.2 Validation Process and Error

The true value of the $\theta = [c_e, c_d, t_r, d]$ parameters are unknown, thus the model calibration is validated with the position error of the resulted model. In order to avoid overfitting, the segments are regenerated with different initial points. Furthermore, in the validation process, all of the segments are applied regardless of the angular velocity is lower than the limit. The position error of a calibrated model containing $\widehat{\theta}$ is calculated for every segment s as,

$$E_{p,s} = \sum_{k=1}^{K} \sqrt{(\widetilde{p}_{x,k} - p_{x,k}(\widehat{\theta}))^2 + (\widetilde{p}_{y,k} - p_{y,k}(\widehat{\theta}))^2}, \tag{18}$$

and the $\overline{E_p}$ average of these are applied as a validation error to evaluate the calibration.

Take into account that the minimum validation error is not zero, because the states of the odometry model (4) at the beginning of the segment are initialized with the filtered pose values in the validation process. To find the reachable minimum error, a GA-based search has been also performed, where only the θ vehicle parameters have been estimated. All of the validation segments have been applied, thus it can be an offline validation method only, but we can determine that the minimum position error in this validation case is 2.22 m.

5.3 Calibration with the Method

In this section, the calibration for a given batch with the proposed method is illustrated step-by-step. The detailed results demonstrate the necessity of both improvements as well, i.e., the integration of initial state estimation and the relative weighting.

5.3.1 Segment Representation with Clusters

In Step 1, GA estimations are performed for every segment individually. The entities of the last 5 generation of the $n = 1$ segment of the batch can be found in Figure 4. Since the loss of the models are between $0.15 - 2.79$, the parameters are in a wide hypercube, e.g. the range of the t_r track widths is $1.43 - 1.62$ m. The entities of the segment are represented with 199

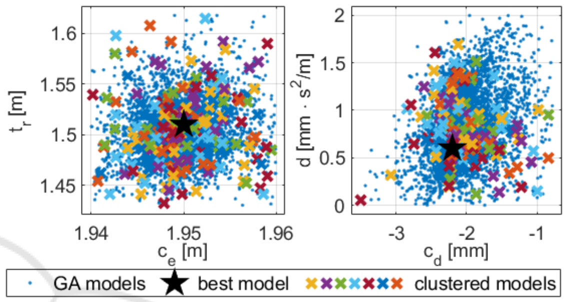

Figure 4: Entities and clustered models of a segment.

clusters, which are sorted by the loss, and the best entity is also used as the first "clustered model". In Step 2, the clustered models are tested on the other $N-1$ segments and the $\overline{L}_{n,1..200}$ average errors (16) are computed. In Figure 5, the loss of the clustered models on

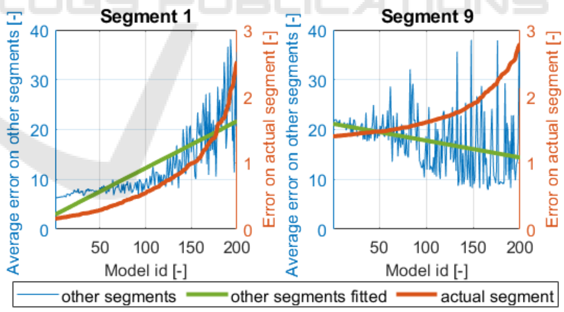

Figure 5: Errors of two segments from the batch.

the actual segment, and the average error (loss by testing on others) are presented for two segments of the batch.

5.3.2 Weight Calculation

In Step 3, the $w_{r,n}$ weights are determined. The fitted straight lines to the $\overline{L}_{n,1..200}$ average errors are illustrated in Figure 5 as well. For segment 1, the models with higher errors in the actual segment also have higher errors in the other segments. In contrast, in segment 9 it is the opposite. Thus, this segment would

Table 1: Relative weight of the segments in the batch.

Seg.	1	2	3	4	5
$p_{1,n}$ $[10^{-2}]$	9.41	7.55	3.65	-2.00	6.19
$w_{r,n}$	3.00	2.71	2.10	1.22	2.50
Seg.	6	7	8	9	10
$p_{1,n}$ $[10^{-2}]$	6.94	-2.15	0.85	-3.40	-1.38
$w_{r,n}$	2.61	1.12	1.66	1.00	1.31

Table 2: Data of the best 2000 entities.

	c_e	c_d	t_r	d	L	$\overline{E_p}$
Min	1.9495	-2.20	1.510	0.88	0.747	2.29
Mean	1.9503	-2.08	1.517	0.93	0.758	2.37
Max	1.9513	-1.94	1.525	1.00	0.766	2.60
$\widehat{\theta}_a$	1.9501	-1.94	1.518	0.88	0.758	2.58
$\widehat{\theta}_b$	1.9502	-2.20	1.513	1.00	0.756	2.33

distort a good optimum of the batch, since the goal is the minimization of the error.

The determined relative weights of the segments with the introduced rule (17) are summarized in Table 1. These are difficult to evaluate alone, but can be done with the examination of the individual optimal parameters of the segments (the first "clusters" of the 200) in Figure 7.

The segments $4, 7, 9, 10$ have the undesirable negative steepness, and the optimal parameters of these segments are the most different from the batch estimated ones, especially for the c_d circumference difference (Figure 7).

5.3.3 Parameter Estimation with Initialization Compensation

In Step 4, the vehicle parameters are estimated in batch mode in parallel with the initial states ($\Theta = [\theta, \vartheta_{1..N}]$) minimizing the loss function (11). It has been mentioned that by integrating the initial states as parameters, the estimation in batch mode is more complicated because a lot of local optimums exist as the ϑ values could compensate each other.

This is demonstrated with the best 2000 entities of the final generation of the GA based estimation. The loss of each of these entities are between $0.747 - 0.766$ which is a very narrow range. The θ vehicle parameters are in a hypercube with limits that can be found in the first three rows of Table 2. This seems to be a narrow parameter space, but the $\overline{E}_p$ validation error of each of these models are in a range of $2.29 - 2.60$ m, which is a significant deviation with respect to the minimum error of 2.22 m. This illustrates the high parameter sensitivity of the odometry model as well.

The reason is, although the loss is similar, the $\widehat{\vartheta}_n$ values of the entities are spread over a wide range resulting in several local minimums. The estimated state initialization parameters (the Δ deviation from the measured $\widetilde{y}_k$ for better view) of the 2000 entities are shown in Figure 6. The average of the differences between the maximum and minimum of the $\Delta p_{x/y}$ positions, and $\Delta\psi$ orientation deviations of the $N = 10$ segments are 1.31 m, 1.28 m and 1.25°, respectively. This illustrates that it is possible to reach the same loss with a wide range of estimated $\widehat{\vartheta}_n$ initial states, due to the mentioned compensation property of the initialization values. The averages of the standard deviations of the segments are also significant with 9.8, and 10.2 cm for the $\Delta p_{x/y}$ values.

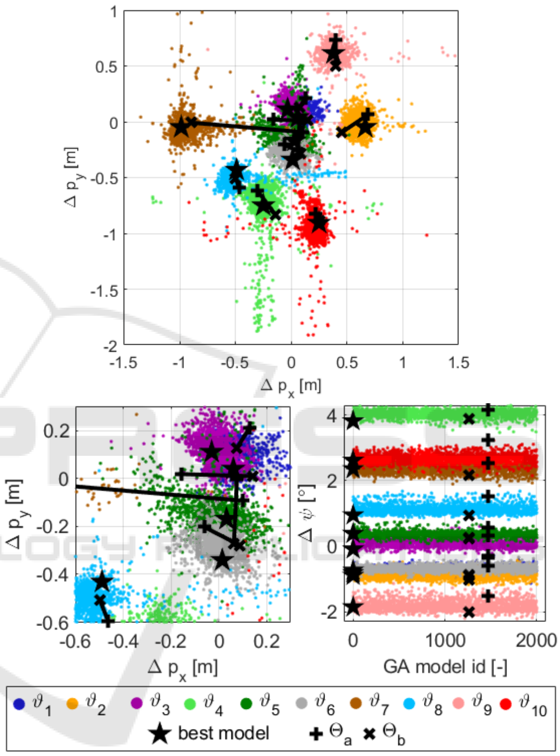

Figure 6: Estimated initial state parameters of the entities.

For example, two entities that have almost the same loss could have completely different estimated initialization, as it is illustrated in Figure 6 with Θ_a and Θ_b. Moreover, the corresponding θ vehicle parameters can lie on the other borders of the parameter hypercube formed by the 2000 entities as it is shown in the last rows of Table 2. Thus, the validation error of the two resulting models are differ significantly (2.58 m vs 2.33 m), but the almost same loss makes it very difficult to infer this during the calibration. This example demonstrates the difficult mathematical optimization generated by the integration of the initial states as parameter and batch estimation mode. However, the genetic algorithm is able to handle this optimization with many local minimums if a sufficiently

Table 3: Model calibrations of the batch with the methods.

Case	c_e	c_d	t_r	d	$\overline{E_p}$
Mean of raws	1.948	-2.49	1.530	0.85	2.78
Batch	1.949	-2.40	1.519	0.07	3.97
Batch weight.	1.949	-2.27	1.525	0.07	3.50
Batch init	1.951	-2.35	1.525	1.02	2.59
Proposed	1.950	-2.13	1.523	0.87	2.29

large population is applied to explore the entire parameter space.

5.3.4 Calibration Results

In Table 3, the estimated parameters with the proposed method are presented, and calibration results in 4 other cases are illustrated as well: the mean of the results of the 10 segment with GA estimations separately (mean of raws), identified parameters in batch mode without any improvement (batch), with the relative weighting structure in batch mode without initialization estimation (batch weighting), and with the estimation including the initialization parameters but without relative weighting (batch init).

It is an interesting result that the model with the mean of the individual (raw) GA estimations has a lower validation error, than the batch calibrated one. The higher error of the batch model is the consequence of the widely different individual loss of the segments, e.g. the loss of segment 4 is 12, while the segments $1,3,5,6$ have lower than 1. Therefore, the estimated optimal parameters of the batch have to be much closer to the individual optimum of the segment with high loss. This is why the relative weighting technique has been introduced.

The relative weighting technique can compensate for this distortion a bit (third row), and when the initial parameters are taken into account this unfavorable distortion effect is greatly reduced (fourth row). However, it is impossible to decide beforehand whether a segment with high loss is completely wrong or only the state initialization is highly inappropriate. Therefore, the integration of both improvements is necessary, and as we can see the validation error is further decreased with the proposed method, which indicates the unbiased calibration.

5.4 Results of Other Batches

In the previous section, the estimation process with the proposed method is presented in detail on a given batch. The robustness of the method is tested with the formulation of 200 different batch from the 1127 segments, and the proposed calibration is performed one by one. Only the validation errors of the batch calibrations are presented with box-plots in Figure 8.

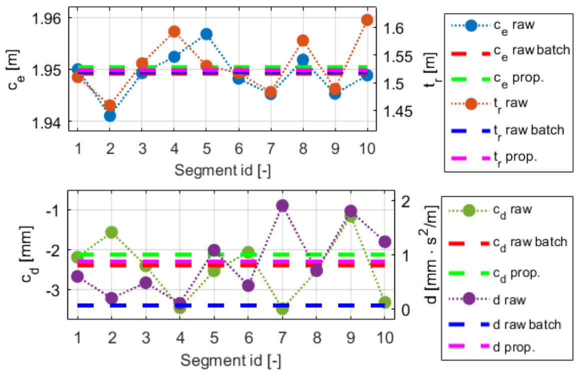

Figure 7: Estimated parameters with the method.

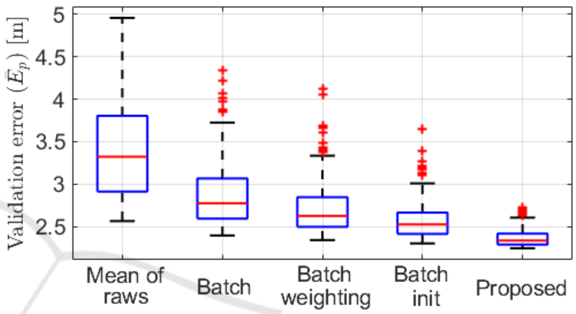

Figure 8: Validation errors of several batches.

The tests illustrate that with the batch formulation the distortion effects can be mitigated, the median of the validation errors decreases to 2.77 m, from the 3.32 m which can be reached with the mean of the individual GA estimations of the segments. The standard deviation is also lower with 20 cm from the 56 cm with the mean models, but the parameter estimation remains significantly biased. With the application of the relative weighting method and with the integration of the initial states as a parameter, the errors and their standard deviations are further decreased, but not substantially. However, with the proposed method when both improvements are applied, the errors are in a range of $2.24-2.70$ m, the median error is only 2.34 m, and the 75th percentile model has 2.42 m validation error. The standard deviation of the errors is relatively low with 11 cm.

6 CONCLUSION

In this paper, a novel algorithm was presented for the calibration task of the nonlinear wheel odometry model of an autonomous vehicle. The proposed method introduces two innovations, the integration of the initial state of the measurement segments as a parameter to be estimated, and a relative weighting in the batch estimation mode. The method was validated by experimental tests with a real vehicle. The main contribution is that when both improvements are applied, the calibration accuracy is improved.

Finally, the authors consider that with the development of a more complex weighting, the bias-free calibration of every batch can be obtained. In the future we would like to integrate a learning-based weighting, however, the generation of training data is complicated, since it is an open question what would be the optimal weights that result in the true value of the parameters.

ACKNOWLEDGEMENTS

The research was supported by the Ministry of Innovation and Technology NRDI Office within the framework of the Autonomous Systems National Laboratory Program. The paper was partially funded by the National Research, Development and Innovation Office (NKFIH) under OTKA Grant Agreement No. K 135512. The work of Máté Fazekas was supported by the ÚNKP-21-3 New National Excellence Program of the Ministry for Innovation and Technology from the source of the National Research, Development and Innovation Fund.

REFERENCES

Antonelli, G. and Chiaverini, S. (2007). Linear estimation of the physical odometric parameters for differential-drive mobile robots. *Autonomous Robots*, 23:59–68.

Antonelli, G., Chiaverini, S., and Fusco, G. (2005). A calibration method for odometry of mobile robots based on the least-squares technique: theory and experimental validation. *IEEE Transactions on Robotics*, 21(5):994–1004.

Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. In *2nd International Conference on Genetic Algorithms*, volume 206, pages 14–21.

Bevly, D. M., Ryu, J., and Gerdes, J. C. (2006). Integrating ins sensors with gps measurements for continuous estimation of vehicle sideslip. *IEEE Transactions on Intelligent Transportation Systems*, 7(4):483–493.

Brunker, A., Wohlgemuth, T., Frey, M., and Gauterin, F. (2017). GNSS-shortages-resistant and self-adaptive rear axle kinematic parameter estimator (SA-RAKPE). In *28th IEEE Intelligent Vehicles Symposium*.

Caron, F., Duflos, E., Pomorski, D., and Vanheeghe, P. (2006). GPS/IMU data fusion using multisensor Kalman-filtering: Introduction of contextual aspects. *Information Fusion*, 7(2):221–230.

Censi, A., Franchi, A., Marchionni, L., and Oriolo, G. (2013). Simultaneous calibration of odometry and sensor parameters for mobile robots. *IEEE Transactions on Robotics*, 29(2):475–492.

Fazekas, M., Gáspár, P., and Németh, B. (2021). Odometry Model Calibration for Self-Driving Vehicles with Noise Correction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Fazekas, M., Németh, B., Gáspár, P., and Sename, O. (2020). Vehicle odometry model identification considering dynamic load transfers. In *28th Mediterranean Conference on Control and Automation (MED)*, pages 19–24.

Funk, N., Alatur, N., and Deuber, R. (2017). https://arxiv.org/abs/1711.00548Autonomous Electric Race Car Design. In *International Electric Vehicle Symposium*.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley.

Jung, D., Seong, J., bae Moon, C., Jin, J., , and Chung, W. (2016). Accurate calibration of systematic errors for car-like mobile robots using experimental orientation errors. *International Journal of Precision Engineering and Manufacturing*, 17(9):1113–1119.

Ljung, L. (1987). *System Identification:Theory for the User*. PTR Prentice Hall.

Ljung, L. (1994). *Modeling of Dynamic Systems*. PTR Prentice Hall.

Ljung, L. (2010). Perspectives on system identification. *Annual Reviews in Control*, 34(1):1–12.

Martinelli, A. and Siegwart, R. (2006). Observability properties and optimal trajectories for on-line odometry self-calibration. In *IEEE Conference on Decision and Control*, pages 3065–3070.

Mathworks (2021a). Matlab global optimization toolbox: Genetic algorithm. https://www.mathworks.com/help/gads/genetic-algorithm.html.

Mathworks (2021b). Matlab statistics and machine learning toolbox: k-means. https://www.mathworks.com/help/stats/kmeans.html.

Maye, J., Sommer, H., Agamennoni, G., Siegwart, R., and Furgale, P. (2016). Online self-calibration for robotic systems. *The International Journal of Robotics Research*, 35(4):357–380.

Schoukens, J. and Ljung, L. (2019). Nonlinear system identification: A user-oriented roadmap. *IEEE Control Systems Magazine*, 39(6):28–99.

Seegmiller, N., Rogers-Marcovitz, F., Miller, G., and Kelly, A. (2013). Vehicle model identification by integrated prediction error minimization. *The International Journal of Robotics Research*, 32(8).

Tangirala, A. K. (2015). *Principles of System Identification*. CRC.