Speed Estimation From Vibrations Using a Deep Learning CNN Approach

Rickard Karlsson and Gustaf Hendeby

Speed Estimation from Vibrations using a Deep Learning CNN Approach

Rickard Karlsson*[†] and Gustaf Hendeby[†]
*NIRA Dynamics AB, SE-583 30 Linköping, Sweden.
e-mail: rickard.karlsson@niradynamics.se
[†] Dept. of Electrical Engineering, Linköping University, Linköping, Sweden
e-mail: {rickard.g.karlsson,gustaf.hendeby}@liu.se

***Abstract*—A novel method for accurate speed estimation of a vehicle using a deep learning *convolutional neural network* (CNN), with accelerometer and gyroscope measurements as input, is presented. It does not suffer from the fundamental drift problem present in all dead reckoning methods, and yet yields about 2 m/s in accuracy. Efficient drift-free vehicle speed estimates are essential in many automotive applications, where internal wheel speed sensors or GPS are unavailable. Using extensive experimental data, the proposed CNN method is compared to an existing frequency analysis method. The proposed method is shown to perform significantly better, particularly during low speed and rapid speed changes where the frequency method struggles.**

***Index Terms*—Inertial measurements, Speed estimation, Deep learning, Convolutional neural network, Vibration analysis.**

I. Introduction and Background

Vehicle speed estimation from noisy sensor data has been an important application area for many years. The solutions can essentially be divided into two dominating approaches based on direct or indirect sensor measurements. Direct methods use measurements of quantities from which the speed can be directly derived. This implies using high level advanced sensors, *e.g.*, radar or GPS, or a *wheel speed sensor* (WSS). For an overview of possible sensors, see [1]. An important benefit of direct methods is that errors do not accumulation over time. Indirect methods often rely on dead-reckoning [2], where measurements of the acceleration are integrated to obtain the speed. Hence, sensor errors accumulate over time, causing the estimate to drift. For automotive applications, load changes and hill-driving add to the normal sensor noise, increasing the error. To mitigate the drift, additional supporting sensors are needed. One option is to analyze chassis vibrations which are embedded in the accelerometer measurements. These vibrations are partially caused by the rotating wheel axle, which is proportional to the vehicle's speed. Speed estimates from these vibrations can help mitigate the dead-reckoning drift.

The axis rotation harmonics visible in the frequency spectrum of the chassis vibrations can be extracted using frequency tracking methods. Generic frequency tracking in periodic signals in noise is a well studied topic, see [3] for an overview. Batch methods based on periodogram maximization, which are applicable when the frequency is almost constant over a period of time, can be found in [4, 5]. Frequency estimation with the purpose of finding the axis rotation speed and providing a virtual speedometer is studied in [6–9]. Difficulties arise during rapid speed changes, as a certain amount of data is needed to estimate the spectrum, and at low speeds, as there are many other sources of vibration (*e.g.*, related to the suspension and the road surface) in that that part of the spectrum.

In recent years, deep learning and *convolutional neural networks* (CNNs, [10–12]) have become very popular. There are numerous success stories in areas such as image classification and segmentation, speech processing, and more. The applications in signal processing are fewer, but a recent application is to analyze accelerometer data with a CNN to categorize human activity, [13, 14]. Estimating periodic signals using CNN has recently been used in ECG applications [15]. In [16], periodic lung motion is tracked utilizing a neural network. Given the success in computer vision, a commonly used technique when dealing with deep learning and signal processing is to treat a batch of data as an image and apply state-of-the-art vision methods such as CNN [17]. Imaged based learning for speed estimation, [18] and road monitoring from inertial sensors [19] are closely related applications and methods. Signal processing applications related to fault diagnosis of periodic phenomena are found in [20].

The main contribution in this paper is a CNN-based method to estimate vehicle speed from accelerometer data. Compared to existing frequency estimating methods, the proposed method extends the handled speed envelope to include low speeds and standstill as well as rapid speed changes. The design allows for standalone speed estimation utilizing, *e.g.*, a smartphone without connection to the vehicle. This could be beneficial in urban areas with multi-path GPS phenomena, in tunnels and other GPS challenged environments.

II. Deep Learning for Speed Estimation

The design of the deep learning method for speed estimation is detailed in three steps. First the available data is described, then the choice of network structure is discussed, followed by information about how to train and evaluate the solution.

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation and partly from CENIIT Project 17.2. Thanks to NIRA Dynamics for providing extensive measurement data.

A. Data Selection and Pre-Processing

Proper data selection and pre-processing is of key importance in any machine learning method. Here, measurements from three accelerometers (a_x, a_y, a_z) (longitudinal, lateral, and vertical, respectively), as well as the yaw rate, $\dot{\psi}$, are available. The accelerometer captures the vibrations related to the rotation of the wheel axle. The yaw rate provides information during cornering, since $a_y = v\dot{\psi}$, where v is the speed. Including yaw rate reduces the error in experiments, in particular when data is only available at a low sample rate.

Another critical design parameter is how much data to include in the batches fed to the net. This is a balance between including sufficient data about the phenomenon studied, and keeping the batch size down to minimize latency, being able to follow quick changes, and reduce computational complexity. Different batch sizes, N are considered. Depending on the data rate the same N will represent data from different time windows affecting the ability to track fast changes. For data rates considered here, $10\,\mathrm{Hz}$ and $100\,\mathrm{Hz}$, the batch sizes $N = 30$ $(3\,\mathrm{s})$ and $N = 250$ $(2.5\,\mathrm{s})$, respectively, was experimentally determined to strike a good balance between the objectives.

To summarize, the input to the net will be an $N \times M$ "image", where M is the number of sensor signals used. Partial overlapping between consecutive batches is employed to use the data effectively.

B. CNN Structure

How to select the proper network structure to solve a specific task is at large an unsolved problem, and craftsmanship and domain knowledge is required to get the most out of available data. The results presented here are the result of evaluating several different network structures; different kernel sizes, number of layers, normalization etc. with respect to the *mean square error* (MSE) of the estimated speed. Visual inspection was also used to study the principal behavior in important scenarios. Nonetheless, the result should be considered as proof of concept for the proposed CNN based method. Tailoring the net for a specific application is expected to further improve the results in that specific setting.

The structure ultimately chosen comprises three layers each containing a convolutional layer followed by a ReLU layer and batch normalization, followed by a $20\,\%$ drop out layer, in front of a fully connected layer and a regression layer. The regression layer was used to minimize the MSE of the speed. The structure is illustrated in Fig. 1 and in TABLE I, with the selected parameters. The first convolutional layer uses a small kernel size (2×3) to capture local effects, followed by two larger kernels $(10 \times 1$ and $20 \times 1)$ to extract larger scale effects. The kernels should be able to, more or less, describe a Fourier transform, as that is the basis in the frequency methods, but not be limited to this. Drop-out and normalization are used to enhance performance.

C. Training and Evaluation

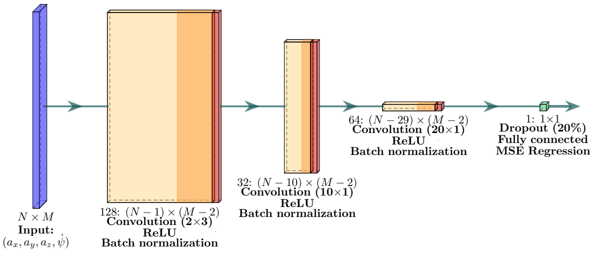


Fig. 1: The CNN layout for speed estimation using accelerometer and yaw rate as input. The number of filters in each layer is highlighted as well as the image sizes and kernels.

TABLE I: Description of the layers used in the network.

Layer Type	Activation	Learnables	
Image input	$30 \times 3 \times 1$	—	
Convolution	$29 \times 1 \times 128$	Weights: $2 \times 3 \times 1 \times 128$	Bias: $1 \times 1 \times 128$
Batch Normalization	$29 \times 1 \times 128$	Offset: $1 \times 1 \times 128$	Scale: $1 \times 1 \times 128$
ReLU	$29 \times 1 \times 128$	—	
Convolution	$20 \times 1 \times 32$	Weights: $10 \times 1 \times 128 \times 32$	Bias: $1 \times 1 \times 23$
Batch Normalization	$20 \times 1 \times 32$	Offset: $1 \times 1 \times 32$	Scale: $1 \times 1 \times 32$
ReLU	$20 \times 1 \times 32$	—	
Convolution	$1 \times 1 \times 64$	Weights: $20 \times 1 \times 32 \times 64$	Bias: $1 \times 1 \times 64$
Batch Normalization	$1 \times 1 \times 64$	Offset: $1 \times 1 \times 64$	Scale: $1 \times 1 \times 64$
ReLU	$1 \times 1 \times 64$	—	
Dropout	$1 \times 1 \times 64$	—	
Fully Connected	$1 \times 1 \times 1$	Weights: 1×64	Bias: 1×1
Regression Output	—	—	

In order to train and evaluate the CNN, the data was divided into separate training, validation, and testing parts, of these the first two were used to train the network and its hyperparameters, and the latter to evaluate the results. Typically, 4 validation sets and 4 testdata sets were used, resulting in about $95\,\%$ of the data being used for training. Performance was evaluated using *root mean square error* (RMSE), where WSS measurements provided the ground truth for the speed. The nets were trained and evaluated using MATLAB 2020 Deep Learning Toolbox [21], with the default settings for loss functions etc.

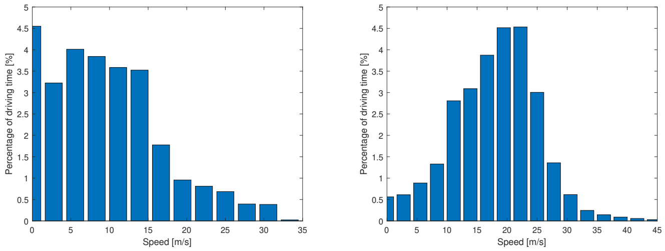
Various combinations of inputs (including or excluding) yaw rate, signal length variation, CNN normalization, etc. were evaluated before arriving at the networks presented in the result section. Note, the learning step is computational intensive, whereas the estimation is very fast; hence, the estimator can easily be implemented in automotive computers or embedded hardware.

III. Experimental Dataset

Two extensive experiments using Audi A6 vehicles are used to evaluate the proposed CNN method. In these accelerometer data and yaw rate information were collected at $100\,\mathrm{Hz}$ and $10\,\mathrm{Hz}$, respectively. The first dataset is a dedicated experiment with one vehicle and one tire set, using a Vector box [22] for data logging in $100\,\mathrm{Hz}$. The latter contains $10\,\mathrm{Hz}$ data collected with NIRA Dynamics' indirect tire pressure monitoring system. In both datasets, ground truth speed is provided by WSS.

The former dataset has the advantage of a higher data rate, but has only a limited amount of data. Typically less than 90 tests of length 20–60 min. The distribution of speeds for approximately $30\,\mathrm{h}$ driving for one vehicle and one tire set can be found in Fig. 2(a).

The second dataset has the advantage of containing many tire sets, and significantly longer tests. It comprises approxi-



(a) Speed profile for 100 Hz data. Total driving time was approximately 30 hours.

(b) Speed profile for 10 Hz data. Total driving time was approximately 158 hours.

Fig. 2: The distribution of speed intervals is depicted in steps of 10 km/h for the 100 Hz and 10 Hz data.

mately 158 h of data from 507 tests, and in total more than 60 unique tire sets. The speed distribution is depicted in Fig. 2(b).

IV. Results

This section presents performance results obtained with the best nets obtained as described above.

A. *Audi A6* 100 *Hz data*

For the dataset collected at 100 Hz both a frequency based method and the CNN can be used. The frequency method is based on *sub-harmonic summation* (SHS, [4, 9]) detailed in [9]. In short, SHS finds the fundamental frequency by maximizing the total energy in all harmonics. The quite high sampling rate allows for capturing the harmonics from the axle rotation in the accelerometer spectrogram. In Fig. 3 the spectrum method using SHS on accelerometer data is depicted for four tests. It is clear that rapid speed changes are poorly handled, as expected as a result of SHS working on batches of data. For relatively high constant speeds the estimates are very good. By analyzing the underlying signal spectrum it is also noted that it is impossible to accurately estimate low speeds or stand-still. Hence, this method cannot be compared to all the tests since the operational speed envelope is too limited. The method sometimes misses the first harmonic, hence overestimating the speed. It is reasonable to believe that adaptive filtering can be used to enhance the performance and reduce outliers.

Fig. 4 shows similar results for the proposed CNN method, with batch length $N = 250$, for the four testdata cases (T100Hz). Contrary to the SHS method, it is clear that the method can handle both low speed and high speed driving, as well as rapid changes in velocity. The good stand-still detection capability is notable. Table II(a) offers evaluation statistics for the evaluated test sequences. The results are favorable.

Studying Fig. 2(a), it is clear that most data is in the low and medium speed range. Hence, it is not surprising that the high speed performance is not as good for CNN. Particularly, if comparing Fig. 3 and Fig. 4 for SHS with CNN, it is clear that high speed (c)–(d) with very constant speed is handled best by the SHS method. This effect of the unbalanced training data can be mitigated by, *e.g.*, collecting more data for the entire speed envelop or weighting the data. For low speed (a)–(b) it is apparent that CNN outperforms SHS.

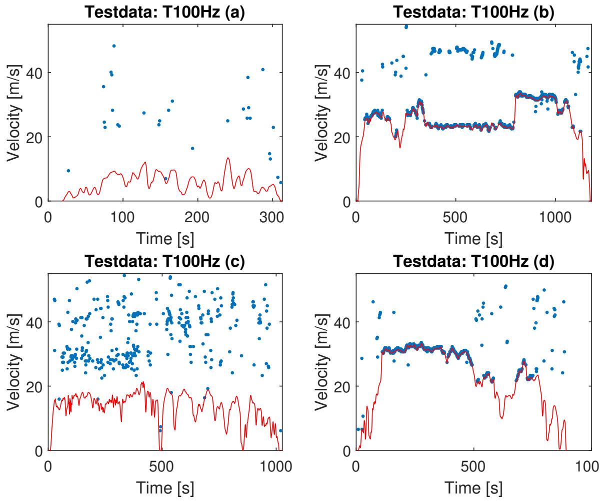


Fig. 3: Acceleration spectrum for speed estimation in 100 Hz data. The data set is the testdata referred as T100Hz (a)–(d). Blue line (estimates), red line (ground truth).

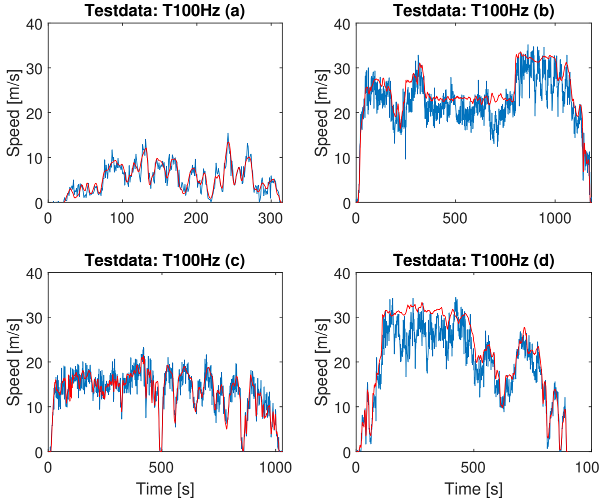


Fig. 4: CNN speed estimation evaluated on four different testdata sets T100Hz (a)–(d) for Vector box 100 Hz data. Blue line (estimates), red line (ground truth).

B. *Audi A6* 10 *Hz data*

The vertical accelerometer is also not available in this vehicle, and hence not used in the CNN. The performance for the four testdata sets is depicted in Table II(b), with

TABLE II: Performance metrics for four selected testdata cases at different data rates. Normalization and yaw rate utilized.

(a) Data at 100 Hz.

Testdata	RMSE [m/s]
T100Hz (a)	1.3
T100Hz (b)	3.5
T100Hz (c)	1.7
T100Hz (d)	4.0

(b) Data at 10 Hz, no a_z.

Testdata	RMSE [m/s]
T10Hz (a)	2.2
T10Hz (b)	2.3
T10Hz (c)	2.5
T10Hz (d)	2.0

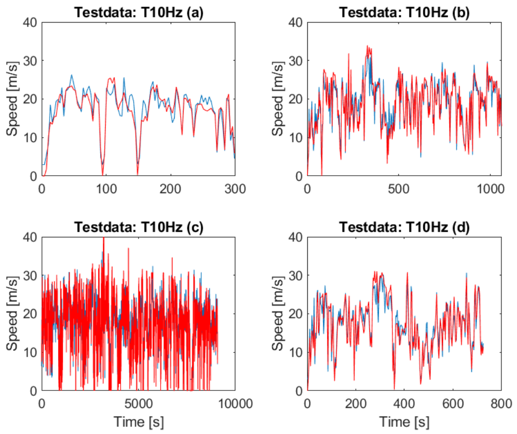


Fig. 5: CNN speed estimation evaluated on four different testdata sets T10Hz (a)–(d) for 10 Hz CAN data. As depicted overtones must be handled as well as discarding outliers. Rapid changes in velocity are particularly challenging. Blue line (estimates), red line (ground truth).

normalization, inclusion of yaw rate, and utilizing the maximal signal length of $N = 30$ samples. The SHS is not used in the comparison, as with 10 Hz sample rate it fails to provide any reasonable results.

Fig. 5 shows the CNN evaluated on four different testdata (T10Hz) sequences. The CNN is successful even if the sample rate is low. The performance is rather similar to the 100 Hz results, which is surprisingly good considering the limited sample frequency and the complete failure of the SHS method.

C. Results and Discussion

Based on the presented results it is concluded that the performance during stand-still and rapid changes is superior for the CNN method compared to the SHS method. While producing reasonable results with 100 Hz data, the SHS completely deteriorated with the 10 Hz data, whereas the proposed CNN only showed a minor performance loss. The CNN also performed much better on segments with low speed, where the spectrum based method struggles. This is mainly due to significant noise terms from suspension and the noise from uneven road segments. The spectrum analysis method performs well on constant speed segments, with sufficient velocity. Overall, the proposed CNN is the best alternative.

V. Conclusions

A new approach for speed estimation using deep learning with a CNN based on accelerometer sensor data has been presented. The method outperformed previous proposed methods utilizing frequency analysis, particularly for low speed regions and handled rapid speed changes much better. In a large experimental automotive study the performance for various setups, data rates etc. were examined. Even for as low sample rate as 10 Hz the absolute RMSE speed was usually below 2 m/s without any additional post-processing. On average the error is about 10 % of the speed within the training interval.

References

[1] I. Skog and P. Händel, “In-car positioning and navigation technologies—a survey,” *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 1, pp. 4–21, 2009.

[2] D. Titterton and J. Weston, *Strapdown Inertial Navigation Technology*, 2nd ed. IET, 2004.

[3] B. G. Quinn and E. J. Hannan, *The estimation and tracking of frequency*, ser. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2001, vol. 9.

[4] D. J. Hermes, “Measurement of pitch by subharmonic summation.” *Journal of the Acoustical Society of America*, vol. 83, no. 1, pp. 257–264, 1988.

[5] P. Stoica, P. Händel, and T. Söderström, “Approximate maximum likelihood frequency estimation,” *Automatica*, vol. 30, no. 1, pp. 131–145, 1994.

[6] R. Karlsson and F. Gustafsson, “The future of automotive localization algorithms: Available, reliable, and scalable localization: Anywhere and anytime,” *IEEE Signal Process. Mag.*, vol. 34, no. 3, pp. 60–69, Mar. 2017.

[7] ——, “Velocity estimation,” U.S. Patent US20 180 231 577A1, Aug., 2015, patent pending.

[8] M. Lindfors, G. Hendeby, F. Gustafsson, and R. Karlsson, “Vehicle speed tracking using chassis vibrations,” in *IEEE IV*, Gothenburg, Sweden, 2016, pp. 214–219.

[9] ——, “Frequency tracking of wheel vibrations,” *IEEE Trans. Control Syst. Technol.*, Apr. 2020.

[10] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York, NY, USA: Springer New York Inc., 2001.

[11] R. Gonzalez, “Deep convolutional neural networks,” *IEEE Signal Process. Mag.*, vol. 35, no. 6, pp. 60–69, Jun. 2018.

[12] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[13] S. Lee, S. Yoon, and H.Cho, “Human activity recognition from accelerometer data using convolutional neural network,” in *IEEE BigComp*, Feb. 2017.

[14] N. Tufek, M. Yalcin, M. Altintas, F. Kalaoglu, Y. Li, and S. K. Bahadir, “Human action recognition using deep learning methods on limited sensory data,” *IEEE Sensors J.*, vol. 20, no. 6, pp. 3101–3112, 2020.

[15] C. Arsene, R. Hankins, and H. Yin, “Deep learning models for denoising ECG signals,” in *27th EUSIPCO*, Sep. 2019.

[16] M. Isaksson, J. Jalden, and M. Murphy, “On using an adaptive neural network to predict lung tumor motion during respiration for radiotherapy applications,” *Medical physics*, vol. 32, pp. 3801–9, 01 2006.

[17] S. Kiranyaz, T. Ince, O. Abdeljaber, O. Avci, and M. Gabbouj, “1-D convolutional neural networks for signal processing applications,” in *IEEE ICASSP*, May 2019.

[18] Y. Gu, Q. Wang, and S. Kamijo, “Intelligent driving data recorder in smartphone using deep neural network-based speedometer and scene understanding,” *IEEE Sensors J.*, vol. 19, no. 1, pp. 287–296, 2019.

[19] A. Basavaraju, J. Du, F. Zhou, and J. Ji, “A machine learning approach to road surface anomaly assessment using smartphone sensors,” *IEEE Sensors J.*, vol. 20, no. 5, pp. 2635–2647, 2020.

[20] A. Shaheryar, Y. Xu, and W. Yousuf, “Robust feature extraction on vibration data under deep-learning framework: An application for fault identification in rotary machines,” *International Journal of Computer Applications*, vol. 167, pp. 37–45, 06 2017.

[21] “MATLAB deep learning toolbox,” 2020, the MathWorks, MA, USA.

[22] Vector Informatik GmbH, *Manual, VN8900 Interface*, 2014.