Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/jtte

Original Research Paper

Learning from the crowd: Road infrastructure monitoring system

Johannes Masino[*], *Jakob Thumm*, *Michael Frey*, *Frank Gauterin*

Institute of Vehicle System Technology, Karlsruhe Institute of Technology (KIT), Karlsruhe 76131, Germany

HIGHLIGHTS

- The paper proposes a system to autonomously and comprehensively monitor the road infrastructure condition.
- The designed methods could incorporate an automatic collection of ground truth data for supervised machine learning.
- The algorithms to compare trajectories are tested in terms of runtime.
- The results suggest to use a range search algorithm coupled with Euclidean distance.

ARTICLE INFO

Article history:
Received 12 January 2017
Received in revised form
13 June 2017
Accepted 15 June 2017
Available online 18 September 2017

Keywords:
Road infrastructure condition
Monitoring
Tree graphs
Euclidean distance
Machine learning
Classification

ABSTRACT

The condition of the road infrastructure has severe impacts on the road safety, driving comfort, and on the rolling resistance. Therefore, the road infrastructure must be monitored comprehensively and in regular intervals to identify damaged road segments and road hazards.

Methods have been developed to comprehensively and automatically digitize the road infrastructure and estimate the road quality, which are based on vehicle sensors and a supervised machine learning classification. Since different types of vehicles have various suspension systems with different response functions, one classifier cannot be taken over to other vehicles. Usually, a high amount of time is needed to acquire training data for each individual vehicle and classifier.

To address this problem, the methods to collect training data automatically for new vehicles based on the comparison of trajectories of untrained and trained vehicles have been developed. The results show that the method based on a *k*-dimensional tree and Euclidean distance performs best and is robust in transferring the information of the road surface from one vehicle to another. Furthermore, this method offers the possibility to merge the output and road infrastructure information from multiple vehicles to enable a more robust and precise prediction of the ground truth.

* *Corresponding author*. Tel.: +49 721 608 41747; fax: +49 721 608 44146.
E-mail addresses: johannes.masino@kit.edu (J. Masino), jakob.thumm@student.kit.edu (J. Thumm), michael.frey@kit.edu (M. Frey), frank.gauterin@kit.edu (F. Gauterin).

Peer review under responsibility of Periodical Offices of Chang'an University.
http://dx.doi.org/10.1016/j.jtte.2017.06.003

1. Introduction

According to the German Federal Statistical Office in 2015 more than 10 billion of Euros were spent on road maintenance projects in Germany to repair road damages (Federal Statistical Office, 2016a). The condition of the road infrastructure is related to rolling resistance and therefore to the amount of CO_2 emissions of combustion engines. Moreover, it affects the range of electric vehicles, driving comfort, vehicle operating costs, and the economy of the country (Ahlin and Granlund, 2002; Molenaar and Sweere, 1981; Soliman, 2006). Faulty streets also have a great influence on the road safety (Ihs, 2004). The German accident statistics prove that more than 1200 accidents were related to road hazards in 2015 (Federal Statistical Office, 2016b). Many roads are regularly inspected by qualified staff to decrease the risk of accidents. A municipality must check the streets in regular intervals and repair all occurred damages in reasonable time. On high heavily busy roads, this happens several times a week, sometimes even daily. In larger cities, many trained inspectors survey the road network daily to log damages of any kind. Smaller communities normally face fewer resources to check their road infrastructure. However, such areas do not have fewer road kilometers that need to be controlled. The road network size in Germany of irregular investigated streets, such as country roads, has a length of 504,700 km and the size of frequently surveyed road, such as national highways, is only 176,800 km (Federal Statistical Office, 2013).

To improve the procedure and enable an autonomous road condition monitoring, we developed a method to estimate the road quality comprehensively and automatically in short and regular intervals. Therefore, we are able to detect many safety related road damages in almost real time. The method is based on a low-cost measurement device, which consists of an inertial sensor and a GPS sensor and is placed near the center of gravity of the vehicle (Masino et al., 2016). Based on a machine learning algorithm and statistics calculated from vibrations and dynamics of the vehicle the system can classify road infrastructure features and estimate the condition. A physical model, which needs lots of computational time and additional sensors at the suspension system of the vehicle, is not required. Since vehicles have different suspension systems, a machine learning model for one vehicle cannot be taken over to other vehicles. They must be trained manually to achieve a high accuracy of classification. Our method addresses this problem and presents an algorithm to collect the required training data automatically. It is based on a comparison of new trajectories to existing ones of trained vehicles.

The developed method can also be used to compare the output of already trained vehicles with each other to provide a more robust and more precise prediction of the road condition and to enable trend recognition by compare trajectory segments with the same location but different timestamps. Overall, with our proposed method, a periodic monitoring of roads can be guaranteed easily. Our system can strongly improve the road safety and quality at comparatively little expense while decreasing the manual and financial effort.

2. Relevant work

There has been research on road infrastructure monitoring based on vehicle sensors, such as accelerometers or acoustic sensors, and machine learning or filters, e.g., Chen et al. (2013), Eriksson et al. (2008), Masino et al. (2017a,b) and Seraj et al. (2016). However, to our knowledge no method has been developed to train new vehicles automatically based on the comparison of trajectories and to get a higher accurate prediction based on the fusion of the information from multiple vehicles.

2.1. *Recognition of street events*

In 2008 the Massachusetts Institute of Technology presented a system that recognizes potholes autonomously (Eriksson et al., 2008). For this purpose, seven taxicabs in Boston were equipped with measuring systems. For each taxicab, a triaxial acceleration sensor measured the vehicle dynamics with a sample rate of 380 Hz and the time, location, speed and direction were acquired from the GPS sensor with 1 Hz. The GPS sensor standard deviation was 3.3 m. The taxi fleet consisted exclusively of the model Toyota Prius from different years of construction. The taxicabs collected 2492 km road data within ten days. The following street event classes were considered good street, pedestrian crossings with thick paint, railway crossings, potholes, manholes, hard stops, turns. The data were labeled with the object, which was run over with the vehicle. A series of filters were applied to the data set to distinguish potholes from other events. Other classes like manholes could not be detected. To test the algorithm, it is applied to both the training data and the large data set from the taxicab fleet. After repeating these tests with random parts of the training data set, potholes could be detected with an accuracy of 92.4%. On well-conditioned roads, the false positive rate lay between 0.12% and 0.63%. On roads with potholes, the false positive rate increased to 14.0%. The algorithm detected 48 potholes in the big data set. A manual verification showed that 39 of these events were actual potholes.

RoADS System from 2014 was intended to recognize road damages and anomalies using smartphones, which were firmly attached to the windscreen of the test vehicles (Seraj et al., 2016). The smartphones had a three-axis acceleration sensor, a gyroscope and a GPS sensor, which were sampled at a frequency of 93 Hz. 45.9 km of road data were collected in two cities by using five different vehicles. In total 100.3 km were traveled. To generate the test data set different street events were run over and the passenger labeled all the important features with an audio recording. The collected data was preprocessed and divided into three classes.

(1) Severe events: sunken manholes, potholes and poorly preserved or heavily patched road sections.
(2) Mild events: all anomalies that occur on only one side of the vehicle, for example cracks, one side patches or one side bumps.
(3) Span events: all events, which extend over the entire width of the road, for example speed bumps, pedestrian crossings, expansion joints and large patched areas.

Before the data was further processed, a high-pass filter had been applied to the vertical acceleration to eliminate cornering, acceleration and deceleration phases. Moreover, the dependence of the vertical acceleration from the speed is removed. For the subsequent feature extraction windows of 2.5 s were built, which overlapped with a factor of 66% or 1.65 s. The data was transformed into the frequency domain and a wavelet transformation was used to suppress noises in data. Both from the time and the frequency domain several features were extracted.

For the anomaly detection, a two-stage support vector machine (SVM) algorithm was used. In the first step, all anomalous windows were distinguished from the normal, so that in the second step the type of event could be determined. To train the SVM, 2073 normal and 993 anomalous windows with an overlap factor of 66% were used. A test of the training, which was performed with a ten-folded cross-validation, achieved an accuracy of 91%. After a successful training of the SVM, the algorithm was tested with a large data set. 264 events were correctly and 43 incorrectly detected, which corresponds to an accuracy of 86%.

The team of Crowdsourcing based road surface monitoring equipped 100 taxicabs in Shenzhen area in China with measurement (Chen et al., 2013). The built-in system was composed of a triaxial acceleration sensor, a GPS module, a GSM module and a microcontroller. The microcontroller detected abnormal road events in real time using a threshold on the vertical acceleration. It was not possible to use a universal threshold of the vertical acceleration for the 100 different vehicles. Therefore, the threshold value is calculated separately for each vehicle with a Gaussian mixture distribution. All events that were detected as abnormal by this method were sent to a central server over the GSM module. On the server, the same filter method was applied, which is described in Eriksson et al. (2008). With this method, potholes could be detected with an accuracy of 90%. Furthermore, the standard deviations of all acceleration values were sent to the server to determine the overall road roughness.

2.2. *Transfer of training data based on trajectories*

Until today, no approach has been presented in any scientific contribution, which allows training data to be transferred to other vehicles. Zhang et al. (2006) presented six different methods to determine the distance and the similarity between two trajectories. These similarity measures were compared in terms of their calculation time and suitability for a clustering algorithm. The goal was to find a method that matches a small distance to similar trajectories.

In Yin and Wolfson (2004) a method is presented which improves the mapping of GPS points on roads, which are stored in a map database. The most widely used function of assigning a GPS point to a road is to assign this GPS point to the nearest lane. However, since the GPS signal is often inaccurate, it may happen that the nearest road is not the one, which was actual driven. The improved method always considers trajectory segments and compares them to the roads nearby. An Euclidean distance is used as a trajectory distance measure. It is important that this approach considers sections and not whole routes.

3. Road infrastructure monitoring system

3.1. *Data acquisition*

The core of our road infrastructure monitoring system is an inertial sensor. A data logger acquires the data from the acceleration and angular rate sensor and the GPS sensor. The data can be transmitted automatically to a server via Wi-Fi, as soon as the vehicle returns to its parking area. The measuring device, which is placed at the center of gravity of the vehicle, has been developed and validated at our institute (Masino et al., 2016). Overall, we acquire the following data of the vehicle: acceleration and rotation rate in three axes with a sample rate of 200 Hz and the GPS position and vehicle velocity with 10 Hz. The accuracy of our GPS system is 1.38 m. To determine the accuracy, we stopped with our vehicle several times at the same position and calculated the standard deviation of the GPS position in meters with the orthodromic distance (Meeus, 1991).

Furthermore, since the GPS sample rate is lower than the inertial sensor sample rate, the GPS data are interpolated with Algorithm 1 (Fig. 1).

A new vector with the length N for the distance x is then calculated based on the time interval $t_k - t_{k-1}$ and the velocity v with the following Eq. (1).

$$x_k = x_{k-1} + v_{k-1}(t_k - t_{k-1}) \quad k = 1, 2, \cdots, N \tag{1}$$

where $x_0 = 0$.

Moreover, the nonuniform data is resampled to uniform data to a fixed rate of 100 samples per meter.

3.2. *Road infrastructure features*

The estimation of the road infrastructure is based on road features, such as potholes, which we want to identify with our measuring device and algorithms. The type and number of road features determines the accuracy and functionality of our data analysis. There might be problems with the assignment of events to the respective features, if there are too many different types of features or if the features are very similar to each other. On the other hand, if only few features are defined, as in the previous literature, only a rough estimation of the road infrastructure condition is possible. Based on civil engineering literature about road infrastructure condition (Beckedahl, 2010) and interviews of experts from civil engineering departments, we define a two-layer approach to monitor the road infrastructure (Table 1). Firstly, it is important to distinguish between the type of road, namely asphalt, concrete, or cobblestone, since the repair measures differ on these deposits and the event of damage depends on the road type. Secondly, we define different street events as follow, which lead to different repair measures. These events can also help us estimate the overall condition of a road.

- Good street: there are no visible or noticeable events on the road surface. The joints occurring on concrete roads at regular intervals are considered normal, as long as the driving is not significantly affected. A cobblestone street

```
Algorithm 1 interpolationGPS
Input: inputInstances, Typ: Instance <time, latitude, longitude, speed>
Output: outputInstances, Typ: Instance <time, latitude, longitude, speed>
cP ← 1                                          ▷ cP := the current Point (1 = start)
currentInstance ← inputInstances(cP)
for i = 2 to count(inputInstances) do
    newInstance ← inputInstances(i)
    if newInstance.time != currentInstance.time then
        startLat ← currentInstance.latitude
        startLon ← currentInstance.longitude
        startSpeed ← currentInstance.speed
        endLat ← newInstance.latitude
        endLon ← newInstance.longitude
        endSpeed ← newInstance.speed
        diffLat ← endLat-startLat
        diffLon ← endLon-startLon
        diffSpeed ← endSpeed-startSpeed
        length ← i-cP
        outputInstances(cP to i).latitude ← startLat +
    (0 to length) · diffLat/length
        outputInstances(cP to i).longitude ← startLon +
    (0 to length) · diffLon/length
        outputInstances(cP to i).speed ← startSpeed +
    (0 to length) · diffSpeed/length
        cP ← i
        currentInstance ← inputInstances(cP)
    end if
end for
return outputInstances
```

Fig. 1 – Interpolating the GPS data through Algorithm 1.

generally has more bumps than the other road classes. Since this cannot be avoided, more irregularities are being admitted here.

- Slight damage: includes all damages that do not require immediate repair, such as cracks, patched areas, and clusters of small potholes. These damages cover a larger area than potholes, but they usually have a lower standard deviation in the vertical acceleration and pitch and roll rate. This damage class does not occur on cobblestones.

Table 1 – Summarizing the events that may occur on the respective road surfaces.

Street event	Asphalt	Concrete	Cobblestone
Good street	x	x	x
Slight damage	x	x	
Pothole	x	x	x
Manhole	x		x
Railway crossing	x		
Bulge	x		x
Speed bump	x		x

Note: “x” means this event can occur on the respective pavement.

- Pothole: on asphalt roads, this corresponds to an outbreak with a minimum depth of 20 mm. On concrete roads, any edge damages are summarized in this class. At cobblestone roads, this event occurs in the form of heavily lowered or missing cobblestones. Typical characteristics of this class are a high standard deviation and high frequencies of the vertical acceleration and the pitch and roll rate.
- Manhole: manhole covers occur in different shapes and sizes, but mainly on asphalt and cobblestones. They have lower frequencies in the pitch rate than potholes.
- Railway crossing: railway crossing mostly occurs on asphalt roads. Because railway crossings extend over the whole width of the road, the standard deviation of the roll rate is not as high as that of a pothole.
- Bulge: bulge represents an increase in the road surface, which occurs on one side of the roadway. Bulges are for example a consequence of tree roots that raise the road surface. This damage only occurs on asphalt and cobblestone roads. Depending on the size of the bulge strong similarities to potholes may occur.
- Speed bump: speed bumps are often used for calming the traffic. They extend over the entire width of the road. Speed bumps have, like railway crossings, no high standard

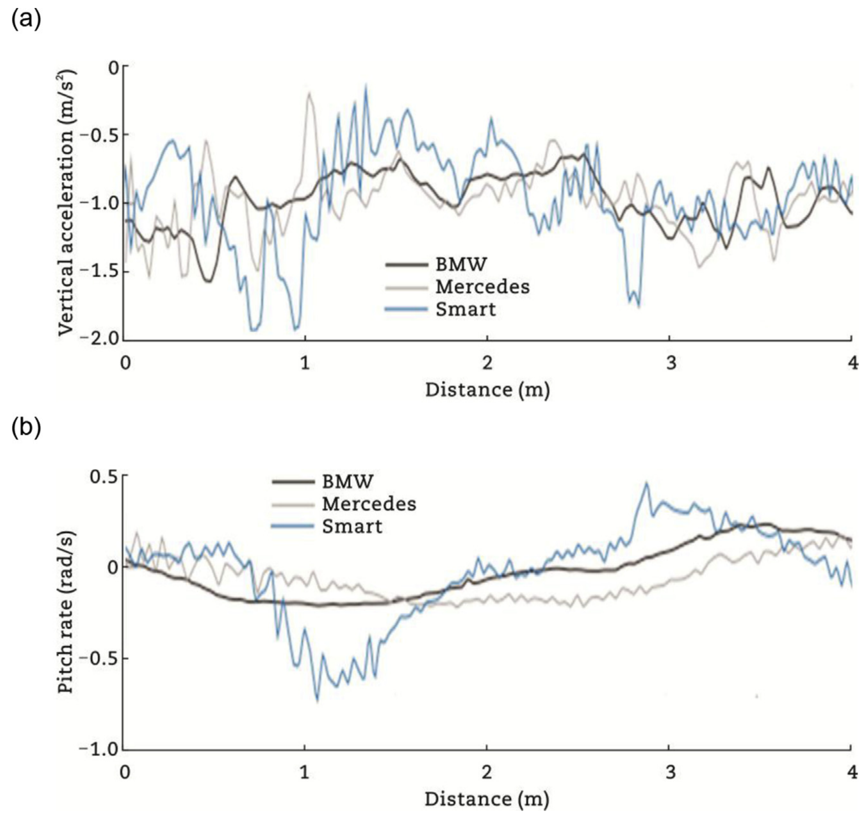


Fig. 2 – Raw data of three vehicles driving over a pothole with a velocity of 7.5 m/s. (a) Vertical acceleration. (b) Pitch rate.

deviation of the roll rate. However, they have a higher amplitude in the vertical acceleration and the pitch rate than railway crossings. Speed bumps are usually used on asphalt roads or cobblestone.

3.3. *Road infrastructure classification*

To estimate road infrastructure features or the road condition we apply a machine learning algorithm, specifically a Support Vector Machine. We calculate features for the algorithm based on the relevant vibrations of the vehicle, such as statistics of the inertial sensor data filtered with Fast Fourier Transformation filters or wavelets. However, the classification model can be easily substituted and this study focuses on algorithms to automatically train new vehicles based on the comparison of the trajectories with already trained vehicles or to fuse the output of multiple vehicles.

4. Learning from the crowd

The vibration behavior of various vehicles differs greatly in the structure, suspension system of the vehicle and the position of the measuring system. In order to prove this hypothesis we analyze the vibrations of three different vehicles, namely a small vehicle (Smart Fortwo), a midsize vehicle (BMW 1 Series) and an upper class vehicle (Mercedes-Benz S 500) with an air suspension system.

Firstly, all three vehicles are driven over the same road hazard, a pothole, at a speed of approximately 7.5 m/s. Fig. 2 shows the raw data of the vertical acceleration and pitch rate of the three different vehicles. The vertical acceleration and pitch rate of the Smart shows higher amplitudes than the other vehicles, whereas the course of the data of the BMW is smoother. Furthermore, the pitch rate of the Mercedes drops down delayed possibly due to the air suspension.

Fig. 3 demonstrates the data from the three vehicles driving over a rough road with a constant velocity of 7.5 m/s. Again, the Smart shows greater oscillations of the z-acceleration than the other two vehicles and the frequency response curve of the pitch rate lies above the two remaining curves. The test results motivate the need for individual classifiers for each vehicle since the statistics and features as input for the classifier differ although the vehicles run over the same event or road condition.

Consequentially, vehicles with individual classifiers based on supervised learning need new training data, which is very time consuming and needs a lot of manual effort. With the following presented method and algorithms we address this problem and automate the training procedure. Furthermore, the output of different vehicles at same positions can be collated and analyzed to get a more precise estimation of the road infrastructure quality. Our method is based on the idea to transfer the information of the road condition to new vehicles, if the new vehicle drives on routes, where data from trained vehicles is already present.

Moreover, we implement several conditions in the process of transferring the label of already existing road infrastructure information to a new vehicle. For example, if the driver of an untrained vehicle avoids to drive over a pothole where

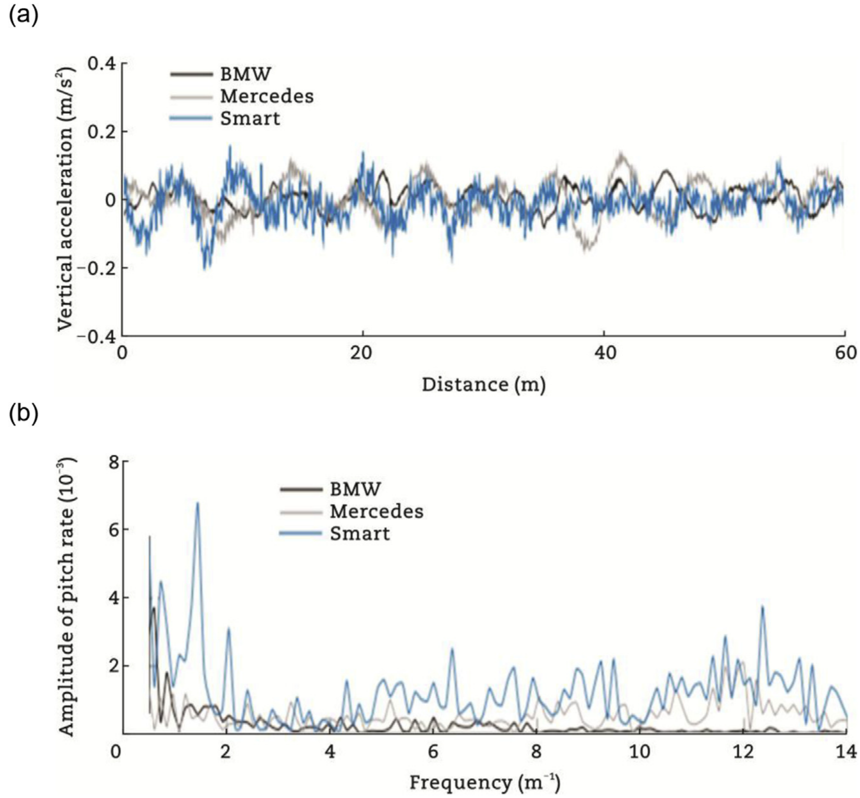


Fig. 3 – Data of three different vehicles driving over the same segment of a rough road at about 7.5 m/s. (a) Vertical acceleration. (b) Amplitude of pitch rate.

already trained vehicles labeled the road segment as potholes, the label is not transferred due to missing amplitudes in the sensor data, and vice versa. Overall, our method only transfers labels or ground truth data to train a new vehicle, if there is a high probability, that the existing label matches with the actual road infrastructure feature and the sensor data.

Before transferring ground truth data to new vehicles or comparing outputs from multiple vehicles at the same position, we compare the trajectories to find road segments, which were overran by these vehicles. For this purpose, we present and compare algorithms to find trajectory segments, which overlap or are very close to each other. To test the algorithms, we apply them to real GPS data collected with our BMW test vehicle, which we drove several times over a railroad crossing from different directions. After the evaluation of the presented algorithms we propose an efficient and robust method for this task.

4.1. *Range search algorithm*

The task of the range search algorithm, which is based on a k-nearest-neighbor (kNN) algorithm, is to find all points q within a radius around a point p (Kakde, 2005). For this purpose, a k-dimensional (k-d) tree is constructed out of all points with Algorithm 2 (Fig. 4), which assigns each point to a node, so that there are roughly equal numbers of points in each node. Fig. 5 shows exemplary a 2-d tree with 100 points and 8 nodes, each consists of 12–13 points on average. For our application, we build a 2-d tree with the GPS coordinates of all vehicles.

After the construction of the tree we want to find all points, that lie within a circle of radius r around a new point p or in our case GPS coordinates that are very close to the trajectories of other vehicles. To achieve this search problem, we apply Algorithms 3, 4 and 5 (Figs. 6–8). The basic idea is to find all nodes, which share an area with the circle around point p. In the second step, we search only these nodes for points, that have an Euclidean distance smaller than r to the point p. With this method we can reduce the time complexity in big O notation from $O(n)$ to $O(\log n)$.

Fig. 9 shows the trajectories of our experiment. With the range search algorithm, we want to find all points that lie in the close range of the red colored trajectory, which represents the trajectory of a new untrained vehicle. The green colored points show the points that the search algorithm finds within a radius of 11.12 m to the corresponding points of the red trajectory. The gray colored points are not close to the red curve.

It is noticeable that even points of trajectories that run in a different direction are recognized as close points. However, the information or label of the road infrastructure at these points must not be transferred to train the new vehicle, since the vehicle did not pass or overrun these road segments. Therefore, we present the following algorithms to compare sections of trajectories in more details as post processing of the range search algorithm to eliminate these points, where the vehicle did definitely not overrun.

Algorithm 2 buildKdTree

Input: A set of points P, current depth d, maximum number of points in node n

Output: The root of the kd-tree storing P.

if P contains only less than k points **then**
  Return a leaf v storing this point
else
  if d is even **then**
    $dim \leftarrow x$
  else
    $dim \leftarrow y$
  end if
  $l \leftarrow median(P_{dim})$ ▷ l is the line, that splits the set of points at the median in dimension dim.
  for All points p_i in P **do**
    if $p_{i,dim} \leq l$ **then**
      add p_i to P_1
    else
      add p_i to P_2
    end if
  end for
  $v_{left} \leftarrow$ buildKdTree(P_1, d+1)
  $v_{right} \leftarrow$ buildKdTree(P_2, d+1)
  Create a node v storing l, make v_{left} the left child of v, and make v_{right} the right child of v.
end if
Return v

Fig. 4 – Building k-d tree with Algorithm 2.

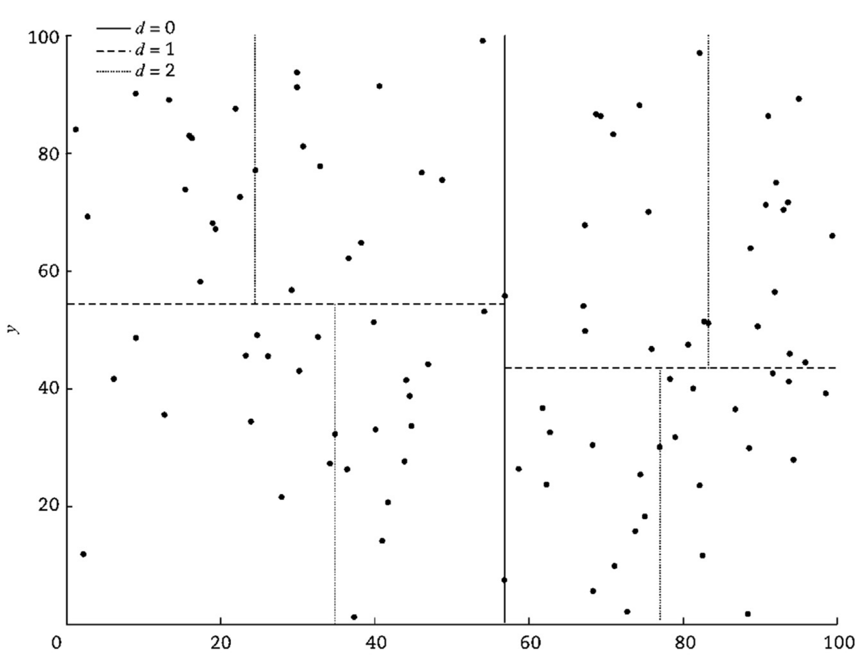


Fig. 5 – A k-dimensional tree with lines, which separates the points (x,y) into 8 nodes with 12–13 points on average.

Algorithm 3 findPointsInRadius

Input: Tree T, Point p:(p_x, p_y), Radius r
Output: All points within the radius around Point p

```
v_0 ← root(T)
leavesList ← findLeaves(v_0, p, r, depth = 0)
pointList ← empty List
for All leaves v_i in leavesList do
    pointList.add(findPointsInNode(v_i, p, r))
end for
Return pointList
```

Fig. 6 – Finding points in radius with Algorithm 3.

Algorithm 4 findLeaves

Input: Node v, Point p:(p_x, p_y), Radius r, Depth d
Output: List of leaves that lie in range $[p_x - r : p_x + r, p_y - r : p_y + r]$

```
if v is a leaf then
    add v to listOfLeaves
else
    if d is even then
        t_min ← p_x − r
        t_max ← p_x + r
    else
        t_min ← p_y − r
        t_max ← p_y + r
    end if
    if l < t_min then                        ▷ l is the splitting value of v
        findLeaves(v_right, p, r, d+1)
    else if l > t_max then
        findLeaves(v_left, p, r, d+1)
    else
        findLeaves(v_right, p, r, d+1)
        findLeaves(v_left, p, r, d+1)
    end if
end if
Return listOfLeaves
```

Fig. 7 – Finding leaves with Algorithm 4.

4.2. *Distance between trajectories*

The trajectory distance algorithm should only output all points from trajectory segments after applying the range search algorithm, which have the same driving direction. We set the size of the trajectory segments to 50 m, which is a good compromise of calculation time and the probability to distinguish turns correctly from driving straight.

A trajectory is a time series of GPS data with the form as follow (Zhang et al., 2006).

$$\left(\left(a_1^x, a_1^y\right), \left(a_2^x, a_2^y\right), \cdots, \left(a_n^x, a_n^y\right)\right) \tag{2}$$

where a^x represents the longitude coordinate, a^y the latitude coordinate. To measure the similarity between two trajectories A with a_k^x and a_k^y and B with b_k^x and b_k^y, we apply different algorithms similar to Zhang et al. (2006) and choose the most efficient and robust one to follow the range search algorithm to eliminate wrong points.

4.2.1. *Euclidean distance*

The Euclidean distance $D_1(\cdot)$ is defined as

$$D_1(A, B) = \frac{1}{N}\sum_{n=1}^{N}\left(\left(a_n^x - b_n^x\right)^2 + \left(a_n^y - b_n^y\right)^2\right)^{\frac{1}{2}} \tag{3}$$

where the length of the compared segments of trajectories A and B must be equal.

Algorithm 5 findPointsInNode

Input: Node v, Point p:(p_x, p_y), Radius r
Output: All points within the radius around Point p

for All points p_i in v **do**
  $d \leftarrow \sqrt{(p_x - p_{ix})^2 + (p_y - p_{iy})^2}$
  if d $\leq$ r **then**
    add p to $listOfPoints$
  end if
end for
Return $listOfPoints$

Fig. 8 – Finding points in node with Algorithm 5.

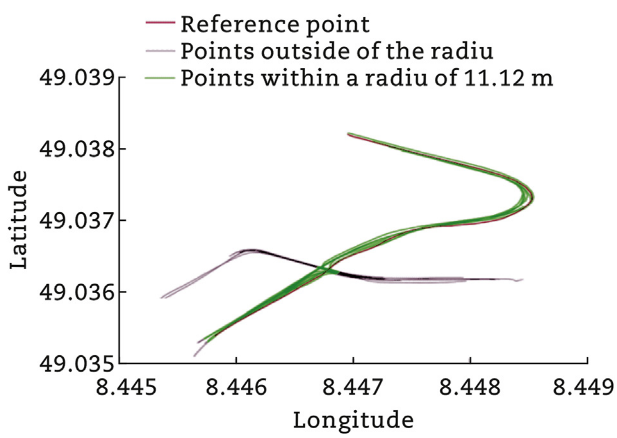


Fig. 9 – Results of our experiment and the application of the range search algorithm.

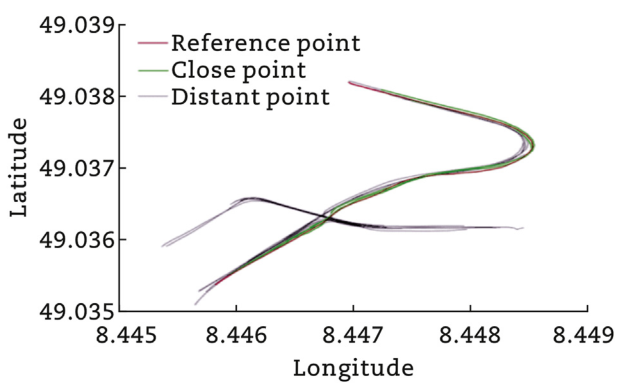


Fig. 10 – The results of range search algorithm and Euclidean distance algorithm in the post processing.

After applying the Euclidean distance we find only segments of trajectories with the same driving direction and therefore on the same driving lane and only segments until the turn on the intersection, as shown in Fig. 10. Compared to only applying the range search algorithm, the trajectory segments of the gray trajectories are colored green in Fig. 10, which have the same driving direction as the red colored trajectory segments.

4.2.2. *Principal component analysis distance*

We calculate the first two principal component analysis (PCA) coefficients a_k^c and b_k^c of trajectories A and B, respectively. Afterwards we calculate the Euclidean distance of the coefficients, whereas a smaller distance indicates a greater similarity of the two trajectories (Bashir et al., 2003).

$$D_2(\mathrm{A},\mathrm{B}) = \left(\sum\nolimits_{k=1}^{2}\left(a_k^c - b_k^c\right)^2\right)^{\frac{1}{2}} \tag{4}$$

Fig. 11 shows the green colored trajectory segments, which were identified by our PCA algorithm as close to the red trajectory. After the application of the PCA algorithm, the trajectory segments that turn on the interaction and drive into the orthogonal direction of the red trajectory are not detected as close. However, the trajectory segments with an opposite direction are detected incorrectly.

To address this problem we apply our Algorithm 6 to consider the driving direction of trajectories A and B (Fig. 12).

After applying our direction algorithm we get similar results as with the Euclidean distance algorithm.

4.2.3. *Hausdorff distance*

The Hausdorff distance algorithm calculates the spatial distance $D_3(\cdot)$ between two trajectories as follow (Lou et al., 2002).

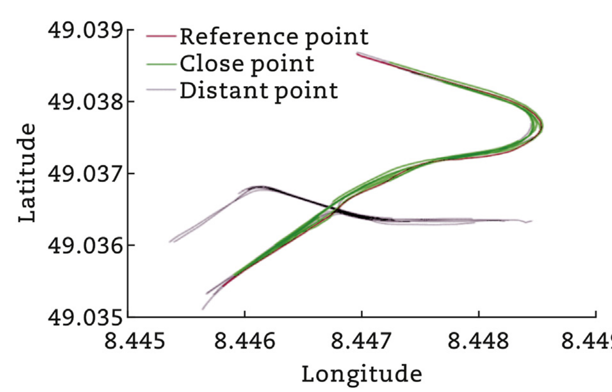


Fig. 11 – The results of range search algorithm in combination with a PCA algorithm and Euclidean distance.

Algorithm 6 considerDirection

Input: x,y, Type: section of trajectory
Output: sameDirection, Type: boolean

```
latitudeFactor ← (x(end)_latitude − x(start)_latitude) * (y(end)_latitude − y(start)_latitude)
longitudeFactor ← (x(end)_longitude − x(start)_longitude) * (y(end)_longitude − y(start)_longitude)
if latitudeFactor ≥ 0 AND longitudeFactor ≥ 0 then
    return true
else
    return false
end if
```

Fig. 12 – Considering the driving direction with Algorithm 6.

$$D_3(A,B) = \max\{d(A,B), d(B,A)\} \tag{5}$$

where $d(A,B) = \max_{a\in A}\min_{b\in B}\|a-b\|$. The application of the Hausdorff distance on our data shows a similar result to the PCA distance, where trajectory segments with an other direction could not be filtered. Therefore, we also have to apply our direction algorithm.

4.2.4. *Dynamic time warping distance*

The dynamic time warping (DTW) algorithm finds the minimum comprehensive path W between two trajectories, which minimizes the cost of the warping. The DTW distance $D_4(\cdot)$ is defined as follow (Keogh and Pazzani, 2000).

$$D_4(A,B) = \min\left\{\frac{1}{K}\left(\sum_{k=1}^{K} w_k\right)^{\frac{1}{2}}\right\} \tag{6}$$

The DTW algorithm returns the similar results as the Euclidean distance.

5. Results

5.1. *Calculation time of distance algorithms*

Since all distance algorithms, partly by applying our direction Algorithm 6, produce similar results, the algorithm is selected based on runtime. In Table 2, a comparison of the algorithms with respect to the measured time per 1000 calculations is shown. If 20 near points are compared, 1000 calculations correspond to a distance of 125 m. The Euclidean distance is outperforming the other algorithms due to its simple computation.

5.2. *Calculation time with and without range search algorithm*

The range search algorithm is only intended to accelerate the algorithm, since all nearby points can also be found using the algorithms discussed in Section 4.2. The speed advantage in the calculation is, however, considerable. A test is performed with a large-area record of 251,177 data points that are up to 70 km away from each other. The traveled distance of 215.65 km corresponds to approximately 6 d of trip at an average mileage of 13,385 km per year.

We apply our method to find close points with and without the range search algorithm. The result in Table 3 shows that the use of the range search algorithm is indispensable. The calculation time with this algorithm is 477.53 times faster than the Euclidean distance algorithm alone.

The performance of the Euclidean distance algorithm with and without range search algorithm is also tested with a high density of trajectories in a small area. Therefore, the intersection data, which has been used in Section 4, is multiplied 50 times. Consequently, there are 1050 trajectories with a total of 48,350 data points in a small area. A test with many vehicles can therefore be simulated. If each car passes an intersection twice a day, the data set corresponds to approximately 18 vessels, which navigate the crossing in 30 d. The results in Table 4 show that the Euclidean distance algorithm performs well even at a high density of trajectories. The calculation time is even smaller including range search algorithm in the pre-processing.

Table 2 – Average calculation time of the algorithms for 1000 trajectories for different lengths of trajectory segments.

Algorithm	Time (s)	
	50 m	200 m
Euclidean distance	0.0050	0.0105
PCA distance	1.0170	1.1600
Hausdorff distance	1.1612	4.8872
DTW distance	0.0598	0.0708

Table 3 – Calculation time to find close trajectory segments in a large area with and without range search algorithm (RSA).

	Time (s)		
	RSA	Euclidean distance algorithm	Sum
Without RSA	–	590.1277	590.1277
With RSA	0.4607	0.6290	1.2358

Table 4 – Calculation time in a dense area with and without range search algorithm (RSA).

	Time (s)		
	RSA	Euclidean distance algorithm	Sum
Without RSA	–	1.8578	1.8578
With RSA	0.0675	0.3853	0.4528

5.3. *Accuracy of the transfer of ground truth data*

To test our method to transfer ground truth data to train a new vehicle, we again use the intersection with trajectories from our BMW test vehicle, which is shown in Section 4. A single road infrastructure event, namely a railway crossing, is approximately at position 8.445, 49.0365. The corresponding data from the BMW test vehicle are marked with railway crossing.

To show the functionality and accuracy of our method to transfer ground truth data, we overran the railway crossing multiple times with a new test vehicle, namely a Smart. We labeled the data of the new vehicle railway crossing manually as soon as we passed it to compare the output of our method to transfer the information automatically.

Fig. 13 shows the result of this test. The green sections are the manually marked railway crossings from the ride with the Smart and the blue sections represent data with the automatic transfer of the label railway crossings from the data, which were previous collected with the BMW. The label railway crossing was transferred successfully except for one case. Latter is due to a bad GPS signal and consequently the route is too far away from the previously traveled routes. However, this underlines our motivation to transfer ground truth data only, if the conditions, e.g., the GPS signal, are well and if there actually was an anomaly in vibration of the vehicle.

6. Discussion

We compare the calculation time for the distance algorithm for our application with the calculation time from Zhang et al. (2006). Table 5 shows the calculation time of the two studies normalized to the calculation time of the Euclidean distance algorithm.

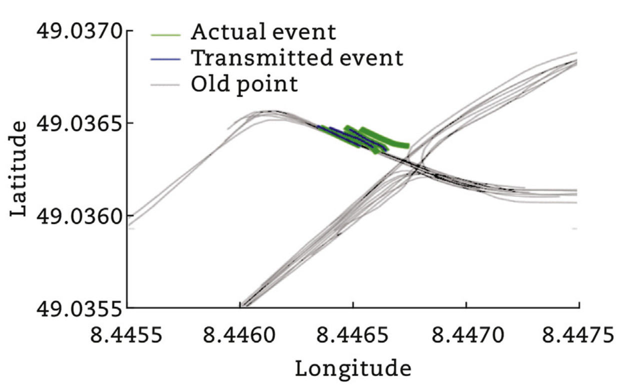


Fig. 13 – The results of the test of our method to transfer ground truth data to new vehicles to train their classifier.

Table 5 – Comparison of the calculation time with different distance algorithms.

	Time (s)			
	Euclidean distance algorithm	PCA distance algorithm	Hausdorff distance algorithm	DTW distance algorithm
Zhang et al. (2006)	1	0.024	130	8
This study	1	203.000	232	12

The comparison indicates that the PCA algorithm has a higher calculation time for our application. The reason is that we use shorter trajectories compared to Zhang et al. (2006) and therefore a multiple of principal components have to be calculated. The trend of the calculation time for Hausdorff and DTW in Zhang et al. (2006) corresponds with our study. The longer calculation time can be again explained with the shorter trajectory lengths in our case.

Our results indicate that our method finds coordinates of already existing training data in our database, which are very close to new data. The trajectories are even on the same traffic lane, which improves the quality of the ground truth data for the new vehicle. With our method, one can transfer the label of already existing training data to data from a new vehicle with a different suspension system to develop an individual classifier.

Furthermore, we consider the following reasons for the transfer of wrong labels.

- Drivers of an unlearned vehicle might avoid to drive over a specific hazard, e.g., a pothole. If earlier drivers overran this event, the transferred label might be wrong.
- Road hazards, such as potholes, might be repaired meanwhile and are not present anymore.
- The previous collected label for training data is wrong or the classification algorithm predicts a wrong event.

Therefore, labels are only transferred if a certain amount of labels from different vehicles and classifiers exist. Secondly, before a label of hazards, which can be avoided from a driver, is transferred, our method reviews the sensor data if there is actually an anomaly in the data, which indicates such an event.

Our method cannot only be applied to automatically collect training data for classifiers of new vehicles with different suspension systems. We can also merge the predictions of the classifiers of different vehicles at specific positions. This brings us to an multiple expert problem (Dawid and Skene, 1979; Raykar et al., 2009; Sheng et al., 2008) and a further method to estimate the most likely road condition or event at this position. Furthermore, with our model we can monitor the road condition over time. For this purpose, we can compare trajectory segments of drivers from different time of specific roads and detect the changes of road conditions or events.

7. Conclusions

This work presents a method to transfer ground truth data automatically from already trained vehicles to new vehicles with various suspension systems to digitize the road infrastructure automatically and comprehensively with various vehicles. The imperative for this method is the different behavior of the vehicle body dependent on the type of vehicle, its structure and suspension system. Therefore, the inertial sensor data, which are the input of our road infrastructure monitoring system, differ over various vehicles, that overrun the same event, such as a pothole. This hypothesis is underlined with data from an experiment under controlled conditions with three different vehicles.

We compare different algorithms to calculate the distance of trajectory segments. For our application, the Euclidean distance algorithm performs best. To boost the calculation time of our method we introduce a range search algorithm as pre-processing of our distance algorithm. With this additional algorithm, we could decrease the calculation time from 590.1277 s with only the distance algorithm to 1.2358 s for trajectory segments in a large area and from 1.8578 s to 0.4528 s in a dense area. Therefore, we could improve the calculation time by a factor of 477.53 or 4.10, respectively.

Moreover, we successfully tested our method if the label from ground truth data from already trained vehicles is transferred correctly to a new vehicle. Our method works defensively, which means that ground truth data is only transferred under good conditions, e.g., strong GPS signal, and if there is an anomaly in the inertial sensor data. Therefore, wrong training data for the new vehicle is avoided, e.g., if a former road hazard which was detected by trained vehicles is repaired before the new vehicle overruns it. However, with our method we can not only automatically transfer ground truth data but we can also compare the estimations of classifiers from different vehicles at the same position. By analyzing the different classifications for several passes, we can calculate the most likely prediction for this position, which is known as a multiple expert problem. Furthermore, we can monitor the road condition over time with our method by comparing the trajectory segments of specific roads over time.

Overall, with the presented method a monitoring system, which is based on vehicular sensor data and a supervised learning algorithm, can automatically learn from the crowd, especially from new vehicles with no training data. The effort to collect training data manually can be drastically reduced.

Acknowledgments

This work was performed within the framework of the project of Technical Aspects of Monitoring the Acoustic Quality of Infrastructure in Road Transport (3714541000), commissioned by the German Federal Environment Agency, and funded by the Federal Ministry for the Environment, Nature Conservation, Building and Nuclear Safety, Germany, within the Environmental Research Plan 2014.

REFERENCES

Ahlin, K., Granlund, N.O.J., 2002. Relating road roughness and vehicle speeds to human whole body vibration and exposure limits. International Journal of Pavement Engineering 3 (4), 207–216.

Bashir, F.I., Khokhar, A.A., Schonfeld, D., 2003. Segmented trajectory based indexing and retrieval of video data. In: International Conference on Image Processing, Barcelona, 2003.

Beckedahl, H.J., 2010. English translation: Pothole, road infrastructure maintenance, Strassenerhaltung, vol. 1. Elsner, Dieburg.

Chen, K., Lu, M., Tan, G., et al., 2013. CRSM: Crowdsourcing based road surface monitoring. In: 2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing (HPCCEUC), Zhangjiajie, 2013.

Dawid, A.P., Skene, A.M., 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. Applied Statistics 28 (1), 20–28.

Eriksson, J., Girod, L., Hull, B., et al., 2008. The pothole patrol: using a mobile sensor network for road surface monitoring. In: The 6th International Conference on Mobile Systems, Applications, and Services, Breckenridge, 2008.

Federal Statistical Office, 2013. Traffic at a Glance. Federal Statistical Office, Wiesbaden.

Federal Statistical Office, 2016a. Sales in Road Construction in Germany for the Years 2009 till 2015. Federal Statistical Office, Wiesbaden.

Federal Statistical Office, 2016b. Traffic Accidents. Federal Statistical Office, Wiesbaden.

Ihs, A., 2004. The influence of road surface condition on traffic safety and ride comfort. In: 6th International Conference on Managing Pavements: the Lessons, the Challenges, the Way Ahead, Brisbane, 2004.

Kakde, H.M., 2005. Range Searching Using Kd Tree. Florida State University, Tallahassee.

Keogh, E.J., Pazzani, M.J., 2000. Scaling up dynamic time warping for data mining applications. In: The 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, 2000.

Lou, J., Liu, Q., Tan, T., et al., 2002. Semantic interpretation of object activities in a surveillance system. In: 16th International Conference on Pattern Recognition (ICPR 2002), Quebec City, 2002.

Masino, J., Frey, M., Gauterin, F., et al., 2016. Development of a highly accurate and low cost measurement device for field operational tests. In: 2016 IEEE 3rd International Symposium on Inertial Sensors and Systems, Laguna Beach, 2016.

Masino, J., Foitzik, M.J., Frey, M., et al., 2017a. Pavement type and wear condition classification from tire cavity acoustic measurements with artificial neural networks. Journal of the Acoustical Society of America 141 (6), 4220–4229.

Masino, J., Pinay, J., Reischl, M., et al., 2017b. Road surface prediction from acoustical measurements in the tire cavity using support vector machine. Applied Acoustics 125, 41–48.

Meeus, J., 1991. Astronomical Algorithms. Willmann-Bell, Richmond.

Molenaar, A.A.A., Sweere, G.T.H., 1981. Road roughness: its evaluation and effect on riding comfort and pavement life. Transportation Research Record 836, 41–49.

Raykar, V.C., Yu, S., Zhao, L.H., et al., 2009. Supervised learning from multiple experts: whom to trust when everyone lies a bit. In: The 26th Annual International Conference on Machine Learning, Montreal, 2009.

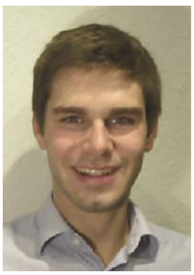
Seraj, F., Van der Zwaag, B.J., Dilo, A., et al., 2016. RoADS: a road pavement monitoring system for anomaly detection using smart phones. Big Data Analytics in the Social and Ubiquitous Context 9546, 128–146.

Sheng, V.S., Provost, F., Ipeirotis, P.G., 2008. Get another label? Improving data quality and data mining using multiple, noisy labelers. In: The 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, 2008.

Soliman, A.M.A., 2006. Effect of Road Roughness on the Vehicle Ride Comfort and Rolling Resistance. SAE Technical Paper 2006-01-1297.

Yin, H., Wolfson, O., 2004. A weight-based map matching method in moving objects databases. In: 16th International Conference on Scientific and Statistical Database Management, Santorini Island, 2004.

Zhang, Z., Huang, K., Tan, T., 2006. Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. In: 18th International Conference on Pattern Recognition (ICPR 2006), Hong Kong, 2006.



Johannes Masino received the BSc and MSc degrees in industrial engineering and management from Karlsruhe Institute of Technology (KIT) in 2011 and 2014, respectively, with semesters abroad in USA and Sweden. He joined the Institute of Vehicle System Technology at Karlsruhe Institute of Technology as a PhD student in 2014. His research interests are signal processing, pattern recognition and machine learning with applications on vehicle sensors.

Jakob Thumm is currently pursuing the BS in mechatronics and information technology at the Karlsruhe Institute of Technology (KIT). His research interests are mechatronic systems, machine learning and vehicle behavior.

Michael Frey received a diploma in mechanical engineering from University of Karlsruhe in 1993. In 2004 he received a doctoral degree in mechanical engineering from University of Karlsruhe. He is the manager of the research group operational strategies and of the research group suspension systems at the Institute of Vehicle Systems Technology.

Frank Gauterin received a diploma degree in physics in 1989 from the Westfälische Wilhelms-Universtät in Münster, Germany, and the Dr. rer. nat. (PhD) degree in 1994 from the University of Oldenburg, Germany. During 1993–2000, he worked as acoustics engineer at Continental AG, Hannover, Germany, and during 2000–2006, as director of noise vibration harshness (NVH) engineering at Continental AG with locations in Germany, USA, and Malaysia. Since 2006 he has been a full professor in vehicle science at the Karlsruhe Institute of Technology (KIT), head of the Institute of Vehicle System Technology, and scientific spokesman of KIT-Center Mobility Systems.