Road-quality classification and bump detection with bicycle-mounted smartphones

Marius Hoffmann, Michael Mock, Michael May
Fraunhofer IAIS
Schloss Birlinghoven, 53754 St Augustin, Germany
first.lastname@iais.fraunhofer.de

Abstract

The paper proposes a embedded surface road classifier for smartphones used to track and classify routes on bikes. The main idea is to provide, along with the route tracking, information about surface quality of the cycling route (is the surface smooth, rough or bumpy?). The main problem is the quantity of accelerometer data that would have to be uploaded along with GPS track, if the analysis was done off-line. Instead, we propose to classify road surfaces online with an embedded classifier, that has been trained off-line. More specifically, we rely on the accelerometer of a bicycle-mounted smartphone for online classification. We carry out experiments to collect cycling tracks consisting of GPS and accelerometer data, label the data and learn a model for classification, which again is deployed on the smartphone. We report on our experiences with classification accuracy on and runtime performance of the classifier on the smartphone.

1 Introduction

The main motivation of this work is to provide a community based cycling route road quality classification service. There are many community based services providing cycling routes together with altitude profiles, but none of them is providing information about the road quality of the route, i.e. whether the road is smooth, rough, or bumpy. Many bicycling community web-portals like [http://www.bikemap.net] offer facilities for uploading and downloading GPS tracks for cycling routes. To our knowledge, none of them provides information about road surface quality of the cycling route. Route quality information could be gathered together with GPS track using the accelerometer data coming from bicycle mounted smartphone. Obviously, including all accelerometer raw data in the data upload would increase data traffic significantly and may not be tolerable for the user, especially when gathering long tracks. The solution is to implement a road surface classification algorithm on the smartphone and to upload the classification results together with the GPS track. Similar approaches already have been successfully applied for other vehicles than bicycles. Pothole detection using GPS data and accelerometer data with dedicated hardware devices mounted in Taxi cabs has already been successfully explored in [Eriksson *et al.*, 2008], [Strazdins12 *et al.*, 2011] and [Mednis *et al.*, 2012] investigate road condition monitoring for vehicular sensor networks based on time series analysis. We investigate experimentally, whether we can achieve a road surface classification using smartphones mounted on bicycles. In order to cope with the restricted computational power of these devices, we apply a machine learning approach: we learn a classifier off-line on a standard PC and apply the classifier online on the smartphone.

We collected GPS tracks and acceleration data (based on the mobile phone's accelerometer sensor) and applied two different approaches for classification of road surface quality, both based on standard machine learning classifiers: in a direct segmentation classification approach, we used manual labeling of road segments of fixed length (smooth, rough, bumpy) to train a classifier, based of various parameter settings for feature extraction. The best result that we obtained in a cross-validation was a 20% increase of accuracy against a standard Kappa-Statistics. In a second approach, we trained a classifier for detecting bumps. Here we achieved an accuracy of 97%. Using this bump detector, we performed a threshold-based road segment classification, which delivered much more comprehensible results. A closer look at input data, manual labeling, classification results, and comparison with the real-world, revealed that the manual labeling was error prone . We conclude that the simple bump-detector based classification approach can be used for road surface quality classification and even does not require further manual labeling of road segments.

2 Classification approach and Results

Most of today's smartphones are equipped with GPS and acceleromater sensors. In order to ensure that our algorithm performs not only on today's top range models, we carried out the experiments with a 2-years old Nokia 5800, one of the first mass models providing accelerometer data. Figure 1 illustrates a track of accelerometer data collected with a smartphone.

Figure 1 shows the length of the accelerometer vector plotted over a track. We see that the data provides a more or less continuos signal (at 37 Hz in our case) over the complete track. As we want to explore a machine learning classification approach for road surface classification, we first have to

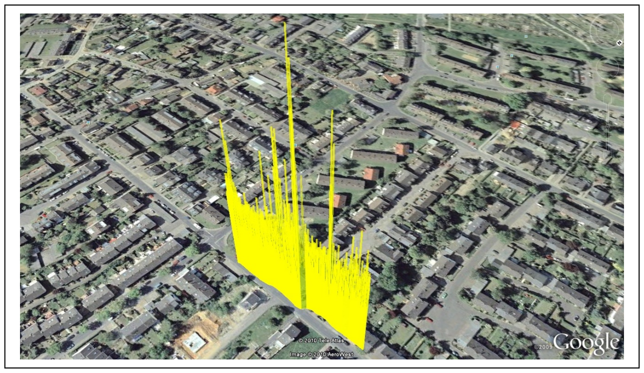

Figure 1: This figure shows a recorded test track of one road. The peeks s in this chart are bumps on the road. The smartphone was attached to the handle of the bike

define the features which are used to train the classifier. The raw data consists of GPS positions and their time stamps, and acceleration values only. Acceleration values are represented by a three-dimensional vector. In a first step, we extract as many features from the data as possible and evaluate experimentally, which feature selection yields the best classification result.

In our first approach to classification described in section 2.1, we divide the road into segments of varying length. In our second approach described in section 2.2, we just consider two subsequent GPS points as boundary of a segment. In both cases, we get a segmentation of the cycling route in a sequence of segments as shown in Figure 2. As a result, the recorded acceleration data is associated to a certain segment.

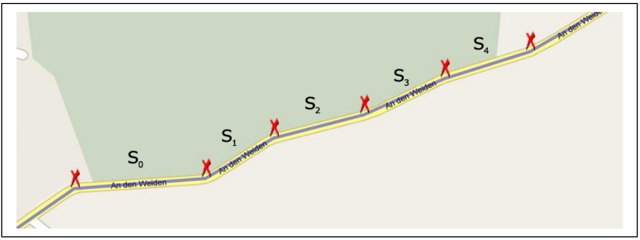

Figure 2: This figure shows how a track will be segmented into a set of segments

The segmentation shown in 2 allows to indicate, which position of the road has a certain surface property or would even contain potholes. We can now analyze segment by segment depending on the data recorded for the segment and make statements about its road surface quality. These information's can be used as features for our machine learning approach. At the end each segment contains GPS and acceleration data which can be used for creating features for this segment.

Features which can be extracted from the GPS data are *speed* and *inclination*. To simplify the handling of the acceleration data provided by the accelerometer, which is made up of an 3D vector, we will further use L2-norm of this vector which is defined as $||x|| = \sqrt{x_1^2 + \ldots + x_n^2}$. The example shown in Figure 1 already illustrates, that changes in these values and sometimes even potholes can be detected by (human) visual inspection of the data. For our machine learning approach, we extract the mean, the variance and the standard deviation of the acceleration values of a segment as features for this segment.

2.1 Direct road surface classification

In this section, we apply standard classification methods to segments of varying length, based on the features described above. For our analysis we consider a number of previous segments which are before the segment that we want to classify. We define a whole road as a set of segments $S = \{s_1, s_2, \ldots, s_n\}$. We consider the previous x segments $s_{i-x}, s_{i-(x-1)}, \ldots, s_i$ of the segment i which we want to classify as features for s_i. In this case the features of the previous segments serve us (primarily our machine learning algorithm) as additional information's for our analysis. How much these feature information's are relevant and how many segments we must consider has be analyzed experimentally.

The organization of the training data is shown in Table 1. Now we use all extracted features of such a set of segments as training data. Every segment has its own row with its own features and additional features of previous segments. Each row in this table also contains the class as entry in the column named *label*. This column contains class which later on will be learned by the machine learning algorithm. For example row 1 has the label *smooth* as class for segment S_1.

$f_{S_{i-2}}$	$f_{S_{i-1}}$	f_{S_i}	label(S_i)
-	f_{S_0}	f_{S_1}	smooth
f_{S_0}	f_{S_1}	f_{S_2}	smooth
f_{S_1}	f_{S_2}	f_{S_3}	smooth
f_{S_2}	f_{S_3}	f_{S_4}	rough
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 1: This table illustrates how the features of each segment are arranged in order to generate a training set of data

We want to evaluate how well the classifiers can learn from the provided data and which features and parameters influence the performance of these classifiers. The goal is to evaluate whether it is possible at all to learn from the data and if so, which are the best parameters (for example segment length, number of segments to be included in the table).

As raw data we recorded one route several times. The route for direct surface classification was recorded 16 times and leads through urban terrain mostly the city of Bonn and they have a length of approximate 13-14km per track (the deviation in length results from the GPS inaccuracy). Each track was labeled for classification by hand with the tool presented in [Guc *et al.*, 2008].

The previously mentioned segment arrangement $s_{i-x}, s_{i-(x-1)}, \ldots, s_i$ will further be called S_{line} which only consist of previous segments and where i is our current position. For the segments length , The other *Fixed* segment length is fixed from the beginning (during the evaluation fixed values of 1m, 2m, 5m, 10m, 15m and 20m are used). For the fixed length parameter the amount of acceleration values can vary, because the amount of values is speed dependant. For classification we will use two different Algorithms the K-Nearest-Neighbor and the Naïve Bayesian Classifier. Five different features were extracted from the training data :*speed*, *inclination*, *acceleration mean*, *acceleration variance*, *acceleration standard deviation*.

The following table shows a compact overview of all parameters which were evaluated.

Parameter type	Parameter Value
ML-algorithm	K-NN, Naive Bayes
segments lengths	variabe length: gps
	fixed length: 1m, 2m, 5m, 10m, 15m, 20m
number of segments	3, 5, 7, 9, 11, 13
extracted features	inclination, speed, acceleration (mean, variance, std)

Table 2: This table gives an overview of all parameters which were changed during evaluation. The acceleration contains three features, acceleration-mean, -variance and -standard deviation)

To measure the performance of the classification algorithm on the evaluation data, a 10-folded cross-validation was included. A N-folded cross validation splits the test data into N equally large sets and then uses $N-1$ set for training to classifier and 1 set for validating the learned concept this is repeated N times where for every iteration a different set of the N sets is used for validation. At the end a confusion matrix is provided from the cross-validation module which consists of the average performance values of the classification.

Additionally we performed a feature selection optimization in order to find the best feature combination. This optimization allows to find a feature combination wich only contains features which influence the learning algorithm positivly and result in hoch accuracy. Features which confuse the learning scheme will not be selected anymore. We found thaht the previously mentioned *speed* and *inclination* feature confuses the learning scheme and results in performances which are worse than the corresponding kappa statistics.

	true smooth	true bumpy	true rough	class precision
pred. smooth	5785	882	92	85,590%
pred. bumpy	836	1052	62	53,949%
pred. rough	151	110	492	65,339%
class recall	85,425%	51,468%	76,161%	**accuracy:** 77,457%

Table 3

The classification performance for the Naive Bayes and the K-NN were almost similar, but the K-NN performed (on average) slightly better than the Naive Bayes.

For K-NN algorithm, the performance increases with an increasing number of the segments which are considered for classification. The Naive Bayes classifier, however, has a more constant performance, independently of the number of segments included in the table. The evaluation also showed that the classification results which use longer segments lengths (15m and 20m) perform much better than the ones with short segment length's (2m). When looking at the influences of all features, we observed that the speed feature does not contribute to the classification. The inclination feature, even worse, confuses the classifier.

The best results (table 3) are achieved with the features acceleration (mean, variance, standard deviation) and a segment length of 20m and 13 segments must be considered for classification. The used segment setup is the S_{line} setup. The corresponding kappa statistic achieves an accuracy of 56,357% which makes a difference of 21,101% between the classifier and its kappa statistic.

The overall results of the classification (at best 78%) are not very satisfying for a classification model. We will see in section 2.2 that the bump detection just based on GPS-defined segments performs much better.

2.2 Bump detection based classification

In this approach, we first consider the detection of single bumps or potholes. The classifier in this first just distinguishes the two classes: "bump" and "no bump". For the bump classification a different route was selected and recorded 15 times. Each of them has a length between 110m and 130m per track (here the deviation in length also results from the GPS inaccuracy). Again each track was labeled for classification by hand via the already mentioned annotator tool.

The performance of the bump classification works out much better compared to the highest accuracy of the surface classification. Again, the feature "speed" turned out to be irrelevant and the feature "inclination" was confusing the classifier. It was also observed that (for surface- not bump-classification), the more segments are considered the more the accuracy declines. The reason for this is that the longer the considered area the more unimportant information is contained in the data which should be classified. In comparison to the surface classification, the bump classification needs shorter segment length's (1m to 5m) to reach high classification accuracy. The longer the segment lengths, the worse the classification performance gets. The long segment also confuse the classification algorithm, this was verified by comparing the results of the classification with the corresponding kappa statistic. The best result were achieved with the segment length *GPS* parameter. This is quite expected, because "bumps" are short term events and GPS-based segmentation (i.e. every two succeeding GPS points define a segment) is the smallest achievable spatial granularity.

	true no bump	true bump	class precision
pred. no bump	404	6	98,537%
pred. bump	2	29	93,548%
class recall	99,507%	82,857%	**accuracy:** 98,186%

Table 4

As we can see it is indeed possible to do pothole and bumpy detection with a very high accuracy, just using the Naive Bayes Classifier on a single segment. This led us to extend this simple approach to be applicable in road surface classification, with the three classes ”smooth”, ”rough”, and ”bumpy”, as described in the following.

Extended bump classification The bump detection can be altered slightly to derive another concept for surface classification. The main idea is to count the number of bumpy segments in a certain road section. Depending on that number, one of the classes ”smooth”, ”rough”, and ”bumpy” is assigned as follows:

- For $0 \leq |bumps| \leq \frac{N}{3}$, the class *smooth* is assigned.
- For $\frac{N}{3} < |bumps| \leq \frac{2N}{3}$, the class *rough* is assigned
- For $\frac{2N}{3} < |bumps| \leq N$, the class *bumpy* is assigned

Not surprisingly, the best results were achieved for N=3, i.e. just considering the GPS-Segments $S_{i-1}, S_i$and S_{i+1} for the classification of GPS-segment S_i. In other words, a GPS segment is considered as, for example, *smooth*, if at most one of its preceding, the GPS-segment itself, and the succeeding GPS-segment have a bump. The results are shown in Table 5.

	true smooth	**true bumpy**	**true rough**	**class precision**
pred. smooth	27865	1113	2129	89,578%
pred. bumpy	2249	1823	3315	24,678%
pred. rough	1488	537	2893	58,825%
class recall	88,175%	52,491%	34,701%	**accuracy:** 75,051%

Table 5: Confusion matrix of a the best performing classification which considered 3 segments during its classification

The classifier with the best accuracy for surface classification achieves $\approx 75\%$ the classifiers from the previous sections which directly learn the labels from the training data perform much worse. For the extended bump classification the K-NN classifier achieves 61% accuracy. A random classifier with the same label distribution performs with $\approx 57\%$ accuracy.

The confusion matrix of the extended bump classifier explains why the accuracy is not higher. The classifier is quite good for *smooth* data, but it confuses *rough* and *bumpy* data. A closer look and comparison with the recorded variances in Figure 3 reveals that most probably, the labeling was not consistent in assigning the labels ”rough” and ”bumpy”.

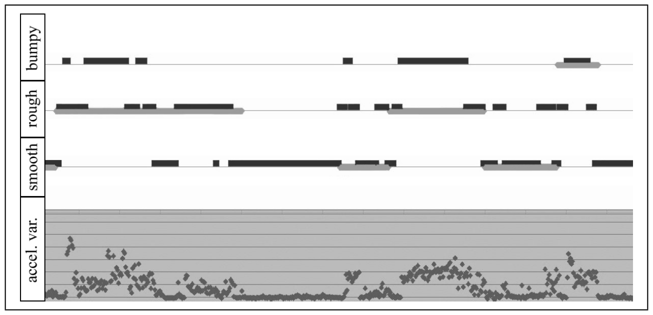

Figure 3: This diagram illustrates the results of inaccurate manual data labeling

Figure 3 visualizes the acceleration variance combined with their manual labels of a section from a recorded track. For each label class the figure shows the manual labels (light gray bars) and the predicted labels (dark gray bars). It can be seen that the light gray labels for the rough class are not modeled with sufficient detailness (on the left side of the diagram). It can also be seen from the acceleration values that this label contains parts of different labels like smooth and bumpy which were not correctly labeled. The diagram shows that the classifier indeed is more often correct than the manual label which is unfortunately the reference for the performance. This is the main reason for the ”bad” performance of the classifiers and explains also the confusion matrix (table 5).

2.3 Classifier implementation on the smartphone

In this section we will discuss the runtime of the whole classification process which was implemented in J2ME. The one of initial goals of this work is to make the classification process possible in the online mode of the client.

Once learned, the classifier has to execute the following steps online on the smartphone.

- calculate the mean, variance and standart deviation of all previous absolute acceleration vector values
- assemble classification data
- applies Naive Bayes classifier for bump detection
- put prediction to bump LIFO (these LIFO stores previous classifications, which are needed to calculate surface prediction)
- put GPS coordinates and prediction for this segment to ObservationBuilder
- builds observation
- sends observation

The execution time of the learned classifier took less than 2 ms in a JME implementation on a Nokia 5800 with an ARM CPU execution at 400 Mhz. The accelerator delivered data at 37 Hz, resulting in 37 values which must be evaluated at each GPS point (given that GPS is running at 1Hz). This means that the overall impact of the classifier on the device performance was very low and that classifier execution finished safely before the next accelerometer values came in.

3 Conclusion

It was shown that in general a surface and a bump classification can be realized via a machine learning approach. It was shown how the data must be preprocessed to achieve good classification results and which features play an important role in this classification process. At the current state,the classification is not as good as it could be. We showed that the correctness and accuracy of the labels in training data should be improved for training a machine learning algorithm. However, we also achieved very good bump detection The learned classifier is fast enough to be executed online on a moderately fast smartphone hardware and needs no further learning or labeling. Surface classification may derived from this. As the classifier performed best for short segments, mainly based on the variance of the length of the acceleration vector, we also see a good chance for just time-series based analysis approaches such as used in [Mednis *et al.*, 2012] or [Mladenov and Mock, 2009] to be applied for road surface classification. As application, biking communities can profit from the presented approach for displaying route quality information on a community portal, or cylcing-friendly cities can monitor the surface quality of their cycling route network for detecting damage and initiating road repair.

Acknowledgements

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 255951 (LIFT Project).

References

[Eriksson *et al.*, 2008] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan. The pothole patrol: Using a mobile sensor network for road surface monitoring. In *Proceeding of the 6th international conference on Mobile systems, applications, and services*, pages 29–39. ACM, 2008.

[Guc *et al.*, 2008] B. Guc, M. May, Y. Saygin, and C. Körner. Semantic Annotation of GPS Trajectories. In *11th AGILE International Conference on Geographic Information Science, Girona, Spain*, 2008.

[Mednis *et al.*, 2012] Artis Mednis, Atis Elsts, and Leo Selavo. Embedded solution for road condition monitoring using vehicular sensor networks. In *Application of Information and Communication Technologies (AICT), 2012 6th International Conference on*, pages 1–5. IEEE, 2012.

[Mladenov and Mock, 2009] M. Mladenov and M. Mock. A step counter service for Java-enabled devices using a built-in accelerometer. In *Proceedings of the 1st International Workshop on Context-Aware Middleware and Services: affiliated with the 4th International Conference on Communication System Software and Middleware (COMSWARE 2009)*, pages 1–5. ACM, 2009.

[Strazdins12 *et al.*, 2011] Girts Strazdins12, Artis Mednis12, Georgijs Kanonirs, Reinholds Zviedris12, and Leo Selavo12. Towards vehicular sensor networks with android smartphones for road surface monitoring. 2011.