VEHICLE LOCALIZATION BY LIDAR POINT CORRELATION IMPROVED BY CHANGE DETECTION

A. Schlichting*, C. Brenner

Institute of Cartography and Geoinformatics, Leibniz Universität Hannover, Germany -
(alexander.schlichting, claus.brenner)@ikg.uni-hannover.de

Commission I, ICWG I/Va

KEY WORDS: Localization, LiDAR, Mobile Mapping, Change Detection, Correlation, Classification

ABSTRACT:

LiDAR sensors are proven sensors for accurate vehicle localization. Instead of detecting and matching features in the LiDAR data, we want to use the entire information provided by the scanners. As dynamic objects, like cars, pedestrians or even construction sites could lead to wrong localization results, we use a change detection algorithm to detect these objects in the reference data. If an object occurs in a certain number of measurements at the same position, we mark it and every containing point as static. In the next step, we merge the data of the single measurement epochs to one reference dataset, whereby we only use static points. Further, we also use a classification algorithm to detect trees.
For the online localization of the vehicle, we use simulated data of a vertical aligned automotive LiDAR sensor. As we only want to use static objects in this case as well, we use a random forest classifier to detect dynamic scan points online. Since the automotive data is derived from the LiDAR Mobile Mapping System, we are able to use the labelled objects from the reference data generation step to create the training data and further to detect dynamic objects online. The localization then can be done by a point to image correlation method using only static objects. We achieved a localization standard deviation of about 5 cm (position) and 0.06° (heading), and were able to successfully localize the vehicle in about 93 % of the cases along a trajectory of 13 km in Hannover, Germany.

1. INTRODUCTION

Accurate localization is essential for highly automated vehicles. Driver assistance systems or self-driving cars need to know their position with an accuracy of some centimeters to a few decimeters. Differential GNSS sensors, even combined with an INS solution, can't guarantee these high accuracies. Especially in cities, with high buildings and trees causing multi-path effects and signal outages, a GNSS solution is not reliable. Therefore additional sensors, like cameras or LiDAR sensors, are needed.
A first vehicle localization approach by using camera data has been published in the 1990s. (Pomerleau and Jochem, 1996) detect lane markers in camera images to determine the position of autonomous driving cars on highways and the road curvature. Nowadays, also LiDAR systems are used to detect lane markers by their reflectivity (Nothdurft et al., 2011) (Yoneda et al., 2015). The detected markers are then matched to a digital map to improve the GNSS/INS position. In (Ziegler et al., 2014) the lane marker detection approach is combined with a 2D feature detection. They use an illumination robust descriptor (DIRD), which is presented in (Lategahn et al., 2014). Though the feature descriptor is not influenced by illumination effects, it still depends on the current vehicle view point and the object surfaces. In (Qu et al., 2015) a geo-referenced traffic sign based localization is presented. The detected traffic signs are used in a local bundle adjustment to decrease the localization error of an additional system.
In (Brenner and Hofmann, 2012) the potential of 3D landmarks, namely pole-like objects, is demonstrated. Poles, extracted from 3D point clouds gathered by LiDAR sensors, are matched to reference landmarks to correct the GPS-based localization.

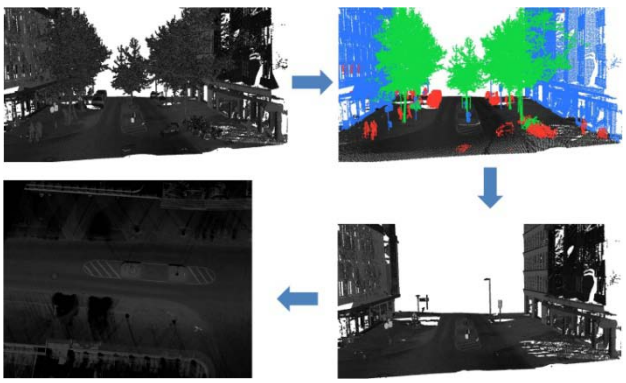

Figure 1. Image generation workflow. Dynamic objects like trees, pedestrians and cars are detected and removed. The reference images are generated using the filtered point clouds.

Disadvantages of this method are the relative low quantity of occurring features and false positive feature detections, especially if an automotive LiDAR sensor is used. Another difficulty is the matching step, which becomes even harder with many false positives. Although this problem is partially solved by using local feature patterns in (Schlichting and Brenner, 2014), the number of correct matched features still is not high enough for a reliable localization.
Instead of using specific landmarks, which means that only a small percentage of the reflected laser beams is taken into account, several approaches make use of the entire LiDAR data. (Yoneda et al., 2014) use the generated 3D point cloud for localization by matching it to a 3D reference cloud in real-time. One big disadvantage of this approach is that storing the reference cloud in a map requires much storage space.

* Corresponding author

(Levinson et al., 2007) are using LiDAR sensors to extract a 2D orthoimage of the ground intensity values. They correlate the extracted images to a given map and localize the vehicle relative to this map using a particle filter. In (Maddern et al., 2015) also a 2D height representation is used. To consider the dynamics of a city, they chose a so called experience-based localization with multiple maps.

In this paper we only want to use one map, containing 2D height and intensity images as a representation of 3D point clouds. For the creation of the reference images, we use several data records gathered by a highly accurate Mobile Mapping System to guarantee a high resolution of a few centimeters. In future scenarios, a cooperative mapping approach of vehicles equipped with automotive LiDAR sensors can record the map instead of using one accurate Mobile Mapping System. The principle of the map generation step is shown in Figure 1. Dynamic objects, like cars, pedestrians or construction sites, are filtered using a change detection algorithm. (Aijazi et al., 2013) detect dynamic objects using a classification approach which divides the objects into predefined categories. Following this, the remaining static cloud is examined for changes by comparing 3D evidence grids. (Xiao et al. 2015) use an occupancy grid based method on the scanning rays without any point voxelization. As we want to detect dynamic objects by the change detection algorithm, we decided to use an object based method. The occurring objects of the single datasets are segmented by a seeded region growing algorithm (Adams and Bischof, 1994) and compared to each other. They are only regarded in the scan images, if they appear in a certain number of measurements at the same position. If not, they are labeled as dynamic objects. Further, we also labeled trees as dynamic objects. They are detected by a random forest classification (Breiman, 2001), similar to methods presented in (Pu et al., 2011) and (Wu et al., 2013).

At this point of research, we use simulated data of a 2D LiDAR sensor mounted in vertical direction on the vehicle to generate the online measurements which perform the online localization. The data is matched to the reference images by a correlation approach. The position deviation to the true vehicle position can be directly derived by the best scan to reference correlation result. A further distinction to previous methods is, that we only want to use static objects in the localization process. For every scan point we calculate a feature vector. Then, the points are classified into static and dynamic points by a random forest classifier. The training data is generated using the labeled objects of the change detection and tree classification step.

The paper is organized as follows. In section 2 the acquisition of the reference point clouds and the simulated online clouds is described. The generation of the reference images, containing the tree classification and the change detection algorithm, is presented in section 3. In section 4 the localization by the point to image matching is described with an evaluation in section 5. Finally, in section 6 conclusions are drawn.

2. DATA ACQUISITION

2.1 Reference Data

For our experiments we used data acquired from a Riegl VMX-250 Mobile Mapping System, containing two laser scanners, a camera system and a localization unit. The system is shown in Figure 2.

The localization is provided by a highly accurate GNNS/INS system combined with an external Distance Measurement Instrument (DMI). The preprocessing step is made by the corresponding Riegl software and additional software for GNSS/INS processing, using reference data from the Satellite

Figure 2. Riegl Mobile Mapping System.

Positioning Service SAPOS. The resulting trajectory is within an accuracy of about 10 to 30 cm in height and 20 cm in position in urban areas. Each scanner measures 100 scan lines per second with an overall scanning rate of 300,000 points per second (Riegl, 2011). The measurement range is limited to 200 meters, the ranging accuracy is 10 mm. Because of its high accuracy and point density, we use the laser scanners to create the reference data.

2.2 Simulated 2D LiDAR sensor

The measurement data of the automotive LiDAR sensor is simulated at this stage of research. The scanner is assumed to be aligned vertically, which means that a single scan line consists of point measurements orthogonal to the driving direction of the vehicle. We sample the data of the Mobile Mapping System in a way that a scan line consists of 580 point measurements with an opening angle of 290 degrees and a resolution of 0.5 degrees. The scanning rate is set to 50 scan lines per second, which yields a point resolution of 20 centimeters in driving direction at a driving speed of 10 m/s.

3. REFERENCE IMAGES

The generation of the reference map is an important part of our vehicle localization approach. The references are realized as 2D representations of highly dense point clouds, separated into a height and an intensity image. As we only want to use static objects for the purpose of localization, we have to filter dynamics from the original point cloud. This is done in two steps. First trees are removed via a classification approach. The remaining dynamic objects are detected by a change detection algorithm, comparing several data sets of the same location measured at different timestamps. To reduce computation time, we perform a downsampling of the point clouds using a voxel grid filter with a leaf size of 10 cm provided by the Point Cloud Library (PCL) (Rusu and Cousins, 2011).

3.1 Alignment

The accuracy of point measurements is directly influenced by the localization accuracy of the Mobile Mapping System. If several point clouds are measured at different dates, the clouds are shifted and rotated relative to each other. As we want to save the data of several measurement campaigns in a single reference set, the point clouds have to be transformed. We solve this issue by determining the transformation to one reference point cloud by using an Iterative Closest Point (ICP) implementation, which was first introduced by (Besl and McKay, 1992). To prevent false point matches, we segment the point clouds into three parts by calculating the point normals: Points with a normal vector in driving direction (x) and

perpendicular to the driving direction (*y*) and ground points with a normal vector in *z* direction. The translation vector consists of the translation values corresponding to the normal vectors of the three point sets. As it turned out, the influence of the rotation angles in this case is not relevant.
To evaluate the alignment accuracy, we pruned three plane regions from the reference point cloud and from five aligned clouds, also in driving direction, perpendicular to driving direction and a ground region. After the alignment step, we calculated the root mean square deviation to the reference plane parameters. The results are shown in Table 1.

direction	σ_{Plane} [mm]	Mean RMS [mm]
x	4,0	11,1
y	4,6	8,3
z	4,7	9,3

Table 1. Results of the alignment to a reference data set.

It can be seen, that the point clouds can be aligned to a reference cloud within an accuracy of about 1 cm, although a downsampling with a 10 cm voxel size was performed in advance.

3.2 Object segmentation

The detection of dynamic objects requires a segmentation of point clouds in single freestanding objects. The segmentation is performed using a two-step seeded region growing algorithm. After computing the local normal vector for every point, we pick the seed points from which the region starts to grow. We sort the points by their height and choose the k lowest points with a normal vector that points up within a certain tolerance. A new point is added to the ground if it is within a radius of 20 cm and its local normal vector in *z* direction points up within a certain tolerance. After every seed point has been processed, we remove the ground from the original point cloud and perform a region growing algorithm on the remaining points. In contrast to the first step, we only use the Euclidean distance as a growing threshold.
It turned out that our approach leads to errors in situations with various ground levels. In this case the segmentation would lead to large objects, connected by the ground. To solve this problem, we again perform the algorithm on the point sets of the objects resulting from the first segmentation.

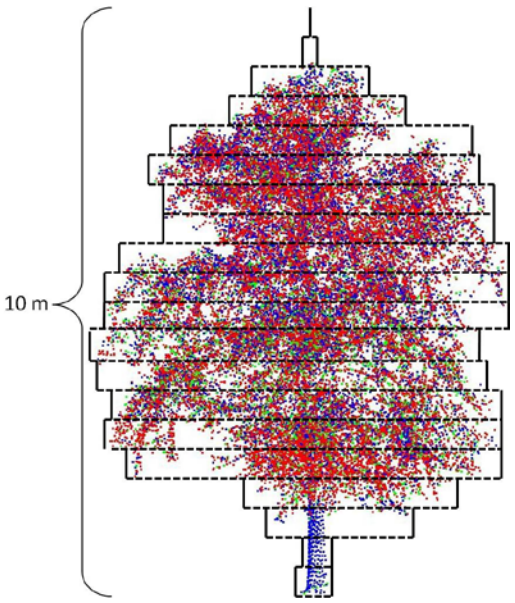

Figure 3. The point sets are divided into at maximum 20 blocks with an overall height of 10 m. For every block the average width, volume and the dimensionality factors are determined.

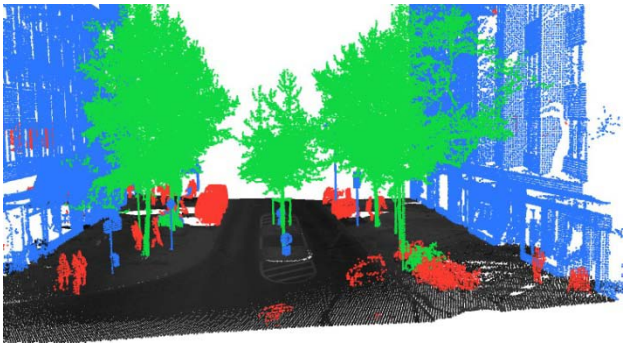

Figure 4: Point cloud with classified trees in green, detected dynamic objects in red and static objects in blue. The ground is colored by the point intensity values.

3.3 Tree classification

Trees are not dynamic objects like cars or pedestrians. Nevertheless, trees may have a bad influence on the localization step, because their appearance may change due of growth, seasonal changes or even because of wind. However, they are hard to detect by a point cloud change detection algorithm, because their basic shape remains the same. Therefore, we decided to detect them by a classification algorithm.
We use a random forest classifier with ten decision trees and a tree depth of ten nodes. The first feature is the tree height, whose calculation is straightforward. For the further features we divide the tree into 20 blocks with a block height of 0.5 m (see Figure 3). Points higher than 10 m are only considered for the calculation of the overall tree height. For every tree block we use the block width and volume as a feature. In addition, we perform a principal component analysis (PCA) for every point using the neighboring points in a radius of 1 m. We then derive the linear (a_1), planar (a_2) and scatter (a_3) dimensionality features by the eigenvalues λ_1, λ_1 and λ_3, like described in (Demantké et al., 2011) and (West et al., 2004):

$$a_1 = \frac{\lambda_1 - \lambda_2}{\lambda_1}, \quad a_2 = \frac{\lambda_2 - \lambda_3}{\lambda_1}, \quad a_3 = \frac{\lambda_3}{\lambda_1}, \tag{1}$$

with $\lambda_1 > \lambda_2 > \lambda_3$ and $a_1 + a_2 + a_3 = 1$.

Figure 3 shows the tree points colored by the dominating dimensionality factor (a_1: blue, a_2: green, a_3: red). It can be seen, that the trunk mostly consists of linear features whereas the crown consists of linear, planar and scatter parts. We calculate the mean values of the dimensionality elements for every block and add them to the feature vector. If a block contains no points, its values are set to zero.
A cross validation with training data consisting of 338 trees and 14793 other objects results in a precision of 96 % and a recall of 95 %. An example of correctly detected trees can be seen in Figure 4.
The two main reasons of false negative tree detections are occlusions and segmentation errors. Objects between the LiDAR sensor and a tree may lead to the effect that not the whole tree but only a part of it is measured by the system. The wrong shape of the object then yields a false classification. In

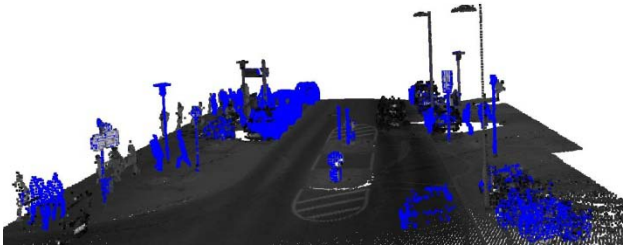

Figure 5: Segments of one point cloud (blue) are compared to points of another point cloud (colored by intensity values).

future work, an occlusion analysis could help to reduce classification errors.
Another problem are trees connected to other objects, especially building facades. If the segmentation algorithm does not divide them into separate objects then as a result the whole segmented object is not classified as a tree. We plan to implement a min-cut approach to divide the objects and improve the segmentation (Sedlacek and Zara, 2009) (Golovinskiy and Funkhouser, 2009).

3.4 Change detection

The goal of the change detection is to detect dynamic objects, like pedestrians or cars. Therefore we do not compare the whole point cloud but only the segmented objects (see 3.2) to five further comparison clouds at the same location. As they all were already aligned to the same reference cloud, they do not have to be aligned to each other. For every point of a segment we perform a radius search with a threshold of 15 cm in the comparison cloud. If for a certain percentage of segment points (here: 25 %) no neighboring points could be found for at least one segment-to-cloud comparison, the object is labeled as dynamic. An exemplary comparison is shown in Figure 5, whereby trees and (only for visualization purposes) buildings are already removed. The segments are all colored blue. Cars and pedestrians have no corresponding points in the other point cloud. Street signs do occur in both clouds and will be labeled as static. Figure 4 shows the resulting dynamic (red) and static (blue) objects. In this case, the algorithm works fine. In general, some errors occur at parking lots, occupied by the same or similar cars. A solution would be another, additional classification approach for cars, pedestrians and cyclists. Of course, other dynamic objects, like for example waste bins, may still be labeled wrongly as static.

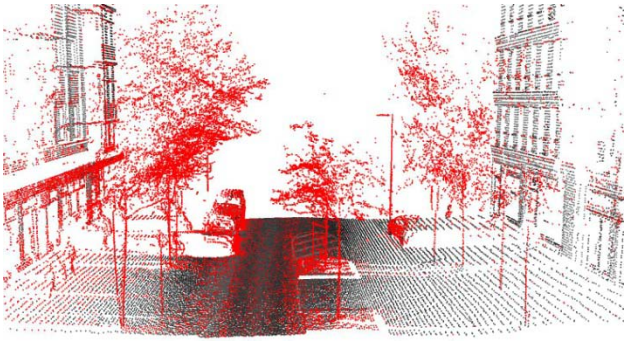

Figure 8: Classified points of the automotive LiDAR data. Dynamic points are colored red, static points are colored by their intensity value.

3.5 Image generation

After filtering dynamic objects from the point clouds, we generate the reference images by reducing the point cloud to a 2D representation. The cloud is divided to a 2D grid with a cell size of 2 cm. Exemplary images can be found in Figure 6 and Figure 7. In Figure 6 the grey value is given by the maximum height value in a grid cell, in Figure 7 the grey value is given by the mean intensity value over all datasets. Overall, we generated 907 images (height and intensity in each case) using eight different data sets for each image on the cluster computer system at the Leibniz University Hannover, Germany.
In future scenarios, instead of using one highly dense and accurate mapping system, the reference data can be recorded by every vehicle equipped with automotive LiDAR sensors. Afterwards, the data may be merged on a server to generate and update dynamic reference images.

4. LOCALIZATION

The localization is performed by matching the simulated data of an automotive LiDAR sensor to the height and intensity reference images. To improve the localization, we label every scan point as dynamic or static using a random forest classification.

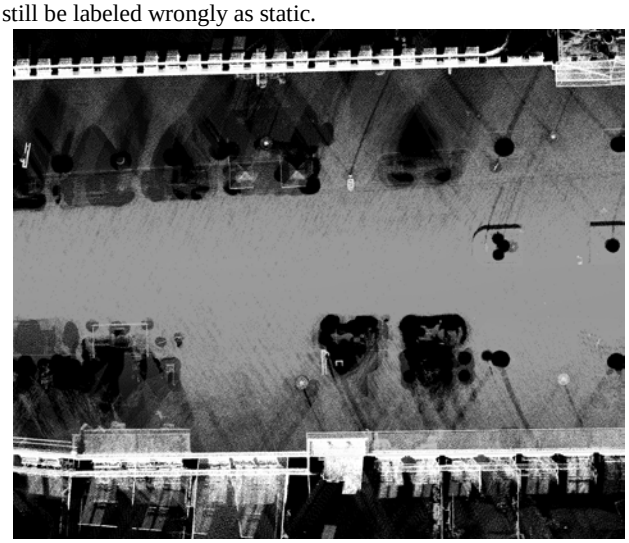

Figure 6. Resulting height reference image.

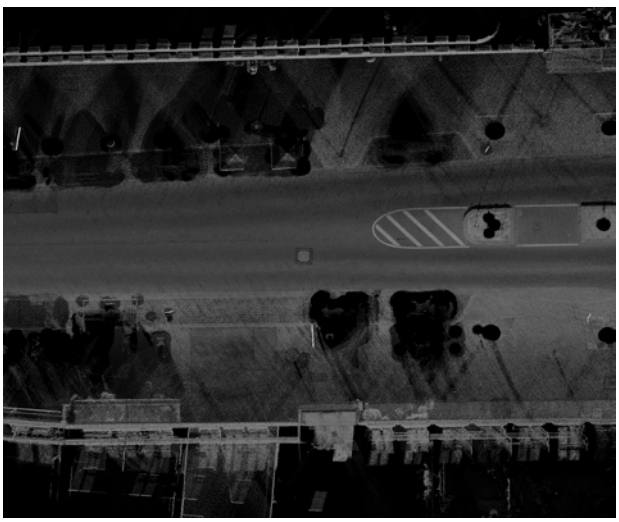

Figure 7. Resulting intensity reference image.

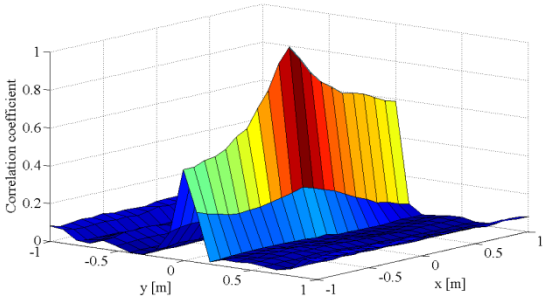

Figure 9. Correlation plot in a range of 2 m x 2 m with a resolution of 0.10 m.

4.1 Online classification

As we only want to use static points and objects in the localization, we also have to detect them in the automotive data. Thus instead of classifying the resulting point clouds in postprocessing, we classify the scan points scan line by scan line. Since we can't integrate neighboring points, the selection of spatial features is limited to one line.

Our feature vector consists of 127 elements. The first values are the position relative to the scanner, namely the height and the horizontal distance. In addition, we use the intensity value und the intensity standard deviation involving the 20 previous and following measurements. Next, we calculate the deviation of the height, the horizontal distance and the intensity values, also in a 20 points neighborhood along the scan line.
The training data is generated by using the labeled dynamic objects, detected by the tree classification and the change detection algorithm (see 3.3 and 3.4). As the street geometry in cities varies from street to street, we decided to divide our test trajectory into 100 m sectors and generated training data of about 50.000 points per sector.
The random forest classifier consists of ten decision trees and a tree depth of 25. We tested the performance of the classification using a cross validation, which achieved a recall of 97 % and a precision of 91 %. False negative detections of dynamic scan points could lead to a wrong localization result whereby false positive detections are only reducing the amount of points in the correlation. Therefore a high recall value, meaning a high completeness of dynamic points, is the major important evaluation criterion.
Figure 8 shows a point cloud with points classified as dynamic shown in red. Scan points on dynamic objects, here cars, pedestrians and trees are correctly detected. Some false positive detections occur on the lower part of the building facade, on the left walkway and on the street light in the back.

4.2 Point to image correlation

We assume that the current vehicle state **x**, which in this simplified case is only given by the position (x,y) and orientation (θ), is known within a range of ± 2 m and a heading of ± 5 degrees. To speed up computation, the transformation $\Delta\mathbf{x}$ that leads to the highest correlation to the reference images is calculated using a pyramid approach with an end resolution of 1 cm and 0.05 degrees. We do not only use one scan line in the matching process, but measurements along a certain range, in this case 20 m.
The measurements are matched to the reference images by determining the corresponding image coordinates for every scan point and comparing the measured height and intensity values to the pixel values of the reference images. As the pixel height value is always given by the highest point, we also only use the measurement with the largest height value that falls into a certain pixel while calculating the height image correlation coefficient.
The overall correlation (*Corr*) of one transformation to the reference data is given by the product of the correlation to the height image ($Corr_1$) and the correlation to the intensity image ($Corr_2$).

$$Corr = Corr_1 \cdot Corr_2 \tag{2}$$

The correlation coefficient can be calculated by the following formula, where $m_{j,i}$ stands for the single measurements and $px_{j,i}$ for the pixel values of the reference image.

$$Corr_j = \frac{\sum_{i=1}^{n}(m_{j,i}-\overline{m_j})(px_{j,i}-\overline{px_j})}{\sqrt{\sum_{i=1}^{n}(m_{j,i}-\overline{m_j})^2 \sum_{i=1}^{n}(px_{j,i}-\overline{px_j})^2}} \text{ , with } j = 1, 2. \tag{3}$$

An example plot of the correlation results in this range for the area that can be seen in Figure 4 is presented in Figure 9, whereby the *x*-axis points to driving direction. Along the driving direction the values are relatively high, compared to the *y*-axis. Mainly because of the building facades, the correlation decreases rapidly from the maximum to both sides on the *y*-axis. Note that in this figure the correlation coefficient is already normalized.
On large streets, in some cases the height variance in the image is relatively low. As a result, the height image matching is very inaccurate. In these cases, we only use the intensity reference image.

5. RESULTS

We tested our localization approach on a trajectory of 13 km length in Hannover, Germany. The reference images were aligned to the same data set that was used to generate the automotive LiDAR data. Consequently, we would expect a perfect matching of the automotive data to be zero in *x* and *y* direction as well as for the heading difference. We only considered correlation results with a deviation Δ_{2D} lower than 0.5 m, with

$$\Delta_{2D_i} = \sqrt{\Delta_{x_i}^2 + \Delta_{y_i}^2}. \tag{4}$$

If the deviation is higher than this threshold, we did not take the corresponding trajectory point into account. The completeness is indicated as the relation of trajectory points with a successful localization (Δ_{2D} < 0.5 m) to the overall number of trajectory points (907).

Figure 11 - Figure 13 show maps where the error ellipses are colored by their 2D deviation. If the 2D deviation is higher than 0.5 m, the corresponding positions are represented by a red bolt. The standard deviations are shown in Table 2 and Table 3. It can be seen, that in most cases a localization with a standard deviation of 0.048 m and 0.064° is possible. Especially in residential areas with narrow streets and buildings standing close, the localization works fine (Figure 11). In this area the completeness is 97 %. The main reasons for localization errors are trees in the reference images that are not detected by the tree classification, low structured pavements and missing height variances as shown in Figure 10. These errors often appear at large streets (Figure 12) and in this case in the city center of Hannover, see Figure 13. Here the completeness values are 91 % (large street) and 89 % (city center). The overall completeness is 93 %. Table 2 also shows the localization results without a classification of dynamic points. Hence the mean deviations are quite lower, the completeness also decreases by about five percent.

	Classification	**No classification**
σ_x	0.043 m	0.046 m
σ_y	0.021 m	0.016 m
σ_{2D}	0.048 m	0.049 m
σ_Θ	0.064°	0.056°
Completeness	0.93	0.88

Table 2: Localization results with and without a classification of dynamic points.

	Residential area	**Large street**	**Center**
σ_x	0.050 m	0.045 m	0.019 m
σ_y	0.021 m	0.022 m	0.017 m
σ_{2D}	0.054 m	0.050 m	0.026 m
σ_Θ	0.067°	0.062°	0.065°
Completeness	0.97	0.91	0.89

Table 3: Localization results separated by city areas.

The time complexity of the correlation value computation and the classification step depends on the number of the scan points and the length of the processed automotive scan segment along the trajectory. To speed up computation, we tested various sampling rates and also varied the processed segment length (see Figure 14 - Figure 17). The evaluation was performed on a Windows 7 64 bit system with a 3.4 GHz i5-3570K quad core

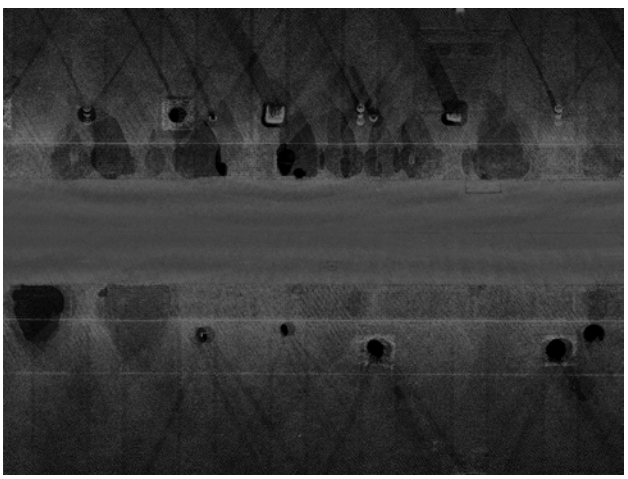

Figure 10: Intensity image of a trajectory point, where a correct localization fails.

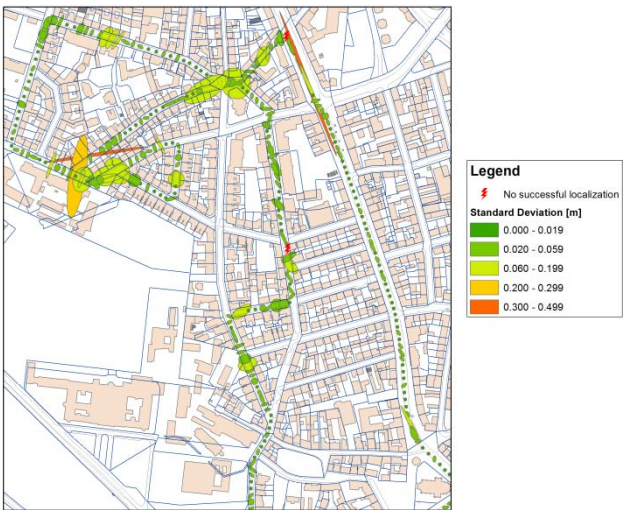

Figure 11: Localization results in a residential area with narrow streets and buildings standing close in Hannover, Germany.

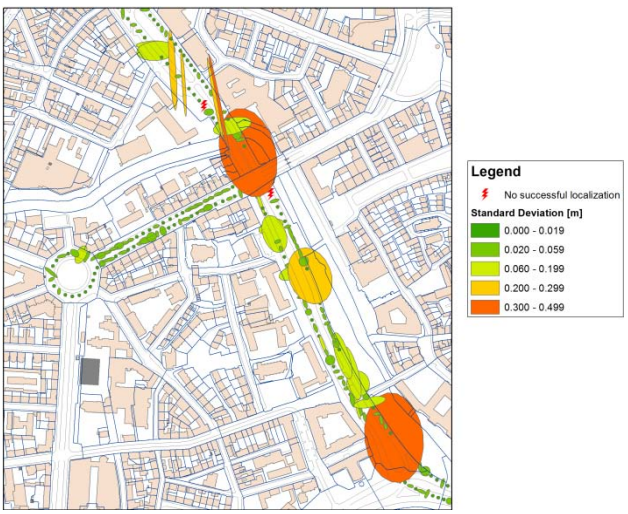

Figure 12: Localization results on a large street in Hannover, Germany.

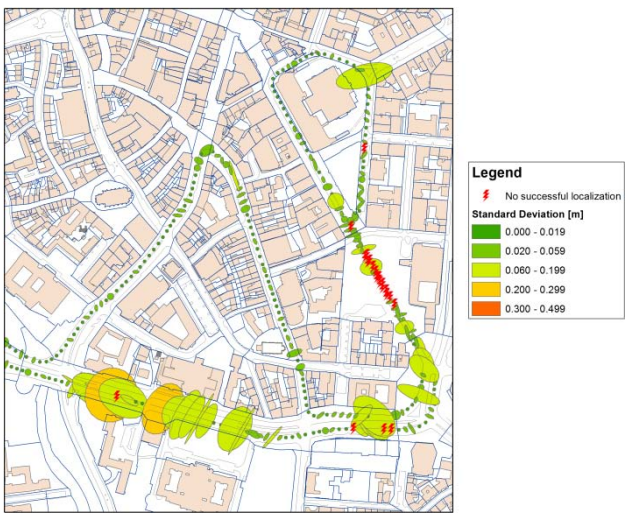

Figure 13: Localization results on a large street (bottom) and in the city center of Hannover, Germany.

processor. It can be seen, that with a higher sampling rate and a smaller analyzed scan segment the accuracy decreases slightly and the completeness decreases strongly. In contrast to the completeness and accuracy, a larger sampling rate has positive influence on the time complexity. At a sampling rate of 5 and a scan segment length of 20 m the correlation analyses with centimeter resolution can be done in less than one second.

6. CONCLUSIONS

In this work we presented a highly accurate vehicle localization approach using an automotive LiDAR sensor. The automotive data was simulated by sampling data of a highly dense and accurate Mobile Mapping System, which we also used to generate the reference data. In the future, we will use original data, e.g. gathered by a SICK LMS500 laser scanner. The reference map contains height and intensity images with a fixed resolution of 2 cm. These reference images are 2D representations of the point clouds, whereby we only considered static objects. Dynamic objects, like trees, cars or pedestrians were filtered by a change detection and a tree classification algorithm. In the majority of cases both algorithms worked well. In the change detection algorithm false positive detections appeared if two objects were connected, like a bicycle standing at a pole light. False negatives appeared if two similar or the same objects occurred at the same positions. Here a classification approach could improve the results. The tree classification did not work in cases where trees are connected to other large objects like building facades. To solve this problem, we plan to implement a min-cut algorithm, which could also reduce the number of false positive dynamic object detections.
The online localization was performed by matching the automotive LiDAR scan points to the reference images. We used a point to image correlation using the height and the intensity values. We also classified the scan points into static and dynamic points to improve the matching. It turned out, that especially in residential areas, the localization works well, with a completeness of about 97 % and an accuracy of 5.4 cm (position) and 0.067° (heading). The overall completeness along a 13 km trajectory is 93 % with a standard deviation of 4.8 cm and 0.064°. The main reasons for localization errors are low structured reference images and remaining trees in the reference images.
We also evaluated the time complexity of the algorithm by varying the sampling rate of the scan points and the length of the analyzed segment at one trajectory point. The computation time for a correlation analyses with a resolution of 1 cm varies between some seconds to about half a second.

ACKNOWLEDGEMENTS

We acknowledge the support of the computer cluster system team at the Leibniz University Hannover, Germany in the production of this work.

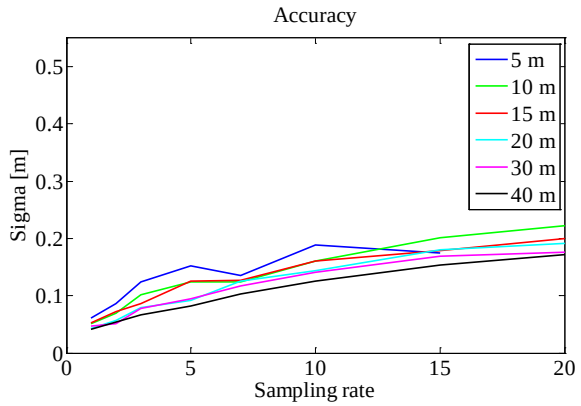

Figure 14: Accuracy (σ2D) at various sampling rates and lengths of processed scan segments.

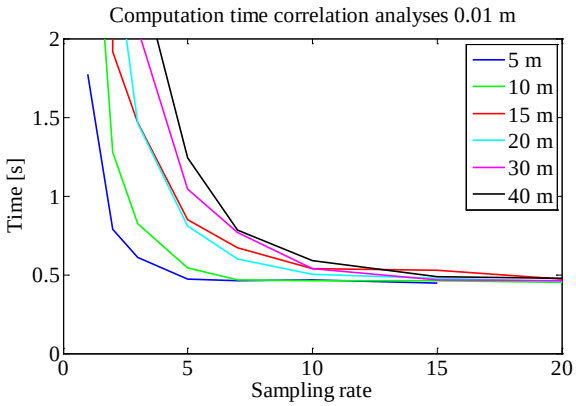

Figure 16: Computation time of the correlation analyses at various sampling rates and lengths of processed scan segments at a resolution of 0.01 m.

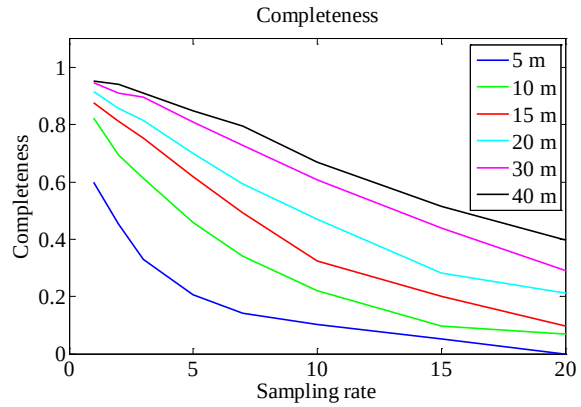

Figure 15: Completeness at various sampling rates and lengths of processed scan segments.

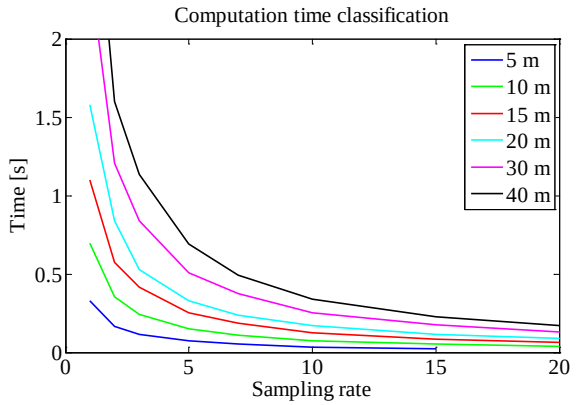

Figure 17: Computation time of the classification step at various sampling rates and lengths of processed scan segments.

REFERENCES

Adams, R., Bischof, L., 1994. Seeded region growing. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 16(6), pp. 641-647.

Aijazi, A. K., Checchin, P., Trassoudaine, L., 2013. Detecting and updating changes in LiDAR point clouds for automatic 3d urban cartography. *ISRPS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2(5), W2.

Besl, P., McKay, N. D., 1992. Method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence 14*, pp. 239-254.

Breiman, L., 2001. Random forests. *Machine Learning*, 45(1). Springer, Heidelberg, pp. 5-32.

Brenner, C., Hofmann, S., 2012. Evaluation of automatically extracted landmarks for future driver assistance systems. *Advances in Spatial Data Handling and GIS*. Springer, Heidelberg, pp. 169-181.

Demantké, J., Mallet, C., David, N., Vallet, B., 2011. Dimensionality based scale selection in 3D LiDAR point clouds. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38(5), W12.

Golovinskiy, A., Funkhouser, T., 2009. Min-cut based segmentation of point clouds. *IEEE International Conference on Computer Vision (ICCV Workshops)*, pp. 39-46.

Lategahn, H., Beck, J., Stiller, C., 2014. DIRD is an illumination robust descriptor. *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, pp. 756-761.

Levinson, J., Montemerlo, M., Thrun, S., 2007. Map-based precision vehicle localization in urban environments. *Proceedings of Robotics: Science and Systems*.

Maddern, W., Pascoe, G., Newman, P., 2015. Leveraging experience for large-scale LiDAR Localization in changing cities. *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1684-1691.

Nothdurft, T., Hecker, P., Ohl, S., Saust, F., Maurer, M., Reschka, A., Bohmer, J., 2011. Stadtpilot: First fully autonomous test drives in urban traffic. *Proceedings of 14th International IEEE Conference on Intelligent Transportation Systems*, pp. 919-924.

Pomerleau, D., Jochem, T., 1996. Rapidly adapting machine vision for automated vehicle steering. *IEEE Intelligent Systems*, 2, pp. 19-27.

Pu, S., Rutzinger, M., Vosselman, G., Oude Elberink, S., 2011. Recognizing basic structures from Mobile Laser Scanning data for road inventory studies. *Journal of Photogrammetry and Remote Sensing*, 66(6), pp. 28-39.

Qu, X., Soheilian, B., Paparoditis, N., 2015. Vehicle Localization using mono-camera and geo-referenced traffic signs. *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, pp. 605-610.

Riegl Laser Measurement Systems GmbH, 2011. Riegl VMX-250, Horn, Austria. Available at: http://products.rieglusa.com/Asset/10_DataSheet_VMX-250_11-11-2011.pdf.

Rusu, R. B. and Cousins, S., 2011. 3D is here: Point Cloud Library (PCL). *IEEE International Conference on Robotics and Automation (ICRA)*.

Schlichting, A., Brenner, C., 2014. Localization using automotive laser scanners and local pattern matching. *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, pp. 414-419.

Sedlacek, D., Zara, J., 2009. Graph cut based point-cloud segmentation for polygonal reconstruction. *Advances in Visual Computing*. Springer, Heidelberg, pp. 39-46.

Wen, X., Vallet, B., Brédif, M., Paparoditis, N., 2015. Street environment change detection from mobile laser scanning point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing 107*, pp. 38-49.

West, K. F., Webb, B. N., Lersch, J. R. Pothier, S., Triscari, J. M., Iverson, A. E., 2004. Context-driven automated target detection in 3D data. *Defense and Security*, International Society for Optics and Photonics, pp. 133-143.

Wu, B., Yu, B., Yue, W., Shu, S., Tan, W., Hu, C., Huang, Y., Wu, J., Liu, H., 2013. A voxel-based method for automated identification and morphological parameters estimation of individual street trees from Mobile Laser Scanning data. *Journal of Phtogrammetry and Remote Sensing*, 5(2), pp. 584-611.

Yoneda, K., Tehrani, H., Ogawa, T., Hukuyama, N., Mita, S., 2014. LiDAR scan feature for localization with highly precise 3-d map. *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, pp. 1345-1350.

Yoneda, K., Yang, C., Mita, S., Okuya, T., Muto, K., 2015. Urban road localization by using multiple layer map matching and line segment matching. *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, pp. 525-530.

Ziegler, J., Lategahn, H., Schreiber, M., Keller, C. G., Knoppel, C., Hipp, J., Haueis, M., Stiller, C., 2014. Video based localization for bertha. *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, pp. 1231-1238.