2013 IEEE International Conference on Robotics and Automation (ICRA)
Karlsruhe, Germany, May 6-10, 2013

Joint Self-Localization and Tracking of Generic Objects in 3D Range Data

Frank Moosmann[1] and Christoph Stiller[1]

***Abstract*—Both, the estimation of the trajectory of a sensor and the detection and tracking of moving objects are essential tasks for autonomous robots. This work proposes a new algorithm that treats both problems jointly. The sole input is a sequence of dense 3D measurements as returned by multi-layer laser scanners or time-of-flight cameras. A major characteristic of the proposed approach is its applicability to any type of environment since specific object models are not used at any algorithm stage. More specifically, precise localization in non-flat environments is possible as well as the detection and tracking of e.g. trams or recumbent bicycles. Moreover, 3D shape estimation of moving objects is inherent to the proposed method. Thorough evaluation is conducted on a vehicular platform with a mounted Velodyne HDL-64E laser scanner.**

I. INTRODUCTION

Two main tasks can be identified for a perception system of a robot: precise self-localization, often performed simultaneously with mapping (SLAM), and the detection and tracking of moving objects (DATMO). While most methods from literature treat the two tasks as being independent, a joint estimation scheme is introduced in this contribution.

A. Self-Localization

The problem of localization is usually understood as the estimation of the robot's pose, i. e. position and orientation. The frame of reference thereby varies. Some approaches seek a global estimate using GPS or global landmarks. Others refer to the relative motion of the robot, specifying the pose w.r.t. the starting point – the goal of this work.

The most widely spread algorithms for range sensors follow the principle of simultaneous localization and mapping (SLAM) [23]. Though the last decade showed a trend towards probabilistic techniques, the computational complexity with 3D data in outdoor environments notably shifts the used method types in favor of scan-matching [18], [11], [3], [16].

Most SLAM methods only estimate the motion of the vehicle w.r.t. a static scene and usually average out objects with different motion. For low outlier ratios these registration methods provide good results. A high portion of moving objects, however, might cause these methods to fail. Only few SLAM methods try to simultaneously detect and track moving objects [26], [25]. Unfortunately, their computational efficiency and robustness in the 3D real world was not yet shown.

[1]Both authors are with the Institute of Measurement and Control, Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany frank.moosmann at kit.edu

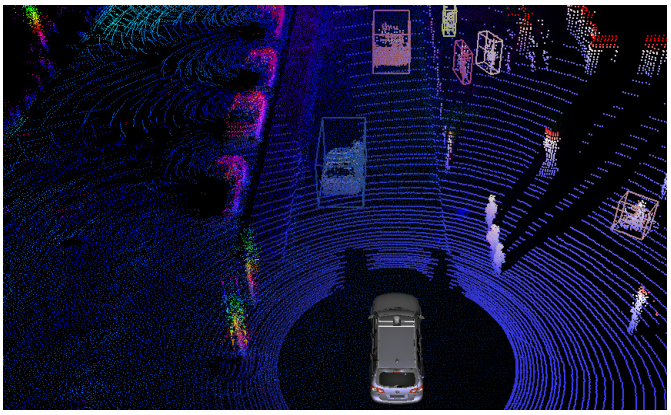


Figure 1. Result of the proposed method: The mapped static environment colored by altitude (left) and tracked moving objects highlighted with a unique coloring in the sensor data (right).

B. Multi Target Tracking

The problem of multi target tracking is usually understood as the task to detect a set of objects in the environment and to characterize them by their position, orientation, extent, and velocity. Existing solutions frequently decompose the problem into two independent stages. The first stage detects objects independently for each point in time. State of the art methods mostly train classifiers for the detection of specific object classes like cars or humans [19]. Only few methods employ generic segmentation methods to detect any kind of object that sticks out well from background [22]. The second stage associates the detections over time in order to get continuous tracks, i. e. estimates of the objects intrinsic state like e. g. position and velocity. Possible generic solutions for this stage are given in [1]. When using dense data, the association of measurements can be ambiguous especially when several detections per object exist. To overcome ambiguities, solutions like fuzzy segmentation [21], segment matching [8] or appearance learning [10] have been proposed.

This two-stage approach has been applied to various kind of sensors, from 2D laser scanners [17] over 3D laser scanners [12], [19] up to time-of-flight cameras [9]. Its major drawback is the dependence on a reliable and repeatable object detector. To the best of our knowledge, no approach exists that can robustly track arbitrary objects.

A completely different methodology is *track before detect* [7]. Sensor data is quantized e. g. at fixed image columns [20] or at fixed intervals in the horizontal plane [4]. Although results seem very promising, finding a good grouping of the tracked partitions, which corresponds to the detection, is still

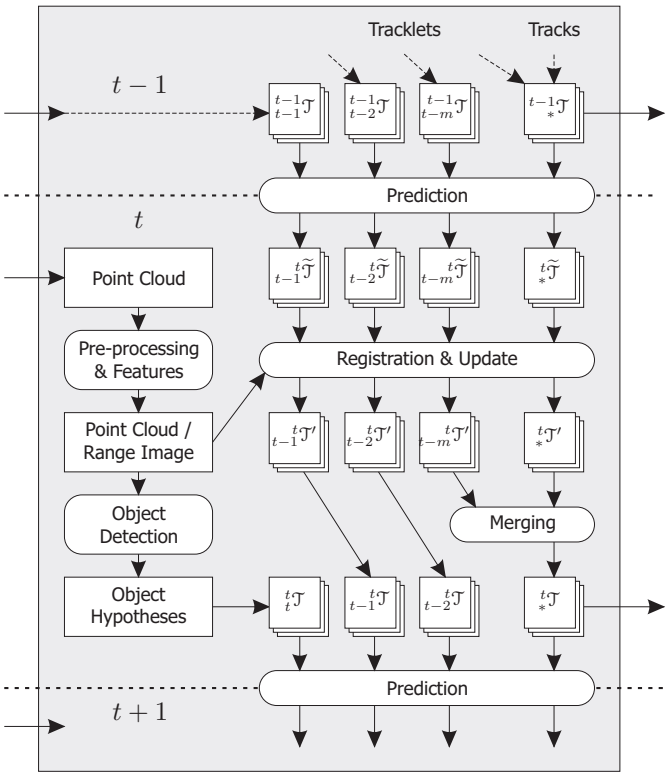


Figure 2. Overview of the proposed method.

an open issue.

One step further is the idea to optimize the partitioning of data (which can here be regarded as object detection) and the motion estimation together. However, the proposed solutions [13], [24] are computationally too complex to be applied in real-time on ordinary computers within the next years.

Hence, all successful object tracking methods seem to be either 2D or model based, which requires manual model construction and model selection through classification.

This work proposes a novel idea for the joint solution of both problems. The combination of a dynamic data partitioning with *track before detect* techniques allows to track arbitrary objects. By treating the static scene as object, mapping is applied to both, moving objects and the static scene, in a unique way. Experiments conducted in a vehicular environment show the applicability to 3D environments with both tracking and self-localization performed with full 6 degrees of freedom.

II. PROPOSED METHOD

Throughout this work, a left superscript ${}^{t}x$ denotes the current time index and a left subscript ${}_{t}x$ the measurement time. For clarity, these are only specified if necessary. All computations are made w.r.t. the sensor coordinate system. No fixed world coordinate frame is used.

A. Overview

Input to the algorithm at each time t is a set of range measurements represented as 3D points ${}^{t}\mathcal{P} = \left\{(x, y, z)^{\mathrm{T}}\right\}$. This point cloud is preprocessed, features are calculated, and

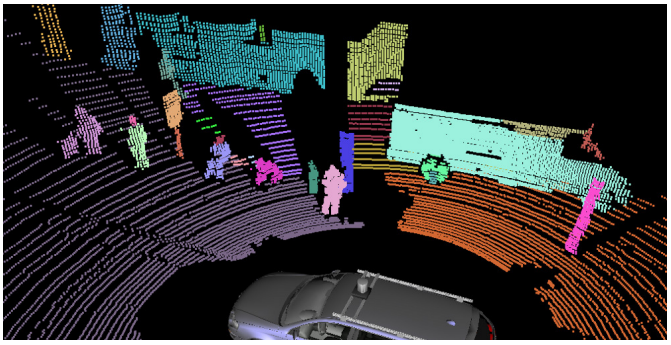


Figure 3. Each segment, indicated by a unique color, is turned into an object hypothesis and verified by tracking across m frames.

object hypotheses are generated. Each object hypothesis ${}^{t}\mathcal{S}$ is turned into a tracklet ${}^{t}_{t}\mathfrak{T}$. Hence, the set of tracklets ${}^{t}_{t}\mathcal{T} = \{{}^{t}_{t}\mathfrak{T}\}$ is created. The only exception is the initialization in the very first frame. Object detection is skipped and one single (static-scene-) track is created from all measurements. The track(let)s are predicted and updated across m frames and finally merged with existing tracks. Note that the registration step uses the unsegmented point cloud as reference, which is in contrast to most existing tracking methods. Output of the algorithm is a set of tracks which includes the track of the static scene. Hence, the sensor motion w.r.t. a fixed world coordinate frame can be deduced as the inverse static scene motion.

B. Pre-processing and Features

The input point cloud is smoothed and two features are calculated for each point $\boldsymbol{p}_i \in \mathcal{P}$: a normal vector $\boldsymbol{n}_i = (n_x, n_y, n_z)^{\mathrm{T}} : \|\boldsymbol{n}_i\| = 1$, representing a local surface plane, and a so-called flatness value $f_i \in [0, 1]$ which characterizes how appropriate the approximation by a surface plane is. The exact calculations are taken from [16], where the normal vectors $\mathcal{N} = \{\boldsymbol{n}_i\}$ are denoted as N and the flatness-values $\mathcal{F} = \{f_i\}$ are denoted as C.

C. Object Detection

The aim of this work is to track any kind of object that is moving. As a consequence, object class specific detectors cannot be used. Better suited are segmentation methods, that split the set of input points $\mathcal{P}$, represented by the set of indices $\mathcal{S} = \{i\}$, into segments $\mathcal{S}_g \subseteq \mathcal{S}, \bigcup_g \mathcal{S}_g = \mathcal{S}, \forall g, h, g \neq h : \mathcal{S}_g \bigcap \mathcal{S}_h = \varnothing$, where each segment corresponds to one object hypothesis. Any meaningful segmentation method can be employed within the proposed tracking framework; here the so-called *local convexity* criterion is used, which was introduced in [15] and improved in [14], also see Fig. 3.

D. Tracking

A tracklet $\mathfrak{T}_g$ is created from each object hypothesis $\mathcal{S}_g$ with a minimum size and can be regarded as object hypothesis in the time domain. A local object coordinate system O_g is introduced, as depicted in Fig. 4. It is specified by a *pose* vector

$$\boldsymbol{\rho}_g = (\phi, \theta, \psi, x, y, z)^{\mathrm{T}} \tag{1}$$

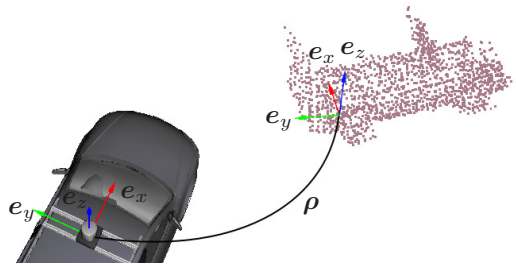


Figure 4. The pose $\boldsymbol{\rho}$ of the state vector defines the position and the orientation of a track coordinate system (top) w.r.t. the scanner coordinate system (bottom). The track appearance is stored as point cloud (violet) with normal vectors and flatness values (both not shown) relative to the track coordinate system.

which defines its orientation and position w.r.t. the sensor coordinate system S. The pose and its derivative constitute the *state* of the tracklet:

$$\boldsymbol{x}_g = \begin{pmatrix} \boldsymbol{\rho}_g \\ \dot{\boldsymbol{\rho}}_g \end{pmatrix} = (\phi, \theta, \psi, x, y, z, \dot{\phi}, \dot{\theta}, \dot{\psi}, \dot{x}, \dot{y}, \dot{z})^{\mathrm{T}} \qquad (2)$$

The 3D points $\mathcal{P}_g \subseteq \mathcal{P}$, the normals $\mathcal{N}_g$, and the flatness values $\mathcal{F}_g$ constitute the *appearance* of the tracklet. They are stored relative to the object coordinate system O_g, see Fig. 4. In total, a tracklet is defined by its state and appearance:

$$\mathfrak{T}_g = (\boldsymbol{x}_g,\ \mathcal{P}_g^{\mathrm{O}_g},\ \mathcal{N}_g^{\mathrm{O}_g},\ \mathcal{F}_g) \qquad (3)$$

It is worth noting that our method for track estimation thus includes the 3D reconstruction of the shape of moving objects.

In the first m frames the appearance of a tracklet is kept constant. On the contrary, the state is re-estimated for each new incoming frame within the *Prediction* and *Update* step of Fig. 2. This makes the appearance move along with the coordinate frame defined by the state. A Kalman Filter with constant velocity model[1] is employed upon the state vector which can express any rigid motion. *Prediction* corresponds exactly to the prediction step of the Kalman Filter. *Registration and update* is performed as in [14] by aligning the track's appearance point cloud with the full input point cloud by means of the ICP algorithm. The predicted pose thereby serves as initial pose of this iterative algorithm. In case the average flatness value of the tracklet exceeds some threshold, the point-to-plane ICP [6] is used, otherwise the point-to-point variant [2]. The measurement covariance for the Kalman Filter update is calculated with the method of [5]. One special treatment is made for the static scene track: instead of registering the track appearance against the input data, the input data is registered against the track appearance as in [16]. This makes the approach faster and more robust and allows for sensor motion compensation.

Note that up to this point, no associations are made yet between the tracklets, since registration is performed with the full input data. Relations are established only in the track management stage described next.

[1]More specific and possibly non-linear models could of course be used for specific object classes to extend and hence improve the method.

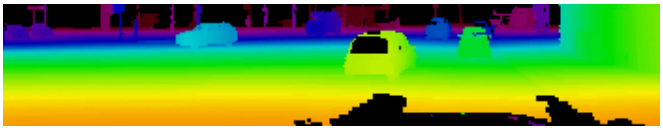


Figure 5. Input points as virtual range image, colored by distance.

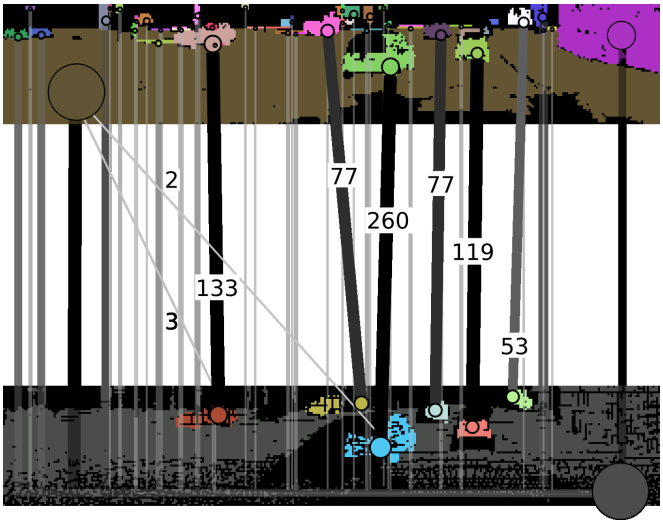


Figure 6. Associations between new tracklets ${}^{t}_{t}\mathcal{T}$ (upper image) and tracks ${}^{t}_{*}\mathcal{T}$ (lower image) are established by overlaying their projections and counting the number of pixels they overlap. Shown are the association strengths for moving objects (in the lower figure); the edges that are not labeled are associations with the static track (gray).

E. Track-Management

This section essentially describes the *Merging* step in Fig. 2, which also handles the transition of tracklets to tracks. Both describe moving objects by their state and appearance. The difference is conceptual only: tracklets are track hypotheses that, after successful verification, can become tracks. As a consequence, tracks are predicted and updated exactly like tracklets.

The merge step takes as input the current set of tracks ${}^{t}_{*}\mathcal{T}'$ and the set of tracklets ${}_{t-m}{}^{t}\mathcal{T}'$ that was (independently) registered across m frames and produces an updated set of tracks ${}^{t}_{*}\mathcal{T}$. First, any track that moved out of the field of view is removed from ${}^{t}_{*}\mathcal{T}'$. Then, each tracklet is compared with the existing tracks and one of three actions is taken:

1) The tracklet is kept and added to the set of tracks if the tracklet was successfuly registered over the last m frames and if it represents an object with a motion different to all existing tracks.
2) The tracklet is merged with the track in case a track on the same object already exists.
3) The tracklet is discarded if none of the above two cases is true.

In all three cases, tracklets are inherently associated and compared with existing tracks. Fig. 6 illustrates the efficient method used to determine these associations: The appearances of all existing tracks and tracklets are projected to two virtual

range images and the number of overlapping pixels determines the association strength a_{gh} between tracklet ${}_{t-m}^{t}\mathfrak{T}_g$ and track ${}_{*}^{t}\mathfrak{T}_h$.

To decide upon the three cases, the tracklet ${}_{t-m}^{t}\mathfrak{T}_g$ is characterized by a feature vector (see Sec. IV). Several characteristics are therefore calculated. Among others is the motion histogram $\boldsymbol{m} = (m_1, m_2, m_3, m_4)^{\mathrm{T}}$. For the tracklet that moved from ${}^{t-m}\boldsymbol{\rho}_g$ to ${}^{t}\boldsymbol{\rho}'_g$, it summarizes how many appearance points moved perpendicular to their normal vector (m_1), aslant to it (m_2 and m_3), and along the normal vector (m_4). This effectively characterizes how reliable motion estimation is, since motion perpendicular to the normal vector is, generally, unreliable. Furthermore, the tracklet is compared to each associated track ${}_{*}^{t}\mathfrak{T}'_h$ with association strength $a_{gh} > 0$. Therefore, the motion of the associated track ${}_{*}^{t}\mathfrak{T}_h$ within the last m frames is applied to the tracklet ${}_{t-m}^{t}\mathfrak{T}'_g$:

$$ {}^{t}\boldsymbol{\rho}''_{g,h} = {}^{t-m}\boldsymbol{\rho}_g + ({}^{t}\boldsymbol{\rho}'_h - {}^{t-m}\boldsymbol{\rho}_h) \tag{4} $$

In case both the tracklet and the track referred to the same object and tracking was successful, ${}^{t}\boldsymbol{\rho}''_{g,h}$ should be very similar to ${}^{t}\boldsymbol{\rho}'_g$. The ICP energy $e_g({}^{t}\boldsymbol{\rho}''_{g,h})$ is calculated using both the Euclidean point-to-point distance [2] and the projective point-to-plane distance [6]. These errors are denoted $e_{g,h,2}$ and $e_{g,h,P}$ in the following as opposed to $e_{g,2}$ and $e_{g,P}$, the errors for the original pose ${}^{t}\boldsymbol{\rho}'_g$. Based on these errors the associated tracks causing minimum error can be determined as well as the track with maximum association strength

$$ h_2 = \underset{h:a_{g,h}>0}{\arg\min}\{e_{g,h,2}\} \tag{5} $$

$$ h_P = \underset{h:a_{g,h}>0}{\arg\min}\{e_{g,h,P}\} \tag{6} $$

$$ h_a = \underset{h}{\arg\max}\{a_{g,h}\} \tag{7} $$

Note that h_2, h_P, and h_a are not necessarily different. The features are gathered within a 52-dimensional feature vector $\boldsymbol{f}_g$, detailed in the appendix. A multi-class support vector machine (SVM) with RBF kernel is used to classify the feature vector in order to decide upon the three cases.

In case a tracklet is kept as new track, the tracklet's appearance is removed from all associated tracks and the tracklet is added to the set of tracks. This implicitly handles track-splits.

In case a tracklet is to be merged with an existing track, the corresponding track still has to be determined. This is performed by calculating for each associated track h a score s_{gh} and choose the track with the highest score. The score is calculated as linear combination of a second feature vector:

$$ s_{gh} = (1 \ \boldsymbol{f}_{gh}{}^{\mathrm{T}}) \cdot \boldsymbol{w} \tag{8} $$

The feature vector $\boldsymbol{f}_{gh}$ is similar to $\boldsymbol{f}_g$ and is detailed in the appendix. The parameter vector $\boldsymbol{w}$ is determined by optimization on a labeled training set. When merging, the state of the associated track remains unchanged. Only the appearance of the tracklet is added to the track. The algorithm for accumulating the appearance is taken from [16]. There, flat areas are contracted to yield sharper surface representations. This so-called *moving object mapping* (MOM) not only makes the results nicer, it also improves the registration result.

Table I
CLASSIFICATION RESULTS ON A LABELED DATA SET FOR TWO DIFFERENT PARAMETER SETTINGS OF THE CLASSIFIER.

	Decision variant A			Decision variant B		
	Keep	Merge	Ignore	Keep	Merge	Ignore
Keep	23	14	89	103	13	10
Merge	0	12208	612	196	12516	108
Ignore	1	208	3614	200	567	3056
Accuracy	94.49%			93.48%		

As compared to [16], one further step is added to process the appearances. This is particularly relevant for non-rigid object in order to avoid tracking inaccuracies. In the projection step illustrated in Fig. 6, each appearance point is removed from the track that yields a closer range value than the range value at the pixel of the current sensor data, see Fig. 5. As consequence, the appearance can adapt to non-rigid objects like pedestrians.

III. RESULTS

The proposed algorithm is evaluated on data captured with a Velodyne HDL-64E laser scanner. The sensor, a 64-beam laser scanner, is mounted on top of a car and yields a 360° view of the environment, as illustrated in Fig. 9. We set $m = 3$ through all the experiments.

The first stage of evaluation concerns the localization precision. Since in static scenes the proposed algorithm for localization equals the algorithm presented in [16], the results are transferable. Two scenarios were evaluated that both represent loops in a non-flat urban environment. These loops can be used to evaluate drift, i. e. the localization imprecision that increases with traveled distance. In average, a position error of 2.66 m after a 1 km drive was determined. This value can be regarded as very low and is about an order of magnitude lower than for common camera-based techniques. More details and discussions are given in [16].

The evaluation of object detection and tracking proceeds in several stages. First, the classifier for track management is evaluated. This classifier decides for each tracklet, i. e. track hypothesis, whether it is to be merged with an existing track, kept as new track, or ignored. Four-fold cross-validation is applied on a dataset that was set-up and labeled manually. The classification accuracy reaches the values listed in table I. The two decision variants correspond to different weightings of the classes during training. With this weighting, the SVM can be pulled towards favoring certain decisions. Variant A favors the ignorance of tracklets, which yields a higher precision but lower recall of the tracker. On the contrary, variant B yields a lower precision but higher recall. Many alternative variants exist, most of them with a classification accuracy between 90% and 95%. For further experiments, variant A is selected.

In order to evaluate the quality of tracking, an experiment was conducted in real traffic using a second car, denoted as

Figure 8. Moving Object Mapping: Appearance of a car accumulated over time (from left to right). Initial points are depicted with double size.

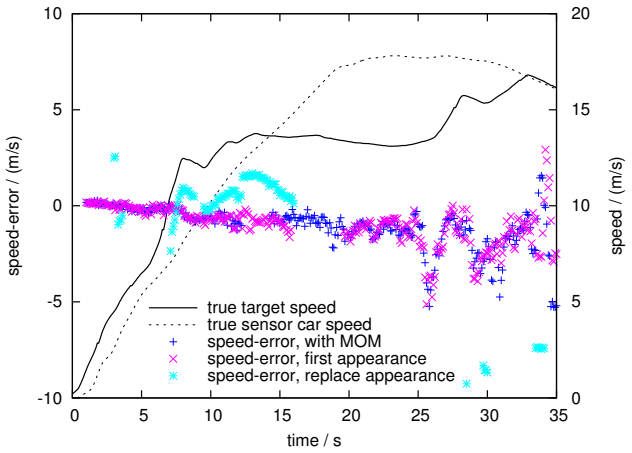


Figure 7. Tracking quality assessed by using a second car, denoted *target car*. Shown are the speed-profiles of both cars and the speed error as difference between the estimated speed (by the tracker) and the true target speed (measured by DGPS/IMU) for different tracking strategies. Missing values indicate a temporary failure of the tracking method.

Table II
TRACKING STATISTICS FOR SPEED COMPARISON EXPERIMENT OF FIG. 7 GENERATED WITHOUT OUTER 10% QUANTILES.

	nb. of tracks	speed error in m/s median	mean	std-deviation
with MOM	2	-0.84	-0.96	±1.16
first appearance	4	-0.82	-1.04	±1.29
replace appearance	12	-10.62	-7.69	±10.27

target car. This target car starts in front of the sensor car, accelerates, and gets overtaken by the sensor car after 26 s. The speed-profiles (measured by DGPS/IMU) as well as the speed-errors are depicted in Fig. 7, some characteristic values are listed in table II. Evident is the advantage of MOM over using only the appearance of one frame. The speed-error is within an acceptable range and the track gets lost only once. Especially during the overtaking maneuver, the car is continuously tracked because the appearance smoothly adapts to the new viewpoints. This adaption is well illustrated in Fig. 8. Using constantly the first appearance leads to three track losses and a slightly higher speed error. Replacing the appearance each frame leads to the worst results. As this technique causes the track to drift, speed errors are high and the track gets lost 11 times.

Additional experiments were conducted around intersections with many moving objects. A video and the data is available on *www.mrt.kit.edu/z/publ/download/velodynetracking/*. Various types were successfully tracked: pedestrians (with rolling case), cyclists, cars, vans, trucks, trams. Fig. 9 shows some tracking results at a big intersection in the city of Karlsruhe. Most moving objects are immediately detected, some slowly moving pedestrians with a short delay. Most tracks are stable, i. e. tracking is successful until the object moves out of view. This is shown by Fig. 10 that lists the distribution of track lengths across the sequence. Cars that move in parallel to the sensor car are tracked for the whole time of movement, i. e. 30 seconds. Most other objects are tracked for several seconds, even in areas where the objects are partly occluded.

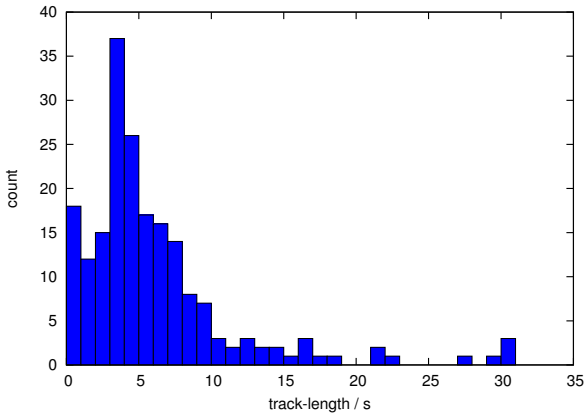


Figure 10. Track lengths on the sequence illustrated in Fig. 9 (total 50 s).

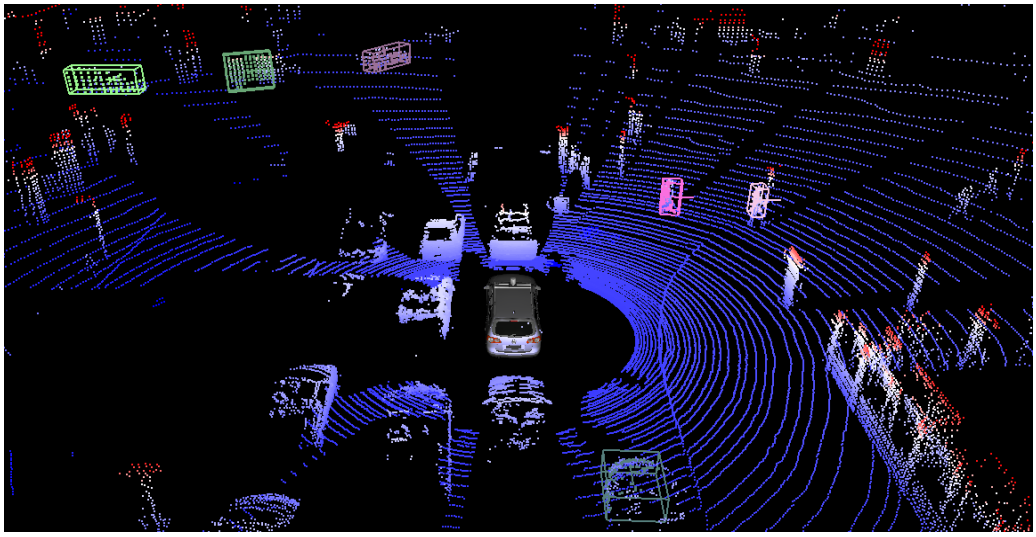
IV. CONCLUSIONS

A novel approach was presented for self-localization and mapping combined with moving object tracking in dense range data. Tracking and mapping was applied to both object hypotheses and the static scene identically. Thus, 3D shape estimation of moving objects is inherent to the proposed method. A classification-based track management was introduced for track verification, merging, and splitting. The applicability of the method was shown for a vehicular platform in a crowded city environment. But these are not the limits of the approach. Since object models were kept generic and tracking is performed in full 3D, the approach is applicable to other sensors and in other application areas, too.

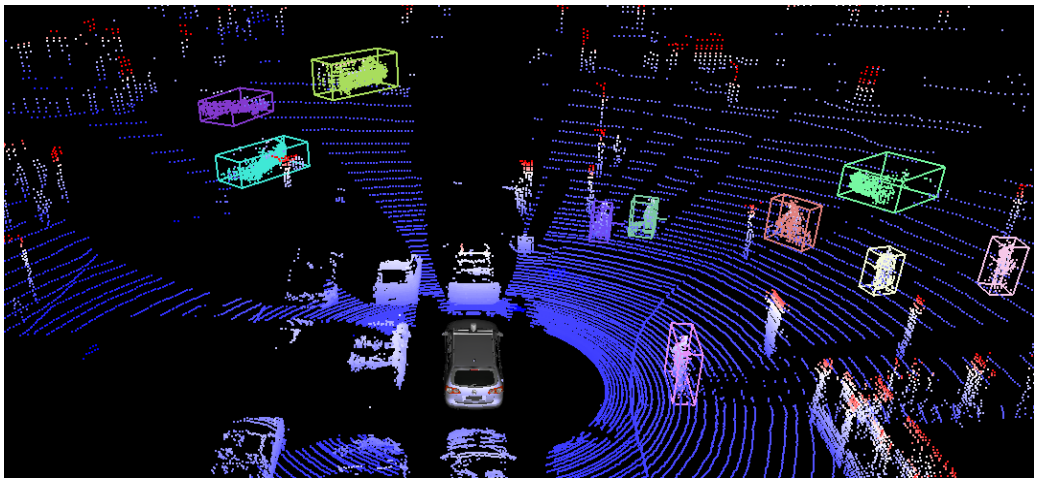
APPENDIX

Let $\log_p(x) := \log(1 + \max\{0, x\})$.

The 52-dimensional feature vector $\boldsymbol{f}_g$ is composed as follows: $\boldsymbol{f}[1] \in \{0, 1\}$ is 1 if the last measurement was successful and 0 otherwise. $\boldsymbol{f}[2] \in \{0, 1, 2, 3\}$ is



(a) After verification interval, t = 0.4 s. Nearly all moving objects are immediately detected and tracked.



(b) t = 7 s. Moving Object Mapping helps bridging occluded areas, e. g. for the turquois car.

(c) t = 26,4 s. Tracking is successful in a wide viewing range.

Figure 9. Tracking results: The sensor data is colored from blue (ground) over white (sensor level) to red (above the sensor). Each tracked object is displayed by a cuboid and the appearance points in a unique color. The cuboid, or bounding box, is calculated in a post-processing step from the appearance.

equal to the number of measurement failures. $\boldsymbol{f}[3] = \log_p \left\| ({}^t\boldsymbol{x}'_g - {}^{t-1}\boldsymbol{x}_g) - ({}^{t-1}\boldsymbol{x}_g - {}^{t-2}\boldsymbol{x}_g) \right\|$ characterizes if the two last relative movements were approximately the same. $\boldsymbol{f}[4] = e_{g,P}$. $\boldsymbol{f}[5] = e_{g,2}$. $\boldsymbol{f}[6] \in \{0,1,2,3\}$ characterizes when the last successful measurement was made. $\boldsymbol{f}[7] = \log_p(|\mathcal{P}_g|)$. $\boldsymbol{f}[8]$ characterizes for the bin of the motion-histogram with the highest value the average move in meters within the last m frames in direction of the normal vector. $\boldsymbol{f}[9] = \log_p(m_4)$. $\boldsymbol{f}[10] = \log_p(m_3)$. $\boldsymbol{f}[11] = \log_p(m_2)$. $\boldsymbol{f}[12] = \log_p(m_4 + m_3)$. $\boldsymbol{f}[13] = m_4/|\mathcal{P}_g|$. $\boldsymbol{f}[14] = (m_4+m_3)/|\mathcal{P}_g|$. $\boldsymbol{f}[15] = \sum_h \min\{1, a_{g,h}\}$. $\boldsymbol{f}[16] = \log_p(\sum_h a_{g,h})$.

The rest of the feature vector is three times a 12-dimensional vector for each of the associated tracks h_2, h_P, and h_a. Exemplary for h_2: $\boldsymbol{f}[17] \in \{0,1\}$ is 1 if the last measurement of h_2 was successful and 0 otherwise. $\boldsymbol{f}[18] = \log_p(a_{g,h_2})$. $\boldsymbol{f}[19] = a_{g,h_2}/\sum_h a_{g,h}$. $\boldsymbol{f}[20] = a_{g,h_2}/\max_h a_{g,h}$. $\boldsymbol{f}[21] \in \mathbb{N}$ characterizes when the last successful measurement of h_2 was made. $\boldsymbol{f}[22] \in \mathbb{N}$ holds the age of h_2, limited by some upper bound. $\boldsymbol{f}[23] = \log_p(|\mathcal{P}_{h_2}|)$. $\boldsymbol{f}[24] = \log_p(|\mathcal{P}_{h_2}|/|\mathcal{P}_g|)$. $\boldsymbol{f}[25] = e_{g,h_2,P}$. $\boldsymbol{f}[26] = e_{g,h_2,2}$. $\boldsymbol{f}[27] = \log_p(e_{g,h_2,P}/e_{g,P})$. $\boldsymbol{f}[28] = \log_p(e_{g,h_2,2}/e_{g,2})$.

The 32-dimensional feature vector $\boldsymbol{f}_{gh}$ is composed as follows: $\boldsymbol{f}[1] = \sum_h \min\{1, a_{g,h}\}$. $\boldsymbol{f}[2] = \log_p(a_{g,h})$. $\boldsymbol{f}[3] = \log_p(\sum_i a_{g,i})$. $\boldsymbol{f}[4] = a_{g,h}/\sum_i a_{g,i}$. $\boldsymbol{f}[5] = a_{g,h}/\max_i a_{g,i}$. $\boldsymbol{f}[6] = e_{g,P}$. $\boldsymbol{f}[7] = e_{g,h,P}$. $\boldsymbol{f}[8] = \log_p(e_{g,h,P}/e_{g,P})$. $\boldsymbol{f}[9] = \log_p(e_{g,h,P}/e_{g,h_P,P})$. $\boldsymbol{f}[10] = e_{g,2}$. $\boldsymbol{f}[11] = e_{g,h,2}$. $\boldsymbol{f}[12] = \log_p(e_{g,h,2}/e_{g,2})$. $\boldsymbol{f}[13] = \log_p(e_{g,h,2}/e_{g,h_2,2})$. $\boldsymbol{f}[14] \in \{0,1,2,3\}$ characterizes when the last successful measurement was made. $\boldsymbol{f}[15] \in \mathbb{N}$ characterizes when the last successful measurement of h was made. $\boldsymbol{f}[16] = \log_p(|\mathcal{P}_g|)$. $\boldsymbol{f}[17] = \log_p(|\mathcal{P}_h|)$. $\boldsymbol{f}[18] = \log_p(|\mathcal{P}_g|/|\mathcal{P}_h|)$. $\boldsymbol{f}[19] \in \{0,1\}$ is 0 if the associated track is the static track and 1 otherwise. $\boldsymbol{f}[20] \in \{0,1\}$ is 0 if the only associated track is the static track and 1 otherwise. $\boldsymbol{f}[21] = \log_p\left(\left\|{}^t\boldsymbol{\rho}'_g - {}^t\boldsymbol{\rho}''_{g,h}\right\|\right)$. $\boldsymbol{f}[22] = \log_p(\mathrm{d}_M({}^t\boldsymbol{\rho}'_g, {}^t\boldsymbol{\rho}''_{g,h}))$. $\boldsymbol{f}[23] = \log_p(\mathrm{d}_M({}^t\boldsymbol{\rho}'_g, {}^t\boldsymbol{\rho}''_{g,h}))$. $\boldsymbol{f}[24] = \log_p(m_4)$. $\boldsymbol{f}[25] = \log_p(m_3)$. $\boldsymbol{f}[26] = \log_p(m_2)$. $\boldsymbol{f}[27] = \log_p(m_4 + m_3)$. $\boldsymbol{f}[28] = (m_4 + m_3)/(m_2 + m_1)$. $\boldsymbol{f}[29] = (m_4 + m_3 + m_2)/(m_1)$. $\boldsymbol{f}[30] = m_4/|\mathcal{P}_g|$. $\boldsymbol{f}[31] = (m_4 + m_3)/|\mathcal{P}_g|$. $\boldsymbol{f}[32]$ characterizes for the bin of the motion-histogram with the highest value the average move in meters within the last m frames in direction of the normal vector.

REFERENCES

[1] Y. Bar-Shalom. *Tracking and data association*. Academic Press Professional, Inc., 1987.

[2] P.J. Besl and H.D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, Feb 1992.

[3] M. Bosse and R. Zlot. Continuous 3D scan-matching with a spinning 2D laser. In *IEEE International Conference on Robotics and Automation*, pages 4312–4319, May 2009.

[4] S. Brechtel, T. Gindele, and R. Dillmann. Recursive importance sampling for efficient grid-based occupancy filtering in dynamic environments. In *IEEE International Conference on Robotics and Automation*, pages 3932–3938, may 2010.

[5] A. Censi. On achievable accuracy for range-finder localization. In *IEEE International Conference on Robotics and Automation*, pages 4170–4175, April 2007.

[6] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. In *IEEE International Conference on Robotics and Automation*, volume 3, pages 2724–2729, 9-11 April 1991.

[7] S. J. Davey, M. G. Rutten, and B. Cheung. A comparison of detection performance for several track-before-detect algorithms. *EURASIP Journal on Advances in Signal Processing*, 2008:428036, January 2008.

[8] B. Douillard, A. Quadros, P. Morton, J.P. Underwood, M. DeDeuge, S. Hugosson, M. Hallstrom, and T.A. Bailey. Scan segments matching for pairwise 3d alignment. In *IEEE International Conference on Robotics and Automation*, pages 3033–3040, May 2012.

[9] B. Fardi, J. Dousa, G. Wanielik, B. Elias, and A. Barke. Obstacle detection and pedestrian recognition using a 3D PMD camera. In *IEEE Intelligent Vehicles Symposium*, pages 225–230, June 2006.

[10] G. Gate and F. Nashashibi. Using targets appearance to improve pedestrian classification with a laser scanner. In *IEEE Intelligent Vehicles Symposium*, pages 571–576, June 2008.

[11] A. Harrison and P. Newman. High quality 3D laser ranging under general vehicle motion. In *IEEE International Conference on Robotics and Automation*, pages 7–12, May 2008.

[12] S. Kammel, J. Ziegler, B. Pitzer, M. Werling, T. Gindele, D. Jagzent, J. Schröder, M. Thuy, M. Goebl, F. v. Hundelshausen, O. Pink, C. Frese, and C. Stiller. Team AnnieWAY's autonomous system for the 2007 DARPA Urban Challenge. *Journal of Field Robotics*, 25(9):615–639, 2008.

[13] O. Michailovich and A. Tannenbaum. Segmentation of tracking sequences using dynamically updated adaptive learning. *IEEE Transactions on Image Processing*, 17(12):2403–2412, Dec. 2008.

[14] F. Moosmann and T. Fraichard. Motion estimation from range images in dynamic outdoor scenes. In *IEEE International Conference on Robotics and Automation*, pages 142–147, May 2010.

[15] F. Moosmann, O. Pink, and C. Stiller. Segmentation of 3d lidar data in non-flat urban environments using a local convexity criterion. In *IEEE Intelligent Vehicles Symposium*, pages 215–220, June 2009.

[16] F. Moosmann and C. Stiller. Velodyne SLAM. In *IEEE Intelligent Vehicles Symposium*, pages 393–398, June 2011.

[17] F. Nashashibi and A. Bargeton. Laser-based vehicles tracking and classification using occlusion reasoning and confidence estimation. In *IEEE Intelligent Vehicles Symposium*, pages 847–852, June 2008.

[18] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann. 6D SLAM—3D mapping outdoor environments: Research articles. *Journal of Field Robotics*, 24(8-9):699–722, 2007.

[19] A. Petrovskaya and S. Thrun. Model based vehicle detection and tracking for autonomous urban driving. *Autonomous Robots*, 26(2-3):123–139, 2009.

[20] D. Pfeiffer and U. Franke. Efficient representation of traffic scenes by means of dynamic stixels. In *IEEE Intelligent Vehicles Symposium*, pages 217–224, June 2010.

[21] S. Reuter and K.C.J. Dietmayer. Fuzzy estimation and segmentation for laser range scans. In *International Conference on Information Fusion*, pages 1974–1981, july 2009.

[22] T. Schamm, J. M. Zöllner, S. Vacek, J. Schröder, and R. Dillmann. Obstacle detection with a photonic mixing device-camera in autonomous vehicles. *International Journal of Intelligent Systems Technologies and Applications*, 5(3/4):315–324, 2008.

[23] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. Intelligent robotics and autonomous agents. MIT Press, September 2005.

[24] J. van de Ven, F. Ramos, and G.D. Tipaldi. An integrated probabilistic model for scan-matching, moving object detection and motion estimation. In *IEEE International Conference on Robotics and Automation*, pages 887–894, May 2010.

[25] T.-D. Vu, J. Burlet, and O. Aycard. Grid-based localization and online mapping with moving objects detection and tracking: new results. In *IEEE Intelligent Vehicles Symposium*, pages 684–689, June 2008.

[26] C.-C. Wang. *Simultaneous Localization, Mapping and Moving Object Tracking*. PhD thesis, Robotics Institute, Carnegie Mellon University, April 2004.