LOCALIZATION CORRECTIONS FOR MOBILE LASER SCANNER USING LOCAL SUPPORT-BASED OUTLIER FILTERING

V.V. Lehtola[a], J-P. Virtanen[a], P. Rönnholm[a], A. Nüchter[b]

[a] Department of Real Estate, Planning and Geoinformatics, Po. Box 15800, 00076 AALTO, Finland, ville.lehtola@iki.fi, juho-pekka.virtanen@aalto.fi, petri.ronnholm@aalto.fi
[b] Robotics and Telematics, University of Würzburg, andreas@nuechti.de

Commission IV, WG IV/7

KEY WORDS: Localization, Intrinsic localization, Noise Filtering, Outlier Removal, Mobile Laser Scanner

ABSTRACT:
Following the pioneering work introduced in [Lehtola et al., ISPRS J. Photogramm. Remote Sens. 99, 2015, pp. 25–29], we extend the state-of-the-art intrinsic localization solution for a single two-dimensional (2D) laser scanner from one into (quasi) three dimensions (3D). By intrinsic localization, we mean that no external sensors are used to localize the scanner, such as inertial measurement devices (IMU) or global navigation satellite systems (GNSS). Specifically, the proposed method builds on a novel concept of local support-based filtering of outliers, which enables the use of six degrees-of-freedom (DoF) simultaneous localization and mapping (SLAM) for the purpose of enacting appropriate trajectory corrections into the previous one-dimensional solution. Moreover, the local support-based filtering concept is platform independent, and is therefore prone to be widely generalizable. The here presented overall method is yet limited into quasi-3D by its inability to recover trajectories with steep curvature, but in the future, it may be further extended into full 3D.

1. INTRODUCTION

Mobile laser scanning (MLS) enables dynamic and rapid data collection. Various MLS systems exist, such as vehicle MLS (e.g., Kaartinen et al., 2013), backpack MLS (e.g. Ellum and El-sheimy, 2000; Naikal et al., 2009; Kukko et al., 2012; Elseber et al., 2013, Nüchter et al., 2015), and rolling MLS (Lehtola et al., 2015). The common problem is to solve accurate orientation of the scanner for all acquired 3D points. Many systems are relying on global navigation satellite systems (GNSS) and an inertial measurement unit (IMU) supported by, for instance, post-processing with a Kalman filter (El-Sheimy et al., 2006). However, GNSS is not feasible in all environments, making the localization of a mobile laser scanner without a reference coordinate system one of the persisting grand problems in laser scanning research. Besides its theoretical importance, any prominent solution is likely to enable applications in indoor environments, construction sites, and other areas that lack satellite coverage, such as urban canyons.

Following the pioneering work conducted in (Lehtola et al., 2015), we set out to extend the concept of intrinsic localization from 1D into 3D. By intrinsic localization, we mean that no other sensor data is used, such as that from inertial measurement units (IMU) or global navigation satellite systems (GNSS). In particular, we attempt to employ non-linear corrections to the theoretical one-dimensional trajectory so that the original three-dimensional scanner trajectory is recovered. For this purpose, we utilize the 3D Toolkit or 3DTK (Nüchter et al., 2011), which implements the well-known 6D SLAM with iterative closest points (ICP) algorithm and globally consistent scan matching. The algorithm is adopted from (Elseberg et al., 2013a), where it was used in a different mobile mapping context, i.e. with multi-wheeled platforms. Unlike other state-of-the-art algorithms, such as Stoyanov and Lilienthal (2009) and Bosse and Zlot (2009), it is not restricted to purely local improvements. Furthermore, we do not make rigidity assumptions, except for the computation of the point correspondences, and neither do we require an explicit motion model of the platform. The semi-rigid SLAM for trajectory optimization works with six degrees of freedom (DoF). Finally, the algorithm requires no high-level feature computation so that we require only the points themselves.

The caveat however is that outliers typically need to be filtered from the data before an ICP-based solution can be successfully employed. Other non-GNSS state-of-the-art works reminiscent to ours by Stoyanov and Lilienthal (2009) and Bosse and Zlot (2009) do not filter outliers. In the first, the error in the point positions is distributed along the vehicle path to produce a locally consistent measurement. The latter voxelizes the data and performs an additional weighting to match constraints with a Lorentzian function in order to reduce the impact of outliers. However, they note that “environments dominated by moving objects can also challenge the algorithm since it becomes difficult to distinguish true outliers”.

In the GNSS-scene, outlier filtering is conducted in the post-processing phase for the already registered data. Lin and Zhang (2015) searched planar objects with the region growth method developed by Vosselman and Klein (2010) and discarded the points that did not belong to segmented objects. Interestingly, however, region growing methods have been reported to suffer from over and under segmentation (Nurunnabi et al., 2015). Leslar et al. (2011) performed two outlier removal algorithms, where they either applied the moving fixed interval smoother algorithm in which trajectory and sensor data is utilized in Kalman filter that compares predicted and computed points and thus detects outliers, or they evaluate each point by fitting a quadratic curved-surface in its neighborhood which reveals remaining outlier points. Voxel-based methods search the number of points in the neighborhooding voxels (e,g., LASTools) or within multiple sifted voxels (Rönnholm et al., 2015). Wang et al. (2012) created MLS-based depth maps and filtered outliers by utilizing bins and distance-based adaptive thresholds. Zhu et al. (2011) projected MLS data to three orthogonal planes and divided them into bins, with a threshold for the bins that contained too few points. Vaaja et al. (2013)

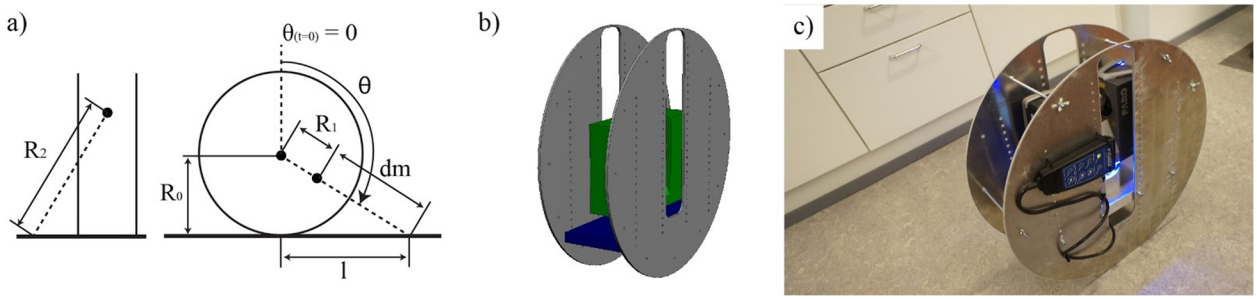

Figure 1. (a) A schematic describing the path parameter θ (t) and constants R_0, R_1, and R_2. Measured distances are marked with d_m. b) VILMA assembly schematics. c) A 2D laser scanner, here Faro Focus 3D in helical mode, is attached in between two metallic discs so that it retains a wide viewing angle through the oval holes carved on the discs. This figure is reproduced from (Lehtola et al., 2015).

detected outlier points by searching if there was less than ten points within a 0.50 meters radius. TerraScan also examines spherical area with given radius when removing unwanted air points utilizing median elevations and standard deviations (TerraScan). To conclude, the noise filtering in these works is based on the assumption that full information is available in the 3D neighborhood of the point in focus. Therefore, the GNSS-scene approach is unsuitable for our needs, as the filtering must take place before the ICP-based registration.

In this paper, we present an inherently local outlier filtering method that is optimized for MLS data, and takes place before the ICP-based registration. Then we show how to correct the relative orientations between the laser scanning profiles obtained from the theoretical solution of (Lehtola et al., 2015).

The paper is organized as follows. The 1D theoretical solution for intrinsic localization is briefly summarized in Section 2. Then, the data processing is done in two steps. First, mobile laser scanner data is filtered, for which we introduce a novel concept of local range support in Section 3.1. Second, a semi-global point cloud matching using 3DTK 6D SLAM is done to adjust the trajectory to its environment, leading to a 3D solution. This is presented in Section 3.2. Results are shown in Section 4, after which follows the Conclusion.

2. INTRINSIC LOCALIZATION FOR A ONE-DIMENSIONAL TRAJECTORY

The localization of the laser data requires a successful reconstruction of the sensor trajectory. The trajectory j(t) is time-dependent with six degrees of freedom, namely, three from location and three from orientation. We write out

$$\mathrm{j}(t) = [\theta(t), \psi(t), \varphi(t), x(t), y(t), z(t)]^T \qquad (1)$$

where θ is the pitch, ψ is the roll, and φ is the yaw angle. Time is denoted by t. Without any reference coordinate system, the successful reconstruction of the trajectory requires that these degrees of freedom are eliminated. Previously, this was done by Lehtola et al. (2015) for a holonomic system in one dimension (1D). We briefly outline this solution here.

In order to capture a 3D environment with a 2D laser scanner, the scanner must be rotated about at least one axis. The 2D scanner is mechanically attached to the platform, so that it can only rotate about one axis of rotation, namely θ. Therefore rotational degrees of freedom are reduced by two, i.e. φ and ψ are constant. By assuming that the platform does not slip against the floor x, y, and z all become direct functions of θ.

The scanner sits on the hypotenuse at a distance of $R_0 - R_1$ from VILMA's central axis, where R_0 = 0.25m is the radius of the metal disc and R_1 = 0.13m, see Fig. 1 (a). Assuming that the floor is flat, simple trigonometry is employed to write

$$\theta = \arccos(R_0 / d), \qquad (2)$$

where $d = d_m + (R_0 - R_1)$, and d_m is the minimum measured distance to the floor over one full 2D circle observation. Considering the minimum distance to the floor, the position of the scanner on disk radius varies between two values, depending on whether the scanner is upside down, Eq. (2), or upside up, in which case $\theta = \pi - \arccos(R_0/d)$, with $d = R_0 + \cos 27.5^{\circ}(d_m - R_2)$, and R_2 = 42cm. Here, the 27.5 degrees is half of the dead angle of the scanner.

The pitch angle θ(t) is the path parameter that describes the scanner trajectory, and obtaining it from the scanner data solves localization in 1D. Initially, the zenith is pointing upwards, θ(t =0) = 0. Then, θ is incremented by 2π for each cycle that the platform rolls. Each time the 2D scanner is perpendicular towards the floor (PTF), $\theta(t) = \pi + 2\pi n$, n = 0, 1, 2, ..., the scanning distance reduces to the minimum R_1. We call this a PTF-observation, and keep track of these occurrences in the laser data obtaining a time series. The PTF observation is robust to error, since data points from a large field of view can be used to interpolate the floor point precisely below the sensor. Also, stochastic errors in PTF observations do not cumulate with time as long as the no-slip condition with the floor applies.
Once θ(t) is obtained, a coordinate transformation for the 2D sensor data (X, Z) is obtained considering the trajectory of a contracted cycloid,

$$\begin{cases} x = X \\ y = R_0\theta + (R_0 - R_1 + Z)\sin\theta \\ z = R_0 + (R_0 - R_1 + Z)\cos\theta \end{cases} \qquad (3)$$

where (x,y,z) are the coordinates of the resulting 3D point cloud. Note that the platform propagates in the positive y-direction.

3. METHODS

Outliers, in general, are caused by multiple factors. All scanners experience some level of inherent noise, but outliers follow also from the environment and from the way the scanner is operated. The latter two are here the main sources of outliers. The built environment typically contains reflecting surfaces, such as mirrors and those made of shiny metals, causing multiple reflections and gaps in data. Also, the platform movement is prone to be non-smooth, such as in our experiments, where the concrete floor is littered with dirt and small rocks. These

reasons alone create the need to filter data for outliers. In (Lehtola et al., 2015), filtering was done only at a global level for the purpose of improving the visual properties of the final 3D point cloud. Here, it is necessary to filter data before the trajectory corrections, which in turn must be done before obtaining the final point cloud. The outlier removal is especially important because the points at long distance have crucial weight on the total SLAM optimization, explained in detail in Section 3.2. Hence, instead of employing global point cloud properties, the filter needs to rely on the temporally local properties of the measured data. For curious readers, the time series of range data is visualized as supplementary material. The Faro Focus 3D scanner in 2D helical mode, which was used in our experiments, has a field of view of 300 degrees. The mirror rotation frequency is 95 Hz.

3.1 LOCAL FILTERING BASED ON SUPPORT

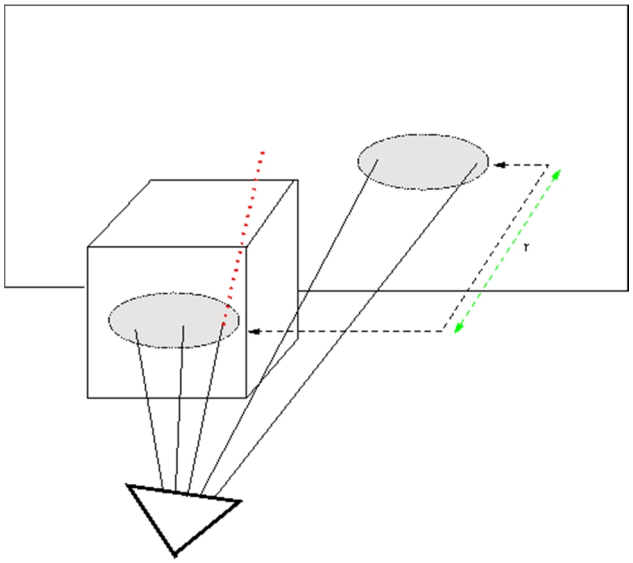

Figure 2. Schematic of local support-based filtering. A 2D scanner is depicted by a triangle. Range measurements from consecutive channels, represented by solid black lines, do not obtain support from each other, if the range difference Δr is too large. Here, the middle point gets support from its left, but not from its right. On the other hand, the red dotted imaginary beam going through the obstacle would get support from the right, but not from the left.

The 3D point cloud data is divided into slices. Each slice, captured during one rotation of the scanner mirror, contains 8534 range measurements. Furthermore, each slice is captured from a different position and angle, because the scanner platform is mobile and rolling. The orientation of each slice is obtained from the procedure explained in Section 2.

We introduce a concept of *support-based local noise filtering*, see Fig 2. The idea is that each point, i.e. a time-of-flight measurement, either has support or is rejected. For this purpose, the point data processing employs two filters: inter-slice support and intra-slice support. Any accepted point must pass both filters. Now, consider a point on slice i and channel c. For the inter-slice support, the change in range is checked against previous i-1 and next i+1 slice, keeping the channel c constant (i.e. the beam angle within the slice). If for both differences Δr > 5.0 m, then the current point does not have support, and is rejected.

Formally, the inter-slice support $\hat{s}_j$ for point j is written as

$$\hat{s}_j = \begin{cases} 1, & if\ \hat{c}_j \geq 1 \\ 0, & otherwise \end{cases}, \quad (4)$$

where the count

$$\hat{c}_j = \sum_{k \in \hat{N}_j} \hat{\theta}_{jk}. \quad (5)$$

Here, $\hat{N}_j$ is the 2-point neighborhood of point j, and $\hat{\theta}$ is a Heaviside step function

$$\hat{\theta}_{jk} = \begin{cases} 1, & if\ \Delta r_{jk} < 5.0\ m \\ 0, & otherwise \end{cases} \quad (6)$$

that responds to the measured range distance $\Delta r_{jk} = |\boldsymbol{r}_j - \boldsymbol{r}_k|$ between points j and k.

For the intra-slice support, 8 nearest neighbors are considered for each point on a 2D slice. Any accepted point must be supported by at least three of these neighbors. Support is gained if the pair-wise difference in range measurement Δr < 0.15 m between the point and its neighbor. Formally, the intra-slice support s_j for point j is written as

$$s_j = \begin{cases} 1, & if\ c_j \geq 3 \\ 0, & otherwise \end{cases} \quad (7)$$

where the count

$$c_j = \sum_{k \in N_j} \theta_{jk}. \quad (8)$$

Here, N_j is the 8-point neighborhood of point j, and θ_{jk} is again a Heaviside step function

$$\theta_{jk} = \begin{cases} 1, & if\ \Delta r_{jk} < 0.15\ cm \\ 0, & otherwise \end{cases}. \quad (9)$$

Let us discuss the support cut-off with respect to the measurement geometry. The most challenging inter-slice geometry is with respect to the floor, as this forms a triangle strongly elongated in one direction. Now, each full cycle contains about from 180 to 600 slices. Therefore, the orientation difference between the slices is of the order of one degree. If the scanner rotates by one degree on a flat floor, the inter-slice support condition in Eqs. (4-6) with $\Delta r_{jk} < 5.0\ m$ means that that the points on the floor two degrees below horizon still pass this condition. On the other hand, any points that violate this rule can be safely discarded as outliers. For the intra-slice geometry, support from a single closest point could suffice in Eqs. (7-9). However, we take into account that one single measurement in the supporting neighborhood may fail for e.g. internal reasons of the scanner. Also, we do not allow that a small set of two or three erroneous points would support each other. Therefore, considering the range discontinuity between two surfaces at different depths, a majority from four points (=3) is required to lie within the threshold in order for the point being considered to gain support.

Considering generalizability, the local properties of the filtering algorithm allow its use for on-chip computation, when a cache of three slices of points can be provided. This is bound to gain relevance as the scanner sizes become smaller, see e.g. Kostamovaara et al. (2015).

3.2 LOCALIZATION CORRECTIONS

We convert the scanner output into a form suitable for 3DTK - The 3D Toolkit (Nüchter et al., 2011). 3DTK is a general point cloud processing framework with an emphasis on registration and mapping. It efficiently implements the well-known 6D SLAM to the similarly well-known iterative closest points (ICP) algorithm and globally consistent scan matching, resulting into an ICP-like solution to the simultaneous localization and mapping (SLAM) problem (Nüchter et al., 2010). To understand the basic idea of the localization correction, we summarize its basis, 6D SLAM. It works similarly to ICP, which minimizes the following error function

$$\mathsf{E}(\mathbf{R},\mathbf{t}) = \frac{1}{N}\sum_{i=1}^{N}\|m_i - (\mathbf{R}\,d_i + \mathbf{t})\| \qquad (10)$$

to solve iteratively for an optimal rotation (**R**, **t**), where the tuples (m_i, d_i) of corresponding model M and data points D are given by minimal distance (Besl, 1992). Instead of the two-scan-equation above, we look at the n-scan case

$$\mathrm{E} = \sum_{j\to k}\sum_i \left\|(\boldsymbol{R_j} m_i + \boldsymbol{t_j}) - (\boldsymbol{R_k} d_i + \boldsymbol{t_k})\right\| \qquad (11)$$

where j and k refer to scans of the SLAM graph, i.e., to the graph modelling the pose constraints in SLAM or bundle adjustment. If they overlap, i.e. closest points are available, then the point pairs for the link are included in the minimization. We solve for all poses at the same time and iterate like in the original ICP. Please note that while there are four closed-form solutions for the original ICP, linearization of the rotation in the second equation is always required.

We also applied an algorithm that improves the entire trajectory of VILMA simultaneously. The algorithm is adopted from (Elseberg et al., 2013a), where it was used in a different mobile mapping context, i.e., on wheeled platforms. Previously, this trajectory correction has also been applied to a sensor-skid inside production environments and to a backpack-mounted scanning solution. Unlike other state-of-the-art algorithms, such as Stoyanov and Lilienthal (2009) and Bosse and Zlot (2009), it is not restricted to purely local improvements. We make no rigidity assumptions, except for the computation of the point correspondences. We require no explicit motion model of a vehicle for instance. The semi-rigid SLAM for trajectory optimization works with six DoF. The algorithm requires no high-level feature computation, i.e., we require only the points themselves.

For VILMA, we do not have separate terrestrial 3D scans. In the current state-of-the-art developed by Bosse and Zlot (2009) for improving overall map quality of mobile mappers in the robotics community the time is coarsely discretized. This results in a partition of the trajectory into sub-scans that are treated rigidly. Then rigid registration algorithms like the ICP and other solutions to the SLAM problem are employed. Obviously, trajectory errors within a sub-scan cannot be improved in this fashion. Applying rigid pose estimation to this non-rigid problem directly is also problematic since rigid transformations can only approximate the underlying ground truth. When a finer discretization is used, single 2D scan slices or single points do not constrain a six DoF pose sufficiently for rigid algorithms.

Mathematical details of our algorithm are given in Elseberg et al. (2013a). Essentially, we first split the trajectory into sections, and match these sections using the automatic high-precise registration of terrestrial 3D scans, i.e., globally consistent scan matching. Here the graph is estimated using a heuristics that measures the overlap of sections using the number of the closest point pairs. After applying globally consistent scan matching on the sections the actual semi-rigid matching as described in Elseberg et al. (2013a) is applied, using the results of the rigid optimization as starting values to compute the numerical minimum of the underlying least square problem. To speed up the calculations, we make use of the sparse Cholesky decomposition.

For VILMA, the amount of data gathered per distance traveled along the trajectory varies, because the platform velocity is not constant. Each cycle contains about 180 to 600 slices. Hence, as input parameters, we choose to perform a 6D SLAM match between every 150 slices, and use 600 slices to perform each of these.

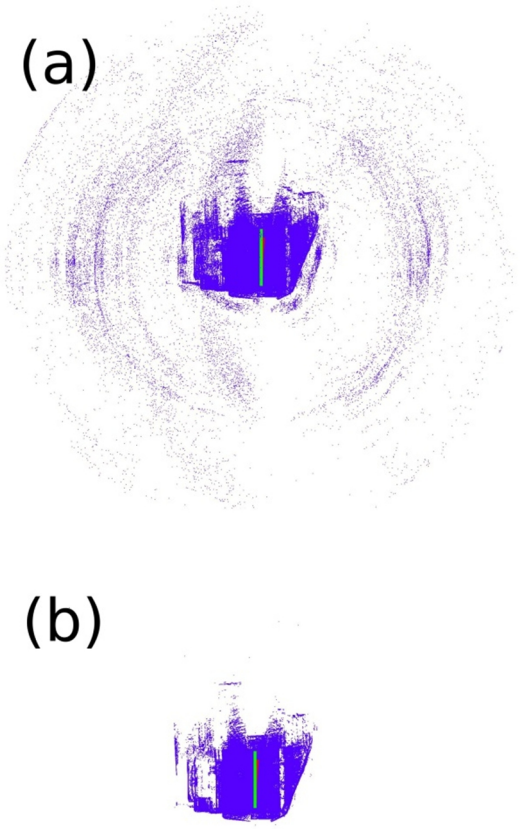

Figure 3. (a) Unfiltered point cloud vs. (b) filtered point cloud visualized from the top. Almost all of the outliers vanish in filtering.

4. RESULTS

The amount of outliers in VILMA data is significant, but the local filtering method introduced in Section 3.1 manages to eliminate almost all of them, see Fig. 3. The point cloud is reduced from 43.1 million to 40.8 million points, equaling to a mere 5.2% reduction. The outliers, and their removal, has impact on both the overall shape of the point cloud, and its inside structure. For convenience, the displayed point clouds are

randomly downsampled before filtering to one tenth of their original size.

Glass and metallic surfaces present in the garage result to specular reflections so that the reflected light follows a different path from the incident light for all cases other than the light normal to the surface. This is one source of outliers for time-of-flight measurements, as the flight path of the laser is elongated by the reflections. Outliers from small reflecting areas, such as those occurring from side mirrors of the cars, are eliminated by the inter-slice support requirement. On the other hand, the intra-slice support requirement keeps the data consistent with respect to most other error sources.

After the filtering, the trajectory is corrected as explained in Section 3.2, and respectively, a new point cloud is written out. Refer to Fig. 4 to see what the visual impact of the correction to the resulting 3D point cloud is. As the trajectory tilts right through correction, recovering its original shape, the shape of the surrounding environment is also simultaneously recovered. For example, apparently oblique walls are straightened.

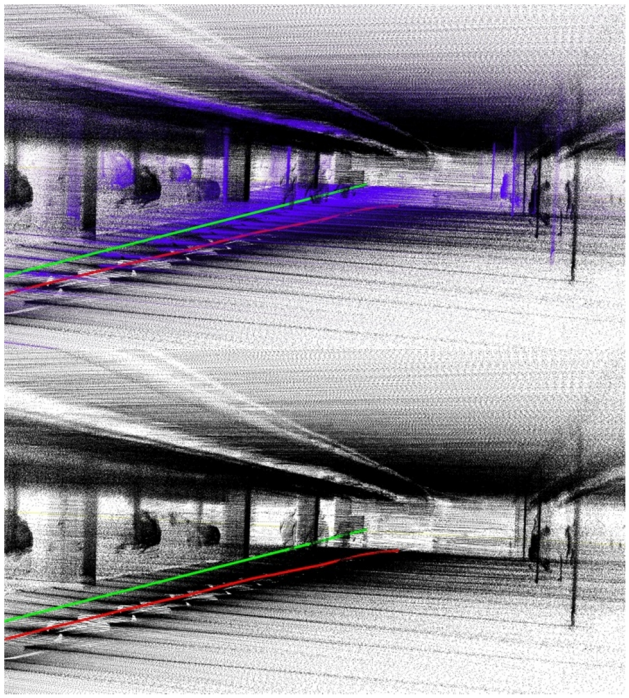

Figure 4. Top: Close-up view on concrete pillars without outlier filtering and a (green) straight trajectory (shown in violet). Objects are dislocated, and may assume oblique shapes. With filtering and corrections, objects are properly located inside the point cloud, and recover their natural shapes (shown in black). The point cloud is derived from the (red) corrected trajectory. Bottom: Only the filtered and corrected point cloud is shown. Both trajectories are visible in both images for visualization purposes.

In order to see if the presented filtering only has a visual effect or a quantifiable effect that as well affects the correction procedure, corrections are run for both unfiltered and filtered point clouds, see Fig 5. The obtained trajectories differ in the sense that the corrections made for the unfiltered point cloud, compared to the filtered case, do not reach the same amplitude. In other words, the corrections are left too small, because trajectory corrections made with the unfiltered point cloud are more likely to get stuck in local nearest neighbor minima than those made on the filtered point cloud. For horizontal displacement Δx, the difference is over 30 cm for a trajectory of the length of 30 m, equaling to a ratio of 1:100. In other words, the proposed filtering is crucial to avoid a significant incrementation of drift, even when the underlying trajectory is close to a straight one. We have also reproduced the plot data from Lehtola et al. (2015) (see Fig. 5 (b)) for comparison purposes. This plot was obtained by manually orientating separate segments of VILMA data to the TLS reference scans leading into - what can be judged from Fig. 5 - a crude estimate.

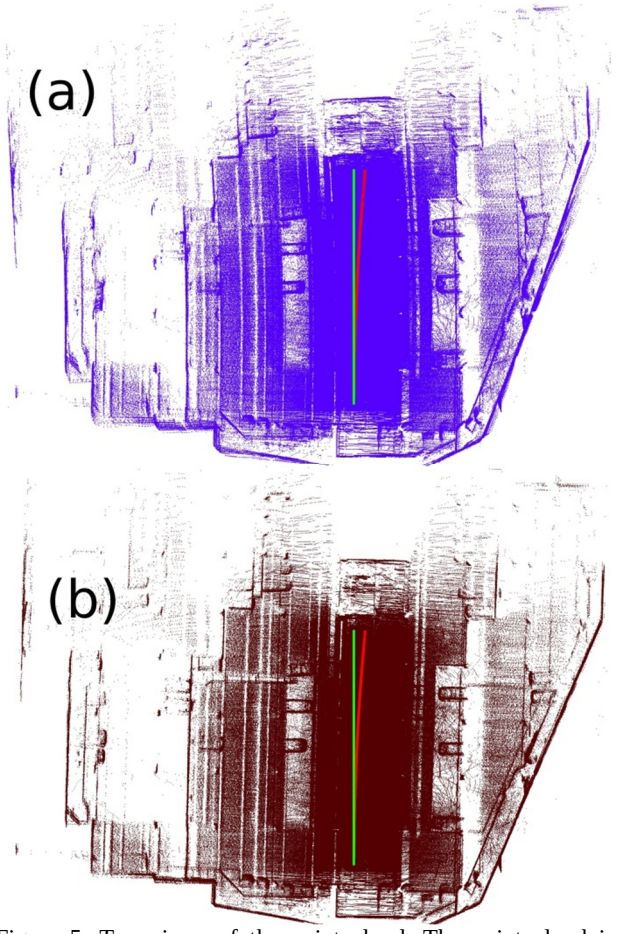

Figure 5. Top views of the point cloud. The point cloud is calculated based on (a) a straight trajectory (shown in green) leading into oblique shapes of straight features, e.g. vertical lines bending left, and (b) the corrected trajectory (shown in red), which has recovered the physical path of VILMA, leading to a point cloud that better corresponds to the physical environment. Both trajectories are visible in both images for visualization purposes.

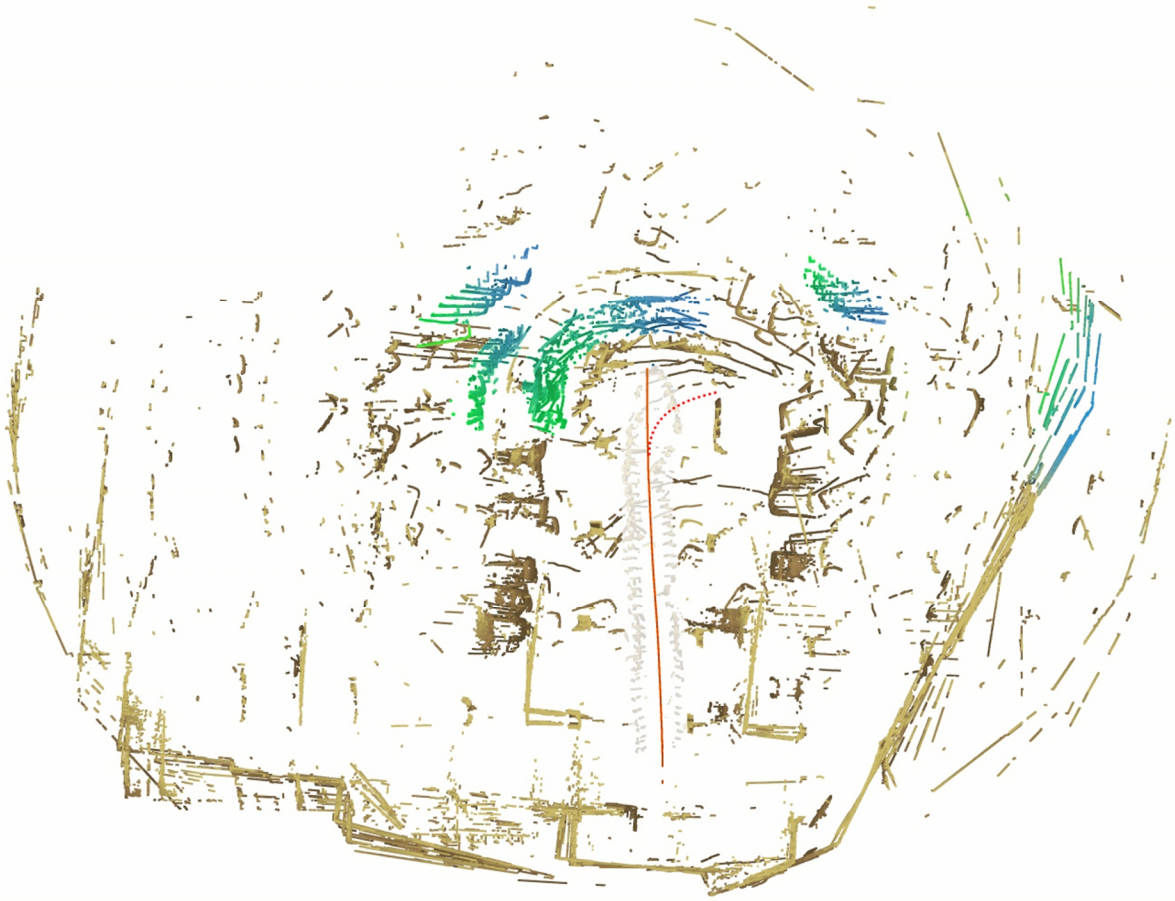

Figure 7. An attempt to correct a steep curve fails. The point cloud is shown from the top with the floor and the ceiling removed. The computed trajectory is plotted in red. The location of the actual trajectory is manually approximated with a dotted red line, and differs from the computed trajectory. Therefore, the walls and objects are displaced by a counter-clockwise rotation. We have colored some of them for visualization purposes with the color fading from green (displaced data) to blue (correctly placed data). However, the trajectory is successfully corrected in the lower part of the image, where the curvature is low enough in both the horizontal and the vertical dimensions.

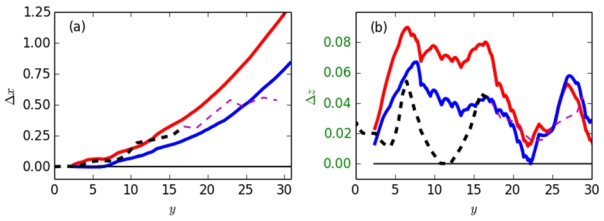

Figure 6. Corrected trajectory plots for filtered (red) and unfiltered (blue) data (in m). (a) The horizontal displacement Δx, and (b) the vertical displacement Δz, from the straight trajectory are shown. The straight trajectory assumption, i.e. without semi-global 6D SLAM corrections, is shown on the bottom with a black solid line. The slashed lines represent the crude manual estimates from Lehtola et al. (2015)(see Fig. 5 (b) there-in), with the black part being more reliable than the magenta part.

The filtering has already proven to be quite effective, since it removed only about 5% of the total points, while both significantly improving the visual properties of the point cloud, and notably improving the functionality of the correction algorithm.

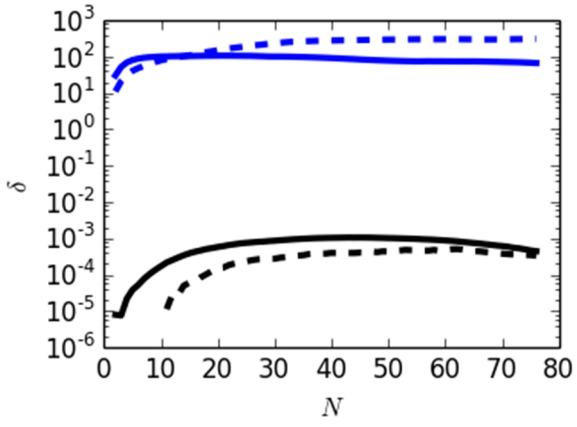

Figure 8. Displacement δ (in cm) from the straight trajectory estimate as a function of N semi-global correction iterations. The displacement of the top (bottom) end of the trajectory is shown with black (blue) lines. Dashed (solid) lines represent absolute displacements in the x (y) -direction. Note the logarithmic scale. Displacements converge after some N=40 iterations.

Surprisingly, the proposed method encounters difficulties when the trajectory contains too steep curves, see Fig. 6. The method fails to recover the underlying physical trajectory, shown with a dotted red line, because it gets stuck in local minima of the ICP-based error function of Eq. (8). Data of the environment captured from this part of the trajectory is displaced accordingly. We have colored some of it for visualization purposes to provide an intuitive view for the reader of the consequences of this failure. For quantitative purposes, the displacement δ from the straight trajectory estimate as a function of N semi-global correction iterations is shown in Fig. 7. Both ends of the trajectory shown in Figure 6 are taken under the magnifying glass, because the sum of all corrections inflicted upon the trajectory is reflected on these. We see that both displacements converge after some N=40 iterations. This means that a local minimum in the n-scan matching is reached, and that further computation does not help in recovering the original physical trajectory. Considering the dotted red line in Fig. 6, the scan matching fails due to a large viewpoint change associated with the steep curvature. Furthermore, the environment is not bountiful enough with suitable data to compensate for this. Ultimately, we must conclude that the usability of the proposed method is limited to so-called quasi-3D, the method working only for rather small displacements from the original 1D trajectory. This sets a goal for future work to close to gap between the theoretical 1D trajectory estimate and - not necessarily a six DoF solution, as this seems quite challenging - but the physical movement on the 2D manifold embedded into 3D space.

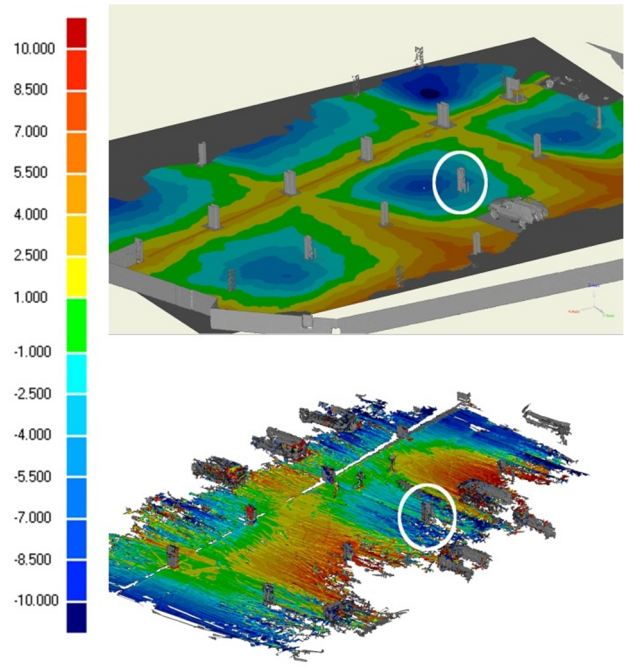

Figure 9. Top: Contour plot from the reference TLS point cloud displays the floor height profile (in cm). The largest displacements from a virtual horizontal plane are shown with deep blue (descent) and orange (ascent). Bottom: The difference of triangulated surface models between the straight trajectory estimate and the *corrected* VILMA data. Both plots use the same units that are in cm, and display one pillar circled with white color to guide the eye.

Finally, we explore the abilities of the presented method to recover the height differences instilled into the recorded data. Conveniently, the car park has a sloped concrete floor in order to guide the water that is created from melted snow brought in by cars, into the sewer sinkholes. In Fig. 8 top image, a contour plot from the reference TLS point cloud displays this floor height distribution. The maximum height deviation along the trajectory is around 12 cm. Next, we check whether comparing the corrected VILMA data with the point cloud obtained with the straight trajectory estimate displays a similarly behaving height profile. Before the difference calculation, the two point clouds were separately oriented using a semi-manual ICP-tool to for a best match with the TLS point cloud. The result is shown in Fig. 8 bottom image. The computed height differences adopt a profile similar to what was obtained from the TLS data, meaning that the 6D SLAM approach is successful also for the vertical dimension. However, in both ends of the trajectory, the method overcorrects the descent due to finite data from the environment. This is shown by the blue regions that extend from one end of the image to its other end.

5. CONCLUSIONS

The presented filtering method proves to be quite effective, removing only about 5% of the total points, while both significantly improving the visual properties of the point cloud, and notably improving the functionality of the correction algorithm. In the future, it may be implemented on-chip for faster computation, as its cache requirement is small. On the other hand, the correction properties of the presented method require further development, as its usability is limited into so-called quasi-3D. In other words, the method works only when the curvature of the underlying physical trajectory to be recovered is not too steep. Finally, this paper paves the road towards the grand goal where one can show that the *intrinsic localization* for a 2D laser scanner is generally feasible in multiple dimensions. By intrinsic localization, we mean that no other sensor data is used, such as that from inertial measurement units (IMU) or global navigations satellite systems (GNSS).

ACKNOWLEDGEMENTS

The authors wish to thank Academy of Finland for funding this research, Grant No. 257755 (VL), CoE-LaSR Grant No. 272195 (VL, J-PV, PR), and Strategic Research Council project COMBAT No. 293389 (VL, J-PV).

REFERENCES

Besl, P. and McKay, N., 1992. A method for Registration of 3-D Shapes. IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI) 14(2), pp. 239–256

Borrmann, D., Elseberg, J., Lingemann, K., Nüchter, A., and Hertzberg, J., 2008. Globally consistent 3D mapping with scan matching. *Robotics and Autonomous Systems*, *56*(2), 130-142.

Bosse, M. and Zlot, R., 2009. Continuous 3D Scan-Matching with a Spinning 2D Laser. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '09), Kobe, Japan, pp. 4312–4319.

El-Sheimy, N., 2006. Kalman Filter Face-Off: Extended vs. Unscented Kalman Filters for Integrated GPS and MEMS Inertial. InsideGNSS, 1(2), pp. 48-54.

Ellum, C. and El-sheimy, N., 2000. The development of a backpack mobile mapping system. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 33, pp. 184–191.

Elseberg, J., Borrman, D., Nüchter, A., 2013. A study of scan patterns for mobile mapping. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XL-7/W2, pp. 75-80.

Kaartinen, H., Hyyppä, J., Kukko, A., Lehtomäki, M., Jaakkola, A., Vosselman, G., 2013. Mobile Mapping - Road Environment Mapping using Mobile Laser Scanning. EuroSDR Official Publication No 52, pp. 49-93.

Kostamovaara, J., Huikari, J., Hallman, L., Nissinen, I., Nissinen, J., Rapakko, H., Avrutin, E., and Ryvkin, B. (2015). On Laser Ranging Based on High-Speed/Energy Laser Diode Pulses and Single-Photon Detection Techniques. Photonics Journal, IEEE, 7(2), 1-15.

Kukko, A, Kaartinen, H., Hyyppä J., and Chen, Y., 2012. Multiplatform Mobile Laser Scanning: Usability and Performance. Sensors, 12 (9), pp. 11712-11733.

LAStools. https://www.cs.unc.edu/~isenburg/lastools/download/lasnoise_README.txt. [Visited on 4 December]

Lehtola, V. V., Virtanen, J. P., Kukko, A., Kaartinen, H., and Hyyppä, H., 2015. Localization of mobile laser scanner using classical mechanics. *ISPRS Journal of Photogrammetry and Remote Sensing*, *99*, 25-29.

Leslar, M., Wang, J-G., Hu, B., 2011. Comprehensive Utilization of Temporal and Spatial Domain Outlier Detection Methods for Mobile Terrestrial LiDAR Data. Remote Sens. 2011, 3, 1724-1742.

Naikal, N., Kua, J., Chen, G., Zakhor, A., 2009. Image Augmented Laser Scan Matching for Indoor Dead Reckoning. Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4134-4141

Nurunnabi, A., West, G., Belton, D., 2015. Robust methods for feature extraction from mobile laser scanning 3D point clouds. In: B. Veenendaal and A. Kealy (Eds.): Research@Locate'15, Brisbane, Australia, 10-12March 2015, pp. 109- 120.

Nüchter, A., Elseberg, J., Schneider, P., and Paulus, D., 2010. Study of parametrizations for the rigid body transformations of the scan registration problem. *Computer Vision and Image Understanding*, *114*(8), 963-980.

Nüchter, A., Borrmann, D., Koch, P., Kühn, M., May, S., 2015. A Man-Portable, IMU-free Mobile Mapping System. ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci., II-3/W5, 17-23.

Nüchter A. et al., 2011. 3DTK – The 3D Toolkit. http://slam6d.sourceforge.net/

Rönnholm, P. Kukko, A., Liang, X., Hyyppä, J., 2015. Filtering the outliers from backpack mobile laser scanning data. The Photogrammetric Journal of Finalnd, 24(2), pp. 20-35.

Stoyanov, T. and Lilienthal, A. J., 2009. Maximum Likelihood Point Cloud Acquisition from a Mobile Platform. In: Proceedings of the IEEE International Conference on Advanced Robotics (ICAR '09), Munich, Germany.

TerraScan. TerraScan User's Guide, 566 pages. https://www.terrasolid.com/download/tscan.pdf. [visited on 4 December]

Vaaja, M., Kukko, A., Kaartinen, H., Kurkela, M., Kasvi, E., Flener, C., Hyyppä, H., Hyyppä, J., Järvelä, J., Alho, P., 2013. Data Processing and Quality Evaluation of a Boat-Based Mobile Laser Scanning System. Sensors, 13, pp. 12497-12515.

Vosselman, G and Klein, R., 2010. Visualization and structuring of point cloud. In: Airborne and Terrestrial Laser Scanning, 1st ed.; Vosselman, G., Maas, H.G., Eds.; Whittles Publising: Dunbeath, pp. 43-79.

Wang, R., Bach, J., Macfarlane, J. and Ferrie, F., 2012. A new upsampling method for mobile LiDAR data. In: IEEE Workshop on Applications of Computer Vision (WACV), pp. 17–24.

Xiangguo Lin, X. and Zhang, J., 2015. Segmentation-based ground points detection from mobile laser scanning point cloud. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XL-7/W4, pp. 99-102.

Zhu, L., Hyyppä, J., Kukko, A., Kaartinen, H., Chen., R., 2011. Photorealistic Building Reconstruction from Mobile Laser Scanning Data. Remote Sensing, 3, pp. 1406-1426.