
# A lidar-only SLAM algorithm for marine vessels and autonomous surface vehicles

Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 55-31 (2022) 229–234

# A lidar-only SLAM algorithm for marine vessels and autonomous surface vehicles

**Artur Engström** * **Domenic Geiseler** * **Krister Blanch** *
**Ola Benderius** * **Iván García Daza** **

* *Chalmers University of Technology, Göteborg, Sweden (e-mail: {arture, domenic}@student.chalmers.se, {krister.blanch,ola.benderius}@chalmers.se).*
** *University of Alcalá, Madrid, Spain (e-mail: ivan.garciad@uah.es)*

**Abstract:** Research into autonomous surface vehicles is noticeably limited in regards to the functionality of the vehicles themselves. Specifically, testing and evaluation typically occurs at speeds considerably lower than what is allowed in an operational setting. For a vessel to be able to take advantage of higher speeds, there must be a robust and tested method for determining localisation and navigation. With an emphasis of development for small vessels with higher impulse capabilities, working in confined and restricted environments, the decision was made to develop a method of navigation that relied solely upon lightweight sensors. For this, a single light ranging sensor was utilised to develop both simultaneous localisation and mapping for the vessel, using the normal distribution transform and iterative closest point methods. Evaluation of the algorithm accuracy as the vessel moved above speeds greater than two metres per second was conducted, and it was feasibly evaluated that there was no observable drift of mapping in horizontal planes, however, there was a accumulated drift in the vertical plane and a transient response in localisation deviation as the vessel changed impulse through the two metre per second window.



*Keywords:* Lidar-based SLAM, Mapping, Localisation, Marine systems

## 1. INTRODUCTION

Autonomous surface vehicles (ASVs) have the potential to provide a robust alternative to routine tasks, especially within confined environments. For this to succeed, however, there must be a robust method for determining the vessels placement within the environment, as well as an accurate representation of the environment itself. This type of function, often referred to as *simultaneous localisation and mapping* (SLAM), has a number of different implementations based on the type of vehicle and environment. For the maritime sector, there are a number of specific unresolved difficulties with forming such map, typically assigned into the categories of speed, accuracy, and validation.

As there are few datasets that tailor to the particular environments that would be expected (Benderius et al., 2021), the first concern is the lack of validation when building automation specifically for vessels. Another problem is that sensors which are commonplace for autonomous land vehicles, come with significant hindrances when moved to the marine sector. For example, current SLAM methods utilising a *light detection and ranging sensor* (lidar), typically coupled with *global navigation satellite systems* (GNSS) and *inertial measurement units* (IMUs), have led to consistent mapping in the horizontal plane while performance in the vertical plane is less studied. The reason is naturally that land based vehicles have a ground plane consistent with the vehicles position, and vertical dynamics is practically non-existent. In the marine sector, platforms are subject to *heave*, which is traversal in the horizontal direction, even when the vessel is stationary. This problem becomes exacerbated when the vessel moves into a GNSS obstructed environment, which is more typical for vessels working alongside structures or within confined areas.

### 1.1 Marine Operations

Marine operations can be classified loosely into two categories, being *restricted* and *unrestricted*. Both of these are still governed by the convention on the international regulations for preventing collisions at sea (COLREGs), as well as local laws. However the generalisation is that there is some form of limit in restricted waters, which for the purposes of this paper, is most likely a speed or wake limit. The Gothenburg region, on the west coast of Sweden, where the experiments of this paper took place is no exception, with the local laws stipulating that in restricted areas, a speed limit of eight or twelve knots (depending on vessel size) is imposed. This translates to four and six metres per second, respectively, and these speeds will be used for the remainder of this paper. This distinction is important, as whilst there has been research into developing localisation and mapping for autonomous watercraft, only limited works have looked at developing a robust method for determining a vessels capability at allowed speeds, and instead trends look at the $2\,\mathrm{m\,s^{-1}}$ benchmark.

As autonomous surface vehicles start moving into roles where impulse precision is a necessity, this shortfall will


10.1016/j.ifacol.2022.10.436

unfortunately limit their capabilities. This paper therefore provides a succinct look at the utilisation of a lidar-driven SLAM at normal maritime speeds in restricted waters, with the vessel moving at speeds greater than $2\,\mathrm{m\,s^{-1}}$, and subject to traversal in full three dimensional (3D) space.

## 2. RELATED WORK

Whilst there are many papers detailing the varying methods of SLAM, the most significant in the field of maritime systems is the work conducted by Papadopoulos et al. (2014) which has been the benchmark for single lidar sensor 3D map building in a marine environment since its iteration in 2014. However, this method is limited to an operating speed of $2\,\mathrm{m\,s^{-1}}$, which is far below the typical operating speed of conventional watercraft. For a more accurate representation of contemporary systems, teams from Data61 and CERBERUS have recently published findings on airborne unmanned vehicles operating in confined GNSS restricted environments (Khattak et al., 2020; Hudson et al., 2021). These systems are more relevant to the proposed marine system, in that the vehicle has to overcome three dimensional spatial awareness in sensor deprived scenarios, with the ultimate goal of achieving both a quick operating speed and a low error rate. The CERBERUS team provided a solution that allowed their system to operate closer to $5\,\mathrm{m\,s^{-1}}$, which is in line with the working speed of restricted maritime areas, however, their system relied upon a multi-sensor method for success.

Alternatively, another method proposed by Nubert et al. (2022) relies on a single sensor and was developed with similar operating constraints. This method utilises a learning based approach on purely simulated data, and then resolves validation by testing within the confined spaces of a semi-symmetrical underground system. Whilst no operational speed was detailed it is worth nothing that their work focused on resolving navigation and mapping with a single lidar in an environment that is symmetrical in nature, which is comparable to the smooth edges of marina pens, artificial shorelines, and the waterline that this paper targets. Furthermore, the overall problems found within their work were also confirmed from the experiments promoting this paper. Other works that have looked at specific marine environments include the works by Han et al. (2019) and Ueland et al. (2017). However, the former used a different sensor family, and the latter worked in a two-dimensional plane and disregarded the third dimension aspect, and neither looked at efficacy at normal operating speeds.

## 3. METHOD

The methods proposed in this paper for localisation and mapping are only utilising data from one lidar sensor, meaning that the desired task is to find an estimate for translating and rotating sets of points between frames. The methods used in this paper are based on a combination of the normal distribution transform (NDT) and the iterative-closest-point (ICP) algorithms. This paper utilised particular versions of NDT and ICP, namely NDT-OMP and ICP-OMP supporting parallel computation. NDT was first introduced by Rosenblatt (1952) concerning multivariate transformation. It is a common registration

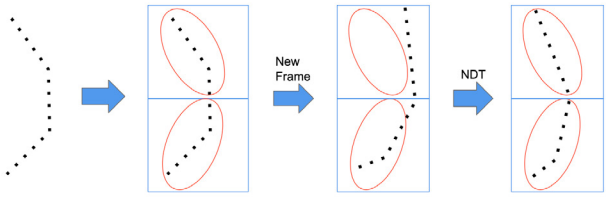


Fig. 1. Two-dimensional illustration of the NDT algorithm. Points from the reference frame are divided into multiple voxels. The red ellipses illustrates the Gaussian distribution in each voxel generated by the points from the reference frame. When the new target frame is generated it is then matched with the Gaussian distribution from the reference frame. A transformation is then performed between the target and reference frame, maximising the overlapping score.

algorithm that is used in many fields for aligning poses. The NDT algorithm provides a *rough registration* of point clouds by matching the target cloud to a reference cloud according to a Gaussian distribution. The Gaussian distribution is generated by splitting up points from the reference cloud into voxels, where a voxel is a three-dimensional equivalent of a pixel, see Fig. 1. The advantage of using the NDT algorithm is that it provides a computationally fast estimate compared to other methods (Huang et al., 2021). Due to NDTs averaging effect, it is also robust to noisy data. However, the performance of the algorithm is sensitive to the size selection of the voxels, meaning that the voxel size needs to be tuned depending on both the sensor and environmental conditions. For data collected in this work, a voxel size of $4.5 \times 4.5 \times 4.5$ [$\mathrm{m}^3$] was used. In previous work (Magnusson et al., 2015), NDT has been concluded to provide accurate results when faced with scan data that has little overlap and weak geometric features. It is therefore, in this work, used as an initial transformation before continuing further with more precise alignments, see Algorithm 1.

ICP is a *refine registration* point-wise algorithm that operates by matching different frames. The principle of ICP is to compute the refine transformation between the reference and target point cloud by minimising the error of point-wise distances (Besl and McKay, 1992). This is performed by minimising the sum of squared differences between the coordinates of the matched pairs through iteration, where the algorithm searches through a combination of rotation and translation. By iteratively updating and refining the relative poses, ICP is able to provide a more precise fit compared to NDT, when there is a sufficient overlap between the two scans. However, the drawback is that it is computationally heavy, especially when working with dense clouds. Due to the non-convexity of the ICP optimisation, the performance of the algorithm is strongly dependent on the initial estimate between frames (Maron et al., 2016). This means that if the poses in the ICP algorithm are too far apart, the algorithm may fail to converge. In order for ICP to work efficiently the alignment of two frames must have a sufficient overlap. Therefore, NDT was executed prior to ICP as an initial estimation before further refinement. It is noted that ICP does not require having a point-wise correspondence since

**Algorithm 1** Natural distribution transform (NDT)

**Input:** $P_r$, $P_t$
**Output:** $P_t$, $T$
(1) Divide $P_r$ into voxels of a specified size
(2) Initialise $T$
(3) Compute the mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ from the points of each voxel
(4) Compute $T$
(5) Map $P_r$ to $P_t$ by rotating and translating each point in $P_r$ according to $T$
(6) Calculate the probability density of each point $\boldsymbol{x_i} \in P_{\text{tar}}$ according to:

$$p(\boldsymbol{x_i}) = \frac{1}{(2\pi)^{\frac{3}{2}}\sqrt{|\boldsymbol{\Sigma}|}} \exp\left(\frac{-(\boldsymbol{x_i}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x_i}-\boldsymbol{\mu})}{2}\right)$$

(7) Calculate the sum of the score of the probability density of each voxel according to:

$$\Psi = \sum_i p(\boldsymbol{x_i})$$

(8) **if** ($\Psi$ is not the smallest)
Use the Newton–Raphson method to compute the new transformation parameters of $T$. Restart from (4)
(9) **return** $P_t$, $T$

**Algorithm 2** Iterative-closest-point

**Input:** $P_r$
**Output:** $P_t$, $T$
(1) Match $P_r$ with $P_t$
(2) Find $T$ by minimising the root mean square cost function:

$$\boldsymbol{R}^*, \boldsymbol{t}^* = \arg\min_{\boldsymbol{R},\boldsymbol{t}} \frac{1}{|P_r|} \sum_i \left|\left|P_t^i - (\boldsymbol{R}\cdot P_r^i + \boldsymbol{t})\right|\right|^2$$

(3) Map $P_r$ to $P_t$ by rotating and translating each point in $P_r$ according to $T$
(4) **if** ($\Delta d >$ `THRESHOLD` or $n_{iter} <$ `MAX_ITER`)
where $\Delta d >$ denotes the change in mean distance. Restart from (2)
(5) **return** $P_t$, $T$

it automatically takes care of differently sized point clouds due to using the closest pair, see Algorithm 2.

For the proposed method, the reference point cloud $P_r$ is defined as the previous frame and the target point cloud $P_t$ as the latest frame. Both frames contain an arbitrary amount of points (not necessary an equal amount). $T$, as seen in Eq. 1, is a $4\times4$ transformation matrix that consists of rotation and translation data, describing how points are transformed from $P_r$ to $P_t$.

$$T = \begin{pmatrix} R_0 & R_1 & R_2 & t_x \\ R_3 & R_4 & R_5 & t_y \\ R_6 & R_7 & R_8 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1}$$

Furthermore, the mean $\boldsymbol{\mu}$ of a voxel is defined as

$$\boldsymbol{\mu} = \frac{1}{|P_v|}\sum_i \boldsymbol{x_i} \tag{2}$$

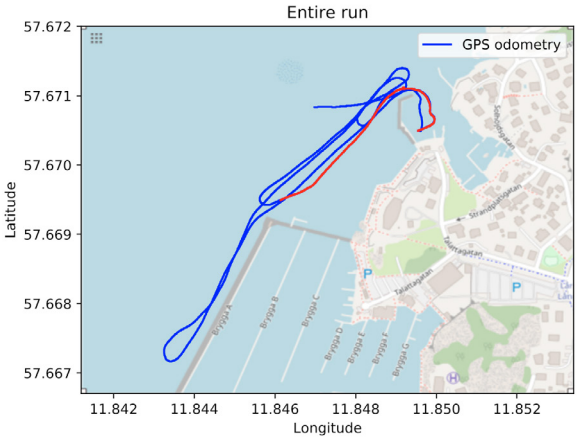


Fig. 2. Path followed in first run. The red path in the first run highlights where the results are evaluated.

and its covariance matrix as

$$\boldsymbol{\Sigma} = \frac{1}{|P_v|}\sum_i (\boldsymbol{x_i}-\boldsymbol{\mu})(\boldsymbol{x_i}-\boldsymbol{\mu})^T \tag{3}$$

where $P_v$ denotes the point cloud inside a voxel, $\boldsymbol{x_i} \in P_v$ is a three-dimensional point, and $|P_v|$ denotes the total number of points in the particular voxel. The final traversal of the point cloud is then performed by combining the transformation matrices of the NDT and ICP algorithm, where the map is generated by translating the target cloud $P_r$ accordingly.

NDT and ICP were implemented using the PCL library (Rusu and Cousins, 2011), using the *DIRECT7* and *KD-tree*-based neighbouring search methods (Magnusson, 2009, p.42). $T$ was initialised to the identity matrix $I$ for Algorithm 1. The parameters `THRESHOLD` and `MAX_ITER` were set to 5 and 200, respectively, for Algorithm 2. An overview of both algorithms is outlined in the following sections.

### *3.1 Experimental setup*

In order to evaluate the proposed algorithms, an experiment was run along the coastline of Gothenburg, see Fig. 2. The run took 25 min and included multiple loops where the vessel's velocity varied between 0.5 m s$^{-1}$ to 3.7 m s$^{-1}$. The total travelled distance was 3086 m. The aim was to analyse the algorithms in multiple scenarios, with a primary focus on testing the function in the relatively calm bay area. This was done in order to conclude how the point cloud registration would perform under mild conditions before moving on to rougher waters. The boat was equipped with sensors on a rigid and compact tripod, as illustrated in Fig. 3.

The lidar used in the experiment was an Ouster OS-2-128, achieving a resolution of 128 beams in the vertical axis and 1024 increments in the horizontal plane at a frequency of 10 Hz. The data was preprocessed using several filtering steps in order to down-sample the data and remove statistical outliers. The preprocessing step includes random down-sampling of the point cloud to 15 %, followed by a voxelgrid down-sampling with a voxel size of 35 cm. Along with the gathered point cloud data, a single antenna

Fig. 3. The sensor platform setup containing an Ouster-OS2-128 lidar and GNSS antenna.

GNSS was used as a ground truth to the measurements. Data capture and hardware interfacing was achieved using the OpenDLV software framework (Benderius and Berger, 2022) powered by libcluon. [1]

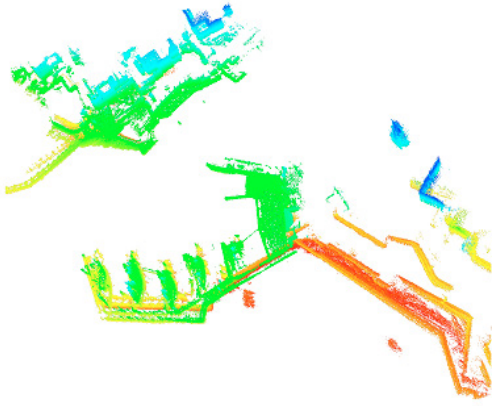

Fig. 4. Diagonal view of the resulting point cloud gathered from the coastline area.

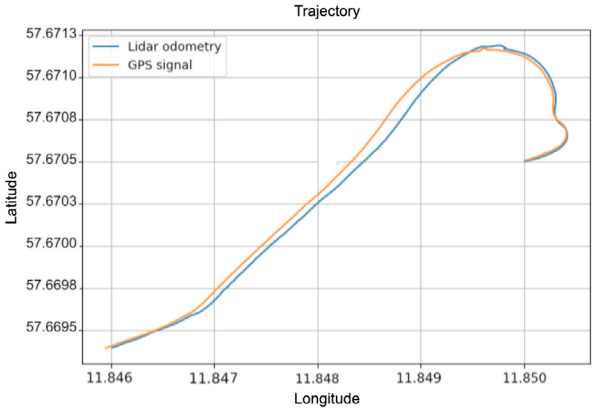


Fig. 5. Lidar odometry compared to the GNSS signal.

[1] https://github.com/chrberger/libcluon

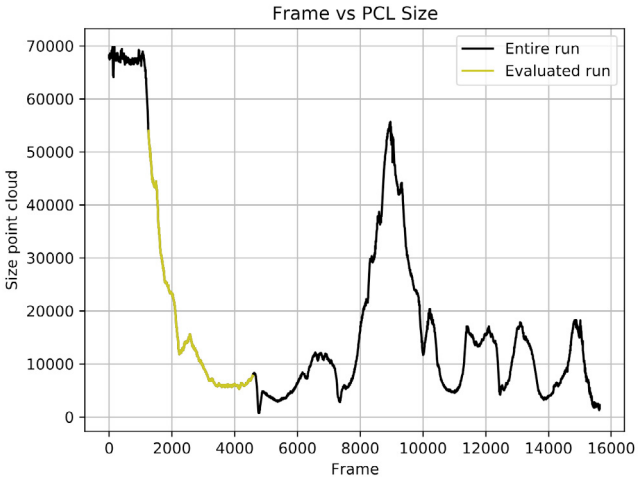


Fig. 6. Size of the point clouds over different frames. The green curve highlights where the run in Fig. 5 was evaluated.

## 4. RESULTS

With the proposed method, the function successfully managed to track the odometry of the lidar in regions with a sufficient amount of points. The measurements were most accurate in close proximity to features such as other vessels, docks, and other points of reference such as buildings, see Fig. 4. The results were validated by comparing the odometry of the lidar with the tracked GNSS data. As the SLAM algorithm does not use any further sensor information from the GNSS, like heading, the lidar odometry had to be rotated in order to fit the trajectory from the localisation system. In Fig. 5, lidar odometry from a 360 m long strip of coastline is presented, and in Fig. 6 it can be seen how the observed features varied along the run. A top-down view of the generated map was used to verify the measurements. It was compared to chart data taken from OpenSeaMap (OpenStreetMap contributors, 2017), see Fig. 7.

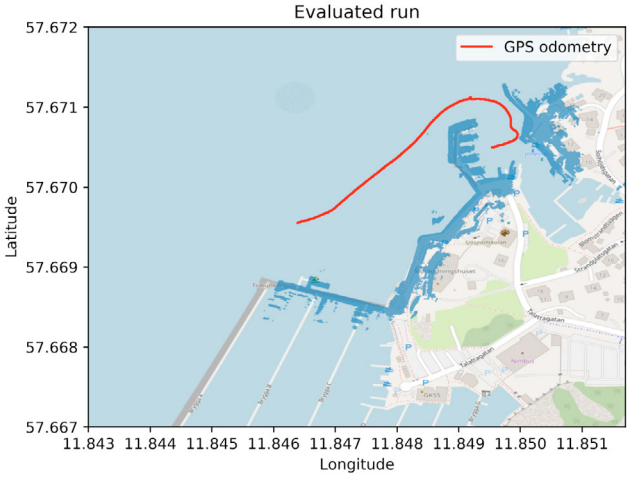


Fig. 7. Top view of the registered point cloud aligned with the corresponding coastline. The red curve shows the path that was traversed for the experiment.

To investigate the behaviour and accuracy of the algorithm under different velocities, a plot of the SLAM deviation

was compared against a plot of the vessel speed in Fig. 8. The deviation was computed by taking the square root distance in the Cartesian plane between the lidar odometry and the GNSS data. It can be observed that with increased speeds, the deviation tends to increase.

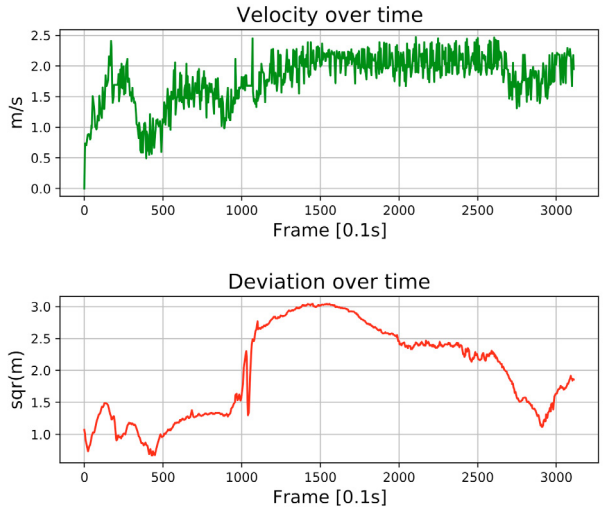


Fig. 8. Deviations between lidar odometry and ground truth over time compared to the velocity.

The deviation of the lidar z-position was plotted over the entire run in Fig. 9 together with the drift. It can be observed that the deviation is accumulated over time, and that the drift was most stable between frames 0 to 1,450.

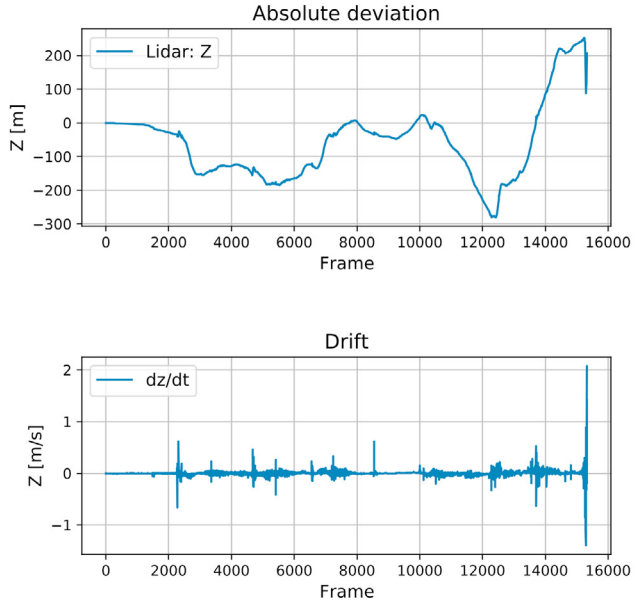


Fig. 9. The top panel show the lidar deviation [m] over the entire run, while the bottom panel show the drift [$\mathrm{m\,s^{-1}}$].

## 5. DISCUSSION AND CONCLUSION

With the proposed method, it was possible to successfully map out a three dimensional environment around the coastline, given that enough features were present. Fig. 4 depicts a diagonal view of the area. The method looks promising around areas with enough reference points. Although deviations were higher, this also performed well for sections with velocities of up to 2.5 $\mathrm{m\,s^{-1}}$. Looking at the trajectory plot in Fig. 5, it is evident that the lidar odometry in the horizontal plane is fairly accurate compared to the GNSS signal with a root mean square error of 2.11 m over a total driven distance of around 360 m.

While evaluating the point cloud from a side view, a drift was noticed in the vertical axis that accumulated over time. Arguably, two sources of error can be connected to these deviations. As the kinematic state of the vessel is not exactly aligned with the global world frame, slight errors in the initial coordinate system could be observed. As the function only relied on point cloud data, it is assumed that the initial position matches the global frame. Even with a slight offset, however, will result in vertical drift over time, as was illustrated in Fig. 9. Another source of error occurs when the lidar detects relatively few features. As the point cloud gets more sparse, the alignment of point cloud frames gets less accurate. If on a given frame the pitch angle gets adjusted, this adds onto the described offset and thereby distort the lidar kinematic state. By comparing Fig. 6 and Fig. 9, it can be concluded that the drift is less significant when the lidar registers a denser point cloud.

While the deviations in the z-axis result in some drift over time, they are constrained to situations with few features and observed at smaller time windows, for example when passing other vessels, the algorithm performs well. For the problem of an offset in the reference frame, a stabiliser like a motorised gimbal can be used in order to keep the lidar at a steady angle in all three dimensions. Furthermore, in a real world application, the system can be improved by performing sensor fusion with an altimeter and an inertial measurement unit in order to keep track of the kinematic changes. This could be accomplished by implementing an *unscented Kalman Filter* (UKF) to estimate the vessel pose. Adding an UKF into the system involves using a boat motion model to predict the pose, to then be automatically corrected in the filter using the measurements and proposed algorithm. Nevertheless, the minimalistic method as presented here aim to show general feasibility of the involved algorithms, and is not directly approaching real-world applications.

As the test platform moved from an asymmetrical to a symmetrical environment (typically as the nose of the boat angled away from the shoreline, and obscuring the lidar to everything except water), there were significant deviations between the GNSS and lidar odometry, regardless of operating speed. For the envisioned working environment of this system, being smaller, high-impulse watercraft working alongside superstructures or in confined marinas, the likelihood of a lidar losing sight of a reference object is unlikely, however, further research could reduce this error margin if it occurs. As noted previously, Nubert et al. (2022), proposed a learning system for symmetrical environments, and this method is a potential candidate for such a solution.

Lastly, and arguably the most important is that the analysis of the environment and odometry was done at post-processing. The next major hurdle for this system

would be to implement it as a real time mapping system, and then finally use this for real time navigation and obstacle avoidance.

## ACKNOWLEDGEMENTS

This research is sponsored by the project *Referensdata och algoritmer till stöd för forskning och utveckling av smarta fartyg* in Sweden (Trafikverket, TRV 2019/1201031). Experimental support was given by the research laboratory Revere at Chalmers University of Technology, as part of preparatory work for the Reeds benchmarking dataset.

## REFERENCES

Benderius, O. and Berger, C. (2022). Opendlv. `https://opendlv.org/`.

Benderius, O., Berger, C., and Blanch, K. (2021). Are we ready for beyond-application high-volume data? the reeds robot perception benchmark dataset. *arXiv preprint arXiv:2109.08250*.

Besl, P.J. and McKay, N.D. (1992). Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, 586–606. Spie.

Han, J., Cho, Y., and Kim, J. (2019). Coastal slam with marine radar for usv operation in gps-restricted situations. *IEEE Journal of Oceanic Engineering*, 44(2), 300–309.

Huang, F., Wen, W., Zhang, J., and Hsu, L.T. (2021). Point wise or feature wise? benchmark comparison of public available lidar odometry algorithms in urban canyons. *arXiv preprint arXiv:2104.05203*.

Hudson, N., Talbot, F., Cox, M., Williams, J., Hines, T., Pitt, A., Wood, B., Frousheger, D., Surdo, K.L., Molnar, T., et al. (2021). Heterogeneous ground and air platforms, homogeneous sensing: Team csiro data61's approach to the darpa subterranean challenge. *arXiv preprint arXiv:2104.09053*.

Khattak, S., Nguyen, H., Mascarich, F., Dang, T., and Alexis, K. (2020). Complementary multi–modal sensor fusion for resilient robot pose estimation in subterranean environments. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, 1024–1029. IEEE.

Magnusson, M. (2009). *The three-dimensional normal-distributions transform: an efficient representation for registration, surface analysis, and loop detection*. Ph.D. thesis, Örebro universitet.

Magnusson, M., Vaskevicius, N., Stoyanov, T., Pathak, K., and Birk, A. (2015). Beyond points: Evaluating recent 3d scan-matching algorithms. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 3631–3637. IEEE.

Maron, H., Dym, N., Kezurer, I., Kovalsky, S., and Lipman, Y. (2016). Point registration via efficient convex relaxation. *ACM Transactions on Graphics (TOG)*, 35(4), 1–12.

Nubert, J., Walther, E., Khattak, S., and Hutter, M. (2022). Learning-based localizability estimation for robust lidar localization. *arXiv preprint arXiv:2203.05698*.

OpenStreetMap contributors (2017). Planet dump retrieved from https://planet.osm.org . `https://www.openstreetmap.org`.

Papadopoulos, G., Kurniawati, H., Shariff, A.S.B.M., Wong, L.J., and Patrikalakis, N.M. (2014). Experiments on surface reconstruction for partially submerged marine structures. *Journal of field robotics*, 31(2), 225–244.

Rosenblatt, M. (1952). Remarks on a multivariate transformation. *The annals of mathematical statistics*, 23(3), 470–472.

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

Ueland, E.S., Skjetne, R., and Dahl, A.R. (2017). Marine autonomous exploration using a lidar and slam. In *International Conference on Offshore Mechanics and Arctic Engineering*, volume 57724, V006T05A029. American Society of Mechanical Engineers.