IEEE Robotics and Automation Letters (RAL) paper presented at
2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

GEERS: Georeferenced Enhanced EKF Using Point Cloud Registration and Segmentation

Rui Bettencourt[1], John Lewis[1], Rodrigo Serra[1], Meysam Basiri[1], Alberto Vale[2], and Pedro U. Lima[1]

***Abstract*—Georeferenced Enhanced EKF using point cloud Registration and Segmentation (GEERS) is a high-accuracy and consistent-rate localization method for outdoor robots. The localization is estimated by an EKF that fuses wheel odometry, IMU and GNSS measurements, in addition to feedback corrections from a registration step. The method improves localization accuracy by registering range sensors with pre-obtained georeferenced 3D maps and providing feedback corrections to the EKF. The continuous fusion of GNSS measurements naturally provides an initial estimate and reduces kidnapped robot situations in symmetric environments. The proposed method can integrate any range sensor (such as RBG-D cameras or 2D and 3D LiDAR). Experimental results in a real-world solar farm, its simulated digital twin, and an open dataset demonstrate localization accuracy improvements. Real-world experiments on a solar farm demonstrated the flexibility and reliability of the proposed method, exposing its advantages towards GNSS-only-based approaches.**

***Index Terms*—Field Robots, Localization, Sensor Fusion, Range Sensing**

I. Introduction

GLOBAL Navigation Satellite System (GNSS) offers a low-cost solution for outdoor robot localization. However, GNSS sensory measurements are error-prone and location-sensitive. Promising solutions, such as using Real-Time Kinematic (RTK) systems, mitigate GNSS errors but do not eliminate them, particularly in environments with sizeable metallic structures [1]. Consequently, to improve localization accuracy in such scenarios, GNSS sensor data can be fused with wheel odometry and Inertial Measurement Unit (IMU) data [2, 3]. However, odometry often fails on uneven terrains for Unmanned Ground Robots (UGVs), and electromagnetic interferences (EMI) can impact IMU data.

In environments with noisy GNSS data, range sensors such as stereo cameras [4, 5], depth cameras [6], or 3D LiDARs [7–10] can be used to improve localization. Existing 3D sensor-based SLAM (Simultaneous Localization and Mapping) techniques [11, 12] generate a 3D spatial map while localizing the agent in the generated map. SLAM approaches can also be utilized to produce LiDAR odometry to be fused in the localization [13–15]. However, given the high computational load of SLAM and 3D sensory measurements [16], SLAM techniques are not preferred in large outdoor environments that are known *a priori*. These issues are addressed by performing mapping from SLAM methods beforehand and later using the generated map for real-time localization. On the other hand, pure 3D approaches to localization may fail in symmetric environments [17].

To integrate GNSS measurements together with range sensors, using a georeferenced 3D map, where all points have corresponding Universal Transverse Mercator (UTM) coordinates, helps transforming points between frames. Previous works such as [18] presented georeferenciation techniques that considered the GNSS at the start of the mapping. Thus, an erroneous initial GNSS can lead to imprecise georeferenciation. A common approach to reduce GNSS error is using the Least Square Method (LSM) [19] in the georeferenciation.

An approach by Perez *et al.* [6] proposes expanding traditional Adaptive Monte Carlo Localization (AMCL) to 3D environments using a 3D map, data from an RGB-D sensor, and radio-based beacons placed in known positions. This approach uses visual odometry based on a camera (prediction step), point cloud matching and radio-based beacons (update step) to localize in real time, but it requires an initial estimate. Therefore, the method fails to recover from the "kidnapped robot" problem without the presence of these beacons. Carrasco *et al.* [20] made one extension of this work by replacing the RGB-D sensor with a 3D LiDAR.

A different approach by Caballero [9] uses a non-linear optimization formulation to estimate the 3D pose of the robot that does not rely on a Bayesian filter. Thus, the solution of the registration between the 3D LiDAR and the 3D map is directly used as a pose estimate.

Other methods that use GNSS data have been proposed. John *et al.* [4] proposed a method that uses GNSS to initialize the estimation, and then solely uses optimization and stereo image registration to improve localization. Given the lack of a global localization, if registration fails at any point ("kidnapped robot"), then it is unlikely it will be able to recover. An extension of this work [5] makes the optimization more efficient and introduces an extra bootstrapping phase to stabilize the GNSS data in the initialization phase. Despite these improvements, the "kidnapped robot" problem persists,

Manuscript received: July, 24, 2023; Revised November, 9, 2023; Accepted December, 13, 2023.

This paper was recommended for publication by Editor Pauline Pounds upon evaluation of the Associate Editor and Reviewers' comments.

*This work was supported by the Portuguese Foundation for Science and Technology (FCT) under the grants 2021.06082.BD and UI/BD/153758/2022, by LARSyS strategic funding under FCT Project UID/50009/2020 and through the DURABLE project, under the Interreg Atlantic Area Programme through the European Regional Development Fund (ERDF) from the European Commission

[1]R. Bettencourt, J. Lewis, R. Serra, M. Basiri and P. Lima are with the Institute for Systems and Robotics, Instituto Superior Técnico, Universidade de Lisboa, Lisbon 1049-001, Portugal (e-mail: {*rui.bettencourt; john.lewis; rodrigo.serra; meysam.basiri; pedro.lima*}*@tecnico.ulisboa.pt*).

[2]A. Vale is with the Instituto de Plasmas e Fusão Nuclear, Instituto Superior Técnico, Universidade de Lisboa, Lisbon 1049-001, Portugal (e-mail: *avale@ipfn.tecnico.ulisboa.pt*).

Digital Object Identifier (DOI): see top of this page.

especially in a symmetric environment

One additional method proposed by Zhang *et al.* [10] uses a georeferenced 3D map and performs registration between LiDAR and the map, together with GNSS data to perform localization through the use of an EKF. However, the mapping is georeferenced with the reading at each instant, which can induce errors in the map. The use of a neural network for registration results in a slow rate of $\sim 1\ Hz$, with error values around 2.19 meters. Another EKF method fusing GNSS and LiDAR data, as well as IMU data, can be found in [14]. This method develops a SLAM-based LiDAR odometry and is highly dependent on it, failing in symmetric environments.

Autonomous driving research has presented many developments towards outdoor localization, especially the fusion of GNSS and LiDAR measurements. However, most of these advancements are centered around SLAM methods [21–23]. Nevertheless, there are alternative localization methods in this area, such as [15]. Through graph optimization, this method fuses GNSS signals with LiDAR odometry (LO) alongside a neural network method, generating single-shot global localization estimates. This approach demands large computational resources and relies on asymmetric, feature-rich environments for both the LO and global localization.

In an effort to ensure consistently accurate localization estimates, irrespective of the incoming 3D data or erroneous GNSS measurements, we propose Georeferenced Enhanced EKF using point cloud Registration and Segmentation (GEERS). To minimize GNSS errors, GEERS uses an LSM-based georeferenciation method [19] instead of using an individual isolated GNSS measurement. GEERS works in real-time and has a high rate due to the use of an EKF and ICP, which are lighter than particle filters or neural networks [10, 15]. Unlike [6, 20], GEERS does not depend on an accurate pose initialization. Continuous fusion of GNSS measurements ensures not only an approximate initial estimate but also reduces the "kidnapped robot" problem, unlike [4, 5]. In contrast to [9], GEERS is robust to registration failures, as the EKF filters out registration outliers. Finally, by allowing the use of any range sensor, the method presents substantial flexibility when compared to most methods that are sensor-specific.

The main contributions of the proposed method are:

- A continuous closed-loop feedback localization technique, utilizing 3D sensors, for large environments with error-prone data from GPS, IMU and odometry. The proposed method is flexibly modular and can be run on any robot with any 3D sensor configuration.

- An implementation of an Extended Kalman Filter (EKF) feedback loop that utilizes an auxiliary corrective input from processed 3D spatial information, used in conjunction with a pre-attained georeferenced 3D map.

- An Adaptive Spherical Voxelization technique that ensures consistent runtime irrespective of the 3D map size or the spatial information perceived by 3D sensors.

This paper is organized as follows: Section II explains the methods used for offline pre-processing, and Section III describes the proposed GEERS method. Section IV discusses the results obtained, and Section V presents our conclusions and future work.

II. Offline Pre-Processing

GEERS is an online localization method that depends on prior 3D mapping and georeferenciation of the map. In this section, we discuss the methods used to achieve this initialization.

A. 3D Mapping - LOAM

To generate a 3D map of the world, we use an offline SLAM method called LiDAR Odometry and Mapping (LOAM) [12]. LOAM results in the 3D point cloud map of the scene, along with the estimated path taken by the robot to build the map. This map is saved for later use as the static map of the world. An example of a 3D map obtained from a solar farm can be seen in Fig. 1.

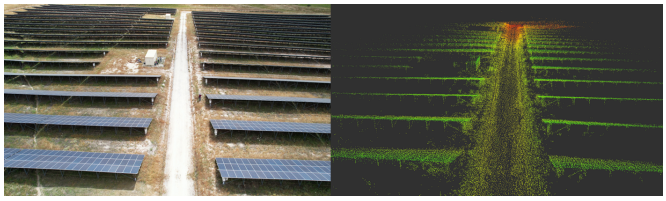


Fig. 1: Top view of the solar farm used for testing (on the left) and its 3D map obtained from LOAM [12] (on the right).

B. Georeference

To georeference the map, an affine transformation is obtained. This transformation is applied to every point of the map, transforming them from the local map frame to the UTM frame, in the form of coordinates. The GNSS coordinates of the robot's trajectory in the UTM frame are acquired when navigating the environment for the mapping performed in Section II-A. From this mapping process, we can also obtain the LOAM estimated path the robot took (poses in the local map frame). We can then match these positions to get the transformation [19]. Note that by using multiple GNSS positions helps reduce errors by averaging out random components of the GNSS noise, mostly related to interference from infrastructures and limitations of the GNSS receiver. Systematic errors are not minimized with this process.

III. Georeferenced Enhanced EKF using point cloud Registration and Segmentation

The necessary offline processes required for GEERS have been explained in Section II. This section explains the proposed GEERS framework (illustrated in Fig. 2). Transformations to the 3D map frame are described in Section III-A, and Adaptive Spherical Voxelization is proposed in Section III-B to address the computational constraints of point cloud processing. The registration component is described in Section III-C, and the final EKF sensor fusion is explained in Section III-D.

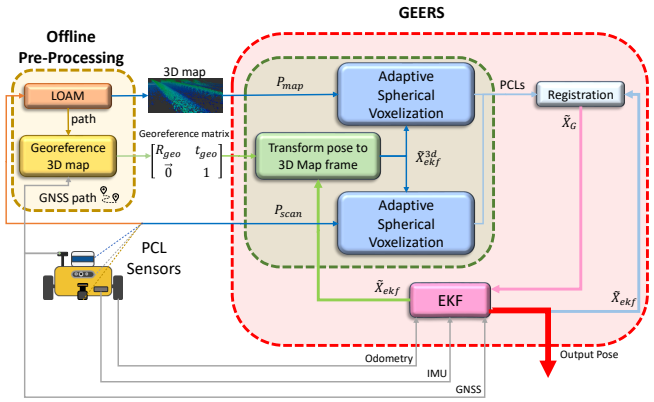


Fig. 2: Block diagram of GEERS.

A. Transform pose to 3D Map frame

The pose estimate of the EKF, $\tilde{X}_{ekf}$, is represented in its local frame and the UTM frame. This pose is transformed to $\tilde{X}^{3d}_{ekf}$, the EKF estimate in the 3D map frame, by using T_{geo}.

A transformation T_{tx} is performed on every new update of P_{scan} to ensure that all computations are carried out in the same frame of reference. T_{tx} comprises the translation matrix t_{ekf} and rotation matrix R_{ekf}[24], as formulated in Eq. (1), where t_{ekf} and R_{ekf} are generated from $\tilde{X}^{3d}_{ekf}$.

$$T_{tx}P_{scan} = \begin{bmatrix} R_{ekf} & t_{ekf} \\ \overrightarrow{0} & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & z_n & 1 \end{bmatrix}' \tag{1}$$

B. Adaptive Spherical Voxelization

As explained in Section II-A, a 3D point cloud map of the environment, P_{map}, is acquired beforehand. GEERS corrects the EKF's estimate by registering the scan acquired from range sensors, P_{scan}, in P_{map}. However, registration algorithms are computationally dependent on the number of points involved in each point cloud (n_{scan} and n_{map}). It would be computationally exhausting to run registration on the whole 3D map of the area (n_{map} around 260000 points for the used maps) for each new scan acquired (n_{scan} around 25000 points for a 3D LiDAR). Added to this, n_{scan} is spatially variant and depends on the perceived environment. A varying n_{scan} can lead to varying computational speed. Deviation from a consistent rate can result in adverse consequences, leading to incorrect EKF estimates.

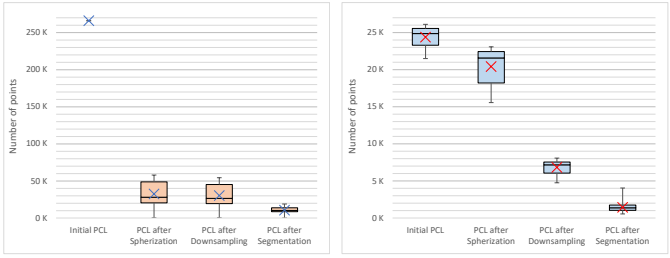
The importance of a consistent and reduced point cloud size can be seen in the improvement in the computation efficiency of [8] compared to its predecessor [7]. In [8], the authors emphasize the importance of maintaining a consistent point cloud size to improve the computational efficiency of LOCUS 2.0. The adaptive voxel grid filter, as proposed in [8], downsamples the incoming scans to a predefined desired size. This approach is favorable when building maps, as outliers can be filtered out in subsequent iterations. Since our primary objective is to improve localization accuracy in a known $3D$ space, achieved by registering P_{scan} with P_{map}, outliers can cumulatively increase the covariance of the localization estimate. Thus, we need a method to remove outliers and minimize both the spatial and computational sizes of P_{scan} and P_{map}. In Section III-B1 and Section III-B2, we elaborate on our approach called Adaptive Spherical Voxelization (ASV) and highlight its benefit.

1) Spherization, Outlier Removal and Downsampling: $\tilde{X}^{3d}_{ekf}$ reliably places the robot within an ellipsoid of uncertainty, proportional to EKF's covariance, C_{ekf}. To increase the accuracy, the eventual registration focuses primarily on an approximate area of interest in the shape of a sphere. An euclidean ball is generated, centered around $\tilde{X}^{3d}_{ekf}$, on both P_{scan} and P_{map}. Points outside these spheres, of radii r_{scan} and r_{map} respectively, are rejected. This method, spherization, relies heavily on the confidence and accuracy of $\tilde{X}^{3d}_{ekf}$. To overcome minor changes in C_{ekf}, we have enabled bounded adaptive tuning of r_{scan}, as shown in Eq. (2). This ensures that radii are bounded within (r^{min}_{scan},r^{max}_{scan}) and proportional to the maximum spatial uncertainty of $\tilde{X}^{3d}_{ekf}$. The adaptive calculations in Eq. (2) use the maximum value of x, y and z variances from C_{ekf} as the variance V^{max}_{ekf}. To make the eventual registration more robust to cumulative estimate errors, we ensure that r_{map} is greater than r_{scan} by a predefined factor of r_{mul} (≥ 1), as shown in Eq. (3). This ensures that we filter enough features from P_{map}, to ensure overlap during registration. Spherization also implicitly helps in limiting the maximum correspondence distance of the registration algorithm to less than r_{scan}.

$$r_{scan} = \begin{cases} r^{max}_{scan} & \text{if } V^{max}_{ekf} \cdot r_{scan} \geq r^{max}_{scan} \\ r^{min}_{scan} & \text{if } V^{max}_{ekf} \cdot r_{scan} \leq r^{min}_{scan} \\ V^{max}_{ekf} \cdot r_{scan} & \text{otherwise} \end{cases} \tag{2}$$

$$r_{map} = r_{scan} \cdot r_{mul} \qquad \text{where } r_{mul} \geq 1 \tag{3}$$

The resultant point cloud sphere is further trimmed using statistical outlier removal (using nearest neighbor search). The resolution of this point cloud is downsampled while remaining with dominant features. Table I illustrates what happens to P_{map} when performing spherization and downsampling with the transition in Table I between a) and b).



(a) 3D map processing (b) 3D LiDAR scan processing

Fig. 3: The reduction of points through ASV, where X is the 600-iteration average pointcloud size.

A statistical analysis of P_{map} and P_{scan} from a 3D LiDAR scan over the point cloud processing can be seen in Fig. 3. It

is possible to see that performing spherization reduces 87.86% the size of P_{map} and 16.53% of P_{scan}.As for downsampling, it reduces a further 7.06% from P_{map} after spherization and 66.57% of P_{scan}. The ASV consistently reduces the pointcloud size to a saturated minimum, as seen in the low variance after segmentation in Fig. 3.

2) Segmentation: The resultant P_{map} and P_{scan}, though reduced in size, consist of redundant information, predominantly in the form of a ground plane. Limited features in the ground plane can be of little help, if not misleading, for point cloud registration. We are aware of the availability of many ground and semantic segmentation algorithms that involve deep learning and feature extraction [25]. In an effort to keep the overall complexity of the algorithm simple, we have incorporated a simple curvature-based segmentation. The curvature normals, $\vec{N}_{map}$ and $\vec{N}_{scan}$, at each point in P_{map} and P_{scan}, are calculated. Normals along the horizontal axis, greater than a predefined value, are assigned as ground and ignored during registration. Since P_{map} and P_{scan} are spherized, the curvature normals are robust to global curvature changes. Thus, spherization enables a simple and efficient ground plane segmentation. In Table I, we can see an example point cloud that results from segmentation. Segmentation reduced P_{map} on average a further 65.11%, after downsampling, and reduced P_{scan} a further 78.95%, when analyzing Fig. 3.

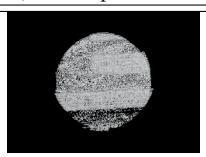
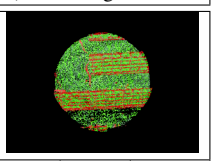
a) Initial Point Cloud	b) After Spherization			c) After Segmentation		
						
Min/Avg/Max	Min	Avg	Max	Min	Avg	Max
2.6e5	385	3.2e4	5.7e4	29	1.0e4	1.9e4

TABLE I: Change in P_{map} through various stages of adaptive spherical voxelization. The table depicts the minimum, average and maximum values of n_{map}, at each stage of ASV, analyzed across 600 iterations. In c), the green points represent the redundant features (primarily ground).

C. Registration

The resultant P_{scan} and P_{map} have reduced size but are feature dominant. A simple point-to-point Iterative Closest Point (ICP) [26] is used in an effort to keep the number of tunable parameters to a minimum. The registration result, T_{icp}, is a 4×4 spatial transformation matrix, that best aligns P_{scan} to P_{map}. T_{icp} can be interpreted as the correction to the EKF's estimate, $\tilde{X}_{ekf}$. The corrected pose, $\tilde{X}_G$, from the GEERS pipeline can be acquired from Eq. (4).

$$\tilde{X}_G = T_{icp} \cdot \tilde{X}_{ekf} \qquad (4) \qquad C_G = {\mathcal{I}_G}^{-1} \qquad (5)$$

Along with $\tilde{X}_G$, we also acquire the Fisher information matrix, $\mathcal{I}_G$. This matrix is a figure of merit of the point cloud alignment. Mathematically [27], it is the approximate inverse of the covariance matrix, C_G (Eq. (5)) [28]. Thus, C_G represents the confidence of the registration along the 6 degrees of freedom, and thus the confidence in the correction $\tilde{X}_G$. $\tilde{X}_G$ and C_G are then fed to the EKF as an auxiliary sensor input. This ensures a seamless fusion with other sensors, as EKF will disregard $\tilde{X}_G$, if the eigenvalues of C_G are over a predefined threshold. In other words, the error is bounded, and this sensor fusion method ensures that the combination can only do better, if not as well as EKF.

D. GEERS' Extended Kalman Filter

In GEERS, an EKF-based sensor fusion is chosen[2], owing to its relatively light computational requirements.

Considering it was implemented for UGVs, the filter has a 2D state space x_k at each time instant k. The state space consists of the position x and y, as well as the yaw rotation, θ, and the corresponding velocities.

The EKF is a non-linear dynamic system described by Eq. (6) from [2], where f is a non-linear state transition function and w_{k-1} is the Gaussian process noise.

$$x_{k+1} = f(x_k) + w_k \qquad (6) \qquad z_k = Hx_k + v_k \qquad (7)$$

The observation models are in the form of Eq. (7), where z_k is the measurement in each time instant k, with Gaussian noise v_k and observation matrix, H. The observation matrix $H = I$, the identity matrix, due to the assumption that each fused sensor produces measurements of the state variables they are estimating.

1) Prediction step: The prediction step is performed through Eqs. (8-10), where f is a standard 3D kinematic model, making the EKF robot-agnostic. Function f is non-linear since additional 3D rotations are performed in Eq. (10) to $\dot{x}_k$ and $\ddot{x}_k$, which are given in the frame of the robot. In Eq. (9), P is the estimated error covariance, projected by the Jacobian of f, F. Q is the added process noise covariance, and in Eq. (10), t is the time interval.

$$\tilde{x}_{k+1} = f(x_k) \qquad (8) \qquad \tilde{P}_{k+1} = FP_kF^T + Q \qquad (9)$$

$$f(x_k) = \begin{cases} x_k + \dot{x}_k t + \frac{1}{2}\ddot{x}_k t^2, & \texttt{if } x_k = \{x, y, \theta\} \\ x_k + \dot{x}_k t, & \texttt{if } x_k = \{\dot{x}, \dot{y}, \dot{\theta}\} \end{cases} \qquad (10)$$

2) Matching Step: The matching step, as explained in section III-C, results in a pose estimate $\tilde{X}_G$ with an associated covariance C_G. This represents the robot's corrected pose, according to GEERS. This is then used in the update step as an auxiliary sensor measurement. The connection between the registration and the EKF is depicted in Fig. 2. Minor outliers are filtered out by not accepting $\tilde{X}_G$ corrections with low confidence or that are further γ meters from $\tilde{X}_{ekf}$ ($|\tilde{X}_G - \tilde{X}_{ekf}| \geq \gamma$), where γ is a predefined threshold.

3) Update Step: The update step is then carried out for every measurement source configured using Eqs. (11-13).

$$K = \hat{P}_k H^T (H\hat{P}_k H^T + R)^{-1} \qquad (11)$$

$$x_k = \hat{x}_k + K(z - H\hat{x}_k) \qquad (12)$$

$$P_k = (I - KH)\hat{P}_k (I - KH)^T + KRK^T \qquad (13)$$

The measurement sources used are the odometry velocities (which provide high-rate measurements of the robot

movement), IMU data (which uses a compass to update the orientation) and GNSS data (converted to the robot's local frame to update x and y). The matching correction X_G is also used as a measurement source in the update step for the position and orientation. The resultant EKF estimate x_k at any time k, $\tilde{X}_{ekf}$, is used to estimate the location of the robot with an associated covariance of C_{ekf}. This estimate is then fed back as an input to Section III-A.

IV. Results

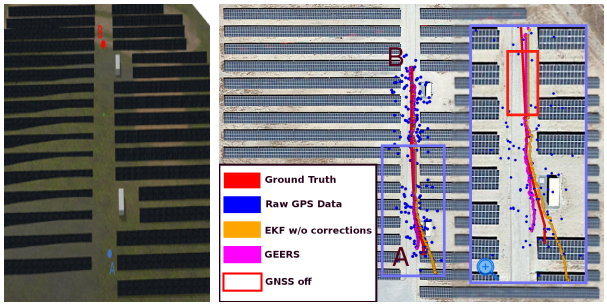
In this section, we present the results[1] comparing the localization accuracy and computational rate of GEERS against EKF fusing only odometry, IMU and GNSS. Two robot setups are used for the experiments: 1)A Jackal robot from Clearpath Robotics that has odometry, a GNSS sensor, an IMU and a Velodyne Puck with 16 layers; 2)A refurbished ATRV-JR from iRobot that has odometry, a Real-Time Kinematic (RTK) GNSS, an IMU, an Hokuyo 2D LiDAR and a RealSense D435 RBG-D camera. All experiments are carried out on an Intel i7 10^{th} Gen processor with 16 Gb of RAM.

The localization environment plays a vital role in tuning the parameters of GEERS. For our simulations and real-world runs, a solar farm was chosen. Solar panel farms offer a structurally symmetric environment, emphasizing the importance of fusing information from GNSS, IMU, and odometry. It also has rigid structures that are vital for point cloud registration. Additionally, the metallic structures and exposure to higher EMI induce dominant errors in the GNSS and IMU readings. These factors, added to the uneven ground, make sensor fusion essential for successful localization. Additional results for longer distances are also presented in section IV-C. These results were obtained from the publicly available dataset - University of Michigan North Campus long-term (NCLT) [29]. We have tuned the parameters of GEERS with $(r^{min}_{scan}, r^{max}_{scan}, r_{mul})$ as $(10, 30, 2)$. The point cloud is downsampled to a voxel size of 0.25 meter.

In order to incorporate different range sensors across varying robot platforms, we have prioritized the modularity and portability of GEERS. To ensure this, GEERS is integrated with well-known Robotic Operating System (ROS) methods *robot_localization* and *move_base* for localization and navigation, respectively. For the 3D mapping, the A-LOAM[2] implementation was chosen.

A. Simulation Results

The simulations are carried out in a digital twin of a solar farm in a gazebo environment, as shown in Fig.4a. For all simulation experiments, the robot is initialized at ***A*** and is tasked to navigate to ***B***. We have mapped the simulated GNSS coordinates to the real-world solar farm. Using the simulated GNSS sensor and the simulated LiDAR, a georeferenced 3D map was generated from the simulation using the methods described in section II. This mapping has a slight offset, as is evident from the slight delta in ***A*** and ***B*** in Fig.4b w.r.t Fig.4a. The wheel odometry and IMU sensor information are virtually error-free in simulation, thus EKF tends to perform better in simulations.



(a) (b)

Fig. 4: (a) - Sample Gazebo Simulation environment; (b) - GNSS denied simulation run, from A to B (≈ 115 m)

1) GPS-Denied Simulation: In this experiment, we test the navigation from A to B in a GPS-denied environment. The red box in Fig. 4b represents the area within which GNSS is disabled. The simulation is carried out with a GNSS covariance of 5 m. Upon comparing the distance errors between the estimate and the ground truth, it is clear that EKF (MSE=2.84 m) is less accurate than GEERS (MSE=1.46 m). This is primarily due to GEERS' ability to correct its initial estimate. Analyzing the trajectory, we can observe that GEERS uses the additional 3D information to correct the initial pose. However, EKF requires a few more iterations to converge to the true pose. When GNSS is disabled, both methods converge and thus can maintain an accurate localization estimate, primarily due to error-free odometry and IMU. However, GEERS' path is closer to the ground truth.

2) Varying GNSS Covariance: In this simulation experiment, we vary the GNSS covariance to compare the localization accuracy of GEERS with EKF. We compare the poses of the robot from both methods with the true pose of the robot acquired from the Gazebo environment. Fig. 5 depicts the mean squared error (MSE) of the true pose against the estimated pose, for a varying GNSS covariance averaged across 25 iterations.

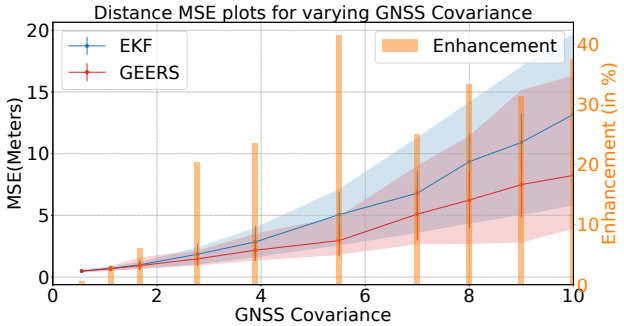


Fig. 5: Comparing the error (in meters) of estimate with groundtruth for varying GNSS covariance.

The line plots depict the average MSE across the covariance, with the shaded regions representing the range between the

[1]https://youtu.be/JX6I0vz2WKU
[2]https://github.com/HKUST-Aerial-Robotics/A-LOAM

minimum and maximum MSE. The orange bar plot depicts GEERS's accuracy enhancement (in percentage) over EKF. For example, GEERS has, on average, 40% less error than EKF for a GNSS covariance of 5.5 meters. We can see a consistent rise in accuracy enhancement with the GNSS covariance, implying that the GEERS consistently outperforms EKF when the GNSS is unreliable. We can also note that the minimum and maximum MSE of GEERS is below that of EKF, thereby ensuring that the localization is bounded within the EKF's limits.

B. Heterogeneous Sensor Fusion in a real Solar Farm

An important aspect of GEERS is that it can use measurements from any range sensor to perform the matching step with the 3D map, as long as they are in a point cloud format. To demonstrate this, the robots were manually teleoperated for 30 meters in a simple straight line along the center of a corridor between two solar panels. This trajectory was chosen due to the GNSS noise caused by the solar panels and so it could easily be plotted on a map, as shown in Fig. 6, under the form of a yellow line. The chosen trajectory has the property that it follows a parallel to the equator, so the ground truth trajectory was obtained by averaging the latitude of the GNSS sensor with RTK corrections (the most accurate sensor available) and visually determining the beginning and the end. This experiment was performed to visually show how the GEERS corrections affect the localization in a real-world experiment and to compare the use of different sensor configurations, as seen in Fig. 7.

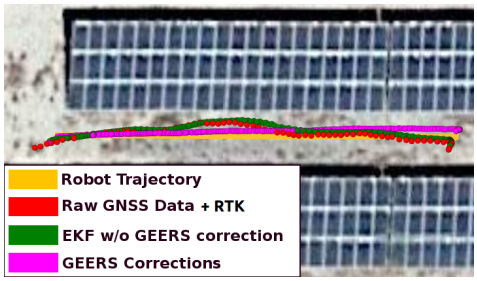


Fig. 6: Trajectory performed by Jackal and its 3D LiDAR.

The results from using GEERS with the different robots with different sensor configurations were compared to this trajectory, and the error was computed along the path. A statistical analysis was performed, which can be seen in Fig. 7. It is worth mentioning that the same georeferenced 3D map was used and obtained from Jackal with the Velodyne in a trajectory different from the one used for localization.

Analyzing Fig. 7 for Jackal, localization with raw GNSS data predictably has a high error. Using EKF to fuse GNSS, odometry and IMU improves the accuracy by 0.4 meter, on average. Using GEERS reduces average localization errors to 0.37 meter without segmentation and 0.28 meter with segmentation. These results show that for this sensor, using segmentation can improve the performance by 24%.

For experiments with ATRV-Jr, an RTK station is set up to correct the incoming GNSS measurements. This increased

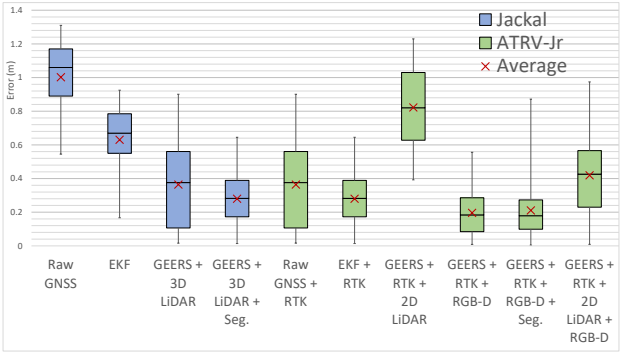


Fig. 7: Comparison between different sensor fusions.

the accuracy of the GNSS measurements to 0.36 meter and 0.28 meter with EKF fusion. When GEERS fuses the RGB-D camera, the accuracy is improved to 0.21 meter with segmentation and to 0.19 meter without segmentation. However, when used with 2D LiDAR, the results under-perform with respect to EKF. This localization accuracy decay is due to a high percentage of outliers (given by the reflective solar panels and the tall vegetation in the field). GEERS with RGB-D camera and 2D LiDAR also underperforms with respect to EKF due to 2D LiDAR outliers.

Table II presents the average GEERS corrections rate for various sensors with and without segmentation. Note that this rate is only for the corrections $\tilde{X}_G$, used to correct the EKF, which runs at a constant 10 Hz. The average pointcloud size across each sensor and various stages of ASV is depicted in Table II. To emphasize the impact of segmentation on the algorithm's rate, the percentage of point reduction achieved by segmentation after Spherization and Downsampling is shown for each sensor. The rate of the algorithm with and without segmentation is also presented. These results are computed from an average of 600 iterations.

TABLE II: Rate of GEERS correction given different sensors

	2D LiDAR	**3D LiDAR**	**RGB-D Camera**
Initial number of points	425	23559	307200
Points after Spherization	411	20425	254311
Points after Downsampling	88	6828	1035
Points after Segmentation	88	1437	829
Segmentation point reduction	—	78.95%	19.95%
Rate without Segmentation (Hz)	5.7	3.0	1.5
Rate with Segmentation (Hz)	—	3.5	1.6

As expected, there is a trade-off between the resolution and frequency of the sensors. With low information sensors, the corrections can be obtained at a higher rate than for sensors with more resolution. Segmentation also increases the algorithm's rate, which is less evident on the RGB-D camera as it's limited field of view records less ground.

C. NCLT Dataset

Cumulative error analysis for long-duration scenarios was carried out using the NCLT dataset [29]. The NCLT dataset

was chosen as it has outdoor data with 3D LiDAR, wheel odometry, IMU data, GNSS data, and a computed groundtruth to perform comparisons. The results were obtained in the first 600 seconds of the session corresponding to January 10th 2013. This corresponds to approximately 660 meters of outdoor navigation in an environment comprising parking lots, trees, and some buildings. This result aims to showcase GEERS performance on a long-distance test in an urban environment and allow for future method comparisons under similar conditions.

A georeferenced map of the world was built using the methods described in section II. The baseline EKF results were obtained with wheel odometry, IMU, and GNSS data. In Fig. 8, it is possible to observe that this baseline solution presents higher error values than the GEERS solution.

Fig. 8: Long distance experiment using the NCLT dataset. EKF without corrections and groundtruth also shown.

A statistical analysis was performed on the results of this dataset, as presented in Table III. In concordance with the previous results, GEERS outperforms the EKF with no registration corrections in every metric. The mean error and RMSE in this experiment are higher than in the previous experiments but still comparable to other state-of-the-art methods, such as the method discussed in [15].

TABLE III: Error comparison on NCLT dataset

	Average (m)	**Std (m)**	**Max (m)**	**RMSE (m)**
EKF w/o GEERS	3.19	2.04	10.98	3.79
GEERS	1.20	0.72	4.64	1.40

A direct comparison can not be made with [15], despite using the same dataset [29], as it is unclear which portion of the dataset was used for the experiments in [15]. Taking that into account, [15] presents 4 results: I) using LO only (RMSE=$8.17m$), II) LO, GNSS, and the global localization (RMSE=$2.19m$), III) these three components together with global graph optimization (RMSE=$1.42m$), and IV) a drift error correction model is added to correct LiDAR odometry (RMSE=$1.21m$). When comparing these results on the NCLT dataset, GEERS (RMSE=$1.4m$) outperforms all results except when the drift error correction model is used. In that case, [15] outperforms GEERS in terms of RMSE by 0.19 meters.

While [15] can provide slightly better results, they use components that are computationally expensive and time-consuming. They are also more error-prone in symmetric environments due to perceptual aliasing, which is not evident in the NCLT dataset. Perceptual aliasing is tackled in GEERS by transforming the sensor information to the last EKF estimate and then applying ASV (section III-B1), which limits the area used in registration.

V. Conclusion

This paper proposes a novel outdoor localization pipeline to tackle outdoor localization. This method, GEERS, utilizes a georeferenced 3D map and point cloud registration to correct the EKF estimates. In structured outdoor environments, where GNSS, odometry, and IMU inputs are highly error-prone, GEERS relies on the 3D sensor inputs for EKF estimate correction. The EKF used for localization fuses wheel odometry, IMU and GNSS to increase reliability in symmetric environments and offset complete reliance on 3D sensor data. The robot avoids the "kidnapped problem" by utilizing GNSS' global localization estimate, and the GNSS error is bounded by the GEERS's 3D sensor fusion. An adaptive spherical voxel filter is introduced to ensure a consistent rate of points for pointcloud registration. This ensures a consistent output rate and helps GEERS to run in real time. Experimental results indicate that computation rates as high as 3.5Hz (for VLP-16) can be achieved.

Localization with GEERS was tested in a real-world solar farm, in a simulated digital twin and in a public dataset [29]. Simulated results show that GEERS achieves better localization than EKF, with increasing simulated GNSS error, with up to 40% improvement in accuracy on simulation, 32% improvement in real-world tests and 37% improvement on NCLT dataset [29]. As for the different sensors used, RGB-D cameras obtained the best results, albeit using RTK and in environments with rich features (the solar panels' legs) close to the sensor. Using a 3D LiDAR resulted likewise in a good performance, with a sensor that has fewer detail measurements but a much larger range and field of view. GEERS is prone to perform better in symmetric environments due to the ASV's spherization that avoids perceptual aliasing situations.

Future work includes using different fusion, registration, and segmentation methods to improve performance. Minimizing systematic errors to improve georeferenciation and, thereby, improve the accuracy of GEERS is also planned.

References

[1] Ming Lu et al. "Positioning and tracking construction vehicles in highly dense urban areas and building construction sites". In: *Automation in construction* 16.5 (2007), pp. 647–656.

[2] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, 2005.

[3] Guo-Sheng Cai, Huei-Yung Lin, and Shih-Fen Kao. "Mobile robot localization using gps, imu and visual odometry". In: *2019 International Automatic Control Conference (CACS)*. IEEE. 2019, pp. 1–6.

[4] Vijay John et al. “Registration of GPS and stereo vision for point cloud localization in intelligent vehicles using particle swarm optimization”. In: *International Conference on Swarm Intelligence*. Springer. 2017, pp. 209–217.

[5] Vijay John et al. “Stereo vision-based vehicle localization in point cloud maps using multiswarm particle swarm optimization”. In: *Signal, image and video processing* 13.4 (2019), pp. 805–812.

[6] Francisco J Perez-Grau et al. “Multi-sensor three-dimensional Monte Carlo localization for long-term aerial robot navigation”. In: *International Journal of Advanced Robotic Systems* 14.5 (2017), p. 1729881417732757.

[7] Matteo Palieri et al. “Locus: A multi-sensor lidar-centric solution for high-precision odometry and 3d mapping in real-time”. In: *IEEE Robotics and Automation Letters* 6.2 (2020), pp. 421–428.

[8] Andrzej Reinke et al. “Locus 2.0: Robust and computationally efficient lidar odometry for real-time 3d mapping”. In: *IEEE Robotics and Automation Letters* 7.4 (2022), pp. 9043–9050.

[9] Fernando Caballero and Luis Merino. “DLL: Direct LIDAR Localization. A map-based localization approach for aerial robots”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 5491–5498.

[10] Junjie Zhang, Kourosh Khoshelham, and Amir Khodabandeh. “Seamless vehicle positioning by lidar-GNSS integration: standalone and multi-epoch scenarios”. In: *Remote Sensing* 13.22 (2021), p. 4525.

[11] Hugh Durrant-Whyte and Tim Bailey. “Simultaneous localization and mapping: part I”. In: *IEEE robotics & automation magazine* 13.2 (2006), pp. 99–110.

[12] Ji Zhang and Sanjiv Singh. “LOAM: Lidar Odometry and Mapping in Real-time.” In: *Robotics: Science and Systems*. Vol. 2. 9. Berkeley, CA. 2014, pp. 1–9.

[13] Jiachen Zhang et al. “GNSS-RTK Adaptively Integrated with LiDAR/IMU Odometry for Continuously Global Positioning in Urban Canyons”. In: *Applied Sciences* 12.10 (2022), p. 5193.

[14] Chao Ban et al. “Smooth and Accurate LiDAR-GNSS-IMU Localization Method with Confidence Estimation”. In: *2023 42nd Chinese Control Conference (CCC)*. IEEE. 2023, pp. 4213–4219.

[15] Jingbin Liu et al. “A ubiquitous positioning solution of integrating GNSS with LiDAR odometry and 3D map for autonomous driving in urban environments”. In: *Journal of Geodesy* 97.4 (2023), p. 39.

[16] Talha Takleh Omar Takleh et al. “A brief survey on SLAM methods in autonomous vehicle”. In: *International Journal of Engineering & Technology* 7.4 (2018), pp. 38–43.

[17] John Lewis, Pedro U Lima, and Meysam Basiri. “Collaborative 3D Scene Reconstruction in Large Outdoor Environments Using a Fleet of Mobile Ground Robots”. In: *Sensors* 23.1 (2022), p. 375.

[18] Fraj Hariz et al. “Direct Georeferencing 3D Points Cloud Map Based on SLAM and Robot Operating System”. In: *2021 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*. IEEE. 2021, pp. 1–6.

[19] Tomáš Janata and Jiří Cajthaml. “Georeferencing of multi-sheet maps based on least squares with constraints—First military mapping survey maps in the area of Czechia”. In: *Applied Sciences* 11.1 (2020), p. 299.

[20] Paloma Carrasco et al. “Monte-Carlo Localization for Aerial Robots using 3D LiDAR and UWB sensing”. In: *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE. 2021, pp. 354–360.

[21] Veli Ilci and Charles Toth. “High definition 3D map creation using GNSS/IMU/LiDAR sensor integration to support autonomous vehicle navigation”. In: *Sensors* 20.3 (2020), p. 899.

[22] Kai-Wei Chiang et al. “Navigation engine design for automated driving using INS/GNSS/3D LiDAR-SLAM and integrity assessment”. In: *Remote Sensing* 12.10 (2020), p. 1564.

[23] Nader Abdelaziz and Ahmed El-Rabbany. “Deep Learning-Aided Inertial/Visual/LiDAR Integration for GNSS-Challenging Environments”. In: *Sensors* 23.13 (2023), p. 6019.

[24] Saeed B Niku. *Introduction to robotics: analysis, control, applications*. John Wiley & Sons, 2020.

[25] Zhihao Shen et al. “Fast Ground Segmentation for 3D LiDAR Point Cloud Based on Jump-Convolution-Process”. In: *Remote Sensing* 13.16 (2021), p. 3239.

[26] Paul J Besl and Neil D McKay. “Method for registration of 3-D shapes”. In: *Sensor fusion IV: control paradigms and data structures*. Vol. 1611. Spie. 1992, pp. 586–606.

[27] Kazuya Fujita, Kensuke Okada, and Kentaro Katahira. “The Fisher information matrix: A tutorial for calculation for decision making models”. In: (2022).

[28] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. “Robust reconstruction of indoor scenes”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 5556–5565.

[29] Nicholas Carlevaris-Bianco, Arash K. Ushani, and Ryan M. Eustice. “University of Michigan North Campus long-term vision and lidar dataset”. In: *International Journal of Robotics Research* 35.9 (2015), pp. 1023–1035.