Delta filter - robust visual-inertial pose estimation in real-time: A multi-trajectory filter on a spherical mobile mapping system*

Fabian Arzberger[1], Fabian Wiecha[1], Jasper Zevering[1], Julian Rothe[2], Dorit Borrmann[3], Sergio Montenegro[2], and Andreas Nüchter[1]

***Abstract*— Precise, real-time, and onboard 3D localization is mandatory for many robotic systems nowdays. Numerous ways exist to achive this: Many state-of-the-art mobile mapping systems accomplish reliable and robust pose estimation utilizing combinations of inertial measurement units (IMUs), global navigation satellite systems (GNSS), visual-inertial- or LiDAR-inertial odometry (VIO/LIO). However, on a spherical mobile mapping system the underlying inherent rolling motion introduces high angular velocities, thus the quality of pose estimates, images, and laser-scans, degrade. In this work we propose a pose filter design that is able to do real-time sensor fusion between two unreliable trajectories - which are sampled asynchronously - into one, more reliable trajectory. It is a simple yet effective filter design that does not require the user to estimate the uncertanty of the sensors. The approach is not limited to spherical robots and theoretically is also suitable for sensor fusion of an arbitrary number of estimators. Furthermore, the filter does not refine its trajectory with information from future measurements and is able to run at $125\,\mathrm{Hz}$ on a Raspberry Pi 4. This work compares this filter against two pose estimation methods on our spherical system: (1) An approach that is based solely on IMU measurements and a motion model, and (2) stereo-VIO with an Intel® RealSense™ tracking camera. The proposed "Delta" filter takes as input (1), (2), and a motion model. Our implementation gets rid of the drift in (1) and (2), estimates the scale of the trajectory, and deals with slow and fast motion as well as driving curves. To quantify our results, we evaluate the trajectories against ground truth pose measurement using an OptiTrack™ motion capturing system. Furthermore, as our spherical system is equipped with a laser-scanner, we evaluate the resulting point clouds against ground truth maps available from a Riegl VZ-400 terrestrial laser scanner (TLS). Our source code can be found on github[1].**

I. Introduction

Spherical mobile mapping systems are just comming of age, as current research in the robotics community shows: The majority of research dealing with spherical systems is about locomotion mechanisms, e.g. [1]–[6]. Using spherical robots for mobile mapping (cf. Figure 1) is a rather novel field. To the best of our knowledge, Borrmann et al. [7] first used a 2D laserscanner mounted on a unicycle's wheel axis, to generate maps via offline- simultaneous localization and mapping (SLAM). In a follow-up study from our own lab [8] we used the same laserscanner inside a spherical robot with a protective outer plastic shell. The robot is capable of self-initiated motion via flywheels utilizing an IBCOAM (impulse by conservation of angular momentum) approach. The idea of using spherical robots for mapping was explored in more depth by the European Space Agency (ESA) in 2021 during a concurrent design facility (CDF) study. This CDF study consideres the general concept of a spherical robot for environment mapping and exploring lunar caves, but also terrestrial vents, to be feasible [9], [10]. Advantages of using spherical robots are a shell that protects internal sensors and a versatile locomotion mechanism that inherently results in sensor rotation leading to optimal coverage of the environment. During SLAM, large and aggressive rotations are the least favorable motions that a mobile mapping system could experience. This is because for any falsely estimated translation, the errors in the resulting environment grow linearly, wheras for rotation these errors grow exponentially with increasing distance. While working with spherical robots, non-centered rotation is the main movement of the

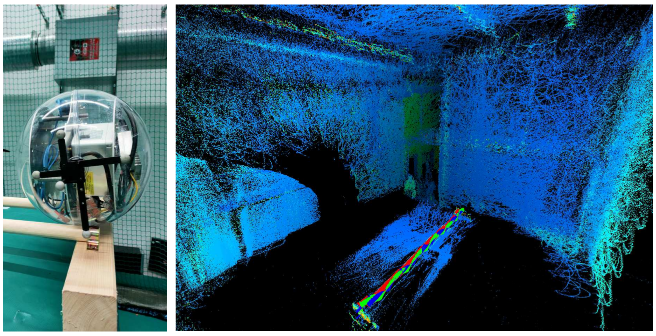


Fig. 1: (Left:) Spherical mobile mapping system equipped with Phidget IMUs, an Intel T265 stereo-tracking camera, Livox Mid-100 LiDAR, and optitrack IR-reflectors. (Right:) Resulting point cloud when applying the estimated trajectory of the proposed Delta-filter, which uses information from the IMUs and internal camera, but not the LiDAR. The optitrack trajectories and resulting point clouds are used for evaluation. A video that demonstrates our implementation is available at https://youtu.be/2yu1RHtTesc.

*We acknowledge funding from the ESA Contract No. 4000130925/20/NL/GLC for the study "DAEDALUS – Descent And Exploration in Deep Autonomy of Lava Underground Structures" within the Open Space Innovation Platform (OSIP) lunar caves-system and the Elite Network Bavaria (ENB) for providing funds for the academic program "Satellite Technology"

[1]The authors are with Computer Science XVII - Robotics, Julius-Maximilians-University Würzburg, 97074 Am Hubland, Germany. Contact: fabian.arzberger@uni-wuerzburg.de

[2]The authors are with Computer Science VIII - Aerospace Information Technology, Julius-Maximilians-University Würzburg, 97074 Am Hubland, Germany. Contact: julian.rothe@uni-wuerzburg.de

[3]Dorit Borrmann is with THWS Robotics, Technische-Hochschule-Würzburg-Schweinfurt, 97421 Schweinfurt, Germany. Contact: dorit.borrmann@thws.de

[1]https://github.com/publish_after_acceptance

internal sensors, which proposes a huge challenge to state of the art SLAM algorithms. In previous work, we proposed initial offline-SLAM solutions for simplified sub-problems, i.e., rotation while descending [11], and rolling on flat surfaces [12]. However, in this work we address only the localization of the system and do not perform offline-SLAM, by introducing a pose estimation filter. Our implementation fuses information from three IMUs and a stereo-tracking camera onboard in real-time. The contributions of this work are as follows:

- A robust yet simple 6-DoF multi-trajectory filter, designed for but not limited to visual-inertial sensor fusion.
- An evaluation of our spherical mobile mapping systems accuracy based not only on ground truth point-clouds, but also on ground truth trajectories, which is stated as an open problem in [12].

The paper is structured as follows: In the next section, we provide an overview of state of the art 6-DoF pose filters, and outline the most similar approaches. Then, we introduce the "Delta"-filter in a general fashion and show an example implementation on a spherical mobile mapping system. Finally, we introduce our accuracy measures and experiments and show that the filter is able to deal with slow and fast motion as well as driving curves.

II. State-of-the-art

Many onboard multi-sensor pose estimation approaches exist in the community. The majority of which being implemented and developed towards autonomous driving cars [13], [14], and unmanned aerial vehicles (UAV) [15], [16]. Soloviev et al. [17] give a broad outline on the sensor types used for navigation: They define a self-contained inertial navigation system (INS) as the primary sensor, as it is available on any platform. Further, the authors consider the following secondary sensors which are qualified for fusion with the INS solution: Global Navigation Satellite System (GNSS) based (e.g. GPS), feature based (e.g. cameras or LiDAR), beacon based (e.g. using specialized navigation signals), or based on signals of opportunity (SoOP) (e.g. radio-frequency signals). In this work we will focus on visual-inertial navigation systems (VINS) and later propose a filter for our spherical system. Santoso et al. [18] categorize popular filters in the robotics community: (1) The Kalman Filter (KF) [19] has been designed to estimate the most likely system state under Gaussian noise by minimizing the covariances of the estimation error. It has since been reinvented and extended serval times, leading to variants such as the Unscented Kalman Filter (UKF) [20], Extended Kalman Filter (EKF) [21], or Multistate Constrained Kalman Filter (MSCKF) [22], just to name a few. KF-based approaches are by far the most popular state estimators among the robotics community. Example implementations on different systems include [16], [23]–[27]. (2) The H_∞ filter approach originates from control theory where it is used as an optimal robust controller. Instead of minimizing the covariance of the estimation error, the H_∞ filter minimizes the worst-case estimation error, which leads to better performance if modelling uncertanties are present [28]. (3) Particle filters (PF), or Monte-Carlo Methods, are known for being applied in many stochastic estimation problems [29]. By now, it is well-known that PF outperforms KF in nonlinear systems underlying non-Gaussian noise [30]. Its biggest drawback is the computational load required for processing many particles representing a single state. (4) Rao-Blackwellized Particle filters (RBPF) combine the advantages of PF and KF while getting rid of their major issues [31]. Therefore, if the system state model contains linear parts with Gaussian noise, these components are seperated and processed using KFs, while nonlinear parts with non-Gaussian noise are dealt with PFs. And finally, in recent years we have noticed the use of (5) graph optimization based methods such as GOMSF [32] and VIRAL-Fusion [33], where the system states are represented and optimized in a pose-graph.

The abovementioned examples solely treat filters implemented on ground vehicles or UAV. Yet other examples exist that implement multi-sensor pose estimation on more challenging systems. Kim et al. [34] fuse data from four sensing modalities on an unmanned underwater vehicle (UUV) using an approach using covariance intersection based on nonlinear optimization. They consider measurements taken via acoustic ultra-short baseline (USBL), Differential GPS (DGPS), Doppler Velocity Logs (DVL), and an INS. Fang et al. [35] use three different sensors for pose estimation on wearable augmented reality (WAR): a monocular camera, a depth sensor, and an INS. They use a KF-based approach in a sliding window fashion. To our knowledge there exists only one onboard pose estimation filter for spherical robots [36]. This approach [36] comes from our own lab and uses only data from inertial measurement units (IMU). The basic idea is to combine the well known IMU orientation filters: the Madgwick filter [37] and Complementary filter [38]. As for translation, the filter performs dead-reckoning using the motion model of a rolling sphere and adding constraints for slipping and sliding effects. Furthermore, the output of the filter in [36] is being utilized as input for the filter proposed in this paper. Lastly, we want to mention another filter that is much simpler than any of the approaches stated above, yet surprisingly effective: Gyrodometry [39]. This filter has been implemented to combine data from wheel encoders (Odometry) with data from a gyroscope by considering not the measured state, but instead the change of state. Therefore, the filter considers the similarity of the measurements to each other to eliminate outliers and update the current state accordingly. The proposed Delta-filter in this paper is similar in these two aspects (change of state and similarity of measurements), but extends the idea to an arbitrary number of estimators in 6-DoF and adds a motion model.

III. Sensor fusion with 6-DoF Delta-filter

In this section we propose a new pose filter design: the "Delta" filter. Its purpose is to receive 6-DoF trajectory estimates from multiple sources, which are known to be

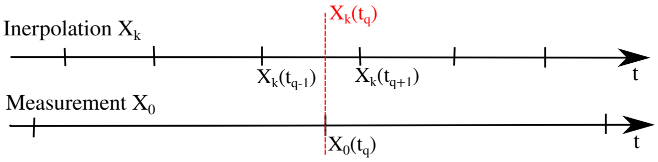


Fig. 2: Timelines showing two sensors publishing pose data at different rates. The sensor having the slower rate is defined as the "measurement", the other trajectories $\boldsymbol{X}_k$ get interpolated at measurement time t_q.

unreliable, and filter them in a probabilistic way. We consider a trajectory "unreliable" if it accumulates drift or makes sudden jumps - which are common effects in IMU- and VIO-based estimators. The filtered trajectory does not use any information from future measurements and is computed in real-time. However, similar to a Kalman filter, the Delta-filter requires a motion model, which is also considered unreliable. In our implementation we filter only two trajectory estimates with a given motion model, yet the Delta-filter is theoretically suitable for an arbitrary number of estimators.

A. *Proposed filter design*

Suppose we have multiple 6-DoF pose estimators $\boldsymbol{X} = [\boldsymbol{R}, \boldsymbol{t}]^\tau \in \mathrm{SE}(3)$, where $\boldsymbol{R}$ is a 3×3 rotation matrix and $\boldsymbol{t}$ is a vector in $\mathbb{R}^3$. The pose of the k-th estimator at time t is denoted by $\boldsymbol{X}_k(t) = [\boldsymbol{R}_k(t), \boldsymbol{t}_k(t)]^\tau : \mathbb{R} \to \mathrm{SE}(3)$. Note that all poses from all estimators must first be transfered in a shared global coordinate frame. As the poses arrive at different time stamps, it is necessary to interpolate between measurements to capture all estimates at the same point in time. Thus, the Delta-filter computes an estimate at the rate of the slowest estimator, denoted as $\boldsymbol{X}_0$, yielding a query time t_q. We call the resulting pose $\boldsymbol{X}_0(t_q)$ the "measurement". All other estimators $\boldsymbol{X}_k$ are queried at time t_q, by interpolating between two measurements at given timestamps $t_{q\pm1}$, as shown in Figure 2. Note that rotation matrices and unit quaternions are isomorphic, thus we use $\boldsymbol{q}_k(t)$ and $\boldsymbol{R}_k(t)$ interchangably as they represent the same elements in $\mathrm{SO}(3)$. Then, the interpolation is constructed using quaternion slerp and linear vector interpolation as described by Equations (1) - (5):

$$\boldsymbol{X}_k(t_q) = [\boldsymbol{R}_k(t_q), \boldsymbol{t}_k(t_q)]^\tau \ , \tag{1}$$

$$\hat{t} = \frac{t_q - t_{q+1}}{t_{q-1} - t_{q+1}} \in [0;1] \ , \tag{2}$$

$$\Omega = \cos^{-1}\left(\boldsymbol{q}_k(t_{q-1}) \cdot \boldsymbol{q}_k(t_{q+1})\right) \ , \tag{3}$$

$$\begin{aligned} \boldsymbol{R}_k(t_q) &= \mathrm{Slerp}\left(\boldsymbol{q}_k(t_{q-1}), \boldsymbol{q}_k(t_{q+1}), \hat{t}\right) \\ &= \frac{\sin((1-\hat{t})\Omega)}{\sin(\Omega)} \cdot \boldsymbol{q}_k(t_{q-1}) + \frac{\sin(\hat{t}\Omega)}{\sin(\Omega)} \cdot \boldsymbol{q}_k(t_{q+1}), \end{aligned} \tag{4}$$

$$\boldsymbol{t}_k(t_q) = (1-\hat{t}) \cdot \boldsymbol{t}_k(t_{q-1}) + \hat{t} \cdot \boldsymbol{t}_k(t_{q+1}) \tag{5}$$

The idea of the Delta-filter is to track the changes between given timestamps t_1 and t_2 (also known as "deltas") of the measurements and interpolations

$$\Delta\boldsymbol{X} = \left[\boldsymbol{R}^{-1}(t_2) \cdot \boldsymbol{R}(t_1) \ , \ \boldsymbol{t}(t_2) - \boldsymbol{t}(t_1)\right]^\tau \tag{6}$$

and estimate a new delta that makes more sense. That is to say that the Delta-filter estimates the most likely pose change between given timestamps. Therefore, the filter first estimates a model delta

$$\begin{aligned} \Delta\boldsymbol{X}_m &= [\Delta\boldsymbol{R}_m, \Delta\boldsymbol{t}_m]^\tau \\ &= f\left(\Delta\boldsymbol{X}_0 \ , \ \{\Delta\boldsymbol{X}_k : k \in \mathbb{N}\}\right) \end{aligned} \tag{7}$$

where f denotes the motion model that estimates the true motion given the measured and interpolated deltas, $\Delta\boldsymbol{X}_0$ and $\Delta\boldsymbol{X}_k$. In a later section we will give an example for the motion model f when implementing the filter on a spherical robot.

1) Measurement, interpolation, and model: The measurement, interpolation, and model deltas $\Delta\boldsymbol{X}_0$, $\Delta\boldsymbol{X}_k$, and $\Delta\boldsymbol{X}_m$ respectivley, are all considered unreliable. They are used to estimate the filtered pose $\boldsymbol{X}_e(t_j)$ by iterativley applying an estimated filtered delta $\Delta\boldsymbol{X}_e$ that happened between t_{j-1} and t_j:

$$\begin{aligned} \boldsymbol{X}_e(t_j) &= \Delta\boldsymbol{X}_e \cdot \boldsymbol{X}_e(t_{j-1}) \\ &= [\Delta\boldsymbol{R}_e \cdot \boldsymbol{R}_e(t_{j-1}), \Delta\boldsymbol{t}_e + \boldsymbol{t}_e(t_{j-1})]^\tau \ . \end{aligned} \tag{8}$$

We seperate the rotation and translation parts by assuming that the measured and interpolated orientations are sufficiently reliable estimates, i.e., they dont drift or jump during a short time period. This assumption is valid for most inertial- and visual-tracking systems. To obtain the estimated filtered rotation delta $\Delta\boldsymbol{R}_e$, we compute

$$\Delta\boldsymbol{R}_e = \mathrm{Slerp}\left(\Delta\boldsymbol{q}_0, \Delta\boldsymbol{q}_k, \frac{1}{2}\right) \tag{9}$$

Note that for more than two estimators, the Slerp in Equation (9) must be replaced with a different quaternion average, e.g. [40]. Furthermore, we assume that the estimated translation deltas $\Delta\boldsymbol{t}_0$, $\Delta\boldsymbol{t}_k$, and $\Delta\boldsymbol{t}_m$ are not sufficiently reliable to just average them, as inertial-tracking tends to drift and visual-tracking tends to jump.

2) Probabilistic weighted geometric mean: Therefore, we use a probabilistic approach that averages the translation direction and then scales it.

$$\Delta\boldsymbol{t}_e = \frac{d}{|\sum_i \Delta\boldsymbol{t}_i|} \cdot \sum_i \Delta\boldsymbol{t}_i \tag{10}$$

where $\Delta\boldsymbol{t}_i$ refers to the measurement, interpolation, and model deltas. An estimate of the true scale of the translated distance d is given by a probabilistically weighted geometric mean:

$$d = \left(\prod_{\omega_i} |\Delta\boldsymbol{t}_i|^{\omega_i}\right)^{\left(\sum_i \omega_i\right)^{-1}} \tag{11}$$

We calculate weights ω_i for each delta that correspond to the similarity of the deltas to their geometric mean, thus outliers

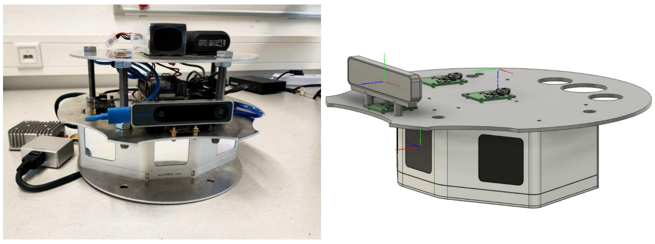


Fig. 3: (Left:) Spherical mobile mapping system without its protecting shell. (Right:) CAD model of the spherical system with sensor frames.

get a damped weighting while similar values get a higher weighting:

$$|\hat{\boldsymbol{t}}| = \left(\prod_{i=1}^{n} |\Delta \boldsymbol{t}_i|\right)^{n^{-1}} , \tag{12}$$

$$s = \sqrt{\frac{1}{n-1}\sum_{i=0}^{n}\left(|\Delta \boldsymbol{t}_i| - |\hat{\boldsymbol{t}}|\right)^2} , \tag{13}$$

$$w_i = 1 - s^{-1} \cdot \left(|\Delta \boldsymbol{t}_i| - |\hat{\boldsymbol{t}}|\right) \tag{14}$$

B. Implementation on a spherical system

The implementation on our spherical robot uses two estimators, the IMU operating at 125 Hz defines the measurement $\boldsymbol{X}_0$, and the camera operating at 200 Hz defines the interpolation $\boldsymbol{X}_k$. For the motion model f of the spherical system with known radius $r = 0.145$ m, we assume that rotation leads to translation, thus we calculate the estimated model delta using the arc length of rotation:

$$f(\Delta \boldsymbol{X}_0, \Delta \boldsymbol{X}_k) = \left[\Delta \boldsymbol{R}_e,\ r \cdot \angle\left(\Delta \boldsymbol{R}_e\right) \cdot \frac{\Delta \boldsymbol{t}_0 + \Delta \boldsymbol{t}_k}{|\Delta \boldsymbol{t}_0 + \Delta \boldsymbol{t}_k|}\right]^{\tau} , \tag{15}$$

where $\angle(\cdot)$ denotes the angle around the axis described by the rotation matrix. Note that we just defined the model rotation $\Delta \boldsymbol{R}_m$ from Equation (8) to be equal to $\Delta \boldsymbol{R}_e$ from Equation (9), as the orientation estimation is considered sufficiently reliable.

The simplicity of the filter design allows for the introduction of simple but effective design choices. As an example we notice that our IMUs tends to drift without the use of a magnetometer, especially in the yaw-axis, wheras the tracking camera does not. Due to the background of our spherical system, we do not want to use the magnetometers by design. Thus, we must rely more on the camera estimations for the yaw angle, which is why we exchange the estimation of the rotation delta in Equation (9). Instead of only using Slerp, which is more universal, we first use Slerp and then replace the yaw-part of the resulting delta with the interpolated camera yaw delta. Hence, the change in yaw is only estimated via the camera.

IV. Experiments and Evaluation

Qualitative results are presented in Figure 4. The IMU-based approach (a) suffers from drift in the yaw axis and overestimates the scale of the trajectory. The visual-inertial (b) tracking approach tends to jump whenever the camera looses track, which happens quite often given the unfavourable type of sensor motion. Our proposed Delta-filter (c) combines both trajectories in real-time at $125\,\mathrm{Hz}$ on a Raspberry Pi 4, gets rid of the drift and jumps, and estimates the scale of the trajectory better. The following sections quantify the results using ground truth trajectories and maps.

A. Error metrics

To quantify the quality of pose estimation, we use two principal approaches: On the one hand, we measure ground truth trajectories with an Optitrack system using IR reflectors. On the other hand, we also compare the resulting point clouds against ground truth measurements in larger environments, when Optitrack is no longer available. We denote the ground truth trajectory $\boldsymbol{X}_{\mathrm{ref}} = [\boldsymbol{R}_{\mathrm{ref}}, \boldsymbol{t}_{\mathrm{ref}}]^{\tau}$, and the other estimated trajectories $\boldsymbol{X}_{\mathrm{est}} = [\boldsymbol{R}_{\mathrm{est}}, \boldsymbol{t}_{\mathrm{est}}]^{\tau}$. For each timestamp in the ground truth trajectory, we sample the closest pose in time from the estimated trajectory for correspondence. Note that all the trajectories must be aligned with the ground truth trajectory. Therefore we align the origins of the trajectories first, as we know that all trajectories started from the same point. Afterwards we rotate around the shared origin using a least-squares alignment according to Umeyama [42]. Note that we only use the estimated rotation of the Umeyama method, since we already aligned the origins. From this point, we use Grupps [43] software for trajectory evaluation. The resulting point clouds are aligned to ground truth using the well-known Iterative Closest Points (ICP) algorithm. We use 3DTK [44] for the processing of the point clouds.

1) Absolute position error: The absolute position error (APE) represents the error of the translation estimation and is given by

$$\mathrm{APE}_i = |\boldsymbol{t}_{\mathrm{est},i} - \boldsymbol{t}_{\mathrm{ref},i}|\ [\mathrm{m}] . \tag{16}$$

2) Relative pose error: The relative pose error (RPE) represents the error of the orientation estimation and is given by

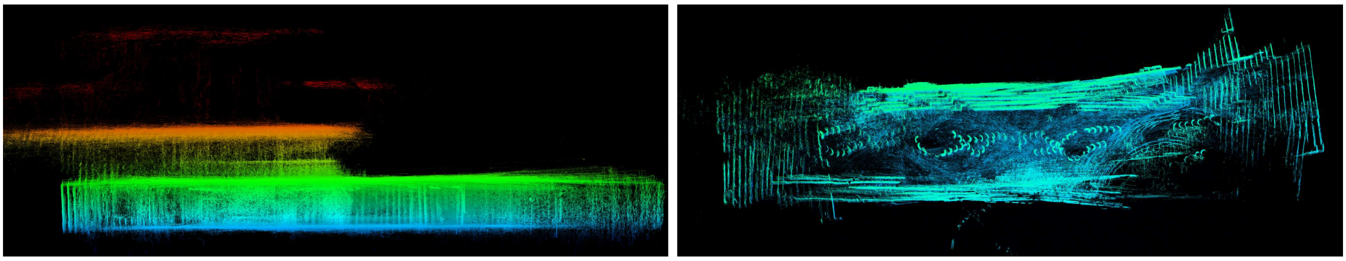
$$\mathrm{RPE}_i = \left|\angle\left(\left(\boldsymbol{R}_{\mathrm{ref},i}^{-1}\boldsymbol{R}_{\mathrm{ref},i-1}\right)^{-1}\left(\boldsymbol{R}_{\mathrm{est},i}^{-1}\boldsymbol{R}_{\mathrm{est},i-1}\right)\right)\right|\ [\mathrm{deg}] . \tag{17}$$

3) Point cloud error: The point cloud error represents the root of the mean squared point-to-point errors (RMSE). Suppose, after matching with ICP, there are N corresponding model- and data-points in the same coordinate frame, denoted $\boldsymbol{m}_i, \boldsymbol{d}_i \in \mathbb{R}^3$ respectivley. Then, the root mean squared error is given by

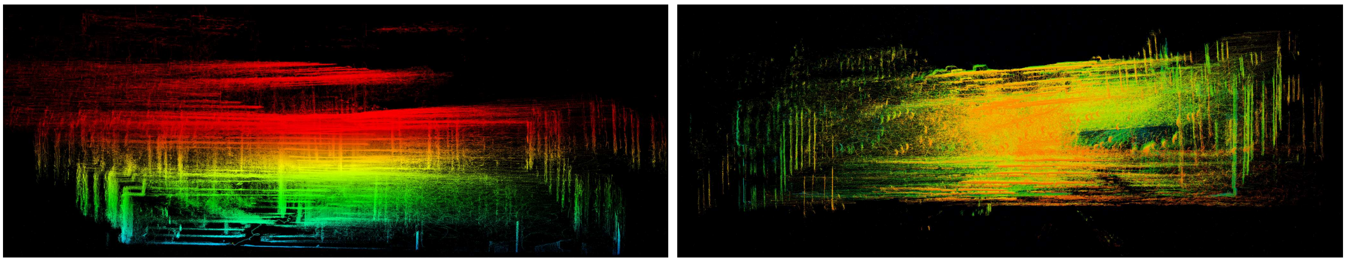
$$\mathrm{RMSE} = \frac{1}{N}\sum_{i=0}^{N}|\boldsymbol{m}_i - \boldsymbol{d}_i|^2 \tag{18}$$

B. Experiments

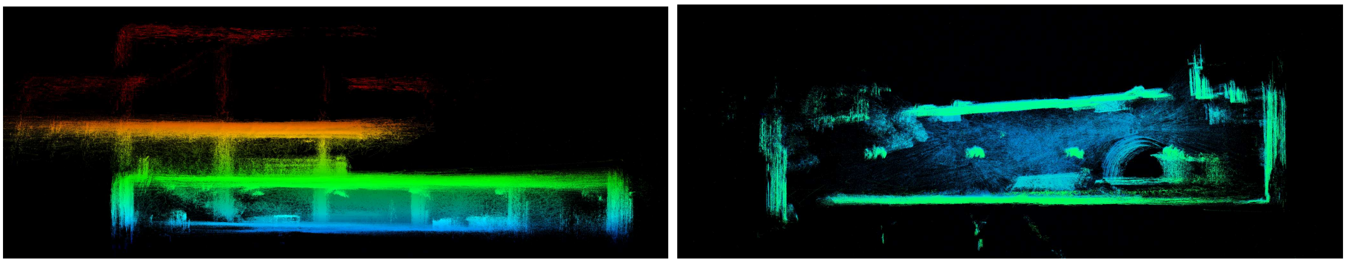
The experiments consist of three types of motion: rolling a straight line slowly, fast, and driving curves at moderate



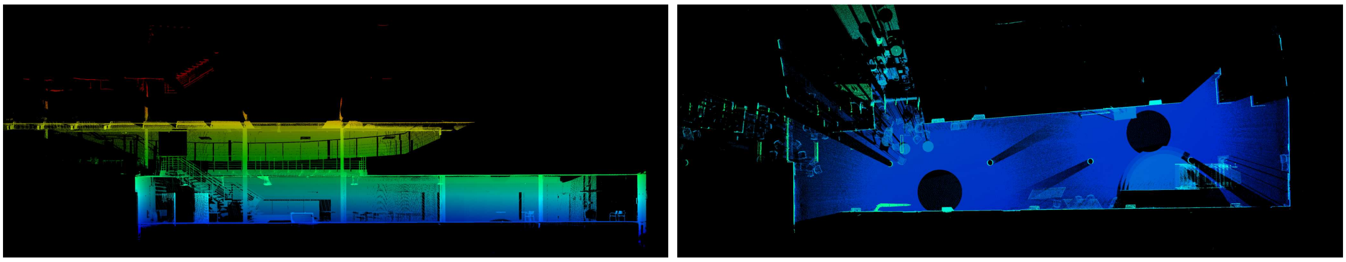
(a) Dead-reckoning IMU-only based pose estimation due to Zevering et al. [41]



(b) Visual-inertial tracking using an Intel RealSense T265 stereo-camera



(c) The proposed Delta-filter using (a), (b), and a motion model



(d) Ground truth point cloud available from a RIEGL VZ-400 TLS

Fig. 4: Resulting point clouds when using three different estimators (a), (b), and (c) are orthographically vizualized. A ground truth point cloud is shown in (d). Images in one column were shot from the same point of view. The colors in the point clouds denotes height. The left column shows sliced views from the side, wheras the right column shows sliced views from the birdseye perspective.

speed. In the first two experiments, an OptiTrack system is available to capture ground truth trajectories, such that we are able to use Equations (17) and (16). However, in the last experiment (driving curves), the environment and trajectory is larger, making the OptiTrack system unavailable. In this experiment, we use a Riegl VZ-400 terrestrial laser scanner (TLS) with an angular resolution of $0.04°$ and accuracy of $5\,\mathrm{mm}$ to provide accurate ground truth point clouds. As our system is equipped with a laser scanner (cf. Figure 3), we compare the resulting point cloud to the ground truth map using Equation (18). Both setups are shown in Figure 5.

1) Fast motion: In this experiment, the sphere traversed a distance of approx. $4\,\mathrm{m}$ in about $10\,\mathrm{s}$. Figure 6 shows the APE (16) of all estimators over time. The T265 suffers from the highest error due to tracking loss, which forces it to rely solely on error prone double integration of acceleration measurements. The IMU-based approach show a considerable increase of error due to the accumulated drift. The error of the proposed Delta-filter are orders of magnitude smaller compared to the IMUs and T265. Figure 7 shows the comparison of RPE (17) over time. Note that the Savgol-filter [45] is applied to the error signals. This is because the ground truth orientations from the OptiTrack system contain many outliers due to mirroring of the IR-reflectors on the spherical shell. The Savgol-filter removes the effect of these outliers but preserves the signal tendency. The RPE of all

TABLE I: Comparison of the estimated translation of the trajectory produced by the Delta-filter with its two source estimators, based on several statistical metrics. Each column compares three values where lower is better.

Error metrics to ground truth trajectories for fast and slow motion with respect to translation								
Estimator	RMSE [m]		Mean [m]		Std. [m]		Max. [m]	
	Slow	Fast	Slow	Fast	Slow	Fast	Slow	Fast
Dead-reckoning INS	1.713	1.736	1.447	1.291	0.917	1.160	2.882	3.001
Intel T265 Stereo-VIO	4.486	7.441	4.012	5.290	2.008	5.234	5.848	13.549
Proposed Delta-filter	**0.114**	**0.248**	**0.103**	**0.193**	**0.049**	**0.165**	**0.189**	**0.428**

TABLE II: Comparison of the estimated rotation of the trajectory produced by the Delta-filter with its two source estimators, based on several statistical metrics. Each column compares three values where lower is better.

Error metrics to ground truth trajectories for fast and slow motion with respect to rotation								
Estimator	RMSE [deg]		Mean [deg]		Std. [deg]		Max. [deg]	
	Slow	Fast	Slow	Fast	Slow	Fast	Slow	Fast
Dead-reckoning INS	1.389	4.318	1.281	3.270	**0.537**	2.819	**2.653**	8.883
Intel T265 Stereo-VIO	**1.374**	**4.213**	**1.264**	**3.190**	0.541	**2.753**	2.701	**8.852**
Proposed Delta-filter	1.384	4.305	1.273	3.248	0.543	2.825	2.752	9.199

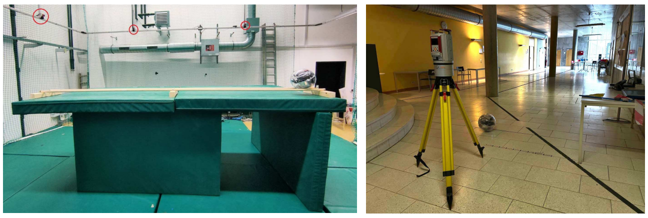


Fig. 5: (Left:) Laboratory test setup in a flycage equipped with an Optitrack system. The sphere has IR reflectors attached to its shell, which are detected by the cameras (red circles). (Right:) Laboratory test setup in the Computer Science building. A RIEGL VZ-400 TLS captures a precise ground truth point cloud for comparison with the spherical mobile mapping system. In both images, motion of the sphere is initiated manually by hand.

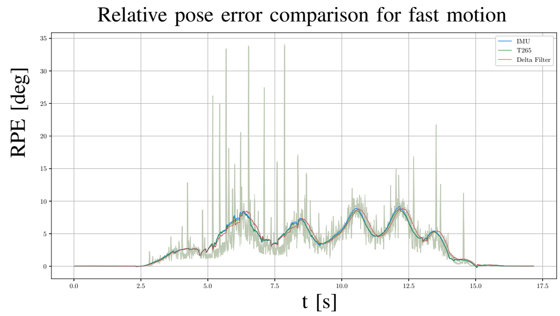


Fig. 7: The relative pose error of all estimators during fast motion over time. The Savgol-filter is applied with a window size of 51 and a polynomial degree of 3 to remove the effect of outliers. In the background the noisy pre-filtered data is shown with low opacity.

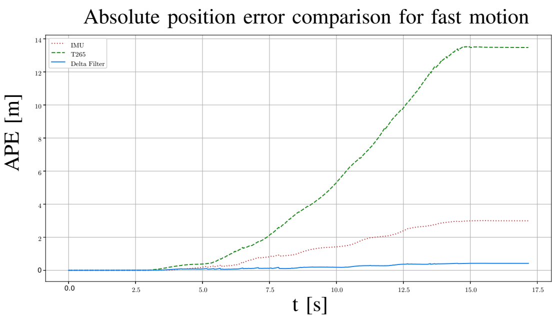


Fig. 6: The absolute position error of all estimators during fast motion over time.

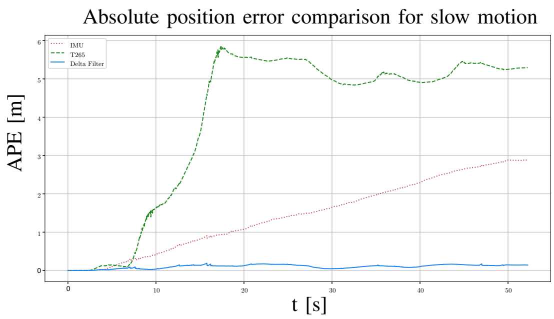


Fig. 8: The absolute position error of all estimators during slow motion over time.

estimators do not differ particularly from each other, which is also evident from the error metrics in Table II. In fact, the RMSE of the RPE of the Delta-filter is between the INS- and T265-solution, which makes sense considering the interpolation in Equation (9).

2) Slow motion: In this experiment, the sphere traversed a distance of approx. $4\,\mathrm{m}$ in about $45\,\mathrm{s}$. Figure 8 shows the comparison of APE over time. The Delta-filter compensates for the linear accumulation of error of the IMU and the sudden jump of the T265, resulting in a lower overall translation error. Table I confirms this observation. Figure 9 presents the comparison of RPE over time. As mentioned above, the Savgol-filter is applied on the error signals. The orientation errors of all estimators are similar to each other, yet overall smaller compared to fast motion.

3) Curves: Figure 10 shows the result of the point cloud analysis. The error to ground truth is vizualized in a point-to-point distance distribution histogram. Note that the large

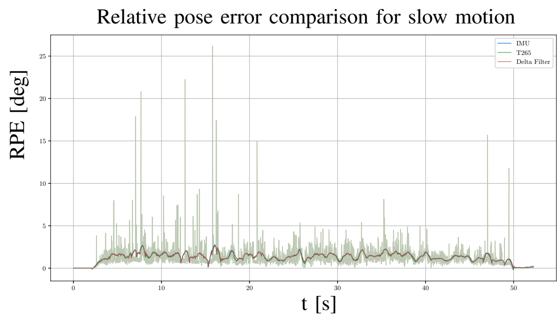


Fig. 9: The relative pose error of all estimators during slow motion over time. The Savgol-filter is applied with a window size of 51 and a polynomial degree of 3 to remove the effect of outliers. In the background the noisy pre-filtered errors are shown with low opacity.

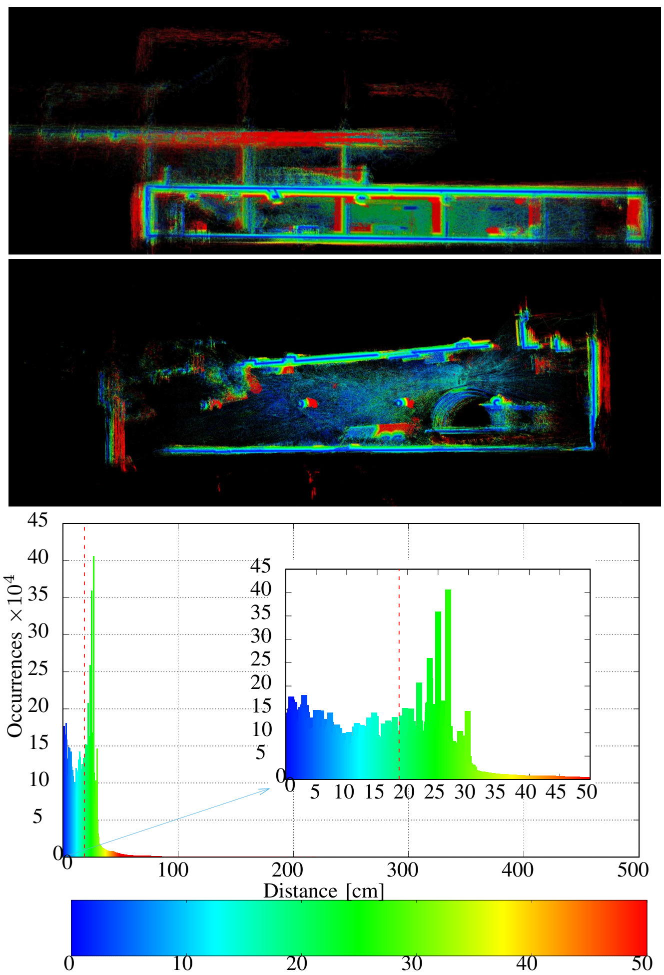


Fig. 10: Resulting point cloud (sliced side-view and birds-eye view) using the trajectory of the proposed filter, as well as a histogram showing a distribution of point-to-point distances. These distances to ground truth are also vizualized using color. The red dashed line in the histogram indicates the mean point-to-point error, which is $18.6\,\mathrm{cm}$

.

errors at the pillars are caused by global filter drift. On the other hand, the errors at the ceiling of the upper floor are rather caused by missing points in the ground truth. These points, however, are so few that they are only barely visible in the histogram. The mean point-to-point error from Equation (18), which is our accuracy estimate for mapping, is $18.6\,\mathrm{cm}$.

C. Discussion

The evaluation shows that the Delta-filter significantly improves the pose estimation accuracy, reduces drift, and eliminates jumps. However, despite reducing the drift, all experiments show that the filter still suffers from global drift regarding translation. Furthermore, in the resulting point clouds, the walls appear to be thicker than in the ground truth point cloud, which comes down to two factors: First, the Livox Mid-100 used in the experiment has higher measurement noise, especially when the laser goes through the plastic shell. And second, the extrinsic calibration of the sensors in the spherical system is rather poor, as all the sensors assume to sit inside the center of the sphere.

V. Conclusions

In this paper we addressed the problem of precise, real-time, and onboard localization in 6-DoF for spherical mobile mapping systems. Usually on these systems, the large angular velocities and constant aggressive dynamics when rolling makes state-of-the-art approaches, e.g. INS- or VIO-based solutions, more difficult. We therefore proposed the simple yet effective Delta-filter, which is able to do real-time sensor fusion of an INS- with a VIO-based solution. The filter needs a motion model defined by the user, greatly decreases the INS drift, and gets rid of the jumps caused by the VIO. We showed that the filter is reliable in slow and fast motion, as well as driving curves. Furthermore, we estimated the mapping accuracy of the spherical mobile mapping system to be $18.6\,\mathrm{cm}$ without the use of offline-SLAM, which is considered to be an improvement to our previous work. Having such a trajectory estimate brings real-time, highly precise laser-based SLAM for spherical robots closer to reality in the near future. However, needlessly to say, a lot of work remains to be done. In the future, we need to address a proper extrinsic calibration between all sensors to further increase the accuracy. We will also incorporate the LiDAR measurements into the localization by building a real-time onboard laser-based SLAM algorithm designed for spherical systems. This will also include the extension of the motion model using environment data to account for slopes, uneven terrain, or free falling for a short period of time.

References

[1] R. Armour, K. Paskins, A. Bowyer, J. Vincent, and W. Megill, “Jumping robots: a biomimetic solution to locomotion across rough terrain,” *Bioinspiration & biomimetics*, vol. 2, no. 3, p. S65, 2007.

[2] K. W. Wait, P. J. Jackson, and L. S. Smoot, “Self locomotion of a spherical rolling robot using a novel deformable pneumatic method,” in *2010 IEEE International Conference on Robotics and Automation*, pp. 3757–3762, IEEE, 2010.

[3] R. Mukherjee, “Spherical mobile robot,” 2001. US Patent 6,289,263.

[4] R. Chase and A. Pandya, "A review of active mechanical driving principles of spherical robots," *Robotics*, vol. 1, no. 1, pp. 3–23, 2012.

[5] D. Liu, H. Sun, Q. Jia, and L. Wang, "Motion control of a spherical mobile robot by feedback linearization," in *2008 7th World Congress on Intelligent Control and Automation*, pp. 965–970, 2008.

[6] J. Zevering, K. Braun, M. Hesse, K. Mathewos, D. Borrmann, A. , Bredenbeck, and A. Nuechter, "The concept the virtual pose instruction plane (vpip) for controlling rod-driven spherical robots," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023. submitted for review.

[7] D. Borrmann, S. Jörissen, and A. Nüchter, "RADLER – A RADial LasER scanning device," in *Proceedings of the International Symposium on Experimental Research*, (Buenos Aires, Argentina), pp. 655–664, 01 2020.

[8] J. Zevering, A. Bredenbeck, F. Arzberger, D. Borrmann, and A. Nüchter, "Luna-a laser-mapping unidirectional navigation actuator," in *Experimental Robotics: The 17th International Symposium*, pp. 85–94, Springer, 2021.

[9] A. P. Rossi, F. Maurelli, V. Unnithan, H. Dreger, K. Mathewos, N. Pradhan, D.-A. Corbeanu, R. Pozzobon, M. Massironi, S. Ferrari, *et al.*, "Daedalus-descent and exploration in deep autonomy of lava underground structures," 2021.

[10] J. Zevering, D. Borrmann, A. Bredenbeck, and A. Nüchter, "The concept of rod-driven locomotion for spherical lunar exploration robots," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5656–5663, 2022.

[11] F. Arzberger, A. Bredenbeck, J. Zevering, D. Borrmann, and A. Nüchter, "Towards spherical robots for mobile mapping in human made environments," *ISPRS Open Journal of Photogrammetry and Remote Sensing*, vol. 1, p. 100004, 2021.

[12] F. Arzberger, J. Zevering, A. Bredenbeck, D. Borrmann, and A. Nüchter, "Mobile 3d scanning and mapping for freely rotating and vertically descended lidar," in *2022 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pp. 122–129, 2022.

[13] Xue, Jian-ru and Wang, Di and Du, Shao-yi and Cui, Di-xiao and Huang, Yong and Zheng, Nan-ning, "A vision-centered multi-sensor fusing approach to self-localization and obstacle perception for robotic cars," *Frontiers of Information Technology & Electronic Engineering*, vol. 18, no. 1, pp. 122–138, 2017.

[14] C. Merfels and C. Stachniss, "Sensor fusion for self-localisation of automated vehicles," *PFG–Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, vol. 85, pp. 113–126, 2017.

[15] G. Abdi, F. Samadzadegan, and F. Kurz, "Pose estimation of unmanned aerial vehicles based on a vision-aided multi-sensor fusion," in *XXII ISPRS Congress, Technical Commission I*, vol. 41, pp. 193–199, 2016.

[16] Du, Hao and Wang, Wei and Xu, Chaowen and Xiao, Ran and Sun, Changyin, "Real-time onboard 3d state estimation of an unmanned aerial vehicle in multi-environments using multi-sensor data fusion," *Sensors*, vol. 20, no. 3, 2020.

[17] A. Soloviev and M. M. Miller, *Navigation in Difficult Environments: Multi-Sensor Fusion Techniques*, pp. 199–229. New York, NY: Springer New York, 2012.

[18] F. Santoso, M. A. Garratt, and S. G. Anavatti, "Visual–inertial navigation systems for aerial robotics: Sensor fusion and technology," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 1, pp. 260–275, 2017.

[19] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[20] E. Wan and R. Van Der Merwe, "The unscented kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pp. 153–158, 2000.

[21] F. Daum, "Nonlinear filters: beyond the kalman filter," *IEEE Aerospace and Electronic Systems Magazine*, vol. 20, no. 8, pp. 57–69, 2005.

[22] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 3565–3572, 2007.

[23] A. Sakai, Y. Tamura, and Y. Kuroda, "An efficient solution to 6dof localization using unscented kalman filter for planetary rovers," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4154–4159, 2009.

[24] G. Ligorio and A. M. Sabatini, "Extended kalman filter-based methods for pose estimation using visual, inertial and magnetic sensors: Comparative analysis and performance evaluation," *Sensors*, vol. 13, no. 2, pp. 1919–1941, 2013.

[25] J. Kelly and G. S. Sukhatme, "Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration," *The International Journal of Robotics Research*, vol. 30, no. 1, pp. 56–79, 2011.

[26] G. Huang, K. Eckenhoff, and J. Leonard, *Optimal-State-Constraint EKF for Visual-Inertial Navigation*, pp. 125–139. Cham: Springer International Publishing, 2018.

[27] Liao, Jianchi and Li, Xingxing and Wang, Xuanbin and Li, Shengyu and Wang, Huidan, "Enhancing navigation performance through visual-inertial odometry in gnss-degraded environment," *Gps Solutions*, vol. 25, pp. 1–18, 2021.

[28] N. Abdelkrim, N. Aouf, A. Tsourdos, and B. White, "Robust nonlinear filtering for ins/gps uav localization," in *2008 16th Mediterranean Conference on Control and Automation*, pp. 695–702, 2008.

[29] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[30] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425–437, 2002.

[31] A. J. Haug, *Bayesian estimation and tracking: a practical guide*. John Wiley & Sons, 2012.

[32] R. Mascaro, L. Teixeira, T. Hinzmann, R. Siegwart, and M. Chli, "Gomsf: Graph-optimization based multi-sensor fusion for robust uav pose estimation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1421–1428, 2018.

[33] Nguyen, Thien-Minh and Cao, Muqing and Yuan, Shenghai and Lyu, Yang and Nguyen, Thien Hoang and Xie, Lihua, "Viral-fusion: A visual-inertial-ranging-lidar sensor fusion approach," *IEEE Transactions on Robotics*, vol. 38, no. 2, pp. 958–977, 2022.

[34] Kim, Kihun and Choi, Hyun-Taek and Lee, Chong-Moo, "Underwater precise navigation using multiple sensor fusion," in *2013 IEEE International Underwater Technology Symposium (UT)*, pp. 1–4, 2013.

[35] Wei Fang and Lianyu Zheng and Xiangyong Wu, "Multi-sensor based real-time 6-dof pose tracking for wearable augmented reality," *Computers in Industry*, vol. 92-93, pp. 91–103, 2017.

[36] J. Zevering, A. Bredenbeck, F. Arzberger, D. Borrmann, and A. Nüchter, "Imu-based pose-estimation for spherical robots with limited resources," in *2021 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pp. 1–8, IEEE, 2021.

[37] S. Madgwick *et al.*, "An efficient orientation filter for inertial and inertial/magnetic sensor arrays," *Report x-io and University of Bristol (UK)*, vol. 25, pp. 113–118, 2010.

[38] Min, Hyung Gi and Jeung, Eun Tae, "Complementary filter design for angle estimation using mems accelerometer and gyroscope," *Department of Control and Instrumentation, Changwon National University, Changwon, Korea*, pp. 641–773, 2015.

[39] J. Borenstein and L. Feng, "Gyrodometry: a new method for combining data from gyros and odometry in mobile robots," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 1, pp. 423–428 vol.1, 1996.

[40] F. L. Markley, Y. Cheng, J. L. Crassidis, and Y. Oshman, "Averaging quaternions," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 4, pp. 1193–1197, 2007.

[41] J. Zevering, A. Bredenbeck, F. Arzberger, D. Borrmann, and A. Nuechter, "Imu-based pose-estimation for spherical robots with limited resources," in *2021 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pp. 1–8, 2021.

[42] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 4, pp. 376–380, 1991.

[43] M. Grupp, "evo: Python package for the evaluation of odometry and SLAM.." `https://github.com/MichaelGrupp/evo`, 2017.

[44] A. Nüchter and K. Lingemann, "3DTK—The 3D Toolkit. 2011." `https://slam6d.sourceforge.io/index.html`, 2011.

[45] A. Savitzky and M. J. Golay, "Smoothing and differentiation of data by simplified least squares procedures.," *Analytical chemistry*, vol. 36, no. 8, pp. 1627–1639, 1964.