# Inertial LiDAR Motion Distortion Correction on Spherical Mobile Mapping Systems for Planetary Exploration

Paul Heinisch[1], Fabian Arzberger[1], and Andreas Nüchter[1]

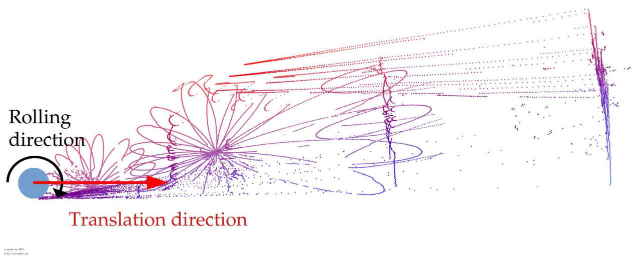
***Abstract*— In this paper, we present an algorithm countering the motion distortion observed in LiDAR point clouds due to the inherent inertial rotations when recorded with a spherical mobile mapping system. State-of-the-art motion distortion correction algorithms focus on more rotationally restricted systems, such as UAV, rovers, or cars, where often a "dead-reckoning" approach is utilized to account for the vehicles linear velocity. Thus, the algorithm presented in this paper is based on a motion model for spherical systems, taking only the angular velocities into account. We show, despite explicitly omitting a model for the linear velocity of the scanner, and assuming constant angular velocity between two frames, that our approach outperforms state-of-the-art algorithms which use a false model for the linear velocities. To verify this, we test our algorithm on 3D point clouds recorded with a Livox MID-100 mounted on a spherical mobile mapping system, and compare the accuracy to an algorithm proposed in the Livox-SDK. To do this, we have ground truth point clouds available from a terrestrial laser scanner (TLS). Our algorithm consistently reduces the motion distortion from low (5 Hz) to high (50 Hz) LiDAR sampling frequencies, such that the point clouds are well-suited for further processing steps such as odometry or mapping. The code for our algorithm is open source [1].**

## I. Introduction

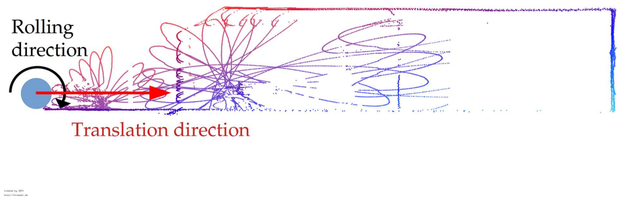
Exploration of extraterrestrial environments has gained traction in the last century: The first humans explored space and the moon in the late 1960s. The only other body in space humankind has visited up until now. Robotics in space has become a necessity, as humans are not fit to explore the vast expanses of our universe, themselves. Today, there are several missions aimed at exploring the surfaces of our moon [2], [3], other planets and their moons [4]–[6], and asteroids [7]. Robots need to perceive their environment with sensors and perform autonomous tasks with the gathered information. This requires sophisticated techniques to make the data as reliable as possible. For many applications in space and also here on Earth, it is mandatory for a robot to locate itself in the environment without the help of global positioning systems such as a navigation satellite system (GNSS). Thus, a robot needs to be able to compute its trajectory on-board in real-time only by using the data it has available. Building reliable representations of the environment makes it finally possible for the robot to navigate unknown terrain [8]–[10].

*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-Universität Würzburg, 97074 Am Hubland, Germany. Contact: `paul.heinisch@stud-mail.uni-wuerzburg.de`



(a) Distorted 3D point cloud due to fast inherent rolling motion.



(b) Distortion corrected 3D point cloud.

Fig. 1: Illustration of AMDC working principle on a spherical robot. Only one principal axis is vizualized, which is a sliced view from the side.

Spherical robots are a recently developed type of robot which is still niche [11]–[14]. In recent years, the European Space Agency (ESA) has evaluated spherical robots to be feasable to be used for the exploration of lunar caves [2]. The spherical shape has many advantages, e.g., protection of the inner system, power efficiency steerability, or sensor coverage by locomotion. In recent work, there have been advances in the locomotion of spherical robots including jumping [12] and pushing itself using rods [15]. However, also many challenges regarding sensor data processing need to be overcome. In this work particularly, we focus on the motion distortion due to the higher than usual angular velocities compared to other types of systems, such as UAV, rovers, or cars. We show that a spherical robot must not use the same correction algorithms than many other systems experiencing high linear velocities [16]. To compensate for the motion distortion induced into the LiDAR point cloud, we rely solely on the angular velocity of the system and ignore its linear velocity. The intuition behind that is, that the motion profile of the LiDAR inside a ball is mostly governed by rotation, leading to much higher distortion as illustrated in Figure 2.

LiDAR systems have been a common tool for perceiving

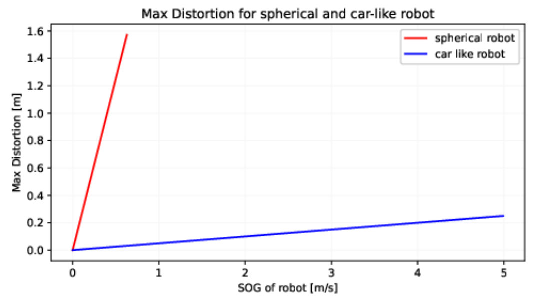


Fig. 2: Motion Distortion due to movement during the recording of a point cloud. The spherical robot (red) with a radius of 20 $\mathrm{cm}$, using rotation as its means of locomotion, has a steep curve for the maximum distortion for objects situated at a distance of ten meters compared to its speed over ground. The distortion of the point clouds for car-like robots (blue) increases much more slowly.

and mapping the environment [17]–[19]. In many applications, especially in space, wheeled robots move at low velocities, or take measurements in a stop-and-go fashion, making the motion distortion of LiDAR point cloud recordings negligable. However, in mobile robots using other means of locomotion this may not hold.

Motion distortion of LiDAR point clouds is a phenomenon happening, once the LiDAR sensor is in motion. Usually, LiDAR systems record a point cloud using laser beams and scan a certain field of view by redirecting the beam and measuring the time of flight or phase shift [20]. This is often done with moving parts in the sensor (for example mirrors or prisms [21], [22]). However, recording one scan while moving, containing multiple points, results in a time interval in which each point has an increasing timestamp. If the sensor is not at a standstill, the points in the final point cloud will appear in false locations, because the sensor itself, which records each point its inertial frame, has moved with respect to the start of the measurement (Figure 1a). Finding the time-dependent transformation between the inertial frame and the global frame is then necessary to transform the points to their true location, representing the actual environment.

We assume a dominating rolling motion for a spherical robot, with the LiDAR sensor mounted rigidly inside the robot, which results in the points being mostly angularly displaced. Artifacts because of angular displacement of the points already appear at very slow velocities. Compared to vehicles dominated by translations (linear velocity), the distortions are more severe, even at slow velocities (Figure 2). In this work, we developed an algorithm countering the distortion by computing a transformation for each point, based on the mean angular motion during the recording of the point cloud (exemplary corrected point cloud with the algorithm in Figure 1). We validate the algorithm on LiDAR point clouds recorded with a Livox MID-100 mounted on a spherical robot. The contributions of this work are as follows:

- A simple algorithm for angular motion distortion correction (AMDC), designed for spherical mobile mapping systems.
- An evaluation of the performance compared to state-of-the-art, based on the comparison of the point clouds recorded with our spherical mobile mapping system with ground truth point clouds.

The paper is structured as follows: In the following section, we provide an overview of the most similar approaches concerned with motion distortion of point clouds. Then, we are going to introduce the angular motion distortion correction algorithm in a general fashion. Finally, we introduce accuracy measurements and experiments and show that the algorithm corrects the distortion induced by the motion of a spherical robot.

## II. Related Work

Ego motion distortion is a well-researched field and is often considered preliminary for further processing of LiDAR point clouds. However, most approaches considering motion distortion are developed with a background in autonomous driving with car-like vehicles. These vehicles are limited in their locomotion and are usually dominated by translations [16], [23]. In [23] Merriaux et al. study the impact of motion distortion concluding that either high linear motions or slow angular velocities can result in large distortions in the point cloud. For mapping, this is fatal as objects may appear twice, distorted, or are missing. A correction method is given considering 2.5D correction: Only taking into account linear motion along X and Y and the yaw angle. In [16] Renzler et al. implement distortion correction for race car applications. The high linear velocity distorts the point clouds. The motion distortion correction is also done in 2.5D, although the car's trajectory during the recording of one frame is considered as well. The combination of 6DoF IMUs and 3D LiDAR system is used by Le Gentil et al. [24] for a calibration framework trying to eliminate motion distortion. It uses the preintegration of upsampled IMU readings to model motion distortion. In [25] Wu et al. use adaptive frame length to reduce motion distortion. The frame length is reduced when the vehicle is moving fast and motion distortion is affecting the point cloud, while lower frame rates are used for slower movements.

Point clouds acquired with a LiDAR are the preferred data source for localization and mapping [26]. SLAM (simultaneous localization and mapping) is based on aligning point clouds and finding transformations that describe the motion of the robot. To use SLAM accurate point clouds are necessary. In [27] Zhang et al. propose a method combining motion distortion correction and mapping in real time. The key idea is to divide the underlying complex problem: Where one algorithm is performing odometry at a high frequency, while a second algorithm runs at a lower frequency performing fine matching and registration of the point cloud. In the abovementioned examples, the translation of the LiDAR sensor during the measurement is either estimated via the IMUs accelerometer, GNSS, or a comination of both. However, on a spherical robot the accelerations experienced by a sensor mounted rigidly inside are mostly governed by centripetal forces, which makes these translation models unreliable. Furthermore, GNSS is not always available for exploration tasks, especially in space. Thus, the intuition behind this work is to only compensate the motion distortion caused by rotation. As for the translation, developing a trochoidal motion model for the sensors placed inside the spherical robot using gyroscope data is beyond the scope of this work and part of future work.

## III. Motion Distortion Correction

### A. General Motion Distortion Correction

This work focuses on the ego-motion distortion correction of point clouds recorded on a spherical mobile mapping system. Thus we are primarily concerned with the angular displacement of points in 3-dimensional space while recording one frame. We assume a point $\boldsymbol{p}_i$ to be recorded at a time $t_i$, which will have the wrong coordinates due to the matching of the time-stamp $t_j$ of the frame and the matched pose of the robot at time $t_j$. The time $\Delta t$ that passes, combined with the motion the robot performs during that time interval until the point is recorded, displaces the point in the final scan. To counter the displacement, we must apply an inverse rigid transformation $\boldsymbol{T}_i^{-1}$ which transforms the point to its actual position. We model the transformation of the sensor $\boldsymbol{T}_i$ that happened between the measurement of two points as a $4 \times 4$ rigid transformation matrix in the $\mathrm{SE}(3)$ group. Note that in our case - ignoring the translation - a rotation matrix $\boldsymbol{R}_i \in \mathrm{SO}(3)$ would be sufficient. However, we will extend this work including a motion model for the translation in the future.

### B. Angular Motion Distortion Correction (AMDC)

For spherical robots, we assume the distortion of the point cloud $\boldsymbol{PC}_j$ to be entirely of rotational nature, as linear motion has a far smaller impact on the distortion and thus quality of the point cloud. We therefore define the transformation of the distortion as a pure rotation and acknowledge that the components of $\boldsymbol{T}^{-1}$ corresponding to translation must be zero. Furthermore, we derive the part of our transformation corresponding to rotation, by using the available data of an IMU. Specifically, we approximate the rotational motion during one frame using the angular velocity $\boldsymbol{\omega}_j$ from the IMUs gyroscope. As a simplification, the algorithm assumes the rotational velocity to be constant during one frame. Thus, we compute the average angular velocity $\bar{\boldsymbol{\omega}}_j$ using $N$ measurements between times $t_j$ and $t_{j-1}$, corresponding to the timestamps of two subsequent LiDAR frames.

$$\bar{\boldsymbol{\omega}}_j = \frac{1}{N} \sum_{k=0}^{N} \boldsymbol{\omega}_{j-k} \tag{1}$$

A reference timestamp must be assigned for each point-cloud frame, which corresponds either to the start or the end of the measurement. For the above Equation 1 it is useful to assign the time stamp for the frame at the end of the measurement, thus the mean velocity is computed for the current time interval $\Delta t_j$. We then find a transform $\boldsymbol{T}_i^{-1}$ for each point by combining the mean angular velocity and an offset to the frame's time stamp. Thus, we introduce a metric $c_i$ measuring the normalized time of the recording of point $\boldsymbol{p}_i$ at time $t_i$ to the time stamp $t_j$.

$$c_i = -\frac{t_i - t_j}{t_j - t_{j-1}} \tag{2}$$

Notice that $c_i$ is a positive number between 0 and 1 as the fraction will always be negative and thus result in a positive offset to the time stamp $t_j$ associated with the last recorded point in the point cloud $\boldsymbol{PC}_j$.

Bringing all the information together, we compute the transformation for each point by integrating the angular velocity up to the point's recording at $t_i$. For this, we first compute the overall rotation $\Delta\bar{\boldsymbol{\Theta}}$ in the $x$, $y$, and $z$ axis in the time interval $\Delta t_j = t_j - t_{j-1}$, and second compute the fraction of this rotation corresponding to each individual point $\boldsymbol{p}_i$.

$$\Delta\bar{\boldsymbol{\Theta}} = \Delta t_j \cdot \bar{\boldsymbol{\omega}}_j \tag{3}$$

$$\Delta\bar{\boldsymbol{\Theta}}_i = -c_i \cdot \Delta\bar{\boldsymbol{\Theta}} \tag{4}$$

Note that the negative sign in Equation 4 accounts for a rotation opposing the direction of motion because the last point in the point cloud is defined as the only one with the correctly assigned pose and timestamp.

The transformation for motion distortion for each point is computed using a rotation matrix $\boldsymbol{R}_{\alpha,\beta,\gamma}$ with the yaw, pitch, and roll angles $\alpha$, $\beta$, and $\gamma$ respectively.

$$\boldsymbol{T}_i^{-1} = \begin{pmatrix} \boldsymbol{R}_{\alpha,\beta,\gamma} & \boldsymbol{0} \\ \boldsymbol{0}^{\tau} & 1 \end{pmatrix} \tag{5}$$

$$\begin{aligned} \boldsymbol{R}_{\alpha,\beta,\gamma} &= \boldsymbol{R}_z(\alpha) \cdot \boldsymbol{R}_y(\beta) \cdot \boldsymbol{R}_x(\gamma) \\ &= \boldsymbol{R}_z(\Delta\bar{\Theta}_{i,z}) \cdot \boldsymbol{R}_y(\Delta\bar{\Theta}_{i,y}) \cdot \boldsymbol{R}_x(\Delta\bar{\Theta}_{i,x}) \end{aligned} \tag{6}$$

The multiplication of the distorted point, given in homogeneous coordinates, with the transformation matrix will then compute to the corrected coordinates of the point $\boldsymbol{p}_{i,\text{corrected}}$.

$$\boldsymbol{p}_{i,\text{corrected}} = \boldsymbol{T}_i^{-1} \cdot \boldsymbol{p}_{i,\text{distorted}} \tag{7}$$

Performing this transformation on all points in $\boldsymbol{PC}_j$ will result in an angular motion distortion corrected point cloud.

## IV. Experiments and Evaluation

We test our algorithm for motion distortion correction on a spherical mobile mapping system, which was designed in the context of the DEADALUS project [2]. The robot consists of two translucent plastic shells that allow the laser beam to pass through, forming a spherical hull when connected. Figure 3 shows the interior structure consisting of several levels. It houses the power supply and sensors, which include three PhidgetSpatial 3/3/3 1044_1b IMUs, an Intel RealSense T265 stereo visual-inertial tracking camera, and a Livox MID-100 LiDAR-scanner. The main computation unit is a Raspberry Pi Model 3b.

The spherical system does not have any actuators, so for our experiments, the robot is pushed along a path manually. The algorithm is implemented with ROS1 [28]. The pose

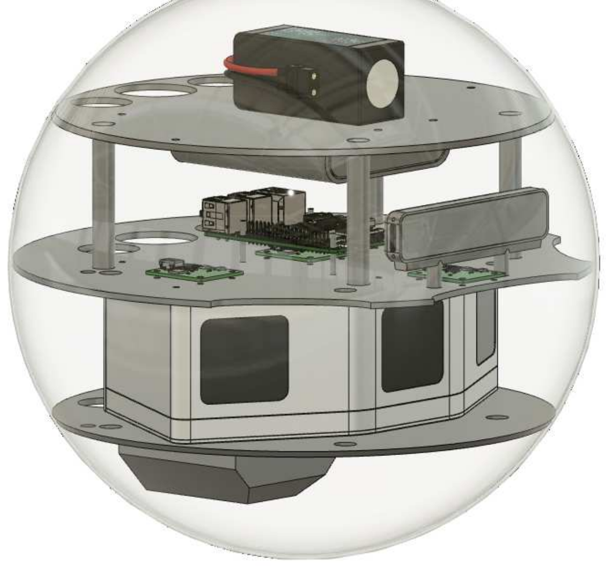


Fig. 3: The spherical robot used for testing our algorithm performing angular motion distortion correction.

of the robot and measurement of motion, especially angular velocity, is of utmost importance for the algorithm and postprocessing. We are using the inertial measurements of the IMUs and a combination of the well known Madgwick filter [29] and the complimentary filter [30] to compute a 6-DoF pose estimate for the robot [31]. Moreover, we use the Delta pose filter [32], to perform sensor fusion of the pose computed from the IMU readings and the pose computed by the Intel RealSense camera. The full 6-DoF pose is not necessary for the distortion correction algorithm, as it relies only on filtered gyroscope readings from the IMUs. However, later processing and visualization of the point cloud requires a corresponding pose for each recorded point cloud. Note that as for the translation, the LiDAR sensor assumes to be placed at the spheres center. Further extrinsic calibration with respect to the spheres center and a corresponding motion correction model for the translation is beyond the scope of this paper and will be part of future work.

### A. *Error metrics*

We compare the distorted and corrected point clouds with a ground truth point cloud, that is recorded with a Riegl VZ-400 terrestrial laser scanner (TLS). This TLS has an accuracy of $5\,\mathrm{mm}$ and an angular resolution of $0.04°$. We merge the individual point cloud frames of our datasets into to a single point cloud, and then align them with the ground truth map. We use 3DTK for all the processing steps [9]. Then, we calculate the mean point-to-point error $(\mathrm{ME})$ of $N$ corresponding points as

$$\mathrm{ME} = \frac{1}{N}\sum_{i=1}^{N} |\hat{\mathbf{p}}_\mathbf{i} - \mathbf{p}_\mathbf{i}| \ , \tag{8}$$

where $\hat{\mathbf{p}}_\mathbf{i} \subset \hat{\boldsymbol{PC}}$ is the predicted point recorded by the spherical robot and $\mathbf{p}_\mathbf{i} \subset \boldsymbol{PC}$ is the corresponding point in the ground truth.

### B. *Experiments*

We test our angular distortion correction algorithm for varying parameters while keeping the others fixed. First, we vary the scanning frequency of the LiDAR while moving with a fixed angular velocity of approximately $115\ \mathrm{deg}\cdot\mathrm{s}^{-1}$. Second, we vary the angular velocity and use a fixed frequency of $10\ \mathrm{Hz}$. In both scenarios, the distortion of the point cloud is expected to increase due to the following phenomena: A lower scanning frequency allows for more time during which the LiDAR system records points and the robot moves further during one frame. For the same reason, increasing the angular velocity also distorts individual points more dramatically. We test the algorithm on different paths: First, we test it on a straight line to make the results for both varying velocity and frequency comparable, and second test it on arbitrary trajectories. Moreover, we compare the results achieved with our algorithm with the results achieved by the motion distortion compensation in the Livox SDK [33]. It is based on the IMU readings, including the accelerometers, such that it accounts not only for rotations, but also for translations.

The results show a decrease of the motion distortion in all scenarios (see Table I and II and Figure 5). For increasing frequency, we see a decreasing ME except for the $50\ \mathrm{Hz}$ frequency, showing the principle influencing the distortion. Our correction algorithm (AMDC), as well as the Livox SDK motion distortion correction algorithm, compute a point cloud which appears less distorted. For $5\ \mathrm{Hz}$, $20\ \mathrm{Hz}$, and $50\ \mathrm{Hz}$ our algorithm outperforms the Livox SDK algorithm, while showing similar performance for $10\ \mathrm{Hz}$. The mean point-to-point errors are in the same order of magnitude, yet showing a clear tendency that AMDC works better on our spherical mobile mapping system. Both algorithms perform better for a frequency of $10\ \mathrm{Hz}$ with a mean error $(\mathrm{ME})$ of below $13.0\ \mathrm{cm}$. The mean error is highest for the lowest frequency of $5\ \mathrm{Hz}$, which is expected. We highlight that our algorithm performs well on the lowest frequency and outperforms the Livox SDK algorithm by the largest margin. The Figures 6 and 7 show the point cloud recorded at $10\ \mathrm{Hz}$ in its distorted state and after the correction with AMDC. In the distorted point cloud, the fanning effect of motion distortion around the major axis of rotation is apparent. This effect does not appear in the corrected point cloud.

We also test our algorithm at increasing velocity. Figure 5 shows the results. We note that for increasing velocity, the distortion increases linearly. The distortion-corrected point clouds have a similar ME and seem to only slightly depend on the velocity. We rationalize about this by considering the larger noise present in the gyroscope readings.

Finally, we evaluate the algorithms on arbitrary paths. Table II shows the ME of the distorted and the corrected point clouds. Note that even after correction, the distortion is higher than when compared with the controlled motion

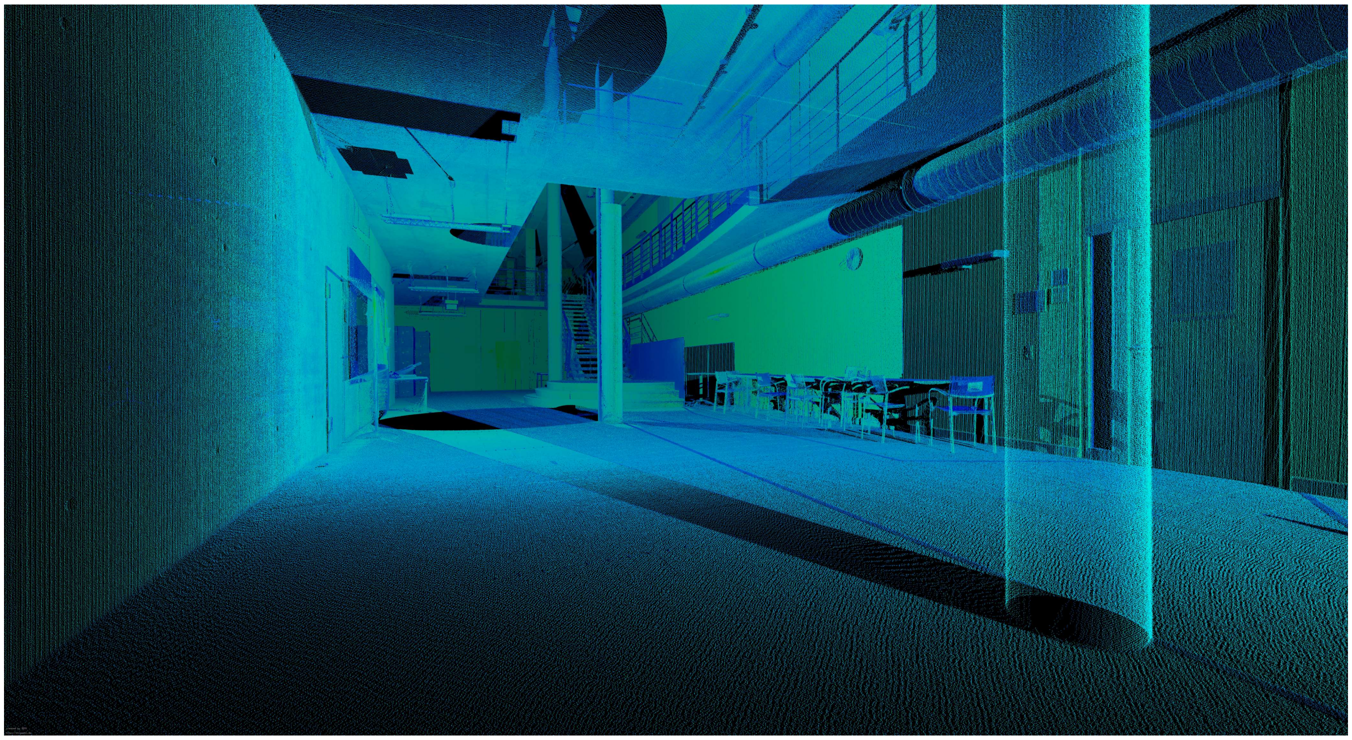


Fig. 4: Ground truth of the site used for the experiments recorded with a Riegl VZ-400. The color of the points corresponds to the reflectivity of the environment.

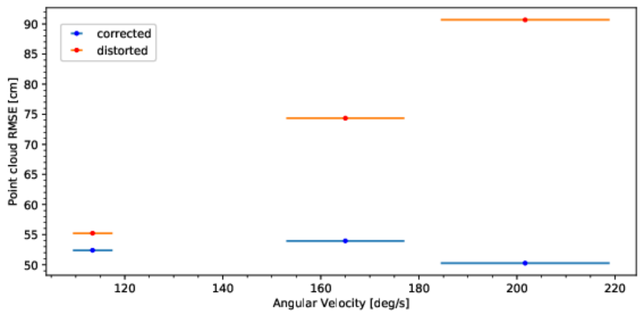


Fig. 5: Motion distortion with increasing velocity. The yellow marks show the distorted point clouds and the blue marks the corrected.

on the track. We explain this by considering the additional larger motions present in the other principle axes.

## C. Discussion

The experiments have shown that a motion compensation algorithm for LiDAR point clouds on spherical platforms is feasable, describing only the rotational motion. The model, only considering the angular velocity and ommiting the linear velocity of the system, especially shows its strengths with lower frequencies. Lower frequencies correspond to more points during a larger time window, and thus more distortion in the point cloud.

Our algorithm, specifically suited for point clouds distorted by angular motion, outperforms the Livox SDK algorithm in many instances and shows similar performance otherwise. This is due to the Livox SDK algorithm utilizing the accelerometer for the translation estimation of the system. This makes sense considering that it is primarily concerned with rotationally more restricted systems. However, inside a primarily rotating system, the accelerometer readings are governed by centripetal forces, making the translation model unreliable.

Testing our algorithm for increasing velocity has shown, that its performance is almost independent of the velocity. In an ideal case the distortion compensated point cloud would have no linearly increasing $\mathrm{ME}$. In our case, however, the increase of the distortion still follows a less steep linear curve. This is due to our imperfectly chosen motion model which averages the rotation between two LiDAR measurements, and increased gyroscope noise.

TABLE I: $\mathrm{ME}$ between point clouds recorded with the spherical mobile platform and ground truth. The $\mathrm{ME}$ of the uncorrected point cloud is compared to the $\mathrm{ME}$ of the Livox distortion correction and the $\mathrm{ME}$ of the AMDC presented in this work. For the experiments, we keep the velocity steady at about $115\ \mathrm{deg}\cdot\mathrm{s}^{-1}$.

| Frame Rate [Hz] | mean error (ME) [cm] | | |
|---|---|---|---|
| | Distorted | AMDC | Livox |
| 5 | 31.842 | **15.274** | 21.483 |
| 10 | 24.030 | 12.880 | **12.435** |
| 20 | 15.714 | **13.423** | 14.957 |
| 50 | 28.358 | **15.009** | 19.482 |

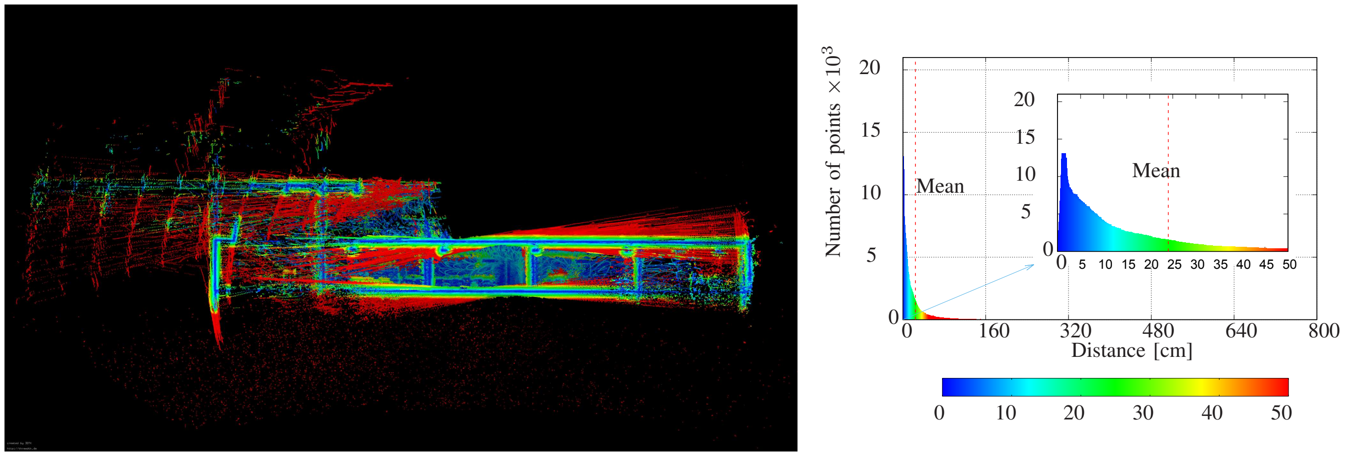


Fig. 6: Distorted point cloud. Side view of the informatics building at the University of Würzburg and the histogram showing the number of points at a given distance from the ground truth. The point cloud was recorded with a frequency of 10 Hz on the track at a speed of around $115\ \mathrm{deg} \cdot \mathrm{s}^{-1}$. Rotation along the rotational axis can be seen by the points marked in red.

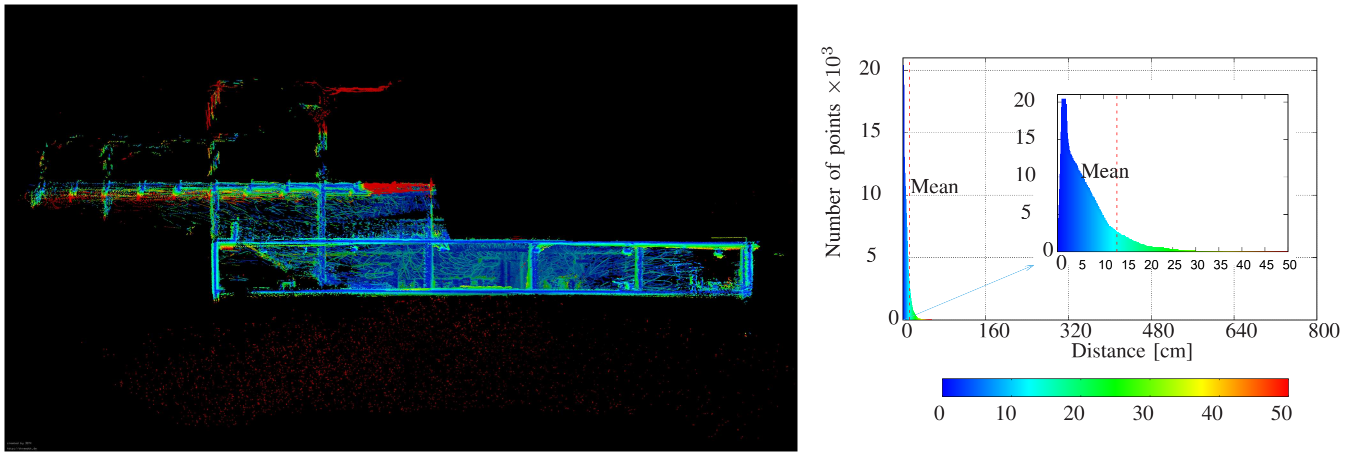


Fig. 7: Corrected point cloud. This is the same recording as in Figure 7 with the AMDC algorithm used for correction. The fanning effect of motion distortion is reduced and most points coincide with the ground truth.

TABLE II: The results of the AMDC motion distorion correction for different paths. The ME of the AMDC and Livox correction are compared to the point cloud without correction. The recording frequency is kept at 10Hz for the experiments.

| Path | mean error (ME) [cm] | | |
|---|---|---|---|
| | Distorted | AMDC | Livox |
| Circular | 30.017 | **22.857** | 22.995 |
| Arbitrary | 29.093 | 21.803 | **21.196** |

## V. Conclusions

In this work, we have presented an algorithm for motion compensation in point clouds on spherical robots. Spherical robots offer many benefits for the exploration of unknown and harsh environments like the moon, with extrem temperatures and radiation. They have a spherical shell that contains all the instruments and are thus protected against damage from the environment. Furthermore, locomotion of the system directly leads to sensor coverage, removing the need of actuators which might break. However, this configuration introduces challenges for measuring the environment with sensors. LiDAR data is recorded in individual frames - smaller point clouds - that are later merged to create a map of the environment or recognize objects in it. The rotating system, however, distorts the data to a greater extent than a system dominated by translations.

The proposed algorithm uses the angular velocity to compute a transformation to correct the position of each point. We have shown that our model for the exclusive rotational motion of the spherical robot performs well and compensates the motion distortion reliably in all tested scenarios. Compared to the algorithm proposed in the Livox SDK, which is not specifically suited for spherical robots, our algorithm outperforms it or at least performs in the same order of magnitude. For a low frequency of 5 Hz, AMDC outperforms the Livox SDK by a large margin, highlighting our tailored motion model. Moreover, we have shown that even for increased velocity, the algorithm keeps performing as expected.

We have shown that, even in its current state, AMDC is a well suited angular motion distortion correction algorithm. The AMDC algorithm offers a good performance for rolling motion, which can be used in further processing for mapping and other purposes. Having access to such data processing algorithms is an important step towards space exploration with spherical mobile mapping systems. However, the algorithm is suitable for the use with any setup and is thus not limited to only spherical systems.

Needlessly to say, a lot of work remains to be done. In future work, we will explore more sophisticated motion models, making it possible to account for trochoidal translational motion as well. This has to include the development of a method for extrinsic calibration of the sensors placed inside the shell with respect to the spheres center, in order to correctly account for the linear motion of the LiDAR. Additionally, we want to conduct further experiments with the algorithm, using also other LiDAR sensors on spherical systems.

## References

[1] P. Heinisch, "LiDAR Distortion Correction on Spherical Robots." `https://github.com/deepcodin/AMDC.git`, 2023.

[2] A. P. Rossi, F. Maurelli, V. Unnithan, H. Dreger, K. Mathewos, N. Pradhan, D.-A. Corbeanu, R. Pozzobon, M. Massironi, S. Ferrari, C. Pernechele, L. Paoletti, E. Simioni, P. Maurizio, T. Santagata, D. Borrmann, A. Nüchter, A. Bredenbeck, J. Zevering, F. Arzberger, and C. A. R. Mantilla, "Daedalus - descent and exploration in deep autonomy of lava underground structures," Tech. Rep. 21, Institut für Informatik, 2021.

[3] M. J. Schuster, S. G. Brunner, K. Bussmann, S. Büttner, A. Dömel, M. Hellerer, H. Lehner, P. Lehner, O. Porges, J. Reill, *et al.*, "Towards autonomous planetary exploration: The lightweight rover unit (lru), its success in the spacebotcamp challenge, and beyond," *Journal of Intelligent & Robotic Systems*, vol. 93, pp. 461–494, 2019.

[4] S. Jin, N. Haghighipour, and W.-H. Ip, *Planetary Exploration and Science: Recent Results and Advances*. 2015.

[5] K. A. Farley, K. H. Williford, K. M. Stack, R. Bhartia, A. Chen, M. de la Torre, K. Hand, Y. Goreva, C. D. Herd, R. Hueso, *et al.*, "Mars 2020 mission overview," *Space Science Reviews*, vol. 216, pp. 1–41, 2020.

[6] C. Weisbin and G. Rodriguez, "Nasa robotics research for planetary surface exploration," *IEEE Robotics & Automation Magazine*, vol. 7, no. 4, pp. 25–34, 2000.

[7] D. Lauretta, S. Balram-Knutson, E. Beshore, W. Boynton, C. Drouet d'Aubigny, D. DellaGiustina, H. Enos, D. Golish, C. Hergenrother, E. Howell, *et al.*, "Osiris-rex: sample return from asteroid (101955) bennu," *Space Science Reviews*, vol. 212, pp. 925–984, 2017.

[8] S. Thrun *et al.*, "Robotic mapping: A survey," 2002.

[9] A. Nuchter, H. Surmann, K. Lingemann, J. Hertzberg, and S. Thrun, "6d slam with an application in autonomous mine mapping," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 2, pp. 1998–2003, IEEE, 2004.

[10] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann, "6d slam—3d mapping outdoor environments," *Journal of Field Robotics*, vol. 24, no. 8-9, pp. 699–722, 2007.

[11] A. Halme, J. Suomela, T. Schönberg, and Y. Wang, "A spherical mobile micro-robot for scientific applications," *Astra*, vol. 96, p. 154, 1996.

[12] F. Wang, C. Li, S. Niu, P. Wang, H. Wu, and B. Li, "Design and analysis of a spherical robot with rolling and jumping modes for deep space exploration," *Machines*, vol. 10, no. 2, 2022.

[14] Q. Zhan, Y. Cai, and C. Yan, "Design, analysis and experiments of an omni-directional spherical robot," in *2011 IEEE International Conference on Robotics and Automation*, pp. 4921–4926, 2011.

[13] Y. Sugiyama, A. Shiotsu, M. Yamanaka, and S. Hirai, "Circular/spherical robots for crawling and jumping," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 3595–3600, 2005.

[15] J. Zevering, D. Borrmann, A. Bredenbeck, and A. Nüchter, "The concept of rod-driven locomotion for spherical lunar exploration robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '22)*, (Kyoto, Japan), pp. 5656–5663, 10 2022.

[16] T. Renzler, M. Stolz, M. Schratter, and D. Watzenig, "Increased accuracy for fast moving lidars: Correction of distorted point clouds," in *2020 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pp. 1–6, 2020.

[17] J. Liu, Q. Sun, Z. Fan, and Y. Jia, "Tof lidar development in autonomous vehicle," in *2018 IEEE 3rd Optoelectronics Global Conference (OGC)*, pp. 185–190, 09 2018.

[18] J. B. Choi, *Environmental Perception for Automated Vehicles: Localization, Mapping and Tracking*. Cuvillier Verlag, 2016.

[19] 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.

[20] B. Behroozpour, P. A. Sandborn, M. C. Wu, and B. E. Boser, "Lidar system architectures and circuits," *IEEE Communications Magazine*, vol. 55, no. 10, pp. 135–142, 2017.

[21] E. Cameron, R. Szumski, and J. West, "Lidar scanning system," European Patent Office, 1991.

[22] R. G. Brazeal, B. E. Wilkinson, and H. H. Hochmair, "A rigorous observation model for the risley prism-based livox mid-40 lidar sensor," *Sensors*, vol. 21, no. 14, 2021.

[23] P. Merriaux, Y. Dupuis, R. Boutteau, P. Vasseur, and X. Savatier, "Lidar point clouds correction acquired from a moving car based on can-bus data," 2017.

[24] C. Le Gentil, T. Vidal-Calleja, and S. Huang, "3d lidar-imu calibration based on upsampled preintegrated measurements for motion distortion correction," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2149–2155, 2018.

[25] W. Wu, J. Li, C. Chen, B. Yang, X. Zou, Y. Yang, Y. Xu, R. Zhong, and R. Chen, "Afli-calib: Robust lidar-imu extrinsic self-calibration based on adaptive frame length lidar odometry," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 199, pp. 157–181, 2023.

[26] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[27] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time.," in *Robotics: Science and Systems*, vol. 2, pp. 1–9, Berkeley, CA, 2014.

[28] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3.2, p. 5, Kobe, Japan, 2009.

[29] 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.

[30] H. G. Min and E. T. Jeung, "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.

[31] 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.

[32] F. Arzberger, F. Wiecha, J. Zevering, J. Rothe, D. Borrmann, S. Montenegro, and A. Nüchter, "Delta filter-robust visual-inertial pose estimation in real-time: A multi-trajectory filter on a spherical mobile mapping system," in *2023 European Conference on Mobile Robots (ECMR)*, pp. 1–8, IEEE, 2023.

[33] Livox-SDK, "Livox cloud undistortion." `github.com/Livox-SDK/livox_cloud_undistortion`, 2022.