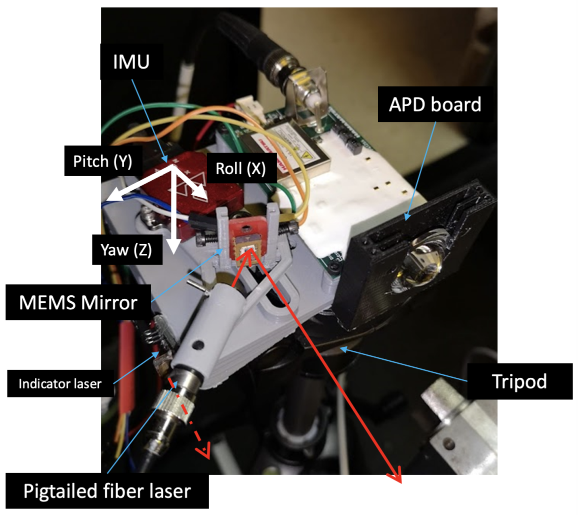
Design of an Adaptive Lightweight LiDAR to Decouple Robot-Camera Geometry

Yuyang Chen*[1], Dingkang Wang*[2], Lenworth Thomas[3], Karthik Dantu[1], Sanjeev J. Koppal[2]

***Abstract*—A fundamental challenge in robot perception is the coupling of the sensor pose and robot pose. This has led to research in active vision where robot pose is changed to reorient the sensor to areas of interest for perception. Further, egomotion such as jitter, and external effects such as wind and others affect perception requiring additional effort in software such as image stabilization. This effect is particularly pronounced in micro-air vehicles and micro-robots who typically are lighter and subject to larger jitter but do not have the computational capability to perform stabilization in real-time. We present a novel microelectromechanical (MEMS) mirror LiDAR system to change the field of view of the LiDAR independent of the robot motion. Our design has the potential for use on small, low-power systems where the expensive components of the LiDAR can be placed external to the small robot. We show the utility of our approach in simulation and on prototype hardware mounted on a UAV. We believe that this LiDAR and its compact movable scanning design provide mechanisms to decouple robot and sensor geometry allowing us to simplify robot perception. We also demonstrate examples of motion compensation using IMU and external odometry feedback in hardware.**



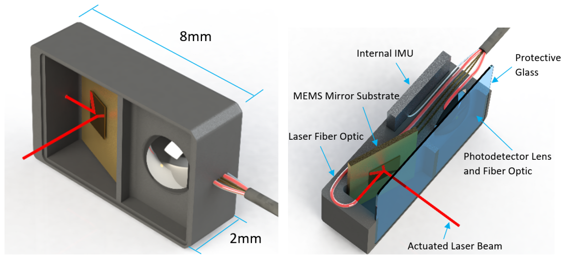


Fig. 1: Our design is given above with the prototype motion-compensated LiDAR (up), and we also prepared a design for future work to integrate this onto smaller platforms.

I. Introduction

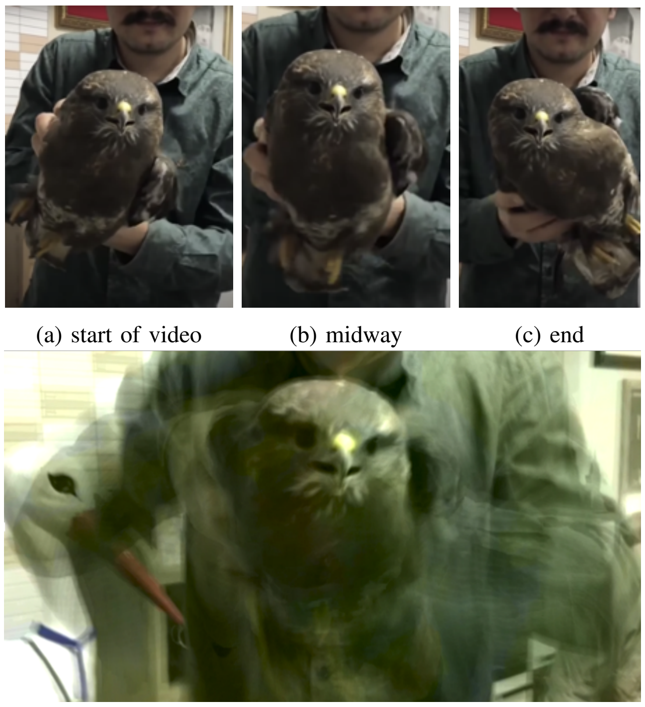
Modern autonomy is largely driven by vision and depth sensors for perception. Most such techniques make an implicit assumption that the relative pose of the sensor w.r.t. the robot is fixed and changes in sensor viewpoint require a change in the robot pose. This implies that fast-moving robots must deal with motion compensation (i.e. camera-robot *stabilization*) and that robots need to reorient themselves to observe the relevant parts of the scene. Correspondingly, stabilization [34, 13, 45, 37] and active vision [6, 4, 62, 38] are well-studied problems.

Let us consider the specific example of image stabilization. While successful, most such methods compensate through *post-capture* processing of sensor data. *We contend that this is simply not feasible for the next generation of fast miniature robots* such as robotic bees [54], crawling and walking robots [19], and other micro-air vehicles [33]. For example, flapping-wing robots such as the RoboBee exhibit a high frequency rocking motion (at about 120 Hz in one design) due to the piezo-electric actuation [18]. Environmental factors such as wind affects micro-robots to a greater extent than a larger robot. There might be aerodynamic instability due ornithopter-based shock absorption [56]. The egomotion of small robots (and onboard sensors) is quite extreme making any sensing challenging. While there have been software methods to correct for such effects for cameras [5] and LiDARs [36], this is often difficult to perform in real-time onboard due to the computational, energy and latency constraints on the robot mentioned above. Without proper motion compensation for miniature devices, we will not be able to unlock the full potential of what is one of the ten grand challenges in robotics [57].

A. *Key Idea: Compensation* during *Imaging*

Our idea is for motion correction to happen in sensor hardware during imaging such that measurements are already compensated without requiring onboard computing. This paper

[1]Yuyang Chen and Karthik Dantu are with the Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY 14228, USA yuyangch@buffalo.edu, kdantu@buffalo.edu
[2,3]Dingkang Wang and Sanjeev Koppal are with the Department of Electrical and Computer Engineering; Lenworth Thomas is with the Department of Mechanical and Aerospace Engineering, University of Florida, Gainesville, FL 32603, USA noplaxochia@ufl.edu, sjkoppal@ece.ufl.edu, lenworth.thomas@ufl.edu
*Equal contributions.


(a) start of video (b) midway (c) end

(d) Average of all images in video

Fig. 2: Biological motion compensation. The position and the angle of the head of the hawk remain stable despite body motion to provide the hawk an stabilized vision. https://www.youtube.com/watch?v=aqgewVCC0k0

shows the motion compensation advantage of decoupling robot-camera geometry, and providing the ability to control the camera properties independent of the robot pose could bring about a new perspective to robot perception and simplify the autonomy pipeline. We demonstrate this through the design of a MEMS-driven LiDAR and perform compensation in two ways - (i) onboard IMU, and (ii) external feedback of robot pose at a high rate.

We are inspired by animal eyes that have fast mechanical movements that compensate for motion, in real-time and at high accuracy [1]. In Figure 2, we show frames $V(t)$ from a video of a hawk (*Buteo jamaicensis*) being moved by a human trainer [16]. We also show the average of the video $\sum_t \frac{V(t)}{T}$ over a time interval T. Note that the averaged image shows motion blurring, except where the eagle mechanically compensates for the shifts. We envision biologically-inspired motion compensation that happens during sensing. These sensors need to adaptively change their orientation, in real-time, and in concert with robot goals such as mapping or navigation. Effectively, the rotation matrix R must cancel out robot motion to provide a ”stable” view of a scene.

B. MEMS Mirror-enabled Adaptive LIDAR

The ability to reorient sensor pose could have many uses in robotics, particularly in image alignment during motion such as in SLAM. If the camera and robot are rigidly attached, then the camera experiences all the motion the robot experiences, including jitter and other potential disturbances that are detrimental to the Visual SLAM task. This could result in spurious features, errors in localization, and incorrect feature association leading to an inaccurate map. In this paper, we describe a sensor design that can perform image reorientation of a LiDAR in hardware without the need for any software processing for such compensation. Previously, pan-tilt-zoom (PTZ) cameras have attempted to address this problem. However, they use mechanical actuation which can react in ones of Hz making it not suitable for tasks such as egomotion compensation in real-time. This is evidenced by the limited use of PTZ cameras on robots - most robots just have sensors rigidly attached.

Our designs break through these past difficulties by exploiting recently available microelectromechanical (MEMS) and opto-mechanical components for changing camera parameters. Opto-MEMS components are famously fast (many kHz), and they allow the changing of the LiDAR projection offset orientation during robot motion, such that the view of LiDAR is effectively static. By changing LiDAR views two orders of magnitude (or more) faster than robot motion, we can effectively allow for camera views to be independent of the robot view. In this work, we can compensate the LiDAR point cloud using an onboard IMU or external feedback such as motion tracking setup. More generally, such compensation allows the robot to focus on the control task while the camera can perform perception (which is required for the control task) independently, and greatly simplifies robot planning as the planner does not need to account for perception and just needs to reason about the control task at hand.

MEMS LiDAR optics have the advantages of small size and low power consumption [46, 24, 25]. Our algorithmic and system design contributions beyond this are:

- We present the design of a novel LiDAR sensor adopting a MEMS mirror similar to this LiDAR MEMS scanner [52]. This design enables wide non-resonant scanning angles for arbitrary orientations. We integrate this with two types of feedback (IMU and external sensors) to demonstrate quick and high-rate motion compensation. Figure 1 shows the design of our sensor.
- We describe and geometrically characterize our sensor, showing that compensation in hardware can reduce the number of unknowns for proprioceptive and exteroceptive tasks. In a simulation, we characterize the effect of compensation delay and compensation rate to identify benefits for robot perception. The quantitative and qualitative results of these simulations are shown in Sect. III.
- We present the compensation control algorithms for our LiDAR. We further characterize the performance of compensation control through experiments and simulations in Section IV.
- We show UAV flight with a proof-of-concept hardware prototype combining external feedback with the MEMS mirror for egomotion compensation. We enable UAV flight by tethering the MEMS modulator to the other heavy necessary components, like the laser, photodetector, optics, the driver circuit, and the signal processing circuitry. The frequencies of the mirror modulation and IMU measurement are much higher than typical robot egomotion. Our prototype MEMS compensated

scan system can perform such compensation in under 10 ms. Please see the accompanying video for proper visualization, and see Fig. 13.

- We provide an implementation of the sensor in the Gazebo simulator. Using this simulated sensor, we propose a framework to adapt modern LiDAR SLAM pipeline to incorporate motion compensation. We adapt a modern LiDAR SLAM pipeline LIO-SAM [41] to incorporate motion compensation to use such a sensor and demonstrate the utility of such motion compensation. We have open-sourced the sensor implementation, the UAV simulation environment, as well as our LIO-SAM adaptations [1] .

II. Related Work

Small, compact LiDAR for small robotics: MEMS mirrors have been studied to build compact LiDAR systems [46, 24, 25]. For instance, Kasturi et al. demonstrated a UAV-borne LiDAR with an electrostatic MEMS mirror scanner that could fit into a small volume of 70 mm $\times$ 60 mm $\times$ 60 mm and weighed about only 50 g [24]. Kimoto et al. developed a LiDAR with an electromagnetic resonant MEMS mirror for robotic vehicles [25].

Comparison to software-based compensation: Motion compensation techniques and image stabilization techniques have been widely used in image captures. Similar to imaging devices, LiDAR point cloud shows point cloud blurring, motion artifacts caused by the motion of the LiDAR or the motion of the target object. Some software-based LiDAR motion compensation use ICP (iterative closest point) [34] and feature matching [13] to find translation and rotation of successive point clouds. Software-based compensation for robotics motion has been studied in great detail in SLAM algorithms [45] or expectation-maximization (EM) methods[37]. Software-based motion compensation have a relative high computation barrier for micro-robotics and may degrade if the point cloud have large discrepancy. Some of the software-based motion compensation relies on the capture of a full frame of point cloud, so it cannot capture the motion frequency higher than the frame rate. For most of the LiDAR (other than flash LiDAR), especially the single scanning beam MEMS LiDAR, the rolling shutter effect caused by the LiDAR motion jitter remains a problem. In contrast to these approaches, we wish to compensate the sensor in hardware, during image capture. Hardware LiDAR motion compensation has several benefits. First, the compensation can be implemented to every LiDAR scanning pulse (for 2D MEMS based LiDAR), which can correct the rolling shutter effect and improve the motion response range. Second, the motion compensation algorithm very simple and can be implement on a low-power microcontroller or FPGA. Third, even if the hardware motion compensation still have some errors, it provides a better initialization for the following software compensation.

These ideas are closer to how PTZ cameras track dynamic objects [27, 20] and assist with background subtraction [42]. However, compared to these approaches, we can tackle egomotion of much higher frequencies, which is a unique challenge of micro-robots. We compensate signals much closer to those seen in adaptive optics and control for camera shake [2, 49, 3]. In addition, our system is on a free moving robot, rather than a fixed viewpoint.

Motorized gimbals: Comparing to motorized image stabilization systems [23], MEMS mirrors not only have smaller size and lighter weight, but their frequency response bandwidth are better than the bulky and heavy camera stabilizer. The MEMS mirror response time is can be less than 10 ms or even less than 1 ms. The servo motor of the camera stabilizer has a bandwidth width less than 30 Hz because they are bulky and have heavy load [29, 39]. This results in a response time higher than 10 ms.

Motion compensation in displays and robotics: Motion-compensated MEMS mirror scanner has been applied for projection, [14], where hand-shake is an issue. In contrast, we deal with the vibration of much higher frequencies, and our approach is closest to adaptive optics for robotics. For example, [44, 43] change the zoom and focal lengths of cameras for SLAM. We compensate using small mirrors, utilizing a rich tradition of compensation in device characterization[31] and to improve SNR [17]. Compared to all the previous methods, we are the first to show IMU-based LiDAR compensation with a MEMS mirror in hardware.

LiDAR SLAM: Ever since the seminal work of [60], successive LIDAR SLAM designs largely follow a LiDAR SLAM architecture similar to Figure 15, where the front end consists De-skew and Feature extraction stages, while the back end usually consists of ICP and Pose Graph Optimization packages such as g2o [26] or GTSAM [8] that globally optimizes the odometry information as estimated by LiDAR visual odometry. Successive efforts moved towards improvement in the following sub areas: 1) tightly coupling LiDAR and IMU [59]; 2) updating the backend PGO optimizer [41]; 3) updating the back end's ICP [35] [58]; 4) updating the front end's point-cloud data structure to do away with ICP's feature dependence [55] *Nevertheless, to the best of our knowledge, all existing LiDAR SLAM systems are designed for LiDARs that are rigidly attached, via fixed joints, to robots and vehicles.*

Sensor reorientation in Active SLAM: There has been a lot of work in the area of perception-aware path planning. A basic assumption of this line of work is that the sensor is rigidly attached to the robot, and therefore, its field of view can be changed only by changing the pose of the robot. [6][38][9] improve SLAM accuracy by actively changing the robot trajectory to improve the field-of-view. Our sensor can simplify these works by changing the FOV in hardware without requiring additional constraints on the path planning.

III. Understanding the Benefits of Compensated LiDAR in Simulation

A. Basic LiDAR geometry

A MEMS-based LIDAR scanning system consists of a laser beam reflected off a small mirror. Voltages control the mirror by physically tilting it to different angles. This allows for

[1]https://github.com/yuyangch/Motion_Compensated_LIO-SAM

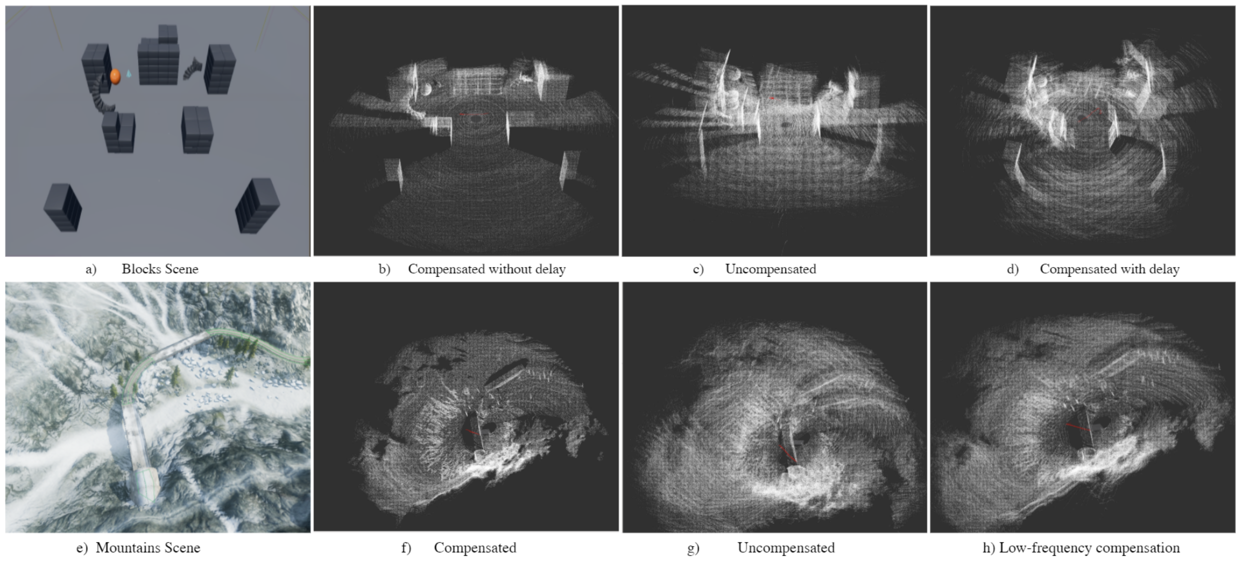

a) Blocks Scene
b) Compensated without delay
c) Uncompensated
d) Compensated with delay
e) Mountains Scene
f) Compensated
g) Uncompensated
h) Low-frequency compensation

Fig. 3: (a) Representative simulation scenario - Blocks scene (b) Mapping the Blocks scene with compensation at 55Hz and no delay (c) Mapping the Blocks scene without compensation (d) Mapping the Blocks scene with compensation at 55Hz and delay of 150ms. (e) Mountains scene (f) mountains scene simulation with 55Hz compensation and 0ms delay (g) mountains scene simulation without compensation. (h) mountains scene simulation with 5Hz compensation and 0ms delay.

LIDAR depth measurements at the direction corresponding the mirror position. Let the function controlling the azimuth be $\phi(V(t))$ and the function controlling the elevation be $\theta(V(t))$, where V is the input voltage that varies with time step t.

To characterize our sensor, we use the the structure-from-motion (SFM) framework with the LIDAR projection matrix $\mathbf{P}$ and the robot's rotation $\mathbf{R}$ and translation t

$$\mathbf{P} = \mathbf{K}\begin{bmatrix}\mathbf{R} & t\end{bmatrix} \tag{1}$$

Where $\mathbf{K}$ is an identity matrix.

In our scenario, the 'pixels' $\mathbf{x}$ relate to the mirror vector orientation $(\theta(V(t)), \phi(V(t))$ on a plane at unit distance from the mirror along the z-axis, and are obtained by projections of 3D points $\mathbf{X}$. Many robotics applications need point cloud alignment across frames, which needs us to recover unknown rotation and translation that minimizes the following optimization.

$$\min_{\mathbf{R},t} \|\mathbf{x} - \mathbf{P}\mathbf{X}\|. \tag{2}$$

This optimization *usually happens in software, after LiDAR and IMU measurements* [47]. Our key idea is that the MEMS mirror provides an opportunity to compensate or control two aspects of the projection matrix $\mathbf{P}$ *before capture, in hardware.*. In this paper, we propose to control a new aspect of the SFM equation in hardware: the rotation matrix $\mathbf{R}$. Given the robot pose (from onboard IMU or other sensing) and the intrinsic matrix, we can easily perform post-capture translation estimation.

$$\min_{t} \|\mathbf{x} - \mathbf{P}\mathbf{X}\|. \tag{3}$$

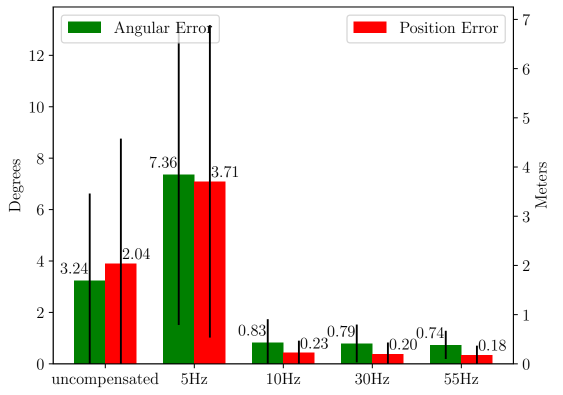
In other words, hardware compensation with MEMS mirrors simplifies the post-capture LIDAR alignment methods to *just finding translation* t, allowing for lightweight and low-latency algorithms to be used with minimal computational effort.

B. Benefits of IMU-compensated LiDAR in SLAM

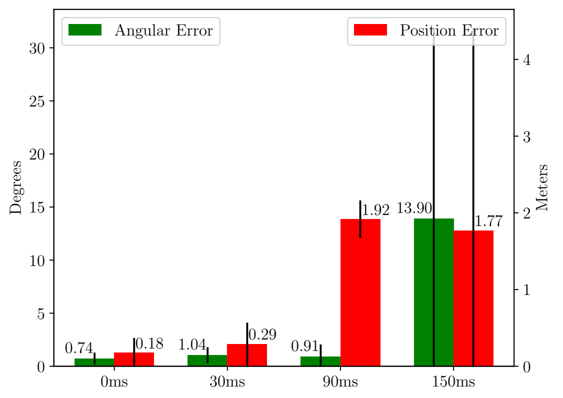
We demonstrate the benefits of motion compensated LiDAR in simulation. Our setup is as follows - we use Airsim [40] running on Unreal Engine 4 for realistic perception and visualization. We tested two scenarios - a scene with geometric objects, called *Blocks scene* shown in Figure 3(a), and an outdoor scene with a bridge and mountains, called *Mountains scene* shown in Figure 3(e). In both scenes, the LiDAR is mounted on a prototype quadrotor UAV. We run LOAM [60], an open-source state-of-the-art LiDAR SLAM system to map the environment and localize the UAV.

As described earlier, motion compensation can be achieved through various means such as a gimble, active compensation of a pan-tilt-zoom camera or MEMS-based hardware compensation like our system. The differences between these methods are along two dimensions - (i) latency of compensation, called *compensation delay* from now on, and (ii) number of times we can compensate in a second, called *compensation rate*. By varying these two parameters in simulation, we compare each method's performance. In order to systematically compensate based on IMU input, we perform some pre-processing of the IMU data. To smooth out the high angular velocity body movements, an angular moving average LiDAR stabilization algorithm is implemented. This method stores the past UAV orientations in a sliding, fix length queue, and reorients the mounted LiDAR towards the average of the past orientations. The average of the orientations is calculated through Linear Interpolation (LERP) of the stored quaternions. We detailed our calculations in IV-B4

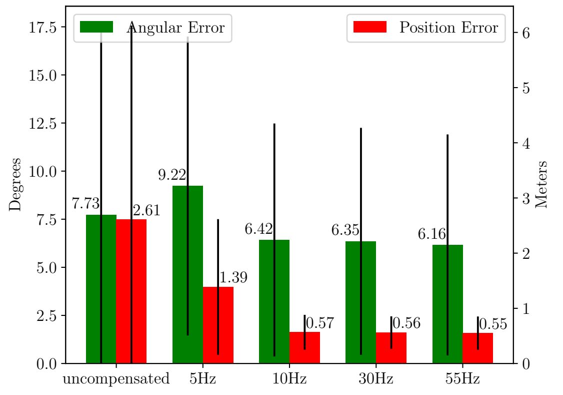
The method is also known as Quaternion L_2-mean [15]. Given the relative short duration of the sliding window, and the relatively small range of rotation that's cover during simulation



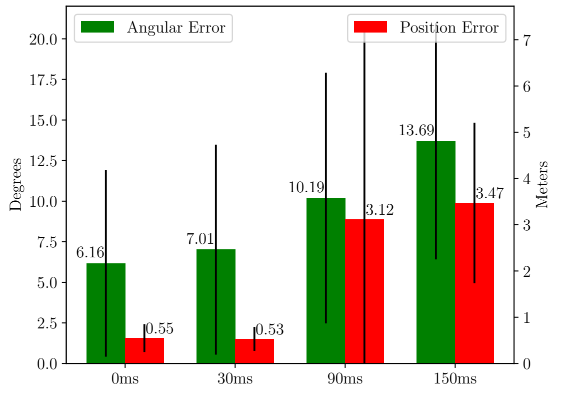
(a) Odometry error vs compensation rate (Blocks scene)



(b) Odometry error vs compensation delay (Blocks scene)



(c) Odometry error vs compensation rate (Mountains scene)



(d) Odometry error vs compensation delay (Mountains scene)

Fig. 4: UAV odometry error while varying compensation rate and compensation delay in two scenes

flights, the prerequisite of using this method is met. It helps remove the impulsive jerky movements that may be observed by the LiDAR, akin to a low-pass filter.

In the experiment, the UAV performs three back-and-forth lateral flights between two way points. During the alternation of way points, the UAV reaches 130°/s in the X body axis. The mounted LiDAR is configured at 16 channels, 360 degree horizontal FOV, 30° vertical FOV and with 150,000 Hz sample rate, akin to commercially available LiDARs.

To quantify performance, we calculate *odometry error*, the difference between the ground truth UAV positions and those positions estimated by LOAM. Figure 4 show the results from our simulations for Blocks scene and Mountains scene. We set the compensation rate to five different values - uncompensated, 5Hz, 10 Hz, 30 Hz, and 55 Hz. We set the compensation delay to five values - no delay (0 ms), 30 ms, 90 ms and 150 ms.

Both the position error and angular error are high when the compensation rate is uncompensated or 5 Hz in the Blocks scene (Figure 4a). It is significantly lower for 10 Hz, 30 Hz and 55 Hz. This shows that smaller rates of compensation as performed by a mechanical gimbal or a PTZ camera (which operate at 5 Hz or lower) are far less effective than a faster compensation mechanism such as the one proposed by us. Similarly, the error in position as well as orientation is low when the compensation delay is either 0ms or 30 ms (Figure 4b).

For larger compensation delays such as 90ms and 150 ms, the error is several times that of when the compensation delay is 30 ms. This shows that as the compensation delay is higher, as it could be with software-based compensation on low-power embedded systems, it is far less effective and leads to greater error in trajectory estimation. This further argues for a system such as ours that is able to perform compensation in hardware, and therefore at a higher rate. The trends are similar, albeit less pronounced in the Mountains scene where features are much less distinct and feature matching is more challenging in general. This proof-of-concept set of simulations encouraged us to build our proposed system.

IV. Novel LiDAR Design

We propose a simple and effective design, where the MEMS mirror and photodetector are placed on a movable head. For image stabilization, we are also able to place the IMU there. A LiDAR engine and accompanying electronics are tethered to this device, which can be light and small enough for micro-robots. To enable both the LiDAR scanning and compensated scanning at high rate, it is important to understand the characterization of the MEMS scanner.

A. The MEMS mirror

All the compensation effects and size advantages described so far will be nullified if the MEMS mirror cannot survive the shock, vibration and shake associated real-world robots. Here we analyze the robustness of the MEMS mirror device for such platforms. Most MEMS mirrors rely on high-quality factor resonant scanning to achieve wide field-of-view (FoV), which leads to heavy ringing effects and overshoot with sudden changes of direction [32, 50]. A suitable MEMS mirror for motion-compensated scanning is expected to have a wide non-resonant scanning angle, smooth and fast step responses ,can operate under common robotics vibration and can survive shock. To achieve this goal, we adopt a popular electrothermal bimorph actuated MEMS mirror design [22, 51] to build this MEMS mirror. The employed MEMS mirror is fabricated with Al/SiO$_2$ based inverted-series-connected (ISC) bimorph actuation structure reported in [22]. This type of MEMS mirror has the advantages of simple and mature fabrication process [61, 53], wide non-resonant scanning angle, linear response and good stiffness. A new electrothermal MEMS mirror is designed and fabricated with the adaption of the motion compensation application. We note that other previously reported MEMS mirrors with electrothermal actuators, electrostatic actuators, or electromagnetic actuators may also be applicable to the motion compensated LiDAR scanning [24, 21, 52].

B. Compensation Algorithm

In the previous sections, we saw the advantages of MEMS mirror-based compensation and the feasibility for use in a robotic LiDAR. Here we focus on the details of the hardware-based rotation compensation algorithm using MEMS mirror scanning LiDAR and sensing for the compensation.

The MEMS mirror reflect a single ray of light towards a point in the spherical coordinate $\{\alpha, \beta, r\}$. The $\{\alpha, \beta\}$ are the two angular control input to the mirror to achieve such target. We will first establish the local(robot) and global (world) frames, then introduce known helper conversion from spherical to Cartesian coordinates, and finally gets into the details of compensation.

1) Preliminaries:

a) Coordinate System: Our LiDAR can compensate for rotation, but it can not compensate for translation. So all discussion here on in drops translation from $SE(3)$ and will only be focus on $SO(3)$. Let the robot have rotation $\boldsymbol{R}^w_{robot} \in SO(3)$ relative to the world frame. In here, the frame of the un-moving base of the LiDAR sensor have Identity rotation $\boldsymbol{R}^w_{base} \in SO(3)$ and therefore identical $SO(3)$ transformation as the robot frame.

b) Spherical-to-Cartesian Conversions: It is important to outline the conversion from the spherical, which is the control coordinate, to normal Cartesian coordinate. Points in the spherical coordinate $\{\alpha, \beta, r\}$ can be converted to Cartesian coordinate via known equations,

$$p_{cartesian} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} r\cos\alpha\cos\beta \\ r\cos\alpha\sin\beta \\ r\sin\alpha \end{bmatrix} \tag{4}$$

and vice versa:

$$p_{spherical} = \begin{bmatrix} \alpha \\ \beta \\ r \end{bmatrix} = \begin{bmatrix} \arctan\frac{z}{\sqrt{x^2+y_M^2}} \\ \arctan\frac{y}{x} \\ \sqrt{x^2+y^2+z^2} \end{bmatrix} \tag{5}$$

Note that both $p_{cartesian}$ and $p_{spherical}$ are points located in the robot's local coordinate frame, $\boldsymbol{R}^w_{robot}$. Other literature's refer to this frame as the local frame, or camera frame.

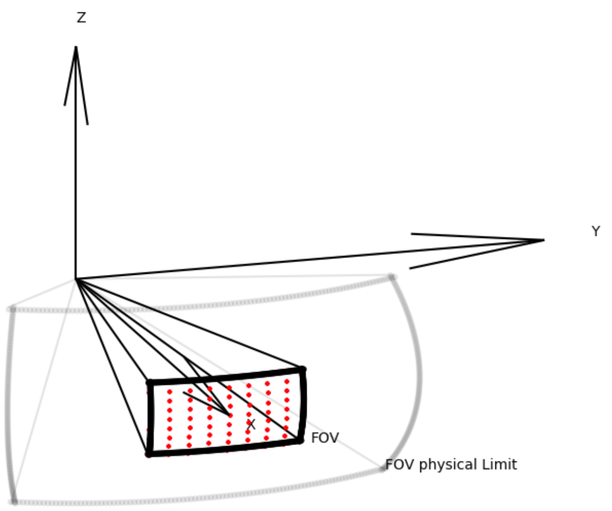


Fig. 5: Depicting Spatial Scanning Grid in the sensor's base frame. In the real rensor, the resolution of the scanning grid is higher at 20x20. The input rotation here is zero. In other words, $\boldsymbol{R}_{control} = \boldsymbol{I}$.

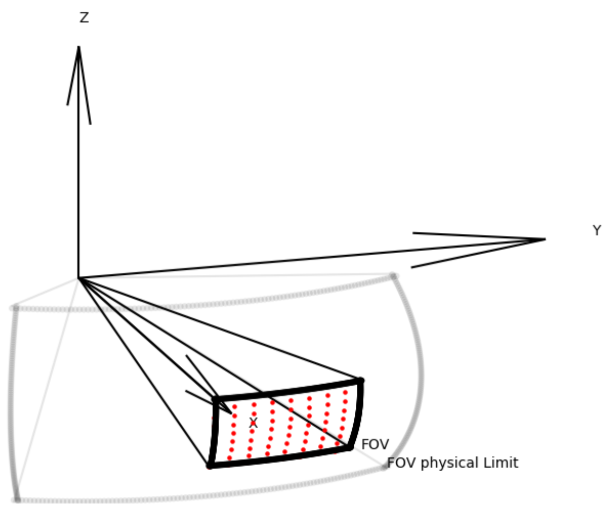


Fig. 6: Depicting Spatial Scanning Grid in the sensor's base frame, The input rotation is is none-zero here.

c) Spatial Scanning: A set of i spherical control coordinates $\{\alpha_i, \beta_i, r_i\}$ defines the scanning pattern of the LiDAR. We use $r_i = 1$ for unit length vectors. In our setup, $\{\alpha_i, \beta_i\}$ defines a rectangular scanning grid in the spherical coordinate, whose center is the principle axis. See figure 5. Within this limit, the mirror can direct its beam to any point desired by the user.

d) Desired Sensor World Frame Rotation: In our design, users can define a desired world frame rotation of the sensor, separately from the world frame rotation of the robot. The rotational decoupling of a sensor and a robot provides many benefits which we demonstrate through various application in this work. Let $\boldsymbol{R}^w_{desired} \in SO(3)$ be the desired rotation target in the world frame. $\boldsymbol{R}^w_{desired}$ can be decided by the users. For example, it can be a slower changing rotation, relative to the robot's body frame. We demonstrated the benefit of such application in III-B. We will touch on the exact details later in IV-B4. Other possibility includes aiming towards a specific world frame target $t \in \mathbb{R}^3$ which we will touch on later, in IV-B5.

2) General Rotation Compensation: Given robot world frame rotation $\boldsymbol{R}^w_{robot}$, user desired sensor rotation $\boldsymbol{R}^w_{desired}$ and a set of spatial scanning, spherical, sensor input coordinates $\{\alpha_i, \beta_i, r_i\}$, we need to find the adjusted sensor input coordinates $\{\alpha^*_i, \beta^*_i, r^*_i\}$, in order to achieve user desired sensor rotation $\boldsymbol{R}^w_{desired}$. We will outline the calculations step by step.

a) Step1: we first translate each $\{\alpha_i, \beta_i, r_i\}$ to Cartesian $p_{cartesian}$ by Equation 4. This step is necessary, in order to calculate points transformation with rotation matrices.

b) Step2: The control rotation input to the sensor $\boldsymbol{R}_{control}$, is the difference between the desired world frame sensor rotation $\boldsymbol{R}^w_{desired}$ and the robot's current world frame rotation $\boldsymbol{R}^w_{robot}$.

Put it formally, Let $\boldsymbol{R}_{control}$ be the rotation from robot rotation $\boldsymbol{R}^w_{robot}$ to the desired rotation $\boldsymbol{R}^w_{desired}$, therefore $\boldsymbol{R}^w_{desired} = \boldsymbol{R}_{control}\boldsymbol{R}^w_{robot}$. We have,

$$\boldsymbol{R}_{control} = \boldsymbol{R}^w_{desired}(\boldsymbol{R}^w_{robot})^T \tag{6}$$

Intuitively, When there is no difference between the desired sensor rotation and the robot rotation, where $\boldsymbol{R}^w_{desired} = \boldsymbol{R}^w_{robot}$ then $\boldsymbol{R}_{control} = \boldsymbol{R}^w_{desired}(\boldsymbol{R}^w_{desired})^T = \boldsymbol{I}$. And $\{\alpha_i, \beta_i, r_i\} = \{\alpha^*_i, \beta^*_i, r^*_i\}$ This default orientation is shown in figure 5. When there is a difference however, an example is shown in Figure 6.

c) Step3: Now, all points in the spatial scanning pattern $p_{cartesian} = \{x_i, y_i, z_i\}$ of the robot frame $\boldsymbol{R}^w_{robot}$ can be transformed to have the desired sensor world frame rotation $\boldsymbol{R}^w_{desired}$:

$$p_{desired-cartesian} = \boldsymbol{R}_{control} p_{cartesian} \tag{7}$$

substituting $\boldsymbol{R}_{control}$, we have

$$p_{desired-cartesian} = \boldsymbol{R}^w_{desired}(\boldsymbol{R}^w_{robot})^T p_{cartesian} \tag{8}$$

d) Step4: Finally, we can translate the rotated points $p_{desired-cartesian_i}$ back to the spherical coordinate $p_{desired-spherical_i}$, via equation 5 for point i's rotation control input to the sensor. Now that we have come to our answer for $\{\alpha^*_i, \beta^*_i, r^*_i\}$.

It is important to note that, this full $SO(3)$ compensation is only achievable because our LiDAR project individual point p_i independently from other points in the set. In the case of a traditional camera or a commercially available LiDAR like Velodyne, The entire set of p_i can be viewed as being projected as a group and correlate to each other. In these other sensors, Full $SO(3)$ compensation is not achievable, even if the sensors are mounted to the robot by a universal joint with 2 degree-of-freedoms α, β. But we will also analyze this special case of grouped points re-projection since our LiDAR can achieve this 2-axis-only compensation.

3) Special Case: 2-axes only compensation: It is important to analyze the case where the sensor can only rotate in two axes relative to the robot. Such setup is commonly seen in robots with cameras mounted by a 2-axis gimbal, as well as PTZ cameras. Another applicable scenario is when we mount a commercially available Velodyne on a UAV via a universal joint, to perform LiDAR SLAM studies such as in III-B and design motion-compensated LiDAR SLAM VI. Furthermore, when it comes to the target aiming IV-B5, 2-axis rotation is often preferred. Our sensor can perform such compensation as well.

In this subsection, we will outline the control not only for our sensor but all sensors, that mount on robots via joints or gimbals with 2-axis orientation controls.

In IV-B2b we define $\boldsymbol{R}_{control}$ as the difference between the sensor orientation and the robot's orientation. Since $\boldsymbol{R}_{control} \in SO(3)$ it requires at least 3-axis rotation control to achieve.

We can collapse this $\boldsymbol{R}_{control}$ matrix into a rotation matrix that is the composition of two Euler angles. The new rotation matrix will not be identical to $\boldsymbol{R}_{control}$, but it keeps the same sensor principle axis ray direction. We herein refer to the collapsed version as $\boldsymbol{R}^*_{control}$. Formally

Let $\boldsymbol{R}_{control}$ be limited to 2-axes rotation only:

$$\boldsymbol{R}^*_{control} = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\beta & -\sin\beta & 0 \\ \sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{9}$$

Here is how to find $\boldsymbol{R}^*_{control}$ from a given $\boldsymbol{R}_{control}$, step by step.

a) Step1: Rotate the principle axis $\boldsymbol{e}_1$, with $\boldsymbol{R}_{control}$ in our case $\boldsymbol{e}_1 = \{x = 1, y = 0, z = 0\}^T$.

$$\boldsymbol{e}_{rotated} = \boldsymbol{R}_{control}\boldsymbol{e}_1 \tag{10}$$

$\boldsymbol{e}_{rotated}$ is now the new principle axis that our sensor should target. Note that, $\boldsymbol{e}_{rotated}$ is closely related to the ray vector from robot to target in the aiming application, more on this later at IV-B5

b) Step2: We then translate this Cartesian coordinate $\boldsymbol{e}_{rotated}$ vector into the Spherical coordinate, using equation 5. We will get $\{\alpha, \beta, 1\}$.

c) Step3: Finally, we can use equation 9 to find the collapsed $\boldsymbol{R}^*_{control}$ with α, β.

Our LiDAR can then use $\boldsymbol{R}^*_{control}$ to perform 2-axis only compensation. We can simply follow the same steps in IV-B2, except we replaces $\boldsymbol{R}_{control}$ in equation 7 with $\boldsymbol{R}^*_{control}$

Further, this compensation can be readily extended to commercially available cameras and LiDARs (such as Velodyne) mounted on a universal joint to the robot frame or a 2-axis gimbal-mounted camera. The 2-axis angles α, β are enough to describe the two joint rotations.

4) Rotational FOV Stabilization: It is often desirable to have relatively slow rotating sensor world frame FOV in many SLAM related applications. We have demonstrated such benefit in III-B. In here we go into details of how it is achieved with our sensor.

a) Quaternion $L_2 - mean$: Supposedly $q_1.....q_n$ is the world frame quaternions store in a queue data structure, representing the robot's world frame rotation in the last n time stamps. We can find its average via LERP, summing and normalizing the quaternions as 4-vectors [15]:

$$q_{avg} = \frac{\sum_{i=1}^{n} q_i}{||\sum_{i=1}^{n} q_i||_2} \tag{11}$$

q_{avg} can be converted into a rotation matrix $\boldsymbol{R}^w_{desired}$. Along with the robot's current world frame rotation $\boldsymbol{R}^w_{robot}$, we can find the adjusted spherical coordinate control input to our sensors $\{\alpha^*_i, \beta^*_i, r^*_i\}$, according to IV-B2.

5) Target Aiming: Let $t^w_{target} \in \mathbb{R}^3$ be the target of interest in the world frame, and let t^w_{robot} be the robot's current world frame translation. Then

$$p_{aim} = (\boldsymbol{R}^w_{robot})^T (t^w_{target} - t^w_{robot}) \tag{12}$$

outlines the ray direction which we want to align our "principle axis", or the projection center point towards. Following a very similar process as to IV-B3 we can find the controls:

a) Step1: We can simply translate Cartesian coordinate p_{aim} to spherical coordinate via equation 5 to find α, β.

b) Step2: Then compose a $\boldsymbol{R}^*_{control}$ via equation 9 for the entire scanning grid. We can simply follow the same steps in IV-B2, except we replaces $\boldsymbol{R}_{control}$ in equation 7 with $\boldsymbol{R}^*_{control}$

For all other sensors mounted on 2-axis gimbals or universal joints, α, β is enough to describe the joint inputs.

6) MEMS Related details: MEMS-related details, relating to the 1-dimension controls of each actuation axis $\{\alpha, \beta\}$, including analysis of robot motion shock on the MEMS as well as preliminary pointcloud stitching, are included in the appendix.

C. LiDAR Hardware Specifics

Our prototype (Figure 7) uses an InGaAs avalanche photodiode module (Thorlabs, APD130C). A fiber with a length of 3 m delivers the laser from the laser source to the scanner head. The gain-switch laser (Leishen, LEP-1550-600) is collimated and reflected by the MEMS mirror. The X-axis of the IMU (VectorNav, VN-100) is parallel to the neutral scanning direction of the MEMS mirror. The in-run bias stability of the gyroscope is $5-7°$/hr typ. The scanner head sits on a tripod so that it can be rotated in the yaw and pitch directions. In the LiDAR base, an Arduino microcontroller is used to process the ToF signals, sample the IMU signals and control the MEMS mirror scanning direction. The data are sent to a PC for post-processing and visualization.

Since our motivation was use with micro-robots, our maximum detection distance is 4 m with a 80% albedo object and the minimal resolvable distance is 5 cm. The maximum ToF measurement rate is 400 points/sec. According to the compensation algorithm described in the previous section, the MEMS mirror scanning direction is updated and compensated for motion at 400 Hz. We now describe our experiments. Please see the accompanying video for further clarification.

D. Compensation experiments with zero translation

1) Handheld Experiments: To demonstrate the effect of compensation, a visible laser is used instead of the LiDAR IR light to visualize the effect of tracking. We mount the LiDAR MEMS scanner on the UAV, as shown in Figure 7. The MEMS mirror desired scanning angle is set to a single point on the target object (0° by 0°) to make it easier for comparing.

Here the entire scanning grid $\{\alpha_i, \beta_i, 1\}$ consist of one single point only at the projection center. We use the general compensation outline in IV-B2

The UAV together with the LiDAR scanner head is held with hand with random rotational motion in yaw/pitch direction. The upper laser trace comes from the laser rigidly connected to the UAV which indicates the UAV's motion. The lower trace is reflected from the MEMS mirror, which shows the compensated/uncompensated scanning laser. The results are shown in Figure 8. The MEMS scanning laser trace area of the compensated scanning is significantly smaller than the uncompensated scanning trace under similar rotational motion disturbance. The videos of the real-time compensation results is available in the supplementary materials.

Then the IR pulse laser is connected to run the LiDAR. An object of interest (in the shape of a +) is placed 2.4 m away from the LiDAR and at the center of the field of view and the background is at 2.8 m, as shown in Figure 9(a). The MEMS mirror performs a raster scanning pattern with an initial field of view of $-3.5°\sim+3.5°$ in both axes to leave the room for compensation. Each frame has 20 by 20 pixels, and the frame refresh rate is 1 fps. To mimic robot vibration, the tripod is rotated randomly in the directions of yaw (Z-axis) and pitch (Y-axis), and the point clouds are shown in Figures 9(d). Despite the motion of the LiDAR head, the point clouds are quite stable. The differences among all of the point clouds are generally less than 2 pixel in either axis, caused by measurement noise.

Figure 9(c) shows the point clouds without compensated scanning, where the relative positions of the target object in the point clouds keep changing. The target object may come out of the MEMS scanning FoV without compensation. With a continuous rotation of 1.5 Hz in the Y-axis, the same structure may appear in multiple positions in the same frame of the point cloud, as shown in the 3rd figure of Figure 9(c). Multiple frames of point cloud are stocked together and shown in the last column of Figure 9. The object can still be identified in the compensated point cloud (Figure 9(f)), but becomes fuzzy caused by the motion jitter when not compensated (Figure 9(e). The videos of the real-time compensation point cloud results is available in the supplementary materials.

2) Motorized input experiments: We use a separate platform to test the 1-d response of our mirror, with disturbance input from a stepper motor, refer to Fig. 10. The MEMS mirror motion compensation system is controlled by an Arduino Mega. The IMU sends the data to the Arduino at 400

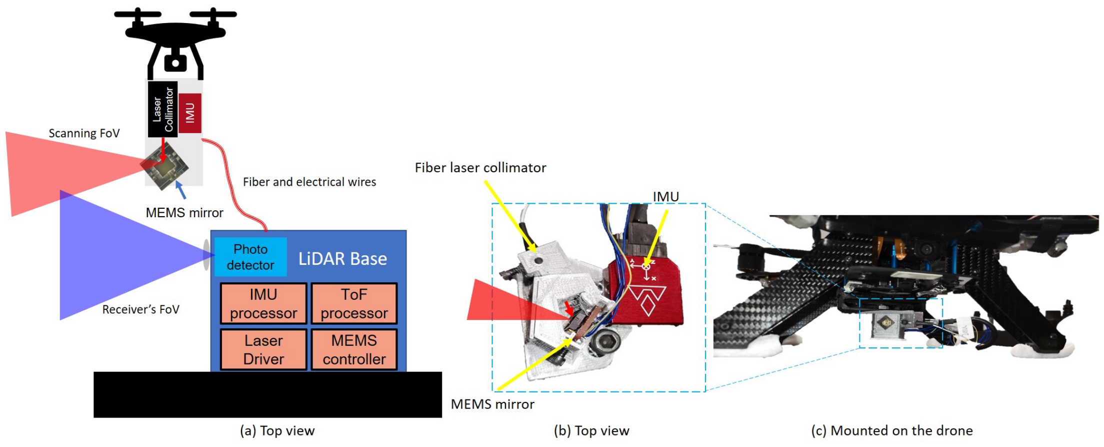


Fig. 7: The movable LiDAR MEMS scanner head, which include the MEMS mirror, an IMU and a fiber laser collimator. (a) shows the top view and (b) shows the LiDAR scanner head mounted to the bottom of the UAV.

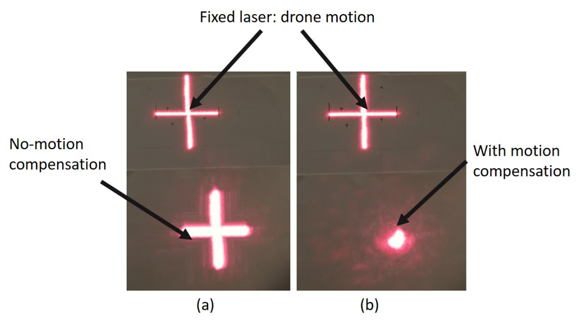


Fig. 8: We use a visible laser to compare the effect of motion compensation of our sensor. The upper laser trace indicates UAV motion, and the lower laser trace indicates the compensated/uncompensated scanning laser reflected from the MEMS mirror. The compensated MEMS scanning (right) shows a much smaller laser trace area than the uncompensated MEMS scanning result.

TABLE I: A comparison of the errors under step motion disturbance and continuous sinusoidal motion disturbance on a step motion with the $H_g(s)$ present or not.

Motor transient time (10-90%)	Mean of Peak Error		Motion disturbance frequency	Mean of Peak Error	
	$H_g(s)$ present	$H_g(s)$ not present		$H_g(s)$ present	$H_g(s)$ not present
55 ms	0.26°	0.36°	6.9 Hz	0.42°	0.56°
85 ms	0.18°	0.26°	3.9 Hz	0.31°	0.24°
150 ms	0.15°	0.22°	2.5 Hz	0.08°	0.16°

Hz. The data is processed, and the compensator $H_g(s)$ (see Supplementary Materials VII)is implemented by the Arduino to get the MEMS angle and the desired driving voltages of the MEMS mirror. The two MEMS orthogonal scanning directions are assembled parallel to two IMU axes. To evaluate the compensation results, the reflected laser is captured by a PSD (position-sensitive detector) sensor fixed on the bench. The PSD sensor is placed 12 cm from the MEMS mirror. The PSD is for compensation evaluation only and is not in the controller loop.

The motion-compensated MEMS scanner test is assembled on a step motor to test the compensation capability under various frequencies. The test bench, including the MEMS mirror, the IMU, and the pigtailed laser are fixed on the shaft of the stepper and rotate with the motor. One of the MEMS scanning directions is coincident with the motor rotation direction. The laser is delivered through a fiber. The stepper has a step size of $1.8°$. With a micro-stepper controller, the approximate minimal step is as small as $0.018°$ for smooth step translation control. The transient time of a $1.8°$ step can be set from 30ms to 500ms. The motion compensation is tested in the pitch direction for smaller errors. The motor is placed horizontally to the ground.

Fig. 11 shows the motion compensation results comparison under various motor speed (t, motor transient time) and with and without the compensator $H_g(s)$. When the motor speed gets faster, the motion compensation errors increase, and $H_g(s)$ can effectively reduce the error.

The motion compensation under continuous sinusoidal drive disturbance is also tested. The motor drives the MEMS mirror scanner head with sinusoidal motions. The MEMS compensated scanning system tries to compensate the scanning angle to an ideal direction. The effect with and without the compensator term is also compared, as shown in Table I.

E. MEMS, Step Response and Robot Motion Shock

We now describes the characteristics of MEMS mirror itself, particularly when it comes to disturbances from robot motion. see Fig. 12a. In a later section V we will show experiment results where the LiDAR is mounted on a UAV and perform target-aiming and spatial scanning tasks.

The mirror has a maximum actuation voltage of 5 V and a scanning FoV of $-4.8°$ to $+5.2°$ in the horizontal axis and $-3.8°$ to $+4.3°$ in the vertical axis (Fig. 12b). The voltage to MEMS tilting angle response is approximately linear. The MEMS mirror can perform non-resonant arbitrary scanning

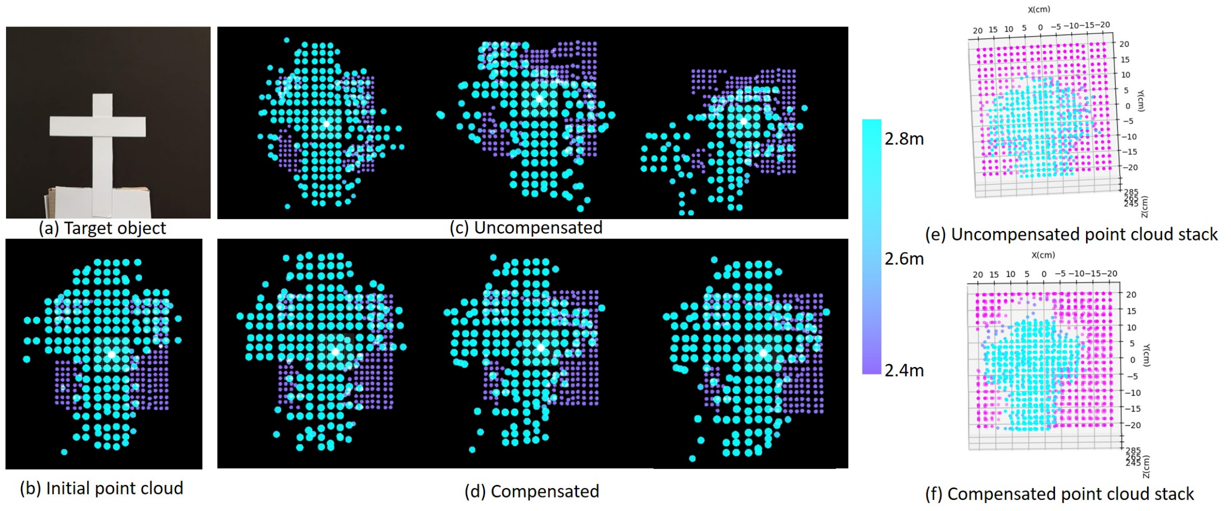


Fig. 9: Motion compensated LiDAR point cloud result with hand held motion disturbance. (a)The target object "+" placed 2.4 m from the LiDAR, along with (b), its initial point cloud scan. (c) and (d) show uncompensated vs. compensated scanning. The hand held rotation angle in (x, y) axes are $(-0.2°, +1.4°), (+1.0°, +1.4°), (-1.5°, +1.7°)$ and compensated angular shake range was $(-1.1°, +1.4°), (+1.2°, -0.5°), (-2.3°, +0.5°)$. (Please see supplementary video). (d and e) The stacking of 5 frame of point cloud of compensated and uncompensated results.

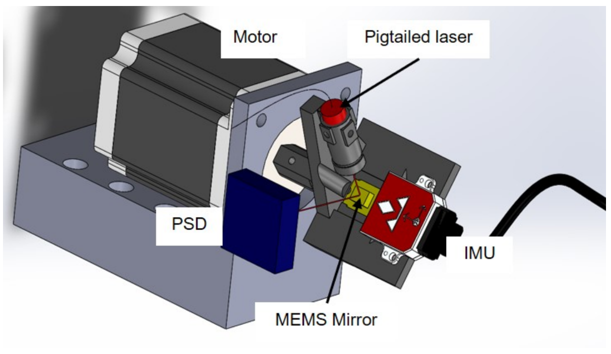


Fig. 10: Input disturbance testing platform with stepper motor.

or pointing according to the control signal. The cross-axis sensitivity is about 6 % in both axes. In the micro-controller, the voltage and the MEMS scanning angle is approximated with a linear relationship with the cross-axis sensitivity taken into considered. The maximum error caused by the non-linearity is $0.3°$.

The step response is 5 ms (Fig. 12c(a)) with very small ringing. To test the frequency response, the frequency of the actuation voltage is swept and the actual tilt angle is measured by tracking the light beam reflected from the mirror plate using a position sensing detector (PSD), shown in Fig. 12d(b). The piston resonant mode is found at f_1=1.07kHz and the tip-tilting resonant modes are at f_2 =1.63 kHz and f_3 = 1.69 kHz.

The tip-tilt scanning response of the MEMS mirror is modeled with a 3rd order system according to [28]. The transfer function $H_1(s)$ can be expressed as,

$$H_1(s) = \frac{\frac{1}{\tau}\omega_n^2}{(s^2 + 2\omega_n\zeta s + \omega_n^2)(s + \frac{1}{\tau})} \tag{13}$$

where τ is the thermal time constant, $\tau \approx t_r/2.2 = 2.3ms$; ω_n is the natural resonant frequency of the mirror rotation, $\omega_n \approx 2\pi(1.65 \text{ kHz})$, and ζ is the damping ratio of the bimorph-mirror plate system, $\zeta \approx 1/2Q = 0.006$. Thus, the transfer function $H_1(s)$ of the MEMS mirror can be obtained by substituting and slightly tuning the parameters in Eq. 13.

Similar to [52], the MEMS mirror is actuated by the PWM signals with a voltage level of 0-5V. The PWM signal can be generated by an Arduino microcontroller at 15 kHz and 8 bit. The ringing of a step response is less than 2 % after about 10 ms. The minimal achievable step is $0.035°$ which is much smaller than the linearization error.

We now show expressions for the acceleration and forces generated by a MEMS mirror scan. The small-angle tip-tilt scanning stiffness k_r is

$$k_\text{r} = I(2\pi f_r)^2 \tag{14}$$

where f_r is the resonant frequency of the tip-tilting modes (f_2, f_3); I is the moment of inertia of the mirror plate alone its tip-tilting axis,

$$I = \frac{1}{12} m_\text{plate}(t^2 + d^2) \tag{15}$$

where t is the thickness of the mirror plate, and d is the length of the mirror plate. The rotation stiffness k_r = 2.16e-6 N·m/rad. With an external angular acceleration of $\ddot{\theta}$ alone on the mirror rotation axis, the excited mirror rotation θ is

$$\theta = -\frac{I\ddot{\theta}}{k_r} = \text{-1e-8} \cdot \ddot{\theta}. \tag{16}$$

Take the mirror scanning step $0.25°$ as the maximum tolerance of the excited mirror plate rotation, the tolerable external angular acceleration is $\ddot{\theta}$ = 44000 rad/s^2. The maximum angular acceleration of a commercialized robot is usually less

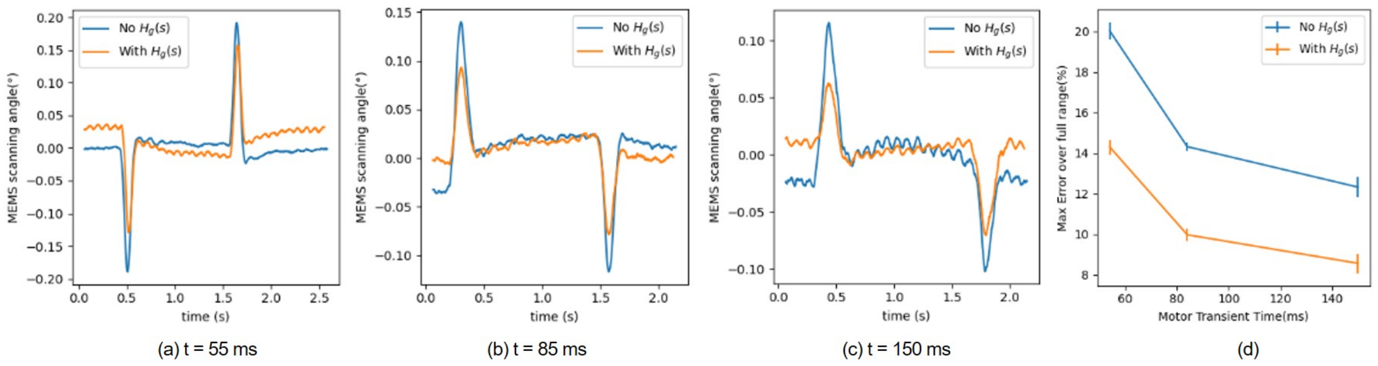


Fig. 11: The motion compensation results comparison under different motor speed (t, motor transient time) and with and without the compensator $H_g(s)$. In these experiments, the mirror is commanded to aim at 0 degree, while the disturbance of 1.8 degrees is input to LiDAR base from a stepper motor. See Fig. 10. Therefore, ideally the MEMS scanning angle stays flat at 0 degree at all times. When the motor speed gets faster, the motion compensation errors become larger. $H_g(s)$ can effectively improve the compensation scanning.

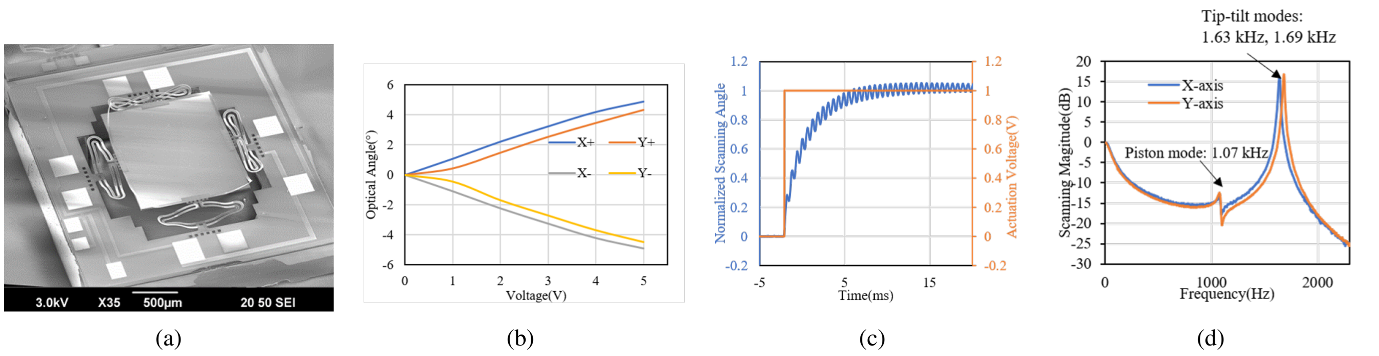


Fig. 12: (a) An SEM image of the fabricated MEMS mirror. (b) The optical scanning angle response of the MEMS mirror. (c) The step response of the MEMS mirror is 5 ms. (d)The frequency response of the MEMS mirror.

than 1000 rad/s^2, and the excited MEMS mirror rotation is less than 6e-4° which can be ignored. Since this MEMS mirror has four identical actuators and the difference on the two axes alone are small, the excited MEMS mirror rotation under robot vibration can be ignored.

We now consider robot crash scenarios. The MEMS mirror can also survive most of the extreme vibration or mechanical shock without failure. The stiffness of the MEMS mirror under shock k_p is:

$$k_\mathrm{p} = m_\mathrm{plate}(2\pi f_1)^2 \tag{17}$$

where m_{plate} is the mass of the mirror plate. Thus, the stiffness the MEMS mirror in piston motion is k_p = 3.2 N/m. The maximum allowable piston displacement of the mirror plate without failure is d_max= 200µm. The maximum tolerable acceleration in the direction perpendicular to the mirror plate is a_{max} is

$$a_{max} = \frac{k_\mathrm{p} d_\mathrm{max}}{m_\mathrm{plate}} = 5500\mathrm{m/s}^2. \tag{18}$$

For most commercial robots, maximum tolerable shock is under 1000 m/s^2. *So the MEMS mirror can survive most of the mechanical shock and vibration of the robot.* External vibration around the resonant frequency will excite large MEMS mirror vibration or even damage the mirror. To avoid the resonance effect, the MEMS mirror should avoid being actuated around the resonant frequency (f_1, f_2, f_3).

F. Robustness to Mirror Control Time Delay

In IV-D2, we quantify the physical system's step response delay time at 5ms see Fig. 12c. In Fig. 11 we further quantify the effects of rotation disturbance of robot, versus compensation error, with various timing profile.

We investigate further the effects of increasing actuation time delay, of either the mirror or the mounting joint, in LiDAR SLAM simulation. We quantify the effects with SLAM odometry error, see Fig. 4b, Fig. 4d in Section III-B.

Further more, in section VI-C6, we further investigate the effects of time delay in our motion-compensated LiDAR Inertial Odometry pipeline, under noisy conditions. To improve on realism, we add control noise, range measurement noise, and IMU noise into our simulation.

G. Robustness to Mirror Control Noise, LiDAR TOF Distance measurement Noise, and IMU Noise

In IV-D2, Fig. 11 we see variations of mirror control noise under disturbances. We use a Gaussian noise model to approximate the control noise, and investigate further on increasing mirror control noise in LiDAR SLAM simulation. We quantify the effects with SLAM odometry error. See section VI,Fig. 18-a).

Further, real IMU has noise, we similarly quantify the effects of increasing IMU noise in VI-C4 and Fig. 18-b).

Furthermore, real LiDAR has TOF distance measurement noise, we similarly quantify the effects of TOF distance measurement noise in VI-C5 and Fig. 18-c).

To conclude, we summarize this section and everything that was described.

- We discuss the key components and characteristics of the proposed LiDAR system design, particularly focusing on the MEMS mirror.
- We outline a rotation compensation control algorithm that uses the MEMS mirror scanning LiDAR and sensing for compensation. We establish the coordinate systems, conversions from spherical to Cartesian coordinates, and details of the compensation process for four applications, including: general rotation compensation, 2-axes only compensation, rotational FOV stabilization and target aiming.
- We analyze the motion compensation control of the proposed LiDAR through real world handheld and motorized input experiments. We present the experiment data related to step response and control error versus input disturbance timing.
- We analyze the robustness of proposed motion compensation control in a tightly coupled LiDAR SLAM simulation. The result is presented in Section VI
- We analyze the survivability of the proposed LiDAR under robot's motion shock.

V. UAV Experiment

Next, we demonstrated the motion compensated LiDAR by flying it on a UAV. The robot pose is from an external motion capture system that tracks the UAV. We vary the robot pose sampling rate and study its effect on the effect of compensation. The UAV is controlled to hover at a designated position with yaw/pitch rotation as motion jitter. Motion compensated LiDAR is set to compensate all the rotational motion, including the controlled rotation and the random motion disturbance. The compensated MEMS scanning laser uses a visible light, and the other visible laser is fixed at a relative higher position on the UAV, as shown in the images in Figure 13b. The target scanning direction is a fixed point on the target.

Here, the entire scanning grid $\{\alpha_i, \beta_i, 1\}$ consist of 20x20 grid pattern points. We use the aiming compensation outline in IV-B5

We trim about 12 s videos in each experiments while the UAV is flying, and the each frames of the videos are accumulated into an image to track the motion of the UAV and the errors of the compensated scanning.

The robot pose sampling rate is set from 1 Hz to 200 Hz to investigate its effect on the compensation results. The controlled UAV rotations are in the yaw and pitch direction. However, the actual motions cause some random motions during the flying. Point clouds are also collected when the UAV is hovering and we overlap several frames. As the robot pose sampling frequency increases from 1Hz, 2Hz to 50Hz, the width of the overlapping area shrink from 10 to 11 points at 1Hz (Fig. 14f), to 6 points at 50 Hz (14d). As the target object in the size of the target object in the point cloud settle down to the a smaller area and the location of the target in the point cloud becomes more certain.

VI. Rotation Compensated LiDAR-Inertial SLAM Design

SLAM is a body of fundamental applications for visual sensors. All exisitng SLAM literatures reason about its odometry in the sensor's local frame, sometimes call camera frame. In this work this frame is the robot frame, with world frame orientation $\boldsymbol{R}^{w}_{robot}$, refer to IV-B1a.

The basic assumption of the existing SLAM is that visual sensor readings use the robot frame with world rotation $\boldsymbol{R}^{w}_{robot}$ as their reference. This assumption is untrue for our sensor, because that our sensor readings use the frame with world orientation $\boldsymbol{R}_{control}\boldsymbol{R}^{w}_{robot}$ as their reference.

Through sections IV-B2 to IV-B5, the additional none-zero rotation $\boldsymbol{R}_{control}$ orients the original scanning grid towards different directions. The existence of $\boldsymbol{R}_{control}$ breaks the basic assumption of existing SLAM.

$\boldsymbol{R}_{control}$ must be compensated for, in order for the existing SLAM pipelines to work with our sensor. This can be done post-capture, we can use either IV-B2 or IV-B3 to compensate. We details the compensation later in VI-B.

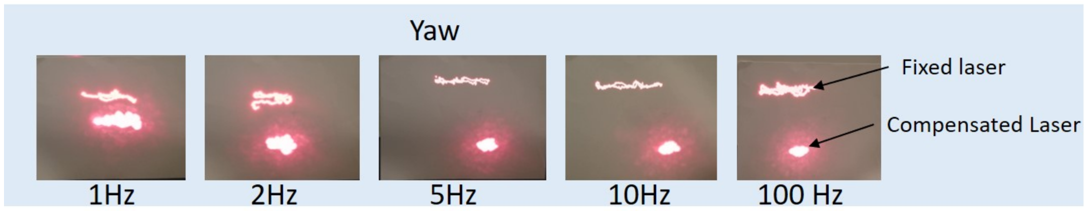
Most LiDAR odometry pipelines utilize Iterative Closest Point (ICP) to match consecutive scans and determine the rotation and translation between the poses. Any rotation of the LiDAR relative to the vehicle would cause errors in the ICP's prior. This would directly impact the quality of ICP's point-cloud registration. Although ICP can tolerate certain levels of error in its prior, in Section VI-C2 we will show that it is far from enough when the magnitude of $\boldsymbol{R}_{control}$ input increases.

A. Motion Compensation for LiDAR SLAM

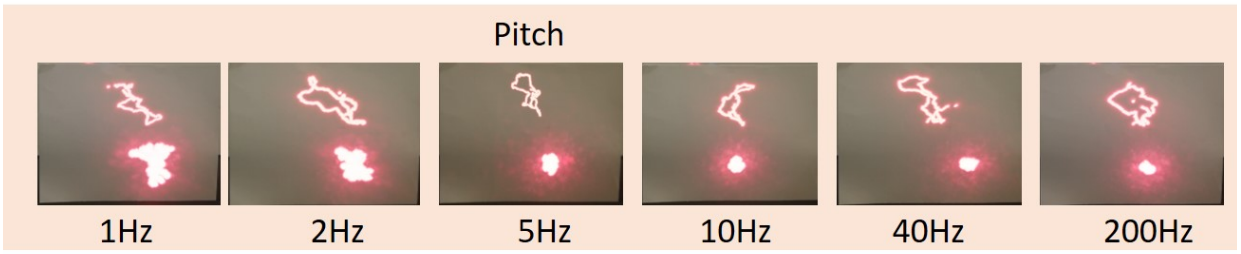
In this simulation, we simulate a 360 degree velodyne LiDAR, that can rotate relative to the vehicle it is mounted on, by a universal joint. A universal joint has rotational DOF similar to a MEMS mirror, both limited to 2 DOF. This setup fits into the compensation framework introduced in the special case IV-B3. In this section, we will demonstrate in simulation that such rotation introduces error in an off-the-shelf LiDAR SLAM pipeline. Additionally, We propose a general method to incorporate such rotation into consideration when performing LiDAR-related SLAM. We demonstrate the effectiveness of the framework in a Rotation Compensated LiDAR-Inertial Odometry and Mapping package, which is publicly available on Github.

For the ease of integration, our framework proposal does not make large edits in the existing paradigm. It only adds a "rotate" stage right after the de-skew stage in the front end, and before feature extraction stage. This edition can be easily integrated with existing pipeline and future designs. The rotate

(a) Our UAV setup with Intel Aero UAV and our LiDAR mounted on it.



(b) Effect of compensation rate on yaw rotation



(c) Effect of compensation rate on pitch rotation

Fig. 13: A comparison of the compensation strength versus the robot pose sampling frequencies. All the images are accumulation of 12s of UAV hovering videos. The compensation target scanning direction is a fixed direction.

stage does one single operation, it rotates the de-skewed point cloud according to the control rotation input to the LiDAR. Our workflow block diagram in shown in Figure 15.

B. The Rotation Stage

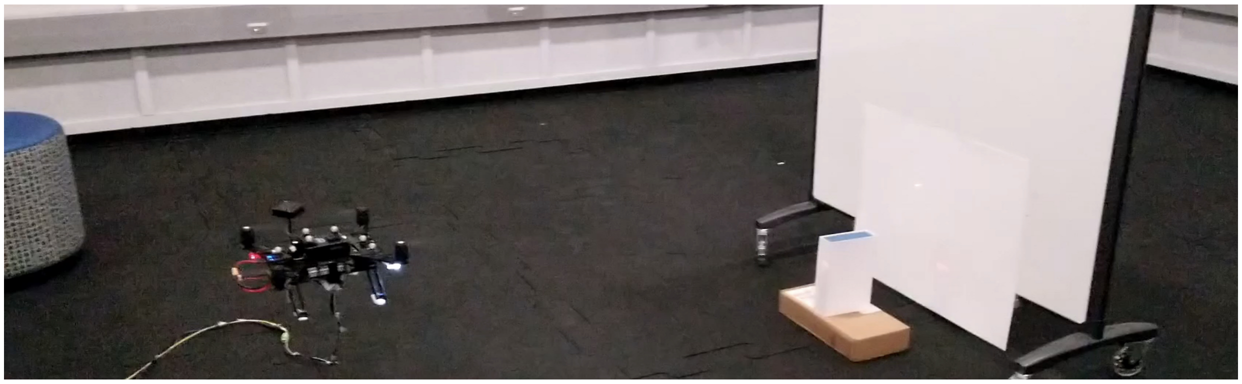
The purpose of this stage of the pipeline, is to rotate the captured LiDAR frame, to a correct position, relative to the LiDAR's base frame of reference. (In this work, the LiDAR's base frame is identical to the vehicle's body frame.)

Let the Lidar's base frame have world rotation $\boldsymbol{R}^{w}_{robot} \in SO(3)$.

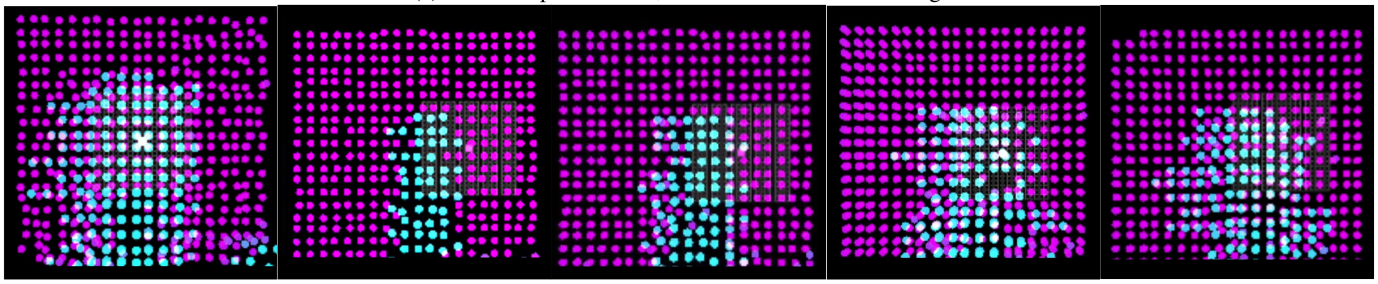
In a traditional LiDAR that doesn't rotate, all points received in a LiDAR frame are position relative to the LiDAR's base frame, with world rotation $\boldsymbol{R}^{w}_{robot}$. However, this assumption is incorrect for our device, where the LiDAR frame is positioned relative to the frame with rotation $\boldsymbol{R}_{control}\boldsymbol{R}^{w}_{robot}$.

The LiDAR's head can rotate $\boldsymbol{R}_{control} \in SO(3)$, relative to its base. This rotation is restricted to azimuth β and elevation directions α. note that in here we analyze a more generalize, special case compensation IV-B3, but the it can be easily extend to full $SO(3)$ compensation IV-B2,

When a LiDAR frame is received, we take the most recently known rotation α, β, in this case the most recent known



(a) Our set up with UAV, LiDAR and the feature target.



(b) Uncompensated (c) One frame (d) 50Hz sampling rate (e) 2Hz sampling rate (f) 1Hz sampling rate

Fig. 14: A comparison of the compensation strength verse the IMU sampling frequencies. The images are accumulation of 20s of point cloud video during the UAV hovering. We use a cuboidal object (as seen in Figure 14a) as object of interest. The width of the target increases due to compensation inaccuracy as we reduce compensation rate from 50 Hz to 1 Hz demonstrating the utility of high rates of compensation even in such static scenarios

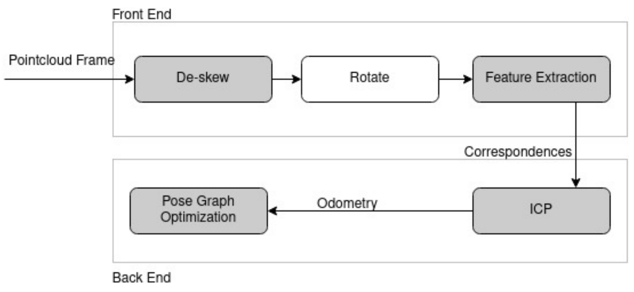


Fig. 15: Illustration of our rotating LiDAR SLAM augmentation pipelines. The existing structures are shown in grey

command rotation, and converts them into a rotation matrix:

$$\boldsymbol{R}_{control} = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{19}$$

And applies the rotation to each point $p \in \mathbb{R}^3$ in the frame point-cloud:

$$p_{rotated} = \boldsymbol{R}_{control} p \tag{20}$$

The rotated point-cloud $p_{rotated}$ now locates at the correct position, relative to the LiDAR's base frame, with world rotation $\boldsymbol{R}^w_{robot}$. The basic assumption of traditional SLAM are now met.

C. Evaluation

Now we evaluate the sensor in simulation to answer a few questions. First, we want to compare traditional LiDAR SLAM and our Motion Compensated SLAM in terms of the handling change in mirror/universal joint orientation magnitude. Next we investigate the effect of noise in the mirror's orientation (say through a faulty IMU or other sensor) on the robustness of our pipeline. We also show the degree to which our pipeline can tolerate such noise.

The proposed SLAM framework should be expected to function, even when the LiDAR users employ control policies that rotate its FOV significantly frame-to-frame. This is Unlike the scenario of running a active stabilization control policy proposed in III-B, Where frame-to-frame variation is minimal. Therefore, in this evaluating section, we use control policy that samples random LiDAR rotation control input from Gaussian distributions at high frequency.

We choose LIO-SAM as the traditional SLAM package to compare against, and built our Motion Compensation framework into it, and open-source it on github. LIO-SAM has all the signature point-cloud processing stages shown in Figure 15. It is relatively new and has good SLAM accuracy performance versus State-of-the-Art. We hope through the open-source code we can demonstrate to the community an example of incorporating our framework.

For Odometry error evaluation, We calculate Average Translation Error (ATE) which is defined by the KITTI benchmark

[11]:

$$E_{trans}(\mathcal{F}) = \frac{1}{|\mathcal{F}|}\sum_{i,j\in\mathcal{F}} \|\hat{T}_j\hat{T}_i^{-1}(T_jT_i^{-1})^{-1}\|_2 \tag{21}$$

Where $\mathcal{F}$ is a set of frames (i, j), T and $\hat{T}$ are the estimated and true LiDAR poses respectively.

1) Experiment Set Up: A simulation study is setup in robotics simulator Gazebo, where a LiDAR with similar sensor characteristics to a VLP-32 [**velodyne**] is mounted on a simulated drone. Further, the LiDAR can rotate in azimuth and elevation via a universal joint. The simulated drone iris, is from the PX4's simulation package. Its onboard IMU have noise added to it according to a noise model outlined in Kalibr [10]. The point-cloud messages from the LiDAR, as well as the IMU messages from the drone, are passed into robotics middleware ROS, where the proposed LiDAR SLAM package runs. The drone is commanded to flight in a diamond waypoint pattern, around a enviornment with different types of resident buildings.

The proposed LiDAR-Inertial SLAM package builds on top of LIO-SAM, which employs the powerful PGO backend GT-SAM [8]. We incoporate the compensation described in VI-B into LIO-SAM, here on in refer to as Motion Compensated LIO-SAM. Naturally, we will compare SLAM performance of Motion Compensated LIO-SAM, against the stock version of LIO-SAM. See Figure 16. To control the orientation of the universal joint, angular commands in α, β, in degrees, are input to the mirror.

2) Level of mirror control orientation magnitude tolerable by an unmodified pipeline vs our system: The two angular commands are sampled from 1-d Gaussian distributions with standard deviation of various degrees, at 10 Hz. Odometry error vs command rotation's gaussian standard deviation is plotted in figure 17. An Gaussian distribution with 8 degree standard deviation generate input angle within +-,8,16,24 degrees, 68,95 and 99.7 percent of the time respectively. Therefore 99.7 percent of the time, angular input span a range of 48 degrees.

In short, by considering mirror rotation, the system can tolerates angular input that span 48 degrees. In contrast, without mirror rotation information, the system can only tolerate angular input that span 12 degrees.

Even in the cases where the input spans less than 12 degrees, by considering mirror rotation, SLAM quality improves in comparison.

3) Level of mirror control noise tolerable : The two angular commands are sampled from 1-d Gaussian distributions with standard deviation of 3 degrees, at 10 Hz. We use our proposed Motion Compensated LIO-SAM here.

Aditionally, Noise rotations in both azimuth and elevation are added on top of each channel. Odometry error vs command rotation's noise Gaussian standard deviation is plotted in Fig. 18-a). The system can tolerate mirror input control noise up to 1.6 degree standard deviation, which spans 9.6 degree.

As Fig. 11 shows, when 1.8 degrees of disturbance from the robot body is generated, there is approximately 10 percent or +-0.18 degrees of peak actuation error, and approximately 3 percent, or +-0.05 degrees of actuation error off peak. Assuming in the worst case a robot generate 18 degrees of disturbances from the robot body, this translate to 1.8 degrees of peak actuation error and 0.5 degrees of actuation error off-peak. Using a Gaussian error profile, we have standard deviation at 1.8/3=0.6 degrees, which is within the noise level that the motion-compensated LIO-SAM can tolerate.

4) Level of IMU noise tolerable: The two angular commands are sampled from 1-d Gaussian distributions with standard deviation of 3 degrees, at 10 Hz.We use our proposed Motion Compensated LIO-SAM here. Additionally, IMU noise are added on top of IMU output, according to an IMU noise model outlined in kalibr [10]. The model is based on [30][48][7]. The base set of noise parameters are outline in the following table.

Parameter	Unit	Value
Gyroscope Noise Density	$\frac{rad}{s}\frac{1}{\sqrt{Hz}}$	3.394e-4
Gyroscope Random Walk	$\frac{rad}{s^2}\frac{1}{\sqrt{Hz}}$	3.879e-05
Gyroscope Turn On Bias Sigma	$\frac{rad}{s}$	8.727e-3
Accelerometer Noise Density	$\frac{m}{s^2}\frac{1}{\sqrt{Hz}}$	4e-3
Accelerometer Random Walk	$\frac{m}{s^3}\frac{1}{\sqrt{Hz}}$	6e-3
Accelerometer Turn On Bias Sigma	$\frac{m}{s^2}$	0.196

The above set of noise parameters are multiply by several factors and their relationship with odometry error are shown in Fig. 18-b). The following set of noise parameters remains constant.

Parameter	Unit	Value
Gyroscope Bias Correlation Time	s	1000
Accelerometer Bias Correlation Time	s	300

In conclusion, the modified pipeline can tolerate IMU noise factor of up to 18.

5) Level of LiDAR TOF Distance Measurement Noise Tolerable: Velodyne's VLP-16 has a Gaussian distance measurement noise profile, with 0 mean, .005-.008 standard deviation [12]. We simulate increasing Gaussian distance measurement noise versus Odometry error.

The two angular commands are sampled from 1-d Gaussian distributions with standard deviation of 3 degrees, at 10 Hz. We use our proposed Motion Compensated LIO-SAM here.

Additionally, we add Gaussian distance measurement noise of 0 mean and varying standard deviation, ranging from 0.02-0.08, which is about 4-10 times of distance noise from a commercially available VLP-16 LiDAR. The result can be seen in Fig. 18-c).

6) Level of actuation delay tolerable under noisy condition: Our modification to LIO-SAM requires timely reporting of actuator joint/MEMS position. Actuation delay can therefore impact the odometry accurary.

Here we evalute motion-compensated LIO-SAM's odometry accuracy with increasing actuation delay.

Different than III-B, to increase realism, we add the above mentioned mirror control noise, IMU noise, LiDAR distance measurement noise.

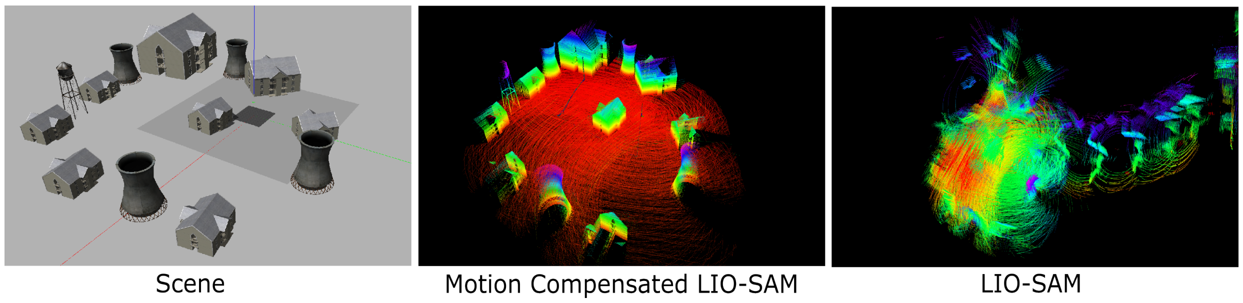


Fig. 16: Illustration of a) our simulator environment, b) mapping results on LIO-SAM with our motion compensation c) mapping results on stock LIO-SAM. It is worth noting that the pointcloud generated from the simulation has rotation with respect to the frame with world rotation $\boldsymbol{R}_{control}\boldsymbol{R}^{w}_{robot}$, which breaks any traditional visual SLAM's assumption, see section VI. It has significant frame-to-frame FOV variations (See section VI-C), which is difficult for any un-compensated, traditional SLAM to handle.

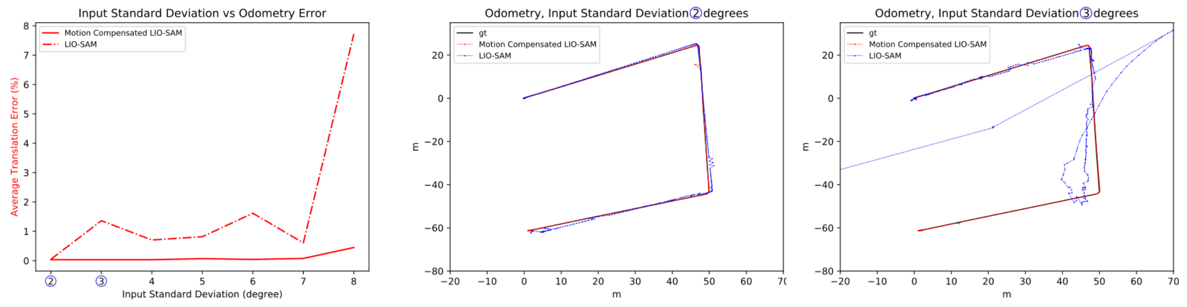


Fig. 17: a) Mirror control magnitude and odometry error, Our Motion Compensated LIO-SAM vs LIO-SAM . As mirror control magnitudes increase, the unmodified LIO-SAM fails completely. b) At 2 degrees standard deviation, Our Motion Compensated LIO-SAM outperforms LIO-SAM c) At 3 degrees standard deviation, degree threshold and beyond, Our Motion Compensatedd LIO-SAM perform normally while the stock LIO-SAM completely fails

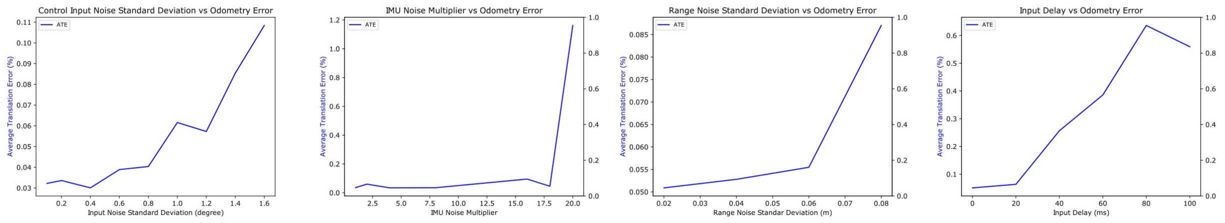


Fig. 18: a) Mirror control noise and odometry error. As the mirror control noise increases, the odometry error also increases. Our pipeline fails after noise standard deviation exceeds 1.6 degree. b) Imu noise and odometry error. As IMU noise factor increase, odometry error also increases. SLAM failure in our pipeline occurs at factor 20. c) As range measurement noise increase, odometry error also increases d)As input delay increases, odometry error also increase. Our system can tolerate input delay of 30ms.

The mirror input control Gaussian noise is set at 0 mean and 0.3 degree standard deviation.

The IMU noise is set at 3X of base IMU noise level mentioned in VI-C4.

The LiDAR distance Gaussian measurement noise is set at 0 mean and 0.008 degrees standard deviation, similar to that of VLP-16.

The two angular commands are sampled from 1-d Gaussian distributions with standard deviation of 3 degrees, at 10 Hz. We use our proposed Motion Compensated LIO-SAM here. The result can be seen in Fig. 18-d).

VII. Limitations and Conclusions

We have designed an adaptive lightweight LiDAR capable of reorienting itself. We have demonstrated the benefits of such a LiDAR in simulation as well as experiment. We have demonstrated in experiment image stabilization in hardware using an onboard IMU. We have also demonstrated viewing an object of interest using this LiDAR through external robot pose feedback. Please see the supplementary material of this paper for some MEMS-related details, including analysis of robot motion shock on the MEMS as well as preliminary point cloud stitching. We also explain how such a sensor can reduce sensing uncertainty. Finally, our accompanying video shows our experiments in action.

We would also like to acknowledge limitations of our study.

- We have indirectly compared to software methods using compensation *delay*. This is because, compared to hardware-compensation, any software-compensation will add delay, and therefore delay is a fundamental metric for hardware-software comparison. For future work we will directly compared with software compensation methods.
- Our design requires the robot to connected to the heavier sensing components using a tether. This limits the fly range and the detection FoV of the system. Although removing the tether restriction is left to future work, we believe that our design is capable of advancing sensing in microrobots significantly, and will help our community in designing microrobots in the future.
- All our results (using IMU as well as Vicon motion capture) are indoor results. We hope to perform future experiments with outdoor effects such as wind.
- In our current system design, there are implementation bottlenecks that limit compensated bandwidth. These are caused by the MEMS mirror and by the signal processing. Tightly coupled on-board designs can reduce these.
- In our current system design, manufacturing and material constraints have limited current MEMS scanners' FoV and speed, making them more suitable for small-sized and lightweight LIDAR applications.

In conclusion, through simulation and a prototype implementation we realize our design shown in Fig 1. We have shown, in simulation on on real hardware experiments, that hardware-compensation using a MEMS mirror improves both reconstruction and mapping. In particular, microrobots which suffer from heavy vibration and motion jitter (such as flapping-wing MAVs [54]) can benefit greatly from the motion compensated MEMS mirror scanning LiDAR for stabilized scene capture. Finally, over the long term, we believe that our design methodology can decouple robot and sensor geometry greatly simplifying robot perception.

References

[1] Thiemo Alldieck et al. "Optical flow-based 3d human motion estimation from monocular video". In: *German Conference on Pattern Recognition*. Springer. 2017, pp. 347–360.

[2] Riccardo Antonello et al. "IMU-aided image stabilization and tracking in a HSM-driven camera positioning unit". In: *2013 IEEE International Symposium on Industrial Electronics*. IEEE. 2013, pp. 1–7.

[3] Moshe Ben-Ezra, Assaf Zomet, and Shree K Nayar. "Jitter camera: High resolution video from a low resolution detector". In: *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.* Vol. 2. IEEE. 2004, pp. II–II.

[4] Andreas Bircher et al. "Receding horizon" next-best-view" planner for 3d exploration". In: *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2016, pp. 1462–1468.

[5] Chao-Ho Chen et al. "Real-time video stabilization based on motion compensation". In: *2009 Fourth International Conference on Innovative Computing, Information and Control (ICICIC)*. IEEE. 2009, pp. 1495–1498.

[6] Gabriele Costante et al. "Perception-aware Path Planning". In: *IEEE Transactions on Robotics* (2016), Epub–ahead.

[7] John L Crassidis. "Sigma-point Kalman filtering for integrated GPS and inertial navigation". In: *IEEE Transactions on Aerospace and Electronic Systems* 42.2 (2006), pp. 750–756.

[8] Frank Dellaert. *Factor graphs and GTSAM: A hands-on introduction*. Tech. rep. Georgia Institute of Technology, 2012.

[9] Xinke Deng et al. "Feature-constrained active visual SLAM for mobile robot navigation". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 7233–7238.

[10] Paul Furgale, Joern Rehder, and Roland Siegwart. "Unified temporal and spatial calibration for multi-sensor systems". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 1280–1286.

[11] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the kitti vision benchmark suite". In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE. 2012, pp. 3354–3361.

[12] Craig L Glennie, Arpan Kusari, and Aldo Facchin. "Calibration and stability analysis of the VLP-16 laser scanner." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 40 (2016).

[13] Zan Gojcic et al. "The perfect match: 3d point cloud matching with smoothed densities". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 5545–5554.

[14] Heinrich Grüger et al. "3.1: MOEMS Laser Projector for Handheld Devices Featuring Motion Compensation". In: *SID Symposium Digest of Technical Papers*. Vol. 38. 1. Wiley Online Library. 2007, pp. 1–3.

[15] Richard Hartley et al. "Rotation averaging". In: *International journal of computer vision* 103.3 (2013), pp. 267–305.

[16] *Hawk Head Stabilization*. Jan. 2020. URL: https://www.youtube.com/watch?v=aqgewVCC0k0.

[17] Tomohiko Hayakawa et al. "Gain-compensated sinusoidal scanning of a galvanometer mirror in proportional-integral-differential control using the pre-emphasis technique for motion-blur compensation". In: *Applied Optics* 55.21 (2016), pp. 5640–5646.

[18] E Farrell Helbling, Sawyer B Fuller, and Robert J Wood. "Pitch and yaw control of a robotic insect using an onboard magnetometer". In: *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2014, pp. 5516–5522.

[19] Katie Lynn Hoffman. "Design and locomotion studies of a miniature centipede-inspired robot". PhD thesis. 2013.

[20] Stefan Hrabar, Peter Corke, and Volker Hilsenstein. "PTZ camera pose estimation by tracking a 3D target". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 240–247.

[21] Kota Ito et al. "System Design and Performance Characterization of a MEMS-Based Laser Scanning Time-of-Flight Sensor Based on a 256×64-pixel Single-Photon Imager". In: *IEEE Photonics Journal* 5.2 (2013), pp. 6800114–6800114.

[22] Kemiao Jia, Sagnik Pal, and Huikai Xie. "An electrothermal tip–tilt–piston micromirror based on folded dual S-shaped bimorphs". In: *Journal of Microelectromechanical systems* 18.5 (2009), pp. 1004–1015.

[23] Ruting Jia et al. "System performance of an inertially stabilized gimbal platform with friction, resonance, and vibration effects". In: *Journal of Nonlinear Dynamics* 2017 (2017).

[24] Abhishek Kasturi et al. "UAV-borne lidar with MEMS mirror-based scanning capability". In: *Laser Radar Technology and Applications XXI*. Vol. 9832. International Society for Optics and Photonics. 2016, p. 98320M.

[25] Katsumi Kimoto et al. "Development of small size 3D LIDAR". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 4620–4626.

[26] Rainer Kümmerle et al. "g 2 o: A general framework for graph optimization". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 3607–3613.

[27] GEN LI et al. "Tracking moving objects by using a pan-tilt-zoom camera". In: *ITC-CSCC: International Technical Conference on Circuits Systems, Computers and Communications*. 2009, pp. 1012–1015.

[28] Mengyuan Li et al. "Modelling and experimental verification of step response overshoot removal in electrothermally-actuated mems mirrors". In: *Micromachines* 8.10 (2017), p. 289.

[29] Quanchao Li et al. "Nonorthogonal Aerial Optoelectronic Platform Based on Triaxial and Control Method Designed for Image Sensors". In: *Sensors* 20.1 (2020), p. 10.

[30] Ming Jun Ma et al. "IEEE standard specification format guide and test procedure for single-axis interferometric fiber optic gyros". In: (1998).

[31] Ievgeniia Maksymova et al. "Detection and Compensation of Periodic Jitters of Oscillating MEMS Mirrors used in Automotive Driving Assistance Systems". In: *2019 IEEE Sensors Applications Symposium (SAS)*. IEEE. 2019, pp. 1–5.

[32] Veljko Milanović et al. "Closed-loop control of gimballess MEMS mirrors for increased bandwidth in LiDAR applications". In: *Laser Radar Technology and Applications XXII*. Vol. 10191. International Society for Optics and Photonics. 2017, 101910N.

[33] Yash Mulgaonkar, Gareth Cross, and Vijay Kumar. "Design of small, safe and robust quadrotor swarms". In: *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2015, pp. 2208–2215.

[34] Frank Neuhaus et al. "Mc2slam: Real-time inertial lidar odometry using two-scan motion compensation". In: *German Conference on Pattern Recognition*. Springer. 2018, pp. 60–72.

[35] Yue Pan et al. "MULLS: Versatile LiDAR SLAM via multi-metric linear least square". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 11633–11640.

[36] Tao Peng and Satyandra K Gupta. "Model and algorithms for point cloud construction using digital projection patterns". In: (2007).

[37] Ethan Phelps and Charles A Primmerman. "Blind Compensation of Angle Jitter for Satellite-Based Ground-Imaging Lidar". In: *IEEE Transactions on Geoscience and Remote Sensing* 58.2 (2019), pp. 1436–1449.

[38] Seyed Abbas Sadat et al. "Feature-rich path planning for robust navigation of MAVs with mono-SLAM". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 3870–3875.

[39] Artur Sagitov and Yuri Gerasimov. "Towards DJI phantom 4 realistic simulation with gimbal and RC controller in ROS/Gazebo environment". In: *2017 10th International Conference on Developments in eSystems Engineering (DeSE)*. IEEE. 2017, pp. 262–266.

[40] Shital Shah et al. "Airsim: High-fidelity visual and physical simulation for autonomous vehicles". In: *Field and service robotics*. Springer. 2018, pp. 621–635.

[41] Tixiao Shan et al. "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping". In: *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE. 2020, pp. 5135–5142.

[42] Jae Kyu Suhr et al. "Background compensation for pan-tilt-zoom cameras using 1-D feature matching and outlier rejection". In: *IEEE transactions on circuits and systems for video technology* 21.3 (2010), pp. 371–377.

[43] Takafumi Taketomi and Janne Heikkila. "Focal length change compensation for monocular slam". In: *2015 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2015, pp. 4982–4986.

[44] Takafumi Taketomi and Janne Heikkilä. "Zoom factor compensation for monocular SLAM". In: *2015 IEEE Virtual Reality (VR)*. IEEE. 2015, pp. 293–294.

[45] Takafumi Taketomi, Hideaki Uchiyama, and Sei Ikeda. "Visual SLAM algorithms: a survey from 2010 to 2016". In: *IPSJ Transactions on Computer Vision and Applications* 9.1 (2017), p. 16.

[46] Zaid Tasneem et al. "Adaptive fovea for scanning depth sensors". In: *The International Journal of Robotics Research* 39.7 (2020), pp. 837–855.

[47] Sebastian Thrun. "Probabilistic robotics". In: *Communications of the ACM* 45.3 (2002), pp. 52–57.

[48] Nikolas Trawny and Stergios I Roumeliotis. "Indirect Kalman filter for 3D attitude estimation". In: *University*

of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep 2 (2005), p. 2005.

[49] Robert K Tyson. "Performance assessment of MEMS adaptive optics in tactical airborne systems". In: *Adaptive Optics Systems and Technology*. Vol. 3762. International Society for Optics and Photonics. 1999, pp. 91–100.

[50] Dingkang Wang, Connor Watkins, and Huikai Xie. "MEMS Mirrors for LiDAR: A review". In: *Micromachines* 11.5 (2020), p. 456.

[51] Dingkang Wang et al. "A Large Aperture 2-Axis Electrothermal MEMS Mirror for Compact 3D LiDAR". In: *2019 International Conference on Optical MEMS and Nanophotonics (OMN)*. IEEE. 2019, pp. 180–181.

[52] Dingkang Wang et al. "A low-voltage, low-current, digital-driven MEMS mirror for low-power LiDAR". In: *IEEE Sensors Letters* 4.8 (2020), pp. 1–4.

[53] Dingkang Wang et al. "An ultra-fast electrothermal micromirror with bimorph actuators made of copper/tungsten". In: *2017 International Conference on Optical MEMS and Nanophotonics (OMN)*. IEEE. 2017, pp. 1–2.

[54] Robert Wood, Radhika Nagpal, and Gu-Yeon Wei. "flight of the robobees". In: *Scientific American* 308.3 (2013), pp. 60–65. ISSN: 00368733, 19467087. URL: http://www.jstor.org/stable/26018027.

[55] Wei Xu et al. "Fast-lio2: Fast direct lidar-inertial odometry". In: *IEEE Transactions on Robotics* (2022).

[56] Dong Xue et al. "Computational simulation and free flight validation of body vibration of flapping-wing MAV in forward flight". In: *Aerospace Science and Technology* 95 (2019), p. 105491.

[57] Guang-Zhong Yang et al. "The grand challenges of Science Robotics". In: *Science robotics* 3.14 (2018), eaar7650.

[58] Heng Yang, Jingnan Shi, and Luca Carlone. "Teaser: Fast and certifiable point cloud registration". In: *IEEE Transactions on Robotics* 37.2 (2020), pp. 314–333.

[59] Haoyang Ye, Yuying Chen, and Ming Liu. "Tightly coupled 3d lidar inertial odometry and mapping". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 3144–3150.

[60] Ji Zhang and Sanjiv Singh. "LOAM: Lidar Odometry and Mapping in Real-time." In: *Robotics: Science and Systems*. Vol. 2. 9. 2014.

[61] Xiaoyang Zhang, Liang Zhou, and Huikai Xie. "A fast, large-stroke electrothermal MEMS mirror based on Cu/W bimorph". In: *Micromachines* 6.12 (2015), pp. 1876–1889.

[62] Zichao Zhang and Davide Scaramuzza. "Perception-aware receding horizon navigation for MAVs". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 2534–2541.

SUPPLEMENTARY MATERIALS

Motion Compensation Controller

The IMU model is provided by the manufacturer, and it is usually simplified as a passive low-pass filter model. For example, the IMU(VN-100) has a bandwidth of $f_2 = 150\text{Hz}$ with a simplified transfer function of

$$H_2(s) = \frac{\frac{f_2}{2\pi}}{s + \frac{f_2}{2\pi}} \tag{22}$$

The compensating components $H_g(s)$ and the high pass filter $H_{pf}(s)$ improve the usable range of the motion compensation. We assume that the relative low frequency spectrum motion is the desired motion, and should not be compensated. It can be achieved by tuning the bandwidth of the high pass filter $H_{pf}(s)$. In this work, a passive high pass filter with a bandwidth of f_h is selected,

$$H_{pf}(s) = \frac{s}{s + 2\pi f_h} \tag{23}$$

To improve the response bandwidth at higher frequency and suppress the noise, $H_g(s)$ is added. The $H_g(s)$ is defined with the inverse of the transfer function of the IMU and the MEMS mirror,

$$H_g(s) = \frac{B(s)}{H_1(s)\dot{H}_2(s)} \tag{24}$$

where $B(s)$ ensure that the order of the numerator is not higher than the order of the denominator. A fourth order Butter-worth filter with a bandwidth of $f_B = 200\text{Hz}$ is used as the low pass filter. The 200 Hz bandwidth is selected to improve the speed while it can still suppress the resonance of the MEMS mirror, which is,

$$B(s) = \frac{1}{\frac{s}{2\pi f_B}^4 + 2.613\frac{s}{2\pi f_B}^3 + 3.414\frac{s}{2\pi f_B}^2 + 2.613\frac{s}{2\pi f_B} + 1} \tag{25}$$

which implies,

$$H_g(s) = \frac{s^4 + 512s^3 + 1.081e08s^2 + 1.486e11s + 4.457e13}{4.457e13} \tag{26}$$

To implement the transfer function in a microcontroller, the continuous-time transfer function $H_g(s)$ is translated to a discrete-time transfer function $G[z]$ with a sampling time of 0.0025s,

$$G[z] = \frac{Y[z]}{X[z]} = \frac{0.843z^4 + 0.93z^3 - 0.32z^2 - 0.045z + 0.0064}{z^4 + 0.32z^3 + 0.11z^2 - 0.017z + 0.0021} \tag{27}$$

$X[z]$ in the input (IMU measurement) and $Y[z]$ is the output (MEMS scanning angle) in the Z domain. Converting the discrete-time transfer function $G[z]$ to the time domain,

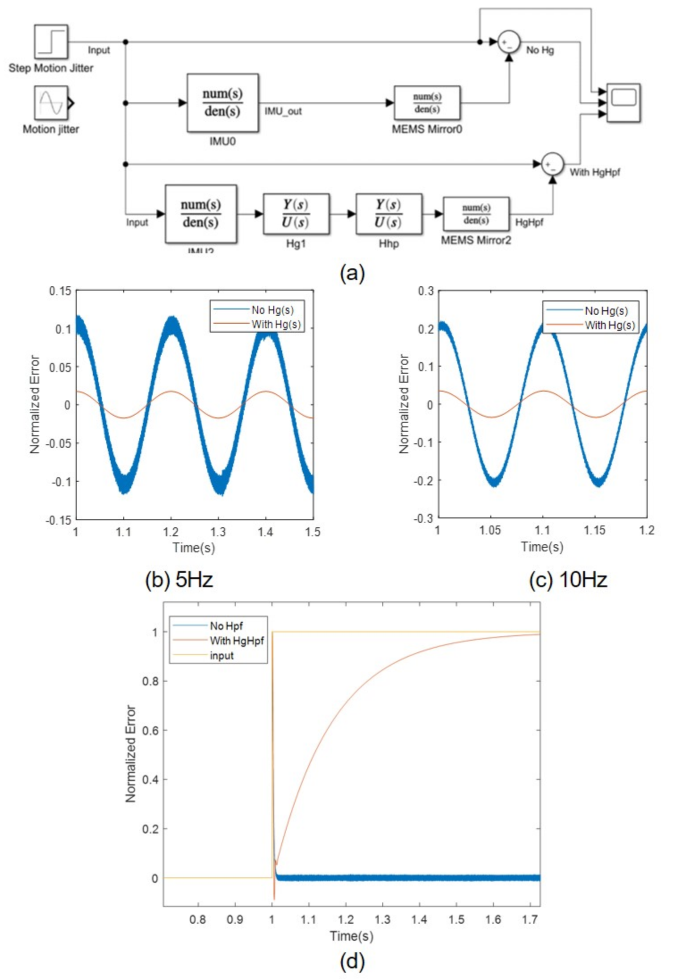


Fig. 19: (a) The Simulink simulation setup of the motion compensation. The input frequency is 5 Hz (b) and 10 Hz (c). (d) The effect of $H_{pf}(s)$ under step response.

$$y[n] = (0.843x[n]+0.93x[n-1]-0.32x[n-2]-0.045x[n-3] + 0.0064x[n-4])- (0.32y[n-1]+0.11y[n-2]-0.017y[n-3]+0.0021y[n-4]). \quad (28)$$

Simulink is used to simulate the performance of the controller and tune the parameters. The Simulink setup is shown in Fig. 19. The input motion jitter is a sinusoid wave. Fig. 19(b) and (c) shows the motion compensation residues with the input motion jitter frequency of 5Hz and 10 Hz. $H_g(s)$ can effectively reduce the residue. Note that the compensation residue is expected to be better than real-world experiments because of the digitization.

Motion Compensation Controller Design

Those algorithms provide motion-compensated MEMS scanning in steady-state. However, as both the IMU and the MEMS mirror have limited response bandwidths, the residue and the speed of the motion compensation may get worse as the frequency of motion jitter increases. Also, the MEMS mirror has a limited scanning range, so the range of motion compensation is also limited. Since the motion compensation system is an open-loop system, a compensating stage $H_g(s)$ can be added to the microcontroller to increase the performance of the system.

A simplified block diagram of the open-loop motion compensation system is shown in Figure 20, where $H_1(s)$ denotes the MEMS mirror tip-tilt model given in Eq. 13, $H_2(s)$ is the IMU model, $H_g(s)$ is the compensating components, $H_{pf}(s)$ is an optional high-pass filter and F is the model for the MEMS mirror driver.

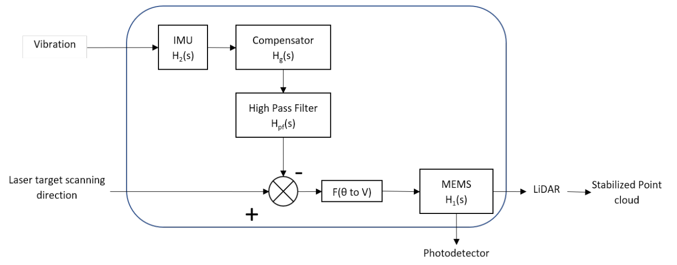


Fig. 20: The block diagram of the open-loop motion compensation system.

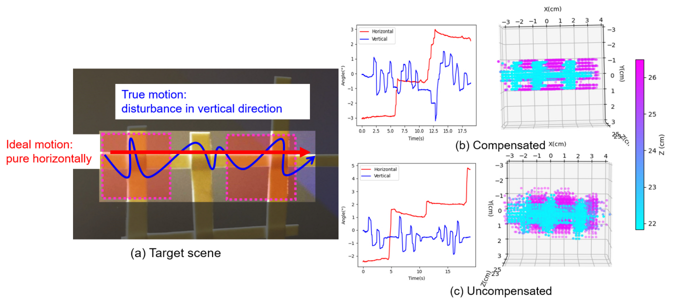


Fig. 21: Compensated point cloud stitching and experiment and its results. (a) The target scene placed at 22 cm from the LiDAR. The red line is the ideal motion, the blue line qualitatively depicts the type of experiment we performed, with vertical disturbances only. The graphs show the actual, measured disturbances. (b) With compensated scanning, the recorded motion and the generated point cloud show sharper depth edges than that with (c) with uncompensated scanning.

Stitching experiments with translation

Here, we performed point-cloud stitching as that sensor moves along an object. The target scanning area is along the horizontal paper-cut figure shown in the highlighted area in the Figure 21 (a). The blue line is an example of the true motion with disturbance only existing in the vertical direction. As the LiDAR rotates from in the horizontal direction from left to right, it is expected to collect the best point cloud covering the highlight area of the object only. The result of compensated scanning and uncompensated scanning are shown in Figures 21 (b) and (c) on the right side, with their measured motions on the left side. Please find other supplementary materials in the accompanying video.