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

Yuyang Chen, Dingkang Wang, Lenworth Thomas, Karthik Dantu, *Senior Member, IEEE*, and Sanjeev J. Koppal, *Senior Member, IEEE*

***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 the robot pose is changed to reorient the sensor to areas of interest for perception. Furthermore, egomotion, such as jitter, and external effects, such as wind and others, affect perception requiring additional efforts in software such as image stabilization. This effect is particularly pronounced in microair vehicles and microrobots that 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 mirror light detection and ranging (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 unmanned aerial vehicle (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 inertial measurement unit (IMU) and external odometry feedback in hardware.**

***Index Terms*—Active perception, motion compensation, robot sensing systems, robot vision systems, simultaneous localization and mapping, vision sensors.**

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 [1], [2], [3], [4] and active vision [5], [6], [7], [8] 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 [9], crawling and walking robots [10], and other microair vehicles [11]. 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 piezoelectric actuation [12]. Environmental factors, such as wind, affect microrobots to a greater extent than larger robots. There might be aerodynamic instability due to ornithopter-based shock absorption [13]. 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 [14] and light detection and ranging (LiDARs) [15], this is often difficult to perform in real-time onboard due to the computational, energy, and latency constraints on the robot mentioned earlier. Without proper motion compensation for miniature devices, we will not be able to unlock the full potential of what is one of the 10 grand challenges in robotics [16].

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 article 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 —1) onboard IMU, and 2) external feedback of the 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 [17]. In Fig. 1, we show frames $V(t)$ from a video of a hawk (*Buteo jamaicensis*) being moved by a human trainer [18].

Manuscript received 31 July 2023; revised 24 December 2023; accepted 29 January 2024. Date of publication 29 February 2024; date of current version 21 March 2024. This paper was recommended for publication by Associate Editor O. Stasse and Editor S. Behnke upon evaluation of the reviewers' comments. The work of Y. Chen and K. Dantu was supported under Grant NSF-1514395 and Grant NSF-1846320. The work of D. Wang, L. Thomas, and S. J. Koppal was supported in part by the ONR under Grant N00014-18-1-2663 and Grant N00014-23-1-2429 and in part by the NSF under Grant 1942444. *(Yuyang Chen and Dingkang Wang contributed equally to this work.) (Corresponding author: Yuyang Chen.)*

Yuyang Chen and Karthik Dantu are with the Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY 14260 USA (e-mail: yuyangch@buffalo.edu; kdantu@buffalo.edu).

Dingkang Wang is with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32603 USA (e-mail: noplaxochia@ufl.edu).

Lenworth Thomas is with the Department of Mechanical and Aerospace Engineering, University of Florida, Gainesville, FL 32603 USA (e-mail: lenworth.thomas@ufl.edu).

Sanjeev J. Koppal is with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32603 USA, and also with the Amazon Robotics, Reading, MA 01864 USA (e-mail: sjkoppal@ece.ufl.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TRO.2024.3371885, provided by the authors.

Digital Object Identifier 10.1109/TRO.2024.3371885

Fig. 1. Biological motion compensation. The position and the angle of the head of the hawk remain stable despite body motion to provide the hawk with a stabilized vision (https://www.youtube.com/watch?v=aqgewVCC0k0). (a) Start of video. (b) Midway. (c) End. (d) Average of all images in the video.

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 the sensor pose could have many uses in robotics, particularly in image alignment during motion such as in simultaneous localization and mapping (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 article, 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 that can react in ones of hertz 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.

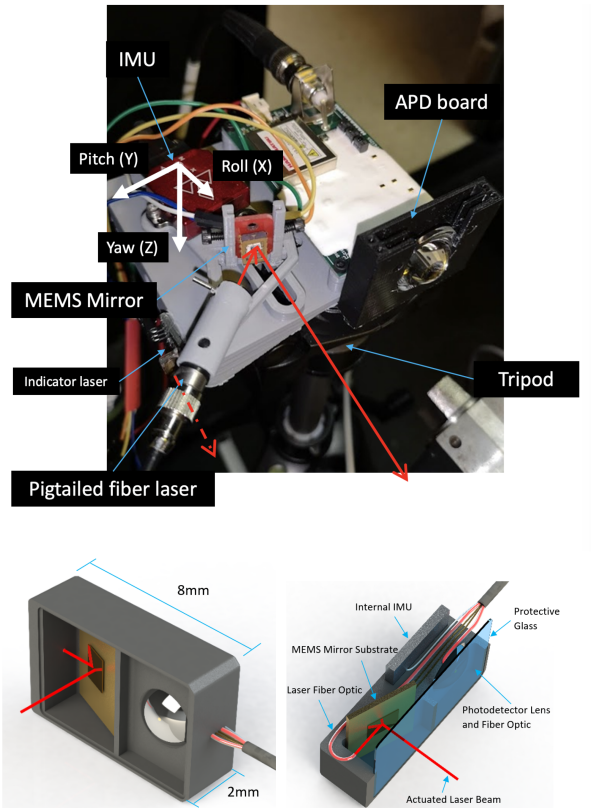


Fig. 2. Our design is given with the prototype motion-compensated LiDAR (up), and we also prepared a design for future work to integrate this onto smaller platforms.

Our designs break through these past difficulties by exploiting recently available microelectromechanical (MEMS) and optomechanical components for changing camera parameters. Opto-MEMS components are famously fast (many kilohertz), 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 of 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 a 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 [19], [20], [21]. Our algorithmic and system design contributions beyond this are as follows.

1) We present the design of a novel LiDAR sensor adopting a MEMS mirror similar to this LiDAR MEMS scanner [22]. This design enables wide nonresonant 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. Fig. 2 shows the design of our sensor.

2) 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 Section III.
3) We present the compensation control algorithms for our LiDAR. We further characterize the performance of compensation control through experiments and simulations in Section IV.
4) We show a UAV flight with a proof-of-concept hardware prototype combining external feedback with the MEMS mirror for egomotion compensation. We enable a UAV flight by tethering the MEMS modulator to the other heavy necessary components, such as laser, photodetector, optics, driver circuit, and 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. See the accompanying video (Supplementary material) for proper visualization and Fig. 12.
5) We provide an implementation of the sensor in the Gazebo simulator. Using this simulated sensor, we propose a framework to adapt a modern LiDAR SLAM pipeline to incorporate motion compensation. We adapt a modern LiDAR SLAM pipeline tightly coupled lidar inertial odometry via smoothing and mapping (LIO-SAM) [23] 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 [19], [20], [21]. For instance, Kasturi et al. [20] demonstrated a UAV-borne LiDAR with an electrostatic MEMS mirror scanner that could fit into a small volume of 70 mm × 60 mm × 60 mm and weighed about only 50 g. Kimoto et al. [21] developed a LiDAR with an electromagnetic resonant MEMS mirror for robotic vehicles.

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 uses iterative closest point (ICP) [1] and feature matching [2] to find the translation and rotation of successive point clouds. Software-based compensation for robotics motion has been studied in great detail in SLAM algorithms [3] or expectation–maximization (EM) methods [4]. Software-based motion compensation has a relatively high computation barrier for microrobotics and may degrade if the point cloud has a 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 2-D MEMS-based LiDAR), which can correct the rolling shutter effect and improve the motion response range. Second, the motion compensation algorithm is very simple and can be implemented on a low-power microcontroller or FPGA. Third, even if the hardware motion compensation still has some errors, it provides a better initialization for the following software compensation.

These ideas are closer to how PTZ cameras track dynamic objects [24], [25] and assist with background subtraction [26]. However, compared to these approaches, we can tackle egomotion of much higher frequencies, which is a unique challenge of microrobots. We compensate signals much closer to those seen in adaptive optics and control for camera shakes [27], [28], [29]. In addition, our system is on a free-moving robot, rather than a fixed viewpoint.

Motorized gimbals: Compared to motorized image stabilization systems [30], MEMS mirrors not only have a smaller size and lighter weight, but their frequency response bandwidths are better than the bulky and heavy camera stabilizers. The MEMS mirror response time can be less than 10 ms or even less than 1 ms. The servomotor of the camera stabilizer has a bandwidth of less than 30 Hz because they are bulky and have a heavy load [31], [32]. 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 [33], where a handshake 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, the authors in [34] and [35] change the zoom and focal lengths of cameras for SLAM. Our prior work [19] changed zoom in LiDARs. In this work, we compensate using small mirrors, utilizing a rich tradition of compensation in device characterization [36] and improving SNR [37]. 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 in [38], successive LiDAR SLAM designs largely follow a LiDAR SLAM architecture similar to Fig. 14, where the front end consists of deskew and feature extraction stages while the back end usually consists of ICP and pose graph optimization packages such as g2o [39] or GTSAM [40] that globally optimizes the odometry information as estimated by LiDAR visual odometry. Successive efforts moved toward improvement in the following subareas:

1) tightly coupling LiDAR and IMU [41];
2) updating the backend PGO optimizer [23];
3) updating the back end's ICP [42];

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

4) updating the front-end's point-cloud data structure to do away with ICPs feature dependence [43].

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 (FoV) can be changed only by changing the pose of the robot. The authors in[5], [8], and [44] improve SLAM accuracy by actively changing the robot trajectory to improve the FoV. 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 LiDAR depth measurements in the direction corresponding to 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 structure-from-motion (SFM) framework with the LiDAR projection matrix $\mathbf{P}$ and the robot's rotation $\mathbf{R}$ and translation t

$$\mathbf{P} = \begin{bmatrix} \mathbf{R} & t \\ 0 & 1 \end{bmatrix}. \tag{1}$$

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 3-D 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:

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

This optimization *usually happens in software, after LiDAR and IMU measurements* [45]. 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 article, 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 postcapture translation estimation

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

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

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 [46] running on Unreal Engine 4 for realistic perception and visualization. We tested two scenarios—1) a scene with geometric objects, called the *blocks scene* shown in Fig. 3(a), and 2) an outdoor scene with a bridge and mountains, called the *mountains scene* shown in Fig. 3(e). In both scenes, the LiDAR is mounted on a prototype quadrotor UAV. We run lidar odometry and mapping (LOAM) [38], 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 gimbal, active compensation of a PTZ camera or MEMS-based hardware compensation like our system. The differences between these methods are along two dimensions—1) latency of compensation, called *compensation delay* from now on, and 2) 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 preprocessing 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, fixed-length queue, and reorients the mounted LiDAR toward 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 Section IV-B4.

The method is also known as quaternion L_2-mean [47]. Given the relatively short duration of the sliding window and the relatively small range of rotation that is covered during simulation 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° horizontal FoV, 30° vertical FoV, and with 150 000-Hz sample rate, akin to commercially available LiDARs.

To quantify performance, we calculate the *odometry error*, the difference between the ground truth UAV positions, and those positions estimated by LOAM. Fig. 4 shows the results from our simulations for blocks scene and mountains scene. We set the compensation rate to five different values—uncompensated, 5 Hz, 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 [see Fig. 4(a)]. 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 0 ms or 30 ms [see Fig. 4(b)].

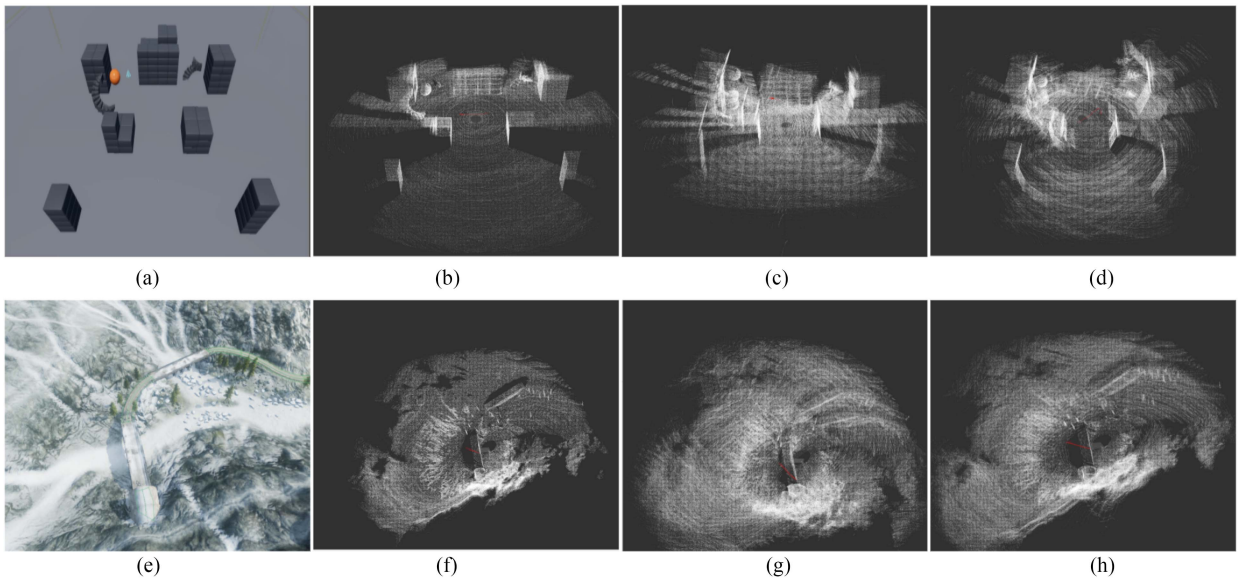


Fig. 3. (a) Representative simulation scenario—Blocks scene. (b) Mapping the blocks scene with compensation at 55 Hz and no delay. (c) Mapping the blocks scene without compensation. (d) Mapping the blocks scene with compensation at 55 Hz and delay of 150 ms. (e) Mountains scene. (f) Mountains scene simulation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

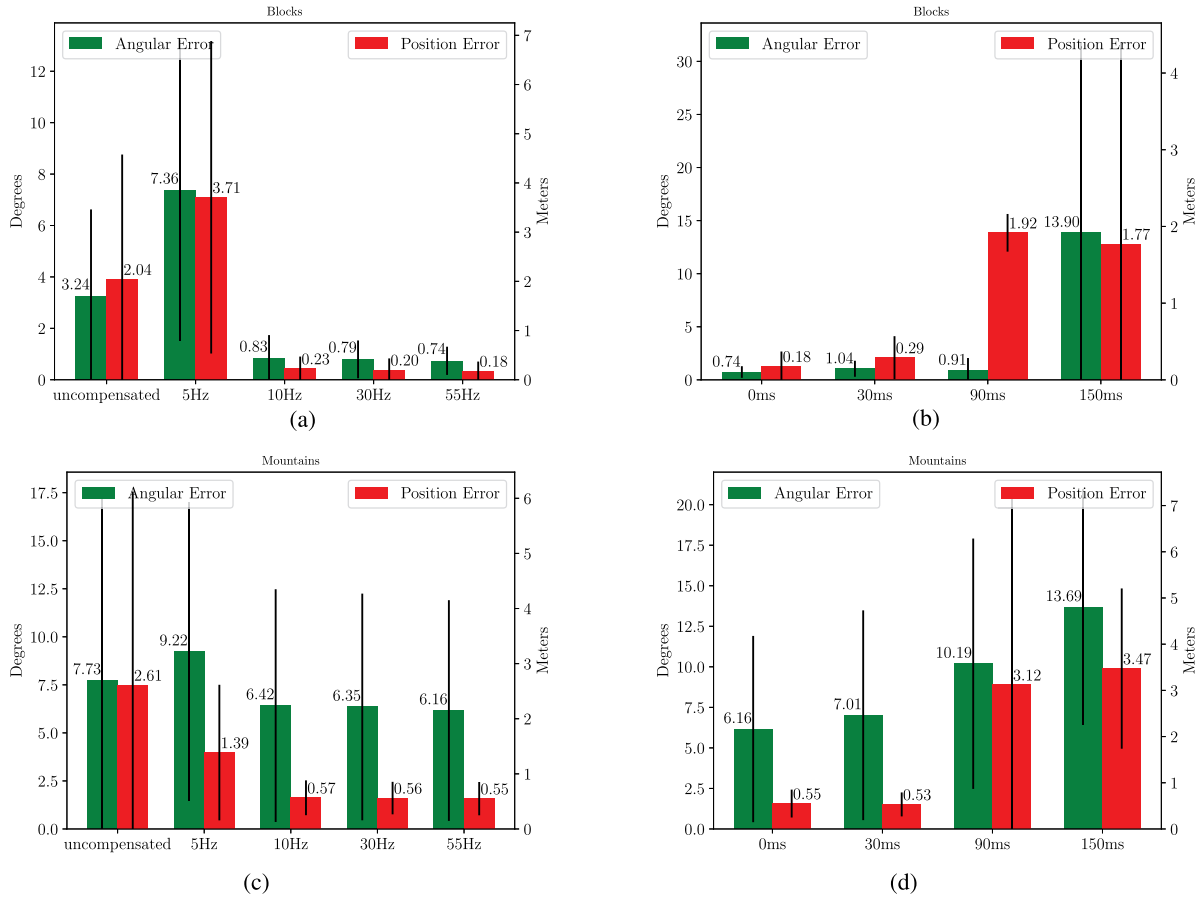


Fig. 4. UAV odometry error while varying compensation rate and compensation delay in two scenes. (a) Odometry error versus compensation rate (blocks scene). (b) Odometry error versus compensation delay (blocks scene). 55-Hz compensation. (c) Odometry error versus compensation rate (mountains scene). (d) Odometry error versus compensation delay (mountains scene). 55-Hz compensation.

For larger compensation delays such as 90 ms 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 microrobots. To enable both the LiDAR scanning and compensated scanning at a high rate, it is important to understand the characterization of the MEMS scanner.

A. 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 with 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 FoV, which leads to heavy ringing effects and overshoot with sudden changes of direction [48], [49]. A suitable MEMS mirror for motion-compensated scanning is expected to have a wide nonresonant scanning angle, smooth and fast step responses, can operate under common robotics vibration, and can survive the shock. To achieve this goal, we adopt a popular electrothermal bimorph-actuated MEMS mirror design [50], [51] to build this MEMS mirror. The employed MEMS mirror is fabricated with Al/SiO_2-based inverted-series-connected bimorph actuation structure reported in [50]. This type of MEMS mirror has the advantages of a simple and mature fabrication process [52], [53], wide nonresonant 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 [20], [22], [54].

B. Compensation Algorithm

In the previous sections, we saw the advantages of MEMS-mirror-based compensation and the feasibility of 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 reflects a single ray of light toward a point in the spherical coordinate $\{\alpha, \beta, r\}$. $\{\alpha, \beta\}$ are the two angular control inputs to the mirror to achieve such a target. We will first establish the local (robot) and global (world) frames, then introduce known helper conversion from spherical to Cartesian coordinates, and, finally, get into the details of compensation.

1) Preliminaries:

a) Coordinate system: Our LiDAR can compensate for rotation but it cannot compensate for translation. Change to So all discussions herein on in drops the translation from $SE(3)$ and will only focus on $SO(3)$. Let the robot have rotation $\boldsymbol{R}^w_{\text{robot}} \in SO(3)$ relative to the world frame. Here, the frame of the unmoving base of the LiDAR sensor has identity rotation $\boldsymbol{R}^w_{\text{base}} \in SO(3)$ 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 a normal Cartesian coordinate. Points in the spherical coordinate $\{\alpha, \beta, r\}$ can be converted to a Cartesian coordinate via known equations

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

and vice versa

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

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

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 Fig. 5(a). 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 applications in this work. Let $\boldsymbol{R}^w_{\text{desired}} \in SO(3)$ be the desired rotation target in the world frame. $\boldsymbol{R}^w_{\text{desired}}$ 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 an application in Section III-B. We will touch on the exact details later in Section IV-B-4. Other possibilities include aiming toward a specific world frame target $t \in \mathbb{R}^3$, which we will touch on later, in Section IV-B-5.

2) General Rotation Compensation: Given robot world frame rotation $\boldsymbol{R}^w_{\text{robot}}$, user-desired sensor rotation $\boldsymbol{R}^w_{\text{desired}}$ and a set of spatial scanning, spherical, sensor input coordinates $\{\alpha_i, \beta_i, r_i\}$, we need to find the adjusted sensor input

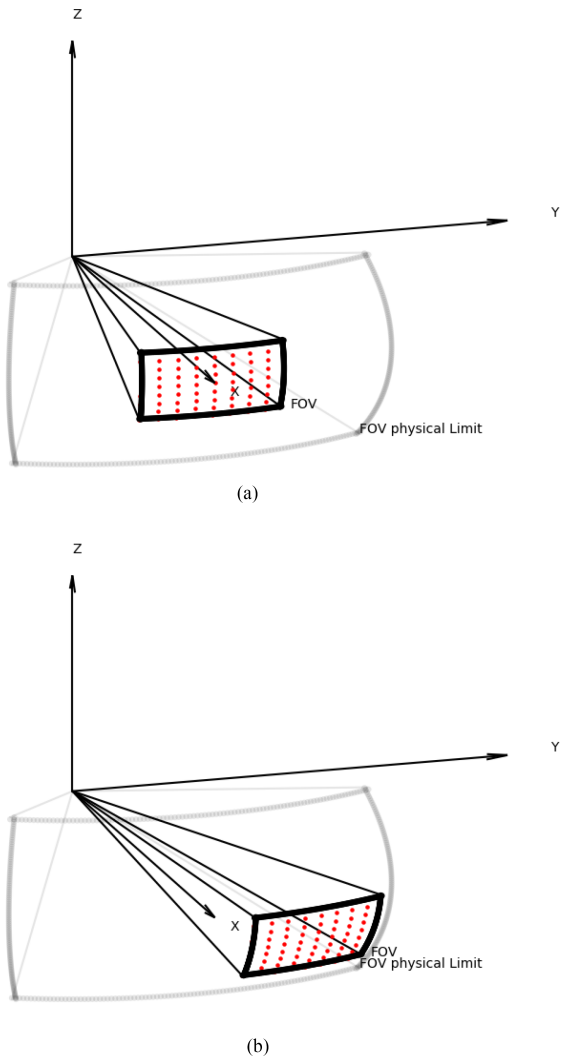


Fig. 5. (a) Depicting spatial scanning grid in the sensor's base frame. In the real sensor, the resolution of the scanning grid is higher at 20×20. The input rotation here is zero. In other words, $\boldsymbol{R}_{\text{control}} = \boldsymbol{I}$. (b) Depicting spatial scanning grid in the sensor's base frame, the input rotation is none-zero here.

coordinates $\{\alpha_i^*, \beta_i^*, r_i^*\}$, in order to achieve user-desired sensor rotation $\boldsymbol{R}^w_{\text{desired}}$. We will outline the calculations step by step.

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

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

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

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

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

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

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

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

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

d) Step4: Finally, we can translate the rotated points $p_{\text{desired-cartesian}_i}$ back to the spherical coordinate $p_{\text{desired-spherical}_i}$ via (5) for point i's rotation control input to the sensor. Now 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 such as Velodyne, the entire set of p_i can be viewed as being projected as a group and correlated 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 reprojection since our LiDAR can achieve this 2-axis-only compensation.

3) Special Case: 2-Axis-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 Section III-B and design motion-compensated LiDAR SLAM in Section VI. Furthermore, when it comes to the target aiming Section IV-B-5, 2-axis rotation is often preferred. Our sensor can perform such compensation as well.

In this section, 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 Section IV-B-2-b, we define $\boldsymbol{R}_{\text{control}}$ as the difference between the sensor orientation and the robot's orientation. Since $\boldsymbol{R}_{\text{control}} \in SO(3)$, it requires at least 3-axis rotation control to achieve.

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

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

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

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

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

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

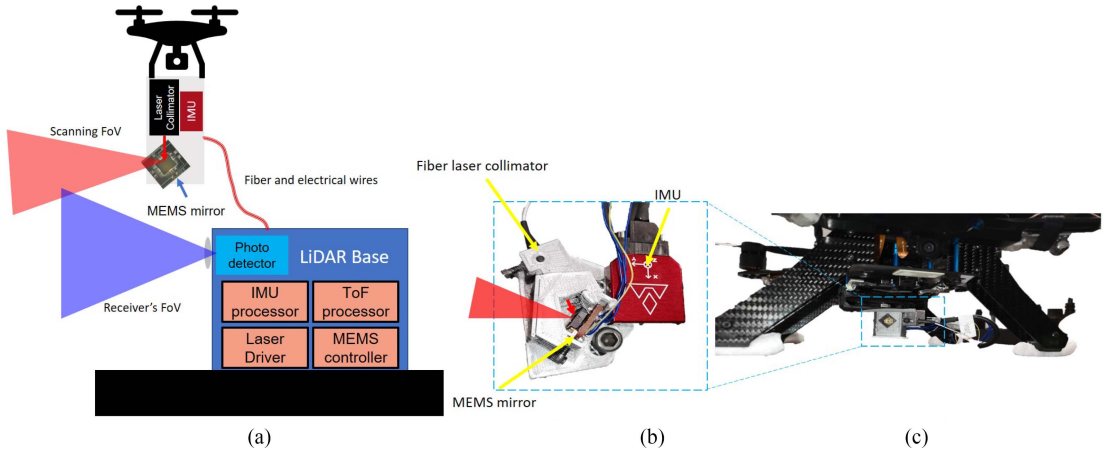


Fig. 6. Movable LiDAR MEMS scanner head, which includes the MEMS mirror, an IMU, and a fiber laser collimator. (a) Top view. (b) LiDAR scanner head mounted to the bottom of the UAV.

$\boldsymbol{e}_{\text{rotated}}$ is now the new principle axis that our sensor should target. Note that, $\boldsymbol{e}_{\text{rotated}}$ is closely related to the ray vector from robot to target in the aiming application, more on this later in Section IV-B-5.

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

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

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

Furthermore, 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 a relatively slow rotating sensor world frame FoV in many SLAM-related applications. We have demonstrated such benefit in Section III-B. Here, we go into details of how it is achieved with our sensor.

a) Quaternion $L_2 - mean$: Supposedly, $q_1 \ldots ..q_n$ is the world frame quaternions stored 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 [47]

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

q_{avg} can be converted into a rotation matrix $\boldsymbol{R}^w_{\text{desired}}$. Along with the robot's current world frame rotation $\boldsymbol{R}^w_{\text{robot}}$, we can find the adjusted spherical coordinate control input to our sensors $\{\alpha^*_i, \beta^*_i, r^*_i\}$, according to Section IV-B-2.

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

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

outlines the ray direction, which we want to align our "principle axis," or the projection center point toward. Following a very similar process as to Section IV-B-3, we can find the controls.

1) *Step1:* We can simply translate Cartesian coordinate p_{aim} to spherical coordinates via (5) to find α, β.
2) *Step2:* Then compose $\boldsymbol{R}^*_{\text{control}}$ via (9) for the entire scanning grid. We can simply follow the same steps in Section IV-B-2, except we replace $\boldsymbol{R}_{\text{control}}$ in (7) with $\boldsymbol{R}^*_{\text{control}}$.

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-D 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 (see Fig. 6) 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 $5 - 7°$/h 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 time-of-flight (ToF) signals, sample the IMU signals, and control the MEMS mirror scanning direction. The data are sent to a PC for postprocessing and visualization.

Since our motivation was to use microrobots, our maximum detection distance is 4 m with an 80% albedo object and the minimal resolvable distance is 5 cm. The maximum ToF measurement rate is 400 points/s. 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. See the accompanying video for further clarification.

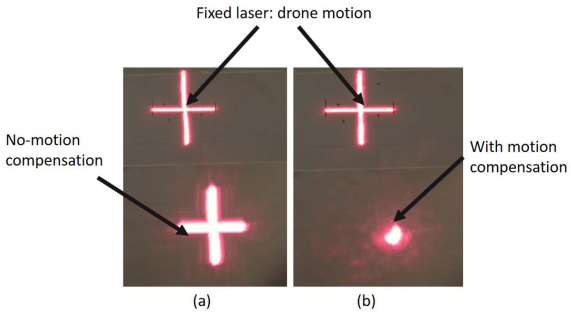


Fig. 7. We use a visible laser to compare the effect of the motion compensation of our sensor. The upper laser trace indicates UAV motion, and the lower laser trace indicates the compensated/uncompensated scanning laser reflected from the MEMS mirror. The compensated MEMS scanning (right) shows a much smaller laser trace area than the uncompensated MEMS scanning result.

TABLE I
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°

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 Fig. 6. The MEMS mirror's desired scanning angle is set to a single point on the target object ($0^\circ \times 0^\circ$) to make it easier to compare.

Here, the entire scanning grid $\{\alpha_i, \beta_i, 1\}$ consists of one single point only at the projection center. We use the general compensation outline in Section IV-B-2

The UAV together with the LiDAR scanner head is held with hand with a random rotational motion in the yaw/pitch direction. The upper laser trace comes from the laser rigidly connected to the UAV, which indicates the UAVs motion. The lower trace is reflected from the MEMS mirror, which shows the compensated/uncompensated scanning laser. The results are shown in Fig. 7. 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 are available in the Supplementary material.

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 FoV and the background is at 2.8 m, as shown in Fig. 8(a). The MEMS mirror performs a raster scanning pattern with an initial FoV of -3.5° to $+3.5^\circ$ in both axes to leave room for compensation. Each frame has 20 $\times$ 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 Fig. 8(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 pixels on either axis, caused by measurement noise.

Fig. 8(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 third image of Fig. 8(c). Multiple frames of the point cloud are stocked together and shown in the last column of Fig. 8. The object can still be identified in the compensated point cloud [see Fig. 8(f)] but becomes fuzzy caused by the motion jitter when not compensated [see Fig. 8(e)]. The videos of the real-time compensation point cloud results are available in the Supplementary material.

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. 9. The MEMS mirror motion compensation system is controlled by an Arduino Mega. The IMU sends the data to the Arduino at 400 Hz. The data are processed, and the compensator $H_g(s)$ 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 position-sensitive detector (PSD) 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 microstepper 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 30 to 500 ms. The motion compensation is tested in the pitch direction for smaller errors. The motor is placed horizontally to the ground.

Fig. 10 shows the motion compensation results from comparison under various motor speeds (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 in an ideal direction. The effect with and without the compensator term is also compared, as shown in Table I.

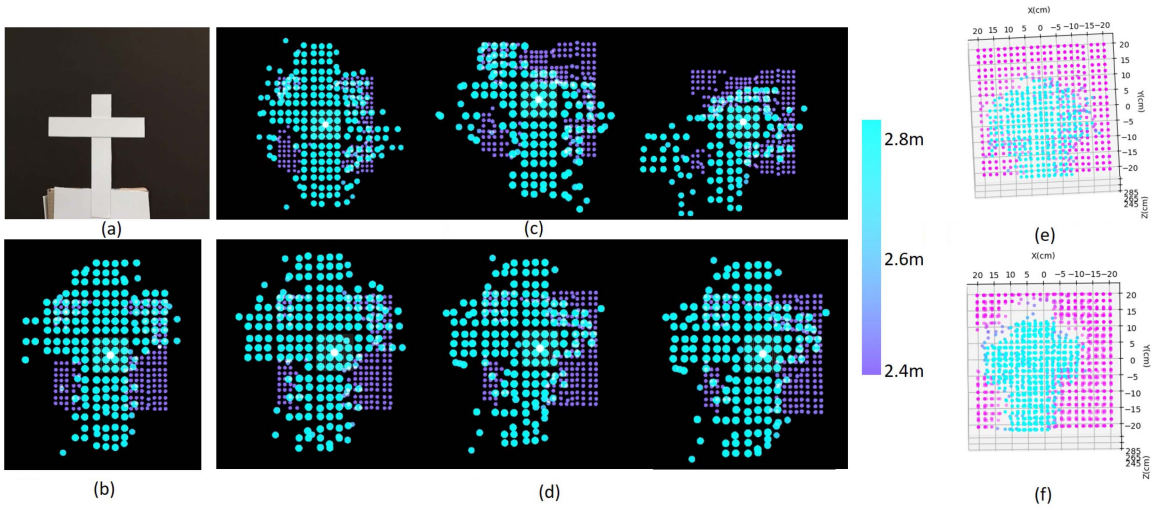


Fig. 8. Motion-compensated LiDAR point cloud result with handheld motion disturbance. (a) Target object "+" placed 2.4 m from the LiDAR, along with (b), its initial point cloud scan. (c) and (d) Uncompensated versus compensated scanning. The handheld rotation angle in (x, y) axes are $(-0.2°, +1.4°), (+1.0°, +1.4°), (-1.5°, +1.7°)$ and compensated angular shake range was $(-1.1°, +1.4°), (+1.2°, -0.5°), (-2.3°, +0.5°)$; see supplementary video. (d) and (e) Stacking of 5 frames of a point cloud of compensated and uncompensated results.

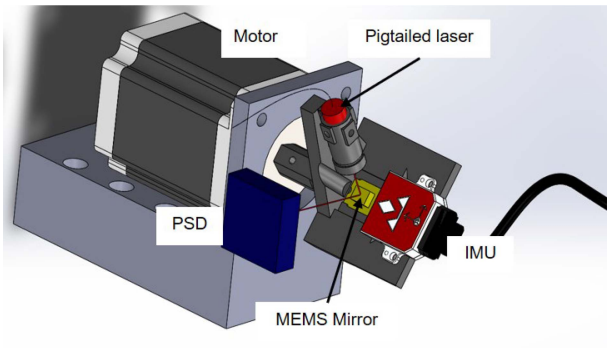


Fig. 9. Input disturbance testing platform with stepper motor.

E. MEMS, Step Response, and Robot Motion Shock

We now describe the characteristics of MEMS mirror itself, particularly when it comes to disturbances from robot motion; see Fig. 11(a). In 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 [see Fig. 11(b)]. The voltage to MEMS tilting angle response is approximately linear. The MEMS mirror can perform nonresonant arbitrary scanning or pointing according to the control signal. The cross-axis sensitivity is about 6% in both axes. In the microcontroller, the voltage and the MEMS scanning angle are approximated with a linear relationship with the cross-axis sensitivity taken into consideration. The maximum error caused by the nonlinearity is $0.3°$.

The step response is 5 ms [see Fig. 11(c)] 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. 11(d). The piston resonant mode is found at f_1=1.07 kHz 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 third-order system according to the work in [55]. 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, $\tau \approx t_r/2.2 = 2.3$ ms; ω_n is the natural resonant frequency of the mirror rotation, $\omega_n \approx 2\pi$ (1.65 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 (13).

Similar to the work in [22], the MEMS mirror is actuated by the pulse width modulation (PWM) signals with a voltage level of 0–5 V. The PWM signal can be generated by an Arduino microcontroller at 15 kHz and 8 b. 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

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

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

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

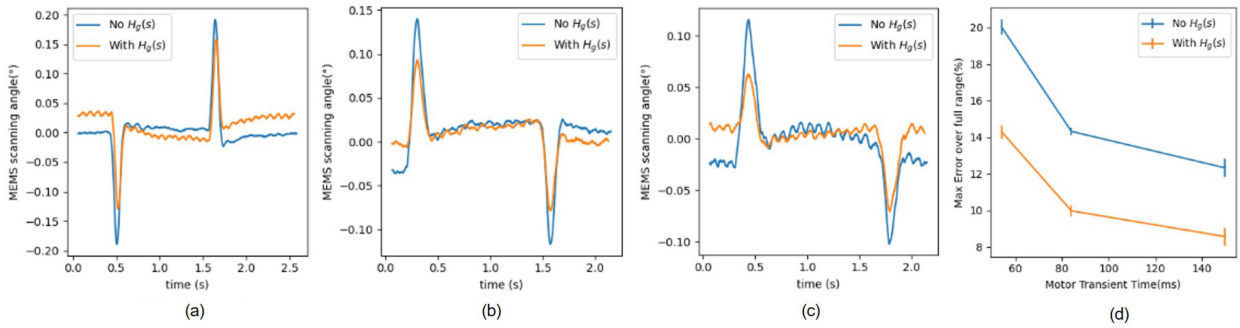


Fig. 10. Motion compensation results comparison under different motor speeds (t, motor transient time) and with and without the compensator $H_g(s)$. In these experiments, the mirror is commanded to aim at 0° while the disturbance of 1.8° is input to the LiDAR base from a stepper motor; see Fig. 9. Therefore, ideally, the MEMS scanning angle stays flat at 0° at all times. When the motor speed gets faster, the motion compensation errors become larger. $H_g(s)$ can effectively improve the compensation scanning. (a) t = 55 ms. (b) t = 85 ms. (c) t = 150 ms. (d) The Compensator $H_g(s)$.

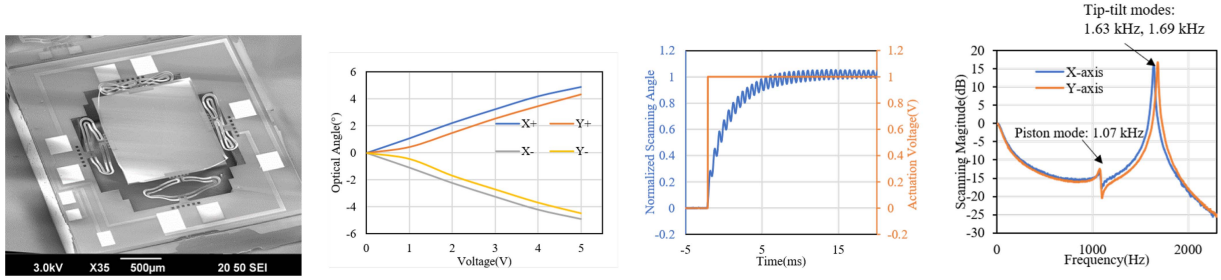


Fig. 11. (a) SEM image of the fabricated MEMS mirror. (b) Optical scanning angle response of the MEMS mirror. (c) Step response of the MEMS mirror is 5 ms. (d) Frequency response of the MEMS mirror.

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

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

Take the mirror scanning step 0.25° as the maximum tolerance of the excited mirror plate rotation, the tolerable external angular acceleration is $\ddot{\theta} = 44\,000$ rad/s^2. The maximum angular acceleration of a commercialized robot is usually less 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 differences 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 of 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 $d_{\max} = 200\,\mu$m. The maximum tolerable acceleration in the direction perpendicular to the mirror plate is $a_{\max}$ is

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

For most commercial robots, the 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 Section IV-D-2, we quantify the physical system's step response delay time at 5 ms; see Fig. 11(c). In Fig. 10, we further quantify the effects of rotation disturbance of the robot, versus compensation error, with various timing profiles.

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. 4(b) and (d).

Furthermore, in Section VI-C-6, we further investigate the effects of time delay in our motion-compensated LiDAR inertial odometry pipeline, under noisy conditions. To improve 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 Section IV-D-2 and Fig. 10, 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 and Fig. 17(a).

Furthermore, real IMU has noise; we similarly quantify the effects of increasing IMU noise in Section VI-C-4 and Fig. 17(b).

Furthermore, real LiDAR has ToF distance measurement noise; we similarly quantify the effects of ToF distance measurement noise in Section VI-C-5 and Fig. 17(c).

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

1) We discuss the key components and characteristics of the proposed LiDAR system design, particularly focusing on the MEMS mirror.
2) 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 a) general rotation compensation, b) 2-axis-only compensation, c) rotational FoV stabilization, and d) target aiming.
3) 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.
4) We analyze the robustness of the proposed motion compensation control in a tightly coupled LiDAR SLAM simulation. The result is presented in Section VI.
5) 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 visible light, and the other visible laser is fixed at a relatively higher position on the UAV, as shown in the images in Fig. 12(b). The target scanning direction is a fixed point on the target.

Here, the entire scanning grid $\{\alpha_i, \beta_i, 1\}$ consists of 20×20 grid pattern points. We use the aiming compensation outline in Section IV-B-5.

We trim about 12-s videos in each experiment while the UAV is flying, and then, 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 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 1 Hz, 2 Hz to 50 Hz, the width of the overlapping area shrinks from 10 to 11 points at 1 Hz [see Fig. 13(f)], to 6 points at 50 Hz [see Fig. 13(d)]. As Fig. 14 demonstrates, the compensation frequency has a clear impact on the quality of the captured point-cloud.

VI. Rotation-Compensated LiDAR-Inertial SLAM Design

SLAM is a body of fundamental applications for visual sensors. All existing SLAM literature reason about its odometry in the sensor's local frame, sometimes called the camera frame. In this work, this frame is the robot frame, with world frame orientation $\boldsymbol{R}^{w}_{\text{robot}}$; refer to Section IV-B-1-a.

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

Through Sections IV-B-2–IV-B-5, the additional none-zero rotation $\boldsymbol{R}_{\text{control}}$ orients the original scanning grid toward different directions. The existence of $\boldsymbol{R}_{\text{control}}$ breaks the basic assumption of existing SLAM.

$\boldsymbol{R}_{\text{control}}$ must be compensated for, in order for the existing SLAM pipelines to work with our sensor. This can be done postcapture, we can use either Sections IV-B-2 or IV-B-3 to compensate. We detail the compensation later in Section VI-B.

Most LiDAR odometry pipelines utilize 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 ICPs prior. This would directly impact the quality of ICPs point-cloud registration. Although ICP can tolerate certain levels of error in its prior, in Section VI-C-2, we will show that it is far from enough when the magnitude of $\boldsymbol{R}_{\text{control}}$ input increases.

A. Motion Compensation for LiDAR SLAM

In this simulation, we simulate a 360° Velodyne LiDAR, which 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, see Section IV-B-3. 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

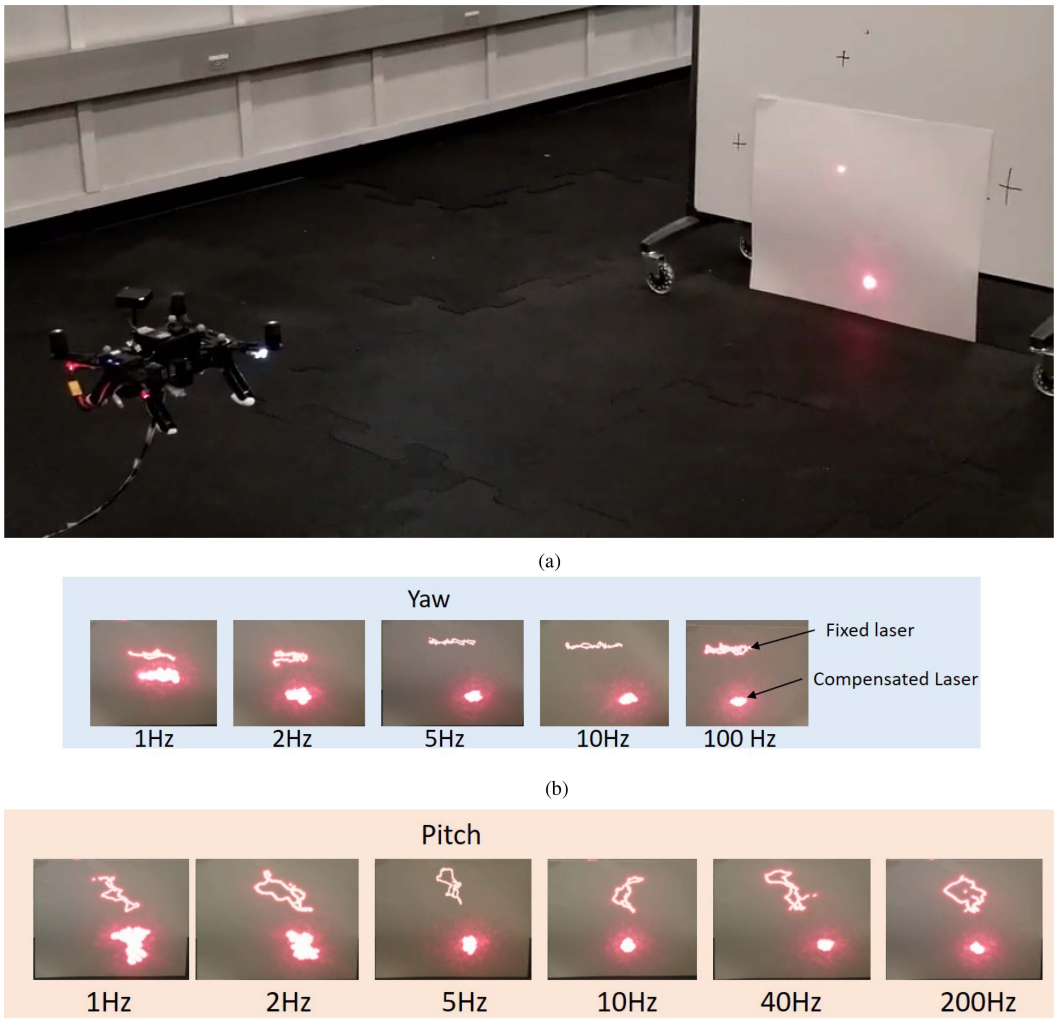


Fig. 12. Comparison of the compensation strength versus the robot pose sampling frequencies. All the images are the accumulation of 12 s of UAV hovering videos. The compensation target scanning direction is a fixed direction. (a) Our UAV setup with Intel Aero UAV and our LiDAR mounted on it. (b) Effect of compensation rate on yaw rotation. (c) Effect of compensation rate on pitch rotation.

LiDAR-inertial odometry and mapping package, which is publicly available on https://github.com/yuyangch/LIO-SAM_rotationGithub. For ease of integration, our framework proposal does not make large edits in the existing paradigm. It only adds a "rotate" stage right after the deskew stage in the front end and before the feature extraction stage. This edition can be easily integrated with existing pipelines and future designs. The rotate stage does one single operation, it rotates the deskewed point cloud according to the control rotation input to the LiDAR. Our workflow block diagram is shown in Fig. 14.

B. 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.)

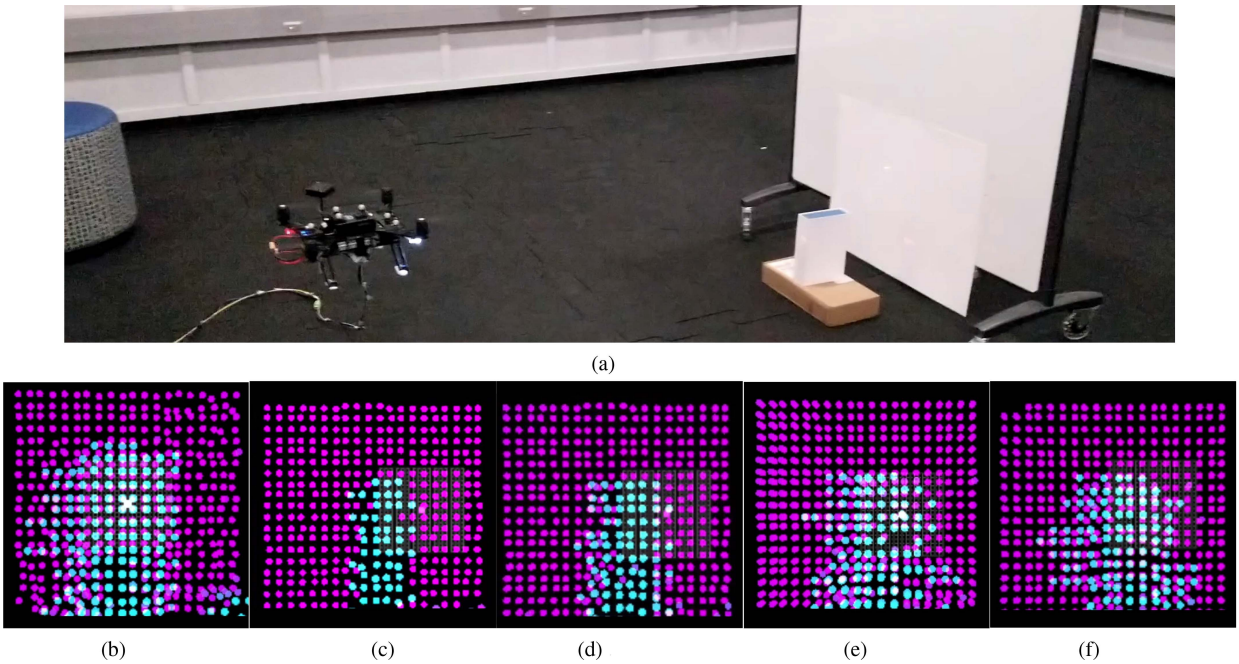
Let the LiDAR's base frame have world rotation $\boldsymbol{R}^{w}_{\text{robot}} \in SO(3)$.

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

The LiDAR's head can rotate $\boldsymbol{R}_{\text{control}} \in SO(3)$, relative to its base. This rotation is restricted to azimuth β and elevation directions α. Note that here, we analyze a more generalized, special case compensation (see Section IV-B-3), but it can be easily extended to full $SO(3)$ compensation (see Section IV-B-2).

When a LiDAR frame is received, we take the most recently known rotation α, β, in this case, the most recent known command rotation, and convert them into a rotation matrix

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



(a) (b) (c) (d) (e) (f)

Fig. 13. Comparison of the compensation strength verse the IMU sampling frequencies. The images are the accumulation of 20 s of point cloud video during the UAV hovering. We use a cuboidal object [as seen in panel (a)] as object of interest. The width of the target increases due to compensation inaccuracy as we reduce the compensation rate from 50 to 1 Hz demonstrating the utility of high rates of compensation even in such static scenarios. (a) Our setup with UAV, LiDAR, and the feature target. (b) Uncompensated. (c) One frame. (d) 50-Hz sampling rate. (e) 2-Hz sampling rate. (f) 1-Hz sampling rate.

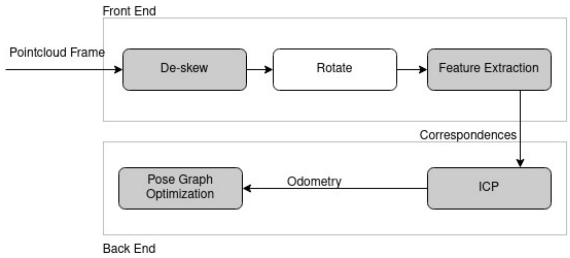


Fig. 14. Illustration of our rotating LiDAR SLAM augmentation (existing modules are shown in gray).

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

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

The rotated point-cloud p_{rotated} is now located at the correct position, relative to the LiDAR's base frame, with world rotation $\boldsymbol{R}^{w}_{\text{robot}}$. The basic assumptions 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 an active stabilization a control policy proposed in Section 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 Fig. 14. 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 the average translation error, which is defined by the KITTI benchmark [56]

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

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

1) Experiment Setup: A simulation study is a setup in the robotics simulator Gazebo, where a LiDAR with similar sensor characteristics to a Velodyne VLP-32 is mounted on a simulated drone. Furthermore, 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 has noise added to it according to a noise model outlined in

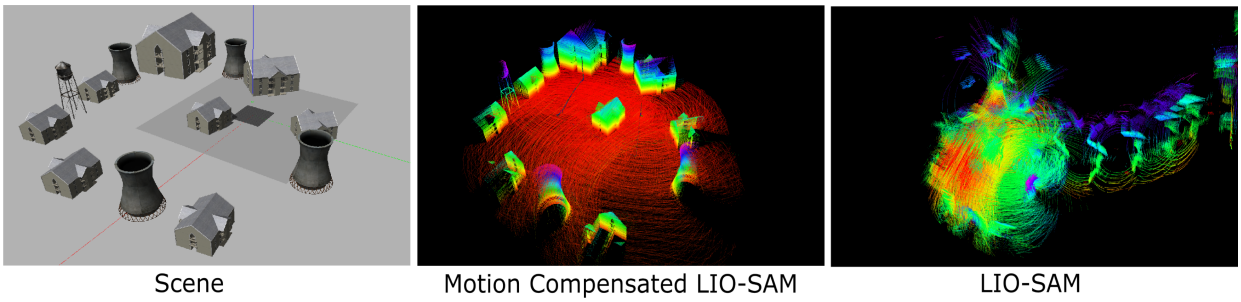


Fig. 15. Illustration of a) our simulator environment, b) mapping results on LIO-SAM with our motion compensation, and c) mapping results on stock LIO-SAM. It is worth noting that the pointcloud generated from the simulation has rotation with respect to the frame with world rotation $\boldsymbol{R}_{\text{control}}\boldsymbol{R}^{w}_{\text{robot}}$, 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 uncompensated, traditional SLAM to handle.

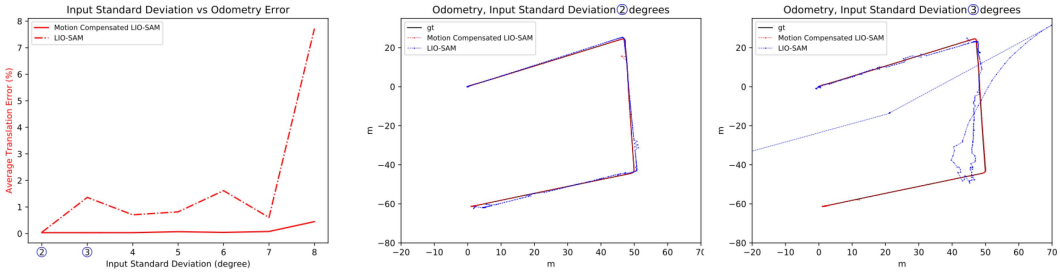


Fig. 16. (a) Mirror control magnitude and odometry error. Our motion-compensated LIO-SAM versus LIO-SAM. As mirror control magnitudes increase, the unmodified LIO-SAM fails completely. (b) At 2° standard deviation, our motion-compensated LIO-SAM outperforms LIO-SAM. (c) At 3° standard deviation, degree threshold, and beyond, our motion-compensated LIO-SAM performs normally while the stock LIO-SAM completely fails.

Kalibr [57]. 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 an environment with different types of resident buildings.

The proposed LiDAR-inertial SLAM package builds on top of LIO-SAM, which employs the powerful PGO backend GTSAM [40]. We incorporate the compensation described in Section VI-B into LIO-SAM, which we refer here as motion-compensated LIO-SAM. Naturally, we will compare the SLAM performance of motion-compensated LIO-SAM, against the stock version of LIO-SAM; see Fig. 15. 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 Versus Our System: The two angular commands are sampled from 1-D Gaussian distributions with a standard deviation of various degrees at 10 Hz. Odometry error versus command rotation's Gaussian standard deviation is plotted in Fig. 16. A Gaussian distribution with 8° standard deviation generates an input angle within ±8°, ±16°, ±24°, 68%, 95%, and 99.7% of the time, respectively. Therefore, 99.7% of the time, angular input spans a range of 48°.

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

Even in the cases where the input spans less than 12°, 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 a standard deviation of 3° at 10 Hz. We use our proposed motion-compensated LIO-SAM here.

Additionally, noise rotations in both azimuth and elevation are added on top of each channel. Odometry error versus command rotation's noise Gaussian standard deviation is plotted in Fig. 17(a). The system can tolerate mirror input control noise up to 1.6° standard deviation, which spans 9.6°.

As Fig. 10 shows, when 1.8° of disturbance from the robot body is generated, there is approximately 10% or ±0.18° of peak actuation errors, and approximately 3%, or ±0.05° of an actuation error off-peak. Assuming in the worst case, a robot generates 18° of disturbances from the robot body; this translates to 1.8° of a peak actuation error and 0.5° of an actuation error off-peak. Using a Gaussian error profile, we have a standard deviation at 1.8/3=0.6°, 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 a standard deviation of 3° at 10 Hz. We use our proposed

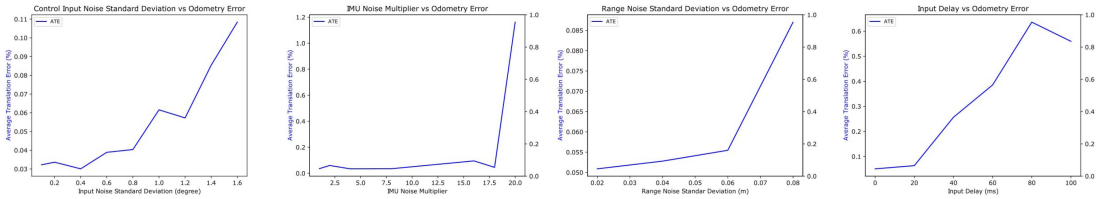


Fig. 17. (a) Mirror control noise and odometry error. As the mirror control noise increases, the odometry error also increases. Our pipeline fails after the noise standard deviation exceeds 1.6°. (b) IMU noise and odometry error. As the IMU noise factor increases, the odometry error also increases. SLAM failure in our pipeline occurs at factor 20. (c) As range measurement noise increases, the odometry error also increases. (d) As input delay increases, the odometry error also increases. Our system can tolerate an input delay of 30 ms.

motion-compensated LIO-SAM here. Additionally, IMU noise is added on top of IMU output, according to an IMU noise model outlined in Kalibr [57]. The model is based on the work in [58], [59], and [60]. The base set of noise parameters is outlined in the following table.

Parameter	Unit	Value
Gyroscope noise density	$\frac{\text{rad}}{\text{s}}\frac{1}{\sqrt{\text{Hz}}}$	3.394e-4
Gyroscope random walk	$\frac{\text{rad}}{\text{s}^2}\frac{1}{\sqrt{\text{Hz}}}$	3.879e-05
Gyroscope turn on bias sigma	$\frac{\text{rad}}{\text{s}}$	8.727e-3
Accelerometer noise density	$\frac{\text{m}}{\text{s}^2}\frac{1}{\sqrt{\text{Hz}}}$	4e-3
Accelerometer random walk	$\frac{\text{m}}{\text{s}^3}\frac{1}{\sqrt{\text{Hz}}}$	6e-3
Accelerometer turn on bias sigma	$\frac{\text{m}}{\text{s}^2}$	0.196

The previous sets of noise parameters are multiplied by several factors and their relationship with odometry errors is shown in Fig. 17(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 an 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, 0.005–0.008 standard deviation [61]. We simulate increasing Gaussian distance measurement noise versus the odometry error.

The two angular commands are sampled from 1-D Gaussian distributions with a standard deviation of 3° 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 to 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. 17(c).

6) Level of Actuation Delay Tolerable Under Noisy Condition: Our modification to LIO-SAM requires timely reporting of the actuator joint/MEMS position. Actuation delay can, therefore, impact the odometry accuracy.

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

Different from Section III-B, to increase realism, we add the aforementioned mirror control noise, IMU noise, and LiDAR distance measurement noise.

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

The IMU noise is set at 3× of the base IMU noise level mentioned in Section VI-C-4.

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

The two angular commands are sampled from 1-D Gaussian distributions with a standard deviation of 3° at 10 Hz. We use our proposed motion-compensated LIO-SAM here. The result can be seen in Fig. 17(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 experiments. We have demonstrated the experiment image stabilization in hardware using an onboard IMU. We have also demonstrated viewing an object of interest using this LiDAR through an external robot pose feedback. See the Supplementary material of this article 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 the limitations of our study, which are as follows.

1) We have indirectly compared 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 compare with software compensation methods.
2) Our design requires the robot to be 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.

3) 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.
4) 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.
5) 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 2. We have shown, in simulation and in real hardware experiments, that hardware-compensation using a MEMS mirror improves both reconstruction and mapping. In particular, microrobots that suffer from heavy vibration and motion jitter (such as flapping-wing MAVs [9]) 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.

ACKNOWLEDGMENT

This article describes work performed at the University of Florida and is not associated with Amazon.

REFERENCES

[1] F. Neuhaus, T. Koß, R. Kohnen, and D. Paulus, "MC2SLAM: Real-time inertial LiDAR odometry using two-scan motion compensation," in *Proc. German Conf. Pattern Recognit.*, 2018, pp. 60–72.
[2] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3D point cloud matching with smoothed densities," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5545–5554.
[3] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual SLAM algorithms: A survey from 2010 to 2016," *IPSJ Trans. Comput. Vis. Appl.*, vol. 9, no. 1, 2017, Art. no. 16.
[4] E. Phelps and C. A. Primmerman, "Blind compensation of angle jitter for satellite-based ground-imaging LiDAR," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 2, pp. 1436–1449, Feb. 2020.
[5] G. Costante, C. Forster, J. Delmerico, P. Valigi, and D. Scaramuzza, "Perception aware path planning," 2016, *arXiv:1605.04151*.
[6] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon "next-best-view" planner for 3D exploration," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 1462–1468.
[7] Z. Zhang and D. Scaramuzza, "Perception-aware receding horizon navigation for MAVs," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 2534–2541.
[8] S. A. Sadat, K. Chutskoff, D. Jungic, J. Wawerla, and R. Vaughan, "Feature-rich path planning for robust navigation of MAVs with mono-SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 3870–3875.
[9] R. Wood, R. Nagpal, and G.-Y. Wei, "Flight of the RoboBees," *Sci. Amer.*, vol. 308, no. 3, pp. 60–65, 2013. [Online]. Available: http://www.jstor.org/stable/26018027
[10] K. L. Hoffman, "Design and locomotion studies of a miniature centipede-inspired robot," Ph.D. dissertation, School Eng. Applied Sci., Harvard University Cambridge, Massachusetts, May 2013.
[11] Y. Mulgaonkar, G. Cross, and V. Kumar, "Design of small, safe and robust quadrotor swarms," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 2208–2215.
[12] E. F. Helbling, S. B. Fuller, and R. J. Wood, "Pitch and yaw control of a robotic insect using an onboard magnetometer," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 5516–5522.
[13] D. Xue, B. Song, W. Song, W. Yang, W. Xu, and T. Wu, "Computational simulation and free flight validation of body vibration of flapping-wing MAV in forward flight," *Aerosp. Sci. Technol.*, vol. 95, 2019, Art. no. 105491.
[14] C.-H. Chen, Y.-L. Kuo, T.-Y. Chen, and J.-R. Chen, "Real-time video stabilization based on motion compensation," in *Proc. 4th Int. Conf. Innov. Comput., Inf. Control*, 2009, pp. 1495–1498.
[15] T. Peng and S. K. Gupta, "Model and algorithms for point cloud construction using digital projection patterns," *J. Comput. Inf. Sci. Eng.*, vol. 7, no. 4, pp. 372–381, 2007.
[16] G.-Z. Yang et al., "The grand challenges of science robotics," *Sci. Robot.*, vol. 3, no. 14, 2018, Art. no. eaar7650.
[17] T. Alldieck, M. Kassubeck, B. Wandt, B. Rosenhahn, and M. Magnor, "Optical flow-based 3D human motion estimation from monocular video," in *Proc. German Conf. Pattern Recognit.*, 2017, pp. 347–360.
[18] "Hawk head stabilization," Jan. 2020. [Online]. Available: https://www.youtube.com/watch?v=aqgewVCC0k0
[19] Z. Tasneem, C. Adhivarahan, D. Wang, H. Xie, K. Dantu, and S. J. Koppal, "Adaptive fovea for scanning depth sensors," *Int. J. Robot. Res.*, vol. 39, no. 7, pp. 837–855, 2020.
[20] A. Kasturi, V. Milanovic, B. H. Atwood, and J. Yang, "UAV-borne LiDAR with MEMS mirror-based scanning capability," in *Proc. SPIE*, vol. 9832, 2016, Art. no. 98320M.
[21] K. Kimoto, N. Asada, T. Mori, Y. Hara, A. Ohya, and S. Yuta, "Development of small size 3D LiDAR," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 4620–4626.
[22] D. Wang, L. Thomas, S. Koppal, Y. Ding, and H. Xie, "A low-voltage, low-current, digital-driven MEMS mirror for low-power LiDAR," *IEEE Sens. Lett.*, vol. 4, no. 8, Aug. 2020, Art. no. 5000604.
[23] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled LiDAR inertial odometry via smoothing and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5135–5142.
[24] G. Li, J. K. Suhr, S.-I. Noh, and J. Kim, "Tracking moving objects by using a pan-tilt-zoom camera," in *Proc. ITC-CSCC: Int. Tech. Conf. Circuits Syst., Comput., Commun.*, 2009, pp. 1012–1015.
[25] S. Hrabar, P. Corke, and V. Hilsenstein, "PTZ camera pose estimation by tracking a 3D target," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 240–247.
[26] J. K. Suhr, H. G. Jung, G. Li, S.-I. Noh, and J. Kim, "Background compensation for pan-tilt-zoom cameras using 1-D feature matching and outlier rejection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 3, pp. 371–377, Mar. 2011.
[27] R. Antonello, R. Oboe, A. Ramello, K. Ito, N. Felicini, and A. Cenedese, "IMU-aided image stabilization and tracking in a HSM-driven camera positioning unit," in *Proc. IEEE Int. Symp. Ind. Electron.*, 2013, pp. 1–7.
[28] R. K. Tyson, "Performance assessment of MEMS adaptive optics in tactical airborne systems," *Proc. SPIE*, vol. 3762, pp. 91–100, 1999.
[29] M. Ben-Ezra, A. Zomet, and S. K. Nayar, "Jitter camera: High resolution video from a low resolution detector," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2, 2004, p. II.
[30] R. Jia, V. K. Nandikolla, G. Haggart, C. Volk, and D. Tazartes, "System performance of an inertially stabilized gimbal platform with friction, resonance, and vibration effects," *J. Nonlinear Dyn.*, vol. 2017, 2017, Art. no. 6594861.
[31] Q. Li, S. Xu, Y. Xu, L. Li, and L. Zhang, "Nonorthogonal aerial optoelectronic platform based on triaxial and control method designed for image sensors," *Sensors*, vol. 20, no. 1, 2020, Art. no. 10.
[32] A. Sagitov and Y. Gerasimov, "Towards DJI phantom 4 realistic simulation with gimbal and RC controller in Ros/Gazebo environment," in *Proc. 10th Int. Conf. Develop. eSyst. Eng.*, 2017, pp. 262–266.
[33] H. Grüger, A. Heberer, C. Gerwig, P. Nauber, M. Scholles, and H. Lakner, "3.1: MOEMS laser projector for handheld devices featuring motion compensation," in *Proc. SID Symp. Dig. Tech. Papers*, 2007, vol. 38, pp. 1–3.
[34] T. Taketomi and J. Heikkilä, "Zoom factor compensation for monocular SLAM," in *Proc. IEEE Virtual Reality*, 2015, pp. 293–294.
[35] T. Taketomi and J. Heikkila, "Focal length change compensation for monocular SLAM," in *Proc. IEEE Int. Conf. Image Process.*, 2015, pp. 4982–4986.

[36] I. Maksymova, P. Greiner, L. C. Niedermueller, and N. Druml, "Detection and compensation of periodic jitters of oscillating MEMS mirrors used in automotive driving assistance systems," in *Proc. IEEE Sensors Appl. Symp.*, 2019, pp. 1–5.
[37] T. Hayakawa, T. Watanabe, T. Senoo, and M. Ishikawa, "Gain-compensated sinusoidal scanning of a galvanometer mirror in proportional-integral-differential control using the pre-emphasis technique for motion-blur compensation," *Appl. Opt.*, vol. 55, no. 21, pp. 5640–5646, 2016.
[38] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," in *Proc. Conf. Robot.: Sci. Syst.*, 2014, vol. 2, pp. 1–9.
[39] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2O: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 3607–3613, doi: 10.1109/ICRA.2011.5979949.
[40] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," Georgia Inst. Technol., Atlanta, GA, USA, Tech. Rep., 2012.
[41] H. Ye, Y. Chen, and M. Liu, "Tightly coupled 3D LiDAR inertial odometry and mapping," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 3144–3150.
[42] H. Yang, J. Shi, and L. Carlone, "TEASER: Fast and certifiable point cloud registration," *IEEE Trans. Robot.*, vol. 37, no. 2, pp. 314–333, Apr. 2021.
[43] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct LiDAR-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053–2073, Aug. 2022.
[44] X. Deng, Z. Zhang, A. Sintov, J. Huang, and T. Bretl, "Feature-constrained active visual SLAM for mobile robot navigation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 7233–7238.
[45] S. Thrun, "Probabilistic robotics," *Commun. ACM*, vol. 45, no. 3, pp. 52–57, 2002.
[46] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "AirSim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*. Berlin, Germany: Springer, 2018, pp. 621–635.
[47] R. Hartley, J. Trumpf, Y. Dai, and H. Li, "Rotation averaging," *Int. J. Comput. Vis.*, vol. 103, no. 3, pp. 267–305, 2013.
[48] V. Milanović, A. Kasturi, J. Yang, and F. Hu, "Closed-loop control of gimbal-less MEMS mirrors for increased bandwidth in LiDAR applications," in *Proc. SPIE*, vol. 10191, 2017, Art. no. 101910N.
[49] D. Wang, C. Watkins, and H. Xie, "MEMS mirrors for LiDAR: A review," *Micromachines*, vol. 11, no. 5, 2020, Art. no. 456.
[50] K. Jia, S. Pal, and H. Xie, "An electrothermal tip-tilt-piston micromirror based on folded dual S-shaped bimorphs," *J. Microelectromech. Syst.*, vol. 18, no. 5, pp. 1004–1015, 2009.
[51] D. Wang, C. Watkins, M. Aradhya, S. Koppal, and H. Xie, "A large aperture 2-axis electrothermal MEMS mirror for compact 3D LiDAR," in *Proc. Int. Conf. Opt. MEMS Nanophotonics*, 2019, pp. 180–181.
[52] X. Zhang, L. Zhou, and H. Xie, "A fast, large-stroke electrothermal MEMS mirror based on CU/W bimorph," *Micromachines*, vol. 6, no. 12, pp. 1876–1889, 2015.
[53] D. Wang, X. Zhang, L. Zhou, M. Liang, D. Zhang, and H. Xie, "An ultra-fast electrothermal micromirror with bimorph actuators made of copper/tungsten," in *Proc. Int. Conf. Opt. MEMS Nanophotonics*, 2017, pp. 1–2.
[54] K. Ito et al., "System design and performance characterization of a MEMS-based laser scanning time-of-flight sensor based on a 256-pixel single-photon imager," *IEEE Photon. J.*, vol. 5, no. 2, Apr. 2013, Art. no. 6800114.
[55] M. Li, Q. Chen, Y. Liu, Y. Ding, and H. Xie, "Modelling and experimental verification of step response overshoot removal in electrothermally-actuated MEMS mirrors," *Micromachines*, vol. 8, no. 10, 2017, Art. no. 289.
[56] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.
[57] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 1280–1286.
[58] M. J. Ma et al. *IEEE Standard Specification Format Guide and Test Procedure for Single-Axis Interferometric Fiber Optic Gyros*, IEEE Std 952-1997, Feb. 1998.
[59] N. Trawny and S. I. Roumeliotis, "Indirect Kalman filter for 3D attitude estimation," Dept. Comput. Sci. Eng., Univ. Minnesota, Minneapolis, MN, USA, Tech. Rep. 2005-002, 2005.
[60] J. L. Crassidis, "Sigma-point Kalman filtering for integrated GPS and inertial navigation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 42, no. 2, pp. 750–756, Apr. 2006.
[61] C. L. Glennie, A. Kusari, and A. Facchin, "Calibration and stability analysis of the VLP-16 laser scanner," *Int. Arch. Photogrammetry Remote Sens. Spatial Inf. Sci.*, vol. 40, pp. 55–60, 2016.

Yuyang Chen received the bachelor of Science degree in electrical engineering from Stony Brook University, Stony Brook, NY, USA, in 2014, and the Ph.D. degree in computer science from the University at Buffalo, Buffalo, NY, in 2024 under the supervision of Prof. Dantu.

Dingkang Wang received the B.E. degree in mechanical engineering from Jilin University, Changchun, China, in 2016 and the Ph.D. degree in electrical engineering from the University of Florida, Gainesville, FL, USA, in 2020.

He is currently a LiDAR and Computer Vision Engineer in the autonomous driving industry. His research interest includes how to use machine learning, multisensor fusion, and generative AI algorithms to help autonomous driving tasks.

Lenworth Thomas received the bachelor of Science degree in mechanical engineering from the University of Florida, Gainesville, FL, USA, in 2021. He is currently working toward the Ph.D. degree in mechanical engineering focusing on robotics with Carnegie Mellon University, Pittsburgh, PA, USA.

His research interests include the design of robotics systems, robotics for environmental data collection, sensor fusion, autonomous navigation, and field robotics.

Karthik Dantu (Senior Member, IEEE) received the M.S. and Ph.D. degrees in computer science from the University of Southern California, Los Angeles, CA, USA, in 2010.

He is currently an Associate Professor in computer science and engineering with the University at Buffalo, State University of New York, Buffalo, NY, USA. He is also the founding Director of the Center for Embodied Autonomy and Robotics, University at Buffalo, a university-wide center for robotics research. He was a Postdoctoral Fellow in EECS with Harvard from 2010 to 2013. His research interests include robot perception, mobile perception, and safe and trustworthy autonomy.

Sanjeev J. Koppal (Senior Member, IEEE) received the B.S. degree in computer science from University of Southern California, in 2003 and M.S. and Ph.D. degrees in robotics from the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA, in 2006 and 2009, respectively.

He is currently an Associate Professor with the Electrical and Computer Engineering Department, University of Florida, Gainesville, FL, USA, and a Kent and Linda Fuchs Faculty Fellow. He also held a UF Term Professorship from 2021 to 2023. He is also the Director of the FOCUS Lab, UF. Since 2022, he has been an Amazon Scholar with Amazon Robotics. Prior to joining UF, he was a Researcher with Texas Instruments Imaging R&D Lab. After CMU, he was a Postdoctoral Research Associate with the School of Engineering and Applied Sciences, Harvard University.