OmniNxt: A Fully Open-source and Compact Aerial Robot with Omnidirectional Visual Perception

Peize Liu[1], Chen Feng[1,†], Yang Xu[2], Yan Ning[1], Hao Xu[1,†], and Shaojie Shen[1]

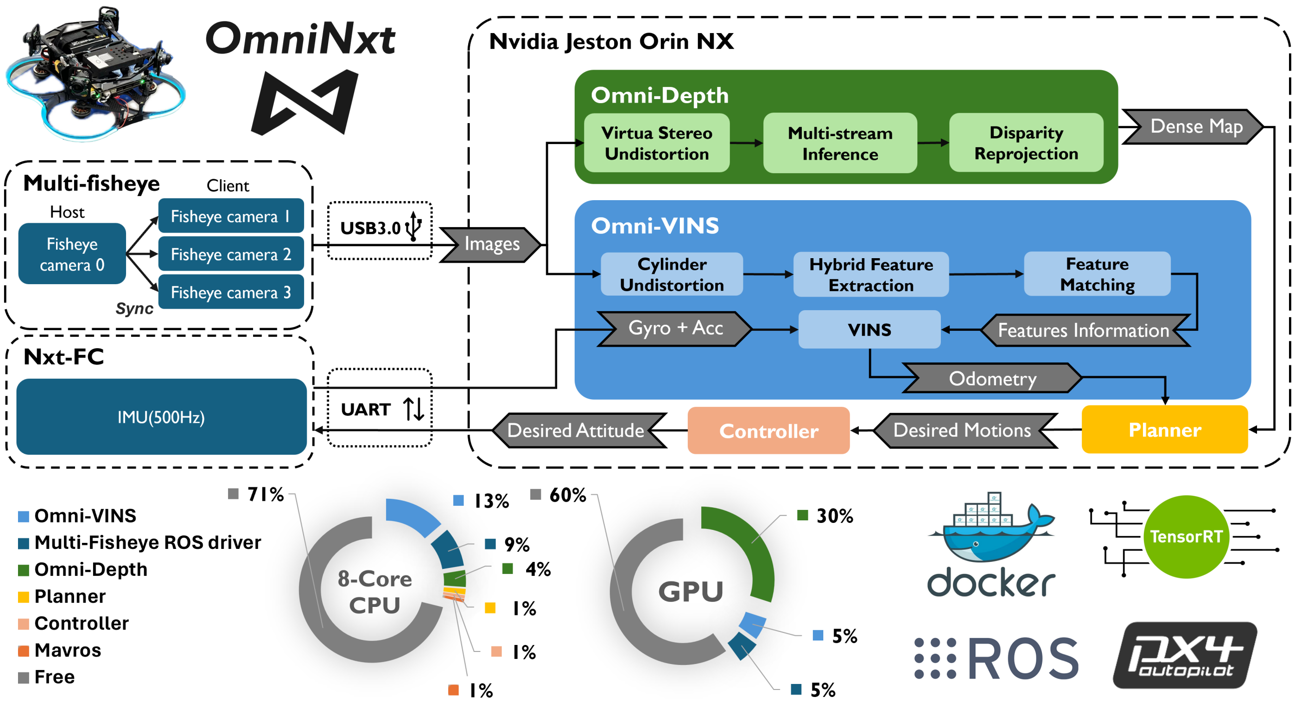

Fig. 1: The system overview of OmniNxt. The hardware architecture includes: Multi-fisheye camera set (camera), Nxt-FC (flight controller and IMU), and Nvidia Jeston Orin NX (onboard computation). The software framework consists of two critical components: Omni-VINS (Sec. III-D) and Omni-Depth (Sec. III-E).

***Abstract*— Adopting omnidirectional Field of View (FoV) cameras in aerial robots vastly improves perception ability, significantly advancing aerial robotics's capabilities in inspection, reconstruction, and rescue tasks. However, such sensors also elevate system complexity, *e.g.*, hardware design, and corresponding algorithm, which limits researchers from utilizing aerial robots with omnidirectional FoV in their research. To bridge this gap, we propose OmniNxt, a fully open-source aerial robotics platform with omnidirectional perception. We design a high-performance flight controller Nxt-FC and a multi-fisheye camera set for OmniNxt. Meanwhile, the compatible software is carefully devised, which empowers OmniNxt to achieve accurate localization and real-time dense mapping with limited computation resource occupancy. We conducted extensive real-world experiments to validate the superior performance of OmniNxt in practical applications. All the hardware and software are open-access at[3], and we provide docker images of each crucial module in the proposed system. Project page: https://hkust-aerial-robotics.github.io/OmniNxt.**

[1]Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong, China.

[2]Division of Emerging Interdisciplinary Areas, The Hong Kong University of Science and Technology, Hong Kong, China.

Email: `{pliuan,cfengag,yxuew,yningaa,hxubc}@ust.hk`, `eeshaojie@ust.hk`

† **Corresponding Authors**

[3]`https://github.com/HKUST-Aerial-Robotics/OmniNxt`

I. Introduction

In recent years, remarkable advancements in open-source aerial robotics platforms have led to a surge in practical applications, *e.g.*, exploration [1], reconstruction [2], and rescue [3]. However, it becomes evident that existing platforms, like FLA [4], MRS [5], and Agilicious [6], are challenging for effectively performing these tasks in increasingly complex and dynamic environments due to their limited FoV.

Contrary to platforms with limited FoV, the early development of aerial robots with omnidirectional FoV by Gao *et al.* [7] demonstrates the potential of omnidirectional FoV in performing the above tasks more robustly and efficiently. As proved by Zhang *et al.* [8] and Wang *et al.* [9], the omnidirectional FoV improves the localization accuracy in challenging environments. Additionally, omnidirectional FoV enables minimizing the indirect-controlled yaw rotation [10] during the flight [10], which enhances the energy efficiency of downstream tasks like autonomous navigation.

However, the lack of open-source omnidirectional FoV platforms can be attributed to the following challenges. **(1)** Sensor configuration: The standard way to achieve omnidirectional FoV [7] is using the multi-fisheye camera set.

Unlike off-the-shelf camera modules, this set requires substantial efforts in structural design, hardware development, and meticulous calibration. **(2)** Algorithm development: To comprehensively leverage the benefits offered by the omnidirectional FoV, all algorithms, *i.e.*, localization, mapping, planner, and controller should be adapted for properly exploiting the increased information. **(3)** System integration: Extensive testing and optimization of each component are essential to ensure stable and reliable performance within systems constrained by size and computational resources. System latency, resource occupancy, and overall performance should be carefully considered and optimized for robust and stable functionality in real-world scenarios.

To tackle these challenges and ensure platforms are broadly applicable to the research community, ***COPE*** criteria have been distilled from existing works and anticipated future challenges to guide the platform design.

- ***Compact***: At the hardware level, sensor components should be compact to minimize the size and weight of the platform, thereby improving maneuverability. Additionally, critical software modules should consume minimal system resources, allowing a flexible environment for further tasks and development.
- ***Open-source***: All hardware and software should be open-source, which facilitates cohesive reproduction, development, and enhancement by the community.
- ***Perceptive***: Comprehensive information about the surroundings and low-noise measurement should be perceived by sensors such as cameras and IMU. On the other hand, corresponding algorithms should be capable of effectively processing and utilizing the input from sensors to improve the system's robustness and stability.
- ***Extendable***: The hardware should be developed to allow seamless interchange and migration across platforms, thus amplifying the utility of the present design. In parallel, the software should be modular and easily updatable to expedite the validation of emerging algorithms and adaptation for subsequent applications.

Following the ***COPE*** criteria, we introduce OmniNxt, the first fully open-source aerial robotics platform with omnidirectional visual perception. OmniNxt boasts a compact size with exceptional computational resources. Thanks to our meticulous development, we endow OmniNxt with an expansive perception range and efficient resource consumption. Thorough evaluations demonstrate the effectiveness and advanced performance in real-world applications. The contributions of our system can be summarized as follows:

1) An open-source and compact hardware platform with omnidirectional FoV: We develop a coin-size yet high-performance flight controller Nxt-FC, which provides 500 Hz low-noise IMU data for the visual inertial odometry (VIO). In addition, we develop a multi-fisheye camera set to support omnidirectional perception. This set also offers synchronized images to VIO, where camera drivers and calibration tools are also provided.

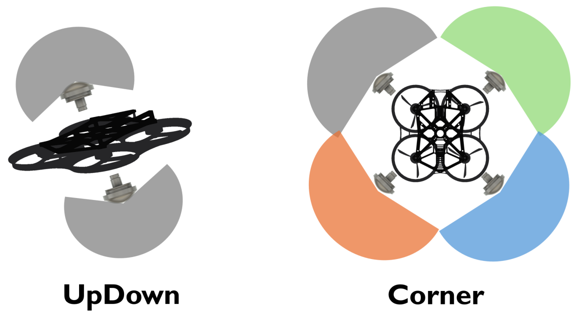

Fig. 2: Typical structures of omnidirectional FoV. Cameras in the UpDown structure are on the top and bottom of the platform, facing upwards and downwards. The Corner structure places cameras on the corners of a plane, covering an omnidirectional view.

2) An open-source and real-time omnidirectional perception framework: It consists of two principal components, Omni-VINS and Omni-Depth. The former provides accurate VIO, while the latter facilitates omnidirectional dense mapping. Both are devised as resource-efficient modules (See Fig. 1), which pave the way for advanced development in downstream tasks.
3) Extensive real-world experiments are conducted to evaluate the proposed system thoroughly. Experiments demonstrate that OmniNxt achieves outperforming localization accuracy and dependable point cloud quality. Moreover, autonomous navigation tests in a cluttered indoor environment validate the practicality of the proposed platform. Experiments also verify the adaptability of OmniNxt, highlighting its compatibility with a range of sensors and its ease of modification to suit diverse applications.

II. Related Works

A. Omnidirectional Visual Inertial Odometry

According to Zhang *et al.* [8], the FoV and the placement of the camera directly impact the odometry accuracy in different scenarios. An adequately installed camera with extended FoV enhances the effectiveness of feature extraction and tracking, contributing to improved odometry accuracy compared with limited FoV. To realize the omnidirectional FoV, existing works commonly adopt two typical configurations in structure: UpDown and Corner (Fig. 2). In the UpDown structure, cameras usually feature 250° to 360° FoV, while the Corner structure usually utilizes three or more cameras with 180° to 210° FoV. Compared with the Corner structure, the UpDown structure suffers more from image distortions, while the Corner structure involves more complex calibration.

Corresponding to the evaluation in [8], systems based on UpDown structure, such as the one proposed by Wang *et al.* [9], demonstrate higher odometry accuracy in indoor environments. This can be attributed to the greater probability of feature extraction and tracking in less distorted areas (center) of the images. However, their performance tends to be less favorable in outdoor settings. This discrepancy arises from

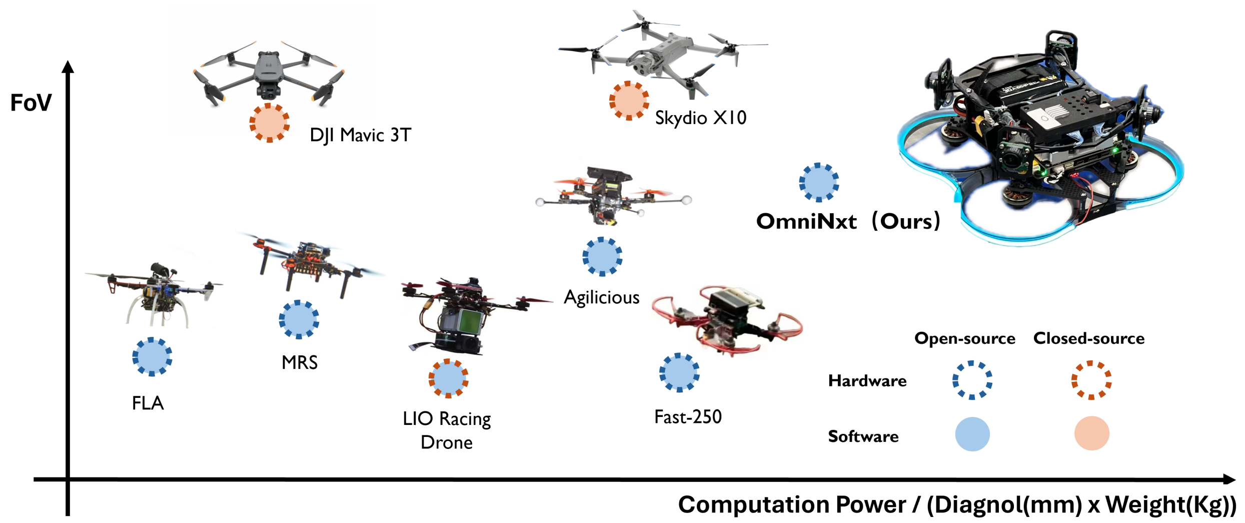

Fig. 3: Platforms comparison. The platforms are compared based on their FoV and the ratio of onboard computation power to the product of size and weight. A higher value on the vertical axis indicates the platform's ability to perceive the surrounding environment more comprehensively. On the horizontal axis, a higher value represents a greater computational power available within a smaller size and weight, indicating a stronger capability of the platform to support downstream tasks.

TABLE I: A comparison of different available consumer and research platforms. We evaluate them in hardware dimensions, weight, sensor types, perception information types, perception range, and extendability. HW: hardware. SW: software.

Platforms	Compact		Open-source		Perceptive				Extendable	
	HW	SW	HW	SW	Visual Info	Geometry Info	Sensor Type	FoV	HW	SW
DJI MAVIC 3T [11]	380mm 920g	-	✗	✗	✓	✓	Multi-fisheye camera set	360°	✗	✗
Skydio X10 [12]	≈380mm 2110g	-	✗	✗	✓	✓	Multi-fisheye camera set	360°	✗	✗
FLA [4]	450mm ≈2500g	Low	✓	✓	✓	✓	Camera & 2D-LiDAR	≈90°	✓	✓
MRS [5]	450mm ≈1500g	Low	✓	✓	✓	✓	Camera & 3D-LiDAR	≈90°	✓	✓
LIO Racing Drone [13]	330mm ≈1300g	Median	✗	✗	✗	✓	3D-LiDAR	≈70°	✗	✓
Agilicious [6]	330mm ≈775g	High	✓	✓	✓	✓	Stereo camera	≈90°	✗	✓
Fast-250 [14]	250mm ≈1000g	Median	✓	✓	✓	✓	Stereo camera	≈90°	✗	✓
OmniNxt(Ours)	**88.92mm 660g**	**High**	✓	✓	✓	✓	Multi-fisheye camera set	**360°**	✓	✓

the fact that features are more likely to be extracted and tracked in the greater distorted areas (margin) when the system performs outdoors, subsequently introducing more noise into the estimation of system status.

Works like Omnidirectional DSO [15], based on Corner structure, achieve higher accuracy in both scenarios because of the less distorted image input. However, compared with the feature-based method like VINS [16], the direct method is unsuitable for agile maneuvering platforms due to its sensitivity to motion blur and changes in image illumination.

B. *Omnidirectional Depth Estimation*

Retrieving dense depth information from multiple images captured by fisheye cameras is challenging due to the significant distortions. Existing solutions to this problem can be categorized into direct and indirect methods.

Most direct methods, such as OmniMVS [17], employ neural networks to obtain dense depth information directly from the raw images. However, these methods often struggle with generalization across different cameras and can hardly meet the real-time requirement on resource-limited platforms.

On the other hand, the indirect methods aim to transfer the raw images into multiple pairs of stereo images captured by the virtual pinhole cameras (See Sec. III-E). Gao *et al.* [7] leverage semi-global matching (SGBM) to obtain a dense depth estimation on resource-limited platforms. While Xie *et al.* [18] employ a CNN to obtain a denser depth estimation. Our approach is similar to [18], but we achieve real-time inference speed on resource-limited platforms while maintaining a more flexible structure in realization (Sec. III-E).

C. Available Platforms

Numerous research groups have made valuable contributions to the community by sharing their design. The design of these platforms varies greatly depending on the research topics they address. Based on the ***COPE*** criteria, TABLE. I provides an overview of currently accessible platforms.

Existing platforms like FLA [4] and MRS [5] are designed for data collection and autonomous navigation in open areas. These platforms share a similar solution, utilizing a PX4 [19] based flight controller, a CPU-only onboard computer, mono or stereo cameras, and LiDAR. However, achieving compactness and lightweight in their design is challenging due to the introduction of LiDAR to obtain dense map. On the other hand, the platform, which is designed for aggressive planning and control, prioritizes maximizing the thrust weight ratio (TWR). One notable platform in this category is Agilicious [6]. Different from those platforms mentioned above, Agilicous [6] achieves compactness and lightweight by adopting a cramped hardware design, which allows the platform to have high TWR but makes it hard to integrate additional sensors. This limitation poses challenges in accessing a comprehensive surrounding environment, limiting its applicability to future tasks. Besides, commercial platforms like DJI MAVIC 3T [11] and SKYDIO X10 [12] are not open-source, hindering further development and algorithm validation. Their large dimensions and weight also limit the range of applications.

III. OmniNxt Platform

A. System Overview

OmniNxt is designed to be a general open-source aerial robotics platform. Therefore, we strictly follow the **COPE** criteria in hardware and software design.

As shown in Fig. 1, OmniNxt consists of three key components in hardware: First, We use Nvidia Jetson Orin NX for the onboard computer, which has an 8-core CPU running at 2.0 GHz and a GPU with 1024 CUDA cores. CPU and GPU share the unified 16G RAM. Second, we adopt a multi-fisheye camera set to capture the omnidirectional image at 20 Hz. All four cameras are synchronized by one camera triggering the other three. Last, to further enhance the robustness of OmniNxt, we develop a low-noise flight controller, which provides 500 Hz low-noise IMU data. Besides, the software framework of OmniNxt can be divided into the following four parts, each serving a specific function:

- **Omni-VINS**: This module combines the high-frequency IMU measurements (500 Hz) and the omnidirectional image (20 Hz) to generate accurate VIO, which is crucial for mapping, planning, and control modules. (Sec. III-D)
- **Omni-Depth**: This module performs real-time omnidirectional dense point cloud generation by processing the omnidirectional image with its virtual-stereo frontend and multi-stream-inference backend. (Sec. III-E)

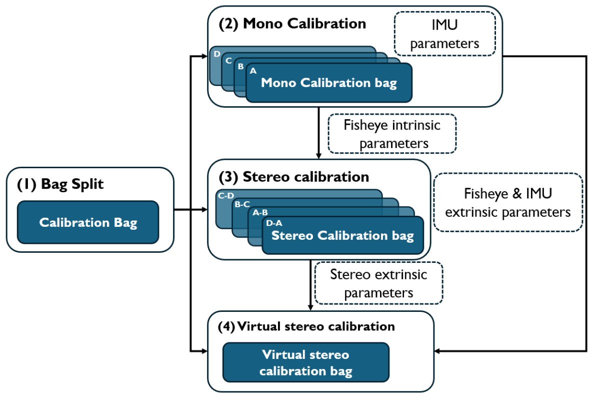
(2) Mono Calibration
IMU parameters
D
C
B
A
Mono Calibration bag
Fisheye intrinsic parameters
(1) Bag Split
Calibration Bag
(3) Stereo calibration
C-D
B-C
A-B
D-A
Stereo Calibration bag
Fisheye & IMU extrinsic parameters
Stereo extrinsic parameters
(4) Virtual stereo calibration
Virtual stereo calibration bag

Fig. 4: The calibration pipeline of multi-fisheye camera set. The numbers indicate the calibration sequence. The letter on the top left of each box corresponds to the camera index in Fig. 5.B

- **Planner**: Based on the Omni-VINS and Omni-Depth, this module generates the trajectory toward the goal position following the aerial robotic dynamics. (Sec. III-F)
- **Controller**: The trajectory generated from the planner is converted to a low-level attitude command by this module and then executed by the flight controller. (Sec. III-F)

The perception modules (Omni-VINS, Omni-Depth, and related hardware drivers) in OminNxt occupy 27% CPU and 40% GPU onboard resources, sparing sufficient resources for the downstream tasks.

B. Nxt-FC

The lack of open-source hardware options in most available flight controllers presents a significant challenge in further development and adaptation on various platforms. To address this issue, we design and open-source a flight controller named Nxt-FC with compact dimensions of 27mm$\times$33mm. Our flight controller is based on open-source autonomous pilot firmware PX4 [19]. We develop a high-frequency raw IMU data stream for robust and accurate VIO. Detailed information is available on our project page.

C. Multi-fisheye Camera Set Calibration

Calibrating all four fisheye cameras' intrinsic and extrinsic parameters together results in a complex optimization problem, thus increasing processing time and lowering the success rate. To address this issue, our pipeline utilizes tartankalibr [20] in fisheye cameras calibrations and kalibrtoolbox [21] [22] in virtual pinhole cameras calibrations. The calibration process is illustrated in Fig. 4.

Initially, we calibrate the intrinsic parameters of each fisheye camera. Once the projection error of each camera is less than 0.5 pixels, we proceed to calibrate the extrinsic parameters between adjacent cameras as well as between cameras and IMU.

After having established the intrinsic and extrinsic parameters of the multi-fisheye camera set, we undistort the image based on the cylindrical camera model (Sec. III-D) to generate virtual stereo pairs (Fig. 5.C) and calibrate both intrinsic and extrinsic parameters of virtual stereo cameras. The transformation between fisheye and virtual stereo cameras is further explained in Sec. III-E.

Our pipeline reduces the calibration time by concurring the calibration process and raises the success rate by allowing the calibration process to resume from the failed case quickly.

D. Omni-VINS

Omni-VINS is optimized for onboard real-time performance based on our previous work D^2SLAM [23]. The significant distortion in fisheye images poses challenges in feature extraction and feature tracking, particularly when using CNN-based methods that are trained with images gathered by the pinhole camera. To address this issue, we employ the MEI model [24] for the intrinsic parameters of raw fisheye cameras and radial-tangential models to formulate the distortion. First, We undistort the fisheye camera image with the cylindrical camera model. The cylindrical camera model [25] can be written as:

$$\begin{aligned}\begin{bmatrix}u\\v\\1\end{bmatrix} &= \begin{bmatrix}f_\phi & 0 & u_0\\ 0 & f_y & v_0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\phi\\ Y/\rho\\ 1\end{bmatrix},\\ \phi &= atan2(X,Z),\\ \rho &= \sqrt{X^2+Z^2},\end{aligned} \tag{1}$$

where f_ϕ and f_y are the focal length. This model offers the advantage of representing virtual cameras with adjustable FoV and rotations relative to the original camera.

In our implementation, we set f_ϕ to $\frac{190^\circ}{W}$ (W is the width of the image) to eliminate the unexposed margin of the image. To ensure the consistency of the undistorted image, we set f_y equal to f_ϕ.

We then perform feature extraction and tracking based on the cylindrical undistorted image. Omni-VINS heritages the hybrid feature extraction strategy developed in D^2VINS [23] but changes the tracking method from Superglue [26] to Lucas-Kanade (LK) [27] method to minimize the GPU occupancy and accelerates the matching process. We perform feature tracking in both previous and latest frames and among adjacent cameras' frames.

Our previous works [16] and [7] have thoroughly formulated the visual-inertial problem. The full state vector in the sliding window is defined as:

$$\begin{aligned}\mathcal{X} &= [x_0, x_1, ..., x_m, x^b_{C_0}, x^b_{C_1}, ..., x^b_{C_n}, \lambda_0, \lambda_1, ..., \lambda_l],\\ x_i &= [p^W_{b_i}, v^W_{b_i}, q^W_{b_i}, b^b_a, b^b_g], k \in [0, m],\\ x^b_{c_j} &= [p^b_{c_j}, q^b_{c_j}], j \in [0, n],\end{aligned} \tag{2}$$

where m is the total number of keyframes, λ_i is the inverse depth of the i^{th} feature from its first observation, and n is the number of fisheye cameras. The visual inertial odometry problem in Omni-D^2VINS is defined as:

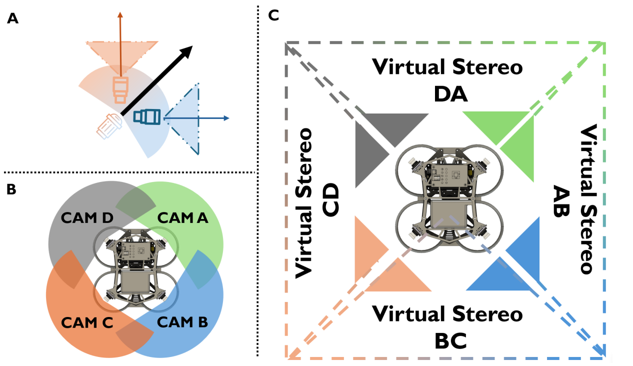

Fig. 5: Illustration of virtual-stereo frontend. **A**: the Z axis of virtual cameras (Orange and Blue) and the fisheye camera (Black). **B**: the placement of four fisheye cameras. **C**: shows the virtual stereo pairs.

$$\min_{\mathcal{X}}\left\{\|r_p - H_p\mathcal{X}\|^2 + \sum_{K\in\mathcal{B}}\left\|r_{\mathcal{B}}(\hat{z}^{b_k}_{b_{k+1}}, \mathcal{X})\right\|^2_{P^{b_k}_{b_{k+1}}} + \sum_{(l,j)\in\mathcal{C}}\left\|r_{\mathcal{C}}(\hat{z}^{c_j}_l, \mathcal{X})\right\|^2_{P^{c_j}_l}\right\} \tag{3}$$

We use Ceres Solver [28] for solving this least squares problem.

E. Omni-Depth

Omni-Depth is designed with a modular structure consisting of a virtual-stereo frontend and a multi-stream-inference backend to ensure the generalization in various stereo-matching techniques.

The virtual-stereo frontend is demonstrated in Fig. 5.C, where the fisheye camera images are undistorted into two perpendicularly placed virtual cylindrical cameras. The FoV of virtual cameras is set to 100°. The extrinsic parameters of the virtual pinhole cameras can be defined as follows:

$$\begin{aligned}\mathbf{T}_{vcam} &= \begin{bmatrix}\mathbf{R}^{vcam}_{fisheye} & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}\mathbf{R}_{fisheye} & \mathbf{t}_{fisheye}\\ 0 & 1\end{bmatrix},\\ \mathbf{R}^{vcam_l}_{fisheye} &= \mathbf{R}_y(-\frac{1}{4}\pi),\\ \mathbf{R}^{vcam_r}_{fisheye} &= \mathbf{R}_y(\frac{1}{4}\pi),\end{aligned} \tag{4}$$

where $vcam_l$ represents the left virtual camera, while $vcam_r$ represents the right one. Since the FoV of these two virtual cameras is close to the standard pinhole camera, we can calibrate them with the pinhole camera model.

In the backend of Omni-Depth, we employ CNN to estimate the disparity of each virtual stereo pair because of its more robust performance in the textureless environment. The Omni-Depth is flexible in accommodating different CNN models in the backend. This allows easy adaptation to meet various platforms' accuracy and inference speed requirements. See Sec. IV-B for the implementation details.

F. Planner and Controller

Combining precise and reliable odometry from Omni-VINS and dense point cloud from Omni-Depth, OmniNxt offers extensive support for various tasks. Benefited from The

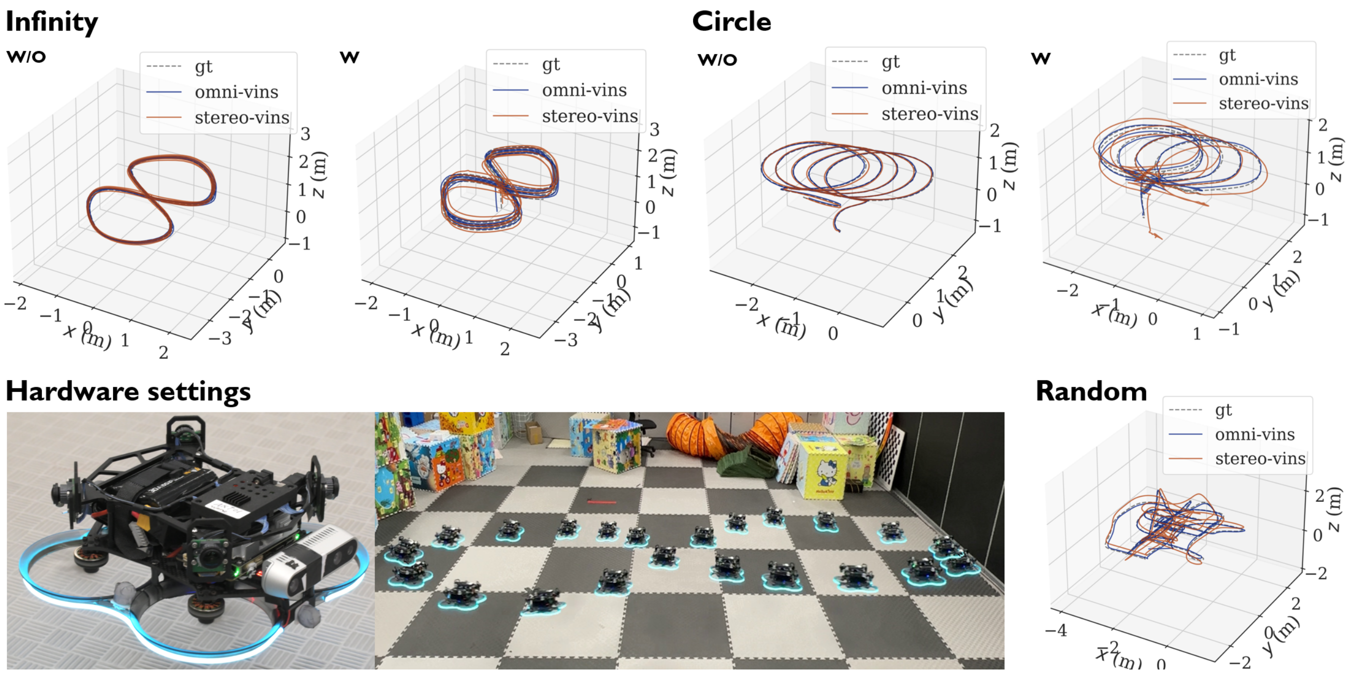

Fig. 6: Omni-VINS evaluation. **Infinity**, **Circle**, and **Random** are the three testing trajectories. **W/O** means flight without rotating the yaw to the speed direction. **gt** is the ground truth. **W** means flight while the yaw direction follows the speed direction. The data collection platform and testing environment are shown in **Hardware settings**. (Sec. IV-A)

omnidirectional perception, yaw rotation can be minimized during the flight, which simplifies the scheme of the trajectory planning in autonomous navigation (Sec. IV-C) while increasing the accuracy of VIO (Sec. IV-A). The command from the tasks level is fed into the controller module and then converted into the desired attitude command in MAVROS format. For efficient control, We utilize the open-source Px4-Controller [14].

IV. Experiments

We conduct a comprehensive analysis of OmniNxt's performance in VIO accuracy (Sec. IV-A) and omnidirectional dense map quality (Sec. IV-B) in the real world. In Sec. IV-C, we demonstrate OmniNxt's ability to autonomously navigate in a narrow indoor environment. In Sec. IV-D, we also adapt OmniNxt with other common visual sensors to ensure its compatibility.

A. *Omni-VINS Evaluation*

We respectively equip OmniNxt with the multi-fisheye camera set and Intel D435 [29] stereo camera to compare the VIO accuracy of omnidirectional VINS (Omni-VINS) and limited-FoV VINS (Stereo-VINS). The ground truth is collected by Opti-Track. In this evaluation, we design three trajectories: **Infinity**, **Cirecle**, and **Random**. In **Infinity** and **Cirecle** cases, OmniNxt follows the trajectory in two manners. One moves without rotating the yaw to the speed direction, and another moves while the yaw follows the speed direction. In the **Random** case, the human pilot controls the robots, randomly performing dashing, spinning, and hovering. Our test environment poses significant challenges to VIO because of the black wall on the right. The testing trajectories and environment are illustrated in Fig. 6.

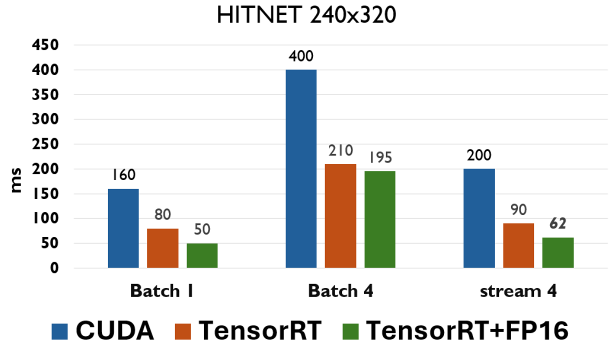

Fig. 7: HITNET inference speed. Batch 1: The inference time of one group of data. Batch 4: The inference time of four groups of data concatenated on batch dimension. Stream 4: The inference time of four groups of data in multi-stream.

Omni-VINS and Stereo-VINS are configured with the same feature extraction and tracking numbers. The sliding window size is set to be 10. Both use good-feature-to-track features and the LK method for tracking. The extrinsic parameters of the camera and IMU in Stereo-VINS are calibrated by Kalibr [22].

Omni-VINS benefits from the omnidirectional FoV, achieving more robust and accurate VIO in all cases. For the root mean squared error (RMSE) of absolute trajectory error (ATE), see TABLE. II (Our evaluate method and tools based on EVO [30]).

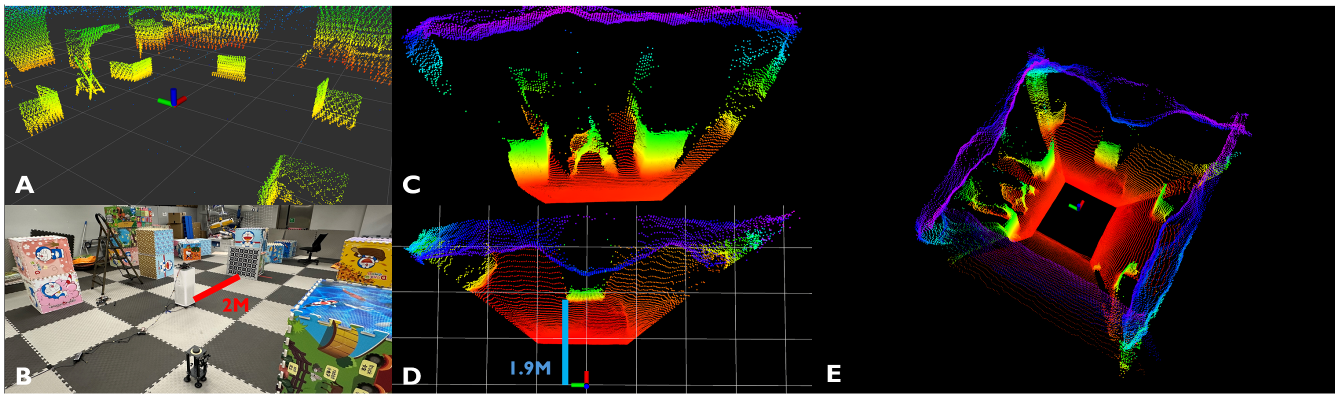

Fig. 8: Omni-Depth evaluation. **A**: the point cloud collected by LIVOX Mid360. **B**: the testing environment. **C and D**: the detailed point cloud generated from the virtual stereo pair DA and CD (grid is 1m) as illustrated in Fig. 5.C. **E**: The overview of the point cloud generated from Omni-Depth. (Sec. IV-B)

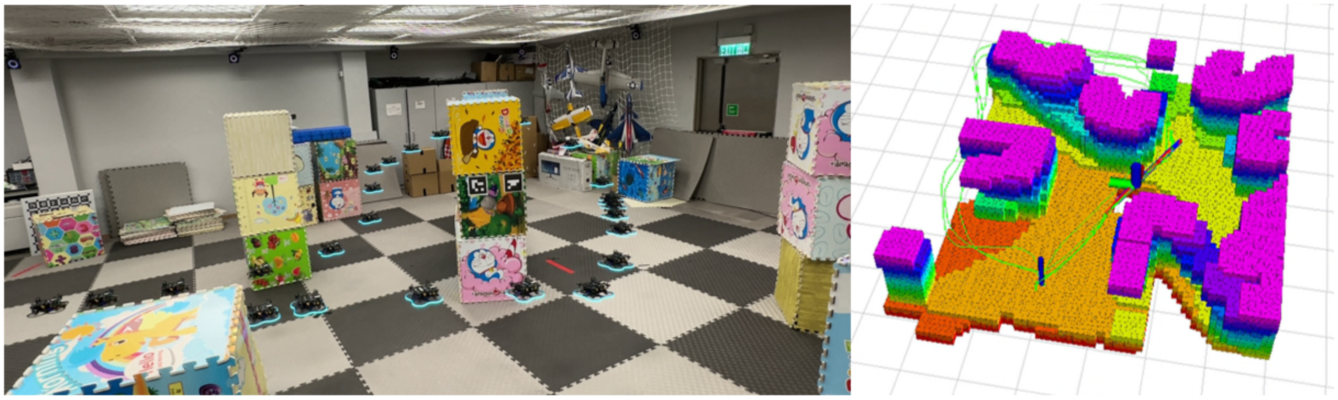
Fig. 9: Real-world autonomous navigation tests. **Left**: OmniNxt autonomously flies without rotating the yaw to the speed direction. **Right**: the executed trajectory and voxel map. (Sec. IV-C)

TABLE II: Localization accuracy comparison (metric: RMSE).

	Omni-VINS	Stereo-VINS	Trajectory Length
Infinity(W)	**0.084m**	0.110m	70.1m
Infinity(W/O)	**0.043m**	0.052m	70.1m
Circle(W)	**0.086m**	0.268m	60.2m
Circle(W/O)	**0.025m**	0.050m	43.0m
Random(W)	**0.098m**	0.460m	60.0m

B. Real-time Depth Estimation

In practice, we use the pre-trained HITNET [31] model with the input dimension (BWHC) $1\times320\times240\times2$ in the backend to balance the estimation quality and inference speed. We accelerate the inference by utilizing Tensor Core and FP16 quantization. The inference time of one virtual stereo drops from 160 ms to 50 ms. Since Omni-Depth involves four sets of inferences for each frame, the sequential execution of these inferences last 200ms. To address this, we explore two approaches to parallelize the inference procedure. Firstly, we concatenate the four groups of input data along the batch dimension, resulting in an input dimension of $4\times320\times240\times2$. However, this approach does not improve the inference time. Then, we adopt NVIDIA multi-stream, which allows four sets of inference to be executed concurrently. Subsequently, the total inference time decreased from 200 ms to 62 ms. Fig. 7 demonstrates the inference speed of HINTET in different manners. The Omni-Depth can run at 15 Hz onboard, providing a real-time dense map. The bias of the dense map is around 10 cm. (See Fig. 8E)

C. Real-world Autonomous Navigation Tests

To demonstrate the practicability of OmniNxt, we conduct autonomous navigation tests in a cluttered indoor environment. Specifically, we modify the trajectory generation strategy in ego-planner [32] by removing the yaw angle trajectory optimization thanks to the omnidirectional perception. This modification enhances the accuracy of localization (as shown in Sec.IV-A) and improves the quality of the dense map by reducing the image blur caused by yaw rotation. This improvement ultimately enhances the efficiency and safety of trajectory planning. For the experiment, we set the maximum speed to 1.0 m/s and the maximum acceleration to 0.6 m/s^2. We represent the surrounding environments with volumetric mapping. All the modules are running onboard. (See Fig. 9)

D. Adaptation of OmniNxt

OmniNxt boasts sophisticated hardware designs that facilitate the swift interchange of visual sensors. Presently, we offer two additional versions of sensor configurations: **Stereo** (Intel D435 [29]) and **RGB-D** (Intel L515 [33]), as depicted in Fig. 10. We have conducted real-world flight experiments to demonstrate their practicality (Stereo in Sec. IV-A, and RGB-D in H_2-Mapping [34]). Detailed information is available on our project page.

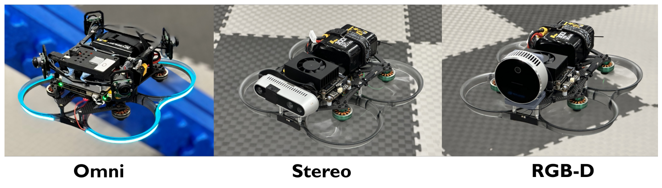

Fig. 10: OmniNxt carries different types of visual sensors. Here, we demonstrate the installation of omnidirectional, stereo, and RGB-D cameras.

V. Conclusion and Future Work

In this paper, we present OmniNxt, a fully open-source and compact aerial robotics platform with omnidirectional visual perception. OmniNxt demonstrates exceptional performance in localization and dense mapping, with limited size and onboard computational resources. We develop Nxt-FC, a coin-size yet high-performance flight controller, which is general to the community. Besides, a multi-fisheye camera set is designed to support omnidirectional perception. Based on our devised hardware, we also propose a real-time omnidirectional perception framework, including Omni-VINS and Omni-Depth. Comprehensive real-world experiments demonstrated the superiority and practicability of OmniNxt. In the future, we aim to improve the capability of omnidirectional visual perception to enable exceptional performance in increasingly demanding environments.

References

[1] B. Zhou, Y. Zhang, X. Chen, and S. Shen, "FUEL: Fast UAV Exploration using Incremental Frontier Structure and Hierarchical Planning," 2020.

[2] C. Feng, H. Li, J. Jiang, X. Chen, S. Shen, and B. Zhou, "FC-Planner: A Skeleton-guided Planning Framework for Fast Aerial Coverage of Complex 3D Scenes," *arXiv preprint arXiv:2309.13882*, 2023.

[3] E. Lygouras, A. Gasteratos, K. Tarchanidis, and A. Mitropoulos, "ROLFER: A fully autonomous aerial rescue support system," *Microprocessors and Microsystems*, vol. 61, pp. 32–42, 2018.

[4] K. Mohta, M. Watterson, Y. Mulgaonkar, S. Liu, C. Qu, A. Makineni, K. Saulnier, K. Sun, A. Zhu, J. Delmerico, K. Karydis, N. Atanasov, G. Loianno, D. Scaramuzza, K. Daniilidis, C. J. Taylor, and V. Kumar, "Fast, autonomous flight in GPS-denied and cluttered environments," *Journal of Field Robotics*, vol. 35, no. 1, pp. 101–120, 2018.

[5] T. Baca, M. Petrlik, M. Vrba, V. Spurny, R. Penicka, D. Hert, and M. Saska, "The MRS UAV System: Pushing the Frontiers of Reproducible Research, Real-world Deployment, and Education with Autonomous Unmanned Aerial Vehicles," *Journal of Intelligent and Robotic Systems*, vol. 102, no. 1, Apr. 2021.

[6] P. Foehn, E. Kaufmann, A. Romero, R. Penicka, S. Sun, L. Bauersfeld, T. Laengle, G. Cioffi, Y. Song, A. Loquercio, and D. Scaramuzza, "Agilicious: Open-source and open-hardware agile quadrotor for vision-based flight," *Science Robotics*, vol. 7, no. 67, p. eabl6259, 2022.

[7] W. Gao, K. Wang, W. Ding, F. Gao, T. Qin, and S. Shen, "Autonomous aerial robot using dual-fisheye cameras," *Journal of Field Robotics*, vol. 37, no. 4, pp. 497–514, 2020.

[8] Z. Zhang, H. Rebecq, C. Forster, and D. Scaramuzza, "Benefit of large field-of-view cameras for visual odometry," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 801–808.

[9] Z. Wang, K. Yang, H. Shi, P. Li, F. Gao, and K. Wang, "LF-VIO: A Visual-Inertial-Odometry Framework for Large Field-of-View Cameras with Negative Plane," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[10] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 2520–2525.

[11] DJI. [Online]. Available: https://enterprise.dji.com/zh-tw/mavic-3-enterprise/

[12] SKYDIO. [Online]. Available: https://www.skydio.com/x10/

[13] D. He, W. Xu, N. Chen, F. Kong, C. Yuan, and F. Zhang, "Point-LIO: Robust High-Bandwidth Light Detection and Ranging Inertial Odometry," *Advanced Intelligent Systems*, vol. 5, no. 7, p. 2200459, 2023.

[14] Fast-LAB. [Online]. Available: https://github.com/ZJU-FAST-Lab/Fast-Drone-250

[15] H. Matsuki, L. von Stumberg, V. Usenko, J. Stückler, and D. Cremers, "Omnidirectional DSO: Direct Sparse Odometry with Fisheye Cameras," *CoRR*, vol. abs/1808.02775, 2018.

[16] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[17] C. Won, J. Ryu, and J. Lim, "End-to-End Learning for Omnidirectional Stereo Matching with Uncertainty Prior," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2020.

[18] S. Xie, D. Wang, and Y. Liu, "Omnividar: Omnidirectional depth estimation from multi-fisheye images," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jun 2023, pp. 21 529–21 538.

[19] L. Meier, D. Honegger, and M. Pollefeys, "Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms," *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6235–6240, 2015.

[20] B. P. Duisterhof, Y. Hu, S. H. Teng, M. Kaess, and S. Scherer, "TartanCalib: Iterative Wide-Angle Lens Calibration using Adaptive SubPixel Refinement of AprilTags," 2022.

[21] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1280–1286.

[22] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, "Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 4304–4311.

[23] H. Xu, P. Liu, X. Chen, and S. Shen, "D^2SLAM: Decentralized and Distributed Collaborative Visual-inertial SLAM System for Aerial Swarm," 2023.

[24] C. Mei and P. Rives, "Single view point omnidirectional camera calibration from planar grids," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 2007, pp. 3945–3950.

[25] E. Plaut, E. B. Yaacov, and B. E. Shlomo, "3D Object Detection from a Single Fisheye Image Without a Single Fisheye Training Image," 2021.

[26] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning Feature Matching with Graph Neural Networks," 2020.

[27] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision (ijcai)," vol. 81, 04 1981.

[28] S. Agarwal, K. Mierle, and T. C. S. Team, "Ceres Solver," 10 2023. [Online]. Available: https://github.com/ceres-solver/ceres-solver

[29] INTEL. [Online]. Available: https://www.intelrealsense.com/depth-camera-d435/

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

[31] V. Tankovich, C. Häne, Y. Zhang, A. Kowdle, S. Fanello, and S. Bouaziz, "HITNet: Hierarchical Iterative Tile Refinement Network for Real-time Stereo Matching," 2023.

[32] X. Zhou, Z. Wang, H. Ye, C. Xu, and F. Gao, "EGO-Planner: An ESDF-free Gradient-based Local Planner for Quadrotors," 2020.

[33] INTEL. [Online]. Available: https://www.intelrealsense.com/lidar-camera-l515/

[34] C. Jiang, H. Zhang, P. Liu, Z. Yu, H. Cheng, B. Zhou, and S. Shen, "H_2-mapping: Real-time dense mapping using hierarchical hybrid representation," *IEEE Robotics and Automation Letters*, vol. 8, no. 10, pp. 6787–6794, 2023.