

Pixel-aligned RGB-NIR Stereo Imaging and Dataset for Robot Vision

Supplemental Document

Jinnyeong Kim Seung-Hwan Baek
POSTECH

In this supplemental document, we provide additional results and details in support of our findings in the main manuscript.

Contents

1. Experimental Prototype	3
1.1. Hardware Stack	3
1.1.1 Imaging Setup	3
1.1.2 Mobile Robot	4
1.1.3 Computational Link and Power Supply	4
1.1.4 Calibration	4
1.2. Software Stack	5
1.2.1 Sensor SDK Implementation	5
1.2.2 Imaging System Pipeline Architecture	5
2. Pixel-aligned RGB-NIR Datasets	6
2.1. Real Dataset	6
2.1.1 Details on Acquisition	6
2.1.2 Samples of Real Dataset	6
2.1.3 Statistics	6
2.2. Synthetic Dataset	9
2.2.1 Synthetic Dataset Augmentation	9
2.2.2 Samples of Synthetic Dataset	11
2.3. Comparison to Other Datasets	11
3. Details on RGB-NIR Feature Fusion and Attentional Fusion	12
3.1. RGB-NIR Image Fusion	12
3.1.1 Network Architecture	12
3.1.2 Loss Function	16
3.1.3 Training Details	16
3.2. RGB-NIR Feature Fusion	16
3.2.1 Network Architecture	16
3.2.2 Loss Function	17
3.2.3 Training Detail	17
4. Additional Result	18
4.1. Implementation of RGB-NIR Image Fusion Methods for Comparative Analysis	18
4.2. RGB-NIR Image Fusion for Object Detection	21
4.2.1 Evaluation	21
4.2.2 Additional Results	21
4.3. RGB-NIR Image Fusion for Structure-from-Motion	22

4.4. Sparse Depth Reconstruction	23
4.5. RGB-NIR Fusion for Depth Estimation	23
4.5.1 Implementation of RGB-NIR Stereo Depth Estimations for Comparative Analysis	23
4.5.2 Evaluation Metrics	24
4.5.3 Additional Quantitative Results	24
4.5.4 Additional Qualitative Samples	24
4.6. Additional Ablation Study	24
4.6.1 Pretrained Feature Encoder Weights	24
4.6.2 Ablation on Feature Fusion Implementation	25

1. Experimental Prototype

1.1. Hardware Stack

1.1.1 Imaging Setup

Table 1 summarizes the detailed specifications of the sensors used in our experimental prototype. To achieve RGB-NIR multispectral imaging and 3D geometric reconstruction, we implemented two pixel-aligned RGB-NIR cameras and a LiDAR system.

Sensor	Quantity	Product	Resolution
RGB-NIR Camera	2	JAI FS-1600D-10GE	180Hz 8bit 1440x1080 BayerRG image 180Hz 8bit 1440x1080 NIR image
LiDAR	1	Ouster OS-1	20Hz 2048x128 point cloud
IMU	1	Ouster OS-1 IMU	100Hz inertial data

Table 1. Sensor specs of our imaging system.

RGB-NIR Stereo Camera Setup The RGB-NIR cameras (JAI FS-1600D-10GE) leverage a dichroic prism to simultaneously capture visible (RGB) and near-infrared (NIR) images, offering distinct advantages in robust feature extraction under varying illumination conditions. This dual-spectrum imaging capability facilitates applications such as material classification, vegetation analysis, and object detection in low-light environments.

Our system integrates two RGB-NIR cameras connected via RJ-45 interfaces using Ethernet cables. These cameras support high frame rates and high-resolution image acquisition, with performance primarily determined by the transmission link speed. The JAI FS-1600D-10GE officially achieves up to 100 fps for RGB-NIR pixel-aligned imaging when operating on a 10 Gbps Ethernet connection.

NIR Active Illumination To enhance imaging in the NIR spectrum, an active illumination source, Advanced Illumination AL295-150850IC, emitting at an 850 nm wavelength, was employed. Table 2 shows detail specification of the NIR illumination. This illumination compensates for ambient lighting variability and improves the quality of the NIR channel, particularly in controlled or low-light environments.

Specification	Parameter
Length	165.6 mm
Weight	68.9 g
Wavelength	850 nm
Photobiological Risk Factor	Exempt (850 nm)
Operating Temperature	0 - 60°C
Compliance	CE, RoHS, IEC 62471
IP Rating	IP50
Lumen Maintenance	L70 = 50,000 Hours

Table 2. Specifications of the active illumination source (AL295-150850IC).

3D LiDAR for Depth Ground Truth We use the Ouster OS-1 LiDAR to obtain accurate depth ground truth. This LiDAR supports up to 2048 samples with 128 channels, providing a depth resolution of 2048×128 for a full 360-degree rotation. The LiDAR can achieve a maximum frame rate of 20 fps at 1024×128 resolution and 10 fps at 2048×128 resolution. We utilized the 20 fps configuration for high-frequency datasets and the 10 fps configuration for lower-frequency datasets. The built-in inertial measurement unit (IMU) measures angular velocity (radians/second) and linear acceleration (G) along the x, y, and z axes at up to 100 Hz, offering additional data for refining LiDAR point clouds. The LiDAR operates using 865 nm structured light, which interacts with NIR cameras and may be affected by external NIR illumination. Nevertheless, advanced features, including multi-sensor crosstalk suppression and programmable settings, mitigate such interference, ensuring high-accuracy depth measurements under challenging illumination conditions.

1.1.2 Mobile Robot

Our imaging system is mounted on a mobile wheeled robot, the Agile-X Ranger Mini 2.0, which provides stable and efficient operation for large-scale data collection in both indoor and outdoor environments. The robot is equipped with a 4-wheel drive (4WD) system and features wheels capable of 180-degree rotation, offering exceptional maneuverability and the capability to navigate tight and complex spaces.

Table 3 shows specifications of the mobile robot. The Agile-X Ranger Mini 2.0 has a compact design with overall dimensions of 738mm \times 500mm \times 338mm and an axle track of 494 mm, enabling smooth traversal across various terrains. It is powered by four 48 V brushless toothed motors, each delivering a rated power of 600 W and a torque of 22 Nm. The robot achieves a maximum speed of 2.6m/s and can climb inclines up to 10° while carrying a maximum load of 150kg. To accommodate different operational needs, the Ranger Mini 2.0 offers two battery configurations: a single battery setup providing 2–8 hours of operation and a multi-battery configuration supporting extended endurance. The lithium-ion batteries can be charged in as little as 1 hour, ensuring minimal downtime. The robot’s advanced suspension system and independent 4-wheel steering enable optimal performance on uneven surfaces and during high-precision maneuvers.

Our setup leverages the Ranger Mini 2.0’s versatility to conduct imaging tasks across diverse settings, including roads, sidewalks, and interior spaces, ensuring comprehensive data collection for research in urban and natural environments. Its rugged design and IP54 rating make it suitable for challenging outdoor conditions while maintaining high stability and reliability for precise imaging.

Specification	Parameter
Maximum Payload	150 kg
Maximum Speed	2.6 m/s
Control Mode	Remote controller / ROS
Steering Type	4-wheel steering
Turning Radius	0 mm (Spin mode) / 810 mm (Ackermann mode)
Battery Life	2–8 hours
Charging Time	1 hour

Table 3. Specifications of Agile-X Ranger Mini 2.0.

1.1.3 Computational Link and Power Supply

To manage the capture pipeline efficiently, we integrated a high-performance laptop (Asus ROG Zephyrus G14) equipped with an Nvidia RTX 4060 GPU (8GB). This setup provides the necessary computational power for real-time data processing and management of our imaging and LiDAR systems. Since both the RGB-NIR cameras and the LiDAR connect via RJ-45 interfaces, we utilized an RJ-45 network switch hub to extend connectivity, accommodating up to 10 devices. This ensures seamless data transfer and system integration, despite the laptop’s limited number of Ethernet ports. Furthermore, to support the power demands of the entire imaging system, we mounted an external AC power bank onto the robot. This setup allows our system to operate continuously for over 3 hours without the need for recharging, ensuring sustained data collection in diverse environments.

1.1.4 Calibration

Stereo Pose Calibration To calibrate a stereo camera system comprising left and right cameras, a standard chessboard pattern is used as a known reference object. The calibration process begins by capturing a series of images of the chessboard from both cameras at multiple orientations. Using these images, point correspondences between the observed 2D chessboard corners in each camera image and the known 3D coordinates of the corners in the chessboard’s coordinate system are established. The intrinsic parameters K_{left} and K_{right} for the left and right cameras, respectively, are estimated through this correspondence, capturing the focal length and principal point of each camera. Next, the relative pose between the left and right cameras is determined. The extrinsic parameters E_{right} , comprising the rotation and translation that map 3D points from the left camera coordinate system to the right camera coordinate system, are computed. This calibration is essential for rectifying stereo image pairs and for accurate 3D reconstruction. By using well-established algorithms [2], the intrinsic matrices K_{left} , K_{right} , and the extrinsic transformation E_{right} are accurately estimated.

LiDAR Pose Calibration To calibrate the LiDAR and left camera, a well-structured real 3D scene is designed and captured to facilitate precise point correspondences between the LiDAR and camera images. Specifically, the left RGB image captured by the camera and the plane image generated from the LiDAR are analyzed to manually annotated corresponding points across both modalities. Each LiDAR coordinate, $(x_{\text{LiDAR}}, y_{\text{LiDAR}}, z_{\text{LiDAR}})$, is mapped into the camera coordinate system as $(x_{\text{camera}}, y_{\text{camera}}, z_{\text{camera}})$ using a transformation matrix $M_{\text{LiDAR} \rightarrow \text{camera}}$. The intrinsic camera matrix K then projects these 3D points onto the 2D image plane, yielding image coordinates (u, v) , as expressed by the following equation:

$$\begin{bmatrix} \hat{u} \\ \hat{v} \\ z_{\text{left}} \end{bmatrix} = K_{\text{left}} M_{\text{LiDAR} \rightarrow \text{camera}} \begin{bmatrix} x_{\text{LiDAR}} \\ y_{\text{LiDAR}} \\ z_{\text{LiDAR}} \\ 1 \end{bmatrix}, \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \frac{\hat{u}}{z_{\text{left}}} \\ \frac{\hat{v}}{z_{\text{left}}} \end{bmatrix} \quad (1)$$

By collecting a sufficient number of $(x_{\text{LiDAR}}, y_{\text{LiDAR}}, z_{\text{LiDAR}})$ and (u, v) correspondences and leveraging the known intrinsic parameters K , the transformation matrix $M_{\text{LiDAR} \rightarrow \text{camera}}$ can be estimated. This estimation process can be formulated as a Perspective-n-Point (PnP) problem, which seeks to determine the camera’s extrinsic parameters (rotation R and translation t) by minimizing the reprojection error between a set of 3D points and their corresponding 2D image projections. The solution to the PnP problem, including the estimation of $M_{\text{LiDAR} \rightarrow \text{camera}}$ follows the approach described in [31], which employs a linear formulation to efficiently estimate the transformation while minimizing the reprojection error. Advanced solvers or iterative methods such as RANSAC can also be applied to enhance robustness in the presence of outliers.

Timing Calibration The JAI MultiSpectral Camera utilized in our system features Precision Timing Protocol (PTP), enabling time measurement with nanosecond-level accuracy. Internally, RGB CMOS and NIR CMOS of a camera are synchronized using PTP. As we employed two separate cameras, it was essential to achieve synchronization between them. To this end, we activated a PTP-based Pulse Generator on each camera, which emits a trigger signal at precisely defined periods. By adjusting the delay settings of the Pulse Generators, we compared timestamps recorded in the response packets of both cameras to align them. This approach ensured that the synchronization between the cameras remained within 100 microseconds. Such a low level of time difference is negligible for 3D vision applications, eliminating the need for further temporal correction in downstream tasks. Additionally, as a precautionary guideline, we discarded any packets that exhibited a timing difference exceeding 1 millisecond.

1.2. Software Stack

1.2.1 Sensor SDK Implementation

JAI Fusion Camera The JAI Fusion camera utilizes the eBUS SDK, which enables configuration of Ethernet-connected cameras and the creation of receiver sockets for the camera’s data streams. Specifically, since this camera has two independent CMOS sensors within a single unit, configuring it as a stereo camera requires handling data from four separate streams. We integrated the C++ SDK library into our CMake project, setting up four distinct ports to create separate stream receiver objects. Each of these receiver objects operates on an independent thread, ensuring smooth data acquisition without blocking. The main thread monitors the timing of the collected image packets from the four streams, grouping them into a single frame if the timestamp difference is less than 1 ms. Once a complete frame is formed, it is published as a ROS2 topic for use by other processes. To minimize the time difference between the stereo cameras, we adjust the delay of the internal pulse generator of the cameras, based on the timing discrepancies measured for each frame.

Ouster OS-1 The Ouster LiDAR sensor is interfaced using its native Python3 SDK library. The LiDAR device is assigned a unique IP address, and its data packets are combined into a 360-degree scan using the SDK’s packet integration tool. However, this integration tool does not accumulate IMU packets. To address this, we extended the packet integration functionality of the SDK, enabling it to integrate accumulated IMU packets into the 360-degree scanning data.

1.2.2 Imaging System Pipeline Architecture

Our imaging system operates on Ubuntu 22.04, utilizing Python 3.12 and ROS2 Humble to provide a robust and efficient platform for multi-sensor data acquisition and processing. The software integrates SDKs for stereo cameras and LiDAR sensors, facilitating seamless file I/O operations and an intuitive UX/UI for user interaction. The system is modularized

into four primary components: Capture Trigger, Response Queue, Storage Management, and User Interface/User Experience (UI/UX).

The Capture Trigger module orchestrates the periodic activation of connected sensors, ensuring synchronized data acquisition from stereo cameras and LiDAR. It maintains precise time synchronization between the stereo cameras, enabling consistent stereo image capturing essential for depth estimation. The LiDAR module processes incoming LiDAR packets, aggregating them into a cohesive 360-degree point cloud representation. Data from the sensors are funneled into the Response Queue, which manages the incoming RGB and LiDAR data streams. This module identifies and pairs sensor data with minimal time discrepancies, adhering to a predefined time difference threshold to maintain data integrity. Once paired, the data is forwarded to the Storage Management module, which handles the efficient local storage of synchronized multi-sensor information. The UI/UX component provides users with a streamlined interface to control the imaging process. A single-button interface allows users to toggle sequential frame capturing (video recording) effortlessly, enabling them to concentrate on the robot's navigation tasks.

The system, deployed on a robot-mounted laptop, supports capturing up to 20 frames per second. When real-time stereo depth estimation is activated, the system maintains a capture rate of up to 10 depth images per second, balancing performance and computational demands. This software architecture ensures reliable synchronization, efficient data management, and user-friendly operation, making it well-suited for real-time robotic applications requiring high-fidelity multi-sensor imaging.

2. Pixel-aligned RGB-NIR Datasets

2.1. Real Dataset

2.1.1 Details on Acquisition

We captured 720×540 resolution BayerRG stereo images using JAI Fusion camera, restoring them to 3-channel RGB images via Bayer interpolation before saving. Simultaneously, 720×540 mono-channel NIR images were captured using JAI FS-1600D-10GE and saved alongside the RGB images. Our dataset comprises 28 video sequences captured at 10 Hz and 15 sequences at 5 Hz. Each video ranges from 100 to 6000 frames. Longer videos encompass continuous transitions through various environments and can be partitioned into shorter clips for specific applications. Data collection occurred across diverse settings, including both indoor spaces like laboratories and lecture halls, and outdoor areas such as courtyards and walkways. Videos were recorded at different times of the day—morning, noon, evening, and night—to capture a wide range of lighting conditions. Some sequences feature transitions between indoor and outdoor environments, while others include dynamic scenarios like a vehicle entering an indoor parking garage from outside. Weather conditions during recording varied from sunny to overcast, adding to the dataset's versatility. Our compact and lightweight camera setup eliminates the need for bulky beam splitters, allowing for extensive and flexible data collection without compromising the field of view. This design facilitates large-scale indoor and outdoor recording sessions, making the dataset suitable for applications in machine vision, autonomous navigation, and robotics.

2.1.2 Samples of Real Dataset

Figure 1 presents samples from our acquired real dataset, which includes RGB stereo images, NIR stereo images, and sparse LiDAR points. Figure 2 showcases our dataset under challenging lighting conditions, such as nighttime and poorly lit indoor environments. We collected over 90,000 frames across diverse settings, including indoor and outdoor environments, bright and dark locations, as well as roads and sidewalks. By capturing data across a wide range of real-world environments and times of day, our dataset is expected to support not only depth estimation but also 3D geometry reconstruction techniques, such as structure-from-motion [34], and various photometric volume reconstruction methods, including Gaussian splatting [20].

2.1.3 Statistics

Figure 3 presents a comprehensive analysis of our dataset. (a) illustrates the proportion of the dataset categorized into 6 distinct lighting conditions: well-lit, dark and complex, which are further detailed as either indoor or outdoor. (b) provides a histogram that depicts the distribution of video scenes based on the number of frames. (c), (d), and (e) show histograms of the exposure times for RGB and NIR cameras under well-lit outdoor, dark outdoor, and Indoor settings, respectively. Notably, due to the independent operation of auto-exposure for the RGB and NIR sensors, the exposure time distributions differ between the two modalities. Graph (f), (g), and (h) further classify scenes based on additional criteria. (f) evaluates the

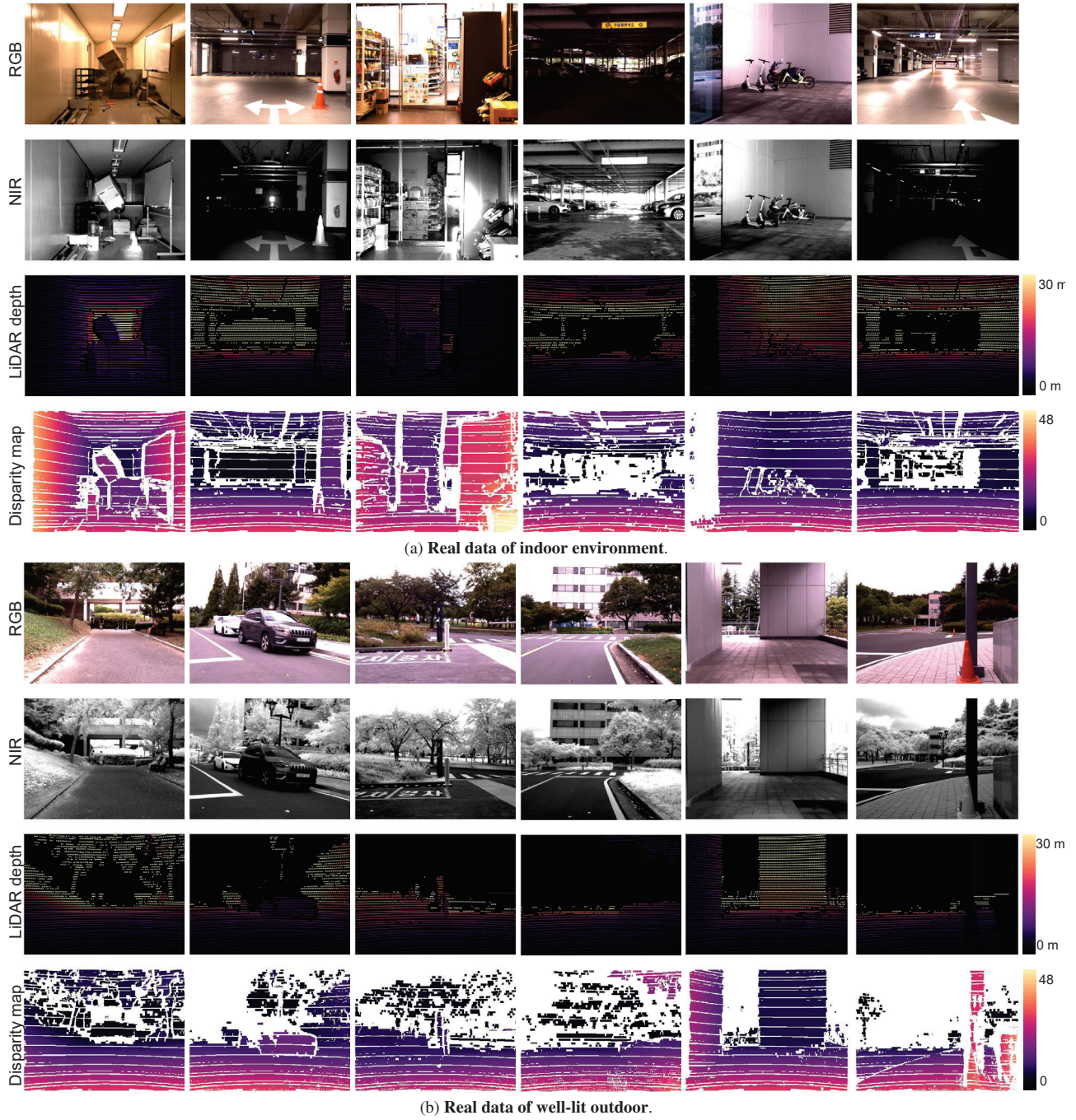
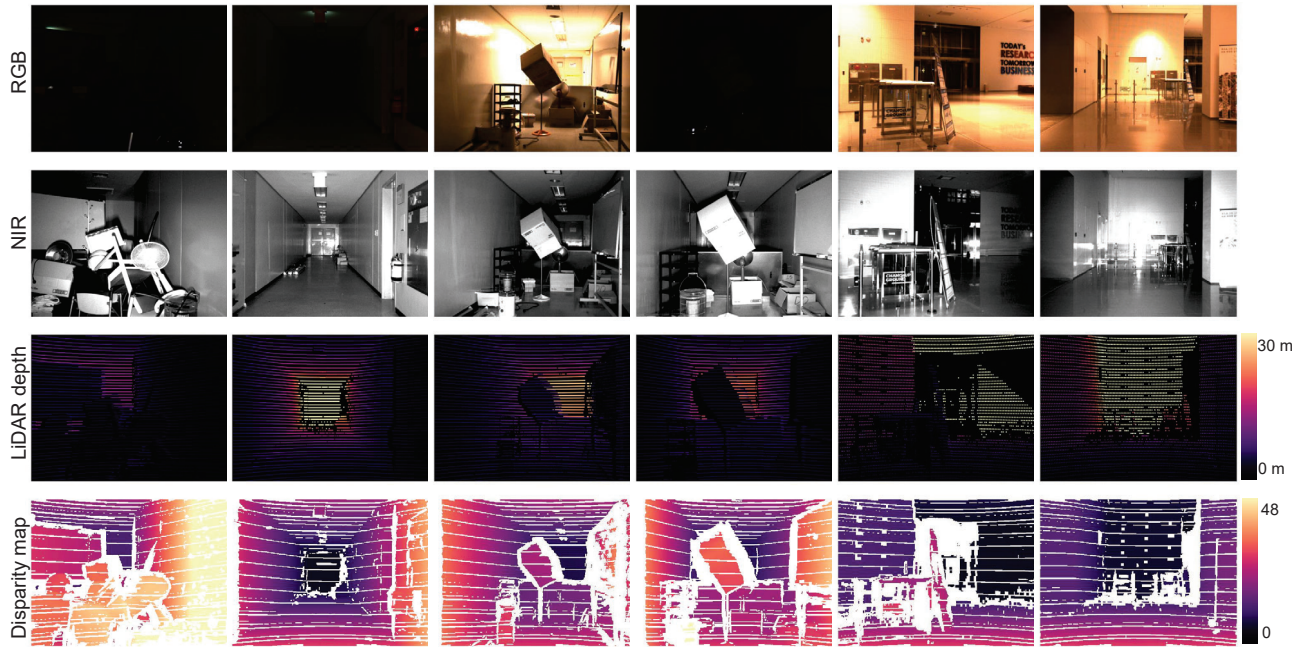
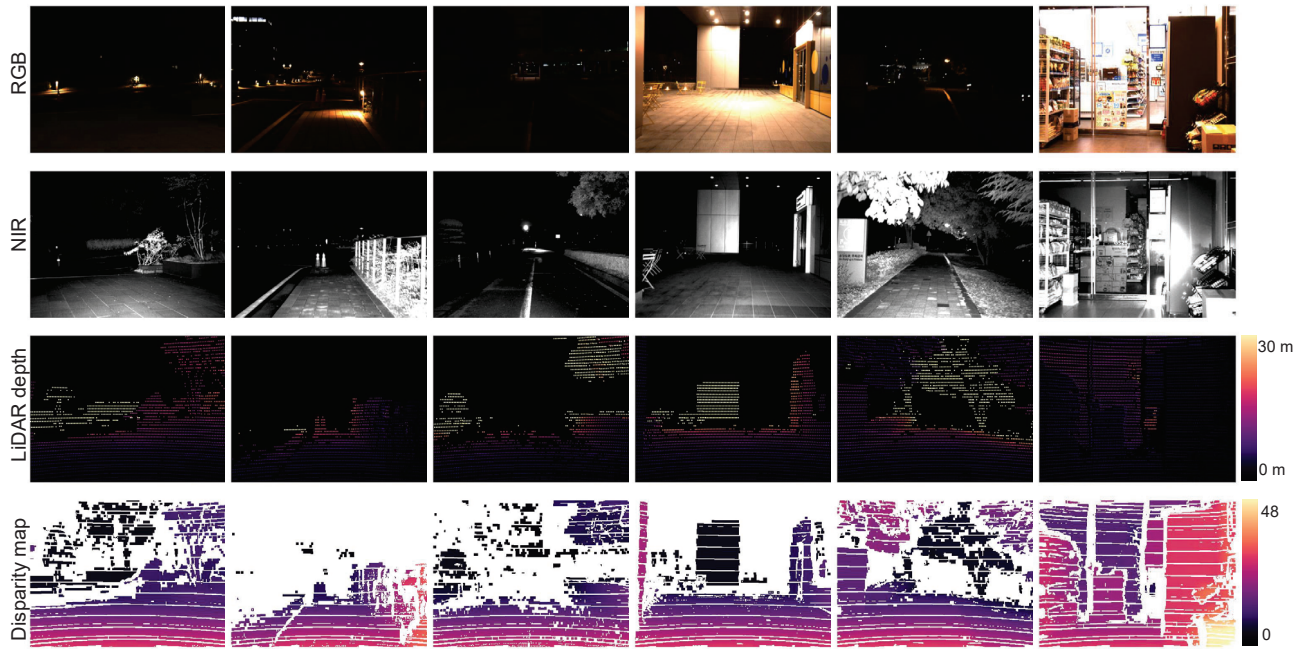


Figure 1. Various real dataset samples

presence of high-specular scenes and overexposure conditions, while (g) categorizes scenes as either indoor or outdoor. (h) details the diverse environments encountered, including road driving, pedestrian pathways, transitions between indoor and outdoor settings, and underground roads. These scenes are further subdivided into Well-Lit, Dark, and Dynamic Lighting conditions. Importantly, the bar graphs in (f), (g), and (h) illustrate that a single scene can belong to multiple categories simultaneously, highlighting the complexity and diversity of our dataset.



(a) Real data of indoor environment with challenging lighting.



(b) Real data of dark outdoor.

Figure 2. Various real dataset samples on challenging lighting condition.

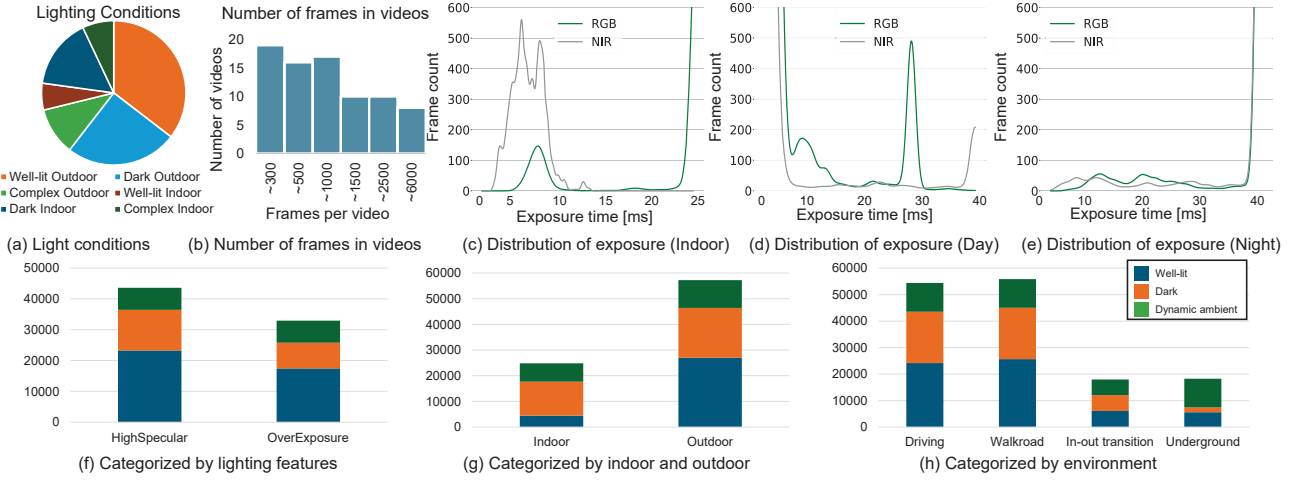


Figure 3. Statistics of our real dataset.

2.2. Synthetic Dataset

2.2.1 Synthetic Dataset Augmentation

Augmentation from Image Formation Equation The image formation of our RGB-NIR pixel aligned camera is denoted as following:

$$I_i^c(p^c) = \eta_1 + g_i(\eta_2 + t_i(R_i^c(p^c)(E_i^c(p^c) + L_i^c(p^c))))). \quad (2)$$

To facilitate training of other vision tasks such as stereo depth estimation, it is essential to have a training dataset with precise ground-truth information, which can be achieved by creating a synthetic dataset. However, generating a large-scale synthetic dataset from scratch—including rendering 3D scenes and extending the three-channel color space to four channels (R, G, B, NIR)—requires substantial resources that may not be readily available. Therefore, we utilized existing large-scale RGB stereo datasets [29, 33] and developed a synthetic rendering pipeline leveraging an image formation model to generate realistic RGB images based on provided depth maps and material segmentation maps.

Figure 4 shows image components in augmentation pipeline. The baseline dataset provides RGB images (a), depth (b) and material index (c). The albedo $R_{i \in \{R, G, B\}}^c(p^c)$ (Fig. 4(d)) is accurately simulated by assigning distinct reflectance values to each material class identified in the segmentation map, thereby ensuring material-specific color representation. NIR albedo $R_{i \in \{NIR\}}^c(p^c)$ (Fig. 4(f)) is pseudo driven from $R_{i \in \{R, G, B\}}^c(p^c)$ by [13].

Normal Map Reconstruction To accurately compute the lighting interactions, normal maps (Fig. 4(e)) are derived from the depth maps (Fig. 4(b)), enabling precise calculation of the incident angles between the light sources and the surface normals for each pixel. The conversion of a depth map into a normal map involves calculating the gradients in the x and y directions to determine the surface normals. This process starts by computing the partial derivatives of the depth values, which are then used to construct the normal vector at each point. Specifically, the normal vector \mathbf{N} can be derived as

$$\mathbf{N} = \left(-\frac{\partial z}{\partial x}, -\frac{\partial z}{\partial y}, z \right), \quad (3)$$

where x, y, z consist of point cloud on camera coordinate, driven by projecting depth map into camera coordinate, followed by normalization to ensure unit length. Detail implementation follows [17].

Ambient Lighting Ambient lighting $E_i^c(p)$ is introduced to emulate diverse environmental illumination conditions by utilizing multiple ambient light sources. Each ambient light source is characterized by its unique position and brightness, contributing cumulatively to the overall ambient illumination at each pixel. Specifically, the ambient lighting is modeled as the sum of contributions from n ambient light sources, as described by the following equation:

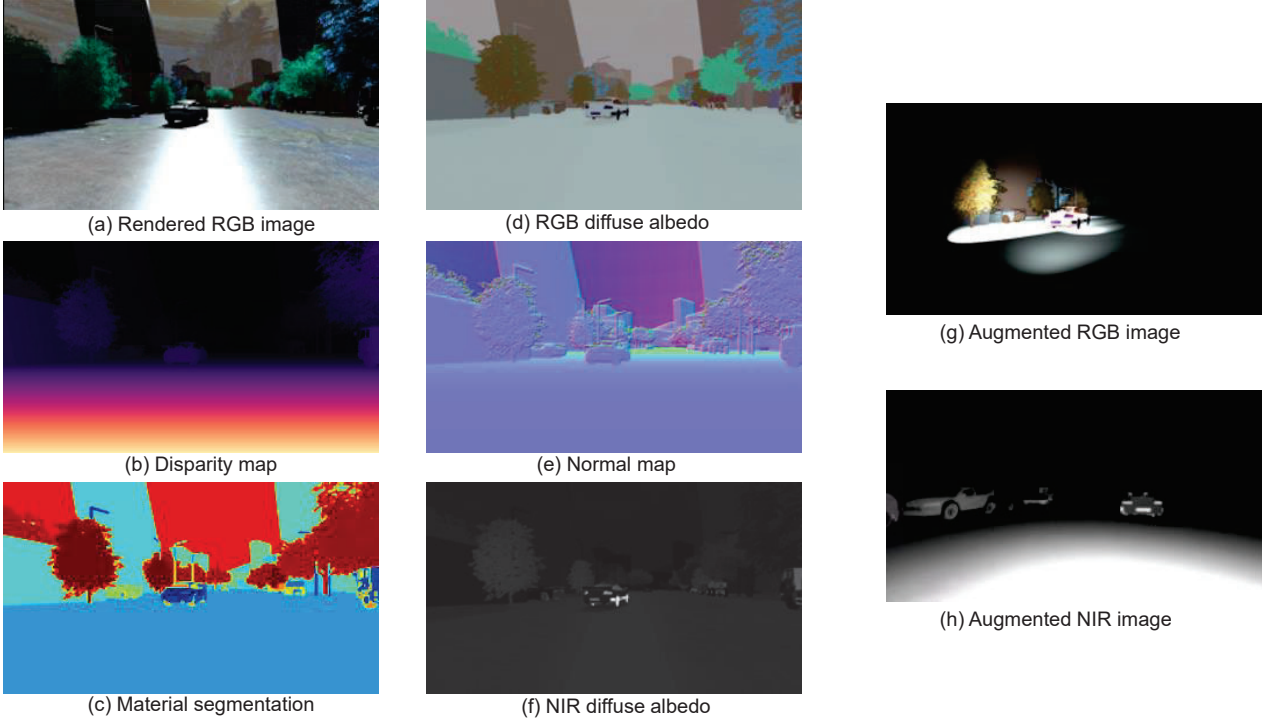


Figure 4. **RGB-NIR synthetic data augmentation.** Sceneflow dataset [29] provide (a) rendered RGB image, (b) disparity map and (c) material index. We assumed RGB albedo (d) with (a) & (b), normal map (e) from (b). We simulated NIR albedo (f) from (d) by [13]. We rendered some light source for RGB (g) and NIR (h).

$$E_i^c(p) = \sum_{j=1}^n \phi_j \cdot \max(0, \mathbf{N}(p) \cdot \mathbf{L}_j(p)), \quad (4)$$

where

- ϕ_j denotes the brightness of the j -th ambient light source.
- $\mathbf{N}(p)$ represents the normal vector at pixel p , derived by 3.
- $\mathbf{L}_j(p)$ is the unit vector pointing from the surface point p to the position of the j -th ambient light source.

This formulation allows for realistic simulation of ambient lighting by accounting for the direction and intensity of multiple light sources, thereby enhancing the visual fidelity of the rendered images.

Active Illumination Active illumination $L_i^c(p)$ is incorporated to provide consistent direct illumination from a single, fixed light source. Unlike ambient lighting, active illumination affects only the Near-Infrared (NIR) channel, leaving the Red (R), Green (G), and Blue (B) channels unaffected. This selective illumination is particularly useful for applications requiring multi-spectral data. The active lighting is modeled without summation, as only one active light source is present, and is defined as follows:

$$L_i^c(p) = \begin{cases} 0, & \text{for } i \in \{R, G, B\} \\ \phi_{\text{active}} \cdot \max(0, \mathbf{N}(p) \cdot \mathbf{L}_{\text{active}}(p)), & \text{for } i = \text{NIR} \end{cases} \quad (5)$$

where:

- ϕ_{active} denotes the brightness of the active light source.
- $\mathbf{L}_{\text{active}}(p)$ is the unit vector pointing from the surface point p to the fixed position of the active light source.
- The term $\max(0, \mathbf{N}(p) \cdot \mathbf{L}_{\text{active}}(p))$ ensures that only positive contributions to the illumination are considered, adhering to the Lambertian reflectance model.

By restricting active illumination to the NIR channel and maintaining a fixed light position and intensity, the model ensures that direct illumination is consistently applied without altering the RGB channels.

The image formation equation incorporates fixed exposure time t_i and gain g_i parameters, which are held constant throughout the simulation to maintain uniform exposure settings. Gaussian noise is systematically added both pre- and post-processing to emulate realistic sensor noise, thereby enhancing the fidelity of the synthetic images. This comprehensive approach integrates material properties, complex lighting interactions, and realistic noise modeling, resulting in high-quality synthetic renderings that are suitable for various applications in computer vision and graphics research.

2.2.2 Samples of Synthetic Dataset

Figure 5 presents samples from our augmented synthetic dataset. We enhanced two distinct environments from the SceneFlow dataset [29]: a driving scene and an indoor environment featuring randomly flying objects. The augmented dataset includes both RGB stereo and Near-Infrared (NIR) stereo images, utilizing existing components of the original dataset such as RGB stereo pairs and disparity maps. This augmented dataset was employed to train a stereo depth estimation network using the original disparity map labels and an image fusion model with the original RGB images.

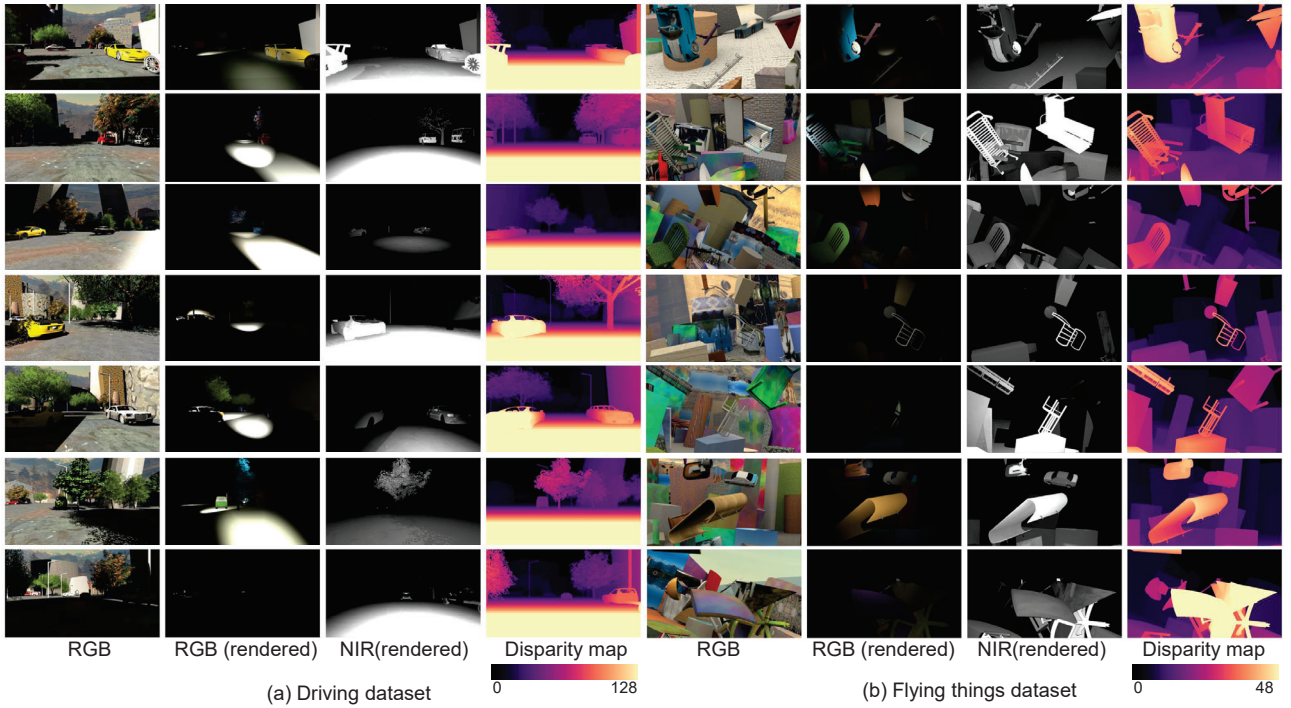


Figure 5. **Samples of augmented synthetic dataset.** SceneFlow [29] includes both outdoor environment (a) and indoor environment (b) dataset. We augmented these datasets relighting RGB stereo images and NIR stereo images.

2.3. Comparison to Other Datasets

We compared our dataset with existing large-scale datasets, including outdoor RGB stereo depth datasets [7, 12], indoor RGB stereo depth datasets [8, 9], datasets with RGB and NIR [3, 5, 30, 36, 38, 41, 42, 45], and RGB-Thermal datasets [6, 14, 21, 28]. The comparison is presented in Table 4, based on several key criteria which are detailed below.

Pixel-aligned RGB-NIR Multispectral datasets primarily include different spectral image information alongside RGB. When two spectral images are pixel-aligned, they can be effectively fused without the need for pose correction, which greatly enhances efficiency. This pixel-level alignment is crucial for applications that leverage multispectral fusion.

Dataset	Pixel-aligned	Multi-View	RGB Stereo	NIR Stereo	GT Depth	Lidar Depth	Video	Spectral Bands	Indoor	Outdoor	Day	Night	Base Platform	Pixel-align Implementation
[12]	X	O	O	X	O	O	O	RGB	X	O	O	O	Vehicle	N/A
[8]	X	O	X	X	O	X	O	RGB	O	X	O	X	Hand-carried	N/A
[7]	X	O	O	X	O	O	O	RGB	X	O	O	O	Vehicle	N/A
[41]	O	X	X	X	X	X	X	RGB, NIR, Thermal	X	O	O	O	N/A	Beam splitter
[28]	Thermal	O	O	X	O	O	O	RGB, Thermal	X	O	O	O	Vehicle	Beam splitter
[9]	X	O	X	X	O	X	O	RGB	O	X	O	X	Hand-carried	N/A
[3]	O	O	X	X	O	O	O	RGB, NIR	X	O	O	X	Tractor	Prism camera
[38]	X	O	X	X	X	X	O	RGB, NIR, FIR, MIR	X	O	O	O	Hand-carried	N/A
[42]	X	O	O	X	O	X	O	RGB, NIR	X	O	O	X	Mobile robot	N/A
[45]	X	O	X	X	X	X	O	RGB, NIR	X	O	O	O	Vehicle	N/A
[5]	X	O	X	X	X	X	O	RGB, NIR	X	O	O	O	Vehicle	N/A
[6]	Thermal	O	O	X	O	O	O	RGB, Thermal	X	O	O	O	Vehicle	Beam splitter
[21]	X	O	O	X	O	O	O	RGB, Thermal	O	O	O	O	Vehicle	N/A
[25]	X	O	O	X	O	O	O	RGB, LWIR	O	O	O	O	Mobile robot, drone	N/A
[14]	Thermal	O	O	Thermal	O	O	O	RGB, Thermal	X	O	O	O	Vehicle	Beam splitter
[36]	X	O	O	O	O	O	O	RGB, NIR, Thermal	X	O	O	O	Vehicle	N/A
[30]	O	O	O	X	O	O	O	RGB, NIR	X	O	O	O	Vehicle	Prism camera
Ours	O	O	O	O	O	O	O	RGB, NIR	O	O	O	O	Mobile robot	Prism camera

Table 4. Summary of RGB-NIR and Multispectral Datasets Characteristics

Multi-View Camera Datasets that include a multi-view camera setup can utilize multi-view geometry, enabling tasks such as depth estimation or optical flow for 3D downstream vision tasks. In this comparison, if a dataset includes more than two cameras, it is considered multi-view. RGB-Stereo and NIR-Stereo specifically denote cases where multiple RGB or NIR cameras are included. For instance, the dataset from [36] includes both RGB stereo and NIR stereo, employing an active stereo camera. On the other hand, datasets such as [5, 28, 30, 38, 42, 45] used only one multispectral camera, limiting their 3D geometry capabilities. Additionally, [8, 9] employed NIR structured light cameras but did not provide the original NIR stereo images, only the estimated depth, which limits further multispectral analysis.

Ground Truth Depth Ground truth depth is essential for evaluating depth estimation methods. Some datasets, such as [8, 9, 42], used RGB-Depth or structured light cameras to obtain depth measurements. However, they do not include more accurate 3D depth measurements from LiDAR, which is a significant limitation for precise ground truth depth generation.

Environment The environmental conditions under which data is collected significantly impact dataset usability. Therefore, we categorize the datasets based on whether they include data from indoor, outdoor, day, or night scenarios. Multispectral datasets are especially useful in scenarios such as low-light conditions or environments with varying lighting (indoor to outdoor), as they provide complementary information to enhance RGB data.

Base Platform For large-scale data collection, the imaging system requires an appropriate mobile platform. Smaller systems can be handheld [8, 9, 38], while larger, heavier systems are typically mounted on vehicles [12, 14, 36], limiting data collection to areas accessible by vehicle. Research using mobile robots or drones [25, 42] improves both mobility and stability, allowing data collection in a wider range of environments.

Pixel-align Implementation Implementations of pixel-aligned multispectral imaging can be broadly categorized into beam splitters and prisms. Beam splitters are used by [6, 14, 28, 41], splitting incoming light into two beams that are then directed to different cameras, achieving pixel-level alignment. Alternatively, dichroic prism-based cameras, such as those used by [3, 30] and our dataset, separate the spectral bands through a prism and direct them to different CMOS sensors. This approach offers a more compact design compared to beam splitters, and provides complete spectral separation, making it preferable for many applications.

3. Details on RGB-NIR Feature Fusion and Attentional Fusion

3.1. RGB-NIR Image Fusion

3.1.1 Network Architecture

Residual Block We implemented the ResNet [15] architecture and its pretrained weights [27] as feature extractors for both the image fusion method and the feature fusion depth method. Table 5 presents the details of its PyTorch implementa-

tion. The basic version of this encoder accepts 3-channel inputs and outputs downsampled feature maps with 256 channels. Additionally, we can adjust the number of input channels, output channels, and the downstream scale factor as needed to accommodate various requirements.

Layer	Layer Name	Input Layer	Description	Output Shape
1	Input	-	Input data	$3 \times H \times W$
2	Conv1	Layer 1	3×3 kernel, padding=1, stride= s	$256 \times H/s \times W/s$
3	Norm1	Layer 2	InstanceNorm2d	$256 \times H/s \times W/s$
4	ReLU1	Layer 3	ReLU	$256 \times H/s \times W/s$
5	Conv2	Layer 4	3×3 kernel, padding=1, stride=1	$256 \times H/s \times W/s$
6	Norm2	Layer 5	InstanceNorm2d	$256 \times H/s \times W/s$
7	ReLU2	Layer 6	ReLU	$256 \times H/s \times W/s$

Table 5. Description of ResidualBlock forward sequence.

Attentional Feature Fusion In our framework, RGB and NIR images are first processed through the ResNet blocks, resulting in two separate 256-channel feature maps, F_v^c and F_n^c . To integrate these feature maps into a unified representation, we employ an attentional feature fusion method that leverages spectral information while preserving essential details. The fusion process comprises two main steps: the self-attention step and the attentional weight summation step.

Self-Attention Step. In this step, channel-level local and global attention mechanisms are applied to the feature maps F_v^c and F_n^c . The attention-enhanced features, A_v^c and A_n^c , are computed as follows:

$$A_u^c = \frac{A_v^c + A_n^c}{M(A_v^c) + M(A_n^c)}, \quad (6)$$

$$A_v^c = F_v^c \circ M(F_v^c), \quad (7)$$

$$A_n^c = F_n^c \circ M(F_n^c), \quad (8)$$

where M denotes the self-attention module introduced in [10], and \circ represents element-wise multiplication. The unified attention map A_u^c is obtained by normalizing the sum of the attention-enhanced features from both modalities.

Attentional Weight Summation Step. Using the unified attention map A_u^c , we compute the fused feature map F_f^c by performing a weighted summation of the attention-enhanced features from RGB and NIR:

$$F_f^c = (A_v^c \circ M(A_u^c)) + (A_n^c \circ (1 - M(A_u^c))). \quad (9)$$

In this equation, $M(A_u^c)$ serves as a weighting factor that dynamically balances the contributions of RGB and NIR features based on the unified attention map. This weighted summation effectively integrates features from both modalities, resulting in a single, comprehensive fused feature map. Figure 6 illustrates the detailed progress of attentional feature fusion using RGB and NIR feature maps. To provide a comprehensive and organized overview of the implementation, we have divided the attentional feature fusion module into four distinct PyTorch implementation tables. Specifically, Table 6 presents the LocalAttention module, Table 7 details the GlobalAttention module, and Table 8 describes the MultiChannelAttention module. These individual components are then integrated to form the complete Attentional Feature Fusion module, as demonstrated in Table 9.

Layer	Layer Name	Input Layer	Description	Output Shape
1	Input	-	Input tensor	$\text{in_channels} \times H \times W$
2	Local_Conv1	Layer 1	Conv2d with kernel size 1×1 , stride 1	$(\text{in_channels}/\text{reduction}) \times H \times W$
3	Local_BN1	Layer 2	BatchNorm applied to the output of Layer 1	$(\text{in_channels}/\text{reduction}) \times H \times W$
4	Local_ReLU	Layer 3	ReLU activation function applied to the output of Layer 2	$(\text{in_channels}/\text{reduction}) \times H \times W$
5	Local_Conv2	Layer 4	Conv2d with kernel size 1×1 , stride 1	$\text{in_channels} \times H \times W$
6	Local_BN2	Layer 5	BatchNorm applied to the output of Layer 4	$\text{in_channels} \times H \times W$
7	Output	Layer 6	Output tensor returned by the module	$\text{in_channels} \times H \times W$

Table 6. LocalAttentionModule

Layer	Layer Name	Input Layer	Description	Output Shape
1	Input	-	Input tensor	$\text{in_channels} \times H \times W$
2	Global_AvgPool	Layer 1	Adaptive Average Pooling to output size 1×1	$\text{in_channels} \times 1 \times 1$
3	Global_Conv1	Layer 2	Conv2d with kernel size 1×1 , stride 1	$(\text{in_channels}/\text{reduction}) \times 1 \times 1$
4	Global_BN1	Layer 3	BatchNorm applied to the output of Layer 2	$(\text{in_channels}/\text{reduction}) \times 1 \times 1$
5	Global_ReLU	Layer 4	ReLU activation function applied to the output of Layer 3	$(\text{in_channels}/\text{reduction}) \times 1 \times 1$
6	Global_Conv2	Layer 5	Conv2d with kernel size 1×1 , stride 1	$\text{in_channels} \times 1 \times 1$
7	Global_BN2	Layer 6	BatchNorm applied to the output of Layer 6	$\text{in_channels} \times 1 \times 1$
8	Output	Layer 7	Output tensor returned by the module	$\text{in_channels} \times 1 \times 1$

Table 7. **GlobalAttentionModule**.

Layer	Layer Name	Input Layer	Description	Output Shape
1	Input	-	Input tensor	$\text{in_channels} \times H \times W$
2a	Local_Attention	Layer 1	LocalAttentionModule	$\text{in_channels} \times H \times W$
2b	Global_Attention	Layer 1	GlobalAttentionModule	$\text{in_channels} \times 1 \times 1$
3	Addition	Layers 2a / 2b	Adds outputs of Layer 2a and Layer 2b	$\text{in_channels} \times H \times W$
4	Sigmoid	Layer 3	Applies Sigmoid activation	$\text{in_channels} \times H \times W$

Table 8. **MultiScaleChannelAttentionModule (MS-CAM)**.

Layer	Layer Name	Input Layer	Description	Output Shape
1a	Input_RGB	-	RGB input tensor	$\text{in_channels} \times H \times W$
1b	Input_NIR	-	NIR input tensor	$\text{in_channels} \times H \times W$
2a	Attention_RGB	Layer 1a	MS-CAM	$\text{in_channels} \times H \times W$
2b	Attention_NIR	Layer 1b	MS-CAM	$\text{in_channels} \times H \times W$
3	Addition	Layers 2a/ 2b	Adds Attention_RGB and Attention_NIR	$\text{in_channels} \times H \times W$
4a	RGB.Scaled	Layers 1a/ 3	Multiplies RGB input by (Layer 2a / Layer 3)	$\text{in_channels} \times H \times W$
4b	NIR.Scaled	Layers 1b/ 3	Multiplies NIR input by (Layer 2b / Layer 3)	$\text{in_channels} \times H \times W$
5	Addition	Layers 4a/ 4b	Adds RGB.Scaled and NIR.Scaled	$\text{in_channels} \times H \times W$
6	Attention_Fusion	Layer 5	MS-CAM	$\text{in_channels} \times H \times W$
7	Feature_Fusion	Layers 6, 1a, 1b	Combines Attention fusion with original RGB and NIR inputs to produce final output: $\text{Output} = (\text{Layer 5}) \times \text{RGB} + (1 - (\text{Layer 5})) \times \text{NIR}$	$\text{in_channels} \times H \times W$

Table 9. **AttentionFeatureFusion**.

Image Fusion Model Table 10 presents a detailed implementation of our image fusion network. Feature maps extracted from a pretrained feature encoder [27] are integrated using an attentional feature fusion mechanism [10]. These fused feature maps are subsequently processed through residual blocks to generate spatially varying weights, α and β , constrained within the range [0, 1]. Utilizing these weights, the fused image is computed as follows:

$$I_{\text{fusion}} = M_{\text{HSV} \rightarrow \text{RGB}}[I_H, I_S, I_V]^T = M_{\text{HSV} \rightarrow \text{RGB}}[I_H, I_S, \alpha I_V + \beta I_{\text{NIR}}]^T \quad (10)$$

Through this approach, we achieve the fusion of RGB and NIR images based on the derived spatially varying weights

NIR Guided RGB Filtering We employ an NIR-guided RGB filtering technique to enhance input RGB images by leveraging a single-channel Near-Infrared (NIR) image as the guiding reference [26]. Our method is grounded in the guided filter framework, where the NIR image I_{NIR} serves as the guide, and the fused image I_{fusion} is the input to be filtered. For each color channel i of I_{fusion} , we first compute the local means μ_{NIR} and $\mu_{\text{fusion},i}$, as well as the covariance $\text{cov}(I_{\text{NIR}}, I_{\text{fusion},i})$ and the variance $\text{var}(I_{\text{NIR}})$ within a window of radius r . The linear coefficients a_i and b_i are then determined using the equations

$$a_i = \frac{\text{cov}(I_{\text{NIR}}, I_{\text{fusion},i})}{\text{var}(I_{\text{NIR}}) + \epsilon}, \quad b_i = \mu_{\text{fusion},i} - a_i \mu_{\text{NIR}}, \quad (11)$$

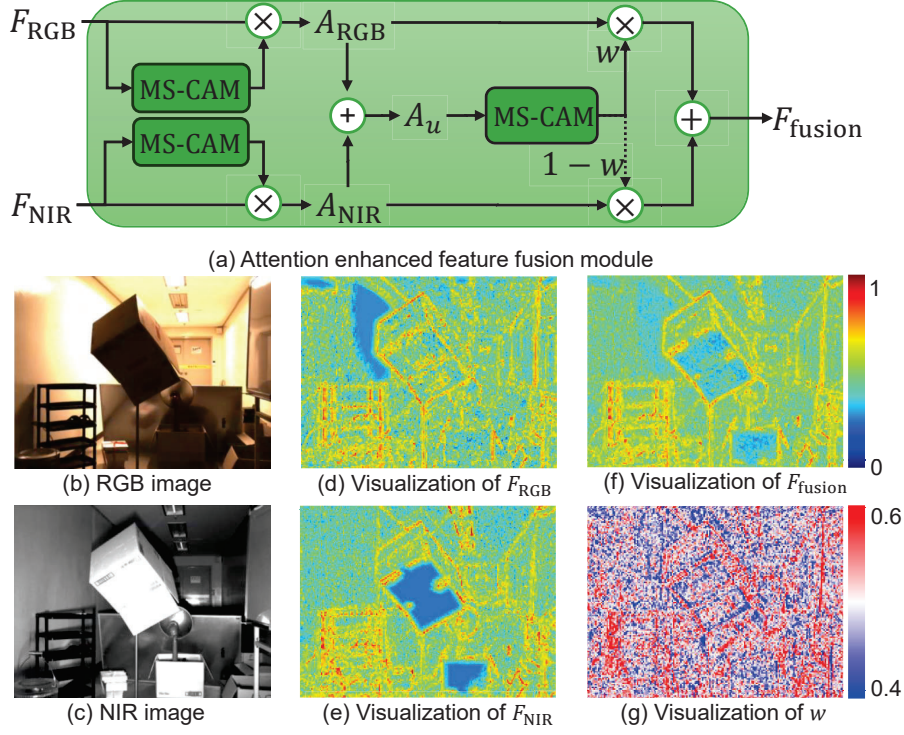


Figure 6. **Attention-enhanced feature fusion.** (a) The structure of attention based feature fusion module, proposed by [10]. MS-CAM is a attention enhancer module. The fusion operation is served by weight sum of attention-enhanced features with w . (b,c) Examples of input RGB and NIR images. (d,e,f,g) Visualizations of F_{RGB} (d), F_{NIR} (e), F_{fusion} (f) and w (g).

where ϵ is a regularization parameter that ensures numerical stability. Subsequently, we calculate the mean values of a^c and b^c within the same window and reconstruct the output RGB image \mathbf{q} for each channel using

$$I_{filtered,i} = \text{mean}(a_i)I_{NIR} + \text{mean}(b_i), \quad (12)$$

ensuring that the resulting pixel values are clamped within the valid range $[0, 255]$. This NIR-guided filtering approach effectively utilizes the structural information from the NIR guide to enhance color fidelity and suppress noise in the images which is generated during fusion process, demonstrating improved performance in various image processing applications.

Layer	Layer Name	Input Layer	Description	Output Shape
1a	Input_RGB	-	Receives RGB input tensor	$3 \times H \times W$
1b	Input_NIR	-	Receives NIR input tensor	$1 \times H \times W$
2	RGB_to_HSV	Layer 1a	Converts RGB tensor to HSV color space	$3 \times H' \times W'$
3a	Encoder_RGB	Layer 1a (RGB)	Passes normalized RGB through BasicEncoder [27]	$256 \times \frac{H'}{2} \times \frac{W'}{2}$
3b	Encoder_NIR	Layer 1b (NIR)	Passes normalized NIR (repeated across channels) through BasicEncoder	$256 \times \frac{H'}{4} \times \frac{W'}{4}$
4	Attentional_Feature_Fusion	Layers 3a and 3b	Combines HSV and NIR feature maps	$256 \times \frac{H'}{4} \times \frac{W'}{4}$
5	Residual_Block	Layer 4	Reduces channel dimensions using ResidualBlocks, followed by Sigmoid activation	$2 \times \frac{H'}{4} \times \frac{W'}{4}$
6	Upsample_Weights	Layer 5	Upsamples weights by a factor of 4 using bilinear interpolation	$2 \times H' \times W'$
7	Combine_HSV_and_NIR	Layers 1b, 2, 6	$[I_H, I_S, I_V] = [I_H, I_S, \alpha I_V + \beta I_{NIR}]$	$3 \times H' \times W'$
8	Convert_to_RGB	Layer 7	Converts combined HSV features back to RGB color space	$3 \times H' \times W'$

Table 10. Forward Pass of our Image Fusion Model

3.1.2 Loss Function

Photometric Loss The synthetic dataset includes original RGB images prior to augmentation. To ensure that the fused output of augmented RGB and NIR images closely matches the original RGB images, we design a photometric loss function defined as

$$\mathcal{L}_{\text{photometric}} = \gamma_{\text{L1}} \|I_{\text{fusion}} - I_{\text{original}}\|_1 + \gamma_{\text{ssim}} \text{SSIM}(I_{\text{fusion}}, I_{\text{original}}). \quad (13)$$

In this formulation, the L1 loss and Structural Similarity Index Measure (SSIM) loss are weighted by coefficients γ_{L1} and γ_{ssim} , respectively, set to 0.85 and 0.15. This combination ensures that the fused image maintains both pixel-wise accuracy and structural similarity to the original RGB image, facilitating effective training of the fusion model.

Ground-truth Disparity Loss To guide our network towards accurate operation, we trained it using a combination of an augmented synthetic dataset and our real dataset, utilizing the L1 loss function. Starting from a pretrained model based on [27], we loaded pretrained weights, excluding the modified structures. Then we fine-tuned our model by freezing weights excluding the modified ones. For the predicted disparity map d_{pred} , we compute L1 loss by

$$\mathcal{L}_{\text{gt}} = \sum_{u,v \in \Omega_{\text{gt}}} |d_{\text{gt}}(u,v) - d_{\text{pred}}(u,v)| \quad (14)$$

where Ω_{gt} is a binary mask indicating valid regions in the labeled disparity map and d_{gt} is ground-truth disparity map from dataset.

3.1.3 Training Details

We utilized our augmented RGB-NIR synthetic dataset, comprising approximately 50,000 original images. For each training batch, different augmentations were dynamically applied to the data to enhance diversity. The fusion model architecture consists of a pretrained feature encoder and a fusion module. During training, the feature encoder was frozen, and only the fusion module was updated to focus on learning effective fusion strategies. To improve the fidelity of the image fusion restoration process, we employed a photometric loss function. Specifically, the fused images were passed through Raft-Stereo to obtain disparity maps, which were then compared to the ground truth disparity maps. Although Raft-Stereo was configured to compute gradients, it was frozen and excluded from the optimizer to ensure that only our fusion network was trained.

For optimization, we used the Adam optimizer in conjunction with gradient scaling, initializing the gradient scaler with a value of 1024. To accelerate training and optimize memory usage, mixed precision training was enabled. The training process was conducted with a batch size of 4 per GPU, utilizing four RTX 3090 GPUs (24 GB each) in a distributed parallel setup, resulting in an effective batch size of 16. The model was trained for approximately 20 epochs using this hardware configuration.

3.2. RGB-NIR Feature Fusion

3.2.1 Network Architecture

We begin by processing the RGB and NIR stereo images through a shared ResNet-based feature extractor f_{enc} . For each input image I_s^c , where $s \in \{\text{RGB}, \text{NIR}\}$ and $c \in \{\text{left}, \text{right}\}$, the encoder transforms it into feature maps:

$$F_s^c = f_{\text{enc}}(I_s^c), \quad \text{for } s \in \{\text{RGB}, \text{NIR}\}, c \in \{\text{left}, \text{right}\}. \quad (15)$$

Next, we apply the attention-based fusion method [10] from Section 3.1.1 to combine the RGB and NIR features:

$$F_{\text{fusion}}^c = f_{\text{fusion}}(F_{\text{RGB}}^c, F_{\text{NIR}}^c). \quad (16)$$

The core of our method lies in constructing correlation volumes that capture the relationships between the left and right feature maps. We compute correlation volumes for both the fused features and the NIR features:

$$V_s(x, y, k) = F_s^{\text{left}}(x, y) \cdot F_s^{\text{right}}(x + k, y), \quad s \in \{\text{fusion}, \text{NIR}\}, \quad (17)$$

where (x, y) denotes the pixel location, k is the disparity index, and \cdot represents the inner product.

We employ an iterative approach using the GRU structure of the RAFT-Stereo network to refine disparity estimates. The process begins with an initial disparity $d_0 = 0$ and progressively updates this estimate. At each iteration n , the update model takes the following inputs:

- The previous disparity estimate, d_n .
- The context feature map from the fused left image, $F_{\text{fusion}}^{\text{left}}$.
- The sampled correlation volume V_{sampled} , which alternates between V_{fusion} and V_{NIR} .

The update model predicts a disparity increment Δd , which is added to the previous disparity to obtain the next estimate:

$$H_{n+1} = \text{ConvGRU}(H_n, d_n, [V_{\text{sampled}}, F_{\text{fusion}}^{\text{left}}]), \quad (18)$$

where H_n is the hidden state of the ConvGRU layer, initialized with $F_{\text{fusion}}^{\text{left}}$.

Then the increment of disparity is computed by a single convolutional layer and it updates the disparity estimate:

$$d_{n+1} = d_n + \text{Conv}(H_{n+1}). \quad (19)$$

By alternating between the fused and NIR correlation volumes at each iteration, our method effectively leverages spectral information from both RGB and NIR images. This approach places more emphasis on NIR features that are robust to environmental lighting, resulting in more accurate depth estimation as demonstrated in Table 16.

We also test our method on other stereo depth modules [4, 43] to highlight strength of feature fusion. We first fine-tune these models on the synthetic training dataset using the disparity reconstruction loss and then further adapt them to the real dataset with the LiDAR loss.

3.2.2 Loss Function

Ground-truth LiDAR Depth Loss We implemented pretrained weight provided by original author, which is trained with RGB stereo dataset, and fine-tuned this using a combination of an augmented synthetic dataset and our real dataset, utilizing the $L1$ loss function. For a series of disparity estimations $\{d_1, d_2, d_3 \dots d_n\}$ we compute $L1$ loss by

$$L_{\text{gt}} = \sum_i^N w^{n-i} \sum_{u,v \in \Omega_{\text{gt}}} |d_{\text{gt}}(u, v) - d_i(u, v)| \quad (20)$$

where w is weight factor for normalized summation of series of d and Ω_{gt} is a binary mask indicating valid regions in the labeled disparity map. We also use LiDAR points $(u_i, v_i, z_{\text{gt},i}) \in \mathbb{R}_{I^l}^3$, which is projected into image I^l coordinate, as ground truth for depth estimation on real-world data:

$$L_{\text{LiDAR}} = \sum_i^N \sum_{(x,y) \in \mathcal{N}(r)} |z_{\text{pred}}(u_i + x, v_i + y) - z_{\text{gt},i}| \quad (21)$$

where $\mathcal{N}(r)$ is a set of point offset, making local neighborhood box and r is radius of it, presently $r = 5$, $z_{\text{pred}} = \frac{f_x \cdot B}{d_n}$ and f_x and B is focal length of camera and baseline distance, which is constant of our system.

3.2.3 Training Detail

Feature Extraction of RGB and NIR Images We employed the same encoder structure and weights for feature extraction from both RGB and NIR images. This pretrained encoder, originally trained on RGB images, is effectively applicable to NIR data. By replicating the single-channel NIR image into three channels, we achieved stereo depth estimation using RAFT-Stereo [27] and CREStereo [23]. Similarly, replicating a single-channel image derived from converting RGB to grayscale or from a monocular visible-light camera into three channels did not present issues within the encoder, regardless of the channel count or target color space. This is because stereo depth estimation primarily involves analyzing the spatial correlation between pixels in the two images, and the correlation definition remains unaffected by changes in the spectral domain. To leverage the advantages of fine-tuning pretrained weights, we used the same encoder for both RGB and NIR images and kept the encoder frozen during training.

Supervised Training We conducted supervised training using our augmented synthetic dataset, which includes RGB-NIR stereo images and highly detailed, pixel-wise disparity maps. This allowed for supervised learning by directly comparing the predicted disparity maps generated by the model with the ground truth disparity maps. To mitigate the domain gap between real and synthetic data, we initially trained for one epoch solely on synthetic data and then incorporated real data into the training dataset. The real data includes LiDAR-derived depth information and pseudo disparity maps, along with occlusion maps obtained through sparse LiDAR reconstruction. Using these components, we were able to perform supervised learning with the previously defined loss function.

Self-supervised Training Once the disparity map d^{left} is obtained, it can be used to warp stereo images. The warped image $I^{\text{left|right}}$ that left image I^{left} is warped into right image coordinate, is computed by:

$$x' = x - d^{\text{left}}(x, y) \quad (22)$$

$$I^{\text{left|right}}(x, y) = I^{\text{left}}(x - d^{\text{left}}(x, y), y) \quad (23)$$

, where $x, y \in \Omega$ and Ω is image plain coordinate.

The photometric consistency between the warped and original images can be leveraged for self-supervised learning:

$$\mathcal{L}_{\text{photometric}} = |I^{\text{left|right}} - I^{\text{right}}| + \text{SSIM}(I^{\text{left|right}}, I^{\text{right}}) \quad (24)$$

However, this approach presents challenges when there are brightness differences between the left and right images, making it difficult for the loss to converge. Furthermore, when scene disparities are not pronounced, the warping loss may not effectively differentiate between accurate and erroneous disparity values, even if the disparity estimates are correct or grossly inaccurate. To ensure comprehensive data collection across both indoor and outdoor environments, we set the stereo baseline to 133mm. This setup results in a relatively narrow disparity range in outdoor scenes. Consequently, we utilized self-supervised learning only as an auxiliary method to support supervised learning on our real dataset.

Finetuning We fine-tuned the pretrained weights of RAFT-Stereo [27] to adapt and extend its capabilities. The correlation sampler, which computes the correlation volume V , is not a neural network structure and was therefore excluded from the learning process. To leverage the performance of the pretrained feature encoder, its parameters were frozen during training. Similarly, the iterative convolutional GRU updater was also frozen to retain its pretrained functionality.

In our design, an attentional feature fusion module [10] was incorporated to generate a fused feature map F_{fusion}^c with $F_{\text{RGB}}^c, F_{\text{NIR}}^c$. As such, the attentional feature fusion module was set to trainable mode to enable learning during fine-tuning.

RAFT-Stereo originally implements the context encoder with a structure identical to the feature encoder, differing only in the number of channels to accommodate separate weights. The context encoder serves as an input to update the hidden state of the GRU updater. To optimize the design and reduce the frequency of attentional fusion operations, we removed the context encoder. Instead, the features obtained from the feature encoder were directly forwarded to the inputs originally designated for context features. Consequently, this modification required additional learning in the process of forwarding the feature map to the GRU.

To ensure robust performance across different scenarios, batch normalization layers were frozen throughout the fine-tuning process. This approach preserved the stability of normalization statistics while allowing for effective adaptation of the model’s new components.

Implement feature fusion on different models MoCha-Stereo [4] and IGEV++[43] are based on an architecture similar to RAFT-Stereo[27], which comprises a feature encoder, a correlation volume, and recursive disparity refinement. This similarity allows for the straightforward adaptation of our feature-fusion modification across these models. In our approach, we insert an attentional feature fusion module after the feature encoder to generate an additional fusion feature alongside the standard RGB and NIR features. We then construct two separate correlation volumes—one from the NIR features and one from the fusion features—and alternate between them during the depth estimation process.

4. Additional Result

4.1. Implementation of RGB-NIR Image Fusion Methods for Comparative Analysis

Figure 7 and 8 illustrate different image fusion methods applied to our real RGB-NIR dataset. We conducted comparative studies on downstream vision applications using our image fusion model alongside other color fusion techniques.

Bayesian Fusion [44] This method employs Bayesian estimation to fuse two different mono-channel images. We implemented per-channel fusion for the R, G, and B channels with NIR using the Bayesian Fusion algorithm. The fusion process involves an iterative optimization procedure that incorporates total variation (TV) regularization to preserve edge details and reduce noise. Specifically, the fusion is achieved by solving the following optimization problem:

$$\min_C \|k_1 * C - D_1\|^2 + \|k_2 * C - D_2\|^2 + \|X - C\|^2 + \lambda \text{TV}(C) \quad (25)$$



Figure 7. **Image fusion visualization of ours and comparison methods.** (a,b) Pixel-aligned RGB and NIR images. (c) Image driven by simple channel sum. (d) YCrCb channel fusion [16]. (e) Bayesian estimation based fusion [44]. (f) Gradient adaptive fusion [1]. (g) DarkVisionNet [18]. (h) VGG based fusion [22]. (i) HSV channel fusion (our baseline) [11], (j) Our fusion method.

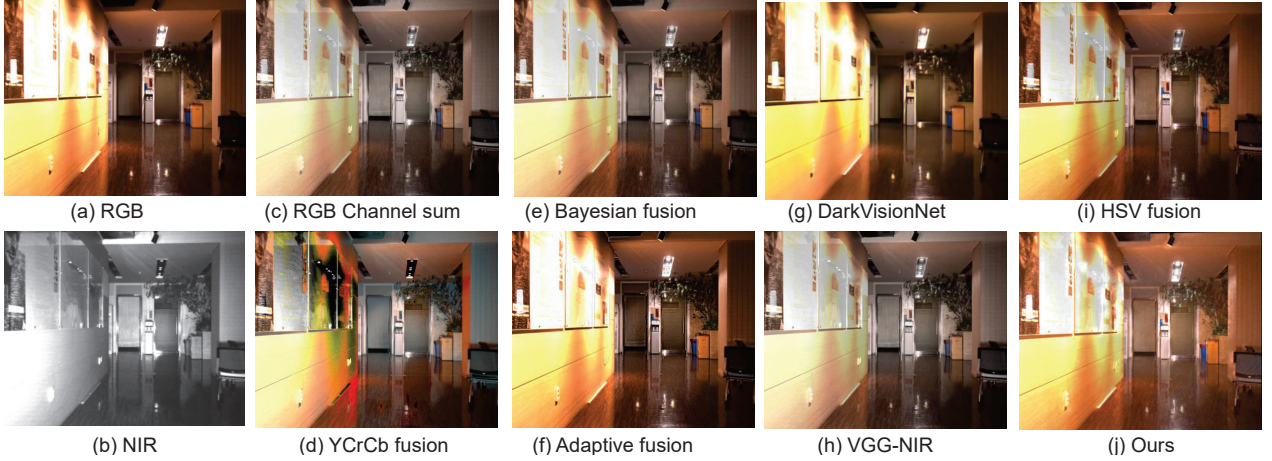


Figure 8. **Image fusion visualization of ours and comparison methods, indoor sample.** (a,b) Pixel-aligned RGB and NIR images. (c) Image driven by simple channel sum. (d) YCrCb channel fusion [16]. (e) Bayesian estimation based fusion [44]. (f) Gradient adaptive fusion [1]. (g) DarkVisionNet [18]. (h) VGG based fusion [22]. (i) HSV channel fusion (our baseline) [11], (j) Our fusion method.

where C is the fused image channel, D_1 and D_2 are the degraded observations from the RGB and NIR channels respectively, k_1 and k_2 are convolution kernels, X is the current estimate of the fused image, and λ is a regularization parameter controlling the strength of the TV term. The optimization is performed using proximal gradient methods, where the TV regularization is handled by the proximal operator:

$$C^{(t+1)} = \text{prox}_{\lambda \text{TV}} \left(\frac{2CY + \rho H}{2C + 2D + \rho} \right), \quad (26)$$

where Y represents the difference between the NIR and RGB images, ρ is a scaling factor, and H is updated through the proximal TV operation. The iterative process continues for a predefined number of iterations to obtain the final fused channel. The fused RGB image is then constructed by stacking the individually fused R, G, and B channels. We prepared the comparison using the source code provided by the authors, ensuring an accurate implementation of the Bayesian Fusion method for our dataset.

DarkVision [18] DarkVision employs a Multi-Layer Perceptron (MLP) network to enhance and denoise dark RGB images with guidance from high-quality NIR images. We utilized the pretrained version of this network for our comparisons, focusing on its ability to improve image quality without incorporating explicit mathematical formulations.

Infrared-VGG Fusion [22] This method is based on the VGG architecture and fuses infrared and visible images using a pretrained VGG module as the encoder. We followed the installation instructions provided by the authors and utilized their publicly available code to set up the comparison environment, enabling a direct evaluation of the fusion performance without delving into the underlying equations.

YCbCr Channel Fusion [16] The YCbCr Channel Fusion method leverages the superior detail preservation of NIR images under various environmental conditions by fusing the luminance (Y) channel in the YCbCr color space. The fusion process is carried out as follows: First, the RGB image is converted to the YCbCr color space, and its grayscale version is obtained.

$$\begin{bmatrix} l_{RGB} \\ C_1 \\ C_2 \end{bmatrix} = M_{RGB \rightarrow YCbCr} I_{RGB} \begin{bmatrix} I_R \\ I_G \\ I_B \end{bmatrix}. \quad (27)$$

The fusion weight is computed by normalizing the difference between the NIR and grayscale images:

$$l_V = \frac{I_{NIR} - l_{RGB}}{I_{\max}}, \quad (28)$$

where I_{NIR} is the intensity of the NIR image and I_{Gray} is the intensity of the grayscale RGB image. I_{\max} is the maximum intensity value for normalization scaling the difference to the range [0, 1]. The fused luminance channel is then calculated as:

$$l_{\text{fused}} = l_{RGB} \cdot l_V + I_{NIR} \cdot (1 - l_V), \quad (29)$$

where l_{RGB} represents the luminance component of the YCbCr image. To enhance the chromatic channels, a scaling factor m is determined by:

$$m = \frac{l_{RGB} - l_{\text{fused}}}{l_{RGB}}, \quad \text{where } m = 0 \quad \text{if } l_{RGB} = 0. \quad (30)$$

This ensures numerical stability by avoiding division by zero. The chromatic channels are then adjusted using the scaling factor and then the fused YCbCr image is reconstructed by stacking the fused luminance and adjusted chromatic channels:

$$C_{1,\text{fused}} = C_1 \cdot (1 + m), C_{2,\text{fused}} = C_2 \cdot (1 + m) \quad (31)$$

$$I_{\text{fused}} = M_{YCbCr \rightarrow RGB} [l_{\text{fused}}, C_{1,\text{fused}}, C_{2,\text{fused}}]^T. \quad (32)$$

This fused YCbCr image is then converted back to the RGB color space to obtain the final fused image. We prepared the comparison based on the equations described in the paper, implementing the fusion process as outlined in the authors' source code.

HSV Channel Fusion [11] This method transforms the images into the HSV color space and fuses the V channel by addition:

$$V_{\text{fused}} = 0.5V_{RGB} + 0.5I_{NIR}. \quad (33)$$

The resulting HSV image is then converted back to RGB. We prepared the comparison by blending the V channel and NIR image at a 0.5:0.5 ratio.

Adaptive RGB-NIR Fusion [1] The Adaptive RGB-NIR Fusion method enhances the edges of the RGB image by leveraging the pixel gradients from the NIR image. The fusion process consists of several key steps: local contrast computation, fusion map generation, high-pass filtering, and image enhancement.

The first step is a local contrast computation. The method computes the local contrast for both the luminance component of the YCbCr-transformed RGB image and the NIR image. The YCbCr transformation is defined as:

$$\begin{bmatrix} l_{RGB} \\ C_1 \\ C_2 \end{bmatrix} = M_{RGB \rightarrow YCbCr} I_{RGB} \begin{bmatrix} I_R \\ I_G \\ I_B \end{bmatrix}, \quad (34)$$

where l_{RGB} is the luminance (Y channel), and C_1, C_2 represent the chrominance channels (Cb and Cr, respectively). For an intensity image I (either luminance l_{RGB} or NIR intensity I_{NIR}), the local contrast LocalContrast_I is defined as a combination of intensity variation and amplitude variation within a local window:

$$\text{LocalContrast}_I = \alpha \cdot (I_{\max} - I_{\min}) + (1 - \alpha) \cdot \text{MaxAmplitude}(I), \quad (35)$$

where:

- I_{\max} and I_{\min} are the maximum and minimum intensity values within the local window.
- $\text{MaxAmplitude}(I)$ is the maximum amplitude obtained from the gradient magnitude computed using Sobel filters:

$$\text{MaxAmplitude}(I) = \max \left(\sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2} \right) \quad (36)$$

- α is a weighting factor (set to 0.5 in our implementation).

The next step is generating the fusion map. Using the local contrasts $\text{LocalContrast}_{l_{\text{RGB}}}$ and $\text{LocalContrast}_{\text{NIR}}$, the fusion map FusionMap is generated to determine the regions where the NIR image can provide enhanced details:

$$\text{FusionMap} = \frac{\max(0, \text{LocalContrast}_{\text{NIR}} - \text{LocalContrast}_{l_{\text{RGB}}})}{\max(\text{LocalContrast}_{\text{NIR}}, \epsilon)}, \quad (37)$$

where ϵ is a small constant (e.g., 1×10^{-6}) to prevent division by zero. This fusion map emphasizes areas where the NIR image has higher local contrast compared to the RGB image.

A high-pass filter is then applied to the NIR image to extract high-frequency details:

$$\text{HPF}(I_{\text{NIR}}) = I_{\text{NIR}} - \text{GaussianBlur}(I_{\text{NIR}}, \sigma), \quad (38)$$

where GaussianBlur applies a Gaussian filter with a specified kernel size (e.g., 19×19) to smooth the NIR image, and σ is the standard deviation of the Gaussian kernel.

The final enhancement is performed by adding the product of the fusion map and the high-pass filtered NIR image to each channel of the YCbCr-converted RGB image. The fused YCbCr image is then converted back to the RGB color space. The final fused RGB image I_{fused} is computed as:

$$I_{\text{HDF}} = \text{FusionMap} \cdot \text{HPF}(I_{\text{NIR}}) \quad (39)$$

$$I_{\text{fused}} = M_{\text{YCbCr} \rightarrow \text{RGB}} \begin{bmatrix} l_{\text{RGB}} + I_{\text{HDF}} \\ C_1 + I_{\text{HDF}} \\ C_2 + I_{\text{HDF}} \end{bmatrix}. \quad (40)$$

Finally, the fused RGB image I_{fused} is clipped to the valid intensity range $[0, 255]$ to ensure proper image representation.

4.2. RGB-NIR Image Fusion for Object Detection

We evaluated the performance of our color fusion approach on object detection tasks [32].

4.2.1 Evaluation

To evaluate the performance of our object detection model, we employed standard metrics commonly used in single-class multi-object detection tasks. Specifically, we utilized precision, recall, F1-score, mean Average Precision (mAP), and average Intersection over Union (IoU) as our primary evaluation metrics. Given that the synthetic dataset was generated with precise object index information, we were able to construct accurate ground-truth annotations for object detection. For the real dataset, we prepared the data by directly creating pseudo labels to serve as ground-truth annotations.

4.2.2 Additional Results

Table 11 presents a comprehensive evaluation of various image fusion methods using YOLO [32] for object detection across multiple performance metrics. Notably, our proposed RGB-NIR fusion technique achieves the highest Detection mAP (0.828), surpassing the second-best Bayesian fusion method (0.773) by a significant margin. This superior performance

is consistently reflected across all evaluated metrics, including Average IOU (0.509), F1 score (0.688), Precision (0.734), and Recall (0.687), indicating a robust enhancement in both localization and classification capabilities. In comparison, traditional methods such as HSV baseline [11] and YCrCb [16] demonstrate lower performance, with Detection mAP values of 0.744 and 0.745, respectively. Furthermore, specialized approaches like DarkVision [18] exhibit considerably lower metrics, highlighting the effectiveness of our fusion strategy. The consistent outperformance across diverse metrics underscores the efficacy of our RGB-NIR fusion method in enhancing object detection accuracy, precision, and recall, thereby establishing it as a superior choice for image fusion in object detection tasks.

Methods	Detection mAP \uparrow	Average IOU \uparrow	F1 score \uparrow	Precision \uparrow	Recall \uparrow
RGB	0.756	0.494	0.623	0.642	0.631
NIR	0.703	0.445	0.558	0.617	0.551
YCrCb [16]	0.745	0.479	0.596	0.651	0.590
Bayesian[44]	0.773	0.485	0.641	0.661	0.647
DarkVision[18]	0.571	0.351	0.465	0.504	0.466
Adaptive[1]	0.762	0.473	0.630	0.665	0.632
VGG-NIR[22]	0.726	0.449	0.573	0.588	0.590
HSV(our baseline) [11]	0.744	0.464	0.612	0.649	0.608
Ours	0.828	0.509	0.688	0.734	0.687

Table 11. **Comparison of image fusion methods for YOLO [32].** Our RGB-NIR image fusion method outperforms other image-fusion methods for object detection.

4.3. RGB-NIR Image Fusion for Structure-from-Motion

Experiments In this study, we evaluated the results of our image fusion approach within the COLMAP framework, a robust Structure-from-Motion (SfM) pipeline, to reconstruct 3D geometry from RGB, Near-Infrared (NIR), and fused images. SfM is a crucial process that estimates both intrinsic and extrinsic camera parameters from multiple images with unknown camera settings, ultimately generating a coherent 3D model of the scene. This reconstruction process begins by selecting a subset of frames to define a world coordinate system, followed by aligning subsequent frames through feature matching, ensuring precise alignment across all images. Effective feature extraction is essential for successful SfM, especially under challenging lighting conditions where regions may suffer from underexposure or overexposure, resulting in unreliable features. Single-channel images, such as those in the NIR spectrum, often lack sufficient color variation, complicating feature extraction. To overcome these limitations, we propose a fusion approach that integrates RGB images, which offer rich color details, with NIR images, which improve feature detection in low-contrast areas. This fusion is designed to enhance feature quality, enabling more accurate and robust 3D reconstructions within the COLMAP environment.

For our experiments, we used a well-illuminated nighttime video dataset, extracting 200 consecutive frames. Each frame contained stereo RGB and stereo NIR images, yielding a total of 400 RGB images, 400 NIR images, and 400 fused images produced using our image fusion method. To ensure consistency across different reconstruction scenarios, we uniformly applied the Simple Radial model for feature extraction across all image sets. Prior to feature extraction, all images underwent stereo calibration to determine intrinsic parameters, which were subsequently used to undistort the images. This preprocessing step mitigated lens distortion and ensured geometric consistency, with the intrinsic parameters provided as presets for the Simple Radial model.

Feature matching was performed using exhaustive matching with a block size of 50, enabling comprehensive correspondence across image pairs. We conducted three reconstruction experiments, each evaluating one of the image modalities: RGB, NIR, and fused images. By comparing the outcomes, we assessed the impact of image fusion on feature detection and overall reconstruction accuracy under challenging lighting conditions.

Result Figure 9 presents a qualitative comparison of COLMAP feature extraction applied to RGB, NIR, and fusion images. Red dots represent the extracted features, while pink dots indicate matched keypoints used for feature correspondence. (c) demonstrates that the fusion images effectively integrate features from both RGB (a) and NIR (b), compensating for dark regions in one spectral domain with information from the other. This highlights the capability of the fusion method to leverage complementary spectral information for robust feature extraction and matching.

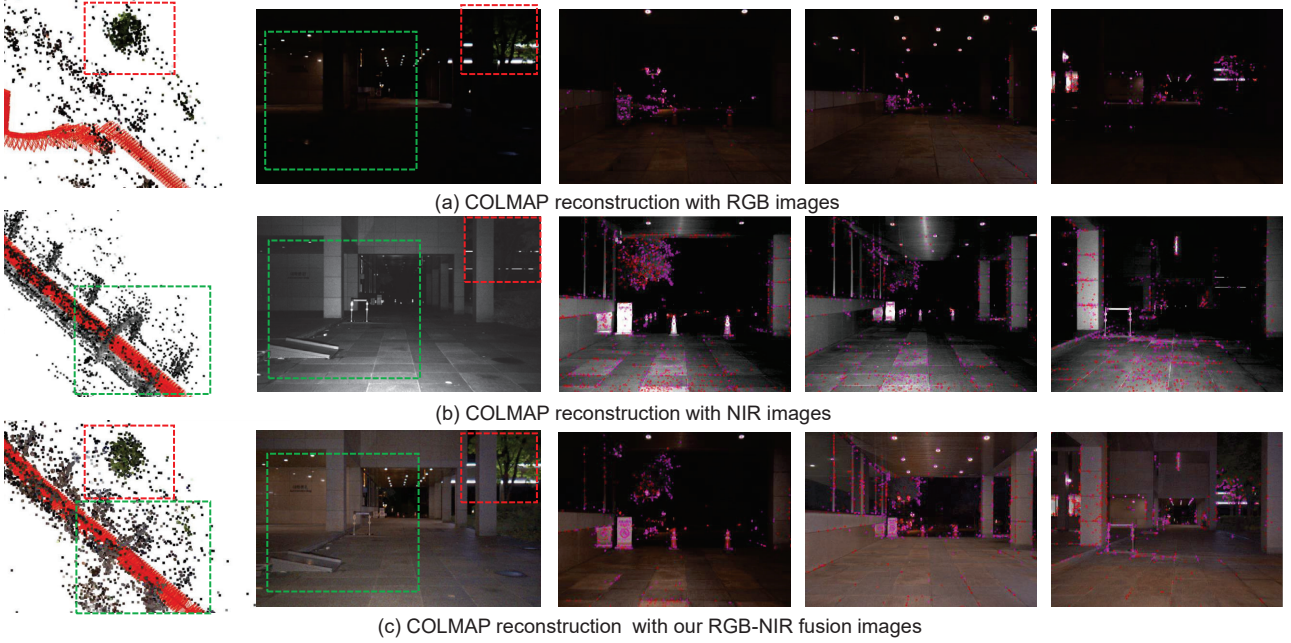


Figure 9. COLMAP reconstruction examples for RGB, NIR and our fused images. (a) RGB images. (b) NIR images. (c) Our fused images.

4.4. Sparse Depth Reconstruction

Sparse depth reconstruction aims to transform a sparse depth map into a full-resolution dense depth map using an RGB image as guidance [19, 39]. Many depth estimation methods, such as LiDAR and RGB-D cameras, provide depth information at resolutions that are significantly lower than those of the accompanying RGB images. High-end LiDAR sensors typically have a maximum resolution of 2048×128 , and stereo depth estimation methods are often considered to have limitations in reliability. Excluding low-confidence regions from these methods' results leads to further sparsity and significant reductions in depth data. Furthermore, when stereo depth maps are computed and warped onto different camera image planes, the detailed pixel-level depth information is often lost. To address these challenges, methods such as CostDCNet [19] and BPNet [39] use RGB images as guidance to reconstruct dense depth maps from sparse data. However, certain regions of the RGB image may lack sufficient detail for depth estimation, and significant lighting variations can further degrade the reconstruction quality. Our proposed RGB-NIR data and RGB-NIR image fusion methodology generates images with excellent photometric consistency even under challenging lighting conditions. As a result, this approach enhances the performance of sparse depth reconstruction by enabling more reliable guidance images and improving depth estimation accuracy.

4.5. RGB-NIR Fusion for Depth Estimation

4.5.1 Implementation of RGB-NIR Stereo Depth Estimations for Comparative Analysis

This study evaluates our proposed feature-fusion method against several state-of-the-art approaches for stereo depth estimation. The methods compared include single-spectral stereo disparity estimation (RGB and NIR) using RAFT-Stereo [27], RAFT-Stereo integrated with our image fusion technique, and other image fusion approaches [1, 11, 16, 18, 22, 24, 44]. Additionally, comparisons were made with fine-tuned versions of cross-spectral stereo depth estimation models [14, 40].

For comparisons using RAFT-Stereo, we tested various input images while maintaining consistent weights in the RAFT-Stereo model to ensure fair pairwise comparisons. For multi-spectral models, we fine-tuned each method using our augmented synthetic dataset and real-world dataset to achieve optimal performance.

These four images are then used for depth estimation. Instead of relying on synthetic images generated via conversion, our approach directly utilizes RGB-NIR stereo pairs as inputs and fine-tunes the depth network accordingly.

The CSPD model [14], originally designed for stereo depth estimation with RGB and thermal stereo pairs, was adapted for this study. We replaced thermal stereo images with NIR stereo pairs and retrained the model using our dataset to facilitate

direct comparison.

Lastly, the DPSNet model [40], which extends RAFT-Stereo by incorporating polarimetry stereo images alongside RGB stereo inputs, was adapted for NIR stereo. We replaced polarimetry stereo images with NIR stereo pairs and conducted supervised training to fine-tune the model for our application.

4.5.2 Evaluation Metrics

We evaluated our stereo depth estimation approach using labeled ground-truth depth data, employing a comprehensive set of metrics to assess performance across both synthetic and real-world datasets.

For the synthetic dataset, ground truth disparity maps were provided. To evaluate the accuracy of the predicted disparity maps, we utilized two primary error metrics: the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE). We computed the absolute error (Mean Absolute Error, MAE) and the root mean square error (RMSE) between the predicted disparity map and the ground truth disparity map as primary evaluation metrics. Additionally, we calculated the proportion of pixels with Euclidean distance errors below thresholds of 1, 3, and 5 pixels to assess the spatial precision of the predictions. These evaluations resulted in five distinct metrics: MAE, RMSE, and the percentage of pixels with errors under 1, 3, and 5 pixels ($e < 1\text{px}$, $e < 3\text{px}$, $e < 5\text{px}$).

For the real-world dataset, which included sparse LiDAR depth labels, we evaluated the predicted depth values by comparing them to ground truth depth measurements at LiDAR-indicated points. The primary metrics applied in this evaluation were the MAE and RMSE of the depth values, expressed in meters, providing a robust measurement of accuracy in real-world scenarios. Furthermore, we computed the ratio

$$\text{ratio} = \max \left(\frac{z_{\text{pred}}(u, v)}{z_{\text{gt}}(u, v)}, \frac{z_{\text{gt}}(u, v)}{z_{\text{pred}}(u, v)} \right) \quad (41)$$

for all pixels (u, v) and defined three thresholds: δ_1 (ratio ≤ 1.25), δ_2 (ratio $\leq 1.25^2$), and δ_3 (ratio $\leq 1.25^3$). These thresholds provided additional evaluation metrics, resulting in a total of five metrics: MAE based on depth, RMSE, δ_1 , δ_2 , and δ_3 .

4.5.3 Additional Quantitative Results

We present the stereo depth estimation results for both our RGB-NIR real dataset and the augmented synthetic dataset. Building upon the summary results presented in the main paper, we have conducted a more comprehensive analysis by categorizing the data based on different environmental conditions and expanding the evaluation metrics to include Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and threshold-based accuracy metrics (δ_1 , δ_2 , δ_3). This detailed approach facilitates a deeper understanding of the algorithm’s performance across diverse scenarios.

Table 12 demonstrates that our fusion method consistently outperforms single modality depth estimation across a range of challenging environments, including well-lit outdoor, dark outdoor, cloudy, well-lit indoor, dark indoor, and complex indoor settings. Moreover, Table 13 illustrates that our fusion strategy not only enhances performance when applied to RAFT stereo but also provides significant improvements with other stereo depth estimation methods [4, 27, 43], highlighting its versatility as a robust baseline approach.

4.5.4 Additional Qualitative Samples

Figure 10 presents a qualitative comparison of our proposed stereo depth estimation network with RAFT-Stereo [27], evaluated on RGB images (a), NIR images (b) and our image fusion method (c). Our method demonstrates superior performance, particularly in regions with overexposure and high specular reflectance. Additionally, Figure 11 showcases qualitative results under challenging lighting conditions. For low-light regions, our feature fusion approach effectively compensates for missing information by leveraging complementary spectral data, resulting in more accurate and consistent depth estimation.

4.6. Additional Ablation Study

4.6.1 Pretrained Feature Encoder Weights

Table 14 and Figure 12 show an ablation study of the pretrained feature encoder. We quantitatively and qualitatively compared the performance of our feature fusion depth estimation using three encoder weights provided by [27], trained on Eth3D [35], SceneFlow [29], and Middlebury [33]. The encoder based on Eth3D was found to be the most suitable among the three.

Depth MAE [m]↓	Well-lit Outdoor	Dark Outdoor	Cloudy Outdoor	Well-lit Indoor	Dark Indoor	Complex Indoor
RGB	5.710	7.491	3.101	3.292	3.201	4.698
NIR	4.576	5.563	3.139	3.062	2.333	4.271
YCbCr [16]	6.759	5.602	3.126	3.025	2.231	6.429
Adaptive [1]	6.857	6.408	3.094	3.227	2.375	6.675
Bayesian [44]	6.225	10.367	7.717	6.187	5.372	7.996
HSV (our baseline) [11]	4.468	5.417	3.121	3.091	2.293	4.293
SIRLUT [24]	4.485	7.725	3.351	5.134	4.498	5.768
VGG-NIR [37]	6.75	6.303	3.021	3.055	2.138	6.682
CSPD [14]	6.965	11.403	8.86	12.132	10.661	13.954
DPS-NET [40]	7.555	6.593	2.75	2.849	2.326	5.144
Image Fusion	4.356	5.156	2.816	2.99	2.431	4.181
Feature fusion	3.651	4.771	2.838	3.1	2.335	4.182

Table 12. **Metrics for Stereo Depth Estimation.** Evaluation conducted on a real-world dataset captured in a nighttime environment. Results are grouped as follows: (1) the first group represents metrics computed using our RGB-NIR dataset with 3-channel stereo input processed by the RAFT-Stereo [27] model, IGEV++ [43] and MoChaStereo [4]; (2) the second group includes results from other multi-spectral stereo depth estimation approaches, fine-tuned on our dataset; and (3) the final group showcases methods from Sections 4.1 and 4.2 of our main paper.

depth MAE [m]↓	Day outdoor	Night outdoor	Cloudy outdoor	Well-lit indoor	Dark indoor	Complex indoor
RAFT-Stereo (RGB) [27]	5.710	7.491	3.101	3.292	3.201	4.698
RAFT-Stereo (NIR) [27]	4.576	5.563	3.139	3.062	2.333	4.271
RAFT-Stereo (HSV) [27]	4.468	5.417	3.121	3.091	2.293	4.293
Mocha-Stereo (RGB) [4]	6.037	16.930	3.915	7.083	8.772	10.841
Mocha-Stereo (NIR) [4]	5.353	8.949	5.203	5.891	3.725	6.338
Mocha-Stereo (HSV) [4]	5.141	10.496	5.167	6.121	4.154	7.634
IGEV++ (RGB) [43]	6.176	9.277	4.261	3.194	3.643	5.958
IGEV++ (NIR) [43]	6.457	5.630	4.189	3.312	2.573	4.692
IGEV++ (HSV) [43]	6.096	5.762	4.130	3.052	2.501	4.915
Our image fusion [27]	4.356	5.156	2.816	2.990	2.431	4.181
Our image fusion [4]	5.191	8.125	3.731	4.144	2.882	5.788
Our image fusion [43]	5.968	7.314	3.737	3.532	2.882	4.848
Our feature fusion [27]	3.651	4.771	2.838	2.695	2.059	3.920
Our feature fusion [4]	4.415	6.063	3.147	4.890	4.282	6.688
Our feature fusion [43]	2.231	3.012	2.368	2.159	1.794	3.091

Table 13. **RGB-NIR feature-based depth estimation.** Our feature-based image fusion method for depth estimation methods [4, 27, 43] outperforms using the single-modality RGB/NIR inputs, hand-craft fusion [11] and our image fusion method (Section 3.1.1).

Pretrained	MAE (m)	RMSE (m)	δ_1	δ_2	δ_3
SceneFlow [29]	2.6191	6.7860	0.6473	0.8768	0.9545
Middlebury [33]	5.0356	8.8270	0.2838	0.4143	0.4594
Eth3d [35]	2.5886	6.7470	0.6526	0.8775	0.9556

Table 14. **Ablation study on different pretrained feature encoder**

4.6.2 Ablation on Feature Fusion Implementation

Table 15 and Figure 13 show an ablation study of different feature fusion methods. We explored five methods: simple feature addition, concatenation with a convolution layer to down-sample concatenated feature channels, pointwise feature multiplication, weighted sum with RGB features at 0.25 and NIR features at 0.75, and our implementation of the attentional feature fusion method inspired by [10]. Although feature multiplication showed the best quantitative results when evaluated using sampled sparse LiDAR information, it performed poorly in qualitative analysis. The attentional feature fusion method demonstrated the best adaptation to spatially varying lighting conditions.

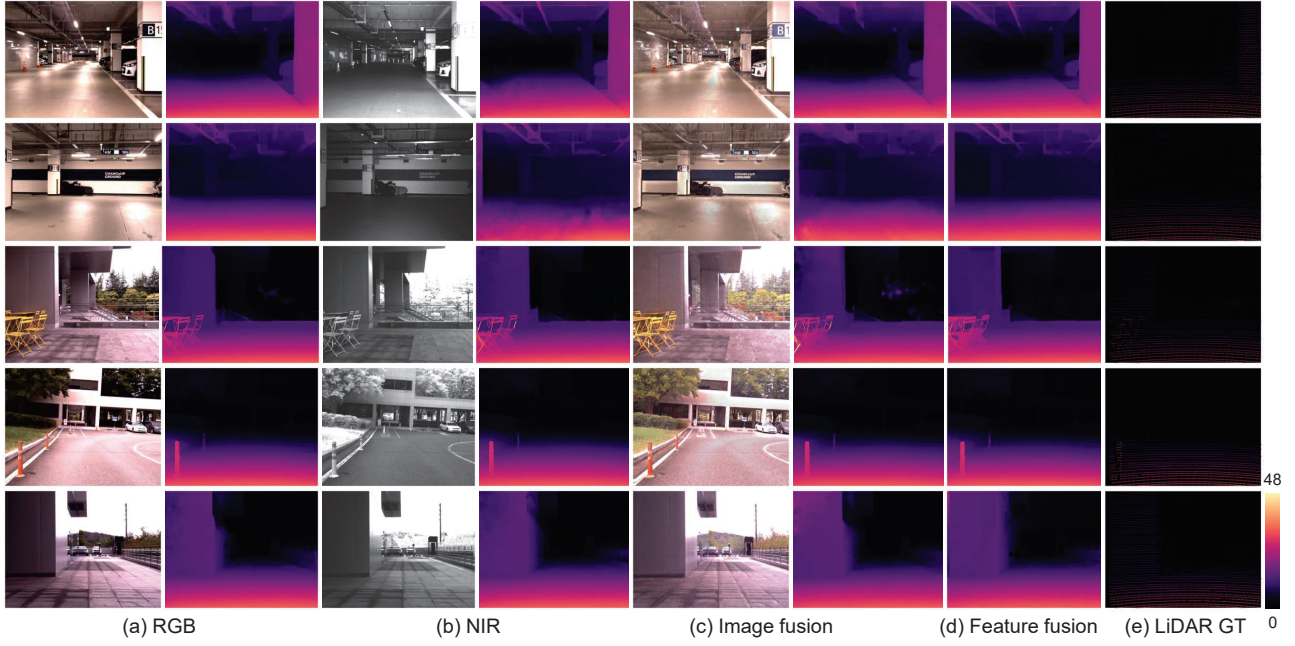


Figure 10. **Depth estimation samples of our feature fusion model.** (a) RGB images and outputs of [27] with them. (b) NIR images and outputs with them. (c) Fused images by our image fusion method and outputs with them. (d) Stereo depth estimation with our feature fusion based method. (e) Ground-truth sparse LiDAR.

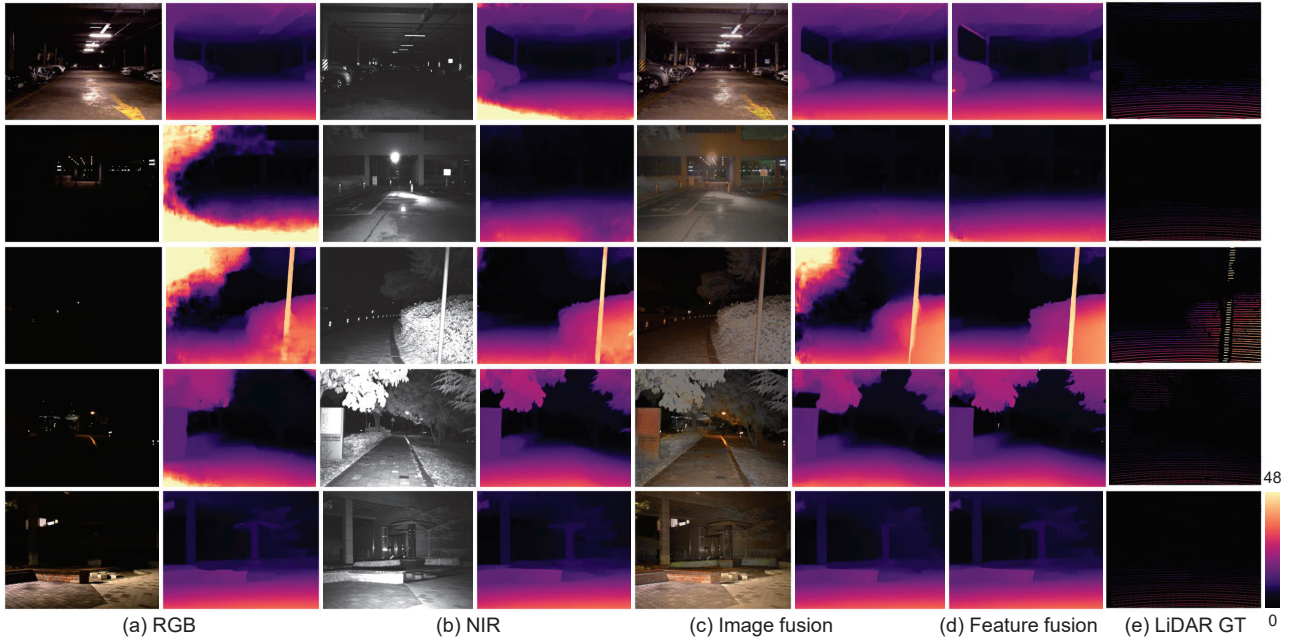


Figure 11. **Depth estimation samples on challenging lighting conditions of our feature fusion model.** (a) RGB images and outputs of [27] with them. (b) NIR images and outputs with them. (c) Fused images by our image fusion method and outputs with them. (d) Stereo depth estimation with our feature fusion based method. (e) Ground-truth sparse LiDAR.

Table 16 and Figure 14 show an ablation study on different correlation volume alternating methods. We compared using only the Fusion volume, alternating between RGB and NIR volumes, alternating among Fusion, RGB, and NIR volumes,

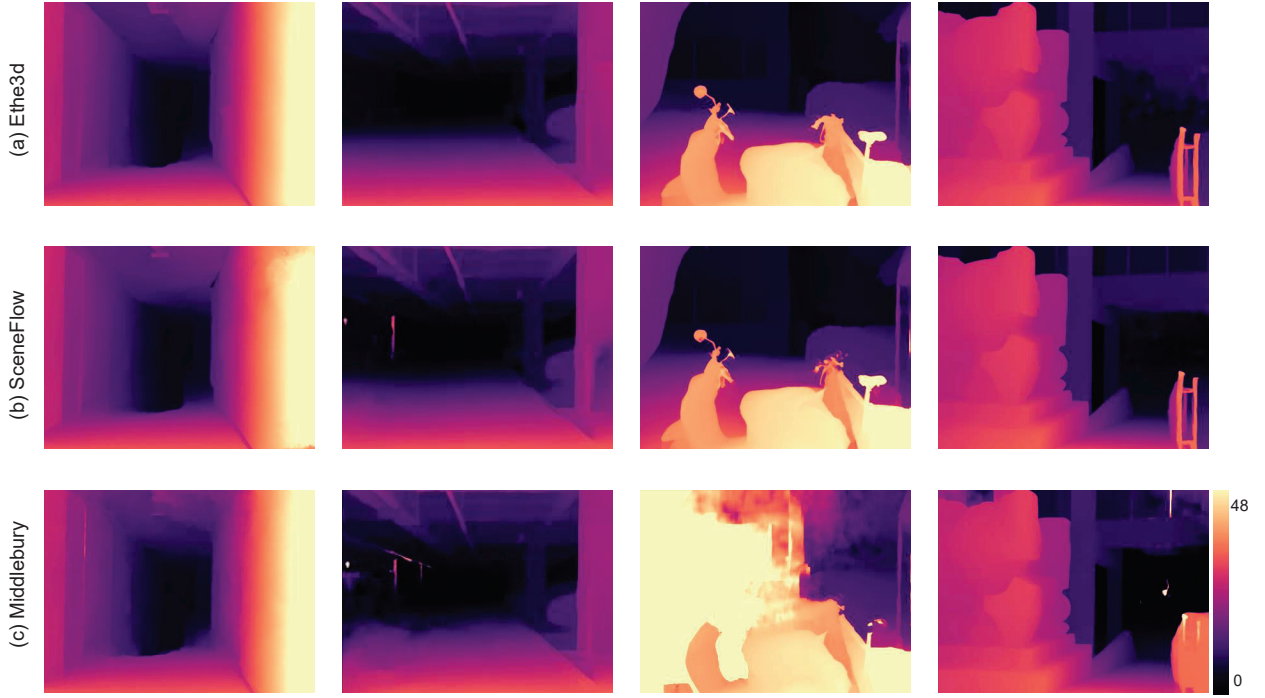


Figure 12. **Ablation study on pretrained encoder.** (a) Pretrained with [35]. (b) Pretrained with [29]. (c) Pretrained with [33].

and alternating between Fusion and NIR volumes. In a quantitative evaluation using sparse LiDAR ground truth, the Fusion-
RGB-NIR alternating method achieved the best performance. However, the most reliable qualitative results were obtained
with the Fusion-NIR alternating method.

Pretrained	MAE (m)	RMSE (m)	δ_1	δ_2	δ_3
Feature addition	2.611	6.5365	0.6526	0.8772	0.9552
Concatenation	2.5719	6.004	0.6455	0.8782	0.954
Multiplication	2.4524	5.8743	0.6634	0.8768	0.9517
Weight sum	2.6251	6.9073	0.6491	0.8686	0.9535
Attentional feature fusion	<u>2.5886</u>	6.7470	0.6422	<u>0.8775</u>	0.9556

Table 15. **Ablation study on different implementations of feature fusion.**

Correlation volumes for disparity estimation	MAE (m)	RMSE (m)	δ_1	δ_2	δ_3
Fusion correlation volumes only	3.3979	7.4405	0.5073	0.7409	0.8073
Alternating RGB-NIR correlation volumes	2.6508	8.5712	0.6575	0.8841	0.9555
Alternating Fusion-RGB-NIR correlation volumes	2.5238	7.4262	0.6336	0.8763	0.9550
Alternating Fusion-NIR correlation volumes	<u>2.5886</u>	6.7470	<u>0.6422</u>	<u>0.8775</u>	0.9556

Table 16. **Ablation study on different implementations of correlation volume alternations.**

References

- [1] Mohamed Awad, Ahmed Elliethy, and Hussein A Aly. Adaptive near-infrared and visible fusion for fast image enhancement. *IEEE Transactions on Computational Imaging*, 6:408–418, 2019. 19, 20, 22, 23, 25
- [2] Jean-Yves Bouguet. Camera calibration toolbox for matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/, 2004. 4

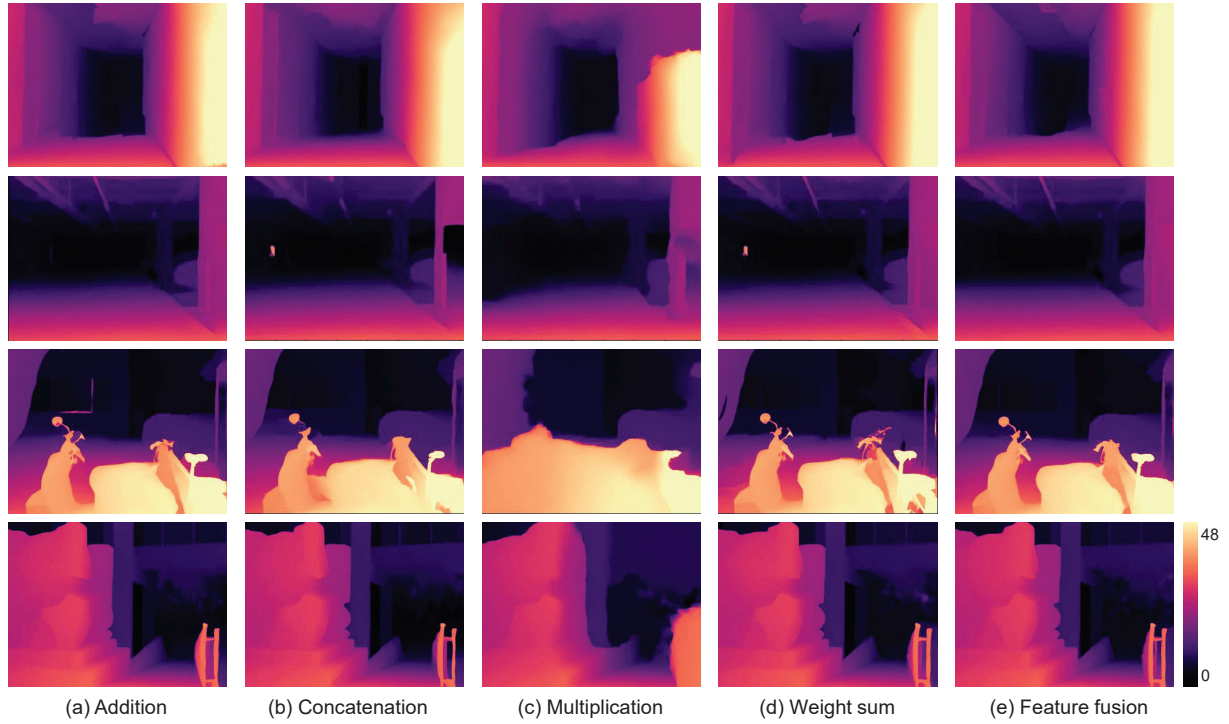


Figure 13. **Ablation study on feature fusion method.** (a) Feature addition. (b) Feature concatenation. (c) Feature multiplication. (d) Feature weight summation. (e) Attentional feature fusion [10]

- [3] Nived Chebrolu, Philipp Lottes, Alexander Schaefer, Wera Winterhalter, Wolfram Burgard, and Cyrill Stachniss. Agricultural robot dataset for plant classification, localization and mapping on sugar beet fields. *The International Journal of Robotics Research*, 36(10):1045–1052, 2017. 11, 12
- [4] Ziyang Chen, Wei Long, He Yao, Yongjun Zhang, Bingshu Wang, Yongbin Qin, and Jia Wu. Mocha-stereo: Motif channel attention network for stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27768–27777, 2024. 17, 18, 24, 25
- [5] Gyeongmin Choe, Seong-Heum Kim, Sunghoon Im, Joon-Young Lee, Srinivasa G Narasimhan, and In So Kweon. RANUS: RGB and NIR urban scene dataset for deep scene parsing. *IEEE Robotics and Automation Letters*, 3(3):1808–1815, 2018. 11, 12
- [6] Yookyung Choi, Namil Kim, Soonmin Hwang, Kibaek Park, Jae Shin Yoon, Kyoungwan An, and In So Kweon. Kaist multi-spectral day/night data set for autonomous and assisted driving. *IEEE Transactions on Intelligent Transportation Systems*, 19(3):934–948, Mar. 2018. 11, 12
- [7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 3213–3223, Las Vegas, NV, USA, June 2016. IEEE. 11, 12
- [8] Camille Couprie, Clément Farabet, Laurent Najman, and Yann Lecun. Indoor semantic segmentation using depth information. In *First International Conference on Learning Representations (ICLR 2013)*, pages 1–8, 2013. 11, 12
- [9] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 11, 12
- [10] Yimian Dai, Fabian Gieseke, Stefan Oehmcke, Yiquan Wu, and Kobus Barnard. Attentional feature fusion. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, page 3559–3568, Waikoloa, HI, USA, Jan. 2021. IEEE. 13, 14, 15, 16, 18, 25, 28
- [11] Clément Fredembach and Sabine Süsstrunk. Colouring the near-infrared. In *Color and imaging conference*, volume 16, pages 176–182. Society of Imaging Science and Technology, 2008. 19, 20, 22, 23, 25
- [12] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, page 3354–3361, Providence, RI, June 2012. IEEE. 11, 12
- [13] Tobias Gruber, Frank Julca-Aguilar, Mario Bijelic, and Felix Heide. Gated2depth: Real-time dense lidar from gated images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1506–1516, 2019. 9, 10

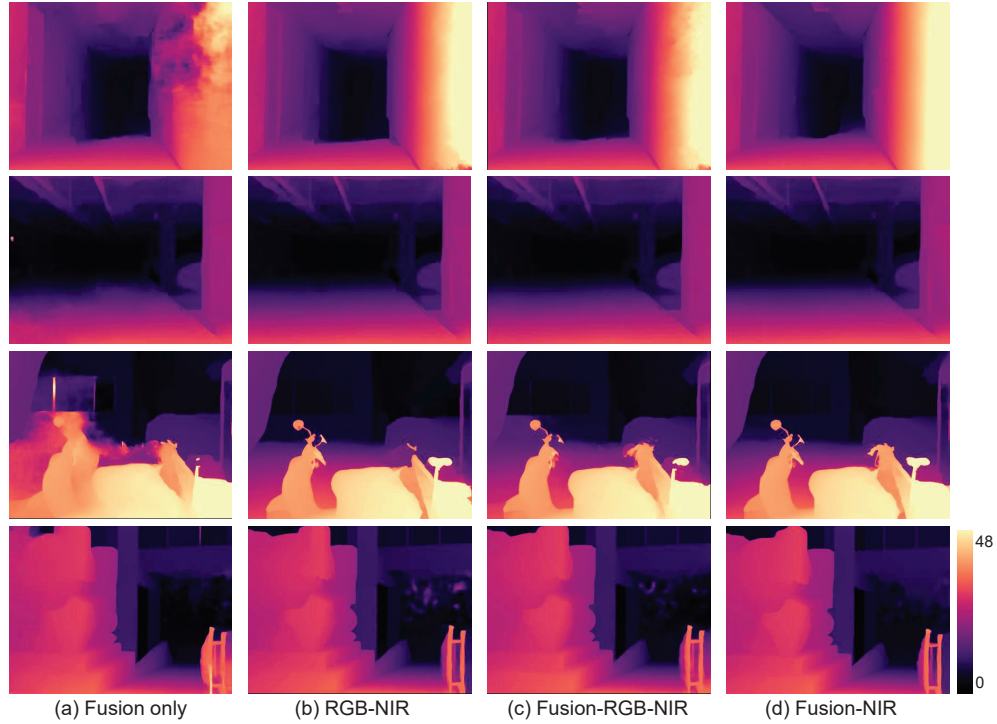


Figure 14. **Ablation study on alternative correlation volume searching.** (a) Fusion correlation volume only. (b) RGB and NIR correlation volumes. (c) Fusion, RGB and NIR correlation volumes. (d) Fusion and NIR correlation volumes.

- [14] Yubin Guo, Xinlei Qi, Jin Xie, Cheng-Zhong Xu, and Hui Kong. Unsupervised cross-spectrum depth estimation by visible-light and thermal cameras. *IEEE Transactions on Intelligent Transportation Systems*, 24(10):10937–10947, Oct. 2023. 11, 12, 23, 25
- [15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 12
- [16] María A Herrera-Arellano, Hayde Peregrina-Barreto, and Iván Terol-Villalobos. Color outdoor image enhancement by v-nir fusion and weighted luminance. In *2019 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)*, pages 1–6. IEEE, 2019. 19, 20, 22, 23, 25
- [17] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. In *Proceedings of the 19th annual conference on computer graphics and interactive techniques*, pages 71–78, 1992. 9
- [18] Shuangping Jin, Bingbing Yu, Minhao Jing, Yi Zhou, Jiajun Liang, and Renhe Ji. Darkvisionnet: Low-light imaging via rgb-nir fusion with deep inconsistency prior. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1104–1112, 2022. 19, 20, 22, 23
- [19] Jaewon Kam, Jungeon Kim, Soongjin Kim, Jaesik Park, and Seungyong Lee. *CostDCNet: Cost Volume Based Depth Completion for a Single RGB-D Image*, volume 13662 of *Lecture Notes in Computer Science*, page 257–274. Springer Nature Switzerland, Cham, 2022. 23
- [20] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):1–14, 2023. 6
- [21] Alex Junho Lee, Younggun Cho, Young-sik Shin, Ayoung Kim, and Hyun Myung. Vivid++: Vision for visibility dataset. *IEEE Robotics and Automation Letters*, 7(3):6282–6289, July 2022. 11, 12
- [22] Hui Li, Xiao-Jun Wu, and Josef Kittler. Infrared and visible image fusion using a deep learning framework. In *2018 24th international conference on pattern recognition (ICPR)*, pages 2705–2710. IEEE, 2018. 19, 20, 22, 23
- [23] Jiankun Li, Peisen Wang, Pengfei Xiong, Tao Cai, Ziwei Yan, Lei Yang, Jiangyu Liu, Haoqiang Fan, and Shuaicheng Liu. Practical stereo matching via cascaded recurrent network with adaptive correlation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 16242–16251, New Orleans, LA, USA, June 2022. IEEE. 17
- [24] Kaijiang Li, Hao Li, Haining Li, Peisen Wang, Chunyi Guo, and Wenfeng Jiang. Sirlut: Simulated infrared fusion guided image-adaptive 3d lookup tables for lightweight image enhancement. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 10220–10228, 2024. 23, 25
- [25] Peize Li, Kaiwen Cai, Muhamad Risqi U. Saputra, Zhuangzhuang Dai, and Chris Xiaoxuan Lu. Odombeyondvision: An indoor multi-modal multi-platform odometry dataset beyond the visible spectrum. In *2022 IEEE/RSJ International Conference on Intelligent*

Robots and Systems (IROS), page 3845–3850, Oct. 2022. 12

- [26] Shutao Li, Xudong Kang, and Jianwen Hu. Image fusion with guided filtering. *IEEE Transactions on Image Processing*, 22(7):2864–2875, July 2013. 14
- [27] Lahav Lipson, Zachary Teed, and Jia Deng. Raft-stereo: Multilevel recurrent field transforms for stereo matching. In *2021 International Conference on 3D Vision (3DV)*, pages 218–227. IEEE, 2021. 12, 14, 15, 16, 17, 18, 23, 24, 25, 26
- [28] Jingjing Liu, Shaoting Zhang, Shu Wang, and Dimitris N Metaxas. Multispectral deep neural networks for pedestrian detection. In *27th British Machine Vision Conference, BMVC 2016*, 2016. 11, 12
- [29] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048, 2016. 9, 10, 11, 24, 25, 27
- [30] Peter Mortimer, Raphael Hagmanns, Miguel Granero, Thorsten Luetzel, Janko Petereit, and Hans-Joachim Wuensche. The goose dataset for perception in unstructured environments. 2024. 11, 12
- [31] Long Quan and Zhongdan Lan. Linear n-point camera pose determination. *IEEE Transactions on pattern analysis and machine intelligence*, 21(8):774–780, 1999. 5
- [32] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. 21, 22
- [33] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *Pattern Recognition: 36th German Conference, GCPR 2014, Münster, Germany, September 2-5, 2014, Proceedings 36*, pages 31–42. Springer, 2014. 9, 24, 25, 27
- [34] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 6
- [35] Thomas Schops, Torsten Sattler, and Marc Pollefeys. Bad slam: Bundle adjusted direct rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 134–144, 2019. 24, 25, 27
- [36] Ukcheol Shin, Jinsun Park, and In So Kweon. Deep depth estimation from thermal image. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 1043–1053, Vancouver, BC, Canada, June 2023. IEEE. 11, 12
- [37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 25
- [38] Karasawa Takumi, Kohei Watanabe, Qishen Ha, Antonio Tejero-De-Pablos, Yoshitaka Ushiku, and Tatsuya Harada. Multispectral object detection for autonomous vehicles. In *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, pages 35–43, 2017. 11, 12
- [39] Jie Tang, Fei-Peng Tian, Boshi An, Jian Li, and Ping Tan. Bilateral propagation network for depth completion. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 9763–9772, Seattle, WA, USA, June 2024. IEEE. 23
- [40] Chaoran Tian, Weihong Pan, Zimo Wang, Mao Mao, Guofeng Zhang, Hujun Bao, Ping Tan, and Zhaopeng Cui. Dps-net: Deep polarimetric stereo depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3569–3579, 2023. 23, 24, 25
- [41] Alexander Toet, Maarten A Hogervorst, and Alan R Pinkus. The triclobs dynamic multi-band image data set for the development and evaluation of image fusion methods. *PloS one*, 11(12):e0165016, 2016. 11, 12
- [42] Abhinav Valada, Gabriel L Oliveira, Thomas Brox, and Wolfram Burgard. Deep multispectral semantic scene understanding of forested environments using multimodal fusion. In *2016 International Symposium on Experimental Robotics*, pages 465–477. Springer, 2017. 11, 12
- [43] Gangwei Xu, Xianqi Wang, Zhaoxing Zhang, Junda Cheng, Chunyuan Liao, and Xin Yang. Igev++: iterative multi-range geometry encoding volumes for stereo matching. *arXiv preprint arXiv:2409.00638*, 2024. 17, 18, 24, 25
- [44] Zixiang Zhao, Shuang Xu, Chunxia Zhang, Junmin Liu, and Jiangshe Zhang. Bayesian fusion for infrared and visible images. *Signal Processing*, 177:107734, 2020. 18, 19, 22, 23, 25
- [45] Tiancheng Zhi, Bernardo R Pires, Martial Hebert, and Srinivasa G Narasimhan. Deep material-aware cross-spectral stereo matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1916–1925, 2018. 11, 12