Adaptive Active Fusion of Camera and Single-Point LiDAR for Depth Estimation

Dang M. Tran, Nate Ahlgren, Chris Depcik, and Hongsheng He

***Abstract*— Depth sensing is an important problem in many applications, such as autonomous driving, robotics, and automation. This article presents an adaptive active fusion method for scene depth estimation by using a camera and a single-point light detection and ranging (LiDAR) sensor. An active scanning mechanism is proposed to guide laser scanning based on critical visual and saliency features, and the convolutional spatial propagation network (CSPN) is designed to generate and refine full depth map from the sparse depth scans. The active scanning mechanism generates a depth mask by using log-spectrum saliency detection, Canny edge detection, and uniform sampling, which indicate critical regions that require a high resolution of laser scanning. To reconstruct a full depth map, the designed CSPN network extracts affinity matrices from the sparse depth scans, while reserving global spatial information in the images. The performance of proposed method was evaluated and compared with the state-of-the-art methods on the NYU depth dataset v2 (NYUv2) and the experiment demonstrated its outperformance in reconstruction accuracy and robustness to measurement noise. The proposed method was also evaluated in real-world scenarios.**

***Index Terms*— Cost-effective light detection and ranging (LiDAR), deep learning, depth completion, sensor fusion, single-point LiDAR.**

I. Introduction

DEPTH sensing provides critical scene information for many applications, such as scene perception, autonomous robotic navigation, safety awareness, and environment modeling. Light detection and ranging (LiDAR) sensors are popularly used in these applications, and LiDAR-based depth estimation techniques have been extensively studied. Recently, low-cost LiDAR sensors have attracted more attention of the research community. A single-point LiDAR system costs below US $300 can scan up to 700,000 points with high accuracy and resolution [1]. As a trade-off, the system is not suitable for real-time applications due to the limited moving speed of its servo mechanism. The system requires 130 mins to capture one indoor scene [2]. In general, it is challenging to obtain complete depth sensing in real time using solely a single-point rangefinder. On the other hand, a camera can quickly capture visual information of an entire scene. It is, therefore, beneficial to fuse cost-effective camera and single-point LiDAR for depth estimation due to their complementary properties.

Cost-efficient LiDAR and RGB cameras have been integrated as affordable alternatives to commercial high-end LiDARs [3]. The classical method for sensor fusion is analytical modeling using inference, optimization, and interpolation [4]. Recently, learning approaches have been applied to sensor fusion and depth estimation. To recover dense depth from sparse laser scan data, sparse-invariant convolutional neural networks (CNNs) were developed for upsampling by introducing an effective sparse convolution layer that explicitly processes missing data during the convolution operation. For fast changing environments, several frameworks were designed to model a sequence of RGB images, e.g., the self-supervised training pipeline in autonomous driving [5]. In addition, domain-specific network architectures for depth estimation have been studied to overcome training bottlenecks and improve fusion performance [6]. Recent work discovered that integrated surface normal and attention map can improve the performance of depth estimation and handle occlusions [7]; however, the proposed networks require dense 3-D LiDAR or planar LiDAR scans [3]. Little progress has been made in the research of depth estimation by using single-point LiDAR, which demands efficient sampling strategies and fusion algorithms to process sparse scans.

The depth estimation methods based on dense data cannot be directly applied to a single-point LiDAR, which only samples one point during a scan. Dense scans for real-time depth sensing are solely available on high-end 3-D LiDARs [8], [9]. The primary challenge in depth estimation using low-cost single-point LiDAR is to balance scanning density and real-time performance. For instance, high-resolution point cloud requires dense sampling, and it takes time to direct the single-point laser for the scans. A mechanism that optimizes the sampling and distribution will significantly improve scanning efficiency and real-time performance. The second challenge is that image-based depth estimation usually produces blurry outputs, such as the depth completion model using CNNs, which are attributed to the inaccurate predictions from local extracted features. To obtain sharp depth maps, modeling of global information and iterative refinement of the depth maps become necessary.

In this article, we propose an adaptive active fusion method for depth estimation using an RGB camera and a laser range finder. The results of the proposed method are demonstrated in Fig. 1. Given an RGB image and a sparse depth map,

Manuscript received 17 October 2022; revised 26 April 2023; accepted 15 May 2023. Date of publication 8 June 2023; date of current version 28 June 2023. This work was supported in part by the Mid-America Transportation Center via a grant from the U.S. Department of Transportation's University Transportation Centers Program. The Associate Editor coordinating the review process was Dr. Martti Kirkko-Jaakkola. *(Corresponding author: Hongsheng He.)*

Dang M. Tran and Hongsheng He are with the Department of Computer Science, The University of Alabama, Tuscaloosa, AL 35487 USA (e-mail: hongsheng.he@ua.edu).

Nate Ahlgren and Chris Depcik are with the School of Engineering, University of Kansas, Kansas City, KS 66045 USA.

Digital Object Identifier 10.1109/TIM.2023.3284129

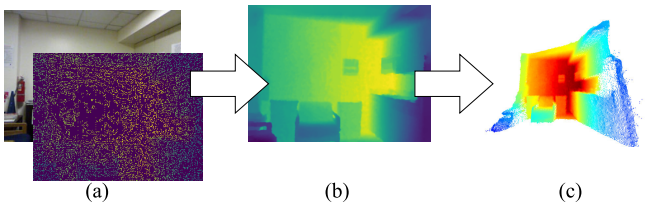

Fig. 1. Depth estimation using adaptive active fusion. (a) Pair of a 304 × 228 RGB image and the sparse depth map. (b) Depth estimation using the depth completing model. (c) Reconstructed 3-D point cloud.

the model estimates a complete dense depth map in real time, which can be converted into dense point cloud based on camera intrinsic parameters (69 312 points per second). We implemented an active scanning mechanism that improves scanning efficiency based on critical visual and saliency features. The mechanism selects informative scene regions while reducing the sampling ratio of the single-point LiDAR. To fuse the sparse depth scans and RGB images, we designed an encoder–decoder architecture with convolutional spatial propagation layers for depth construction and refinement. The spatial propagation layer is designed to preserve global information, which works as a bilateral filter that reduces estimation noise [10]. The proposed method was evaluated on the NYU depth dataset v2 (NYUv2) [11] in comparison with the state-of-the-art methods, including sparse-to-dense [6] and convolutional spatial propagation network (CSPN) [12]. The proposed method demonstrated the competitive performance in accuracy and robustness to measurement noise. The method improves the accuracy of depth estimation as compared with pure RGB-based methods, and it could be applied to improve the performance of line-scanning LiDARs.

The contributions of the article are summarized as follows.

1) An active scanning mechanism based on saliency detection and feature extraction has been implemented. The novel mechanism produces a visual mask that guides the scanning process in an efficient manner.
2) A depth estimation method with balanced sensing accuracy and speed has been designed in an encoder–decoder architecture with convolutional spatial propagation layers to fuse RGB images and sparse depth scans.
3) A cost-effective LiDAR system has been developed using the proposed mechanisms and evaluated in real-world scenarios.

II. Adaptive Active Fusion

The framework of the adaptive active fusion method is shown in Fig. 2, where the model takes RGB images and sparse depth scans as input and generates dense depth maps. The model contains an active scanning mechanism (Section II-A), an encoder–decoder architecture, and a convolutional spatial propagation layer (Section II-B). The active scanning mechanism takes an RGB image as input and produces three maps: saliency map, edge map, and an uniform grid. These three channels are fused as a scan map that indicates the regions that the single-point LiDAR should scan. The RGB image and the sparse depth scans are fed into the encoder–decoder model, and depth scans are also rerouted to the spatial propagation layer to produce affinity matrices. The outputs of the encoder–decoder are combined with results of the CSPN layer, which generates affinity matrices for iterative refinement. The final output of the model after iteration is the refined dense depth map, which is used to construct 3-D point cloud (Section II-C).

A. Active Scanning Mechanism

To improve the efficiency of the single-point LiDAR, we propose an active scanning mechanism that utilizes a various resolution across a scene, where regions with detailed textures or depth variations are sampled with a high resolution. As such, this mechanism allows the depth estimation model to reduce redundant scanning and highlight most critical information. The active scanning mechanism that integrates visual saliency and feature maps using the Hadamard product is defined as follows:

$$\mathcal{D} = \mathcal{S}(f) \odot E(G, \Theta) \odot \mathcal{G} \tag{1}$$

where $\mathcal{S}(f)$ is the filtered saliency map, $E(G, \Theta)$ is the detected edge map, and $\mathcal{G}$ is the uniform distributed grid mask.

Humans pay more attention to important and saliency scene regions. Such a mechanism can significantly improve the search efficiency and may boost up the performance of the depth estimation. In this article, we use the log-spectrum saliency detection method to compute the saliency map $\mathcal{S}^*(f)$ [13]. The log-spectrum saliency detection is a static method that applies a log-form representation of an image for salient localization. Log spectrum of an image, which is a frequency domain representation, has been commonly used in fast semantic categorization problem [14]. Images having the same semantic characteristics (such as indoor/outdoor and maximum capturing distance) will belong to the same log patterns. Given an image $X \in \mathbb{R}^{m \times n \times c}$, we can compute its log spectrum $\mathcal{L}(f)$ in frequency domain, using a Fourier transform method $\mathfrak{J}$. Spectral residuals of an image $\mathcal{R}(f)$ can be obtained from $\mathcal{L}(f)$ using statistical analysis as follows:

$$\mathcal{R}(f) = \mathcal{L}(f) - \frac{1}{n^2} J_n * \mathcal{L}(f) \tag{2}$$

where $*$ is a convolution operation, $(1/n^2)J_n * \mathcal{L}(f)$ is a heuristic estimation of the log spectra shape in the frequency domain, and J_n is the unit matrix size $n \times n$. Spectral residual $\mathcal{R}(f)$ is a compressed representation of saliency map in the frequency domain. It suppresses unimportant information and preserves nontrivial parts only. To obtain the saliency map $\mathcal{S}(f)$ in the spatial domain, inverse Fourier transformation $\mathfrak{J}^{-1}$ is applied on the computed spectral residuals $\mathcal{S}(f) = \mathfrak{J}^{-1}[\exp(\mathcal{R}(f))]^2$.

The inclusion of edge feature maps in the active scanning mechanism was inspired by the observation on the saliency maps. Saliency detection focuses on textures or regions with high entropy but disregards geometric shapes and local boundaries, which are crucial for depth estimation. Therefore, we integrated an edge detection model in the active scanning mechanism. In this article, we adopt the Canny edge detector [15] considering its computational efficiency for real-time performance. Canny detection algorithm uses

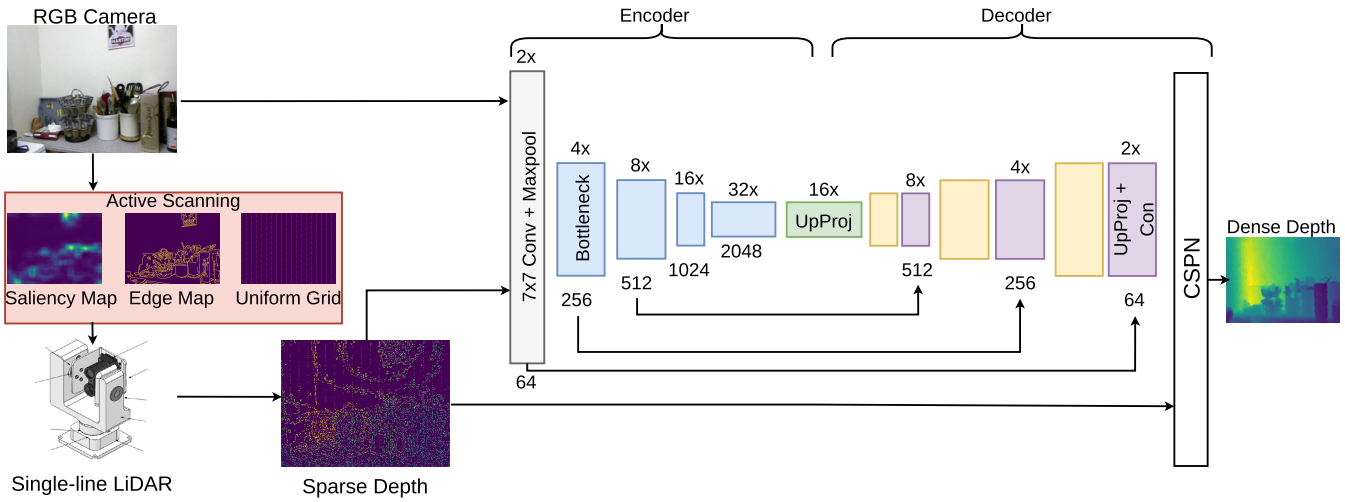

Fig. 2. Framework of the adaptive active fusion. Given RGB-D input (304 × 228 × 4), the tensor first goes through 7 × 7 convolutional layer, following with four downsampling steps and four upsampling steps. Sparse depth map is directly mapped as affinity to guide the depth completing process.

intensity characteristics of an edge to do recognition. Gradient first-order derivatives in horizontal direction G_x and vertical direction G_y can be obtained by convolution with a Sobel kernel. The obtained derivatives can be used to estimate the local gradient $G = (G_x^2 + G_y^2)^{1/2}$ and the direction angle $\Theta = \arg\tan(G_y/G_x)$. Once the gradient G and orientation angle Θ at each pixel are obtained, the Canny algorithm can be applied to detect valid edges. The Canny detector performs nonmaximum suppression on each pixels and returns all possible edges existing in the image. The hysteresis thresholding is then used to filter out invalid edges $E(G, \Theta)$.

Though the saliency and edge maps generate a compact representation of a scene as centralized regions, depth estimation using these featured regions alone achieves lower overall accuracy as compared with the uniform depth sampling [12], especially in noncentralized regions. To increase the generality of the scanning mask, we further refined the scanning mask by reducing the point density and integrating a uniform grid mask to capture information from noncentralized regions. The grid mask $\mathcal{G}$ contains pixels uniformly distributed across the entire image.

The scanning mask selects critical points to depth estimation and guides the scanning of the single-point LiDAR. To achieve a fast preprocessing speed, we typically choose a larger kernel size $n = 5$ for the log-spectrum-based saliency detection algorithm. The influencing parameters in the Canny detector are the minimal and maximal threshold values. Using the hold-out method on the NYUv2 dataset, we found the thresholds (100, 200) produced ideal edge features. The sampling percentage of the grid mask is set as 50% out of total points, allowing a sufficient number of nonzero pixels on the saliency map.

B. Depth Reconstruction

The computed saliency mask is combined with the RGB image as the input to the depth estimation model. Although a vanilla autoencoder could do a good job in depth predicting, the obtained results are usually blurred and have a lower resolution comparing with the original input [16]. The main challenge in the depth reconstruction problem is to maintain global spatial information during feature encoding. Although convolution can capture local relations, traditional convolutional-based models lack the ability to capture global information. Thus, we designed a spatial propagation layer that can help depth prediction process by providing global information propagated from the original inputs [12], where global features are represented in the form of affinity matrices. The model structure is shown in Fig. 2.

To reconstruct depth, the model takes an RGB image $X \in \mathbb{R}^{m\times n\times c}$ combined with saliency masks as input. A sparse depth image $D_0 \in \mathbb{R}^{m\times n}$ is rerouted directly to the spatial propagation layer, applying affinity matrices on the predicted depth. The autoencoder architecture starts with a convolution layer with a 7 × 7 size kernel followed by a max pooling. The generated feature map goes through four consecutive downsampling blocks, each of which will reduce the dimension by half while doubling the feature channel dimension. Following the work [17], each downsampling block is designed to be the three consecutive convolutional layers with the kernel sizes of [1, 3, 1], respectively, followed by rectified linear unit (ReLU) activation functions. The downsampling blocks are used to compress information of the input feature map into a latent representation. The output feature map from the encoder is fed to the decoder, which is a sequence of four upsampling blocks. Each upsampling block contains a padded convolutional layer while preserving the shape of the feature map, and an up-projection block, followed by a ReLU layer. Each upsampling step uses information from the feature channel to estimate surrounding pixels in the upscale feature map, which increases the shape of feature map by 2 while reducing the feature channel dimension by 2^2.

The spatial propagation layer combines global information in the original sparse depth to refine depth prediction. More specifically, the sparse depth map $D_0 \in \mathbb{R}^{m\times n}$ is rerouted directly to the CSPN, generating affinities matrices D_n through n iteration steps. SPNs [12] use the recurring relation between

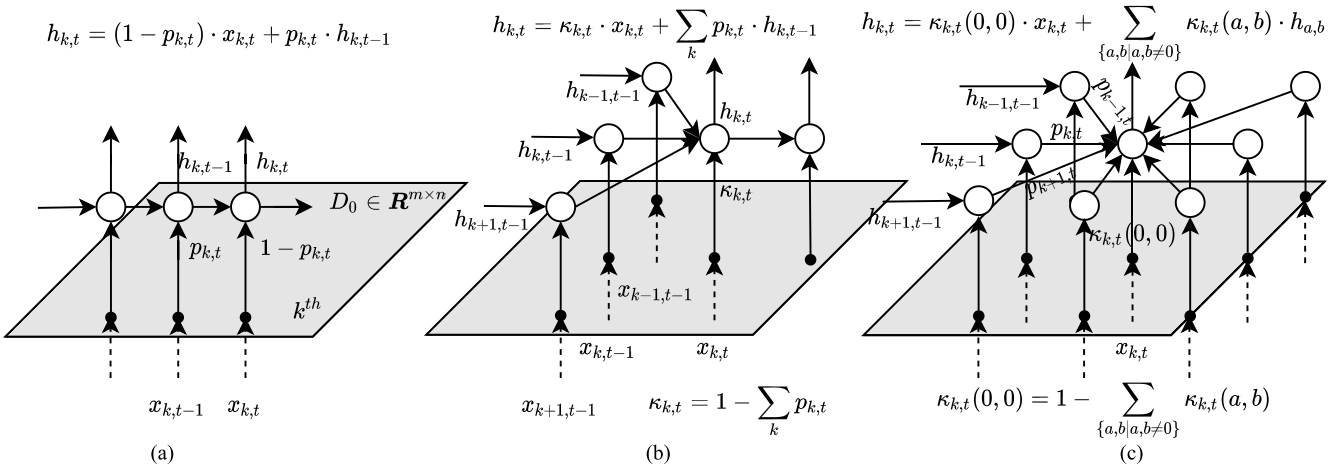

Fig. 3. Different spatial propagation strategies. (a) Spatial information is propagated linearly in specific direction as RNN. (b) Spatial information is propagated in specific many-to-one pattern (three-way connector). (c) Spatial information is propagated using neighbor pixels, similar to convolutional kernel operator.

nearby pixels to propagate spatial information in a specific direction. Affinity matrices are generated by combining the propagation results from different directions. An CSPN uses convolution kernels to instruct the propagating that computes affinity matrices in a single iteration in the anisotropic diffusion process. This approach increases the computation efficiency and learning ability [12].

The CPSN logic works as follows. Given sparse depth scans $D_0 \in \mathbb{R}^{m\times n}$, the spatial propagation layer generates a feature map in the hidden space $\boldsymbol{H} \in \mathbb{R}^{m\times n\times c}$, where c is the feature dimension. We denote $H_{i,j}$ as a cell of feature maps $\boldsymbol{H}$, where H_0 is the feature map $\boldsymbol{H}$ at iteration 0, and $H_{i,j,t}$ is the cell at i, j of $\boldsymbol{H}$ at iteration t. Using these notations, the CSPN defines the recurrence relation of hidden states as follows [18]:

$$H_{i,j,t+1} = \kappa_{i,j}(0,0) \odot H_{i,j,0} + \sum_{a,b=\frac{-(k-1)}{2}}^{\frac{(k-1)}{2}} \kappa_{i,j}(a,b) \odot H_{i-a,j-b,t} \tag{3}$$

where $\kappa \in \mathbb{R}^{k\times k}$ is the normalized kernel size. More specifically, $\kappa_{i,j}$ can be defined by normalizing equation

$$\kappa_{i,j}(a,b) = \begin{cases} \dfrac{\hat{\kappa}_{i,j}(a,b)}{\sum_{a,b\neq 0} \left|\hat{\kappa}_{i,j}(a,b)\right|}, & \text{if } a,b \neq 0 \\ 1 - \sum_{a,b\neq 0} \kappa_{i,j}(a,b), & \text{otherwise} \end{cases} \tag{4}$$

where $\hat{\kappa} \in \mathbb{R}^{k\times k}$ is a kernel of the size k. After n iterations, the CSPN returns a tensor $\boldsymbol{H}_n \in \mathbb{R}^{m\times n\times c}$, which is concatenated with the feature map from encoder–decoder components.

Fig. 3 explains the spatial propagation mechanism. Pixels of the depth image are used as the input to recurrent neural network (RNN) models. Information of pixel $x_{k,t-1}$ is propagated toward pixel $x_{k,t}$ through the recurrence relation between hidden states $h_{k,t-1}$, $h_{k,t}$. Linear recurrent propagation through 1-D sequence [18] was first proposed, as shown in Fig. 3(a). The idea was extended for multiple connectors [18] [Fig. 3(b)] and convolutional propagation [12] [Fig. 3(c)].

To train multiple regression models, we adopted the reversed Huber loss function [19] for depth estimation. The reversed Huber function is designed to be less sensitive to large weight values than $\mathcal{L}_2$ and less sensitive to small weight values than $\mathcal{L}_1$ [6]. More specifically, reversed Huber (denoted as berHu) is defined as follows:

$$\mathcal{B}(e) = \begin{cases} |e|, & \text{if } |e| \leq c \\ \dfrac{e^2 + c^2}{2c}, & \text{otherwise} \end{cases} \tag{5}$$

where e is the absolute difference between predicted depth $D_n \in \mathbb{R}^{m\times n}$ and ground truth $D^* \in \mathbb{R}^{m\times n}$, and c is 20% of the maximal absolute error within the working batch.

C. Point Cloud Representation

The estimated depth maps are converted into 3-D point clouds for noise filtering and refinement. To interchangeably transform depth maps to point clouds, we applied the depth-to-cloud conversion algorithm by using intrinsic camera parameters obtained from chessboard calibration. Given a depth image $D_n \in \mathbb{R}^{304\times 228}$, the algorithm maps each pixel at the uth row and vth column to appropriate point coordinate (x, y, z) using focal lengths f_x, f_y, center point offsets c_x, c_y, and the skew value s

$$D(u,v)\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{pmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{6}$$

where $D(u, v)$ is the depth value of the pixel (u, v) on the depth map. More specifically, the point coordination can be defined as follows:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{pmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}^{-1} D(u,v) \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{7}$$

where f_x, f_y, c_x, c_y, and s are intrinsic camera parameters.

III. Point Cloud Refinement

The reconstructed point cloud from a single 2-D depth scene will certainly be noisy and incomplete, due to the lack of full 3-D perception from a single view. Therefore, a noise filtering algorithm is required in postprocessing to improve

the quality and visualization of point clouds. Although surface interpolation [20] and projection-based filtering [21] can effectively remove outliers, these approaches are not suitable for nonsmooth surfaces and sparsely distributed point clouds. In addition, the point cloud obtained by the developed sensor is commonly discontinuous and multinomial distributed. In this article, we propose the iterative statistical outliers removal (ISOR) algorithm that can adaptively filter noises from contaminated point clouds by analyzing their statistical representation. The ISOR method is suitable for unorganized point cloud representation, especially for outdoor scenes. Different from the SOR method, which requires a complete point cloud as input, our proposed ISOR method can handle incomplete point clouds, using placeholder memory. Point clouds are divided into reasonable small patches and shifted into a shared memory. We developed a state machine to filter these small patches in parallel. The filtering decisions are based on the number of points in each patch and the movements of the servo motors.

Based on the current status of each patch, the statistical filtering operator is triggered, which determines by its geometric properties if a section in a patch should be removed. The local neighborhood distance is one of the intuitive geometric metrics for outlier detection. More specifically, given the point cloud $P = \{p_1, p_2, \ldots, p_n\}$, local densities are generated using the Kd-tree method. Each local density $Q_i = \{q_1^i, q_2^i, \ldots, q_k^i\}$, with respect to the querying point p_i ($i \in n$), contains k nearest neighbor points surrounding p_i. The average distance of Q_i with respect to p_i is computed by [22]

$$d_{p_i} = \frac{1}{k}\sum_{j=1}^{k} \left\| p_i - q_j^i \right\| \tag{8}$$

where $q_j^i \in Q_i$ and $\|p_i - q_j^i\|$ is the Euclidean distance between p_i and q_j^i. The extracted geometric features d_{p_i} are used to estimate local distribution by

$$LD(p_i) = \frac{1}{k}\sum_{j=1}^{k} e^{\left(\frac{-\|p_i - q_j^i\|}{d_{p_i}}\right)} \tag{9}$$

where $LD(p_i) \in [0, 1]$ is a statistical estimation of p_i and d_{p_i} is the average distance of Q_i with respect to p_i. The local distribution $LD(p_i)$ indicates the likelihood of point p_i to be an outlier. A point p_i is a local outlier if its distance to surrounding points p_j is significantly large. More specifically, outliers can be defined as follows:

$$\text{outliers} = \{p_i | LD(p_i) \geq \text{threshold}\}. \tag{10}$$

An equivalent representation of outlier set using cutoff value can be defined as follows:

$$L(p_i) = 1 - LD(p_i) \tag{11}$$

$$\text{outliers} = \{p_i | L(p_i) \leq \delta\} \tag{12}$$

where $\delta \in [0, 1]$ is the predefined cutoff value. For scenes with dense point distribution, we typically choose $\delta = 0.1$, which successfully filters unwanted outliers.

IV. Experiment

We evaluated the performance of the proposed adaptive active fusion method for depth estimation. Experiments were conducted on the NYUv2 dataset [11] and real-world scenarios. Though the RealSense can capture both RGB and depth streams, only the RGB channel was used for depth estimation. We compared the proposed method with three depth estimation approaches: CSPN [12], sparse-to-dense [6], and monocular [16]. We further investigated the impact of the active scanning mechanism on depth estimation in different sampling scales. We also compared the ISOR method with the SOR approach [22] on real-time performance and accuracy.

A. Experiment Setup

We trained and validated the proposed method on the NYUv2 dataset [11], which consists of 464 diverse indoor scenes with RGB and depth frames. We used 249 scenes for training and preserved 215 for testing. We also followed the augmentation procedure described in [6]. In total, there are 47 585 generated samples for training and 655 samples for testing. The dataset consists of various types of indoor environments, such as a living room, a basement, a kitchen, and a bedroom. Each data pair includes an RGB image $X \in \mathbb{R}^{640\times480}$ and a depth map $D^* \in \mathbb{R}^{640\times480}$. The RGB images and depth maps are downsampled and center-cropped into a fixed size of 304×228. The sparse depth map $D_0 \in \mathbb{R}^{304\times228}$ is generated from the complete depth map D^* using the sampling strategy described in Section II. The complete depth maps D^* are used as the ground truth during training.

The model was trained for 100 epochs using an adaptive learning rate (lr) initialized with $lr = 10^{-2}$. The weights of the encoder–decoder were first initialized from the pretrained model on the ImageNet dataset. With the learning rate $lr = 10^{-2}$, the training process took two days to complete; while with $lr = 10^{-3}$, the training process extended to weeks but became overfitting. To optimize the training process, we adopted the adaptive learning optimizer with $lr = 10^{-3}$ and reduced the lr by 20% when the root mean square error (RMSE) is sufficiently small (<0.12).

B. Performance of Depth Reconstruction

The qualitative results of the proposed method are shown in Fig. 4, which includes seven different indoor scenes from the NYUv2 dataset. The second column represents sparse depth maps D_0, which contain approximately 6000 points; the third column represents predicted depth maps, and the fourth column represents the ground truth. The red rectangles indicate the discrepancies between the estimation and the ground truth. As we can see in Fig. 4, the estimated depth and the corresponding ground truth are significantly similar in the close range (green) and far range (yellow). The proposed method can obtain high-resolution outputs in regions surrounding objects. Meanwhile, red rectangles between rows 1 and 7 mostly locate at planar areas, which are not commonly captured by the active scanning mechanism. The degraded performance on these regions is due to the lack of depth

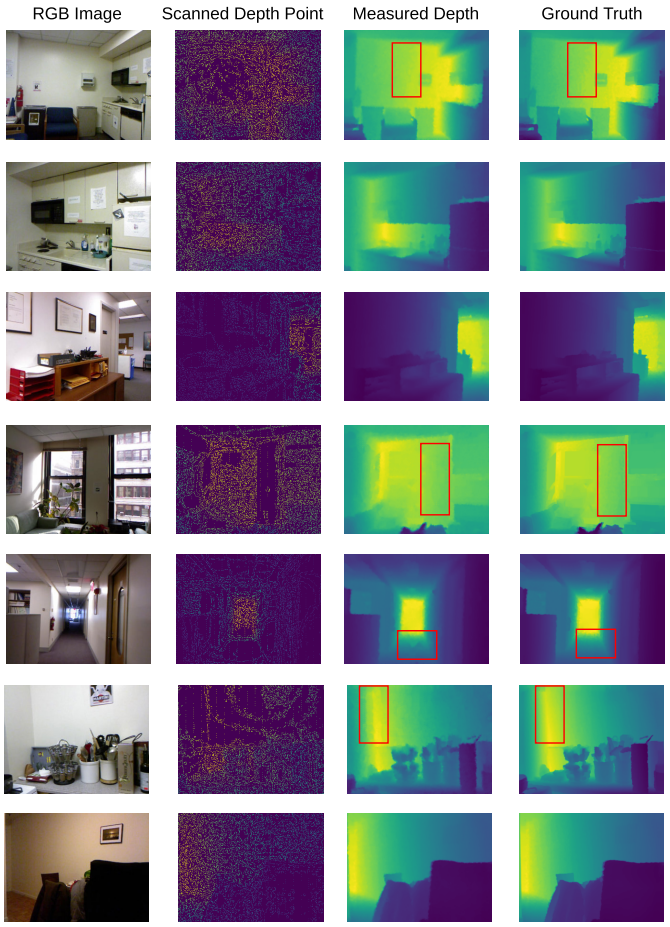

Fig. 4. Qualitative results of the proposed active fusion model.

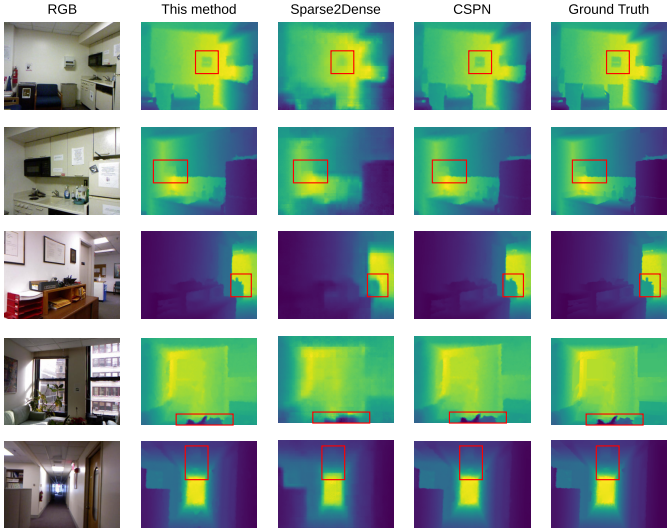

Fig. 5. Comparison between three depth estimation methods: 1) adaptive active fusion; 2) sparse-to-dense; and 3) CSPN.

TABLE I
QUANTITATIVE COMPARISON OF THE DEPTH COMPLETION MODELS ON NYUV2 VALIDATION SET

	This paper	CSPN [12]	Sparse2Dense [6]	Monocular [16]
MSE↓	0.003	0.005	0.034	–
RMSE↓	0.055	0.070	0.185	0.388
MAE↓	0.019	0.022	0.087	0.103
$\boldsymbol{\delta_{1.02}}$↑	99.9%	99.8%	98.7%	89.2%
$\boldsymbol{\delta_{1.05}}$↑	100.0%	100.0%	99.8%	96.4%
$\boldsymbol{\delta_{1.10}}$↑	100.0%	100.0%	99.9%	98.2%

information. In general, the proposed model performed well on both centralized regions and noncentralized regions.

We compared the proposed method with depth estimation approaches: CSPN [12] and sparse-to-dense [6], as shown in Fig. 5. The experiment demonstrates that the proposed method outperforms sparse-to-dense and CSPN in local regions surrounding objects (red rectangles). It should be noticed that the experiment was conducted for indoor scenes only, where the normal depth distances is less than 10 m.

To compare depth estimation performance, we used three different metrics that measure depth residuals: 1) RMSE, $E_{\text{RMSE}}(D, D^*) = ((1/|D|)\sum_{d\in D}|d^* - d|^2)^{1/2}$; 2) MSE, $E_{\text{MSE}}(D, D^*) = (1/|D|)\sum_{d\in D}(d^*-d)^2$; and 3) mean absolute error (MAE), $E_{\text{MAE}}(D, D^*) = \sum_{d\in D}|d^* - d|/|D|$, where $D \in \mathbb{R}^{304\times228}$ is the predicted depth map, $D^* \in \mathbb{R}^{304\times228}$ is the ground truth, and d, d^* are corresponding pixels of D, D^*. Beside residual measurements, we also evaluated model accuracy using pixel relative similarities with predefined thresholds. The method is known as delta, which is defined as $\delta_t = \max(d^*/d, d/d^*) < t$, where t is a threshold value and $\delta_t \in [0, 1]$. The common choices for t are $\{1.02, 1.05, 1.10\}$. Delta with lower values of t is much more sensitive when comparing pixel similarity between two depth maps.

Table I compares the performance of different depth estimation methods. Residual metrics (↓) compute the difference between depth estimation and ground truth. Smaller residual values indicate better model performance. On the other hand, delta metrics (↑) compute the similarity between the depth estimation and the ground truth. Larger δ_t values indicate that it is harder to visually distinguish differences between the estimation and the ground truth. According to the table, the proposed model obtains smallest residual values ($E_{\text{MSE}} = 0.003$, $E_{\text{RMSE}} = 0.055$, and $E_{\text{MAE}} = 0.019$) and highest δ_t values ($\delta_{1.02} = 99.9\%$, $\delta_{1.05} = 100\%$, and $\delta_{1.10} = 100\%$), which demonstrate the outperformance over other methods. More specifically, the proposed model achieves 1.6× better residual values than CSPN, 11.3× better than sparse-to-dense in MSE, and 7× better than the monocular model in RMSE. The proposed model is the only method with over 99% accuracy in the three scale t values.

We additionally evaluated the runtime performance of the three models on the NYUv2 dataset. All the depth estimation methods can produce dense depth maps within 0.2 s, which is faster than average human reaction. More specifically, the proposed method and CSPN can produce full dense depth maps (304 × 228) within 5–19 ms, and sparse-to-dense model

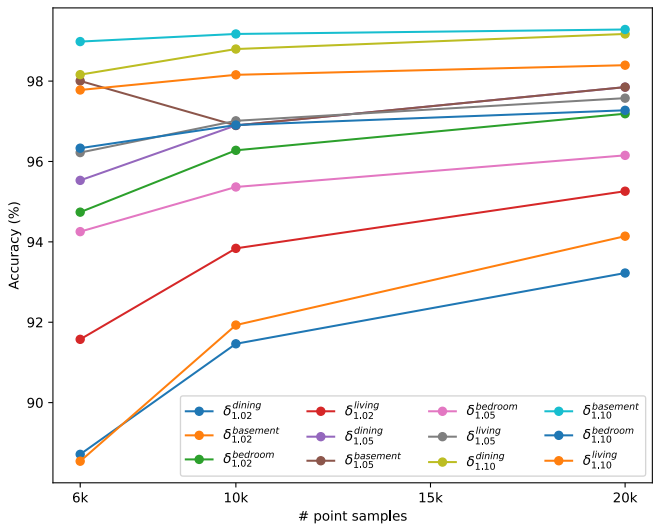

Fig. 6. Performance of model based on different depth scale inputs.

can produce a depth map within 0.8–4 ms. However, the depth maps constructed from CSPN and the proposed method have significantly higher resolutions than those from the sparse-to-dense, demonstrating a trade-off between resolution and real-time performance.

C. Robustness to Scanning Frequencies

To analyze the impact of the active scanning mechanism on the accuracy of depth estimation, we retrained and evaluated the proposed model using different sampling percentages. We performed experiments on depth densities from 6000 to 20 000 nonzero pixels (which is ∼30% of the image pixels). As we can see from Fig. 6, the larger the depth density, the better the model performs. The model performance boosted up by 3% (from 92.79% to 95.07%) on average in $\delta_{1.02}$ as the density increased from 6000 to 10 000. When the depth map is sufficiently dense, the impacts of the active scanning mechanism became negligible, which proves that the mechanism can indeed discover useful information for depth estimation, especially when the sampling frequency is low.

D. Point-Cloud Refinement

We evaluated point-cloud refining algorithms on the estimated depth maps and compared the developed ISOR method with the standard SOR filtering on five different indoor scenes by runtime performance and D-mean accuracy. D-mean metrics are defined as $d(P, P^*) = (1/2) \cdot d'(P, P^*) + (1/2) \cdot d'(P^*, P)$ where P, P^* are origin point cloud and filtered point cloud, respectively, d is D-mean point cloud distance, and $d'(P_{\text{target}}, P_{\text{ref}}) = \sum_{i=1}^{N} \|p_i - q_i\|$ is the Euclidean distance between p_i in target point cloud P_{target} and q_i in P_{ref}, which is the closest point to p_i. Additional outliers are added uniformly into the point cloud at different scales.

The qualitative results of the two methods are shown in Fig. 7. As we can see from the figure, the ISOR and SOR methods can filter out "global outliers"—outliers that are recognizable to not be a part of the origin point cloud. For less

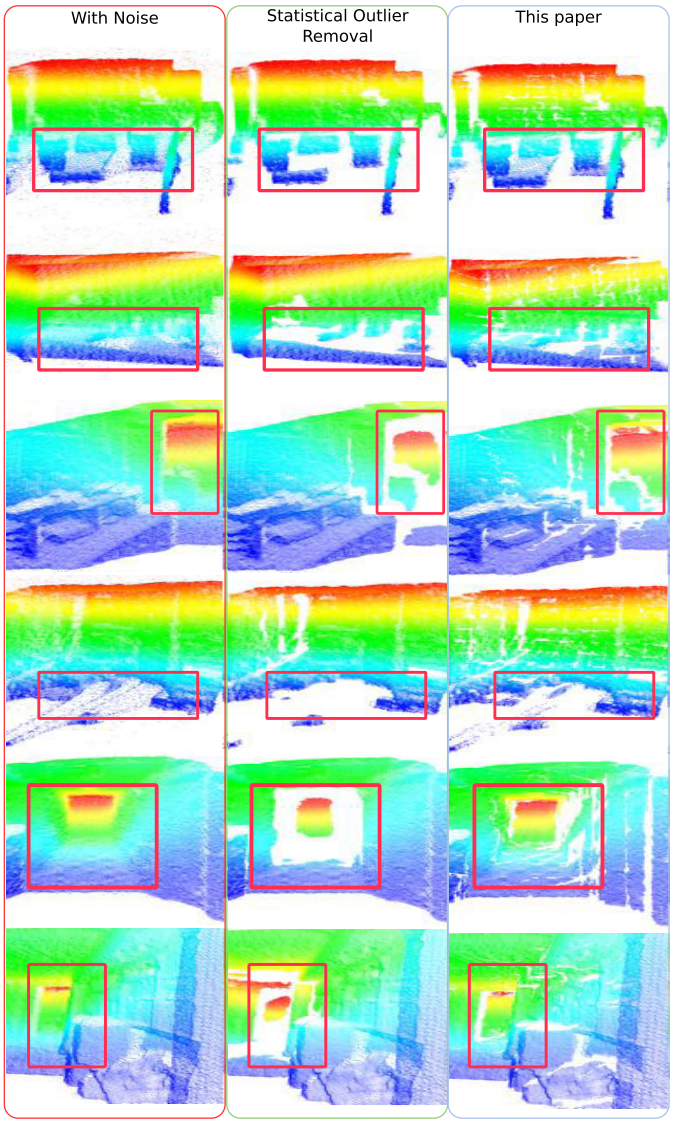

Fig. 7. Qualitative results of the point cloud refinement methods.

obvious regions, we can see that ISOR and SOR methods agree on which points are outliers. The major difference between the two methods is that the SOR method filters harder than the ISOR. The white strikes in the filtered results by ISOR are generated by the local iterative computing.

We compared the runtime performance and accuracy (using D-mean) of the two methods in Table II. The ISOR algorithm runs faster than SOR algorithms in small- and medium-sized point clouds. For the point cloud with a size $69\,312 = 304 \times 228$, ISOR took less than 1 s while SOR required 14 s. When the point cloud scales to $\geq 1\,00\,000$, the runtime performance become a critical issue. As we can see, the results from the ISOR method also have a smaller D-mean value in all scenarios. By observing more complete and global context, SOR removed much more points, which may be important information. On the other hand, ISOR filters less points by focusing on a local region, and thus, it runs fast and retains more information.

TABLE II
QUANTITATIVE COMPARISON OF TWO STATISTICAL OUTLIER REMOVAL METHODS

	This Paper (ISOR)		SOR	
	Runtime (s)↓	D-mean↓	Runtime (s)↓	D-mean↓
Office 1	1.132	32.288	14.664	556.623
Office 2	1.081	27.598	15.277	689.443
Office 3	1.064	3.6011	15.386	21.200
Office 4	1.008	43.447	14.108	4661.4
Office 5	1.072	32.288	14.204	556.623

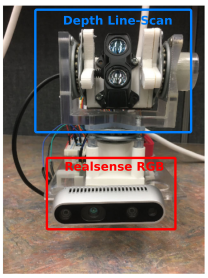

Fig. 8. Prototype of the camera-LiDAR system. The Garmin LiDAR-lite scanner rotates vertically and horizontally with defined angles, and the RealSense sensor captures RGB frames.

E. Real-World Evaluation

To evaluate the depth estimation model in real-world scenarios, we first collected testing data using a single-point depth estimation system. We have developed a prototype of the camera-LiDAR system, which includes a single-point laser scanner, Garmin LiDAR-lite v3, a RealSense D435i sensor, a processor Raspberry Pi v3, and a secure digital (SD) card for data storage, as shown in Fig. 8. The total cost for the development of the system is less than US $300, and the system is small enough to be mounted on a bicycle [23]. The system contains two servo motors that rotate horizontally and vertically to accumulate scanned points into 3-D point clouds. The system can be controlled using MATLAB, Python, and C++ programs. For the RGB channel, we used a RealSense D435i sensor that captures both RGB and depth frames; however, in this article, we only use the RGB channel of RealSense, with 640 × 480 resolution configuration.

We collected RGB images and sparse depth maps in indoor and outdoor environments using the developed LiDAR system and evaluated the proposed method for depth estimation in real-world scenarios. The results of the experiment are shown in Fig. 9. The first column shows the obtained RGB images, the second column shows the measured point clouds by the RealSense sensor, and the third column shows the depth estimation after ISOR filtering. Though the model was trained on the indoor dataset, the estimated depth on outdoor scenes (rows 6–9) matches the ground truth. In contrast, the depth estimation in indoor scenes (rows 1–5 and 10) is much more sensitive and noisy as compared with the outdoor scenarios. The noise in the estimated depth map may be primarily caused by the sensor noises and accumulated errors. The average RMSE, MSE, and MAE of depth estimation for ten real-world

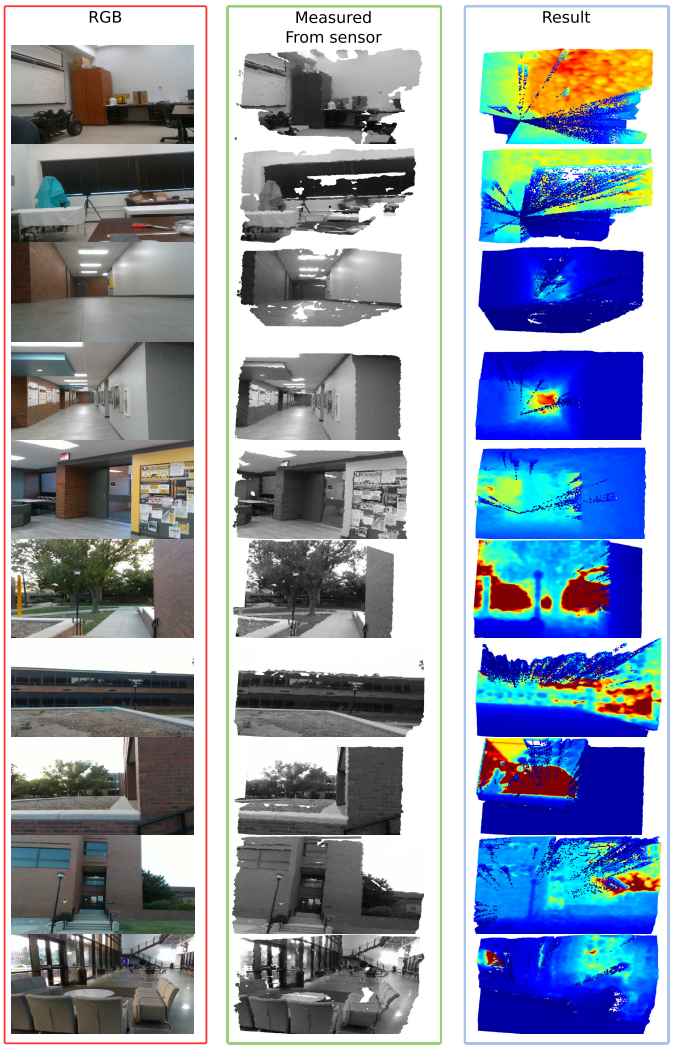

Fig. 9. Real-world performance of point cloud reconstruction in real-time.

office scenarios are 0.0942, 0.0107, and 0.0107, respectively, as compared with the depth map measured by the RealSense D435i sensor.

The performance of depth estimation model in real-world scenarios seems to be worse than the performance on training dataset NYUv2, both on indoor and outdoor scenes. It is unlikely to be overfitting problem, since we integrated various overfitting prevention techniques during training, such as early stopping (with patience = 10). One possible reason for such low performance in real-world scenes is the maximum distance used in the training dataset. More specifically, in NYUv2 indoor scenes, depth distances are between 0 and 10 m. Meanwhile, our captured outdoor scenes have depth values beyond this interval.

V. CONCLUSION

This article presented a novel depth estimation method using cost-efficient camera and single-point LiDAR sensors. We have shown that the active scanning mechanism helps improve the performance of depth estimation while reducing the requirement of depth scans. Through experiments, the active scanning mechanism and the depth estimation method

have been proved to be effective in terms of accuracy and computing speed. The method was designed for indoor environment with a maximum depth distance of 10 m. In addition, the system is currently not suitable for dynamic environments, due to the hardware limitations of the single-point LiDAR.

ACKNOWLEDGMENT

The contents reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein, and are not necessarily representative of the sponsoring agencies.

REFERENCES

[1] I. Blankenau, D. Zolotor, M. Choate, A. Jorns, Q. Homann, and C. Depcik, "Development of a low-cost LiDAR system for bicycles," in *Proc. SAE Tech. Paper Ser.*, Apr. 2018, pp. 1–9.

[2] J. Bennett et al., "Design of an efficient, low-cost, stationary LiDAR system for roadway condition monitoring," in *Proc. Saf. Eng., Risk, Rel. Analysis, Res. Posters*, vol. 13, Nov. 2021, Art. no. V013T14A047.

[3] C. Fu, C. Dong, C. Mertz, and J. M. Dolan, "Depth completion via inductive fusion of planar LiDAR and monocular camera," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 10843–10848.

[4] D. Held, J. Levinson, and S. Thrun, "Precision tracking with sparse 3D and dense color 2D data," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 1138–1145.

[5] F. Ma, G. V. Cavalheiro, and S. Karaman, "Self-supervised sparse-to-dense: Self-supervised depth completion from LiDAR and monocular camera," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 3288–3295.

[6] F. Ma and S. Karaman, "Sparse-to-dense: Depth prediction from sparse depth samples and a single image," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 4796–4803.

[7] J. Qiu et al., "DeepLiDAR: Deep surface normal guided depth prediction for outdoor scene from sparse LiDAR data and single color image," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3308–3317.

[8] C. Fu, C. Mertz, and J. M. Dolan, "LiDAR and monocular camera fusion: On-road depth completion for autonomous driving," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 273–278.

[9] Z. Sun, D. B. Lindell, O. Solgaard, and G. Wetzstein, "SPADNet: Deep RGB-SPAD sensor fusion assisted by monocular depth estimation," *Opt. Exp.*, vol. 28, no. 10, pp. 14948–14962, May 2020.

[10] J. T. Barron and B. Poole, "The fast bilateral solver," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 617–632.

[11] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2012, pp. 746–760.

[12] X. Cheng, P. Wang, and R. Yang, "Depth estimation via affinity learned with convolutional spatial propagation network," 2018, *arXiv:1808.00150*.

[13] X. Hou and L. Zhang, "Saliency detection: A spectral residual approach," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.

[14] A. Torralba and A. Oliva, "Statistics of natural image categories," *Netw., Comput. Neural Syst.*, vol. 14, no. 3, pp. 391–412, Jan. 2003.

[15] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[16] H. Basak, S. Ghosal, M. Sarkar, M. Das, and S. Chattopadhyay, "Monocular depth estimation using encoder–decoder architecture and transfer learning from single RGB image," in *Proc. IEEE 7th Uttar Pradesh Sect. Int. Conf. Electr., Electron. Comput. Eng. (UPCON)*, Nov. 2020, pp. 1–6.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015, *arXiv:1512.03385*.

[18] S. Liu, S. D. Mello, J. Gu, G. Zhong, M.-H. Yang, and J. Kautz, "Learning affinity via spatial propagation networks," 2017, *arXiv:1710.01020*.

[19] A. B. Owen, "A robust hybrid of lasso and ridge regression," *Contemp. Math.*, vol. 443, no. 7, pp. 59–72, 2007.

[20] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Point set surfaces," in *Proc. Visualizat. (VIS)*, Oct. 2001, pp. 21–29.

[21] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or, "Consolidation of unorganized point clouds for surface reconstruction," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 1–7, Dec. 2009.

[22] X. Ning, F. Li, G. Tian, and Y. Wang, "An efficient outlier removal method for scattered point cloud data," *PLoS ONE*, vol. 13, no. 8, Aug. 2018, Art. no. e0201280.

[23] T. Wiklund et al., "Design and development of a cost-effective LiDAR system for transportation," in *Proc. Saf. Eng., Risk, Rel. Anal.*, vol. 13, Nov. 2019, Art. no. V013T13A028.

Dang M. Tran received the B.S. degree in computer science from the School of Computing, Wichita State University, Wichita, KS, USA, in 2022.

He is currently a Ph.D. candidate with the University of Alabama, Tuscaloosa, AL, USA. His research interests include robotics, natural language processing, and machine learning.

Nate Ahlgren received the B.S. and M.S. degrees in mechanical engineering from the University of Kansas, Kansas City, KS, USA, in 2020 and 2022, respectively.

He is currently a Calibration Engineer at the University of Kansas.

Chris Depcik received the B.S. degree in mechanical engineering from the University of Florida, Gainesville, FL, USA, in 1997, and the M.S. degree in mechanical engineering, the M.S. degree in aerospace engineering, and the Ph.D. degree in mechanical engineering from the University of Michigan (UM), Ann Arbor, MI, USA, in 1999, 2002, and 2003, respectively.

He worked at UM as a Post-Doctoral Research Fellow and a Lecturer. He is an American Society of Mechanical Engineers Fellow and a Professor with the Mechanical Engineering Department, University of Kansas (KU), Kansas City, KS, USA. In addition, he has a Courtesy Appointment with the Aerospace Engineering Department, University of Kansas (KU). His laboratory conducts research revolving around a sustainable approach to energy and the transportation infrastructure. This includes analysis of electric vehicles, biofuels, combustion, and energy recovery potential. His group has authored or coauthored over 100 refereed articles.

Dr. Depcik received the Society of Automotive Engineers Ralph R. Teetor Award for his transportation-related research and educational activities.

Hongsheng He received the Ph.D. degree in electrical and computer engineering from the National University of Singapore, Singapore, in 2012.

He was an Assistant Professor with the School of Computing, Wichita State University, Wichita, KS, USA, from 2017 to 2023. He is currently an Associate Professor with the Department of Computer Science, The University of Alabama, Tuscaloosa, AL, USA. His research interests include robotics, artificial intelligence, and automation.