International Journal of Computer Vision
https://doi.org/10.1007/s11263-024-02188-3

Deep Unrolled Weighted Graph Laplacian Regularization for Depth Completion

Jin Zeng[1] · Qingpeng Zhu[2] · Tongxuan Tian[1] · Wenxiu Sun[2] · Lin Zhang[1] · Shengjie Zhao[1]

Received: 28 June 2023 / Accepted: 10 July 2024

Abstract
Depth completion aims to estimate dense depth images from sparse depth measurements with RGB image guidance. However, previous approaches have not fully considered sparse input fidelity, resulting in inconsistency with sparse input and poor robustness to input corruption. In this paper, we propose the deep unrolled Weighted Graph Laplacian Regularization (WGLR) for depth completion which enhances input fidelity and noise robustness by enforcing input constraints in the network design. Specifically, we assume graph Laplacian regularization as the prior for depth completion optimization and derive the WGLR solution by interpreting the depth map as the discrete counterpart of continuous manifold, enabling analysis in continuous domain and enforcing input consistency. Based on its anisotropic diffusion interpretation, we unroll the WGLR solution into iterative filtering for efficient implementation. Furthermore, we integrate the unrolled WGLR into deep learning framework to develop high-performance yet interpretable network, which diffuses the depth in a hierarchical manner to ensure global smoothness while preserving visually salient details. Experimental results demonstrate that the proposed scheme improves consistency with depth measurements and robustness to input corruption for depth completion, outperforming competing schemes on the NYUv2, KITTI-DC and TetrasRGBD datasets.

Keywords Depth completion · LiDAR sensor · Graph Laplacian regularization · Deep neural network

1 Introduction

The acquisition of precise scene depth is crucial for various applications, including automatic driving (Li et al., 2022), scene understanding (Chang et al., 2017), and augmented reality (Du et al., 2020), etc. However, depth sensors, such as LiDAR sensors, typically provide sparse depth maps, as illustrated in Fig. 1b and g. These sparse maps are impractical for downstream applications where dense depth maps are required (Uhrig et al., 2017). Therefore, numerous research efforts have been focused on depth completion from sparse depth maps using RGB image guidance to provide dense depth maps, as shown in Fig. 1c, d, h, i (Uhrig et al., 2017; Ma and Karaman, 2018; Van Gansbeke et al., 2019).

Recent works have made remarkable improvements in depth completion using deep neural networks (DNNs) (Cheng et al., 2019; Park et al., 2020; Hu et al., 2021; Zhao et al., 2021; Lopez-Rodriguez et al., 2022). However, existing schemes do not fully consider the preservation of sparse inputs, resulting in two issues: (1) the output dense depth is not guaranteed to be consistent with the sparse input, and (2) the accuracy of the output depth is sensitive to input corruption, such as sensor noise or misalignment between depth measurements and RGB images caused by synchronization or occlusion (Zeng et al., 2019; Qiu et al., 2019), as illustrated in Fig. 1g. Even state-of-the-art algorithms (Park

Communicated by Yasuyuki Matsushita.

✉ Qingpeng Zhu
zhuqingpeng226@gmail.com

Jin Zeng
zengjin@tongji.edu.cn

Tongxuan Tian
tongxuan259@gmail.com

Wenxiu Sun
sunwenxiu@sensetime.com

Lin Zhang
cslinzhang@tongji.edu.cn

Shengjie Zhao
shengjiezhao@tongji.edu.cn

1 School of Software Engineering, Tongji University, Shanghai, China

2 SenseTime Research, Shenzhen, China

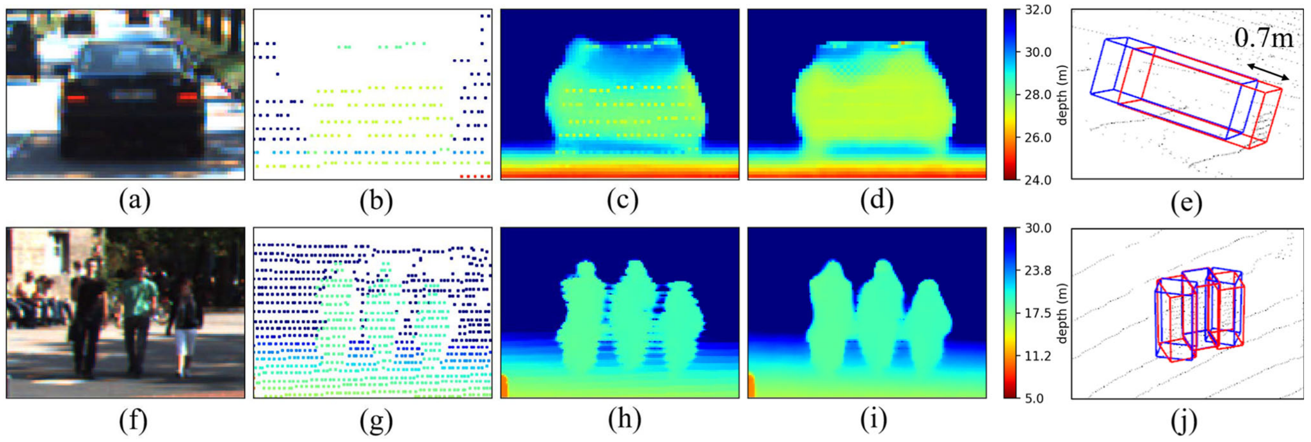


Fig. 1 Depth completion with different approaches: (**a**, **f**) RGB input; (**b**, **g**) sparse depth input; (**c**, **h**) results of NLSPN (Park et al., 2020), where the depth estimation is not consistent with input values in (**c**), and the edge accuracy degrades due to mis-alignment between depth input and RGB in (**h**); (**d**, **i**) results of proposed HUGNet with input constraints, where the results well preserve the input in (**d**) and is robust to depth-RGB mis-alignment in (**i**); (**e**, **j**) results of 3D object detection, colored in blue for NLSPN and red for HUGNet, where HUGNet provides more accurate car positions and pedestrian orientations (Color figure online)

et al., 2020; Liu et al., 2023; Zhang et al., 2023) suffer from the above two issues. In Fig. 1, we show the results of non-local spatial propagation network (NLSPN) (Park et al., 2020) for demonstration. For instance, Fig. 1c exemplifies the input inconsistency, showing an approximate 1 m offset between depth estimation and input LiDAR samples in the car's region. Moreover, Fig. 1h demonstrates the high sensitivity to noise, where the edge accuracy is severely degraded due to misalignment between input depth and RGB images. This hinders the application of depth maps in downstream tasks, e.g., using the predicted depth in 3D object detection (Ma et al., 2020), NLSPN results generate inaccurate car position in Fig. 1e and inconsistent pedestrian orientations in Fig. 1j.

To improve input fidelity, approaches based on spatial propagation networks (SPNs) replaced depth estimation at sample locations with valid input values during propagation (Cheng et al., 2019; Park et al., 2020). However, *such input replacement procedure for post-processing refinement leads to discontinuity surrounding sample locations*, as shown in Fig. 1c. Meanwhile, some methods (Eldesokey et al., 2020; Chodosh et al., 2018) incorporated model-based approaches with input constraints into deep learning frameworks to explicitly propagate the input to its neighbors. By considering input reliability in the propagation, these methods further enhance input corruption robustness. However, compared to recent schemes, these methods are less accurate in structural detail enhancement and prediction in large input-invalid areas, which is *due to the limited expressiveness of their model-based network designs*.

In this paper, we propose *deep unrolled Weighted Graph Laplacian Regularization* (WGLR) which enhances input fidelity and noise robustness, as shown in Fig. 1d, i. This further improves the accuracy of downstream 3D object detection application, where the proposed approach provides more accurate car positions in Fig. 1e and pedestrain orientations in Fig. 1j. In particular, we assume graph Laplacian regularization (GLR) as the data smoothness prior for depth completion optimization in an uncertainty-aware manner, and derive the WGLR solution by solving the optimization in the continuous domain with correct boundary term to enforce input consistency. Based on its anisotropic diffusion interpretation, WGLR solution is efficiently implemented by unrolling WGLR into iterative filtering, and thus can be recast into a trainable module in the deep network. The unrolled WGLR is then incorporated into the DNN framework to develop high-performance yet interpretable network with explicit input constraints. Additionally, the unrolled WGLR is adopted in a hierarchical manner to successively diffuse depth in multiple scales, ensuring global smoothness while preserving structural details. The proposed network is named as ***H**ierarchical Deep **U**nrolled W**G**LR Network, referred to as* **HUGNet** for short.

Compared with SPN-based schemes (Cheng et al., 2019; Park et al., 2020), we provide continuous-domain analysis to explain how the proposed WGLR corrects the boundary term to avoid discontinuity in SPN-based schemes. Compared with pure DNN-based schemes (Zhao et al., 2021; Hu et al., 2021), our approach incorporates the WGLR prior with input constraint in the network design so as to explicitly enforce input fidelity. While there are also schemes incorporating input constraints in the network (Eldesokey et al., 2020; Chodosh et al., 2018), they sacrifice detail preservation due to limited expressiveness of the model-based network design. Instead, we combine the merits of the two, i.e., the robustness of data prior and learning power of DNN, to produce a network that ensures input fidelity while maintaining structural details. The contributions of our work are summarized as follows.

- We assume GLR prior in depth completion optimization and derive the WGLR solution via continuous-domain analysis to enforce confidence-aware input consistency;
- We design unrolled WGLR to efficiently implement WGLR solution based on its diffusion interpretation, enabling WGLR to be recast into a trainable module in the deep network;
- We incorporate unrolled WGLR into DNN to develop high-performance yet interpretable network with explicit input constraints; unrolled WGLR is adopted to diffuse the depth in a hierarchical manner to ensure global smoothness while preserving structural details.

Experiments validate the advantages of the proposed deep unrolled WGLR in input consistency and noise robustness with detail preservation. The paper is organized as follows. Related works are discussed in Sect. 2, and Sect. 3 provides a detailed discussion of the proposed algorithm with the network design in Sect. 4. Comparison with competing schemes and ablation study are demonstrated in Sect. 5. The work is concluded in Sect. 6.

2 Related Works

In this section, we will first overview the learning-based schemes for depth completion, and focus on the learning-based schemes with input constraints for network design that are most related to the proposed approach.

2.1 Learning-Based Depth Completion

With the development of DNNs, deep learning based methods provide the state-of-the-art performance for depth completion and outperform model-based methods (Ferstl et al., 2013; Liu et al., 2015; Barron and Poole, 2016) by a wide margin. Early methods relied only on sparse depth maps and designed sparsity-invariant convolution for sparse input (Uhrig et al., 2017; Huang et al., 2019), but the resulting depth completion suffered from blurry edges and missing structural details, so recent methods utilize RGB images as the guidance for accurate detail preservation in depth prediction.

RGB-guided depth completion focuses on the fusion between two modalities (Ma and Karaman, 2018; Van Gansbeke et al., 2019; Li et al., 2020; Hu et al., 2021; Zhao et al., 2021). Sparse-to-Dense (Ma and Karaman, 2018) concatenated RGB and sparse depth map and fed them into an encoder-decoder network for final prediction. FusionNet (Van Gansbeke et al., 2019) fused the local information from sparse depth and the global guidance map learnt from RGB and sparse depth, weighted by their respective confidence maps. PENet (Hu et al., 2021) generated depth maps from color-dominant and depth-dominant branches respectively then adaptively fused the two complimentary depth maps. ACMNet (Zhao et al., 2021) designed attention-based graph propagation to extract multi-modal features from RGB and depth which were then combined with the gated fusion. Some works introduced intermediate representations to explore extra constraints for depth estimation. For example, DeepLiDAR (Qiu et al., 2019) and DepthNormal (Xu et al., 2019) estimated the surface normal as the intermediate representation used as an additional constraint for depth estimation. GAENet (Chen et al., 2022) integrated implicit 3D geometric structure information into 2D learning architecture to guide depth estimation.

Recent learning-based approaches achieve high accuracy and sharp details, but do not guarantee consistency with sparse input since the input values are not explicitly preserved, and are often vulnerable to input corruption. This motivates works to incorporate input constraints in the network design discussed as follows.

2.2 Learning-Based Depth Completion with Input Constraints

Recent works notice the offset between output depth estimation and input depth measurement, thus incorporate input constraints into network design to improve input fidelity. For example, CSPN (Cheng et al., 2019) and its variants, *e.g.*, NLSPN (Park et al., 2020) and GraphCSPN (Liu et al., 2022), adopted input replacement which replaced estimation at sample locations with input values during propagation. However input replacement resulted in discontinuity at sample locations as illustrated in Fig. 1c.

On the other hand, some works combined the prior knowledge of depth image and deep neural network, enabling more interpretable control on output than pure learning-based methods, since they explicitly propagated and rectified corrupted measurements. In Chodosh et al. (2018), the depth completion was formulated as compressed sensing with input constraint, which was solved with a deep recurrent auto-encoder based on Alternating Direction Neural Networks. PNCNN (Eldesokey et al., 2020) assumed the depth image as a multivariate normal distribution and explicitly propagated the input to its neighbors conditioned on its reliability using the learnt applicability function. Although the prior knowledge enabled control on output, these methods were less effective in reconstructing structural details and estimating depth in large input-invalid area, since the incorporated prior limited their network expressive power.

Therefore, it is challenging to combine the advantages of both in the network design: (1) accurate structural details using learning-based approaches and (2) input fidelity using model-based approaches. *In contrast, our proposed deep unrolled WGLR combines the merits of both to improve confidence-aware input fidelity while maintaining the structural details.*

2.3 Depth Rectification with Sparse Measurements

While sparse input fidelity is essential in depth completion, it is also considered in other depth estimation scenarios. For example, Pseudo-LiDAR++ (You et al., 2020) rectified the stereo depth with the sparse LiDAR depth via graph-based depth correction (GDC) for accurate 3D object detection, but GDC required solving linear system and was not implemented efficiently. To speed up GDC, FusionDepth network (Feng et al., 2022) fused monocular image features and sparse LiDAR features to correct monocular depth estimation with sparse LiDAR input using GDC as the teacher module, though FusionDepth was less accurate than GDC. Unlike previous works that do not well balance the accuracy and efficiency in depth rectification with sparse input, *we implement the depth completion with input constraint efficiently without sacrificing accuracy via algorithm unrolling based on the diffusion interpretation of WGLR.*

3 Problem Formulation and Algorithm Design

In this section, we propose the unrolling of weighted graph Laplacian regularization (WGLR) for depth completion which enforces input consistency conditioned on its reliability. First, by solving the depth completion optimization in continuous domain, we derive WGLR solution by extending the point integral method so as to properly propagate the sparse input values. Then, to speed up implementation, we unroll the WGLR solution into iterative filtering via its anisotropic diffusion interpretation, based on which the network architecture is designed in the sequel.

3.1 Problem Formulation

In this paper, we are interested in reconstructing the dense depth map $\mathbf{x} \in \mathbb{R}^N$ from the sparse depth map $\mathbf{y} \in \mathbb{R}^N$ produced by the depth sensor with inherent noise $\boldsymbol{\epsilon} \in \mathbb{R}^N$, where $\mathbf{x}$, $\mathbf{y}$ and $\boldsymbol{\epsilon}$ are in the vectorized form, N is the total number of pixels in the depth map. The relation between $\mathbf{x}$ and $\mathbf{y}$ is described as

$$\mathbf{y} = \mathbf{m} \odot \mathbf{x} + \boldsymbol{\epsilon}, \tag{1}$$

where $\odot$ is Hadamard product, $\mathbf{m}$ is the binary mask, *i.e*, $m_i = 1$ indicates the depth at the i-th pixel is measured by the sensor, $i = 1, \ldots, N$.

Due to the ill-posedness of the problem, extra prior knowledge describing the characteristics of $\mathbf{x}$ is required to facilitate the reconstruction (Milanfar, 2012). Here we use the following energy functional which is widely adopted in depth image processing (Barron and Poole, 2016; Pang and Cheung, 2017; Hu et al., 2013),

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2} \sum_{i,j} w_{i,j} \left(x_i - x_j\right)^2, \tag{2}$$

where $w_{i,j}$ is the (i, j)-th element of the adjacency matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$, indicating the similarity between pixel i and j. With $\mathbf{W}$, we then define the undirected weighted graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$ where $\mathcal{V}$ is composed of the N pixels as nodes connected via edges $\mathcal{E}$ with weights $\mathbf{W}$ on the neighborhood graph $\mathcal{G}$. Diagonal degree matrix $\mathbf{D}$ has diagonal entries $D_{i,i} = \sum_j w_{i,j}, \forall i$. A *graph Laplacian matrix* $\mathbf{L}$ is defined as $\mathbf{L} \triangleq \mathbf{D} - \mathbf{W}$, then (2) can be rewritten as

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2} \mathbf{x}^\top \mathbf{L} \mathbf{x}, \tag{3}$$

which is referred to as the well-known *graph Laplacian regularizer* (GLR) (Ortega et al., 2018), which is validated to well preserve the piece-wise smooth property of depth images (Cheung et al., 2018; Pang and Zeng, 2021), and thus adopted as signal prior in our approach.

Considering the reliability of depth samples varies, we generalize the binary mask $\mathbf{m}$ to the confidence map $\mathbf{c} \in \mathbb{R}^N$ in the range of $[0, 1]$. To seek the *maximum a posteriori* (MAP) solution of $\mathbf{x}$ (Bishop and Nasrabadi, 2006), the optimization is formulated as

$$\min_{\mathbf{x}} \; \frac{1}{2} \|\mathbf{c} \odot (\mathbf{x} - \mathbf{y})\|_2^2 + \frac{\lambda}{2} \mathbf{x}^\top \mathbf{L} \mathbf{x} \tag{4}$$

where λ is the weight for the GLR prior. The first term is the uncertainty-aware data fidelity term, where $\mathbf{x}$ is optimized to be close to $\mathbf{y}$ conditioned on the confidence $\mathbf{c}$. Meanwhile, the second term is the GLR prior which enforces $\mathbf{x}$ to be smooth with respect to the graph $\mathcal{G}$.

3.2 WGLR Solution Derivation

Next, we derive solution for (4). While previous works solve (4) in the discrete domain and suffer from input inconsistency, we derive the solution in continuous domain to mitigate input inconsistency.

3.2.1 GLR Solution in Discrete Domain

Previous works solved the optimization in (4) by differentiating the objective function with respect to $\mathbf{x}$, and set the result equal to 0 to get the solution as follows which we denote as *GLR solution*:

$$\mathbf{C}(\mathbf{x}-\mathbf{y})+\lambda\mathbf{L}\mathbf{x}=0, \tag{5}$$

where $\mathbf{C}=\text{diag}(\mathbf{c})$. SPN-based approaches (Cheng et al., 2019; Park et al., 2020) are special cases of GLR solution which we will discuss in Sect. 3.3. However, it is observed that GLR solution does not preserve input. Direct input replacement as adopted in Cheng et al. (2019), Park et al. (2020) results in discontinuity at sample locations. For demonstration, we test on a sample image from KITTI-DC dataset (Uhrig et al., 2017) where we show the output depth image of NLSPN (Park et al., 2020) in Fig. 2c with discontinuity at sample locations. In Fig. 2e, we plot the depth values along the green line highlighted in RGB, where the output depth has an offset of about 1 m from input, validating that GLR solution does not preserve input.

3.2.2 Proposed WGLR Solution in Continuous Domain

In contrast to previous works, we solve the optimization in continuous domain. We assume the pixels in $\mathbf{x}$ are discrete samples on the continuous manifold $\mathcal{M}$ (Osher et al., 2017), then the solution in the continuous form is a Laplace-Beltrami equation (Shi et al., 2018) over the manifold $\mathcal{M}$ as follows,

$$c(i)(x(i)-y(i))+\lambda\Delta_{\mathcal{M}}x(i)=0, \quad \mathbf{p}_i\in\mathcal{M}, \tag{6}$$

where $\Delta_{\mathcal{M}}$ is the Laplace-Beltrami operator on $\mathcal{M}$. $\mathbf{p}_i$ is the point on $\mathcal{M}$ corresponding to the i-th depth pixel. x, y and c are the continuous functionals with $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{c}$ as their discrete counterparts. For indexing, we write $x(\mathbf{p}_i)$ as $x(i)$ for simplification.

The Laplace-Beltrami operator in (6) is defined on continuous $\mathcal{M}$, but the observations of $\mathcal{M}$ is discrete and finite. To solve (6), we adopt the point integral method (Li et al., 2017) to approximate (6) with the following integral,

$$c(i)(x(i)-y(i))+\lambda\int_{\mathcal{M}}(x(i)-x(j))w(i,j)d\mathbf{p}_j$$
$$+2\mu\lambda\int_{\partial\mathcal{M}}(x(j)-y(j))w(i,j)c(j)d\mathbf{p}_j=0, \tag{7}$$

where $\partial\mathcal{M}$ is the boundary of $\mathcal{M}$ and the sparse input $\mathbf{y}$ lie on $\partial\mathcal{M}$. The third term is referred to as boundary term

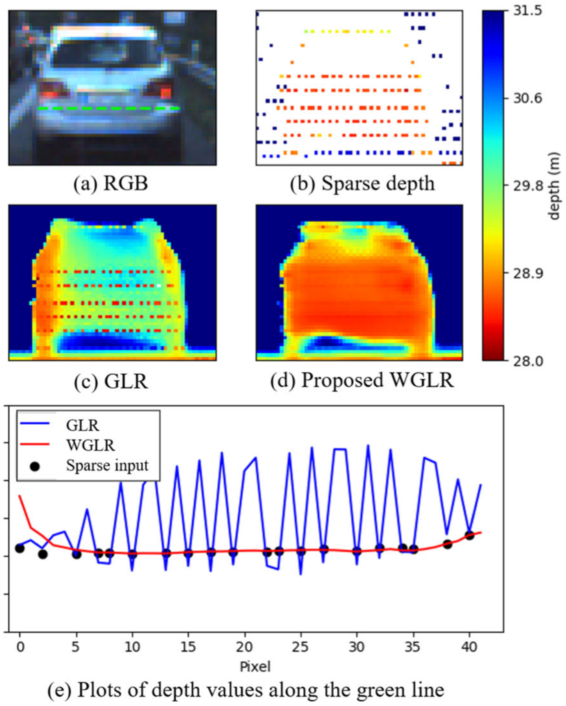


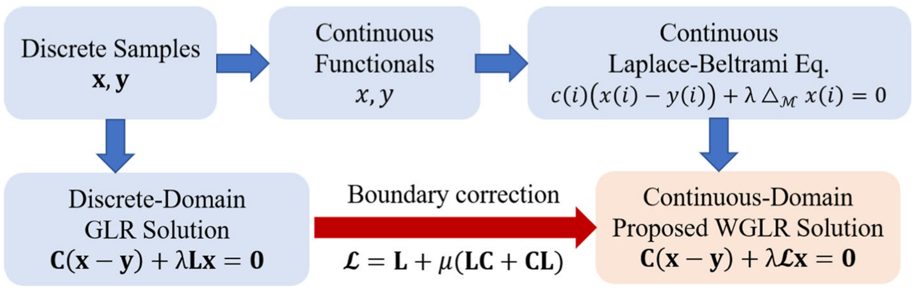
Fig. 2 Depth completion comparison between GLR and WGLR: (**a**) RGB image with green line indicating the locations of plotted depth values in (**e**); (**b**) sparse input; (**c**, **d**) results of GLR and WGLR solutions; (**e**) plots of depth values along the green dashed line in (**a**), where GLR has about 1 m offset with input, and WGLR well preserves input (Color figure online)

with weight μ which makes the sparse input values correctly spread to neighboring pixels and gives much better recovery. We then discretize (7) with

$$c_i(x_i-y_i)+\lambda\sum_{j=1}^{N}(x_i-x_j)w_{i,j}$$
$$+2\mu\lambda\sum_{j=1}^{N}(x_j-y_j)w_{i,j}c_j=0, \forall i. \tag{8}$$

However, the linear systems in (8) is asymmetric which makes the numerical solver inefficient. Based on the smoothness assumption of $\mathbf{x}$ and the local support of the weight $w_{i,j}$, we replace x_j with x_i in the third term to achieve symmetry. Assume y_j is approximated with x_j, (8) becomes

$$c_i(x_i-y_i)+\lambda\sum_{j=1}^{N}w_{i,j}(x_i-x_j)$$
$$+\mu\lambda\sum_{j=1}^{N}(c_jw_{i,j}+c_jw_{j,i})(x_i-x_j)=0. \tag{9}$$



Discrete Samples
x, y
Continuous Functionals
x, y
Continuous Laplace-Beltrami Eq.
$c(i)\big(x(i)-y(i)\big)+\lambda\,\Delta_{\mathcal{M}}\,x(i)=0$
Discrete-Domain GLR Solution
$\mathbf{C}(\mathbf{x}-\mathbf{y})+\lambda\mathbf{L}\mathbf{x}=\mathbf{0}$
Boundary correction
$\mathcal{L}=\mathbf{L}+\mu(\mathbf{LC}+\mathbf{CL})$
Continuous-Domain Proposed WGLR Solution
$\mathbf{C}(\mathbf{x}-\mathbf{y})+\lambda\mathcal{L}\mathbf{x}=\mathbf{0}$

Fig. 3 Summary of key concepts, where a blue arrow pointing from block A to block B means B is derived from A (Color figure online)

By rewriting (9) in matrix form, the solution is expressed as

$$\mathbf{C}(\mathbf{x}-\mathbf{y})+\lambda(\mathbf{L}+\mu\mathbf{LC}+\mu\mathbf{CL})\mathbf{x}=0, \tag{10}$$

which is equivalent to reweight the original graph Laplacian $\mathbf{L}$ as $\mathcal{L}=\mathbf{L}+\mu(\mathbf{LC}+\mathbf{CL})$ to give the solution

$$\mathbf{C}(\mathbf{x}-\mathbf{y})+\lambda\mathcal{L}\mathbf{x}=0. \tag{11}$$

Hence we call this solution the *weighted graph Laplacian regularization*, named **WGLR** for short.

3.2.3 Comparison Between GLR and WGLR

By comparing WGLR in (10) with GLR in (5), GLR drops an non-negligible term $\mu(\mathbf{LC}+\mathbf{CL})\mathbf{x}$ thus suffers from the sample discontinuity, while WGLR corrects the boundary term in (7) and enforces input preservation as shown in Fig. 2d.

3.2.4 Interpretation of WGLR

WGLR alleviates discontinuity by enlarging the weight for high confidence samples, i.e., typically the depth samples. Specifically, using the original $\mathbf{L}$, when the samples are very sparse, even if the summation of the difference between the depth samples and its neighbors is large, it is overwhelmed by the summation over invalid pixels. In this case, the continuity is sacrificed in the optimization. On the other hand, when using the weighted $\mathcal{L}$, the weight for the summation over depth samples gets enlarged, so the solution is enforced to adhere to the input samples and alleviate the discontinuity. A special case is when the samples are dense, i.e., $\mathbf{C}=\mathbf{I}$, then $\mathcal{L}$ becomes the original $\mathbf{L}$.

The key concepts are summarized in Fig. 3. The question that remains is how to efficiently solve (11), which is addressed as follows.

3.3 Unrolled WGLR Algorithm

Even if the linear system in (11) is sparse, symmetric and positive definite and can be solved efficiently with conjugate gradient (CG) based approaches (Shewchuk, 1994), it is still computationally expensive and cannot enable real-time implementation. Therefore, to further speed up the algorithm, we unroll the WGLR solution into iterative filtering based on its anisotropic diffusion interpretation, which is efficiently accomplished with layers of convolution. Algorithm unrolling has also been used in image processing (Monga et al., 2021), but we focus on anisotropic image completion which has not been addressed. Detailed algorithm design is as follows.

Let $\mathcal{D}$ and $\mathcal{W}$ denote the degree and adjacency matrices of $\mathcal{L}$, (11) is rewritten as

$$\mathbf{C}(\mathbf{x}-\mathbf{y})+\lambda(\mathcal{D}-\mathcal{W})\mathbf{x}=0. \tag{12}$$

The solution in (12) can be obtained with the diffusion scheme by running the following solution procedure,

$$\partial_t\mathbf{x}=g(\Delta\mathbf{x})(\mathbf{C}(\mathbf{y}-\mathbf{x})-\lambda(\mathcal{D}-\mathcal{W})\mathbf{x}),\quad \mathbf{x}^0=\mathbf{y}, \tag{13}$$

where $\mathbf{x}^0$ is the initial state, Δ is the discrete Laplace operator on the image grid, $g(\Delta\mathbf{x})$ is the diffusion coefficient as a function of $\Delta\mathbf{x}$ (Perona and Malik, 1990). We set $g(\Delta\mathbf{x})$ as $(\mathbf{C}+\lambda\mathcal{D})^{-1}$ to give,

$$\partial_t\mathbf{x}=\frac{\mathbf{C}(\mathbf{y}-\mathbf{x})-\lambda(\mathcal{D}-\mathcal{W})\mathbf{x}}{\mathbf{C}+\lambda\mathcal{D}}, \tag{14}$$

leading to a geometry-driven diffusion that diffuses more at pixels with sharp discontinuity where elements in $\mathcal{D}$ are small (Strong and Chan, 1996). Moreover, the diffusion diffuses less at high-confidence pixels where elements in $\mathbf{C}$ are large. By replacing $\partial_t\mathbf{x}$ with $\mathbf{x}^{t+1}-\mathbf{x}^t$ as done in Cheng et al. (2018); Liu et al. (2017), (14) becomes,

$$\mathbf{x}^{t+1}-\mathbf{x}^t=\frac{\mathbf{C}(\mathbf{y}-\mathbf{x}^t)-\lambda(\mathcal{D}-\mathcal{W})\mathbf{x}^t}{\mathbf{C}+\lambda\mathcal{D}}, \tag{15}$$

$$\mathbf{x}^{t+1}=\frac{\mathbf{Cy}+\lambda\mathcal{W}\mathbf{x}^t}{\mathbf{C}+\lambda\mathcal{D}}. \tag{16}$$

From (16), we can see at each time step t, $\mathbf{x}^{t+1}$ is obtained by the convolutional transform of $\mathbf{x}^t$ with kernel specified by $\mathcal{W}$, followed by fusion with the initial state $\mathbf{Cy}$. In the convolution procedure, the input values get propagated to the neighboring pixels defined by adjacency matrix $\mathcal{W}$. With

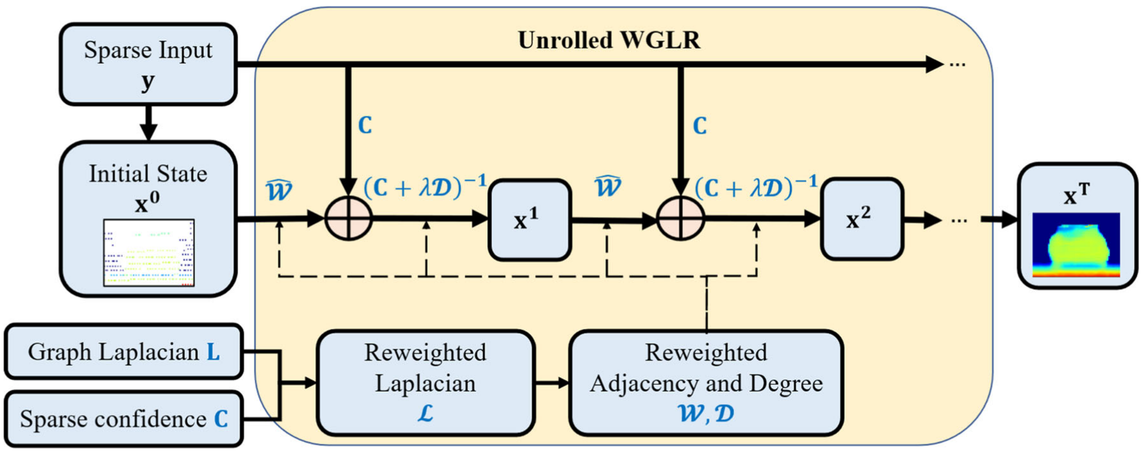

Sparse Input
$\mathbf{y}$
Initial State
$\mathbf{x}^0$
Unrolled WGLR
$\mathbf{C}$
$\widehat{\mathcal{W}}$
$(\mathbf{C}+\lambda\mathcal{D})^{-1}$
$\mathbf{x}^1$
$\mathbf{x}^2$
$\mathbf{x}^T$
Graph Laplacian $\mathbf{L}$
Sparse confidence $\mathbf{C}$
Reweighted Laplacian $\mathcal{L}$
Reweighted Adjacency and Degree $\mathcal{W}, \mathcal{D}$

Fig. 4 Illustration of Unrolled WGLR: one iteration of WGLR executes a convolutional transform and then linear fusion and thus can be recast into a network layer; by stacking the layers together a deep network is formed. The trainable parameters in the network are colored in blue (Color figure online)

Algorithm 1 Unrolled Weighted Graph Laplacian Regularization

```
Require: Sparse input depth with noise corruption $\mathbf{y}$ and corresponding
    confidence $\mathbf{c}$, graph Laplacian matrix $\mathbf{L}$, weight for GLR prior $\lambda$,
    weight for boundary correction $\mu$, iteration number $T$
Ensure: Dense output depth $\mathbf{x}$
1: Reweight $\mathbf{L}$ with $\mathbf{c}$ and $\mu$ to obtain weighted graph Laplacian $\mathcal{L} =$
   $\mathbf{L} + \mu(\mathbf{LC} + \mathbf{CL})$
2: Obtain corresponding $\mathcal{W}$ and $\mathcal{D}$ from $\mathcal{L}$
3: Obtain the normalized affinity $\hat{\mathcal{W}}$ with $\mathcal{D}^{-1}\mathcal{W}/\lambda$
4: Initialize $\mathbf{x}^0 = \mathbf{y}$
5: for $t = 0 : T - 1$ do
6:     Transform $\mathbf{x}^t$ with convolutional kernel $\hat{\mathcal{W}}$
7:     Fuse with $\mathbf{y}$ via $\mathbf{c}$ as specified in (16) to update $\mathbf{x}^{(t+1)}$
8: end for
```

a larger λ, the output will be better aligned with the structure specified by $\mathcal{W}$. In the fusion procedure, given the confidence $\mathbf{C}$ in the range [0, 1], if the input sample y_i is highly reliable, the output x_i^{t+1} will preserve the corresponding sample.

To ensure stability of the diffusion, we normalized $\mathcal{W}$ with $\hat{\mathcal{W}} = \mathcal{D}^{-1}\mathcal{W}/\lambda$. By recurrently repeating the above procedure, we obtain the solution to the optimization in (4) enforcing input fidelity conditioned on its confidence. We call this algorithm *Unrolled WGLR*, which is summarized in Algorithm 1 and illustrated in Fig. 4.

Relation to SPN-based Approaches With similar unrolling algorithm, the GLR solution can be implemented with the following diffusion scheme,

$$\mathbf{x}^{t+1} = (\mathbf{C}\mathbf{y} + \lambda\mathbf{W}\mathbf{x}^t)/(\mathbf{C} + \lambda\mathbf{D}). \tag{17}$$

If we use $\mathbf{C} = \mathbf{I}$, (17) boils down to CSPN (Cheng et al., 2019). Therefore, WGLR solution improves over SPN-based approaches in terms of input fidelity via (1) reweighting graph Laplacian matrix $\mathcal{L}$ which corrects boundary term of $\mathbf{L}$ so that sparse input properly diffuses to neighboring pixels; (2) considering input reliability $\mathbf{C}$ which prohibits propagation of noisy input values. This is validated experimentally in Sect. 5.

4 Network Design

In this section, the proposed unrolled WGLR algorithm is integrated into deep learning framework to develop HUGNet, a high-performance yet interpretable network for depth completion. First, we describe the motivation for executing unrolled WGLR algorithm as a deep neural network. Then we introduce the network design of hierarchical unrolled WGLR that applies depth diffusion at multi-scales. Following that, we discuss supporting modules of spatial-variant graph learning and confidence estimation to provide structural guidance and input conditioning for hierarchical unrolled WGLR.

4.1 Deep Unrolled WGLR

The unrolled WGLR algorithm in Sect. 3.3 is built upon prior knowledge of data smoothness with respect to the graph structure. Due to its interpretability as an anisotropic diffusion process, the algorithm explicitly enforces input constraints and enhances robustness to input corruption. However, the challenges are how to learn accurate structural features for graph construction, and how to estimate input confidence for prohibiting noisy input from spreading to neighbors.

To address the above challenges, we execute each iteration of unrolled WGLR as a convolutional transform and then linear fusion, which is thus recast into a network layer; by stacking the layers together forms a deep neural network

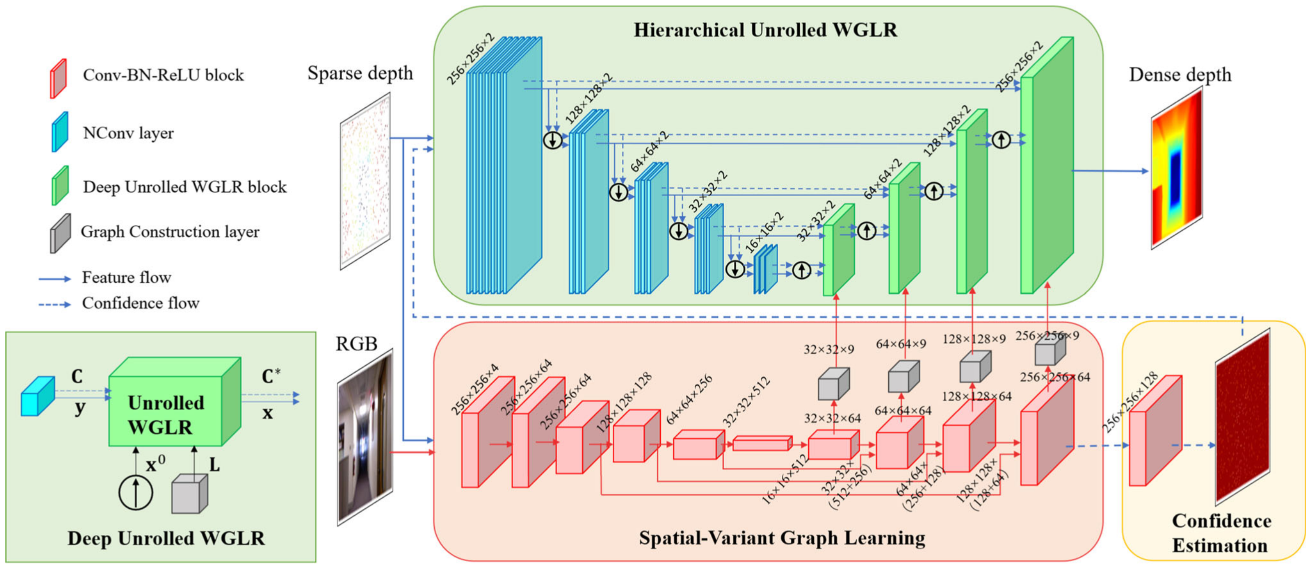


Fig. 5 Overview of the proposed HUGNet consists of three modules: (**1**) hierarchical unrolled WGLR module for depth completion; (**2**) spatial-variant graph learning module for graph Laplacian construction; (**3**) confidence estimation module for input confidence generation. An input with size 256 $\times$ 256 is used for demonstration (Color figure online)

where the trainable parameters in the network are colored in blue as shown in Fig. 4. Specifically, the graph Laplacian $\mathbf{L}$ and input confidence $\mathbf{C}$ corresponding to $\mathbf{y}$ are learnt from neural networks. The deep network of unrolled WGLR is named *Deep Unrolled WGLR*, which is then integrated as a trainable module in the proposed network with a simplified illustration in the green box shown in the lower-left of Fig. 5.

4.2 Network Architecture

The overview of the network is illustrated in Fig. 5, which is composed of three modules: (1) hierarchical unrolled WGLR module for depth completion, where the encoder pre-filters the sparse input to generate multi-scale dense depth and confidence, and the decoder hierarchically diffuses the depth via deep unrolled WGLR; (2) spatial-variant graph learning module for graph Laplacian construction; (3) confidence estimation module for semantic confidence generation. Next, we will discuss the design motivation and detailed implementation for each module.

4.2.1 Hierarchical Unrolled WGLR

This is the fundamental module in the network which takes the sparse depth and confidence as input and output the dense depth. The unrolled WGLR is implemented in a hierarchical manner, where the encoder pre-filters the sparse input to generate multi-scale depth and confidence, while the decoder hierarchically diffuses the depth via deep unrolled WGLR. The last unrolled WGLR block outputs the final dense depth. The structure and feature size are shown at the top of Fig. 5.

Encoder for Input Pre-filtering Although the sparse depth and confidence are ready to be used in the unrolled WGLR, using the sparse depth as the initial state requires large number of iterations to reach the steady state in the diffusion process. Therefore, to accelerate the diffusion convergence, we pre-filter the sparse depth to generate dense depth with noise reduction and value-preserved densification. We adopt NConv layers (Eldesokey et al., 2019) to jointly filter depth and confidence, which explicitly rectifies the disturbed measurements with confidence-equipped convolution layers, and removes the bias in NConv to satisfy the requirement for input preservation. As shown in Fig. 6b, the encoder generates densified depth feature while preserving the input values, which is consistent with the design purpose. Note that the encoder feature is not required to be fully densified because the depth diffusion mainly relies on the decoder.

Additionally, to provide input for hierarchical WGLR, a multi-scale architecture based on U-Net (Ronneberger et al., 2015) is adopted, and the depth feature $\mathbf{y}$ and confidence $\mathbf{C}$ with corresponding scale are used as input for unrolled WGLR blocks at the decoder.

Decoder for Hierarchical Diffusion When using unrolled WGLR at full resolution, the convergence is slow. Moreover, it is observed that the result is less smooth especially in large input-invalid area. This is because the learnt graph with localized connectivity is less effective in ensuring global smoothness.

In light of this, single-scale unrolled WGLR is insufficient, which motivates us to adopt unrolled WGLR at four different scales in a hierarchical manner as shown Fig. 5. The depth is successively refined, where the WGLR output from

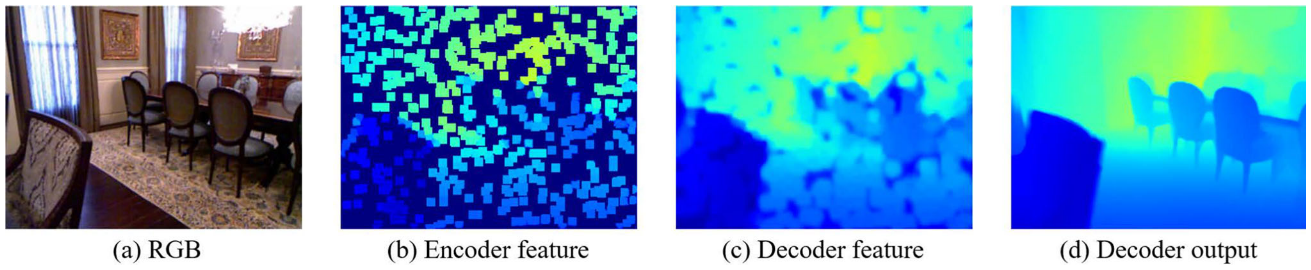


Fig. 6 Visualization of encoder and decoder features in hierarchical unrolled WGLR module (Color figure online)

previous scale shown in Fig. 6c is used as $\mathbf{x}^0$ in the next scale. Then with the learnt graph structure $\mathbf{L}$, the decoder refines $\mathbf{x}^0$ conditioned on the encoder feature input $\mathbf{y}$, and generates output shown in Fig. 6d. In this way, global smoothness would be corrected at small resolution, while detail enhancement would take place at the full resolution.

4.2.2 Spatial-Variant Graph Learning

This module extracts structural features from RGB and sparse depth images to estimate the multi-scale graph Laplacian $\mathbf{L}$. *We emphasize that we do not limit the network architecture for this module, and more accurate fusion network for semantic feature learning can be adopted.* In this work, we adopt the encoder-decoder structure built upon ResNet-34 backbone (He et al., 2016). The outputs of the decoder features are fed into the graph construction layer, which outputs the spatial-variant kernels to compute $\mathbf{L}$ connecting pixels in the local 3×3 region. $\mathbf{L}$ at four scales are then used to enable hierarchical unrolled WGLR.

4.2.3 Confidence Estimation

Along with the graph learning, the RGB-D fusion network outputs the semantic confidence for sparse input depth. The confidence is then fed into the NConv layer and pre-filtered along with the sparse depth to provide $\mathbf{C}$ in the unrolled WGLR as the corresponding confidence for input depth to prohibit noise propogation.

4.3 Loss Function

For accurate prediction of dense depth map, we train our network with the following loss function below supervised by the ground truth depth as follows:

$$L(\mathbf{x}^{\text{gt}}, \mathbf{x}^{\text{pred}}) = \frac{1}{\mid \mathcal{V} \mid} \sum_{v \in \mathcal{V}} \mid \mathbf{x}_v^{\text{pred}} - \mathbf{x}_v^{\text{gt}} \mid^p \tag{18}$$

where $\mathbf{x}^{\text{gt}}$ is the ground-truth depth, $\mathbf{x}^{\text{pred}}$ is the final predicted dense depth. $\mathbf{x}_v$ and $\mid \mathcal{V} \mid$ denote the depth values at pixel index v and the number of valid pixels, respectively. p is set to 1 for l_1 loss and 2 for l_2 loss. Note that we do not have any supervision on the confidence because there is no ground truth; therefore, it is indirectly trained based on L.

5 Experiments

In this section, we experimentally validate the effectiveness of the proposed HUGNet for depth completion via comparison with competing schemes on indoor NYUv2 dataset (Silberman et al., 2012) and outdoor KITTI depth completion (KITTI-DC) dataset (Uhrig et al., 2017), with extra focus on illustration of input preservation and noisy robustness. We additionally verify the effectiveness of each module in HUGNet via ablation study. Furthermore, TetrasRGBD dataset (Sun et al., 2023) is used to demonstrate performance on noisy RGB-D data captured by mobile devices, showing the potential of proposed scheme to support commodity RGB-D cameras with low imaging quality.

5.1 Experiment Setting

Dataset We adopt NYUv2 dataset (Silberman et al., 2012) and KITTI-DC dataset (Uhrig et al., 2017) for evaluation. NYUv2 dataset (Silberman et al., 2012) contains images collected from 464 different indoor scenes with Microsoft Kinect. We use the official split of data, where 249 scenes are used for training and the remaining 215 for testing, with sample number in the sparse depth input set to 500. For training and testing, the input images are resized to 320×240 and then center-cropped to 304×228. KITTI-DC dataset (Uhrig et al., 2017) is an outdoor dataset for autonomous driving, which contains 85k color images and corresponding dense annotated depth maps and sparse raw LiDAR scans for training, 6k for validation, and 1k for testing. The sparse LiDAR scans are captured by the real LiDAR sensor with noise corruption (Geiger et al., 2012). We use the selected validation set for evaluation in our experiment. For training, we crop the first 100 rows of color and depth images (which have no corresponding ground-truth depth) and then randomly crop color and depth images to 1216×240.

Training Details We use Adam optimizer with initial learning rate set to $1e^{-3}$ and decayed at epoch [10, 20, 30, 40] with decay rate 0.5. We initialize the weights of the encoder in graph learning module with model pretrained on ImageNet (Deng et al., 2009). The model is trained for 50 epochs using l_1 and l_2 losses to balance RMSE and MAE. We implement with PyTorch (Paszke et al., 2017) on 2 NVIDIA GeForce RTX 3090 GPUs, and batch size for NYUv2 and KITTI-DC is 16 and 6, respectively. For parameter setting, we set $\lambda = 1$, $\mu = S/N$ which is the sampling rate with S as the input sample number. Iteration number is set empirically with $T = 5$ for WGLR at 1/2, 1/4, 1/8 scales and $T = 10$ for the last WGLR block at full resolution.

Evaluation Metrics For quantitative evaluation, we adopt the commonly used metrics, including RMSE, MAE, iRMSE, iMAE, REL, δ_τ as used in Park et al. (2020):

- RMSE: root mean squared error

$$\sqrt{\frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}}(d_v^{\text{gt}} - d_v^{\text{pred}})^2};$$

- MAE: mean absolute error

$$\frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}}|d_v^{\text{gt}} - d_v^{\text{pred}}|;$$

- iRMSE: RMSE of inverse depth

$$\sqrt{\frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}}\left(\frac{1}{d_v^{\text{gt}}} - \frac{1}{d_v^{\text{pred}}}\right)^2};$$

- iMAE: MAE of inverse depth

$$\frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}}\left|\frac{1}{d_v^{\text{gt}}} - \frac{1}{d_v^{\text{pred}}}\right|;$$

- REL: mean absolute relative error

$$\frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}}\left|\frac{d_v^{\text{gt}} - d_v^{\text{pred}}}{d_v^{\text{gt}}}\right|;$$

- δ_τ: percentage of pixels satisfying

$$\max\left(\frac{d_v^{\text{gt}}}{d_v^{\text{pred}}}, \frac{d_v^{\text{pred}}}{d_v^{\text{gt}}}\right) < \tau, \tau \in \{1.25, 1.25^2, 1.25^3\}.$$

In addition, we focus on the accuracy at sample locations and edge areas using the following metrics:

- SMAE: spot mean absolute error measures MAE for pixels within 5×5 neighborhood $\mathcal{N}_\mathcal{S}$ of the sample locations $\mathcal{S}$, which is computed as

$$\frac{1}{|\mathcal{N}_\mathcal{S}|}\sum_{v\in\mathcal{N}_\mathcal{S}}|d_v^{\text{gt}} - d_v^{\text{pred}}|.$$

  Lower SMAE value indicates better confidence-aware input preservation and continuity around sample locations.
- EWMAE: edge weighted mean absolute error (Sun et al., 2023) measures the weighted average of absolute error, assigning larger weights to regions with larger depth discontinuity, which is computed as

$$\left(\sum_{v\in\mathcal{V}} g_v\,|d_v^{\text{gt}} - d_v^{\text{pred}}|\right)\Big/\left(\sum_{v\in\mathcal{V}} g_v\right),$$

  where g_v denote the weight coefficient at pixel v computed following (López-Randulfe et al., 2017). Lower EWMAE value indicates more accurate structural details preservation.

5.2 Comparison with State-of-the-Art Schemes

5.2.1 NYUv2 Dataset

The proposed HUGNet is compared with competing schemes including Sparse-to-dense (S2D) (Ma and Karaman, 2018), PNCNN (Eldesokey et al., 2020), NConv with RGB guidance using EncDec-Net (Eldesokey et al., 2019), CSPN (Cheng et al., 2019), NLSPN (Park et al., 2020) and PENet (Hu et al., 2021). S2D and PENet are pure data-driven methods, PNCNN and NConv are representative methods for networks with prior knowledge to incorporate input constraints, CSPN and NLSPN are GLR-based methods with input replacement.

In real-world scenarios, the captured depth is highly likely to be corrupted by noise. However, NYUv2 samples the sparse depth from the ground-truth depth without noise corruption. Therefore, in addition to the original setting (Park et al., 2020), *we simulate corrupted sparse input depth to evaluate performance of input fidelity under noise attack.* Next, we will describe the result comparison for the above two cases.

Original NYUv2 To test existing schemes, we use the pretrained models provided by the authors, except for S2D and PENet that did not provide NYUv2 pretrained models, so we use the provided model structures and retrain the models. Following (Park et al., 2020), we use the metrics of RMSE, REL and δ_τ. The top part entitled "Original Setting" in Table 1 shows the quantitative comparison among different schemes on original NYUv2 dataset and the proposed HUGNet outperforms competing schemes.

Table 1 Quantitative evaluation on NYUv2 dataset compared with existing schemes

Noise	Metric	S2D (Ma and Karaman, 2018)	PNCNN (Eldesokey et al., 2020)	NConv (Eldesokey et al., 2019)	CSPN (Cheng et al., 2019)	NLSPN (Park et al., 2020)	PENet (Hu et al., 2021)	HUGNet
Original setting	$\downarrow$ RMSE (m)	0.224	0.256	0.123	0.117	0.092	0.113	**0.091**
	$\downarrow$ REL (m)	0.043	0.049	0.017	0.016	**0.012**	0.022	**0.012**
	$\uparrow$ $\delta_{1.25}$	97.8	97.6	99.1	99.2	**99.6**	99.3	**99.6**
	$\uparrow$ $\delta_{1.25^2}$	99.5	99.4	99.8	99.9	99.9	99.9	99.9
	$\uparrow$ $\delta_{1.25^3}$	99.9	99.9	100.0	100.0	100.0	100.0	100.0
Outlier $p = 0.2$	$\downarrow$ RMSE (m)	0.245	0.217	0.190	0.180	0.136	0.149	**0.129**
	$\downarrow$ MAE (m)	0.813	0.116	0.109	0.107	0.055	0.080	**0.054**
	$\downarrow$ iRMSE (1/m)	0.176	0.038	0.034	0.032	0.131	0.037	**0.026**
	$\downarrow$ iMAE (1/m)	0.109	0.020	0.020	0.019	0.010	0.015	**0.009**
	$\downarrow$ SMAE (m)	0.891	0.101	0.099	0.096	0.041	0.071	**0.039**
	$\downarrow$ EWMAE (m)	0.942	0.442	0.399	0.381	0.313	0.354	**0.311**
AWGN $\sigma = 0.2$	$\downarrow$ RMSE (m)	0.209	0.206	0.175	0.169	0.140	0.148	**0.137**
	$\downarrow$ MAE (m)	0.138	0.119	0.101	0.107	0.077	0.084	**0.075**
	$\downarrow$ iRMSE (1/m)	0.044	0.038	0.034	0.036	0.072	0.037	**0.034**
	$\downarrow$ iMAE (1/m)	0.027	0.022	0.020	0.020	0.016	0.017	**0.014**
	$\downarrow$ SMAE (m)	0.122	0.105	0.093	0.096	0.071	0.075	**0.066**
	$\downarrow$ EWMAE (m)	0.414	0.421	0.378	0.367	0.312	0.344	**0.311**
Synthetic TetrasRGBD	$\downarrow$ RMSE (m)	0.190	0.206	0.176	0.166	0.116	0.124	**0.112**
	$\downarrow$ MAE (m)	0.439	0.112	0.100	0.098	0.053	0.062	**0.051**
	$\downarrow$ iRMSE (1/m)	0.055	0.038	0.036	0.031	0.038	0.020	**0.017**
	$\downarrow$ iMAE (1/m)	0.057	0.020	0.019	0.017	0.009	0.010	**0.008**
	$\downarrow$ SMAE (m)	0.265	0.093	0.083	0.085	0.040	0.061	**0.039**
	$\downarrow$ EWMAE (m)	0.497	0.429	0.383	0.366	0.302	0.343	**0.300**
Complexity	Runtime (ms)	6.18	10.31	6.18	45.34	32.98	63.89	18.55
	FLOPs (G)	43	18	77	261	220	137	221
	#Params (M)	5.54	0.67	0.48	218.12	25.84	132.12	26.67

Both original setting without noise, and noisy settings with outlier, additive white Gaussian noise (AWGN) and synthetic TetrasRGBD are tested. The best results are marked in bold

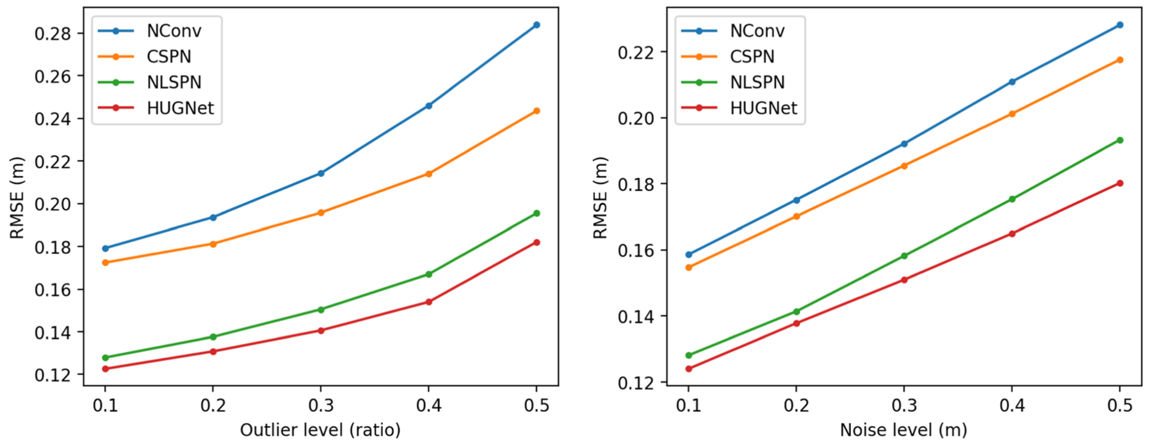


Fig. 7 Quantitative evaluation on NYUv2 dataset with different input noise settings, left for outlier level from 0.1 to 0.5; right for noise level from 0.1 to 0.5

Note that the advantage of HUGNet is less obvious in this setting because the sparse input contains no noise thus not realistic, making the task relatively easy and less meaningful. This motivates us to evaluate the noisy setting closer to real sensor data, which is more meaningful to reveal the performance in real practice.

NYUv2 with Synthetic Noise According to the statistical analysis of Time-of-Flight (ToF) depth sensor noise Mufti and Mahony (2011), the noise distribution in ToF depth approximates Gaussian distribution where the variance depends on the Signal-to-Noise ratio (SNR), except for low-SNR regions where the depth values scatter across the unambiguous range. Therefore, two types of noise are considered, (1) outliers with value randomly sampled from the uniform distribution from 0 to 10 ms and outlier ratio over total number of pixels set to p; (2) additive white Gaussian noise (AWGN) with standard deviation σ. To adopt the models to noise, we retrain all the competing schemes as well as HUGNet with noise data augmentation to enhance noise robustness. That is, during training, we apply noise corruption described above to the input sparse depth, where p uniformly distributed from 0.0 to 0.3 and σ uniformly distributed from 0.0 to 0.3 m.

In Table 1, we choose noise setting of outlier ratio $p = 0.2$ and AWGN $\sigma = 0.2$ as a median level of corruption for testing. Furthermore, we simulate noise based on the statistical analysis of TetrasRGBD (Sun et al., 2023) to generate noise closer to real sensor noise. Specifically, we select pixels at certain distance d_s from 1 to 7 ms and compute the error variance and outlier ratio (error larger than 3% depth value). The statics show that the noise variance approximately increases linearly with distance, while the noise ratio does not show a correlation with distance. Accordingly, we include three types of noise: (1) AWGN with distance-aware variance $\sigma_i = 0.0079 + 0.0118 \times d_i$ at i-th pixel with depth d_i, and the coefficients are regressed from the noise statistics; (2) outlier ratio $p = 0.04$ using the mean value of outlier ratios at different distances; (3) edge noise following the supplementary of Barron and Malik (2013), where we replace each pixel in the disparity map with the bilinearly interpolated value of a location near that pixel, and the shift is drawn from a normal distribution with $\sigma = 0.5$. The noise setting is named Synthetic TetrasRGBD used for testing.

Since REL and δ_τ cannot distinguish performance, we instead use MAE, iRMSE, iMAE, SMAE and EWMAE, where we can see HUGNet outperforms other schemes in all metrics. For example in the outlier setting, HUGNet reduces iRMSE by at least **15.6%** than competing schemes. Also, HUGNet has the lowest SMAE and EWMAE, due to the utilization of the reweighted graph Laplacian matrix to enforce input preservation, and the use of confidence map to prohibit propagation of noisy input and promote structural detail fidelity.

Additionally, we have accordingly included the complexity comparison in Table 1, showing the average running time, FLOPs, and the parameter size. The runtime is tested on one GeForce RTX 1080 Ti GPU. As shown in Table 1, the most competitive method, *i.e.*, NLSPN, consumes a higher runtime, while HUGNet achieves higher accuracy at moderate complexity with 43% runtime reduction.

To evaluate the generalization ability, we test with different levels of noise corruption using the same model. Figure 7 shows the RMSE results for HUGNet and competing schemes where $p, \sigma \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$. NConv, CSPN, NLSPN are used for comparison as they show competitive performance in Table 1. Our HUGNet outperforms existing schemes with lower error at all noise levels, demonstrating higher robustness to noise.

The results show that CSPN and NLSPN degrade when noise attack is involved. This is because the adjacency matrices are not corrected in SPN schemes which makes it hard to adhere to the input, especially when the input is not

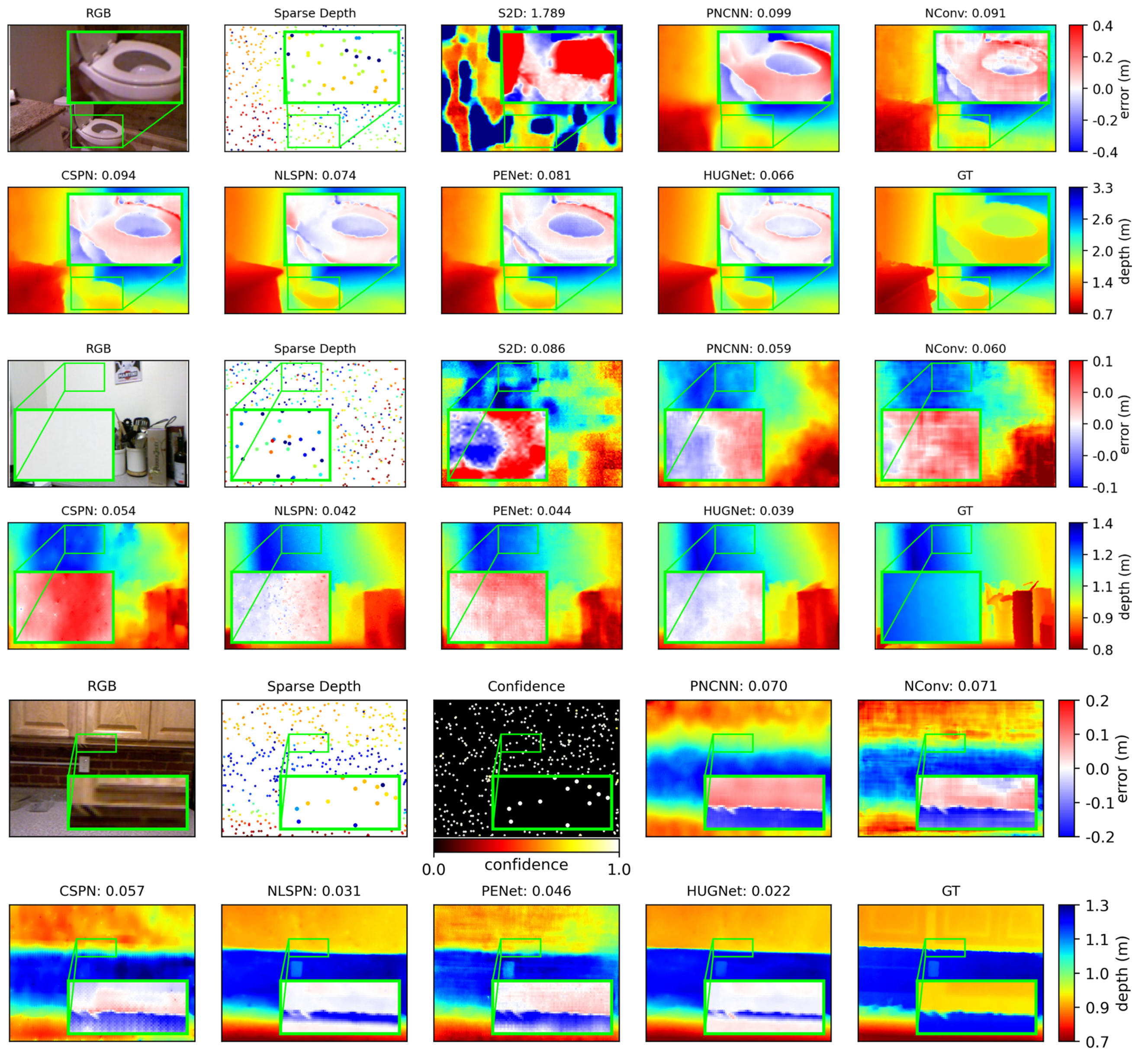


Fig. 8 Depth completion with different algorithms, tested on NYUv2 sample data with outlier $p = 0.2$ (upper part), AWGN $\sigma = 0.2$ (mid part), and synthetic TetrasRGBD noise (lower part). RMSE is shown on top of each result image, and the error map of the selected patch is enlarged and visualized in the green box. The proposed HUGNet outperforms existing schemes with input consistency and noise robustness (Color figure online)

smooth; moreover, they do not consider input confidence during propagation, thus invulnerable to noise corruption. Although NLSPN utilizes the confidence in the propagation, the confidence map is to quantify the confidence of predicted initial dense depth, which does not necessarily reflect the confidence of input sparse depth. For NConv, even if input constraint is incorporated in the network, the accuracy is less satisfying since the model-based network design limits its expressiveness in utilizing structural features from RGBD fusion. In contrast, HUGNet shows more accurate input fidelity as well as noise robustness, without sacrificing of detail preservation.

This is further validated in visual comparison in Fig. 8, where we show our depth completion results with outlier $p = 0.2$, noise $\sigma = 0.2$ and synthetic TetrasRGBD. Pure data-driven S2D and PENet cannot preserve input, and model-based PNCNN and NConv show blurry edges due to limited model expressiveness. This is consistent with the quantitative evaluation using SMAE and EWMAE shown in Table 1, where S2D, PNCNN, NConv and PENet exhibit

Table 2 Comparison of quantitative evaluation on KITTI-DC validation dataset among existing schemes

Method	RMSE (mm)↓	MAE (mm)↓	iRMSE (1/km)↓	iMAE (1/km)↓
PNCNN	1255.5	278.3	4.5	1.0
CSPN	1047.8	239.1	3.2	1.1
NConv	908.8	209.6	2.5	0.9
GAENet	813.8	245.1	2.6	1.2
DNormal	811.1	236.7	2.5	1.1
FusionNet	802.5	214.0	2.2	0.9
ACMNet	786.9	216.2	2.3	1.0
NLSPN	771.8	197.3	2.0	0.8
HUGNet	756.8	202.2	2.0	0.8

large SMAE and EWMAE values, indicating poor input preservation and inaccurate edge details. CSPN, NLSPN shows provide sharper depth details but suffer from discontinuity at sample locations due to the lack of boundary correction in the adjacency matrices, *e.g.*, on the flat wall in Fig. 8, CSPN and NLSPN show uneven results, where the error is large except for input locations. On the contrary, for our proposed HUGNet, the input is properly diffused to neighboring pixels enforcing input consistency and global smoothness; moreover, the WGLR prior does not affect the model expressiveness due to the design of deep unrolled WGLR, thus the structural details are preserved.

Discussion on Confidence Map We notice that the confidence map is more effective in defending outlier than AWGN. In the lower part of Fig. 8, the confidence has much smaller values at outlier pixels but does not vary much at pixels with moderate AWGN noise. This is because confidence not only depends on the reliability of input, but also the accuracy of $\mathbf{L}$ in (4). If $\mathbf{L}$ provides accurate structural features of $\mathbf{x}$, then the optimization relies less on confidence for noise resistance. Therefore in the case of moderate AWGN where noise can be removed via the diffusion based on $\mathbf{L}$, $\mathbf{c}$ does not show strong correlation with input noise in Fig. 8.

5.2.2 KITTI-DC Dataset

For evaluation on KITTI-DC, we additionally include FusionNet (Van Gansbeke et al., 2019) and ACMNet (Zhao et al., 2021) for pure data-drive methods, DNormal (Xu et al., 2019) and GAENet (Chen et al., 2022) for methods utilizing extra constraints, and DySPN (Lin et al., 2022), RigNet (Yan et al., 2022), MFFNet (Liu et al., 2023), CompletionFormer (CFormer) (Zhang et al., 2023) and BEV@DC (Zhou et al., 2023) which achieve state-of-the-art performance.

Tables 2 and 3 show the quantitative evaluation of our HUGNet on KITTI-DC validation and online test dataset with comparison with above mentioned competing schemes. The values are those originally reported in their respective papers. We have additionally included the complexity comparison in Table 3, including average running time (s) and FLOPs (G) with 1216 × 352 input size, and parameter size (M) of the model. Two types of runtime are provided. One is "RT Online" quoted from the KITTI benchmark website. For a fair comparison, we include "RT@1080Ti" which is

Table 3 Comparison of quantitative evaluation on KITTI-DC test dataset among existing schemes

Method	RMSE (mm)↓	MAE (mm)↓	iRMSE (1/km)↓	iMAE (1/km)↓	RT Online (s)↓	RT@1080Ti (s)↓	FLOPs (G)↓	Params (M)↓
CSPN (Cheng et al., 2019)	1019.64	279.46	2.93	1.15	1.00	–	–	–
NLSPN (Park et al., 2020)	741.68	199.59	1.99	0.84	0.22	0.159	1355	25.84
RigNet (Yan et al., 2022)	712.66	203.25	2.08	0.90	0.20	–	–	65.00
DySPN (Lin et al., 2022)	709.12	192.71	1.88	0.82	0.16	–	–	26.79
CFormer (Zhang et al., 2023)	708.87	203.45	2.01	0.88	0.12	0.214	1072	82.51
BEV@DC (Zhou et al., 2023)	**697.44**	**189.44**	**1.83**	**0.82**	0.13	–	–	–
PNCNN (Eldesokey et al., 2020)	960.05	251.77	3.37	1.05	0.02	0.023	109	0.67
NConv (Eldesokey et al., 2019)	829.98	233.26	2.60	1.03	0.02	0.014	356	0.36
DNormal (Xu et al., 2019)	777.05	235.17	2.42	1.13	0.10	–	–	28.99
GAENet (Chen et al., 2022)	773.90	231.29	2.29	1.08	0.05	–	–	4.19
FusionNet (Van Gansbeke et al., 2019)	772.87	215.02	2.19	0.93	0.02	–	–	2.55
ACMNet (Zhao et al., 2021)	744.91	206.09	2.08	0.90	0.08	–	544	4.90
PENet (Hu et al., 2021)	730.08	210.55	2.17	0.94	0.03	0.144	816	131.92
MFFNet (Liu et al., 2023)	**719.85**	208.11	2.21	0.94	0.05	–	–	–
HUGNet	724.64	**200.28**	**2.02**	**0.88**	0.09	0.087	1359	26.67

The online methods are listed in the lower section while the non-online methods are listed in the upper section

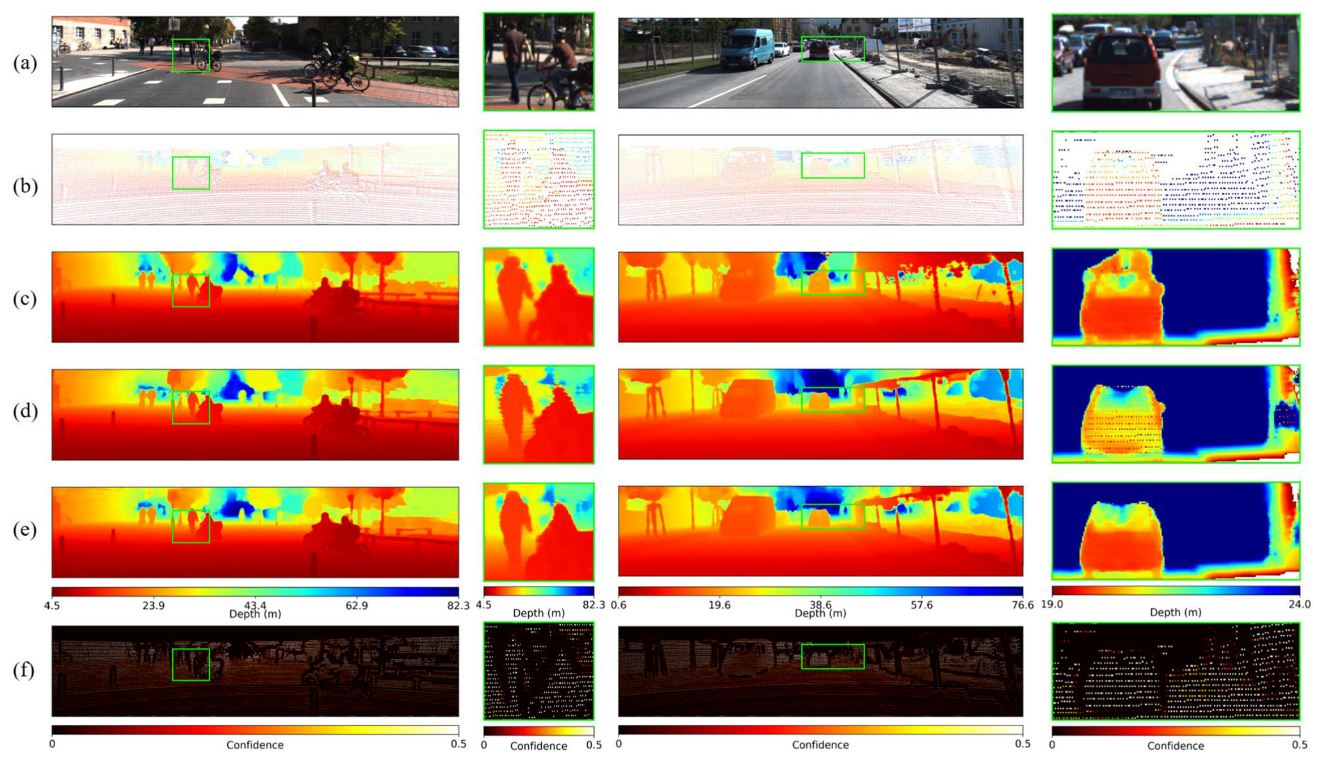


Fig. 9 Depth completion comparison with competing schemes on KITTI-DC test set: **a** RGB input, **b** sparse depth input, results of **c** NConv, **d** NLSPN, **e** proposed HUGNet and **f** confidence estimation in HUGNet. On the left scene, HUGNet shows higher robustness to depth-RGB mis-alignment; on the right scene, HUGNet shows better input preservation (Color figure online)

tested on one GeForce RTX 1080 Ti GPU with source codes released by the authors.

Similar to the evaluation in MFFNet (Liu et al., 2023), we refer to methods faster than 10 Hz as online methods since the sampling frequency of most LiDARs is 10 Hz (Geiger et al., 2012). In Table 3, the methods are listed in order of RMSE, where HUGNet achieves SOTA performance among online methods. As compared to non-online works, HUGNet achieves competitive results to RigNet (Yan et al., 2022) with 55% reduced runtime. We acknowledge that HUGNet is lower in accuracy than SOTA CFormer (Zhang et al., 2023) and BEV@DC (Zhou et al., 2023), but with 59% speed-up than CFormer, HUGNet enables online processing which is vital for downstream tasks.

The proposed HUGNet shows competitive performance among existing schemes due to the confidence-aware input preservation, which is visually validated in Figs. 9 and 10. Similar to results on NYUv2, NConv shows blurry edges, while NLSPN provides sharper structural details but suffers from input inconsistency and noise sensitivity. On the contrary, with deep unrolled WGLR, HUGNet not only preserves input but also resists noisy input via confidence estimation shown in Figs. 9f and 10h where the low confidence values prohibit noisy input from diffusing, but also preserves structural details. In addition, in Fig. 10, we include the visualization of the error maps of the sparse depth maps so as to demonstrate the relation between the sparse input error in Fig. 10g and the learned confidence in Fig. 10h. As shown in Fig. 10, there is a strong correlation between input error and the confidence, *e.g.*, in the left column, the confidence in Fig. 10h zeros out the values at the locations of the green input samples in Fig. 10f, which are outliers due to RGB-D mis-alignment along the object boundary. This explains how the confidence map enhances the noise robustness of HUGNet.

5.3 Ablation Study

To better understand how HUGNet works, in the ablation study as follows, we investigate the effect of each module in the network, including the confidence estimation, the spatial-variant graph learning, graph Laplacian re-weighting, and the hierarchical architecture of unrolled WGLR. For quantitative evaluation, we test with KITTI-DC validation set and results are shown in Table 4. For all the variants of HUGNet, the baseline model is the single-scale deep unrolled GLR and the filter kernels are learnt from depth input without graph Laplacian reweighting. The components to be investigated are then added to the baseline.

Confidence Estimation via RGBD Fusion (FConf) Based on how confidence is estimated in the baseline, we

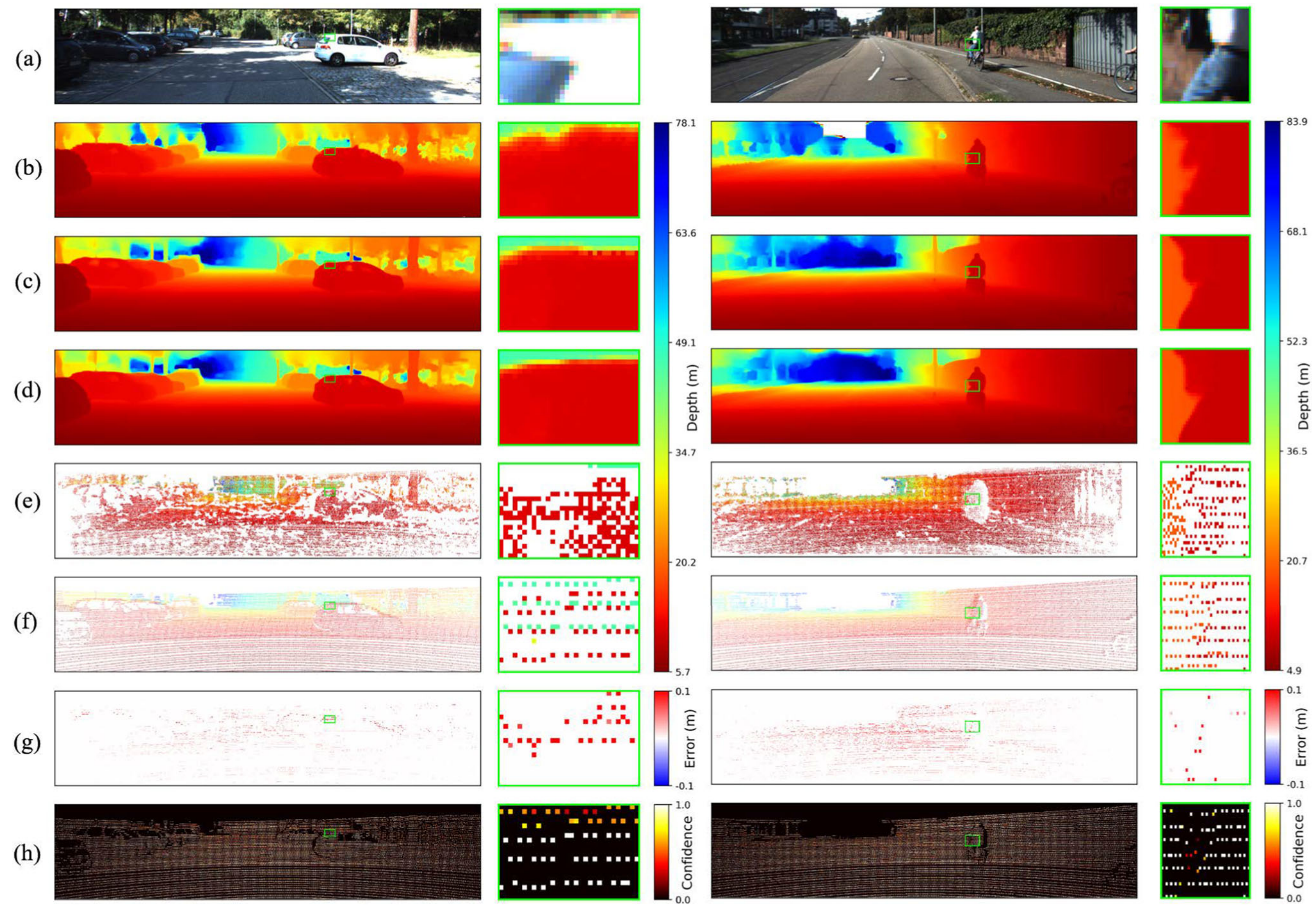


Fig. 10 Depth completion comparison with competing schemes on KITTI-DC validation set: **a** RGB input, results of **b** NConv, **c** NLSPN, **d** proposed HUGNet, **e** GT depth, sparse depth **f** input and **g** error, **h** confidence estimation in HUGNet which reflect input reliability. HUGNet shows higher robustness to noise corruption in sparse input (Color figure online)

Table 4 Quantitative evaluation on KITTI-DC validation set for ablation study

Method	Variants				Metrics			
	FConf	GLR	WGL	HUG	RMSE	MAE	iRMSE	iMAE
DConf					1180.3	268.5	3.4	1.1
FConf	✓				1053.4	259.2	3.0	1.0
FConf+GLR	✓	✓			847.5	220.5	3.2	1.0
FConf+WGLR	✓	✓	✓		796.4	210.9	2.2	0.9
FConf+GLR+HUG	✓	✓		✓	809.0	214.2	2.8	0.9
FConf+WGLR+HUG (HUGNet)	✓	✓	✓	✓	**756.8**	**202.2**	**2.0**	**0.8**

have two variants: one is estimating confidence from depth with an UNet prefixed to the baseline similar to Eldesokey et al. (2020), named DConf; one is estimating confidence from the RGBD fusion network, named as FConf. As shown in Table 4, FConf shows improvement in accuracy by learning a more accurate confidence with semantic RGBD fusion features than with only depth features as shown in Fig. 11, leading to a decrease of 126.9 mm in RMSE. Moreover, by sharing weight with the graph learning module, FConf saves the computational cost than DConf which involves an extra network for confidence estimation. In the following variants, the FConf module is adopted by default.

Graph Learning for GLR Solution (GLR) The spatial-variant graph learning produces adaptive kernels for constructing graph Laplacian used in GLR solution, and the variant is named as GLR. Without graph learning, the convo-

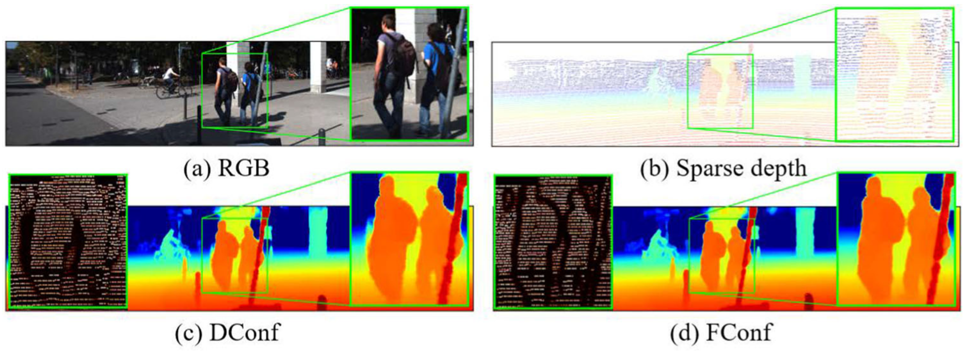


Fig. 11 Evaluation of confidence estimation module with sample KITTI-DC data, where the FConf learnt from RGBD fusion generates more accurate confidence than DConf learnt from depth input. Depth estimation in the green boxes are enlarged and shown on the right, and confidence maps are shown on the left (Color figure online)

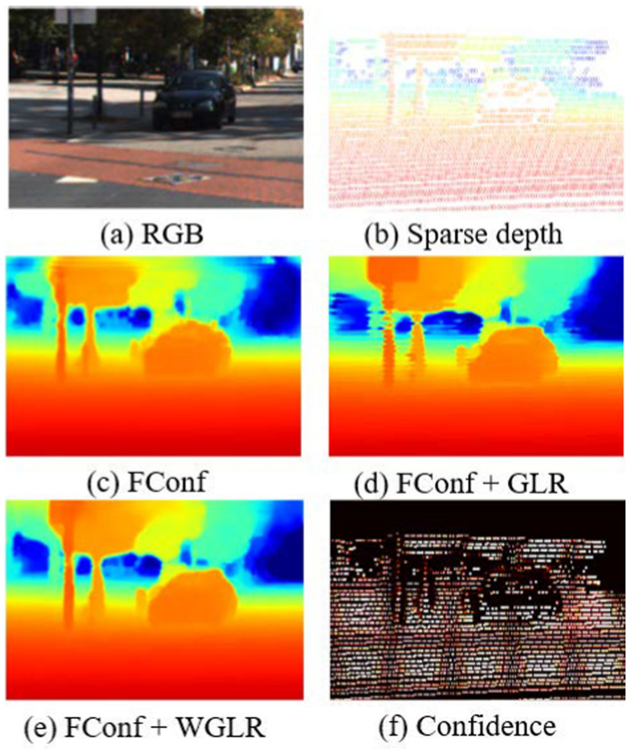


Fig. 12 Evaluation of graph learning and re-weighting graph Laplacian with sample KITTI-DC data. With graph learning, (**d**) GLR and (**e**) WGLR show sharper structural details than (c) FConf; with Laplacian re-weighting, (**e**) WGLR further enhances continuity at sample locations and robustness to input corruption than (**d**) GLR (Color figure online)

lutional kernels are static for all pixels and learnt from depth input only. With graph learning from RGBD fusion network, GLR solution improves the accuracy by a large margin shown in Table 4, *e.g.*, FConf+GLR outperforms FConf in RMSE by 205.9 mm, showing much better edge accuracy in Fig. 12.

Re-Weighting Graph Laplacian (WGL) While GLR does not enforce input preservation, WGLR is used to correct boundary term of GLR to mitigate the input inconsistency via re-weighting the graph Laplacian (WGL). As shown in Table 4, FConf+WGLR outperforms FConf+GLR by 51.1 mm, which is visually validated in Fig. 12.

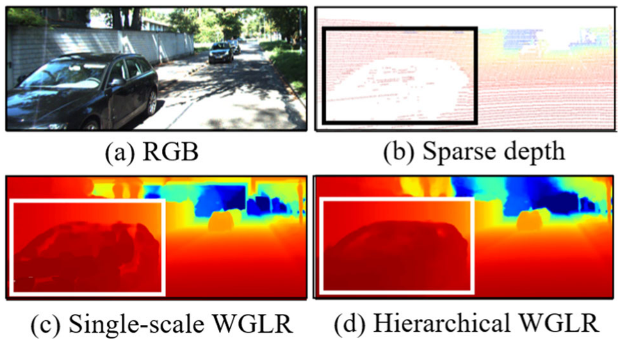


Fig. 13 Evaluation of hierarchical WGLR with sample KITTI-DC data, where hierarchical WGLR shows enhanced global smoothness than single-scale WGLR due to the use of multi-scale diffusion, especially in large input-invalid area highlighted in white rectangles (Color figure online)

Hierarchical Unrolled WGLR (HUG) The module of Hierarchical Unrolled WGLR adopts deep unrolled WGLR blocks in a hierarchical manner, which successively diffuses the depth at multi-scale. FConf+WGLR+HUG in Table 4 is HUGNet, and the variants using HUG outperforms those without HUG, *e.g.*, RMSE of HUGNet is 39.6 mm less than FConf+WGLR. Moreover, in Fig. 13, HUGNet entitled "hierarchical WGLR" shows better smoothness than FConf+WGLR entitled "single-scale WGLR", validating that hierarchical implementation promotes global smoothness while maintaining local details.

Iterations and Scales in HUG Module In addition, we provide an analysis of the impact of the scales and the corresponding iteration numbers in hierarchical unrolled WGLR, which we denote as T_s where s is the scale. The analysis of s and T_s is based on the evaluation on NYUv2 test dataset. Since there are substantial combinations of s and T_s settings, we first examine the impact of T_1 for full scale and $T_{1/2}$ for 1/2 scale excluding other scales, then investigate the impact of T_s for smaller scales. Specifically, we start with the two-scale unrolled WGLR and set T_1 to 6, 10, 14, 18 and $T_{1/2}$ to 0, 3, 5, 7, and investigate their impact on performance. RMSE values are plotted in Fig. 14 where we can see: 1) limited T_1 (*e.g.*, $T_1 = 6$) is insufficient to diffuse the depth, resulting in

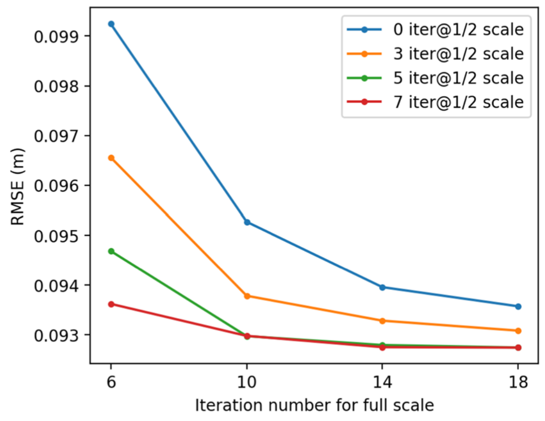


Fig. 14 Impact of the number of iterations and scales on the prediction RMSE on NYUv2 testset (Color figure online)

Table 5 Impact of scales on RMSE and complexity on NYUv2 testset

Scales	Iterations	RMSE (m)	FLOPs (G)
1	[10 0 0 0 0]	0.0953	5.487
1/2	[10 5 0 0 0]	0.0930	2.011
1/4	[10 5 5 0 0]	0.0915	0.981
1/8	[10 5 5 5 0]	0.0913	0.494
1/16	[10 5 5 5 5]	0.0913	0.165

The FLOPs number is for the diffusion block at each scale including graph learning module

poor RMSE; 2) larger T_1 leads to performance convergence, while larger $T_{1/2}$ (*e.g.*, $T_{1/2} = 5, 7$) fastens the convergence and more importantly enhances the accuracy.

To balance the efficiency and effectiveness, we set $T_1 = 10$, $T_{1/2} = 5$ respectively, based on which we further investigate the impact of multi-scale diffusion, and add scales of $1/4$, $1/8$ and $1/16$ successively. Since the required iteration number is related to the density of the input feature, we set fewer iterations for smaller scales, *i.e.*, $T_s \leq T_{1/2} = 5$ for $s \leq 1/2$. Moreover, since the FLOPs number for diffusion decreases linearly with the scales as shown in Table 5, so for simplicity, we set $T_s = 5$ for $s \leq 1/2$. The RMSE values are shown in Table 5 where involving more scales enhances the performance and the result converges at $1/8$ scale, so we include scales up to $1/8$ input size. In sum, we set $T_1 = 10$, and $T_s = 5$ for $s \in \{1/2, 1/4, 1/8\}$ for the experiments.

Network Structure for Graph Learning As mentioned in Sect. 4.2.2, the model structure for spatial-variant graph learning is not limited. Here we compare three typical network structures, i.e., Res34-UNet used in HUGNet based on ResNet, JCAT in CFormer (Zhang et al., 2023) based on transformer, and ENet in PENet (Hu et al., 2021) based on multi-branch fusion.

The HUGNet variants are trained with NYUv2 dataset following the training strategies in the original papers. For Res34-UNet and JCAT, we initialize the encoder with models pretrained on ImageNet (Deng et al., 2009), while ENet is trained from scratch since no pretrained model is provided. The evaluation results are shown in Table 6 where only Res34-UNet produces competitive results.

However, when we initialize Res34-UNet and JCAT with the pre-trained models of NLSPN and CFormer trained on NYUv2 dataset, JCAT shows improved accuracy while Res34-UNet is not affected. This is because JCAT and ENet are more difficult to train than Res34-UNet. This implies that Res34-UNet possesses training stability and competitive accuracy without the requirement of dedicated training strategy. On the other hand, with dedicated training, more sophisticated networks, such as those based on transformers, have the potential to achieve superior performance.

The complexity comparison is included in Table 6, showing the average running time tested on one GeForce RTX 1080 Ti GPU and the parameter size. While Res34-UNet and JCAT both achieve state-of-the-art performance, Res34-UNet reduced the runtime by 58.2%. Therefore, we choose Res34-UNet as the backbone for HUGNet to balance accuracy and efficiency.

5.4 Evaluation on Mobile Device

To demonstrate the generalization to real world practice, we adopt the TetrasRGBD dataset (Sun et al., 2023) for testing with real data captured by mobile devices, where the input depth suffers from large sensor noise. TetrasRGBD provides real data collected from a fixed camera equipped with calibrated RGB and ToF sensors. However, the corresponding ground-truth depth maps are not provided, thus we use the pretrained model using NYUv2 dataset with synthetic noise augmentation.

For quantitative evaluation of the results, we adopt two blind depth assessment metrics, Blind Depth Quality Metric (BDQM) (Farid et al., 2015) and depth confidence measure (DCM) (Cong et al., 2016). In Table 7, the results for HUGNet surpasses those of NConv and NLSPN. In addition, we show visual comparison in Fig. 15 which is consistent with the metric results. While NConv and NLSPN show uneven surfaces in the background of first row images, and inaccurate estimation in large input-invalid areas in second and third row images, the proposed HUGNet shows strong robustness to noise and enhanced global smoothness due to the use of hierarchical WGLR. Additionally, NLSPN shows discontinuity highlighted in red rectangle in the second row, HUGNet demonstrates better input preservation. This additionally validates the generalization ability of HUGNet to real sensor noise when trained with synthetic noise, showing the potential of HUGNet to support commodity RGB-D cameras with low imaging quality.

Table 6 Ablation study of model structure for spatial-variant graph learning module tested on NYUv2 dataset

Backbone & Pretrain	RMSE (m)	MAE (m)	Runtime (ms)	#Params (M)
Res34-UNet w/ ImageNet (HUGNet)	0.0913	0.0348	18.55	26.67
Res34-UNet w/ NLSPN	0.0911	0.0348		
JCAT w/ ImageNet	0.0933	0.0361	44.38	83.34
JCAT w/ CFormer	0.0904	0.0350		
ENet from scratch	0.1132	0.0513	52.22	132.95

Table 7 Blind assessment of depth completion results on TetrasRGBD dataset with real noise

Metric	NConv	NLSPN	HUGNet
↑ BDQM	76.0	122.7	**125.6**
↑ DCM	0.2657	0.2747	**0.2854**

5.5 Limitations and Future Works

We have investigated the input preservation with noise robustness, and the noise mainly refers to sensor noise and RGB-D misalignment in areas within sensing range. Nevertheless, there are other types of "corruption" in sparse input, *e.g.*, large input-invalid area that is of low reflectance or out of sensing range, then depth completion merely relies on monocular RGB features. Though HUGNet is able to handle large input-invalid area via hierarchical unrolled WGLR, the accuracy is not justified due to the lack to ground-truth depth. Therefore, for future study, the investigation will be devoted to a more comprehensive processing of input corruption via self-supervised learning or multi-view RGB input settings.

Moreover, the current graph learning module does not utilize non-local neighborhoods, while existing schemes such as NLSPN and DySPN utilize non-local neighbors to enhance performance. Therefore, efficiently utilizing non-local neighbors in the WGLR will be our future direction.

6 Conclusion

In this paper, we propose Hierarchical Deep Unrolled WGLR Network (HUGNet) for depth completion that enforces input preservation and enhances noise robustness. Based on continuous domain analysis, the weighted graph Laplacian regularization (WGLR) solution is derived to solve the depth completion optimization with input preservation. Then WGLR solution is unrolled into iterative filtering via its anisotropic diffusion interpretation, which is efficiently implemented via convolutional transforms and integrated

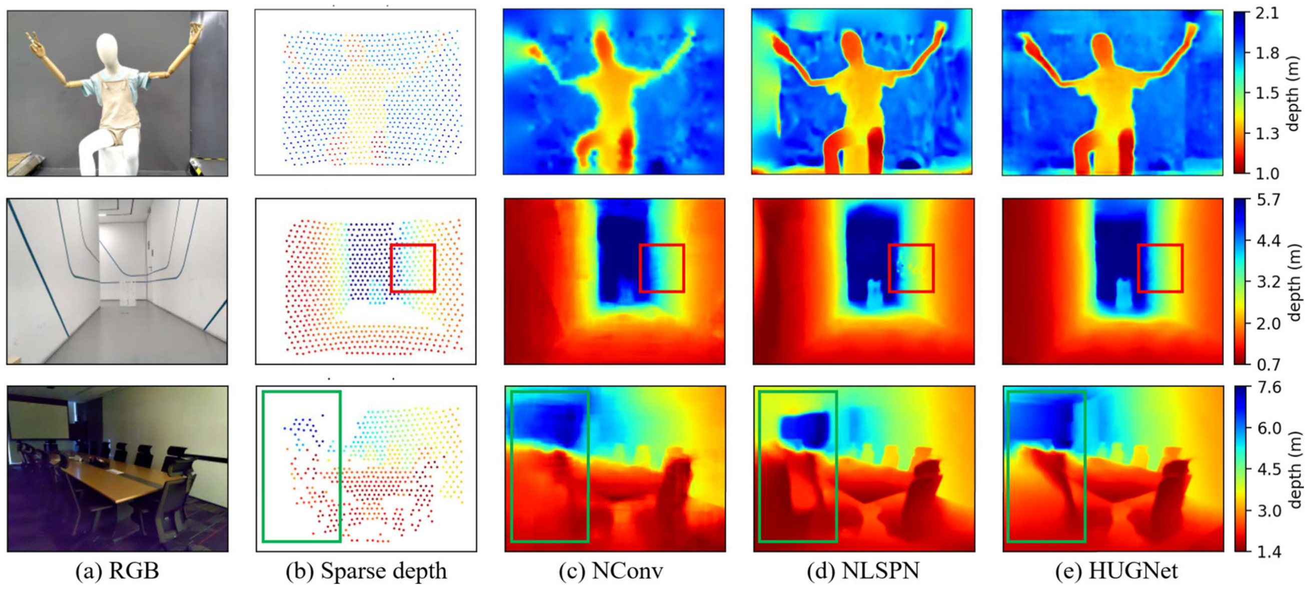


Fig. 15 Depth completion with different approaches on TetrasRGBD set, where HUGNet shows advantages in **1** global smoothness in the background of first row images, **2** input preservation highlighted in red rectangles of second row images, **3** accuracy in large input-invalid area highlighted in green rectangles of third row images (Color figure online)

into the deep learning framework as a trainable module. With hierarchical unrolled WGLR, HUGNet is designed which shows global smoothness and accurate structural detail, and strong generalization ability that robustifies the network against input corruption. Experimental results show that HUGNet improves input preservation and noise robustness over competing schemes in datasets with synthetic noise as well as real sensor noise.

Acknowledgements This work was supported in part by National Natural Science Foundation of China under Grant 62201389, and in part by Shanghai Sailing Program under Grant 22YF1451200.

References

Barron, J. T., & Malik, J. (2013). Intrinsic scene properties from a single rgb-d image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 17–24).

Barron, J. T., & Poole, B. (2016). The fast bilateral solver. In *European Conference on Computer Vision (ECCV)* (pp. 617–632). Springer

Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern Recognition and Machine Learning* (pp. 161–162). New York: Springer.

Chang, A., Dai, A., Funkhouser, T., Halber, M., Niebner, M., Savva, M., Song, S., Zeng, A., & Zhang, Y. (2017). Matterport3d: Learning from rgb-d data in indoor environments. In *International Conference on 3D Vision (3DV)* (pp. 667–676).

Chen, H., Yang, H., Zhang, Y., et al. (2022). Depth completion using geometry-aware embedding. In *International Conference on Robotics and Automation (ICRA)* (pp. 8680–8686). IEEE.

Cheng, X., Wang, P., & Yang, R. (2018). Depth estimation via affinity learned with convolutional spatial propagation network. In *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 103–119).

Cheng, X., Wang, P., & Yang, R. (2019). Learning depth with convolutional spatial propagation network. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 42*(10), 2361–2379.

Cheung, G., Magli, E., Tanaka, Y., & Ng, M. K. (2018). Graph spectral image processing. *Proceedings of the IEEE, 106*(5), 907–930.

Chodosh, N., Wang, C., & Lucey, S. (2018). Deep convolutional compressed sensing for lidar depth completion. In *Asian Conference on Computer Vision (ACCV)* (pp. 499–513). Springer.

Cong, R., Lei, J., Zhang, C., Huang, Q., Cao, X., & Hou, C. (2016). Saliency detection for stereoscopic images based on depth confidence analysis and multiple cues fusion. *IEEE Signal Processing Letters, 23*(6), 819–823.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 248–255). Ieee.

Du, R., Turner, E., Dzitsiuk, M., Prasso, L., Duarte, I., Dourgarian, J., Afonso, J., Pascoal, J., Gladstone, J., Cruces, N., et al. (2020). Depthlab: Real-time 3d interaction with depth maps for mobile augmented reality. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (pp. 829–843).

Eldesokey, A., Felsberg, M., Holmquist, K., & Persson, M. (2020). Uncertainty-aware cnns for depth completion: Uncertainty from beginning to end. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 12014–12023).

Eldesokey, A., Felsberg, M., & Khan, F. S. (2019). Confidence propagation through CNNS for guided sparse depth regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 42*(10), 2423–2436.

Farid, M. S., Lucenteforte, M., & Grangetto, M. (2015). Blind depth quality assessment using histogram shape analysis. In *2015 3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON)* (pp. 1–5). IEEE

Feng, Z., Jing, L., Yin, P., Tian, Y., & Li, B. (2022). Advancing self-supervised monocular depth learning with sparse lidar. In *Conference on Robot Learning* (pp. 685–694). PMLR.

Ferstl, D., Reinbacher, C., Ranftl, R., Rüther, M., & Bischof, H. (2013). Image guided depth upsampling using anisotropic total generalized variation. In *International Conference on Computer Vision (ICCV)* (pp. 993–1000).

Geiger, A., Lenz, P., & Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3354–3361). IEEE.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778).

Hu, W., Li, X., Cheung, G., & Au, O. (2013). Depth map denoising using graph-based transform and group sparsity. In *IEEE International Workshop on Multimedia Signal Processing* (pp. 001–006). IEEE.

Hu, M., Wang, S., Li, B., Ning, S., Fan, L., & Gong, X. (2021). Penet: Towards precise and efficient image guided depth completion. In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 13656–13662). IEEE.

Huang, Z., Fan, J., Cheng, S., Yi, S., Wang, X., & Li, H. (2019). Hms-net: Hierarchical multi-scale sparsity-invariant network for sparse depth completion. *IEEE Transactions on Image Processing, 29*, 3429–3441.

Li, Y., Yu, A. W., Meng, T., Caine, B., Ngiam, J., Peng, D., Shen, J., Lu, Y., Zhou, D., Le, Q.V., et al. (2022). Deepfusion: Lidar-camera deep fusion for multi-modal 3d object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 17182–17191).

Li, A., Yuan, Z., Ling, Y., Chi, W., Zhang, C., et al. (2020). A multi-scale guided cascade hourglass network for depth completion. In *IEEE/CVF Winter Conference on Applications of Computer Vision* (pp. 32–40).

Lin, Y., Cheng, T., Zhong, Q., Zhou, W., & Yang, H. (2022). Dynamic spatial propagation network for depth completion. *In Proceedings of the AAAI Conference on Artificial Intelligence,36*, 1638–1646.

Li, Z., Shi, Z., & Sun, J. (2017). Point integral method for solving poisson-type equations on manifolds from point clouds with convergence guarantees. *Communications in Computational Physics, 22*(1), 228–258.

Liu, S., De Mello, S., Gu, J., Zhong, G., Yang, M.-H., & Kautz, J. (2017). Learning affinity via spatial propagation networks. *Advances in Neural Information Processing Systems,30*.

Liu, X., Shao, X., Wang, B., Li, Y., & Wang, S. (2022). Graphcspn: Geometry-aware depth completion via dynamic gcns. In *European Conference on Computer Vision (ECCV)* (pp. 90–107). Springer

Liu, L.-K., Chan, S. H., & Nguyen, T. Q. (2015). Depth reconstruction from sparse samples: Representation, algorithm, and sampling. *IEEE Transactions on Image Processing, 24*(6), 1983–1996.

Liu, L., Song, X., Sun, J., Lyu, X., Li, L., Liu, Y., & Zhang, L. (2023). Mff-net: Towards efficient monocular depth completion with multi-modal feature fusion. *IEEE Robotics and Automation Letters, 8*(2), 920–927.

López-Randulfe, J., Veiga, C., Rodríguez-Andina, J. J., & Farina, J. (2017). A quantitative method for selecting denoising filters, based on a new edge-sensitive metric. In *2017 IEEE International Conference on Industrial Technology (ICIT)* (pp. 974–979). IEEE

Lopez-Rodriguez, A., Busam, B., & Mikolajczyk, K. (2022). Project to adapt: Domain adaptation for depth completion from noisy and sparse sensor data. *International Journal of Computer Vision,* 1–17.

Ma, F., & Karaman, S. (2018). Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 4796–4803). IEEE.

Ma, X., Liu, S., Xia, Z., Zhang, H., Zeng, X., & Ouyang, W. (2020). Rethinking pseudo-lidar representation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16* (pp. 311–327). Springer.

Milanfar, P. (2012). A tour of modern image filtering: New insights and methods, both practical and theoretical. *IEEE Signal Processing Magazine, 30*(1), 106–128.

Monga, V., Li, Y., & Eldar, Y. C. (2021). Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine, 38*(2), 18–44.

Mufti, F., & Mahony, R. (2011). Statistical analysis of signal measurement in time-of-flight cameras. *ISPRS Journal of Photogrammetry and Remote Sensing, 66*(5), 720–731.

Ortega, A., Frossard, P., Kovačević, J., Moura, J. M., & Vandergheynst, P. (2018). Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE, 106*(5), 808–828.

Osher, S., Shi, Z., & Zhu, W. (2017). Low dimensional manifold model for image processing. *SIAM Journal on Imaging Sciences, 10*(4), 1669–1690.

Pang, J., & Zeng, J. (2021). Graph spectral image restoration. *Graph Spectral Image Processing,133*.

Pang, J., & Cheung, G. (2017). Graph Laplacian regularization for image denoising: Analysis in the continuous domain. *IEEE Transactions on Image Processing, 26*(4), 1770–1785.

Park, J., Joo, K., Hu, Z., Liu, C.-K., & So Kweon, I. (2020). Non-local spatial propagation network for depth completion. In *European Conference on Computer Vision (ECCV)* (pp. 120–136). Springer.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., & Lerer, A. (2017). Automatic differentiation in pytorch.

Perona, P., & Malik, J. (1990). Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence, 12*(7), 629–639.

Qiu, J., Cui, Z., Zhang, Y., Zhang, X., Liu, S., Zeng, B., & Pollefeys, M. (2019). Deeplidar: Deep surface normal guided depth prediction for outdoor scene from sparse lidar data and single color image. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 3313–3322).

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-assisted Intervention* (pp. 234–241). Springer.

Shewchuk, J. R., et al. (1994). *An introduction to the conjugate gradient method without the agonizing pain*. Department of Computer Science, Carnegie-Mellon University.

Shi, Z., Sun, J., & Tian, M. (2018). Harmonic extension on the point cloud. *Multiscale Modeling & Simulation, 16*(1), 215–247.

Silberman, N., Hoiem, D., Kohli, P., & Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision (ECCV)* (pp. 746–760). Springer.

Strong, D. M., & Chan, T. F. (1996). Spatially and scale adaptive total variation based regularization and anisotropic diffusion in image processing. In *Diusion in Image Processing, UCLA Math Department CAM Report*. Citeseer.

Sun, W., Zhu, Q., Li, C., Feng, R., Zhou, S., Jiang, J., Yang, Q., Loy, C. C., Gu, J., Hou, D., et al. (2023). Mipi 2022 challenge on rgb+ tof depth completion: Dataset and report. In *European Conference on Computer Vision (ECCV) Workshop* (pp. 3–20). Springer.

Uhrig, J., Schneider, N., Schneider, L., Franke, U., Brox, T., & Geiger, A . (2017). Sparsity invariant cnns. In *International Conference on 3D Vision (3DV)* (pp. 11–20). IEEE

Van Gansbeke, W., Neven, D., De Brabandere, B., & Van Gool, L. (2019). Sparse and noisy lidar completion with rgb guidance and uncertainty. In *International Conference on Machine Vision Applications (MVA)* (pp. 1–6). IEEE.

Xu, Y., Zhu, X., Shi, J., Zhang, G., Bao, H., & Li, H. (2019). Depth completion from sparse lidar data with depth-normal constraints. In *International Conference on Computer Vision (ICCV)* (pp. 2811–2820).

Yan, Z., Wang, K., Li, X., Zhang, Z., Li, J., & Yang, J. (2022). Rignet: Repetitive image guided network for depth completion. In *European Conference on Computer Vision* (pp. 214–230). Springer.

You, Y., Wang, Y., Chao, W.-L., Garg, D., Pleiss, G., Hariharan, B., Campbell, M., & Weinberger, K. Q. (2020). Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. In *International Conference on Learning Representations (ICLR)*.

Zeng, J., Tong, Y., Huang, Y., Yan, Q., Sun, W., Chen, J., & Wang, Y. (2019). Deep surface normal estimation with hierarchical rgb-d fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 6153–6162).

Zhang, Y., Guo, X., Poggi, M., Zhu, Z., Huang, G., & Mattoccia, S. (2023). Completionformer: Depth completion with convolutions and vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 18527–18536).

Zhao, S., Gong, M., Fu, H., & Tao, D. (2021). Adaptive context-aware multi-modal network for depth completion. *IEEE Transactions on Image Processing, 30*, 5264–5276.

Zhou, W., Yan, X., Liao, Y., Lin, Y., Huang, J., Zhao, G., Cui, S., & Li, Z. (2023). Bev@ dc: Bird's-eye view assisted training for depth completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 9233–9242).