# Revisiting Deformable Convolution for Depth Completion

Xinglong Sun[1], Jean Ponce[2], and Yu-Xiong Wang[3]

***Abstract*— Depth completion, which aims to generate high-quality dense depth maps from sparse depth maps, has attracted increasing attention in recent years. Previous work usually employs RGB images as guidance, and introduces iterative spatial propagation to refine estimated coarse depth maps. However, most of the propagation refinement methods require several iterations and suffer from a fixed receptive field, which may contain irrelevant and useless information with very sparse input. In this paper, we address these two challenges simultaneously by revisiting the idea of deformable convolution. We propose an effective architecture that leverages deformable kernel convolution as a single-pass refinement module, and empirically demonstrate its superiority. To better understand the function of deformable convolution and exploit it for depth completion, we further systematically investigate a variety of representative strategies. Our study reveals that, different from prior work, deformable convolution needs to be applied on an estimated depth map with a relatively high density for better performance. We evaluate our model on the large-scale KITTI dataset and achieve state-of-the-art level performance in both accuracy and inference speed. Our code is available at https://github.com/AlexSunNik/ReDC.**

## I. INTRODUCTION

Reliable depth information is fundamental in many real-world applications, such as robotics [1], autonomous driving [2], and 3D mapping [3]. It is critical for agents to better sense the surrounding environment for more accurate and effective actions. However, existing depth sensors usually produce depth maps with incomplete data. For example, LiDAR sensors have a limited number of scanlines and operating frequencies, thus producing very sparse depth measurements. Therefore, depth completion, which aims to estimate dense depth maps from the sparse ones, has received increasing attention recently.

A wide variety of deep convolutional neural network [4] based approaches have been developed for effective depth completion. They often formulate the task as a learned interpolation, using high-quality RGB images from camera sensors as guidance to up-sample sparse points into dense depth images. Inspired by monocular depth estimation [5]–[8], these approaches [9]–[19] typically use an encoder-decoder network. In addition, to better leverage the two different modalities of color and depth data, two-branch network architectures are adopted.

[1]Xinglong Sun is a graduate student in the Department of Computer Science at Stanford University, xs15@stanford.edu

[2]Jean Ponce is a Professor in the Computer Science Department of Ecole normale supérieure (ENS-PSL, CNRS, Inria) and a Global Distinguished Professor in the Courant Institute and the Center for Data Science at New York University, jean.ponce@ens.fr

[3]Yu-Xiong Wang is an assistant professor in the Department of Computer Science at University of Illinois Urbana-Champaign, yxw@illinois.edu

Nonetheless, the depth maps directly output from the decoder are often still blurry or low in quality. Concretely, as observed in [20], depth maps directly produced by deep decoders may not preserve the input depth values at valid pixels. Many approaches [20]–[24] thus introduce efficient spatial propagation to refine coarse depth maps. However, most of the propagation refinement methods suffer from two major drawbacks. First, propagation at each pixel is performed with a uniform-size receptive field and a regular convolution. This limitation originates from the fixed geometric structure of the convolution module. A convolution unit can only sample the input feature map at fixed locations, which may contain *irrelevant and useless information with very sparse input*. Second, the propagation refinement module usually needs to refine the coarse depth map for *several iterations* to obtain satisfactory performance, which is not ideal for good hardware latency and memory footprint.

In this paper, we address these two challenges simultaneously by **Re**visiting the idea of ***D**eformable **C**onvolution* [25] (**ReDC**), considering its capability of adaptively generating different receptive fields by learning offsets on regular grid positions. *Our key insight* is that such a capability (i) is desirable for depth completion so as to adaptively attend to the most informative regions, and (ii) should be exploited to refine coarse depth maps. To this end, we propose an effective and efficient architecture that leverages deformable kernel convolution as a refinement module and empirically demonstrate its superiority. We build upon the PENet [26] backbone, one of the publicly released state-of-the-art methods. We observe a significant improvement over PENet with our architecture. Notably, we found that with our deformable refinement module, the coarse depth *only needs a single pass of refinement*, distinguishing our model from those iterative ones [22]–[24], [26] and enjoying noticeable inference speedup.

Our approach is also different from and substantially outperforms prior work that uses deformable convolution for depth completion tasks [23], [24], [27], whose performance is still inferior to the state of the art. For example, Deformable Kernel Network (DKN) [27] obtains strong performance on the NYU-v2 benchmark [28], but it performs poorly on the more challenging KITTI benchmark [14], [29], where the input depth maps exhibit much higher sparsity. Similarly, DSP [24] and NLSPN [23] fail to achieve competitive performance on KITTI, despite several iterations of refinement.

To better understand the function of deformable convolution and leverage it for depth completion, we systematically investigate a variety of representative strategies other than

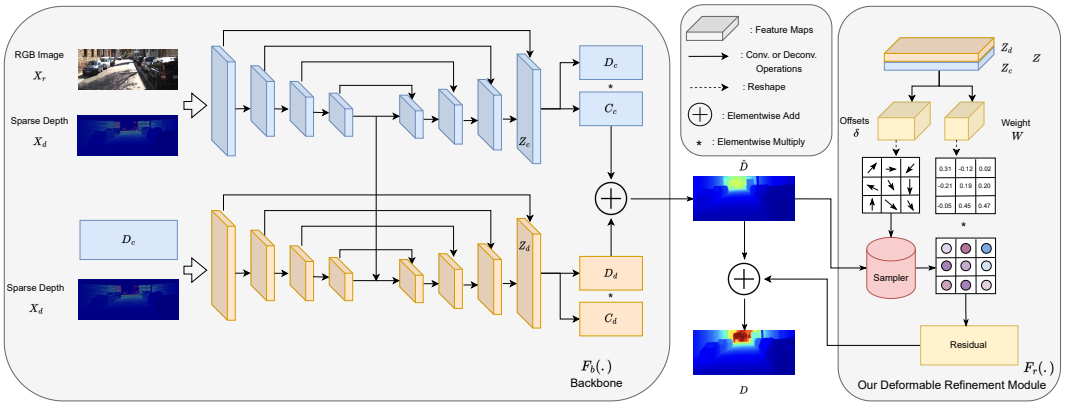


Fig. 1. Overview of our method ReDC. We first generate a coarse depth map $\hat{D}$ from the backbone. Then, we pass it through our deformable refinement module. We feed the feature maps coming from the final layer of the backbone decoder into a $1 \times 1$ convolution layer to regress a set of weights and offsets for performing the interpolation on $\hat{D}$ in a deformable manner.

our proposed approach. Our study reveals that different ways of employing deformable convolution lead to significantly different performance. *Importantly, it needs to be applied on a depth map with a relatively high density for better performance*. In other words, a direct application of deformable convolution as in DKN [27] does not work with high input depth sparsity like KITTI. This further supports our proposed method of using deformable refinement over coarse depth.

We summarize our contributions as follows:

- We conduct a thorough study on the most effective way of leveraging deformable convolution, promoting a deeper understanding of deformable convolution within the depth completion research community.
- We propose a novel and effective depth completion architecture that requires only a single pass through the depth refinement module, achieving more hardware and memory-friendly performance.
- We achieve competitive state-of-the-art level depth completion performance in accuracy and inference speed on the challenging KITTI dataset and surpass prior work by a clear margin.

## II. RELATED WORK

### A. Depth Completion

Depth completion aims to generate dense depth maps by completing sparse depth maps. Some work [14], [30] produces dense depth maps from monocular sparse depth maps only, while the majority of work [16], [20], [31], [32] employs guidance or reference images like high-quality RGB images to assist the dense depth map generation. These approaches try to solve the challenge of irregular and extremely sparse input depth and the two completely different modalities of color and depth with sparse and spatially-invariant convolution [14], [31], [33], [34], uncertainty exploration [15], [35], and model fusion strategies [36]. Additionally, some representative work tries to draw insights from other tasks and exploits multi-scale features [34], [37]–[39], surface normals [16], [17], semantic information [13], [40], and context affinity [20], [22], [23] to improve depth completion performance. Recently, PENet is proposed and incorporates several effective features useful to boost depth completion performance. Moreover, it proposes a simple convolutional layer to encode 3D geometric cues which are shown to further improve the performance. PENet is one of the best publicly released models on the KITTI depth completion benchmark, and we study our deformable refinement module upon it and observe a clear improvement.

### B. Deformable Convolution

The idea of deformable convolution is first introduced and explored in DCN [25] to enhance the transformation modeling capability of convolutional neural networks (CNNs). Due to the intrinsic nature of the depth completion task, deformable convolution is quickly exploited to extract more useful information from the very sparse depth maps. DKN [27] explicitly learns sparse and spatially-invariant kernels for depth completion by regressing offsets to sample pixels at adaptive locations. It obtains superior performance on the NYU-v2 depth [28] completion benchmark, indicating the potential effectiveness of the DKN module on the depth completion task. However, our investigation shows that its performance is degraded on the more challenging KITTI benchmark with input depth of much higher sparsity. DSP [24] adaptively learns a different receptive field and affinity matrix between non-local pixels by generating memory-costly embedding for each non-local pixel, which is impractical on edge devices. Still, it fails to reach state-of-the-art level performance on KITTI and requires a minimum of 3 iterations of refinement for decent performance, which is also the case in another deformable model NLSPN [23]. We conduct a thorough study on deformable convolution and explore its more effective usage with very sparse input depth maps like KITTI. We propose a new efficient architecture which achieves state-of-the-art level performance while using much less inference time, and we comprehensively analyze its effectiveness through experiments.

## III. METHODOLOGY

We design an end-to-end trainable model ReDC with our proposed deformable refinement module, as illustrated in Fig. 1. We first feed a sparse depth map and a guidance RGB image into the backbone model, which produces a

coarse depth map. We then use our refinement module based on deformable kernel convolution to improve its quality. As mentioned previously, we only pass it through the refinement module *once*.

### A. Overall Procedure

We represent the input sparse depth map as $X_d$, the guidance RGB image as $X_r$, and the provided groundtruth dense map in the training set as $Y$. We denote our completion model as $F(.)$ which produces a dense depth map $D$ from $X_d$ and $X_r$. Concretely, this is expressed as $D = F(X_d, X_r)$. The completion model contains two parts, namely, the backbone $F_b(.)$ which produces a coarse depth map $\hat{D}$ and the refinement module $F_r(.)$ which improves the quality of the coarse depth. The overall model can be expressed as:

$$D = F_r(F_b(X_d, X_r)). \tag{1}$$

We next discuss the backbone choice and our deformable refinement module.

### B. Backbone

We follow PENet [26] for the backbone choice. The backbone network has a two-branch architecture, with one branch extracting the color-dominant information and the other extracting the depth-dominant information. Each branch is of an encoder-decoder structure, and the results coming from both branches are fused to generate the coarse depth $\hat{D}$. Depth generated by the color-dominant branch is denoted as $D_c$, and depth from the depth-dominant branch is denoted as $D_d$. For fusing the two depth maps, confidence masks are also generated, which are $C_c$ for $D_c$ and $C_d$ for $D_d$. We perform the fusion following PENet and FusionNet [35]. Concretely, the final fused depth $\hat{D}$ is computed as:

$$\hat{D}(u,v) = \frac{D_d(u,v) \cdot e^{C_d(u,v)} + D_c(u,v) \cdot e^{C_c(u,v)}}{e^{C_d(u,v)} + e^{C_c(u,v)}}, \tag{2}$$

where $(u, v)$ denotes the location of each pixel. In practice, a softmax is applied on $C_c$ and $C_d$ for normalization.

### C. Refinement Module

We now present our deformable refinement module in detail and how we utilize it to improve the quality of the generated depth. Following previous work [27], [41]–[44], we represent each point of our refined depth $D$ as a weighted average of points sampled from the coarse depth $\hat{D}$. Specifically, we represent it as:

$$D_p = \sum_{q \in \mathcal{N}(p)} W_{pq}(Z) \hat{D}_q, \tag{3}$$

where $p = (u, v)$ represents the point coordinate on the depth map, $\mathcal{N}(p)$ defines a set of neighbors near the point $p$, and $Z$ is some guidance like extracted feature maps. Most of the approaches [22], [45]–[47] mainly focus on learning the weights $W$ and employ a fixed square grid as predefined neighbors to sample points. Deformable approaches [24], [25], [27] try to learn a set of offsets added to each sampled location in the regular square grid, producing a deformable neighborhood region.

Here, we show how we design the architecture to learn the kernel weights $W$ and neighborhood $\mathcal{N}$ offsets. We denote the feature map coming from the last layer of the decoder of the color-dominant branch as $Z_c$ and that from the depth-dominant branch as $Z_d$, as illustrated in Fig. 1. We concatenate the two feature maps channel-wise and represent the combined feature as $Z = \{Z_d, Z_c\}$. Later, we add a $1 \times 1$ convolutional layer taking the combined feature $Z$ as input. It outputs a feature map of size $k^2 \times 1 \times 1$, where $k$ represents the size of the interpolation kernel $W$. We then apply $sigmoid(.)$ on it to estimate the interpolation kernel weights. Similar to [27], we found that a softmax instead of sigmoid layer as used in [48]–[50] yields no performance improvement. Same as the weights, we add another $1 \times 1$ convolutional layer on feature $Z$ to regress the offsets as well. It produces an output feature map of size $2k^2 \times 1 \times 1$ containing the relative offsets (both in $x$ and $y$ directions) we use to apply to sampled locations on a regular grid. For example, with $k = 3$, we would learn 9 offsets used to sample 9 points sparsely chosen from a large neighborhood area. Similar to [27], we choose to compute the residual instead of directly generating the final output. The refinement of the coarse depth $\hat{D}$ can thus be expressed as follows:

$$D_p = \hat{D}_p + \sum_{q \in \mathcal{N}(p)} W_{ps}(Z) \hat{D}_{s(q)}, \tag{4}$$

where $\mathcal{N}(p)$, as previously described, is a local $k \times k$ regular square window centered at location $p$. We express $s(q)$ the sampled position as $s(q) = q + \delta q$, where $\delta q$ comes from the $2k^2 \times 1 \times 1$ shaped offset map $\delta$ we regress. Since $s(q)$ could be fractional, in order to compute $\hat{D}_{s(q)}$, we utilize a two-dimensional bilinear kernel as in [25], [51] to compute its value based on its 4 neighbors, detailed as follows:

$$\hat{D}_{s(q)} = \sum_{t \in \mathcal{R}(s(q))} G(s(q), t) \hat{D}_t, \tag{5}$$

where $\mathcal{R}(s(q))$ lists all of the four integer neighbors of $s(q)$ and $G$ is a sampling kernel defined as:

$$G(s,t) = \max(0, 1 - |s_x - t_x|) \cdot \max(0, 1 - |s_y - t_y|). \tag{6}$$

### D. Training Loss

We use a combined $\ell_2$ and $\ell_1$ loss for training our model, which is defined as:

$$\begin{aligned} L(D, Y) = \alpha \| (D - Y) \odot \mathbb{1}(Y > 0) \|^2 + \\ (1 - \alpha) \| (D - Y) \odot \mathbb{1}(Y > 0) \|. \end{aligned} \tag{7}$$

Here, $\odot$ is the elementwise multiplication. For datasets like KITTI, even the groundtruth depth map is of relatively high sparsity, containing many invalid pixels. We follow the common practice and only consider those having valid depth values with the mask $\mathbb{1}(Y > 0)$.

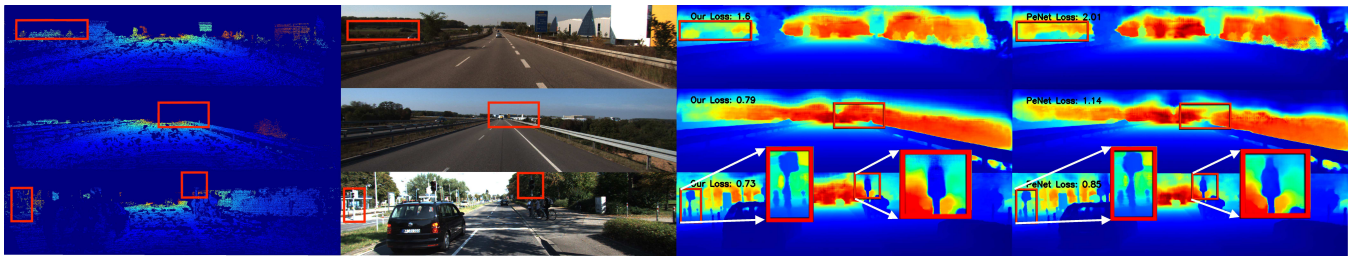


Fig. 2. Visualization of inference results obtained by our model ReDC and PENet [26] on three challenging samples from the KITTI *training* set. From **left** to **right**: groundtruth dense depth map, guidance RGB image, inference from our model, and inference from PENet. $\ell_2$ loss computed between the groundtruth and inference result is presented at the top left corner. Regions to focus on are highlighted in red boxes, showing that our ReDC achieves better performance with more accurate depth and finer detail.

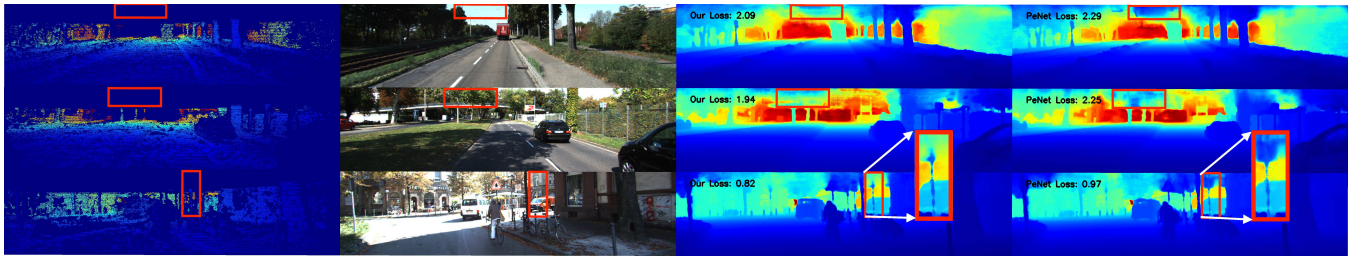


Fig. 3. Visualization of inference results obtained by our model ReDC and PENet [26] on three challenging samples from the KITTI *validation* set. Notation follows Figure 2. Consistent with Figure 2, our ReDC achieves better performance with more accurate depth and finer detail.

## IV. EXPERIMENTS

### A. Experiment Settings

**Dataset:** We evaluate our approach on the KITTI [29] Depth Completion benchmark [14], with sparse depth maps obtained by projecting 3D LiDAR points to corresponding image frames. The density of the valid depth points is around $5\%$ for the sparse depth maps and $16\%$ for the groundtruth dense depth maps. The dataset contains $93K$ data samples, with $86K$ for training, $7K$ for validation, and $1K$ for testing.

**Evaluation Metrics:** We follow the common practice and evaluate our approach under four standard metrics [14], including root mean square error (RMSE) in $mm$, mean absolute error (MAE) in $mm$, root mean squared error of the inverse depth (iRMSE) in $1/km$, and mean absolute error of the inverse depth (iMAE) in $1/km$. We also report the inference time for a single depth-image completion in seconds on RTX 2080Ti GPU with a 2.5GHz processor.

**Baselines:** We mainly compare with competitive *publicly* released and *peer-reviewed* depth completion methods, including PENet [26] and DSP [24]. In addition to the discussed methods in Sec. II, we include very recent baselines like MFF-Net [18], RigNet [52], and GAENet [53], as well as well-referenced baselines like PwP [17] and UberATG [11]. Finally, we evaluate alternative strategies of using deformable convolution for depth completion, such as DKN [27].

**Implementation Detail:** We implement our approach and conduct all experiments using PyTorch [54] with four NVIDIA RTX 2080Ti GPUs running in parallel. We train with an Adam optimizer [55] with weight decay set to $1e-5$, $\beta_1$ set to 0.9, and $\beta_2$ set to 0.99. We run the experiment for a total of 30 epochs with an initial learning rate set to $1e-3$, decaying in a cosine annealing fashion. The batch size is set to 2 samples per GPU for each step. We choose $k = 3$ which is the size of the interpolation kernel $W$ discussed in Section III-C. The loss function is the combined $\ell_2$ and $\ell_1$ losses with $\alpha$ set to 0.5. In terms of training data augmentation, we follow the common practice and adopt random cropping, random flipping, and color jittering [56], [57].

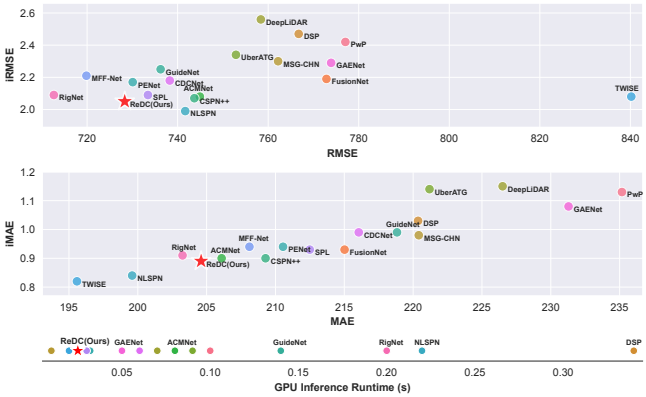


Fig. 4. Our model ReDC clearly outperforms competitive publicly released methods in terms of accuracy metrics and inference runtime on the KITTI test set. All metrics are the lower the better. Numerically, our ReDC achieves 728.31 (RMSE), 204.60 (MAE), 2.05 (iRMSE), and 0.89 (iMAE).

### B. Main Results

**Quantitative Comparisons:** Figure 4 summarizes the comparisons with baselines on the KITTI test set. Notice that all methods are trained and tuned on the *KITTI released training and validation splits* using sparse and dense depth maps of the *provided default density levels*. In terms of the inference runtime, we only include the reported results from baselines *measured on GPU for a fair comparison*. Our method ReDC achieves *state-of-the-art* level performance in both *accuracy* (RMSE, MAE, iRMSE, and iMAE) and *inference runtime*. Notably, our approach clearly outperforms PENet, which our backbone is based on (Sec. III). This result

validates the effectiveness of our deformable refinement module. Moreover, we include the comparison with methods like DSP and NLSPN which also use deformable convolution as part of their depth completion models. In addition to the superior accuracy, our model significantly surpasses them in terms of inference speed, demonstrating the strength of our architecture.

**Visualizations:** We now visually compare the generated depth maps from our model and the top-performing baseline PENet in Figures 2 and 3. Note that, while the quantitative metrics (Section IV-A) are computed only over the pixel locations with valid groundtruth depth values, these visualizations provide more comprehensive depth completion results over *all* pixels. Therefore, they are complementary to demonstrate the superiority of our approach.

In Figure 2, we show the three most challenging scenes from the training set for both PENet and our method. We visualize the groundtruth depth maps, RGB images, and inference results from PENet and our model. We also provide the $\ell_2$ loss computed between the groundtruth depth maps and inference results at the top left corner as an informative indicator of the quality of the generated depth maps. We highlight some challenging regions of interest in red boxes, where our model performs substantially better than PENet. We discuss the observation in detail as follows.

(I) *Ours outperforms PENet on points with groundtruth.* In the highlighted box of the first hard training sample, we see valid depth points in the groundtruth depth map. Our generated depth correctly predicts the depth values in this region, with the same visualization color as the groundtruth. By contrast, PENet produces incorrect depth values.

(II) *Ours also outperforms PENet on points without groundtruth.* In the highlighted box of the second hard training sample, we see that there are completely no valid data in the provided groundtruth depth map. For reasonable estimation in this case, the models have to rely on the RGB image or leverage the depth data from informative neighborhood. Our model correctly estimates the depth values in this region and maintains smoothness with the surrounding area. This is attributed to our deformable refinement module, which is able to *adaptively attend to the informative regions*. However, PENet fails on such a challenging region, producing a very sharp transition and gap in the middle.

(III) *Ours produces depth with finer detail and structure.* In the highlighted box of the third sample, we observe thin traffic light poles in the image. Our model produces their depth with finer detail and structure than that of PENet.

We also provide the same visualization analysis on the validation set in Figure 3. We observe the similar phenomenon that our model leads to more accurate depth estimation in challenging regions which contain *very few or a limited number of valid depth points* in groundtruth images, and produces object depth with finer detail and structure. For example, in the highlighted box of the third hard validation example, we see that our model generates the depth of a tall traffic pole with very high quality; whereas, PENet produces a bunch of messy point clouds around the top of the pole.

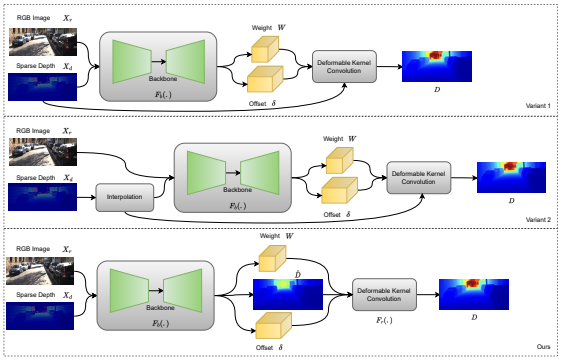


Fig. 5. Illustration of variants of leveraging deformable convolution for depth completion. Our strategy works the best, which generates an intermediate coarse depth map and then refines it by our deformable refinement module.

TABLE I

Investigation on variants of leveraging deformable convolution for depth completion on the KITTI validation set, demonstrating the superiority of our approach. 'NN' denotes nearest-neighbor.

| Model | RMSE (↓) | MAE (↓) | iRMSE (↓) | iMAE (↓) |
|---|---|---|---|---|
| Variant 1 w/ smaller backbone [27] | 1130.47 | 241.58 | 4.00 | 0.97 |
| Variant 1 w/ our backbone | 900.54 | 251.32 | 2.59 | 0.99 |
| Variant 2 w/ NN interpolation | 875.75 | 235.58 | 2.40 | 0.98 |
| ReDC (Ours) | **813.05** | **218.30** | **2.32** | **0.89** |

TABLE II

Investigation on different interpolation techniques. Results are reported as directly testing interpolated depth on the KITTI validation set without any model learning. Nearest-neighbor interpolation performs the best overall.

| Interpolation | RMSE (↓) | MAE (↓) | iRMSE (↓) | iMAE (↓) |
|---|---|---|---|---|
| RBF | **2014.52** | 794.89 | inf[1] | inf |
| Linear | 2265.33 | 507.93 | inf | inf |
| Cubic | 3022.93 | 644.99 | inf | inf |
| Nearest | 2139.25 | **454.42** | **6.45** | **1.83** |

### C. Investigation on Variants of Our Model

Our evaluation so far has validated the importance of deformable convolution for depth completion, in a way of refining the estimated coarse depth. Here we further investigate alternative strategies of exploiting deformable convolution, and show that our current approach is the most effective to employ it to improve depth completion performance, *especially when the input depth maps are of very high sparsity*. Figure 5 illustrates several representative variants and Table I summarizes their performance.

Specifically, (I) we start from a straightforward use of deformable convolution as **Variant 1**, where we directly apply the learned weights and offsets on the *input* sparse depth $X_d$ following DKN [27]. We use the same backbone as in DKN. This method has been shown to achieve high performance on the NYU-v2 dataset [28] where the density of the input depth maps is much higher than KITTI [27]. However, as shown in Table I, it achieves very poor performance with RMSE of 1130.47 on KITTI. (II) We then upgrade **Variant 1 with a more powerful backbone** – the one used in both PENet and our current model. Despite the observed improvement (900.54 v.s. 1130.47 RMSE), it still fails to match competitive depth completion performance on KITTI. This shows that it is difficult to operate deformable convolution directly

[1]iRMSE and iMAE are inverse metrics. They attain an 'inf' (infinite) value, when a model generates 0 in output depth at valid pixels.

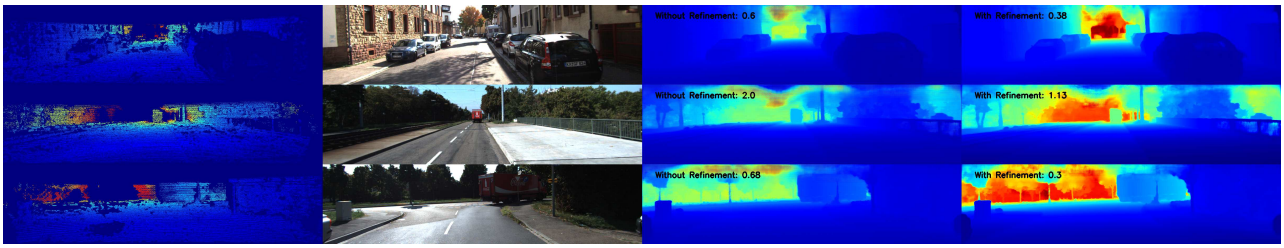


Fig. 6. Visualization of inference results obtained by our model ReDC before and after the deformable refinement module. From **left** to **right**: groundtruth dense depth, guidance RGB image, coarse depth $\hat{D}$ before refinement, and final depth output $D$ after refinement. $\ell_2$ loss computed between the groundtruth and inference result is presented at the top left corner. The coarse depth map is significantly refined by our deformable refinement module *with only a single pass*.

on *input depth of high sparsity*, irrespective of an improved feature extractor or backbone.

We thus conjecture that deformable kernel convolution requires a relatively high density for input depth maps to obtain good performance. Based on this assumption, we pre-process the depth maps to increase their density by using various interpolation techniques. We directly test the interpolated depth from the sparse depth maps against the groundtruth on the KITTI validation set, and report the results in Table II. We observe that the nearest-neighbor interpolation, filling the empty depth values based on their nearest neighbors, achieves the best overall performance.

Based on this result, (III) we further explore **Variant 2** (Figure 5), where we feed the nearest-neighbor interpolated depth into Variant 1 with a strong backbone. As expected, we observe an improvement (875.75 v.s. 900.54 RMSE) in Table I, which validates our assumption that the density of the input depth maps is crutial for the performance of deformable convolution. Compared with Variant 2, our model is substantially better under all the metrics, since we use a *learned* coarse depth map. Collectively, the comparisons with Variants 1 and 2 in Table I demonstrate the superiority of our strategy – an intermediate coarse depth map is first generated under the guidance of an RGB image via a deep backbone; then, it gets improved by our deformable refinement module.

### D. Ablation Studies

We conduct a variety of ablation studies to further demonstrate the performance improvement brought by our refinement module, and investigate the effect of different components and hyperparameter settings in our approach.

**Coarse-to-Fine Procedure:** Figure 6 visually compares the generated coarse depth $\hat{D}$ before our refinement module and the final depth map $D$. We also provide the $\ell_2$ loss computed between the groundtruth depth maps and the generated depth maps at the top left corner as in Section IV-B. We observe that the quality of the generated depth gets significantly improved by our refinement module. Depth values in many areas of the coarse depth maps get corrected, and the depth detail of objects also gets refined. We also provide the result of training and validating the backbone alone without our refinement module in Table III which shows obviously worse performance (835.37 v.s. 813.05 RMSE), highlighting the effectiveness of our refinement.

**Hyperparameter Settings:** We study the impact of different hyperparameter settings in Table III. (I) **Learning rate schedule:** Instead of decaying the learning rate in a cosine annealing fashion, we use a staircase schedule. Specifically, the learning rate gets decayed by $\{0.5, 0.2, 0.1\}$ at epoch $\{10, 15, 25\}$. The performance is not as good as cosine annealing (820.76 v.s. 813.05 RMSE). (II) **Training objective:** The performance gets worse (815.53 v.s. 813.05 RMSE), when the training objective is switched to only an $\ell_2$ loss instead of a combined $\ell_2$ and $\ell_1$ loss. The performance further drops, when we use both staircase decay and $\ell_2$ loss (821.53 v.s. 813.05 RMSE). (III) **Deformable kernel size:** We investigate whether a larger deformable kernel $W$ would improve the completion performance. We observe that, when $k = 5$, the performance gets significantly degraded (850.34 v.s. 813.05 RMSE). These ablations show that our current hyperparameter setting achieves the best performance.

TABLE III

Ablation studies on hyperparameter settings on the KITTI validation set. Our current setting achieves the best performance.

| Model | RMSE (↓) | MAE (↓) | iRMSE (↓) | iMAE (↓) |
|---|---|---|---|---|
| Backbone Only | 835.37 | 235.00 | 2.47 | 0.97 |
| ReDC w/ staircase lr decay | 820.76 | 228.31 | **2.30** | 0.95 |
| ReDC w/ $\ell_2$ loss | 815.53 | 227.35 | 2.38 | 0.95 |
| ReDC w/ $\ell_2$ loss + staircase | 821.53 | 233.90 | 2.32 | 0.98 |
| ReDC $k = 5$ | 850.34 | 234.31 | **2.30** | 0.95 |
| ReDC (Ours) | **813.05** | **218.30** | 2.32 | **0.89** |

## V. CONCLUSION

In this paper, we revisited deformable convolution and studied its best usage on depth completion with very sparse depth maps. We found that different ways of employing deformable convolution could lead to completely distinct performance, and it needs to be applied on a depth map with a relatively high density for optimal performance. We proposed a new, effective depth completion architecture that requires only a single pass through our deformable refinement module, which separates our model from previous ones which usually require a minimum of three passes to get good performance. We achieved competitive state-of-the-art level depth completion performance on the challenging KITTI dataset and surpassed previous work by a clear margin. We believe that this work will deepen the understanding of deformable convolution in the depth completion community.

## ACKNOWLEDGMENT

This work was supported in part by NSF Grant 2106825, NIFA Award 2020-67021-32799, the Jump ARCHES endowment, the NCSA Fellows program, the Inria/NYU collaboration, the Louis Vuitton/ENS chair on artificial intelligence

and the French government under management of Agence Nationale de la Recherche as part of the Investissements d'avenir program, reference ANR19-P3IA0001 (PRAIRIE 3IA Institute). We thank Beomjun Kim and Bumsub Ham for providing their DKN code.

## REFERENCES


[1] K. Shankar, M. Tjersland, J. Ma, K. Stone, and M. Bajracharya, "A learned stereo depth system for robotic manipulation in homes," *RA-L*, 2022. 1

[2] O. Natan and J. Miura, "End-to-end autonomous driving with semantic depth cloud mapping and multi-agent," *T-IV*, 2022. 1

[3] T. Hervier, S. Bonnabel, and F. Goulette, "Accurate 3D maps from depth images and motion sensors via nonlinear kalman filtering," in *IROS*, 2012. 1

[4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 1998. 1

[5] Z. Li and N. Snavely, "MegaDepth: Learning single-view depth prediction from internet photos," in *CVPR*, 2018. 1

[6] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *CVPR*, 2018. 1

[7] X. Pan, T. Zhang, and H. Wang, "A method for handling multi-occlusion in depth estimation of light field," in *ICIP*, 2019. 1

[8] S. Kumari, R. R. Jha, A. Bhavsar, and A. Nigam, "AUTODEPTH: Single image depth map estimation via residual CNN encoder-decoder and stacked hourglass," in *ICIP*, 2019. 1

[9] F. Ma and S. Karaman, "Sparse-to-dense: Depth prediction from sparse depth samples and a single image," in *ICRA*, 2018. 1

[10] S. Shivakumar, T. Nguyen, I. Miller, S. Chen, V. Kumar, and C. Taylor, "DFuseNet: Deep fusion of RGB and sparse depth information for image guided dense depth completion," in *ITSC*, 2019. 1

[11] Y. Chen, B. Yang, M. Liang, and R. Urtasun, "Learning joint 2D-3D representations for depth completion," 2019. 1, 4

[12] S. Zhao, M. Gong, H. Fu, and D. Tao, "Adaptive context-aware multi-modal network for depth completion," *TIP*, 2021. 1

[13] M. Jaritz, R. De Charette, E. Wirbel, X. Perrotton, and F. Nashashibi, "Sparse and dense data with CNNs: Depth completion and semantic segmentation," in *3DV*, 2018. 1, 2

[14] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger, "Sparsity invariant CNNs," in *3DV*, 2017. 1, 2, 4

[15] A. Eldesokey, M. Felsberg, and F. S. Khan, "Confidence propagation through CNNs for guided sparse depth regression," *PAMI*, 2019. 1, 2

[16] J. Qiu, Z. Cui, Y. Zhang, X. Zhang, S. Liu, B. Zeng, and M. Pollefeys, "DeepLiDAR: Deep surface normal guided depth prediction for outdoor scene from sparse LiDAR data and single color image," in *CVPR*, 2019. 1, 2

[17] Y. Xu, X. Zhu, J. Shi, G. Zhang, H. Bao, and H. Li, "Depth completion from sparse LiDAR data with depth-normal constraints," in *ICCV*, 2019. 1, 2, 4

[18] L. Liu, X. Song, J. Sun, X. Lyu, L. Li, Y. Liu, and L. Zhang, "MFF-Net: Towards efficient monocular depth completion with multi-modal feature fusion," *RA-L*, 2023. 1, 4

[19] X. Liang and C. Jung, "Selective progressive learning for sparse depth completion," in *ICPR*, 2022. 1

[20] X. Cheng, P. Wang, and R. Yang, "Depth estimation via affinity learned with convolutional spatial propagation network," in *ECCV*, 2018. 1, 2

[21] ——, "Learning depth with convolutional spatial propagation network," in *PAMI*, 2019. 1

[22] X. Cheng, P. Wang, C. Guan, and R. Yang, "CSPN++: Learning context and resource aware convolutional spatial propagation networks for depth completion," in *AAAI*, 2020. 1, 2, 3

[23] J. Park, K. Joo, Z. Hu, C.-K. Liu, and I. So Kweon, "Non-local spatial propagation network for depth completion," in *ECCV*, 2020. 1, 2

[24] Z. Xu, H. Yin, and J. Yao, "Deformable spatial propagation networks for depth completion," in *ICIP*, 2020. 1, 2, 3, 4

[25] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *ICCV*, 2017. 1, 2, 3

[26] M. Hu, S. Wang, B. Li, S. Ning, L. Fan, and X. Gong, "PENet: Towards precise and efficient image guided depth completion," in *ICRA*, 2021. 1, 3, 4

[27] B. Kim, J. Ponce, and B. Ham, "Deformable kernel networks for guided depth map upsampling," *IJCV*, 2020. 1, 2, 3, 4, 5

[28] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *ECCV*, 2012. 1, 2, 5

[29] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *IJRR*, 2013. 1, 4

[30] A. Eldesokey, M. Felsberg, K. Holmquist, and M. Persson, "Uncertainty-aware CNNs for depth completion: Uncertainty from beginning to end," in *CVPR*, 2020. 2

[31] J. Hua and X. Gong, "A normalized convolutional neural network for guided sparse depth upsampling." in *IJCAI*, 2018. 2

[32] J. Liu and X. Gong, "Guided depth enhancement via anisotropic diffusion," in *PCM*. 2

[33] A. Eldesokey, M. Felsberg, and F. S. Khan, "Propagating confidences through CNNs for sparse data regression," *BMVA*, 2018. 2

[34] Z. Huang, J. Fan, S. Cheng, S. Yi, X. Wang, and H. Li, "HMS-Net: Hierarchical multi-scale sparsity-invariant network for sparse depth completion," *TIP*, 2019. 2

[35] W. Van Gansbeke, D. Neven, B. De Brabandere, and L. Van Gool, "Sparse and noisy lidar completion with RGB guidance and uncertainty," in *MVA*, 2019. 2, 3

[36] J. Tang, F.-P. Tian, W. Feng, J. Li, and P. Tan, "Learning guided convolutional network for depth completion," *TIP*, 2020. 2

[37] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *NeurIPS*, 2014. 2

[38] B. Wang, Y. Feng, and H. Liu, "Multi-scale features fusion from sparse LiDAR data and single image for depth completion," *Electronics Letters*, 2018. 2

[39] A. Li, Z. Yuan, Y. Ling, W. Chi, S. Zhang, and C. Zhang, "A multi-scale guided cascade hourglass network for depth completion," in *WACV*, 2020. 2

[40] N. Schneider, L. Schneider, P. Pinggera, U. Franke, M. Pollefeys, and C. Stiller, "Semantically guided depth upsampling," in *GCPR*, 2016. 2

[41] K. He, J. Sun, and X. Tang, "Guided image filtering," *PAMI*, 2012. 3

[42] J. Kopf, M. F. Cohen, D. Lischinski, and M. Uyttendaele, "Joint bilateral upsampling," *ToG*, 2007. 3

[43] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *ICCV*, 1998. 3

[44] H. Wu, S. Zheng, J. Zhang, and K. Huang, "Fast end-to-end trainable guided filter," in *CVPR*, 2018. 3

[45] T.-W. Hui, C. C. Loy, and X. Tang, "Depth map super-resolution by deep multi-scale guidance," in *ECCV*, 2016. 3

[46] Y. Li, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep joint image filtering," in *ECCV*, 2016. 3

[47] ——, "Joint image filtering with deep convolutional networks," *PAMI*, 2019. 3

[48] S. Bako, T. Vogels, B. McWilliams, M. Meyer, J. Novák, A. Harvill, P. Sen, T. Derose, and F. Rousselle, "Kernel-predicting convolutional networks for denoising monte carlo renderings," *TOG*, 2017. 3

[49] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive convolution," in *CVPR*, 2017. 3

[50] T. Vogels, F. Rousselle, B. McWilliams, G. Röthlin, A. Harvill, D. Adler, M. Meyer, and J. Novák, "Denoising with kernel prediction and asymmetric loss functions," *TOG*, 2018. 3

[51] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in *NeurIPS*, 2015. 3

[52] Z. Yan, K. Wang, X. Li, Z. Zhang, J. Li, and J. Yang, "RigNet: Repetitive image guided network for depth completion," in *ECCV*, 2022. 4

[53] W. Du, H. Chen, H. Yang, and Y. Zhang, "Depth completion using geometry-aware embedding," in *ICRA*, 2022. 4

[54] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *NeurIPS*, 2019. 4

[55] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015. 4

[56] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *arXiv preprint arXiv:1708.04552*, 2017. 4

[57] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *NeurIPS*, 2012. 4