Abstract

We propose a simple, efficient, yet powerful framework for dense visual predictions based on the conditional diffusion pipeline. Our approach follows a “noise-to-map” generative paradigm for prediction by progressively removing noise from a random Gaussian distribution, guided by the image. The method, called DDP, efficiently extends the denoising diffusion process into the modern perception pipeline. Without task-specific design and architecture customization, DDP is easy to generalize to most dense prediction tasks, e.g., semantic segmentation and depth estimation. In addition, DDP shows attractive properties such as dynamic inference and uncertainty awareness, in contrast to previous single-step discriminative methods. We show top results on three representative tasks with six diverse benchmarks, without tricks, DDP achieves state-of-the-art or competitive performance on each task compared to the specialist counterparts. For example, semantic segmentation (83.9 mIoU on Cityscapes), BEV map segmentation (70.6 mIoU on nuScenes), and depth estimation (0.05 REL on KITTI). We hope that our approach will serve as a solid baseline and facilitate future research.

1. Introduction

Dense prediction tasks are the foundation of computer vision research, including a wide range of perceptual tasks such as semantic segmentation [19, 91], depth estimation [28, 63, 67], and optical flow [26, 28]. These tasks require correctly predicting the discrete labels or continuous values for all pixels in the image, which provides detailed contextual understanding and enables various applications.

Numerous methods have rapidly improved the result of perception tasks over a short period of time. In general terms, these methods can be divided into two paradigms: discriminative-based [27, 88, 78, 17] and generative-based [77, 31, 35, 40, 81]. The former approach, which directly learns the mapping between input-output pairs and predicts in a single forward step, has become the current de-facto choice due to its simplicity and efficiency. Whereas, generative models aim at modeling the underlying distribution of the data, conceptually having a greater capacity to handle challenging tasks. However, they are often restricted by complex architecture customization as well as various training difficulties [60, 37, 6].

These challenges have been largely addressed by the diffusion and score-based models [32, 64, 68]. The solutions, based on denoising diffusion process, are conceptually simple: they apply a continuous diffusion process to transform data into noise and generate new samples by simulating the time-reversed diffusion process. These methods now enable easy training and achieve superior results on various generative tasks [50, 58, 56, 53]. Witnessing these great successes, there has been a recent surge of interest to introduce diffusion models to dense prediction tasks, including
Figure 2. The proposed DDP framework. The image encoder extracts feature representation from the input image $x$ as the condition. The map decoder takes the noisy map $y_t$ as input and produces the denoised prediction under the guidance. During training, the noisy map $y_t$ is constructed by adding Gaussian noise to the encoded ground truth. In inference, the noisy map $y_t$ is randomly sampled from the Gaussian distribution and iteratively refined to obtain the desired prediction $y_0$.

semantic segmentation [1, 13, 75, 74] and depth estimation [61]. However, these methods simply transfer the heavy frameworks from image generation tasks to dense prediction, resulting in low efficiency, slow convergence, and suboptimal performance.

In this paper, we introduce a general, simple, yet effective diffusion framework for dense visual prediction. Our method named as DDP, which extends the denoising diffusion process into the modern perception pipeline effectively (see Figure 2). During training, the Gaussian noise controlled by a noise schedule [51] is added to the encoded ground truth to obtain the noisy maps. Then these noisy maps are fused with the conditional features from the image encoder, e.g., Swin Transformer [45]. Finally, these fused features are fed to a lightweight map decoder to produce the predictions without noise. At the inference phase, DDP generates predictions by reversing the learned diffusion process, which adjusts a noisy Gaussian distribution to the learned map distribution under the guidance of the test images (see Figure 1).

Compared to previous cumbersome diffusion perception models [75, 74, 61], DDP decouples the image encoder and map decoder. The image encoder runs only once, while the diffusion process is performed only in the lightweight decoder head. With this efficient design, our proposed method can easily be applied to modern perception tasks. Furthermore, unlike previous single-step discriminative models, DDP is capable of performing iterative inference multiple times using the shared parameters and exhibits the following appealing properties: (1) dynamic inference to trade off computation and prediction quality and (2) natural awareness of the prediction uncertainty.

We evaluate DDP on three representative dense prediction tasks, including semantic segmentation, BEV map segmentation, and depth estimation, using six popular datasets (ADE20K [91], Cityscapes [19], nuScenes [7], KITTI [28], NYU-DepthV2 [63], and SUN RGB-D [67]). Our experimental results demonstrate that DDP significantly outperforms existing state-of-the-art methods. Specifically, on ADE20K, DDP achieves 46.1 mIoU with a single sampling step, which is significantly better than UperNet [76] and K-Net [87]. On nuScenes, DDP yields an mIoU of 70.3, which is clearly better than the BEVFusion [47] baseline that achieves an mIoU of 62.7. Furthermore, by increasing the sampling steps, DDP can achieve even higher performance on both ADE20K and nuScenes, reaching mIoU of 47.0 and 70.6, respectively. Moreover, the gains are more versatile for different model architectures as well as model sizes. DDP achieves 83.9 mIoU on Cityscapes with the ConvNeXt-L backbone and produces a leading REL of 0.05 on KITTI with the Swin-L backbone.

Overall, our contributions in this work are three-fold.

- We formulate the dense visual prediction tasks as a general conditional denoising process, with simple yet highly effective designs.
- Our “noise-to-map” generative paradigm offers several appealing properties, such as the ability to perform dynamic inference and uncertain awareness.
- We conduct extensive experiments on three representative tasks with six diverse benchmarks. The results demonstrate that our method, which we refer to as DDP, achieves competitive performance when compared to previous discriminative methods.

2. Related Work

Diffusion Model. Diffusion [32, 64] and score-based generative models [66] have been particularly successful as generative models and achieve impressive results across various modalities, including images [53, 59, 24, 50, 23, 23], video [33, 34], audio [38], and biomedical [2, 70, 62, 20]. Given the notable achievements of diffusion models in these respective domains, leveraging such models to develop generation-based perceptual models would prove to be a highly promising avenue to push the boundaries of perceptual tasks to newer heights.
Dense Prediction. The perception of real-world scenes via pixel-by-pixel classification or regression is commonly formulated as dense prediction tasks, such as semantic segmentation [19, 91], depth estimation [28, 63, 67], and optical flow [26, 28]. Numerous methods have emerged and achieved tremendous progress, and these advances can be roughly divided to: multi-scale feature aggregation [8, 9, 76], high-capacity backbone [78, 89, 54] and powerful decoder head [69, 87, 18, 36]. In this paper, as shown in Figure 1, which differs from previous discriminative-based methods, we explore a generative “noise-to-map” paradigm for general dense prediction tasks.

Diffusion Models for Dense Prediction. With the recent success of diffusion models in generation tasks, there has been a noticeable rise in interest to incorporate them into dense visual prediction tasks. Several pioneering works [75, 1, 74, 13, 61, 11] attempted to apply the diffusion model to visual perception tasks, e.g. image segmentation or depth estimation task. For example, Wolleb et al. [74] explore the diffusion model for medical image segmentation. Pix2Seq-Net [13] applies the bit diffusion model [15] for panoptic segmentation. Our concurrent work DepthGen [61] involves diffusion pipeline to the task of depth estimation. For all the diffusion models listed above, one or two parameter-heavy convolutional U-Nets [57] are adopted, leading to low efficiency, slow convergence, and sub-optimal performance. In this work, as illustrated in Figure 2, we introduce a simple yet effective diffusion framework, which extends the denoising diffusion process into the modern perception pipeline while maintaining accuracy and efficiency.

3. Methodology

3.1. Preliminaries

Dense Prediction. The objective of dense prediction tasks is to predict discrete labels or continuous values, denoted as $y$, for every pixel present in the input image $x \in \mathbb{R}^{3 \times H \times W}$.

Conditional Diffusion Model. The conditional diffusion model, which is an extension of the diffusion model [32, 64, 68], belongs to the category of likelihood-based models inspired by non-equilibrium thermodynamics. The conditional diffusion model assumes a forward noising process by gradually adding noise to the data sample, which is defined as:

$$q(z_t \mid z_0) = \mathcal{N}(z_t; \sqrt{\alpha_t}z_0, (1-\alpha_t)I),$$

which transforms the data sample $z_0$ to a latent noisy sample $z_t$ for $t \in \{0, 1, \ldots, T\}$. The constants $\alpha_t := \prod_{s=0}^{t} \alpha_s = \prod_{s=0}^{t} (1-\beta_s)$ and $\beta_s$ represents the noise schedule [51, 32]. During training, the reverse process model $f_\theta(z_t, x, t)$ is trained to predict $z_0$ from $z_t$ under the guidance of condition $x$ by minimizing the training objective function (i.e., $l_2$ loss). At the inference stage, predicted data sample $z_T$ is reconstructed from a random noise $z_T$ with the model $f_\theta$, conditional input $x$, and a translation rule $[32, 65]$ in a markovian way, i.e., $z_T \rightarrow z_{T-\Delta} \rightarrow \ldots \rightarrow z_0$, which can be formulated as:

$$p_\theta(z_{0:T} \mid x) = p(z_T) \prod_{t=1}^{T} p_\theta(z_{t-1} \mid z_t, x).$$

In this paper, our goal is to solve dense prediction tasks via the conditional diffusion model. In our setting, the data samples are the ground truth map $z_0 = y$, and a neural network $f_\theta$ is trained to predict $z_0$ from random noise $z_t \sim \mathcal{N}(0, I)$ conditioned on the corresponding image $x$.

3.2. Architecture

Since the diffusion model generates samples progressively, it requires multiple runs of the model in the inference stage. Previous methods [75, 61, 74] apply the model $f_\theta$ in multiple steps on the raw image $x$, which significantly increases the computational overhead. To alleviate this issue, we separate the entire model into two parts: image encoder and map decoder, as shown in Figure 2. The image encoder forwards only once to extract the feature map from the input image $x$. Then the map decoder employs it as the condition rather than the raw image $x$, to gradually refine the prediction from the noisy map $y_t$.

Image Encoder. The image encoder receives the raw image $x$ as input and generates multi-scale features at 4 different resolutions. Subsequently, these multi-scale features are fused using the FPN [44] and aggregated by a 1×1 convolution. The produced feature map, with the resolution of $256 \times \frac{5}{4} \times \frac{5}{4}$, is employed as the condition for the map decoder. In contrast to the previous methods [1, 75, 61], DDP is able to work with modern network architectures such as ConvNext [46] and Swin Transformer [45].

Map Decoder. The map decoder $f_\theta$ takes as input the noisy map $y_t$ and the feature map from the image encoder via concatenation and performs a pixel-by-pixel classification or regression. Following the common practice [17, 92, 86] in modern perception pipelines, we simply stack six layers of deformable attention as the map decoder. Compared to previous works [1, 75, 61, 13, 74] that use the parameter-intensive U-Nets, our map decoder is lightweight and compact, allowing efficient reuse of the shared parameters during the multi-step reverse diffusion process.

3.3. Training

During training, we first construct a diffusion process from the ground truth $y$ to the noisy map $y_t$ and then train the model to reverse this process. The training procedure
for DDP is provided in Algorithm 1 (for more details please refer to ??).

**Label Encoding.** Standard diffusion models assume continuous data, which makes them a convenient choice for regression tasks with continuous values (e.g., depth estimation). However, existing studies [13, 15] show that they are unsuitable for discrete labels (e.g., semantic segmentation). Therefore, we explore several encoding strategies for the discrete labels, including: (1) One-hot encoding, which represents categorical labels as binary vectors of 0 and 1; (2) Analog bits encoding [13], which first converts discrete integers into bit strings, and then casts them as real numbers; (3) Class embedding, which uses a learnable embedding layer to project discrete labels into a high-dimensional continuous space, with a sigmoid function for normalization. For all of these strategies, we normalize and scale the range of encoded labels within [−scale, +scale], as shown in Algorithm 1. Notably, the scaling factor scale controls the signal-to-noise ratio (SNR) [13, 12], which is an important hyper-parameter for diffusion models. We compare these strategies in Table 5a and find class embedding work best. More discussions are in Section 4.5.

**Map Corruption.** We add Gaussian noise to corrupt the encoded ground truth, obtaining the noisy map $y_t$. As shown in Equation (1), the intensity of corruption noise is controlled by $\alpha_t$, which adopts the monotonically decreasing schedule for $\alpha_t$ in different time steps $t \in [0, 1]$. Different noise scheduling strategies, including cosine schedule [51] and linear schedule [32], are compared and discussed in Section 4.5. We found that the cosine schedule usually worked best in our benchmark tasks.

**Objective Function.** Standard diffusion models are trained with $l_2$ loss, which is reasonable for dense prediction tasks, but we found that adopting a task-specific loss works better for supervision, e.g., cross-entropy loss for semantic segmentation, sigloss for depth estimation.

3.4. Inference

Given a test image as condition input, the model starts with a random noise map sampled from a Gaussian distribution and gradually refines the prediction, we summarize the inference procedure in Algorithm 2.

**Sampling Rule.** We choose the DDIM update rule [65] for the sampling. In each sampling step $t$, the random noise $y_t$ or the predicted noisy map $y_{t+1}$ from the last step is fused with the conditional feature map, and sent to the map decoder $f_θ$ for map prediction. After getting the predicted result of the current step, we compute the noisy map $y_t$ for the next step using the reparameterization trick. Following [14, 13, 11], we use the asymmetric time intervals (controlled by a hyper-parameter $td$) during the inference stage, and $td = 1$ works best in our method.

**Sampling Drift.** As displayed in Figure 3a, we empirically observe that the model performance improves in a few sampling steps and then declines slightly as the number of steps increases. Similar observations can also be found in [11, 10, 61]. This performance decline can be attributed to the “sampling drift” challenge, which refers to the discrepancy between the distribution of training and sampling data. During training, the model is trained to inverse the ground truth map, while during testing, the model is inferred to remove noise from its “imperfect” prediction, which drifts away from the underlying corrupted distributions. This drift becomes pronounced with smaller time steps $t$, owing to the compounded errors, and is further intensified when a sample deviates more substantially from the distribution of ground truth [22].

To verify our hypothesis, in the last 5k iterations of training, we construct $y_t$ using the model’s prediction rather than the ground truth. The approach transforms the training target to remove the added noise on its own predictions, thereby aligning the data distribution of training and testing. We name this approach “self-aligned denoising.” As revealed in Figure 3a, this approach tends to produce satura-
tion instead of performance degradation. Our findings suggest that incorporating the diffusion process into perception tasks could enhance efficacy compared to image generation (e.g., about 50 DDIM steps for image generation). In other words, the proposed DDP can improve efficiency (e.g., satisfied results in 3 iterative steps) while retaining the benefits of the diffusion model. More discussions can be found in ??.

Multiple Inference. By virtue of the multi-step sampling procedure, our method supports dynamic inference, which has the flexibility to trade compute for prediction quality. Besides, it naturally enables the assessment of the reliability and uncertainty of model predictions.

4. Experiment

We first present the appealing properties of our DDP, followed by empirical evaluations of its performance against leading methods on several representative tasks, including semantic segmentation, BEV map segmentation, and monocular depth estimation. Finally, we provide ablation studies on the DDP components. Due to space limitations, more implementation details and experimental results are provided in ?? and ??, respectively.

4.1. Main Properties

We explore and show properties of DDP in Figure 3 using the default setting in Section 4.2. With such a multi-step sampling procedure, we have the flexibility to trade computational cost for prediction quality. Furthermore, the stochastic sampling process allows the computing of pixel-wise uncertainty maps of the prediction.

Dynamic Inference. We evaluate DDP with ConvNext-T and ConvNext-L backbones by increasing their sampling steps from 1 to 10. The results are presented in Figure 3a. It can be seen that the DDP can continuously improve its performance by using more sampling steps. For example, DDP with ConvNext-T shows an increase from 82.33 mIoU (1 step) to 82.60 mIoU (3 steps), and we visualize the inference trajectory in Figure 3b. In comparison to the previous single-step method, our approach boasts the flexibility to balance computational cost against accuracy. This means our method can be adapted to different trade-offs between speed and accuracy under various scenarios without the need to retrain the network.

Uncertainty Awareness. In addition to the performance gains, the proposed DDP can naturally provide uncertainty estimates. In the multi-step sampling process, we can simply count the pixels where the predicted result of each step differs from the result of the previous step, and finally, we simply normalize this change count map to 0-1 and obtain an uncertainty map. In comparison, DDP is naturally and easily capable of estimating uncertainty, whereas previous methods [48, 30] require complicated modeling such as Bayesian networks.

4.2. Semantic Segmentation

Datasets. We evaluate the proposed DDP using two widely used datasets: ADE20K [91] and Cityscapes [19]. ADE20K is a large-scale scene parsing dataset with over 20,000 images, and Cityscapes is a street scene dataset with high-quality pixel-level annotations for 5,000 images.

Settings. In the training phase, following common practices [73, 16, 78, 72], the crop size is set to 512×512 for ADE20K, and 512×1024 for Cityscapes. We optimize our DDP models using the AdamW [49] optimizer, with an initial learning rate of 6×10−5 and a weight decay of 0.01. All models are trained for 160k iterations and compared fairly.
with previous non-diffusion methods.

**Results on ADE20K.** Table 1 presents the semantic segmentation performance of DDP on ADE20K [91], which shows that our method consistently outperforms many representative methods [76, 21, 87, 5] and the non-diffusion baseline across different backbones. For instance, when using Swin-T [45] as the backbone, our DDP (step 1) yields a promising result of 46.1 mIoU, surpassing the non-diffusion baseline (DDP w/o diffusion process) by 1.2 points (44.9 vs. 44.9). Moreover, our DDP (step 3) can further enhance the performance to 47.0 mIoU, attaining a remarkable gain of 0.9 points by multi-steps of denoising diffusion. With the Swin-L backbone, our DDP (step 3) achieves the best performance of 53.2 mIoU, which is 1.1 points (53.2 vs. 52.1) better than UperNet with comparable FLOPs. These results suggest that our DDP not only achieves a performance gain but also offers more flexibility than previous methods.

**Results on Cityscapes.** We compare our DDP with various representative models on Cityscapes [19] in Table 2, such as Segmenter [69], SETR [89], SegFormer [78], DiversePatch [29], and Mask2Former [17], and so on. As shown, we conduct extensive experiments based on ConvNext [46] and Swin [45] with different model sizes. When using ConvNext-L$^1$ as the backbone, our DDP (step 1) produces a competitive result of 82.95 mIoU, and it can be further boosted to 83.21 mIoU (step 3). This phenomenon was also observed when taking Swin-T as the backbone, and the mIoU increased from 80.96 to 81.24 through additional 2 sampling steps. These experimental results demonstrate the scalability of our methodology, which can be applied to different model structures of arbitrary size. Moreover, once again, the experimental results show that DDP achieves progressive improvements through multi-step denoising diffusion while keeping comparable computational overhead.

**Discussion.** The original intention of DDP is to design a diffusion-based general framework for various dense prediction tasks. Although its segmentation performance is slightly lower than its specialized counterpart Mask2Former [17], it remains highly competitive and has several attractive features. How to design a segmentation-specific diffusion framework to achieve better performance than Mask2Former is left for future research.

### 4.3. BEV Map Segmentation

**Dataset.** We conduct our experiments of BEV map segmentation on the nuScenes [7] dataset. It is a large-scale autonomous driving perception dataset, which includes over 1000 urban road scenes covering different time periods and weather conditions in two cities, Boston and Singapore.

**Settings.** We further verify the DDP framework on the BEV map segmentation task. Specifically, we equip our method with the representative method BEVFusion [47], where we directly replace its segmentation head with the proposed map decoder for the diffusion process. We follow evaluation protocol from [47] and compare the results with state-of-the-art methods [79, 82, 47, 4]. We report the mIoU of 6 background classes, including drivable space (Dri), pedestrian crossing (Ped), walk-way (Wal), stop line (Sto), car-parking area (Car), and lane divider (Div), and use the

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>#Param</th>
<th>FLOPs</th>
<th>mIoU</th>
<th>+MS</th>
</tr>
</thead>
<tbody>
<tr>
<td>UperNet [76]</td>
<td>Swin-T</td>
<td>60M</td>
<td>236G</td>
<td>44.5</td>
<td>45.8</td>
</tr>
<tr>
<td>Region Rebalance [21]</td>
<td>Swin-T</td>
<td>60M</td>
<td>236G</td>
<td>45.0</td>
<td>46.5</td>
</tr>
<tr>
<td>MaskFormer [18]</td>
<td>Swin-T</td>
<td>42M</td>
<td>55G</td>
<td>46.7</td>
<td>48.8</td>
</tr>
<tr>
<td>Mask2Former [17]</td>
<td>Swin-T</td>
<td>47M</td>
<td>74G</td>
<td>47.7</td>
<td>49.6</td>
</tr>
<tr>
<td>K-Net [87]</td>
<td>Swin-T</td>
<td>73M</td>
<td>256G</td>
<td>45.8</td>
<td>46.3</td>
</tr>
<tr>
<td>SenFormer [5]</td>
<td>Swin-T</td>
<td>144M</td>
<td>179G</td>
<td>46.0</td>
<td>46.4</td>
</tr>
<tr>
<td>Non-diffusion Baseline</td>
<td>Swin-T</td>
<td>35M</td>
<td>111G</td>
<td>44.9</td>
<td>46.1</td>
</tr>
<tr>
<td>DDP (step 1)</td>
<td>Swin-T</td>
<td>40M</td>
<td>113G</td>
<td>46.1</td>
<td>47.6</td>
</tr>
<tr>
<td>DDP (step 3)</td>
<td>Swin-T</td>
<td>40M</td>
<td>252G</td>
<td>47.0</td>
<td>47.8</td>
</tr>
<tr>
<td>UperNet [76]</td>
<td>Swin-S</td>
<td>81M</td>
<td>259G</td>
<td>47.6</td>
<td>49.5</td>
</tr>
<tr>
<td>DDP (step 1)</td>
<td>Swin-S</td>
<td>61M</td>
<td>136G</td>
<td>48.4</td>
<td>49.7</td>
</tr>
<tr>
<td>DDP (step 3)</td>
<td>Swin-S</td>
<td>61M</td>
<td>276G</td>
<td>48.7</td>
<td>49.7</td>
</tr>
<tr>
<td>UperNet [76]</td>
<td>Swin-B</td>
<td>121M</td>
<td>297G</td>
<td>48.1</td>
<td>49.7</td>
</tr>
<tr>
<td>DDP (step 1)</td>
<td>Swin-B</td>
<td>99M</td>
<td>173G</td>
<td>49.2</td>
<td>50.8</td>
</tr>
<tr>
<td>DDP (step 3)</td>
<td>Swin-B</td>
<td>99M</td>
<td>312G</td>
<td>49.4</td>
<td>50.8</td>
</tr>
<tr>
<td>UperNet [76]</td>
<td>Swin-L$^1$</td>
<td>234M</td>
<td>411G</td>
<td>52.1</td>
<td>53.5</td>
</tr>
<tr>
<td>DDP (step 1)</td>
<td>Swin-L$^1$</td>
<td>207M</td>
<td>285G</td>
<td>53.1</td>
<td>54.4</td>
</tr>
<tr>
<td>DDP (step 3)</td>
<td>Swin-L$^1$</td>
<td>207M</td>
<td>425G</td>
<td>53.2</td>
<td>54.4</td>
</tr>
</tbody>
</table>

Table 1. Semantic segmentation on ADE20K val set. We report single-scale (SS) and multi-scale (MS) mIoU. The FLOPs are measured with 512×512 inputs. Backbones pre-trained on ImageNet-22K are marked with †.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>#Param</th>
<th>FLOPs</th>
<th>mIoU</th>
<th>+MS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDP (step 1)</td>
<td>Swin-T</td>
<td>39M</td>
<td>885G</td>
<td>80.96</td>
<td>82.25</td>
</tr>
<tr>
<td>DDP (step 3)</td>
<td>Swin-T</td>
<td>39M</td>
<td>1992G</td>
<td>81.24</td>
<td>82.46</td>
</tr>
<tr>
<td>DDP (step 1)</td>
<td>Swin-S</td>
<td>61M</td>
<td>1067G</td>
<td>82.17</td>
<td>83.06</td>
</tr>
<tr>
<td>DDP (step 3)</td>
<td>Swin-S</td>
<td>61M</td>
<td>2174G</td>
<td>82.41</td>
<td>83.21</td>
</tr>
<tr>
<td>DDP (step 1)</td>
<td>Swin-B</td>
<td>99M</td>
<td>1357G</td>
<td>82.37</td>
<td>83.36</td>
</tr>
<tr>
<td>DDP (step 3)</td>
<td>Swin-B</td>
<td>99M</td>
<td>2464G</td>
<td>82.54</td>
<td>83.42</td>
</tr>
<tr>
<td>DDP (step 1)</td>
<td>ConvNext-T</td>
<td>40M</td>
<td>883G</td>
<td>80.33</td>
<td>82.00</td>
</tr>
<tr>
<td>DDP (step 3)</td>
<td>ConvNext-T</td>
<td>40M</td>
<td>1989G</td>
<td>82.60</td>
<td>83.15</td>
</tr>
<tr>
<td>DDP (step 1)</td>
<td>ConvNext-S</td>
<td>62M</td>
<td>1059G</td>
<td>82.37</td>
<td>83.38</td>
</tr>
<tr>
<td>DDP (step 3)</td>
<td>ConvNext-S</td>
<td>62M</td>
<td>2166G</td>
<td>82.69</td>
<td>83.58</td>
</tr>
<tr>
<td>DDP (step 1)</td>
<td>ConvNext-B</td>
<td>100M</td>
<td>1340G</td>
<td>82.59</td>
<td>83.47</td>
</tr>
<tr>
<td>DDP (step 3)</td>
<td>ConvNext-B</td>
<td>100M</td>
<td>2447G</td>
<td>82.78</td>
<td>83.49</td>
</tr>
<tr>
<td>DDP (step 1)</td>
<td>ConvNext-L$^1$</td>
<td>209M</td>
<td>2139G</td>
<td>82.95</td>
<td>83.76</td>
</tr>
<tr>
<td>DDP (step 3)</td>
<td>ConvNext-L$^1$</td>
<td>209M</td>
<td>3245G</td>
<td>83.21</td>
<td>83.92</td>
</tr>
</tbody>
</table>

Table 2. Semantic segmentation on Cityscapes val set. We report single-scale (SS) and multi-scale (MS) mIoU. The FLOPs are measured with 1024×2048 inputs. Backbones pre-trained on ImageNet-22K are marked with †.
We evaluate the depth estimation performance of our approach, over existing state-of-the-art methods. Specifically, in the camera-only scenario, our DDP (step 1) attains a 59.3 mIoU score on the nuScenes validation dataset, which surpasses the previous best method X-Align [4] by 1.3 mIoU (59.3 vs. 58.0). By iteratively refining the output of the model, DDP (step 3) sets a new state-of-the-art record of 59.4 mIoU solely based on camera modality. In the multi-modality setting, we improve the segmentation results of our DDP (step 1) to 70.3 mIoU by combining LiDAR information, significantly higher than the current state-of-the-art methods [47, 4] by at least 4.6 mIoU. Remarkably, this performance can be further enhanced to a maximum of 70.6 mIoU by leveraging the benefits of iterative denoising diffusion. In summary, these results demonstrate that DDP can be easily generalized to other tasks and obtain performance gains, proving the effectiveness and generalization of our approach.

### 4.4. Depth Estimation

#### Datasets

We evaluate the depth estimation performance of DDP on three prominent datasets, namely KITTI [28], NYU-DepthV2 [63], and SUN RGB-D [67]. (1) The KITTI dataset encompasses stereo image pairs and corresponding ground truth depth maps for outdoor scenes captured by a car-mounted camera. Following common practices [25, 41], we use about 26K left-view images for training and 697 images for testing. (2) The NYU dataset contains RGB-Depth images for indoor scenes captured at a resolution of 640×480. Similar to prior research [41], the model is trained on 24K train images and evaluated on the reserved 652 images. (3) The SUN RGB-D dataset is a vast collection of around 10K indoor images. We employ it to evaluate the generalization abilities of our NYU pre-trained models. The results on KITTI are shown in the main paper, while others will be provided in the supplementary material.

#### Settings

We incorporate the DDP model into the codebase developed by [41] for depth estimation experiments. We excluded the discrete label encoding module as the task requires continuous value regression. All experimental settings are the same as [41] for a fair comparison.

#### Metrics

Typically, the evaluation of depth estimation methods employs the following metrics: accuracy under threshold (δ_i < 1.25^i, i = 1, 2, 3), mean absolute relative error (REL), mean squared relative error (SqRel), root mean squared error (RMSE), root mean squared log error (RMSE log), and mean log10 error (log10).

#### Results

Table 4 shows the depth estimation results on the KITTI dataset. We compare the proposed DDP models with state-of-the-art depth estimators. Specifically, we choose DepthFormer [41] and DepthGen [61] as our main competitors, in which DepthFormer is a strong counterpart and achieved leading performance, while DepthGen is a concurrent work of ours and is also a diffusion-based depth estimator. As we can observe, although the performance on this benchmark tends to be saturated, our DDP models still outperform all the competitors with clear margins in most metrics, such as REL, SqRel, and RMSE. For instance, equipped with Swin-L^†, our method achieves a state-of-the-art RMSE log of 0.076 by 3 steps of denoising diffusion. Compared with the concurrent diffusion-based model [61], we find that: (1) DDP outperforms DepthGen with clear margins, particularly in regards to the RMSE log, metric (2.072 vs. 2.985), which can be contributed by the equipped advanced pipeline design (e.g., Swin Transformer vs. U-Net). (2) DDP is more lightweight and efficient compared to DepthGen, as the denoising diffusion process occurs solely on the decoder head, whereas with DepthGen, the process occurs on the entire model.

### 4.5. Ablation Study

We conduct ablation studies on the ADE20K semantic segmentation. All models are trained using our DDP with Swin-T [45] backbone for 160k iterations. Other settings are the same as the settings in Section 4.2.

#### Label Encoding

Since the labels of semantic segmentation are discrete, we need to encode them first. As shown in Table 5a, here we study the effect of three different strategies. For each of them, we search the optimal scaling factor.

#### Signal Scale

As shown in Table 5b, we search for the optimal scaling factor for the class embedding strategy. As can be seen, when we use a larger scaling factor than 0.01, the

<table>
<thead>
<tr>
<th>Method</th>
<th>Modality</th>
<th>Dri</th>
<th>Ped</th>
<th>Wal</th>
<th>Sto</th>
<th>Car</th>
<th>Div</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF [59]</td>
<td>C</td>
<td>74.0</td>
<td>35.3</td>
<td>45.9</td>
<td>27.5</td>
<td>35.9</td>
<td>33.9</td>
<td>42.1</td>
</tr>
<tr>
<td>LSS [52]</td>
<td>C</td>
<td>75.4</td>
<td>38.8</td>
<td>46.3</td>
<td>30.3</td>
<td>39.1</td>
<td>36.5</td>
<td>44.4</td>
</tr>
<tr>
<td>CVT [90]</td>
<td>C</td>
<td>74.3</td>
<td>36.8</td>
<td>39.9</td>
<td>25.8</td>
<td>35.0</td>
<td>29.4</td>
<td>40.2</td>
</tr>
<tr>
<td>M^2BEV [79]</td>
<td>C</td>
<td>77.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>40.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BEVFusion [47]</td>
<td>C</td>
<td>81.7</td>
<td>54.8</td>
<td>58.4</td>
<td>47.4</td>
<td>50.7</td>
<td>46.4</td>
<td>56.6</td>
</tr>
<tr>
<td>X-Align [4]</td>
<td>C</td>
<td>82.4</td>
<td>55.6</td>
<td>59.3</td>
<td>49.6</td>
<td>53.8</td>
<td>47.4</td>
<td>58.0</td>
</tr>
<tr>
<td>DDP (step 1)</td>
<td>C</td>
<td>83.2</td>
<td>58.5</td>
<td>61.6</td>
<td>52.4</td>
<td>51.1</td>
<td>48.9</td>
<td>59.3</td>
</tr>
<tr>
<td>DDP (step 3)</td>
<td>C</td>
<td>83.6</td>
<td>58.3</td>
<td>61.8</td>
<td>52.3</td>
<td>51.4</td>
<td>49.2</td>
<td>59.4</td>
</tr>
</tbody>
</table>

Table 3. BEV map segmentation on nuScenes val set. We report the IoU of 6 background classes and the mean IoU. “C” and “L” denote the camera modality and LiDAR modality, respectively.
Table 4. Depth estimation on the KITTI val set. Backbones pre-trained on ImageNet-22K are marked with †. We report the performance of DDP with 3 diffusion steps. The best and second-best results are bolded or underlined, respectively. ↓ means lower is better, and ↑ means higher is better. * denotes best results of our concurrent work [61].

Table 5. DDP ablation experiments with Swin-T [45] on ADE20K semantic segmentation. We report the performance with 3 sampling steps in (a), (b), (c), and (d). If not specified, the default settings are: the label encoding strategy is class embedding, the scaling factor is 0.01, the noise schedule is cosine, and the map decoder has a depth of 6. Default settings are marked in gray.

**Table 4. Depth estimation on the KITTI val set.** Backbones pre-trained on ImageNet-22K are marked with †. We report the performance of DDP with 3 diffusion steps. The best and second-best results are bolded or underlined, respectively. ↓ means lower is better, and ↑ means higher is better. * denotes best results of our concurrent work [61].

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>(\delta_1)</th>
<th>(\delta_2)</th>
<th>(\delta_3)</th>
<th>REL ↓</th>
<th>SaRel ↓</th>
<th>RMSE ↓</th>
<th>RMSE log ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>BinSformer [42]</td>
<td>Swin-L↓</td>
<td>0.974</td>
<td>0.997</td>
<td>0.999</td>
<td>0.052</td>
<td>0.151</td>
<td>2.098</td>
<td>0.079</td>
</tr>
<tr>
<td>DepthFormer [41]</td>
<td>ResNet-50 + Swin-L↑</td>
<td>0.975</td>
<td>0.997</td>
<td>0.999</td>
<td>0.052</td>
<td>0.151</td>
<td>2.098</td>
<td>0.079</td>
</tr>
<tr>
<td>AdaBins [3]</td>
<td>EfficientNet-B5 + Mini-VIT</td>
<td>0.964</td>
<td>0.995</td>
<td>0.999</td>
<td>0.058</td>
<td>0.190</td>
<td>2.360</td>
<td>0.088</td>
</tr>
<tr>
<td>DepthFormer [41]</td>
<td>ResNet-50 + Swin-T</td>
<td>0.966</td>
<td>0.995</td>
<td>0.999</td>
<td>0.056</td>
<td>0.177</td>
<td>2.252</td>
<td>0.086</td>
</tr>
<tr>
<td>TransDepth [80]</td>
<td>ResNet-50 + Vit-B</td>
<td>0.956</td>
<td>0.994</td>
<td>0.999</td>
<td>0.064</td>
<td>0.252</td>
<td>2.755</td>
<td>0.098</td>
</tr>
<tr>
<td>DPT [54]</td>
<td>ResNet-50 + Vit-B</td>
<td>0.959</td>
<td>0.995</td>
<td>0.999</td>
<td>0.062</td>
<td>-</td>
<td>2.573</td>
<td>0.092</td>
</tr>
<tr>
<td>BTS [59]</td>
<td>DenseNet-161</td>
<td>0.956</td>
<td>0.993</td>
<td>0.998</td>
<td>0.059</td>
<td>0.245</td>
<td>2.756</td>
<td>0.096</td>
</tr>
<tr>
<td>VNL [83]</td>
<td>ResNet-101</td>
<td>0.938</td>
<td>0.990</td>
<td>0.998</td>
<td>0.072</td>
<td>-</td>
<td>3.258</td>
<td>0.117</td>
</tr>
<tr>
<td>VNL [83]</td>
<td>ResNet-101</td>
<td>0.932</td>
<td>0.984</td>
<td>0.994</td>
<td>0.072</td>
<td>-</td>
<td>3.258</td>
<td>0.117</td>
</tr>
<tr>
<td>DORN [27]</td>
<td>ResNet-101</td>
<td>0.932</td>
<td>0.984</td>
<td>0.994</td>
<td>0.072</td>
<td>-</td>
<td>3.258</td>
<td>0.117</td>
</tr>
<tr>
<td>58.4</td>
<td>0.999</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.056 0.177 2.252 0.086</td>
<td>0.975 0.997 0.999</td>
<td>0.052 0.151</td>
<td>2.098 0.079</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table 5. DDP ablation experiments** with Swin-T [45] on ADE20K semantic segmentation. We report the performance with 3 sampling steps in (a), (b), (c), and (d). If not specified, the default settings are: the label encoding strategy is class embedding, the scaling factor is 0.01, the noise schedule is cosine, and the map decoder has a depth of 6. Default settings are marked in gray.

**Performance degraded significantly. This is because using a larger scaling factor, more easy cases are reserved with the same time step \(t\). In addition, we found the best scaling factor (i.e., 0.01) for class embedding is typically smaller than analog bits [13] and one-hot (i.e., 0.1).

**Noise Schedule.** As shown in Table 5c, we compare the effectiveness of the cosine schedule [51] and linear schedule [32] in DDP for semantic segmentation, and find that the model using the cosine schedule achieves notably better performance (47.0 vs. 45.1). This is attributed to the cosine schedule’s mechanism of simulating the realistic scenario of gradually weakening signal influence, which prompts the model to learn stronger denoising capabilities, in contrast to the simple linear schedule.

**Decoder Depth.** We study the effect of decoder depth in Table 5d and observe that the map decoder requires a suitable depth. Initially, the model accuracy improves as the depth increases, but eventually decreases. Therefore, we finally adopted a map decoder with 6 blocks, which only has 8.4M parameters. Overall, the map decoder is lightweight and efficient, compared with representative methods K-Net [87] (41.5M) and UperNet [76] (31.5M).

**Accuracy vs. Efficiency.** We show the dynamic trade-off of DDP between accuracy and efficiency in Table 5e. Compared with the representative discriminative method K-Net [87], DDP yields a better mIoU when using only one sampling step, with fewer FLOPs and higher FPS. When adopting three sampling steps, the performance is further boosted to 47.0 mIoU, while maintaining comparable FLOPs and FPS. These results show that DDP can iteratively infer multiple times with reasonable time cost.

5. Conclusion

This paper introduced DDP, a simple, efficient, yet powerful framework for dense visual predictions based on conditional diffusion. It extends the denoising diffusion process into modern perception pipelines, without requiring architectural customization or task-specific design. We demonstrate DDP’s effectiveness through state-of-the-art or competitive performance on three representative tasks and six diverse benchmarks. Moreover, it additionally exhibits multiple inference and uncertainty awareness, which contrasts with previous single-step discriminative methods. These results indicate that DDP can serve as an important baseline.
for future research in dense prediction tasks. One potential drawback of DDP is its non-negligible additional computational cost for multi-step inference. Besides, while DDP has demonstrated excellent improvement on several benchmark datasets for dense visual prediction tasks, further research is necessary to determine its efficacy in other domains.

Acknowledgement. This project was supported in part by the National Key R&D Program of China No.2022ZD0161000, General Research Fund of Hong Kong No.17200622 and the Fundamental Research Funds for the Central Universities XJ2023000701 and XJ2023004101.

References


[19] Marius Corde, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 1, 2, 3, 5, 6


