Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

DepthHist: Depth Estimation Using Differentiable Histogram with SliceStitch Augmentation

Mohammed Chaouki ZIARA

chaouki.ziara@univ-sba.dz

Université Djillali Liabes

Mohammed ELBAHRI

Université Djillali Liabes

Nasreddine TALEB

Université Djillali Liabes

Kidiyo KPALMA

Institut National des Sciences Appliquées de Rennes

Research Article

Keywords: Depth Estimation, Loss Function, Differentiable Histogram, Image Augmentation

Posted Date: May 21st, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9670481/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

DepthHist: Depth Estimation Using Differentiable Histogram with SliceStitch Augmentation

Mohammed Chaouki Ziara[1*], Mohamed Elbahri[2], Nasreddine Taleb[1], Kidiyo Kpalma[3]

[1*]Department of Electronics, RCAM Laboratory, Djillali Liabes University, Sidi Bel Abbes, 22000, Algeria.
[2]Department of Computer Science, RCAM Laboratory, Djillali Liabes University, Sidi Bel Abbes, 22000, Algeria.
[3]IETR-UMR 6164, University of Rennes, INSA Rennes, CNRS, Rennes, F-35000, France.

*Corresponding author(s). E-mail(s): chaouki.ziara@univ-sba.dz;
Contributing authors: mohamed.elbahri@univ-sba.dz;
ne_taleb@univ-sba.dz; Kidiyo.Kpalma@insa-rennes.fr;

Abstract

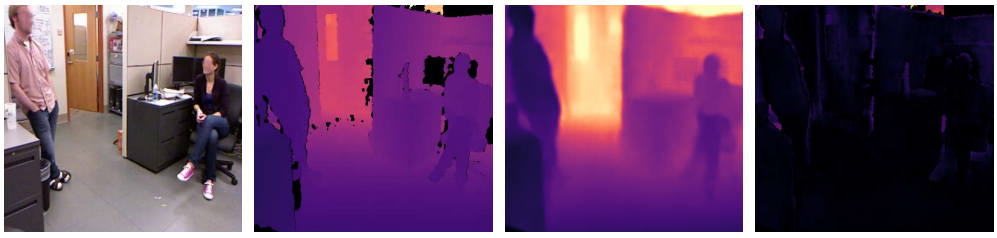
Recent monocular depth estimation methods frequently rely on discretizing the depth range and predicting probabilities over learned bins. Such formulations implicitly treat depth as a classification-regression problem and recover the final estimate by combining per-bin probabilities with their associated depth values. In contrast, we introduce *DepthHist*, a soft-binning framework that constructs a differentiable histogram directly from intermediate feature maps. Our approach predicts per-channel bin centers and scales through a channel-wise attention module. These parameters define a differentiable kernel that measures the similarity between feature responses and their associated centers and scales, yielding continuous soft histogram maps that preserve the structure of the feature space. Depth is then obtained by center-weighted aggregation of these similarity fields, thereby avoiding the discretization artifacts and probability–range coupling inherent in prior binning-based methods. To guide the learning of the soft histogram representation, we propose two complementary training objectives. A Chamfer-based alignment encourages the predicted bin centers to match the empirical distribution of ground-truth depth values, while a histogram-alignment loss enforces consistency between the predicted soft histogram and one computed from ground truth using the same kernel. We further introduce *SliceStitch*, a simple spatial mixing augmentation that improves robustness to abrupt scene transitions.

Extensive experiments across the KITTI, NYUv2, and SUNRGB-D benchmarks demonstrate that *DepthHist* achieves superior accuracy and efficiency relative to recent state-of-the-art methods, while requiring fewer parameters. Code will be released at https://github.com/Chaouki-AI/DepthHist.

Keywords: Depth Estimation, Loss Function, Differentiable Histogram, Image Augmentation

1 Introduction

Monocular Depth Estimation (MDE) predicts scene depth from a single image for applications such as robotics and 3D reconstruction [1, 2]. Traditionally framed as a regression problem, these methods often show limited performance due to the highly non-linear and heavy-tailed nature of depth distributions, the difficulty of optimizing pixel-wise regression losses in the presence of sharp depth discontinuities, and the tendency of regression models to produce overly smooth predictions in ambiguous or textureless regions [3]. To overcome this limitation, researchers have reformulated MDE as a classification problem by discretizing depth into bins [4, 5]. This improves quantitative results but can also introduce visual artifacts such as discontinuities. Building on this idea, hybrid regression–classification approaches like AdaBins tackle extreme depth variations by dynamically predicting bin centers and combining probabilistic distributions with regression-refined values [6]. This architecture, which inspired many subsequent models [7–11], has significantly advanced MDE performance. Nevertheless, these so-called "histogram-inspired" methods treat bin centers as learned depth candidates rather than modeling true statistical distributions. As a result, they fail to capture structural characteristics such as skewness or modality that genuine histograms preserve.



RGB Image | Ground Truth | Predicted Depth | Error Map

Fig. 1: Illustration of the depth and error maps from ***DepthHist***. From left to right: RGB image, Ground truth depth map, Predicted depth map, and Error map (dark indicates low prediction error, while brighter regions highlight areas of higher discrepancy).

Loss functions grounded in histograms remain largely unexplored in MDE, primarily because standard histogram operations are non-differentiable and incompatible with gradient-based optimization [12]. A recent workaround [13] approximates histograms with activation functions, yet as shown in Figure 2, these approximations (orange) lack the structural consistency and statistical fidelity of true distributions (pink).

In this work, we address these limitations by integrating histogram representations into the MDE pipeline through architectural and loss-function contributions, complemented by a targeted data augmentation strategy. Our main contributions are:

- **Differentiable Histogram Layer**: A novel layer that learns bin centers and scales, then applies a soft binning operation to decode the feature map into a histogram-based representation. Unlike earlier approaches that simply treat bin centers as classification labels, our method interprets the histogram itself as a learnable, fully differentiable structure that remains directly connected to the underlying data distribution.
- **Histogram-Based Loss Function**: We introduce a loss that compares predicted histograms with target histograms built from ground-truth depth and learned bin parameters. This loss not only measures alignment but also penalizes mismatches in the overall distribution, encouraging predictions that are statistically and structurally consistent.

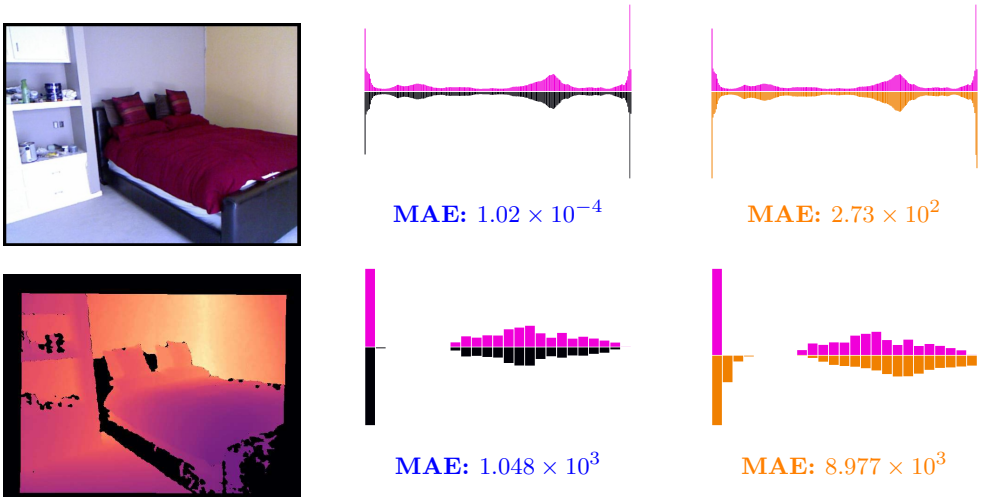


Fig. 2: Comparison of histogram generation methods for RGB images (top row) and depth maps (bottom row). The first column shows the input image and ground truth depth map. The second column compares soft binning histograms (in dark) against ground truth (in pink). The third column presents results using the activation-based method of Avi-Haron *et al.* [13] (in orange). The error between estimated and ground truth histograms is indicated as the mean absolute error (MAE, ↓). (Zoom in to better see histogram differences.)

- ***SliceStitch* Augmentation**: We propose a simple yet effective augmentation technique that merges two randomly selected image–depth pairs along either a vertical or horizontal axis by forcing the model to handle abrupt structural changes. This method reduces reliance on global context and improves its ability to make reliable local depth predictions.

2 Related works

Monocular Depth Estimation (MDE) predicts scene depth from a single image for applications such as largely rely on discretization: the depth range is split into adaptive intervals whose centers are learned or predicted. Bhat *et al.* [6] proposed a design based on Transformers [14] that predicts such adaptive bin centers. Their pipeline divides the feature map into 16×16 patches, uses transformer layers to capture long-range relations between patches, and maps the resulting embeddings with an MLP to a set of classes interpreted as bin centers. This lets the model adapt the bin distribution to scene context and recover continuous depth via weighted combinations of bin centers and decoded features. Following this direction, Zhenyu *et al.* [7] and Zhu *et al.* [11] enriched the idea with extra transformer modules and a dual-MLP design that yields two complementary bin vectors: one that refines features (via a scalar product with the input) and another that predicts the adaptive bin centers. Agarwal *et al.* [9] take a related route by predicting bin centers from pixel-level queries at the coarsest resolution. These global formulations work well in structured scenes but tend to break down in heterogeneous environments: a single global set of bins cannot always account for localized depth changes, especially when well-defined foreground objects stand against ambiguous or low-contrast backgrounds [8]. To address this limitation, recent works have moved toward local binning. Bhat *et al.* [8] split images into subregions and learn bin centers independently for each part, which helps the model adapt to local structures such as occlusion boundaries or weakly textured areas. Based on the idea of localized binning Shao *et al.* [10] built a GRU-driven iterative refinement. Regional bin centers are recalibrated across several stages, and the cyclic updates progressively sharpen the predicted depth maps. it is important to note, however, that the "bins" used in these MDE methods are not the same as classical histogram bins. In a histogram, a bin denotes the frequency of values in a given interval and the collection of bins describes a distribution. The discretization used in depth methods functions more as an adaptive regression scaffold than as a true distributional descriptor. This distinction motivates our work: we aim to reintroduce the histogram concept into MDE by estimating histograms explicitly, both as part of the loss and as histogram maps that feed into the final depth prediction. Although prior histogram works were developed for tasks beyond depth estimation, the following approaches nevertheless provide useful tools and insights. Wang *et al.* [15] propose a trainable histogram layer with learnable centers and widths; the layer summarizes class-confidence distributions across the image and can guide prediction refinement. Peeples *et al.* [16] incorporate localized histogram computations inside CNN backbones, using RBFs [17] to approximate local feature distributions and produce spatially aware texture descriptors. Another complementary work focuses on histogram-based supervision: Avi-Aharon *et al.* [13]

introduce differentiable histogram losses for image-to-image translation, using kernel density estimation to form smooth histogram representations and comparing them with mutual information and distance measures. Together, these works show how distributional information can be modeled and exploited, and so our contribution is to adapt and integrate these histogram ideas specifically for monocular depth estimation.

Beyond architectural choices, data augmentation remains central to improving MDE robustness. Standard transforms (rotations, flips, cropping and color jitter) are still widely used [18]. Shao *et al.* proposed CutFlip, which swaps vertical halves of an image to reduce dependence on vertical priors and improve generalization [19]. Still, many depth errors stem from abrupt depth transitions at object boundaries or other structural discontinuities. To confront this issue, we introduce an augmentation strategy that mixes inputs and outputs either horizontally or vertically across scenes; this reduces over-reliance on local context and makes the model more robust to sharp changes in depth.

3 Methodology

In this section, we go more deeply into the main components of the *DepthHist* model. First of all, we introduce the layer dedicated to creating the histogram representations, and then we analyze in detail the loss functions and the augmentation technique used, along with their impact on model optimization.

Model Overview: *DepthHist*, shown in Figure 3, is implemented based on encoder–decoder architecture. The encoder is a Swin Transformer [20] that replaces global self-attention with shifted window-based attention [21]. The input is partitioned into local windows where attention is computed, and shifting these windows across layers enables cross-window interactions. Patch merging produces pyramidal feature maps [22], enabling the encoder to represent spatial relations at multiple scales while remaining computationally efficient for high-resolution inputs. The decoder follows a U-Net–inspired cascade of convolutional blocks [23]. Each block merges the upsampled output from the previous stage with the corresponding encoder features via skip connections. Upsampling is implemented with transposed convolutions [24]; merged features are normalized, passed through a non-linear activation, and connected with internal residual paths to improve gradient flow and preserve structural detail [25]. The decoder thus reconstructs spatially precise feature maps while maintaining scene-level context [26]. The pivotal element of the model is the *HistLayer*, a module that integrates three information sources: the RGB input, the reduced-resolution feature map from the encoder bottleneck, and the upsampled feature map from the decoder. It converts these inputs into a histogram representation via a soft-binning operation parameterized by the predicted centers and scales. The final dense depth estimate is produced by the element-wise product of this histogram with the denormalized centers, summed across bins.

3.1 *HistLayer* Overview

The *HistLayer* block receives three feature maps as input: the raw RGB input image, a feature map from the encoder backbone, and a corresponding decoded feature map

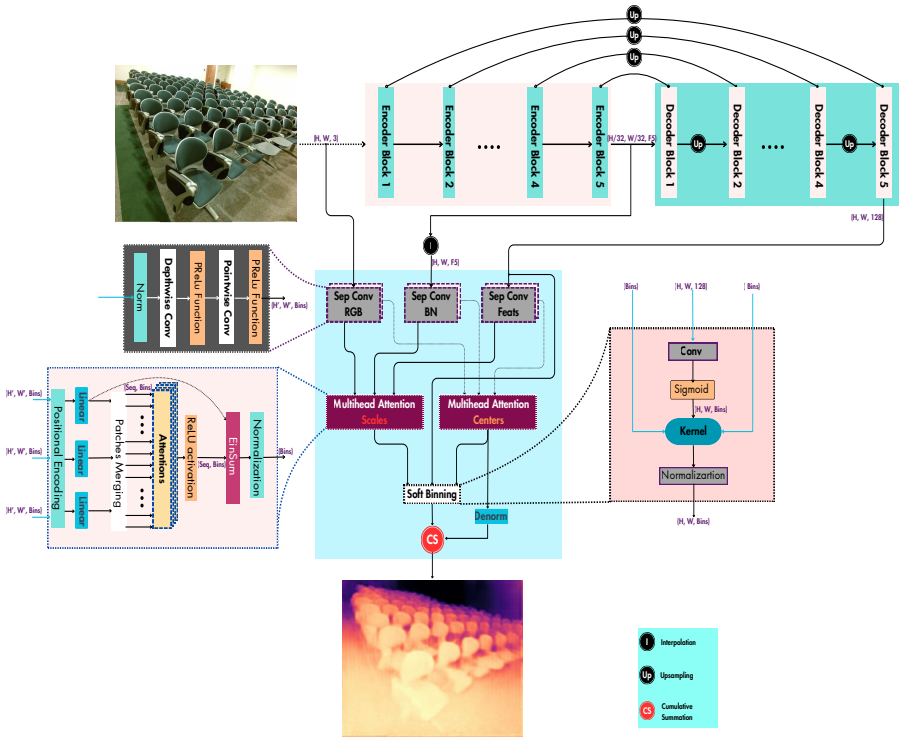


Fig. 3: **Architecture of *DepthHist***: The RGB image is processed through a series of encoder-decoder blocks, producing two outputs: a decoded feature map with a shape of $(H, W, 128)$ and a bottleneck representation with a shape of $(\frac{H}{32}, \frac{W}{32}, \text{Feats})$. These outputs, along with the RGB image, are fed into *HistLayer* to generate a histogram representation, which is then used to produce the final depth map.

from the decoder. Each input is processed through a dual-path separable convolution block, implemented using depthwise separable convolutions to efficiently capture spatial and cross-channel interactions [27]. Inspired by the dual-path design [28], each block contains two parallel branches to enable both feature reuse and novel feature extraction, resulting in two output tensors per input for a total of six tensors, which are split into two groups of three tensors: one group is used to predict histogram bin centers, and the other to predict bin scales. Each group is then passed through its own attention pipeline. Within each pipeline, positional encodings with linear projection are first added to preserve spatial context. The tensors are then partitioned into patches, merged into tokens, and processed by multi-head attention mechanism to model long-range dependencies. This is followed by an element-wise gating, and an aggregation operation for adaptive feature reweighting with a final normalization layer is applied to stabilize training. The result is one representative vector from each pipeline, referred to as the centers and scales vectors. These vectors are then used to construct a histogram representation through a soft binning process.

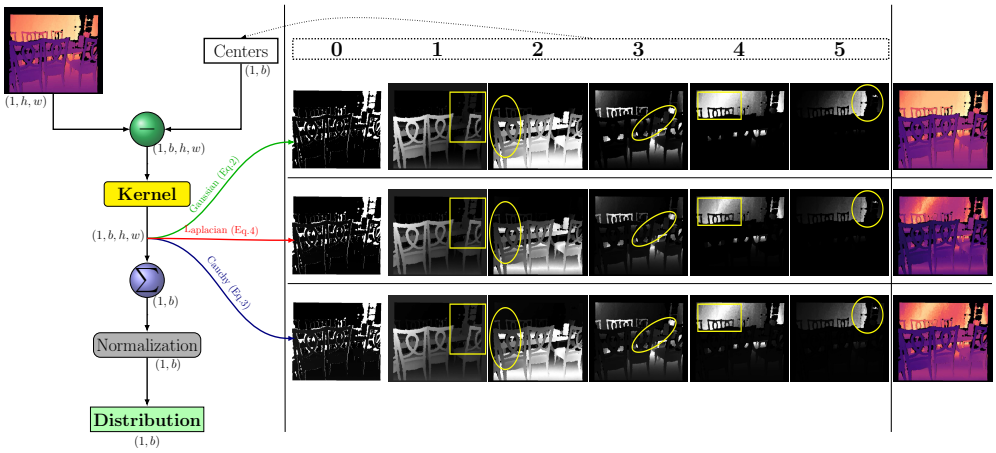


Fig. 4: **Architecture of *Soft Binning* and Kernel Impact:** The left panel shows the soft binning process applied to a single-channel depth map, where centers $C_i = 0, \ldots, 5$ with dispersion 0.4 to generate depth response maps (visualized as images in the middle) using the three kernels. The right panel presents the reconstructed depth maps from cumulative summation (Fig. 5 - middle), with reconstruction quality reflected in SSIM scores of 0.85 (Gaussian), 0.72 (Laplacian), and 0.73 (Cauchy) against the ground truth [29].

Soft Binning Formulation: The goal is to build a frequency distribution (histogram), which can then be incorporated in deep learning networks both as an architectural element as well as in the loss function. For that, an integrable, non-negative, real-valued, and symmetric kernel must be used. This kernel ensures differentiable projection of the input samples onto bins so that it can be incorporated in deep learning pipelines seamlessly.

Let X be a feature map of size (f, h, w), where f is the number of channels ($f = 1$ in Fig. 4), and h, w are the spatial dimensions. C is vector of length b, with entries representing the bin centers of the histogram. The first step is to compute the distance between each feature value and the bin centers:

$$D_{f,i,h,w} = X_{f,h,w} - C_i, \quad \forall i \in \{1, 2, \ldots, b\} \tag{1}$$

The contribution of each feature value to the ith bin is then quantified using a kernel function. A common choice is the Gaussian kernel:

$$S_{f,i,h,w} = \exp\left(-\frac{D^2_{f,i,h,w}}{\sigma^2}\right) \tag{2}$$

where σ controls the dispersion around the bin center. Small differences ($D_{f,i,h,w} \approx 0$) maximize contributions, while larger ones decay exponentially.

An alternative is the Cauchy kernel:

$$S_{f,i,h,w} = \frac{1}{1 + \left(\frac{D_{f,i,h,w}}{\sigma}\right)^2} \tag{3}$$

which declines polynomially, allowing distant values to still contribute meaningfully. This slower decay also improves robustness to outliers and helps capture long-range dependencies. For sharper locality, one may employ the Laplacian kernel:

$$S_{f,i,h,w} = \exp\left(-\left|\frac{D_{f,i,h,w}}{\sigma}\right|\right) \tag{4}$$

Among these kernels, the Laplacian decays most rapidly (absolute exponential), yielding highly localized contributions. The Gaussian exhibits a smoother exponential fall-off, while the Cauchy decays more slowly, supporting long-range influence.
In order to standardize the contributions, each $S_{f,i,h,w}$ is divided by the total across the bin dimension:

$$\hat{H}_{f,i,h,w} = \frac{S_{f,i,h,w}}{\sum_{i=1}^{b} S_{f,i,h,w}} \tag{5}$$

Aggregating across feature and spatial dimensions gives the histogram distribution:

$$H_i = \sum_{f,h,w} \hat{H}_{f,i,h,w} \tag{6}$$

Using the normalized contributions $\hat{H}_{f,i,h,w}$ and the bin centers C_i, we can reconstruct an approximation of the original feature map:

$$\hat{X}_{f,h,w} = \sum_{i=1}^{b} \hat{H}_{f,i,h,w} C_i \tag{7}$$

The reconstruction quality depends on the chosen kernel, the bin centers C_i, and their dispersions σ. Proper calibration ensures that $\hat{H}_{f,i,h,w}$ preserves the original feature distribution, while poorly defined parameters introduce bias and distortions. Fig. 4 illustrates how kernel choice affects contribution maps and reconstructed features in the context of depth estimation.

The formulation described so far relies on fixed bin centers and dispersions. While this design is interpretable and differentiable, it is inadequate for the diverse, non-uniform statistics of natural scenes, particularly in monocular depth estimation (MDE). To overcome this problem, we introduce a learnable extension, the *HistLayer*, which predicts histogram parameters directly from the input. Specifically, a convolutional projection with sigmoid activation generates the feature map, while attention mechanisms applied to multi-scale features predict the centers and dispersions. The optimization process enforces learning appropriate feature maps, centers, and dispersions, which are then used to construct histograms. Finally, these histogram maps,

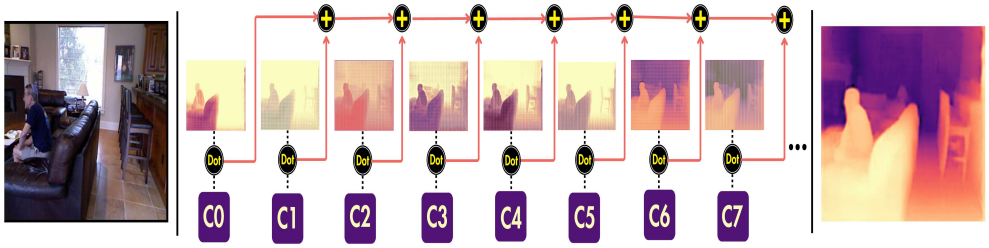


Fig. 5: Illustration of soft binning and reconstruction. Left: input RGB image. Middle: feature map generated via soft binning (top) and the corresponding bin centers (bottom). Right: reconstructed depth map obtained through the cumulative sum of the element-wise multiplication between contribution maps and the scaled bin centers. The scaled centers are derived by denormalizing the attention-predicted centers (originally in $[0, 1]$) using the maximum dataset depth.

together with scaled centers, are combined via cumulative summation to predict the depth map. By reframing MDE as a histogram alignment problem, we move beyond scalar regression, classification or even hybrid approaches and instead we model depth as a structured probabilistic distribution over learned histogram bins.

3.2 Loss Functions

To ensure stable learning, we define a composite loss with three complementary terms:

$$\mathcal{L}_{total} = \alpha \mathcal{L}_{SILoss} + \beta \mathcal{L}_{Chamfer} + \gamma \mathcal{L}_{2DLoss} \tag{8}$$

The scale-invariant loss ($\mathcal{L}_{SILoss}$), introduced by Eigen *et al.* [30], accounts for logarithmic scale differences between predicted and ground-truth depths and is widely used in works such as BTS [31] and AdaBins [6]. The Chamfer loss ($\mathcal{L}_{Chamfer}$) aligns denormalized bin centers with ground-truth depth values by minimizing pairwise distances. Lastly, the 2D histogram loss ($\mathcal{L}_{2DLoss}$) captures local spatial patterns and enforces alignment between predicted and ground-truth depth distributions.

3.2.1 Scale-invariant Loss ($\mathcal{L}_{SILoss}$)

$$\mathcal{L}_{SILoss} = \sqrt{\frac{1}{T}\sum_{i=1}^{T} g_i^2 - \frac{1}{T^2}\left(\sum_{i=1}^{T} g_i\right)^2}, \text{where} \quad g_i = \log(\hat{d}_i) - \log(d_i) \tag{9}$$

Here d_i is the ground-truth depth, $\hat{d}_i$ the prediction, and T the number of valid pixels (depth $> 10^{-3}$).

3.2.2 Chamfer Loss ($\mathcal{L}_{Chamfer}$)

We use the bi-directional Chamfer loss [32] to regularize bin centers C_i against ground-truth depths d_j:

$$\mathcal{L}_{Chamfer} = Chamfer(d_j, C_i) + Chamfer(C_i, d_j) \tag{10}$$

with the Chamfer distance defined as

$$Chamfer(S_1, S_2) = \frac{1}{|S_1|} \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2 + \frac{1}{|S_2|} \sum_{y \in S_2} \min_{x \in S_1} \|y - x\|_2. \tag{11}$$

3.2.3 2D histogram loss ($\mathcal{L}_{2DLoss}$)

As outlined in Section 3.1, the *HistLayer* builds histogram-like distributions from predicted bin centers and their dispersions. The scale-invariant loss $\mathcal{L}_{SILoss}$ (Eq. 9) still supervises the final depth map, and the Chamfer loss $\mathcal{L}_{Chamfer}$ (Eq. 10) pushes the bin centers closer to the global depth distribution. Both are useful, but notice that they operate either at a scalar level or across the whole distribution. They don't really touch the inner structure of the histogram itself. Since the *HistLayer* is supposed to model depth explicitly as a distribution over learned prototypes, we found it necessary to guide this intermediate representation more directly. That is why we introduce the 2D histogram loss.

Because ground-truth histograms do not exist, we construct a surrogate. We apply soft binning to the ground-truth depth map d_{gt}, using the predicted centers C_i and dispersions σ_i:

$$S(i, h, w) = softbinning(d_{\text{gt}}, C_i, \sigma_i). \tag{12}$$

The task is then simple in spirit: make the predicted histogram $\hat{S}(i, h, w)$ aligned with this surrogate. To do so, we rely on two terms: a magnitude loss with directional similarity loss and combined as:

$$\mathcal{L}_{2DLoss} = \mathcal{L}_{\text{mag}} + \mathcal{L}_{\text{ort}}. \tag{13}$$

Rather than comparing raw values, we focus on spatial gradients which capture local structure and they are less sensitive to absolute scale, which is exactly what we want: geometry over raw numbers. Concretely, we compute horizontal and vertical gradients:

$$\begin{aligned} \nabla_{S_w} &= \lim_{\Delta w \to 0} \frac{S(i, h, w + \Delta w) - S(i, h, w)}{\Delta w}, \\ \nabla_{S_h} &= \lim_{\Delta h \to 0} \frac{S(i, h + \Delta h, w) - S(i, h, w)}{\Delta h}. \end{aligned} \tag{14}$$

The same is done for $\hat{S}$. These are then combined into gradient magnitudes:

$$\Psi_{S_i} = \sqrt{\nabla^2_{S_w} + \nabla^2_{S_h}}, \tag{15}$$

We measure the difference between Ψ_{S_i} and $\Psi_{\hat{S}_i}$ using Mean Absolute Percentage Error (MAPE). The usual version is asymmetric and unstable in some cases, so we compute it both ways and sum them:

$$\mathcal{L}_{mag} = \mathcal{M}_{APE}(\Psi_{S_i}, \Psi_{\hat{S}_i}) + \mathcal{M}_{APE}(\Psi_{\hat{S}_i}, \Psi_{S_i}). \tag{16}$$

The second term checks orientation. We use cosine similarity between gradient vectors. For instance, in the w-direction (and similarly for Sim_h):

$$Sim_w = \frac{\nabla_{S_w} \cdot \nabla_{\hat{S}_w}}{2 \cdot \|\nabla_{S_w}\| \cdot \|\nabla_{\hat{S}_w}\|}, \tag{17}$$

These similarities (Sim_h, Sim_w) are averaged, normalized, and inverted to form the orientation loss:

$$\mathcal{L}_{ort} = 1 - \frac{(Sim_h + Sim_w) + 1}{2}. \tag{18}$$

3.3 Augmentation using *SliceStitch*

Shao *et al.* [19] recently pointed out a persistent problem in monocular depth estimation: many models lean too heavily on vertical position as a shortcut. In other words, pixels near the bottom of the frame are often treated as "close," while those near the top are treated as "far." This happens because most training data come from scenes laid out on flat surfaces (roads, floors), so the model learns an almost default gradient: near at the bottom, far at the top. As a result, depth predictions may depend more on spatial location than on actual scene content.

Algorithm 1 SliceStitch Augmentation

Require: Two RGB images $I^A, I^B \in \mathbb{R}^{H \times W \times 3}$
Require: Two maps $D^A, D^B \in \mathbb{R}^{H \times W}$
Ensure: Augmented image I', augmented map D'
Sample $r \sim \mathcal{U}(0.3, 0.7)$
Choose $axis \sim$ RandomChoice($\{$HORIZONTAL, VERTICAL$\}$)
if $axis$ is horizontal **then**
    $cut \Leftarrow \lfloor r \cdot H \rfloor$
    $I' \Leftarrow$ concat top of I^A with bottom of I^B
    $D' \Leftarrow$ concat top of D^A with bottom of D^B
else
    $cut \Leftarrow \lfloor r \cdot W \rfloor$
    $I' \Leftarrow$ concat left of I^A with right of I^B
    $D' \Leftarrow$ concat left of D^A with right of D^B
end if
return I', D'

Our *SliceStitch* augmentation builds on this observation but pushes it further. Instead of slicing a single image, as in Shao *et al.*, we create new samples by stitching together parts of two different images. The process is simple: (i) pick two RGB–depth pairs at random, (ii) select a slicing axis (horizontal or vertical), and (iii) merge the complementary pieces into a new image–depth pair. The procedure is summarized in Algorithm 1, and examples are shown in Figure 6.



Fig. 6: Visual illustration of the proposed *SliceStitch* augmentation technique. The top row in each block shows RGB image pairs before and after the augmentation, with yellow dashed boxes highlighting the stitched regions from two different scenes. The bottom row presents their corresponding depth maps, showing the resulting discontinuities at the stitched boundaries

The intuition is straightforward. A horizontal cut stitches the top of one scene with the bottom of another, deliberately breaking the "near–far" vertical gradient. A vertical cut, by contrast, blends left and right halves, disrupting cues such as road edges, building alignments, or vanishing lines. In both cases, the well orderd global layout is violated. What remains are abrupt seams and mismatched structures, hence forcing the model to rely less on positional bias and more on semantic and object-level cues. By introducing such local discontinuities, *SliceStitch* discourages shortcut learning and encourages context-aware reasoning. In practice, this makes the model more robust and better able to generalize to unfamiliar or heterogeneous environments.

4 Experiment

We have carried out an extensive experimental study on reference datasets for monocular depth estimation, covering both indoor and outdoor scenes. This section begins with a concise presentation of the datasets used and the evaluation metrics, followed by a detailed description of the technical implementation, and finally the experimental obtained results. We then present a quantitative comparison with state-of-the-art methods.

4.1 Datasets

The KITTI dataset comprises stereo image pairs and 3D laser scans of outdoor scenes, captured using a LiDAR sensor [33] mounted on a moving vehicle [34]. The RGB images have a resolution of approximately 1241×376 pixels, while the corresponding depth maps are extremely sparse, with a maximum measurable depth of up to 80 meters. *DepthHist* is trained on a random subset of 26,000 images, with random cropping to 704×352 pixels. For evaluation, we use 697 images from the test split defined by Eigen *et al.* [30], apply the cropping scheme of Garg *et al.* [35], and perform bilinear resampling to align predictions with the ground-truth resolution. The final prediction is obtained by averaging each result with that of its horizontally flipped counterpart.

The NYU Depth v2 dataset provides a comprehensive collection of indoor RGB-D images, captured at a resolution of 640×480 pixels using the Microsoft Kinect sensor [36]. The dataset consists of 120,000 training samples and 654 test images [37]. In our experiments, we use a subset of 50,000 training samples and the full test set, considering a maximum depth range of 10 meters. Following the central cropping strategy introduced by Eigen *et al.* [30], evaluation is restricted to the central region of each test image. For inference, we compute the final prediction as the average of the model's output on the original image and its horizontally flipped version—a standard augmentation technique known to improve robustness and consistency in depth estimation tasks.

4.2 Evaluation Metrics

We evaluate our method using six standard metrics widely adopted in previous works:

- **Root Mean Squared Error (RMSE)**:

$$RMSE = \sqrt{\frac{1}{n}\sum_{p=1}^{n}(y_p - \hat{y}_p)^2} \tag{19}$$

- **Average Relative Error (REL)**:

$$REL = \frac{1}{n}\sum_{p=1}^{n}\frac{|y_p - \hat{y}_p|}{y_p} \tag{20}$$

- **Logarithmic Error**:

$$\log_{10} = \frac{1}{n}\sum_{p=1}^{n} |\log_{10}(y_p) - \log_{10}(\hat{y}_p)| \tag{21}$$

- **Accuracy under threshold δ_i**:

$$\delta_i = \max\left(\frac{y_p}{\hat{y}_p}, \frac{\hat{y}_p}{y_p}\right) < 1.25^i \tag{22}$$

- **Squared Relative Error (SqRel)**:

$$SqRel = \frac{1}{n}\sum_{p=1}^{n} \left(\frac{y_p - \hat{y}_p}{y_p}\right)^2 \tag{23}$$

- **Root Mean Squared Logarithmic Error ($\mathbf{RMSE}_{Log}$)**:

$$RMSE_{Log} = \sqrt{\frac{1}{n}\sum_{p=1}^{n} (\log(y_p) - \log(\hat{y}_p))^2} \tag{24}$$

Here, y_p and $\hat{y}_p$ denote pixel values in the ground-truth and predicted depth maps, respectively, and n is the total number of valid pixels.

4.3 Implementation details

DepthHist is formulated using PyTorch [38], allowing rigorous manipulation of computational aspects through the AdamW optimizer [39]. The latter is parameterised with a weight decay equals to 0.01, aimed at minimising overfitting by explicit regularisation of the parameters [40]. The learning process is governed by a One Cycle Learning policy [41], where the learning rate reaches a maximum of 1.25×10^{-3} before decaying in a linear anealing manner, ensuring convergence in 40 epochs. The depth estimate is quantified by a histogram discretised into 36 intervals, which is the number of histogram bins in *HistLayer*. The total number of trainable parameters of *DepthHist* is 125 million, where 86.7 million used by encoder, 6.8 million for the decoder and 31.5 million used by the *HistLayer*. The total loss function elaborated in Eq. 8 is weighted as the following: $\alpha = 100$ and $\beta = \gamma = 10^{-1}$. The values of α, β, and γ were empirically determined based on the convergence behavior and the obtained results. Reducing α while increasing β and γ led to suboptimal performance, both quantitatively and qualitatively.

4.4 Results and discussion

In this section, we compare the performance of the *DepthHist* on different datasets by comparing it against a diverse set of representative depth estimation models.

Table 1 highlights how our model, *DepthHist*, stands against the leading methods on KITTI. In particular, it delivers the lowest Absolute Relative Error at 0.048 (a 17.2% drop compared to AdaBins [6]) and the highest accuracy with a δ_1 score of 0.983, edging past IEBins (0.978). These results, together with the qualitative examples in Figure 7, show that *DepthHist* encodes scene structure more consistently, especially around thin edges and depth discontinuities.

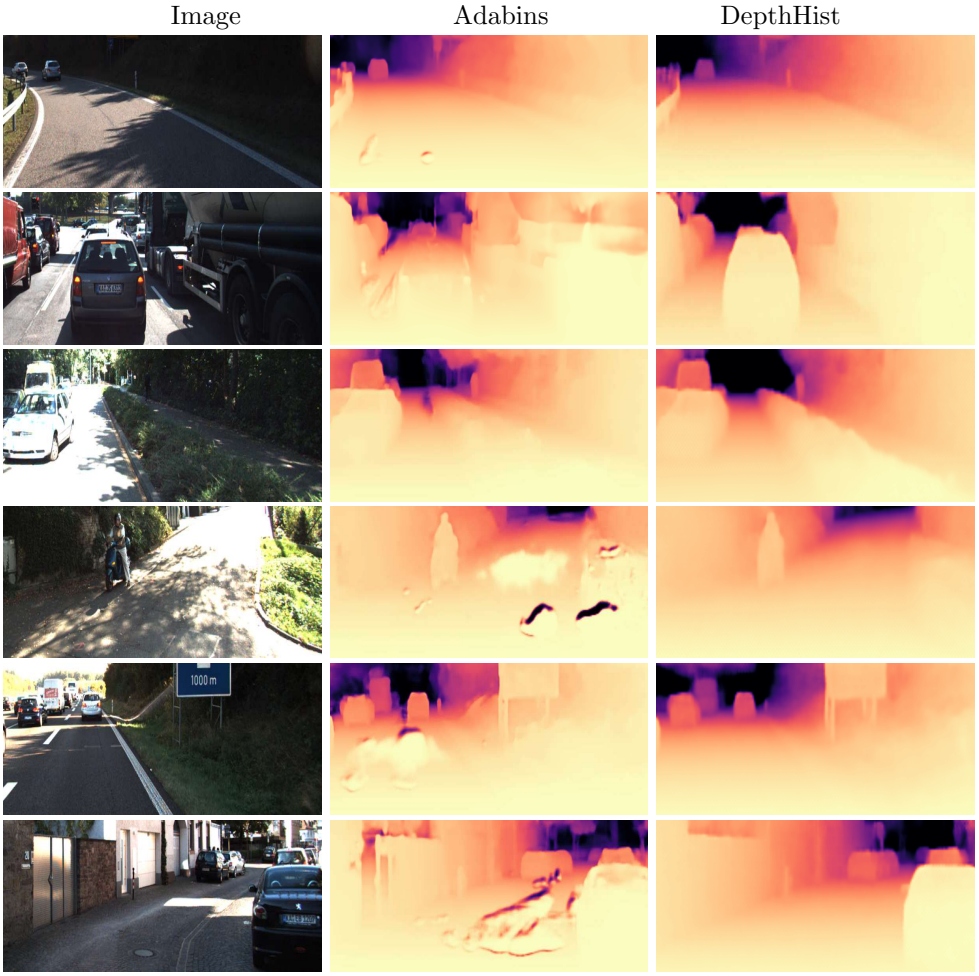


Fig. 7: Qualitative comparison with the AdaBins Model on KITTI dataset.

For indoor scenes, Table II shows that *DepthHist* extends well to NYU-Depth-v2. It reaches the lowest Absolute Relative Error of 0.076, which is 26.2% better than AdaBins and 12.6% better than IEBins, and achieves the highest δ_1 accuracy at 0.962, outperforming IEBins (0.936) by 2.8%. As shown in Figure 8, these improvements

Method	Class	Par. (M)	δ_1	δ_2	δ_3	Rel	SqRel	RMSE	RMSElog
Eigen et al. [30]	Regression	–	0.702	0.898	0.967	0.203	1.548	6.307	0.282
Godard et al. [42]	Regression	31	0.861	0.949	0.976	0.114	0.898	4.935	0.206
Cao et al. [43]	Classification	82	0.887	0.963	0.982	0.115	–	4.712	0.198
Repmono [44]	Classification	82	0.893	0.984	0.983	0.103	0.731	4.446	0.178
DualDepth* [45]	Regression	–	0.911	0.967	0.984	0.091	0.703	4.215	0.171
DORN [4]	Classification	93	0.932	0.984	0.994	0.072	0.307	2.727	0.120
DSC-MDE [46]	Regression	–	0.957	0.995	0.998	–	0.242	2.468	0.102
DepthRL-L (WS) [47]	Regression	213	0.969	0.996	0.999	0.055	0.175	2.237	0.100
Liu et al. [48]	Regression	213	0.969	0.996	–	0.053	0.169	2.221	–
AdaBins [6]	Hybrid	78	0.964	0.995	**0.999**	0.058	0.190	2.360	0.088
NeWCRFs [49]	Regression	270	0.974	<u>0.997</u>	**0.999**	0.052	0.155	2.129	0.079
Zhang et al. [50]	Hybrid	–	0.974	<u>0.997</u>	**0.999**	0.053	<u>0.142</u>	2.218	0.079
URCDC-Depth [19]	Regression	270	0.977	<u>0.997</u>	**0.999**	<u>0.050</u>	<u>0.142</u>	2.032	0.076
IEBins [10]	Hybrid	273	<u>0.978</u>	**0.998**	**0.999**	<u>0.050</u>	<u>0.142</u>	<u>2.011</u>	<u>0.075</u>
Ours	Histogram	125	**0.983**	**0.998**	**0.999**	**0.048**	**0.119**	**1.978**	**0.070**

Table 1: Performance comparison on the KITTI dataset (evaluation range: up to 80 meters). *DepthHist* (histogram-based on Cauchy) is evaluated against regression, classification, and hybrid methods. Top-performing results are shown in bold, and second-best results are underlined. Parameter counts (in millions) are included where available. Methods evaluated using depth maps with a resolution of 640 × 192 are marked with (*).

lead to clearer indoor reconstructions, especially in challenging areas such as cluttered layouts or reflective surfaces.

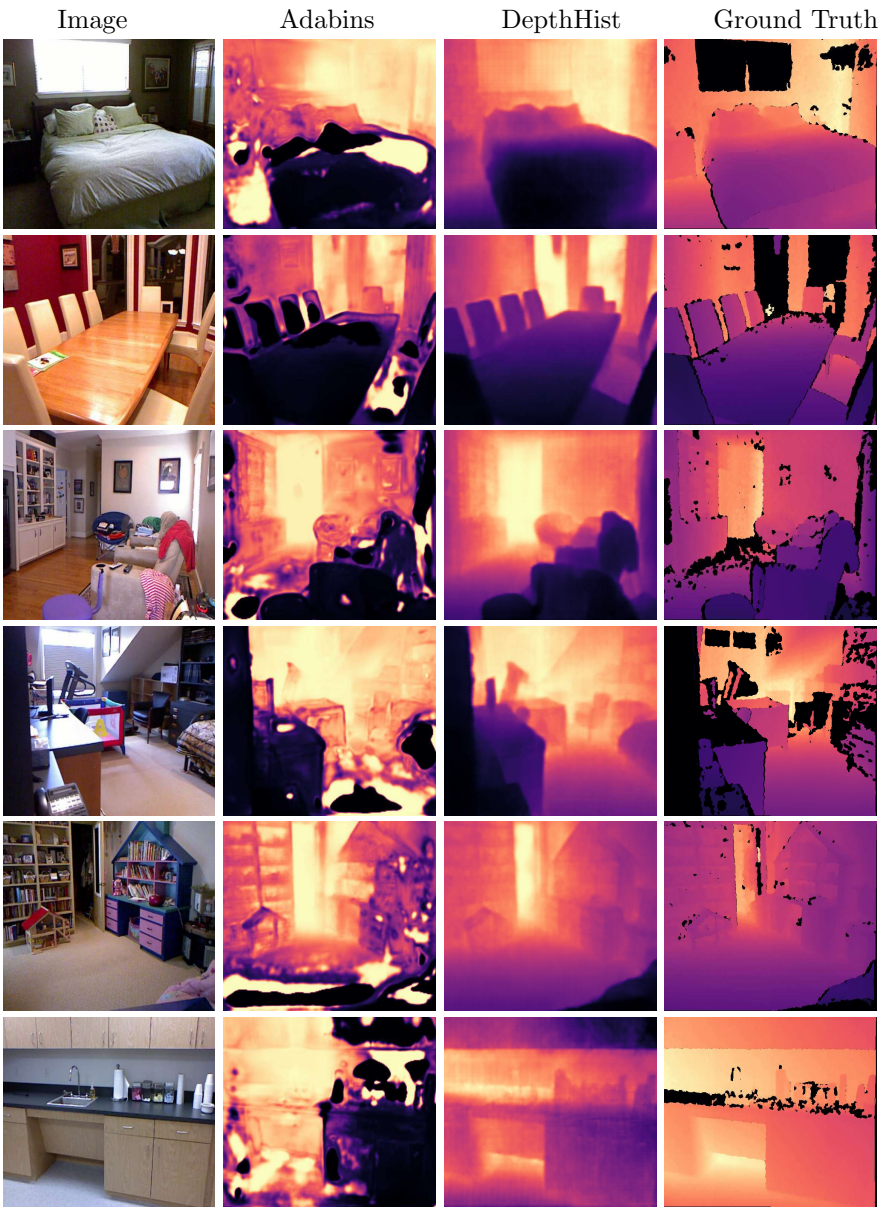


Fig. 8: Qualitative comparison with the AdaBins Model on the NYU-Depth-v2 dataset.

Method	Class	Par. (M)	δ_1	δ_2	δ_3	Rel	RMSE	$\log_{10}$
Eigen et al. [30]	Regression	-	0.769	0.950	0.988	0.158	0.641	–
Fu et al. [4]	Classification	93	0.828	0.965	0.992	0.115	0.509	0.051
NF-GLPN [51]	Regression	25	0.871	–	–	–	0.348	0.042
BTS [31]	Regression	47	0.885	0.978	0.994	0.110	0.392	0.047
AdaBins [6]	Hybrid	78	0.903	0.984	0.997	0.103	0.364	0.046
LocalBins [8]	Hybrid	74	0.907	0.987	0.998	0.099	0.357	0.042
DepthFormer [52]	Regression	-	0.921	0.992	0.998	0.096	0.339	0.041
NeWCRFs [49]	Regression	270	0.922	0.992	0.998	0.095	0.334	0.041
URCDC-Depth [19]	Regression	270	0.933	0.992	0.998	0.088	0.316	0.038
IEBins [10]	Hybrid	273	0.936	0.992	0.998	0.087	0.314	0.038
DepthRL-L (WS) [47]	Regression	213	0.948	0.994	**0.999**	0.080	**0.285**	0.034
Ours	Histogram	125	**0.962**	**0.996**	**0.999**	**0.076**	0.292	**0.032**

Table II: Performance comparison on the NYU-Depth-v2 dataset (evaluation range: 0 to 10 meters). DepthHist (Histogram-based on Laplacian) is evaluated against regression, classification, and hybrid methods. The top-performing outcomes are bolded, while the second-best outcomes are underlined. Parameter counts (millions) are included where available.

Computational efficiency: To evaluate the computational efficiency of *DepthHist*, we benchmarked the average inference time on various input resolutions and compared it to two models: *IEBins* [10] and *NeWCRFs* [49]. Experiments were run on a system with Ubuntu 24.04.2 LTS, 64GB RAM, Intel Core i7-10700 (8 cores, 16 threads) CPU, and an NVIDIA RTX 3060 GPU with 12GB VRAM. All models were tested with batch size 1 and over than 30 iterations. Figure 8 shows the inference time at different popular resolutions.

Cross-validation on the SUNRGBD dataset: To illustrate the generalization capacity of our model and its resilience to data distribution shifts, we took *DepthHist* based on the Laplacian kernel, trained on the NYU-Depth-v2 dataset, and directly evaluated it on the SUNRGBD dataset [53]. This dataset contains 5,050 high-resolution RGB-depth image pairs, taken in a wide range of indoor scenes. We used the same evaluation protocol as for NYUv2. As can be seen from our results in Table II, *DepthHist* achieves the best performance across all reported metrics. Compared to the second-best method (IEBins), it reduces the Absolute Relative Error by *1.5%*, the Root Mean Square Error by *0.25%*, and improves the $\boldsymbol{\delta_1}$ accuracy by *0.97%*.

5 Ablation study

In order to rigorously assess the influence of the proposed contributions on depth estimation, we carried out a series of experiments on NYUv2 Dataset based on the following variations:

Number of Bins: Choosing the number of bins used by *DepthHist* is critical for both accurate depth estimation and computational performance. The model partitions the continuous depth interval of the image using a pre-specified number of bins, each representing a particular range of the depth distribution. Using too few bins causes

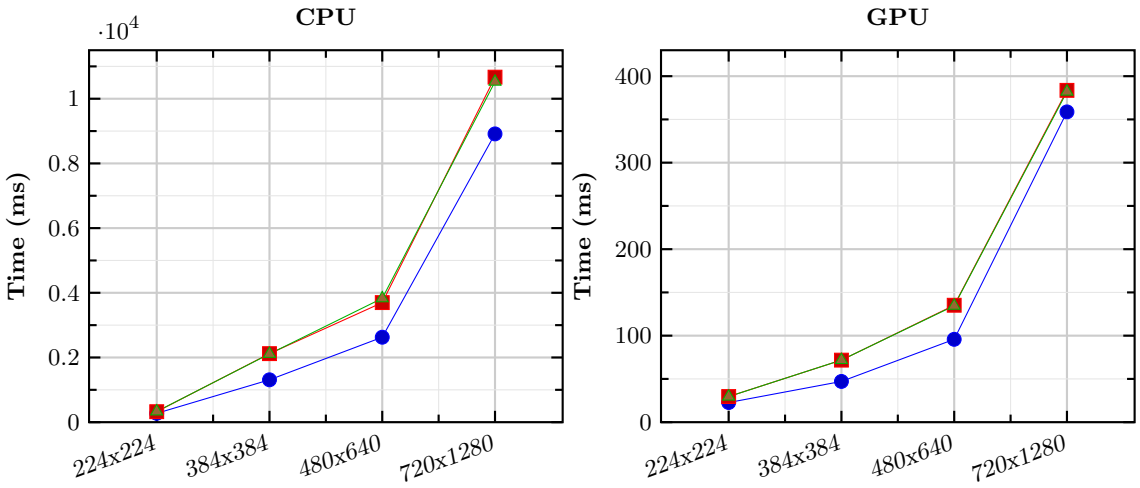


Fig. 8: Inference time comparison across resolutions for CPU and GPU execution. Lower values indicate better performance. Models are distinguished by: • **Ours**, ■ **IEBins**, and ▲ **NeWCRFs**.

Method	Abs Rel ↓	RMSE ↓	δ_1 ↑	δ_2 ↑	δ_3 ↑
BTS [31]	0.172	0.515	0.740	0.933	0.980
AdaBins [6]	0.159	0.476	0.771	0.944	0.983
LocalBins [8]	0.156	0.467	0.777	0.949	0.985
PixelFormer [9]	0.144	0.441	0.802	0.962	0.990
BinsFormer [7]	0.143	0.422	0.805	0.963	0.990
IEBins [10]	<u>0.135</u>	<u>0.405</u>	<u>0.822</u>	<u>0.971</u>	<u>0.993</u>
Ours	**0.133**	**0.404**	**0.830**	**0.987**	**0.999**

Table II: Cross-dataset validation on SUNRGBD dataset using models trained on NYU-Depth-v2. The best results are highlighted in **bold**, and the second-best are underlined.

over-quantization: depth values congregate, spatial detail is lost, and depth discontinuities are poorly preserved. For example, quantizing ten depth levels (0–9) into five bins produces assignments that fail to preserve the original distribution. Conversely, using too many bins leads to over-partitioning, increasing computational cost without yielding a commensurate accuracy gain. To explore this trade-off, we systematically varied the number of bins and evaluated performance using the Root Mean Squared Error (RMSE). As shown in Figure 10, the RMSE decreases from 0.401 with four bins to 0.308 with thirty-six bins, further increases yield only marginal improvements, indicating a saturation effect.

Loss Functions: In order to develop a better understanding of how different loss formulations affect performance, experiments were run with several combinations. We

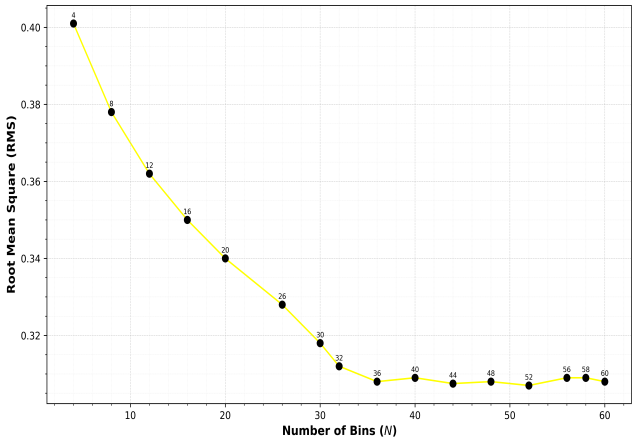


Fig. 10: a comprehensive visualization of the number of bins and RMSE trade-off, illustrating the impact of different bin numbers.

started with the basic scale-invariant loss ($\mathcal{L}_{SILoss}$), then summmed the the basic scale-invariant loss with the Chamfer loss ($\mathcal{L}_{\boldsymbol{Chamfer}}$), then summed the basic scale-invariant loss with the 2D histogram loss ($\mathcal{L}_{2DLoss}$), and finally summed all the three loss functions together. Each setting was evaluated to understand its influence on the accuracy and error measures of the model. Table IV summarizes the results and provides a brief comparison among these different training approaches.

Loss	$\boldsymbol{\delta_1}\uparrow$	$\boldsymbol{\delta_2}\uparrow$	RMSE $\downarrow$	Rel $\downarrow$
$\mathcal{L}_{SILoss}$	0.948	0.994	0.312	0.083
$\mathcal{L}_{SILoss} + \mathcal{L}_{\boldsymbol{Chamfer}}$	0.951	0.994	0.309	0.080
$\mathcal{L}_{SILoss} + \mathcal{L}_{2DLoss}$	<u>0.955</u>	**0.996**	<u>0.306</u>	**0.079**
$\mathcal{L}_{SILoss} + \mathcal{L}_{\boldsymbol{Chamfer}} + \mathcal{L}_{2DLoss}$	**0.962**	**0.996**	**0.292**	**0.076**

Table IV: Comparison of performance with respect to the choice of loss function using *DepthHist*-based Laplacian kernel (Eq. 4) on the NYU-Depth-v2 dataset. The best results are bolded, while the second-best are underlined.

Histogram Kernels: The three soft binning kernels introduced in Methodology section are compared to evaluate their effect on loss calculation and overall model performance. Quantitative results, presented in Table V, highlight the contribution of each kernel to depth estimation accuracy, while Figure 11 provides qualitative comparisons of the resulting depth maps. In addition to internal comparisons, the proposed kernels are also evaluated against a previous method based on activation function mappings introduced by Avi *et al.* [13], within a unified evaluation framework.

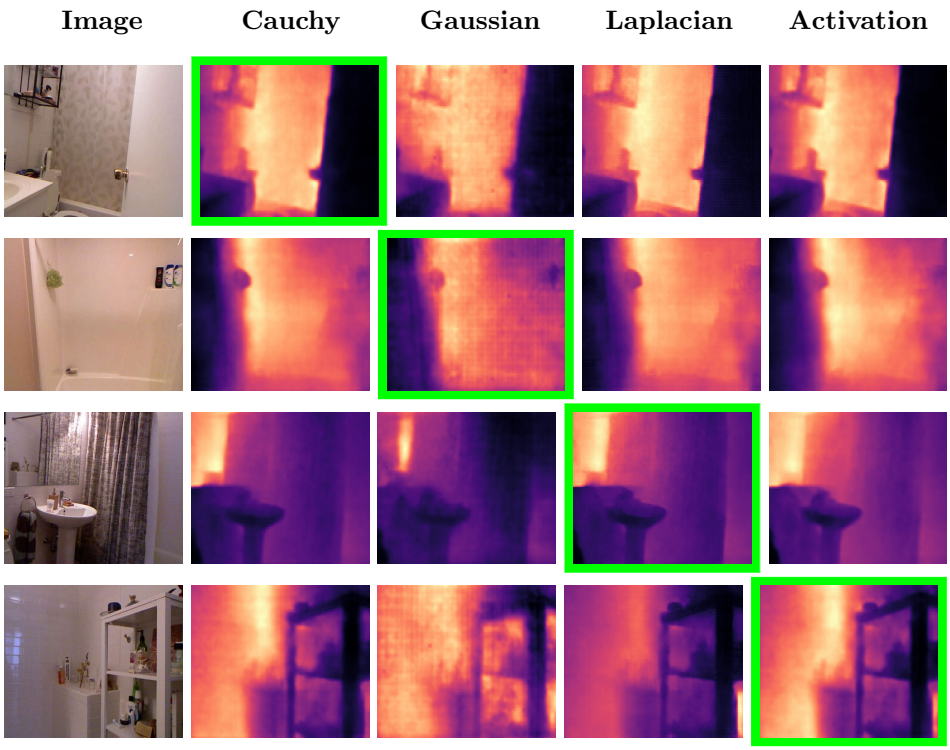


Fig. 11: **Qualitative comparison of soft binning kernels and activation-based mapping.** The image highlighted with a green rectangle corresponds to the prediction with the highest $\boldsymbol{\delta_1}$ score.

Among the tested variants, Cauchy and Laplacian kernels consistently produce smoother and more coherent depth predictions, demonstrating strong adaptability to the depth estimation task. To increase interpretability of each binning approach, we measured how often each method produced the highest $\boldsymbol{\delta_1}$ score across the test set. The Laplacian kernel was best in **38.2%** of samples and the Cauchy kernel in **36.7%** (together $\approx$ **74.9%**), while the Gaussian, activation-based, and tied cases occurred in **9.1%**, **6.3%**, and **9.7%** of samples, respectively. These results mirror the kernels' decay near zero: the Laplacian's ($|\boldsymbol{x}|$) driven steep drop favors fine local discrimination, the Gaussian's earlier but shallower ($\boldsymbol{e^{-x^2}}$) decay is less discriminative for fine features, and the Cauchy combines a steep near-zero drop with long polynomial tails ($\boldsymbol{\propto 1/x^2}$), balancing local accuracy and global coherence. The activation-based method's relatively flat response explains its weaker performance. Each kernel's response is illustrated in Figure 12.

Augmentation Techniques: To assess the contribution of the proposed augmentation method, *SliceStitch*, we performed an analysis comparing it to the *CutFlip* method. All training settings are exactly the same for all experiments, with the only difference being the selection of the augmentation approach in order to provide a fair

Kernel	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	RMSE $\downarrow$	Abs Rel $\downarrow$
Gaussian (Eq. 2)	0.892	0.989	0.998	0.394	0.110
Laplacian (Eq. 4)	**0.962**	**0.996**	**0.999**	**0.292**	**0.076**
Cauchy (Eq. 3)	0.955	**0.996**	**0.999**	0.308	0.079
Activation [13]	0.923	0.994	**0.999**	0.363	0.096

Table V: Comparative performance of kernel-based methods and activation-based approach. Best results are bolded, and second-best are underlined. The Laplacian kernel achieves the highest accuracy in most metrics, closely followed by Cauchy.

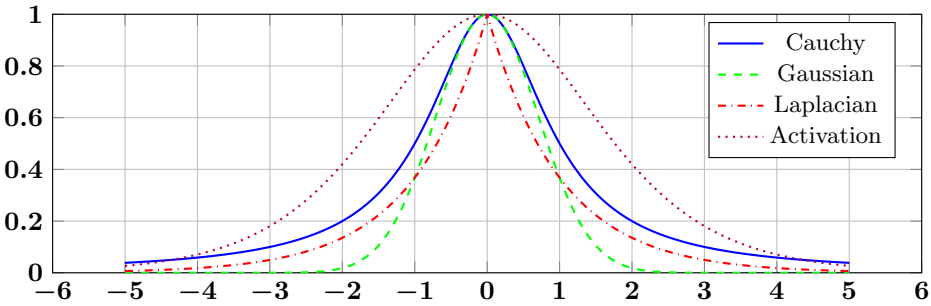


Fig. 12: **Response of binning kernels:** Sharper kernels assign values more accurately when data points are near bin centers, while broader kernels yield less precise assignments for distant values.

and controlled comparison. From Table VI, it can be seen that adding *SliceStitch* provides steady performance improvements for both *AdaBins* [6] as well as our suggested *DepthHist* model. Comparatively to *CutFlip*, *SliceStitch* shows a greater ability to enhance generalization. While *CutFlip* perturbs depth priors by decoupling depth and vertical image position correlation, its augmentation is still limited to a single scene and maintains some semantic coherence.

Method	Rel $\downarrow$	RMSE $\downarrow$	$\log_{10} \downarrow$	$\delta_1 \uparrow$	$\delta_2 \uparrow$
AdaBins [6]	0.103	0.364	0.046	0.903	0.988
+ CutFlip	0.099	0.345	0.045	0.907	0.989
+ SliceStitch	**0.095**	**0.321**	**0.044**	**0.912**	**0.990**
DepthHist (Ours)	0.085	0.351	0.036	0.947	0.994
+ CutFlip	0.082	0.333	0.035	0.951	0.995
+ SliceStitch	**0.079**	**0.308**	**0.034**	**0.955**	**0.996**

Table VI: Ablation study comparing baseline, CutFlip, and SliceStitch augmentations for *AdaBins* and *DepthHist* models.

By contrast, *SliceStitch* creates more discontinuous and semantically richer spatial perturbations by concatenating two different image-depth pairs along a horizontal or vertical axis. This design de-emphasizes the model's use of global structural priors and forces the model to rely more on localized geometric cues.

6 conclusion

In this work, we have presented a novel approach for depth estimation using histograms based on a soft binning strategy. As shown by our experiments on the KITTI and NYUv2 datasets, our method provides substantial improvement compared to state-of-the-art methods such as AdaBins, IEBins and NewCRfs both quantitatively and qualitatively. These results point toward the potential of soft binning to advance depth estimation in deep learning frameworks, guiding the field toward a formulation where true histogram-based modeling plays a decisive role. We plan to incorporate the soft binning mechanism into the internal structures of neural networks in future works. This integration will examine its use as an alternative to softmax in applications like filter pruning. We also expect to explore its application in variational inference methods, where the probabilistic nature of soft binning can be further leveraged. Finally, we presented *SliceStitch*, an augmentation technique that encourages spatial generalization by stitching together regions of different images. Its proven impact on depth estimation indicates its potential for general applications in vision tasks that demand strong structural reasoning, such as semantic segmentation, surface normal estimation, or optical flow.

Acknowledgements. This work was supported by the Algerian Directorate General for Scientific Research and Technological Development (DGRSDT).

Declarations

- Funding: This work was supported by the Algerian Directorate General for Scientific Research and Technological Development (DGRSDT).
- Conflict of interest: Not applicable
- Consent for publication: Not applicable
- Data availability: Not applicable
- Materials availability: Not applicable
- Code availability: Not applicable
- Author contribution: Not applicable

References

[1] Diamantas, S.C., Oikonomidis, A., Crowder, R.M.: Depth estimation for autonomous robot navigation: A comparative approach. In: IEEE Conf. on Imaging Sys. and Tech., pp. 426–430 (2010). https://doi.org/10.1109/IST.2010.5548483

[2] Cao, Y., Shen, C., Shen, H.T.: Exploiting depth from single monocular images for object detection and semantic segmentation. IEEE Trans. on Image Processing **26**(2), 836–846 (2017) https://doi.org/10.1109/TIP.2016.2621673

[3] Carvalho, M., Saux, B.L., Trouvé-Peloux, P., Almansa, A., Champagnat, F.: On regression losses for deep depth estimation. In: IEEE Conf. on Image Proces. (ICIP), pp. 2915–2919 (2018). https://doi.org/10.1109/ICIP.2018.8451312

[4] Fu, H., Gong, M., Wang, C., Batmanghelich, K., Tao, D.: Deep ordinal regression network for monocular depth estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018). https://doi.org/10.48550/arXiv.1806.02446

[5] Cao, Y., Zhao, T., Xian, K., Shen, C., Cao, Z., Xu, S.: Monocular depth estimation with augmented ordinal depth relationships. IEEE Transactions on Image Processing, 1–1 (2018) https://doi.org/10.1109/TIP.2018.2877944

[6] Farooq Bhat, S., Alhashim, I., Wonka, P.: Adabins: Depth estimation using adaptive bins. In: IEEE Conf. on Comp. Vis. and Pat. Rec. (CVPR) (2021). https://doi.org/10.1109/cvpr46437.2021.00400 . http://dx.doi.org/10.1109/CVPR46437.2021.00400

[7] Li, Z., Wang, X., Liu, X., Jiang, J.: Binsformer: Revisiting adaptive bins for monocular depth estimation. In: IEEE Trans. on Image Processing, vol. 33, pp. 3964–3976 (2024). https://doi.org/10.1109/TIP.2024.3416065

[8] Bhat, S.F., Alhashim, I., Wonka, P.: Localbins: Improving depth estimation by learning local distributions. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022, pp. 480–496. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-19769-7_28

[9] Agarwal, A., Arora, C.: Attention attention everywhere: Monocular depth prediction with skip attention. In: IEEE Winter Conf. on App. of Comp. Vis. (WACV), pp. 5850–5859 (2023). https://doi.org/10.1109/WACV56688.2023.00581

[10] Shao, S., Pei, Z., Chen, W., Chen, P.C.Y., Li, Z.: Iebins: Iterative elastic bins for monocular depth estimation and completion. International Journal of Computer Vision **133**(5), 2463–2486 (2025) https://doi.org/10.1007/s11263-024-02293-3

[11] Zhu, R., Song, Z., Liu, L., He, J., Zhang, T., Zhang, Y.: Ha-bins: Hierarchical adaptive bins for robust monocular depth estimation across multiple datasets. IEEE Trans. on Circuits and Systems for Video Technology, 4354–4366 (2024) https://doi.org/10.1109/TCSVT.2023.3335316

[12] Damadi, S., Moharrer, G., Cham, M.: The Backpropagation algorithm for a math student (2023). https://arxiv.org/abs/2301.09977

[13] Avi-Aharon, M., Arbelle, A., Raviv, T.R.: Differentiable histogram loss functions for intensity-based image-to-image translation. IEEE Trans. on Pattern Analysis and Machine Intelligence, 11642–11653 (2023) https://doi.org/10.1109/TPAMI.2023.3278287

[14] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conf. on Learning Representations (2021). https://openreview.net/forum?id=YicbFdNTTy

[15] "Wang, Z., Li, H., Ouyang, W., Wang, X.: "learnable histogram: Statistical context features for deep neural networks". In: "Computer Vision – ECCV", pp. 246–262. "Springer International Publishing", ??? ("2016"). https://doi.org/10.48550/arXiv.1804.09398"

[16] Peeples, J., Xu, W., Zare, A.: Histogram layers for texture analysis, pp. 541–552 (2022). https://doi.org/10.1109/TAI.2021.3135804

[17] Marrs, A.D., Webb, A.R.: Exploratory data analysis using radial basis function latent variable models. In: Proceedings of the 12th International Conference on Neural Information Processing Systems. NIPS'98, pp. 529–535. MIT Press, Cambridge, MA, USA (1998). https://doi.org/10.1109/TIP.2025.3533207

[18] Shorten, C., Khoshgoftaar, T.M.: A survey on image data augmentation for deep learning. Journal of Big Data **6**(1), 60 (2019) https://doi.org/10.1186/s40537-019-0197-0

[19] Shao, S., Pei, Z., Chen, W., Li, R., Liu, Z., Li, Z.: Urcdc-depth: Uncertainty rectified cross-distillation with cutflip for monocular depth estimation. IEEE Trans. on Multimedia **26**, 3341–3353 (2024) https://doi.org/10.1109/TMM.2023.3310259

[20] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: IEEE Int. Conf. on Comp. Vis. (ICCV), pp. 9992–10002 (2021). https://doi.org/10.1109/ICCV48922.2021.00986

[21] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS'17, pp. 6000–6010. Curran Associates Inc., Red Hook, NY, USA (2017). https://doi.org/10.5555/3295222.3295349

[22] Yu, Y., Zhao, D., Shen, G., Dong, Y., Zeng, Y.: Brain-inspired stepwise patch merging for vision transformers. In: Int. Joint Conf. on Artificial Intelligence, IJCAI, pp. 2305–2313 (2025). https://doi.org/10.24963/ijcai.2025/257

[23] Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28

[24] Dumoulin, V., Visin, F.: A guide to convolution arithmetic for deep learning (2018). https://arxiv.org/abs/1603.07285

[25] Alzubaidi, L., Zhang, J., Humaidi, A.J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M.A., Al-Amidie, M., Farhan, L.: Review of deep learning: concepts, cnn architectures, challenges, applications, future directions. Journal of Big Data **8**(1), 53 (2021) https://doi.org/10.1186/s40537-021-00444-8

[26] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conf. on Comp. Vis. and Pat. Rec. (CVPR), pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90

[27] Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: IEEE Conf. on Comp. Vis. and Pat. Rec. (CVPR) (2017)

[28] Chen, Y., Li, J., Xiao, H., Jin, X., Yan, S., Feng, J.: Dual Path Networks (2017). https://arxiv.org/abs/1707.01629

[29] Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing **13**(4), 600–612 (2004) https://doi.org/10.1109/TIP.2003.819861

[30] Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2. NIPS'14, pp. 2366–2374. MIT Press, Cambridge, MA, USA (2014). https://doi.org/10.5555/2969033.2969091

[31] Lee, J.H., Han, M.-K., Ko, D.W., Suh, I.H.: From big to small: Multi-scale local planar guidance for monocular depth estimation. In: arXiv Preprint arXiv:1907.10326 (2021). https://doi.org/10.48550/arXiv.1907.10326 . https://arxiv.org/abs/1907.10326

[32] Wu, T., Pan, L., Zhang, J., Wang, T., Liu, Z., Lin, D.: Density-aware Chamfer Distance as a Comprehensive Metric for Point Cloud Completion (2021). https://arxiv.org/abs/2111.12702

[33] Dehong, C., Liangqi, Z., Pengpeng, S., Zaiyang, T., Yuhao, M., Yong, W.: Design and implementation of lidar navigation system based on triangulation measurement. In: 2017 29th Chinese Control And Decision Conference (CCDC), pp. 6060–6063 (2017). https://doi.org/10.1109/CCDC.2017.7978258

[34] Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. Int. J. Rob. Res. **32**(11), 1231–1237 (2013) https://doi.org/10.1177/0278364913491297

[35] Garg, R., B.G., V.K., Carneiro, G., Reid, I.: Unsupervised cnn for single view depth estimation: Geometry to the rescue. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision – ECCV 2016, pp. 740–756. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_45

[36] Zhang, Z.: Microsoft kinect sensor and its effect. IEEE MultiMedia **19**(2), 4–10 (2012) https://doi.org/10.1109/MMUL.2012.24

[37] Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) Computer Vision – ECCV 2012, pp. 746–760. Springer, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33715-4_54

[38] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: an imperative style, high-performance deep learning library. Curran Associates Inc., Red Hook, NY, USA (2019). https://doi.org/10.5555/3454287.3455008

[39] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: Int. Conf. on Learning Representations (2019). https://openreview.net/forum?id=Bkg6RiCqY7

[40] Bos, S., Chug, E.: Using weight decay to optimize the generalization ability of a perceptron. In: Int. Conf. on Neural Networks, vol. 1, pp. 241–2461 (1996). https://doi.org/10.1109/ICNN.1996.548898

[41] Smith, L.N., Topin, N.: Super-convergence: very fast training of neural networks using large learning rates. In: Art. Int. and Machine Learning for Multi-Domain Operations Applications, p. 1100612 (2019). https://doi.org/10.1117/12.2520589 . https://doi.org/10.1117/12.2520589

[42] Godard, C., Aodha, O.M., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: IEEE Conf. on Comp. Vis. and Pat. Rec. (CVPR), pp. 6602–6611 (2017). https://doi.org/10.1109/CVPR.2017.699

[43] Cao, Y., Wu, Z., Shen, C.: Estimating depth from monocular images as classification using deep fully convolutional residual networks. In: IEEE Trans. Circuits and Systems for Video Technology, pp. 3174–3182 (2018). https://doi.org/10.1109/TCSVT.2017.2740321

[44] Zhang, G., Tang, X., Wang, L., Cui, H., Fei, T., Tang, H., Jiang, S.: Repmono:

a lightweight self-supervised monocular depth estimation architecture for high-speed inference. Complex & Intelligent Systems **10**(6), 7927–7941 (2024) https://doi.org/10.1007/s40747-024-01575-0

[45] Feng, C., Zhang, C., Chen, Z., Hu, W., Lu, K., Ge, L.: Self-supervised monocular depth estimation with dual-path encoders and offset field interpolation. IEEE Trans. on Image Processing **34**, 939–954 (2025) https://doi.org/10.1109/TIP.2025.3533207

[46] Yan, W., Dong, L., Ma, W., Mi, Q., Zha, H.: Dsc-mde: Dual structural contexts for monocular depth estimation. Knowledge-Based Systems **263**, 110308 (2023) https://doi.org/10.1016/j.knosys.2023.110308

[47] Chen, H., Wang, Y., Zhang, J., Zhang, J., Pan, Z., Jia, S., Huang, S.: Depthrl: a weakly supervised approach for monocular depth estimation using deep reinforcement learning. Complex & Intelligent Systems **11**(8), 344 (2025) https://doi.org/10.1007/s40747-025-01967-w

[48] Liu, L., Tian, J., Luo, G., Xu, S., Zhang, C., Hu, H., Tao, W.: Image depth estimation assisted by multi-view projection. Complex & Intelligent Systems **11**(1), 66 (2024) https://doi.org/10.1007/s40747-024-01688-6

[49] Yuan, W., Gu, X., Dai, Z., Zhu, S., Tan, P.: Neural window fully-connected crfs for monocular depth estimation. In: IEEE Conf. on Comp. Vis. and Pat. Rec. (CVPR), pp. 3906–3915 (2022). https://doi.org/10.1109/CVPR52688.2022.00389

[50] Duan, Z., Chen, Y., Yu, H., Hu, B., Chen, C.: Rgb-fusion: Monocular 3d reconstruction with learned depth prediction. Displays **70**, 102100 (2021) https://doi.org/10.1016/j.displa.2021.102100

[51] Chen, Y.-H., Chen, Y.-J., Hsu, C.-C.: Enhanced near-field depth estimation: Integrating global and local path networks. Knowledge-Based Systems **328**, 114198 (2025) https://doi.org/10.1016/j.knosys.2025.114198

[52] Li, Z., Chen, Z., Liu, X., Jiang, J.: Depthformer: Exploiting long-range correlation and local information for accurate monocular depth estimation. Machine Intelligence Research, 837–854 (2023) https://doi.org/10.1007/s11633-023-1458-0

[53] Song, S., Lichtenberg, S.P., Xiao, J.: Sun rgb-d: A rgb-d scene understanding benchmark suite. In: IEEE Conf. on Comp. Vis. and Pat. Rec. (CVPR), pp. 567–576 (2015). https://doi.org/10.1109/CVPR.2015.7298655