Learning to Adapt CLIP for Few-Shot Monocular Depth Estimation

Xueting Hu[1] Ce Zhang[1] Yi Zhang[1] Bowen Hai[1] Ke Yu[1] Zhihai He[1,2*]
[1]Department of Electronic and Electrical Engineering, Southern University of Science and Technology, Shenzhen, China
[2]Pengcheng Laboratory, Shenzhen, China
{huxt2022, zhangc2019, zhangyi2021, haibw2022, yuk2020}@mail.sustech.edu.cn, hezh@sustech.edu.cn

Abstract

Pre-trained Vision-Language Models (VLMs), such as CLIP, have shown enhanced performance across a range of tasks that involve the integration of visual and linguistic modalities. When CLIP is used for depth estimation tasks, the patches, divided from the input images, can be combined with a series of semantic descriptions of the depth information to obtain similarity results. The coarse estimation of depth is then achieved by weighting and summing the depth values, called depth bins, corresponding to the predefined semantic descriptions. The zero-shot approach circumvents the computational and time-intensive nature of traditional fully-supervised depth estimation methods. However, this method, utilizing fixed depth bins, may not effectively generalize as images from different scenes may exhibit distinct depth distributions. To address this challenge, we propose a few-shot-based method which learns to adapt the VLMs for monocular depth estimation to balance training costs and generalization capabilities. Specifically, it assigns different depth bins for different scenes, which can be selected by the model during inference. Additionally, we incorporate learnable prompts to preprocess the input text to convert the easily human-understood text into easily model-understood vectors and further enhance the performance. With only one image per scene for training, our extensive experiment results on the NYU V2 and KITTI dataset demonstrate that our method outperforms the previous state-of-the-art method by up to 10.6% in terms of MARE[1].

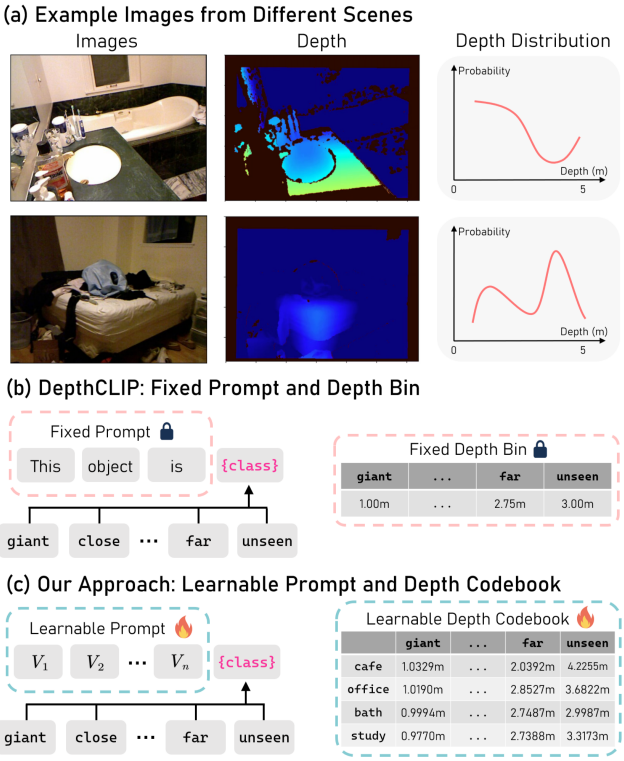

Figure 1. **Illustration of our motivation and major idea.** (a) Depth estimation and distributions of two example images from `bathroom` and `bedroom` scenes. (b) Fixed prompt and depth bin in DepthCLIP [48]. (c) Proposed learnable prompt and depth codebook in our approach.

1. Introduction

Recently, attention has been drawn toward large-scale pre-trained Vision-Language Models (VLMs). VLMs are pre-trained on massive amounts of pairs of texts and images available on the Internet, allowing them to gain a deeper understanding of the connection between language and vision and learn more comprehensive visual representations [16, 31]. Given these capabilities, VLMs can play a crucial role in processing multi-modal tasks. Notably, the Contrastive Language-Image Pre-Training (CLIP) [31] model stands out for its remarkable image and text matching capability. During pre-training, CLIP encodes the input image and text using separate image and text encoders to obtain corresponding feature representations. By measuring the cosine similarity between the image feature and text feature, CLIP identifies the image-text pair with the highest similar-

*Corresponding author.
[1]This work is supported by Center for Computational Science and Engineering at Southern University of Science and Technology.

ity as the positive sample and the rest as negative samples. With a contrastive learning loss function, the CLIP model is trained on a dataset consisted of 400 million image-text pairs. Empirical evidence has demonstrated that such pre-trained models are highly effective in tasks such as few-shot image classification [49, 50, 53], transfer learning [39, 40], and image captioning [22, 43].

Monocular depth estimation, a key downstream task in computer vision, holds significant applications in autonomous driving, virtual reality and robotics. Traditional machine learning algorithms such as support vector machines (SVM) and random forests are commonly employed in this context [9, 33]. In contrast, deep learning-based depth estimation methods leverage deep neural networks to acquire intricate feature representations and perform depth prediction tasks. These methodologies typically exploit deep learning models like convolutional neural networks and autoencoders [10, 41]. Both machine learning-based and deep learning-based methods entail substantial data requirements for successful training, and are computationally expensive and time-consuming [30].

Recently, a novel approach called DepthCLIP [48] proposes to perform zero-shot monocular depth estimation using the large-scale CLIP model. In this approach, the patches of the input image respond to a specific semantic distance token, which determine the contribution of this patch to various depth descriptions. The depth descriptions, which are predefined by setting corresponding depth values manually, are then weighted and summed with the relevant depth values to yield the final depth estimation results. However, we recognize that images from different scenes have various depth distributions. As shown in Figure 1 (a), the two images from different scenes have totally different depth distributions. Therefore, pre-setting specific parameters, such as the depth values corresponding to the classes describing the depth, may lead to model performance that is strong in certain scenarios but suboptimal in others.

We observe that the key to achieving optimal performance for monocular depth estimation models lies in their ability to generalize effectively. To overcome the generalizability issues outlined previously, in this work, we design a few-shot learning approach. We endeavor to achieve a harmonious balance between the traditional approach, which can be time-consuming and labor-intensive, and the zero-shot method, which may result in limited generalization capabilities.

Specifically, we propose a novel approach to adapt the CLIP model for few-shot monocular depth estimation. Given its nature as a model architecture that integrates visual and textual elements, it is inherently advantageous to extract comprehensive and enhanced features from both visual and textual domains. (1) From the visual perspective, an image can be examined globally or analyzed locally. A global view of a scene image allows for the extraction of its scene features. These scene features serve as evaluative criteria when determining the optimal depth bin selection. Here, unlike previous methods which relied on artificially-set depth values corresponding to pre-defined depth descriptions, we design a depth codebook, which enables the model to identify the most appropriate depth value in accordance with the current scene feature of the image during inference. A local perspective of the image entails dividing it into patches, and for each patch, the image feature is extracted. All these extracted features play a crucial role in the subsequent stages of the process. (2) From the textual perspective, we leverage a simple network to preprocess the input text by converting its content from easily human-understood text into easily model-understood vectors, enhancing the model's suitability for monocular depth estimation. A comparison of our approach and previous state-of-the-art DepthCLIP [48] is shown in Figure 1 (b) and (c). Note that our approach is conducted with the few-shot setting. Specifically, we opt to utilize one image from each category for the purposes of training, during which several model parameters are adjusted and fine-tuned to yield optimal performance. With only one image per scene for training, our extensive experiment results on the NYU V2 dataset [38] demonstrate that our method outperforms the previous state-of-the-art method by up to 10.6% in terms of MARE.

The major contributions of our paper are summarized as follows:

1. We explore the monocular depth estimation task using vision-language models in a new few-shot setting, which aims to improve the depth estimation performance of pre-trained large-scale models by leveraging a small set of annotated samples.

2. We recognize that images from different scenes have various depth distributions. To address this issue, we design learnable prompts and learnable depth codebooks to adapt the CLIP model for different scenes effectively.

3. We evaluate our proposed method on NYU V2 dataset [38]. With only one image per scene for training, our method outperforms the previous state-of-the-art method by up to 10.6% in terms of MARE, and can even be compared to fully-supervised methods.

2. Related Work

In this section, we review related works on vision-language models, prompt tuning methods for VLMs, and monocular depth estimation.

(1) Vision-language models. Large-scale pre-trained vision-language models (VLMs) have been developed to learn general visual representation under the supervision of

natural languages [7, 13, 20, 31, 34]. These models utilize extensive datasets to acquire representations that encompass the semantic comprehension of images and their accompanying textual descriptions. For instance, CLIP [31] is derived through the application of contrastive learning on a dataset of 400 million curated image-text pairs, while ALIGN [16] utilizes 1.8 billion noisy image-text pairs. Several other studies have been conducted along the direction of VLMs, like CoCa [45], Flamingo [1], and IDEA [14].

In recent years, large-scale pre-trained VLMs have been applied to and shown great performances on various cross-modal alignment, zero-shot and few-shot image recognition tasks [11, 31, 47, 51], and other visual tasks including image retrieval [8, 26], visual grounding [22, 44], and visual question answering [8, 19, 54].

(2) Prompt tuning methods for VLMs. The concept of "prompt tuning" involves adapting pre-trained basic models for downstream tasks, especially in scenarios with limited or no prior training data. Initially introduced in the context of text input in language models, prompting aimed to enhance their output by providing specific instructions or examples as part of the input [2, 32]. In subsequent research [36, 37], the approach involved transforming the downstream task into a "cloze" task using predefined patterns or templates, eliminating the need for task-specific classifiers. However, discovering the most effective patterns can be time-consuming, prompting recent studies to explore soft prompts learned in a continuous manner [21, 24].

Prompt tuning techniques for fine-tuning VLMs have focused on devising intricate prompts and incorporating adaptable context to extract task-relevant information from the encoded knowledge [52, 53]. In recent advancements, several notable prompt tuning methods have demonstrated significant improvements [5, 17, 27, 42]. CoOp [53], a pioneering work in this field, optimizes prompt context by employing a series of learnable vectors, either in a unified or class-specific manner. Building upon CoOp, CoCoOp [52] enhances the method by learning to generate vectors conditioned on individual image, effectively solving the problem of generalization to unseen classes.

(3) Monocular depth estimation. Monocular depth estimation is a crucial task in computer vision and finds numerous applications in autonomous driving, virtual reality, and robotics. Recent advancements in monocular depth estimation can be categorized into two main approaches: *supervised* and *unsupervised* methods.

Supervised methods leverage ground truth depth maps during the training process to grasp semantic priors and extract semantic relationships [3, 15, 18]. Recently, Li *et al.* [25] introduced Depthformer, which incorporates additional modules to enhance Transformer features through element-wise interaction. It models the affinity between Transformer and CNN features in a set-to-set translation manner, further improving depth estimation performance. However, it would be costly and labor-intensive to gather fully-annotated data, which ultimately hinders the scalability of this approach.

Unsupervised methods, in contrast, typically try to pre-train the models to teach them to discover the semantic parts and objects within an image instead, and do not require additional annotated samples. Examples of unsupervised methods for monocular depth estimation include vid2depth [28]. This approach considers the 3D geometric structure of the scene and enforces consistency in estimated 3D point clouds and ego-motion on consecutive frames. Another work along this direction is from Zhang *et al.* [46], who also proposed a new unsupervised hybrid geometric-refined loss to explicitly explore the precise geometric relationship between the input color image and the predicted depth map.

A recent notable approach called DepthCLIP [48] has gained significant attention. It leverages the text-image correlation capabilities of CLIP [31] to enable zero-shot depth prediction. In this work, we follow DepthCLIP [48] to further explore the potential of CLIP in depth estimation tasks.

3. Method

In this section, we present our proposed method of learning to adapt CLIP for few-shot monocular depth estimation in detail.

3.1. Contrastive Language-Image Pre-Training

The effectiveness of CLIP [31] has been demonstrated in the multi-modal task of image-text matching. By leveraging contrastive learning methods, CLIP trains powerful text encoder and image encoder using a training dataset of 400 million text-image pairs. Without the need for any training on new datasets, text and image features can be obtained separately by text encoder and image encoder, and the features are simply interacted with by dot product to obtain the image-text similarity, then the model can accurately select the text that best fits the input image from a number of texts based on the similarity.

To address the downstream task of depth estimation, we can leverage few-shot methods to extract information from limited data samples and conduct model fine-tuning. When constructing the model, we explore two main directions: In the text perspective, what kind of text is best understood by the model, how to better extract text features, and how to make the text features help the image task to the maximum extent. In the image perspective, how to extract local and global information of an image and combine them with text to be applied to the downstream task of depth estimation.

3.2. Method Overview

In Figure 2, we present an overview of our proposed method for few-shot depth estimation. The training process

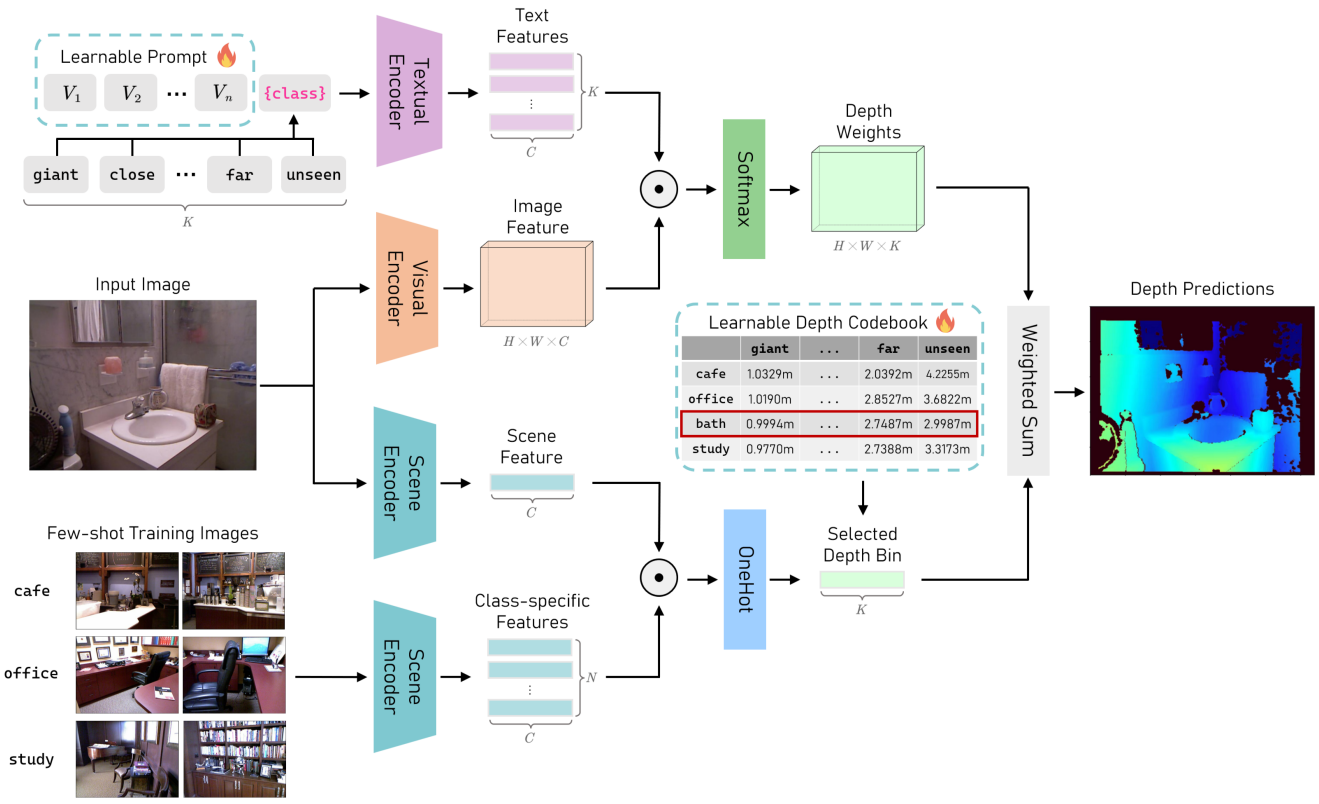

Figure 2. **Overview of our proposed method.** We incorporate learnable prompts and learnable depth codebooks to adapt the CLIP model on few-shot samples for better depth estimation. Three different encoders are included in this work. The textual encoder and scene encoder are directly from the original CLIP [31] model with ResNet-50 backbone. The visual encoder is the CLIP's ResNet-50 image encoder but without the final pooling layer. The fire icon means the parameters will be updated during few-shot training.

of our few-shot-based method requires one image from each scene category as a training sample. Preceding the training phase, we fetch all scene samples into the scene encoder to obtain a set of class-specific features. Simultaneously, we input all depth category descriptors into the text encoder to acquire a set of textual features.

During the training phase, the input image will be processed through the visual encoder and scene encoder, yielding image feature $\mathbf{F}_v$ through patch stitching and scene feature $\mathbf{F}_s$ through encoding the entire image. Subsequently, the cosine similarity is computed between the image feature and the set of textual features to estimate the depth category for each patch. The resulting similarity scores are then normalized to generate the depth estimation weight. Likewise, the scene feature and the set of scene features undergo cosine similarity calculations, enabling the identification of the most similar scene within the scene set for the current input image. This identified scene is then utilized to select the corresponding depth bin from the depth codebook using the one-hot coding approach. The depth estimation weight is then suitably weighted and merged with the depth bin to obtain the final depth values on a patch-by-patch basis. The depth of all pixels within each patch is equal to the depth of that patch.

3.3. Learnable Prompt

The prompt text for manually designed inputs in CLIP is characterized by textual features, while in DepthCLIP [48], the textual input takes the form of "This object is `[Depth CLASS]`." We artificially classify the continuous variable of depth into `[Depth CLASS]` and map it to discrete values. Specifically, the `[Depth CLASS]` includes "giant", "extremely close", "close", "not in distance", "a little remote", "far", and "unseen" with corresponding values that can be mapped onto numerical measurements. For example, "unseen" may represent a distance of 5 m.

Although text prompts designed for human comprehension conform to English semantic and grammatical rules, they are often not intuitive for the model. Consequently, in our method, we employed few-shot training techniques to convert these human-friendly words ("This", "object", "is") into computationally-friendly vectors, denoted as $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n]$, where n is a hyperparameter, which represents the number of vectors after converting words into vectors, the number of vectors does not need to match that of words. $\mathbf{v}_i$ is a row vector, representing a word token. To

ensure the model's sensibility toward various depth classes in the textual input, we adopt a unified text encoder, which takes the form of "`[Vector] + [Class]`". Suppose we have K depth classes and those K classes are represented by K tokens $\mathbf{m}_1, \cdots, \mathbf{m}_K$. This approach facilitates model comprehension and processing of textual features in these prompts.

$$\mathbf{T} = \begin{bmatrix} \mathbf{V} & \mathbf{m}_1 \\ \mathbf{V} & \mathbf{m}_2 \\ \vdots & \vdots \\ \mathbf{V} & \mathbf{m}_K \end{bmatrix}. \tag{1}$$

All these prompts can be transformed into text features $\mathbf{F}^{\star}_{t,i}$ via a text encoder:

$$\mathbf{F}^{\star}_{t,i} = \mathbf{\Phi}_t(\mathbf{T}_i), \tag{2}$$

where text features $\mathbf{F}^{\star}_{t,i} \in \mathbb{R}^{1\times C}$, $i = 1, \cdots, K$, $\mathbf{\Phi}_t$ represents the text encoder, it is directly from CLIP model, and T_i is the i-th row of $\mathbf{T}$, C is the length of the feature.

In the training stage, the prompt vectors can be updated by the root-mean-square error of the depth estimation result with the ground truth, given by

$$\mathcal{L} = \sqrt{\frac{1}{hw}\sum_{i=1}^{h}\sum_{j=1}^{w}(\hat{d} - d_{g.t.})^2}, \tag{3}$$

where h,w are given by the original size of the input image, $\hat{d}$ and $d_{g.t.}$ denote predicted depth and ground truth depth value for each pixel, respectively.

3.4. Learnable Depth Codebook

The depth codebook $\mathbf{\Theta}$ is the key to convert this discrete value of classification results into depth values. Our sample is composed by taking one image from each class. Accordingly, each scene is given a depth bin in our codebook.

For the scene images $\mathbf{S}_j$ used for the few-shot, they are first fed into the scene encoder $\mathbf{\Phi}_s$ to obtain the class-specific features

$$\mathbf{F}^{\star}_{s,j} = \mathbf{\Phi}_s(\mathbf{S}_j). \tag{4}$$

The scene encoder is directly from the CLIP's ResNet-50 visual backbone. The class-specific features $\mathbf{F}^{\star}_{s,j} \in \mathbb{R}^{1\times C}$, $j = 1, ..., N$, where N is the number of scenes in the depth-codebook $\mathbf{\Theta}$.

When the image $\mathbf{I}$ is input in, $\mathbf{I}$ went through the scene encoder to get the input scene features $\mathbf{F}_s$.

$$\mathbf{F}_s = \mathbf{\Phi}_s(\mathbf{I}), \tag{5}$$

where $\mathbf{F}_s \in \mathbb{R}^{1\times C}$.

Consequently, we compute the similarity between the scene features of the input image and the class-specific features, facilitating the identification of the most appropriate depth bin within the codebook $\mathbf{\Theta}$ through a one-hot encoding technique. The similarity of the input image $\mathbf{I}$ to the scene image s^s_j is calculated by

$$s^s_j = \frac{\mathbf{F}_s {\mathbf{F}^{\star}_{s,j}}^{\top}}{\parallel \mathbf{F}_s \parallel \parallel \mathbf{F}^{\star}_{s,j} \parallel}. \tag{6}$$

The evaluated scene class $j^{\star}$ of the image $\mathbf{I}$ is expressed as

$$j^{\star} = \arg\max_j s^s_j. \tag{7}$$

Then the selected depth bin is the $j^{\star}$-th row of $\mathbf{\Theta}$:

$$\boldsymbol{\theta}^{\star} = \mathbf{\Theta}_{j^{\star}}. \tag{8}$$

The codebook $\mathbf{\Theta}$ is assigned an initial value and subsequently updated throughout the training process by the RMSE loss given in Equation (3).

3.5. Depth Prediction

Upon completing the tasks related to the image scene analysis and input text pre-processing, we proceed to promote the interaction between the input text and the input image. For the input image $\mathbf{I}$, we feed it into the image encoder without the pooling layer to obtain the image feature $\mathbf{F}_v$:

$$\mathbf{F}_v = \mathbf{\Phi}_v(\mathbf{I}), \tag{9}$$

where $\mathbf{F}_v \in \mathbb{R}^{H\times W\times C}$ and $\mathbf{\Phi}_v$ denotes the visual encoder, HW is the number of image patches and C is the feature space dimension. Here, the visual encoder is slightly different from the scene encoder. It is also from the ResNet50-CLIP but without the final pooling layer.

Then, we calculate the cosine similarity between $\mathbf{F}^{\star}_{t,i}$ and $\mathbf{F}_v$ to get its similarity score

$$s^v_i = \frac{\mathbf{F}^{\star}_{t,i}\mathbf{F}_v^{\top}}{\parallel \mathbf{F}^{\star}_{t,i} \parallel \parallel \mathbf{F}_v \parallel}. \tag{10}$$

The similarity score vector $\mathbf{s}^v = [s^v_1, \cdots, s^v_K]$ is softmaxed to get the depth weight $\mathbf{d}$:

$$\mathbf{d} = \mathrm{Softmax}(\mathbf{s}^v). \tag{11}$$

Depth weight $\mathbf{d} \in \mathbb{R}^{H\times W\times K}$, $H \times W$ is the number of image patches and K is the number of depth description class. The dimensionality of such a tensor means that for each of all $H\times W$ patches, its probability distribution across the K depth categories is provided.

At this juncture, we have optimized and consolidated the accessible textual data, integrated and leveraged the global and local image information, and derived the depth weights and depth boxes. Ultimately, a straightforward weighted summation suffices to acquire the depth estimation d for each patch

$$\hat{d} = \boldsymbol{\theta}^{\star}\mathbf{d}^{\top}. \tag{12}$$

The depth estimation result obtained here is based on patches, and the depth of each pixel in each patch is equal to the value of the depth estimation result of the patch.

4. Experiments

In this section, we present performance comparisons with state-of-the-art methods on monocular depth estimation, and ablation studies to demonstrate the effectiveness of our proposed method.

4.1. Datasets

NYU V2 [38] is a dataset for indoor scene depth estimation. It consists of 1,449 RGB images and corresponding depth maps for indoor scenes. The resolution of each image is 480 $\times$ 640, and the depth range for each pixel is 0-10m. The training set covers 36,253 images from 249 scenes, while the testing set includes 654 images from the remaining 215 scenes. To remove frames, all samples are cropped to a resolution of 416 $\times$ 512 pixels.

KITTI [12] is a large scale outdoor dataset with a resolution of 375 $\times$ 1242 pixels. It consists of RGB and depth image pairs captured by cameras and depth sensors in an autonomous driving car. For a fair comparison, we adopt the split strategy of ASTransformer [4]. The training set is composed of 23,488 from 32 scenes while the testing set contains 697 images from the remaining 29 scenes. The training and testing samples are cropped to a resolution of 352 $\times$ 704 pixels.

4.2. Implementation Details

We used PyTorch to build the model framework. Our method is built upon the pre-trained ResNet-50 CLIP model. For the learnable prompts, we directly initialize them randomly. We selected one image per scene category to participate in the few-shot training. For the NYU V2 [38] dataset, we set the initial dimension of the depth codebook as 27 $\times$ 7. Specifically, there are 27 scene categories to choose from, and the depth of each scene is manually classified into 7 categories: [‘giant’, ‘extremely close’, ‘close’, ‘not in distance’, ‘a little remote’, ‘far’, ‘unseen’], with the same initial depth values assigned to these descriptive words: [1.00, 1.50, 2.00, 2.25, 2.50, 2.75, 3.00]. A depth bin contains a set of depth values, and for each scene class, we construct a class-dependent depth bin. Our depth codebook contains 27 class-dependent depth bins, which will be updated in subsequent training.

In our experiments, the learning rate is set to 0.5 for the prompt training and 0.01 for the depth codebook training, a decay factor of 1×10^{-5} is used in both cases to prevent overfitting. After training for 200 epochs, we obtained the current results. All the experiments are conducted on a single NVIDIA RTX 3090 GPU.

4.3. Evaluation Metrics

We evaluated the effectiveness of the method using four common metrics, which are divided into accuracy metrics and error metrics. The error metrics include the mean absolute relative error (MARE), mean squared relative error (MSRE), absolute error in log space (AELS), and root mean square error (RMSE). These error metrics we used can be computed by

$$\mathrm{MARE} = \frac{1}{N}\sum_{i=1}^{N}\frac{(|d_i - \hat{d}_i|)^2}{d_i}, \tag{13}$$

$$\mathrm{MSRE} = \frac{1}{N}\sum_{i=1}^{N}\frac{|d_i - \hat{d}_i|}{d_i}, \tag{14}$$

$$\mathrm{AELS} = \frac{1}{N}\sum_{i=1}^{N}|\log(d_i) - \log(\hat{d}_i)|, \tag{15}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(d_i - \hat{d}_i)^2}. \tag{16}$$

Threshold accuracy is a binary classification metric used in depth estimation tasks. It measures the probability of correct classification of depth estimation algorithms after binarizing the true depth values. We selected three thresholds of 1.25, 1.25^2 and 1.25^3 for threshold accuracy calculation:

$$\delta_{th} = \frac{1}{N}\sum_{i=1}^{N}[\max(\frac{d_i}{\hat{d}_i}, \frac{\hat{d}_i}{d_i}) < \delta], \delta = 1.25, 1.25^2, 1.25^3, \tag{17}$$

where N is the number of samples, d_i represents the ground truth depth, and $\hat{d}_i$ represents the predicted depth.

4.4. Performance Results

In Table 1, we compare our results with other monocular depth estimation methods on NYU dataset. The upper half part is the prediction result of the supervised learning method, while the lower half part includes the performance results of the previous zero-shot depth estimation approach. The results obtained by our method is presented in the last row. It is trained based on a few-shot setting which randomly selects one picture per category. The lower bound row is obtained by making random predictions for each pixel within the depth ranging from 0 to 10 meters. Our few-shot-based method has notable improvement compared to the original zero-shot DepthCLIP [48], exceeding the lower bound by a wider margin. And the adapted method has advantages in all metrics over other zero-shot transferring methods which have been pre-trained on specific datasets for monocular depth estimation (unsupervised KITTI video [12]). Moreover, our method could obtain fairly performance of some fully-supervised methods like Make3D [35], even surpassing it in some metrics. It has considerable accuracy under $\delta < 1.25^3$ and MARE deviation, along with notably lower RMSE of 1.049 (compared

Method	Pre-training	Supervision	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$	MARE $\downarrow$	AELS $\downarrow$	RMSE $\downarrow$
Make3D [35]	-	full	0.447	0.745	0.897	0.349	-	1.214
DORN [10]	-	full	0.828	0.965	0.992	0.115	0.051	0.509
ASTransformer [4]	-	full	0.902	0.985	0.997	0.103	0.044	0.374
DepthFormer [25]	-	full	0.921	0.989	**0.998**	0.096	0.041	0.339
RPSF [29]	-	full	**0.952**	**0.989**	0.997	**0.072**	**0.029**	**0.267**
LORN [23]	ImageNet-1k [6]	few-shot†	0.703	0.923	0.979	1.008	0.222	-
Lower Bound	-	-	0.140	0.297	0.471	1.327	0.323	2.934
vid2depth [28]	KITTI video [12]	0-shot	0.268	0.507	0.695	0.572	-	1.637
Zhang *et al.* [46]	KITTI video [12]	0-shot	0.350	0.617	0.799	0.513	0.529	1.457
DepthCLIP [48]	CLIP [31]	0-shot	0.394	0.683	0.851	0.388	0.156	1.167
Ours	CLIP [31]	1-shot	**0.428**	**0.732**	**0.898**	**0.347**	**0.140**	**1.049**

Table 1. **Performance comparisons of our proposed method and previous state-of-the-art methods on the NYU V2 dataset [38].** Lower bound is obtained by randomly making predication for each pixel within depth range 0-10m. We report the results obtained by previous state-of-the-art on fully-supervised and zero/few-shot settings. †LORN uses 200 images and 2,500 partial images for training. Note that our method uses only one image per scene for better estimation.

Method	Pre-training	Supervision	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$	MARE $\downarrow$	MSRE $\downarrow$	RMSE $\downarrow$	AELS $\downarrow$
DORN [10]	-	full	0.932	0.984	0.994	0.072	0.307	2.727	0.120
ASTransformer [4]	-	full	0.963	0.995	0.999	0.103	-	-	0.374
DepthFormer [25]	-	full	**0.975**	**0.997**	**0.999**	**0.052**	**0.158**	**2.143**	**0.079**
DepthCLIP [48]	CLIP [31]	0-shot	0.281	0.531	0.696	0.473	6.007	12.958	0.680
Ours	CLIP [31]	1-shot	**0.312**	**0.569**	**0.739**	**0.384**	**4.661**	**12.290**	**0.632**

Table 2. **Performance comparisons of our proposed method and previous state-of-the-art methods on the KITTI dataset [12].** The depth range on this task is 0-80m. We manually fine-tune the depth bin values for DepthCLIP [48] to obtain these results. Note that in our method, the depth bins are learnable, so we only need to initialize the depth bins randomly. We only use 27 images to train our model.

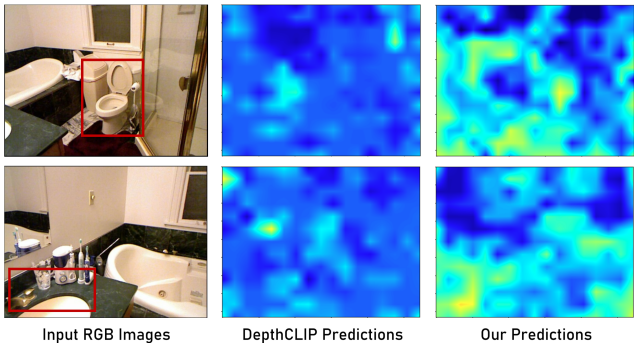

Figure 3. **Qualitative comparisons on the NYU V2 dataset [38].** We present the depth estimation results obtained by DepthCLIP [48] and our proposed methods on three images to show the effectiveness of our proposed method.

with 1.214 of Make3D [35] and 1.167 of DepthCLIP [48]). These extensive experimental results demonstrate the effectiveness of our few-shot approach in adapting the vision-language models for monocular depth estimation.

In Table 2, we compare our results with other monocular depth estimation methods on KITTI dataset. Following the same principle of comparison of NYU dataset, our method still outperforms DepthCLIP by a substantial margin.

4.5. Qualitative Comparisons

In Figure 3, we present qualitative comparisons of our proposed method and DepthCLIP [48] method on the NYU V2 dataset [38]. We include three images for performance comparisons. We can see that, our method leads to better depth predictions than DepthCLIP [48] in all three images, especially within the red rectangular regions. These results indicate that our approach effectively leverages few-shot training to adapt vision-language models, resulting in improved depth predictions.

4.6. Ablation Studies

To systematically evaluate our proposed method, we provide an empirical analysis of our design choices in this section. In Table 3, we present the ablation study results. All the experiments are conducted on the NYU V2 dataset [38]. Our method has two major new components, namely learnable prompt and learnable depth codebook introduced in Section 3. In the first row of Table 3, we report the results of the baseline method (DepthCLIP [48]). From the table, we can see that each algorithm component has a significant contribution to the overall performance.

Learnable prompt design. In the previous vision-

Method	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$	MARE $\downarrow$	RMSE $\downarrow$	AELS $\downarrow$
Baseline	0.394	0.683	0.851	0.388	0.156	1.167
Baseline + Learnable Prompt	0.429	0.711	0.878	0.386	0.145	1.076
Baseline + Learnable Prompt + Learnable Depth Bin	**0.433**	0.715	0.884	0.378	0.143	1.066
Baseline + Learnable Prompt + Learnable Depth Codebook	0.428	**0.732**	**0.898**	**0.347**	**0.140**	**1.049**

Table 3. **Impact of different algorithm components.** We report the performances with different design choices. 'Learnable Depth Bin' means to learn a single class-independent depth bin. Our depth codebook is designed to learn a class-dependent depth bin for each scene.

Patch size	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$	MARE $\downarrow$	RMSE $\downarrow$	AELS $\downarrow$
8×8	0.336	0.617	0.806	0.379	0.179	1.300
16×16	0.354	0.630	0.817	0.514	0.172	1.229
32×32	**0.428**	**0.732**	**0.898**	**0.347**	**0.140**	**1.049**

Table 4. **Effects of different patch sizes.** In our experiments, we set the patch size to 32×32.

language models, the prompt is consisted of a grammatically correct English sentence, it helps fulfill the text input which was then passed through a text encoder to obtain the associated text features. In the context of the depth estimation task, the sentence "This object is `[Depth CLASS]`" is utilized [48]. To optimize this, the words in the sentence are transformed into a vector that can be learned, and this vector is subsequently used to replace the input language sentence, forming a concatenated vector. We call this learnable prompts. The second row shows results with prompt tuning, it shows that with learnable prompt setting, our model could be better adapted to the training set. Note that, the addition of learnable prompts improves the despth estimation performance considerably.

Learnable depth codebook design. To enhance the model's generalization ability, our method introduces a class-dependent depth codebook. After obtaining the scene class by one-hot coding, we attach a specific learnable depth bin to each semantic language token. Then the selected bin would be combined with depth weights to obtain the final prediction. In other words, we have mapped the same semantic token to different class-dependent depth bin according to scene category. As we can see in Table 3, the third row shows results with both learnable prompt and one learnable class-independent depth bin for all scenes. And the last row shows the final results, replacing the single depth bin with a scene-adapted depth codebook. we could notice the obvious improvement in all metrics except the percentage of $\delta < 1.25$. This shows the superiority of our learnable depth codebook.

Patch size. The patch size is related to the selected backbone, which is ResNet-50 in our experiments. Specifically, after feature extraction by CLIP's image encoder, the the obtained feature dimensions are the original size of the input image divided by 32. This is determined by the internal structure of ResNet-50 and the fixed parameters of the CLIP model. So in our experiments, we naturally set the patch size to 32. To ablate the effects of patch size, we conduct experiments with patch sizes 8 and 16 by using the middle layer features of the encoder as the extracted image features. The results are shown in Table 4. The results indicate that the patch size of 32 leads to the best performance.

Method	Supervision	Time	Params.	RMSE $\downarrow$
DORN [10]	full	$>$ 30 h	51M	0.509
DepthCLIP [48]	0-shot	0	0	1.147
Ours	1-shot	70 min	8K	1.049

Table 5. **Trade-of between efficiency and estimation accuracy.** We report the training time and number of parameters to be updated of fully-supervised, zero-shot, and our few-shot methods. We conduct the experiment on one single RTX 3090 GPU.

Trade-of between efficiency and estimation accuracy. Fully-supervised methods demand a significant amount of training time to achieve satisfactory accuracy, whereas zero-shot approaches can obtain a basic level of accuracy without any training. Our method aims to strike a balance between training efficiency and generalization performance. Table 5 provides a comparative analysis of training time, parameter count, and RMSE across different methods, further illustrating this balance. For instance, DORN [10] has 51M parameters and need to be trained by all taining data. The zero-shot DepthCLIP [48] incurs no training cost in terms of time and parameters but yields the highest RMSE. In contrast, our approach only updates 8K parameters by training on a mere 27 images for 70 minutes, and yet achieves a lower RMSE.

5. Conclusion

In this paper, we propose a few-shot-based method that aims to adapt CLIP for monocular depth estimation. This approach focuses on striking a balance between training costs and generalization ability. Specifically, it assigns varying depth bin to different scenes, allowing the model to select the appropriate bins during inference. Furthermore, we introduce a learnable prompt to preprocess the input text, facilitating the conversion of easily understandable human text into vectors that are readily interpreted by the model. Remarkably, with only one image per scene for training, our extensive experiments on the NYU V2 and KITTI dataset demonstrate that our method surpasses the previous state-of-the-art approach by up to 10.6%.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *NeurIPS*, volume 35, pages 23716–23736, 2022. 3

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, volume 33, pages 1877–1901, 2020. 3

[3] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In *AAAI*, volume 33, pages 8001–8008, 2019. 3

[4] Wenjie Chang, Yueyi Zhang, and Zhiwei Xiong. Transformer-based monocular depth estimation with attention supervision. In *BMVC*, 2021. 6, 7

[5] Guangyi Chen, Weiran Yao, Xiangchen Song, Xinyue Li, Yongming Rao, and Kun Zhang. Prompt learning with optimal transport for vision-language models. In *ICLR*, 2023. 3

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 7

[7] Karan Desai and Justin Johnson. Virtex: Learning visual representations from textual annotations. In *CVPR*, pages 11162–11173, 2021. 3

[8] Jiali Duan, Liqun Chen, Son Tran, Jinyu Yang, Yi Xu, Belinda Zeng, and Trishul Chilimbi. Multi-modal alignment using representation codebook. In *CVPR*, pages 15651–15660, 2022. 3

[9] Shuai Fang, Ren Jin, and Yang Cao. Fast depth estimation from single image using structured forest. In *ICIP*, pages 4022–4026. IEEE, 2016. 2

[10] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, pages 2002–2011, 2018. 2, 7, 8

[11] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *IJCV*, pages 1–15, 2023. 3

[12] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 6, 7

[13] Lluis Gomez, Yash Patel, Marçal Rusinol, Dimosthenis Karatzas, and CV Jawahar. Self-supervised learning of visual features through embedding images into text topic spaces. In *CVPR*, pages 4230–4239, 2017. 3

[14] Xinyu Huang, Youcai Zhang, Ying Cheng, Weiwei Tian, Ruiwei Zhao, Rui Feng, Yuejie Zhang, Yaqian Li, Yandong Guo, and Xiaobo Zhang. Idea: Increasing text diversity via online multi-label recognition for vision-language pre-training. In *ACM MM*, pages 4573–4583, 2022. 3

[15] Philip TG Jackson, Amir Atapour Abarghouei, Stephen Bonner, Toby P Breckon, and Boguslaw Obara. Style augmentation: data augmentation via style randomization. In *CVPRW*, volume 6, pages 10–11, 2019. 3

[16] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, pages 4904–4916, 2021. 1, 3

[17] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *ECCV*, 2022. 3

[18] Riccardo La Grassa, Ignazio Gallo, Cristina Re, Gabriele Cremonese, Nicola Landro, Claudio Pernechele, Emanuele Simioni, and Mattia Gatti. An adversarial generative network designed for high-resolution monocular depth estimation from 2d hirise images of mars. *Remote Sensing*, 14(18):4619, 2022. 3

[19] Jie Lei, Linjie Li, Luowei Zhou, Zhe Gan, Tamara L Berg, Mohit Bansal, and Jingjing Liu. Less is more: Clipbert for video-and-language learning via sparse sampling. In *CVPR*, pages 7331–7341, 2021. 3

[20] Jimmy Lei Ba, Kevin Swersky, Sanja Fidler, et al. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *CVPR*, pages 4247–4255, 2015. 3

[21] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *EMNLP*, pages 3045–3059, 2021. 3

[22] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *CVPR*, pages 10965–10975, 2022. 2, 3

[23] Shuai Li, Jiaying Shi, Wenfeng Song, Aimin Hao, and Hong Qin. Few-shot learning for monocular depth estimation based on local object relationship. In *ICTAI*, pages 1221–1228. IEEE, 2019. 7

[24] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL*, pages 4582–4597, 2021. 3

[25] Zhenyu Li, Zehui Chen, Xianming Liu, and Junjun Jiang. Depthformer: Exploiting long-range correlation and local information for accurate monocular depth estimation. *Machine Intelligence Research*, pages 1–18, 2023. 3, 7

[26] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, volume 32, 2019. 3

[27] Yuning Lu, Jianzhuang Liu, Yonggang Zhang, Yajing Liu, and Xinmei Tian. Prompt distribution learning. In *CVPR*, pages 5206–5215, 2022. 3

[28] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In *CVPR*, pages 5667–5675, 2018. 3, 7

[29] Mazen Mel, Muhammad Siddiqui, and Pietro Zanuttigh. End-to-end learning for joint depth and image reconstruction from diffracted rotation. *arXiv preprint arXiv:2204.07076*, 2022. 7

[30] Yue Ming, Xuyang Meng, Chunxiao Fan, and Hui Yu. Deep learning for monocular depth estimation: A review. *Neurocomputing*, 438:14–33, 2021. 2

[31] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 1, 3, 4, 7

[32] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019. 3

[33] Anirban Roy and Sinisa Todorovic. Monocular depth estimation using neural regression forest. In *CVPR*, pages 5506–5514, 2016. 2

[34] Mert Bulent Sariyildiz, Julien Perez, and Diane Larlus. Learning visual representations with caption annotations. In *ECCV*, pages 153–170, 2020. 3

[35] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE TPAMI*, 31(5):824–840, 2008. 6, 7

[36] Timo Schick and Hinrich Schütze. Exploiting cloze questions for few shot text classification and natural language inference. In *EACL*, pages 255–269, 2021. 3

[37] Timo Schick and Hinrich Schütze. It's not just size that matters: Small language models are also few-shot learners. In *NAACL*, pages 2339–2352, 2021. 3

[38] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. *ECCV*, 7576:746–760, 2012. 2, 6, 7

[39] Yushun Tang, Qinghai Guo, and Zhihai He. Cross-inferential networks for source-free unsupervised domain adaptation. In *ICIP*, pages 96–100. IEEE, 2023. 2

[40] Yushun Tang, Ce Zhang, Heng Xu, Shuoshuo Chen, Jie Cheng, Luziwei Leng, Qinghai Guo, and Zhihai He. Neuromodulated hebbian learning for fully test-time adaptation. In *CVPR*, pages 3728–3738, 2023. 2

[41] Fabio Tosi, Filippo Aleotti, Matteo Poggi, and Stefano Mattoccia. Learning monocular depth estimation infusing traditional stereo knowledge. In *CVPR*, pages 9799–9809, 2019. 2

[42] Feng Wang, Manling Li, Xudong Lin, Hairong Lv, Alexander G Schwing, and Heng Ji. Learning to decompose visual features with latent textual prompts. In *ICLR*, 2023. 3

[43] Ning Wang, Jiangrong Xie, Hang Luo, Qinglin Cheng, Jihao Wu, Mingbo Jia, and Linlin Li. Efficient image captioning for edge devices. In *AAAI*, volume 37, pages 2608–2616, 2023. 2

[44] Yuan Yao, Ao Zhang, Zhengyan Zhang, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Cpt: Colorful prompt tuning for pre-trained vision-language models. *arXiv preprint arXiv:2109.11797*, 2021. 3

[45] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *TMLR*, 2022. 3

[46] Mingliang Zhang, Xinchen Ye, Xin Fan, and Wei Zhong. Unsupervised depth estimation from monocular videos with hybrid geometric-refined loss and contextual attention. *Neurocomputing*, 379:250–261, 2020. 3, 7

[47] Renrui Zhang, Longtian Qiu, Wei Zhang, and Ziyao Zeng. Vt-clip: Enhancing vision-language models with visual-guided texts. *arXiv preprint arXiv:2112.02399*, 2021. 3

[48] Renrui Zhang, Ziyao Zeng, Ziyu Guo, and Yafeng Li. Can language understand depth? In *ACM MM*, pages 6868–6874, 2022. 1, 2, 3, 4, 6, 7, 8

[49] Yi Zhang, Ce Zhang, Xueting Hu, and Zhihai He. Unsupervised prototype adapter for vision-language models. In *PRCV*, 2023. 2

[50] Yi Zhang, Ce Zhang, Zihan Liao, Yushun Tang, and Zhihai He. Bdc-adapter: Brownian distance covariance for better vision-language reasoning. In *BMVC*, 2023. 2

[51] Yi Zhang, Ce Zhang, Yushun Tang, and Zhihai He. Cross-modal concept learning and inference for vision-language models. *arXiv preprint arXiv:2307.15460*, 2023. 3

[52] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *CVPR*, pages 16816–16825, 2022. 3

[53] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *IJCV*, 130(9):2337–2348, 2022. 2, 3

[54] Mingyang Zhou, Licheng Yu, Amanpreet Singh, Mengjiao Wang, Zhou Yu, and Ning Zhang. Unsupervised vision-and-language pre-training via retrieval-based multi-granular alignment. In *CVPR*, pages 16485–16494, 2022. 3