IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 30, 2022, accepted February 8, 2022, date of publication February 11, 2022, date of current version March 2, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3151108

Learning Depth Estimation From Memory Infusing Monocular Cues: A Generalization Prediction Approach

YAKUN ZHOU[1,2], JINTING LUO[1], MUSEN HU[1], (Student Member, IEEE), TINGYONG WU[1], (Member, IEEE), JINKUAN ZHU[1], XINGZHONG XIONG[2], (Member, IEEE), AND JIENAN CHEN[1], (Senior Member, IEEE)

[1]National Key Laboratory of Science and Technology on Communications, University of Electronic Science and Technology of China, Chengdu 611731, China
[2]School of Automation and Information Engineering, Sichuan University of Science and Engineering, Yibin 643002, China

Corresponding author: Tingyong Wu (wuty75@uestc.edu.cn)

This work was supported in part by the Artificial Intelligence Key Laboratory of Sichuan Province under Grant 2020RZJ02, and in part by the Sichuan Science and Technology Program under Grant 2021YFG0127.

ABSTRACT Depth estimation from a single image is a challenging task, yet this field has a promising prospect in automatic driving and augmented reality. However, the prediction accuracy is degraded significantly when the trained network is transferred from the training dataset to real scenarios. To solve this issue, we propose MonoMeMa, a novel deep architecture based on the human monocular cue, which means humans can perceive depth information with one eye through the relative size of objects, light and shadow, etc. based on previous visual experience. Our method simulates the process of the formation and utilization of human monocular visual memory, including three steps: Firstly, MonoMeMa perceives and extracts real-world objects feature vectors (encoding). Then, it maintains and replaces the extracted feature vector over time (storing). Finally, MonoMeMa combines query objects feature vectors and memory to inference depth information (retrieving). According to the simulation results, our model shows the state-of-the-art results on the KITTI driving dataset. Moreover, MonoMema exhibits remarkable generalization performance when our model is migrated to other driving datasets without any finetune.

INDEX TERMS Long short-term memory (LSTM), monocular depth estimiation, multi-layer perceptron (MLP), region proposal network (RPN).

I. INTRODUCTION

Obtaining accurate depth from images is one of the most important tasks in computer vision. In recent years the depth estimation has attracted a wide range of applications in many fields such as automatic driving, robot navigation, 3D depth reconstruction and augmented reality. Although LIDAR technology is quite popular, attaining depth from images is worth more preference. Compared with LIDAR sensors, using a camera to collect depth information has several potential advantages: cheap, easy to be installed, and adaptive in various environments.

The popular solution of visual depth prediction so far is the stereo estimation, which infers disparity (i.e., the inverse of depth) using two or more cameras from different points of view. However, these binocular approaches are limited by the problems of calibration error and synchronization. Therefore, currently predicting depth from a single image has become a hot area in depth estimation.

Monocular depth estimation is a very challenging work as the image is the projection of the 3-D scene, while the projection only captures the 2-D information. Different from the binocular depth estimation methods, monocular depth estimation regards predicting depth as a regression problem and focuses on finding a relationship between the pixel value and the depth value [1]. To achieve this goal, early methods [1]–[4] use techniques in machine learning to build monocular depth estimation models. With the development of deep learning in recent years, monocular depth estimation approaches [5]–[8] based on deep neural networks become popular. Based on these studies [5], [7], [9], experimental results obtained by monocular depth estimation exhibit excellent performance, which indicates that the deep neural

The associate editor coordinating the review of this manuscript and approving it for publication was Felix Albu.

FIGURE 1. Overview of the proposed MonoMeMa. Input image from KITTI dataset (top) [10]. Predicted results by our MonoMeMa (bottom).

networks are suitable for the task of mapping the pixel value with to depth values.

However, the critical challenge is the generalization task performance. Although the deep learning methods display excellent results on a single dataset such as KITTI [10], they rarely show the equally good performance in the generalization tasks where the input data has various aspect ratios, different camera settings, and distinctive vehicle poses.

Therefore, the approaches based on deep neural networks generally lack generalization ability. Although the literature [11] develop tools that enable mixing multiple datasets with incompatible annotations, it is unrealistic and impractical to obtain datasets of all scenes in the real world [12]. So the generalization ability is important to support the modern intelligent applications, such as automatic driving tasks.

Human beings can predict the depth of pictures taken by various cameras with different configurations. That is to say, people can estimate the object depth from an image, even without the pre-knowledge of camera specifications. The reason is shown in [13], humans perform well at monocular depth estimation by exploiting monocular cues such as perspective, scaling relative to the known size of familiar objects. To perceive depth in new scenes, humans utilize the monocular cues by comparing the size of the unfamiliar objects with the size of the familiar objects which are memorized before. It is precise because we have formed a rich and structural understanding of the world through the past visual experience, so we humans can model the real-world scenes well [14].

Inspired by the monocular cues in human depth perception, in this work, we propose Monocular Memory Matching (MonoMeMa) architecture to estimate object depth from a single image based on monocular cues. In the first stage, we use an encoder to perceive and extract real-world objects feature vectors. Then we utilize the extract feature vectors to search for empirical information stored in the memory storage, which stores monocular cues extracted from past experiences such as size, type of the objects and depth labels (The storage maintains and replaces the extracted feature vector over time). Finally, we use these matching information obtained from the memory and a decoding network to inference the depth.

(1) An external memory storage:The memory can simulate humans past visual experience and store monocular cues for target objects in order to restore depth in new scenes.
(2) An encoder-decoder architecture: The architecture cooperates with the external storage to restore the depth. It can extract the feature of target objects from a single picture and send it to the decoder to restore the depth of the object through the combination of similar past experience information in the external storage.
(3) A novel memory storage control mechanism:The control mechanism can determine whether the current training data is valuable for future prediction tasks, and learn to write as little information as possible while maintaining considerable accuracy.

II. RELATED WORK

In this section, we review the literature relevant to our work concerned with stereo and monocular depth estimation approaches.

A. STEREO DEPTH ESTIMATION

Given a pair of rectified stereo images, the goal of stero depth estimation is to compute the disparity d for each pixel in the reference image. Disparity refers to the difference in horizontal location of a pixel in the left and right image — a pixel at position (x, y) in the left image appears at position $(x - d, y)$ in the right image. Then the depth of this pixel is calculated by $\frac{f*B}{d}$, where f is the camera's focal length and B is the distance between two camera centers.

Most conventional dense stereo algorithms calculate disparity based on the four steps summarized by [15]. These methods rely on 2-frame stereo correspondence and are organized by matching cost computation, cost support aggregation, disparity computation and optimization, or disparity refinement. Current state-of-the-art studies focus on how to compute the matching cost accurately and how to refine the disparity map. With the rapid development of deep learning, convolutional neural networks (CNNs) have been applied to learn how to match corresponding points, and a deep network trained to match 9×9 image patches was shown by [16] to produce then state-of-the-art results. [17] regards the correspondence problem as a multi-scale task and propose a notably faster Siamese network. [18] design a deep network to compute disparity from the images. A popular and effective approximation to refine the disparity map is the Semi-Global Matching (SGM) of [19], where dynamic programming optimizes a pathwise form of the energy function in many directions.

Recently, end-to-end networks have been developed to predict whole disparity maps without post-processing. Mayer *et al.* [20] created a large synthetic dataset to train an end-to-end network for disparity estimation (DispNet) and optical flow (FlowNet), improving the state-of-the-art.Kendall *et al.* [21] introduce GC-Net, an end-to-end network to efficiently learn context in the disparity cost

volume using 3-D convolutions. Chen [22] proposed a novel pyramid stereo matching network (PSMNet) to exploit global context information in stereo matching. PSMNet use Spatial pyramid pooling (SPP) [23] and dilated convolution [24] to enlarge the receptive fields and improve the utilization of global context information.

The methods above rely on a large amount of ground truth disparity data. However, image pairs they use for training are hard to obtain in the real world; calibration errors and synchronization problems can also reduce the accuracy of the training data.

B. MONOCULAR DEPTH ESTIMATION

Monocular depth estimation refers to the problem setup where only a single image is available at test time. Before the deep learning era, some monocular depth estimation methods [1]–[4] are based on machine learning techniques. Saxena et al. [1] treat the task of recovering depth from pixels as a regression problem. They utilize Markov Random Field (MRF) and some hand-designed multi-scale texture features to incorporates multiscale local and global image features, modeling both depths at individual points as well as the relation between depth at different points. With the increasing availability of the ground truth data, supervised approaches outperform the previous works.

Eigen *et al.* [5] propose a model employing two deep network stacks where one makes a coarse global prediction based on the entire image and another that refines this prediction locally. Liu *et al.* [7] present a deep convolutional neural field model for estimating depths from single monocular images and design a deep structured learning scheme which learns the unary and pairwise potentials of continuous conditional random field (CRF) in a unified deep CNN framework to avoid hand-crafted features. Li *et al.* [25] combines deep learning features on image patches with hierarchical CRFs defined on a superpixel segmentation of the image. Work by Laina *et al.* [26] models the ambiguous mapping between monocular images and depth maps using a fully convolutional architecture encompassing residual learning. They also introduce the reverse Huber loss that is particularly suited for the tasks driven by the value distributions commonly presenting in depth maps. Some methods combine depth map prediction with semantic segmentation, Ladick *et al.* [27] simplify the deep prediction to a classification problem and proposed a new pixel-wise classifier, that can jointly predict a semantic class and a depth label from a single image. Liu et al's method [28] semantic segmentation of the scene and then use the semantic labels to guide the 3D reconstruction.

Recently, some unsupervised depth estimation methods have also been proposed. Compared to general supervised learning, these methods do not need to use vast amounts of manually labelled data to train them. Garg *et al.* [29] propose a stereopsis based encoder-decoder architecture,which predicts depth by training on an image reconstruction loss. *Zhou et al.* [14] present an unsupervised learning framework for the task of monocular depth and camera motion estimation from unstructured video sequences. Guo *et al.* [30] propose a framework that can make full use of Cross-domain synthetic data, which uses the stereo matching networks as a proxy to learn depth from synthetic data, and uses predicted stereo disparity maps to supervise training monocular depth estimation networks. [31] makes monocular camera move in an unknown indoor environment acquiring continuous images sequences. Depth estimation and object detection is respectively implemented through FCN and Faster RCNN.

Finally, mostly related to our work is the work by J. Konrad *et al.* [32]. Their approaches regard the depth estimation task as a matching problem. Instead of relying on a deterministic scene model for the input 2D image, they propose to ''learn'' the model from a large dictionary of stereo pairs such as YouTube 3D. Based on the assumption that two stereo pairs whose left images are photometrically similar are likely to have similar disparity fields, they predict the depth information by matching the input image with the stereo-pair images in the dictionary. Inspired by their work, we design a network with memory where the memory contains past useful experiences collected during the training step. Thus, our model can predict the depth depending on the feedback obtained by inquiring the input image feature from memory.

III. MONOCULAR MEMORY MATCHING

In this section, we describe the detailed MonoMeMa architecture designed to infer accurate depth estimation for an object in a supervised manner from a single image. We begin with the encoder-decoder structure of our model, and then depict the memory control strategy used to accumulate useful memories. Finally, we present our training loss in each part of our model. Figure2 shows an overview of our framework, depicting an input frame and the outcome of MonoMeMa.

A. MODEL ARCHITECTURE

Our model focuses on the depth of the specific target objects instead of the pixel depth as the object depth estimation is more useful and practical for the real-world application like auxiliary driving, where detecting the critical objects such as cars, pedestrians and obtaining their depth is an efficient way to parsing the scenes.

The structure of the proposed MonoMeMa is shown in Figure 2, which consists of an object encoder and a depth decoder and memory storage.

The purpose of our object detection network is to detect and encode specific targets such as vehicles from an image. The object encoder consists of a feature extractor network based on CNNs, and a proposal network (RPN) proposed in [33], which enables the object detection by the regression of bounding boxes [33], [34] and the non-maximum suppression (NMS). Firstly, the image is processed by convolution and pooling layers to obtain the feature map. Then the encoder will extract a fixed-length feature vector from the feature map by using the ROI pooling layer. After that, each feature vector is fed into two parallel output layers. One layer is to perform eight classifications (Car, Van, Truck, Pedestrian,

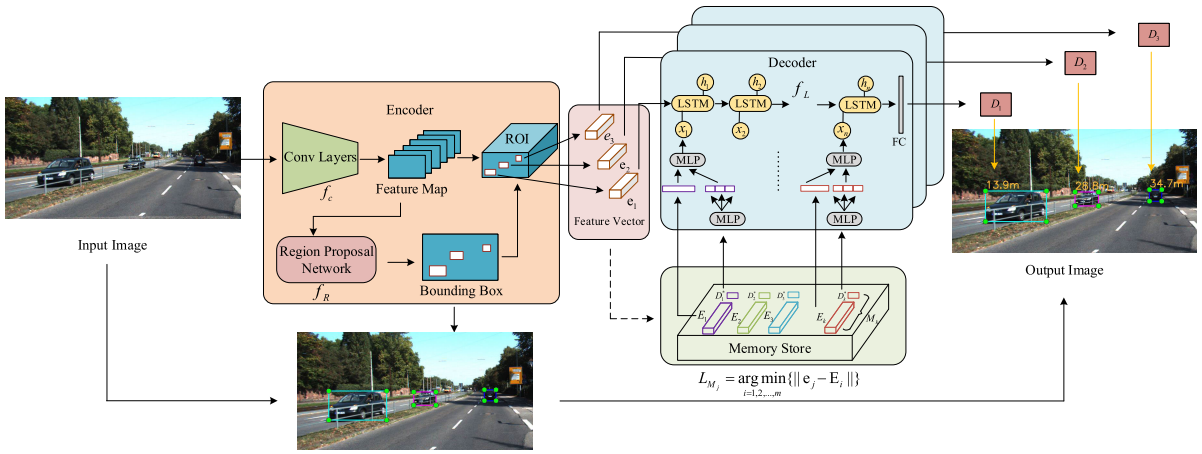


FIGURE 2. The whole network structure of our model. From left to right: input image, object encoder, depth decoder, memory and output image.

Person_sitting, Cyclist, Tram, Misc), and output the probability distribution of each RoI for the target on eight object classes. The other layer outputs four real values (Bounding-Box regression) for each of the eight objects. The other layer outputs four real values (Bounding-Box regression) for each of the ten objects. Consequently, these real values represent the bounding-box locations for each class.

Inspired by the monocular depth cues, we design an external memory storage that collects the high-dimensional knowledge of the depth cues viewed in the training process such as the class, the size, and the distance of different samples. By matching the target object with historical objects in memory, the external memory storage proposes the memory vectors and their ground truth depth labels.

After doing this, the decoder predicts the depth based on the vectors and labels obtained from both the object encoder network and the memory. Since there are multiple target objects in an image, we make predictions for each query feature vector separately, and we choose the recursive decoder LSTM that shares weights as the model decoder. For a specific query vector, we can use the KNN algorithm to find multiple similar historical feature vectors from the Memory Store. We use the MLP network to map the true depth labels corresponding to the historical vector from low dimensions to high dimensions. Then the high-dimensional vector and the memory feature vector are stitched, and the new vector is used as the input of each time step of the LSTM. The LSTM is followed by a fully connected network to output the depth prediction for the target object.

Our design requires that the decoder must be able to recover the depth of the target object from memory. Recursive decoder such as LSTM has the internal memory and the hidden activation is similar to the register. The decoder can mix information across multiple time steps of operation and select different weights for different memories to recover the depth information of the query vector. We also compared other decoders in the Ablation Study. Several more complex feed-forward neural network decoders and LSTM comparisons are also mentioned in the [35] to prove the performance of LSTM.

Suppose that the capacity of the memory is m, which has been obtained during the training process. Each memory segment M_i involves two elements, the feature vector E_i and the corresponding depth label L_i, so that the set $M = \{E_i, D_i^* | i = 1, 2, \ldots, m\}$. The depth label can be obtained from the disparity value and the camera parameters provided in the dataset.

The goal of the inference process is to obtain the depth of each specific object from the given image I. Assuming that the CNN function f_C, the RPN function f_R and the LSTM function f_L have been optimized during the training process. In the object encoder network, the extracted feature vectors $\mathbf{e} = \{e_1, e_2, \ldots, e_N\}$ (N represents the number of the specific objects in a single image) are obtained by four steps as shown in Figure 2 : firstly, a feature map F of the input raw image I is computed by the CNNs as $F = f_c(I)$; then we use the RPN network [33] to predict the boundaries B_j and classes C_j (background or foreground), which are given as $\{B_j, C_j\} = f_R(F)$; after that, we use boundary and class information to extract the region of interests (ROIs) R_j and reshape them to a uniform size through a pooling layer; finally, the feature vectors e_j are obtained by stretching the ROIs to one-dimensional vectors as $e_j = MLP\left(R_j\right)$, $j = 1, 2 \ldots N$.

The decoder in Figure 2 shows that the depth decoding step for each feature vector involves two parts: (1) searching for the matching memory vectors and (2) decoding through LSTM. We use the Euclidean distance to measure the similarity between the jth query feature vector and the ith memory vector, and the similarity is given as $d_i^j = \left|\left|e_j - E_i\right|\right|$. Based on the similarity, we select the first k vectors with the smallest Euclidean distance as the output of the memory store. By the way, k denotes the output data size of memory each time and the value of k depends on the LSTM size. We will give the detail value in section V(C). In the search and match phase

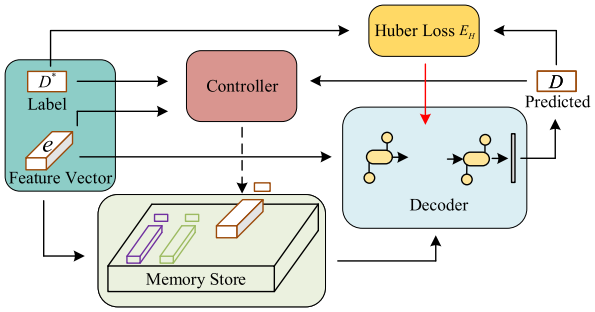


FIGURE 3. Data Flow in Memory. This figure shows the data flow in the memory where e is the feature vector, D is the predicted depth and D^* is the ground truth label. In the training process, the feature vector obtains the prediction depth through the decoder. On the one hand, the prediction depth and label are used to calculate the Huber Loss. On the other hand, the prediction depth and truth value are sent to the control module to decide whether to store the vector.

our output is $M_i^j = \{E_i, D_i^*\}, i = 1, 2, \ldots, k$. Next, We use the MLP network to map the true depth labels from low dimensions to high dimensions $MLP(D_i^*)$, and then stitching the high-dimensional vector and the memory eigenvector to get the new vector $x_i = MLP(E_i, MLP(D_i^*)), i = 1, 2, \ldots, k$. This new vector is used as the input of each time step of the LSTM.

Finally, we can obtain the predicted depth for the jth object as

$$D_j = f_F(f_L\left(e_j, x_i | i = 1, 2, \ldots, k\right)), \tag{1}$$

where f_F and f_L represent full connection layer and LSTM respectively.

To summarize, our model consists of two main parts: an object encoder extracts the depth features of the specific object, and a depth decoder network recovers the depth.

B. MEMORY CONTROL

In order to make our model work more efficiently, the memory should learn to write as little information as possible while maintaining considerable accuracy. To this aim, we design a memory control strategy that is used to judge whether the present training data is valuable for the later predicting missions. Based on such two assumptions: (1) a model will be improved by the labels of the data that is not accurately predicted (we call this data as valuable information) and (2) there is no need for a model to store the data that is already precisely predicted (we call this data as valueless information), we expect the memory which can provide our model with valuable information. When the predictions are quite precise for present training data, we choose to skip them. When our model shows a bad performance on the present training data, we assume this data contains information that our model didn't learn well before. Hence, we write such data into the memory to help with further predictions.

Based on these considerations, we design a controller to measure the value of each training data and set a memory control threshold ζ_a to adjust the capacity of the memory. In the controller, we calculate the Absolute Relative Error as $\sigma_a = \frac{|D^* - D|}{D^*}$, where D is the predicted depth and D^* is the ground truth label. If $\sigma_a > \zeta_a$, it means that there is an unacceptable difference between the predicted depth and the ground truth so that this valuable vector should be added to the memory. This will enable the stored ground truth labels to greatly assist the decoder in correcting predictions the next time a similar query vector is encountered.On the contrary, if $\sigma_a < \zeta_a$, it indicates our model has already learned it well, we only need to use the existing memory and decoder to predict the depth well, so there is no need to store it and its labels. By changing the value of ζ_a in the Ablation Study, we can not only show a positive effect of memory on depth estimation tasks but also display the robustness of our work. Figure 3 shows a data flow during the training step. Noticeably, the backpropagation is only realized in the process marked by the red line and the process marked by the black line works only in the forward calculation. It means that the selection process of the memory in the training step does not need backpropagation, which reduces the time complexity of the calculation.

C. LOSS FUNCTION

As our model consists of two main networks, the loss function is also composed of the object encoder loss E_O and the depth decoder loss E_H.

1) OBJECT ENCODER LOSS

This part measures the model's ability to demarcate specific objects from a single image. It consists of a regression loss in finding the bounding boxes and a classification loss caused by recognizing whether it's foreground or background. The object encoder loss E_O is given as

$$E_o = \frac{1}{N_{cls}} \sum_i E_{cls}(C_i, C_i^*) + \tau \frac{1}{N_{bbox}} \sum_i C_i^* E_{bbox}(B_i, B_i^*), \tag{2}$$

where i is the index of a bounding box, N_{cls} represents the batch size, and N_{bbox} represents the number of the bounding boxes. C_i and C_i^* represent the predicted class and the corresponding label where 1 stands for foreground and 0 stands for the background. B_i and B_i^* are the four-dimensional predicted bounding boxes and their labels. $E_{bbox} = R\left(B_i - B_i^*\right)$ where R is the robust loss proposed in [34] and E_{cls} is cross-entropy function. τ is a constant factor set for weighting these two parts. The complete reference can be find in [33] equation 1.

2) DEPTH DECODER LOSS

This part measures the model's ability to decode depth from feature vectors. In order to compensate for the inaccuracy brought by the small difference between predictions and the ground truth or the large distribution of distance in the dataset, we adapt the Huber loss [36] that is particularly suited for

the task [26].

$$E_H = \begin{cases} |D^* - D|, & |D^* - D| < c \\ \dfrac{(D^* - D)^2 + c^2}{2c}, & |D^* - D| > c, \end{cases} \tag{3}$$

where D and D^* are the predicted depth and the corresponding labels, c is a constant.

IV. EXPERIMENT

In this section, we describe the datasets, implementation details, metrics, and then present exhaustive evaluations of MonoMeMa on various training/testing configurations, showing that our method outperforms supervised state-of-the-art approaches. As standard in our experiment, we assess the performance of monocular depth estimation techniques following the protocol by Eigen *et al.* [5], extracting data from the KITTI [10] dataset, CityScapes [37] dataset and ApolloScape [38] dataset. By the way, we extract the depth labels in 2D layer and the label depth is obtained from the center pixel points. So that the depth value will not disturb by the background regions. Additionally, we also perform an exhaustive ablation study proving that the decoder based on LSTM and memory enables our strategy to improve the predicted depth accuracy. Additionally, we also perform an exhaustive ablation study proving that the decoder based on LSTM and memory enables our strategy to improve the predicted depth accuracy.

A. DATASETS

1) KITTI

The KITTI dataset [10] is a collection of several outdoor scenes concerning driving scenarios. It consists of 61 scenes that contain about 42382 stereo frames. The standard image size is 1242 × 375 pixels. Each image contains up to 15 vehicles and 30 pedestrians with different degrees of occlusion. The LIDAR device measures the depth information. Since the encoder needs to be trained, we chose the pictures in the KITTI dataset that contain the detection box labels and this dataset provides the true depth labels for the target objects. So we can directly use the detection box labels and the true depth labels to train the encoder and decoder. We split 7481 images from the object detection dataset of KITTI and select 6058 of them for the training set, 674 of them for validating set, and 749 of them for the test set.

2) CITYSCAPES

The CitysScapes dataset [37] includes stereo pairs (contains about 22973 frames) covering 50 cities in Germany captured by a moving vehicle in various weather conditions. Its standard image shape is 2084× 1024 pixels. In our experiment, we select 1525 from the 5000 images with fine annotations in three main cities as the test set for the generalization task. Note that the CitysScapes dataset does not provide target detection box labels and disparity maps, but we only perform generalization tasks for depth prediction on this data set, so we do not need target detection box labels. First we use the SGM algorithm to calculate the disparity depth map of the Cityscapes dataset as the ground truth. Then use the trained encoder to find the target objects in the original image. Finally, we calculate the depth label of the target objects by average pooling the depth map with the target objects mask.

FIGURE 4. Results on KITTI [10]. The predicted depth (yellow) is on the upper side of the box and the ground truth depth (red) is on the lower side of the box.

3) APOLLOSCAPE

The ApolloScape dataset [38] is a large-scale dataset for autonomous driving. It's composed of 140K images with a shape of 3130 × 960 pixels in three Chinese cities captured in various traffic conditions, the number of moving objects averages from tens to over one hundred. We select 1000 images in four distinctive regions from two cities as the test set for the generalization task. The method of obtaining the depth labels of the ApolloScape dataset is the same as the previous one.

B. IMPLEMENTATION DETAILS

The network which is implemented in Tensorflow [39] contains 132 million trainable variables (126 million for the encoder and 6.5 million for the decoder) and takes around 20 hours (16 hours for training the object encoder and 4 hours for decoder) using a single GTX1080 GPU on the dataset of 6 thousand images. The inference takes less than $160ms$, or more than 6 frames per second, for a 1242×375 image, including transfer times to and from the GPU. Figure 4 shows the results of the KITTI dataset. Please see our code for more details.

During training, we set the learning rate for the object encoder to $\alpha_{obj} = 1e^{-3}$and the learning rate in decoder to $\alpha_{dep} = 1e^{-4}$. As for the memory capacity, we set the max capacity to $m = 100$ and the threshold to $\zeta_a = 0.1$. When calculating metrics, we regard the object as the minimal calculating unit. For example, 2926 objects with corresponding depth are generated by our model from 749 images on the KITTI testing set. Thus, we assess the average performance in objects instead of images. Moreover, for a fair comparison with other methods, we use the bounding boxes obtained from the object encoder to split the depth maps obtained in other methods and calculate the object depth by average pooling [40]. Generally, the depth labels provided in datasets are disparities. Thus, we obtain the depth maps using the formula $D = b * f/d$, where D is depth, b is the baseline, f represents focal of the camera and d represents disparity.

C. EVALUATION METRICS

The main evaluation indicators of the object detection network is mean Average Precision(mAP). And in the depth estimation task we evaluate our model using the metrics proposed in prior works [5].

$$\text{RMSE}: \sqrt{\frac{1}{N}\sum_{i=1}^{N} ||\rho(x_i) - g(x_i)||_2^2},$$

$$\text{RMSE (log)}: \sqrt{\frac{1}{N}\sum_{i=1}^{N} ||\log(\rho(x_i)) - \log(g(x_i))||_2^2},$$

$$\text{Abs Rel}: \frac{1}{N}\sum_{i=1}^{N} \frac{|\rho(x_i) - g(x_i)|}{g(x_i)},$$

$$\text{Sq Rel}: \frac{1}{N}\sum_{i=1}^{N} \frac{|\rho(x_i) - g(x_i)|^2}{g(x_i)},$$

$$\text{Accuracy}: \frac{\sum_{i=1}^{N} |max\left(\frac{\rho(x_i)}{g(x_i)}, \frac{g(x_i)}{\rho(x_i)}\right) = \delta < thr|}{N},$$

$$\text{Maximum Relative Error}: \quad \delta = \max\left(\frac{\rho(x_i)}{g(x_i)}, \frac{g(x_i)}{\rho(x_i)}\right),$$

where x_i represents the index of the objects, $\rho(x_i)$ represents the predicted depth, $g(x_i)$represents the ground truth depth and N is the number of objects.

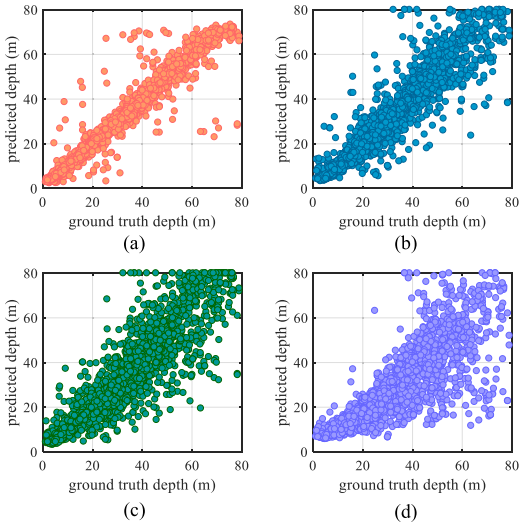


FIGURE 5. Scatter map of statistical comparison with state-of-art methods. (a) Ours. (b) Tosi *et al.* [41]. (c) Godard *et al.* [6]. (d) Ibraheem Alhashim *et al.* [8].

V. RESULTS

A. COMPARISON WITH THE STATE-OF-THE-ART

In this section, we compare our framework with state-of-the-art approaches for monocular depth estimation on the KITTI dataset. For a fair competition, we adopt the code that formally published in their GitHub and evaluate the metrics using pre-trained models provided by the authors. Figure 5 shows the statistical results of our model comparing with others. We can see the predicted depth value of our model converges well on the diagonal from $1m$ to $80m$, but traditional pixel-by-pixel prediction depth while other methods tend to obviously diverge in the long-distance prediction, which indicates that our algorithm outperforms all other existing methods especially in estimating the depth of the objects that are far from the camera.

FIGURE 6. Results on generalization tasks. (Left) Results on CityScapes [37]. (Right) Results on ApolloScape [38]. The predicted depth (yellow) is on the upper side of the boxes and the ground truth depth (red) is on the lower side of the boxes.

TABLE 1. Comparison with the state-of-the-art methods on the KITTI dataset. K represents KITTI dataset with maximum depth set to 80m.

				Lower is better		Higher is better			
Method	Dataset	Abs Rel	Sq Rel	RMSE	RMSE(log)	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	mAP
[8]	K	0.234	2.497	10.016	0.374	0.616	0.840	0.926	
[6]	K	0.184	1.747	7.659	0.266	0.743	0.923	0.967	
[42]	K	0.150	1.234	6.132	0.225	0.840	0.955	0.978	
[43]	K	0.164	1.447	6.432	0.243	0.813	0.933	0.971	
ours	**K**	**0.086**	**0.578**	**4.219**	**0.164**	**0.936**	**0.972**	**0.986**	**0.586**

TABLE 2. Comparison with other approaches on generalization tasks on CityScapes and ApolloScape.

				Lower is better		Higher is better			
Method	Dataset	Abs Rel	Sq Rel	RMSE	RMSE(log)	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	
[8]	C	0.250	2.154	7.412	0.281	0.592	0.895	0.977	
[6]	C	0.281	4.181	14.391	0.412	0.412	0.718	0.902	
[42]	C	0.295	3.833	13.178	0.418	0.246	0.719	0.919	
[43]	C	0.283	4.579	15.711	0.396	0.449	0.750	0.904	
ours	**C**	**0.157**	**1.405**	**8.319**	**0.212**	**0.754**	**0.960**	**0.992**	
[8]	A	0.263	2.370	9.373	0.374	0.518	0.804	0.923	
[6]	A	0.222	2.830	9.861	0.361	0.615	0.836	0.912	
[42]	A	0.179	1.491	7.324	0.274	0.708	0.909	0.963	
[43]	A	0.190	1.884	8.469	0.292	0.671	0.885	0.954	
ours	**A**	**0.138**	**0.920**	**5.872**	**0.222**	**0.838**	**0.951**	**0.978**	

The mAP measures the accuracy of our method in the feature extraction stage. And Table 1 also shows quantitative results evaluated on all kinds of metrics. We can observe from the table that in terms of Abs Rel, our method surpasses [41] by 42%, [6]by nearly 53%, [8]by 63% and [42] by 58%. This proves that the depth estimation ability of our model in supervised learning tasks on the KITTI dataset is obviously better than other methods.

B. GENERALIZATION TO OTHER DATASETS

To illustrate that our model can efficiently generalize to other datasets, we compare our approach with several methods on the CityScapes and ApolloScapes datasets. Traditional methods generally need finetune when generalizing to other datasets, but this is not appropriate in real scenes, because we are difficult to capture a large number of real ground datasets. Compared with these methods, thanks to the memory module brings powerful depth clues and LSTM modules in the coding phase act as valid and reasoned elements to strengthen the capability of the network to take advantage of what has been internally stored, we can see from the table that without any finetune, our approach outperforms our competitors in the generalization task. Figure 6 shows qualitative results on both of the datasets. A quantitative comparison is also displayed in Table 2. The numerical result in Abs Rel exceeds 37% and 47% of work by Ibrahhem [8], 44% and 38% of work by Goard [6], 47%, 23% of work by Fabio [41] and 44%, 27% of work by Godard *et al.* [42] on CityScapes and ApolloScape respectively. We also note that if we increase one bit of the storage memory, the model performance will double. Although the value of mAP is enough to support significant object detection, there are some scenes that may exist miss or

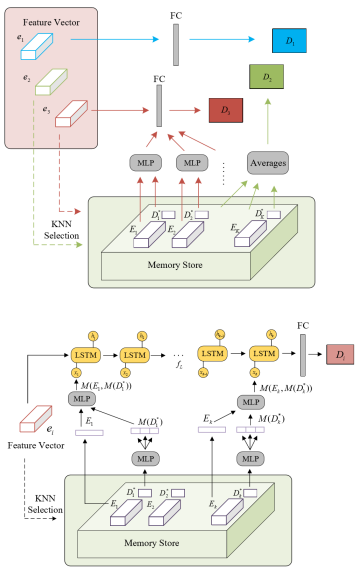


FIGURE 7. Different decoder structures. The figure in the top shows three different decoder structures, among them, blue represents memoryless linear neural network model, green represents parameterless KNN model, and red represents a basic parameter model. The figure in the bottom shows our LSTM decoder structure.

wrong detection. Generally, this characterisitic is denoted by Average Recall(AR). According to our experiment, the value of AR is 0.62. This value approaches the best case of CNN. Besides, we detect objects in the constant process in self-driving. It means that there is always a moment can detect the missed object. If the missed detection occurs frequently, we can reduce the IOU threshold (our IOU value here is 0.5) to obtain more detection bounding box.

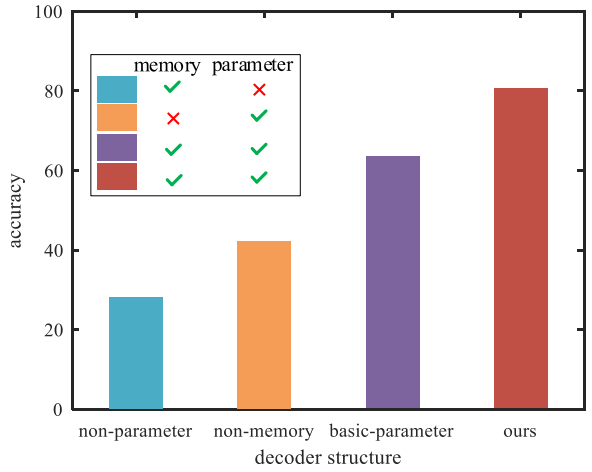


FIGURE 8. Accuracy of the four kinds of decoder structures. For each object, we regard the prediction whose Abs Rel is lower than 0.1 as a correct prediction.

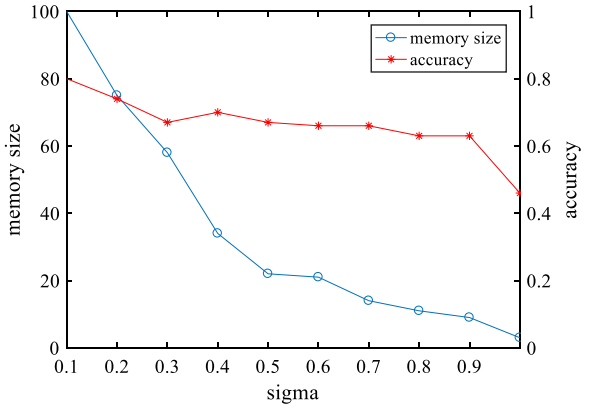


FIGURE 9. Accuracy as a function of examples written to memory. Redline ranges from 0 to 1, it corresponds to the accuracy evaluated on the KITTI dataset. Blueline ranges from 0 to 100, it related to the size of memory.

C. ABLATION STUDY

To verify the function of the LSTM and the memory in the decoder, we replace the depth decoder module with the other three structures: (1) a non-memory model ablating both the LSTM and memory where a linear layer is added behind the object encoder. (2) A non-parameter model which ablates the LSTM and predicts the depth by averaging the depth labels obtained from the k nearest memory vectors. (3) A basic-parameter model ablating the LSTM where we concatenate the k nearest memory vectors and send the concatenated vectors into a fully connected layer. The fully connected layer predicts the depth. Three different decoder structures are shown in Figure 7. The parameters in the encoder is fixed when training the decoder. We have trained encoder before training the decoder.

1) LSTM ANALYSIS

We think the selection of LSTM is important, because the recursive decoder such as LSTM can mix information across multiple time steps of operation and select different weights for different memories to recover the depth information of the query vector. Figure 8 shows the comparison results on KITTI. We can see from the figure that: (1) The basic-parameter model outperforms the non-parameter model, which means that the neural parameters can more effectively analyze the monocular depth cues from the memory and use them for prediction. (2) The basic-parameter model shows lower performance than our model, which means our module contributes to the accuracy by improving the ability to match the present data with the historical information. In other words, our model can better dig out relative size clues and familiar size clues, and can use LSTM to recover depth information more accurately. Ramalho *et al.* [35] compared relational self-attention feed-forward decoder, relational working memory decoder and LSTM decoder for the classification case and found they perform equally well.

2) MEMORY ANALYSIS

One of the key reasons why human can accurately predict the depth is that we have formed a rich understanding of the world through the past visual experience, and stored a large number of past experience. Human memory selection is not random, but more able to remember those failed cases. Similarly, our model pays special attention to the imprecise prediction cases in the training process. We evaluate the contribution that memory makes to our model by ablating the memory capacity. In the training phase, we set the memory capacity from 0 to 100 by adjusting the memory control threshold ζ_a. Figure 9 shows that the accuracy is slightly affected by the decrease of memory size, this indicates our model has good robustness. While the memory size drops below the number of LSTM (in this paper, we set it to 10), the accuracy decreases sharply, it's because the model has no enough prior information used as references. It should be noted that when the memory size decreases to zero, the LSTM model degenerates into the non-memory model (shown in Figure 9), where the prediction only relies on the linear layer.

VI. CONCLUSION

In this paper, we proposed MonoMeMa, a novel framework for monocular depth estimation for specific objects. It combines (1) an object encoder for extracting the object feature and (2) a recursive neural network decoder used to predict the depth based on the memory mechanism. To choose memory efficiently, we also design a memory control strategy that allows the input data that brings additional information to be written in the memory. Noteworthily, our model not only outperforms present approaches in the supervised tasks on the KITTI dataset but also shows a state-of-the-art experimental result in the generalization tasks on CityScapes and ApolloScape datasets. Through exhaustive experiments, we prove that the decoder network based on the LSTM structure is flexible for depth prediction, and the memory leads to a more accurate network. In addition, we think that the task we are considering is novel, which transforms the depth

prediction problem pixel by pixel into the depth estimation for the target object. And we propose an original and innovative strategy to combine object knwoledge and depth estimation with the aim of taking full advantage of their strong connection in the real world.

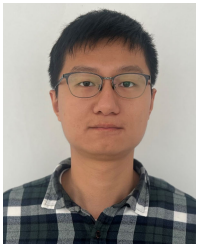
In future work, we will consider migrating our model to the real-time auxiliary driving tasks. In the real world, road scenes vary rapidly. Hence, we expect to decrease the amount of the parameters in our model for higher processing speed. It may be a scheme to realize it by infusing some knowledge in meta-learning and using lightweight object detection networks.

REFERENCES

[1] A. Saxena, S. H. Chung, and A. Y. Ng, "Learning depth from single monocular images," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 1161–1168.
[2] J. Michels, A. Saxena, and A. Y. Ng, "High speed obstacle avoidance using monocular vision and reinforcement learning," in *Proc. 22nd Int. Conf. Mach. Learn.*, 2005, pp. 593–600.
[3] A. Saxena, J. Schulte, and A. Y. Ng, "Depth estimation using monocular and stereo cues," in *Proc. IJCAI*, vol. 7, Jan. 2007, pp. 2197–2203.
[4] A. Saxena, S. H. Chung, and A. Y. Ng, "3-D depth reconstruction from a single still image," *Int. J. Comput. Vis.*, vol. 76, no. 1, pp. 53–69, 2008.
[5] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2366–2374.
[6] C. Godard, O. M. Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 270–279.
[7] F. Liu, C. Shen, G. Lin, and I. Reid, "Learning depth from single monocular images using deep convolutional neural fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 2024–2039, Oct. 2016.
[8] I. Alhashim and P. Wonka, "High quality monocular depth estimation via transfer learning," 2018, *arXiv:1812.11941*.
[9] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2002–2011.
[10] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.
[11] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 3, pp. 1623–1637, Mar. 2022.
[12] A. Carlson, K. A. Skinner, R. Vasudevan, and M. Johnson-Roberson, "Modeling camera effects to improve visual learning from synthetic data," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 1–16.
[13] I. P. Howard, *Perceiving in Depth: Basic Mechanisms*, vol. 1. London, U.K.: Oxford Univ. Press, 2012.
[14] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1851–1858.
[15] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. J. Comput. Vis.*, vol. 47, nos. 1–3, pp. 7–42, Apr. 2002.
[16] J. Žbontar and Y. LeCun, "Stereo matching by training a convolutional neural network to compare image patches," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2287–2318, Oct. 2016.
[17] W. Luo, A. G. Schwing, and R. Urtasun, "Efficient deep learning for stereo matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5695–5703.
[18] Z. Chen, X. Sun, L. Wang, Y. Yu, and C. Huang, "A deep visual correspondence embedding model for stereo matching costs," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 972–980.
[19] H. Hirschmüller, "Accurate and efficient stereo processing by semi-global matching and mutual information," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, pp. 807–814.
[20] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4040–4048.
[21] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry, "End-to-end learning of geometry and context for deep stereo regression," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 66–75.
[22] J.-R. Chang and Y.-S. Chen, "Pyramid stereo matching network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5410–5418.
[23] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2881–2890.
[24] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," 2015, *arXiv:1511.07122*.
[25] B. Li, C. Shen, Y. Dai, A. van den Hengel, and M. He, "Depth and surface normal estimation from monocular images using regression on deep features and hierarchical CRFs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1119–1127.
[26] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *Proc. 4th Int. Conf. 3D Vis. (3DV)*, Oct. 2016, pp. 239–248.
[27] L. Ladicky, J. Shi, and M. Pollefeys, "Pulling things out of perspective," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 89–96.
[28] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1253–1260.
[29] R. Garg, B. G. V. Kumar, G. Carneiro, and I. Reid, "Unsupervised CNN for single view depth estimation: Geometry to the rescue," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham, Switzerland: Springer, 2016, pp. 740–756.
[30] X. Guo, H. Li, S. Yi, J. Ren, and X. Wang, "Learning monocular depth by distilling cross-domain stereo networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 484–500.
[31] C. Hou, X. Zhao, and Y. Lin, "Depth estimation and object detection for monocular semantic SLAM using deep convolutional network," in *Proc. IEEE 20th Int. Conf. Softw. Quality, Rel. Secur. Companion (QRS-C)*, Dec. 2020, pp. 256–263.
[32] J. Konrad, G. Brown, M. Wang, P. Ishwar, C. Wu, and D. Mukherjee, "Automatic 2D-to-3D image conversion using 3D examples from the internet," *Proc. SPIE*, vol. 8288, Feb. 2012, Art. no. 82880F.
[33] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[34] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.
[35] T. Ramalho and M. Garnelo, "Adaptive posterior learning: Few-shot learning with a surprise-based memory module," 2019, *arXiv:1902.02527*.
[36] A. B. Owen, "A robust hybrid of lasso and ridge regression," *Contemp. Math.*, vol. 443, no. 7, pp. 59–72, 2007.
[37] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.
[38] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang, "The ApolloScape dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 954–960.
[39] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," 2016, *arXiv:1603.04467*.
[40] D. Neven, B. D. Brabandere, S. Georgoulis, M. Proesmans, and L. Van Gool, "Fast scene understanding for autonomous driving," 2017, *arXiv:1708.02550*.
[41] F. Tosi, F. Aleotti, M. Poggi, and S. Mattoccia, "Learning monocular depth estimation infusing traditional stereo knowledge," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9799–9809.
[42] C. Godard, O. M. Aodha, M. Firman, and G. Brostow, "Digging into self-supervised monocular depth estimation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3828–3838.

YAKUN ZHOU received the B.E. degree in communication engineering from Yanshan University, Qinhuangdao, China, in 2020. She is currently pursuing the M.S. degree in communications with the National Key Laboratory of Science and Technology, University of Electronic Science and Technology of China. Her current research interests include neural network accelerators and digital VLSI design.

JINTING LUO received the B.E. degree from the School of Information Science and Technology, Dalian Maritime University, in 2019. He is currently pursuing the M.Sc. degree in communications with the National Key Laboratory of Science and Technology, University of Electronic Science and Technology of China. His current research interests include computer vision, deep reinforcement learning, and selfdriving.



MUSEN HU (Student Member, IEEE) received the B.E. degree from the School of Communication Engineering, Ningbo University, in 2021. He is currently pursuing the M.Sc. degree in communications with the National Key Laboratory of Science and Technology, University of Electronic Science and Technology of China. His current research interests include deep reinforcement learning and PCB auto-routing.

TINGYONG WU (Member, IEEE) received the B.E., M.S., and Ph.D. degrees in communication systems from the University of Electronic Science and Technology of China, Chengdu, China, in 1998, 2001, and 2007, respectively. He is currently an Associate Professor with the National Key Laboratory of Science and Technology on Communications, University of Electronic Science and Technology of China. His current research interests include signal processing in wireless communication, circuit-system design, and artificial intelligence for communication.

JINKUAN ZHU received the B.E. degree from the University of Electronic Science and Technology of China, in 2019, where he is currently pursuing the M.Sc. degree with the School of Computer Science. His current research interests include computer vision and deep learning.

XINGZHONG XIONG (Member, IEEE) received the B.S. degree in communication engineering from the Sichuan University of Science and Engineering, Zigong, China, in 1996, and the M.S. and Ph.D. degrees in communication and information system from the University of Electronic Science and Technology of China (UESTC), in 2006 and 2009, respectively. In 2012, he completed a Research Assignment from the Postdoctoral Station of Electronic Science and Technology at UESTC. He is currently a Professor with the School of Automation and Electronic Information, Sichuan University of Science and Engineering. His research interests include wireless and mobile communications technologies, intelligent signal processing, the Internet of Things technologies, and very large-scale integration (VLSI) designs.

JIENAN CHEN (Senior Member, IEEE) received the B.S. and Ph.D. degrees in communication systems from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2007 and 2014, respectively. He was also worked with the School of Electrical and Computer Engineering, University of Minnesota, Minneapolis, as a Visiting Scholar, in 2012 and 2014, and then as a Postdoctoral Scholar with the University of North Texas. He is currently a Professor with the National Key Laboratory of Science and Technology on Communications, UESTC. His current research interests include VLSI circuit designs, low-power circuit designs, stochastic computation-based system designs, machine learning-based signal processing, artificial intelligence for networking, and circuit-system design. He also served as the Symposium Chair for Globalsip and a TPC Member for Globecom and ICC.