Continual Stereo Matching of Continuous Driving Scenes with Growing Architecture

Chenghao Zhang[1,2], Kun Tian[1,2], Bin Fan[3], Gaofeng Meng[1,2,4*], Zhaoxiang Zhang[1,4], Chunhong Pan[1]
[1] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences
[3] School of Automation and Electrical Engineering, University of Science and Technology Beijing
[4] CAS Centre for Artificial Intelligence and Robotics, HK Institute of Science and Innovation
{chenghao.zhang, kun.tian, gfmeng, zxzhang, chpan}@nlpr.ia.ac.cn, {bin.fan}@ieee.org

Abstract

The deep stereo models have achieved state-of-the-art performance on driving scenes, but they suffer from severe performance degradation when tested on unseen scenes. Although recent work has narrowed this performance gap through continuous online adaptation, this setup requires continuous gradient updates at inference and can hardly deal with rapidly changing scenes. To address these challenges, we propose to perform continual stereo matching where a model is tasked to 1) continually learn new scenes, 2) overcome forgetting previously learned scenes, and 3) continuously predict disparities at deployment. We achieve this goal by introducing a Reusable Architecture Growth (RAG) framework. RAG leverages task-specific neural unit search and architecture growth for continual learning of new scenes. During growth, it can maintain high reusability by reusing previous neural units while achieving good performance. A module named Scene Router is further introduced to adaptively select the scene-specific architecture path at inference. Experimental results demonstrate that our method achieves compelling performance in various types of challenging driving scenes.

1. Introduction

The reconstruction of the surrounding 3D scene structure is the foundation of many vision tasks. Depth is the useful precursor to sensing 3D geometry, which is preferentially recovered by well-posed stereo matching due to its simple settings, high accuracy, and acceptable cost. Benefiting from the convolutional neural networks (CNNs), deep stereo methods have achieved remarkable progress recently in driving scenes [3, 5, 6, 14, 45, 48], constantly improving stereo benchmarks like KITTI [11, 26].

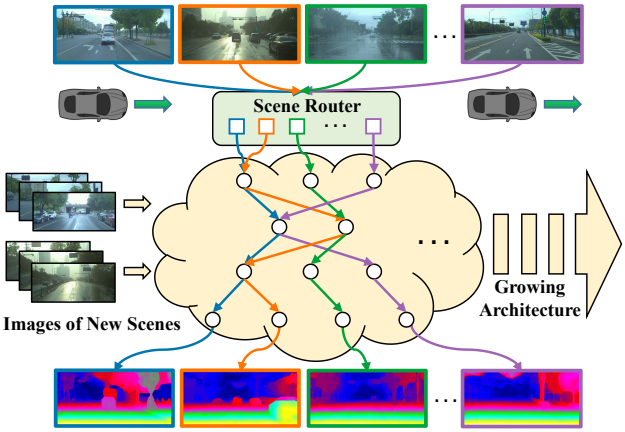


Figure 1. Schematic representation of our framework deployed on real-world continuous driving scenes. The scene-specific architecture path will be loaded for inference according to Scene Router.

However, deep stereo models suffer from performance degradation when deployed to unseen scenarios [41]. This is often caused by the gap between training and testing data domains, *e.g.*, synthetic [25] and real-world data [11, 26]. Domain-adaptive methods [22, 29, 39] can achieve good performance, but they inherently rely on the types of scenes available at training time. Unfortunately, collecting enough data from all kinds of scenes at hand, such as various weather and road conditions in autonomous driving, is quite expensive and unfeasible in reality.

Imagine a car driving in real-world scenarios shown in Fig. 1. The car may experience continuous scenes changing from cloudy to rainy, or from the city to the countryside. A stereo model with a single fixed architecture can hardly perform well on all types of scenes. Moreover, it is also difficult to continue to learn new scenes without forgetting previous knowledge. For optimal performance, an ideal model

*Corresponding author.

should adaptively load suitable architectures for the scenes at inference. The model also needs to avoid the performance drop on previous scenes while learning on the new scenes.

The previous method MADNet [43] utilizes an online learning scheme to continuously adapt to current scenes. The follow-up work MAD++ [31] leverages prior labels to improve the performance while alleviating catastrophic forgetting in stereo. Nevertheless, continuous online gradient updates are required at inference even when the model has adapted well to the current scene, which is computationally unnecessary. Besides, when faced with rapidly changing scenes at inference, MAD++ still needs buffer time to adapt since it cannot leverage previously learned scenes to help. In contrast, we reformulate this problem as a *continual stereo matching* problem. By doing this, the model can *continually* learn to estimate the disparity of new heterogeneous scenes and quickly adapt to rapidly changing *continuous* scenes without online gradient updates at inference.

In this work, we propose a Reusable Architecture Growth (RAG) framework to address the continual stereo problem. RAG can overcome the catastrophic forgetting by freezing the model parameters learned in the previous scenes. Since different scenes vary in color, lighting, and disparity distributions, we assign task-specific neural units for each new scene and adapt the model to them by architecture growth. To obtain a more compact architecture, we explicitly reuse the learned neural units during architecture growth and thereby achieve a balance between model performance and parameter efficiency. Under different weather and road conditions, our method achieves comparable or better performance compared with the state-of-the-art methods. At deployment time, we further propose a module called Scene Router to automatically select the scene-specific architecture path according to the scene type of the input.

Our contributions are summarized as follows:

- We propose a Reusable Architecture Growth framework consisting of task-specific neural units search and architecture growth. The framework can continually learn to estimate the disparity of new scenes without catastrophic forgetting while exhibiting good reusability of the learned neural units.
- A Scene Router module is further introduced to adaptively select the scene-specific architecture path for the current scene at inference. In contrast to continuous adaptation methods [31, 43], our method can quickly adapt to rapid scene switches and is more computationally efficient.
- Experiments demonstrate that our method achieves compelling performance under different challenging weather and road conditions on DrivingStereo [46], KITTI raw [10], and Virtual KITTI [2] datasets.

2. Related Work

Deep Stereo Matching. DispNet [25] is the first end-to-end deep stereo model utilizing a correlation layer to encode matching information. Along this line, the residual learning is exploited [18, 28] to obtain more accurate disparity. Besides, semantic cues [47] and edge information [40] are also incorporated. AANet [45] further boosts the performance through adaptive aggregation. Another category of deep stereo methods represented by GC-Net [14] has focused on building a 4D cost volume and leveraging 3D convolutions to regress disparity. Subsequent improvements in the network structure include PSMNet [3], CSPN [5], and GANet [48]. Other methods [12, 13, 37] explore to construct better regular or cascade cost volume. They achieve remarkable performance on the KITTI benchmarks [11, 26]. LEAStereo [6] further improves the benchmark score through hierarchical neural architecture search.

Adaptive Stereo Matching. It is observed in [41] that deep stereo models pre-trained on synthetic data will suffer from performance degradation when exposed to real-world scenarios. To alleviate the problem, a series of unsupervised domain adaptation methods [22, 29, 39] have made efforts to narrow the domain gap. Apart from the offline learning mechanism, online learning can also be employed like temporal information exploitation [50], meta-learning scheme [42], and modular adaptation [43]. The latter is further improved by prior labels to alleviate catastrophic forgetting [31]. However, online learning requires continuous gradient descent updates at inference, which is computationally extensive. In contrast, our method can perform continuous inference without online gradient updates at deployment while not forgetting the previous scenes.

Neural Architecture Search (NAS). Automatic network design has attracted increasing interest in recent years. Most of the methods targeted on image classification tasks to search top-performing architectures by reinforcement learning [20, 30, 51, 52], evolutionary algorithms [32], and one-shot search [21]. In addition, some recent work attempts to develop NAS to semantic segmentation [19], object detection [4], and other tasks. AutoDispNet [35] first applies NAS to stereo matching by searching cell units of a U-shape architecture. Subsequently, LEAStereo [6] achieves the top performance on several stereo benchmarks through hierarchical NAS. In this study, we leverage NAS to perform task-specific neural unit search and architecture growth to learn the continually incremental scenes.

Continual Learning. Continual learning methods aim to overcome catastrophic forgetting, which are mainly divided into three families [7], $i.e.$, replay, regularization, and parameter isolation. The replay methods store representative samples in memory [33] or construct pseudo-samples by generative models [38]. The regularization-based meth-

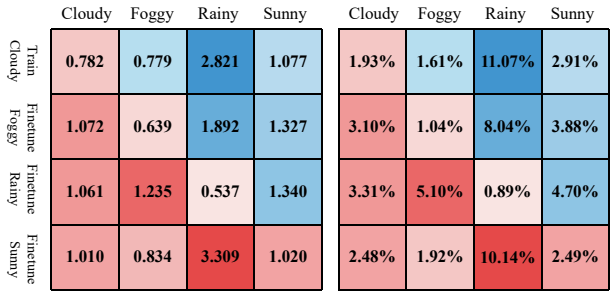


Figure 2. Catastrophic forgetting in stereo. The deep stereo model is first trained on the *cloudy* scene and then sequentially finetuned on each of the other scenes. The red boxes refer to the performance on each scene learned so far, while the blue boxes refer to the poor generalization performance. The EPE (left) and D1-all (right) metrics are used for evaluation. Light colors represent lower error rates. Best viewed in color.

ods [15,17,23] preserve the previous knowledge by adding a regularization term to the loss function without extra memory requirements. Nevertheless, these two families still can not fully maintain the previously learned knowledge. To avoid any possible forgetting, the parameter isolation methods protect the model parameters of previous tasks when learning new tasks. One way is to select a sub-network for each task from a fixed network [8, 24, 36], but the fixed network has a limited capacity for continually increasing tasks. Another way utilizes dynamic architectures [1, 34] by allocating a single model for each task separately, which does not consider the reusability of previous models. Other methods [16,44] selectively expand new units or adapt from old units when learning a new task, which reduces the speed of model expansion to some extent. However, these methods all focus on image classification tasks. In this study, we advance a further step to the dense regression task for continual stereo matching and realize it with the high reusability of the learned neural units.

3. Continual Stereo Matching

3.1. Catastrophic Forgetting in Stereo Macthing

We first demonstrate the catastrophic forgetting phenomenon in stereo, that is, deep stereo models often suffer from serious performance drop on previously learned scenes when adapting to new scenes. To this end, we choose the off-the-shelf state-of-the-art model LEAStereo [6] for our discussions.

For illustration, we construct a task sequence using four kinds of weather conditions of the DrivingStereo [46] dataset, *i.e.*, *cloudy* $\rightarrow$ *foggy* $\rightarrow$ *rainy*$\rightarrow$ *sunny*. The training and validation sets are divided for each scene. We first train the model on the *cloudy* scene and then sequentially finetune it on the other scenes. The evaluation is performed on each scene learned so far and unseen scenes as shown in Fig. 2. We observe that sequential finetuning usually achieves the best performance after finetuning on the current scene, but it performs badly on previously learned scenes or unseen scenes. This is especially obvious after finetuning on the *rainy* scene since it is completely different from the other three scenes in terms of color, lighting, and texture. The experimental results show that catastrophic forgetting is common in heterogeneous driving scenes.

3.2. Problem Statement

The continual stereo problem $\mathcal{T}$ can be formulated as learning N consecutive task sequences $\{\mathcal{T}^1, \mathcal{T}^2, ..., \mathcal{T}^N\}$ with each corresponding to a heterogeneous scene. Let $\Omega = \{\Omega^1, \Omega^2, ..., \Omega^N\}$ denote the data of the N continual scenes, which is divided into training data $\Omega_{(train)}$ and testing data $\Omega_{(test)}$. For each task $\mathcal{T}^t$, the corresponding training data consist of M triples, that is $\Omega^t_{(train)} = \{(I^t_{l_1}, I^t_{r_1}, D^t_{gt_1}), ..., (I^t_{l_M}, I^t_{r_M}, D^t_{gt_M})\}$, where I_l, I_r, and D_{gt} are left image, right image, and the ground truth disparity, respectively. Denote the learning model h^t that contains all the parameters learned up to the current task $\mathcal{T}^t$. The model learns on the tasks from $\mathcal{T}^1$ to $\mathcal{T}^N$ sequentially, and only the training data of the current task $\Omega^t_{(train)}$ can be used. Both $\Omega^t_{(train)}$ and $\Omega^t_{(test)}$ will not be available in the subsequent tasks. The goal of the continual stereo is to continually learn the ability to estimate the disparity of incremental new scenes by *maximizing* the performance of h^t on the current task $\mathcal{T}^t$ while *minimizing* the forgetting for previously learned tasks from $\mathcal{T}^1$ to $\mathcal{T}^{t-1}$. To achieve the maximum goal, we can minimize the following objective function,

$$\mathcal{L}(h^N; \Omega_{(train)}) = \sum\nolimits_{t=1}^{N} \mathcal{L}_t \left(h^t; \Omega^t_{(train)}\right) \quad (1)$$

$$\mathcal{L}_t \left(h^t; \Omega^t_{(train)}\right) = \frac{1}{M} \sum\nolimits_{i=1}^{M} \mathcal{L}_{reg} \left(D^t_{pred_i}, D^t_{gt_i}\right) \quad (2)$$

where $\mathcal{L}_{reg}$ is the smooth l_1 loss of the predicted disparity $D^t_{pred_i} = h^t \left(I^t_{l_i}, I^t_{r_i}\right)$ and the ground truth disparity $D^t_{gt_i}$.

Note that we have no access to all the training data at once, thus Eq. (1) can not be directly optimized. Actually, for the current task $\mathcal{T}^t$, the parameters learned on the previous tasks have been optimized. By freezing the previously learned parameters, we can get an approximate optimization target of optimizing Eq. (2) for the current model h^t while achieving the minimization goal.

4. Method

4.1. Overview

In this study, we propose the Reusable Architecture Growth to better deal with the current task. Following [6],

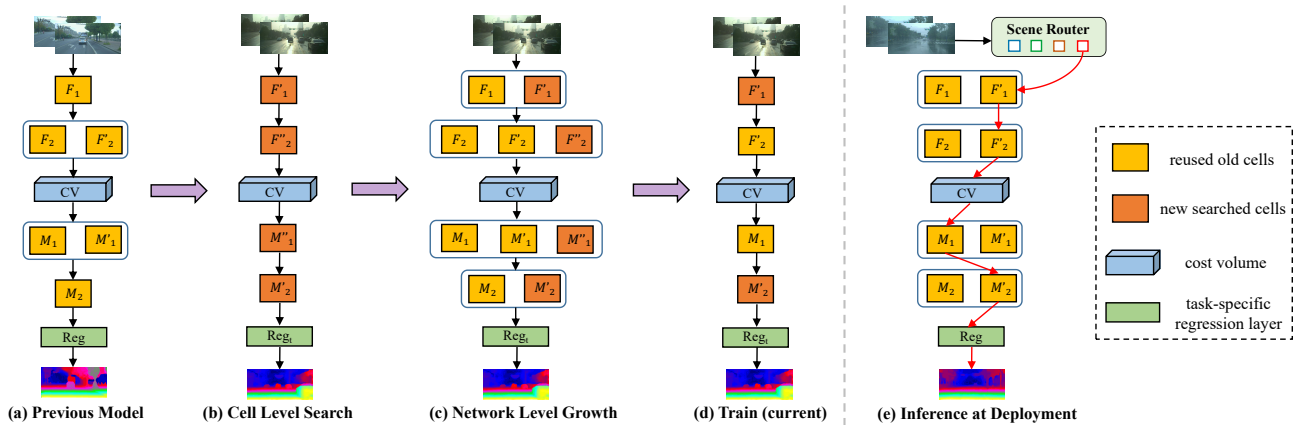


Figure 3. Overview of our Reusable Architecture Growth. For the current task $\mathcal{T}^t$, based on the previous model (a), we first search task-specific neural units of the Feature Net (marked as F) and Matching Net (marked as M) (b), then select suitable units to make the network grow (c), and finally train the selected specific model (d). At test time, the scene-specific architecture path (marked in red) is selected for inference according to the Scene Router (e). Best viewed in color.

our basic model includes Feature Net, cost volume, Matching Net, and disparity regression layers. Fig. 3 shows the overall pipeline of our RAG. For the current task $\mathcal{T}^t$, we first search for task-specific neural units since different scenes have different colors, textures, and disparity distributions. To adapt the model to $\mathcal{T}^t$ without forgetting previous tasks, we then perform architecture growth to determine whether to use the reused old units or the searched new ones for each layer in the network level. The selected new units are retained while the unadopted are deleted. Finally, the model of the current task will be trained. At test time, we further introduce the Scene Router module to adaptively select a specific architecture path to predict the disparity of continuous image data stream. In the following subsections, we will introduce our RAG and Scene Router in detail.

4.2. Reusable Architecture Growth

Cell Level Search. The neural unit in our model is a searchable cell that consists of a fully-connected directed acyclic graph (DAG). Following [6], the cell consists of two input nodes from the preceding two layers, three intermediate nodes, and one output node. Let $\mathcal{O}$ denote a set of candidate operations. The set includes the 3×3 2D convolution and skip connection for Feature Net. For Matching Net, it includes the $3 \times 3 \times 3$ 3D convolution and skip connection. Then each intermediate node s_j can be formulated as $s_j = \sum_{i \sim j} o_{ij}(s_i)$, where $\sim$ indicates that node i is connected to j, and o_{ij} is the operation between them containing $K(=2)$ operations.

For each operation $o_{ij} = \{o_{ij}^1, ..., o_{ij}^K\}$, we allocate a validation score $\Delta = \{\delta_{ij}\}$, an iteration record $C = \{c_{ij}\}$, and a candidate probability $P = \{p_{ij}\}$. Both the validation score Δ and the iteration record C are initialized to zero, and the candidate probability P is uniformly initialized.

To update P, we apply the MdeNAS algorithm in [49] to our cell level search with modifications. Define the selected set of operations after each sampling as m^*. Assuming the error rate (D1-all) on the validation set is σ^* after one epoch, we have the validation score formulated as

$$\delta_{ij}^{m^*} = 1 - \sigma^*. \tag{3}$$

The iteration record correspondingly increases by one.

Intuitively, the operation with fewer iterations and higher validation scores should be preferentially selected, thus getting probability gain. Otherwise it should get probability decay for penalty. Following this point of view, the updating policy of P is as follows:

$$\begin{aligned} p_{ij}^{m^*} = p_{ij}^{m^*} + \alpha\Big(&\sum_{k=1}^{K} \mathbb{I}(c_{ij}^{m^*} < c_{ij}^k, \delta_{ij}^{m^*} > \delta_{ij}^k) - \\ &\sum_{k=1}^{K} \mathbb{I}(c_{ij}^{m^*} > c_{ij}^k, \delta_{ij}^{m^*} < \delta_{ij}^k)\Big), \end{aligned} \tag{4}$$

where $\mathbb{I}(,)$ denotes as the indicator function that equals to one if its condition is true and α is the update momentum. To ensure that the sum of the probabilities is 1, we adopt the softmax operation on the candidate probability as $p_{ij}^{m^*} = \frac{\exp(p_{ij}^{m^*})}{\sum_{k=1}^{K} \exp(p_{ij}^k)}$. Finally, select the operations with the highest probability as the final operations of the cell. Extensive experiments demonstrate that task-specific neural units achieve superior performance to fixed units.

Network Level Growth. We define the network level search space as the arrangement of cells of old and new tasks. For the current task $\mathcal{T}^t$, the candidate architecture is composed of cells in each layer, *i.e.*, $h^t = \bigcup_{j=1}^{L} \beta_j$, where L is the number of layers and β_j is composed of the reused old cells $\{o_j^1, ..., o_j^{t-1}\}$ and the searched new cell o_j^t.

Similar to the cell level search, we also allocate a validation score $\Delta = \{\delta_j\}$, an iteration record $C = \{c_j\}$, and

a selected probability $P = \{p_j\}$ to each cell. The update of the selected probability follows the policy in Eq. (4). Finally, the cells with the highest probability are selected to form the model h^t of the current task $\mathcal{T}^t$.

It is worth noting that the architecture growth will inevitably increase the model parameters. We hope to select reused old cells as many as possible to improve the reusability of the learned cells while maintaining good performance. To this end, we modify the updated policy in two aspects: initialization and validation score.

Initialization. Unlike the cell level search, the reused cells learned on the old tasks have already been trained, so their iteration records should be initialized to a non-zero constant c_0 while the new cells are still initialized to zero. Besides, to improve the reuse rate of the old cells, the selected probability of old cells should be higher than that of new cells. Here, we initialize the probability of the old cells to γ times that of the new cell, *i.e.*,

$$p_j^k = \gamma \cdot p_j^t = \frac{\gamma}{\gamma(t-1)+1}, k \in \{1, 2, ..., t-1\}. \quad (5)$$

Validation Score. A simple way is to directly use the error rate as the validation score like Eq. (3), but the reusability of the model is excluded. To keep the model more compact during growth, we explicitly integrate the model parameters into the growth evaluation. After each sampling, the selected architecture path is marked as m^*, and the number of parameters of the selected reused cells is ϕ^{m^*}. Assuming that the error rate on the validation set after one epoch is σ^*, then the validation score can be composed of the error rate and model parameters as below:

$$\delta_j^{m^*} = \sqrt{1-\sigma^*} \cdot \log\left(\frac{\phi^{m^*}}{\phi} + 1\right), \quad (6)$$

where ϕ is the target number of parameters of the reused cells. The above formula shows that the lower error rates and more old reused cells make $\delta_j^{m^*}$ larger. In addition, $\delta_j^{m^*}$ can roughly fall within the range of $(0, 1)$. See the supplementary material for more design details. The ablation study in Section 5.4 shows that the above strategy improves the parameter efficiency while keeping good performance.

4.3. Scene Router

At inference, a scene-specific architecture path needs to be selected to predict the disparity of continuous stereo images. Although the path can be manually chosen, it is often difficult for the drivers to judge what the current scene is. To break this dilemma, we propose a Scene Router module to automatically select the path for the current scene.

As stated in [1], for the current task $\mathcal{T}^t$, we train a one-layer autoencoder for each task. The task-specific autoencoder takes the image feature representation x_n^t as input and is trained using the mean square error loss $\mathcal{L}_{mse}$ between the reconstructed output $\hat{x}_n^t$ and the input x_n^t. At test time, the test image is input into different task-specific autoencoders to reconstruct itself, and the scene corresponding to the autoencoder with the smallest reconstruction error is selected as the final choice.

The above training mechanism can achieve good performance in image classification [1], but it fails in some cases in the driving scenarios by experiments in Section 5.4. An autoencoder trained in one scene may yield a lower reconstruction error for the other due to the correlation between different scenes. To alleviate this problem, we advocate to explicitly enlarge the reconstruction loss of the image of the new scene on the previously trained autoencoders. Assuming that the reconstruction output obtained by the autoencoder trained on the old task $\mathcal{T}^i$ is $\hat{x}_o^i (i < t)$. Our goal is to make the reconstructed output $\hat{x}_n^t$ *close* to the input x_n^t and *far* from $\hat{x}_o^i$, *i.e.*,

$$\mathrm{sim}_{nn}\left(\hat{x}_n^t, x_n^t\right) \gg \mathrm{sim}_{no}\left(\hat{x}_n^t, \hat{x}_o^i\right), \quad (7)$$

where $\mathrm{sim}(x, y)$ is the similarity measurement using the inverse of the mean square error. Inspired by [27], we propose a *scene contrastive loss* to impose this constraint as

$$\mathcal{L}_{con} = -\log\left(\frac{\exp(\mathrm{sim}_{nn}/\tau)}{\exp(\mathrm{sim}_{nn}/\tau) + \sum_i \exp(\mathrm{sim}_{no}/\tau)}\right), \quad (8)$$

where τ is the temperature (typically set to 2). Thus the entire training loss becomes to $\mathcal{L}_{auto} = \mathcal{L}_{mse} + \lambda \cdot \mathcal{L}_{con}$, where $\lambda = 0.1$. Experiments in Section 5.4 show that $\mathcal{L}_{con}$ can significantly improve the accuracy of scene division.

5. Experiments

5.1. Datasets and Evaluation Metrics

Datasets. We evaluate the proposed method on **DrivingStereo** [46], **KITTI raw** [10], and **Virtual KITTI** [2, 9] datasets. DrivingStereo is a large-scale outdoor stereo dataset in driving scenarios. From the entire dataset, we use the specially selected four weather conditions (*cloudy*, *foggy*, *rainy*, *sunny*) for continual stereo. KITTI raw collects real-world outdoor stereo video sequences covering heterogeneous environments, namely *residential*, *city*, *road* and *campus*. Virtual KITTI is a synthetic clone of the real KITTI dataset containing 5 sequences of Scene 01, 02, 06, 18, and 20. Since the weather conditions have been considered in DrivingStereo, we select the 5 scenes with 4 camera configurations for continual stereo in Virtual KITTI.

Metrics. The stereo performance is accessed using the standard end-point-error (EPE) and D1-all metrics. To evaluate the performance for continual stereo, we use Final Average Error (FAE) to represent the average performance of the final model on each task learned so far. Backward Transfer (BWT) in [23] is also adopted to evaluate the forgetting of

Table 1. Results of different continual learning methods on DrivingStereo, KITTI raw, and Virtual KITTI datasets. The D1-all and EPE in terms of FAE and BWT are compared. Red and blue represent the best and the second best results on the corresponding datasets.

Methods	DrivingStereo				KITTI raw				Virtual KITTI			
	FAE		BWT		FAE		BWT		FAE		BWT	
	EPE↓	D1↓	EPE↓	D1↓	EPE↓	D1↓	EPE↓	D1↓	EPE↓	D1↓	EPE↓	D1↓
Incremental Finetuning	0.937	2.60%	0.342	1.43%	0.594	0.78%	0.050	0.04%	0.892	5.63%	0.353	2.66%
EWC [15]	1.053	4.02%	0.095	0.56%	0.620	0.90%	0.008	0.02%	0.934	5.85%	0.070	0.45%
iCaRL [33]	0.866	1.78%	0.225	0.50%	0.586	0.72%	0.011	0.03%	0.863	5.01%	0.143	0.80%
Expert Gate [1]	0.681	1.52%	0.0	0.0%	0.586	0.74%	0.0	0.0%	0.697	3.91%	0.0	0.0%
Learn to Grow [16]	0.662	1.34%	0.0	0.0%	0.598	0.77%	0.0	0.0%	0.836	5.03%	0.0	0.0%
Joint Training (Ideal)	0.673	1.38%	-	-	0.554	0.64%	-	-	0.737	4.18%	-	-
RAG (Ours)	0.637	1.21%	0.0	0.0%	0.583	0.71%	0.0	0.0%	0.717	4.18%	0.0	0.0%

previously learned knowledge. In addition, we further denote a metric of Average Reuse Rate (ARR) to evaluate the reusability of the learned cells, which calculates the average percentage of the number of parameters of reused cells in the current architecture. We provide more details about the datasets and metrics in the supplementary material.

5.2. Implementation Details

Our method is implemented with PyTorch[1]. Images are randomly cropped to the size of 384×192 with no other data augmentation. We adopt the SGD optimizer with momentum 0.9, weight decay 0.0003, and cosine learning rate from 0.025 to 0.001. The update momentum α is set to 0.01. We set the initial value of the iterative record $c_0 = 10$, the multiple $\gamma = 2$, and the target number of parameters $\phi = \Phi/2$. The network structure of the base model and other training details can be found in the supplementary material. Code is available at https://github.com/chzhang18/RAG.

5.3. Evaluation of Continual Stereo

Baselines. We compare our method with several representative continual learning methods, which are regularization-based EWC [15], replay-based iCaRL [33], as well as Expert Gate [1] and Learn to Grow [16] based on dynamic structure. We also adopt the other two widely used baselines, incremental finetuning and joint training using all data from various scenes (ideal conditions).

Comparisons. The overall performance comparisons with the above baselines are listed in Table 1. It is clear that our method achieves remarkable performance on all three datasets across various weather and road conditions. Benefiting from the task-specific neural units, our RAG outperforms Learn to Grow and achieves the best performance on DrivingStereo. A surprising result is that our RAG even surpasses the performance of joint training in ideal on DrivingStereo and Virtual KITTI except for KITTI raw. This suggests that joint training on all the scenes may cause the model to only find the global solution of the mixed scenes, but the solution is not always optimal for each scene, especially when the difference across scenes is large. On the contrary, learning each scene in sequence by architecture growth can decouple the mixed scenes to find a local solution for each scene. The claim is further validated that Expert Gate obtains superior results on Virtual KITTI since it allocates a separate base model to each scene. Each base model can focus on the current scene better under sufficient heterogeneous data, but the number of parameters will increase sharply at the same time. By contrast, our method

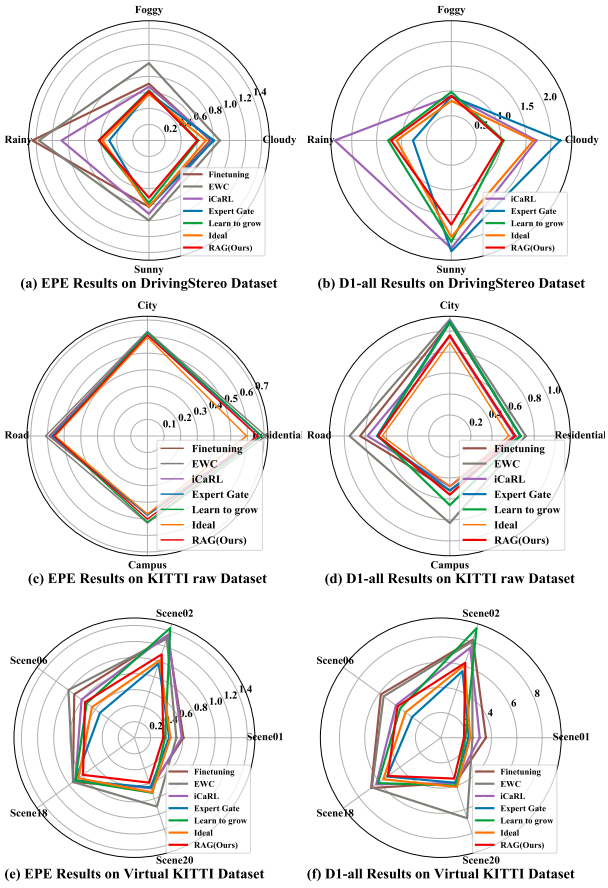


Figure 4. Comparison with various continual learning methods on each scene of DrivingStereo, KITTI raw, and Virtual KITTI datasets. EPE and D1-all results are reported.

[1]We also provide a version implemented by MindSpore.

Table 2. Ablation study of neural units search and architecture growth on the DrivingStereo dataset.

Cells Search	Fea. Net Grow	Mat. Net Grow	# param (M)	Cloudy EPE ↓	Cloudy D1 ↓	Foggy EPE ↓	Foggy D1 ↓	Rainy EPE ↓	Rainy D1 ↓	Sunny EPE ↓	Sunny D1 ↓	FAE EPE ↓	FAE D1 ↓
			0.61	0.782	1.93%	0.639	1.03%	0.541	0.93%	1.020	2.49%	0.746	1.60%
✓			0.61	0.601	1.03%	0.603	0.82%	0.546	0.99%	0.650	1.34%	**0.600**	**1.05%**
✓	✓		0.17	0.733	1.74%	0.704	1.40%	0.806	2.30%	0.696	1.66%	0.735	1.78%
✓		✓	0.38	0.685	1.40%	0.712	1.39%	0.585	1.08%	0.707	1.69%	0.672	1.39%
	✓	✓	0.41	0.618	1.03%	0.604	0.84%	0.645	1.33%	0.753	1.90%	0.655	1.28%
✓	✓	✓	0.48	0.601	1.03%	0.612	0.90%	0.620	1.21%	0.715	1.70%	**0.637**	**1.21%**

Table 3. Ablation results of different strategies for the model reusability on the DrivingStereo dataset.

(a) **Initial value.**

c_0	EPE↓	D1↓	ARR↑
0	0.818	2.12%	**81.5%**
10	**0.655**	**1.34%**	40.7%
20	0.676	1.42%	31.5%

(b) **Multiple.**

γ	EPE↓	D1↓	ARR↑
1	**0.655**	**1.34%**	40.7%
2	0.671	1.36%	48.1%
5	0.703	1.55%	**49.3%**

(c) **Validation score.**

$\delta_j^{m^*}$	EPE↓	D1↓	ARR↑
Eq. (3)	0.671	1.36%	48.1%
Eq. (6)	**0.660**	**1.32%**	**50.9%**

(d) **Target parameters.**

ϕ	EPE↓	D1↓	ARR↑
$\Phi/3$	0.664	1.40%	48.2%
$\Phi/2$	**0.637**	**1.21%**	**50.1%**
Φ	0.660	1.32%	50.9%

can keep high reusability during architecture growth while achieving competitive performance with fewer parameters. For catastrophic forgetting, incremental finetuning suffers the most as evidenced by the highest BWT. Compared with regularization and replay based methods, our method does not forget any previous knowledge since our BWT is zero.

We further compare the FAE on each scene with these methods as shown in Fig. 4. The D1-all results of incremental finetuning and EWC are excluded on DrivingStereo due to poor performance. Our RAG achieves the best results in most scenes on all the datasets despite being inferior in some scenes, such as *rainy* on DrivingStereo, *campus* on KITTI raw, and Scene 06 on Virtual KITTI. It seems that the knowledge learned from previous scenes is not always sufficient to support the model to achieve the best on each scene under reusable constraints. Despite this, we still achieve the top performance in terms of the final average error.

5.4. Ablation Study

In this part, we run comprehensive ablations on the DrivingStereo dataset to analyze the component of RAG and the Scene Router module.

Neural Units Search. In continual stereo, data from heterogeneous scenes are quite different in terms of color, illumination, and disparity distribution. Hence, it is better to design task-specific neural units for each scene. For illustration, we train a separate model with fixed cells (searched on synthetic data) and searchable cells for each scene, respectively. The experimental results are listed in the top two rows of Table 2. It is clear that using task-specific cells achieves remarkable advantages on three scenes. Even in the *rainy* scene, considerable performance is realized. This indicates that searching for task-specific cells is better than using fixed cells.

Architecture Growth. The architecture to grow comes from two parts, $i.e.$, Feature Net and Matching Net. The results in the last four rows in Table 2 demonstrate the contributions of each part. Only the Feature Net growth can achieve reasonable performance, but it is inferior to the model using fixed cells. Only the Matching Net growth achieves better performance benefiting from the more powerful 3D convolutions. The growth of both parts further reduces the error rates and EPE. This suggests that the simultaneous growth of the two parts can maximize the performance of the model on the continual stereo. Our RAG even achieves comparable performance to the multiple models with task-specific cells (2^{nd} row) yet has fewer parameters.

Reusability. Our method will inevitably increase the model parameters as the scenes increase. To tackle this problem, we propose a series of strategies to improve the model reusability while maintaining good performance. Here, four factors are considered, $i.e.$, the initial value c_0, the multiple γ, the validation score $\delta_j^{m^*}$, and the target parameters ϕ.

Table 3a shows the results of several different initial values of c_0. Zero initialization for the well-trained old cells can get a high ARR but the performance of the model drops rapidly as the scenes increase. In contrast, non-zero initialization achieves more stable performance despite decreasing the ARR. We set $c_0 = 10$ in this work. To improve the reusability of the old cells, we set their initial probability to γ times that of the new cells. One can observe in Table 3b that larger multiples can increase the ARR but the error rates also increase. We choose $\gamma = 2$ as the trade-off between performance and reuse rates. Table 3c compares the results of using only the error rate as the validation score like Eq. (3) with our design in Eq. (6). The proposed strategy improves both the performance and ARR, which indicates that better old cells are selected for the new task during architecture growth. We further explore the impact of the target number of parameters of the old cells ϕ. As listed in Table 3d, using a larger target number can get a higher reuse rate, but the error rate does not decrease linearly. We finally

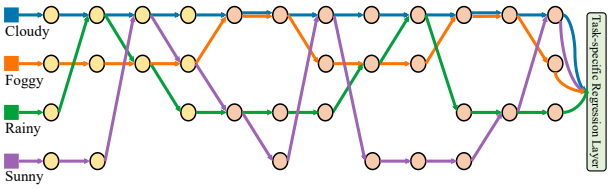


Figure 5. Visualization of the growing architecture on the DrivingStereo dataset. The four architecture paths represent the inference paths of the four kinds of weather. We only depict cells in the Feature Net (yellow circle) and Matching Net (red circle).

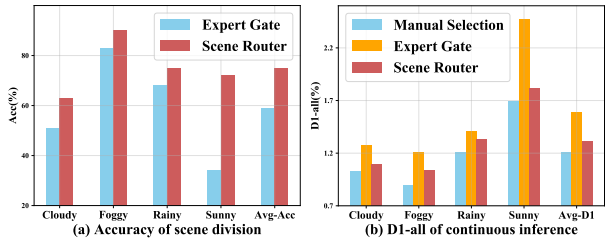


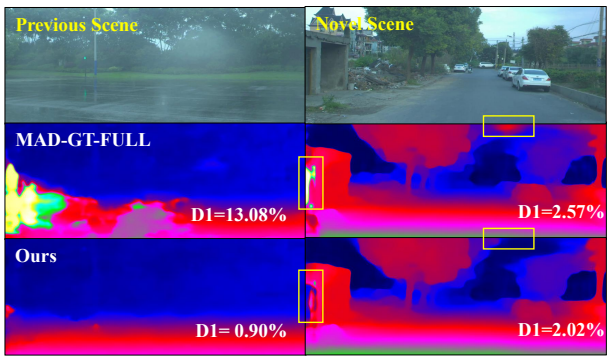
Figure 6. The effectiveness of Scene Router tested on the DrivingStereo dataset including scene division and continuous inference.

choose the suitable intermediate value $\phi = \Phi/2$ where Φ represents the number of parameters of a base model. The visualization of the growing architecture in Fig. 5 gives a more intuitive description of the reusability of old cells.

Scene Router. At deployment, we adopt Scene Router to adaptively select the scene-specific architecture path for continuous disparity estimation. Fig. 6(a) shows the accuracy of scene division of our method and Expert Gate [1]. Serious misjudgments occur in Expert Gate in a certain scene like *sunny*, which makes the model unable to be well-excavated. In contrast, our Scene Router can achieve balanced results in various scenes and improve the mean accuracy by about 15% benefiting from the proposed scene contrastive loss. Despite our method failing in some cases, it does not mean that the final stereo performance will drop a lot. Actually, images in different scenes are sometimes difficult to distinguish clearly, such as *sunny* and *cloudy* scenes. The misclassification of these two scenes may yield comparable stereo performance. To this end, we further compare our Scene Router with Expert Gate and manual selection. As depicted in Fig. 6(b), Scene Router significantly surpasses Expert Gate, achieving considerable performance to manual selection.

5.5. Comparison with Continuous Adaptation

We further compare our method with the supervised version of the continuous adaptation method [43], called MADNet-GT-Full, to show the ability to overcome forgetting and the adaptability to novel scenes at inference. As shown in Fig. 7(a), after learning or adapting to a series of scenes (*cloudy* $\rightarrow$ *foggy* $\rightarrow$ *rainy* $\rightarrow$ *sunny*), MAD-GT-Full yields severe errors when tested on the previously learned scene (*e.g.*, *rainy* days) since it has forgotten previous knowledge. It needs some buffer time to re-adapt to this scene through continuous online gradient updates. In contrast, our method can overcome catastrophic forgetting to achieve good results. Benefiting from the Scene Router, our method can quickly adapt to the rapid scene switches. We also show the generalization performance of direct inference without adaptation in Fig. 7(b). When exposed to a novel scene, such as *overcast* days at dusk, our Scene Router can adaptively select the suitable architecture path to predict better disparity. In this case, the architecture corresponding to the *cloudy* scene is selected for inference since the scene type of the input image is closest to a cloudy day. See supplementary material for more cases.



(a) Overcome catastrophic forgetting (b) Adaptability to novel scenes

Figure 7. Comparison with the continuous adaptation method [43]. Our method can overcome catastrophic forgetting (a) and have better adaptability (boxed in yellow) to the novel scene (b).

6. Conclusion and Future Work

The proposed RAG framework successfully tackles the continual stereo problem through task-specific neural unit search and architecture growth. During architecture growth, our RAG can achieve high reusability to make the model more compact. We have shown the efficacy of RAG by experimental analysis, which surpasses various continual learning methods in challenging weather and road conditions. At deployment, we utilize Scene Router to adaptively select the architecture path to adapt to rapid scene switches without online gradients update. In future work, we plan to extend RAG to other dense regression tasks. We would also like to investigate alternative approaches to realize the function of the Scene Router.

Acknowledgements

This research was supported by the National Key Research and Development Program of China under Grant No. 2018AAA0100400, the National Natural Science Foundation of China under Grants 61976208, 62071466, 62076242, the InnoHK project, and CAAI-Huawei MindSpore Open Fund.

References

[1] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *CVPR*, pages 3366–3375, 2017. 3, 5, 6, 8

[2] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2, 2020. 2, 5

[3] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *CVPR*, pages 5410–5418, 2018. 1, 2

[4] Yukang Chen, Tong Yang, Xiangyu Zhang, Gaofeng Meng, Xinyu Xiao, and Jian Sun. Detnas: Backbone search for object detection. In *NeurIPS*, pages 6642–6652, 2019. 2

[5] Xinjing Cheng, Peng Wang, and Ruigang Yang. Learning depth with convolutional spatial propagation network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(10):2361–2379, 2019. 1, 2

[6] Xuelian Cheng, Yiran Zhong, Mehrtash Harandi, Yuchao Dai, Xiaojun Chang, Tom Drummond, Hongdong Li, and Zongyuan Ge. Hierarchical neural architecture search for deep stereo matching. In *NeurIPS*, pages 22158–22169, 2020. 1, 2, 3, 4

[7] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[8] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017. 3

[9] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, pages 4340–4349, 2016. 5

[10] 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. 2, 5

[11] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, pages 3354–3361, 2012. 1, 2

[12] X. Gu, Z. Fan, S. Zhu, Z. Dai, and P. Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *CVPR*, pages 2495–2504, 2020. 2

[13] Xiaoyang Guo, Kai Yang, Wukui Yang, Xiaogang Wang, and Hongsheng Li. Group-wise correlation stereo network. In *CVPR*, pages 3273–3282, 2019. 2

[14] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, and Peter Henry. End-to-end learning of geometry and context for deep stereo regression. In *ICCV*, pages 66–75, 2017. 1, 2

[15] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of the Sciences*, pages 3521–3526, 2017. 3, 6

[16] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *ICML*, pages 3925–3934, 2019. 3, 6

[17] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 2935–2947, 2017. 3

[18] Zhengfa Liang, Yiliu Feng, Yulan Guo, Hengzhu Liu, Wei Chen, Linbo Qiao, Li Zhou, and Jianfeng Zhang. Learning for disparity estimation through feature constancy. In *CVPR*, pages 2811–2820, 2018. 2

[19] Chenxi Liu, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan L Yuille, and Li Fei-Fei. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In *CVPR*, pages 82–92, 2019. 2

[20] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *ECCV*, pages 19–34, 2018. 2

[21] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018. 2

[22] Rui Liu, Chengxi Yang, Wenxiu Sun, Xiaogang Wang, and Hongsheng Li. Stereogan: Bridging synthetic-to-real domain gap by joint optimization of domain translation and stereo matching. In *CVPR*, pages 12757–12766, 2020. 1, 2

[23] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *NeurIPS*, pages 6467–6476, 2017. 3, 5

[24] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *CVPR*, pages 7765–7773, 2018. 3

[25] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, pages 4040–4048, 2016. 1, 2

[26] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *CVPR*, pages 3061–3070, 2015. 1, 2

[27] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 5

[28] Jiahao Pang, Wenxiu Sun, Jimmy Ren, Chengxi Yang, and Qiong Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *ICCV*, pages 878–886, 2017. 2

[29] Jiahao Pang, Wenxiu Sun, Chengxi Yang, Jimmy Ren, Ruichao Xiao, Jin Zeng, and Liang Lin. Zoom and learn: Generalizing deep stereo matching to novel domains. In *CVPR*, pages 2070–2079, 2018. 1, 2

[30] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In *ICML*, pages 4095–4104, 2018. 2

[31] Matteo Poggi, Alessio Tonioni, Fabio Tosi, Stefano Mattoccia, and Luigi Di Stefano. Continual adaptation for deep

stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[32] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *AAAI*, pages 4780–4789, 2019. 2

[33] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 2001–2010, 2017. 2, 6

[34] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016. 3

[35] Tonmoy Saikia, Yassine Marrakchi, Arber Zela, Frank Hutter, and Thomas Brox. Autodispnet: Improving disparity estimation with automl. In *ICCV*, pages 1812–1823, 2019. 2

[36] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *ICML*, pages 4548–4557, 2018. 3

[37] Zhelun Shen, Yuchao Dai, and Zhibo Rao. Cfnet: Cascade and fused cost volume for robust stereo matching. In *CVPR*, pages 13906–13915, June 2021. 2

[38] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *arXiv preprint arXiv:1705.08690*, 2017. 2

[39] Xiao Song, Guorun Yang, Xinge Zhu, Hui Zhou, Zhe Wang, and Jianping Shi. Adastereo: a simple and efficient approach for adaptive stereo matching. In *CVPR*, pages 10328–10337, 2021. 1, 2

[40] Xiao Song, Xu Zhao, Liangji Fang, Hanwen Hu, and Yizhou Yu. Edgestereo: An effective multi-task learning network for stereo matching and edge detection. *International Journal of Computer Vision*, 128(4):910–930, 2020. 2

[41] Alessio Tonioni, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. Unsupervised adaptation for deep stereo. In *ICCV*, pages 1614–1622, 2017. 1, 2

[42] Alessio Tonioni, Oscar Rahnama, Thomas Joy, Luigi Di Stefano, Thalaiyasingam Ajanthan, and Philip H S Torr. Learning to adapt for stereo. In *CVPR*, pages 9661–9670, 2019. 2

[43] Alessio Tonioni, Fabio Tosi, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. Real-time self-adaptive deep stereo. In *CVPR*, pages 195–204, 2019. 2, 8

[44] Wenjin Wang, Yunqing Hu, and Yin Zhang. Lifelong learning with searchable extension units. *arXiv preprint arXiv:2003.08559*, 2020. 3

[45] Haofei Xu and Juyong Zhang. Aanet: Adaptive aggregation network for efficient stereo matching. In *CVPR*, pages 1959–1968, 2020. 1, 2

[46] Guorun Yang, Xiao Song, Chaoqin Huang, Zhidong Deng, Jianping Shi, and Bolei Zhou. Drivingstereo: A large-scale dataset for stereo matching in autonomous driving scenarios. In *CVPR*, pages 899–908, 2019. 2, 3, 5

[47] Guorun Yang, Hengshuang Zhao, Jianping Shi, Zhidong Deng, and Jiaya Jia. Segstereo: Exploiting semantic information for disparity estimation. In *ECCV*, pages 660–676, 2018. 2

[48] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *CVPR*, pages 185–194, 2019. 1, 2

[49] Xiawu Zheng, Rongrong Ji, Lang Tang, Baochang Zhang, Jianzhuang Liu, and Qi Tian. Multinomial distribution learning for effective neural architecture search. In *ICCV*, pages 1304–1313, 2019. 4

[50] Yiran Zhong, Hongdong Li, and Yuchao Dai. Open-world stereo video matching with deep rnn. In *ECCV*, pages 101–116, 2018. 2

[51] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016. 2

[52] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *CVPR*, pages 8697–8710, 2018. 2