Zero-Shot Neural Architecture Search for Efficient Deep Stereo Matching

Alessio Mingozzi[1*], Stefano Mattoccia[1], Matteo Poggi[1], and Fatma Güney[2]

[1]Alma Mater Studiorum University of Bologna
[2]KUIS AI Center, Koç University
[*]alessio.mingozzi2@unibo.it

Abstract. This paper introduces a novel and efficient architecture for deep stereo matching obtained through Zero-Shot Neural Architecture Search (NAS). Although accurate and capable of good generalization across datasets, state-of-the-art iterative stereo models are often too computationally expensive for low-power devices. In order to address this limitation, this work employs NAS to efficiently explore a search space of different layer types and hyperparameters, including efficient residual layers from Ghost Modules. Instead of relying on extensive training, the method evaluates candidate architectures using a combined zero-cost proxy score based on the AZ-NAS score and the number of parameters, thus promoting the selection of smaller, efficient models. Applied to RAFT-Stereo, this process yields a significantly smaller – 1.14M parameters, compared to the original 11M – and substantially faster network. The resulting architecture maintains competitive performance on various stereo benchmarks while running 3× faster, demonstrating the effectiveness of Zero-Shot NAS in optimizing deep stereo networks for resource-constrained environments.

Keywords: Neural Architecture Search · Zero-Shot · Stereo Matching.

1 Introduction

Stereo matching, a foundational problem in computer vision, aims to determine dense correspondences between a pair of rectified stereo images, thereby generating a dense disparity map. This cue is crucial for many applications, such as robotics and augmented reality, and purposely, many strategies have been proposed to obtain robust disparity maps. In contrast to the conventional use of 3D convolutional networks in stereo matching, RAFT-Stereo [18] utilizes 2D convolutions and a lightweight cost volume. The network employs multi-level GRUs to effectively propagate information across the image, enabling both real-time inference and accurate disparity estimation even when dealing with high-resolution images. Notably, RAFT-Stereo exhibits remarkable cross-dataset generalization capabilities, achieving impressive performance on real-world datasets even after training solely on synthetic data. Nevertheless, given its characteristics, the standard version struggles with low-performance or memory-constrained systems, such as edge devices.

Although the advent of deep neural networks has brought substantial advancements across different disciplines, their design often necessitates substantial human effort and expertise. To address this issue, Neural Architecture Search (NAS) aims to automate the design process of neural networks by identifying optimal network architectures through an iterative search process. However, since each candidate needs to be evaluated, this process can be computationally expensive and time-consuming. Purposely, One-Shot and Zero-Shot NAS paradigms have been proposed. The former strategy employs a hyper-network to share candidate operations, reducing training time significantly compared to traditional multi-shot approaches. The latter further enhances efficiency by eliminating training during the search phase, specifically, by employing various proxy scores to predict the accuracy of candidate network architectures during the initialization stage. These proxies, often grounded in theoretical analyses of deep neural networks, offer valuable insights into the characteristics of high-performing architectures. Employing Zero-Shot proxies that accurately reflect the nature of network performance, particularly in constrained scenarios, is essential to fully harness the potential of this strategy in the presence of a low computational budget.

The main contribution of this work is leveraging the Zero-Shot NAS paradigm to obtain a fast and lightweight network with a minimal loss in accuracy compared to RAFT-Stereo.
The source code is available at https://github.com/amingozz/RAFT-StereoZero.

2 Related Work

We briefly review the literature relevant to our work.

Deep Stereo Matching. After years of developments in hand-crafted algorithms [31] already, end-to-end architectures starting with DispNetC [22] and developed with 2D [45] and 3D [12] convolutions became the dominant solutions in stereo matching. This led to the exploration of new paradigms, stereo video processing [47], fusion with active sensors [24, 3], self-adapting models [36, 25, 26] or strategies to deal with challenging conditions such as non-Lambertian surfaces [46, 27, 7] or depth discontinuities [39, 2, 37]. The latest advances, thoroughly reviewed in [38], introduced recurrent architectures [18] inspired by RAFT [35]. This category represents the preferred choice nowadays, as it proved great generalization capabilities without requiring any explicit design choice [5]: among them, RAFT-Stereo [35] introduces an all-pair cost volume used to guide an iterative disparity estimation process. Despite the advances brought by deep learning, state-of-the-art architectures seldom fit low-power devices and when feasible, they are extremely slow.

Neural Architecture Search. Recent advancements in Automated Machine Learning (AutoML) [11] have notably accelerated network development. Neural Architecture Search (NAS), a subset of AutoML, has proven successful in tasks such as classification [44], object detection [33], and semantic segmentation [20]. However, NAS demands substantial computational resources. To mit-

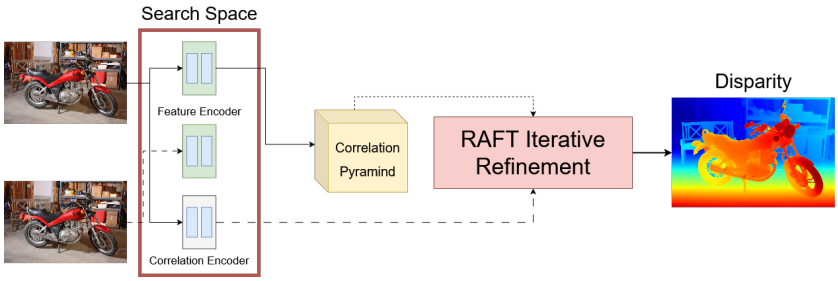

Fig. 1: **RAFT-Stereo architecture.** The search process is conducted across the two encoders.

igate these computational demands, researchers have developed more efficient methods like Differentiable Architecture Search (DARTS) [20], which introduces a hierarchical NAS approach. DARTS has been applied to stereo, resulting in architectures like AutoDispNet [29] obtained in 42 GPU days. LEAStereo [6] further extended this approach to volumetric stereo matching, achieving state-of-the-art results. Recently, Zero-Shot NAS methods have emerged to reduce computational burdens by using low-cost proxies to evaluate candidate architectures without extensive training. Various proxies [1, 42, 34, 21] have been developed to estimate architecture quality. ZenNAS [17] demonstrated the capability to measure architectural expressivity with minimal resources, identifying a competitive ImageNet architecture in 0.5 GPU days. Similarly, ZiCo [15] achieves better performances evaluating gradients over input data.Despite findings that many zero-shot proxies correlate similarly with test set accuracy compared to simpler metrics like parameter count and FLOPs [14], the recent AZ-NAS [13] proposes higher accuracy correlation by providing four distinct scores computed over a single forward pass, addressing the limitations of previous proxies and improving discrimination among top-performing architectures.

Given the absence of prior applications of Zero-Shot NAS methods to deep stereo problems and the high computational cost of traditional approaches, this work applies these methodologies for such tasks for the first time.

3 Method

This section describes the methodology for obtaining a faster and lightweight RAFT-Stereo-like network by leveraging a NAS framework and a Zero-Shot strategy, in the form of an adapted AZ-NAS score.

3.1 Search Space

As shown in Figure 1, RAFT-Stereo architecture consists of two primary components: the feature encoder and the context encoder, usually sharing the same structure, followed by the iterative refinement process carried out by ConvGRUs.

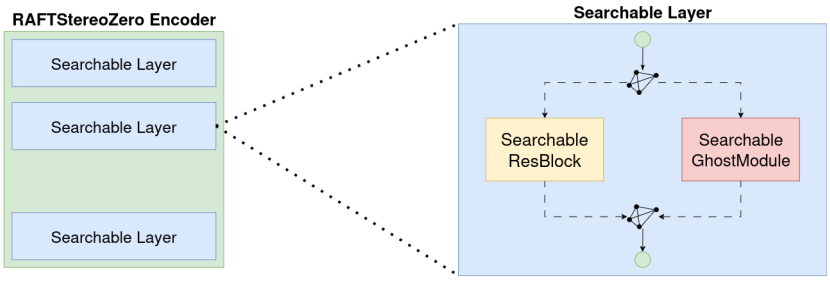

Fig. 2: **Encoder search space.** Layer selection is made between the original Resnet-like block and a GhostBlock.

Hence, the search space is constructed starting from these two modules. To achieve a smaller and more efficient architecture, we defined the global search space over two properties: the hyperparameters of the various layers, including the convolution feature dimensions or kernel size, and the layer type. Concerning the latter, the original RAFT-Stereo implementation uses 2D convolutions and residual blocks within the encoders. In contrast, we add depthwise convolutions and a more efficient residual layer derived from GhostModule [9, 10] to the search space, as it can reduce computational cost and model size without sacrificing performance. In summary, the search space comprises the original RAFT-Stereo layers, depthwise convolutions, and GhostModules, with all their hyperparameters, as summarized in Figure 2.

3.2 Search Strategy

NAS methodologies rely on a search strategy to define how to mutate the architecture to generate new candidates. Various strategies have been proposed in the literature, ranging from genetic algorithms to reinforcement learning. One such strategy is regularized evolution [28], which relies on evolutionary algorithms, a class of optimization techniques inspired by natural selection. This framework employs a randomized population of architectures as its starting point. During each search cycle, the architecture with the highest fitness score is selected as a parent for the new generation.

3.3 Evaluation Strategy

To determine the best-performing architecture, evaluating and selecting candidates based on desired performance characteristics is crucial. An exhaustive evaluation of all candidates is computationally impractical, often requiring thousands of GPU days. To address this, techniques like One-Shot and Zero-Shot methodologies have been proposed. Specifically, we utilize the latter for efficiently ranking architectures, as it is the fastest and least resource-intensive. Among zero-shot methods, AZ-NAS scores [13] show a good correlation with

test set accuracy, using different scores to capture various network characteristics and provide better ratings among candidates. However, zero-cost scores tend to optimize the expressive capabilities of networks, driving the search toward larger networks, which hinders the identification of smaller, more efficient combinations. To address this, we define a metric by summing AZ-Score and the number of parameters, aiming to find a network with the lowest parameters while retaining a high AZ-Score value. The AZ-Score comprises four metrics:

Expressivity($\mathcal{E}$). It evaluates how features are distributed across orientations in the feature space, given randomly initialized network weights and Gaussian random inputs. Given the score of the l-th primary block, $s_l^{\mathcal{E}}$, as an entropy score, posing L1-normalized coefficients of PCs as probabilities:

$$s_l^{\mathcal{E}} = \sum_{i=1}^{c} -\hat{\lambda}_l(i) log \hat{\lambda}_l(i) \tag{1}$$

where $\hat{\lambda}_l(i)$ is a set of L1-normalized coefficient. The overall network score consists of the sum of all block-level scores:

$$s^{\mathcal{E}} = \sum_{l=1}^{L} s_l^{\mathcal{E}} \tag{2}$$

Progressivity($\mathcal{P}$). The difference of the block-level $\mathcal{E}$ scores from Eq.(1):

$$s^{\mathcal{P}} = \min_{lin2,\ldots,L} s_l^{\mathcal{E}} - s_{l-1}^{\mathcal{E}} \tag{3}$$

Trainability($\mathcal{T}$). Given the approximation of the Jacobian matrix A_l:

$$A_l^{\top} = \frac{1}{n} \sum_{p=1}^{n} g_{l-1}(p) g_l^{\top}(p), \tag{4}$$

The trainability score $s^{\mathcal{T}}$ is defined as:

$$s^{\mathcal{T}} = \frac{1}{L-1} \sum_{l=2}^{L} -\sigma_l - \frac{1}{\sigma_l} + 2, \tag{5}$$

which reaches its maximum value when the spectral norm of A_l, denoted as σ_l, is equal to 1 for all l.

Complexity($\mathcal{C}$). Given the correlation between network dimension and performance, [13] proposes using the FLOPs count as a complexity proxy.

$$s^{\mathcal{C}}(i) = log_{10}(FLOPs(i)) \tag{6}$$

We use the log_{10} to reduce the summed score dimension.

Score aggregation. Since the NNI Framework allows only a single floating point value as the score, we cannot rely on the ranking approach used by AZ-NAS, which instead will be used to select the final architecture. For this reason,

Algorithm 1 Evolutionary search using the combined score

Input: search space Z; number of NAS iteration T.
Output: selected architecture A^*.
Randomly initialize first architecture A_1 for evolutionary search
while $i = 1 < T$ **do**
    Compute proxy scores $s^{\mathcal{E}}(i)$, $s^{\mathcal{T}}(i)$, $s^{\mathcal{P}}(i)$ for the architecture A_i.
    Append the architecture A_i to the population history $\mathbb{A}$.
    Compute #PARAMS of A_i.
    Compute the score $s(A_i)$.
    Generate a new candidate architecture A_{i+1} which belongs to the search space Z by mutating the top architecture based on s score.
end while
Compute the ranking over the history set $\mathbb{A}$ and select the best candidate architecture A^*

we aggregate all scores into a single value. Given an architecture candidate, we compute the set of scores $\mathcal{M}$ and define the i^th architecture score $s^{AZ}(i)$ as:

$$s^{AZ}(i) = \sum_{\mathcal{M} \in \mathcal{E},\mathcal{T},\mathcal{P},\mathcal{C}} s^{\mathcal{M}}(i) \tag{7}$$

In order to account for the architectural dimensions in the scoring process, the combined final score is formulated as:

$$s^{\Omega}(i) = ln(s^{AZ}(i)) - ln(\#PARAMS(i)) \tag{8}$$

We compute all proxy scores for the possible candidate architectures during the evolutionary search process. Subsequently, to generate new candidates, we mutate the top-scoring architecture based on the s^{Ω} value. Finally we select the best-performing architecture using the AZ ranking formula:

$$S^{\Omega}(i) = \sum_{\mathcal{M} \in \{\mathcal{E},\mathcal{P},\mathcal{T},\mathcal{C}\}} \log\left(\frac{\text{Rank}\left(s^{\Omega}(i)\right)}{m}\right) \tag{9}$$

where m is the number of candidate architecture. The chosen architecture A^{Ω} will be the first ranked candidate. The process is shown in Algorithm 1.

4 Experiments

In this section, we describe the application of our combined score within the neural architecture search process to identify an optimal stereo architecture. Then, we evaluate its performance against the original RAFT-Stereo, as well as with its more efficient variant, RAFT-Stereo *realtime* (RT), using a shared backbone for features context extraction and bi-level GRUs at $\frac{1}{8}$ and $\frac{1}{16}$ resolution [19].

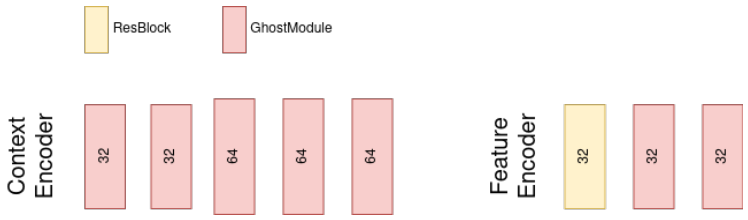

Fig. 3: **RAFT-StereoZero architecture.** The encoders consist predominantly of GhostModules, except for the initial layer in the feature encoder. Each layer employs distinct hyperparameter configurations.

Model	Middlebury 2014		ETH3D		KITTI 2012		KITTI 2015	
	EPE	D1	EPE	D1	EPE	D1	EPE	D1
Setup A	3.21	18.87	0.67	16.92	0.90	**4.26**	1.51	**6.34**
Setup B	3.13	**18.19**	0.58	12.29	**0.89**	4.37	1.29	6.72
Setup C	**2.91**	18.89	**0.40**	**6.13**	0.90	4.51	**1.22**	6.44

Table 1: **Ablation study – impact of the training data.** We evaluate the impact of different training data configurations.

4.1 Implementation details

The architecture search was implemented in PyTorch using the NNI framework, using the NeRF stereo dataset [40] to compute scores over a single batch sample of the dataset during evaluation.

Search process. Upon concluding the search process, which took 0.1 GPU days, we applied the ranking process over the architecture history to identify the network with the best overall score. The final architecture configuration, dubbed RAFT-StereoZero and shown in Fig.3, counts 1.14M trainable parameters, in contrast to the original 11M, translating into lower memory occupancy. While RAFT-Stereo consists of ResBlocks of a variable number of channels, RAFT-StereoZero modules are mostly GhostModules counting 32 or 64 channels. Given the iterative nature of the prediction process, we further optimize RAFT-StereoZero complexity and speed by conducting a Hyperparameter Optimization (HPO) search to identify the best combination of training and validation iterations. Using the Tree-structured Parzen Estimator (TPE) algorithm [43], each combination was trained over a small subset of the training dataset, i.e., 400 samples for 5000 steps, with the error on Middlebury [30] and KITTI 2015 [23] used as evaluation scores, leading to 8 training and 10 validation iterations as the best trade-off between accuracy and speed.

Evaluation metrics. To assess the accuracy of each model, we measure the average end-point error (EPE) between predicted and ground-truth disparity maps, as well as the percentage of pixels with error larger than a threshold (D1), fixed to 2, 1, and 3 for Middlebury, ETH3D, and KITTI datasets, respectively.

Model	Middlebury 2014		ETH3D		KITTI 2012		KITTI 2015	
	EPE	D1	EPE	D1	EPE	D1	EPE	D1
RAFT-Stereo	1.57	11.50	0.27	2.61	0.83	3.86	1.13	5.68
RAFT-Stereo (Setup C)	1.79	11.78	0.88	24.17	0.76	2.88	1.08	4.86
RAFT-StereoZero	1.87	13.06	0.31	4.32	0.78	3.57	1.12	5.69
RAFT-Stereo (RT)	2.53	15.57	0.58	5.76	0.92	4.31	1.15	5.82
RAFT-StereoZero (RT)	2.93	18.59	0.49	9.28	0.89	4.03	1.13	5.62

Table 2: **Quantitative results.** When compared with the original architecture, RAFT-StereoZero achieves competitive results across different datasets.

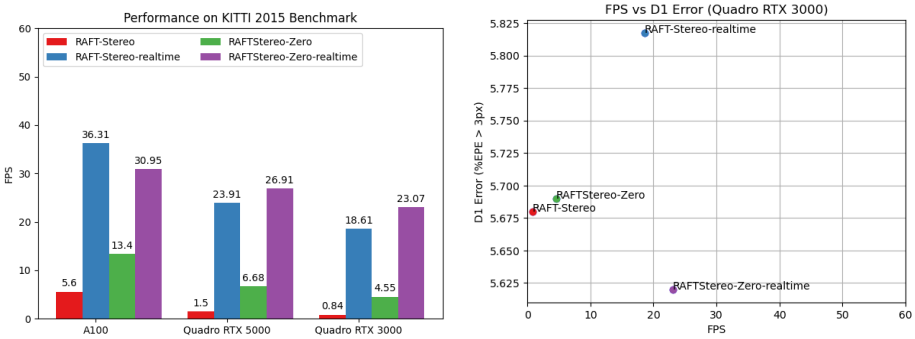

Fig. 4: **Runtime analysis.** Left: framerate measured on three different GPUS; right: accuracy-framerate plot on KITTI 2015 using Quadro RTX 3000 GPU.

4.2 Training

Once the search process is concluded, we trained the architecture using the same protocol as the original RAFT-Stereo, while exploring the use of more recent datasets – such as NERF-Stereo [40] and CREStereo [16] – alongside SceneFlow [22], Sintel Stereo [4], and Falling Things [41]. After balancing, our training set counts 760,836 samples during training. We trained RAFT-StereoZero on four A100 GPUs for over 800,000 steps, with a total batch size 32.

4.3 Evaluation

In order to assess the performance of RAFT-StereoZero compared to the original RAFT-Stereo, we run the evaluation over the zero-shot benchmark [40] composed of four standard different stereo datasets: Middlebury 2014 [30], ETH3D [32], KITTI 2012[8] and KITTI 2015 [23].

Impact of training datasets. We study the effect of each additional dataset we introduce for training. Purposely, we train different instances RAFT-StereoZero following the same protocol, yet using three training data setup: first, we use just Sceneflow, Falling Things and Sintel Stereo (Setup A), then add NERF-Stereo (Setup B), and finally, we add CREStereo (Setup C). Table 1 reports the outcome of this evaluation. Specifically, we can see an average 8% relative improvement

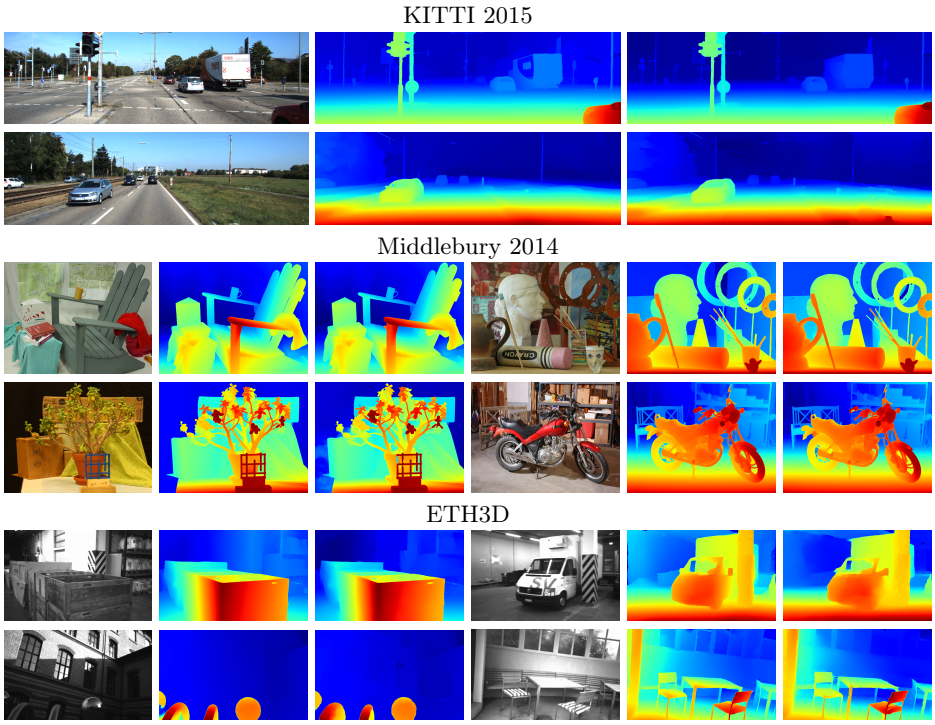

Fig. 5: **Qualitative results.** We report examples from KITTI 2015, Middlebury 2014 and ETH3D, showing the reference color image (left) and the disparity maps predicted by RAFT-Stereo (middle) and RAFT-StereoZero (right).

when adding NeRF Stereo, and a further 10% when adding CREStereo. In particular, the accuracy over the ETH3D benchmark benefits the most, with an improvement of 31% EPE and a 50% D1.

Accuracy comparison with RAFT-Stereo. Table 2 reports a comparison between the original RAFT-Stereo and our model at the top. For a fair comparison, we also train a further instance of RAFT-Stereo on the same training data used for our model (Setup C): this yields some improvements on KITTI datasets, although being surprisingly ineffective on ETH3D. We can appreciate how RAFT-StereoZero achieves almost equivalent accuracy on any dataset with marginal drops only – lower than 0.1 EPE and about 1% D1 on Middlebury and KITTI datasets against RAFT-Stereo Setup C, about 0.03 EPE and 1.3% D1 on ETH3D against the original RAFT-Stereo.

At the bottom, we report a comparison between RAFT-Stereo (RT) variant and a *realtime* counterpart searched through our approach, referred to as RAFT-StereoZero (RT). Again, our model achieves marginal drops in accuracy on Middlebury and ETH3D, while gaining some accuracy on KITTI datasets.

The small drops exposed by both our models against their counterparts allow for training accuracy with efficiency, as we are going to discuss in the remainder.

Inference Speed. Figure 4 reports the accuracy-speed trade-off achieved by the original RAFT-Stereo models and our searched variants. On the left, the runtime required to process KITTI 2015 images when measured on three GPUs: nVidia A100, Quadro RTX 5000, and Quadro RTX 3000. RAFT-StereoZero achieves a speed-up ranging from 2.3× to 5.4× against RAFT-Stereo and an increase of 5FPS when considering (RT) variants. On the right, we report an accuracy-framerate plot concerning performance on the Quadro RTX 3000. It highlights that our models achieve faster and more accurate results on KITTI 2015 than their original counterparts.

4.4 Qualitative Results

We conclude by reporting a qualitative comparison between the predictions by RAFT-Stereo and RAFT-StereoZero. Figure 5 shows examples from KITTI 2015, Middlebury 2014 and ETH3D datasets. We can appreciate how our model retains the generalization capability of the original RAFT-Stereo, as well as fine-grained details in the predicted disparity maps.

5 Conclusions

In this paper, we propose RAFT-StereoZero, a novel architectural configuration based on RAFT-Stereo [18] for two-view stereo, identified using the Zero-Shot Neural Architecture Search methodology. By leveraging the capabilities of the AZ-Score to evaluate the quality of candidate architectures, combined with a performance-focused approach, we have discovered a model that not only achieves competitive results but also exhibits faster performance than the original architecture.

References

1. Abdelfattah, M.S., Mehrotra, A., Łukasz Dudziak, Lane, N.D.: Zero-cost proxies for lightweight nas (2021), https://arxiv.org/abs/2101.08134
2. Aleotti, F., Tosi, F., Ramirez, P.Z., Poggi, M., Salti, S., Mattoccia, S., Di Stefano, L.: Neural disparity refinement for arbitrary resolution stereo. In: 3DV (2021)
3. Bartolomei, L., Poggi, M., Tosi, F., Conti, A., Mattoccia, S.: Active stereo without pattern projector. In: ICCV (2023)
4. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: ECCV (2012)
5. Cai, C., Poggi, M., Mattoccia, S., Mordohai, P.: Matching-space stereo networks for cross-domain generalization. In: 3DV (2020)
6. Cheng, X., Zhong, Y., Harandi, M., Dai, Y., Chang, X., Li, H., Drummond, T., Ge, Z.: Hierarchical neural architecture search for deep stereo matching. NeurIPS (2020)
7. Costanzino, A., Ramirez, P.Z., Poggi, M., Tosi, F., Mattoccia, S., Di Stefano, L.: Learning depth estimation for transparent and mirror surfaces. In: CVPR (2023)
8. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The Kitti vision benchmark suite. In: CVPR (2012)

9. Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., Xu, C.: GhostNet: More Features from Cheap Operations (Mar 2020). https://doi.org/10.48550/arXiv.1911.11907
10. Han, K., Wang, Y., Xu, C., Guo, J., Xu, C., Wu, E., Tian, Q.: GhostNets on Heterogeneous Devices via Cheap Operations (Jan 2022). https://doi.org/10.48550/arXiv.2201.03297
11. Hutter, F., Kotthoff, L., Vanschoren, J. (eds.): Automated Machine Learning: Methods, Systems, Challenges. The Springer Series on Challenges in Machine Learning, Springer (2019). https://doi.org/10.1007/978-3-030-05318-5
12. Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., Bry, A.: End-to-end learning of geometry and context for deep stereo regression. In: ICCV (2017)
13. Lee, J., Ham, B.: AZ-NAS: Assembling Zero-Cost Proxies for Network Architecture Search. In: CVPR (2024)
14. Li, G., Hoang, D., Bhardwaj, K., Lin, M., Wang, Z., Marculescu, R.: Zero-Shot Neural Architecture Search: Challenges, Solutions, and Opportunities. TPAMI pp. 1–19 (2024)
15. Li, G., Yang, Y., Bhardwaj, K., Mărculescu, R.: ZiCo: Zero-shot NAS via Inverse Coefficient of Variation on Gradients. ICLR (2023)
16. Li, J., Wang, P., Xiong, P., Cai, T., Yan, Z., Yang, L., Liu, J., Fan, H., Liu, S.: Practical Stereo Matching via Cascaded Recurrent Network with Adaptive Correlation. In: CVPR (2022)
17. Lin, M., Wang, P., Sun, Z., Chen, H., Sun, X., Qian, Q., Li, H., Jin, R.: Zen-NAS: A Zero-Shot NAS for High-Performance Image Recognition. In: ICCV (2021)
18. Lipson, L., Teed, Z., Deng, J.: Raft-stereo: Multilevel recurrent field transforms for stereo matching. In: 3DV (2021)
19. Lipson, L., Teed, Z., Deng, J.: RAFT-Stereo: Multilevel Recurrent Field Transforms for Stereo Matching (Sep 2021). https://doi.org/10.48550/arXiv.2109.07547
20. Liu, H., Simonyan, K., Yang, Y.: DARTS: Differentiable Architecture Search (Apr 2019). https://doi.org/10.48550/arXiv.1806.09055
21. Lopes, V., Alirezazadeh, S., Alexandre, L.A.: EPE-NAS: Efficient Performance Estimation Without Training for Neural Architecture Search. In: ICANN (2021)
22. Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: CVPR (2016)
23. Menze, M., Geiger, A.: Object scene flow for autonomous vehicles. In: CVPR (2015)
24. Poggi, M., Pallotti, D., Tosi, F., Mattoccia, S.: Guided stereo matching. In: CVPR (2019)
25. Poggi, M., Tonioni, A., Tosi, F., Mattoccia, S., Di Stefano, L.: Continual adaptation for deep stereo. TPAMI **44**(9), 4713–4729 (2021)
26. Poggi, M., Tosi, F.: Federated online adaptation for deep stereo. In: CVPR (2024)
27. Ramirez, P.Z., Costanzino, A., Tosi, F., Poggi, M., Salti, S., Mattoccia, S., Stefano, L.D.: Booster: A benchmark for depth from images of specular and transparent surfaces. TPAMI **46**(1), 85–102 (2024)
28. Real, E., Aggarwal, A., Huang, Y., Le, Q.V.: Regularized Evolution for Image Classifier Architecture Search (Feb 2019). https://doi.org/10.48550/arXiv.1802.01548
29. Saikia, T., Marrakchi, Y., Zela, A., Hutter, F., Brox, T.: Autodispnet: Improving disparity estimation with automl. In: ICCV (2019)
30. Scharstein, D., Hirschmüller, H., Kitajima, Y., Krathwohl, G., Nešić, N., Wang, X., Westling, P.: High-resolution stereo datasets with subpixel-accurate ground truth. In: GCPR (2014)

31. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. IJCV **47**, 7–42 (2002)
32. Schops, T., Schonberger, J.L., Galliani, S., Sattler, T., Schindler, K., Pollefeys, M., Geiger, A.: A multi-view stereo benchmark with high-resolution images and multi-camera videos. In: CVPR (2017)
33. Tan, M., Pang, R., Le, Q.V.: EfficientDet: Scalable and Efficient Object Detection (Jul 2020). https://doi.org/10.48550/arXiv.1911.09070
34. Tanaka, H., Kunin, D., Yamins, D.L., Ganguli, S.: Pruning neural networks without any data by iteratively conserving synaptic flow. In: NeurIPS (2020)
35. Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: ECCV 2020 (2020)
36. Tonioni, A., Tosi, F., Poggi, M., Mattoccia, S., Stefano, L.D.: Real-time self-adaptive deep stereo. In: CVPR (2019)
37. Tosi, F., Aleotti, F., Ramirez, P.Z., Poggi, M., Salti, S., Mattoccia, S., Di Stefano, L.: Neural disparity refinement. TPAMI (2024)
38. Tosi, F., Bartolomei, L., Poggi, M.: A survey on deep stereo matching in the twenties. arXiv preprint arXiv:2407.07816 (2024), https://arxiv.org/abs/2407.07816, extended version of CVPR 2024 Tutorial "Deep Stereo Matching in the Twenties" (https://sites.google.com/view/stereo-twenties)
39. Tosi, F., Liao, Y., Schmitt, C., Geiger, A.: Smd-nets: Stereo mixture density networks. In: CVPR (2021)
40. Tosi, F., Tonioni, A., De Gregorio, D., Poggi, M.: NeRF-Supervised Deep Stereo. In: CVPR (2023)
41. Tremblay, J., To, T., Birchfield, S.: Falling things: A synthetic dataset for 3d object detection and pose estimation. In: CVPR Workshops (2018)
42. Wang, C., Zhang, G., Grosse, R.: Picking winning tickets before training by preserving gradient flow (2020), https://arxiv.org/abs/2002.07376
43. Watanabe, S.: Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance (May 2023). https://doi.org/10.48550/arXiv.2304.11127
44. Wu, B., Dai, X., Zhang, P., Wang, Y., Sun, F., Wu, Y., Tian, Y., Vajda, P., Jia, Y., Keutzer, K.: FBNet: Hardware-Aware Efficient ConvNet Design via Differentiable Neural Architecture Search (May 2019). https://doi.org/10.48550/arXiv.1812.03443
45. Yin, Z., Darrell, T., Yu, F.: Hierarchical discrete distribution decomposition for match density estimation. In: CVPR (2019)
46. Zama Ramirez, P., Tosi, F., Poggi, M., Salti, S., Di Stefano, L., Mattoccia, S.: Open challenges in deep stereo: the booster dataset. In: CVPR (2022)
47. Zhang, Y., Poggi, M., Mattoccia, S.: TemporalStereo: Efficient spatial-temporal stereo matching network. In: IROS (2023)