Seeking attention: Using full context transformers for better disparity estimation

Nadir Bengana, Janne Mustaniemi, and Janne Heikkilä

University of Oulu, Oulu, Finland
mohamed.bengana@oulu.fi

Abstract. Until recently, convolutional neural networks have dominated various machine vision fields—including stereo disparity estimation—with little to no competition. Vision transformers have shaken up this domination with the introduction of multiple models achieving state of art results in fields such as semantic segmentation and object detection. In this paper, we explore the viability of stereo transformers, which are attention-based models inspired from NLP applications, by designing a transformer-based stereo disparity estimation as well as an end-to-end transformer architectures for both feature extraction and feature matching. Our solution is not limited by a pre-set maximum disparity and manages to achieve state of the art on SceneFlow dataset.

Keywords: Stereo Disparity · Depth estimation · Vision Transformer.

1 Introduction

What is the next step for stereo disparity estimation? Until recently, convolutional neural networks (CNN) dominated computer vision applications. However, a novel method called transformers has proved to be effective in various fields including but not limited to classification [1], semantic segmentation [2], image inpainting [3], and super-resolution [4].

Unlike Convolutional Neural Networks, transformers are attention based models. The term attention was used to refer to how we pay attention to specific areas of an image or words in a sentence. In machine learning context, attention [5] was born to address an issue in sequence to sequence problems where only the output of the decoder was considered. Attention, on the other hand, takes the output at each step and assigns a weight to it, allowing the decoder to focus on the most important parts of the sequence.

Unlike early attention based models, transformers rely exclusively on attention to draw global dependencies between the input and output [6]. Transformers revolutionized the field of Natural Language Processing (NLP) with models such as GPT [7], and BERT [8] vastly outperforming their recurrent neural network counterparts.

Shortly after transformers became mainstream in NLP applications, transformer based models for machine vision began to appear. Some of the early

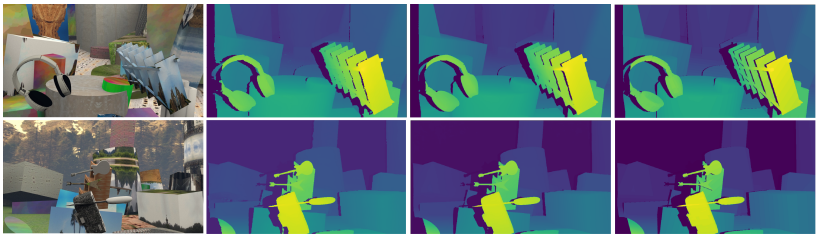

Fig. 1. Samples from SceneFlow Dataset. From left to right: left images, STTR output, output of our model, Ground Truth (GT) disparity.

examples tried to incorporate transformer concepts into convolution [9], however, these solutions did not provide the substantial jump in accuracy seen in NLP. Recently, more pure transformer based machine vision models were developed. Vision Transformer (ViT) [1] is an example model that that tries to be as close as possible to the original transformer. The equivalent for words is patches in the image. However, these solutions still need to be creative to successfully include transformers. For example, most semantic segmentation models use CNN decoders.

In the field of stereo matching and disparity estimation, CNNs reign supreme with most models following the same pipeline of feature extraction, cost volume generation, feature matching, and disparity refinement. All of these steps heavily rely on CNNs, notably the feature matching step which uses 3D CNNs. Therefore, to incorporate transformers, a new feature matching approach must be used. Stereo Transformer (STTR) [10] is one of very few transformer based disparity estimation models. In STTR, the feature matching is done using a dynamic programming method introduced in by Ohta *et. al.* [11] which relies only on the epipolar lines thus making the model blind to any context outside those lines. The solution they came up with is adding a CNN post processing method to regain some global context.

In this paper, we introduce a method to have global context while still keeping the computational complexity low. We also introduce an end to end transformer based model for stereo disparity estimation. The flowcharts are shown in Fig. 2. We achieve state of the art results on the SceneFlow dataset. Samples are shown in Fig. 1.

2 Related Work

2.1 Vision Transformers

Since their inception, transformers became very successful in NLP applications. Models such as BERT [8] and GPT-3 [7] substantially improved the performance of various NLP tasks. In the original paper [6], transformers are described to use self-attention to compute representations of its input and output without the

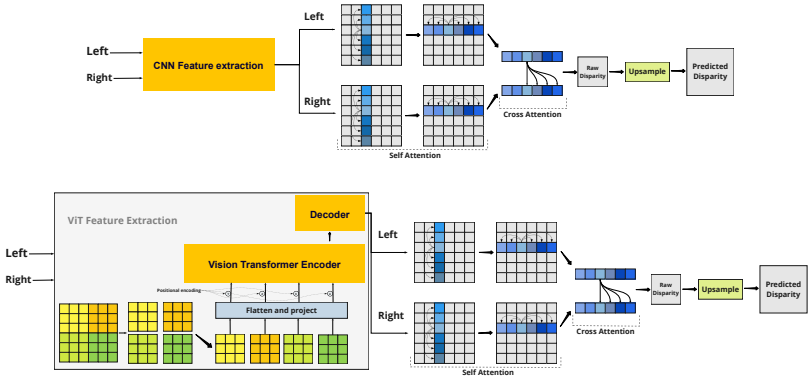

Fig. 2. Architectures of our models. *Top* CNN based model. *Bottom* ViT based model.

use of RNNs or CNNs. Self-attention allows the transformer to learn long-range dependencies in a more effective way compared to RNNs. Therefore, efforts have been made to port this success to computer vision tasks.

In their early form, transformers were used alongside convolution or as a way to tackle the downsides of convolution. Examples of these early methods include deformable convolutions [9] which use self-attention to allow convolution kernels to change shape, and Squeeze-and-Exitation networks [12] which address channel-wise feature responses by selecting information features to emphasise on.

To fully replace convolutional layers, transformers needed to be adapted to image data. Equating pixels to words would result in large memory consumption and a computation complexity of $O(h^2w^2)$ where h and w are the height and with of the image respectively. There are various ways this issue has been handled. Methods applied by Han *et. al.* [13], Zheng *et. al.*, [2] and Strudel *et. al.* [14] among others, tackled this issue by dividing the image into patches essentially mimicking the behaviour of convolutional kernels, however, unlike convolution, these patches do not rely on fixed pattern matching but instead attentively determine aggregation weights, i.e, something akin to a dynamic kernel. Another method called Axial DeepLab introduced by the DeepLab team [15] chose the patches to be 1 pixel wide vectors along the x and y axis consecutively to build full context. However, this method is a hybrid between convolution and self-attention.

2.2 Stereo Disparity estimation

In a stereo image setup, finding the stereo disparity relies on finding a match across these two images for every pixel. Classical methods such as semi-global matching [16] relied on finding differences between patches along the epipolar lines from both images. However, these methods are not reliable when it comes to un-textured objects and noisy images. Deep Learning (DL) methods brought

a substantial improvement over previous methods where a CNN learn to extract features from both images followed by concatenation and finally passing through fully connected layers which learn to find the disparity [17]. Subsequently, CNN based methods advanced further using a cost volume and cost aggregation where a 4D vector is build out of the feature maps where the 4th dimension is the candidate disparities and using a 3D CNN to find the best match. These methods follow a similar pipeline of feature extraction, calculating matching cost, cost aggregation, and disparity refinement [18–20].

To translate stereo disparity estimation to transformers a new approach needed to be followed. Although transformers have not been used much in the subject of disparity estimation, some works have explored and proved its feasibility. Wang *et. al.* [21] proposed a generic parallax-attention mechanism (PAM), which does not rely on self-attention, but only on cross-attention across epipolar lines as a way to match features.

Li *et. al.* [10] (STTR) introduced a full ViT approach to stereo matching where both self and cross-attention are used. STTR relies on finding self and cross-attention across the horizontal lines. The matching is obtained using the dynamic programming method introduced in 1985 [11], where similarities between pixel intensities were used to find the equivalent match. The pixel intensities are replaced in STTR with attention which gives better contextual information for each pixel and long range associations. Since the transformer can only attend to epipolar lines, it misses the context across the $y-axis$ of the feature map. To tackle that issue a post processing convolutional network has been introduced which aims at capturing context from the the original image.

3 Methods

Stereo disparity estimation methods that rely on DNN follow a pipeline consisting of feature extraction, cost volume generation, stereo matching, and disparity calculation. However, using transformers the pipeline is slightly different, where the cost volume steps are omitted in favor of direct matching.

3.1 Feature extraction

To test various options for feature extraction we designed multiple feature extraction models based on both CNNs and ViT. This variation serves to perform an extensive ablation study on what works best for a transformer feature matching.

For our CNN based model, the feature extraction network consists of 42 layers in an encoder-decoder architecture as shown in Fig. 3. Our model is slightly inspired by PSMNet [18]. It is meant to be lightweight keeping the runtime and memory consumption low as well as the resolution of the feature map consistently at $\frac{1}{4}H, \frac{1}{4}W, C$ where H and W are the height and weight of the input images consecutively, and C is the depth of the feature map.

Transformers are notorious for requiring huge datasets for pretraining. Therefore, to select an option for our transformer based feature extraction, we chose

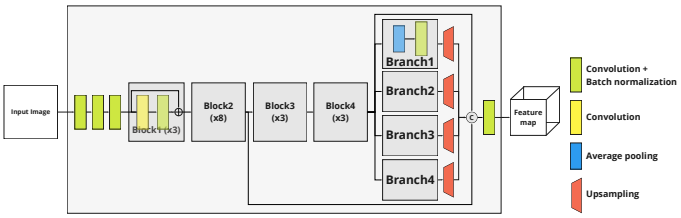

Fig. 3. Architecture of Feature extraction network.

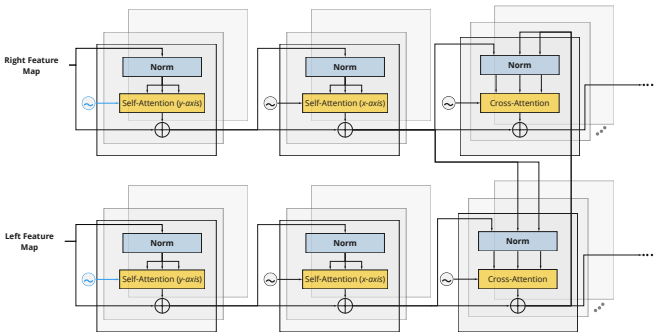

Fig. 4. Architecture of attention layers. The sine sign represents the positional encoding.

Visual Transformer (ViT) since it is pretrained on various large datasets on the task of classification. We use a model pretrained on ImageNet21K [22], which contains over 16 million samples. According to Alexey *et. al.* [1], Larger datasets yield better results, in fact, their in house JFT 300M which contains over 300 million samples, outperformed ImageNet 21K by consistently improving the classification accuracy by up to 5%. We chose ImageNet21K due to its open availability. We modified the ViT model to fit to the task of feature extraction by introducing an encoder-decoder architecture. The model architecture is shown in Fig. 2. The output of the ViT feature extraction is set to match the same output stride and feature depth as the CNN feature extraction. Additionally, we tested the feature extraction network introduced in STTR [10].

3.2 Full context stereo transformers

In CNN based stereo matching, a cost volume is constructed containing all the candidate disparities. However, the model cannot attain to disparities outside the preset maximum disparity. Most CNN models set this maximum value to be 192 or 384. However, even the mainstream datasets, the maximum disparity can surpass 384 pixels. The main reasons why the cost volume is limited are memory

consumption and complexity. Therefore, most models try to find a balance that would achieve the best test results, which would not necessarily translate well to situations where the disparity is higher than the preset limit.

With transformers, the cost volume generation step is avoided in favor of attention layers. Transformers contain 4 main sub-components: the self-attention layers, the cross-attention layers, the positional encoding, and the feed forward layers [6]. STTR [10], replaced the cost volume with dynamic programming [11], which originally relies on comparing the intensity of pixels. Using self-attention, weights are associated with each pixel, giving it richer information than simply its intensity. This extra information may be its relative distance to a landmark or how color changes in nearby pixels. Subsequently, cross-attention layers are used to compare these weighted pixels in the stereo views.

In previous transformer based methods, either local context [21] or epipolar only context is used [10]. Therefore, a lot of information that can be useful in stereo matching is omitted. The perfect scenario would be to include the whole feature map, that is, however, not feasible due to memory and computation constraint. Hence, we developed a method that can leverage the information from not only the epipolar lines ($x - axis$), but also from the vertical lines ($y - axis$). Our method illustrated in Fig. 4 is based on and improves upon the Stereo Transformer (STTR) [10]. As shown in Fig. 4, the feature maps go through self-attention along the vertical $y - axis$ before the horizontal $x - axis$ as shown in Fig. 2. This setup allows the self-attention layers to have information regarding the whole vertical $y - axis$ at each point in the epipolar lines. The intuition is that each pixel in the feature map would first be enhanced with weights from the vertical $y - axis$ before being matched along the epipolar lines.

The attention layers take 3 inputs called the Key, $Query$, and $Value$. These are obtained by passing the feature map through a linear feed forward layer. These terms are borrowed from retrieval systems. The query is what we are searching for, the key is a unique identifier and the value is the main data. The attention layer aims at finding weights to be associated with the values. These weights, as we will see, are obtained by selecting from a set of keys using a query.

The linear equation used in the attention is presented as follows:

$$F(x, W, b) = Wx + b \tag{1}$$

where x is the input, W contains the weights, and b is the bias.

In the case of self-attention layers, the Key, $Query$, and $Value$ are obtained using the output of the previous cross-attention layer or the feature maps in the first layer. For the cross-attention layers, while $Value$ is similarly obtained from the previous self-attention layers, the Key, and $Query$ of the left transformer are obtained from the previous self-attention output of the right transformer and vice-versa.

Each attention layer is divided into 8 heads which take a portion of the feature map. The Key, $Query$, and $Value$ are obtained using the following equations:

$$\begin{aligned} K_i &= F(f, W_K, b_K)_i \\ Q_i &= F(f, W_Q, b_Q)_i \\ V_i &= F(f, W_V, b_V)_i \end{aligned} \tag{2}$$

where $f = o + PE$ with o representing the output of the feature extraction network in the case of the first layer or the output of the previous attention layers. PE is the positional encoding. K_i, Q_i, and V_i are the Key, $Query$, and $Value$ of the ith head, respectively. Therefore, the selection of the $Keys$, $Querys$, and $Values$ is done by the model in the learning phase. The Positional Encoding (PE) is an important part of transformers. The goal of PE is to add information regarding the position of each pixel. For the feature matching network, we used sinusoidal positional encoding similar to the one used in the original Transformer paper [6]:

$$\begin{aligned} PE_{(pos,2i)} &= \sin(pos/10000^{2i/d}) \\ PE_{(pos,2i+1)} &= \cos(pos/10000^{2i/d}) \end{aligned} \tag{3}$$

where pos is the position, i is the dimension, and d is the dimension of the model. Multiple forms of positional encoding can be used. The two requirements for the positional encoding are the ability to represent PE_{pos+k} as a linear function of PE_{pos}, and unambiguity, that is, no two positions have the same encoding. In the feature extraction ViT network, the positional encoding is done using standard learnable 1D position embedding similar to what was introduced in the original ViT paper [1].

The attention layers first extract the attention weights using the following equation:

$$\alpha_i = \frac{Q_i^T K_i}{\sqrt{C_i}} \tag{4}$$

where α_i is the attention weight of the ith head, and C_i is the depth of the ith $value$. The output value V is obtained with the following equation:

$$V = Concat(\alpha_1 V_i, ..., \alpha_H V_H) \tag{5}$$

Finally, the output of each layer is obtained as follows:

$$O = F(V, W_o, b_o) \tag{6}$$

4 Experiments and Results

4.1 Datasets and Metrics

SceneFlow: The SceneFlow dataset [23] comprises of more than 39000 stereo frames in 960×540 pixel resolution. The dataset is divided into 3 subsets. Like most previous works, we use the FlyingThings3D subset with the default training, validation, and test subsets.

KITTI: KITTI dataset [24] consists of 200 training scenes and 200 test scenes obtained from a moving vehicle.

Metrics The metrics used are the percentage of errors larger than 3 pixels, known as 3px error, and Expected Prediction Error (EPE).

4.2 Training

The optimizer used is AdamW, with $1e^{-4}$ as weight decay and a learning rate of $1e^{-4}$. The pretraining is done with a fixed learning rate on SceneFlow for 15 epochs while finetunning is done with a learning rate decay of 0.99 for up to 400 epochs. The training was performed on multiple Nvidia GPUs. The feature extraction transformer has 12 self-attention layers with the output depth being 128 with an output stride of 4. The feature matching has 6 self and cross-attention layers. The output stride for the CNN feature extraction is 4. The image size is the default for SceneFlow and KITTI.

4.3 Experiments

In our ablation studies, we test 3 feature extraction models and 3 feature matching models. This would result in 9 experiments, but, we only test the new attention models with the best performing feature extraction models.

4.4 Results and comparison

Comparison with other methods We compare our method with prior stereo disparity estimation methods, notably, works holding the state of the arts in the datasets we are studying. The architecture of our method is based on CNN feature extraction with 42 layer and feature matching with transformers using self-attention across the vertical $y - axis$, then self-attention across horizontal $x - axis$ followed by cross-attention across the epipolar lines from both views. The results are shown in Table 1.

In the SceneFlow dataset, our method holds the SOA 3px results. The current epe SOA holder is HITNet [25] which—unlike most CNN based methods—does not rely on 3D convolutions. Instead, it employs a fast multi-resolution step followed by geometric propagation. The SOA method for KITTI 2015 is LEAStereo [19], which uses a classical cost volume and 3D convolution for matching. However, it uses Neural Architectural Search (NAS) to find the optimal model within the search space they employed. Our method fails to achieve similar outstanding results with KITTI. The reason, we theorize is the size of the dataset. Although, we pretrained the models on SceneFlow, KITTI still is different enough and does not have as many samples. Therefore, the transformers could not learn enough to overcome CNNs.

Table 1. Results with SceneFlow pretraining

	KITTI 2015			SceneFlow	
	bg	fg	all	epe	3px
STTR [10]	1.70	3.61	2.01	0.45	1.26
PSMNet [18]	1.71	4.31	2.14	1.03	3.60
AANet [26]	1.80	4.93	2.32	0.87	2.38
LEAStereo [19]	**1.29**	**2.65**	**1.51**	0.78	2.60
HITNet [25]	1.74	3.20	1.98	**0.36**	2.21
Ours	2.00	4.20	2.38	0.38	**1.10**

Feature extraction results We tested the feature extraction methods explained previously. We refer to the feature extraction used in STTR as Dense FE, the ViT based feature extraction as ViT FE, and our convolution based feature extraction model as Conv FE. The feature matching architecture used in this section uses self-attention on horizontal $x - axis$ only. Table 2 shows the results obtained using different feature extraction methods.

Table 2. SceneFlow results using different feature extraction networks

Experiment	3px error	EPE	Runtime (s)	Training time (1epoch)
Dense FE	1.26	0.45	0.79	11h28m12s
Conv FE	1.20	0.40	0.52	6h54m40s
ViT FE	1.86	0.52	0.41	2h14m40s

From these results, we can deduce that a very deep CNN feature extraction method with an output stride of 0 is not necessary. Previous works using NAS showed that shallower DNNs can perform better than their deeper counterparts [27].

Transformer matching results We tested multiple transformer architectures. xSAxCA refers to attention across $x - axis$ only in both self and cross-attention. xySAxCA refers to attention across $x - axis$ and $y - axis$ in self-attention and rows only in cross-attention. xySAxyCA refers to attention across $x - axis$ and $y - axis$ in both self and cross-attention.

Intuitively, objects in the $y - axis$ might not seem to be very useful since they change position depending on their disparity. However, we are not trying to match across both views using the y-axis, instead, we are trying to add more data to pixels especially in difficult regions such as texturless ones. To know whether it helps to include the vertical $y - axis$ self-attention prior to the horizontal $x - axis$ self-attention, we visualize the self and cross final attention weights at a certain point in the left image in all the layers in Fig. 5, and Fig. 6. These figures show the weights with the highest values. They give us an idea as to what the transformer is seeing as important landmarks to identify to corresponding

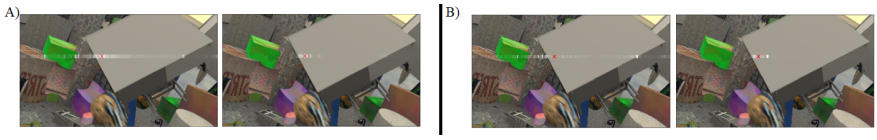

Fig. 5. Self attention of a single pixel (marked with red x).
A): Self attention of pixel from STTR. *B)*: Self attention from out model.
self-attention of pixel in the first and last self-attention layer respectively.

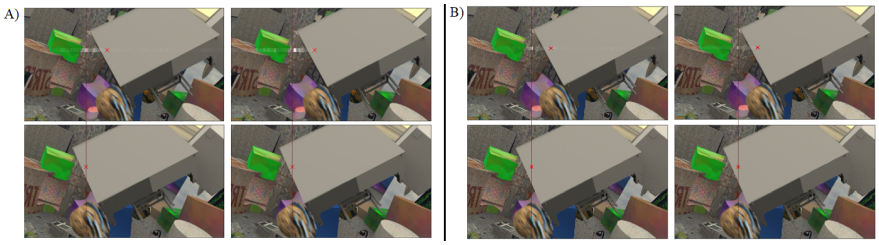

Fig. 6. Cross attention of a single pixel (marked with red x).
A): Cross attention of pixel from STTR. *B)*: Cross attention from out model.
Top: cross-attention of pixel in the first and last cross-attention layer respectively.
Bottom: right image with marked GT corresponding pixel and a line going through the left image showing where the GT is.

pixel in the right view. We can see that with out method, the output in each layer is sharper. That is, the model finds it easier to select landmarks in self attention and finding corresponding pixels in the cross attention. In the cross-attention figure (Fig. 6), we can observe that the STTR algorithm fails to select the correct corresponding pixel while our method is more accurate. The feature extraction used in this section is the Conv FE.

The dataset used for this ablation study is SceneFlow. The results are displayed in Table 3.

Table 3. SceneFlow results using different transformer architecture

Experiment	3px error	EPE
xSAxCA	1.20	0.40
xySAxCA	**1.10**	**0.38**
xySAxyCA	1.15	0.41

We observe that the results improve with the inclusion of a self-attention on the $y - axis$. However, no such improvement is obtained with the inclusion of cross-attention along the $y - axis$.

5 Conclusion

We demonstrated a method that rivals CNNs in stereo matching achieving state of the art in SceneFlow dataset. We introduced a feature matching architecture that leverage the full context of the images. We showed that a shallower feature extraction method is sufficient to achieve good results. Our solution still relies on CNN for feature extraction. However, ViT based feature extraction performed well especially considering it is the fastest configuration which would be beneficial for time critical applications.

References

1. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *CoRR*, vol. abs/2010.11929, 2020. [Online]. Available: https://arxiv.org/abs/2010.11929
2. S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr, and L. Zhang, "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 6881–6890.
3. Y. Zhou, C. Barnes, E. Shechtman, and S. Amirghodsi, "Transfill: Reference-guided image inpainting by merging multiple color and spatial transformations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 2266–2276.
4. F. Yang, H. Yang, J. Fu, H. Lu, and B. Guo, "Learning texture transformer network for image super-resolution," in *CVPR*, June 2020.
5. D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *CoRR*, vol. abs/1409.0473, 2015.
6. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.
7. T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901.
8. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.
9. J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE Computer Society, oct 2017, pp. 764–773.

10. Z. Li, X. Liu, N. Drenkow, A. Ding, F. X. Creighton, R. H. Taylor, and M. Unberath, "Revisiting stereo depth estimation from a sequence-to-sequence perspective with transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 6197–6206.
11. Y. Ohta and T. Kanade, "Stereo by intra- and inter-scanline search using dynamic programming," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-7, pp. 139–154, 1985.
12. J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141.
13. H. Hu, Z. Zhang, Z. Xie, and S. Lin, "Local relation networks for image recognition," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 3463–3472.
14. R. Strudel, R. Garcia, I. Laptev, and C. Schmid, "Segmenter: Transformer for semantic segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 7262–7272.
15. H. Wang, Y. Zhu, B. Green, H. Adam, A. Yuille, and L.-C. Chen, "Axial-deeplab: Stand-alone axial-attention for panoptic segmentation," in *Computer Vision – ECCV 2020*. Springer International Publishing, 2020, pp. 108–126.
16. H. Hirschmuller, "Accurate and efficient stereo processing by semi-global matching and mutual information," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, 2005, pp. 807–814 vol. 2.
17. J. Zbontar and Y. LeCun, "Computing the stereo matching cost with a convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
18. J.-R. Chang and Y.-S. Chen, "Pyramid stereo matching network," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5410–5418.
19. G. Yang, J. Manela, M. Happold, and D. Ramanan, "Hierarchical deep stereo matching on high-resolution images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5515–5524.
20. S. Khamis, S. Fanello, C. Rhemann, A. Kowdle, J. Valentin, and S. Izadi, "Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.
21. L. Wang, Y. Guo, Y. Wang, Z. Liang, Z. Lin, J. Yang, and W. An, "Parallax attention for unsupervised stereo correspondence learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.
22. O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.
23. N. Mayer, E. Ilg, P. Häusser, 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 *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
24. M. Menze, C. Heipke, and A. Geiger, "Joint 3d estimation of vehicles and scene flow," in *ISPRS Workshop on Image Sequence Analysis (ISA)*, 2015.

25. V. Tankovich, C. Hane, Y. Zhang, A. Kowdle, S. Fanello, and S. Bouaziz, "Hitnet: Hierarchical iterative tile refinement network for real-time stereo matching," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 14 362–14 372.
26. H. Xu and J. Zhang, "Aanet: Adaptive aggregation network for efficient stereo matching," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1959–1968.
27. X. Cheng, Y. Zhong, M. Harandi, Y. Dai, X. Chang, H. Li, T. Drummond, and Z. Ge, "Hierarchical neural architecture search for deep stereo matching," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 22 158–22 169.