Fast Depth Estimation for View Synthesis

Nantheera Anantrasirichai
Visual Information Laboratory
University of Bristol
Bristol, UK
n.anantrasirichai@bristol.ac.uk

Majid Geravand
Braendler Engineering
Turin, Italy
majid.geravand@braendler.com

David Braendler
Braendler Engineering
London, UK
david.braendler@braendler.com

David R. Bull
Visual Information Laboratory
University of Bristol
Bristol, UK
dave.bull@bristol.ac.uk

***Abstract*—Disparity/depth estimation from sequences of stereo images is an important element in 3D vision. Owing to occlusions, imperfect settings and homogeneous luminance, accurate estimate of depth remains a challenging problem. Targetting view synthesis, we propose a novel learning-based framework making use of dilated convolution, densely connected convolutional modules, compact decoder and skip connections. The network is shallow but dense, so it is fast and accurate. Two additional contributions - a non-linear adjustment of the depth resolution and the introduction of a projection loss, lead to reduction of estimation error by up to 20% and 25% respectively. The results show that our network outperforms state-of-the-art methods with an average improvement in accuracy of depth estimation and view synthesis by approximately 45% and 34% respectively. Where our method generates comparable quality of estimated depth, it performs 10 times faster than those methods.**

***Index Terms*—depth estimation, disparity estimation, deep learning, CNN, view synthesis**

I. Introduction

In the human visual system, a stereopsis process creates a perception of three-dimensional (3D) depth from the combination of the two spatially separated signals received by the brain from our eyes. The fusion of these two slightly different pictures gives the sensation of strong three-dimensionality by matching similarities. To provide stereopsis in machine vision applications, two images are captured simultaneously from two cameras with parallel camera geometry, and an implicit geometric process is used to extract 3D information from these images. Binocular disparity d is computed and depth z is obtained from (1),

$$z = \frac{fB}{d}. \tag{1}$$

where f and B are a focal length and a baseline between two cameras, respectively.

3D information, or depth, is utilised in many applications, including 3D reconstruction [1], view synthesis [2], object recognition [3] and multi-view video compression [4]. Traditional methods search the corresponding points between the left and the right images using block-based [4], [5] or mesh-based matching [2]. More sophisticated approaches, e.g. dynamic programming [3], [6], produce better results as they do not introduce blocking artefacts or noisy depth maps. Most methods however involve an iterative process to minimise an error function, to further refine the depth map, particularly around the edge of the object [7], and to improve geometric projection [8]. Such iterations are time-consuming and are not suitable for real-time applications. For example, a 3D patch-based minimum spanning tree (3DMIST [9]), one of the top five in Middlebury Stereo benchmark [10], takes about 25 sec to process one 450×350 image pair. On the KITTI benchmark, where the stereo pairs are captured in driving scenes [11], convolutional neural networks (CNNs) have shown significantly faster computation (<2.5 sec at the resolution of 1392×512 pixels). However, the estimated disparity maps are mainly used for visual odometry, 3D object detection and 3D tracking, not for 3D reconstruction or view synthesis, where precise estimation at depth discontinuities is crucial.

In this paper, we present a new learning-based approach that achieves both high quality estimated depth for view synthesis and fast computation. We adapt a DenseMapNet [12] with additional compact decoder and skip layers to include the low-level features for finer estimation. Therefore the network is significantly shallower than many state-of-the-art methods, whilst producing comparably accurate depth results. Stereo matching is difficult in homogeneous areas and traditional methods solve this issue by using large windows. In the CNN, this can be solved using a big receptive field, so we employ several dilation rates to capture disparity. As our depth estimation method is intended to be used for view synthesis, we herein propose an exponential adjustment for depth values during training process. This will concentrate more on the near objects, which are more salient than the far ones and the background. In addition, we propose a projection loss, where the weights in the convolution layers are also adjusted according to the error from the synthesised image.

The remainder of this paper is organised as follows. Related work on CNN-based depth estimation is presented in Section II. The proposed scheme is described in Section III. The performance of the method is evaluated in Section IV. Finally, Section V presents the conclusions of this work.

II. CNNs for disparity/depth estimation

CNNs were first introduced for stereo matching by Zbontar and LeCun [13] to replace the computation of the matching cost and to learn a similarity measure on small image patches. This method significantly accelerates the process of disparity estimation; as a result, most recent methods employ CNNs.

This work was supported by Innovate UK: Palantir – Real time inspection and assessment of wind turbine blade health.

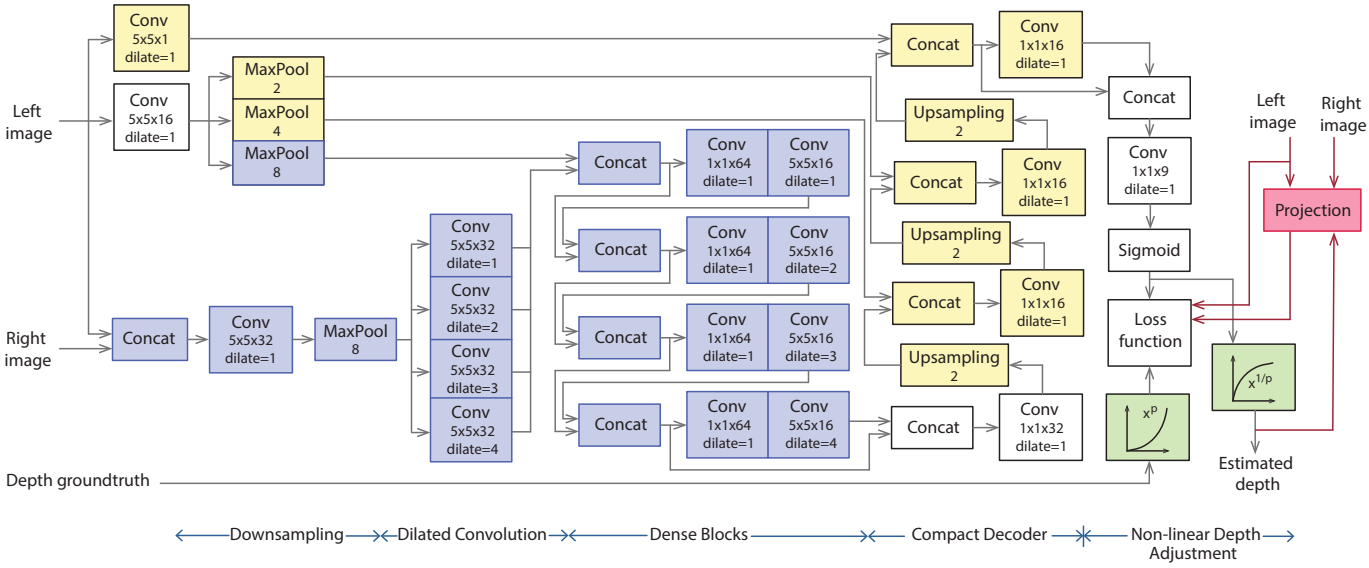

Fig. 1. Proposed network architecture for depth estimation. The blue blocks are for feature matching, the yellow blocks are additional compact decoder part to ensure sharp upsampled feature maps, and the green blocks enhance depth resolution of the near objects. Each Conv module comprises convolution, batch normalisation, and ReLu layers.

Generally, the learning-based disparity estimation methods comprise two modules: i) *feature extraction*, applied to the left and the right image pair separately, but with the learnable weights and biases shared between them in the training process (e.g. siamese networks), ii) *disparity estimation*, where the output of the feature extraction process, *a cost volume*, is employed to compute a disparity map. The 3D cost volume stores the costs for choosing a disparity value for each pixel. The simplest cost is the absolute differences of the intensities. The size of the cost volume thus increases with the search space.

As the accuracy of the estimated disparity is significantly improved and real-time computation becomes feasible, CNNs have been gained more attention. A number of network architectures have been proposed, along with slight changes of parameters and transfer learning for specific applications. For example, a pyramid stereo matching network (PSMNet) [14] exploits global context information in stereo matching using pyramid pooling module and dilated convolution is applied to further enlarge the receptive field. It employs a stacked hourglass architecture [15] to estimate disparity values. The Sparse Cost Volume Network (SCV-Net) [16] was proposed to improve complexity efficiency by shifting feature maps with a stride of 3. Guided Aggregation Net (GA-Net) [17] employs a stacked hourglass CNN to extract features of the left and right image pair, giving the output as a 4D cost volume. The cost aggregation module then calculates a disparity map.

III. Proposed Method

A. Network architecture

The proposed network architecture is shown in Fig. 1. We adapt the correspondence network from DenseMapNet [12] (blue blocks in Fig. 1) to find correspondences of the input stereo pair. The matching process of corresponding points between the left and the right views is performed at a lower resolution, and down-sampled via a max-pooling layer (a downscale factor of 8 is used throughout this paper). This improves computation speed, reduces memory requirements, and overcomes problems of large disparities. Then, dilated convolution with different dilation factors (l=1-4) is applied. Dilated convolution enlarges the field of view of the filters to incorporate larger context by expanding the receptive field without loss of resolution. The dilated convolutions are defined in Eq.2 [18], where F is a feature map, k is a filter, $*_l$ is a convolution operator with a dilation factor l.

$$(F *_l k)(\mathbf{p}) = \sum_{\mathbf{s}+l\mathbf{t}=\mathbf{p}} F(\mathbf{s})k(\mathbf{t}). \tag{2}$$

Four one-layer Dense Blocks [19] are then employed to capture corresponding features. A Dense Block uses feature maps from multiple preceding layers as inputs leading to more connections amongst layers. Subsequently, the feature maps are enlarged to the original resolution. Instead of applying upsampling only once like in DenseMapNet, we propose a compact decoder using a step-wise upsampling of 2 (yellow blocks in Fig. 1). In addition, we add skip connections by merging the low-level feature maps of the left image in every upsampling step. This ensures pixel-wise co-locations between the RGB image and the depth map in both full resolution and feature levels.

If the ground truth data for training is based on disparity values, it is scaled to $[0, 1]$, equivalent to $[0, d_{\max}]$, where $d_{\max}$ is the maximum value of disparity. If the network is to estimate depth, the ground truth $\hat{z}$ is normalised and subtracted from 1, i.e. $\hat{z} = 1 - \frac{z}{z_{\max}}$. The prediction output of the network is done

via a Sigmoid activation function, $\hat{x} = (1 + e^{-x})^{-1}$, which scales the output x of the last convolution layer to $[0, 1]$.

B. Depth adjustment

For view synthesis, foreground objects are often more salient and incorrect depth estimates can result in noisy visualisation, particularly at the edges of the objects or where there exists discontinuity of the depth. Here we propose a nonlinear adjustment to the depth ground truth (green blocks in Fig. 1) so that the closer objects have higher depth resolution than distant objects or background. During the training process, exponentiation is applied to the ground truth $\hat{z}$, as in (3), where p is an exponent. For the prediction process, the final estimated depth $\tilde{z}$ is computed using (4), where $\hat{x}$ is the output of the Sigmoid activation layer.

$$\hat{z}' = \hat{z}^p,\ \hat{z} \in [0, 1] \text{ and } p \geqslant 1. \tag{3}$$

$$\tilde{z} = \hat{x}^{\frac{1}{p}},\ \hat{x} \in [0, 1]. \tag{4}$$

Fig. 2 (left) demonstrates how $\hat{z}$ values are adjusted with the exponential function when p=1.5. The blue plot shows that $\hat{z}$ values close to 1 (areas near to the cameras) are stretched out gaining higher resolution, whilst the values close to 0 are shrunk (areas far from the cameras). This technique improves the validation loss by approximately 15%, as shown in Fig. 2 (right). The optimal value of p depends on applications and the positions of the salient objects in the scene. We initialise the p value using curve fitting to the histogram of the training ground truth.

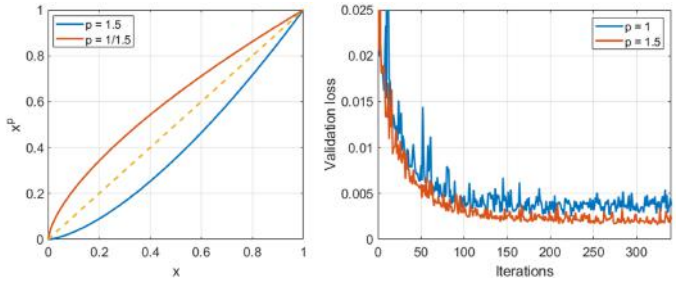

Fig. 2. Exponential functions (left) and validation losses with and without depth adjustment.

C. Loss function

Most networks for disparity estimation mentioned in Section II employ a smooth ℓ_1 loss function (mean absolute error) as it is robust to outliers and disparity/depth discontinuities. However, outliers can still produce undesirable projected pixels - seen as noisy edges in the synthesised views. Therefore we employ the ℓ_2 loss function (mean square error). Note that we tested several loss functions, including cross entropy, ℓ_1, ℓ_2 and perceptual loss with pre-trained VGG [20]. While the qualities of estimated depths do not differ significantly, the best view synthesis is accomplished with the ℓ_2 loss.

In this paper, we propose using the ℓ_2 losses calculated from the predicted depth map L_{ℓ_2} and the reconstructed left image from the right image, referred as the projected loss $L_{R\to L}$. The prediction loss L_{ℓ_2} equals to $\sum(\hat{z} - z)^2$, where z and $\hat{z}$ are real and estimated depth values. For the projection loss, if needed, the depth is first converted to disparity d through the relationship in (1), i.e. $\hat{d} = fB/\hat{z}$. The pixel $(i,\ j)$ on the reconstructed left image is derived from the pixel $(i - \hat{d},\ j)$ on the right image. The final loss function is a weighted combination between two losses as in (5), where α_z and α_p are the weights of the prediction and projection losses (We simply use α_z=α_p=1 in this paper). I^L, I^R, N_z, and N_p are the left image, the right image, the total number of the pixels on each image, and the total number of existing pixels on the reconstructed left image, respectively. Experimental results show that adding the projected loss improves the prediction performance by approximately 20%, compared to using L_{ℓ_2} alone.

$$\begin{aligned} L &= \alpha_z L_{\ell_2} + \alpha_p L_{R\to L} \\ &= \frac{1}{N_z}\sum_k (\hat{z}_k - z_k)^2 + \frac{1}{N_p}\sum_{ij}(I^R_{i-\hat{d},j} - I^L_{ij})^2 \end{aligned} \tag{5}$$

IV. Results and Discussion

The proposed network is implemented on Keras with a Tensorflow backend (available at https://github.com/pui-nantheera/DepthEstimation). The network is trained with the Adam optimizer, which is an extension to stochastic gradient descent (SGD), in which the procedure updates network weights iteratively based on training data. To prevent overfitting, we drop 20% of features extracted in the convolution layers. The model is initialised by training the network with our synthetic datasets, described in Section IV-A, and subsequently using it as the initial model when training with other test stereo sequences. This transfer learning strategy by fine-tuning a pre-trained network speeds up the training process. Referring the validation loss of p=1.5 in Fig. 2, the estimated depth maps during the training are shown in Fig. 3. The depth is quickly learnt, getting sharper and settles around the 250$^{\text{th}}$ iteration. Experimental results and state-of-the-art comparison of depth estimation and view synthesis are presented in Section IV-B.

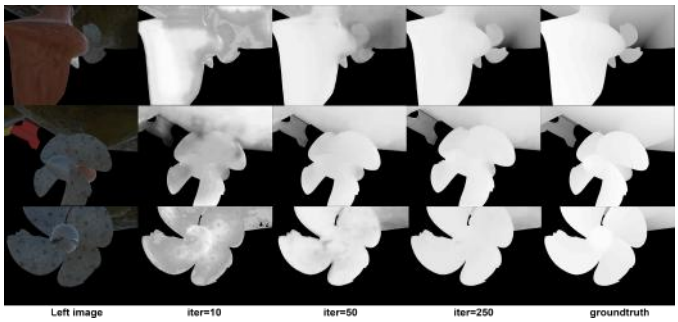

Fig. 3. Estimated depth maps of *Propeller* dataset at the iterations of 10, 50 and 250 (column 2–4) of the stereo pair, where the left image and the true depth are shown in the first and the last column, respectively. Note that the intensity is normalised and adjusted for visualisation.

A. Synthetic datasets

For robustness, we include eight scenes of simulated stereo sequences (total 25,000 stereo pairs with a resolution of 480×640 pixels), created using Unity software [21] and the Elastic Fusion algorithm [22] by Braendler Engineering[1]. Three cameras are attached together. Two of them are identical and used to capture RGB images with parallel camera configurations, whilst the other captures depth images (range 0-15m). The depth images are recorded, corresponding to the left image. All control parameters are set to replicate the real scenario. The examples of the synthetic datasets are illustrated in Fig. 3 and 4. The scenes include both simple and complicated structures, with narrow objects at different depths. The cameras are moved around the target generating various values. These datasets are available on https://go.aws/37zlsTs.

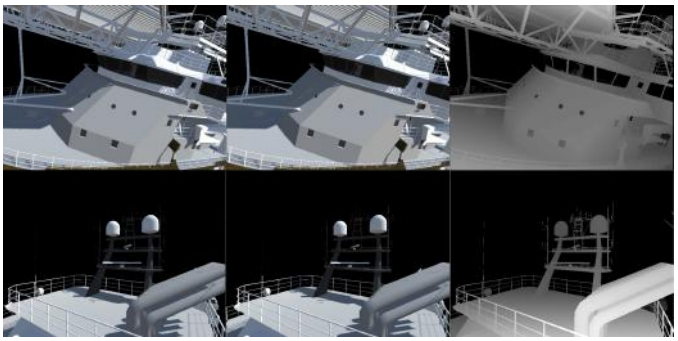

Fig. 4. Stereo pairs of '*Ship*' and '*Antenna*' scenes in the top and bottom rows, respectively. The depth maps are shown in the last column where the intensity represents the depth – brighter indicating closer to the camera.

B. Performance

We tested our network with standard test sequences, namely i) *Sintel* [23], containing animated humans, animals in various types of background scenes, ii) *Driving* and *Monkaa* [24], containing different sizes of objects and visually challenging fur. The *Driving* scenes replicate the KITTI2015 dataset [11], but provide dense disparity groundtruth. We also include one of our synthetic datasets, *Propeller*, for testing. For this dataset, the depth maps are estimated, instead of disparity maps. We randomly selected 90% of the stereo pairs for training and used the remainder for testing.

The performance of the proposed network was compared with three state-of-the-art methods, i.e. DispNet [24], DenseMapNet [12], and PSMNet [14]. The results of the depth/disparity estimation are shown in Table I. We also compared the quality of synthesised views generated using the estimated depth or disparity. Table I shows the mean absolute error (MAE) of the synthesised right images projected from the left images. The results show that our network achieves the best performance on the *Driving* and *Propeller* datasets, whilst the PSMNet outperforms others on the *Sintel* and *Monkaa* datasets. However, when comparing the results of view synthesis, our network and PSMNet shows an insignificant difference on the *Sintel* and *Monkaa* datasets, but our network outperforms the PSMNet by approximately 20% and 10% on he *Driving* and *Propeller* datasets, respectively. This is because our method places greater emphasises on the foreground.

[1]http://braendler.com/home

TABLE I
PERFORMANCE COMPARISON SHOWING MEAN END-POINT-ERROR (EPE) OF DEPTH/DISPARITY ESTIMATION ($\hat{z}$) AND MEAN ABSOLUTUE ERROR (MAE) OF RECONSTRUCTED RIGHT VIEW ($\hat{I}^R$). THE EPES OF SINTEL, DRIVING, AND MONKAA ARE IN PIXELS, WHILST THOSE OF PROPELLER ARE IN CM.

Method	*Sintel*		*Driving*		*Monkaa*		*Propeller*	
	$\hat{z}$	$\hat{I}^R$	$\hat{z}$	$\hat{I}^R$	$\hat{z}$	$\hat{I}^R$	$\hat{z}$	$\hat{I}^R$
DispNet	5.38	9.87	15.62	13.93	5.99	12.84	9.73	59.34
DenseMapNet	4.41	8.34	6.56	9.43	4.45	9.78	8.67	54.69
PSMNet	**3.85**	**7.94**	8.12	10.32	**3.88**	**7.36**	2.38	41.22
Proposed	3.95	7.99	**6.42**	**8.24**	4.08	7.93	**2.32**	**37.31**

Amongst the state-of-the-art methods, DenseMapNet was reportedas having the fastest speed (less than 0.03 sec per stereo pair using NVIDIA GTX 1080Ti) [12], [14], [24]. This is followed by DispNet and PSMNet, of which the runtimes are approximately twice and ten times that of DenseMapNet, respectively. We tested DenseMapNet on an NVIDIA Tesla M60 (which is less powerful than NVIDIA GTX 1080Ti) and found that it has a computational time of 0.20 sec per image pair. Our network processes one stereo pair within 0.21 sec which is only 5% slower than the DenseMapNet, whilst the quality improvement is in excess of 70%. Comparing to PSMNet, the accuracies of disparity estimation for *Sintel* and *Monkaa* datasets are 4% more than that of our method, whilst the performance is about 2% worse when estimating depths of *Propeller*. This could be because the PSMNet is designed for disparity estimation and the precision of the results is limited to 1 pixel. In contrast, our method can estimate the depth values directly, which means the precision can be up to 64 bits using double-precision floating-point format. Note that PSMNet takes more than 2 sec per stereo pair using NVIDIA GTX 1080Ti platform, which is 10 times more than our network, whilst realising comparable accuracy.

The synthesised right views are shown in Fig. 5. The images are generated from the left images using the groundtruth disparity maps, and those estimated from DenseMapNet [12] and our method. Our synthesised results show better quality, particularly around the edges of the objects.

V. CONCLUSIONS

This paper presents a new architecture to estimate depth or disparity targetted at view synthesis applications. The proposed network is employs densely connected convolutional modules and dilated convolutions. We improve the estimation by adding a compact decoder and connecting it with the low-level features via skip layers. Additionally, we modify the depth values during training so that the near objects, which are more salient, have better depth resolution. We

(a) (b) (c) (d) (e)

Fig. 5. View synthesis of *Sintel* and *Driving*. (a) Left image. (b) Right image. (c) Reconstructed right image using groundtruth (d) Reconstructed right image using estimated depth map by DenseMapNet [12]. (e) Reconstructed right image using estimated depth map by our method.

also propose a new loss function that integrates a projection loss to maximise the quality of view synthesis. The results show an improvement in prediction accuracy by up to 45%. Moreover, it significantly improves the perceived quality of 3D reconstruction.

REFERENCES

[1] Yi Zhou, Guillermo Gallego, Henri Rebecq, Laurent Kneip, Hongdong Li, and Davide Scaramuzza, "Semi-dense 3d reconstruction with a stereo event camera," in *European Conf. on Computer Vision (ECCV)*, 2018.

[2] Guilherme P. Fickel and Claudio R. Jung, "Disparity map estimation and view synthesis using temporally adaptive triangular meshes," *Computers & Graphics*, vol. 68, pp. 43 – 52, 2017.

[3] Yizhou Wang, M. Brookes, and P. L. Dragotti, "Object recognition using multi-view imaging," in *2008 9th International Conference on Signal Processing*, Oct 2008, pp. 810–813.

[4] N. Anantrasirichai, C. N. Canagarajah, D. W. Redmill, and D. R. Bull, "In-band disparity compensation for multiview image compression and view synthesis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 4, pp. 473–484, April 2010.

[5] Dimitrios Tzovaras, Michael G. Strintzis, and Haralambos Sahinoglou, "Evaluation of multiresolution block matching techniques for motion and disparity estimation," *Signal Processing: Image Communication*, vol. 6, no. 1, pp. 59 – 67, 1994.

[6] N. Anantrasirichai, C. N. Canagarajah, D. W. Redmill, and D. R. Bull, "Dynamic programming for multi-view disparity/depth estimation," in *IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, May 2006, vol. 2, pp. II–II.

[7] R. Fan, X. Ai, and N. Dahnoun, "Road surface 3d reconstruction based on dense subpixel disparity map estimation," *IEEE Transactions on Image Processing*, vol. 27, no. 6, pp. 3025–3035, June 2018.

[8] S. Ince, E. Martinian, S. Yea, and A. Vetro, "Depth estimation for view synthesis in multiview video coding," in *2007 3DTV Conference*, May 2007, pp. 1–4.

[9] Lincheng Li, Xin Yu, Shunli Zhang, Xiaolin Zhao, and Li Zhang, "3d cost aggregation with multiple minimum spanning trees for stereo matching," *Appl. Opt.*, vol. 56, pp. 3411–3420, 2017.

[10] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, vol. 47, pp. 7–42, 2002.

[11] Moritz Menze and Andreas Geiger, "Object scene flow for autonomous vehicles," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[12] R. Atienza, "Fast disparity estimation using dense networks," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 3207–3212.

[13] Jure Žbontar and Yann LeCun, "Stereo matching by training a convolutional neural network to compare image patches," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2287–2318, 2016.

[14] J. Chang and Y. Chen, "Pyramid stereo matching network," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2018, pp. 5410–5418.

[15] Alejandro Newell, Kaiyu Yang, and Jia Deng, "Stacked hourglass networks for human pose estimation," in *Computer Vision - ECCV 2016*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, Eds., Cham, 2016, pp. 483–499, Springer International Publishing.

[16] Chuanhua Lu, Hideaki Uchiyama, Diego Thomas, Atsushi Shimada, and Rin ichiro Taniguchi, "Sparse cost volume for efficient stereo matching," *Remote sensing*, vol. 10, no. 11, pp. 1–12, 2018.

[17] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip H.S. Torr, "GA-Net: Guided aggregation net for end-to-end stereo matching," *arXiv:1904.06587v1*, 2019.

[18] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *International Conference on Learning Representations*, 2016.

[19] G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 2261–2269.

[20] Justin Johnson, Alexandre Alahi, and Li Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*, 2016.

[21] Unity, "Unity real-time development platform," 2019.

[22] Thomas Whelan, Renato F Salas-Moreno, Ben Glocker, Andrew J Davison, and Stefan Leutenegger, "Elasticfusion: Real-time dense slam and light source estimation," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1697–1716, 2016.

[23] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black, "A naturalistic open source movie for optical flow evaluation," in *Computer Vision – ECCV 2012*, Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, Eds., Berlin, Heidelberg, 2012, pp. 611–625, Springer Berlin Heidelberg.

[24] 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, arXiv:1512.02134.