symmetry

MDPI

Article

BlockNet: A Deep Neural Network for Block-Based Motion Estimation Using Representative Matching

Junggi Lee [1,†], Kyeongbo Kong [1,†], Gyujin Bae [2] and Woo-Jin Song [1,*]

[1] Department of Electrical Engineering, Pohang University of Science and Technology, San 31, Hyoja-dong, Nam-gu, Pohang, Gyungbuk 37673, Korea; leejk12@postech.ac.kr (J.L.); kkb4723@postech.ac.kr (K.K.)
[2] LG Display Co., Ltd., E2 Block LG Science Park, 30, Magokjungang 10-ro, Gangseo-gu, Seoul 07796, Korea; gyujin.bae@lgdisplay.com
* Correspondence: wjsong@postech.ac.kr
† The authors contribute equally.

Received: 25 April 2020; Accepted: 17 May 2020; Published: 20 May 2020

Abstract: Owing to the limitations of practical realizations, block-based motion is widely used as an alternative for pixel-based motion in video applications such as global motion estimation and frame rate up-conversion. We hereby present BlockNet, a compact but effective deep neural architecture for block-based motion estimation. First, BlockNet extracts rich features for a pair of input images. Then, it estimates coarse-to-fine block motion using a pyramidal structure. In each level, block-based motion is estimated using the proposed representative matching with a simple average operator. The experimental results show that BlockNet achieved a similar average end-point error with and without representative matching, whereas the proposed matching incurred 18% lower computational cost than full matching.

Keywords: motion estimation; block-based motion; block matching; representative matching; deep neural network

1. Introduction

Motion estimation is a process that searches for movement between two sequential images. It is widely used in video applications such as video compression [1], global motion estimation [2,3], and frame rate up-conversion [4,5]. Recently, deep neural networks, which exhibit superiority in the field of computer vision [6–12], were used to estimate motion [13–16]. Dosovitskiy et al. [13] proposed FlowNet by directly applying a deep network to estimate motion from input image pairs. Ilg et al. [14] proposed FlowNet2, which was designed as a cascade of FlowNet. The first network in FlowNet2 estimates the motion between the current and reference images. This motion is exploited to warp the reference image, which constitutes the input to the following network. Although FlowNet2 achieved outstanding performance, the network had over 160M parameters. Recently, Sun et al. [15] and Hui et al. [16] proposed lightweight and effective networks that estimate the coarse-to-fine motion by exploiting a pyramidal structure for features named PWC-Net and LiteFlowNet, respectively. However, the aforementioned methods suffer from extremely high computational costs and large numbers of model parameters because they estimate motion in the pixel domain, which constitutes a limitation in practical realizations.

Block-based motion is widely used as an alternative to pixel-based motion [17]. To estimate block- based motion, block-matching algorithms, where blocks of two sequential images are compared, are usually used owing to their simplicity. A naïve block-matching method consists of exploiting all pixels of each candidate block for matching. Although the naïve block-matching method can be used to find the optimal motion between input image pairs, the computational complexity is high. To reduce

this complexity in a matching manner, several alternative methods using representative values for each block were studied, such as sub-block mean [18], certain patterns [19], and some noticeable pixels [20]. However, because these methods matched each block of input image pairs in the intensity domain, it is difficult to find the effective values for representing the block.

In this paper, we propose a new deep neural architecture for block-based motion estimation. Our contribution is as follows. First, unlike conventional representative matching in the intensity domain [18–20], we conduct representative matching in the feature domain where the features are obtained from convolutional neural networks (CNNs). Owing to the powerful ability of CNNs to represent features, the proposed representative matching achieves similar performance to naïve full matching with lower computation complexity. Secondly, it can be easily implemented by the typical pooling operator used in deep learning because the process to find the representative value in each feature is shared. Finally, to maximize the efficiency of the proposed representative matching, we optimize deep neural network using dilated convolution, which can expand the receptive field preserving feature shape without the additional computation and pyramidal structure that causes BlockNet to use a small search range.

2. BlockNet

Figure 1 shows the architecture of BlockNet, the proposed block-based motion estimation network using representative matching. Next, we explain each component of BlockNet in detail.

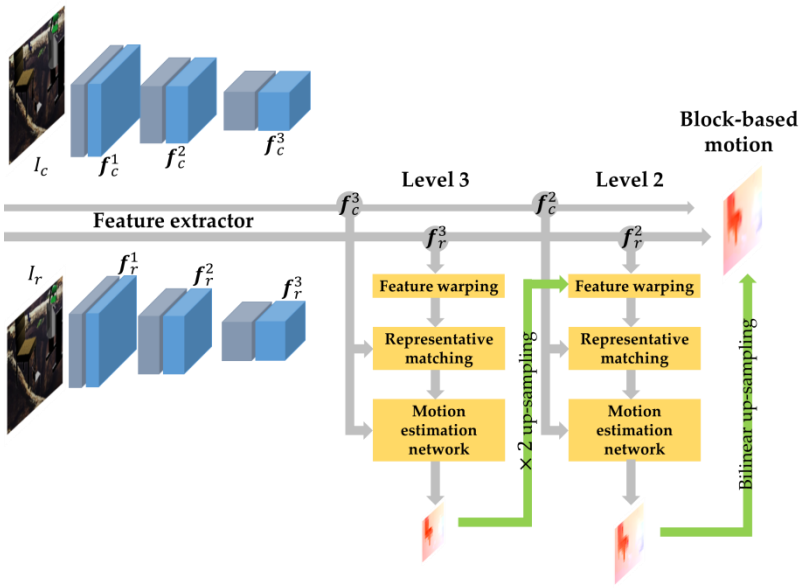

Figure 1. Overall architecture of BlockNet. It first extracts features for a pair of input images using shared weights and then estimates coarse-to-fine motion from top to bottom. To estimate the residual motion at levels 3 and 2, the current level of BlockNet uses the reference feature warped by the motion from the previous level. In each level, block-based motion is estimated using the proposed representative matching with a simple operator and a motion estimation network.

2.1. Feature Extractor

When a current image I_c and reference image I_r are given as input, BlockNet extracts features using three convolutional layers that construct compact architecture. In each layer, a convolution with a filter size 3×3 and stride of 2 is first used to obtain the appropriate spatial shape in each level; a dilated convolution with filter size of 3×3 and rate of 2 is then used to enlarge the receptive field while retaining the spatial shape. The numbers of filters in each convolutional layer are 16, 32, and 64, respectively, similar to [15]. The filter weights in each convolutional layer are shared across two input images, namely I_c,I_r, to extract features with common patterns.

2.2. Representative Matching for Motion Estimation

2.2.1. Proposed Algorithm

In a compact deep neural network, a hand-designed architecture such as the matching approach may perform better to estimate motion [13]. Based on this, it is reasonable for our network to utilize a standard method of block-matching motion estimation. Block matching consists of finding the most similar block by comparing the block in the current feature and candidate blocks of the reference feature in the search range (Figure 2, left side).

For pixel matching, the matching cost is defined in [13] as the correlation of vectors corresponding to each pixel of the current and reference features. This correlation is accumulated for the search range of each pixel. As a result, a 3D cost volume, which has dimensions of $R^2 \times H \times W$, where $R \times R$ is the search range and H and W respectively denote the height and width of the features, is constructed. When this pixel matching is applied on block matching, the matching cost is calculated at block level instead of pixel level. Thus, the 3D cost volume has dimensions $R^2 \times \frac{H}{d} \times \frac{W}{d}$, where $d \times d$ is block size, that is:

$$\textbf{\textit{Cost Volume}}(i,j) = \frac{1}{Nd^2}\left(\boldsymbol{b}_i^c\right)^{\mathrm{T}}\boldsymbol{b}_j^r \tag{1}$$

where $\boldsymbol{b}_i^c$ is the column vector vectorizing the i-th block of the current feature, $\boldsymbol{b}_j^r$ is the column vector vectorizing the j-th candidate block of the reference feature in the search range, and Nd^2 is the dimension of the column vector.

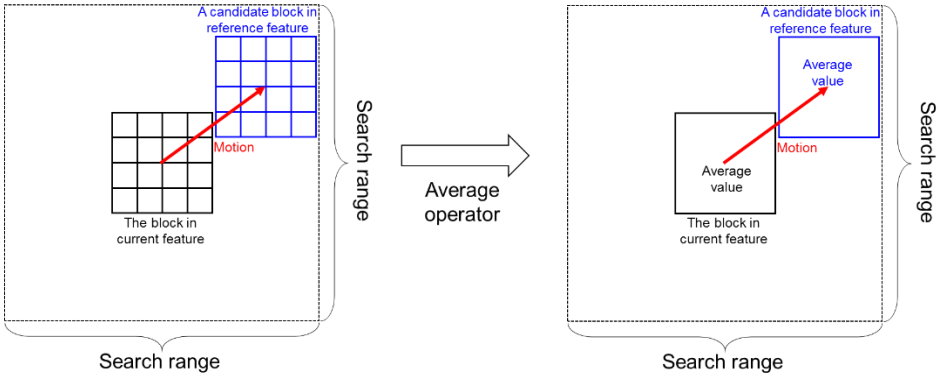

Figure 2. Generation procedure of the matching cost between the block in the current feature and a candidate block in the reference feature: (**left**) conventional block matching method, (**right**) the proposed representative matching using the average operator.

When the block size is $d \times d$ and the search range is $R \times R$, the number of multipliers for a block matching is $Nd^2 \times R^2$. If the block size is larger, the total number of the computation is much higher and thus it may not be suitable for practical realizations. To reduce the computation in the block matching, we could find the representative value in the block. In this study, we propose a representative matching method using the simple average operator defined next (see Figure 2):

$$\textbf{\textit{Cost Volume}}(i,j) = \frac{1}{N}\left(\overline{\boldsymbol{b}}_i^c\right)^{\mathrm{T}}\overline{\boldsymbol{b}}_j^r \tag{2}$$

where $\overline{\boldsymbol{b}}_i^c$ and $\overline{\boldsymbol{b}}_j^r$ are the average of the column vectors $\boldsymbol{b}_i^c$ and $\boldsymbol{b}_j^r$, respectively. Owing to the proposed representative matching, the number of multipliers for block matching is reduced as much as $d \times d$.

In a conventional block-matching algorithm using representative values of the block [18], the average value of the block is also used to reduce the computational cost. However, because the representative matching is applied in the intensity domain, it is insufficient to represent the block. By contrast, because the image can be analyzed as various features through a CNN, our representative matching works well in the feature domain.

With the 3D cost volume and current feature as input, block-based motion is obtained using a CNN with filter size 3 × 3 and stride 2 (Figure 1, motion estimation network). The numbers of filters at each convolutional layer are 32, 24, 16, and 8, respectively.

2.2.2. Implementation Details

Instead of extracting the representative values whenever performing the block-level matching, it can be considered that the representative value of each feature is first extracted, and then matching is performed to implement efficiently. To this end, the average-pooling operator, which is widely exploited in the deep-learning framework, can be used.

The implementation procedure of the proposed representative matching is described in Algorithm 1. Because the representative values of the block in the current and reference features should be extracted at intervals of block size and pixel, respectively, the stride of the average-pooling operator is set to block size and 1, respectively. Moreover, the average-pooling size is the same as the block size (steps 1 and 2). Then, the 3D cost volume is obtained by a matching process that extracts the patches from each average-pooled feature in the search range (steps 3–5) and multiplies them (step 6).

Algorithm 1. Proposed representative matching

Definition
$avg\_pool(\cdot)$: The average-pooling operator
$repeat(\cdot)$: Duplication of each element of the matrix
$extract\_patch(\cdot)$: Extraction of the patches repeatedly
Input: current feature f_c, reference feature f_r, block size $d \times d$, and search range $R \times R$

1. $\bar{f}_c = avg\_pool\left(f_c,\ size = [d,d],\ stride = [d,d]\right)$
2. $\bar{f}_r = avg\_pool\left(f_r,\ size = [d,d],\ stride = [1,1]\right)$
3. $\bar{f}_c^{repeat} = repeat\left(\bar{f}_c,\ size = [R,R]\right)$
4. $p_c = extract\_patch(\bar{f}_c^{repeat},\ size = [R,R],\ stride = [R,R])$
5. $p_r = extract\_patch\left(\bar{f}_r,\ size = [R,R],\ stride = [d,d]\right)$
6. ***Cost Volume*** $= p_c * p_r$

Output: ***Cost Volume***

2.3. Pyramidal Structure with Feature Warping

To maximize the efficiency of our representative matching, we adopted the pyramidal structure in PWC-Net [15]. At level l, the reference feature is warped toward the current feature using a ×2 up-sampled motion estimated from the previous level (Figure 1, feature warping). We first estimate the motion utilizing a 3-level pyramid structure with $\left\{f_c^3, f_r^3\right\}$, and $\left\{f_c^2, f_r^2\right\}$ among the 4 possible levels. We then simply up-sample the estimated motion as much as the remaining levels to obtain the final motion. This architecture can reduce the computation complexity while obtaining the motion with a similar accuracy to that reported in [13].

3. Experiments

3.1. Experimental Setup

To train BlockNet, we used the FlyingChairs dataset [13], which is composed of 22,872 image pairs with ground-truth motion. We cropped 384 × 512 images to 384 × 448 patches and used 90% and 10% of the dataset for training and to test, respectively. We used the multi-scale training loss $\mathcal{L}(\theta)$ described in [15] as follows:

$$\mathcal{L}(\theta) = \sum_{l} \alpha_l \sum_{\boldsymbol{x}} \| \boldsymbol{MV}^l_{\theta}(\boldsymbol{x}) - \boldsymbol{MV}^l_{GT}(\boldsymbol{x}) \|_2 + \gamma \| \theta \|_2 \quad (3)$$

where θ is the network parameter, α_l is the loss weight for layer l, $\boldsymbol{x}$ is the block index, $\boldsymbol{MV}^l_{\theta}$ is the estimated block-based motion vector in layer l, $\boldsymbol{MV}^l_{GT}$ is the ground-truth block-based motion vector in layer l, $\| \cdot \|_2$ is the L_2 norm operator, and γ is the regularization parameter. To obtain the ground truth of block-based motion, we down-sampled the pixel-level ground-truth motion by a factor given by the block size. As in [15], the ground truth was down-sampled by a factor of 2 at each level. Moreover, it was identically scaled by $1/20$ at all levels. This made the estimated motion have identical scale at all levels. Thus, the up-sampled motion had to be scaled from the previous level before passing through the warping operator. We set the scale values for the up-sampled motion as $20/2^3$ and $20/2^2$ at levels 3 and 2, respectively. We used a block size of 4×4 and a search range of 15×15, which are determined by experiments on hyperparameters in Section 3.2.

We first trained BlockNet using the MPI Sintel dataset [21] with 600 epochs. We fine-tuned the network using the FlyingChairs dataset. The initial learning rate was 0.0001. It was halved at iterations 0.2 M, 0.25 M, 0.3 M, and 0.35 M. We used a mini-batch size of 4 and the Adam optimizer [22]. The weights were set to $\alpha_4 = 0.32$, $\alpha_3 = 0.08$, and $\alpha_2 = 0.02$, and the regularization parameter γ was set to 0.0004 as in [15]. BlockNet was implemented using TensorFlow 1.7.0.

3.2. Results

To verify the effectiveness of the proposed deep neural architecture, BlockNet was compared to a conventional block motion estimation (BME) that exploits all pixels of each candidate block in the search range for matching. We also compared each algorithm with or without the proposed representative matching (RM). All results are evaluated in terms of end-point error (EPE), with the L_2 norm between the estimated motion and ground truth [15].

The average and standard deviation of EPE are summarized in Table 1. Experimental results show that BlockNet with full matching had lower average EPE than BME with full matching. This is because the CNNs in BlockNet can extract rich features, and the matching errors of BlockNet were lower than those of BME. Moreover, average EPEs of BlockNet with full matching and proposed RM were similar, while average EPEs of BME with full matching and proposed RM significantly differed. This result implies that the proposed representative value, which reduced the computational complexity as much as 1/16 for each matching, was more effective in the feature domain than in the intensity domain. Figure 3 shows qualitative results of BlockNet with full matching and proposed RM. The results of BlockNet with full matching and proposed RM are quite similar (Figure 3; top, chair leg). However, proposed RM occasionally fails to estimate the detailed motion of an object compared to full matching (Figure 3; bottom, chair leg).

Table 1. Quantitative results on the FlyingChairs dataset.

	BME with Full Matching	BME with Proposed RM	BlockNet with Full Matching	BlockNet with Proposed RM
Average EPE	10.33	16.86	3.74	4.09
Std. of EPE	7.64	7.09	3.53	3.57

Detailed experiments were conducted to verify the effect of some hyper-parameters (block size, search range) in BlockNet with RM (Figure 4). Although a large block size reduced the computational complexity, the average EPE was increased because of the reduction in the resolution of the estimated motion. For the search range, the average EPE with a large value was slightly decreased at the expense of high computational complexity. The proposed RM was reduced by 18% compared to full matching,

using the best hyper-parameter (Figure 4, red diamond), with respect to computational complexity while archiving similar average EPE.

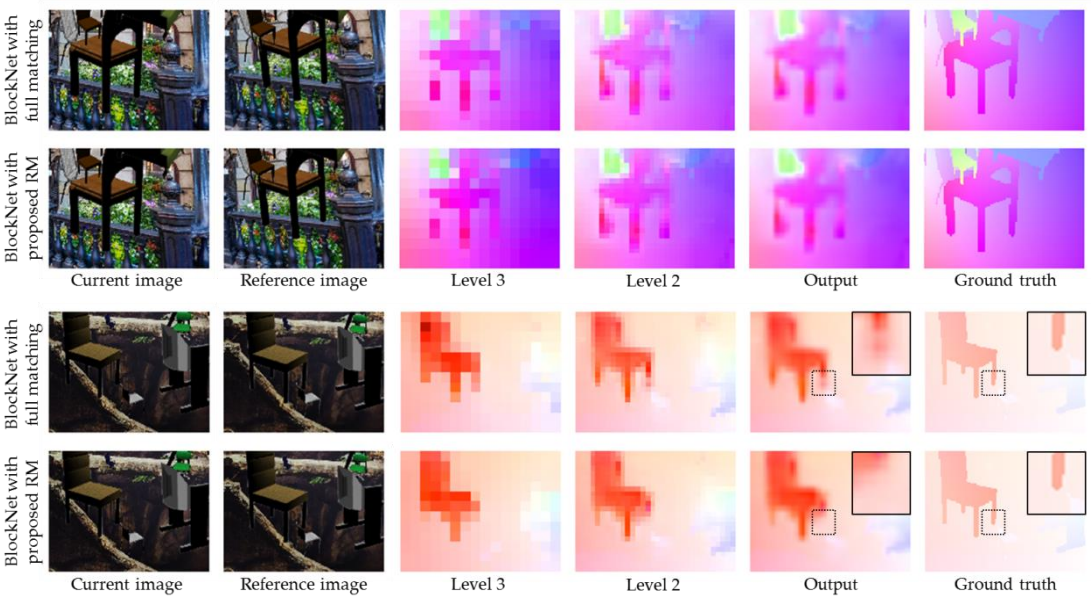

Figure 3. Visual results on the FlyingChairs dataset.

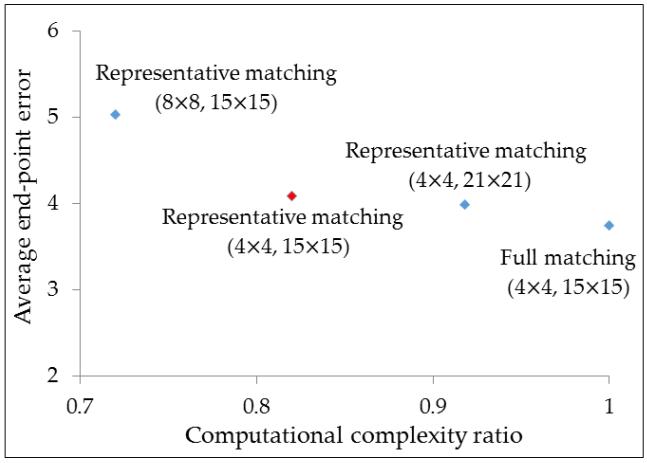

Figure 4. Average end-point error and computational complexity ratio on BlockNet with several methods using FlyingChairs dataset. Computational complexity ratio represents the value of all methods divided by that of full matching. The values in the parentheses represent the block sizes and search ranges, respectively.

4. Conclusions

In this paper, we proposed BlockNet using an efficient representative matching. The proposed network can extract rich features for block-based motion estimation. A representative matching was performed with these features by using the average operator and implemented simply by using the average-pooling operator, widely employed in the deep-learning framework. To maximize the efficiency of the proposed representative matching, a pyramidal structure with feature warping was adopted in BlockNet. Experimental results show that BlockNet with and without our representative matching achieved similar average EPE, while our matching exhibited lower computational cost than full matching. In future work, we will apply BlockNet to various real-time applications based on

motion estimation, such as frame rate up-conversion because it has less computational cost and is easy to implement.

Author Contributions: J.L. and K.K.: methodology, software, writing—original draft preparation, writing—review and editing; G.B.: project administration; W.-J.S.: supervision. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ICT Consilience Creative program (IITP-2019-2011-1-00783) supervised by the IITP (Institute for Information & communications Technology Planning & Evaluation), and LG Display under LGD-POSTECH Research Program.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Kuo, C.-M.; Hsieh, C.-H.; Jou, Y.-D.; Lin, H.-C.; Lu, P.-C. Motion estimation for video compression using Kalman filtering. *IEEE Trans. Broadcast.* **1996**, *42*, 110–116.
2. Su, Y.; Sun, M.-T.; Hsu, V. Global motion estimation from coarsely sampled motion vector field and the applications. *IEEE Trans. Circuits Syst. Video Technol.* **2005**, *15*, 232–242.
3. Kong, K.; Shin, S.; Lee, J.; Song, W.-J. How to estimate global motion non-Iteratively from a coarsely sampled motion vector field. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *29*, 3729–3742. [CrossRef]
4. Kang, S.-J.; Yoo, S.; Kim, Y.H. Dual motion estimation for frame rate up-conversion. *IEEE Trans. Circuits Syst. Video Technol.* **2010**, *20*, 1909–1914. [CrossRef]
5. Yoo, D.-G.; Kang, S.-J.; Kim, Y.H. Direction-select motion estimation for motion-compensated frame rate up-conversion. *J. Disp. Technol.* **2013**, *9*, 840–850.
6. Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 834–848. [CrossRef] [PubMed]
7. Kong, K.; Lee, J.; Song, W.-J.; Kang, M.; Kwon, K.J.; Kim, S.G. Multitask bilateral learning for real-time image enhancement. *J. Soc. Inf. Disp.* **2019**, *27*, 630–645. [CrossRef]
8. Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; Fei-Fei, L. Large-scale video classification with convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 24–27 June 2014; pp. 1725–1732.
9. Kong, K.; Lee, J.; Kwak, Y.; Kang, M.; Kim, S.G.; Song, W.-J. Recycling: Semi-supervised learning with noisy labels in deep neural networks. *IEEE Access* **2019**, *7*, 66998–67005. [CrossRef]
10. Bouwmans, T.; Javed, S.; Sultana, M.; Jung, S.K. Deep neural network concepts for background subtraction: A systematic review and comparative evaluation. *Neural Netw.* **2019**, *117*, 8–66. [CrossRef] [PubMed]
11. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
12. Ciaparrone, G.; Sánchez, F.L.; Tabik, S.; Troiano, L.; Tagliaferri, R.; Herrera, F. Deep learning in video multi-object tracking: A survey. *Neurocomputing* **2020**, *381*, 61–88. [CrossRef]
13. Dosovitskiy, A.; Fischer, P.; Ilg, E.; Hausser, P.; Hazirbas, C.; Golkov, V.; Van Der Smagt, P.; Cremers, D.; Brox, T. Flownet: Learning optical flow with convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 2758–2766.
14. Ilg, E.; Mayer, N.; Saikia, T.; Keuper, M.; Dosovitskiy, A.; Brox, T. Flownet 2.0: Evolution of optical flow estimation with deep networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2462–2470.
15. Sun, D.; Yang, X.; Liu, M.Y.; Kautz, J. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 8934–8943.
16. Hui, T.W.; Tang, X.; Change Loy, C. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 8981–8989.

17. Metkar, S.; Talbar, S. *Motion Estimation Techniques for Digital Video Coding*; Springer: Cham, Switzerland, 2013; p. 13.
18. Sun, X.; Jin, G.; Huang, M.; Xu, G. A novel partial block-matching motion estimation algorithm. In Proceedings of the Third International Symposium on Multispectral Image Processing and Pattern Recognition, Beijing, China, 20–22 October 2003; pp. 839–842.
19. Liu, B.; Zaccarin, A. New fast algorithms for the estimation of block motion vectors. *IEEE Trans. Circuits Syst. Video Technol.* **1993**, *3*, 148–157. [CrossRef]
20. Chan, Y.-L.; Siu, W.-C. New adaptive pixel decimation for block motion vector estimation. *IEEE Trans. Circuits Syst. Video Technol.* **1996**, *6*, 113–118. [CrossRef]
21. Butler, D.J.; Wulff, J.; Stanley, G.B.; Black, M.J. A naturalistic open source movie for optical flow evaluation. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 611–625.
22. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.