DDCNet-Multires: Effective Receptive Field Guided Multiresolution CNN for Dense Prediction

Ali Salehi, Madhusudhanan Balasubramanian
Tennessee, United States

Abstract

Dense optical flow estimation is challenging when there are large displacements in a scene with heterogeneous motion dynamics, occlusion, and scene homogeneity. Traditional approaches to handle these challenges include hierarchical and multiresolution processing methods. Learning-based optical flow methods typically use a multiresolution approach with image warping when a broad range of flow velocities and heterogeneous motion is present. Accuracy of such coarse-to-fine methods is affected by the ghosting artifacts when images are warped across multiple resolutions and by the vanishing problem in smaller scene extents with higher motion contrast. Previously, we devised strategies for building compact dense prediction networks guided by the effective receptive field (ERF) characteristics of the network (DDCNet). The DDCNet design was intentionally simple and compact allowing it to be used as a building block for designing more complex yet compact networks. In this work, we extend the DDCNet strategies to handle heterogeneous motion dynamics by cascading DDCNet based sub-nets with decreasing extents of their ERF. Our DDCNet with multiresolution capability (DDCNet-Multires) is compact without any specialized network layers. We evaluate the performance of the DDCNet-Multires network using standard optical flow benchmark datasets. Our experiments demonstrate that DDCNet-Multires improves over the DDCNet-B0 and -B1 and provides optical flow estimates with accuracy comparable to similar lightweight learning-based methods.

Keywords: Dense prediction, optical flow estimation, dilated convolution, multi-resolution analysis, compact network, gridding artifact

Preprint submitted
1. Introduction

Having a fast and accurate optical flow estimation method is an essential step in many computer vision applications. Decades of research resulted in many successful dense pixel matching methods. Nevertheless, presence of certain conditions such as large displacements (e.g., hundreds of pixels), a high percentage of occlusion, and large homogeneous regions are still challenging.

A powerful machine learning model with a good degree of generalization and easy to fine-tune could be a promising solution for many applications. With learning-based models, many of the assumptions generally employed for modeling and estimating motion from image sequences can be avoided by allowing the models to learn generic transformation of frames in an image sequence. Deep learning methods have been shown to be successful for dense prediction tasks including dense flow estimation [18, 17, 19, 28, 9, 7, 14].

In many computer vision applications, estimating optical flow is an essential intermediate step. Therefore, compactness, speed, and accuracy of the optical flow estimation algorithms are necessary. Inspired by classical approaches, some deep learning-based flow estimation models have designed custom layers such as explicit feature mapping [6, 25, 11]. Those layers proved to be effective in flow estimation. Using standard non-specialized layers in the network, however, can facilitate easier integration of the network within the processing pipeline of other computer vision applications such as activity recognition.

Estimating large displacements (in the order of hundreds of pixels) in the presence of heterogeneous motion dynamics is one of the main challenges that are yet to be addressed by modern flow estimation algorithms. Multi-resolution approaches have been extensively used in classical and modern methods as a strategy to estimate large flow vectors [11, 4, 22]. Many of these methods use spatial pyramid approach to achieve this. In general, a coarser-level optical flow estimated at a given Gaussian pyramid level of frames in an image sequence is used to first warp either a target or reference frame to the next finer level in the pyramidal decomposition. A finer resolution of optical flow estimate is estimated at the next pyramidal level using the warped image and the corresponding reference or target frames at the respective pyramid level. This coarse-to-fine level of optical flow estimation process is repeated until the desired or full resolution of optical flow estimates are achieved. Other hierarchical approaches to estimate large motions in the presence of heterogeneous motion dynamics include using various levels of global flow.
estimates in a scene or regions within a scene to hierarchically propagate a more accurate initial optical flow estimate to each pixel and thereby allowing the method to converge to the ground truth optical flow quickly. In this approach, only the flow information is scaled at multiple levels while retaining the original frame resolution [3].

There are issues associated with methods that use spatial pyramid to deal with large displacements. Inaccurate flow estimates at a coarse level, for example, due to scene occlusion or homogeneity, may propagate to the finer levels. Another important issue with those coarse-to-fine methods is the introduction of ghosting artifact during frame warping [15]. For example, a foreground object (occluding object) moving over a stationary background will introduce a ghost copy of the occluding object after warping. Though warping images or feature maps can guide the network to achieve finer-scale estimations and reduce the search range during feature matching, the region with ghosting artifacts will lead to estimation errors due to multiple matching candidate locations. In addition, smaller objects moving at a higher velocity likely are often ignored (vanished) due to scale mismatch in a coarse-to-fine strategy [22]. Coarse-to-fine strategies, however, are useful for large motion estimation with magnitudes higher than the extent of the receptive field of deeper neural networks.

Previously, we developed a lightweight deep dilated CNN architecture (DDCNet) and strategies for dense prediction problems [24]. In brief, DDCNet-B0 and -B1 were designed based on effective use of dilated convolutional layers for dense prediction tasks. Effective Receptive Field (ERF) of candidate networks was used as guiding principle to design compact networks for dense optical flow estimation. Two key strategies or design recommendations for building DDCNets were: 1) preserving spatial information throughout the network with minimal feature downsampling and 2) designing networks with large enough receptive fields or effective receptive fields. While techniques such as use of deconvolution layers could be used to recover spatial resolution in a deeper network, preserving resolution of spatial features within the network is essential for dense prediction tasks. Because the primary task of dense prediction problems such as optical flow estimation is to estimate pixel-level coordinate transformation of a scene, every output unit in the network should have access to large enough spatiotemporal extent of the input image sequences.

In this work, we present a multi-resolution architecture called DDCNet-Multires that neither utilizes image-pyramidal architecture nor any interme-
rate warping to achieve multi-resolution property. DDCNet-Multires extends the DDCNet design strategies to handle heterogeneous motion dynamics by cascading sub-nets with decreasing extents of their ERF. With this additional design strategy, coarse optical flow estimates from a sub-net with a larger ERF are refined using one or more sub-nets with narrower ERF. We demonstrate the performance of DDCNet-Multires model using standard optical flow benchmark datasets.

2. Related Work

SpyNet [22], PWC-Net [25] and all three variants of LiteFlowNet [11, 12, 10] take advantage of classical coarse-to-fine approach in their designs. SpyNet builds image pyramid and warps the images using estimated flow in the coarser levels. PWC-Net and LiteFlownets build pyramids from feature maps and do warping on feature maps instead of images.

FlowNet-Simple and FlowNet-Correlation also achieve some kind of built-in multi-resolution by estimating flow from low-resolution features maps and repeating the process in all consecutive layers of its refinement section [6]. FlowNet2 stacks several versions of FlowNets but all its networks work on full-resolution data. It takes advantage of the warping layer to help the higher-level networks to deal with shorter displacements [11, 10]. Our method differs from FlowNet-Simple: we estimate flow after each sub-net comprised of about 15 layers and each sub-net takes full image features as input. Unlike FlowNet2 we do not use any warping layers.

LiteFlowNet2 and LiteFlowNet3 are designed to address some of the issues associated with propagation of wrong estimated form coarse levels. LiteFlowNet2 increases the speed of the original model along with improvements in the quality of the flow estimates [12]. LiteFlowNet3 incorporates two additional modifications on LiteFlowNet2 to deal with propagation of wrong flow estimates from coarser levels when there is occlusion or homogeneous regions in a scene [10].

Effects of the vanishing problem can be reduced by building the pyramid using dense features instead of raw images such as the methods used in PWC-Net and LiteFlowNet [25, 10]. The vanishing problem cannot be alleviated even by feature warping technique used in LiteFlowNet [11]. Therefore, a more complicated modification to the cost function was necessary for LiteFlowNet3 to deal with propagation of errors from coarse level to fine levels due to the ghosting problem [10]. DDCNet-Multires integrates DDCNet and
multi-resolution strategies in such a way to avoid vanishing and ghosting artifact problems.

DDCNet-B0 shows the efficacy of our compact dense-prediction design strategy [24]. Our network architecture is simple while performing better than similar networks such as FlowNet-Simple in terms of speed and accuracy. DDCNet-B1 demonstrated improved effectiveness of the network by dropping odd layers (layers with odd dilation rates) and reducing the resolution of the feature maps. The central part of both of these networks is a sub-net called flow feature extractor that consists of several convolution layers with increasing dilation rates (see Figure 1). In this work, we incorporate classical multiresolution strategies along with the DDCNet design strategies to further improve the accuracy of dense flow estimation.

3. DDCNet-Multires

DDCNet-B0 and DDCNet-B1 networks were designed based on the unique characteristics of dense estimation tasks such as dense flow estimation. Their design is intentionally simple and compact which makes them ideally suited for serving as building blocks for many other applications as well as for building more complex architectures using simpler building blocks. To improve accuracy, we utilize the sub-modules within these basic networks and build more elaborate networks yet within the class of lightweight networks with few trainable parameters and lower processing time.

Figure 1 shows all important building elements, namely spatial feature extractor and flow feature extractor that were used for building a multiresolution network called DDCNet-Multires. DDCNet-Multires has DDCNet-B1 [24] as foundation and has all the necessary layers to learn optical flow estimation. In fact, by training this simple network in Figure 1 we were able to attain more comparable optical flow estimates with other complicated methods in literature such as FlowNet-Simple.

Spatial feature extractor sub-net is comprised of three (or more) convolution layers with dilatation rates of 1, 2, and 3 respectively. It takes the first and second frames of a sequence separately and after applying these shared convolutions on them, concatenates the feature maps from each image and passes them to subsequent layers in the network. Flow feature extractor is a sub-net made of consecutive convolution layers with increasing dilation rates. When dilation rates of this sub-net are constant or only increasing in steps of one, we call that flow feature refiner (Figure 1). Systematic and strategic
Figure 1: Main components of DDCNet architecture: spatial feature extractor) three (or more) convolution layers with dilatation rates of 1, 2, and 3 respectively. flow feature extractor) is made of consecutive convolution layers with increasing dilatation rates. When dilatation rates of this sub-net is constant or only increasing in steps of one, we call this flow feature refiner.
arrangement of flow feature extractor and flow feature refiner can be used to build networks with larger receptive fields with appropriate field shape and field smoothness.

3.1. Multi-resolution

We utilize our flow feature extractor module as a building block for an improved multi-resolution flow estimation network. By adjusting the dilation rates or by passing a downsampled inputs to this module, the ERF of this module can be adjusted to generate intermediate flow estimates at the desired scale or resolution. For example, using a flow feature extractor module with a larger ERF, coarse and large flows in the scene can be estimated. Therefore, by cascading several flow feature extractor modules with decreasing extents of ERFs, initial coarse and large motion estimates can be refined successively to generate finer flow estimates for all flow magnitudes.

Following the original DDCNet design strategies [24], we make sure that the ERF of the DDCNet-Multires architecture covers the majority of large flow vectors i.e. large enough ERF extent, and has suitable ERF shape and smoothness (see ERF of the network in Figure 9). Figure 2 shows a multi-resolution network architecture built using one spatial feature extractor module, one flow feature extractor module, and two cascaded flow feature refiner modules. For preprocessing and illumination invariant spatial feature extraction, raw image sequences are fed to a spatial feature extractor module resulting in 64 feature maps for each of the images in the sequence. Concatenated spatial feature maps are then passed to a flow feature extractor module whose detailed architecture is shown in Figure 1. Two convolution filters were applied to the 128 feature maps at 1/4th resolution from the flow feature extractor module to generate coarse flow estimates at 1/4th resolution.

Coarse flow estimates at 1/4th resolution from the flow feature extractor were upsampled to full resolution and concatenated with the original spatial feature maps. The concatenated coarse flow estimates and spatial feature maps were fed to a flow feature refiner module. Again, two convolutional filters were applied to the 128 feature maps at 1/2 resolution from the first flow feature refiner module to generate finer flow estimates at 1/2 resolution.

Upsampled coarse flow estimates from the flow feature extractor module, upsampled finer flow estimates from the first flow feature refiner module and the original spatial feature maps were concatenated and fed to a second flow feature refiner module comprised of 15 convolutions with no dilation.
The final optical flow estimates at the original resolution were obtained by applying one layer of convolution on the feature maps from the second flow feature refiner module.

One important distinction with our multi-resolution architecture is that neither do we utilize image-pyramidal architecture nor any intermediate warping to achieve multi-resolution property. Despite naming the different parts of the network as separate modules, they are all sub-nets of a single network. Therefore, these sub-nets could be more efficiently trained end-to-end using the raw image sequences and the corresponding ground-truth optical flow for each of the training sequences.

4. Experiments

4.1. Datasets

*Flying Chairs* dataset was used both for design purposes (selecting more promising networks for further tests) and initializing all other training for faster convergence. This dataset contains 22,872 image pairs with ground truth flow. Sequences were generated by applying random 2D affine transformations of a set of renderings of 3D chair models put on Flicker images from different categories as background [6].

*Flying Things3D* is a synthetic stereo flow benchmark dataset with approximately 77 thousand sequences. It provides ground truth stereo disparity and optical flow maps. While the Flying Chairs dataset contains only planar motion, Flying Things3D benchmark sequences have 3D motions and lighting effects and were designed to be more realistic. In addition to linear motions, 3D objects in the scene also undergo non-rigid transformations. Due to large object movements within the scene, optical flow magnitudes, as well as the extent of occlusions, are also larger. Therefore, Flying Things3D dataset, with more diverse scene characteristics and motion dynamics (Figure 3b), is a primary dataset commonly used for training supervised networks for dense flow estimation [20].

*MPI Sintel Dataset* is among the most challenging dataset with higher flow magnitudes, motion heterogeneity, and large occlusions. Two different versions of this dataset namely *clean* and *final* are used in this study. Clean sequences describe specific illumination conditions for the scene. The final sequences include atmospheric effects such as fog along with motion blur. For each dataset version (clean and final), 1,041 sequences with ground truth are available [5].
Figure 2: DDCNet-Multires: A multi-resolution network architecture built using one spatial feature extractor module, one flow feature extractor module, and two cascaded flow feature refiner modules.
KITTI2012 and KITTI2015 are real-word sparse optical flow datasets. Because the samples are taken from a driver’s view, these are limited to only a specific type of motion known as ego-motion that captures a rigid scene as the camera moves within the scene. In total, approximately 400 sequences with larger displacements and a larger variety of lighting conditions are present in these datasets. Ground truth occlusion maps are also available [8, 21].

Middlebury includes real scenes with fluorescent tagged textures used for generating ground truth optical flow and realistic computer-generated synthetic scenes. It contains only 8 two-frame sequences with small displacements, around 10 pixels [2].

![Histograms of flow vectors in Sintel and Flying Things3D datasets with the u and v components of the flow velocities along the horizontal and vertical axes respectively. Frequency of observing u, v is in log scale and color coded for clarity. Each location in the histogram represents the frequency of a specific flow velocity u, v in the respective dataset with the zero velocity ((u, v) = [0, 0]) at the center.](image)

4.2. Training and Network Details

Sub-nets within DDCNet-Multires can be individually trained when there are memory or GPU limitations or the whole DDCNet-Multires with cascaded sub-nets can be trained end-to-end depending on the available GPU capacity. Initial training for DDCNet-Multires started with the Flying Chair dataset as it is simple in terms of scene dynamics and magnitude of motions. In general, a suitable initial learning rate was identified as the parameter choice resulting in a downward trend in the network error. Then, we continued training on the Flying Things 3D dataset that has larger motion vectors (see a histogram of its motion vectors in Figure 3). This is still a synthetic dataset but with more samples, diverse objects, textures, and lighting con-
This dataset was designed to have image and scene statistics similar to other benchmark datasets as much as possible.

Train-time-augmentation method was used, following the work of [6], to improve accuracy and generalization of the network. Geometric augmentations and photometric augmentations were applied to each batch. For geometric augmentations, all images within each training sequence and the corresponding ground truth optical flow were rotated randomly between -30 and 30 degrees and resized with a random scaling factor uniformly chosen between 0.5 and 1.0. For photometric augmentations, Gaussian noise with zero mean and a standard deviation chosen uniformly from \([0.01, 0.08]\) was added to only images in each sequence. A multiplier for adjusting contrast was chosen randomly from \([0, 1.5]\). Images were converted to HSV and their saturation channels were multiplied by a random saturation factor chosen from \([0, 1]\) then converted back to RGB. To modify brightness, a random number, selected from \([-0.3, 0.3]\), was added to all channels of images. After all these perturbations, pixel values were scaled to be in range \([0, 1]\).

The following network training parameters were heuristically obtained. A batch size of 4 was used for training. Learning rates were set to 0.0001 for the first 900k iterations and divided by 2 after the network loss stays flat for a few epochs (about 60k iterations). As we mentioned in Figure 1, all the filters were \(3 \times 3\) in size and were initialized by the He-initialization method. ReLU activation was used for most of the layers except those that generate flow maps. For the layers that generate flow maps, linear activation was used to generate any real number (positive or negative values).

4.3. Results

4.3.1. Quantitative and Qualitative Evaluation on Benchmark Datasets

Average endpoint error (AEE) metric was used for assessing network performance during our network design and development. We also used \(Fl_{all}\) metric on KITTI datasets, where \(Fl_{all}\) measures the percentage of outlying estimates based on the number of pixel coordinates with an endpoint error (EE) \(\geq 3\) pixels of flow and EE \(\geq 5\%\) of the magnitude of its ground-truth flow vector were counted as outliers).

Quantitative performance of our DDCNet models and other state-of-the-art lightweight and heavyweight deep learning models for the Sintel, KITTI12, KITTI15, and Middlebury datasets are presented in Table 1. For datasets with large displacements such as Sintel, DDCNet-Multires outperformed DDCNet-B0 and DDCNet-B1. This can be observed from sequences...
Table 1: Average endpoint error of selected deep learning-based methods on benchmark datasets. Entries with parentheses indicate the testing performance on data that were previous used for fine-tuning the network. $F_{all}$ measures the percentage of outlier estimates; pixels with $EE \geq 3$ and $EE \geq 5\%$ of the magnitude of its ground-truth flow vector were counted as outliers.

| Method                  | Sintel clean | Sintel final | KITTI12 | KITTI15 | Middlebury |
|-------------------------|--------------|--------------|---------|---------|------------|
|                         | train | test | train | test | train | test | (Fl-all) | (Fl-all) | train | test |
| **Heavyweight CNN**     |        |     |       |       |       |       |         |         |       |     |
| FlowNetS [6]            | 4.50  | 7.42 | 5.45  | 8.43 | 6.20  | 1.09  |
| FlowNetS ft-sintel [6]  | (3.66) | 6.96 | (4.44) | 7.76 | 7.52  | 9.1   | 0.98    | 1.15    |        |     |
| FlowNetC [6]            | 4.31  | 7.28 | 5.87  | 8.81 | 9.35  | 1.15  |
| FlowNetC ft-sintel [6]  | (3.78) | 6.85 | (5.28) | 8.51 | 8.79  | 1.15  | 0.93    | 1.15    |        |     |
| FlowNet2 [13]          | 2.02  | 3.96 | 3.54  | 6.62 | 4.01  | 10.08 | 29.99%  | 3.54    | 9.94  | 28.02%|
| FlowNet2 ft-sintel [13] | (1.45) | 4.16 | (2.19) | 5.74 | 5.54  | 9.42  | 28.02%  | 3.54    | 9.94  | 28.02%|
| FlowNet2 ft-kitti [13]  | 3.43  | 4.83 | (1.43) | 1.8  | (2.36) | (8.88%) | 11.48%  | 0.56    |       |     |
| **Lightweight CNN**     |        |     |       |       |       |       |         |         |       |     |
| SPyNet [22]            | 4.12  | 6.69 | 5.57  | 8.43 | 9.12  | 0.33  | 0.58    | 0.33    | 0.58   |     |
| SPyNet ft-sintel [22]   | (3.17) | 6.64 | (4.32) | 8.36 | 3.36  | 4.1   | 35.07%  | 3.36    | 4.1   | 35.07%|
| PWC-Net+ ft-sintel [20] | (1.71) | 3.45 | (2.34) | 4.60 | (0.99) | 1.4   | (1.47) | (7.59%) | (7.72%)|     |
| LiteFlowNet [11]       | 2.48  | 4.04 | 4.00  | 6.09 | 1.29  | 1.7   | 10.39%  | 1.29    | 1.7   | 10.39%|
| LiteFlowNet ft-sintel [11] | (1.64) | 4.86 | (2.33) | 6.09 | (1.29) | 1.7   | 10.39%  | 1.29    | 1.7   | 10.39%|
| LiteFlowNet ft-kitti [11] | (1.30) | 3.48 | (1.62) | 4.69 | (0.95) | 1.4   | (1.33) | (4.32%) | (7.62%)|     |
| LiteFlowNet2 ft-sintel [12] | (1.32) | 2.99 | (1.76) | 4.45 | (0.91) | 1.3   | (1.26) | (8.82%) | (7.34%)|     |
| LiteFlowNet2 ft-kitti [12] | (0.99) | 1.46 | (1.06) | 2.46 | (0.91) | 1.3   | (1.26) | (8.82%) | (7.34%)|     |
| LiteFlowNet3 ft-sintel [10] | (1.32) | 2.99 | (1.76) | 4.45 | (0.91) | 1.3   | (1.26) | (8.82%) | (7.34%)|     |
| LiteFlowNet3 ft-kitti [10] | (0.99) | 1.46 | (1.06) | 2.46 | (0.91) | 1.3   | (1.26) | (8.82%) | (7.34%)|     |
| DDCNet-B0 ft-sintel [22] | (1.71) | 2.20 | (1.41) | 2.46 | (2.17)| 1.35  | 15.29%  | 47.78%  | 0.69   |     |
| DDCNet-B1               | 4.12  | 5.46 | 5.48  | 8.57 | 16.43 | 59.03%| 1.2     | 1.2     |     |     |
| DDCNet-B1 ft-sintel [22] | (1.96) | 6.19 | (2.25) | 6.91 | 6.65  | 11.22 | 52.66%  | 1.14    | 1.14   |     |
| DDCNet-B1 ft-kitti [13] | 0.65  | 0.38 | (1.76) | 4.2 | (2.57) | (15.56)% | 38.23%  | 1.74    | 1.74   |     |
| DDCNet-Multires         | 2.71  | 4.14 | 4.95  | 7.54 | 13.54 | 43.12%| 0.49    | 0.49    |     |     |
| DDCNet-Multires ft-sintel [13] | (1.36) | 5.34 | (1.70) | 5.86 | 5.41  | 12.59 | 40.30%  | 0.58    | 0.58   |     |
| DDCNet-Multires ft-kitti [10] | 0.86  | 7.54 | (0.92) | 3.2 | (1.33) | (5.58%) | 24.66%  | 0.72    | 0.72   |     |

with large displacements in them. For example, B0 fails in regions with motions larger than 250 pixels in the ‘large motion’ sequence (second row) in Figure 7 and in Figure 8 whereas B1 performs better and Multires has the best performance. But for Middlebury with small to moderate flow velocities, B0 performed better than B1. The endpoint error of DDCNet-Multires is less than B0 even on the Middlebury dataset demonstrating the effectiveness of the flow refiner modules.

One of the principal applications of the DDCNet sub-nets is in its use as a building block for designing lightweight, more efficient, and more effective optical flow estimation networks such as the DDCNet-Multires. FlowNet-Simple has been used as a building block for numerous optical flow estimation methods [13, 16, 23, 27]. LiteFlowNet and its variants are among the best performing lightweight learning-based models [11, 10]. Therefore, we focus on comparing the performance of our DDCNet-Multires primarily with FlowNet and LiteFlowNet.
Figure 4 shows the ground truth optical flow and optical flow estimated by FlowNet-Simple, FlowNet2, LiteFlowNet3, and DDCNet-Multires on Sintel dataset. As summarized in Table 2, FlowNet-Simple has 6 times more parameters (38 million vs 5.54) than DDCNet-Multires. Despite having an additional variational-based refiner step on top of the main network, FlowNet-Simple has a higher endpoint error on almost all of the benchmark datasets. Superior quality of DDCNet-Multires optical flow estimates when compared to FlowNet-Simple can be observed in Figure 4.

FlowNet2 is modeled as a stack of several sub-networks. The endpoint error of FlowNet2 is better than our Multires model, but FlowNet2 has 25 times more learnable parameters than the Multires model. Further, FlowNet2 is more than 6 times slower during testing and it is expected to require a longer training duration since each network needs to be trained separately. We could likely improve the performance of DDCNets using a similar stacking strategy while retaining the strengths of our network modeling approach. On Kitti2015 dataset, while FlowNet2 has a lower Fl-all error metric compared to DDCNet-Multires, it fails when it comes to motion boundaries (it can be seen in the result of sample #09 in Figure 5).

Among lightweight methods, DDCNet-Multires outperforms SPyNet on challenging Sintel and Kitti datasets but falls behind PWC-Net. LiteFlowNet models perform better than our DDCNet-Multires on Sintel clean test data. Our model outperforms LiteFlowNet on more challenging Sintel final test dataset. LiteFlowNet2 and LiteFlowNet3 have complicated designs but outperform all other methods on Sintel and Kitti datasets.

### 4.3.2. Model Size / Compactness and Processing Time

Table 2 shows a summary of model parameters (number of layers, and number of learnable parameters) and computational speed of processing Sintel image sequences.

The number of trainable parameters for DDCNet-B0 and DDCNet-B1 was 1.03 and 2.99 million respectively. DDCNet-Multires has 5.54 million parameters which is comparable with the current compact / lightweight optical flow models. In contrast, more elaborate and heavyweight models such as FlowNet models have 38 million to 162 million trainable parameters.

All of the DDCNets were developed using Tensorflow whereas other lightweight models were mainly implemented using Caffe. FlowNet implementations are available in both Tensorflow and Caffe. In general, Caffe implementations are several times faster than Tensorflow models. Since our GPU has more RAM,
Figure 4: Performance of our DDCNet-Multires model, LiteFlowNet3 (lightweight), FlowNet-Simple (heavyweight) and FlowNet2 (heavyweight) models for estimating optical flow for selected Sintel image sequences.
Figure 5: Performance of our DDCNet-Multires model, LiteFlowNet3 (lightweight) and FlowNet2 (heavyweight) models for estimating optical flow for selected Kitti2015 image sequences.
Table 2: Number of trainable parameters and computational speed of processing Sintel image sequences for lightweight and heavyweight models. Runtime is measured using Sintel image sequences with a frame size of 1024 × 436 pixels. General speed differences between the computational frameworks such as Tensorflow and Caffe should be considered when comparing the run times of various networks.

| Method                  | Number of layers | Number of parameters (m) | Framework | GPU (NVIDIA) | Time (ms) | FPS |
|-------------------------|------------------|--------------------------|-----------|--------------|-----------|-----|
| DDCNet-B0               | 31               | 1.03                     | TF2       | Quadro RTX 8000 | 76   | 13  |
| DDCNet-B1               | 30               | 2.99                     | TF2       | Quadro RTX 8000 | 30   | 33  |
| DDCNet-Multires         | 52               | 5.54                     | TF2       | Quadro RTX 8000 | 88   | 11  |
| Possible Caffe          |                  |                          | Quadro RTX 8000 | 17   | 58  |
| FlowNet Simple          | 17               | 38                       | TF1       | Tesla K80     | 86   | 11  |
|                         |                  |                          | Caffe     | GTX 1080      | 18   | 55  |
| FlowNet Correlation     | 26               | 39.16                    | TF1       | Tesla K80     | 179  | 5   |
|                         |                  |                          | Caffe     | GTX 1080      | 32   | 31  |
| FlowNet2                | 115              | 162.49                   | TF1       | Tesla K80     | 692  | 1   |
|                         |                  |                          | Caffe     | GTX 1080      | 123  | 8   |
| LiteFlowNet             | 94               | 5.37                     | Caffe     | GTX 1080      | 88.53 | 12  |
| SPyNet                  | 35               | 1.2                      | Torch     | GTX 1080      | 129.83 | 8   |
| PWC-Net+                | 59               | 8.75                     | Caffe     | TITAN Xp      | 39.63 | 25  |

we limited the batch size to 1 to make test times comparable. We report the average time for running one thousand batches of image sequences. With 30 ms processing time in Tensorflow, DDCNet-B1 model is about 3 times faster than Tensorflow implementation of FlowNet-Simple on a less powerful GPU. B1 is also 3 times faster than LiteFlowNet implemented using Caffe and runs on GTX 1080. DDCNet-Multires has test time comparable to FlowNet-Simple and accuracy close to FlowNet2 while being several times faster than it.

Figure 6 shows the frame processing rate of DDCNet models as a function of frame sizes. All DDCNet models are very fast (processing time per frame < 10 ms) for frame size smaller than 200 × 200 pixels. B1 is the fastest model across all frame sizes. Though B0 has the least number of learnable parameters with full resolution feature maps within the network, the number of floating-point operations is much higher compared to the B1 model.

4.4. Ablation study

In order to understand the efficacy of the designed DDCNet-Multires, in this section, we will compare its results on several challenging test cases from Sintel dataset. Later we will investigate its receptive field and intermediate flow estimates to see effectiveness of its sub-nets.
Figure 6: Testing time of the DDCNet models as a function of sequence frame size. B1 is computationally more efficient than B0 while being more accurate. Tests are performed on a single GPU Quadro RTX 8000 with batch sizes of 1.

4.5. Visual Inspection of Flow Estimates by Different DDCNet Models

Figure 7 shows performance of each of the DDCNet models on a broad variety of six example sequences with varied scene composition and motion dynamics from the Sintel Training Clean datasets. The examples are namely 1) fine motion sequence (first row in Figure 7) involving finer motions of smaller objects in various directions; 2) large motion sequence (second row) involving higher flow velocities; 3) disparate motion sequence (third row) involving varied flow dynamics in various parts of the scene with varied flow velocities; 4) homogeneous texture sequence (fourth row) with motions involving scene segments with homogeneous texture; 5) high texture sequence (fifth row) with motions involving highly textured scene segments; and 6) high occlusion sequence (sixth row) with aperture problem, entire or partial object views entering the scene and / or leaving the scene.

Figure 8 shows the corresponding error maps for each of the example sequences. The error map represents the magnitude of the difference between the estimated and ground-truth flow vector at each pixel. Therefore, locations with larger estimation errors will appear brighter and locations with lower estimation errors will appear darker in the error maps.

For the fine motion sequence, the Multires model captured the most finer flow profiles and B0 ignored finer motion details and motion boundaries.
Figure 7: Optical flow estimates of DDCNet-B0, B1 vs Multires for qualitative assessment of the network performance. Corresponding error maps are shown in [8].
Figure 8: Optical flow estimation error maps of DDCNet-B0, B1 vs Multires for the optical flow estimates shown in\cite{7}. Brighter locations indicate locations with larger estimation error.
Difficulties of the B0 model in detecting larger motions and clear motion boundaries in the larger motion sequence (e.g. near the dragon’s legs in the scene) is likely due to the smaller extent of its ERF. With a broader ERF extent, and peaking ERF near each reference pixel, the Multires model was able to detect larger motions and with significantly clearer motion boundaries. In images sequences with multi-directional flows in the scene (evident from multiple colors in the color-coded optical flow) as in the disparate motion sequence, we observed that activations of neurons in the latest layers due to multi-directional flows could cancel each other leading to less accurate flow estimates. Multires was again the best performing model followed by B0 and B1. All DDCNet models performed well on homogeneous texture sequence with Multires leading the list. For the high texture sequence all of the methods have difficulties identifying fine-grained flow estimates and motion boundaries in the high-textured zones. Multires and B0 were able to estimate finer flow velocities compared to B1. For the most challenging high occlusion sequence, all DDCNet models faced difficulties in the occluded zones or zones with aperture problem. Multires again performed better than B0 and B1 models.

In summary, Multires outperforms B0 and B1 models in terms of estimation accuracy with a similar processing time as B0. And Multires is also more accurate in estimating large flows, finer flows, and enforcing clearer motion boundaries in the estimated flows.

4.5.1. Visual Inspection of Multi-resolution Estimates in DDCNet-Multires

Figure 9 shows ERFs of each of the multi-resolution segments of the DDCNet-Multires model (one coarse segment and two refiner segments). These ERF maps are obtained by taking derivatives of two intermediate and one final flow layer with respect to the input layer. For example, ERF in Figure 9a is obtained by considering coarse flow layer after flow feature extractor sub-net as output of the network (see Figure 2). For illustrating distinctions of the multi-resolution levels, smaller frame size sequences were used for generating these ERF plots.

ERF corresponding to the coarsest output layer is wider with a Gaussian shape but with gridding artifacts. Adding the first flow refiner module minimizes ERF gridding artifacts while retaining a larger ERF extent. The final flow refiner module introduces a sharp peak in the ERF at each of the reference pixel locations emphasizing nearby pixel locations during matching pursuits. This likely helps the network to generate sharper and more
accurate results along motion boundaries. Figure 10 illustrates intermediate optical flow estimates generated within the Multires model at two coarser levels (Figures 10 a, b) used to generate a final fine-resolution optical flow estimate (Figure 10 c). The first-level estimate is the coarsest and suffers from gridding artifacts as it is expected based on the ERF of this segment (Figure 10 a). Results after the first flow refiner module are smooth and most of the artifacts are resolved (Figure 10 b). The final flow refiner module does not improve the accuracy significantly, but it is further refining the estimate and removing even more noises especially along the motion boundaries (Figure 10 c).

It is possible to reconfigure the Multires model to generate coarser flow estimates with a faster processing time (suitable for real-time applications) or finer flow estimates with longer processing time. For example, finer flow estimates can be achieved using three levels of feature refiner modules. Such high-resolution finer flow estimates may be especially important when flow
Figure 10: Intermediate and final flow estimates of DDCNet-Multires: (a) and (b) are intermediate optical flow estimates of the network. (c) is the final estimate from the network.
estimation is part of a high-level prediction task such as activity recognition or video compression.

5. Software

The DDCNet-Multires model along with necessary instructions for running the software are available to the public in the following URL: https://github.com/alisaaalehi/DDCNet.

6. Conclusion

In this work, we have devised a compact deep dilated CNN for dense prediction problems using 1) network design strategies guided by the effective receptive field characteristics of the network and 2) a cascaded sub-net approach to achieve multiresolution capability for handling large heterogeneous motion. In the cascaded sub-net approach, each sub-net with a varying ERF extent and ERF characteristics provides an optical flow estimate. Sub-nets with decreasing or varying ERF extents and characteristics are interconnected to achieve a multiresolution capability without image or feature warping. Thus, DDCNet-Multires avoids the ghosting artifacts and minimizes the vanishing problem. Desired overall ERF of the network can be achieved by effective combination and arrangement of DDCNet sub-nets with varying ERF shape, extent and smoothness characteristics. Accuracy of our compact DDCNet-Multires network on standard optical flow benchmark datasets was better than FlowNet-Simple and comparable to LiteFlowNet. In conclusion, our three strategies or recommendations namely 1) preserving spatial information throughout the network, 2) optimal ERF characteristics, and 3) multiresolution through cascaded sub-nets with varying ERF characteristics are useful for building more compact networks for dense prediction problems using standard network elements.

Acknowledgement

The research was supported in part by financial support in the form of a Herff Fellowship from the Herff College of Engineering, The University of Memphis (UoM); tuition fees support from the Department of Electrical and Computer Engineering, UoM; a summer student fellowship from the Fight for Sight organization; and in part by the National Institutes of Health, National Eye Institute Grant EY020518.
References

[1] Padmanabhan Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2(3):283–310, 1989.

[2] Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International journal of computer vision*, 92(1):1–31, 2011.

[3] Madhusudhanan Balasubramanian. *A Computational Framework for the Structural Change Analysis of 3D Volumes of Microscopic Specimens*. PhD thesis, Louisiana State University, 2006.

[4] James R Bergen, Patrick Anandan, Keith J Hanna, and Rajesh Hingorani. Hierarchical model-based motion estimation. In *European conference on computer vision*, pages 237–252. Springer, 1992.

[5] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, October 2012.

[6] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.

[7] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.

[8] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[9] Shir Gur and Lior Wolf. Single image depth estimation trained via depth from defocus cues. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7683–7692, 2019.
[10] Tak-Wai Hui and Chen Change Loy. Liteflownet3: Resolving correspondence ambiguity for more accurate optical flow estimation. In European Conference on Computer Vision, pages 169–184. Springer, 2020.

[11] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8981–8989, 2018.

[12] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. A lightweight optical flow cnn-revisiting data fidelity and regularization. arXiv preprint arXiv:1903.07414, 2019.

[13] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In IEEE conference on computer vision and pattern recognition (CVPR), volume 2, page 6, 2017.

[14] Eddy Ilg, Tonmoy Saikia, Margret Keuper, and Thomas Brox. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In Proceedings of the European Conference on Computer Vision (ECCV), pages 614–630, 2018.

[15] Joel Janai, Fatma Guney, Anurag Ranjan, Michael Black, and Andreas Geiger. Unsupervised learning of multi-frame optical flow with occlusions. In Proceedings of the European Conference on Computer Vision (ECCV), pages 690–706, 2018.

[16] J Yu Jason, Adam W Harley, and Konstantinos G Derpanis. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In European Conference on Computer Vision, pages 3–10. Springer, 2016.

[17] Chenxi Liu, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan L Yuille, and Li Fei-Fei. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 82–92, 2019.

[18] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE
conference on computer vision and pattern recognition, pages 3431–3440, 2015.

[19] William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104*, 2016.

[20] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4048, 2016. This is DispNet. They have introduced several methods here.

[21] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3061–3070, 2015.

[22] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, page 2. IEEE, 2017.

[23] Zhe Ren, Junchi Yan, Bingbing Ni, Bin Liu, Xiaokang Yang, and Hongyuan Zha. Unsupervised deep learning for optical flow estimation. In *AAAI*, pages 1495–1501, 2017.

[24] Ali Salehi and Madhusudhanan Balasubramanian. DDCNet: Deep dilated convolutional neural network for dense prediction. Submitted, 2021.

[25] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018.

[26] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Models matter, so does training: An empirical study of cnns for optical flow estimation. *IEEE transactions on pattern analysis and machine intelligence*, 42(6):1408–1423, 2019.
[27] Yang Wang, Yi Yang, Zhenheng Yang, Liang Zhao, Peng Wang, and Wei Xu. Occlusion aware unsupervised learning of optical flow. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4884–4893, 2018. URL http://openaccess.thecvf.com/content_cvpr_2018/papers/Wang_Occlusion_Aware_Unsupervised_CVPR_2018_paper.pdf.

[28] Junbo Zhang, Yu Zheng, Junkai Sun, and Dekang Qi. Flow prediction in spatio-temporal networks based on multitask deep learning. IEEE Transactions on Knowledge and Data Engineering, 32(3):468–478, 2019.