InterpoNet, A brain inspired neural network for optical flow dense interpolation Supplementary

Shay Zweig[1] and Lior Wolf[2,3]

[1]The Gonda Multidisciplinary Brain Research Center, Bar Ilan University, Israel
[2]The Blavatnik School of Computer Science, Tel Aviv University, Israel
[3]Facebook AI Research
shayzweig@gmail.com, wolf@cs.tau.ac.il

A. Bi-directional Averaging

	FF	DF	CPM	DM
Ours bidi	5.470	6.142	5.851	5.971
Ours no-bidi	5.363	6.141	5.768	6.017
Epic bidi	6.225	6.837	6.521	6.261
Epic no-bidi	5.815	6.625	6.337	6.441

Table S.1: Comparison of the results of our method and EpicFlow for the Sintel validation set with and without applying bi-directional averaging to the input in evaluation time .

We found that the training process results declined when the average number of missing pixels in the training flow maps was too high. Some of the matching algorithms, in particular DeepMatching, did produce sparse maps like these. To tackle this problem, we calculate the flow map bi-directionally (From I to I' and from I' to I) using the matching algorithm. We invert the second flow map and average the two maps. This simple step solves the sparseness problem for all of the matching algorithms we used. This procedure added to the computation time of our method. However most matching algorithms already compute bi-directional maps for consistency check and false matches filtering purposes and so we did not need to apply them twice. Importantly, we found that the bi-directional averaging is critical mostly for training the network and specifically for DeepMatching outputs. Training on FlowFields non averaged maps, for instance, gives comparable results to training with the averaged maps. Interestingly, applying EpicFlow on the bi-directional average of the DeepMatching algorithm output also slightly improved their results (Table S.1). For consistency reasons, we choose to present in this paper the results gained using the bi-directional averaged maps for training and evaluation. However, for all matching algorithms using only the original, non averaged map, in evaluation time yields results similar to those presented (Table S.1). The analysis in this section was performed without the variational post processing for both our method and EpicFlow.

B. Choice of Training and Validation Sets

The validation sets for both KITTI2012 and KITTI2015 datasets were the last 20% of the pairs in each. For the Sintel dataset, due to the temporal dependencies within scenes which are a pitfall for over-fitting, we define 4 whole scenes including 167 image pairs as a validation set rather than a random sample. We use the same validation set in the pre-training and Sintel fine tuning phases.

C. Early Stopping

Early stopping served as our only regularization method. The number of steps before performing the stop was 5000,1000 and 400 for training on the flying chairs, Sintel and KITTI datasets respectively. We use 4 rounds of early stopping in which we divide the learning rate by two starting with a learning rate of 5×10^{-5} for the pre-training and 5×10^{-6} for the fine tuning. After 4 rounds, we choose the weights that yielded the best performance on the validation set throughout the training.

D. Quantitative comparison To EpicFlow

Table S.2 shows the results gained using our method compared to EpicFlow for both KITTI datasets. Our method surpassed EpicFlow in all measurements (excluding %Out for KITTI 2012 using CPM). To further investigate our performance compared to EpicFlow, we looked at the EPE over all noisy pixels (pixels with $EPE > 3$) and missing pixels from all the flow maps in the Sintel validation set. To make a fair comparison for this analysis, we performed our prediction without bi-directional averaging so the number of noisy and missing pixels in the input to our network and EpicFlow was identical. We found that our performance were better than EpicFlow's in both of these

Method	KITTI 2012		KITTI 2015	
	EPE	%Out-all	EPE	%Out-all
FF+Ours	**2.363**	**11.11**	**7.921**	**29.00**
FF+Epic	3.518	11.25	16.100	33.00
CPM+Ours	**2.271**	11.3	**6.92**	**26.04**
CPM+Epic	3.337	**11.16**	15.135	32.48
DF+Ours	**2.074**	**9.01**	**6.626**	**24.29**
DF+Epic	2.92	12.34	11.680	30.34
DM+Ours	**2.168**	**9.57**	**6.733**	**28.84**
DM+Epic	3.515	14.20	14.068	35.12

Table S.2: Comparison of our model to EpicFlow on the KITTI 2012 and KITTI 2015 validation sets. The %Out is the percentage of pixels with $EPE > 3$ pixels.

areas, but it was significantly better only for the missing pixels ($Mean \pm SEM$ difference between Epic EPE and Our EPE: 0.08 ± 0.1, 1.11 ± 0.42 pixels; paired t-test p=0.42, $p < 0.01$ for noisy and missing pixels respectively, n=167). This emphasize our superiority over EpicFlow, Especially in large missing regions, as was demonstrated in Figure 5 of the main text.

E. Supplemental Figures

The supplemental figures presented here show further examples on top of the ones presented in the figures in the main text. Figure S.1 shows the progression of the prediction process in the network as appears in the output of the different detour layers, similar to figure 3a in the main text. Notice here also how the network first performs a simple interpolation and then refines the predictions in the deeper layers. Figure S.2 presents the predictions of networks with and without the edges input, similar to Figure 4a in the main text. The progression of the predictions in the different layers in those network is presented in figure S.3. These two figures illustrate how the edges input function in the network - acting as a stopper for spread of activation. Notice how the bottom "simple interpolation" layers perform similarly in both networks. However, starting from layer 4, the refinement process is very different. The network that receives the edges as input utilizes them to act as motion boundaries. Finally, figures S.4, S.5 and S.6 shows additional examples to the ones presented in figure 5 in the main text, for the comparison between the performance of our method and EpicFlow on the Sintel, KITTI 2012 and KITTI 2015 validation sets.

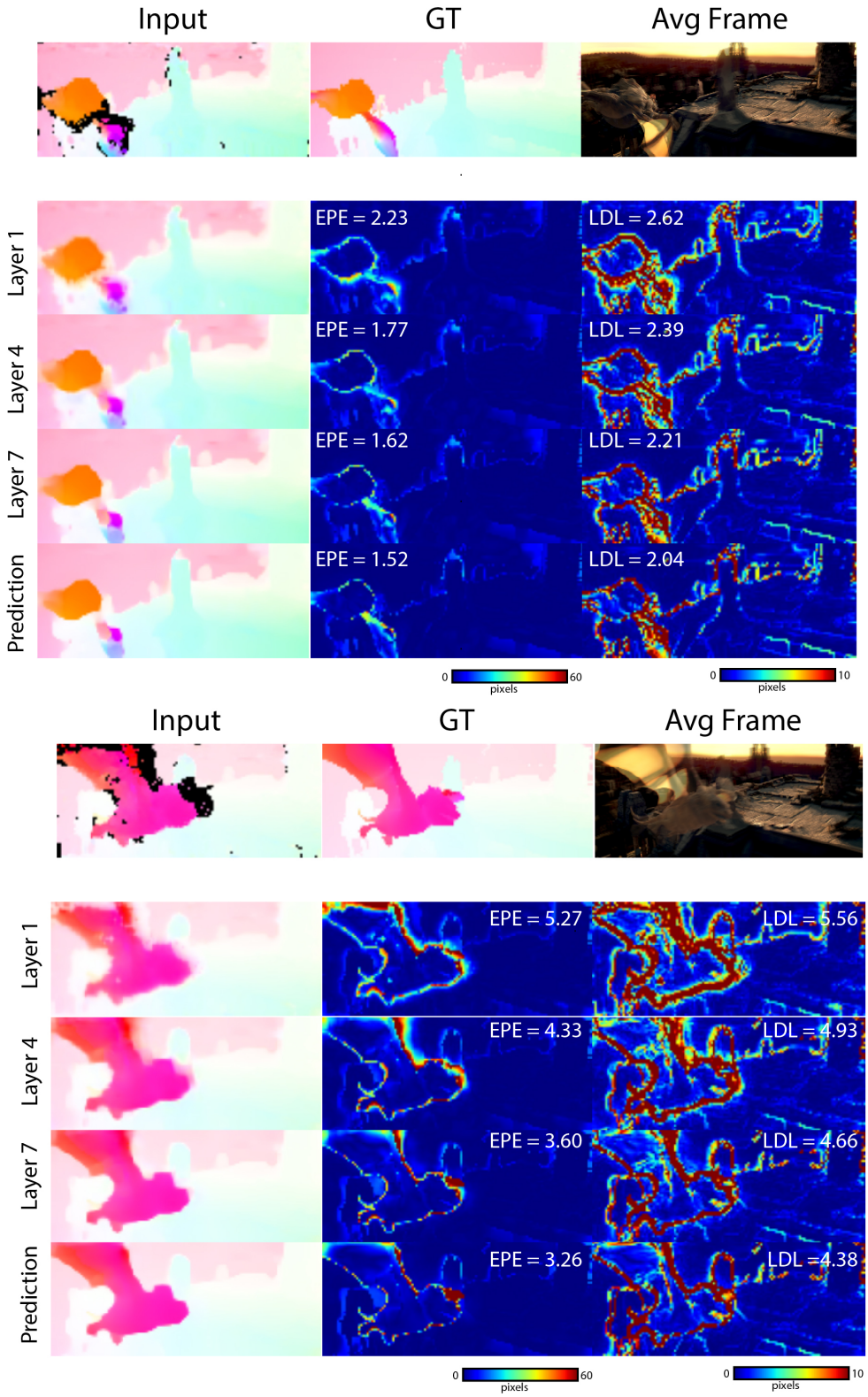

Figure S.1: Predictions in different layers – additional examples to figure 3a in the main text. The progression of the prediction process throughout the different layers in the network, as shown by the detour networks outputs. Starting from the second row, the second and third columns are the EPE and LD loss maps respectively.

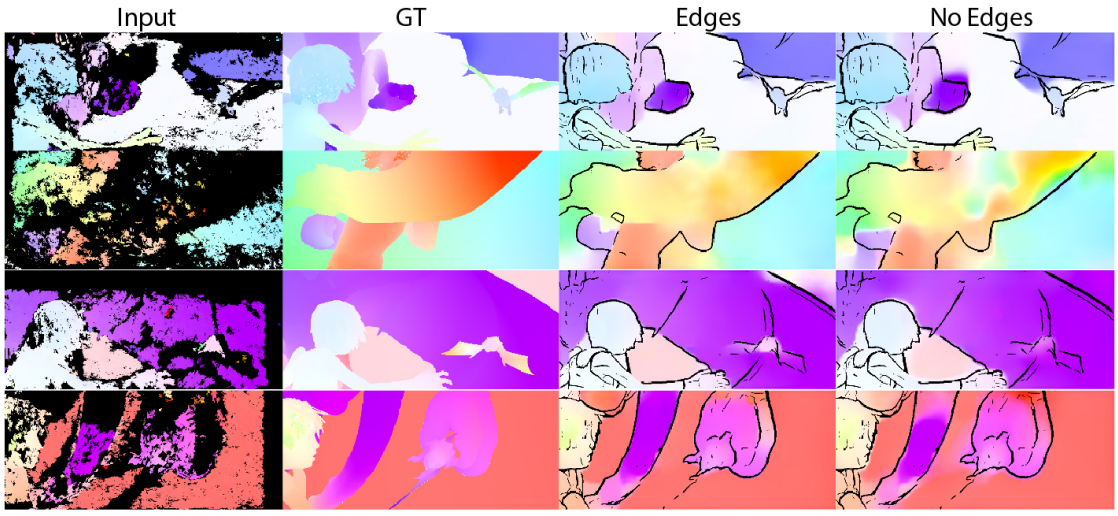

Figure S.2: The predictions of the networks with and without the edges input (additional examples to figure 4a in the main text). Edges are marked with black lines.

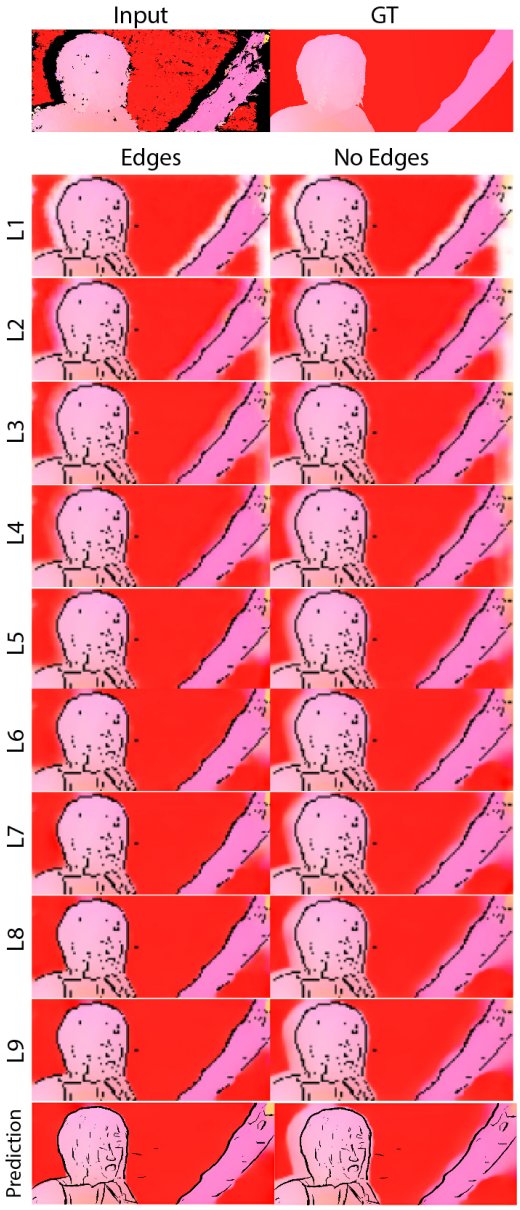

Figure S.3: The progression of the prediction process throughout the different layers in the network, as shown by the detour networks outputs for networks with and without the edges input, notice the similarity in the bottom layers and then the divergence starting from layer 4.

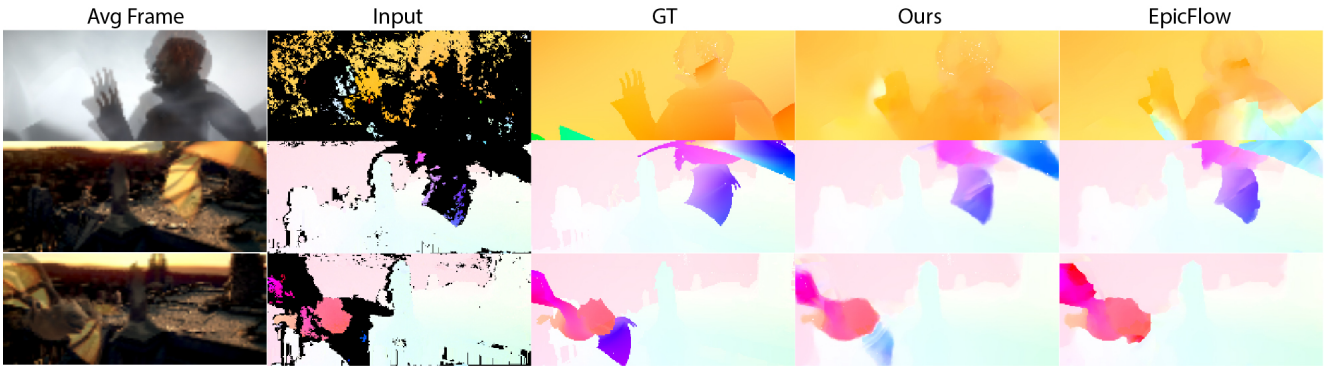

Figure S.4: A comparison of the predictions of our network to EpicFlow on examples from the Sintel validation set. (additional examples to figure 5 in the main text).

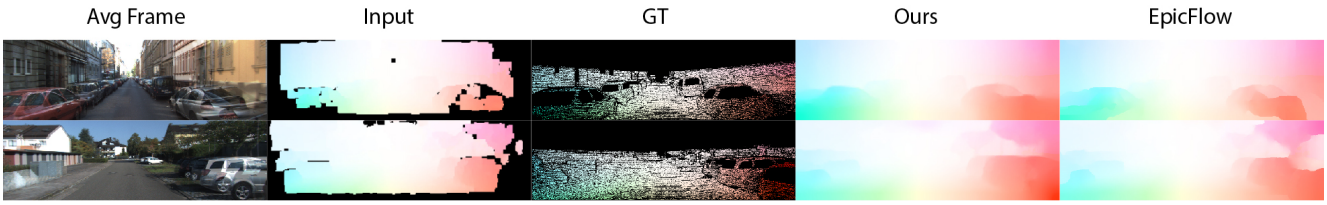

Figure S.5: A comparison of the predictions of our network to EpicFlow on examples from the KITTI 2012 validation set.

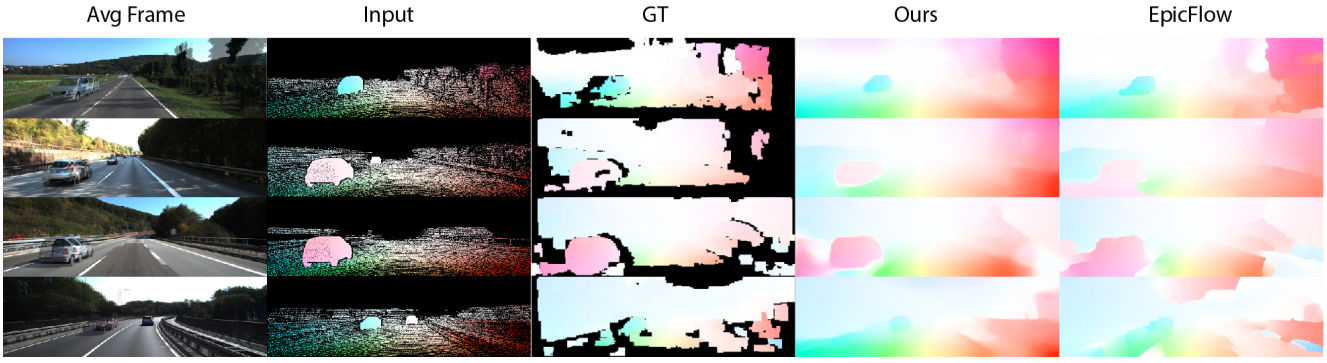

Figure S.6: A comparison of the predictions of our network to EpicFlow on examples from the KITTI 2015 validation set.