Towards Equivariant Optical Flow Estimation with Deep Learning - Supplementary Material -

Stefano Savian[1,2], Pietro Morerio[1], Alessio Del Bue[1], Andrea A. Janes[2], and Tammam Tillo[3]

{*stefano.savian, pietro.morerio, alessio.delbue*}*@iit.it, ajanes@unibz.it, tammam@iiitd.ac.in*
[1]Pattern Analysis & Computer Vision (PAVIS), Istituto Italiano di Tecnologia, Genova, Italy
[2]Free University of Bozen-Bolzano, Bolzano, Italy
[3]Indraprastha Institute of Information Technology Delhi (IIITD), Delhi, India

Contents

1. Introduction

As pointed out in the main paper, this supplementary material addresses and expands certain aspects of our work which could not be presented due to space limitations. In **section** 2, we demonstrate a theoretical upper bound to the sign imbalance. In **section** 3, we investigate and highlight what are the differences between the presented aggregation strategies to obtain O^*, (O^* is necessary to produce I in order to compute our auxiliary sign imbalance loss). In **section** 4, we explicitly present different ways to aggregate I to one or two scalar values. We also integrate the experimental part by further analyzing the experimental data to provide a more comprehensive evaluation of the sign imbalance, and by presenting the results with bar plots. In **section** 5, we report more details on the design of the Kaleidoscope dataset. In **section** 6, we report further testing and training data statistics related to the experiments in the paper that are useful to better identify possible training bias. In **section** 7, the approximation of T_{lr}, T_{ud} with T_{180° is further investigated. In **section** 8, we provide a more in depth analysis of the full-frame results, vertical and horizontal sign imbalance assessment. In **section** 9, we report the results on the thresholded groundtruth motion magnitude.

We also provide more information on the sign imbalance mitigation using our auxiliary loss. In **section** 10, we provide the results for the preliminary experiments with FlowNetC. In **section** 11, we report more details for the experiments with RAFT, already reported in the main paper, but here explained in greater detail.

2. On Relating the sign imbalance and EPE

Example. Suppose that the considered frames are of size 7×7 pixel and that frame $F_n^{'}$ and $F_n^{''}$ are two identical white frames with a dark pixel in the center of the frames. Let the single and double quotes superscript represent two scenarios of motion, namely, scenario I and scenario II. A pictorial representation of the two identical frames is shown in Fig. 1 (a). In scenario I the dark pixel moves up by 2 pixels and to the right by 3 pixels, whereas, in scenario II the dark pixel moves down by two pixels and to the left by three pixels. These two scenarios of motion are depicted in Fig. 1 (a) and (b).

In reference to scenario I let:

- vector $g^{'}$ be used to represent the motion of the pixel
- vector $o^{'}$ be used to represent the estimated motion of the pixel with a hypothetical optic flow estimator

- vector $e^{'}$ be used to represent the difference between the estimated motion with the said estimator and the actual motion, this is to say $e^{'} = g^{'} - o^{'}$. This vector is going to be named the error

In the above mentioned vectors the prime symbol is used as a superscript to indicate scenario I of motion. As for vectors related to scenario II of motion a double prime symbol is used as a superscript. The above mentioned vectors for scenario I, and scenario II, are depicted in Fig. 1 (c). It is worth noticing that $g^{''} = -g^{'}$. One might expect to have $o^{''} = -o^{'}$. If $o^{''}$ is not similar to $-o^{'}$ then this indicates an imbalance behavior of the optic flow estimator. The larger the difference between $o^{''}$ and $-o^{'}$ the larger the said imbalance. Let $o^{\star} = -o^{''}$. Fig. 1 (d) and (e) depicts $o^{'}$, $o^{\star}$, and the vector i where $i = o^{'} - o^{\star}$. Vector i represents the sign imbalance of the optic flow estimates for the two scenarios of motion. Furthermore, Fig. 1 (e) represents the vector $g^{'}$ and $e^{'}$, in addition to the vector $e^{\star}$, where $e^{\star} = g^{'} - o^{\star} = -e^{''}$ The color code used in Fig. 1 (c), (d) and (e), is as follows: vectors g, e and i are depicted in green, blue and red color, respectively.

2.1. Imbalance magnitude upper bound

In section 2 a general hypothetical estimator was considered. Let this generalization be used to evaluate a possible upper bound to the sign imbalance vector magnitude. For a given magnitude of imbalance $||i||$ would it be possible to find a relationship between the said value and $||e^{'}||$ and $||e^{\star}||$?

Theorem 1 (Euclidean Sign imbalance and EPE relation upper bound)**.** *Considering two generic opposite scenarios of motion, scenario I obtained starting from two generic input frames F_n, F_{n+1}, and scenario II generated starting from $T_{180^\circ}(F'_n), T_{180^\circ}(F'_{n+1})$ and additionaly rotating the produced OF, as:*

$$o' = o(F_n, F_{n+1}); \tag{1}$$
$$o^* = T_{180}(o(T_{180^\circ}(F'_n), T_{180^\circ}(F'_{n+1}))). \tag{2}$$

Equation 1 shows the OF for scenario I and eq. 2 shows the OF for scenario II. Considering g' the groundtruth vector for scenario I, and $e' = g' - o'$,$e^* = g' + o^*$ the errors respectively for scenario I and scenario II; the sign imbalance vector associated to e', e^* is $i = o + o^*$. The Euclidean sign imbalance is $||i||$, the Euclidean norm of i. For any pixel considered, the Euclidean sign imbalance $||i||$ is always lower or equal to the sum of the EPE of scenario I and scenario II, $||e||, ||e^*||$, as:

$$||i|| \leq ||e'|| + ||e^*||. \tag{3}$$

Proof. We prove the statement geometrically by considering the triangle with sides e', e^*, i (similar to the one in Fig. -e)). By applying the triangle inequality to the mentioned triangle e', e^*, i, the upper bound of $||i||$ is defined as:

$$||i|| \leq ||e'|| + ||e^*|| \qquad \blacksquare. \tag{4}$$

Equation 4 show that the Euclidean sign imbalance is always lower or equal than the sum of the EPE $||e'||, ||e^*||$. $\square$

And a consequence of theorem 1 is the statement in the next corollary.

Corollary 1. *Equation 3 considers a single pixel. The triangle inequality also holds when applying this reasoning to the expected value (over a set of pixels $\mathcal{P}$) of $||i||, ||e'||, ||e^*||$, and noted as $\overline{||i||}, \overline{||e'||}, \overline{||e^*||}$. For the monotonicity and linearity of the expected value the triangle inequality can be extended as:*

$$\overline{||i||} \leq \overline{||e'||} + \overline{||e^*||}. \tag{5}$$

Equation 5 shows that eq. 3 can be generalized to the mean of the set of pixels considered. Notably, if $\overline{||e'||} = \overline{||e^*||}$, then:

$$\overline{||i||} \leq 2 \cdot \overline{||e'||}. \tag{6}$$

Equation 6 shows that in this scenario, the sign imbalance upper bound becomes two times the EPE, on average.

3. Obtaining O^*

To evaluate the sign imbalance loss it is necessary to produce O^*. When testing the networks, O^*, is obtained by two independent inference of the network. When training the network, O^*, can be obtained similarly. However it is important to maintain the backward computation graph for differentiating during the backward pass. We here report again the strategy involving multiple forward propagations of the network before the backpropagation, to explain it to a finer granularity.

Algorithm 1 Multiple forward propagations.

```
 1: procedure TRAINING-ITERATION($F_n, F_{n+1}, G$)
 2:     $O$ = inference($F_n, F_{n+1}$)
 3:     if FWDs => detach gradients.
 4:     Generate $F_n^{180^\circ}, F_{n+1}^{180^\circ}$
 5:     $O^{180^\circ}$ = inference($F_n^{180^\circ}, F_{n+1}^{180^\circ}$)
 6:     $O^*$ = $T_{180^\circ}(O^{180^\circ})$
 7:     Compute $\mathcal{L}_E(O,G)$
 8:     Compute $\mathcal{L}_I(O,O^*)$
 9:     $\mathcal{L}$= $\mathcal{L}_E$+ $\beta\cdot$ $\mathcal{L}_I$
10:     if FWDg => $\mathcal{L}$= $\mathcal{L}/2$
11:     Compute gradients
12:     Update learnable weights
```

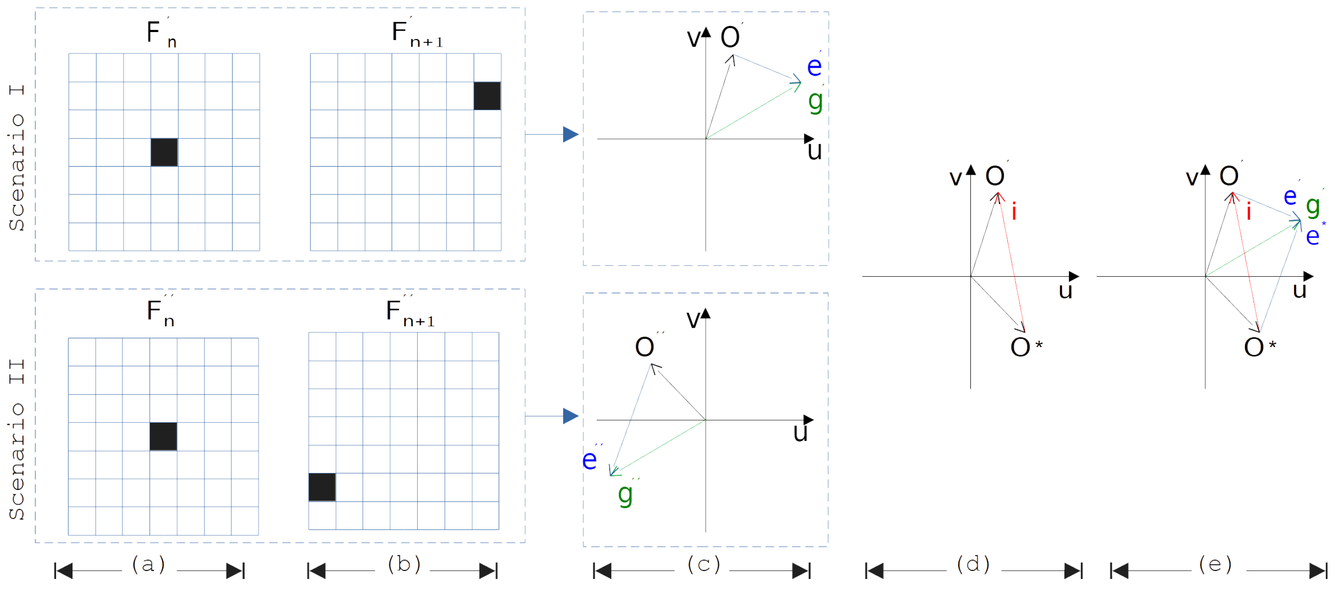


Figure 1: Two scenarios of motion of one dark pixel in two consecutive frames are shown in (a) and (b); (c) is to show for each scenario of motion the actual motion of the dark pixel g, the estimated motion O, and the difference between the estimated motion and the actual motion, this is to say the error e; (d) is to show the sign imbalance of the optic flow estimates for the two scenarios of motion; (e) is to show the sign imbalance of the optic flow estimates in addition to the errors committed by the estimator for the two scenarios of motion.

Referring to Algorithm 1, for every training iteration, F_n, F_{n+1}, are forward propagated to obtain O. After that, $F_n^{180°}, F_{n+1}^{180°}$, are computed and forward propagated into the same network to obtain $O^{180°}$, and then 180°rotated again to obtain O^*. The sign imbalance loss is then computed and the gradients are backpropagated w.r.t. the inputs. The double forward propagation does not require to change the network input size and can be applied to virtually any learnt optical flow estimator.

Using the double forward propagation does not change the architecture, however, the double inference before backpropagation changes the operations to calculate the backward graph. We use pytorch[1] as deep learning framework. Pytorch dynamic computation generates a backward computation graph every time the model is forward propagated. The backward computation graph is then populated once the gradient is calculated by backpropagation. The chain rule is used to calculate the gradient w.r.t. every learnable weight for each operation in the network. Weights are updated based on the gradient values and according to the optimizer. In these conditions, two scenarios are possible, i) when adding the second forward propagation we could either interrupt the backward gradient flow during on the dynamic graph second propagation (FWDs), or populate the second dynamic graph with the gradient values (FWDg).

Figure 2, shows a simplified example of multiple forward propagations of the model before computing the gradients. We use Fig. 2 to further explain the difference between FWDs and FWDg strategies. We assume $f(x, w)$ to be a differentiable function in the network and its output z defined by a value x and learnable weight w. The node x is the input value of the network dependant on the previous operations and states in the graph, finally $L(g, o, o^{180})$ is the loss function calculated at the output side. In 2-a) the network generates o starting from F_n, F_{n+1}. In this step the intermediate states are computed. In 2-b) the network generates o^{180} starting from $F_n^{180°}, F_{n+1}^{180°}$. At step 2-c) (dashed lines) the derivatives of the loss w.r.t. the input are computed with the backpropagation algorithm: the derivatives w.r.t. every element of the network are computed starting from the output and using the chain rule, as in eq. 7, and eq. 8.

$$\frac{\partial L(g,o,o^{180})}{\partial w} = \frac{\partial L(g,o,o^{180})}{\partial o}\frac{\partial o(z)}{\partial z}\dots\frac{\partial f(x,w)}{\partial w}; \tag{7}$$

$$\frac{\partial L(g,o,o^{180})}{\partial w} = \frac{\partial L(g,o,o^{180})}{\partial o^{180}}\frac{\partial o^{180}(z^{180})}{\partial z^{180}}\dots\frac{\partial f(x,w)}{\partial w}. \tag{8}$$

As shown in Fig. 2 the network derivatives of the two branches will display different numerical values for every operation in the network, but will act on the same network weights. Networks states and derivatives in case a) and b) beside the learnable weight w, which is the same weight

[1]www.pytorch.org/

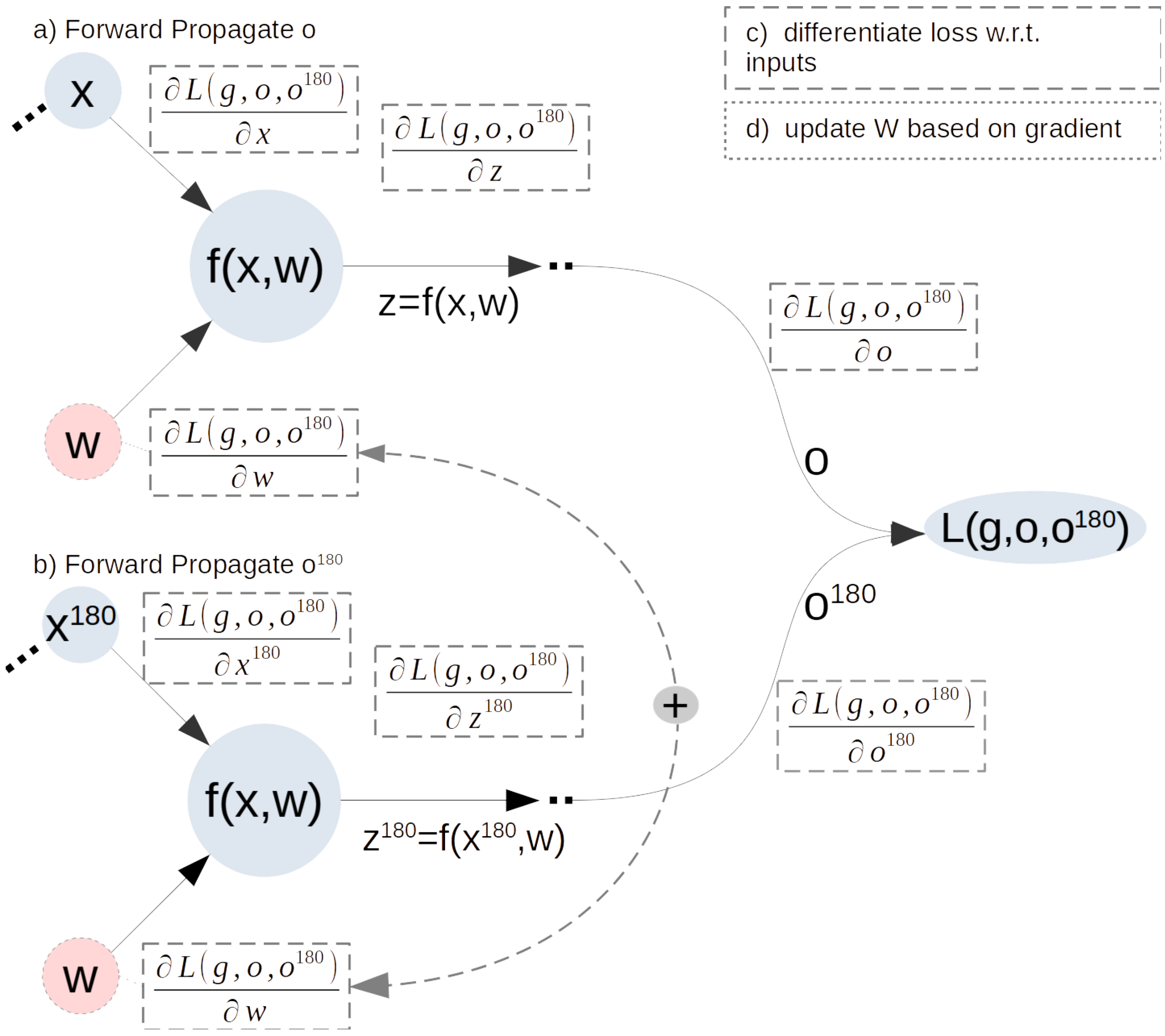

a) Forward Propagate o
x
f(x,w)
z=f(x,w)
w
$\frac{\partial L(g,o,o^{180})}{\partial x}$
$\frac{\partial L(g,o,o^{180})}{\partial z}$
$\frac{\partial L(g,o,o^{180})}{\partial w}$
c) differentiate loss w.r.t. inputs
d) update W based on gradient
$\frac{\partial L(g,o,o^{180})}{\partial o}$
o
L(g,o,o180)
o180
b) Forward Propagate o180
x180
$\frac{\partial L(g,o,o^{180})}{\partial x^{180}}$
$\frac{\partial L(g,o,o^{180})}{\partial z^{180}}$
$\frac{\partial L(g,o,o^{180})}{\partial o^{180}}$
+
f(x,w)
z180=f(x180,w)
w
$\frac{\partial L(g,o,o^{180})}{\partial w}$

Figure 2: Example of double forward propagation before gradient computation for a generic operation $z = f(x, w)$ in the network. Blue circles represent network internal inputs and functions, fine dotted pink circles represent the learnable weights. In step a) the network is forward propagated starting from F_n, F_{n+1}, in b) the network is forward propagated starting from $F_n^{180^\circ}, F_{n+1}^{180^\circ}$, in c) (dashed grey boxes) the differentials are backpropagated starting from the last operation of the loss function, to compute the derivatives for every state in the network. The dashed squares represent the derivatives calculated at every operation in the network. In d) the learnable weight w is updated based on its calculated differential. The differential of w is the accumulation of its differential in step a) and step b), the grey dashed arrow underlines this.

both in case a) and b). Equation 7 for case a), and, eq. 8 for case b); are obtained by explicitly expanding the derivatives as they are computed during step c). Given that both forward passes act on the same weight w, the derivative stored in w (dashed line) is the accumulation (the sum) of case a) and case b), as show by the dashed line in Fig. 2. Finally, in step d) the optimizer updates w. It can be noted that the derivative used for the update step is given by the sum of the derivatives of both the forward propagations. For simplicity, assuming the weight w is updated with batch gradient descent, as in eq. 9, the update step is dependant on the average of the gradients of the two forward propagations as in eq. 10 for FWDg. On the contrary, when stopping the gradients during the second forward propagations, all the derivatives in the second branch are set to zero as in eq. 11 and the update step only depends on the first forward propagation, whereas the second forward propagation is only used to obtain O^*. From a practical perspective, this means that when using the FWDg strategy, the total loss, or the learning rate should be halved, as in eq. 10.

$$w = w - \eta \frac{\partial L(g, o, o^{180})}{\partial w} \tag{9}$$

$$w = w - \eta \frac{1}{2}\left(\frac{\partial f(x,w)}{\partial w} \cdot \frac{\partial o(z)}{\partial z} \cdot \ldots \cdot \frac{\partial L(g,o,o^{180})}{\partial o} + \right.$$
$$\left. + \frac{\partial f(x,w)}{\partial w} \cdot \frac{\partial o(z^{180})}{\partial z^{180}} \cdot \ldots \cdot \frac{\partial L(g,o,o^{180})}{\partial o^{180}} \right) \tag{10}$$

$$w = w - \eta \left(\frac{\partial f(x,w)}{\partial w} \cdot \frac{\partial o(z)}{\partial z} \cdot \ldots \cdot \frac{\partial L(g,o,o^{180})}{\partial o} + 0 \cdot \ldots \cdot 0 \right) \tag{11}$$

4. Testing and Training metrics

The sign imbalance matrix $I = O + O^*$, is a two layer matrix containing the pixelwise horizontal imbalance I_u and vertical imbalance I_v. The matrices I_u and I_v represent the pixel-wise imbalance of the estimated horizontal, and vertical displacement of the optical flow estimator, respectively. A non-zero entry in these matrices indicate an imbalanced behavior associated with the pixel in frame $F_n^{'}$, at the same position of the non-zero entry. A positive value indicates an over-estimate of the up-to-down, and left-to-right, motion, while a negative value indicates the opposite.

Assuming $\mathcal{P}$, the set of considered pixels, which for the full frame statistics is: $\mathcal{P} = h \cdot w$, where h and w represent O height and width. To quantify the sign imbalance behavior over $\mathcal{P}$ a statistical mean measure could be used. We choose the generalized mean, over the set of pixels. So, to quantify the sign imbalance behavior of the estimated vertical displacement of an optical flow estimator for a set of pixels $\mathcal{P}$ the following equation could be used:

$$\overline{I}_v = \left(\frac{1}{\mathcal{P}} \sum_{\forall p \in \mathcal{P}} |i_v(p)|^m \right)^{\frac{1}{m}} . \tag{12}$$

In eq. 12, m determines the statistical mean, the value $(\mathcal{P})$ is the number of pixels in the set $\mathcal{P}$, and $i_v(p)$ the vertical imbalance for a pixel. The usage of the absolute value for $i_v(p)$, in the above equation, imply that the direction of over-estimation is neglected.

Different strategies can be used to aggregate I to one meaningful scalar. Accordingly, the sign imbalance could be calculated for every pixel using the L1 norm, or L2 norm, and averaged using first or second order statistical mean, respectively:

$$\overline{|I|} = \frac{1}{2\mathcal{P}} \sum_{\forall p \in \mathcal{P}} \left(|i_u(p)| + |i_v(p)| \right); \tag{13}$$

$$\overline{||I||} = \frac{1}{(\mathcal{P})} \sum_{\forall p \in \mathcal{P}} ||i(p)||; \tag{14}$$

$$\overline{I_{m=2}} = \frac{1}{2} \left(\frac{1}{\mathcal{P}} \sum_{\forall p \in \mathcal{P}} |i_u(p)|^2 \right)^{\frac{1}{2}} + \frac{1}{2} \left(\frac{1}{\mathcal{P}} \sum_{\forall p \in \mathcal{P}} |i_v(p)|^2 \right)^{\frac{1}{2}} . \tag{15}$$

Equation 13 shows $\overline{|I|}$ the L1 loss averaged. Equation 14 shows $\overline{||I||}$, the arithmetic mean of the per pixel Euclidean distance, also used in the main paper. Equation 15 shows $\overline{I_{m=2}}$, the sign imbalance averaged with statistical mean of power $m = 2$. The value 2 in the denominator of eq. 13 and eq. 15, is to account for the vertical and horizontal sign imbalance equally. When training the networks sign imbalance metric is arbitrary, however, we prefer to weigh the sign imbalance similarly to the training metric used for the accuracy error e, which is commonly the EPE.

5. Kaleidoscope dataset generation

We compose Kaleidoscope by image formation. This dataset contains exactly the same motion magnitude for each direction, thus can be used for an unbiased evaluation of the sign imbalance. Kaleidoscope OF is generated by superimposing circles of different radii over both frames F_n, F_{n+1} on the same position, except for a constant drawn from a Gaussian distribution representing the displacement, for each circle. Each frame pair and OF is composed of four repetitions of the same texture and motion. We refer to each repetition as quadrant. In this paper, quadrant has the same spatial meaning as in the Cartesian system, considering the origin to be located at the center of the frame, q1 is the upper right quadrant, q2 the upper left quadrant, q3 is the lower left quadrant and q4 is the lower right quadrant.

Texture density, motion distribution, object size, object number and frame size are all parameterized. This allows to generate a wide variety of samples. Nonetheless, the motion has been constrained to avoid overlapping objects in frame I_n and to avoid out of quadrant objects in frame I_{n+1}. To guarantee statistical significance we have generated 1000 samples per dataset. The OF has been drawn from a Gaussian distribution with zero mean and 60 px standard deviation. Given the wide variety of samples that could be generated even in these restricted conditions, to ensure a fair comparison of different networks and to ensure results replicability, the random seed has been fixed. Finally, different background and foreground textures have been tested, resulting in similar performance of the network if compared to the “Lena” and “Goldhill” pair; which are the textures used. The scripts used to generate this dataset will be available in the paper repository. Some visualizations are shown in Fig. 3.

6. Testing and training datasets statistics

Table 1 summarize L1 and L2 motion magnitudes $\overline{|G_u|}, \overline{|G_v|}, \overline{||G||}$ for the datasets used in this chapter. Moreover, it specifies the positive and negative amount of motion of $\overline{G_u}, \overline{G_v}$, for the forward and backward optical flow (when available). Finally, the same metrics are evaluated for the training datasets FlyingChairs, FlyingChairs2, FlyingChairsOcc, FlyingThings. We note that for natural sequences, the horizontal motion has always an higher magnitude compared to the vertical motion. We also note that FlyingChairs is more balanced than FlyingThings3D. FlyingThings3D contains a larger motion magnitude to the left, and a larger motion magnitude going downwards. Still referring to table 1, we note that when datasets provide forward and backward optical flow, their $G_u > 0, G_u < 0, G_v > 0, G_v < 0$ values can show a small difference (e.g. FlyingChairs2), or very large differences as for FlyingThings3D depending on the mapping direction.

To evaluate the extent of sign imbalance for different displacements range, we thresholded the groundtruth OF (G) in regions of homogeneous motion. Table 4 $\mathcal{P}$ represent the set of all pixels of the tested sequence; table 2, $\mathcal{P}$ is the set of pixels within a certain magnitude range. Thresholds have been chosen to uniformly cover the data displacement range of the testing datasets. Based on the table 2 for the Sintel row, we note that displacements in the range $[0, 5[$ px cover $\simeq 55\%$ of the moving pixels, $[5, 20[$ cover $\simeq 27\%$ of the moving pixels, $[20, \inf[$ cover $\simeq 18\%$ of the moving pixels. Thus, we chose to mask G using the same values. On monkaa these thresholds uniformly splits the data.

The training dataset FlyingChairs also show a large number of pixels with low displacements, whereas FlyingThings3D show more pixels with a large displacement.

7. Approximating T_{ud}, T_{lr} with T_{180}

In this section we evaluate eq.(12) for $m = 1$, for the horizontal and vertical direction, to obtain the mean of the absolute values, $\overline{I}_u, \overline{I}_v$. We used two approaches to evaluate Eq.(12), these are : 1) T_{ud} to evaluate $\overline{I}_v$ and T_{lr} to evaluate $\overline{I}_u$; 2) T_{180° to evaluate $\overline{I}_v$ and $\overline{I}_u$. This should help understanding if the composition of T_{ud},T_{lr}, is equivalent to the usage of T_{180°, to produce $\overline{I_u}, \overline{I_v}$.

Table 3 reports $\overline{I_u}, \overline{I_v}$ for the transformations T_{lr}, T_{ud}, and T_{180° Ideally the columns $\overline{I}_v, \overline{I}_u$ should display the same values independently on the transformation used. For readability we report the same results in Fig. 4, for Sintel *final*. Referring to Fig.4 it can be noted that all networks can well approximate the two transforms, with the only exception being $\overline{I_u}$ for IRR-PWC fine tuned on KITTI and RAFT trained with mirroring fine tuned on the KITTI dataset R-M'(K) where the sign imbalance difference is around the 20% of the sign imbalance magnitude $\overline{I_u(T_{lr})}$. The results for the Kaleidoscope dataset are similar, all networks can well approximate $\overline{I}_v, \overline{I}_u$, beside IRR-PWC fine tuned on KITTI, where the $\overline{I}_u$ shows a moderate difference.

Furthermore, referring to table 3, by evaluating the ratio $a_{eu} = |\overline{I_v}(T_{ud}) - \overline{I_v}(T_{180^\circ})| \cdot 100/\overline{I_v}(T_{ud})$ (and similarly $a_{ev} = |\overline{I_u}(T_{ud}) - \overline{I_u}(T_{180^\circ})| \cdot 100/\overline{I_u}(T_{ud})$) for all the columns, it is possible to evaluate how well the estimates obtained by the T_{180° rotation approximates the T_{lr}, T_{ud} transforms. Results show, that for all networks the approximation difference on Sintel *clean*, 8% for $\overline{I_u}$ and 3% for $\overline{I_v}$, for the *final* pass is 10% for $\overline{I_u}$ and 4% for $\overline{I_v}$. For the Kaleidoscope dataset the approximation error is 8% for $\overline{I_u}$ and 4% for $\overline{I_v}$ and is 9% for $\overline{I_u}$. The largest deviation in this case is given by RAFT fine tuned on Sintel, for which the approximation error ratio is 55% and 67% respectively on Sintel *clean* and *final*. However when tested on the image formation sequence the error is $\approx 5\%$. The second highest approximation difference on Sintel is given by R-M'(K), (fine tuned on KITTI), which is also the most imbalanced network on the image formation datasets. To conclude, with a good approximation it can be said that the T_{180° transform approximates the T_{lr}, T_{ud} transforms.

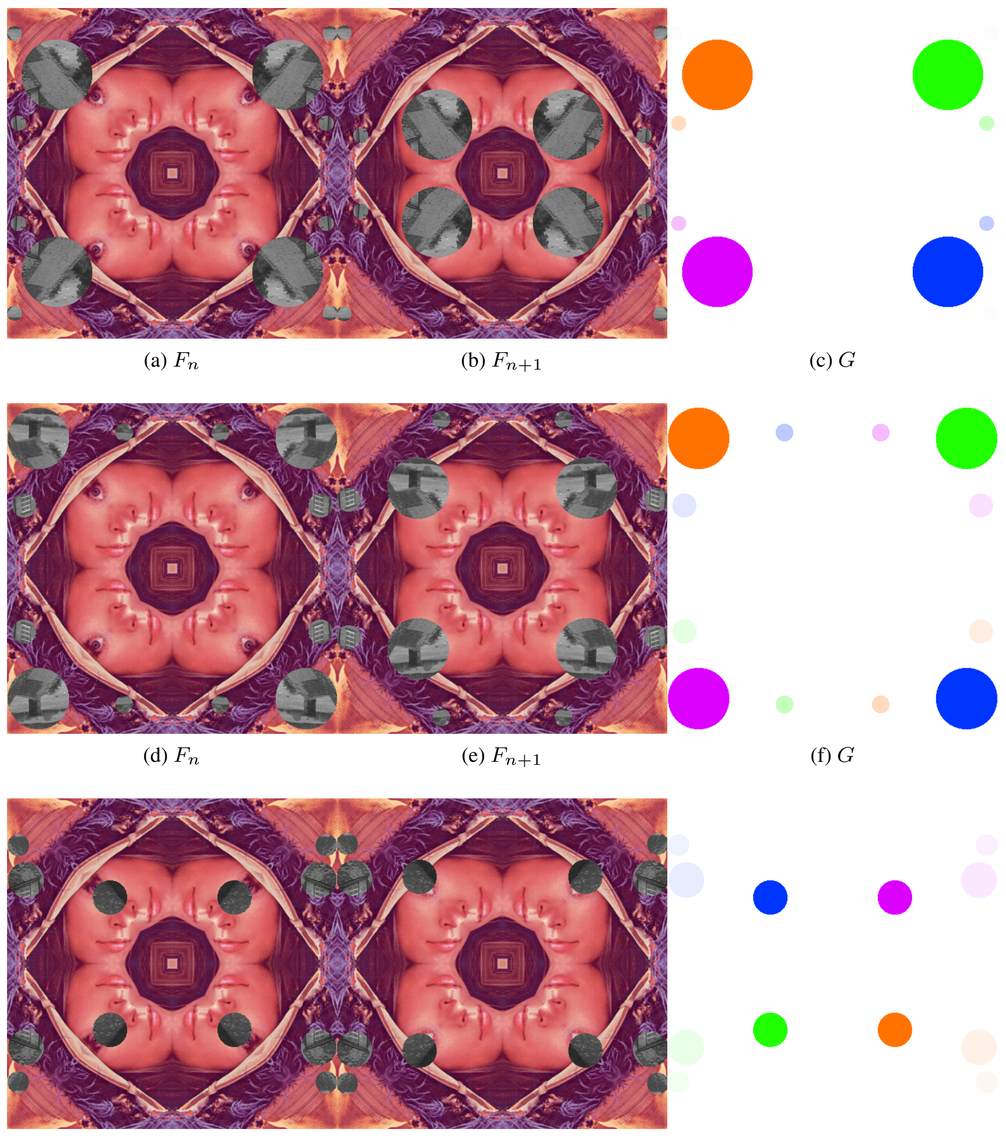
8. Full frame evaluation

We evaluate eq. 12 for $m = 1$ to obtain the mean of the absolute values, $\overline{I}_u$, $\overline{I}_v$, $\overline{||I||}$. Results are evaluated on the full frame for various algorithms for estimating the optical flow. We refer to the table 3 for $\overline{I}_u, \overline{I}_v$. We refer to table 4 for the evaluation of $\overline{||I||}$. Given the large amount of information contained in the tables, results are also presented with bar graphs. Figure 5 reports $\overline{||I||}$ and EPE for all for Sintel *final*, Fig. 7 reports $\overline{I}_u, \overline{I}_v$, Fig. **??** reports the EPE for the Kaleidoscope dataset.

8.1. Vertical and Horizontal Sign Imbalance

By referring to Tab. 3 and Fig. 7 we can evaluate $\overline{I_u}, \overline{I_v}$ to check if the models show higher vertical or horizontal imbalance. We use the results on the Kaleidoscope dataset to evaluate if the models show different imbalance values depending on motion direction. In fact, as shown in Sec. 6 Sintel$|G_u| > |G_v|$, whereas the Kaleidoscope dataset show exactly the same layerwise L1 magnitudes $\overline{|G_u|} = |G_v|$.

By looking at Fig. **??** we note that $\overline{I_u}, \overline{I_v}$ show very similar values for all the models tested, beside models fine tuned on KITTI. In fact, IP(K), R(K) show $\overline{I_u} >> \overline{I_v}$. When evaluated on Sintel the difference is higher, however as noted, this testing dataset is unbalanced. Finally, if frame pairs and groundtruth optical flow are rotated anticlockwise of 90°, for all networks tested on Sintel the sign imbalance show


(a) F_n (b) F_{n+1} (c) G

(d) F_n (e) F_{n+1} (f) G

(g) F_n (h) F_{n+1} (i) G

Figure 3: Visualization of three frame pairs and OF drawn from the Kaleidoscope dataset.

an increase of $\overline{I_u}$ and decrease of $\overline{I_v}$ (results not shown for brevity).

9. Masked regions evaluation

We here evaluate how the sign imbalance varies based on the groundtruth motion magnitude. To do that we masked the OF O in different regions based on the groundtruth motion magnitude $||G||$, and evaluate the sign imbalance in these regions. In this section, $\mathcal{P}$ is the set of pixels within a certain magnitude $||G||$, in the range $0 \leq ||G|| < 5$, $5 \leq ||G|| < 20$, $||G|| \geq 20$ and referred in this section as small, medium and large displacements. The regions thresholds are chosen, based on the considerations of Sec. 6. Results are presented in Tab. 5 for Sintel, and with bar graphs in Fig. 8 and 9 for Sintel *final*. The bar graph in figure 8 shows the sign imbalance evaluated on Sintel *final* thresholded regions. Displacements masked in the range $[0, 5[$, $[5, 20[$, $[20, inf[$ are presented respectively with bars of darker shades of blue with no texture, with sparse white lines, light blue with finer line density. The bar graph in 9 is normalized over the thresholded grountruth magnitudes. Thus, showing $\frac{\overline{||I||}}{||G||}$, the extent of sign imbalance over the groundtruth motion, for each region of with a certain motion magnitude.

The bar graph in Fig. 8 shows that in all cases regions with a larger groundtruth motion magnitude presents

Table 1: Groundtruth motion statistics per motion direction and orientation. L1 and L2 motion magnitudes. FWD and BCK refer to the presence of forward and backward optical flow.

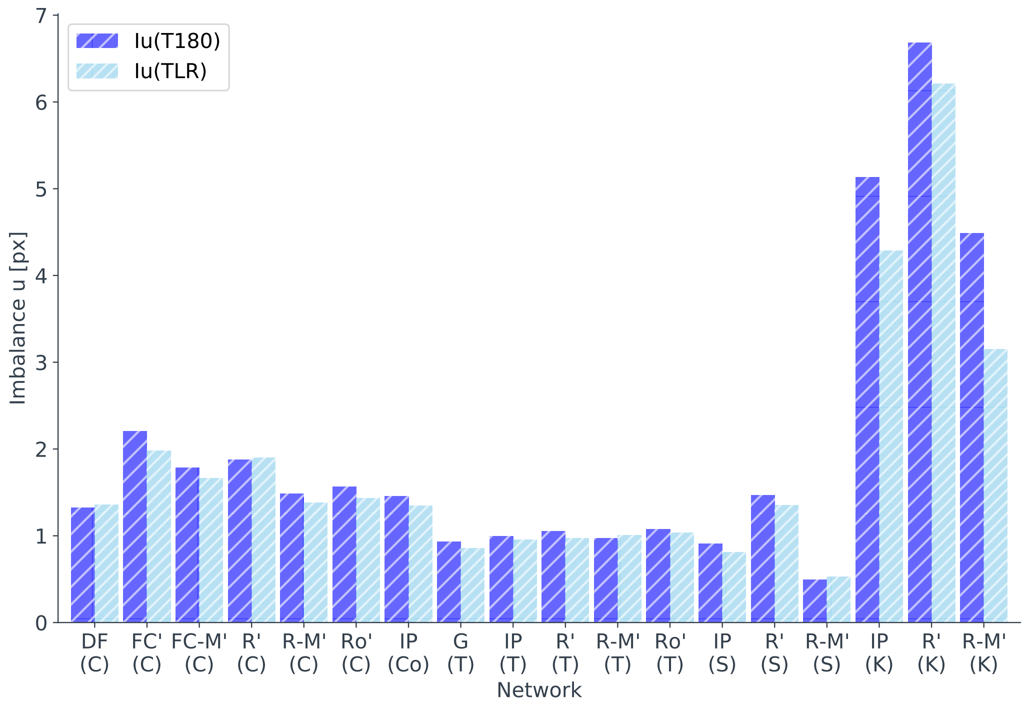
Data	FWD	BCK	$Gu > 0$	$Gu < 0$	$Gv > 0$	$Gv < 0$	$\|Gu\|$	$\|Gv\|$	$\|\|G\|\|$
FlyingChairs	x		7.09	-7.25	7.12	-7.36	11.11	7.1	7.16
FlyingChairs2	x		7.16	-7.27	7.18	-7.51	10.14	6.45	6.55
FlyingChairs2	x	x	7.16	-7.39	7.3	-7.6	10.24	6.5	6.64
FlyingChairs2		x	7.17	-7.5	7.42	-7.68	10.34	6.55	6.73
FlyingChairsOcc	x		7.34	-7.14	7.39	-7.32	11.16	7.09	7.21
FlyingChairsOcc	x	x	7.28	-7.28	7.39	-7.4	11.23	7.14	7.25
FlyingChairsOcc		x	7.23	-7.43	7.39	-7.48	11.3	7.18	7.29
FlyingThings3D	x	x	34.62	-45.39	34.2	-22.27	53.19	40	28.24
FlyingThings3D	x		33.63	-32.94	21.94	-21.03	43.22	33.28	21.48
FlyingThings3D		x	35.62	-57.54	46.09	-23.55	63.14	46.72	35
Kaleidoscope	x		29.51	-29.51	29.51	-29.51	7.7	5.45	5.45
Sintel	x		12.11	-9.66	7.42	-6.5	13.5	10.08	6.66
Kitti2015	x		38.25	-34.98	15.1	-3.81	33.00	11.35	36.7
Hd1k	x		10.67	-11.64	6.94	-2.14	8.98	7.88	3.39
Monkaa	x	x	17.6	-18.03	4.95	-4.92	19.44	17.16	4.75
Monkaa	x		22.43	-11.81	5.44	-5.07	19.83	17.4	5.05
Monkaa		x	10.66	-22.36	4.47	-4.76	19.05	16.92	4.44

Table 2: Thresholded groundtruth motion mean magnitude and pixel counts ratio.

				$\overline{\|\|G\|\|}$ [px]		Pixel Distribution [%]		
data	FWD	BCK	$[0, 5[$	$[5, 20[$	[20,inf[	[0,5[	[5,20[	[20,inf[
FlyingChairs	x		1.58	10.76	39.1	50.03	32.46	17.51
FlyingChairs2	x		1.45	10.63	38.17	52.35	31.99	15.66
FlyingChairs2	x	x	1.45	10.63	38.56	52.3	31.9	15.8
FlyingChairs2		x	1.45	10.63	38.94	52.26	31.81	15.93
FlyingChairsOcc	x		1.54	10.77	38.4	49.4	32.68	17.92
FlyingChairsOcc	x	x	1.55	10.78	38.48	49.25	32.71	18.05
FlyingChairsOcc		x	1.55	10.78	38.56	49.09	32.73	18.18
FlyingThings3D	x	x	3.06	11.74	69.36	10.72	38.92	50.35
FlyingThings3D	x		3.09	11.85	69.75	9.64	38.41	51.95
FlyingThings3D		x	3.04	11.63	68.95	11.8	39.44	48.76
Kaleidoscope	x		0.06	12.7	58.3	83.16	4.75	12.09
Sintel	x		1.65	10.45	54.51	54.95	27.15	17.9
Kitti2015	x		0.08	11.76	63.99	20.10	26.07	53.83
Hd1k	x		1.19	10.59	35.79	55.11	30.73	14.16
Monkaa	x	x	1.73	11.06	51.09	39.15	31.36	29.49
Monkaa	x		1.73	11.06	51.71	39.14	31.34	29.52
Monkaa		x	1.74	11.05	50.47	39.16	31.38	29.46

a larger sign imbalance. For all networks beside R-M'(S), the sign imbalance on Sintel *final* for high displacements is higher than 5 px, (approx 10% of the groundtruth magnitude), as pointed in Fig. 9. When normalized, low and medium displacements are more imbalanced relatively to the groundtruth motion magnitude. For medium displacements the sign imbalance is often around 1 px, or the 20% of the groundtruth motion. Smaller displacements show a lower imbalance, $\overline{||I||} \approx 0.3px$. However, small displacements show the highest sign imbalance over groundtruth magnitude ratio; small displacements are highly unbalanced, up to 60% the groundtruth testing motion.

The bar graph in Fig. 9 shows, for all networks trained on almost any dataset, that the network imbalance is never lower than 10% of $\overline{||G||}$ (excluding the models fine tuned on the testing set, R'(S), R-M'(S) and IP(S)). On average, all networks trained on FlyingChairs present an higher imbalance compared to fine tuning on FlyingThings. When



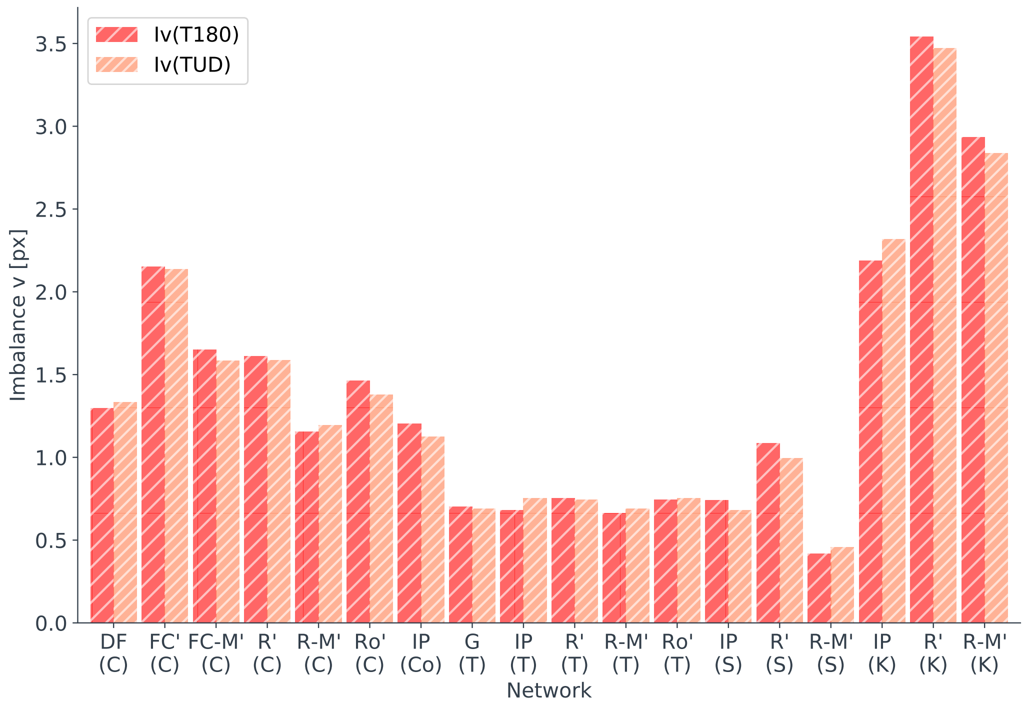


Figure 4: bar graphs showing the approximation of T_{180°,T_{lr}, T_{ud} when evaluating $\overline{I}_u, \overline{I}_v$ on Sintel *final*. The bar graphs color coding is blue with ascending lines for $\overline{I_u}$, red with ascending lines for $\overline{I_v}$; T_{180° transformation is noted by darker tones and double lines, whereas T_{lr}, T_{ud} are noted by lighter tones and single lines. The abscissa shows the model name and training data, the ordinate displays $\overline{I}_v, \overline{I}_u$ measured in pixels. T_{180° can well approximate T_{lr}, T_{ud} for all networks. RAFT-M fine tuned on KITTI (R-M'(K)) shows the highest difference between $I_u(T_{180^\circ})$ and $I_u(T_{lr})$

Table 3: Transforms approximation evaluated on Sintel. All networks can well approximate the sign imbalance $\overline{I_u}, \overline{I_v}$, RAFT and IRR-PWC fine tuned on KITTI show the greatest approximation difference. RAFT fine tuned on Sintel display the highest approximation error a_{eu}.

	Sintel											
	clean						final					
	$\overline{I_u}$		a_{eu}	$\overline{I_v}$		a_{ev}	$\overline{I_u}$		a_{eu}	$\overline{I_v}$		a_{ev}
Label	$T_{180°}$	T_{lr}	%	$T_{180°}$	T_{ud}	%	$T_{180°}$	T_{lr}	%	$T_{180°}$	T_{ud}	%
DF(C)	1.07	1.1	2.4	1.03	1	3.28	1.32	1.36	2.7	1.3	1.33	2.7
FC'(C)	1.82	1.63	11.8	1.69	1.7	0.5	2.21	1.98	11.5	2.15	2.14	0.8
FC-M'(C)	1.46	1.34	9.2	1.38	1.3	4.36	1.79	1.66	7.4	1.65	1.58	4.2
Ro'(C)	0.79	0.81	2.6	0.75	0.8	6.8	1.56	1.43	9.1	1.46	1.38	6.1
R-M'(C)	0.76	0.76	0.4	0.75	0.7	1.75	1.49	1.38	7.6	1.15	1.2	3.4
R'(C)	0.98	0.98	0.3	0.92	1	5.23	1.88	1.9	1.2	1.61	1.59	1.5
Ro(C)	0.78	0.71	9.9	0.81	0.8	0.78	1.51	1.38	9.6	1.43	1.4	1.9
R-M'(C2)	0.71	0.7	1.6	0.69	0.7	8.19	1.19	1.1	7.6	1.02	0.96	6.8
R-M'(C2f)	0.75	0.71	5	0.75	0.7	4.58	1.28	1.15	10.7	1.31	1.22	7.8
IP(Co)	0.89	0.91	2.4	0.75	0.7	6.93	1.46	1.35	8	1.2	1.12	7
G(T)	0.51	0.48	7.4	0.4	0.4	6	0.93	0.86	8.4	0.7	0.69	2
IP(T)	0.65	0.64	2.5	0.51	0.5	2.94	0.99	0.96	4	0.68	0.75	9.8
Ro'(T)	0.71	0.76	6.4	0.44	0.5	4.2	1.07	1.04	3.7	0.74	0.75	1.1
R-M'(T)	0.55	0.56	1.7	0.41	0.4	1.67	0.97	1.01	3.4	0.66	0.69	3.8
R'(T)	0.88	0.82	7	0.47	0.5	1.12	1.05	0.97	8.4	0.75	0.74	1.2
Rs(T)	0.91	0.82	10.8	0.69	0.7	4.96	1.23	1.14	7.5	0.87	0.86	0.9
Ro(T)	0.6	0.57	4.6	0.44	0.5	3.41	1.01	1.02	0.8	0.72	0.78	8
R-M'(Tf)	0.69	0.69	0	0.41	0.4	5.17	1.05	0.96	9.1	0.68	0.63	7.6
R-M'(C2-T)	0.56	0.62	10.2	0.4	0.4	6.48	0.9	0.86	5.1	0.73	0.7	3.4
IP(S)	0.68	0.65	3.7	0.55	0.6	0.95	0.91	0.81	11.9	0.74	0.68	8.6
Ro(S)	0.48	0.31	55	0.4	0.4	1.01	0.79	0.47	66.5	0.69	0.7	1.7
R-M'(S)	0.39	0.34	14.1	0.3	0.3	4.84	0.49	0.53	6	0.42	0.46	9
R'(S)	0.76	0.73	3	0.56	0.5	3.5	1.47	1.35	8.5	1.08	0.99	9
IP(K)	6.08	5.69	6.9	2.57	2.5	1.43	5.13	4.29	19.8	2.19	2.32	5.7
R-M'(K)	3.02	2.35	28.8	1.98	2	1.13	4.48	3.15	42.4	2.93	2.84	3.4
R'(K)	4.11	4.21	2.3	2.35	2.5	7.29	6.68	6.21	7.5	3.54	3.47	1.9
Ro'(K)	3.33	3.56	6.4	1.92	1.8	5.49	5.05	4.82	4.7	2.68	2.55	4.8
MEAN	1.22	1.15	8.1	0.86	0.9	3.79	1.72	1.55	11.1	1.25	1.23	4.6

networks are fine tuned on FlyingThings, the sign imbalance is reduced on average by 40% its value when trained on FlyingChairs. Referring to Fig. 8 it can be noted that large displacements sign imbalance is noticeably higher for FlyingChairs w.r.t. FlyingThings. At a closer look this also holds for medium and low displacements. This can be better observed in Fig. 9. When normalizing the sign imbalance over the groundtruth, it can be noted that low displacements sign imbalance is drastically reduced of 50% (compared to its value on FlyingChairs), for all displacements range. However, it is not possible to directly compare FlyingChairs and FlyingThings3D because FlyingChairs do not provide backward optical flow. Referring to Tab. 5 by evaluating, IP(Co) R-M'(C2), and IP(T), R-M'(C2-T), we can more accurately evaluate the effect of fine tuning on FlyingThings. By comparing R-M'(C2), R-M'(C2f), we notice that the two networks show to the same imbalance and same EPE for all the thresholds. Also R-M'(C) shows similar performance, however it displays an higher imbalance on Sintel *final*. However, these result show that the penalty on Sintel *final* is not due to the lack of forward and backward optical flow of FlyingChairs.

Finally, we note that the sign imbalance mitigation provided by training FlyingChairs2 does not lead to better performance after fine tuning on FlyingThings. This is shown by comparing R-M'(T) and R-M'(C2-T) (Tab. 5). Retraining on FlyingChairs2 on forward and backward optical flow does not further reduce the sign imbalance if compared to fine tuning on FlyingChairs as R-M'(T) and R-M'(C2-T) show the same imbalance.

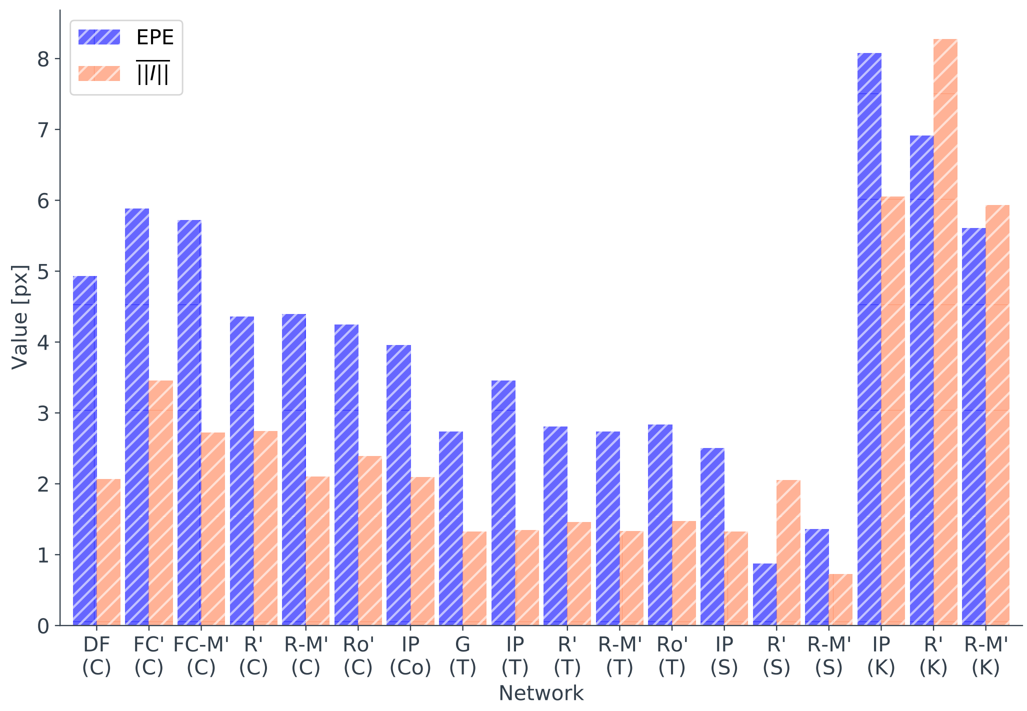


Figure 5: Full frame Sintel *final* results. Sign imbalance and EPE are reduced when fine tuning on FlyingThings3D. Fine tuning on KITTI leads to largely unbalanced models.

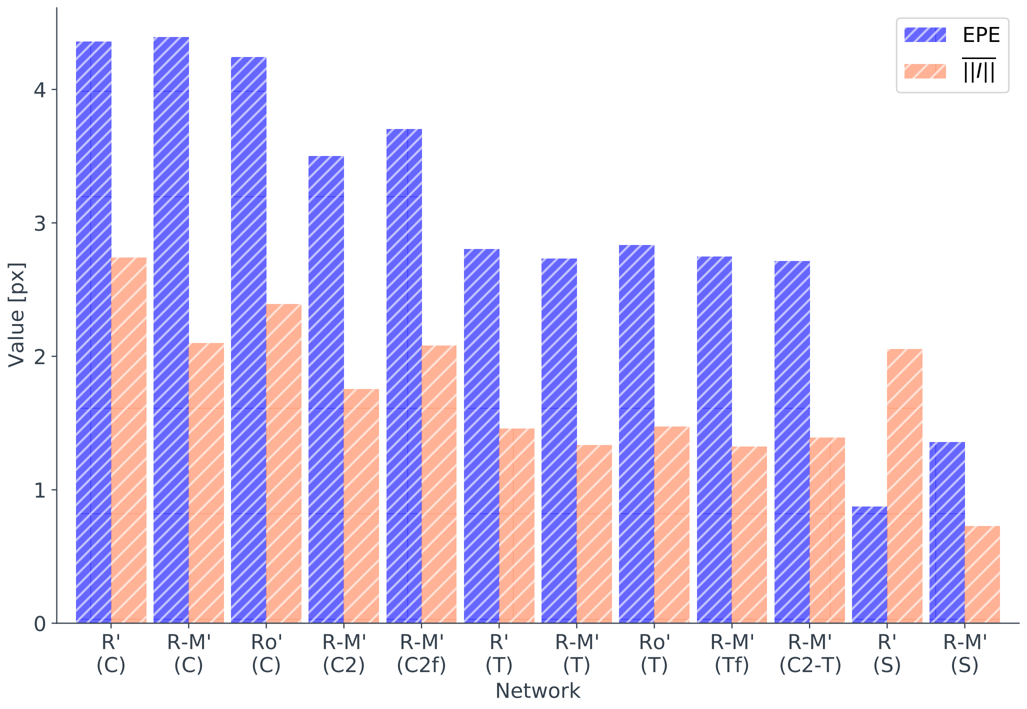


Figure 6: RAFT Full frame results on Sintel *final*, for different training datasets. Training on forward and backward optical flow does not mitigate the sign imbalance significatively.

Table 4: Full frame results on Sintel and on the image formation datasets. Yellow highlighting is used to show the effect of mirroring, green highlighting is used to show the effect of training on forward and backward optical flow. Mirroring partially mitigates the sign imbalance, whereas training on forward and backward optical flow has a limited effect.

	Sintel				Kaleidoscope
	clean		final		
Label	EPE	$\overline{\|\|I\|\|}$	EPE	$\overline{\|\|I\|\|}$	EPE
DF(C)	3.85	1.66	4.93	2.06	5.07
FC'(C)	4.57	2.77	5.88	3.45	4.6
FC-M'(C)	4.42	2.24	5.72	2.72	4.52
Ro'(C)	2.19	1.22	4.24	2.39	1.41
R-M'(C)	2.19	1.2	4.39	2.1	1.41
R'(C)	2.25	1.49	4.36	2.74	1.59
Ro(C)	2.15	1.27	4.44	2.35	1.43
R-M'(C2)	2.15	1.11	3.5	1.75	1.51
R-M'(C2f)	2.14	1.19	3.7	2.08	1.46
IP(Co)	2.34	1.3	3.96	2.09	3.66
G(T)	1.3	0.73	2.73	1.32	1.26
IP(T)	1.87	0.92	3.46	1.35	2.66
Ro'(T)	1.58	0.95	2.83	1.47	1.13
R-M'(T)	1.42	0.78	2.73	1.33	1.08
R'(T)	1.54	1.13	2.8	1.46	1.16
Rs(T)	2.13	1.28	3.31	1.7	2.58
Ro(T)	1.47	0.84	2.71	1.4	1.14
R-M'(Tf)	1.43	0.91	2.71	1.39	1.18
R-M'(C2-T)	1.45	0.77	2.75	1.32	1.18
IP(S)	1.91	0.98	2.5	1.32	4.5
Ro(S)	0.74	0.7	1.19	1.17	1.14
R-M'(S)	0.83	0.55	1.36	0.73	1.14
R'(S)	0.56	1.05	0.87	2.05	1.25
IP(K)	7.41	7.07	8.07	6.05	8.46
R-M'(K)	3.95	4.03	5.6	5.93	3.99
R'(K)	4.63	5.24	6.91	8.27	4.36
Ro'(K)	4.43	4.27	6	6.23	4.01

When fine tuning RAFT on Sintel and testing on Sintel, the sign imbalance is halved compared to only training on FlyingThings, for small and medium displacements; large displacements sign imbalance is reduced but not halved. This can be observed by comparing IP(T) and R-M'(T), and IP(S) and R-M'(S), in Fig. 8. However, the sign imbalance in Fig. 9, seems noticeably reduced if compared to its value when trained FlyingThings. If R-M'(T), R-M'(S) are evaluated on a completely different dataset, the Kaleidoscope dataset, R-M'(S) sign imbalance is not reduced, but is slighlty worsen compared to R-M'(T), for large displacements. Finally, as noted by IP(S) and R-M'(S), the sign imbalance is dramatically worsen when fine tuning on KITTI, accounting for over the 40 % motion magnitude for all thresholds.

We evaluate the performance of different models trained on FlyingChairs, DF(C), FC-M'(C), R-M'(C), IP(Co). DDFlow show the lowest imbalance among the models mentioned. RAFT is the second least imbalanced network, but shows a noticeably larger sign imbalance for small displacements, compared to DF(C), Fig. 9. We note that R-M'(C2) and R-M'(C2F) show similar sign imbalance values as R-M'(C), Tab. 5. IRR-PWC shows the same trend of RAFT. FlowNetC shows the highest imbalance, reaching almost 60% of the motion magnitude in Fig. 9.

Models highlighted in yellow in Tab. 5 can be used to evaluate the effects of the mirroring data augmentation, for different networks and for different datasets. FlowNetC-M, sign imbalance $\overline{||I||}$, is reduced of 2 px and the EPE is reduced of 0.6 px, for large displacements. For medium displacements the sign imbalance is reduced of 0.3 px and the EPE is almost unchanged. For small displacements the sign imbalance is reduced of 0.13 px and the EPE is unchanged. Similar effects can also be observed for RAFT trained on FlyingChairs and on FlyingThings. When fine tuning on KITTI the sign imbalance is greatly reduced for

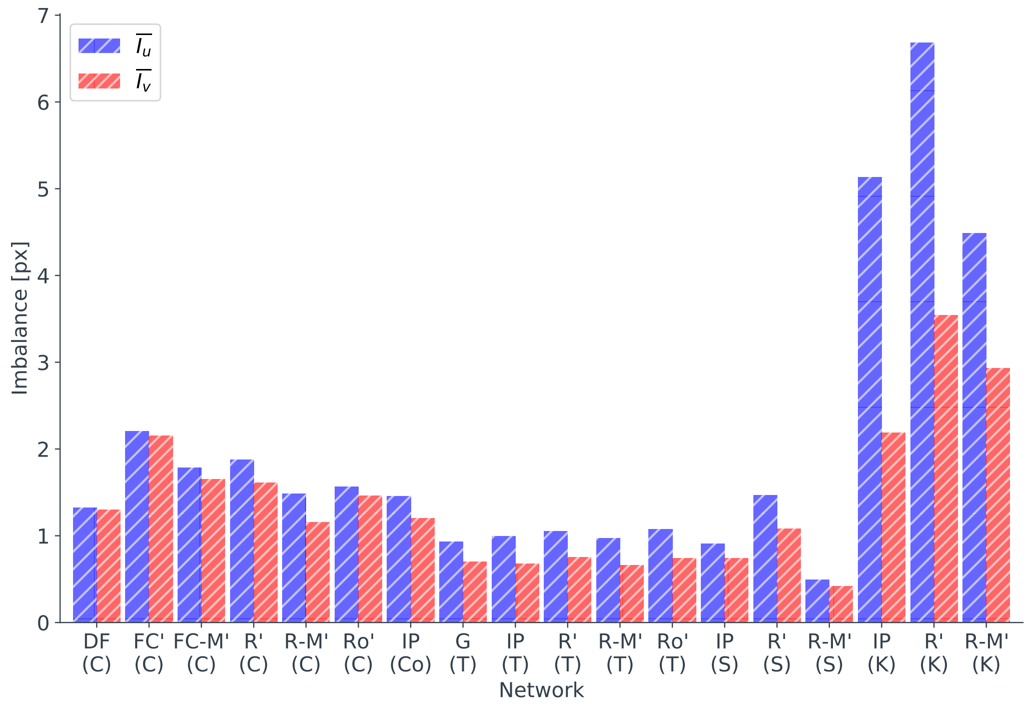


Figure 7: Per axis L1 imbalance evaluated on Sintel *final* for all networks, on this test set there seems to be a slighly larger horizontal imbalance. Fine tuning on KITTI lead to a higher horizontal imbalance.

large displacements when applying the mirroring data augmentation. Similarly, when fine tuning on Sintel the sign imbalance is significantly reduced for all displacements.

For all networks, the mean relative imbalance $I_R = \overline{||I||/EPE}$ for small and medium displacements is around 0.75 , for large displacements $I_R \approx 0.65$. This trend suggests the sign imbalance is overall almost constant relatively to the EPE, but also reduced of approximately 10% for high displacements. However, if examined one by one, the networks show different trends. For all displacements range, the relatively most imbalanced network is still RAFT fine tuned on Sintel or KITTI, for which the sign imbalance is higher than the EPE for low displacements. IRR-PWC fine tuned on Sintel shows a relative imbalance of 0.60, not substantially higher than IRR-PWC trained on Things. IRR-PWC shows also a relative lower imbalance if compared to RAFT, mostly for smaller displacements. RAFT overall presents a relative imbalance of 0.7 for displacements up to 20 px and 0.50 for higher displacements. GMA shows a higher relative imbalance for low displacements if compared to RAFT. FlowNetC and FlowNetC-M shows an overall high imbalance for low displacements, but a moderate to low imbalance for higher displacements. DDFlow, shows a constant imbalance for all thresholds, obtaining the lowest relative imbalance among all networks.

10. Preliminary experiments with FlowNetC

In this section we present the experiments carried with FlowNetC. The section is structured as follows. Section 10.1 compares different aggregation strategies to obtain O^*. Section 10.2 illustrates the hyperparameter tuning procedure. Finally, Sec. 10.3 presents the results in terms of EPE and sign imbalance mitigation. FlowNetC baseline training details are the same used in the main paper.

10.1. Comparing different aggregation approaches

The impact on accuracy of the different strategies, FlowNet-FWDs, and FlowNet-FWDg (as described in sec. 3) is evaluated by removing the mirroring data augmentation stage and setting $\beta = 0$ during training. Still referring to table 7 it can be noted that in all cases for $\beta = 0$, the changes introduced increase the EPE of roughly 0.2 px. The effect is more severe on Sintel *final*.

10.2. Hyperparameter tuning

Results are reported for the most relevant values of β, for models trained with the full schedule. As mentioned in Sec. 4 during training we match the accuracy training metric of the model, which for FlowNetC is the EPE. Thus, in this section the sign imbalance loss is based on $\overline{||I||}$.

Figure 10 shows the loss function for different β

Table 5: Full frame results on Sintel. Yellow highlighting is used to show the effect of mirroring, green highlighting is used to show the effect of training on forward and backward optical flow. Mirroring partially mitigates the sign imbalance for large displacements, particularly when fine tuining on Sintel and KITTI. Training on forward and backward optical flow has a limited effect.

	Sintel											
	clean						final					
	$0 \leq \overline{\|\|G\|\|} < 5$		$5 \leq \overline{\|\|G\|\|} < 20$		$\overline{\|\|G\|\|} \geq 20$		$0 \leq \overline{\|\|G\|\|} < 5$		$5 \leq \overline{\|\|G\|\|} < 20$		$\overline{\|\|G\|\|} \geq 20$	
label	EPE	$\overline{\|\|I\|\|}$	EPE	$\overline{\|\|I\|\|}$	EPE	$\overline{\|\|I\|\|}$	EPE	$\overline{\|\|I\|\|}$	EPE	$\overline{\|\|I\|\|}$	EPE	$\overline{\|\|I\|\|}$
DF(C)	0.48	0.22	2.28	1.01	16.61	7.05	0.64	0.26	3.03	1.36	21.01	8.67
FC'(C)	1.08	0.9	3.36	2.24	17.16	9.28	1.31	1.06	4.3	2.95	22.33	11.57
FC-M'(C)	1.02	0.77	3.38	1.93	16.48	7.25	1.29	0.88	4.4	2.47	21.35	8.73
Ro'(C)	0.42	0.29	1.35	0.92	8.9	4.5	0.67	0.48	2.51	1.97	17.84	8.88
R-M'(C)	0.4	0.25	1.33	0.78	8.98	4.75	0.68	0.43	2.69	1.57	18.38	8.04
R'(C)	0.41	0.34	1.39	1.21	9.21	5.45	0.63	0.56	2.69	2.32	18.34	10.06
Ro(C)	0.41	0.29	1.31	0.95	8.76	4.76	0.71	0.53	2.52	1.91	18.79	8.59
R-M'(C2)	0.4	0.25	1.29	0.78	8.82	4.27	0.54	0.31	2.1	1.25	14.69	6.93
R-M'(C2f)	0.39	0.25	1.38	0.87	8.64	4.54	0.63	0.4	2.43	1.38	15.07	8.31
IP(Co)	0.35	0.2	1.25	0.74	10.13	5.54	0.57	0.3	2.44	1.51	16.66	8.5
G(T)	0.21	0.17	0.72	0.61	5.54	2.65	0.39	0.25	1.61	0.9	11.61	5.25
IP(T)	0.3	0.19	1.06	0.59	7.9	3.69	0.54	0.26	2.16	1.12	14.38	5.04
Ro'(T)	0.27	0.21	1.26	1.01	6.11	3.15	0.39	0.26	1.81	1.16	11.87	5.66
R-M'(T)	0.24	0.17	0.9	0.63	5.87	2.85	0.39	0.25	1.6	1.05	11.62	5.08
R'(T)	0.27	0.24	1.22	1.34	5.93	3.54	0.37	0.26	1.5	1.02	12.23	5.8
Rs(T)	0.36	0.26	1.32	0.99	8.81	4.87	0.48	0.32	2.07	1.52	13.84	6.22
Ro(T)	0.26	0.19	0.97	0.73	5.98	3.01	0.38	0.25	1.62	0.99	11.54	5.54
R-M'(Tf)	0.25	0.21	0.93	0.97	5.82	2.97	0.39	0.25	1.55	0.91	11.59	5.64
R-M'(C2-T)	0.26	0.17	0.98	0.67	5.81	2.78	0.38	0.23	1.56	0.91	11.8	5.31
IP(S)	0.24	0.15	0.86	0.52	8.63	4.2	0.3	0.18	1.13	0.64	11.33	5.85
Ro(S)	0.14	0.13	0.5	0.43	2.92	2.84	0.2	0.18	0.68	0.59	5.01	5.09
R-M'(S)	0.15	0.11	0.52	0.36	3.39	2.17	0.19	0.13	0.72	0.47	5.91	2.93
R'(S)	0.13	0.16	0.43	0.55	2.11	4.53	0.17	0.25	0.58	0.94	3.48	9.27
IP(K)	0.76	1.19	3.96	3.41	33.06	30.67	0.84	0.7	5.14	4.85	34.72	24.29
R-M'(K)	0.68	0.7	2.76	3.01	15.8	15.78	0.89	0.96	3.88	4.34	22.69	23.61
R'(K)	0.56	0.68	2.84	3.2	19.84	22.35	0.84	1.06	4.23	5.53	29.61	34.58
Ro'(K)	0.65	0.66	2.68	2.95	18.67	17.38	0.87	0.85	3.6	4.07	25.4	26.05

for FlowNetC-FWDs. Despite some minor differences, FlowNetC-ST and FlowNetC-FWDg show similar trends. Fig. 10 b) shows the loss function for the entire training, Fig. 10 a) is a zoom in of the first 2000 iterations. Referring to 10 b) the black thick line shows the EPE loss for $\beta = 0$, the thin black line with X markers shows the sign imbalance loss. Still referring to Fig. 10, for $\beta = 0$ it can be noted that $LE[1] \approx 207$ and $LE[300k] \approx 16$, instead it can be noted that the sign imbalance starts relatively low, $LI[1] \approx 37$, reaches its maximum at $LI[850] \approx 125$ and decreases naturally to $LI[300k] \approx 17$. The highest imbalance is produced when the EPE loss strongly decreases. Finally, with $\beta = 0$ it can be noted that EPE and sign imbalance loss present similar values.

Still referring to Fig. 10 a), the effect of different β values on the sign imbalance training loss can be evaluated. Overall, for all architectures, a slight imbalance reduction is observed for $\beta = 0.3$, values between $0.3 > \beta > 0.6$ further lower EPE and sign imbalance. The best trade-off between imbalance and accuracy is found at $\beta = 0.6$ higher values of β reduce imbalance but penalize accuracy due to the dominance of the auxiliary loss backpropagated gradients. Moreover, as it can be observed in Fig. 10 b) β has a damping effect on imbalance, higher values of beta reduce the initial imbalance loss overshoot amplitude, but on the contrary large values of β delay the learning of the EPE loss.

10.3. Results

In the remainder of this chapter results are presented with tables. The tables show the EPE and the sign imbalance $\overline{||I||}$ on the full frame of the testing datasets. In the tables,

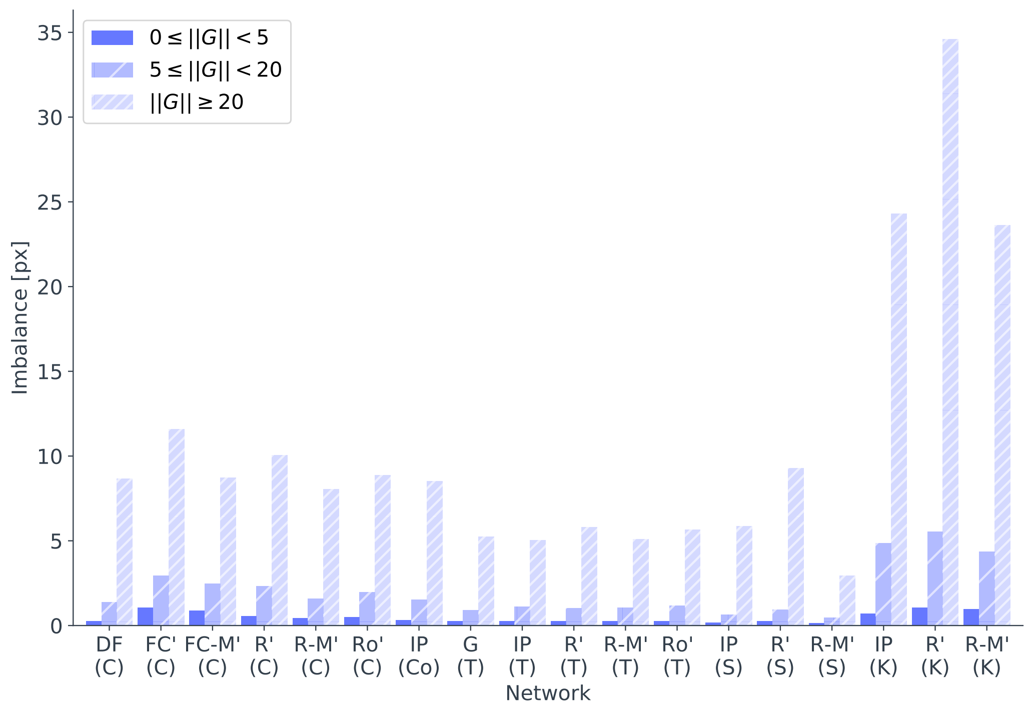


Figure 8: L2 sign imbalance evaluated on masked regions of Sintel *final* for all networks. Higher displacements lead to a higher sign imbalance. Fine tuning on FlyingThings3D considerably reduces the sign imbalance for all magnitudes.

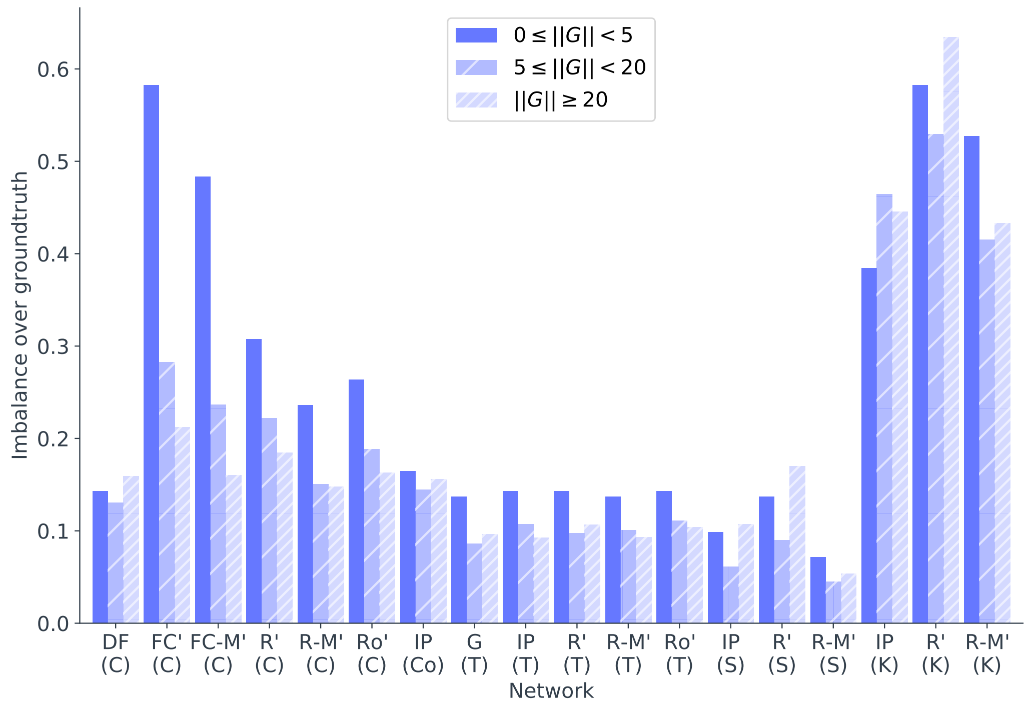
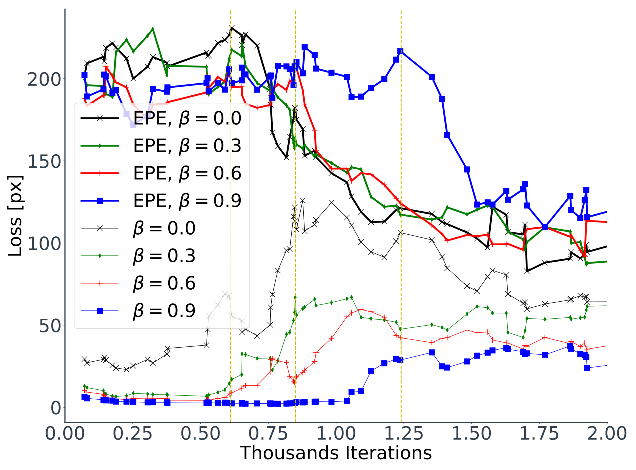
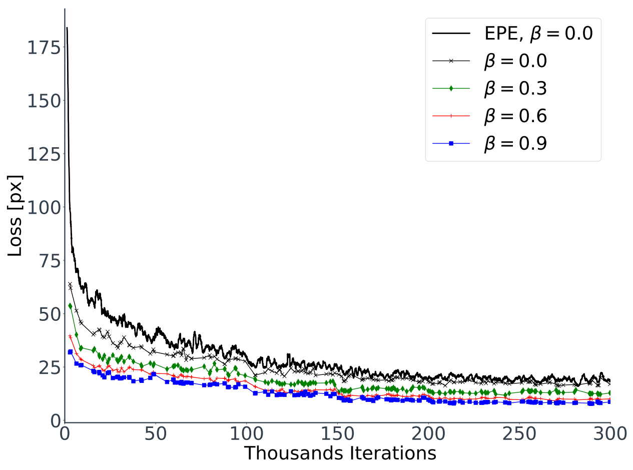


Figure 9: L2 sign imbalance evaluated on masked regions of Sintel *final* for all networks normalized over $\overline{||G||}$. FlowNetC is largely imbalanced for small displacements. Training on FlyingThings3D reduces the normalized imbalance for all thresholds.



(a)



(b)

Figure 10: FlowNetC-FWDs training loss function for different values of β, the abscissa axis shows the (thousands) iterations, the ordinate shows the loss values. Loss values are smoothed by moving average with a window of 50 samples in b) and of 5 samples in a). In b) the black thick line shows the EPE for $\beta = 0$, for clarity the EPE loss for other values of β is not shown since the difference is minimal. The β values are $0.0, 0.3, 0.6, 0.9$ and displayed respectively by black line with X marker, green line with diamond marker, red line with cross marker, blue line with squared markers. The thinner lines represent the value of imbalance loss. It can be seen that higher values of β lower the sign imbalance loss. a) first 2000 iterations zoom in. Thicker lines refer to EPE loss, thin lines show imbalance loss, legend is the same as in b), for clarity loss values for $\beta = 0.3$ are not shown. Vertical dashed yellow lines show where the EPE loss starts decreasing for different β values. For $\beta = 0$ at the sudden decrease of EPE loss corresponds a sudden increase of sign imbalance loss. The same effect is observed for other values of β. β has a damping effect on the sign imbalance, larger values of β reduce imbalance loss overshoot but also shift forward the optical flow learning process.

models marked with “x” use mirroring data augmentation, whereas models marked with “-” do not use data augmentation during training. The table cells are conditionally formatted, red highlighted cells are values showing a considerable penalty compared to the baseline, green highlighted cells are values considerably better than the baseline, cells highlighted in yellow are values very close to the baseline. Darker red color shades mean the error value is higher than others, darker green color shades means the error is lower than others, darker yellow shades mean the error is close to the baseline, but showing slightly higher values.

Table 6 reports the results in terms of EPE and L2 imbalance on Sintel for FlowNetC-FWDs and FlowNetC-FWDg. Overall, higher values of β lead to an higher sign imbalance mitigation. FlowNetC-FWDg with mirroring data augmentation, for $\beta = 0.3$ obtains substantial gains on Sintel *final* both in accuracy and sign imbalance mitigation, if compared with FlowNetC-FWDg without mirroring data augmentation for the same beta value. For FlowNetC-FWDg and FlowNet-FWDs the optimal value of β is 0.3 to minimize the EPE and 0.6 to minimize the sign imbalance. Overall, there is an EPE decrease only on Kaleidoscope, whereas Sintel *final* suffer the highest EPE increase. On the contrary, the sign imbalance is considerably reduced on all the testing datasets. FlowNetC-FWDs and FlowNetC-FWDg benefit of the mirroring data augmentation stage only for $\beta \leq 0.3$. Gains are both in terms of accuracy and sign imbalance mitigation. Higher values of β penalize the accuracy, especially on Sintel *final*. Finally, it is important to note that due to the nature of convolutional neural networks, different training runs lead to slight different results, during the experiments 0.1 px EPE or imbalance difference can be observed when evaluating different training runs of the same model with the same parameters. The large value of EPE and sign imbalance loss is due to the fact that for each upsampling FlowNetC layer, the EPE loss for FlowNetC is summed for every pixel, at different resolutions, resulting in a very large loss.

Table 6: FlowNetC trained on FlyingChairs results. Cell color coding is red for values showing higher EPE or imbalance compared to the baseline, green for error values lower than the baseline. Darker color shades mean a larger deviation from the baseline.

			Sintel				Kaleido-
			clean		final		scope
	Mir.	beta	EPE	$\overline{\|\|I\|\|}$	EPE	$\overline{\|\|I\|\|}$	EPE
base.	-	-	4.57	2.77	5.88	3.45	4.60
	x	-	4.42	2.24	5.72	2.72	4.52
ST	-	0	4.87	3.07	6.16	3.71	4.57
	-	0.3	4.37	1.88	5.71	2.43	4.35
	x	0.3	4.55	1.96	5.92	2.55	4.34
	-	0.6	4.50	1.62	5.78	2.00	4.68
	x	0.6	4.54	1.61	5.91	1.96	4.73
	-	0.9	Not Converging				
FWDS	-	0	4.72	2.97	6.17	3.80	4.44
	-	0.3	4.73	2.21	6.14	2.72	4.29
	x	0.3	4.54	2.07	6.01	2.55	4.09
	-	0.6	4.62	1.95	6.01	2.41	4.34
	x	0.6	4.76	1.89	6.18	2.27	4.23
	-	0.9	4.75	1.64	6.06	2.00	4.93
FWDG	-	0	4.72	2.69	6.07	3.36	4.36
	-	0.3	4.96	3.04	6.26	3.51	4.91
	x	0.3	4.49	1.95	5.86	2.33	4.07
	-	0.6	4.83	2.66	6.12	3.12	4.78
	x	0.6	4.76	1.58	6.09	1.93	4.33
	-	0.9	4.78	2.15	6.12	2.44	4.76

11. Experiments with RAFT

This section presents the experiments carried with RAFT. Section 11.1 present the baseline values per training dataset. Section 11.2 present the evaluation of RAFT trained on FlyingChairs, Sec. 11.3 presents the evaluation of RAFT fine tuned on FlyingThings. Section 11.4 presents the results when fine tuning RAFT on the target dataset. Finally, RAFT is trained on the error e using the L1-norm. The experiments with RAFT are carried with L1 sign imbalance loss, where not otherwise stated.

11.1. Baseline

Table 7 shows the performance of RAFT for different training runs. The training runs are grouped by training set, C refers to training on FlyingChairs and T refers to pretraining on FlyingChairs and fine tuning on FlyingThings. The mean and standard deviation of all training runs is also shown. When modifying the batch size, the total number of flow update iterations has been changed accordingly, to maintain an equal total number of optical flow updates as the baseline. The rows highlighted in yellow are used as baselines for the EPE and the sign imbalance. Models are considered to perform similarly to the baseline when their computed EPE and sign imbalance do not diverge more than two times the standard deviation computed in table 7.

By comparing the EPE and the sign imbalance values of the baseline in table 7 for FlyingChairs and for FlyingThings, we note that the sign imbalance and EPE is considerably reduced of around 0.8 px and 1.7 px respectively on Sintel *clean* and *final*. The sign imbalance is also reduced of 0.4 and 0.8 px respectively on Sintel *clean* and final. The kaleidoscope dataset shows the lowest EPE and sign imbalance reduction, which is around 0.3 px for both the EPE and the sign imbalance.

11.2. FlyingChairs trained models comparison

Table 8 helps evaluating the effects of different β values, when different strategies, FWDs and FWDg, are used to obtain O^* during training. For $\beta = 0$, the baseline and FWDg, FWDs models with input mirroring perform equally, whereas FWDs without mirroring show a considerably higher imbalance and a slightly higher EPE. Increasing β reduces the sign imbalance $\overline{||I||}$ for all models on all testing sets. Values of $\beta \in [0.4, 0.6]$ minimize EPE and $\overline{||I||}$ on all testing sets. Higher values of β further reduce imbal-

Table 7: Baseline models trained and their average EPE and sign imbalance value in pixels.

				Sintel				Kaleido-
				Clean		Final		scope
Model	Mir.	batch	Data	EPE	$\overline{\lVert I \rVert}$	EPE	$\overline{\lVert I \rVert}$	EPE
RAFT	x	8	C	2.19	1.22	4.39	2.10	1.41
RAFT	x	16	C	2.29	1.23	4.51	2.43	1.44
RAFT	x	6	C	2.24	1.11	4.52	1.98	1.49
RAFT	x	8	C	2.25	1.24	4.50	2.14	1.47
mean				2.25	1.20	4.48	2.16	1.45
std				0.04	0.06	0.06	0.19	0.03
RAFT	x	10	T	1.42	0.77	2.75	1.36	1.15
RAFT	x	5	T	1.43	0.79	2.72	1.33	1.08
RAFT	x	5	T	1.38	0.79	2.67	1.36	1.23
mean				1.41	0.78	2.72	1.35	1.15
std				0.03	0.01	0.04	0.02	0.07

ance and EPE on Sintel, but slighlty increase the EPE on the Kaleidoscope dataset. The EPE penalty on the Kaleidoscope dataset is worsen when β increases. However, this is sometimes in contrast with the results on Sintel. A notable example is RAFT-FWDs for $\beta = 1.6$ (without mirroring) which records the lowest EPE and sign imbalance on Sintel, but also records a higher EPE on the Kaleidoscope dataset, up to 0.8 px compared to the baseline. For this reason the models in table 8 showing the highest imbalance mitigation but an EPE increase on Kaleidoscope, and the models with values of beta leading to the lowest EPE independently on the sign imbalance mitigation, have been tested on two additional datasets: Monkaa and KITTI. Table 9 show the results of this evaluation. As it can be noticed, no network showing an EPE penalty in table 8 on Kaleidoscope, shows a penalty on these additional datasets. On the contrary, on this datasets these models perform similarly. Furthermore, models trained with a higher β obtain better performance compared to lower values of β. FWDg for $\beta \in [0.4, 0.6]$ perform worse than the baseline.

To evaluate the difference between allowing the gradient during the second forward propagation (FWDg) or stop it (FWDs) we refer to the models in table 8 labelled with an "x", meaning the training runs use mirroring data augmentation. We note that FWDg and FWDs show similar trends, but FWDg show a slightly higher imbalance reduction, and an higher EPE penalty on Sintel *clean* and Kaleidoscope, for the same β values. Additionaly, table 8 reports EPE and sign imbalance values when RAFT-FWDs is trained with or without mirroring training data augmentation. This should help understanding if mirroring is still beneficial for the sign imbalance and the EPE when applying the auxiliary loss. We note that FWDs without mirroring leads to a considerably lower EPE. This can be observed by comparing the two strategies for values of $\beta = 0.8$ and $\beta = 1.6$, for these beta values, the EPE gain is around 0.6 px when the mirroring data augmentation is off. When the data augmentation is applied, the lowest EPE on Sintel *final* is around 4.07 px, and is reached by $\beta = 0.6$ and $\beta = 1.8$. However the penalty on Sintel *clean* is higher in the latter case. Both networks show a similar penalty on the Kaleidoscope dataset.

We evaluated how different aggregation metrics can be used during training on FlyingChairs. We tested the metrics described in Sec. 4, $\overline{||I||}$ and $\overline{I_{m=2}}$. Given that different aggregation metrics lead to different loss function values, we tested the training on $\overline{||I||}$ for values in the range $\beta \in [0.001, 1]$. As a result, we noticed that the training diverge after around 1000 iterations for all values of β. Differently, when plugging the statistical imbalance $\overline{I_{m=2}}$ during training, different outcomes are observed. Table 10 reports the converging training runs using $\overline{I_{m=2}}$. Overall for all the β values there is an EPE penalty on Sintel *clean* and Kaleidoscope; the EPE on Sintel *final*, instead, is considerably reduced. However, differently from all the other training runs on this chapter, increasing β leads to unexpected outcomes. For $\beta = 0$ and $\beta = 0.2$ there is an higher imbalance reduction is compared to $\beta = 0.4$. Furthermore, $\beta = 0.2$ records the lowest sign imbalance among all the models trained on FlyingChairs, however, this value of beta leads to a very large EPE on Sintel *clean* and Kaleidoscope. $\beta = 0.6$ has also a similar imbalance reduction, but show a considerably lower EPE compared to $\beta = 0.2$.

11.3. FlyingThings trained models comparison

Table 11 compares different RAFT training runs, using the models pretrained on FlyingChairs (table 8), to fine tune

Table 8: Evaluation of RAFT trained FlyingChairs for different aggregation strategies.

			Sintel				Kaleido-scope
			Clean		Final		
	Mir.	β	EPE	$\overline{\|\|I\|\|}$	EPE	$\overline{\|\|I\|\|}$	EPE
baseline	x	-	2.25	1.20	4.48	2.16	1.45
FWDS	-	0	2.22	1.59	4.67	3.42	1.48
	-	0.2	2.22	1.47	4.39	2.45	1.54
	-	0.4	2.18	1.18	4.26	2.24	1.49
	-	0.6	2.21	1.00	4.22	1.86	1.44
	-	0.8	2.16	0.79	3.92	1.52	1.66
	-	1	2.18	0.78	4.07	1.27	1.63
	-	1.2	2.24	0.61	4.06	0.99	1.83
	-	1.4	2.32	0.48	3.88	0.93	2.25
	-	1.6	2.28	0.46	3.77	0.72	2.20
FWDS	x	0	2.29	1.27	4.55	2.09	1.49
	x	0.2	2.22	1.02	4.41	2.33	1.43
	x	0.4	2.30	1.15	4.38	2.02	1.44
	x	0.6	2.14	0.85	4.07	1.61	1.49
	x	0.8	2.31	0.99	4.29	1.68	1.55
	x	1	2.22	0.78	4.29	1.43	1.64
	x	1.2	2.22	0.70	4.29	1.42	1.65
	x	1.4	2.29	0.74	4.19	1.20	1.84
	x	1.6	2.32	0.65	4.16	1.17	2.06
	x	1.8	2.43	0.70	4.09	1.08	2.14
FWDG	x	0	2.22	1.24	4.47	2.05	1.41
	x	0.2	2.18	0.99	4.45	1.99	1.47
	x	0.4	2.21	0.87	4.53	1.64	1.53
	x	0.6	2.23	0.93	4.29	1.47	1.64
	x	0.8	2.21	0.68	4.22	1.49	1.76
	x	1	2.29	0.66	4.11	1.14	1.88
	x	1.2	2.39	0.85	4.25	1.35	2.03
	x	1.4	2.40	0.48	4.23	1.14	2.40

on FlyingThings. The models listed use different strategies, FWDs and FWDg. Referring to table 11 the column labelled with β_C refers the value of β applied during the training on FlyingChairs, and similarly β_T is the β used during the fine tuning on FlyingThings.

Section 11.1 has shown that when fine tuning on FlyingThings, the highest gain is on Sintel *final*, moderate improvements are noticed on Sintel *clean* and a small improvement is observed on the Kaleidoscopoe dataset. Similar patterns are shown in table 11 when the models are fine tuned on FlyingThings3D with $\beta > 0$.

To evaluate what is the contribution of pretraining on FlyingChairs and what is the loss function contribution on FlyingThings, we trained the models pre-trained on FlyingChairs with $\beta \in [0.4, 0.8]$, on FlyingThings, using only the mirroring data augmentation for the fine tuning stage. Beside a slight EPE penalty on Kaleidoscope, the EPE is unchanged. Instead, the sign imbalance is considerably reduced following the usual pattern, higher imbalance reduction on Sintel *final*, moderate reduction on Sintel *clean* and Kaleidoscope.

Given the large search space for β when fine tuning on FlyingThings, the heuristics used to choose which models trained on FlyingChairs to fine tune onFlyingThings3D are three. i) The first rule is picking the model showing the lowest EPE and the highest imbalance mitigation, without showing any EPE penalty on any of the tested datasets. ii) The second rule is picking the model maximizing the sign imbalance mitigation, also allowing small EPE increases compared to the baseline. iii) The third rule is to weigh imbalance and EPE loss equally, $\beta = 1$. These heuristics have been applied to every strategy (FWDs, FWDg with or without mirroring). Finally, for FWDs with mirroring we additionaly evaluated the effects of halving β when fine

Table 9: Results on Monkaa and KITTI for certain networks trained on FlyingChairs showing a slight EPE penalty when tested on Kaleidoscope.

			Monkaa				KITTI	
			clean		final			
	Mir.	β	EPE	$\overline{\vert\vert I\vert\vert}$	EPE	$\overline{\vert\vert I\vert\vert}$	EPE	$\overline{\vert\vert I\vert\vert}$
baseline	x	-	3.53	1.72	4.54	2.21	10.78	4.78
FWDS	-	0.8	3.43	1.59	4.37	1.72	10.28	3.39
	-	1.6	3.24	0.60	3.94	0.79	11.63	1.91
FWDS	x	0.6	3.45	1.36	4.39	1.72	9.57	3.33
	x	0.8	3.53	1.38	4.40	1.65	9.67	3.16
	x	1	3.31	1.09	4.23	1.34	10.09	3.43
FWDG	x	0.6	3.40	1.10	4.36	1.34	9.98	3.20
	x	1	3.29	0.80	4.18	1.12	10.49	2.81
	x	1.4	3.33	0.82	4.12	0.88	11.37	2.19

Table 10: Different aggregation metrics when training on FlyingChairs. * means RAFT has been trained on the statistical imbalance for $m = 2$.

			Sintel				Kaleido-scope
			clean		final		
	Mir.	β	EPE	I	EPE	I	EPE
baseline	x	-	2.25	1.20	4.48	2.16	1.45
FWDs*	x	0.00	2.44	0.66	4.13	1.28	2.34
	x	0.20	3.22	0.41	4.41	0.54	4.67
	x	0.40	2.24	1.17	4.37	2.00	1.47
	x	0.60	2.74	0.54	4.21	0.75	3.53

tuning on FlyingThings3D (compared to its value on FlyingChairs). We also evaluated the effect of increasing β during the fine tuning stage. The β values chosen can be found in table 11.

By comparing all networks in table 8 and 11 we note that fine tuning on FlyingThings3D shrinks the network performance difference. Roughly, the per-strategy EPE standard deviation of the network fine tuned, is always lower than 0.12 px, and the sign imbalance standard deviation is maximum 0.16 px (recorded on Sintel *final* for FWDg). Thus, the performance gap between different values of beta is squeezed if compared to training only on FlyingChairs. We can compare the difference between stopping the gradient and allowing the gradient on both forward propagations by evaluating models trained with $\beta_T = 1$ in table 11. The training runs stopping the gradient, FWDs, (with or without mirroring) perform very similarly, with the highest difference being of 0.11 px on the Kaleidoscope EPE. Letting the gradient flow during the second forward propagation, (FWDg), for $\beta = 1$, show the highest imbalance mitigation on all the testing sets. For the same β, the difference between FWDs and FWDg on Sintel *final* and Kaleidoscope is minimal (< 0.1 px). However, on Sintel *clean* the sign imbalance is significatively reduced of more than 50% the baseline value, but at the expense of an EPE increase of 0.2 px ($\approx 15\%$ increase) compared to FWDs.

Moreover, by comparing the models with gradient stopped for $\beta > 1$ we note that the sign imbalance is significantly reduced on Sintel *final* only for very large values of β, ($\beta = 1.6$). However, similarly to FWDg for $\beta = 1$ the higher imbalance mitigation leads to an EPE penalty on Sintel *clean* and Kaleidoscope. The models showing this penalty have been tested on Monkaa and KITTI and do not show an EPE penalty, beside for FWDs without mirroring for $\beta = 1.6$. The results are shown in table 12. Secondly, we note that pretraining on FlyingChairs with $\beta_C = 1$ and fine tuning with $\beta_T = 1.6$, do not lead to a much different performance if compared to fixing beta to 1.2 or 1.6. On the other hand, by halving β to 0.5 when fine tuning we notice an EPE reduction, but roughly a 0.2 px imbalance increase. Furthermore, when beta is halved, the performance are very similar to fine tuning models previously trained on FlyingChairs on FlyingThings, by only applying mirroring data augmentation. This means that for the mentioned model (FWDs with $\beta_C = 1.0, \beta_T = 0.5$) there is not a significant gain compared to fine tuning on FlyingTh-

Table 11: Results on FlyingThings3D for different RAFT training runs. A noticeable sign imbalance reduction is obtained when fine tuning for $\beta \geq 1$

		C	T	Sintel				Kaleido-
				clean		final		scope
	Mir.	β_C	β_T	EPE	$\overline{\|\|I\|\|}$	EPE	$\overline{\|\|I\|\|}$	EPE
baseline	x	-	-	1.41	0.78	2.72	1.35	1.15
	x	0.4	-	1.45	0.62	2.69	1.04	1.24
	x	0.6	-	1.44	0.58	2.83	0.91	1.27
	x	0.8	-	1.43	0.51	2.69	0.85	1.27
	x	1.40	-	1.48	0.74	2.67	1.32	1.18
FWDS	x	0.6	0.6	1.49	0.56	2.68	0.90	1.26
	x	1	0.5	1.44	0.60	2.66	0.91	1.24
	x	1	1	1.49	0.47	2.71	0.69	1.29
	x	1	1.6	1.54	0.42	2.76	0.74	1.40
	x	1.2	1.2	1.45	0.55	2.72	0.67	1.36
	x	1.6	1.6	1.53	0.49	2.77	0.54	1.51
FWDS	-	0.6	0.6	1.50	0.44	2.87	1.04	1.24
	-	0.8	0.8	1.56	0.51	2.72	0.81	1.35
	-	1	1	1.51	0.53	2.75	0.69	1.38
	-	1.6	1.6	1.68	0.41	2.81	0.48	1.64
FWDG	x	0.4	0.4	1.45	0.49	2.68	0.82	1.27
	x	1	1	1.63	0.34	2.75	0.60	1.39

Table 12: Testing on Monkaa and Kitti. The models tested are the ones trained on FlyingThings3D, showing a large imbalance mitigation but a small EPE penalty on Kaleidoscope. Almost all models trained with high β do not show an EPE penalty on Monkaa and KITTI.

		C	T	Monkaa				KITTI	
				clean		final		Clean	
	Mir.	β_C	β_T	EPE	$\overline{\|\|I\|\|}$	EPE	$\overline{\|\|I\|\|}$	EPE	$\overline{\|\|I\|\|}$
baseline	x	-	-	2.35	0.83	3.10	0.93	4.96	1.83
FWDS	x	1.00	1.00	2.41	0.47	3.10	0.57	5.06	1.10
	x	1.60	1.60	2.52	0.40	3.19	0.52	5.67	1.03
FWDG	x	1.00	1.00	2.45	0.36	3.13	0.49	5.67	0.91

ings3D without the additional loss function.

To sum up, when fine tuning the models on FlyingThings, there is not a large performance difference between different values of beta, if compared to the results on FlyingChairs. Moreover, the models trained with by letting the gradient propagate, further reduce the sign imbalance, compared to the models trained by stopping the gradient. In fact, the best model for imbalance mitigation is FWDg for $\beta = 1$ which can reduce the sign imbalance more than two times if compared to the baseline, on all the testing datasets.

11.4. Fine tuning comparison

Based on the considerations of Sec. 11.2 and 11.3 we fine tuned raft on Sintel and KITTI for $\beta = 1$. If we observe an EPE penalty, we halve β during the last training stage. Table 13 show the results of fine tuning FWDs on Sintel and KITTI. Fine tuning on Sintel considerably reduces the sign imbalance with a slight EPE penalty. Halving beta during fine tuning limits the sign imbalance mitigation, but do not show an EPE penalty on any datasets. Fine tuning on KITTI with $\beta = 1$ dramatically reduces the sign imbalance. As it can be noticed the sign imbalance goes from almost 6 px to just 1.66 px on Sintel *final*, similar results are also observed on the *clean* pass. The loss function also benefit the EPE on all datasets, beside on Kaleidoscope.

Table 13: Results of fine tuning RAFT-FWDs on Sintel and on KITTI. Fine tuning on KITTI for $\beta = 1$ drastically reduce the sign imbalance.

						Sintel				Kaleido-
		C	T	S	K	clean		final		scope
Fine Tune	Mir.	β_C	β_T	β_S	β_K	EPE	$\overline{\lvert\lvert I \rvert\rvert}$	EPE	$\overline{\lvert\lvert I \rvert\rvert}$	EPE
	x	-	-	-	-	0.83	0.55	1.36	0.73	1.14
Sintel	x	1	1	0.5	-	0.86	0.39	1.39	0.55	1.18
	x	1	1	1	-	0.92	0.31	1.48	0.45	1.31
	x	-	-	-	-	3.95	4.03	5.60	5.93	3.99
KITTI	x	1	1	1	0.5	4.15	3.73	5.86	5.18	3.94
	x	1	1	1	1	4.07	1.37	4.94	1.66	5.00