Supplementary Material

Flow Fields: Dense Correspondence Fields for Highly Accurate Large Displacement Optical Flow Estimation

1. Introduction

This supplementary material document is only intended for readers that have read the article "Flow Fields: Dense Correspondence Fields for Highly Accurate Large Displacement Optical Flow Estimation", as we assume the notations terms and experiments introduced/presented in the article to be known. First, we present the parameters determined for the public results of our approach in Section 2. In Section 3 we describe why we did not incorporate the matching error for outlier filtering. In Section 4 we provide guidelines for parameter selection. This section was created for our conference approach [1], but should also be mostly valid for our improved approach. In Section 5 we evaluate the variance of our approach due to random search. In Section 6 we describe inaccuracies of the ∼Flow Fields curve. In Section 7 we test our conference approach [1] with different data terms.

Figure 7 shows an example for an evolving flow field. Figure 7 in the paper shows snippets from here. In Figure 1 we show what happens if the experiment presented in Figure 8 a) in the paper is only performed with one sample as initialization (see figure caption).

2. Parameters

We found the following optimal parameters for the test sets:

- MPI-Sintel: $\epsilon = 5$, $e = 4$, $s = 50$,
- Middlebury: $\epsilon = 1$, $e = 7$, $s = 50$
- KITTI 2012: $\epsilon = 1$, $e = 7$ (*Flow Fields+*), $e = 9$ (*Flow Fields*), $s = 150$
- KITTI 2015 $\epsilon = 1.5$, $e = 7$, $s = 100$

3. Using matching error for outlier filtering

In this section we describe, why we did not use the matching error for outlier filtering. As far as we know there is no study so far that evaluates if it makes sense to combine consistency checks and matching errors. As can be seen in Figure 6, the matching error is a much weaker measure for finding outliers than the consistency check. Nevertheless, there is some tendency that a smaller matching error leads to fewer outliers – at least in some range. However, there is a high variability in this tendency. On the clean set of MPI-Sintel the smaller matching error leads to less outliers from an error of 20 up to around 300. In contrast, on the final set this rule is reliable from around 10 to 100, while there is much more gain in this range. We tried to bring these different requirements of clean and final together to define a variable consistency check filter threshold ϵ_{E_d} that depends on the matching error. However, except from being extremely effortful the gain is very limited even if the training sequence is used for testing. When splitting into training and test sequence the quality might even be less, due to overfitting. As a result, we find that it is not worth to consider the matching error if a much more powerful consistency check measure is available.

4. Parameter Selection

Here we describe the effects of our parameters in more detail and provide guidelines for parameter selection. This section is bases on our conference approach [1] and was barely touched for our journal extension. Still most statements should still be true. Not all statements in this section are theoretically or experimentally evaluated. Some statements are assumptions of the authors due to their experience and expertise.

A larger patch size r usually improves (to some extend) the matching robustness, but also leads to more loss of detail. Thus, the optimal r can be a tradeoff between reasonable robustness and reasonable loss of detail. A novel property of our approach is that more robustness cannot only be achieved with a larger patch size r, but also with a larger k. Both robustness factors complement each other. r is important for robust patch comparison (which is still the foundation of our approach), while k allows it due to the blur and the scaled matching to increase the initial patch radius even much further (to $k \times r$) without loss of most details

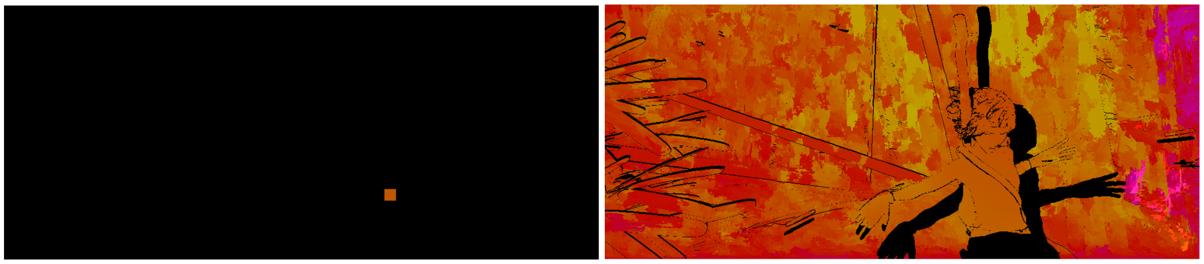
Figure 1. The figure shows what happens if the example in Figure 5 a) in the paper is only initialized with one seed point instead of two. The correct flow outside of the person cannot be found as it is out of range of the random walk.

(in contrast to an enlargement of r). Especially, connected details that are part of a larger body with similar flow are hardly negatively affected by a larger k (e.g. a nose on a head, but also an arm at a body if the arm has not a too strong movement compared to the body). Mainly small fast moving objects[1] suffer form a larger k, although the negative effect is still quite small up to some k ($k \approx 3$ for small objects on MPI-Sintel, see paper) so that the positive effect of more robustness prevails.

Summarized: basic robustness is provided by r. k provides extra robustness on top with much less loss of detail, but it cannot replace r as matching patches with radius r is still the foundation of our approach. If independent objects with fast moment compared to their size matter then k is also a tradeoff between robustness and loss of detail. Otherwise, k is only limited by the image size, although the robustness gain might already get negligible small beforehand. For very large k a kd-tree initialization is unnecessary – a zero initialization can be used instead.

Smaller l decrease similar to larger k the amount of initial resistant outliers. However, only with scales k the outlier sieves can be used. Furthermore, it seems (we did not evaluate it deeply) that determining samples on less positions leads even without scales to better results. This might or might not be (partly) due to collisions of resistant outliers. Lets assume the following scenarios:

1. $k_1 = 0, l_1 = 1$
2. $k_2 = 3, l_2 = 8$.

In both scenarios the same amount of kd-tree samples is created. In scenario 1 all resistant outliers are keep, while in scenario 2 only one resistant outlier per pixel can be kept if more than one is found at a pixel. This leads in total to less resistant outliers. In our paper we simply use $l = 8$ as it performs good and as it was used by [5], which increases comparability.

r_2 should be set only slightly smaller than r to widely preserve the robustness of r, while it should be set different enough to show a different behavior. In our tests the pair $r = 8$ and $r_2 = 6$ performed slightly better than $r = 8$ and $r_2 = 7$. For smaller r it is better to use $r_2 = r - 1$. Different behavior can also be archived by choosing $S \neq S_2$ for SIFT flow. As r_2 is smaller it is obvious that we choose S_2 larger. A larger S_2 improves robustness, which is desirable as the smaller patch radius r_2 decreases robustness (we want to have different behavior and not less robustness). Note that we set S and S_2 to achieve a similar runtime to the census transform. In our tests the SIFT features used for SIFT flow are OpenCV 2.4 SIFT features with a key point size of 0.5 (see OpenCV documentation). (Note: SIFT features in journal extension use a traditional key point size of 1). Larger e, s and smaller ϵ lead to more strict outlier filtering.

[1] Fast moving compared to their size

5. Variance due to the random component

Due to random search our approach has a random component. In this section we evaluate the deviation of matches in EPE from each other due to random search. The evaluation is performed on the MPI-Sintel dataset. For every frame we run FlowFields+ 10 times. This gives us 10 samples for each pixel and allows to determine a mean to determine deviations from that mean. The EPE deviation from the mean is plotted in Figure 2. Furthermore, average deviation, variance, maximum deviation and the probability that a point is an outlier ($> 3px$ from mean) are given in Table 1.

In Figure 2 we use a bin size of 0.01. The total amount is normalized to 1. This means that for around 60% of all pixels the deviation is < 0.005 pixels (bin from -0.005 to $+0.005$). As random search is randomized some points do not convert in time. This is probably why neighboring bins to the inner most bin also contain a large part of the matches.

Only around 2.1% of matches can be considered as real outliers (see Table 1). Most of these outliers occur in occluded regions. Here 12% are outliers while only around 8% are in the inner bin (-0.005 to $+0.005$) as can be seen in Figure 2. This shows that the derivation mainly happens in occluded pixels.

However, we think that other effects that also make pixels unmatchable like textureless regions also causes larger derivations. To test our claim we eliminate unmatchable

Method	Mean difference	σ	Max difference	outliers ($> 3px$ from μ)
Matchable (see caption)	0.029	0.57	209	0.18%
Non occluded	0.160	1.99	366	1.4%
All	0.263	3.32	532	2.1%
Occluded	1.922	10.26	532	11.7%

Table 1. Column 2 to 4 are mean difference, standard deviation and maximum difference of EPE of a pixel when running Flow Fields 10 times on the same frame. The last one is the probability that a match deviates by more than 3 pixels from the mean match of that pixel. Evaluated on MPI-Sintel. "Matchable" is non occluded where the at least 2 of 10 matches have an EPE below 3 pixels. This indicates that the pixel is somehow matchable as at least 2 of 10 matches are correctly matched.

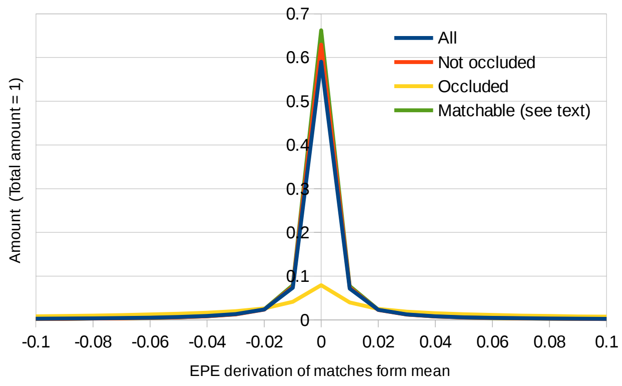

Figure 2. Derivation of matches from mean. See text.

pixels by defining a pixel as matchable if the EPE of at least 2 of the 10 matches is below 3 pixels. This indicates that the pixel can at least sometimes be correctly matched. As seen in Table 1 the outlier rate for matchable pixels decreases to only 0.18%, which is again nearly ten times lower than for non occluded regions. This shows that derivations strongly correlate with the inability to match a pixel.

The mean difference and standard derivation in Table 1 are larger than one might expect from Figure 2. This is due to the fact that outliers with large EPE difference have a large impact on these values. As the max difference column shows there are single outliers with very large difference.

5.1. Dis(advantage) due to derivation

We think that a derivation like in our approach is an advantage, as it can be exploited to detect unreliable regions. This is for instance exploited by our outlier filter. One can argue that there is also an disadvantage as our approach is inconsistent i.e. if we run our approach more than once on a frame it does not give the same output. While this is true it is easy to fix. We simply can initialize the random number generator in every frame with the same seed value. Then our approach gives also perfectly consistent results.

6. Inaccuracies in ∼Flow Fields curve

Here we discuss some of the inaccuracies in the ∼Flow Fields curve. First of all, we assume for the cure that random search always succeeds in reintroducing outliers in its search range. However, this is not the case. Figure 3 demonstrates illustrative that it is actually likely that random search fails in doing so. This means that the ∼Flow Fields curve is based on the factual worst case for the random search operation.

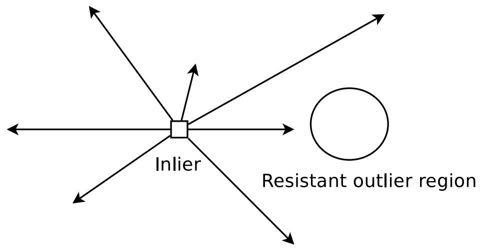

Figure 3. Random search attempts can easily fail in finding a resistant outlier region.

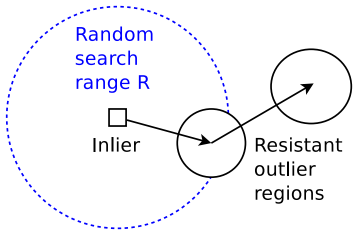

Figure 4. Random search attempts can go above random search range $\mathcal{R}$ if there is more than one outlier region.

Using the worst case is justified as this automatically means that the filter effect that works for the worst case also works for the real Flow Fields – potentially even better.

We could also simulate random search. However, simulating it independently would probably draw a too positive picture as propagation is not considered, that can support random search in gradient decent. Simulating propagation is out of scope as it requires to leave the context of a single pixel and simulate the whole Flow Fields approach. At this level even the Kd-tree initialization is relevant, although it is not part of the outlier sieve effect. Conclusion: using the worst case avoids very difficult but irrelevant questions, as our observations also work with the worst case.

In our model we just consider the outlier and the inlier. However, the combination of several random search oper-

Feature/Method	r	r_2	ϵ	Epic	Epic noc.
Census transform	8	6	5	4.03	2.04
SIFT flow	5	4	0.8	4.14	2.22
EpicFlow [6]	-	-	-	4.34	2.48

Table 2. Results on the Sintel training dataset (for simplicity and comparability we use the same subset as in the paper). We use $s = 50$ for both features and $S = 6$ and $S_2 = 10$ for SIFT flow (S and S_2 are runtime tradeoffs to obtain a runtime that is similar to the Census transform). Unmentioned parameters are set to their standard value mentioned in the paper.

Feature/Method	r	r_2	e	Epic
Census transform	8	6	7	0.239
SIFT flow	6	5	9	0.248
EpicFlow [6]	-	-	-	0.380

Table 3. Results on the Middlebury training dataset. We use $s = 50$ for both features. Unmentioned parameters are set to their standard value mentioned in the paper (i.e. $S = 3$).

ations on a scale and further not considered outliers allow it to exceed the limit $\mathcal{R}$ (or $\mathcal{R}^+$) to some extend as shown in Figure 4. We think that this is a minor effect. First of all there is still a absolute limit for random search of $c\mathcal{R}$ with c being the number of random search operations i.e. for large distances the effect is still guaranteed. Secondly, it is very unlikely to get even close to that limit. While a random walk with random distance 0...1 ends up on median 0.5 pixels from the origin, 4 such random walks in a row only end up on median 0.78 pixels from the origin. The reason for this small difference is that a random walk can also go backwards towards the origin. In the unlikely case that a random walk goes only in one direction it also has to end up in positions with always decreasing matching error to get accepted. The fact that more random search operations improve the optical flow in our tests also supports that this effect is minor – at least it harms less that the benefits obtained from more random search operations.

7. Additional experiments

In this section we present our results with our conference approach *Flow Fields* with the SIFT flow data term on MPI-Sintel [3] and Middelbury [2] as well as with the census transform data term on KITTI [4]. This section was mainly intended as supplementary material for our conference approach [1], but for completeness we also keep it as supplementary material for our extended journal article. In all result tables that are presented in this section we mark our Flow Field approach (with data terms mentioned in the tables) blue and the original EpicFlow [6] approach red.

As can be seen in Table 2 and 3, we can clearly outperform the original EpicFlow with our SIFT flow data term on MPI-Sintel and Middlebury, but less than with the census transform. As a result, our Flow Fields + Epic with the SIFT flow data term outperform the original EpicFlow approach on all three tested datasets i.e. our Flow Fields with SIFT flow are in general superior to Deep Matching descriptors [7] if EpicFlow is applied. Note that SIFT flow in general requires a smaller patch radius r than the census transform (see tables), as SIFT flow pixels consider not only the pixel color itself but also the surrounding of the pixel. Despite the good results census transform, still performs better on MPI-Sintel and Middlebury.

Feature/Method	>3 pixel nocc.	>3 pixel all	EPE nocc.	EPE all
SIFT flow	5.23 %	12.58 %	1.27 px	2.94 px
EpicFlow [6]	7.49 %	16.75 %	1.38 px	3.48 px
Census transform	11.38 %	19.70 %	2.18 px	4.55 px

Table 4. Results on KITTI training set. nocc. means non-occluded. >3 pixel means an endpoint error above 3 pixel. We use $r = 5, r_2 = 4, \epsilon = 5$, $e = 8$ and $s = 100$ for the census transform. All other parameters are set to their standard value mentioned in the paper. For SIFT flow we use the parameters used on the test set. Both our results are for their respective circumstances very good. See text and Figure 5 for a description of the challenging circumstances we have to deal with on KITTI.

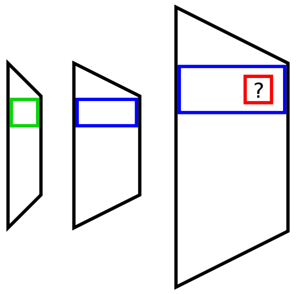

Figure 5. An example of the deformations (blue) an image patch (green) can undergo on a wall (black) in KITTI. Left: the original patch. Middle: With angular deformation only. Right: with angular and scale deformation (a common case on KITTI). A unmodified patch based approach like ours can only match the green patch to the red patch or a moved (but not deformed) version of it. It is clear that this cannot work very well, as the correct patch (blue) that would match the green patch is strongly deformed compared to the red patch. Considering this fact our results on KITTI are very good.

On KITTI (Table 4) the census transform does not perform that well. As mentioned in the paper this is probably because (unmodified) patch based approaches are not suited for datasets like KITTI where image patches of walls and the street can undergo strong scale changes and deformations (See Figure 5). Nevertheless, we can obtain very good results with the census transform considering the challenging circumstances. The problem in Figure 5 also applies to our SIFTFlow data term. However, as SIFT (and SIFT flow as well) is to some extend robust to deformation it is possi-

	Matching Error																			
Filter distance ε	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90	90–100	100–110	110–120	120–130	130–140	140–150	150–160	160–170	170–180	180–190	190–200
0.0 – 0.2	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
0.2 – 0.4	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	1%	1%
0.4 – 0.6	2%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%
0.6 – 0.8	4%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	2%	2%	2%	2%	2%	2%	2%	2%
0.8 – 1.0	7%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	3%	3%	3%	3%	3%	3%
1.0 – 1.2	7%	3%	2%	2%	2%	2%	2%	2%	3%	3%	3%	3%	3%	3%	3%	4%	4%	4%	4%	4%
1.2 – 1.4	17%	4%	3%	3%	3%	3%	3%	4%	4%	4%	4%	4%	4%	5%	5%	5%	5%	5%	6%	6%
1.4 – 1.6	12%	4%	4%	4%	4%	4%	4%	4%	5%	5%	5%	5%	5%	6%	6%	6%	6%	7%	7%	7%
1.6 – 1.8	15%	5%	5%	5%	5%	5%	5%	6%	6%	6%	7%	7%	7%	7%	8%	8%	8%	8%	9%	9%
1.8 – 2.0	16%	6%	6%	6%	6%	6%	7%	7%	7%	7%	8%	8%	8%	9%	9%	9%	10%	10%	11%	11%
2.0 – 2.2	41%	8%	7%	7%	7%	8%	8%	8%	8%	9%	9%	9%	10%	10%	11%	11%	12%	12%	12%	13%
2.2 – 2.4	48%	10%	8%	8%	9%	9%	9%	9%	10%	10%	11%	11%	11%	12%	12%	13%	13%	14%	14%	15%
2.4 – 2.6	38%	11%	10%	10%	10%	11%	11%	11%	12%	12%	13%	13%	13%	14%	14%	15%	15%	16%	16%	17%
2.6 – 2.8	24%	14%	12%	11%	12%	12%	12%	13%	13%	14%	14%	15%	15%	15%	16%	16%	17%	18%	18%	19%
2.8 – 3.0	26%	17%	14%	14%	14%	14%	14%	14%	15%	15%	16%	16%	16%	17%	17%	18%	18%	19%	20%	20%
3.0 – 3.2	24%	16%	16%	15%	14%	15%	16%	16%	16%	17%	18%	18%	18%	19%	19%	20%	20%	21%	21%	22%
3.2 – 3.4	53%	21%	17%	17%	18%	17%	18%	18%	18%	19%	20%	20%	21%	21%	21%	22%	22%	23%	24%	24%
3.4 – 3.6	38%	18%	18%	17%	18%	19%	20%	20%	20%	21%	21%	22%	22%	23%	23%	24%	24%	25%	25%	26%
3.6 – 3.8	38%	21%	22%	21%	22%	22%	22%	22%	23%	23%	23%	24%	24%	25%	25%	25%	26%	27%	27%	28%
3.8 – 4.0	33%	24%	22%	22%	22%	24%	23%	24%	24%	25%	26%	27%	26%	27%	27%	28%	28%	29%	29%	29%
4.0 – 4.2	43%	28%	25%	26%	25%	27%	25%	27%	27%	28%	28%	29%	29%	29%	30%	30%	31%	31%	32%	32%
4.2 – 4.4	46%	28%	27%	27%	26%	30%	29%	29%	30%	31%	32%	32%	32%	33%	33%	33%	34%	34%	35%	35%
4.4 – 4.6	30%	31%	29%	29%	31%	33%	33%	34%	34%	34%	35%	35%	36%	36%	36%	37%	37%	38%	38%	39%
4.6 – 4.8	52%	35%	37%	36%	37%	38%	37%	38%	38%	40%	41%	40%	40%	41%	41%	41%	41%	42%	43%	43%
4.8 – 5.0	69%	43%	39%	41%	42%	43%	41%	43%	43%	43%	44%	45%	46%	46%	47%	47%	47%	47%	48%	49%
5.0 – 5.2	60%	48%	46%	46%	48%	49%	50%	50%	50%	51%	51%	51%	52%	52%	52%	52%	53%	54%	54%	55%
5.2 – 5.4	49%	47%	48%	49%	52%	53%	52%	53%	54%	54%	55%	56%	57%	56%	57%	57%	58%	58%	59%	60%
5.4 – 5.6	75%	55%	52%	55%	55%	56%	58%	57%	57%	58%	60%	58%	59%	59%	60%	60%	60%	60%	62%	62%
5.6 – 5.8	54%	56%	54%	56%	58%	58%	57%	59%	59%	60%	60%	61%	61%	63%	62%	62%	63%	63%	64%	64%
5.8 – 6.0	61%	54%	57%	60%	60%	62%	62%	62%	62%	63%	63%	64%	63%	64%	65%	64%	65%	65%	66%	66%
Frequency	2.9%	3.5%	5.0%	5.8%	6.0%	6.1%	5.9%	5.7%	5.5%	5.2%	4.9%	4.6%	4.3%	4.0%	3.7%	3.4%	3.1%	2.8%	2.5%	2.3%

	Matching Error																				
Filter distance ε	200–210	210–220	220–230	230–240	240–250	250–260	260–270	270–280	280–290	290–300	300–310	310–320	320–330	330–340	340–350	350–360	360–370	370–380	380–390	390–400	Frequnecy
0.0 – 0.2	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	35.2%
0.2 – 0.4	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	26.4%
0.4 – 0.6	1%	1%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	1%	2%	1%	1%	1%	1%	1%	1%	13.5%
0.6 – 0.8	2%	2%	2%	3%	3%	3%	3%	3%	3%	3%	2%	3%	2%	2%	2%	2%	2%	2%	2%	2%	7.6%
0.8 – 1.0	3%	3%	4%	4%	4%	4%	4%	4%	4%	4%	4%	3%	3%	3%	3%	3%	3%	3%	3%	3%	4.7%
1.0 – 1.2	4%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	4%	4%	4%	4%	4%	4%	3.1%
1.2 – 1.4	6%	6%	6%	7%	7%	7%	7%	7%	7%	7%	7%	7%	7%	7%	7%	6%	6%	6%	6%	5%	1.9%
1.4 – 1.6	8%	8%	8%	8%	9%	9%	9%	9%	9%	9%	9%	9%	9%	8%	8%	8%	8%	8%	7%	7%	1.4%
1.6 – 1.8	9%	10%	10%	10%	11%	11%	11%	11%	11%	11%	11%	11%	11%	11%	11%	11%	10%	10%	10%	10%	1.0%
1.8 – 2.0	12%	12%	12%	12%	13%	13%	13%	13%	13%	14%	14%	14%	13%	13%	13%	13%	12%	13%	12%	11%	0.8%
2.0 – 2.2	13%	14%	14%	15%	15%	15%	15%	16%	15%	15%	16%	16%	15%	15%	15%	15%	14%	14%	13%	14%	0.6%
2.2 – 2.4	16%	16%	16%	17%	17%	18%	18%	18%	18%	18%	18%	18%	18%	18%	19%	18%	16%	16%	15%	13%	0.5%
2.4 – 2.6	17%	18%	19%	19%	19%	19%	20%	20%	20%	21%	21%	21%	20%	20%	19%	20%	19%	17%	19%	15%	0.4%
2.6 – 2.8	20%	20%	20%	21%	22%	22%	22%	23%	23%	23%	23%	23%	23%	22%	21%	22%	20%	20%	19%	17%	0.3%
2.8 – 3.0	21%	21%	22%	23%	23%	24%	24%	25%	24%	25%	25%	25%	25%	24%	24%	23%	23%	21%	20%	20%	0.3%
3.0 – 3.2	22%	24%	24%	25%	25%	25%	26%	26%	26%	27%	26%	26%	26%	26%	25%	25%	24%	23%	20%	21%	0.3%
3.2 – 3.4	25%	25%	25%	26%	27%	27%	28%	28%	28%	28%	28%	28%	28%	28%	27%	27%	25%	24%	23%	21%	0.2%
3.4 – 3.6	26%	27%	28%	28%	29%	29%	30%	30%	30%	30%	30%	30%	30%	30%	29%	28%	26%	26%	24%	21%	0.2%
3.6 – 3.8	29%	29%	30%	30%	30%	31%	31%	32%	31%	32%	32%	31%	31%	31%	31%	29%	29%	27%	25%	24%	0.2%
3.8 – 4.0	31%	31%	32%	33%	33%	33%	34%	34%	34%	34%	34%	33%	33%	33%	33%	31%	30%	28%	27%	27%	0.2%
4.0 – 4.2	33%	34%	34%	35%	35%	35%	36%	36%	36%	37%	36%	35%	36%	36%	34%	33%	33%	31%	27%	27%	0.2%
4.2 – 4.4	36%	36%	37%	37%	38%	38%	38%	38%	39%	39%	38%	38%	38%	38%	36%	34%	32%	34%	32%	29%	0.1%
4.4 – 4.6	40%	40%	40%	40%	41%	41%	42%	42%	42%	42%	42%	42%	41%	41%	39%	39%	36%	35%	34%	33%	0.1%
4.6 – 4.8	43%	44%	45%	45%	45%	45%	47%	47%	46%	47%	46%	46%	46%	47%	44%	45%	43%	42%	40%	40%	0.1%
4.8 – 5.0	50%	50%	51%	51%	52%	52%	52%	52%	52%	53%	53%	53%	53%	54%	54%	52%	54%	52%	53%	52%	0.1%
5.0 – 5.2	55%	56%	57%	57%	57%	58%	58%	60%	60%	60%	60%	62%	62%	61%	62%	63%	63%	64%	64%	64%	0.1%
5.2 – 5.4	59%	60%	61%	61%	62%	62%	62%	64%	64%	64%	66%	65%	66%	67%	69%	69%	70%	70%	73%	74%	0.1%
5.4 – 5.6	63%	62%	63%	64%	65%	66%	66%	66%	67%	68%	68%	68%	69%	71%	72%	72%	73%	75%	77%	76%	0.1%
5.6 – 5.8	65%	65%	65%	66%	66%	67%	67%	69%	69%	69%	70%	71%	72%	73%	73%	76%	75%	76%	79%	81%	0.1%
5.8 – 6.0	66%	66%	66%	68%	69%	69%	70%	70%	70%	70%	72%	72%	72%	73%	75%	76%	77%	78%	81%	79%	0.1%
Frequency	2.0%	1.8%	1.6%	1.4%	1.2%	1.0%	0.9%	0.7%	0.6%	0.5%	0.4%	0.3%	0.2%	0.2%	0.1%	0.1%	0.1%	0.1%	0.0%	0.0%	

(a) The outlier probabilities for clean

	Matching Error																			
Filter distance ε	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90	90–100	100–110	110–120	120–130	130–140	140–150	150–160	160–170	170–180	180–190	190–200
0.0 – 0.2	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
0.2 – 0.4	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%
0.4 – 0.6	2%	2%	2%	1%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%
0.6 – 0.8	3%	3%	3%	2%	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%
0.8 – 1.0	4%	4%	3%	3%	4%	4%	4%	4%	4%	4%	4%	4%	4%	4%	4%	4%	4%	4%	4%	4%
1.0 – 1.2	5%	5%	5%	4%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%
1.2 – 1.4	5%	6%	6%	6%	7%	7%	7%	7%	7%	7%	7%	7%	7%	7%	7%	7%	7%	7%	7%	7%
1.4 – 1.6	8%	7%	7%	7%	8%	8%	8%	8%	8%	9%	9%	9%	8%	8%	8%	8%	8%	8%	9%	9%
1.6 – 1.8	8%	8%	9%	9%	10%	10%	10%	10%	11%	10%	10%	11%	10%	10%	10%	10%	10%	10%	10%	10%
1.8 – 2.0	7%	10%	10%	11%	11%	12%	12%	12%	12%	12%	12%	12%	12%	12%	12%	12%	12%	12%	12%	12%
2.0 – 2.2	8%	10%	12%	12%	13%	13%	14%	14%	14%	14%	14%	14%	13%	13%	13%	13%	13%	13%	13%	13%
2.2 – 2.4	9%	12%	13%	14%	15%	15%	15%	16%	16%	16%	16%	15%	15%	15%	15%	15%	15%	15%	15%	15%
2.4 – 2.6	9%	13%	14%	14%	17%	17%	17%	18%	18%	18%	18%	18%	17%	17%	17%	17%	17%	17%	17%	17%
2.6 – 2.8	11%	13%	15%	16%	18%	18%	19%	19%	19%	19%	19%	19%	19%	19%	19%	19%	19%	19%	19%	19%
2.8 – 3.0	12%	15%	16%	18%	20%	20%	20%	21%	21%	21%	21%	21%	21%	21%	21%	21%	20%	20%	21%	21%
3.0 – 3.2	14%	16%	17%	19%	21%	21%	22%	23%	23%	22%	22%	22%	22%	22%	22%	22%	22%	22%	22%	22%
3.2 – 3.4	14%	17%	19%	21%	22%	23%	24%	24%	25%	25%	25%	25%	25%	25%	24%	24%	24%	24%	24%	25%
3.4 – 3.6	16%	18%	21%	22%	24%	25%	26%	26%	26%	26%	26%	27%	27%	27%	26%	26%	26%	26%	26%	27%
3.6 – 3.8	19%	21%	22%	24%	25%	26%	27%	28%	28%	28%	28%	29%	29%	29%	28%	28%	28%	28%	29%	29%
3.8 – 4.0	22%	22%	24%	27%	29%	29%	30%	30%	30%	30%	30%	31%	30%	31%	31%	30%	31%	31%	31%	31%
4.0 – 4.2	22%	23%	25%	27%	30%	32%	31%	32%	32%	33%	32%	33%	32%	32%	32%	32%	32%	32%	33%	33%
4.2 – 4.4	21%	26%	28%	30%	32%	34%	34%	34%	35%	35%	36%	35%	35%	36%	35%	35%	35%	35%	35%	36%
4.4 – 4.6	27%	29%	30%	33%	35%	35%	36%	36%	37%	38%	37%	38%	38%	38%	37%	38%	38%	38%	38%	39%
4.6 – 4.8	28%	29%	32%	36%	38%	39%	40%	39%	39%	40%	41%	41%	41%	41%	41%	41%	41%	41%	41%	41%
4.8 – 5.0	36%	32%	35%	39%	41%	42%	42%	43%	44%	44%	44%	44%	44%	44%	43%	44%	44%	44%	45%	45%
5.0 – 5.2	31%	36%	39%	42%	46%	47%	47%	47%	47%	47%	47%	47%	47%	47%	48%	47%	48%	48%	48%	49%
5.2 – 5.4	44%	40%	41%	45%	49%	50%	50%	51%	51%	51%	51%	50%	51%	51%	51%	50%	50%	51%	51%	51%
5.4 – 5.6	44%	41%	44%	49%	52%	53%	53%	53%	52%	52%	53%	53%	53%	52%	53%	53%	53%	54%	54%	55%
5.6 – 5.8	49%	45%	48%	50%	54%	53%	55%	55%	55%	55%	55%	55%	55%	55%	55%	55%	55%	56%	57%	57%
5.8 – 6.0	53%	50%	49%	52%	55%	56%	56%	57%	57%	56%	57%	57%	57%	57%	57%	57%	57%	58%	58%	59%
Frequency	5.3%	5.3%	6.6%	6.6%	6.2%	5.7%	5.3%	5.0%	4.7%	4.4%	4.2%	4.0%	3.8%	3.6%	3.5%	3.2%	3.0%	2.8%	2.5%	2.3%

	Matching Error																				
Filter distance ε	200–210	210–220	220–230	230–240	240–250	250–260	260–270	270–280	280–290	290–300	300–310	310–320	320–330	330–340	340–350	350–360	360–370	370–380	380–390	390–400	Frequnecy
0.0 – 0.2	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	29.7%
0.2 – 0.4	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	21.5%
0.4 – 0.6	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	2%	12.7%
0.6 – 0.8	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%	2%	2%	2%	2%	2%	2%	2%	8.2%
0.8 – 1.0	4%	4%	4%	4%	4%	4%	4%	4%	4%	4%	4%	3%	4%	3%	3%	3%	3%	3%	3%	3%	5.7%
1.0 – 1.2	5%	5%	5%	6%	5%	5%	5%	5%	5%	5%	5%	4%	4%	4%	4%	4%	4%	4%	4%	4%	4.3%
1.2 – 1.4	7%	7%	7%	7%	7%	7%	7%	7%	7%	7%	7%	6%	6%	6%	6%	5%	6%	5%	6%	5%	2.9%
1.4 – 1.6	9%	9%	9%	9%	9%	9%	9%	9%	8%	8%	8%	8%	7%	7%	7%	7%	7%	7%	6%	6%	2.2%
1.6 – 1.8	11%	11%	11%	11%	10%	10%	10%	10%	10%	10%	10%	9%	9%	9%	9%	9%	9%	7%	9%	8%	1.7%
1.8 – 2.0	12%	12%	12%	12%	12%	12%	12%	12%	12%	12%	11%	11%	11%	11%	10%	10%	10%	10%	9%	9%	1.4%
2.0 – 2.2	14%	13%	14%	14%	14%	14%	14%	14%	14%	14%	13%	13%	12%	12%	12%	11%	11%	11%	10%	9%	1.2%
2.2 – 2.4	15%	15%	16%	16%	16%	16%	16%	16%	16%	15%	15%	14%	14%	14%	14%	13%	13%	13%	11%	11%	1.0%
2.4 – 2.6	17%	18%	18%	18%	18%	18%	18%	18%	17%	18%	17%	17%	17%	15%	16%	15%	15%	15%	13%	13%	0.8%
2.6 – 2.8	19%	19%	20%	20%	20%	20%	20%	20%	20%	20%	19%	19%	18%	18%	17%	17%	16%	17%	15%	16%	0.7%
2.8 – 3.0	21%	21%	22%	22%	22%	22%	22%	22%	22%	21%	21%	21%	20%	20%	19%	19%	18%	18%	16%	16%	0.7%
3.0 – 3.2	23%	23%	23%	23%	23%	24%	24%	24%	23%	23%	23%	22%	21%	21%	20%	20%	18%	20%	17%	17%	0.6%
3.2 – 3.4	25%	25%	25%	25%	26%	26%	26%	26%	26%	25%	24%	24%	24%	23%	23%	22%	22%	19%	20%	18%	0.5%
3.4 – 3.6	27%	27%	28%	27%	28%	28%	27%	28%	27%	27%	27%	27%	26%	25%	25%	24%	23%	22%	21%	20%	0.5%
3.6 – 3.8	29%	29%	29%	30%	30%	30%	30%	30%	30%	30%	30%	29%	27%	27%	27%	25%	24%	23%	23%	20%	0.4%
3.8 – 4.0	31%	32%	32%	32%	32%	32%	32%	32%	32%	32%	31%	31%	30%	29%	29%	28%	28%	26%	26%	23%	0.4%
4.0 – 4.2	33%	34%	34%	35%	35%	35%	35%	35%	34%	34%	33%	32%	31%	31%	31%	29%	28%	28%	25%	24%	0.4%
4.2 – 4.4	36%	36%	37%	37%	37%	38%	37%	37%	37%	36%	36%	35%	35%	34%	33%	34%	30%	28%	29%	30%	0.3%
4.4 – 4.6	39%	39%	40%	40%	40%	40%	40%	40%	41%	40%	38%	39%	39%	37%	37%	36%	34%	34%	33%	30%	0.3%
4.6 – 4.8	42%	42%	42%	43%	43%	44%	44%	44%	44%	43%	43%	43%	42%	41%	40%	40%	38%	38%	37%	37%	0.3%
4.8 – 5.0	46%	46%	47%	47%	47%	48%	48%	48%	48%	47%	47%	47%	47%	47%	48%	47%	45%	46%	42%	43%	0.3%
5.0 – 5.2	49%	50%	50%	51%	51%	52%	52%	53%	52%	53%	52%	53%	53%	53%	52%	54%	53%	56%	54%	54%	0.3%
5.2 – 5.4	52%	53%	53%	55%	55%	55%	55%	56%	56%	55%	56%	57%	57%	58%	58%	59%	59%	59%	61%	60%	0.2%
5.4 – 5.6	54%	55%	56%	57%	58%	58%	58%	59%	59%	59%	60%	61%	60%	61%	62%	62%	64%	64%	65%	65%	0.2%
5.6 – 5.8	57%	58%	59%	59%	60%	61%	61%	61%	61%	62%	62%	62%	62%	64%	65%	66%	65%	68%	69%	69%	0.2%
5.8 – 6.0	59%	60%	61%	61%	62%	62%	62%	63%	63%	64%	64%	64%	65%	67%	66%	69%	68%	68%	71%	73%	0.2%
Frequency	2.0%	1.8%	1.5%	1.3%	1.1%	0.9%	0.8%	0.6%	0.5%	0.4%	0.3%	0.2%	0.2%	0.1%	0.1%	0.1%	0.1%	0.0%	0.0%	0.0%	

(b) The outlier probabilities for final

Figure 6. The Figure shows the probability that a point on our Flow Maps is an outlier for different matching errors (column) and different filter thresholds ϵ (row) on the clean and final datasets of MPI-Sintel. We use the standard parameters presented in the paper. This includes a 2x consistency check. The outlier threshold is set to 5 pixels i.e. a point is an outlier if it varies by more than 5 pixels from the ground truth. the maximum possible matching error is $3(2r+1)*(2r+1) = 867$ (3 color channels). However, values greater 400 are negligible.

ble to obtain state-of-the-art results with it – but only if our novel Flow Field approach is used for matching.

References

[1] C. Bailer, B. Taetz, and D. Stricker. Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation. In *ICCV*. IEEE, 2015. 1, 4

[2] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, 2011. 4

[3] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *Computer Vision–ECCV 2012*, pages 611–625. Springer, 2012. 4

[4] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, page 0278364913491297, 2013. 4

[5] K. He and J. Sun. Computing nearest-neighbor fields via propagation-assisted kd-trees. In *Computer Vision and Pat-*

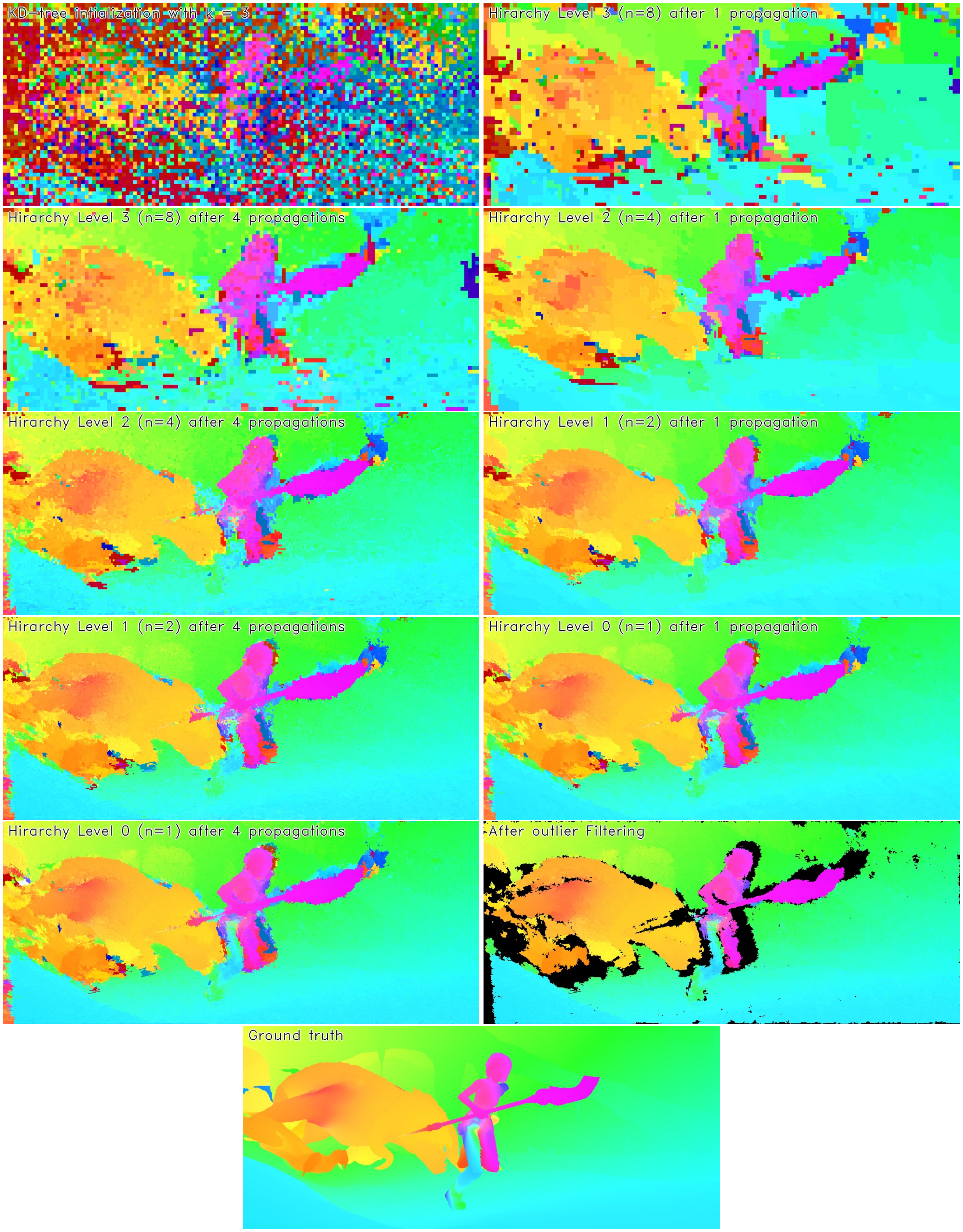

Figure 7. Image sequence shows how a flow field evolves. Figure 6 in paper is a snippet of these figures.

tern Recognition (CVPR), 2012 IEEE Conference on, pages 111–118. IEEE, 2012. 2

[6] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow. 2015. 4

[7] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. DeepFlow: Large displacement optical flow with deep matching. In *IEEE Intenational Conference on Computer Vision (ICCV)*, Sydney, Australia, Dec. 2013. 4