DEGRAF-FLOW: EXTENDING DEGRAF FEATURES FOR ACCURATE AND EFFICIENT SPARSE-TO-DENSE OPTICAL FLOW ESTIMATION

Felix Stephenson[1], *Toby P. Breckon*[1], *Ioannis Katramados*[2]
Durham University, UK[1] | NHL Stenden University of Applied Sciences, Netherlands[2]

ABSTRACT

Modern optical flow methods make use of salient scene feature points detected and matched within the scene as a basis for sparse-to-dense optical flow estimation. Current feature detectors however either give sparse, non uniform point clouds (resulting in flow inaccuracies) or lack the efficiency for frame-rate real-time applications. In this work we use the novel Dense Gradient Based Features (DeGraF) as the input to a sparse-to-dense optical flow scheme. This consists of three stages: 1) efficient detection of uniformly distributed Dense Gradient Based Features (DeGraF) [1]; 2) feature tracking via robust local optical flow [2]; and 3) edge preserving flow interpolation [3] to recover overall dense optical flow. The tunable density and uniformity of DeGraF features yield superior dense optical flow estimation compared to other popular feature detectors within this three stage pipeline. Furthermore, the comparable speed of feature detection also lends itself well to the aim of real-time optical flow recovery. Evaluation on established real-world benchmark datasets show test performance in an autonomous vehicle setting where DeGraF-Flow shows promising results in terms of accuracy with competitive computational efficiency among non-GPU based methods, including a marked increase in speed over the conceptually similar EpicFlow approach [3].

Index Terms— optical flow, Dense Gradient Based Features, DeGraF, automotive vision, feature points

1. INTRODUCTION

Optical flow estimation is the recovery of the motion fields between temporally adjacent images within a sequence. Since its conception over 35 years ago [4, 5] it remains an area of intense interest in computer vision in terms of accurate and efficient computation for numerous applications [6].

Dense optical flow estimation aims to accurately recover per-pixel motion vectors from every pixel in a video frame to the corresponding locations in the subsequent (or previous) image frame in the sequence. These vector fields form the basis for applications such as scene segmentation [7], object detection and tracking [8], structure from motion and visual odometry [9]. The development of autonomous vehicles has revealed the necessity for real-time scene understanding [10]. This has lead to increased pressure to improve both the quality and computational efficiency of dense optical flow estimation.

To date, recent progress in this area has been driven by the introduction of increasingly challenging benchmark datasets [11, 12, 10, 13] providing accurate ground truth optical flow for comparison. Approaches which are robust to scene discontinuities (occlusions, motion boundaries) and appearance changes (illumination, chromacity) have shown strong results under static scene conditions [11]. However, the recent KITTI optical flow benchmarks [10, 13], which comprise video sequences of dynamic real-world urban driving scenarios, presents additional challenges. In particular, significant motion vectors between subsequent video frames due to vehicle velocity through the scene, present a key problem in accurate optical flow estimation [14] for such large displacement vectors.

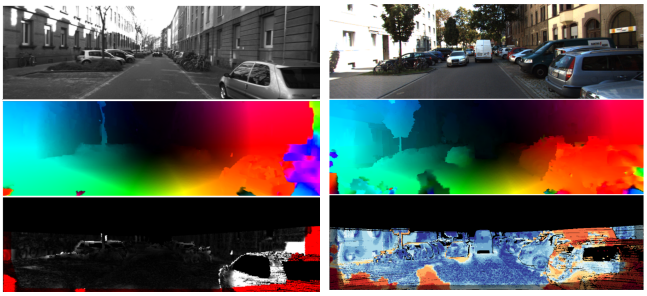

Fig. 1. Dense optical flow results and error maps from KITTI 2012 [10] (left) and KITTI 2015 [13] (right).

To cope with such challenges, contemporary optical flow methods use a sparse-to-dense estimation scheme, whereby a sparse set of points on a video frame are matched to points in the subsequent frame. The sparse set of optical flow vectors recovered from this matching are then used as input to a refinement scheme to recover dense optical flow [14, 15, 16, 3, 17, 18]. Such matching of image points can accurately recover long range motions, however, as the matches are sparse (only cover a fraction of the image) they can lead to loss of accuracy at motion boundaries [3] in the refined dense optical flow. To address this issue, the recent work of [3] (EpicFlow) incorporates a novel state-of-the-art interpolator, Edge Preserving Interpolation of Correspondences (EPIC), which recovers dense flow from sparse matches using edge detection to preserve accuracy at motion boundaries. This shows improved results compared against previous interpolation methods [19] and is currently used as a popular post-processing step in contemporary state-of-the-art dense optical flow estimation methods [20, 21, 22].

Many of these sparse matching techniques are highly accurate but are notably incapable of real-time performance re-

quired for applications such as vehicle autonomy [3, 17, 18, 23]. By contrast, computational efficiency is more readily achievable via sparse point tracking whereby flow estimation only takes place on a fraction of the image. Interestingly, the seminal Lucas-Kanade [5] sparse point tracker proposed over 30 years ago still forms the basis for many contemporary state-of-the-art sparse flow techniques [24, 25, 2]. Robust Local Optical Flow (RLOF) [2] is one such derivative that shows state-of-the-art accuracy on the KITTI benchmark [10, 13]. The most recent work on RLOF [26] demonstrates that combining sparse flow field with interpolators can achieve both efficient and accurate dense optical flow.

All methods that employ sparse feature tracking require a well defined feature set upon which matching can take place. Furthermore, sparse flow vectors with uniform spatial coverage is an ideal for accurate dense optical flow recovery [26] making uniform feature distribution across the scene a key conduit to success. Common feature choices are Harris [27] or FAST key-points [28] due to their relative speed. However these detectors do not guarantee uniform spatial feature distribution as they locate points only on highly textured image regions (e.g corners and edges). To address this issue, a current state of the art sparse flow method, denoted Fast Semi Dense Epipolar Flow (FSDEF) [25], forces uniformity of FAST key-point feature by use of a block-wise selection; this however adversely effects saliency causing increased erroneous matches. Further to this, FAST points do not provide sub-pixel precision so further refinement to the point matches is required to recover accurate optical flow. By contrast, recent work on the use of Dense Gradient Based Features (DeGraF) [1] address these above issues. This approach provides spatially tunable feature density and uniformity in addition to sub-pixel accuracy. DeGraF has comparable speed to FAST and has been shown to be superior in terms of noise and illumination invariance [1].

Motivated by these desirable attributes, our proposed method, DeGraF-Flow, takes a spatially uniform grid of DeGraF feature points as an input to a sparse-to-dense optical flow estimation scheme. Given two temporally adjacent images in a sequence, DeGraF points are detected in the first image and then efficiently tracked to the subsequent image using RLOF [2]. Finally dense optical flow is recovered using the established EPIC interpolation approach [3].

2. APPROACH

Dense optical flow is recovered from two temporally adjacent images (Figure 2A) using a three step process:

Point detection on the first image is carried out by calculation of an even grid of DeGraF points [1] shown in Figure 2B. A sliding window is passed over the image with step size of δ. A key-point is detected within the window at each step as follows.

For an image region I of dimensions $w \times h$ containing grayscale pixels, two centroids, C_{pos} and C_{neg} are defined

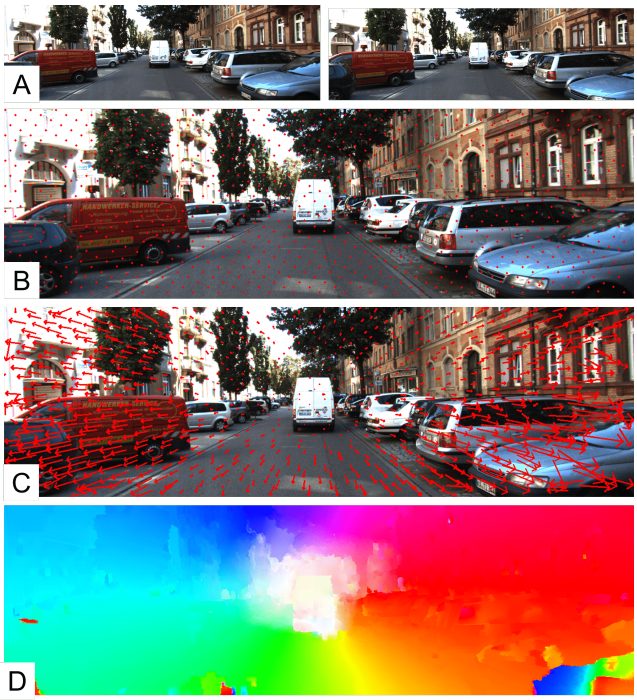

Fig. 2. An overview of the DeGraF-Flow pipeline

which define a gradient vector $\overrightarrow{C_{pos}C_{neg}}$ for the region. C_{pos} is computed as the spatially weighted average pixel value:

$$C_{pos}(x_{pos}, y_{pos}) = C_{pos}\left(\frac{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1} iI(i,j)}{S_{pos}}, \frac{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1} jI(i,j)}{S_{pos}}\right) \quad (1)$$

where $S_{pos} = \sum_{i=0}^{h-1}\sum_{j=0}^{w-1} I(i,j)$. The negative centroid C_{neg} is similarly defined as the weighted average of inverted pixel values:

$$C_{neg}(x_{neg}, y_{neg}) = C_{neg}\left(\frac{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1} i(1+m-I(i,j))}{S_{neg}}, \frac{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1} j(1+m-I(i,j))}{S_{neg}}\right) \quad (2)$$

where $S_{neg} = \sum_{i=0}^{h-1}\sum_{j=0}^{w-1}(1+m-I(i,j))$ and $m = \max_{(i,j)} I(i,j)$. Inverted pixel values are normalised ($1 \rightarrow 256$) to avoid division by zero.

The key-point in each region is taken as the location of the most stable centroid, i.e if $S_{neg} > S_{pos}$ then the key-point is at (x_{neg}, y_{neg}) and vice versa. This choice is made because the larger value from S_{neg} and S_{pos} is less sensitive to noise and so the corresponding centroid is more robust.

Sparse Point tracking of each DeGraF point to the subsequent image is carried out using the Robust Local Optical Flow approach of [2]. The sparse optical flow vectors recovered are shown in Figure 2C.

Interpolation of sparse vectors to recover the dense flow field in Figure 2D is achieved using EPIC (Edge Preserving Interpolation of Correspondences) [3]. An affine transformation is fitted to the k nearest support flow vectors, estimated using a geodesic distance which penalises crossing of image edges. This work uses image gradients instead of structured edge maps for defining image edges. The result is a dense optical flow field estimation as illustrated in Figure 2D.

3. EVALUATION

Evaluation is carried out on the KITTI optic flow estimation benchmark data sets (denoted as KITTI 2012 [10] and KITTI 2015 [13]).

Statistical accuracy is measured using the established End-Point Error (EPE) metric [10, 13]. For a predicted optical flow vector $\mathbf{u}^p$ at every pixel with corresponding ground flow truth vector $\mathbf{u}^{gt}$, the EPE is then defined as the average difference between the predicted and ground truth vectors over the image:

$$\text{EPE} = \frac{1}{N}\sum_i \|\mathbf{u}_i^p - \mathbf{u}_i^{gt}\|^2, \tag{3}$$

where N is the number of pixels and EPE is hence measured in pixels.

Our method (DeGraF / DeGraF-Flow) was implemented in C++ and all experiments run on a Core i7 using four CPU cores. All timings reported are for the run-time of the algorithm excluding image input/output and display.

Parameters: DeGraF window size, $w = h = 3$ and step size, $\delta = 9$. For RLOF, the global motion and illumination models are used as per [2] with the adaptive cross based support region described in [29]. This algorithm is termed RLOF(IM-GM) in reported results in Table 2. For EPIC we use $k = 128$ [3].

3.1. Comparison of Feature Detectors

To justify the use of DeGraF points, we compare with established feature detectors for dense flow computation on KITTI 2012 [10].

As is shown in [26], when using RLOF and interpolation to recover dense optical flow, a uniform grid of points is a superior input compared to other feature point detectors. Here we repeat the experiment of [26] but with the addition of DeGraF (Table 1).

Table 1 shows the EPE on a KITTI image pair from using popular feature detectors for the first stage of optical flow estimation. To allow meaningful comparison, each detector is tuned to ensure a comparable number of points are detected. DeGraF shows the best performing EPE and efficient detection, equal to FAST.

Point Detector	# Points	EPE	Detection Time(s)
DeGraF [1]	5400	**1.34**	0.07
SURF [30]	5282	1.43	0.90
SIFT [31]	5400	1.77	1.80
AGAST [32]	5624	1.92	0.11
FAST [28]	5562	2.88	0.07
ORB [33]	5400	5.60	0.28

Table 1. Comparison of differing point detectors for dense optical flow computation on KITTI 2012 example #164 with each tuned to detect approximately 5400 points per image.

3.2. Benchmark Comparison - KITTI 2012

The KITTI 2012 benchmark [10] comprises 194 training and 194 test image pairs (1240 $\times$ 376 pixels) depicting static road scenes. Semi dense (50%) ground truth, calculated using a LiDAR, is provided.

Table 2 shows the results on the KITTI test set for CPU methods that can process an image pair in under 20 seconds. All such methods that perform better than DeGraf-Flow in terms of EPE are included in the table. Not all less accurate methods are shown. Below the dividing line the best sparse flow methods are shown. Note how, although these methods have very low error, this error is only reported over a greatly reduced density of points. The result of standalone Pyramidal Lucas-Kanade (PLK) [24] is shown as a baseline reference. The first two columns give the percentage of estimated flow vectors that have an EPE of more than 3px. The next two columns give average EPE values over the test set. Noc denotes statistics on only the non occluded pixels, where occluded pixels are those which appear in the first image but not in the second. Methods are ranked in order of increasing non-occluded outlier percentage (Out-Noc).

DeGraF-Flow shows promising results in terms of balancing computational efficiency (run-time, Table 2) and accuracy. In Table 2 it ranks thirteenth in accuracy (Out-Noc) and is shown to be the *fourth fastest CPU method* that has a percentage outlier of non occluded pixels of less than 10%. Other faster methods such as PCA-Flow and DIS-Fast show significantly higher EPE . PCA-Flow and PCA-Layers both employ an interpolation scheme for computing dense flow. The superior results of DeGraF-Flow show that the EPIC interpolator is the correct choice, which agrees with the findings in [26]. At the time of testing, DeGraF-Flow was the 18th fastest overall, including GPU methods with an overall accuracy rank of 56 from a total of 95 submissions within KITTI 2012.

EpicFlow and RLOF(IM-GM) are highlighted in Table 2 as these are the two constituent components of our method. DeGraF-Flow is outperformed by EpicFlow but runs almost five times faster. RLOF(IM-GM) represents near state of the art in sparse optical flow, making it an excellent candidate for tracking DeGraF points. RLOF is second only to FSDEF [25] which is comtemporary work to that presented here.

3.3. Benchmark Comparison - KITTI 2015

The KITTI 2015 benchmark [13] comprises 200 training and 200 test image pairs (1242 $\times$ 375 pixels) with the increased

Method	Out-Noc	Out-All	Avg-Noc	Avg-All	Density	Run-time	Environment
SPS-Fl [23]	3.38 %	10.06 %	0.9 px	2.9 px	100.00 %	11 s	1 core @ 3.5 Ghz
SDF [34]	3.80 %	7.69 %	1.0 px	2.3 px	100.00 %	TBA	1 core @ 2.5 Ghz
MotionSLIC [35]	3.91 %	10.56 %	0.9 px	2.7 px	100.00 %	11 s	1 core @ 3.0 Ghz
RicFlow [36]	4.96 %	13.04 %	1.3 px	3.2 px	100.00 %	5 s	1 core @ 3.5 Ghz
CPM2 [37]	5.60 %	13.52 %	1.3 px	3.3 px	100.00 %	4 s	1 core @ 2.5 Ghz
CPM-Flow [37]	5.79 %	13.70 %	1.3 px	3.2 px	100.00 %	4.2s	1 core @ 3.5 Ghz
MEC-Flow [38]	6.95 %	17.91 %	1.8 px	6.0 px	100.00 %	3 s	1 core @ 2.5 Ghz
DeepFlow [18]	7.22 %	17.79 %	1.5 px	5.8 px	100.00 %	17 s	1 core @ 3.6Ghz
RecSPy+ [39]	7.51 %	15.96 %	1.6 px	3.6 px	100.00 %	0.16 s	1 core @ 2.5 Ghz
RDENSE(anon)	7.72 %	14.02 %	1.9 px	4.6 px	100.00 %	0.5 s	4 cores @ 2.5 Ghz
EpicFlow [3]	7.88 %	17.08 %	1.5 px	3.8 px	100.00 %	15 s	1 core @ 3.6 Ghz
SparseFlow [17]	9.09 %	19.32 %	2.6 px	7.6 px	100.00 %	10 s	1 core @ 3.5 Ghz
DeGraF-Flow	9.41 %	16.93 %	2.5 px	8.4 px	100.00 %	3.2 s	4 cores @ 2.5 Ghz
PCA-Layers [19]	12.02 %	19.11 %	2.5 px	5.2 px	100.00 %	3.2 s	1 core @ 2.5 Ghz
PCA-Flow [19]	15.67 %	24.59 %	2.7 px	6.2 px	100.00 %	0.19 s	1 core @ 2.5 Ghz
DB-TV-L1 [40]	30.87 %	39.25 %	7.9 px	14.6 px	100.00 %	16 s	1 core @ 2.5 Ghz
DIS-FAST [41]	38.58 %	46.21 %	7.8 px	14.4 px	100.00 %	0.023s	1 core @ 4 Ghz
FSDEF [25]	1.07 %	1.17 %	0.7 px	0.7 px	41.81 %	0.26s	4 cores @ 3.5 Ghz
RLOF(IM-GM) [2]	2.48 %	2.64 %	0.8 px	1.0 px	11.84 %	3.7 s	4 core @ 3.4 Ghz
RLOF [42]	3.14 %	3.39 %	1.0 px	1.2 px	14.76 %	0.488 s	GPU @ 700 Mhz
BERLOF [43]	3.31 %	3.60 %	1.0 px	1.2 px	15.26 %	0.231 s	GPU @ 700 Mhz
PLK [24]	27.44 %	31.04 %	11.3 px	17.3 px	92.33 %	1.3 s	4 cores @ 3.5 Ghz

Table 2. KITTI 2012 Benchmark Results - comparison of the best CPU methods that can process an image pair in under 20 seconds. Below the line = sparse flow algorithms; Noc = pixels that are not occluded in the second image; First two columns = percentage of flow vectors that have an EPE of greater than 3px; methods are ordered by Out-Noc. Full table of results can be found on the KITTI benchmark website [44].

Rank	Method	Fl-bg	Fl-fg	Fl-all	Run-time	Environment
65	EpicFlow [3]	25.81 %	28.69 %	26.29 %	15 s	1 core @ 3.6 Ghz
67	DeepFlow [18]	27.96 %	31.06 %	28.48 %	17 s	1 core @ 3.5 Ghz
68	DeGraF-Flow	28.78 %	29.69 %	28.94 %	3.2 s	4 cores @ 2.5 Ghz

Table 3. KITTI 2015 Benchmark Results - percentage of pixels with EPE ≥ 3 pixels are given; fg and bg refer to motion of foreground moving vehicles and the static background scene respectively.

challenges of dynamic scene objects (vehicles). These exhibit far larger pixel displacements in some areas resulting in lower algorithm performance on KITTI 2015 (Table 3) than on KITTI 2012 (Table 2).

Table 3 shows the results of DeGraF-Flow on the test set compared to DeepFlow [18] and EpicFlow [3] (which both use DeepMatch as the sparse matching technique). The percentage of flow vectors with an EPE greater than 3px are shown, with fg and bg referring to foreground objects and the background scene respectively.

As with the KITTI 2012 benchmark results our approach shows comparable accuracy with significantly reduced run-time over EpicFlow and DeepFlow. Over all submissions to the KITTI 2015 benchmark, DeGraF-Flow reports *the 10th fastest CPU method*. In terms of accuracy it places 13th (out of 22) from CPU methods that take less than than 10 seconds to process an image pair. Over the total of 90 submissions DeGraF-Flow places 68 in terms of Fl-all and 20 in terms of run-time.

4. CONCLUSION

In this paper we present DeGraF-Flow, a novel optical flow estimation method. DeGraF-Flow uses a rapidly computed grid of Dense Gradient based Features (DeGraF) and then combines an existing state of the art sparse point tracker (RLOF [2]) and interpolator (EPIC [3]) to recover dense flow. With only minimal impact on accuracy (within 2% of EPIC-Flow [3] and DeepFlow [18] across all metrics) our approach offers significant gains in computational performance for dense optic flow estimation.

We show that the invariability, density and uniformity of DeGraF points yield superior dense flow results compared to other popular point detectors. The rapid end-to-end optic flow estimation time is also conducive to real-time applications such as scene understanding for future autonomous vehicle applications.

On the KITTI 2012 and 2015 benchmarks [10, 13] our method shows competitive run-time and comparable accuracy with other CPU methods. Future work will exploit the tracking of DeGraF features for applications in an autonomous vehicle setting.

5. REFERENCES

[1] I. Katramados and T. P. Breckon, "Dense gradient-based features (DEGRAF) for computationally efficient and invariant feature extraction in real-time applications," *Int. Conf. Image Process.*, pp. 300–304, 2016.

[2] T. Senst, J. Geistert, and T. Sikora, "Robust Local Optical Flow : Long-Range Motions and Varying Illuminations," *Int. Conf. Image Process.*, pp. 4478–4482, 2016.

[3] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "EpicFlow : Edge-Preserving Interpolation of Correspondences for Optical Flow," *Conf. Comput. Vis. Pattern Recognit.*, pp. 1164–1172, 2015.

[4] K. P. Horn and B. G. Schunck, "Determining Optical Flow," *Artif. Intell.*, vol. 17, pp. 185–203, 1981.

[5] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision.," *Int. Jt. Conf. Artif. Intell.*, pp. 674–679, 1981.

[6] D. Fortun, P. Bouthemy, and C. Kervrann, "Optical flow modeling and computation : a survey," *Comput. Vis. Image Underst.*, vol. 134, pp. 1–21, 2015.

[7] L. Sevilla-Lara, D. Sun, V. Jampani, and M. Black, "Optical Flow with Semantic Segmentation and Localized Layers," *Conf. Comput. Vis. Pattern Recognit.*, pp. 3889–3898, 2016.

[8] K. Kale, S. Pawar, and P. Dhulekar, "Moving Object Tracking using Optical Flow and Motion Vector Estimation," *4th Int. Conf. Reliab. Infocom Technol. Optim.*, pp. 1–6, 2015.

[9] M. O. A. Aqel, M. Marhaban, M. I. Saripan, and N. B. Ismail, "Review of visual odometry: types, approaches, challenges, and applications," *Springerplus*, vol. 5, no. 1, 2016.

[10] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," *Comput. Vis. Pattern Recognit.*, pp. 3354–3361, 2012.

[11] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *Int. J. Comput. Vis.*, vol. 92, no. 1, pp. 1–31, 2011.

[12] D. Butler, J. Wulff, G. Stanley, and M. Black, "A Naturalistic Open Source Movie for Optical Flow Evaluation," *Eur. Conf. Comput. Vis.*, pp. 611–625, 2012.

[13] M. Menze and A. Geiger, "Object Scene Flow for Autonomous Vehicles," *Conf. Comput. Vis. Pattern Recognit.*, pp. 3061–3070, 2015.

[14] T. Brox and J. Malik, "Large displacement optical flow: Descriptor matching in variational motion estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 3, pp. 500–513, 2011.

[15] G. Eilertsen, P. Forssén, and J. Unger, "BriefMatch : Dense binary feature matching for real-time optical flow estimation," *Proc. Scand. Conf. Image Anal.*, pp. 221–233, 2017.

[16] C. Liu, J. Yuen, and A. Torralba, "Sift flow: Dense correspondence across scenes and its applications," *Dense Image Corresp. Comput. Vis.*, pp. 15–49, 2015.

[17] R. Timofte and L. Van Gool, "SparseFlow: Sparse Matching for Small to Large Displacement Optical Flow," *IEEE Winter Conf. Appl. Comput. Vis.*, pp. 1100–1106, 2015.

[18] P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "DeepFlow : Large displacement optical flow with deep matching," *Int. Conf. Comput. Vis.*, pp. 1385–1392, 2013.

[19] J. Wulff and M. Black, "Efficient sparse-to-dense optical flow estimation using a learned basis and layers," *IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 120–130, 2015.

[20] C. Bailer, B. Taetz, and D. Stricker, "Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation," *Int. Conf. Comput. Vis.*, pp. 4015–4023, 2015.

[21] Q. Chen and V. Koltun, "Full Flow : Optical Flow Estimation By Global Optimization over Regular Grids," *Conf. Comput. Vis. Pattern Recognit.*, pp. 4706–4714, 2016.

[22] J. Janai, F. Güney, J. Wulff, M. Black, and A. Geiger, "Slow Flow : Exploiting High-Speed Cameras for Accurate and Diverse Optical Flow Reference Data," *Conf. Comput. Vis. Pattern Recognit.*, pp. 3597–3607, 2017.

[23] K. Yamaguchi, D. McAllester, and R. Urtasun, "Efficient joint segmentation, occlusion labeling, stereo and flow estimation," *Lect. Notes Comput. Sci.*, vol. 8693, pp. 756–771, 2014.

[24] J-Y. Bouguet, "Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm," *Intel Corp.*, 2001.

[25] M. Garrigues and A. Manzanera, "Fast Semi Dense Epipolar Flow Estimation," *IEEE Winter Conf. Appl. Comput. Vis.*, pp. 427–435, 2017.

[26] J. Geistert, T. Senst, and T. Sikora, "Robust local optical flow: Dense motion vector field interpolation," *Pict. Coding Symp.*, pp. 1–5, 2016.

[27] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," *Procedings Alvey Vis. Conf.*, pp. 147–151, 1988.

[28] E. Rosten, E. Rosten, R. Porter, R. Porter, and T. Drummond, "Faster and better: a machine learning approach to corner detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 105–19, 2010.

[29] T. Senst, T. Borgmann, and T. Sikora, "Cross based robust local optical flow," *Int. Conf. Image Process.*, pp. 1967–1971, 2014.

[30] H. Bay, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, 2008.

[31] D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[32] E. Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger, "Adaptive and generic corner detection based on the accelerated segment test," *Eur. Conf. Comput. Vis.*, pp. 183–196, 2010.

[33] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," *IEEE Int. Conf. Comput. Vis.*, pp. 2564–2571, 2011.

[34] M. Bai, W. Luo, K. Kundu, and R. Urtasun, "Exploiting Semantic Information and Deep Matching for Optical Flow," *Eur. Conf. Comput. Vis.*, pp. 154–170, 2016.

[35] K. Yamaguchi, D. McAllester, and R. Urtasun, "Robust monocular epipolar flow estimation," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 1862–1869, 2013.

[36] Y. Hu, Y. Li, and R. Song, "Robust Interpolation of Correspondences for Large Displacement Optical Flow," *IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 4791–4799, 2017.

[37] Y. Li, Y. Hu, R. Song, P. Rao, and Y. Wang, "Coarse-to-fine PatchMatch for Dense Correspondence," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 1, no. c, pp. 1–12, 2017.

[38] M. A. Mohamed, M. H. Mirabdollah, and B. Mertsching, "Monocular epipolar constraint for optical flow estimation," in *Computer Vision Systems*, (2017), pp. 62–71.

[39] P. Hu, G. Wang, and Y. Tan, "Recurrent Spatial Pyramid CNN for Optical Flow Estimation," *IEEE Trans. Multimed.*, vol. 14, no. 8, 2018.

[40] C. Zach, T. Pock, and H. Bischof, "A Duality Based Approach for Realtime TV-L 1 Optical Flow," *Ann. Symp. Ger. Assoc. Patt. Recogn*, pp. 214–223, 2007.

[41] T. Kroeger, R. Timofte, D. Dai, and L. Van Gool, "Fast Optical Flow using Dense Inverse Search," *Lect. Notes Comput. Sci.*, vol. 9908, 2016.

[42] T. Senst, V. Eiselein, and T. Sikora, "Robust local optical flow for feature tracking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 9, pp. 1377–1387, 2012.

[43] T. Senst, J. Geistert, I. Keller, and T. Sikora, "Robust local optical flow estimation using bilinear equations for sparse motion estimation," *Int. Conf. Image Process.*, pp. 2499–2503, 2013.

[44] KITTI website, "www.cvlibs.net/datasets/kitti," (Visited: 2018-03-30).