Fast Semi Dense Epipolar Flow Estimation

Matthieu Garrigues, Antoine Manzanera

Fast Semi Dense Epipolar Flow Estimation

Matthieu Garrigues Antoine Manzanera
U2IS-Robotics & Vision, ENSTA-ParisTech
Paris-Saclay University, FRANCE
firstname.lastname@ensta-paristech.fr

Abstract

Optical flow computation consists in recovering the apparent motion field between two images with overlapping fields of view. This paper focuses on a subset of optical flow problems, called epipolar flow, where the camera moves inside a scene containing no moving objects. Accurate solutions exist but their high computational complexities make them non suitable for a large panel of real-time applications.

We propose a new epipolar flow approach with low computational complexity achieving the best error rate on the non dense KITTI optical flow 2012 benchmark and running $1000\times$ *faster than the second ranked approach. On a 4core 3GHz processor, our multi-core implementation computes a semi dense optical flow field of a* $450k$ *pixels image in* $260ms$*. It is a significant advance in reducing the running time of accurate optical flow computation.*

To achieve such results we rely on the epipolar constraints and the local coherence of the optical flow not only to increase accuracy but also to reduce computational complexity.

Our contribution is twofold. It is first, the acceleration and the accuracy increase of current RANSAC based visual odometry algorithms via the estimation of a robust sparse flow field, well distributed over the image domain. And then, the estimation of a semi dense flow field leveraging epipolar constraints and a propagation scheme to speedup the estimation and reduce error rates.

1. Introduction

With the raise of augmented reality, virtual reality and robotics, capturing the geometry of a real-world 3d environment has attracted considerable research efforts in academic and industrial laboratories.

The problem consists in computing the 3d localization of objects observed using an on-board sensor. The ultimate goal of this challenge is to achieve the best accuracy while reducing the manufacturing cost and power consumption [13].

As of the year 2016, the most precise sensors are LIDARs. They measure the depth of neighboring object by timing the round trip of a light pulse. Their main disadvantage is their high price tag and high power consumption.

Much cheaper than LIDARs, active structured-light 3D scanners are well-spread: the Microsoft Kinect V1 [20], and other portable devices such as Google Tango [6]. They work well in indoor and small range environments but their accuracy drop under the sunlight and large distances. This makes them unsuitable for autonomous driving applications for example. The second problem of these sensors is their high power consumption due to the emission of the structured light.

The cheapest sensors are the RGB cameras. However, they do not provide depth information, then different methods to recover it were proposed. They usually involve one or two steps: First, if they are not already known, the estimation of the relative camera poses (also known as visual odometry), and the optical flow computation. The power consumption of such systems is smaller than LIDAR and structured-light scanners but their accuracy highly depends on the texture of the scene. Furthermore, the higher complexity of the problem makes it harder to build fast and accurate applications, especially for the non-calibrated case and where the relative poses of the different points of view are unknown.

In this paper, we focus on epipolar flow. In this paper, we focus on epipolar flow. It consists in the estimation of a disparity map from an image pair captured by a camera moving inside a static scene. Despite its inability to capture the depth of moving objects, it is useful in applications like 3D scene reconstruction. It also has two advantages over stereo-vision done with a pair of aligned cameras: First, it only uses one camera and second, it is more flexible with respect to the viewpoint variation.

We propose a novel approach to compute the optical flow between two images. It is faster and more precise than the current state of the art algorithms in the static scene/moving camera case. According to the *KITTI optical flow* bench-

mark [5] 98.3% of the estimated flow vectors have a Euclidean distance to the ground truth inferior to 3 pixels and, in average, it covers 50% of the image pixels. It is ranked first for the partial estimation in this benchmark, while being 1000× faster than the second. Our following three contributions span over the different stages of the epipolar flow pipeline:

- A well distributed and accurate **sparse flow vector field estimation** that increases robustness and reduces runtime of RANSAC based fundamental matrix estimation.
- A fast and almost error free estimation of a **semi-dense optical flow**, narrowing the search to the epipolar lines.
- An **error filter** relying on the Lucas-Kanade optical flow [8] and a simple local coherence criterion that filters most of the erroneous flow vectors.
- A **fast open-source C++ implementation** optimized for multi-core CPU and processing a *KITTI* image pair in 260 milliseconds on a 4-core 3GHz processor. We are currently submitting the code to the open source (under the MIT licence) C++ image processing library *Video++* [4].

The paper is organized as follows. In Section 2, we present how our approach is related to the state of the art, and also how it differs from it. Then in Section 3, we go through the notation used in this paper. In Section 4, we describe our improved fundamental matrix estimation and in Section 5, we present the semi dense optical flow estimation. Finally, we evaluate our approach with the *KITTI optical flow benchmark* in Section 6.

2. Related Works

During the last years, many works were conducted to improved the quality and/or reduce the computation time of optical flow from an image pair. The *KITTI optical flow benchmark* references and ranks 78 methods. Competing algorithms can rely on 3 kinds of additional information: First, they may estimate and make use of epipolar constraints, then they may use stereo image pairs, and finally they may use a sequence of more than two images. On Figure 1, we draw the distribution of the algorithms competing on the partial estimation section of this benchmark. We observe that a high proportion of algorithms have running time ranging from 10s to 1000s, and error rates ranging from 4% to 10%. Accurate and fast running algorithms are fewer, with **only 5, including ours, running under 1s and with error rate lower than 10%**.

The best approaches often make hypothesis about the 3D structure of the scene: In [1, 16, 17, 19, 15], it is considered

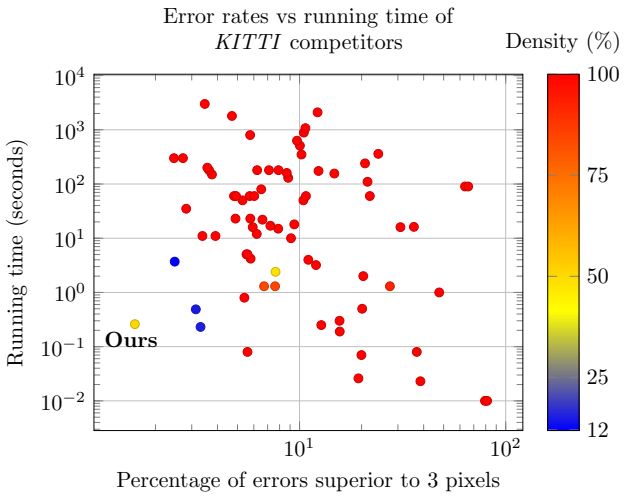

Figure 1. Error rates vs running time of the algorithms referenced by the *KITTI optical flow* section "partial estimation", on a logarithmic scale. Colors encode the density of the computed optical flow. Only 5 competitors have both a running time under 1s, and an error rate lower than 10%.

as a union of **slanted 3D plane portions** (facets). This allows to shift the problem from the pixel scale to the super-pixel (projected facet) scale, significantly reducing the number of unknowns. These approaches formulate a **global energy function** based on the brightness constancy and other data, with regularization terms. To find a solution close to the global minimum, they rely on an energy minimization technique like particle convex belief propagation [9], graph cuts [15], or customized iterative algorithms [17] to jointly estimate the super-pixel segmentation and the associated slanted facets. These techniques have the advantage of providing an accurate dense optical flow but their computation time remains high: **from 10s to several minutes per image**.

While these solutions focus on accuracy rather than running time, accelerating the estimation of optical flow to real time ($> 10fps$) processing also drew much research interest. In [10], the authors present eFOLKI, derived from the multi-scale iterative Lucas Kanade [8] algorithm. To speedup the computation, they reduce the number of pixel interpolations by **wrapping the whole image after each iteration** of the gradient descent. To increase accuracy, they use a **dynamic integration window** size, to avoid over-smoothing the optical flow field. Running on the GPU, this is **one of the fastest optical flow** computation of the *KITTI optical flow benchmark*, but it is ranked $60th$ in terms of accuracy.

As shown in Figure 1, our proposal is distinguished by a low running time and the lowest error rate. To achieve this, it first combines several ideas from the state of the art:

- It constraints the estimation of optical flow to the

epipolar lines as in [17, 18].

- It uses a fast **local gradient descent** minimizing an SSD distance between two patches to compute each optical flow vector, similarly to Lucas Kanade [8].
- Its **parallel implementation** takes advantage of a multi-core CPU architecture as in [10] which compute the optical flow on a GPU.

The low running time of our approach is also partly due to the region growing algorithm that propagates and refines optical flow vectors. Unlike the leading algorithms of the KITTI benchmark, this step leverages local coherence not to build an expensive global energy minimization but to reduce running time and errors.

Another particularity of our solution is the use of the Lucas-Kanade algorithm not to compute actual optical flow vectors, but to help filtering the errors.

3. Notations

In the following, we will use the following notations:

- I_1 and I_2 are the two input images.
- A 3D point X is projected on images I_1 and I_2 at the 2D pixel coordinates x_1 and x_2 respectively.
- $\tilde{x}$ is the 3D homogeneous coordinates of a 2D vector x.
- F is the 3×3 fundamental matrix, relating $\tilde{x}_1$ and $\tilde{x}_2$.

4. Fundamental Matrix Estimation

Generic optical flow algorithms have to estimate, for each pixel, two unknowns: the displacement along the two dimensions of the image. They always face two major challenges: repetitive texture causing erroneous matches, and the large dimension of the search space increasing the running time.

However, in a static scene captured by a moving camera, the optical flow projected on the focal plane is only related to the distance of the objects and the camera motion. If the camera rotation and translation are known, only the 1D disparity of each pixel remains to be estimated. The search space is then reduced from a 2D domain to a line, called the epipolar line.

This greatly simplifies the computation of the optical flow. First, because the probability that more than one similar pixels appear is lower in a smaller space, thus reducing the number of errors. And second, it reduces the number of comparisons required to find the best match, thus speeding up the search.

The camera motion (as well as its intrinsic parameters) is usually encoded as the fundamental matrix F, with the following property: if $\tilde{x}_1$ and $\tilde{x}_2$ are the homogeneous coordinates of the projection of the same 3D point X in the first and second image respectively, we have:

$$\tilde{x}_2^T F \tilde{x}_1 = 0 \tag{1}$$

In other words, it means that X, its two projections and the two optical centers lie on the same 3D plane. When x_1 and F are known and x_2 is unknown, the coefficients of the line where x_2 lies can be computed as follows: if $l_2 = \tilde{x}_1^T F$, we get $l_2^T \tilde{x}_2 = 0$ which defines a line in the I_2 coordinate space, and allows us **to reduce the matching of a 2D point to a search along a single 1D line**.

The fundamental matrix also provides the common point of all epipolar lines, which is usually called epipole and denoted e_1 (resp. e_2) for the first (resp. second) image :

$$\tilde{e}_2^T F = \vec{0} \tag{2}$$

$$F \tilde{e}_1 = \vec{0} \tag{3}$$

The estimation of the fundamental matrix F usually relies on a set of correspondences between pixels of I_1 and I_2. Theoretically, only seven correspondences are needed to estimate F, but it is common to rely on few hundreds to increase the robustness to matching errors.

There are existing solutions [7, 11] to estimate F from a set of correspondences. Their accuracy and runtime depend on two criteria:

- The precision of the 2D/2D correspondences.
- The uniform distribution of the correspondences over the image domain.

We propose in this section a method to extract the sparse set of correspondences maximizing these two criteria. It has four advantages:

1. It minimizes the number of iterations of RANSAC schemes by prefiltering outliers, speeding up the estimation of the fundamental matrix.
2. It estimates small displacements as well as large ones.
3. It ensures a good distribution of 2D/2D correspondences over the image domain, increasing the accuracy of the estimation
4. It realizes a good trade-off between accuracy and running time on a multi-core processor.

In the remaining of this section, we first give an overview of the algorithm, then go through each part into deeper details, and finally benchmark its accuracy and speed on the KITTI optical flow benchmark.

4.1. Overview

The Fundamental matrix estimation pipeline begins with the matching of two FAST [12] keypoint sets, extracted from the two images. Grid indexing on the keypoint sets speeds up the search. Then, the displacement vectors are refined using a pyramidal Lucas-Kanade [8] scheme. A filter is applied on the set of resulting matches to reduce the number of errors. The fundamental matrix is finally estimated with a classical method [7, 11].

4.2. Evaluation

To evaluate the precision of the fundamental matrix associated to an image pair, we estimate the deviation of the epipolar lines from the ground truth optical flow.

Using the set of pixels with ground truth vectors GT provided by the KITTI optical flow benchmark, we define the fundamental matrix F error E_F as the maximum distance between the epipolar line associated to a pixel x_1 and its ground truth match x_2.

$$E_F = \max_{(x_1,x_2)\in GT} \frac{|\tilde{x}_2^T l|}{\sqrt{l_a^2 + l_b^2}}, \tag{4}$$

with:

$$l = F\tilde{x}_1 = \begin{pmatrix} l_a \\ l_b \\ l_c \end{pmatrix}. \tag{5}$$

In the following, we analyze the impact of several parameters on the fundamental matrix error and on the runtime of the whole pipeline.

4.3. Sparse FAST to FAST keypoint matching

The first step of the fundamental matrix estimation is the computation of a sparse optical flow vector field. Its role is to feed the fundamental matrix estimation and to provide seeds for the semi dense epipolar flow estimation (Section 5).

To compute a sparse optical flow vector field, lots of methods rely on the fact that the displacement is small and can be recovered by locally optimizing a SSD distance *via* a gradient descent [8]. The quality of these methods highly depends on the magnitude of the displacements, even in a multi-scale framework [2]. In our work, to match large displacement as well as small displacements, we preferred to match the FAST keypoints of I_1 against the FAST keypoints of the I_2 image. This way, we rely on the keypoint repeatability instead of the small displacement assumption. For each keypoint selected in the first image, we search for the keypoint in the second image that has the smallest SSD distance to it (The SSD is computed using a 11×11 neighborhood patch). Thanks to the keypoint detector repeatability, it can be expected that most of the keypoints in the first image will also be detected in the second image.

However, the FAST detector is not appropriate to feed a fundamental matrix estimator: the distribution of the FAST keypoints highly depends on the content of the image, as the FAST detector is not designed to guarantee an even spatial distribution, which is detrimental to the estimation of the fundamental matrix.

In order to correct this distribution problem, we set up the following blockwise selection strategy: in each image block of a $Nc \times Nc$ grid, we select the FAST keypoint with the highest score, as long as it is higher than a threshold $keypointTh$. This blockwise selection strategy ensures that only one keypoint per block gets selected. This way, keypoints are more evenly distributed over the image domain and capture more motion information, thus improving the estimation of F.

In spite of these advantages, such blockwise selection alters the repeatability of the FAST detector and causes some erroneous matches. A filtering strategy is proposed in Section 4.4 to remove the errors of this matching step.

We tested the fundamental matrix estimator with all couples of parameters Nc and $keypointTh$, and averaged the measures on the 50 first images of the KITTI dataset. The results confirm our hypothesis: Figure 2 shows that thanks to the better distribution induced by the blockwise strategy, a smaller grid resolution Nc can be used to reduce the runtime by an order of magnitude, without increasing the error on the estimation of F.

4.4. Speeding up the F Estimator with Erroneous Matches Filtering

Because the blockwise FAST keypoint strategy alters the repeatability of the keypoints, some keypoints in the first image may not be detected in the second image and sometimes matched with some other similar keypoints, leading to erroneous matches.

These errors are usually detected and filtered out using a RANSAC estimator of the fundamental matrix. However, RANSAC iterations of the F estimator are time-consuming, so the fewer outliers we have, the faster the F estimator. Then, to save some of these expensive iterations, we filter out matches diverging from their local median displacement using the following filter:

A flow vector v is kept only if it respects the following constraint:

$$||v - med|| < medTh, \tag{6}$$

with med the median flow vector calculated within the image block containing v in the $Rf \times Rf$ partition grid. Figure 3 shows the performances of a set of runs of the full F estimation pipeline with varying $medTh$ parameters. Lower runtimes correspond to smaller $medTh$ values.

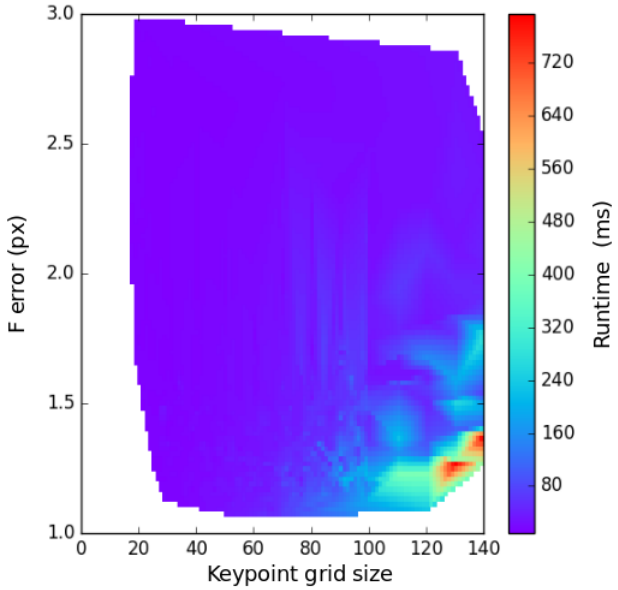

Figure 2. **Impact of the keypoint grid resolution on the runtime of the fundamental matrix estimator. We used a linear interpolation to generate a dense mapping from our set of experiments.**

It can be observed that our filter is able to reduce running time by approximately 30% without impacting the error on F. The variance along the F error is due to fluctuations in the RANSAC precision.

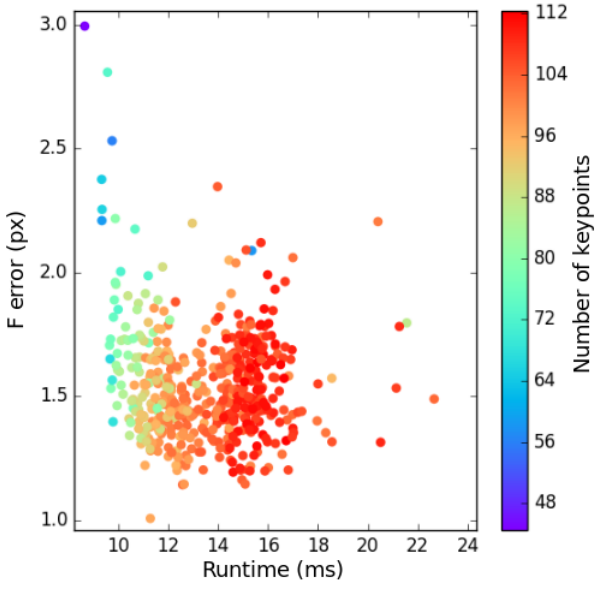

Figure 3. **Efficiency of the Erroneous Matches Filtering**

4.5. Matching refinement with a two Scale Lucas-Kanade Scheme

Because the sparse keypoint matching computes correspondences between FAST keypoints, their precision is not subpixelic. This reduces the precision of the fundamental matrix estimation. To overcome this problem, we refine the matches with a two scale Lucas-Kanade gradient descent [2].

4.6. Fundamental matrix estimation

From the set of refined matches, the fundamental matrix can be estimated with one of the state of the art estimator wrapped in a RANSAC scheme, to be robust to the remaining outliers. Despite the fact that they all benefit from our evenly distributed and error-filtered set of optical flow vectors, the runtime and error of the RANSAC scheme is highly dependent on the inlier acceptance threshold $ransacTh$.

On the presented experiments, the fundamental matrix was estimated using the 8-point algorithm [7] with the RANSAC scheme thanks to the `cv::findFundamentalMat` OpenCV function [3]. Figure 4 shows the influence of the RANSAC threshold on the runtime and on the F error. It can be observed that up to the value 0.3, increasing this parameter reduces the runtime without impacting the F error.

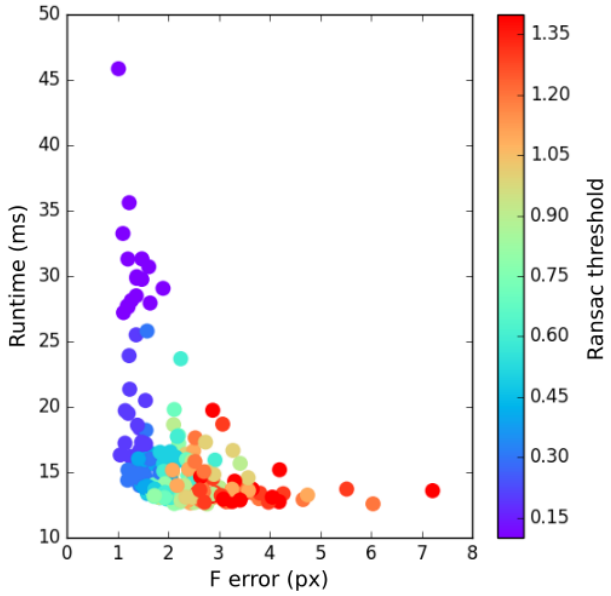

Figure 4. **Impact of the RANSAC threshold on the F error and on the runtime of the F estimator**

5. Guided Semi Dense Optical Flow Computation

We present in this section how, using the epipolar constraints, we guide the estimation of the optical flow to increase robustness and decrease computation time.

5.1. Disparity estimation and propagation

The disparity $D(x_1)$ that we are looking for minimizes for a given pixel x_1 the SSD distance between a patch centered on a pixel in I_2 and the neighborhood patch of x_1 in I_1. Thanks to the epipolar constraint, the domain of this minimization is reduced to the line l_2, which can be parameterized as follows:

$$l_2(x_1, d) = e_2 + dv \tag{7}$$

with $d \in \mathbb{R}$ the disparity, v a direction vector of the epipolar line and e_2 the coordinates of the epipole in I_2.

The computation of the disparity field D can be formulated as the following minimization:

$$D(x_1) = \arg\min_d \sum_{u \in [-3,3]^2} \Big[I_1(x_1 + u) - I_2(l_2(x_1, d) + u) \Big]^2 \tag{8}$$

To compute D, we propose the following algorithm:

Initialization. For all sparse 2D optical flow vectors previously (section 4.3) computed for the fundamental matrix estimation, we save their corresponding disparities in the 2D disparity field at their location. We call them "seed disparities" in the following.

Disparity optimization and propagation. Once the disparity field initialized, we refine and propagate the seed disparities as follows: for each pixel on a propagation front, we refine its disparity by a gradient descent on the epipolar line locally minimizing the SSD distance (equation 8).

After processing a pixel we enqueue each one of its c8-neighbors $\left\{y_1^i\right\}_{i=1..8}$ if it meets the two following conditions:

- The local gradient norm is superior to the $minGradient$ threshold.
- The disparity at point y_1^i has not already been computed or is too different from the disparity we want to propagate (absolute difference greater than $dispPropTh$).

The first criterion prevents the propagation on textureless regions while the second avoids searching twice for the same local minimum.

Note that because there are several seeds, several propagation fronts coexist, and then the same pixel may be examined several times. When re-examining a pixel, we save the new disparity only if it reduces the SSD distance.

Speeding up the search on multi-core processors. The advantage of this propagation scheme is that the processes attached to different seeds are independent and can run in separate threads, as long as they do not process simultaneously the same pixel. This allows a significant speedup on multi-core processors.

5.2. Error detection and filtering

The optimization of the disparity field presented in the previous section restricts its search domain to the epipolar lines. To verify the validity of each optical flow vector estimated at this step, we propose to check if they actually minimize the SSD distance in all the directions.

We do this by running one iteration of the Lucas-Kanade optical flow algorithm [8] on each estimated vector. If the computed optical flow vector does not correspond to a local minimum in the 2D space, the Lucas-Kanade algorithm will diverge from the epipolar line.

Errors are then detected when at least one of the following conditions is met:

- **Non local minimum:** If the distance between the destination of the optical flow vector and the epipolar line is greater than eth.
- **Local incoherence:** If the distance between a vector v and the vectors from its $Nv \times Nv$ neighborhood is greater than dth for at least $dp\%$ of its neighbors, it is considered incoherent. The best results were obtained with $Nv = 15$ and $dp = 60$.

To evaluate the performance of our filter, we use the following protocol: For each parameter pair $\{eth, dth\}$, we count the proportion of flow vectors that are rightly or wrongly classified as an error, the "true" errors being defined as the vectors which differs from the KITTI ground truth of more than 3 pixels. Figure 5 shows the evolution of these two proportions over the different filter configurations. To reduce the parameter space of the evaluation, we set a proportional relation between the two parameters as follows: $eth = dth/4.3$.

Note that the use of Lucas-Kanade [8] here is novel: we not only use it to refine the flow field, but also to decorrelate the erroneous flow vectors, by spreading the wrongly matched points (which stick to the epipolar line) in inconsistent directions. This highly helps to better filter the errors, and other optical flow methods could benefit from it.

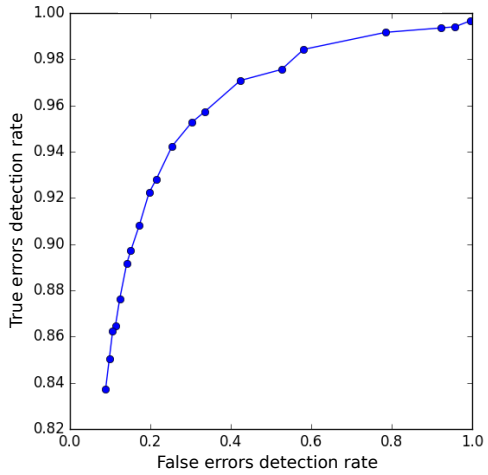

Figure 5. Accuracy of the proposed error filter.

5.3. Second propagation

Error filtering creates holes in the optical flow field. And while large holes are hard to fill, optical flow in small holes can be recovered with satisfying precision from neighboring flow vectors. We therefore replace missing vectors by the average of the vectors located in the 7×7 neighborhood. In our experiments on the KITTI dataset, this method could recover 15% of the image pixels with a negligible error rate increase.

6. Results and comparison with the state of the art

We evaluate our approach on the *KITTI Optical Flow 2012* benchmark. This base contains a collection of image pairs acquired from a forward looking camera embedded on a car and their ground truth optical flow computed from LIDAR data. The dataset features several types of camera motion like forward translations, or rotations.

The table 1 presents the ranking of the leading algorithms of the benchmark with respect to the error rates, for the non-dense optical flow computation. Our approach ranks first, provides error rates $1.5\times$ smaller and running time $1000\times$ smaller than the second ranked one [15]. It is worth noting that the latter covers 100% of the image pixels while ours covers 50%. The best method having similar running time [14] uses a GPU, covers only 15.26% of the pixels and has error rates two times higher. Figure 6 presents the optical flow on a *KITTI* image pair and Figure 7 shows the errors according to the ground truth.

7. Conclusion and future works

We proposed in this paper an algorithm to compute the camera motion and the optical flow in the case of a moving camera and a static scene. Our approach provides a semi dense flow field, covering in average 50% of the image pixels, with an error rate smaller than the previously first ranked approach on the *KITTI optical flow* benchmark, while running $1000\times$ faster.

To obtain such results we optimized each steps of the pipeline with two goals in mind: low running time and when possible, better accuracy. First, by ensuring a good spatial distribution of keypoints and by prefiltering outliers, we could accelerate visual odometry by an order of magnitude while keeping the best accuracy for the fundamental matrix estimation. Then, by leveraging the epipolar geometry and local coherence, we proposed a fast computation of the optical flow and a filter able to filter out 95% of its errors.

Even if it achieves the best error rate on the *KITTI optical flow* benchmark, our approach leaves room for many improvements using ideas from the state of the art. For example, a facet model of the scene could be used as in [17], or *a priori* knowledge about the 3D structure of the scene as in [15]. However, these techniques often rely on expensive global energy minimization and it remains a challenge to leverage them without impacting the running time.

Table 1. *KITTI optical flow 2012* ranking of the seven best algorithms with respect to the error rate (where the errors correspond to estimated vectors whose difference with the ground truth is greater than 3 pixels). The error rate is averaged on the estimated pixels. Our approach (FSDEF) computes an optical flow field with fewer errors and has a running time which is three orders of magnitude lower than the second ranked approach. The only algorithm with lower (-11%) computation time is BERLOF [14], which uses a GPU and covers only 15% of the pixels. For each algorithm, *Out-Noc* represents the error rates of the non occluded pixels, *Density* is the proportion of estimated pixels and *Runtime* the processing time in seconds for one image pair.

Method	Out-Noc	Density	Runtime
FSDEF	**1.59 %**	50.57 %	0.26 s
PRSM	2.46 %	100.00 %	300 s
GME-IM-RLOF	2.48 %	11.84 %	3.7 s
VC-SF	2.72 %	100.00 %	300 s
SPS-StFl	2.82 %	100.00 %	35 s
RLOF	3.14 %	14.76 %	0.488 s
BERLOF	3.31 %	15.26 %	**0.231 s**

References

[1] M. Bleyer, C. Rother, and P. Kohli. Surface stereo with soft segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1570–1577. IEEE, 2010.

[2] J.-Y. Bouguet. Pyramidal implementation of the affine Lucas Kanade feature tracker description of the algorithm. *Intel Corporation*, 5(1-10):4, 2001.

[3] G. Bradski. *Dr. Dobb's Journal of Software Tools*, 2000.

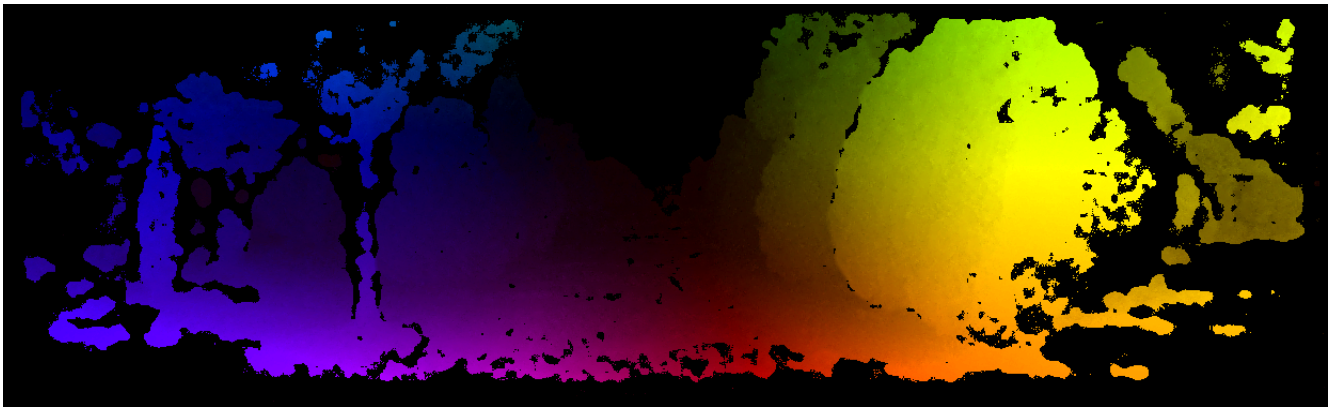

Figure 6. Optical flow computed on a *KITTI* image pair. Intensity (resp. hue) represents the magnitude (resp. orientation) of the flow vectors.

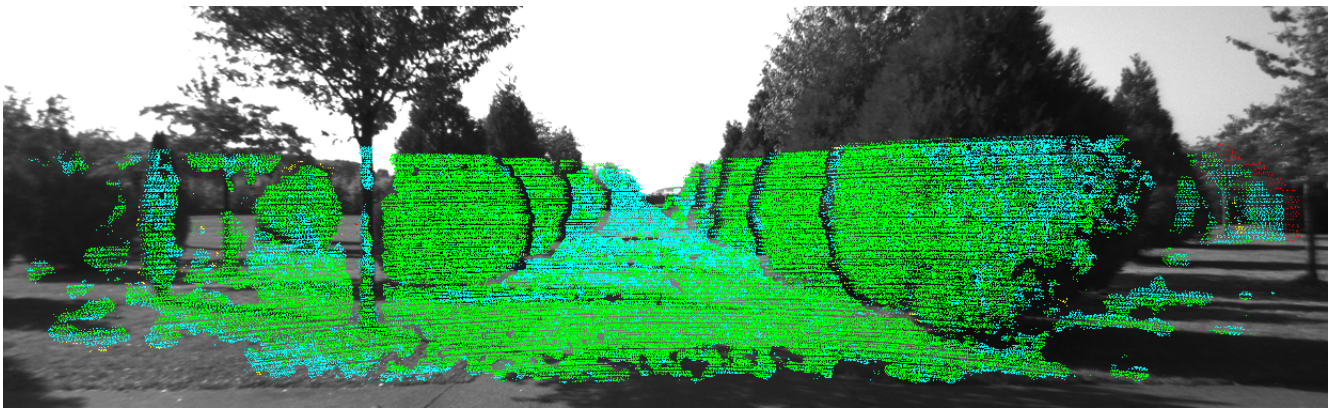

Figure 7. Errors in pixels of the optical flow computed on a *KITTI* image pair. The color code corresponds to the error range. Green: $[0, 1[$ pixels, Cyan: $[1, 3[$ pixels, Yellow: $[3, 10[$ pixels and Red: $[10, \infty[$ pixels.

[4] M. Garrigues and A. Manzanera. Video++, a modern image and video processing C++ framework. In *Design and Architectures for Signal and Image Processing (DASIP), 2014 Conference on*, pages 1–6. IEEE, 2014.

[5] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[6] Google. ATAP Project Tango Google, Feb. 2014.

[7] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. 1987.

[8] B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. In *IJCAI*, volume 81, pages 674–679, 1981.

[9] J. Peng, T. Hazan, D. McAllester, and R. Urtasun. Convex max-product algorithms for continuous MRFs with applications to protein folding. In *Proc. ICML*, 2011.

[10] A. Plyer, G. Le Besnerais, and F. Champagnat. Massively parallel Lucas Kanade optical flow for real-time video processing applications. *Journal of Real-Time Image Processing*, 11(4):713–730, 2016.

[11] H. Richard. In defence of the 8-point algorithm. In *Proceedings of the 5th International Conference on Computer Vision, IEE Computer Science Press, Cambridge*, pages 1064–1070, 1995.

[12] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision (ECCV'06)*, volume 1, pages 430–443, May 2006.

[13] G. Sansoni, M. Trebeschi, and F. Docchio. State-of-the-art and applications of 3d imaging sensors in industry, cultural heritage, medicine, and criminal investigation. *Sensors*, 9(1):568–601, 2009.

[14] T. Senst, J. Geistert, I. Keller, and T. Sikora. Robust local optical flow estimation using bilinear equations for sparse motion estimation. In *ICIP*, pages 2499–2503, 2013.

[15] C. Vogel, K. Schindler, and S. Roth. 3D scene flow estimation with a piecewise rigid scene model. *International Journal of Computer Vision*, 115(1):1–28, 2015.

[16] K. Yamaguchi, T. Hazan, D. McAllester, and R. Urtasun. Continuous Markov random fields for robust stereo estimation. In *European Conference on Computer Vision*, pages 45–58. Springer, 2012.

[17] K. Yamaguchi, D. McAllester, and R. Urtasun. Robust monocular epipolar flow estimation. In *Proceedings of the*

IEEE Conference on Computer Vision and Pattern Recognition, pages 1862–1869, 2013.

[18] K. Yamaguchi, D. McAllester, and R. Urtasun. Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In *European Conference on Computer Vision*, pages 756–771. Springer, 2014.

[19] Y. Zhang, M. Gong, and Y.-H. Yang. Local stereo matching with 3D adaptive cost aggregation for slanted surface modeling and sub-pixel accuracy. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4. IEEE, 2008.

[20] Z. Zhang. Microsoft Kinect sensor and its effect. *IEEE MultiMedia*, 19(2):4–10, Apr. 2012.