

# Towards Real-time Motion Estimation in High-Definition Video Based on Points of Interest

Petr Pulc Faculty of Information Technology, Czech Technical University in Prague, Thákurova 9, 160 00 Prague Email: petr.pulc@fit.cvut.cz

*Abstract*—Currently used motion estimation is usually based on a computation of optical flow from individual images or short sequences. As these methods do not require an extraction of the visual description in points of interest, correspondence can be deduced only by the position of such points.

In this paper, we propose an alternative motion estimation method solely using a binary visual descriptor. By tuning the internal parameters, we achieve either a detection of longer time series or a higher number of shorter series in a shorter time. As our method uses the visual descriptors, their values can be directly used in more complex visual detection tasks.

# I. INTRODUCTION

**M**OTION ESTIMATION can be considered one of the basic tasks in video signal processing. The detection of object position changes is fundamental for areas such as robotics and video surveillance. Generalisation to the tracking of interest point positions is then used in many fields of image processing, including image stitching (to acquire panorama [1] or super-resolution [2]), tracking of facial features (for emotion detection [3]) or image stabilisation [4].

Many other methods can utilise a tracking algorithm to reduce the complex payload computation. Processing (based e.g. on Deep Convolutional Neural Networks [5]) could be limited to only several "best" frames. The extracted features would be then extrapolated to the rest of the video segment if needed [6]. Therefore, the motion estimation approach could present a lightweight counterpart to an expensive processing of all frames in new Video to Text (VTT) tasks.

An important motivation for our interest in motion estimation is the *search for suitable meta-features in multimedia*, as the extracted motion can be used either directly as a metafeature or as a basis for obtaining other meta-features.

In the next section, we present state of the art in video motion estimation, including the traditionally used Kanade-Lucas-Tomasi (KLT) algorithm, and its drawbacks. In II-B we also briefly discuss other feature point selection and description algorithms. Moreover, in II-C we provide a summary of our previous experiments. Section III discusses a proposal of Oriented FAST and Rotated BRIEF (ORB) feature descriptor use for interest point registration between two consecutive frames. Section IV introduces the use of registered key points for the deduction of continuous motion time series. Section V Martin Holeňa Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod Vodárenskou věží 2, 182 07 Prague Email: martin@cs.cas.cz

provides the preliminary results and compares our approach with the KLT feature tracker. We will discuss possible extensions of the herein presented method in Section VI.

# II. STATE OF THE ART

A naïve motion estimation algorithm could be, in theory, constructed by the selection of a patch from one video frame and its convolution with the next frame. An area of the frame containing similar intensities would give a strong response. However, with transformations other than translation, such a method would be prone to many errors. Therefore, more sophisticated motion estimation algorithms are commonly used.

# A. Kanade-Lucas-Tomasi

One of the widely adopted approaches to point tracking is the Lucas-Kanade algorithm [7] combined with feature point selection of Shi-Tomasi [8]. This point tracking algorithm exploits the assumption that motion in video sequences with sufficient frame rate can be estimated by a smooth optical flow function. Therefore, the new position of a feature point is expected to be in a relatively small proximity.

For computation of the new feature point position, a corresponding spatial intensity gradient of a window around a given point of interest (typically  $31 \times 31$  pixels) is to be found. The iteratively discovered displacement of the intensity gradients is then considered as the optical flow vector and translated to motion vectors of the nearby points of interest. To ensure scale invariance and faster detection in a larger neighbourhood, the Lucas-Kanade algorithm uses a pyramidal approach, where the optical flow is computed on the coarsest level of the pyramid and then refined on the lower levels, ultimately in the full resolution. Backwards tracking is used to assess the error of the point tracking. Feature points with a bidirectional error of more than 3 pixels are usually considered as lost.

As Shi and Tomasi stated in [8], the point tracking algorithm will yield insufficient results if unsuitable feature points are selected in the first frame. However, "good features to track" should be based only on the ability to track them, not an a priori quality. This definition is rather unfortunate, as we cannot use a multi-pass approach in the real-time environment. Requirements on the feature points stated by Shi and Tomasi are however widely accepted.

The Kanade-Lucas-Tomasi (KLT) is a powerful and fast keypoint tracker. However, it has several drawbacks:

- (i) Lost feature points are not reconnected if they reappear.
- (ii) New feature points should be added in place of lost ones.
- (iii) The feature points are described only in the first frame.

The point replacement, mentioned in (ii), should favour an ability to track new objects as soon as possible. Detection of new features, however, slows down the motion estimation. Omitting the feature point regeneration, on the other hand, degrades the quality of tracked points throughout time.

If opposed to (iii), the description of feature points is available in all frames, it can be passed directly to more complex image processing methods.

## B. Feature Points and Visual Descriptors

The Harris-Stephens Combined Corner and Edge Detector [9] uses linked edges from the Canny detector [10]. The corner candidates are then filtered by the Harris response function which is still widely used for an assessment of corner quality.

The above mentioned Shi-Tomasi detector [8] directly computes the minimal eigenvalue. If it is close to zero, the considered point is not added to the set of corners.

Some corner detectors compare the intensity of the proposed corners to the intensities of points in a circular mask. If the area with similar intensities is small enough, SUSAN [11] detects a corner. FAST [12] makes this method faster by selecting the pixels for comparison in a pre-trained order.

Other feature point detectors search for scale-space extrema in the Laplacian of Gaussians [13] (or a box filter [14]) on each octave. This introduces the possibility of a multi-scale feature that is considered more stable regarding detection repeatability under various deformations. Modifications of FAST (Oriented FAST [15] and AGAST [16]) also use the scale pyramid to provide the scale information.

For the use in image registration and other tasks, detected points are passed to a visual descriptor. To enable scale and rotation invariance of the description, dominant direction and scale are used from the interest point detection. Therefore, the descriptor is usually connected with a particular point detector.

SIFT [13] and SURF [14] are based on a histogram of neighbouring gradient orientation (or wavelet response, respectively). The local texture information is scale and rotation invariant due to information passed from its detector.

ORB [15] and BRISK [17] (Binary robust invariant scalable keypoints), both binary feature descriptors, use the pixel intensity differences to the detected corner. Order of pixels in the descriptor is based on the gradient orientation (and scale in BRISK).

In the following sections, we will consider only the use of ORB, for several reasons: According to both [18] and our experiments [19], ORB provides the same or even better results on approximately registered features than SIFT and SURF. The computation time of both corner detection and description is however significantly reduced. The principles of our motion estimator can be, however, used in combination with any point detector and appropriate descriptor, only with a slight compromise on the speed of feature detection.

# C. Our Previous Research

In our previous work [19], we have shown that interest point registration across consecutive frames is a feasible solution for motion estimation. The resulting motion vectors were grouped by hierarchical clustering to propose objects on the scene. Then we utilised an extended min-cut algorithm to acquire subpixel precise segmentation boundaries.

The main caveat of this approach was, however, the speed of feature registration. Each feature descriptor was matched in a high-dimensional space (especially for SIFT and SURF). The Approximate Nearest Neighbors [20] approach was in some cases even slower than a brute-force search. Also, spatially distant feature points were commonly misregistered. However, this can be (under assumptions of smooth optical flow, discussed in [21]) eliminated by considering only the neighbourhood of the key point in the next frame. We elaborate on this idea further in this paper. Mainly with a requirement to speed up the process of key point registration. Our approach is discussed in the next section.

Another issue is that we use hierarchical clustering on a relatively large number of not very distinct motion vectors (represented by position, length and direction). To provide more information for future clustering, we propose in Section IV to gather time series of key point position history.

#### III. VIDEO SEQUENCE POINT OF INTEREST REGISTRATION

Key point registration describes a process through which the points with similar visual neighbourhood are mapped onto each other. Instead of actual pixels, feature descriptors are used to both speed up the method of matching and to introduce invariance to common deformations.

The search for a matching descriptor is then usually carried out with the nearest neighbour operator. A point of interest is represented as a vector in feature space, and an appropriate measure is used to find its distance to other points. To eliminate a majority of unpromising comparisons, vectors are usually indexed in a k-d tree [22] or a PCP tree [23]. FLANN [20] is a widely used framework for approximate nearest neighbour search in a k-means tree structure.

A video sequence consisting of a single shot with sufficient framerate presents only minor changes between consecutive frames. Points of interest are therefore spatially almost registered, and detected motion is to some extent smooth. Hence, it is beneficial to index the key points by their location as well. Only the adjacent key points are then checked for descriptor correspondence. Because such index is only two-dimensional, k-d tree indexing is chosen as the approach with the fastest query time and the smallest overhead [20].

Our approach uses a k-d tree for indexing of the interest points by their position. Only points closer than a given radius (by default 2%) and with the Hamming distance of its descriptor smaller than 64, as proposed in [15], are considered as candidates. The point with the lowest Hamming distance is considered a match and excluded from the k-d tree.

For k-d tree indexing and search, we used a header-only implementation nanoflann [24], which allowed us to control the indexing of points fully. The time required for the construction of the index is negligible –  $\mathcal{O}(n \log n)$ , where n is the number of points indexed. The radius search is, however, significantly improved. Instead of comparing distances to all points and sorting in  $\mathcal{O}(n+n \log n)$ , the search in the k-d tree takes only  $\mathcal{O}(n^{\frac{1}{2}}+m)$ , where m is the number of results. We can also estimate the feature position from the previous frame, as we discuss in the next section.

# IV. DEDUCTION OF OPTICAL FLOW

After processing the first two frames and with an assumption of smooth motion, the approximate position of the key point in the next frame can be estimated. Such an assumption enables us to provide a more likely query point for the radius search as well as a possibility to reduce the radius to increase the speed of registration.

To this end, we keep a list of active tracked features with current position and ORB description, position delta, radius and history of positions. Each successive frame is then registered by the algorithm presented in the previous section and matched features are updated accordingly. Unmatched features increase the radius and can, therefore, recover for some time. When radius exceeds a threshold, the feature becomes lost. The rest of the newly detected points of interest is appended to the set of active points with default uncertainty value and zero position delta, as no history is presumably available.

As a result, we obtain a list of features with their description and history of positions. This information can be used during the processing of multimedia content for online recognition and detection scenarios, or offline for multimedia content classification and deeper analysis.

# V. MOTION ESTIMATION RESULTS AND PERFORMANCE

Figure 1 presents a comparison of the original Kanade-Lucas-Tomasi algorithm with our proposed method based on matching ORB descriptors. The test sequences were captured just for this experiment by Lumix FZ80, as it provides a 4K video recording. The camera moves downwards during the shot and continuously tilts upwards. The background is therefore visually moving downwards, and the tap with the marble fountain moves up throughout the considered sequence.

Figure 1(a) presents a result of feature detection on the first frame and tracking by the Lucas-Kanade algorithm with recommended settings, yet without filtering. As can be seen, the fountain presents many points that are considered as convenient for tracking, whereas the Shi-Tomasi detector proposes no key point in the background, even though visually good (but unsharp) edges are available on the bench and lamppost. From the visual perspective, several severe misdetections accumulate during the processing.

The result of our method, presented in Figure 1(b) shows features tracked in this frame with rather short history lines.



(a) Kanade-Lucas-Tomasi on all frames without cross checking and feature renewal



(b) Active motion vectors detected by ORB matching with history, proposed by our algorithm

Fig. 1: Visual comparison of motion estimation. For full resolution, please refer to http://github.com/petrpulc/orb-flow/ tree/master/img

We assume that this is mainly caused by the overly precise matching of the point descriptors, which is however crucial for the stability of the registration algorithm. When all gathered timelines are overlaid, the combined history length is comparable. Our proposed algorithm is, however, capable of considering new points of interest as soon as they appear without a need to wait for lost features. Due to that, several good features are detected on the bench, providing at least some information about the background movement.

Our other goal was to make the motion estimation possible in a higher resolution video in real time. According to the results presented in Table I(b), we are currently able to detect motion on circa 350 newly detected interest points for each frame in 1440p resolution. Higher parallelism (e.g. in the feature point matching stage) would lead to higher utilisation of available resources and possibly even better results.

The processing time of Kanade-Lucas-Tomasi (see Table I(a)), is dependent almost only on the resolution, which indicates a slow detection of key points. If the detection is not carried out on every frame, the time required for processing is appropriately reduced. However, the problem of connecting segmented motion time series emerges.

In our implementation, the motion detection speed can be increased by decreasing the lost point threshold. This, however, results in detection of many shorter sequences that would need to be reconnected in further processing. Darker green represents processing above 30 fps, lighter green above 25fps.

(a) Kanade-Lucas motion estimation with Shi-Tomasi point of interest detection in each frame

						Verti	cal resolu	ution				
Points	180	360	540	720	900	1080	1260	1440	1620	1800	1980	2160
100	2.86	8.0	16.01	27.8	43.03	62.79	84.3	106.87	137.56	167.6	203.23	246.07
200	3.27	8.47	16.83	28.25	43.98	63.57	84.61	107.06	133.76	164.99	201.39	245.36
300	3.6	8.78	16.86	28.62	43.94	63.03	84.8	109.33	135.98	165.48	201.85	246.09
400	3.58	9.16	17.53	28.98	44.72	64.06	84.6	109.88	134.55	169.59	202.4	245.68
500	3.56	9.59	17.93	30.23	45.35	64.08	83.47	108.63	135.25	164.1	205.1	242.84
600	3.56	9.95	18.3	30.38	45.46	64.67	84.81	109.08	139.97	167.74	201.41	243.62
700	3.57	9.03	18.4	30.41	45.02	64.77	87.1	107.72	136.34	167.45	202.32	248.48
800	3.58	9.03	19.08	30.64	46.46	65.12	86.88	108.62	136.68	167.77	207.38	244.43
900	3.59	9.06	17.36	30.77	46.62	65.5	86.66	111.48	138.7	168.93	206.52	247.6
1000	3.56	9.05	17.57	31.71	46.34	66.9	87.77	112.43	135.08	171.98	203.19	253.53

(b) Our algorithm with the detection of interest points by ORB on each frame and descriptor matching

	Vertical resolution											
Points	180	360	540	720	900	1080	1260	1440	1620	1800	1980	2160
100	2.42	4.54	7.62	11.43	18.24	24.44	32.67	39.83	45.92	55.08	69.95	77.39
200	2.65	5.74	9.61	14.18	19.4	26.92	32.86	41.63	51.48	60.62	68.93	78.53
300	2.87	6.75	11.93	17.32	23.53	30.86	38.1	47.13	54.38	65.07	77.42	84.01
400	3.02	8.14	14.48	20.27	28.07	34.91	40.86	51.47	60.89	68.53	80.15	86.53
500	3.0	9.21	17.19	23.95	33.3	39.2	46.42	57.09	67.17	78.61	89.89	95.32
600	3.01	10.18	19.85	28.85	39.74	46.02	52.78	64.24	75.12	87.28	95.61	103.02
700	2.99	11.25	22.48	32.84	42.71	52.23	59.47	70.97	84.52	95.67	106.25	116.92
800	2.99	11.5	24.23	36.37	49.08	57.02	66.23	78.34	95.32	104.19	121.68	126.52
900	3.0	11.92	26.53	40.93	54.26	65.53	74.12	82.9	97.13	114.92	130.15	146.47
1000	2.99	12.2	28.65	46.52	58.58	72.27	84.01	92.25	111.85	127.17	144.6	152.49

TABLE II: Number of detected features by our algorithm during a 120 frame long sequence

	vertical resolution											
Points	180	360	540	720	900	1080	1260	1440	1620	1800	1980	2160
100	4186	6690	6369	7773	8098	7766	8034	7983	8089	8186	8141	8185
200	4117	9211	10774	14691	15692	13477	16084	15980	16559	16465	16605	16645
300	4614	9901	15577	20011	23007	18790	22798	23117	24293	24402	24810	25031
400	4504	11447	17824	21768	28857	23195	26167	29449	31814	31952	31897	32821
500	4544	13646	21827	24138	35209	28372	26089	35675	38383	39437	39369	40825
600	4544	15303	23188	27623	36086	33597	29883	39590	44788	46700	46878	48054
700	4544	16125	24480	28197	36883	36693	33786	39441	50117	51451	51938	55420
800	4544	16645	24374	31692	40790	40678	38020	41938	55291	58064	58996	60780
900	4544	17225	26527	34282	40792	43610	42742	44220	53890	63718	64920	67925
1000	4544	17811	29024	37062	40665	45416	46872	47854	54732	68837	70498	73681

In Table II, we present the number of detected sequences with an arbitrary history length in a sequence of 120 frames. With increasing resolution of the frame, the key point detector is less likely to select the same point and the more likely the feature becomes lost. Also, such features may be based on noise or not very robust image areas.

#### VI. CONCLUSION

In comparison to the baseline in real-time feature tracking, Kanade-Lucas-Tomasi algorithm, our approach is able to provide a sufficient number of tracked points of interest even on a FullHD or higher resolution without utilising GPU. The time sequences gathered from our motion estimation are, however, relatively short and may require additional processing.

Our future research will be, therefore, aimed mainly at improving the matching capabilities of our algorithm, resulting in longer time sequences. These sequences will be then more suitable for clustering of the gathered time sequences for the purpose of faster object segmentation and ultimately object detection and description.

# ACKNOWLEDGEMENTS

This reported research was supported by the Czech Science Foundation (GAČR), grant  $N^{\circ}$  17-01251. The work of Petr Pulc was also partially supported by by the Grant Agency of the Czech Technical University in Prague, grant  $N^{\circ}$  SGS17/210/OHK3/3T/18.

#### References

- M. Brown and D. G. Lowe, "Automatic panoramic image stitching using invariant features," *International Journal of Computer Vision*, vol. 74, no. 1, pp. 59–73, Aug 2007. doi: 10.1007/s11263-006-0002-3
- [2] Z. Yuan, P. Yan, and S. Li, "Super resolution based on scale invariant feature transform," in *International Conference on Audio, Language and Image Processing*, 2008. doi: 10.1109/ICALIP.2008.4590265
- [3] W. Zheng, H. Tang et al., Emotion Recognition from Arbitrary View Facial Images. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 490–503. ISBN 978-3-642-15567-3
- [4] A. Censi, A. Fusiello, and V. Roberto, "Image stabilization by features tracking," in *Proceedings 10th International Conference on Image Analysis and Processing*, 1999. doi: 10.1109/ICIAP.1999.797671
- [5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems* 25, 2012, pp. 1097–1105.
- [6] S. Gould, J. Arfvidsson et al., "Peripheral-foveal vision for real-time object recognition and tracking in video," in *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, 2007.
- B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," *IJCAI'81*, pp. 674–679, 1981.
   [Online]. Available: http://dl.acm.org/citation.cfm?id=1623264.1623280
- [8] J. Shi and C. Tomasi, "Good features to track," in 1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Jun 1994. doi: 10.1109/CVPR.1994.323794. ISSN 1063-6919 pp. 593–600.
- [9] C. Harris and M. Stephens, "A combined corner and edge detector," in Proceedings of the Alvey Vision Conference. Alvety Vision Club, 1988. doi: 10.5244/C.2.23 pp. 23.1–23.6.
- [10] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, Nov 1986. doi: 10.1109/TPAMI.1986.4767851
- [11] S. M. Smith and J. M. Brady, "Susan—a new approach to low level image processing," *International Journal of Computer Vision*, vol. 23, no. 1, pp. 45–78, May 1997. doi: 10.1023/A:1007963824710
  [12] E. Rosten and T. Drummond, "Fusing points and lines for high perfor-
- [12] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking," in *IEEE International Conference on Computer Vision* (*ICCV*), 2005. doi: 10.1109/ICCV.2005.104. ISSN 1550-5499
- [13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov 2004. doi: 10.1023/B:VISI.0000029664.99615.94
- [14] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," 9th European Conference on Computer Vision (ECCV 2006), pp. 404–417, 2006. doi: 10.1007/11744023\_32
- [15] E. Rublee, V. Rabaud *et al.*, "Orb: An efficient alternative to sift or surf," in 2011 International Conference on Computer Vision, Nov 2011. doi: 10.1109/ICCV.2011.6126544. ISSN 1550-5499 pp. 2564–2571.
- [16] E. Mair, G. D. Hager *et al.*, "Adaptive and generic corner detection based on the accelerated segment test," *11th European Conference on Computer Vision (ECCV)*, 2010. doi: 10.1007/978-3-642-15552-9\_14
- [17] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2011. doi: 10.1109/ICCV.2011.6126542
- [18] A. Canclini, M. Cesana *et al.*, "Evaluation of low-complexity visual feature detectors and descriptors," in 2013 18th International Conference on Digital Signal Processing (DSP), July 2013. doi: 10.1109/ICDSP.2013.6622757. ISSN 1546-1874 pp. 1–7.
- [19] P. Pulc, E. Rosenzveig, and M. Holeňa, "Image processing in collaborative open narrative systems," in *Fourth International Workshop on Computational Intelligence and Data Mining (WCIDM 2016)*, vol. 1649. CEUR, 2016. ISSN 1613-0073 pp. 155–162.
- [20] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *In VISAPP International Conference on Computer Vision Theory and Applications*, 2009. doi: 10.5220/0001787803310340 pp. 331–340.
- [21] S. S. Beauchemin and J. L. Barron, "The computation of optical flow," ACM Comput. Surv., vol. 27, no. 3, pp. 433–466, Sep. 1995. doi: 10.1145/212094.212141
- [22] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, Sep. 1975. doi: 10.1145/361002.361007
- [23] K. C. Zatloukal, M. H. Johnson, and R. E. Ladner, "Nearest neighbor search for data compression," in *Data Structures, Near Neighbor Searches, and Methodology*, 1999.
- [24] J. L. Blanco-Claraco, "nanoflann," 2011. [Online]. Available: https://github.com/jlblancoc/nanoflann