FEATURE SELECTION WITH GEOMETRIC CONSTRAINTS FOR VISION-BASED UNMANNED AERIAL VEHICLE NAVIGATION

Maria E. Angelopoulou and Christos-Savvas Bouganis

Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2BT, UK
email: {m.angelopoulou, christos-savvas.bouganis}@imperial.ac.uk

ABSTRACT

Vision-based egomotion estimation can be employed to endow with navigation ability an Unmanned Aerial Vehicle (UAV) that is equipped with an on-board camera. The egomotion estimation block computes the 3D UAV motion, taking as an input a 2D optical flow map that is constructed for each of the captured video frames. This work considers sparse optical flow estimation, and thus the navigation system that is developed includes a feature selection unit, which initially identifies the points of the optical flow map. This paper demonstrates that the feature selection process, and in particular the geometry of the selected feature set, decisively determines the overall system performance. Various computation schedules, which combine geometric constraints with a textural quality metric for the image features, are thus investigated. This paper shows that imposing appropriate distance constraints in the feature selection process significantly increases the output precision of the egomotion estimation unit, thus enabling accurate vision-based UAV self-navigation.

Index Terms— feature selection, optical flow, egomotion estimation, Unmanned Aerial Vehicle (UAV), self-navigation

1. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) are highly suitable when aerial operations are required, but the presence of a pilot is dangerous, impossible or simply expensive [1, 2]. This pertains to a wide range of applications, such as surveillance, search and rescue, aerial mapping, and defence operations.

To provide a useful 'eye in the sky', an UAV is typically equipped with an on-board camera. In the captured video frames, the points of the 3D real-world scene are projected on a 2D plane. The motion between adjacent frames can be described with a 2D motion field, known as the *optical flow map*. The inverse problem of extracting the 3D camera motion from the 2D optical flow map is known as egomotion estimation [3]. An on-board egomotion estimation processing unit that is mounted on an UAV enables its self-navigation.

This work is co-funded by the Technology Strategy Board. The video sequence of the real-data experiment is captured by a digital camera that is mounted on the Barnard Microsystems InView unmanned aircraft.

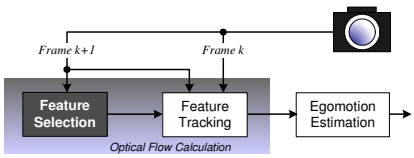
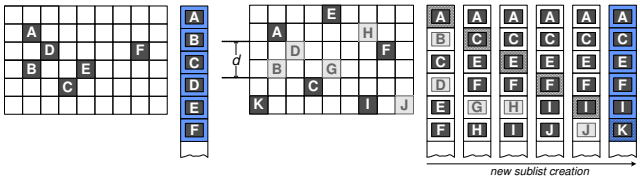


Fig. 1. The vision-based system for UAV self-navigation.

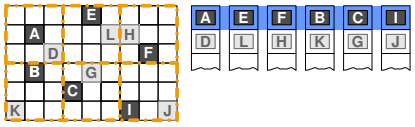
The optical flow map can be either *dense* or *sparse*, depending on whether a 2D motion vector is computed on every pixel. When sparse optical flow is considered, as is the case in this work, a feature selection process precedes the actual tracking that calculates the 2D motion vectors. In this manner, the tracker takes as an input a subset of distinctive image features that are 'good' to track [4, 5, 6]. This reduces the computational load and excludes from the calculations nondistinctive features that may disorientate the tracking algorithm.

As shown in Fig. 1, the complete vision-based UAV self-navigation system consists of the *Optical Flow Calculation* and the *Egomotion Estimation* units. The former is further divided in two processing blocks. These are the *Feature Selection* block [4], which identifies the distinctive features on the reference frame, and the *Feature Tracking* block [6, 7], which tracks those features on the adjacent frame to estimate the 2D optical flow that describes the inter-frame motion.

Occupying the beginning of the processing chain, the *Feature Selection* (FS) block determines the course of further processing and decisively affects the precision of the estimated 3D motion. In the literature, FS methods identify distinctive features based on a textural quality metric, such as the minimum eigenvalue of a 2×2 matrix containing sums of derivatives [4]. This paper demonstrates that when FS is executed as part of an egomotion estimation system, not only the textural properties of the features but also their geometry should be taken into account when forming the selected feature set. This produces an informative set of distinctive image features that are evenly distributed all over the frame, which significantly increases the precision of the estimated 3D motion. In the sections that follow, three possible FS computation schedules are identified and discussed, and their effect on the performance of egomotion estimation is evaluated, for certain distance constraints and parameter values.



(a) Brute-Force FS (b) Dynamic Block-Based FS, $d = 2$



(c) Static Block-Based FS

Fig. 2. Visualizing the three FS computation schedules, for $N = 6$.

2. FS APPROACHES AND FEATURE GEOMETRY

With respect to the geometric constraints that can be imposed to the feature set, this section identifies three possible FS computation schedules and discusses their suitability for vision-based UAV self-navigation. To assess feature textural quality, all schedules employ as a quality metric the minimum eigenvalue λ of matrix G_m [4]. For a certain pixel, G_m is computed on its $(2 \times \omega + 1)^2$ neighborhood, as follows:

$$G_m = \sum_{x=p_x-\omega}^{p_x+\omega} \sum_{y=p_y-\omega}^{p_y+\omega} \begin{pmatrix} X_{(x,y)}^2 & X_{(x,y)}Y_{(x,y)} \\ X_{(x,y)}Y_{(x,y)} & Y_{(x,y)}^2 \end{pmatrix} \quad (1)$$

where (p_x, p_y) are the pixel coordinates, and $X_{(x,y)}$, $Y_{(x,y)}$ denote the horizontal and vertical image derivatives at (x, y).

2.1. Brute-Force FS

In *Brute-Force FS* (BF), G_m is formed by computing (1) for each pixel, the λ values are sorted in decreasing order, and the N top features of the list comprise the selected set [4]. As shown in Fig. 2(a), the frame is globally treated, and a single sorting operation is executed that includes all of the pixels.

A drawback of BF is that, due to the absence of geometric constraints, the selected features may concentrate on a limited number of Regions of Interest (ROIs) of high textural quality. The computed motion vectors are thus overcrowded on these ROIs, leaving the rest of the frame void of motion information. Moreover, if the limited set of ROIs also includes regions with local irregularities, such as occlusions or inter-frame illumination variation, an increased number of inaccurate motion vectors is produced, dominating further processing and degrading the overall performance. For the above reasons, an unbalanced optical flow map comprises an input of the egomotion estimation unit that is less informative of the actual motion, thus decreasing the robustness of egomotion calculation (Sect. 3). Due to the limited height of flight of an UAV and the limited field of view of its on-board camera, the number of individual ROIs also tends to be limited. This renders feature overcrowding significantly common in UAV egomotion estimation, resulting in a navigation system that suffers from decreased output accuracy (Sect. 3).

2.2. Dynamic Block-Based FS

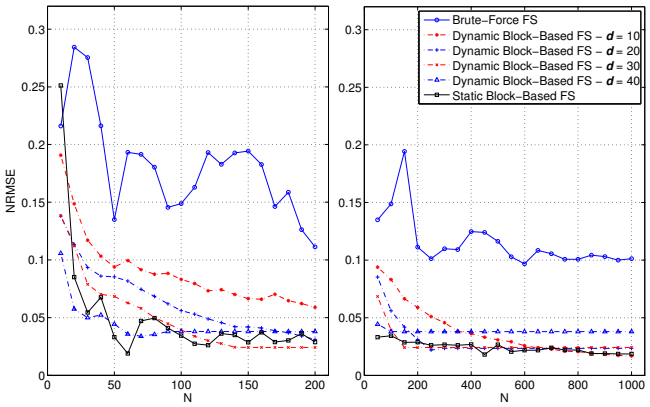
A *Dynamic Block-Based FS* (DBB) approach is next presented that tackles the drawbacks of BF. As Fig. 2(b) shows, the frame is globally treated, as in BF, but now multiple sorting stages occur that take feature location into account. The first stage considers feature A, which is at the top of the sorted list, and removes all lower-priority features within its $(2d + 1)^2$ neighborhood, *i.e.* B and D. A new sublist is produced, and the second higher-priority feature, C, is next considered in a similar manner. This process continues for N sorting stages, giving N sublists of decreasing length. The N top features of the final sublist comprise the selected set.

In DBB, N blocks of size $(2d + 1)^2$ pixels are considered. These are dynamically located on each frame, with their centers residing at the pixels that occupy the top N places of the final sorted sublist. This scheme can render features that are evenly distributed all over the frame and also exploit high textural quality, occupying top places in a list that is sorted based on textural criteria. The degree of 'feature scattering' depends on d, the minimum distance in pixels between adjacent features. Sect. 3 demonstrates how d affects the precision of the estimated 3D motion. A drawback of DBB derives from the fact that a λ value should be computed and stored for each and every pixel of the frame before the sorting operations begin. This limits the possible parallelism in the processing of FS and thus the performance for FS hardware implementation.

2.3. Static Block-Based FS

A *Static Block-Based FS* (SBB) scheme is next proposed, where the FS blocks have a fixed center. Each frame is statically segmented into N predefined blocks with overlapping contours. In particular, a border of $\omega + 1$ pixels is required to compute the derivatives and λ values at the block edges. From each block a single feature is included in the output set: the one with the best textural quality. Fig. 2(c) provides a simplified visualization of SBB for $N = 6$; no overlapping contours are shown. A sorted list is produced for each block, and the top features of these lists comprise the selected feature set.

The parallelism in feature selection can be now increased by a factor of N, since N blocks of pixels can be independently processed up to the end of FS operations, thus making the method suitable for real-time hardware implementation. A drawback of this approach is the fact that an individual FS block may not possess any significantly distinctive features, if its textural quality is low. To overcome the above, a threshold can be imposed to λ, and blocks whose best feature does not meet that threshold can be excluded from further processing.



(a) (b)

Fig. 3. Mean NRMSE of the estimated egomotion parameters for the indicated FS approaches, where (a) $N \in [10, 200]$ with a step of 10 and (b) $N \in [50, 1000]$ with a step of 50.

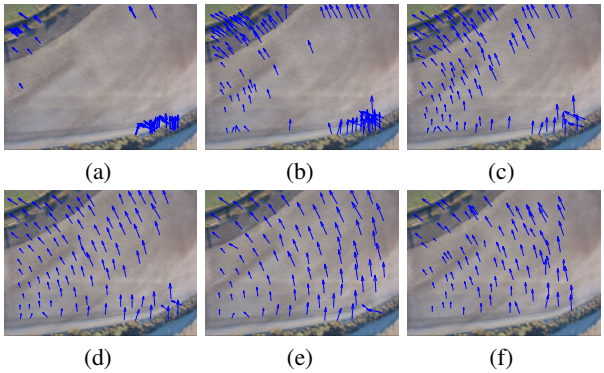
3. PERFORMANCE EVALUATION

This section evaluates the FS schedules of Sect. 2, to quantify their effect on the final egomotion estimation outputs, for certain distance constraints and parameter values. Feature tracking is implemented with multiresolution block-matching [7, 6], while egomotion estimation is executed through planar deformation determination [3]. The latter assumes constant depth in the camera coordinate system. This assumption is realistic only under certain constraints for normal navigation, but it is intrinsically highly suitable for plane navigation, due to the large distances between the camera and the scene. Six 3D motion parameters are rendered; (A, B, C) describe the translation motion component, β corresponds to a planar rotation around the optical axis, while θ and α describe an out-of-plane rotation that distorts the original camera coordinate system [3]. A ground-truth and a real-data experiment are performed. The test sequences include VGA (480×640) frames.

The graphs of Fig. 3 demonstrate the mean of the *Normalized Root Mean Square Error* (NRMSE) for the egomotion parameters between their estimated and ground-truth values, for various FS computation schedules and values of N. The input of the ground-truth experiment comprises an 11-frame semi-synthetic test sequence, which has been generated by applying both rotational and translational motion, of various angles and magnitudes, on the first frame of the captured video sequence that is shown in Fig. 4. In this manner, the ground-truth egomotion parameters are known and are used to estimate the error. As Fig. 3 shows, BF renders significantly higher NRMSE, compared to approaches that involve adequate geometric constraints. For $N \leq 100$, the DBB error decreases as d increases from 10 to 40. For larger N, increasing d beyond 30 does not further ameliorate accuracy. SBB error levels are similar to those of the optimal DBB schedules.

The above are confirmed with a real-data experiment,

Fig. 4. Setup for real-data testing. The Barnard Microsystems In-View unmanned aircraft is equipped with an on-board digital camera, and the captured video sequence is used in the tests.



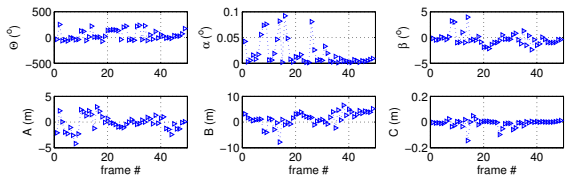
(a) (b) (c)

(d) (e) (f)

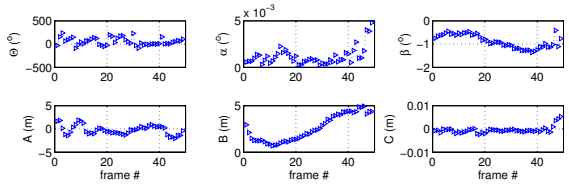
Fig. 5. Computed optical flow map for frame-pair $f_{44} - f_{45}$ of the real-data test, with $N = 100$ and (a) BF, (b-e) DBB with d equal to 10, 20, 30, 40, respectively, and (f) SBB.

which employs a 50-frame video sequence, captured by an UAV with an on-board camera (Fig. 4). Fig. 5 illustrates the calculated optical flow for $f_{44} - f_{45}$, $N = 100$, and various FS schedules. Fig. 5(a) demonstrates the unbalanced optical flow map that is rendered by BF, as a result of feature cluttering on a limited number of ROIs. Figures 5(b-e) show the gradual spreading of the motion vectors, as d increases from 10 to 40, for DBB. Fig. 5(f) shows their even distribution for SBB, which is, in addition, the hardware-friendly version of FS (Sect. 2). The optical flow of Fig. 5(f) facilitates not only egomotion estimation but also feature tracking, as its features are selected with textural quality criteria, leading to increased overall system performance as demonstrated next.

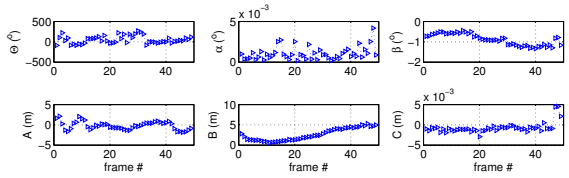
Fig. 6 shows the computed egomotion parameters, when applying the indicated FS approaches on the real-data sequence of Fig. 4, for $N = 100$. To map the translation parameters to meters, a focal length of 50×10^{-3} m and a mean UAV height of 150 m are considered. For each parameter set, Fig. 7 presents a flight simulation that visualizes the estimated UAV motion for the last 20 frames of the se-



(a) With BF

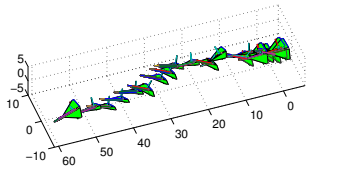


(b) With DBB, $d = 30$

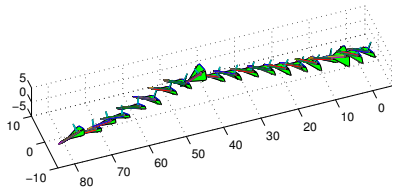


(c) With SBB

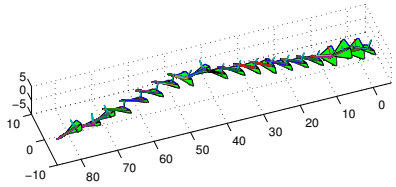
Fig. 6. Computed egomotion parameters for the real-data test, when employing the indicated FS methods and $N = 100$.



(a) With BF



(b) With DBB, $d = 30$



(c) With SBB

Fig. 7. Flight simulation for the parameters of Fig. 6, for frames $f_{31} - f_{50}$, when employing the indicated FS methods and $N = 100$.

quence, *i.e.* $f_{31} - f_{50}$. The UAV trajectory for BF (Fig. 7(a)) possesses unrealistic discontinuities, while its magnitude deviates significantly from the cases where distance constraints are included in FS. Such cases are shown in Figures 7(b,c), which illustrate smooth flight trajectories that are also in accordance with respect to their magnitudes. These demonstrate the robustness of egomotion estimation that is achieved when applying appropriate distance constraints, as in both DBB with $d = 30$ and SBB. The above observations are consistent with the low NRMSE values corresponding to these schemes in the ground-truth experiment (Fig. 3).

4. CONCLUSION AND FUTURE WORK

This work investigates how the inclusion of geometric constraints in the feature selection process affects the output accuracy of a vision-based egomotion estimation system for UAV self-navigation. Feature distance constraints distribute the vectors of the optical flow map evenly all over the frame, thus providing an informative input set for the egomotion estimation block. Such constraints are combined appropriately with textural feature selection criteria, to identify features of distinctive texture and thus facilitate the tracker. As results demonstrate, such a combination renders an improved feature set that enables both accurate feature tracking and egomotion estimation, hence significantly increasing the precision of the output 3D motion parameters, compared to brute-force FS approaches. Future work includes the development and integration of methods that perform saliency detection for indicating good candidate points [8], as well as methods that effectively identify outliers among the 2D motion vectors.

5. REFERENCES

[1] B. Sinopoli, M. Micheli, G. Donato, and T.J. Koo, "Vision Based Navigation for an Unmanned Aerial Vehicle," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2001, vol. 2, pp. 1757 – 1764.

[2] N. Guenard, T. Hamel, and R. Mahony, "A Practical Visual Servo Control for an Unmanned Aerial Vehicle," *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 331–340, Apr. 2008.

[3] C. Jonchery, F. Dibos, and G. Koepfler, "Camera Motion Estimation Through Planar Deformation Determination," *Journal of Mathematical Imaging and Vision*, vol. 32, no. 1, pp. 73–87, Sep. 2008.

[4] J. Shi and C. Tomasi, "Good Features to Track," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 1994, pp. 593–600.

[5] R.T. Collins, Y. Liu, and M. Leordeanu, "Online Selection of Discriminative Tracking Features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1631–1643, Oct. 2005.

[6] A. Yilmaz, O. Javed, and M. Shah, "Object Tracking: A Survey," *ACM Computing Surveys*, vol. 38, no. 4, pp. 1–45, Dec. 2006.

[7] J.-Y. Bouguet, "Pyramidal Implementation of the Lucas Kanade Feature Tracker: Description of the algorithm," Microprocessor Research Labs, Intel Corporation, 2002.

[8] Y. Liu, C.-S. Bouganis, and P.Y.K. Cheung, "A Spatiotemporal Saliency Framework," in *IEEE International Conference on Image Processing (ICIP)*, Oct. 2006, pp. 437–440.