Void detection for UAV based on optical flow and vanishing points

Débora N.P. Oliveira * **Davi J.G. Sousa** * **Marcos R.A. Morais** ** **Antonio M.N. Lima** **

* *Electrical Engineering Graduate Program, Universidade Federal de Campina Grande (debora.oliveira, davi.j.g.sousa@ee.ufcg.edu.br).*
** *Department of Electrical Engineering, Universidade Federal de Campina Grande (morais,amnlima@dee.ufcg.edu.br).*

Abstract: Indoor micro aerial vehicle (MAV) navigation is mostly based on high-computational cost obstacle detection algorithms. In this paper, we propose a path planning framework based on perspective cues. The vanishing points were detected by using the Hough-Canny transform. Due to the cluttering, multiple vanishing points candidates are arranged according to the proximity to the optical flow focus of expansion. Obstacles' depths are detected via the optical flow vector clustering. When multiple planes are recognized, the elected vanishing point is considered the center of the furthest empty plane from the camera. This remark guides the direction of the MAV to a particularly free-of-obstacles void area. Preliminary experimental results indicate that the proposed method is faster than other visual-based navigation algorithms.

Keywords: image processing, navigation, obstacle avoidance, vanishing point, optical flow.

1. INTRODUCTION

The lightweight and small size make the micro aerial vehicle (MAV) the most suitable agent for exploring and surveying missions in cluttered environments in which one cannot use the GPS, like indoor settings. In these conditions, it's required to address navigation techniques that create a stable and free of obstacles trajectories for a MAV to reach its target.

In-home environments, the GPS signal lacks precision or is unrecognizable (Granillo and Beltran, 2018; Zhou et al., 2015). Moreover, laser sensors such as LIDARs, which are bulky and power-consuming, are only suitable for heavy-load MAVs that support the battery weight (Prophet et al., 2017). Thus, for in-home applications, a vision-based solution is a feasible alternative that does not require any other accessory devices. As an example, Zhou et al. (2015) shows how to navigate a MAV through open windows by using features cascade classifier for stereo images. Other vison-based navigation and localization UAV applications are discussed in Al-Kaff et al. (2018).

Moreover, vision sensors support MAV navigation in a 3D space, despite capturing 2D data. Heng et al. (2015); Youn et al. (2020); Sa et al. (2013); Moura et al. (2021); Wang et al. (2020) use simultaneous visual localization and mapping (SLAM) to build a three-dimensional map of the room and to indicate non-obstructed trajectories. Nonetheless, this approach is limited to environments with distinct feature points visible between frames. Such a solution is not appropriate to surfaces that devoid of trackable features, such as walls (Bills et al., 2011).

Another vision-based pose estimation method is the recognition of landmarks signals, such as QR code-based (Chie and Juin, 2020) or Haar-like features (Nakamura et al., 2016). Howbeit, both SLAM and landmark identification require reliable odometry and intrinsic parameters of the camera, such as focal distance and distortion coefficients.

To refine vision-based pose estimation, Pestana et al. (2015); Chowdhary et al. (2013); Zhang et al. (2018); Niu et al. (2021) rely on inertial measurement unit (IMU) data fusion trough extended Kalman filter (EKF) or particle filter (Gronwall et al., 2017). Regardless of their robustness, these techniques bound to non-commercial MAVs due to the need for access to an IMU or the estimator states.

Furthermore, trajectory tracking algorithms must follow fast dynamic rates to guarantee the autonomy of the MAV in non-supervised conditions. Considering the aim to guide the drone to a static void, it's satisfactory to avoid solely the obstructions between the current position and the aim point. Hence, recognizing all objects in the captured scene, as in SLAM and data-fusion, processes non-useful data. A simpler approach can be applied using low-complexity perspective cues, such as vanishing points (Yu and Zhu, 2019) and optical flow (Horn and Schunck, 1981).

In this work, we rely on algorithms that instruct the best MAV's direction of navigation based on perspective cues. Firstly, we extract optical flow vectors and vanishing points candidates from a pair of images captured via a monocular camera. These points are arranged using the focus of expansion from the optical flow field to estimate the furthest empty plane from the camera to which the linear trajectory from the quadcopter is free of obstacles.

This paper is structured in the following sections: in section 2, we describe the computation of the suggested method, while the experimental results and conclusion are detailed in sections 3 and 4, respectively.

2. PROPOSED FRAMEWORK

Vanishing points (VP) are geometric single image perspective cues that have lower computational-cost than visual SLAM and have already been used to navigate quadcopter through hallways (Bills et al., 2011; Padhy et al., 2019). Within empty spaces, it is simple to identify wall and floor boundaries. However, it is challenging to identify the wall limits within cluttered environments due to the foreground obstructions. In such circumstances, likewise (Yu and Zhu, 2019) one can use motion relationship acquired through optical flow to vote for the most adequate VP candidate.

The proposed algorithm for commercial MAV obstacle avoidance in GPS-denied cluttered indoor scenes consists of four steps: image pre-processing and feature extraction, optical flow computation and field clustering, vanishing points extraction and vanishing point voting. The workflow of the algorithm can be seen in Fig. 1.

The integration of the proposed method to the MAV's control loop is illustrated in Fig. 2. The speed of the propellers is updated synchronously at the same rate as the inner control loop. On the other hand, image capturing for the outer loop control occurs at each t_k. Hence, the second image is captured at t_{k+1}, viz. an instant subsequently. Hence, the velocity of the drone between t_k and t_{k+1} must be slow enough to capture non blurred images. The blocks *I2L* and *L2I* represent the transformation of the coordinates on the inertial reference system to the image plane parallel to the lens, and vice-versa.

The path planner yields the next reference position (x_0, y_0, z_0) to the control loop without considering the obstacles. This trajectory is thus refined to (x^*, y^*, z_0) by using the void detection algorithm, which indicates the preferred direction of navigation on image plane coordinates (x_d, y_d). The altitude of navigation z_0 remained constant, and thus propagated. In the case of non-valid direction (x_d, y_d) indicated by the detection algorithm, the selection switch S can bypass the path planner and a new reference position is supplied to the control loop.

In this paper, we need access to the coordinates provided by the path planner to the controller. The sensors' signals, such as the ones issued by the barometer, accelerometer,

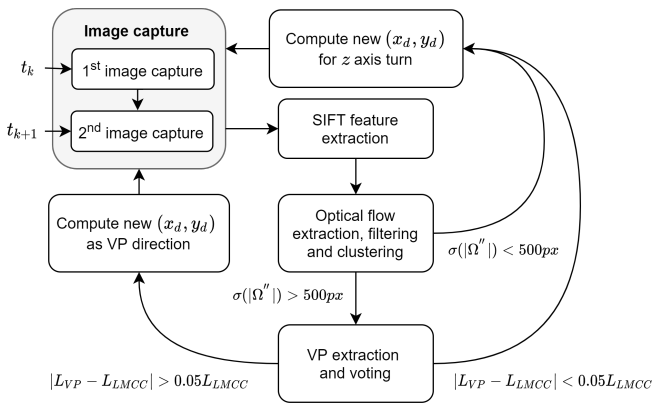


Figure 1. Structure of the proposed workflow. The output is the coordinates (x_d, y_d), which indicates the direction to the furthest empty plane from the camera.

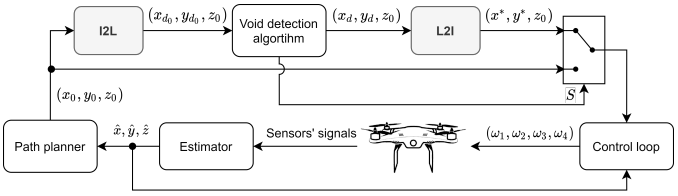


Figure 2. Architecture of the MAV's control and navigation loop.

gyroscope, and ultrasound, are not used in the void detection algorithm. As in many commercial drones, these signals are private data; hence indicated in the flow only for understanding purposes. The void detection algorithm steps are described in the following sections.

2.1 Image pre-processing and feature extraction

One can compare two sequential images' features to estimate a moving object's instantaneous velocity using the optical flow method. The basic assumptions are that voxels and pixels do not change for these two images $I(t)$ and $I(t + \Delta t)$ of the same object (Horn and Schunck, 1981). In this paper, we collected SIFT (Scale-Invariant Feature Transform) features, as they are sparser than ORB (Oriented fast and Rotated Binary robust independent elementary features) and BRISK (Binary Robust Invariant Scalable Keypoints) features and more computationally efficient than SURF (Speeded Up Robust Features) (Tareen and Saleem, 2018).

After SIFT features extraction, if less than 100 points are recognized in at least one of the two images $I(t)$ or $I(t + \Delta t)$, the algorithm prevents an empty optical flow field after the outlier filter — as described in the section 2.2. redThis avoidance is done through the selection switch (S), which allows one to bypass the path planner position output as a new reference point to the controller when an obstacle appears in picture and no non-obstructed vanishing point is found.

2.2 Optical flow computation and field clustering

For movement extraction, we applied Lucas and Kanade (1981) optical flow method. This algorithm provides the coordinates of the feature points matched between adjacent frames. The pair of coordinates of a matched pixel in the first and second frame compose an optical flow vector. All these optical flow vectors identified between the two adjacent frames constitute the optical flow field.

The obtained optical flow vectors are filtered by distribution, length, and angular analysis and clustered to maximize processing speed and accuracy. The initial set of N optical flow vectors is indicated as $\Omega = \{l_1, l_2, ..., l_N\}$.

Distribution filtering Firstly, we execute the Lucas-Kanade algorithm on the SIFT features from frame 1 to frame 2 $(\Omega_{1,2})$ and vice-versa $(\Omega_{2,1})$. Then, the optical vectors that refer to the same feature, and are spatially distant more than half pixel are considered outliers — i.e $\Omega^{'} = \{l_i \mid -0.5 < l_i(2,1) - l_i(1,2) < 0.5\}$, for $i = 1, 2, ..., N$; $N^{'}$ denotes the number of lines in $\Omega^{'}$.

Length filtering Lower magnitude optical flow lines usually refer to non-trackable feature surfaces, such as walls, light sources, or patterned surfaces. Thus, we assume that, in a set of flow vectors ordered by ascending magnitude, the first 10% items are outliers. Hence, the output set is defined as $\Omega^{''} = \{l_k, l_{k+1}, .., l_{N'}\}$, for $k = 0.1N^{'}$.

Angular filtering In virtue of the same discernment of the length filtering, we treat as outliers the vectors, which angle α to the horizontal line is outside the interval $p\sigma$ of the mean value α_m of the distribution, for σ the standard deviation of $\alpha_i = \angle l_i, \forall l_i \in \Omega''$. The value p is defined as the first integer in the range $[1, 9]$ that represents a decay of 10% of the outlier quota. Hence, the final set is denoted as $\Omega^{'''} = \{l_i \mid -p\sigma < \alpha_i - \alpha_m < p\sigma\}$, for $i = 1, 2, ..., N^{''}$.

Clustering To minimize the computation-cost, the optical flow vectors are clustered through k-means algorithm by spatial and angle distribution. The number of clusters n is also automatically defined as the first integer in the range $[1, 15]$ that represents the biggest Calinski-Harabasz (CH) index and the lowest Davies-Bouldin (DB) index.

Due to perspective geometry, when a camera capturing multiple-planes scene moves, the furthest plane — i.e. background — moves less than the foreground objects. Based on this analysis, we consider that the outermost point must be near the lowest magnitude optical flow vector coordinates (LMCC) in the second frame.

The length of the vector at LMCC is estimated through k-nearest neighbours algorithm. The nearest neighbours number value is defined as the first integer k, for $k = 5i$ and $i = 1, 2, ..., 8$, that represents a variance of 5% of the estimated clusters magnitude. If the standard deviation of the optical flow vectors of each cluster center is smaller than 500 pixels, we consider that the captured scene has only one plane — please note that this value must be changed according to the camera's resolution. Hence, the MAV assumes the default behaviour for wall avoidance, such as turning about it's z axis.

2.3 Vanishing points extraction

The vanishing points candidates of the second frame are computed using the Canny edge detector and Hough Probabilistic Transform (HPT) (Matas et al., 2000) to identify the long lines. The set of N extracted segments by using the HPT is indicated as $\Gamma = \{l_1, l_2, ..., l_K\}$ (Yu and Zhu, 2019), to which β_i denotes the angle of $l_i \in \Gamma$ to a horizontal line, and thus $\beta_i \in [0°, 180°]$.

Finding long lines in cluttered environments is a challenging task, so a region of interest (ROI) is defined to the lower half of the image, as it contains the floor lines. This approach avoids processing uppermost crowded scene (i.e. shelves, frames), and thus decays computational cost.

Because of foreground obstacles in a cluttered environment, it's impossible to assert that all identified lines are in the same plane. As each cluster represents a plane, we organized the lines in n groups by the euclidean distance to the cluster centers. Since all the lines in the same cluster are on the same plane, their intersection point represent a VP candidate (Bills et al., 2011).

However, not all lines intersect at the same point. Hence, Bills et al. (2011) searches for the highest density region of pairwise intersection. The image is divided into a 10x10 pixel grid and the intersections in each grid element are counted. The coordinates of the densest intersection element grid is defined as the VP candidate of the cluster.

In this work, we select one VP candidate per plane. As the search must conclude a VP on the horizon, we consider the new set of $K^{'}$ valid segments by $\Gamma^{'} = \{l_i \mid 40° < \beta_i < 140°\}$. In opposition to road or sidewalk recognition, within a cluttered environment, it is not always possible to identify the parallel lines on both sides of an object. Therefore, we only count the intersection between the pair of lines l_u and l_v to which $5° < |\beta_v - \beta_u| < 15°, \forall \beta^{'} \in [155°, 180°]$ for $\beta^{'} = \beta + 180°$ if $\beta < 90°$.

Let (x_k, y_k), for $k \in [1, K]$, be the intersection point of two lines $l_u, l_v \in \Gamma^{'}$, i.e. $y = m_u x + c_u = m_v x + c_v$. Furthermore, consider G a $(w/10) \times (h/10)$ matrix that represents the number of line intersections falling in the grid element (a, b), for w and h the width and height of the image Bills et al. (2011) — i.e:

$$G_{a,b} = \sum_{k=1}^{K} \left(\frac{x_k}{10} < a\right) \wedge \left(\frac{y_k}{10} < b\right) \tag{1}$$

Therefore, the grid with the maximum number of intersection is:

$$(a^*, b^*) = \arg\max_{a,b} G_{a,b} \tag{2}$$

The coordinate of the middle of the densest intersection element grid and VP candidate is:

$$(x^*, y^*) = 10(a^* + 0.5, b^* + 0.5) \tag{3}$$

2.4 Vanishing points voting

There are n or fewer VPs candidates, for n the number of clusters. In perspective geometry, the radial center of the optical flow field is the focus of expansion (FOE). As states Hashimoto et al. (1996); Guo et al. (2018), the FOE is equivalent to the VP. Therefore, the nearest VP candidate to the LMCC is elected the principal VP.

However, the VP can be hidden behind a foreground obstacle. Hence, the magnitude of the optical flow vector L_{VP} around the VP location and L_{LMCC} on the LMCC can be estimated using the same k-nearest neighbour algorithm described in Section 2.2. When $|L_{VP} - L_{LMCC}| < 0.05L_{LMCC}$, the VP candidate is elected as principal VP.

As previously said in Bills et al. (2011); Padhy et al. (2019), the principal VP direction (x_d, y_d) is the preferred direction of navigation for the MAV. In the case of non-valid VP candidates, the output (x_d, y_d) must point to the LMCC, as it is the nearest known markdown of the FOE. The validity criteria is that the OF vector mean magnitude around the selected VP must be less than 0.05% away from the magnitude at LMMC. These values are retrieved from the clustering obtained in section 2.2.

3. RESULTS AND DISCUSSION

To test the algorithm we applied the NavigationNet dataset (Huang et al., 2018). Four flat scenes were used to

simulate a MAV navigation: a wide room (A), a corridor (B), and a narrow door (C) and a dead-end room (D). Only the frontal camera images are utilized.

For image set A and C, the optical flow field before and after the filtering processes are illustrated in Fig. 3. Comparing Fig. 3a with Fig. 3b, one can clearly see the outlier removal on walls and windows. Furthermore, the difference between Fig. 3c and Fig. 3d highlights the angular filter rejection on patterned surfaces, such as curtains. Both comparisons demonstrate the filter efficiency while retaining movement information from hard edges — e.g. wall-floor and wall-obstacle boundaries.

The value of p and the number of dropped vectors on each filter step is described in Table 1. It's is important to note that these values are automatically generated as described in section 2, and thus not tuned ad hoc.

Table 1. Filter and cluster parameters per set.

Image set	p	n	k	$N^{'}$	$N^{''}$	$N^{'''}$
A	2	12	10	1725	1553	1482
B	3	13	10	331	298	298
C	3	12	10	1192	1073	1053
D	3	15	10	538	485	480

Fig. 4 depicts the length, angular, and cluster filter parameters' choice for image set A and C. Both Fig. 4a and Fig. 4b illustrate how the magnitude filter rejects the lower 10% length OF vectors, which commonly represent outliers. Subsequently, Fig. 4c and Fig. 4d show how the angular filter discards the lines which angles are outside the threshold boundary $p\sigma$ to the mean value α_m.

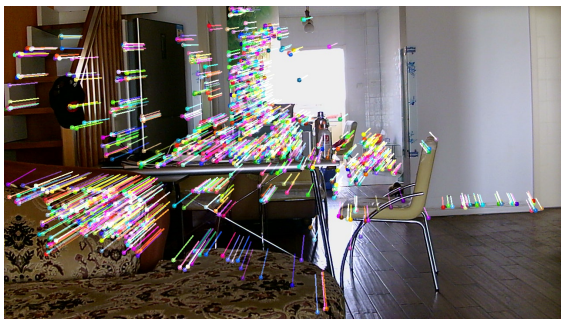
According to section 2, the value $p\sigma$ depends on the outlier quota, which according to Table 1, is 172 vectors for $\sigma = 2$ for set A and 33 vectors for $\sigma = 3$ for set C. On the condition that these drops represent less than 10% in relation to the number of outliers rejected with $\sigma = 1$ for set A and $\sigma = 2$ for set C, the iteration must stop in $\sigma = 2$ for set A and $\sigma = 3$ for set C, and the value of the boundary is chosen as shown in Fig. 4e and Fig. 4f. Likewise, Fig. 4i and Fig. 4j depict the higher CH and lower DB index for 12 clusters, which defines the parameter n. Lastly, Fig. 4g and Fig. 4h represent the selection of nearest neighbours value k, which stopped at the second iteration ($k = 10$) for both described dataset.

Comparing $N^{''}$ between the image sets A/C and B/D, it is evident that the efficiency of the proposed method does not depend on the total of optical flow vectors after filtering. However, the performance of the algorithm relies on the similarity and separation of the cluster centers.

The optical flow field, cluster centers and LMCC are shown in Fig. 5. Likewise, the number of clusters n and k nearest neighbours computed for each image set is described in Table 1. Despite clustering for X, Y, and slope data, it is clear that the cluster centers also aggregate the optical flow field in magnitude. Furthermore, one can notice that the LMCC locates on the lower magnitude region, which is highlighted by the darker hues in the color map.

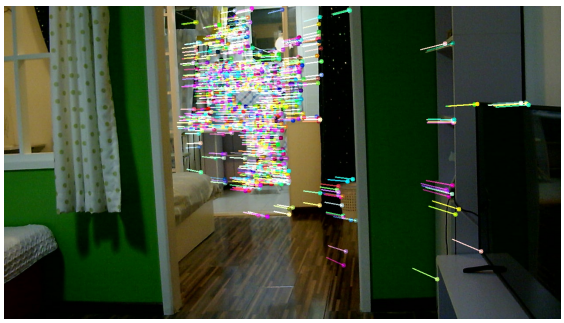
In Fig. 5d, one can see that major part of the optical flow vectors' magnitudes are below 60% of the maximum vector magnitude. This case is an example of a plane directly in front of the camera, which must be avoided, indicating (x_d, y_d) as a turn around the MAV's z axis.

(a)



(b)

(c)



(d)

Figure 3. Optical flow field vectors before (3a) (3c) and after (3b) (3d) filtering for image set A and C, respectively.

In Fig. 6 are illustrated the lines detected by using the Canny and Hough transform. The highlighted lines assert the angular rules described in Section 2.3, and thus are not approximately vertical nor horizontal. The parameters for the Canny and Hough transform were manually adjusted to achieve an average of 10 lines per image pair. Analysing

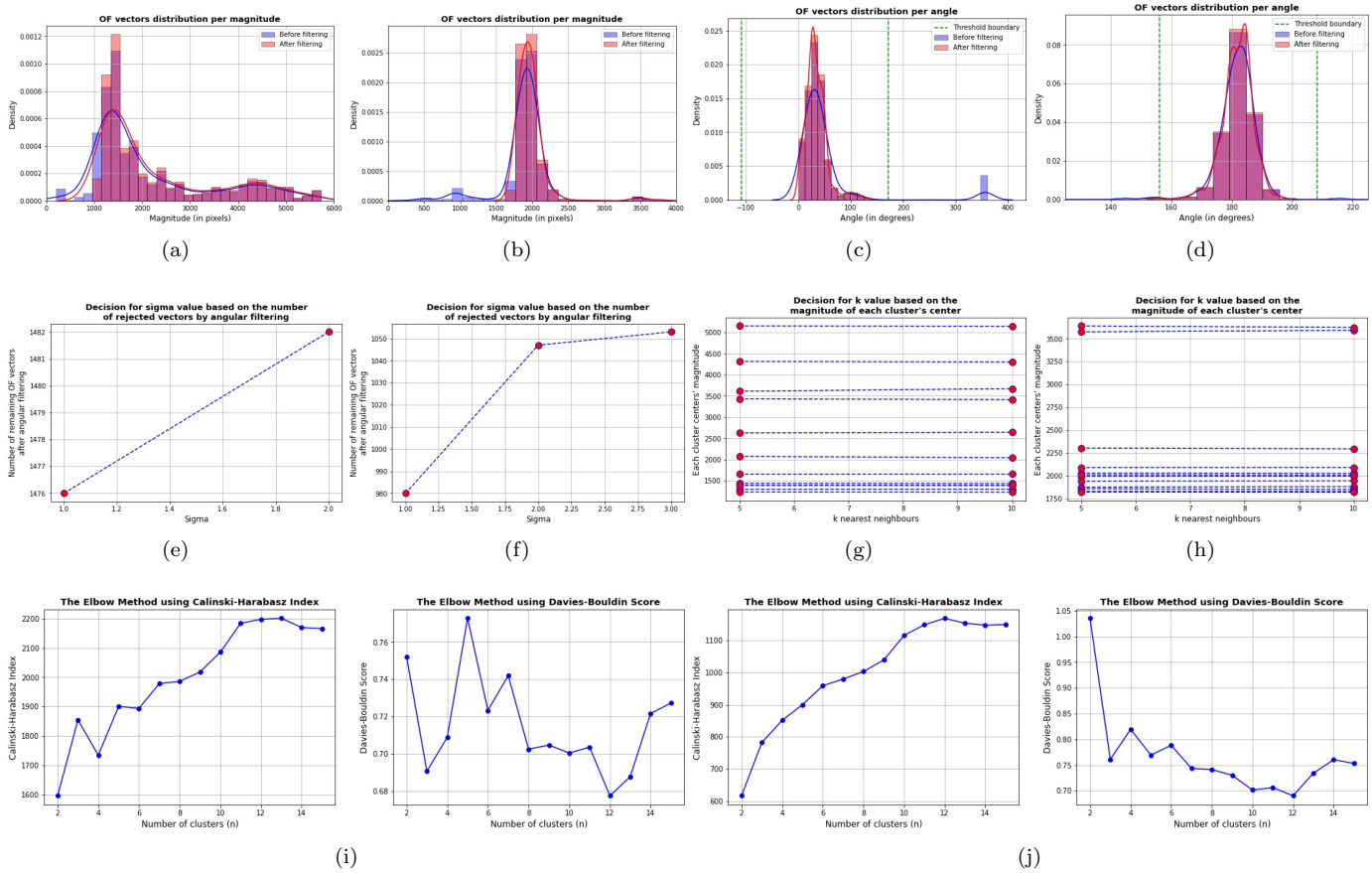


Figure 4. Magnitude filter (4a) (4b) and angular filter output (4c) (4d) for image set A and C, respectively. Decision of parameters angular filter threshold boundary σ (4e) (4f), k nearest neighbours (4g) (4h), and n number of clusters (4i) (4j) are also depicted, respectively, for image set A and C.

Fig. 6a, Fig. 6b and Fig. 6c, it is also geometrically evident that the FOE is near the closest VP to the LMCC. Furthermore, because of the ROI, one can see that on Fig. 6a and Fig. 6b mainly the floor lines are identified. This approach thus guarantees a better VP approximation.

For image set A, two out of three line groups are composed of approximately parallels lines. As states Section 2.3, the intersection of these lines are not considered in the VP extraction algorithm. Thus, only the VP of the remaining group is identified, as illustrated in Fig. 6a. For datasets A and B, the magnitude of the optical flow field at the VP coordinates is 0.05% away from the magnitude at LMMC. Therefore, in both cases, the VP (x_d, y_d) is elected as the navigation direction.

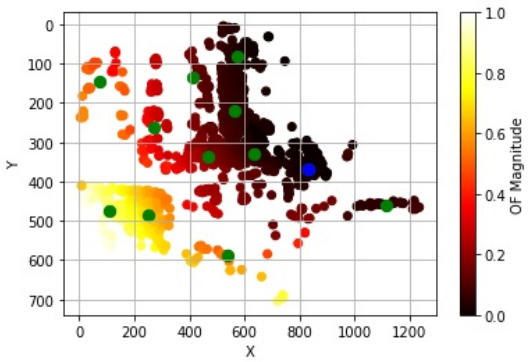
For image set C, the identified lines are arranged in three groups as shown in Fig. 6c. However, the magnitude of the optical flow field at the closest VP to the LMCC is more than 0.05% away from the magnitude at LMMC. Therefore, we consider that the VP is hidden behind a foreground object. In this case, the LMCC must be indicated as (x_d, y_d). In Table 2, one can see the cumulative time of each algorithm step and the total number of instruction calls obtained using deterministic profiler (Forbes, 2017) running in a Google Collab notebook. The parameters p, k and n were only estimated in the first void detection loop, as the environment remains static.

Table 2. Cumulative time and instruction calls of each proposed algorithm step per loop type.

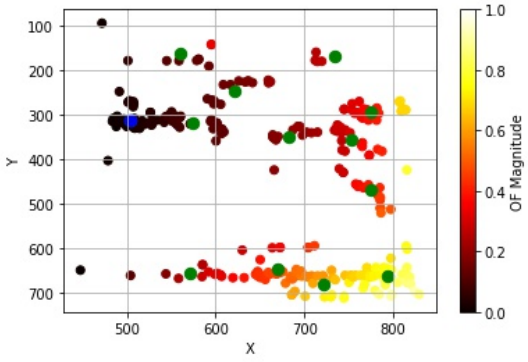
Section	Algorithm step	First loop	Other loops
2.1	Feature extraction	1.050s	1.025s
2.2	OF extraction and filtering	0.304s	0.297s
	OF clustering	3.869s	0.393s
2.3	VP extraction	0.119s	0.120s
2.4	VP voting	0.003s	0.003s
Total time		5.345s	1.838s
# instructions calls		1,234,247	131,354
# instructions calls / Total time		230kHz	71kHz

Comparing the columns of Table 2, it is evident the runtime increase owed to the parameters' estimation in the optical flow clustering step. One solution would be to reduce the test interval for n, p and k. Moreover, the control loop and void detection algorithm illustrated in Fig. 2 must not follow the same rate. If one can consider the environment static, no obstacle will obstruct the path between the MAV and the planned direction (x_d, y_d) between sampled updates. Therefore, the void detection algorithm can be executed exclusively if the MAV moves m meters in the (x^*, y^*) direction.

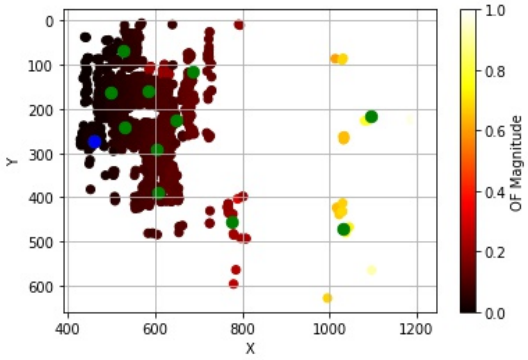
The equal profiler (Forbes, 2017) with a SLAM algorithm (Baggio et al., 2012) reports a 2.6s requirement. Despite this result being faster than the suggested algorithm's first loop, one can conclude that the proposed approach has a



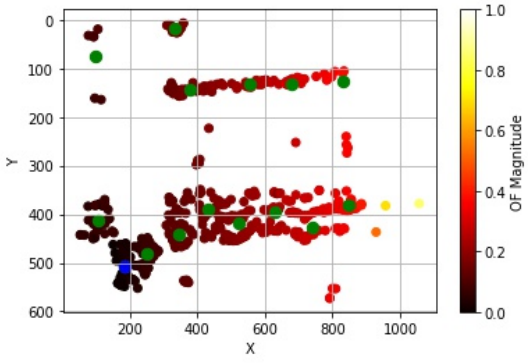
(a) Optical flow cluster for image set A



(b) Optical flow cluster for image set B

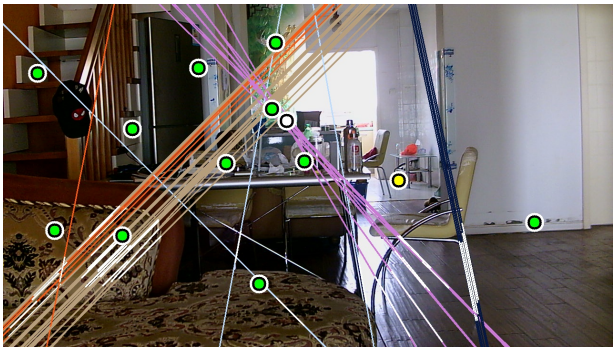


(c) Optical flow cluster for image set C

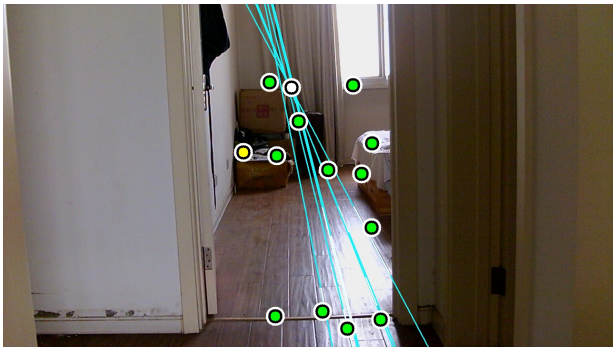


(d) Optical flow cluster for image set D

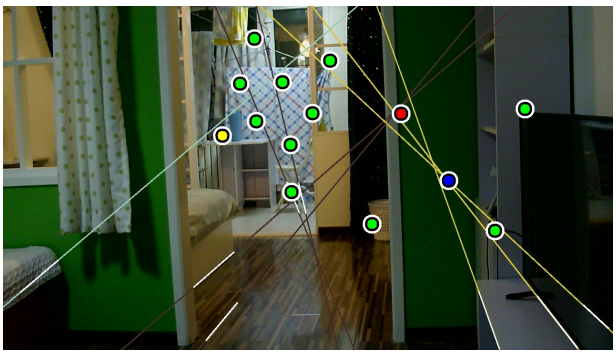
Figure 5. Optical flow field for set A (5a), B (5b), C (5c) and D (5d). The green scatter plot represents the cluster centers. The blue point indicates the LMCC.



(a) Image set A



(b) Image set B



(c) Image set C

Figure 6. Detected lines by using the Canny and HPT for image set A (6a), B (6b) and C (6c). The green scatter plot represents the cluster centers. The yellow, white, red and blue dots indicate, respectively, the LMCC, the elected VP, the rejected VP and the valid but non-elected VP.

lower computational cost for the following iterations. As long as the frequency of the embedded MAV's processor is known, one can use the rate of each loop type shown in Table 2 to estimate the run-time.

4. CONCLUSIONS

This paper proposes a flexible and efficient approach for embedded MAV navigation in a cluttered environment based on perspective cues, such as vanishing points. As experimentally verified, multiple VPs candidates can be determined by using Hough-Canny transform. Also, experimental results show that the true VP is near the radial center of the optical flow field, which is noisy and thus filtered among spatially distributed characteristics. As these outliers filter parameters are updated between scenes, the

proposed system is adaptive to the abundance of obstruction. The reasonable limits of the interval between captures are that the two images must non-blurred and share at least 100 matched features. Likewise, the sensor brightness sensitivity must be customized, so at least two intercepting lines visible after the Hough-Canny transform. As the optical flow and vanishing point extraction do not require camera parameters such as SLAM, the proposed technique is suitable for commercial MAVs. On the other hand, we must have access to the high-level control loop to replace the reference pose issued by the path planner with the one found by the void detection algorithm.

REFERENCES

Al-Kaff, A., Martín, D., García, F., de la Escalera, A., and Armingol, J.M. (2018). Survey of computer vision algorithms and applications for unmanned aerial vehicles. *Expert Systems with Applications*, 92, 447–463.

Baggio, D., Emami, S., Escrivá, D., Ievgen, K., Mahmood, N., Saragih, J., and Shilkrot, R. (2012). *Mastering OpenCV with Practical Computer Vision Projects*, chapter Exploring Structure from Motion Using OpenCV. Packt Publishing.

Bills, C., Chen, J., and Saxena, A. (2011). Autonomous MAV flight in indoor environments using single image perspective cues. In *Conf. Rec. IEEE/ICRA*.

Chie, L.C. and Juin, Y.W. (2020). Artificial landmark-based indoor navigation system for an autonomous unmanned aerial vehicle. In *Conf. Rec. IEEE/ICIEA*.

Chowdhary, G., Johnson, E.N., Magree, D., Wu, A., and Shein, A. (2013). GPS-denied indoor and outdoor monocular vision aided navigation and control of unmanned aircraft. *J. of Field Robotics*, 30(3).

Forbes, E. (2017). *Learning Concurrency in Python*, chapter Profilers. Packt Publishing.

Granillo, O.D.M. and Beltran, Z.Z. (2018). Real-time drone (UAV) trajectory generation and tracking by optical flow. In *Conf. Rec. IEEE/ICMEAE*.

Gronwall, C., Rydell, J., Tulldahl, M., Zhang, E., Bissmarck, F., and Bilock, E. (2017). Two imaging systems for positioning and navigation. In *Workshop Rec. IEEE/RAS/RED-UAS*.

Guo, X., Li, Q., and Sun, C. (2018). A new road tracking method based on heading direction detection. *J. of Automobile Eng.*, 233(2).

Hashimoto, H., Yamaura, T., and Higashiguchi, M. (1996). Detection of obstacle from monocular vision based on histogram matching method. In *Conf. Rec. IEEE/IECON/ICoIEI*.

Heng, L., Lee, G.H., and Pollefeys, M. (2015). Self-calibration and visual SLAM with a multi-camera system on a micro aerial vehicle. *J. Auton. Robots*, 39(3).

Horn, B.K. and Schunck, B.G. (1981). Determining optical flow. *J. Artificial Intelligence*, 17(1).

Huang, H., Shen, Y., and Sun, J. (2018). Navigationnet: A large-scale interactive indoor navigation dataset.

Lucas, B. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *Proc. 7th IJCAI*.

Matas, J., Galambos, C., and Kittler, J. (2000). Robust detection of lines using the progressive probabilistic hough transform. *Comput. Vision and Image Understanding*.

Moura, A., Antunes, J., Dias, A., Martins, A., and Almeida, J. (2021). Graph-SLAM approach for indoor UAV localization in warehouse logistics applications. In *Conf. Rec. IEEE/ICARSC*. IEEE.

Nakamura, T., Haviland, S., Bershadsky, D., Magree, D., and Johnson, E.N. (2016). Vision-based closed-loop tracking using micro air vehicles. In *Conf. Rec. IEEE/AeroConf*.

Niu, G., Zhang, J., Guo, S., Pun, M.O., and Chen, C.S. (2021). UAV-enabled 3d indoor positioning and navigation based on VLC. In *Conf. Rec. IEEE/ICC*. IEEE.

Padhy, R.P., Xia, F., Choudhury, S.K., Sa, P.K., and Bakshi, S. (2019). Monocular vision aided autonomous UAV navigation in indoor corridor environments. *IEEE Trans. Sustain. Comput.*, 4(1).

Pestana, J., Sanchez-Lopez, J.L., de la Puente, P., Carrio, A., and Campoy, P. (2015). A vision-based quadrotor multi-robot solution for the indoor autonomy challenge of the 2013 int. micro air vehicle competition. *J. of Intell. & Robotic Syst.*, 84(1-4).

Prophet, S., Scholz, G., and Trommer, G.F. (2017). Collision avoidance system with situational awareness capabilities for autonomous MAV indoor flights. In *Proc. IEEE/ICINS*.

Sa, I., He, H., Huynh, V., and Corke, P. (2013). Monocular vision based autonomous navigation for a cost-effective MAV in GPS-denied environments. In *Conf. Rec. IEEE/ASME/AIM*.

Tareen, S.A.K. and Saleem, Z. (2018). A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK. In *Proc. IEEE/iCoMET*.

Wang, H., Wang, Z., Liu, Q., and Gao, Y. (2020). Multi-features visual odometry for indoor mapping of UAV. In *Conf. Rec. IEEE/ICUS*. IEEE.

Youn, W., Ko, H., Choi, H., Choi, I., Baek, J.H., and Myung, H. (2020). Collision-free autonomous navigation of a small UAV using low-cost sensors in GPS-denied environments. *J. of Control, Automat. and Syst.*, 19(2).

Yu, Z. and Zhu, L. (2019). Roust vanishing point detection based on the combination of edge and optical flow. In *Workshop Rec. IEEE/IROS*.

Zhang, X., Du, Y., Chen, F., Qin, L., and Ling, Q. (2018). Indoor position control of a quadrotor UAV with monocular vision feedback. In *Conf. Rec. IEEE/Chinese Control Conference*.

Zhou, S., Flores, G., Bazan, E., Lozano, R., and Rodriguez, A. (2015). Real-time object detection and pose estimation using stereo vision. an application for a quadrotor MAV. In *Workshop Rec. IEEE/RAS/RED-UAS*.