Motion-based MAV Detection in GPS-denied Environments

Erik Vroon [*1], Jim Rojer [†2], and Guido C. H. E. de Croon [‡1]

[1]MAVLab, Control and Operations, Faculty of Aerospace Engineering, TU Delft, The Netherlands
[2]Netherlands Organisation for Applied Scientific Research (TNO), The Hague, The Netherlands

Abstract

Drones need to detect and localize each other if they are to collaborate in multi-robot teams or swarms. In this paper, a method based on dense optical flow (OF) is developed that detects dynamic objects. This is achieved by comparing the flow vectors with the direction to the Focus of Expansion (FoE) in the image plane. A simulation in AirSim is developed to validate this approach and to create a data set for motion-based dynamic object detection. This simulation includes ground-truth FoE, depth, OF and IMU data. The results show that our method performs well if the OF vector's magnitude is large enough and its angle is sufficiently different from those of static world points. We expect that the presented method will serve as a useful baseline for deep learning methods using dense optical flow as input.

1 Introduction

Nowadays, Micro Air Vehicles (MAVs) are becoming more and more common. Reasons for their popularity include their high maneuverability, vertical take-off capabilities and ability to perform tasks that humans cannot endure [1]. To further enhance the capabilities of MAVs and overcome the individual limitations of MAVs, swarms of MAVs were introduced. To enable the proper functioning of the swarm, sensing of the environment and the other MAVs is paramount. In particular, the relative locations of MAVs inside the swarm are needed for collision avoidance and swarm coordination [2]. The most basic and robust method of obtaining the locations of other MAVs, is by exchanging positions obtained from Global Navigation Satellite System (GNSS) signals. However, GNSS signals are not always available, for example when the signals are blocked, spoofed, jammed or distorted by multipath effects.

*Email address: erik.vroon22@gmail.com
†Email address: jim.rojer@tno.nl
‡Email address: g.c.h.e.decroon@tudelft.nl

Computer vision is a promising alternative, because cameras are small/lightweight and provide a vast amount of information [2]. There are two main types of approaches solving the relative localization problem. The first is to create a shared map of the environment and have the MAVs exchange their location in this map. Simultaneous Localization and Mapping (SLAM) is a wide field of research that targets the first type [3]. The second type of relative localization focuses on the detection of the MAVs themselves. This process is often simplified by the use of physical devices called markers. These can be either infrared (see the work of Walter et al. [4]) or ultraviolet (see the work of Roberts et al. [5]) LEDs, or colored objects. Markers require specific hardware changes to the device, which may not always be desirable. Markerless detection represents a more difficult problem. Some methods quite successfully rely on stereo vision [6, 7]. Of course, for resource minimization, methods using a single camera are of interest. Currently, the main approach with a single camera is to employ deep neural networks that detect other MAVs in a single image [8, 9]. These neural networks show promising results, but it is not yet clear how well the trained networks can deal with cluttered backgrounds. Moreover, if the drone appearance or environment changes substantially with respect to the training set, retraining may be necessary.

In order to obtain a solution that does not depend on stereo vision or markers and that is more generic compared to appearance-based methods, it may be useful to use optical flow. Optical flow has multiple advantages over its alternative vision-based methods. Firstly, MAVs will possibly be detected in situations where they are barely distinguishable for the human eye due to background clutter. Additionally, optical flow based methods are less dependent on shapes and appearances of MAVs compared to other methods. Finally, optical flow can offer a larger maximum detection range compared to active markers.

Some papers incorporate motion into their appearance-based neural network, the so-called hybrid methods, such as the work by Yoshihashi et al. [10], where the temporal information improves the performance of the object detector in situations where there is little contrast between the background and

foreground. Nonetheless, it is a ground-based method and uses static cameras, which is less challenging compared to the situation where the observer is moving.

To our knowledge, the research done by Li et al. [11] is the only work using purely optical flow without artificial neural networks for the detection of MAVs from a moving observer in the air. With their method, they were able to detect other MAVs, even when they were barely visible because of their size and the cluttered background. It has approximately the same detection accuracy (87%) as appearance-based neural networks applied to MAV detection (maximum accuracy of approximately 90%) [8, 12]. However, it is based on some assumptions. The method of Li et al. is based on a combination of background subtraction and Lucas-Kanade optical flow. The background subtraction process assumes that the tracked objects have a very different motion compared to a distant background, of which the motion is modelled with a homography transformation.

This paper focuses on motion-based object detection to detect MAVs from onboard a moving MAV in more general, 3D environments. Specifically, we present an optical-flow-based algorithm to detect dynamic objects in video feeds from a moving camera. This is done by comparing the flow vectors with the direction to the Focus of Expansion (FoE) in the image plane. This method is applied to simulations run in AirSim [13]. These simulations output ground-truth FoE, depth, optical flow and IMU data, which are valuable for the development and validation of motion-based object detection techniques. The proposed algorithm's image processing pipeline is mostly 'traditional', exploiting knowledge on the properties of the (derotated) optical flow field. We believe that eventually purely deep learning motion-based methods will achieve higher performance, but expect that the presented, completely comprehensible pipeline will be a useful benchmark method. Moreover, the results of our method show some of the challenges that will also be faced by deep learning methods, including difficult detection for small optical flow, flow directions similar to those of static world points, and the fact that other dynamic objects are not differentiated from MAVs in our current pipeline. On this last point, in this paper all moving objects are assumed to be MAVs, except for the clouds, which we detect with a deep neural network. To output only MAVs in an environment with other types of dynamic objects, the pipeline has to be extended to differentiate MAVs from other moving objects.

2 Detection method

The object detection method is illustrated in figure 2. First, the optical flow (OF) field is derotated using the rotation rates of the IMU. The location of the FoE is calculated using the derotated flow. FoE is the point where the translational flow is 0. This is the motion direction of the camera. All static points in the environment move away from the FoE. Points that are closer to the camera in terms of depth, have larger flow. Points that are further away from the FoE have larger flow as well. Dynamic objects may move in other directions. Then the associated flow vectors do not point away from the FoE. Unfortunately, they may move away from the FoE leading to flow that is similar to static objects. The angle κ between the vector pointing towards the FoE and the flow vector is calculated, as illustrated in figure 1. The larger κ, the more likely a pixel belongs to an object moving relative to the camera. In the following subsections, the individual steps of the method are explained in detail. All code used to reproduce this method and its results can be found publicly online[1].

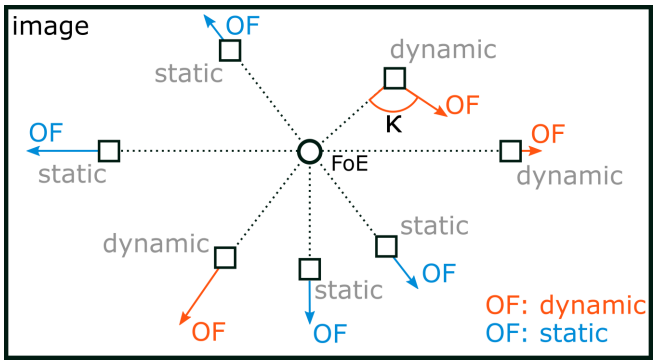


Figure 1: Illustration of κ for a camera moving forward. The κ angle denotes the difference between angle of the vector pointing at the FoE and the angle of the flow vector. For pixels of static objects, κ is approximately zero. For dynamic pixels, κ is non-zero, except when the object moves away from the FoE.

2.1 Calculating optical flow

As the object detection method relies on optical flow, an accurate dense optical flow estimator must be used. In figure 3, four neural networks estimating optical flow are compared. They illustrate that on the MIDGARD [14] dataset, LiteFlowNet [15] and Maskflownet [16] perform worse compared to RAFT [17] and FlowNet2 [18]. For all networks, the default weights were used. By visual inspection, FlowNet2 appears to perform best for small moving objects. Therefore, FlowNet2 is used for the results in the rest of this paper.

2.2 Derotation

Derotation has to be applied to the optical flow field to estimate an accurate FoE. The derotation technique in this paper is based on the work of Dinaux et al. [19]. The derotation vector per pixel coordinate can be calculated

[1] `https://github.com/evroon/mav-detection`

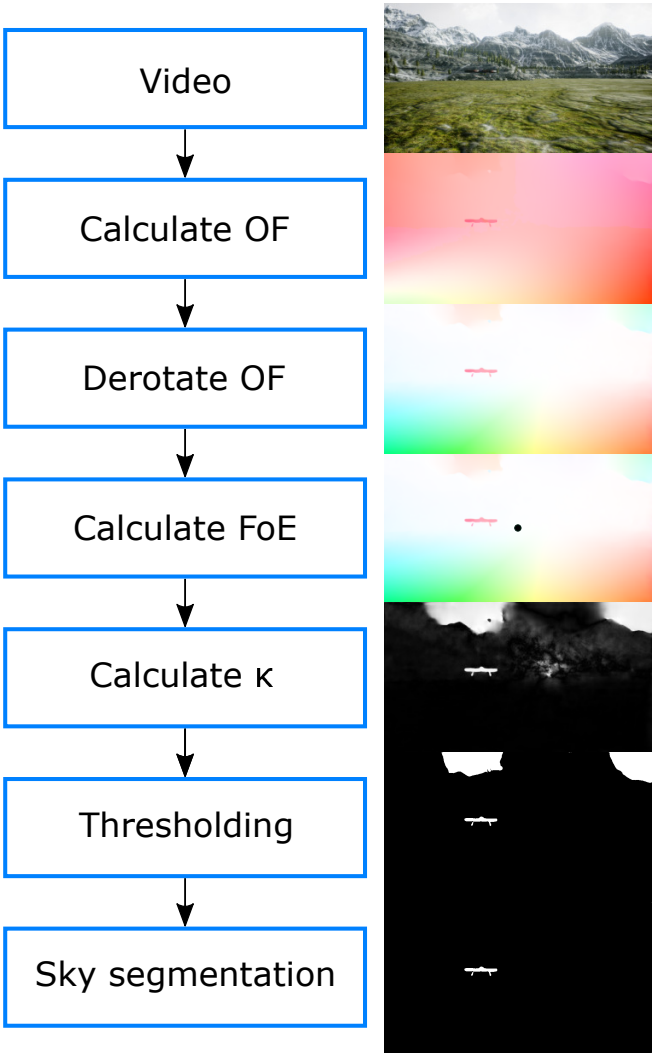
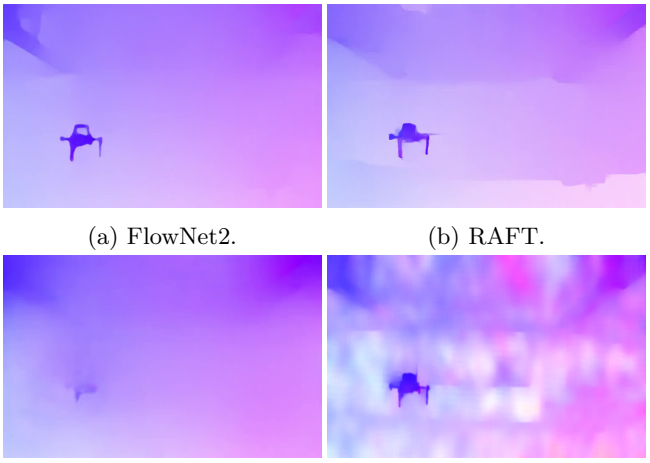


Figure 2: The proposed image processing pipeline.

from equation 1 describing optical flow (u, v) for a world point i in terms of ego-motion (U, V, W being the body velocities in X, Y, and Z direction and A, B, and C the rotations around those same axes) and the coordinates of the observed point (X_i, Y_i, Z_i, with image coordinates $x_i = \frac{X_i}{Z_i}$ and $y_i = \frac{Y_i}{Z_i}$), cf. [20].

$$u_i = -\frac{U}{Z_i} + x_i\frac{W}{Z_i} + Ax_iy_i - Bx_i^2 - B + Cy_i = u_T + u_R$$
$$v_i = -\frac{V}{Z_i} + y_i\frac{W}{Z_i} - Cx_i + A + Ay_i^2 - Bxy_i = v_T + v_R \quad (1)$$

The optical flow can be split into two factors: the rotational (u_R, v_R) and translational (u_T, v_T) parts. The rotational part is only dependent on the pixel coordinate and rotational rates of the camera (A, B, C). Therefore, the structure of the scene (in particular, depth) has no influence on the rotational part of optical flow. The Inertial Measurement Unit (IMU) of an MAV can be used to measure the rotational rate.



(a) FlowNet2. (b) RAFT.

(c) Maskflownet. (d) Liteflownet.

Figure 3: Different neural networks estimating optical flow compared using the MIDGARD [14] dataset.

2.3 *Calculation of the FoE*

The Focus of Expansion (FoE) is the point where all flow vectors point towards or originate from when an observer moves through an environment. This point can lie outside the camera's Field of View, but in this paper it is assumed to lie in the image plane. Nonetheless, the method does work for FoEs outside the Field of View.

The FoE is calculated as presented in figure 4. First, two optical flow vectors are randomly sampled. The intersection of the two vectors is calculated. This process is repeated N times, where N equals 1000. A RANSAC scheme [21] is applied to the set of intersections to make it more robust against outliers. The RANSAC method calculates a location in the image where most intersections have a distance to this point that is lower than a certain threshold. The resulting location is taken as the location of the FoE.

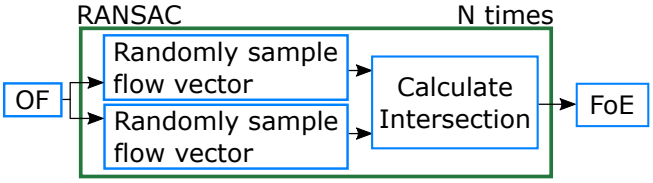


Figure 4: FoE method flowchart.

2.4 *Sky segmentation*

In outdoor environments, clouds in the sky can also move independently from the camera and generate substantial flow. Therefore, we segment clouds and sky by appearance and mask them out from the result. To this

end, we use HRNet-OCR [22] with the default weights trained on the Cityscapes dataset [23]. By comparing the depth buffer from AirSim with the segmentation mask for the sky, one can validate the performance of the segmentation. Because of the visual simplicity of the environment in AirSim, the TPR of the sky segmentation is at least 99.5% and the FPR is less than 0.1%. The sky segmentation is performed at half the resolution of the captured images from AirSim, to reduce memory and computational effort of the GPU.

2.5 Thresholding and detection output

The output of the algorithm is based on the angle κ as illustrated in figure 1. The larger κ, the more likely it is that that pixel belongs to an object moving relative to the observer. Pixels with a κ angle larger than 15° are marked as moving objects. Out of these marked pixels, flow vectors with a magnitude smaller than 1 pixel/frame are discarded, because the angle of such vectors is sensitive to noise. However, the threshold on κ can be more substantiated by analyzing how the error in the angle of the flow vectors behaves for various magnitudes of flow. One would expect that the error of the estimated OF direction increases for decreasing OF magnitude. This is the case, as shown in figure 5. For 100 FlowNet2 images, the radial error with respect to the ground truth OF data is plotted for all pixels (except the sky) versus the magnitude of the OF. The white line of $0.25 \pm (0.5 + \frac{8}{|\mathrm{OF}|})$ is fitted manually. The flow magnitude and value of κ that lie in the area between the upper and lower parts of this function, are discarded. Additionally, flow vectors with a magnitude lower than 0.5 pixels/frame are removed. The performance difference when using this 'dynamic' method of thresholding depending on the flow's magnitude is presented in the results section (see figure 10).

3 AirSim

Simulations in AirSim [13] are carried out for various reasons. Most importantly, simulations can provide ground truth optical flow and FoE data that cannot be retrieved in real life. The ground truth optical flow makes it easier to develop a motion-based object detector, because the ground truth optical flow has no noise or artifacts. Simulations also enable validation of the algorithm on a low level, by for example comparing the FoE estimation with the ground truth FoE. Specifically, AirSim is chosen because of its realistic rendering and support for MAVs, including various simulated sensors.

3.1 Environments

One environment is used in AirSim: LandscapeMountains[2]. LandscapeMountains is a freely available

[2] `https://www.unrealengine.com/marketplace/en-US/product/landscape-mountains`

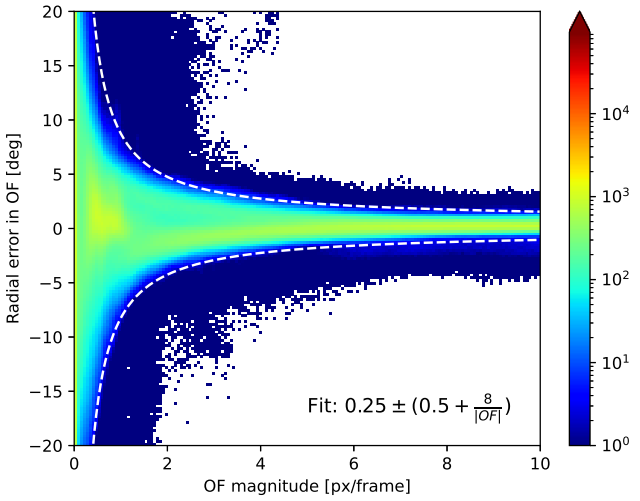


Figure 5: Histogram of the radial error in FlowNet2 (compared to the ground truth OF) versus the magnitude of the OF. Averaged over 100 OF fields.

project from Epic Games, the publisher of Unreal Engine. It was chosen because of its realism, while at the same time being not too demanding. To diminish the influence of visual effects on the estimation of optical flow and the performance of the object detector, most of these influences were removed from the simulation. All moving actors (gates to fly through, birds) are made invisible. The clouds are translated vertically by 500m such that they appear above the terrain. Additionally, to avoid reflections, the ice is replaced by a grass material and the fog is disabled. This limits the method to a set of real-world environments, but in a large range of applications these assumptions can still be considered valid. The only visible visual effect is the shadow of the terrain and MAVs.

3.2 MAV control

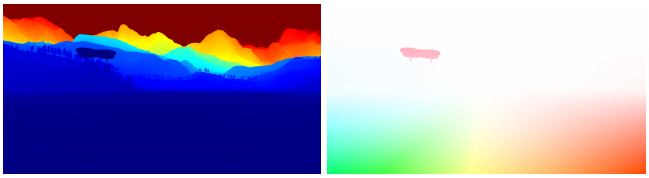
The MAVs are controlled using Python scripts. A loop is run for each simulation configuration, in which the MAVs are controlled and the data from AirSim is captured. First, the control inputs are calculated for the MAV to detect and the observing MAV. The time is advanced for 43ms (23Hz) and lastly, the data from AirSim is collected. The simulation is paused while obtaining the data of AirSim, such that the IMU data and camera frames are taken at the same timestep. The MAVs follow their flight path with a maximum deviation of 0.14m.

Two types of sequences are recorded. Firstly, collision courses, where the MAVs fly towards the same point at the same time at 4m/s. Secondly, sideways trajectories in which one MAV moves sideways in front of the observing MAV, which moves forwards at 4m/s.

3.3 Data acquisition

There are three visual outputs of the simulations: the RGB camera image, the depth in the camera image and the segmentation mask of the MAV inside the images. These three outputs are taken from the same camera, so all use the same projections. These outputs are shown in figures 6a to 6c. The camera image and segmentation mask are saved as PNG files, while the depth image is saved in AirSim's pfm format, enabling the use of floats. Additionally, sensor data is stored of both MAVs. This includes IMU and GPS data, but also contains collision data, the control inputs, FoE coordinates and camera properties. The ground truth FoE is calculated using the view projection matrix of the observer's camera and the observer's velocity vector. The images are collected at a resolution of 1920x1024 pixels with a framerate of approximately 23Hz. The field of view of the camera is 90° and there is no distortion or noise in the image.

(a) RGB camera output. (b) G.t. segmentation mask.



(c) Depth output. (d) Ground truth optical flow.

Figure 6: The different ground truth (g.t.) output frames captured in AirSim (a-c) and the g.t. optical flow (d) calculated from the depth output.

3.4 Ground truth optical flow

AirSim has no built-in method of calculating dense ground truth optical flow. However, it can be calculated from the depth image and the viewprojection matrix of the camera. This method is based on the work of Mayer et al. [24]. A visualization of the ground truth optical flow is shown in figure 6d and the steps of the method are shown in figure 7. Using the depth image, one can deduce the 3D world positions of all projected pixels by multiplying the inverse of the viewprojection matrix with the homogeneous pixel coordinates. This will result in a point cloud. From these 3D points, one can calculate their 3D positions one timestep ago. Finally, by applying the viewprojection matrix of the previous frame to the 3D points, one obtains the 2D coordinates of the original pixels one timestep ago. The difference between the original and the reprojected coordinates yields the ground truth OF. The optical flow calculation has some limitations. For example, the flow of visual effects is not taken into account. This includes shadows, animations of vegetation, reflections/refractions etc.

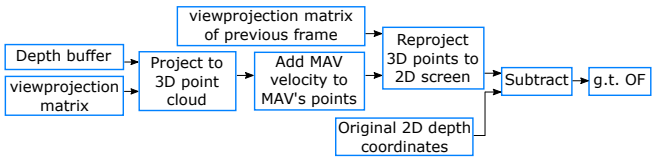


Figure 7: Flowchart for calculating the ground truth OF.

3.5 IMU

The IMU is modeled using the default IMU in AirLib, the library implementing MAV dynamics and sensors inside AirSim. The biases and random walks of the gyroscope and accelerometer are set to zero, leaving IMU noise to future work. The IMU data is used for OF derotation, cf. subsection 2.2.

3.6 Overview of parameters

An overview of all parameters for the simulations and the object detection method is shown in table 1.

Table 1: Parameters of the simulation and method.

Parameter	Value
Resolution	1920x1024 px
Framerate	23 Hz
Field of View	90°
Observing MAV speed	4m/s
Fixed OF magnitude threshold	1 px/frame
Fixed OF radial threshold	15°
Number of collision course sequences	6
Number of sideways sequences	9
Number of FoE validation sequences	3

4 Results

This section will present the results in terms of performance on the FoE estimation and object detection for the simulations in AirSim.

4.1 AirSim

Because the accuracy of the object detection depends on the quality of the FoE estimation, the error between the estimated and ground-truth FoE is analyzed for different situations. A histogram of the FoE errors for one sequence is shown in figure 8. This is recorded for an MAV moving (without rotation) at 4 m/s with an FoE 20 pixels from the left and right edges of the image and an FoE in the center. Two characteristics are notable. For a forward moving MAV, the estimated FoE is on average slightly offset upwards (by 7.2 pixels) and to the

right (by 2.8 pixels), but this is small compared to the total resolution of the image and therefore negligible for the majority of all pixels. Moreover, the location of the FoE affects the mean of the x distribution slightly, as the estimated FoE tends towards the center.

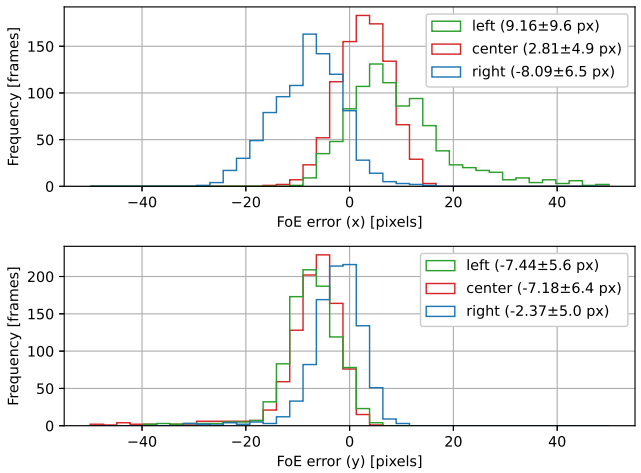


Figure 8: Histograms showing the error (in x and y direction) between the g.t. FoE and estimated FoE, for a FoE in the far left (at $x = 20$ px), center (at $x = 960$ px) and far right (at $x = 1900$ px) part of the image. The legend includes the means and standard deviations of the distributions.

The performance of the object detection method is determined by the True Positive Rate (TPR) and the False Positive Rate (FPR). TPR is the percentage of pixels from dynamic objects that are identified as dynamic object pixels. FPR is the percentage of pixels from static objects that are identified as dynamic object pixels. Ideally, one would have a large TPR for a very small FPR. In this case, the FPR is always relatively small, but the TPR varies considerably. This is shown in figure 9, where the TPR is plotted against κ for various speeds of the MAV to detect. As can be seen, the object detector is less accurate for slower moving objects.

The relation between the TPR/FPR and the magnitude of the OF of the detected object is presented in figure 10). The average TPR for κ between 180° and 90° is taken as measure of performance. It is clear that lower OF magnitudes decrease the TPR, but the FPR is unaffected. As hypothesized in section 2.5, a threshold that is dependent on the magnitude of the OF vector (a dynamic threshold) indeed results in a higher TPR for slower moving objects. However, this also increases the FPR to 0.5% - 2.0%, which could be considered acceptable depending on the application. In situations where the object to detect has a large OF vector, a fixed threshold would be more suitable.

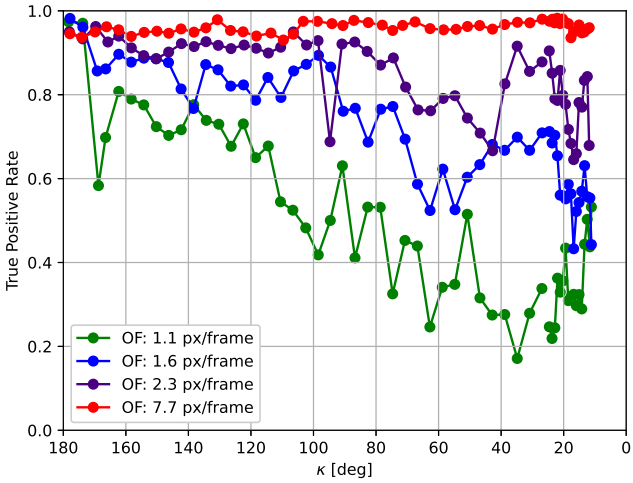


Figure 9: TPR vs κ, where MAV to detect moves from left to right with four different speeds (thus four magnitudes of OF) at a relative distance of 5m, decreasing κ from 180° to 0°. A dynamic threshold is applied.

Additionally, lower values of κ degrade the performance of the object detector. This is illustrated in figure 11, in which the angle κ is visualized. A higher intensity in the image indicates a higher value of κ, meaning that the flow vector is not pointing towards the FoE. Thus, such a flow vector does not only correspond to the flow created by the translation of the camera, but also to the motion of the object belonging to that pixel. In figure 11a, κ is large and therefore the MAV is easy to detect. In figure 11b, the MAV is more challenging to detect and in figure 11c, the method is unable to detect the MAV as κ is close to zero.

To test the method in more complex circumstances, data was recorded for a collision course where the flight paths of the MAVs cross at an angle of 75°, shown in figure 11d. In this case, the MAV to detect remains at the same location in the image during the sequence, but becomes closer and therefore larger in the image. It can be seen that the right part has a κ angle close to zero. However, using a dynamic threshold, the TPR is still high (0.98) at the cost of a relatively high FPR ($2.8 \cdot 10^{-2}$). Unfortunately, this is only the case for short distances. For a collision course, the flow magnitude at large distances is too small to properly estimate κ.

5 Discussion and Conclusion

We have introduced an optical-flow-based algorithm for detecting other moving objects, where our interest lies in the detection of other drones. The object detection method in this paper proves to work successfully if the angle of the optical flow vector of the object to detect is sufficiently different from the background flow, as

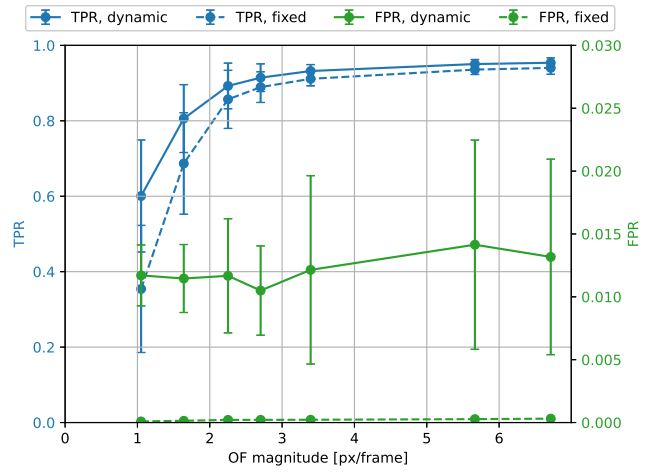
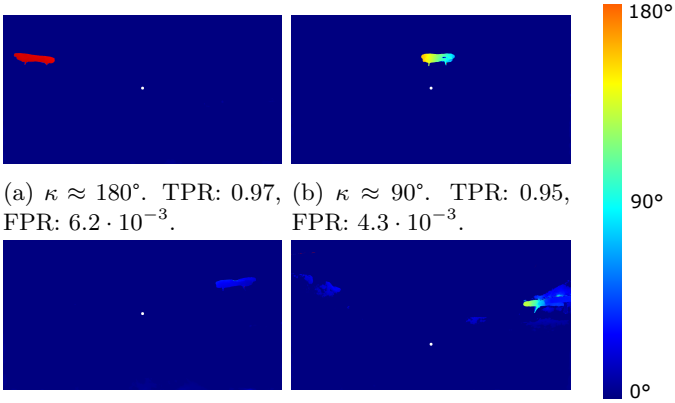


Figure 10: TPR and FPR vs the magnitude of the OF of the MAV to detect for $\kappa > 90°$ using a fixed and dynamic threshold.



(a) $\kappa \approx 180°$. TPR: 0.97, FPR: $6.2 \cdot 10^{-3}$.

(b) $\kappa \approx 90°$. TPR: 0.95, FPR: $4.3 \cdot 10^{-3}$.

(c) $\kappa \approx 0°$. TPR: 0.93, FPR: $1.5 \cdot 10^{-2}$.

(d) CC. TPR: 0.98, FPR: $2.8 \cdot 10^{-2}$.

Figure 11: κ displayed for various situations. In (a) to (c), the MAV moves sideways from left to right. In (d), the observer and target are on a collision course (CC) of 75°. The white dot represents the FoE. A dynamic threshold is applied to calculate the TPR and FPR.

illustrated in figure 11. This means that objects moving towards the FoE, which are crossing the flight path of the observer and are thus considered dangerous, can be successfully detected. Although the method is based on assumptions of the OF, it does not assume a specific appearance of the moving object, which makes it suitable for a wide range of applications.

The method in this paper has the following limitations. Most importantly, if the observer is stationary or the dynamic object has no optical flow, detection by means of flow direction will not succeed. Therefore, MAVs on head-on collision courses cannot be detected in this way because they have the same flow field as the surroundings. A solution would be to utilize the divergence of the OF field to detect head-on colliding objects (just as for static objects). Another limitation is the computational effort of our current implementation, which is large due to the remaining deep network parts of the pipeline. For example, FlowNet2 runs on approximately 1.7 Hz on an RTX 2070 for 1920x1024 images. This would be too slow to use in real-time on MAVs themselves. Therefore, the resolution has to be reduced and/or another optical flow method must be used onboard.

References

[1] Shweta Gupte, Paul Infant Teenu Mohandas, and James M. Conrad. A survey of quadrotor unmanned aerial vehicles. *Conference Proceedings - IEEE SOUTHEASTCON*, 2012.

[2] Mario Coppola, Kimberly N. McGuire, Christophe De Wagter, and Guido C. H. E. de Croon. A Survey on Swarming With Micro Air Vehicles: Fundamental Challenges and Constraints. *Frontiers in Robotics and AI*, 7, feb 2020.

[3] Danping Zou, Ping Tan, and Wenxian Yu. Collaborative visual SLAM for multiple agents: A brief survey. *Virtual Reality & Intelligent Hardware*, 1(5):461–482, oct 2019.

[4] Viktor Walter, Nicolas Staub, Antonio Franchi, and Martin Saska. UVDAR System for Visual Relative Localization with Application to Leader-Follower Formations of Multirotor UAVs. *IEEE Robotics and Automation Letters*, 4(3):2637–2644, jul 2019.

[5] James F. Roberts, Timothy Stirling, Jean Christophe Zufferey, and Dario Floreano. 3-D relative positioning sensor for indoor flying robots. *Autonomous Robots*, 33(1-2):5–20, 2012.

[6] Matous Vrba, Daniel Hert, and Martin Saska. Onboard Marker-Less Detection and Localization of Non-Cooperating Drones for Their Safe Interception by an Autonomous Aerial System. *IEEE Robotics and Automation Letters*, 4(4):3402–3409, 2019.

[7] Changhong Fu, Adrian Carrio, Miguel A. Olivares-Mendez, Ramon Suarez-Fernandez, and Pascual Campoy. Robust real-time vision-based aircraft

Tracking from Unmanned Aerial Vehicles. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 5441–5446. IEEE, may 2014.

[8] Bilal Taha and Abdulhadi Shoufan. Machine Learning-Based Drone Detection and Classification: State-of-the-Art in Research. *IEEE Access*, 7:138669–138682, 2019.

[9] Fabian Schilling, Julien Lecoeur, Fabrizio Schiano, and Dario Floreano. Learning Vision-Based Flight in Drone Swarms by Imitation. *IEEE Robotics and Automation Letters*, 4(4):4523–4530, oct 2019.

[10] Ryota Yoshihashi, Tu Tuan Trinh, Rei Kawakami, Shaodi You, Makoto Iida, and Takeshi Naemura. Differentiating objects by motion: Joint detection and tracking of small flying objects. *arXiv*, 2017.

[11] Jing Li, Dong Hye Ye, Timothy Chung, Mathias Kolsch, Juan Wachs, and Charles Bouman. Multi-target detection and tracking from a single camera in Unmanned Aerial Vehicles (UAVs). *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4992–4997, oct 2016.

[12] Cemal Aker and Sinan Kalkan. Using Deep Networks for Drone Detection. *arXiv*, jun 2017.

[13] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. *arXiv*, pages 1–14, may 2017.

[14] Viktor Walter, Matous Vrba, and Martin Saska. On training datasets for machine learning-based visual relative localization of micro-scale UAVs. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 10674–10680, 2020.

[15] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. A Lightweight Optical Flow CNN - Revisiting Data Fidelity and Regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2020.

[16] Shengyu Zhao, Yilun Sheng, Yue Dong, Eric I.Chao Chang, and Yan Xu. Maskflownet: Asymmetric feature matching with learnable occlusion mask. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, c:6277–6286, 2020.

[17] Zachary Teed and Jia Deng. RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12347 LNCS:402–419, 2020.

[18] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of optical flow estimation with deep networks. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:1647–1655, 2017.

[19] Raoul Dinaux, Nikhil Wessendorp, Julien Dupeyroux, and Guido de Croon. FAITH: Fast iterative half-plane focus of expansion estimation using event-based optic flow. *arXiv*, feb 2021.

[20] H.C. Longuet and K. Prazdny. The Interpretation of a Moving Retinal Image. *Proceding of the royal society of london*, 208(1173):385–397, 1980.

[21] Martin A. Fischler and Robert C. Bolles. Random sample consensus. *Communications of the ACM*, 24(6):381–395, jun 1981.

[22] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical Multi-Scale Attention for Semantic Segmentation. *arXiv*, pages 1–11, may 2020.

[23] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-Decem:3213–3223, 2016.

[24] N Mayer, E Ilg, P Häusser, P Fischer, D Cremers, A Dosovitskiy, and T Brox. A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4040–4048, 2016.