Detection of fast incoming objects with a moving camera

Fabio Poiesi
fabio.poiesi@qmul.ac.uk
Andrea Cavallaro
a.cavallaro@qmul.ac.uk

Centre for Intelligent Sensing
Queen Mary University of London
London, UK

Abstract

Using a monocular camera for early collision detection in cluttered scenes to elude fast incoming objects is a desirable but challenging functionality for mobile robots, such as small drones. We present a novel moving object detection and avoidance algorithm for an uncalibrated camera that uses only the optical flow to predict collisions. First, we estimate the optical flow and compensate the global camera motion. Then we detect incoming objects while removing the noise caused by dynamic textures, nearby terrain and lens distortion by means of an adaptively learnt background-motion model. Next, we estimate the time to contact, namely the expected time for an incoming object to cross the infinite plane defined by the extension of the image plane. Finally, we combine the time to contact and the compensated motion in a Bayesian framework to identify an object-free region the robot can move towards to avoid the collision. We demonstrate and evaluate the proposed algorithm using footage of flying robots that observe fast incoming objects such as birds, balls and other drones.

1 Introduction

A quick detection of fast incoming objects that become visible only a few seconds before impact can help a flying robot avoid a collision. While collision avoidance can use data from 3D laser scanners, sonars or stereo cameras [15], monocular cameras are preferable because of their small weight, power efficiency and ease of deployment. However, a single uncalibrated camera cannot in general make accurate predictions about the time to collision (or time to contact, TTC) with incoming objects of an unknown size.

Colliding objects can be detected using feature points to measure the expansion of image patches [13] or to restrict the computation of the TTC to certain locations [2, 14]. However, feature point based methods [13, 14] or generic object detectors (e.g. EdgeBox [22]) are inadequate with textureless objects. Dense depth maps generated by monocular cameras while hovering [3] or collision-free trajectories built in the disparity space with stereo cameras [11] can also be used. Alternatively, stereo block-matching can build a disparity map for a limited range of distances to localise obstacles [5]. Recently, Watanabe *et al*. [19] proposed a method to compute the TTC from image intensity values when a light source illuminates the colliding surface.

Pages 146.1-146.11
DOI: https://dx.doi.org/10.5244/C.30.146

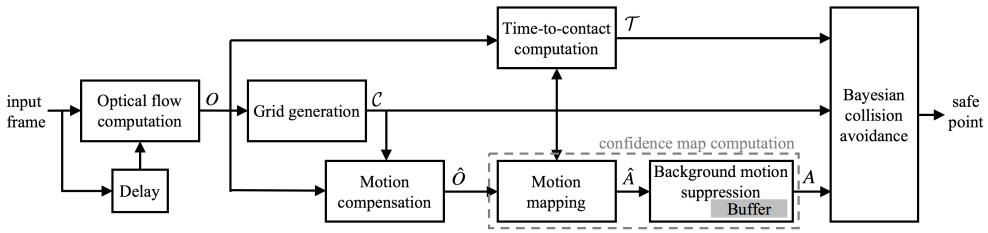

Figure 1: Block diagram of the proposed object detection and avoidance algorithm.

The optical flow can be used to detect incoming objects via the TTC [7]. However, the optical flow may generate several false positives due to the regularisation noise and the aperture problem. In fact, the optical flow is used alone for obstacle avoidance only in virtual [2] or controlled scenarios [17]. In real scenarios, the optical flow is used along with an ultrasonic sensor for altitude stabilisation [8], with inertial sensors for indoor navigation in textured environments [23] and with a feature point detector for obstacle avoidance [1]. While methods exist that tackle the problem of early collision detection from (fixed-wing) unmanned aerial vehicles, these methods are designed to work at high altitudes where clutter is generated by clouds and therefore morphological operators suffice as post-processing [12]. At low altitudes, which are common for micro aerial vehicles, the problem is considerably more challenging as the surrounding environment is often cluttered.

In this paper, we propose a method to address the problem of detecting fast incoming objects when they become visible from a moving camera shortly before an impact (Fig. 1). To detect these objects we use only the optical flow computed from an uncalibrated camera without using any feature points. We calculate the motion induced by the camera to infer the position of the objects on the image plane and the TTC to estimate their proximity. To reduce the influence of camera motion in object detection we adaptively learn the background motion. The proposed adaptation strategy allows the background motion model to cope with varying camera velocities and with different scene depths. Finally, we merge the optical flow information and use a Bayesian collision avoidance method to locate object-free regions, whose centre is represented as a *safe point*. The robot can then use the position of the safe point and avoid the incoming object(s).

2 Time to contact

Let a flying robot (e.g. a small drone) capture with its forward facing camera a video whose frame at instant t is $I(t)$. $I(t)$ is composed of N pixels with positions $\Phi = \{(x_i, y_i)\}_{i=1}^{N}$.

We first estimate the dense optical flow, $O(t)$, between the current, $I(t)$, and the previous frame, $I(t-1)$ [9]:

$$O(t) = \{(x_i, y_i, u_i(t), v_i(t))\}_{i=1}^{N}, \tag{1}$$

where (x_i, y_i) is the position and $(u_i(t), v_i(t))$ are the velocity components of the motion vector associated to pixel i. For simplicity we will omit t in the remainder of the paper.

Distant objects generally produce motion vectors whose magnitude is smaller than that of motion vectors generated by nearby objects. We therefore calculate the time to contact (TTC) to estimate the number of frames prior to the potential contact with an observed object [6]. Let $\hat{\mathcal{T}} = \{\hat{\tau}_i\}_{i=1}^{N}$ be the set of TTC values, $\hat{\tau}_i$, associated to each pixel i and calculated as

$$\hat{\tau}_i = \frac{||(x_i - \tilde{x}, y_i - \tilde{y})||_2}{||(u_i, v_i)||_2}, \tag{2}$$

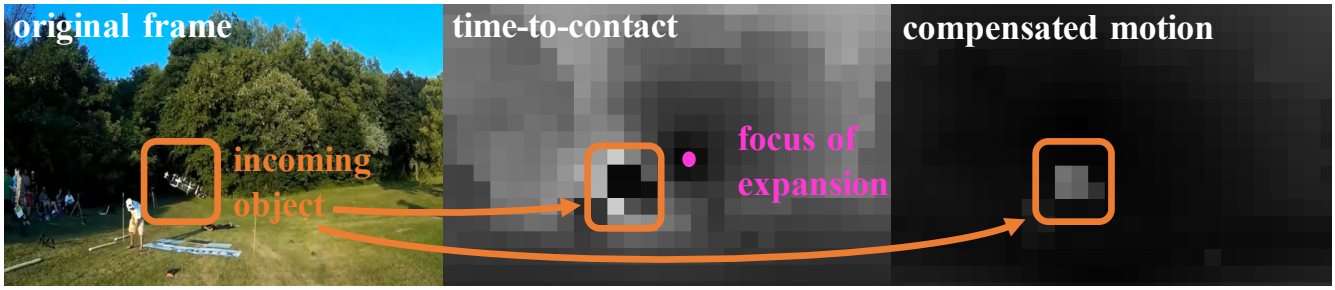

Figure 2: Example of time to contact (TTC) and compensated motion for an incoming object. (a) A small drone (orange box) is about to collide with the drone recording the egocentric video. (b) The focus of expansion F (magenta point) generates small values of TTC (τ_j) that may generate false incoming-object detections. (c) Using the compensated motion (m_j) produces instead a more accurate object detection.

where $||\cdot||_2$ is the ℓ-2 norm and $F = (\tilde{x}, \tilde{y})$ is the focus of expansion (FOE) [16]. The FOE defines the translational direction of the flying robot as the point of divergence of the optical flow on the image plane. The FOE, which can be calculated using O via least squares [16, 18], enables us to compute the TTC regardless of the direction of motion. Because motion vectors near the FOE may generate a small value for the numerator of Eq. 2, the detection of incoming objects around F is unreliable as they may appear to be still [11].

In principle, one could segment regions with small (decreasing) TTC values to detect incoming objects. However, in practice, the location of these regions can change between consecutive frames as the position of F, which is obtained in each frame independently, varies with the orientation of the optical flow field. To address this problem, we estimate the location of an object and its closeness to the camera after global motion compensation and based on TTC, respectively (Fig. 2). This solution is discussed in the next section.

3 Incoming object detection

We compensate the motion O by calculating the optical flow induced by the moving camera. Let the camera-induced optical flow, $\hat{O}$, be defined as

$$\hat{O} = \{(x_i, y_i, \hat{u}_i, \hat{v}_i)\}_{i=1}^{N}, \tag{3}$$

where $(\hat{u}_i, \hat{v}_i)$ is the motion vector at (x_i, y_i) induced by the camera motion. We compute $\hat{O}$ by applying a least square affine transformation to O [20]. Differences between $\hat{O}$ and O highlight regions on the image plane whose motion differs from that of the camera. Given the magnitude of the motion-vector difference $\hat{m}_i = ||(\hat{u}_i - u_i, \hat{v}_i - v_i)||_2$, our goal is to localise incoming objects in each frame by detecting regions with large motion differences while discarding large noisy motion vectors.

We localise the incoming objects using a lower resolution representation, later referred to as *confidence map*, which is generated by analysing the $\hat{m}_i$ values grouped into C *cells* of $g \times g$ pixels. Let $\mathcal{C} = \{c_j\}_{j=1}^{C}$ be the set of the centres $c_j = (x_j, y_j)^T$ of each cell j and $\hat{A} = \{\hat{\alpha}_j\}_{j=1}^{C}$ be the confidence map, where $\hat{\alpha}_j \in [0, 1]$ is the confidence value for cell j. The larger $\hat{\alpha}_j$, the higher the confidence that cell j contains a moving object.

We calculate $\hat{\alpha}_j$ by appropriately mapping the $(100\rho)^{th}$ percentile of the $\hat{m}_i$ values of cell j (Eq. 5). The $(100\rho)^{th}$ percentile, m_j, is defined by considering the cumulative density

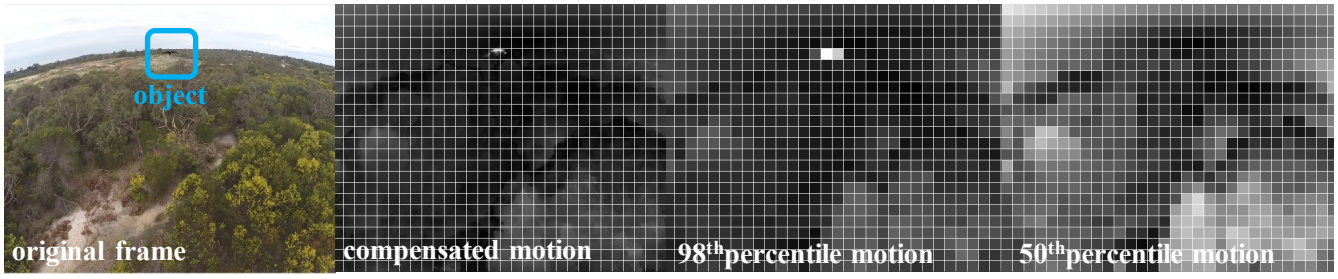

Figure 3: Example of compensated motion using cells. A high percentile (m_j at 98^{th} percentile) can be used to detect a moving object that is far from the camera, also when only a portion of the object falls within a cell.

function:

$$\sum_{\hat{m}_i=\hat{m}_1}^{m_j} f(\hat{m}_i) = \rho, \tag{4}$$

i.e. the area under the probability density function $f(\hat{m}_i)$ to the left of m_j is ρ (e.g. $\rho = 0.98$). Note that an object that is small or far from the camera may occupy only a portion of a cell. We do not use the maximum $\hat{m}_i$ within a cell in order to discard noisy values that are likely to appear in the last percentiles (Fig. 3). Each m_j is then mapped to the corresponding confidence value $\hat{\alpha}_j$ with a sigmoid function that uses as mean value the optical flow magnitude induced by the camera motion:

$$\hat{\alpha}_j = \left(1 + e^{-(m_j - M)}\right)^{-1}, \tag{5}$$

where M is the $(100\rho)^{th}$ percentile of the compensated motion magnitude computed over the whole frame (i.e. not per cell), defined by

$$\sum_{\hat{m}_i=\hat{m}_1}^{M} f(\hat{m}_i) = \rho. \tag{6}$$

To infer the closeness of incoming objects to the camera we define $\mathcal{T} = \{\tau_j\}_{j=1}^{C}$, whose elements τ_j are the $(100(1-\rho))^{th}$ percentile of the TTC in cell j:

$$\sum_{\hat{\tau}_i=\hat{\tau}_1}^{\tau_j} f(\hat{\tau}_i) = 1 - \rho. \tag{7}$$

A value of τ_j closer to zero indicates a closer object.

Ideally, the compensated motion should be non-zero only for moving objects whose motion differs from that of the camera. However, noisy optical flow vectors are generated by untextured areas, lens distortion, the aperture problem and may be amplified by the regularisation. Moreover, when the robot flies close to the terrain, large motion vectors are generated on the lower part of the frame. This spurious motion, which we refer to as *background motion*, may produce erroneously large $\hat{\alpha}_j$ values (Fig. 4).

We learn the background motion by collecting past $\hat{\alpha}_j$ samples in a temporally shifting *buffer*, $B = \{B_j\}_{j=1}^{C}$, which is composed of local buffers B_j (one for each cell j). The parameters of B_j are T_1, the number of frames used to learn the background motion (e.g. T_1=15), and T_2, a learning delay (e.g. T_2=3). This learning delay hiders the buffer to use the motion

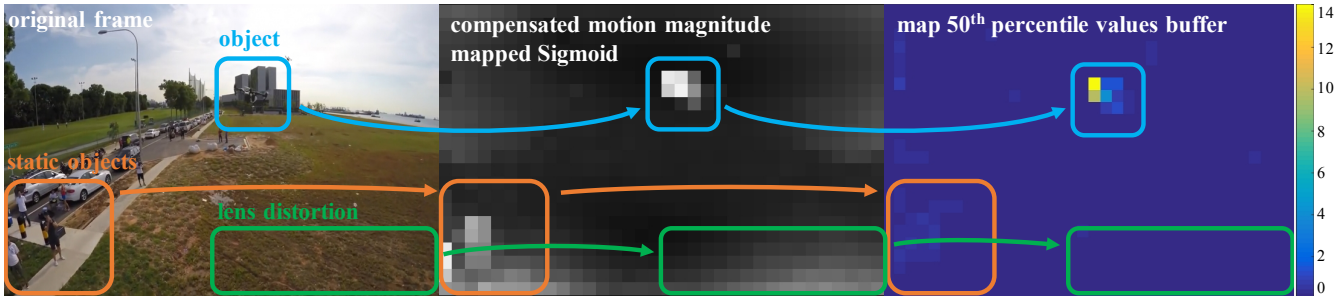

Figure 4: Removal of spurious motion caused by static objects and lens distortion. (a) Original frame. (b) Compensated motion magnitude mapped via a sigmoid function ($\hat{A}$). (c) Map with the 50^{th} percentile distance computed from the background motion buffer (A).

of an incoming object to update the background model (i.e. using T_2 increases the discrimination of moving objects when they become visible).

Let the samples $\hat{\alpha}_j$ stored in B_j be represented as $\hat{\alpha}_j^b$, with $b = 1,...,T_1$. We compare these samples with the incoming samples to detect moving objects using an approach similar to ViBe [4]. ViBe was however used for image intensities and without any normalisations. Let the difference d_j^b between the incoming $\hat{\alpha}_j$ value and the stored $\hat{\alpha}_j^b$ be defined as

$$d_j^b = \frac{|\hat{\alpha}_j - \hat{\alpha}_j^b|}{\hat{\alpha}_j^\rho} \quad \forall b = 1,...,T_1, \tag{8}$$

where the normalisation factor $\hat{\alpha}_j^\rho$ is the $(100\rho)^{th}$ percentile of the values in B_j, which is defined by

$$\sum_{\hat{\alpha}_j^b = \hat{\alpha}_j^1}^{\hat{\alpha}_j^\rho} f(\hat{\alpha}_j^b) = \rho. \tag{9}$$

If $\hat{\alpha}_j$ is dissimilar from the background motion, cell j is deemed to belong to a moving object. In this case the value to update the buffer for cell j is chosen randomly from one of the eight closest neighbouring cells that are considered to represent background motion. If no neighbours satisfy this condition, the buffer is updated with a random value selected from the whole confidence map of the current frame.

A value $\hat{\alpha}_j$ is considered to be similar to the background motion when its d_j^b is below a threshold δ for more than half of the $\hat{\alpha}_j^b$ values in the buffer. When $\hat{\alpha}_j$ is similar to the background motion, its value is used to update the buffer via a first-in-first-out policy.

Note that the values $\hat{\alpha}_j$ are distributed within $[0,1]$ depending on the scene, thus affecting differently the range of d_j^b values. An example where the values of d_j^b span different intervals due to different distributions of $\hat{\alpha}_j$ is shown in Fig. 5. We therefore design an adaptive threshold δ that depends on the distribution of the distances d_j^b. We first determine d^ρ as $(100\rho)^{th}$ percentile of d_j^b as

$$\sum_{d_j^b = d_j^1}^{d^\rho} f(d_j^b) = \rho. \tag{10}$$

Then, assuming a normal distribution with variance d^ρ for the distances, an incoming object is detected when d_j^b exceeds three standard deviations from the zero mean (i.e. we set $\delta =$

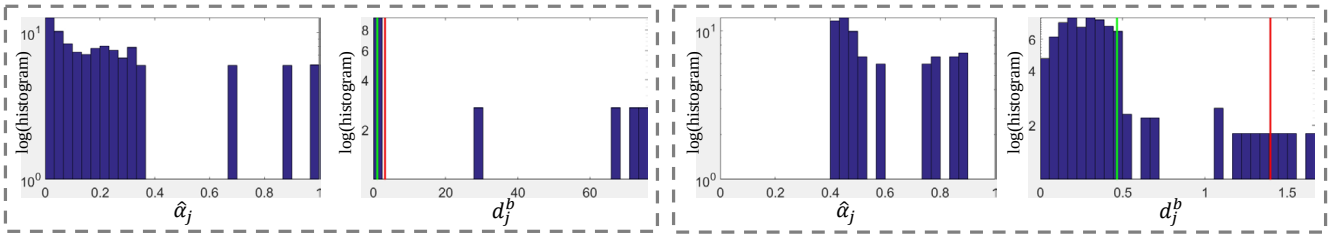

Figure 5: Example of adaptive threshold used to determine if $\hat{\alpha}_j$ belongs to the background motion model. The two distributions of $\hat{\alpha}_j$ lead to different ranges of d_j^b. The green line indicates the 98^{th} percentile at 1.05 in (a) and at 0.46 in (b); whereas the red line indicates three times the 98^{th} percentile at 3.16 in (a) and at 1.39 in (b). The vertical axis has a logarithmic scale to facilitate the comparison.

$3d^\rho$). The elements of the output confidence map $A = \{\alpha_j\}_{j=1}^{C}$ are therefore defined as

$$\alpha_j = \begin{cases} \hat{\alpha}_j & \text{if } \left(\sum_{b=1}^{T_1} \mathbf{1}(d_j^b)\right) > \frac{T_1}{2} \vee \hat{\alpha}_j \geq .75 \\ 0 & \text{otherwise,} \end{cases} \tag{11}$$

where $\mathbf{1}(\cdot)$ is the indicator function that is equal to 1 when $d_j^b > 3d^\rho$. Moreover, the first condition is logically *or*-ed with $\hat{\alpha}_j \geq .75$ in order to detect objects that have become part of the background after a long permanence in the same cell.

4 Collision avoidance

We aim to infer the position of a *safe point* on the image plane that a flying robot can use to produce steering commands to avoid an incoming object. For example, the position of the safe point can be mapped to the heading body direction of the robot.

To define regions to be avoided we use a *prior* distribution for the safe point $\Phi_q = (x_q, y_q)$ and a *likelihood* function that accounts for the position of potential incoming objects and the TTC measures. We then generate a *posterior* distribution over the image plane Φ using a Bayesian approach [1]. Let the distribution of possible safe points be a probability density function (*pdf*) built with a mixture of inverted Gaussians, whose minima are centred on the detected moving objects and whose variances are inversely proportional to their TTC. The closer an object to the camera, the smaller the (minimum) value corresponding to the location of the object on the image plane.

Let V_q define the validity of the *prior* safe point quantified via a probability distribution whose value is the maximum of the distribution when the safety is the highest. Let Σ_p be the covariance of the prior distribution: a smaller covariance makes the maximum of the prior distribution larger and the desired safe point easier to follow. The prior distribution is defined as

$$p(V_q = 1|\Phi, \Phi_q) = e^{-\frac{1}{2}(\Phi-\Phi_q)^T \Sigma_p^{-1} (\Phi-\Phi_q)}. \tag{12}$$

Let the level of security of a safe point V_j be proportional to the size of the region the likelihood function generates to avoid a potential incoming object. The *likelihood* function, which uses α_j and τ_j for each cell centre c_j, is defined as

$$p(V_j = 1|\Phi, c_j, \tau_j, \alpha_j) = 1 - \alpha_j e^{-\frac{1}{2}(\Phi-c_j)^T \Sigma_o (\Phi-c_j)}, \tag{13}$$

where the elements of $\Sigma_o = diag(\sigma_o^2, \sigma_o^2)$, which regulates the variance of a mode localised on an object, are inversely proportional to the TTC (note that Σ_o is purposefully not inverted):

$$\sigma_o = \varepsilon + \kappa \left(1 - e^{-\frac{1}{2}\left(\frac{\tau_j}{\sigma_\tau}\right)^2} \right), \tag{14}$$

where κ modulates the contribution of the Gaussian function, σ_τ is the cut-off coefficient for TTC values and ε is the coefficient that limits the expansion of the variance of the inverted Gaussian. The smaller ε and κ, the larger the standard deviation of the constructed inverted Gaussian around a likely incoming object. We refer to this area as *risk region*. The smaller the TTC, the larger the minimum of the posterior distribution that contains this risk region.

Finally, the *posterior* distribution is defined as

$$p(\Phi|\Phi_q, \mathcal{C}, \mathcal{T}, A, \mathcal{V}, V_q) \propto p(V_q = 1|\Phi, \Phi_q) \prod_{j=1}^{C} p(V_j = 1|\Phi, c_j, \tau_j, \alpha_j), \tag{15}$$

where $\mathcal{V} = \{V_j\}_{j=1}^{C}$ and the V_j are assumed to be *i.i.d.* for computational efficiency.

If $\alpha_j = 0 \; \forall j$, no objects have to be avoided, the likelihood function generates a uniform *pdf* and the safe point remains at $\mathcal{S} = \Phi_q$. A large α_j is likely to represent a colliding object and the risk region will expand based on the proximity of the object to the camera. This leads the location of the safe point $\mathcal{S}$ to move with the maximum of the posterior

$$\mathcal{S} = \arg\max_{\Phi} \left(p(\Phi|\Phi_q, \mathcal{C}, \mathcal{T}, A, \mathcal{V}, V_q) \right) \tag{16}$$

when the TTC of the detected objects is sufficiently small.

5 Results

We evaluate the proposed method using a dataset with eight videos (S1,..., S8) with drones undergoing mid-air collisions with a bird, a ball and other drones[1]. These videos include several challenges such as far and close terrain views, different flying robot and incoming object speeds, and dynamic backgrounds (e.g. water). The dataset consists of 1367 frames, all resized to 720×480 pixels, at 25Hz. We compute the optical flow with the method from [9] using its original parameters. The other parameters we used in the experiments are: cell size $g = 20$, percentile $\rho = 0.98$, background motion buffer size $T_1 = 15$, buffer delay $T_2 = 3$, modulation coefficient $\kappa = 0.2$, cut-off coefficient $\sigma_\tau = 70$ and expansion coefficient $\varepsilon = 0.01$.

We compare our method with an approach for obstacle avoidance based on SURF [13] and an approach for moving object segmentation based on the analysis of the phase change of the Fourier spectra from the pixel intensities [21]. We also compare with three baseline methods for moving object detection and with one method that uses a combination of optical flow and SURF. The methods are: TB (threshold-based), which thresholds m_j, the magnitude difference of the compensated motion; TB+FP, which uses the TB optical flow features to select SURF feature points (FP) on moving regions (this approach resembles the method proposed in [1]); MF (mapping function), which uses a threshold applied to the motion difference mapped with the sigmoid function; UB (unnormalised buffer), which uses a fixed

[1] Videos and results are available at: http://www.eecs.qmul.ac.uk/~andrea/avoidance.html

	S1		S2		S3		S4		S5		S6		S7		S8		F-score
	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	
[13]	.01	.54	.00	.07	.00	.63	.01	.17	.04	.14	.01	.04	.03	.30	.03	.27	**.01**
[21]	.23	.49	.02	.83	.01	.63	.25	.42	.27	.30	.44	.04	.33	.21	.34	.44	**.10**
Ours	.97	.77	.26	.93	.04	.63	.76	.57	.99	.48	.94	.58	.77	.76	.46	.38	**.51**
TB	.66	.49	.10	.97	.02	.94	.76	.46	.71	.63	.45	.88	.81	.20	.24	.62	**.30**
TB+FP	.38	.33	.00	.03	.00	.56	.14	.21	.55	.23	.04	.06	.07	.18	.01	.52	**.01**
MF	.95	.51	.38	.83	.03	.44	.99	.40	1	.40	.29	.05	.32	1	.31	.32	**.38**
UB	.96	.67	.26	.90	.04	.63	.80	.48	.99	.47	.86	.39	.77	.76	.42	.33	**.47**

Table 1: Incoming object detection performance in terms of Precision (P) and Recall (R) for each sequence (S1,..., S8) and average F-Score across all the sequences for the methods under analysis. Key – Ours: the proposed approach; TB: threshold based; TB+FP: threshold based + feature points; MF: mapping function. UB: unnormalised buffer.

Figure 6: Sample false positive (yellow), true positive (green) and false negative (red) results.

δ as threshold (similarly to ViBe [4]), without the normalisation of the elements within the buffer. The thresholds of these alternative methods are chosen to provide the best results.

We quantify the incoming object detection accuracy using Precision, Recall and F-score. As ground truth, we annotated a moving object with a bounding box when its size is at least 5×5 pixels. Table 1 compares the incoming object detection performance of the methods under analysis. Overall, the proposed approach outperforms the other methods. Feature point based methods (i.e. [13] and TB+FP) perform worse than the other methods. The method proposed in [21] has lower Precision than TB. A major problem with [21] is with large camera speeds. In fact, [21] has high Recall in S8 as the camera is almost stationary but a large number of false positives in S2 and S3 as the robot flies at high speed generating large displacements between consecutive frames. The compensated motion provides more accurate detections than [21] when objects are small. For example, in S3 TB can detect an incoming object when it is farther away than [21]. The performance of MF improves with respect to that of TB. The sigmoid function can self-modulate its own mean value based on the global motion, which makes the moving object detection adaptive to the speed of the flying robot. Compared to UB, the buffer normalisation we use generates a higher Recall.

Fig. 6 shows three scenarios from S3 (left), S4 (centre) and S7 (right). In S3, we can observe two large false positives: both regions are due to large-magnitude optical flow that is not learnt by the background motion model. In S4, we observe that the proposed method can detect multiple moving objects: the top object is located on a simple background, whereas the bottom object is located on a cluttered background. In S7, a false negative is generated by a highly noisy optical flow that prevents the object and the background (noise) from being distinguishable.

We also quantify the relative time between the instant the safe point is computed and the instant the impact occurs. In the absence of incoming objects, we set the safe point to be in the centre of the frame. The maximum possible distance between the desired safe point, Φ_q, and the computed safe point, $\mathcal{S}$, is half the frame diagonal. We use this maximum

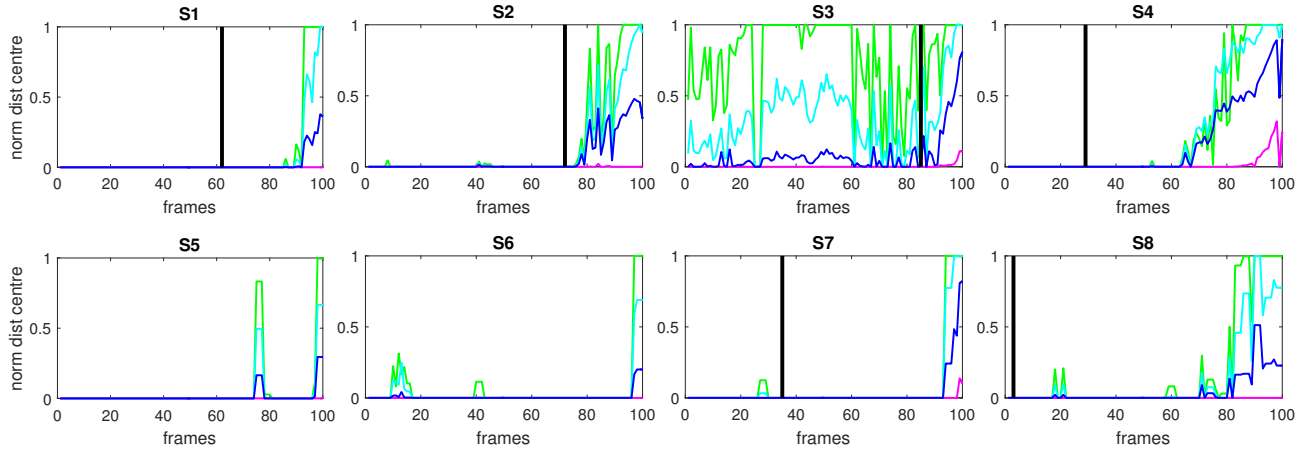

Figure 7: Normalised distance between the desired and the computed safe points up to the collision (occurring at frame 100) when varying ε for sequence S1 to S8. Green: $\varepsilon = 0.001$; cyan: $\varepsilon = 0.005$; blue: $\varepsilon = 0.01$, magenta: $\varepsilon = 0.05$. The black vertical line indicates when the object starts being included in the ground truth (note that in S5 and S6 the black line is at frame 0).

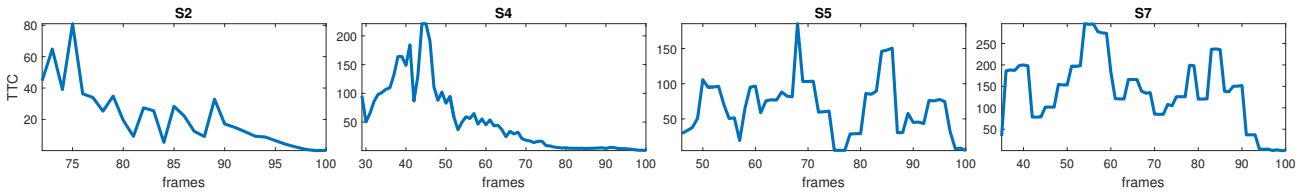

Figure 8: Time to contact (TTC) of an incoming object up to the collision (occurring at frame 100) for sequence S2, S4, S5 and S7. S2 and S4 show a decreasing trend. S5 and S7 are affected by a large amount of noise that is due to inaccuracies of the optical flow.

distance to normalise the distance of $\mathcal{S}$ from the centre of the frame. Fig. 7 shows when the Bayesian avoidance method generates the safe point before the collision occurs at varying values of ε. The value $\varepsilon = 0.01$ is a good trade-off between stability and sensitivity of safe points. Smaller values of ε lead to high instability when false positive detections occur (see S3), whereas larger values lead to late or missed generations of safe points (see S1). The generation of the safe point is also influenced by the noise in the optical flow, which corrupts the TTC measures. As a consequence, the generation of safe points may occur considerably later than the time instant when an incoming object becomes visible (see S1, S5, S7, S8 in Fig. 7). For example, the optical flow noise makes the object not detectable between frame 80 and 95 in S5.

Finally, Fig. 8 shows the TTC measured during the last 100 frames before the collision. In S2 and S4 it is visible from the trend of the TTC over time that the objects are getting closer. This is also reflected in the corresponding results of Fig. 7, where we can see a variation of the safe point happens as soon as the incoming object becomes visible. However, in S5 and S7 the noisy TTC does not provide evidence to detect incoming objects as the TTC values do not produce a decreasing trend. In fact, S5 and S7 show mostly no variations of the safe point position in Fig. 7.

6 Conclusions

We presented a visual collision detection method that is based on optical flow and is applicable also when incoming objects become visible only a few seconds before collision with a moving camera. We combined time to contact and compensated motion features with a Bayesian collision avoidance method to infer safe points on the image plane. We also proposed a methodology to learn the background motion to reduce false positive detections. The proposed method can detect incoming objects 10 to 40 frames prior to collision and can be synergistically used with other features or collision detection methods.

Future work includes improving the motion estimation accuracy to anticipate incoming objects and to reduce the regularisation noise. Moreover, we will use temporal filtering to reduce false positive detections and make predictive decisions about safe points. Finally, we will model the steering manoeuvres to be selected based on the posterior *pdf* (Eq. 15) and on quadcopter controllers designed for collision avoidance [10].

Acknowledgments

This work was supported in part by the ARTEMIS JU and the UK Technology Strategy Board (Innovate UK) through the COPCAMS Project, under Grant 332913.

References

[1] G. Alenya, A. Negre, and J.L. Crowley. Time to contact for obstacle avoidance. In *European Conference on Mobile Robots*, pages 19–24, Mlini, HR, Sep. 2009.

[2] G. Alenya, A. Negre, and J.L. Crowley. A comparison of three methods for measure of time to contact. In *Proc. of Intelligent Robots and Systems*, pages 4565–4570, St. Louis, USA, Oct. 2009.

[3] H. Alvarez, L.M. Paz, J. Sturm, and D. Cremers. Collision avoidance for quadrotors with a monocular camera. In *Experimental Robotics*. Springer, 2016.

[4] O. Barnich and M. Van Droogenbroeck. ViBE: a universal background subtraction algorithm for video sequences. *Trans. on Image Processing*, 20(6):1709–1724, Jun. 2011.

[5] A. J. Barry and R. Tedrake. Pushbroom stereo for high-speed navigation in cluttered environments. In *Proc. of Robotics and Automation*, pages 3046–3052, Seattle, USA, May 2015.

[6] T. Camus. Calculating time-to-contact using real-time quantized optical flow. In *National Institute of Standards and Technology NISTIR 5609*, 1995.

[7] W. Green and P. Oh. Optic-flow-based collision avoidance. *IEEE Robotics & Automation Magazine*, 15(1):96–103, Mar. 2008.

[8] D. Honegger, L. Meier, P. Tanskanen, and M. Pollefeys. An open source and open hardware embedded metric optical flow cmos camera for indoor and outdoor applications. In *Proc. of Robotics and Automation*, pages 1736–1741, Karlsruhe, GE, May 2013.

[9] C. Liu. *Beyond Pixels: Exploring New Representations and Applications for Motion Analysis*. PhD thesis, Massachusetts Institute of Technology, May 2009.

[10] Z. Liu, R. Sengupta, and A. Foina. Quadrotors in smart cities avoiding helicopters. In *Proc. of American Control Conference*, Boston, USA, Jul. 2016.

[11] L. Matthies, R. Brockers, Y. Kuwata, and S. Weiss. Stereo vision-based obstacle avoidance for micro air vehicles using disparity space. In *Proc. of International Conference on Robotics and Automation*, Hong Kong, CN, May 2014.

[12] L. Mejias, S. McNamara, J. Lai, and J. Ford. Vision-based detection and tracking of aerial targets for UAV collision avoidance. In *Proc. of Intelligent Robots and Systems*, pages 18–22, Taipei, TW, Oct. 2010.

[13] T. Mori and S. Scherer. First results in detecting and avoiding frontal obstacles from a monocular camera for micro unmanned aerial vehicles. In *Proc. of International Conference on Robotics and Automation*, pages 1750–1757, Karlsruhe, GE, May 2013.

[14] D. Muller, J. Pauli, C. Nunn, S. Gormer, and S. Muller-Schneiders. Time to contact estimation using interest points. In *Proc. of Intelligent Transportation Systems*, pages 1–6, St. Louis, USA, Oct. 2009.

[15] M. Nieuwenhuisen, D. Droeschel, M. Beul, and S. Behnke. Obstacle detection and navigation planning for autonomous micro aerial vehicles. In *Proc. of International Conference on Unmanned Aircraft Systems*, pages 1040–1047, Orlando, USA, May 2014.

[16] P. O'Donovan. Optical flow: Techniques and applications. Technical Report 502425, University of Saskatchewan, CA, Apr. 2005.

[17] S. Pundlik, E. Peli, and G. Luo. Time to collision and collision risk estimation from local scale and motion. In *Proc. of Advances in Visual Computing*, pages 728–737, Las Vegas, USA, Sep. 2011.

[18] M. Tistarelli, E. Grosso, and G. Sandini. Dynamic stereo in visual navigation. In *Proc. of Computer Vision and Pattern Recognition*, pages 186–193, Maui, USA, Jun. 1991.

[19] Y. Watanabe, F. Sakaue, and J. Sato. Time-to-contact from image intensity. In *Proc. of Computer Vision and Pattern Recognition*, pages 4176–4183, Boston, USA, Jun. 2015.

[20] P. Yao, G. Evansy, and A. Calway. Face tracking and pose estimation using affine motion parameters. In *Proc. of Scandinavian Conference on Image Analysis*, pages 531–536, Bergen, NO, Jun. 2001.

[21] B. Zhou, X. Hou, and L. Zhang. A phase discrepancy analysis of object motion. In *Proc. of Asian Conference on Computer Vision*, pages 225–238, Queenstown, NZ, Nov. 2010.

[22] C. L. Zitnick and P. Dollar. Edge boxes: Locating object proposals from edges. In *Proc. of European Conference on Computer Vision*, pages 391–405, Zurich, CH, Sep. 2014.

[23] J.-C. Zufferey and D. Floreano. Fly-inspired visual steering of an ultralight indoor aircraft. *Trans. on Robotics*, 22(1):137–146, Feb. 2006.