2013 IEEE International Conference on Robotics and Automation (ICRA)
Karlsruhe, Germany, May 6-10, 2013

Moving Target Detection for Sense and Avoid Using Regional Phase Correlation *

Kaaren May and Nicholas Krouglicof, *Member, IEEE*

Abstract— **This paper outlines a video-based method for detecting intruder aircraft to assist with sense and avoid for small, unmanned aerial vehicles (UAVs). A key consideration is that the algorithm is suitable for real-time implementation on field-programmable gate arrays (FPGAs). The method begins by estimating the motion in the scene using regional phase correlation, and then fitting the positional predictions obtained using these regional motion vectors to an affine model representing the effect of camera motion on the background imagery. A combination of metrics, including phase correlation peak height (a confidence measure) and the error between the position predicted by the affine model and that obtained using the measured phase correlation vector, is used to indicate regions of interest where moving targets are present. The ability of the algorithm to detect approaching aircraft is analyzed using a number of aerial video sequences with different encounter geometries.**

I. Introduction

Sense and avoid remains one of the greatest challenges to deployment of UAVs for commercial applications, such as surveillance, environmental monitoring, and traffic control. In a non-cooperative sense and avoid system, the first step is to detect conflicting air traffic using passive (e.g., electro-optical and infrared cameras) or active (e.g., radar, laser) sensors [1]. An overview of current research on sense and avoid is provided in [2].

For small UAVs, size, weight, and power (SWaP) restrictions limit the sensor payload and the ability to do on-board processing. The wide availability of small, low-cost video cameras has made them a popular sensor choice for small UAVs, and high-definition digital video cameras are becoming more common. However, the high pixel density and frame rate pose additional challenges in terms of processing and bandwidth.

Reconfigurable hardware such as an FPGA provides a flexible framework for real-time on-board processing. The advantages offered by FPGAs include massive parallel processing and low power consumption. Furthermore, the availability of soft processor cores such as Altera's NIOS II and Xilinx's MicroBlaze allow tradeoffs between hardware and software to maximize performance. For these reasons, interest in using FPGAs for image processing on board small UAVs is growing. For example, in [3] a Xilinx FX20 FPGA was used for onboard calculation of an affine homography representing frame-to-frame movement for micro UAV applications. In [4] an Elementary Motion Detector array was implemented on a Virtex2 Xilinx FPGA for application to navigation of a micro UAV.

The goal of the current work is to develop a computationally efficient algorithm, suitable for implementation on an FPGA, for detecting conflicting traffic by distinguishing between the apparent motion of the background and the motion of an approaching aircraft. The research forms part of a project to develop an "active vision system" that automatically detects and tracks other airborne targets for use in sense and avoid systems deployed on small UAVs. The overall system design includes several novel features, including replacement of the traditional gimbal used for sensor steering by a three degree-of-freedom parallel kinematic mechanism that employs Printed Circuit Board (PCB)-based voice coil actuators. The resulting device offers fast dynamics, high accuracy, and a large motion range for a small physical volume.

II. Moving Target Detection

A. Detection from a Stationary Platform

The most basic approach to moving target detection involves frame differencing to indicate where changes have occurred, with some additional processing to ensure that the differencing process is robust (e.g., to noise or changes in lighting conditions). However, when the camera is mounted on a moving platform such as a UAV, it is necessary to compensate for the overall scene motion. The goal is to distinguish between the movements of the aircraft (vehicle trajectory and vibration) and the movement of any objects in the field of view of the camera. It should be noted that if an aircraft is approaching head on, then there is no apparent motion, and other detection methods must be used (e.g., [5]).

B. Detection from a Moving Platform

A general approach is to measure motion at a number of points in the scene and to combine the motion measurements to estimate overall scene motion. For instance, in [3], a Harris Feature Detector is combined with a correlator, and then the RANdom Sample Consensus (RANSAC) method is used to calculate an affine homography. While [3] does not proceed further, typically the next step would be to register the current image to the previous image by applying an affine or projective mapping. Moving targets would then be detected by subtracting the registered frames. However, image registration is non-trivial, and small errors in registration may be misinterpreted as motion. Therefore, some post-processing (e.g., tracking) is needed to ensure that spurious differences leading to false alarms are eliminated ([6], [7]).

Another method is to calculate a dense motion field representing local motion over the entire scene, and then to

*Research supported by the Boeing Company, the Atlantic Canada Opportunities Agency, and the Research & Development Corporation of Newfoundland and Labrador.

Both authors are with the Faculty of Engineering and Applied Science, Memorial University of Newfoundland, St. John's, NL A1B 3X5 Canada (e-mail: {kmay, nickk}@mun.ca).

compare this to an estimate of the background motion (e.g., [8], [9]). Common motion estimation approaches, such as those considered in [9] for detection in sense and avoid, include block matching, the Lucas-Kanade method ([11], [12]) and the Horn-Schunck method [13]. However, these approaches have various limitations.

Block matching, for example, takes a block in the current image and searches for a block in another image that provides the best match by minimizing an error measure such as the mean absolute difference. A key challenge with block matching is how to conduct the search efficiently, since for large motions the search region may be very large. Various fast block-matching algorithms have been proposed that eliminate the need for a full search (e.g., [10]). However, block matching tends to generate spurious motion estimates that may not represent the true motion in the scene.

The Lucas-Kanade method is another popular approach to motion estimation. However, it is limited by the assumptions that displacements between frames are small, and that the intensity does not change between frames when this displacement is taken into account. One solution to the small inter-frame displacement issue is to use a multi-resolution approach in which the algorithm is first applied to low-resolution versions of the original images, and then refined at increasing image resolutions (e.g., [12]). However, this adds to the computational complexity. Furthermore, because the Lucas-Kanade method is a local method that relies on spatial intensity gradients, it fails in uniform regions of the image. It also fails if there are significant changes in lighting conditions.

The Horn-Schunck method, by comparison, places a smoothness constraint on the motion field. While this allows motion to be estimated in uniform regions, the smoothness constraint can affect the ability to detect the motion of a small target relative to the background. Furthermore, the method is sensitive to noise.

C. Considerations for Sense and Avoid

A particular challenge in motion estimation for sense and avoid is that very little image detail may be present in large parts of the scene, particularly in the sky region. This can lead to unreliable motion estimates if, for example, weak features in the clouds are used to measure motion. Therefore, an approach such as the visual moving target indicator for UAVs outlined in [14], which detects and tracks point features and then analyses tracked feature positions to detect moving ground targets, can lead to many false alarms. Furthermore, because [14] relies on a multi-stage approach to reduce the number of false alarms, it is unsuitable for a real-time application such as sense and avoid.

Secondly, the method must be robust to significant changes in light intensity in an outdoor environment. Thirdly, it must be able to handle a large range of motions. This is because the motion of the camera platform may result in significant changes in the background scene between frames. Furthermore, the apparent motion of the target will increase as the target gets nearer to the camera, provided that the target is not approaching head on. Finally, the requirement for on-board image processing means that computational complexity must be kept as low as possible.

To address these issues, a regional (i.e., block-based) phase correlation approach is proposed. The initial processing stage follows the method introduced in [15] for motion estimation in broadcast applications such as temporal standards conversion and video compression. These applications share the requirements for real-time implementation in hardware, handling of fast motions, robustness to changes in luminance, and the ability to measure motion in general scenes where feature-based approaches may fail. A computationally efficient way of extending this method for camera motions other than simple panning is proposed.

III. PHASE CORRELATION

Phase correlation is a frequency-domain image registration technique that exploits the translation property of the Fourier transform. According to this property, a translation of an image in the spatial domain is equivalent to a phase shift in the frequency domain. Therefore, if one image is a translated version of another image, i.e., $f_2(x, y) = f_1(x - x_0, y - y_0)$, then the normalized cross power spectrum of the two images can be used to estimate the translational shift (x_0, y_0):

$$Z(u, v) = F_2(u, v)F_1^*(u, v)/(|F_2(u, v)||F_1^*(u, v)|), \quad (1)$$

which is equivalent to

$$Z(u, v) = \exp(-i2\pi(x_0 u/M + y_0 v/N)), \quad (2)$$

where $F_1(u, v)$ and $F_2(u, v)$ are the 2-D discrete Fourier transforms (DFTs) of the M x N images $f_1(x, y)$ and $f_2(x, y)$, and F^* denotes the complex conjugate. The inverse transform of $Z(u, v)$ results in an impulse function whose coordinates correspond to the shift (x_0, y_0). In practice, one image is never a perfect translation of another, and instead of an impulse, a correlation surface $c(x, y)$ is obtained with peaks whose locations represent the translational motion of objects in the images under comparison [15]. The height of a peak serves as a measure of confidence in the corresponding motion estimate.

A windowing function is typically applied to the input images $f_1(x, y)$ and $f_2(x, y)$, in order to reduce the edge effects or "frequency leakage" caused by the periodic nature of the DFT. If the inputs are not windowed, then luminance differences at the left and right (top and bottom) edges of the input images create sharp transitions when the signal is repeated, and the resulting high-frequency components can cause a spurious peak at zero displacement. The disadvantage of windowing is that it reduces the contribution of pixels near the borders of $f_1(x, y)$ and $f_2(x, y)$. Thus, the motion of an object located near a border may not be measured accurately and the velocity range is, in effect, reduced.

While it is possible to estimate multiple motions from a single correlation surface [15], this relies on the objects having fairly distinct frequency signatures. In the current implementation, the image is divided into blocks that overlap by 50% in the horizontal and vertical directions (to compensate for windowing), and only the dominant motion (highest peak) within each block is considered. The use of the highest peak assumes that there is little detail in the background (e.g., sky) over which the object of interest (aircraft) is moving. However, for small objects moving

against a detailed background, the highest peak would likely correspond to the background motion, and therefore it would be necessary to consider multiple peaks.

The optimal block size involves a trade off between two factors: large blocks allow large motions to be measured, but small blocks increase the likelihood that only one object is moving within each block. For the sense and avoid problem, the likelihood that there are several moving targets in one area of the image is very small, since civil aircraft are required to maintain a safe distance from other aircraft.

Example correlation surfaces are provided in Fig. 1. Regions corresponding to an intruder aircraft and to a less detailed cloud region are shown. Note that the "zero" position of the correlation surface has been shifted to the midpoint from the top-left corner. The input blocks are shown before and after windowing in Fig. 1 (a) and (c). It can be seen the displacement of the peak from the center of the correlation surface corresponds to the shift between frames. Furthermore, in the cloud region, where there is little detail that can be used to estimate motion, the corresponding peak height is low. In the absence of any significant detail, the peak location tends towards zero, as it primarily represents the displacement of the windowing function between frames.

Weather (e.g., limited visibility of the intruder aircraft due to clouds or fog) could reduce the peak height corresponding to the aircraft motion. However, the surrounding background would be less visible than the aircraft, and therefore the highest peak should still correspond to the aircraft. That said, in cases where visibility is severely limited, sensors other than EO cameras are more suitable for detection.

Subpixel motion estimates are obtained by interpolating the peak position using the pixels immediately above and below and to the left and right of the highest point on the surface.

A summary of the phase correlation stage of the algorithm is shown in Fig. 2. Note that in a pure phase correlation algorithm, all the frequency bins are, in effect, assigned unit magnitude prior to computing the inverse DFT. However, in the current implementation, the magnitude of the DFT of the block in the previous image is compared against a fixed threshold. If the magnitude exceeds the threshold, then that frequency bin is assigned a magnitude of 1 prior to calculating the inverse DFT. Otherwise, the bin is set to 0. By setting the threshold close to the estimated noise level, the effect of bins that contain little signal information is reduced, thus filtering out some noise on the correlation surface.

It should also be noted that because the input to the forward DFT is real, symmetry properties of the DFT can be exploited so that two DFTs on real inputs are performed as one DFT on a complex input ([15], [16]). Furthermore, the inverse DFT can be computed using a forward DFT algorithm by calculating the DFT of $Z^*(u, v)$, i.e., reversing the sign of the phase differences, which yields $MNc^*(x, y)$. Because the correlation surface is real, $c^*(x, y) = c(x, y)$. Therefore, the measured peak height is simply scaled by $1/MN$ before treating it as a "confidence" measure.

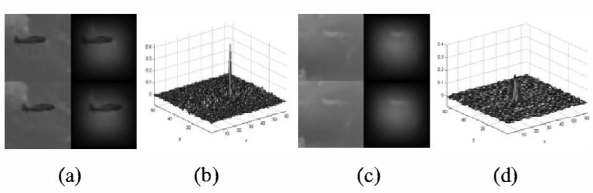

Figure 1. Examples of phase correlation: (a) and (c) are the input blocks from successive frames before and after windowing, and (b) and (d) are the resulting phase correlation surfaces

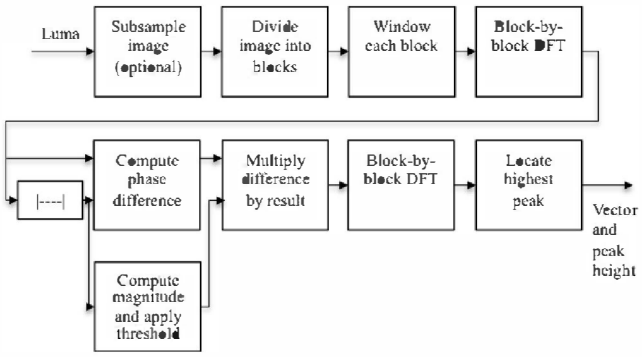

Figure 2. Phase correlation stage

IV. Estimation of Rotation and Scale

When the camera is mounted on a UAV, the motion is not constrained to a simple panning motion, i.e., translation. While phase correlation can also be used to estimate rotation and scale (e.g., [17]), this requires further co-ordinate transformations in the frequency domain. The procedure involves estimating rotation and scale in one step, then compensating for these before estimating the residual translation. This significantly increases the computational complexity.

A simpler approach takes advantage of the fact that the image has been divided into smaller blocks before applying phase correlation. Within each block, it is assumed that the effects of rotation and scale can be ignored. Rotation and scale are instead accounted for by fitting an affine model to the positional predictions obtained using block-based phase correlation, where the phase correlation vector is applied to the pixel at the center of each block. In other words, for each block j, we have:

$$\begin{bmatrix} x'(j) \\ y'(j) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x(j) \\ y(j) \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \end{bmatrix}, \quad (3)$$

where $(x(j), y(j))$ corresponds to the center of the phase correlation block, and $(x'(j), y'(j)) = (x(j) + x_0(j), y(j) + y_0(j))$ using the shift measured from phase correlation. Counter-clockwise rotation by an angle θ about the origin, scaling by a factor s, and translation by (T_x, T_y) results in an equation of the form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = s \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \end{bmatrix}. \quad (4)$$

If we let $\alpha = s\cos\theta$ and $\beta = s\sin\theta$, then:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x & -y & 1 & 0 \\ y & x & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ T_x \\ T_y \end{bmatrix}, \quad (5)$$

where α, β, T_x, T_y are the unknown parameters. If the motion estimates from P phase correlation blocks are available, then there are $2P$ equations in 4 unknowns:

$$\begin{bmatrix} x'(1) \\ \vdots \\ x'(P) \\ y'(1) \\ \vdots \\ y'(P) \end{bmatrix} = \begin{bmatrix} x(1) & -y(1) & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ x(P) & -y(\mathrm{P}) & 1 & 0 \\ y(1) & x(1) & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ y(P) & x(P) & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ T_x \\ T_y \end{bmatrix}, \quad (6)$$

which can be written as $\mathbf{b} = \mathbf{A}\mathbf{x}$, and a least-squares approach provides an estimate of $\mathbf{x}$, where $\mathbf{x} = [\alpha\ \beta\ T_x\ T_y]^{\mathrm{T}}$.

Two modifications can be used to improve estimation of the affine model parameters. The first is to segment the phase correlation motion estimates into two groups: "zero or near zero" vectors and "non-zero" vectors. Only the non-zero vectors are used to estimate the affine model parameters. The rationale is that in regions with little image detail, like the sky, the measured phase correlation vectors will tend towards zero, even though they may not represent the camera motion. Therefore, zero or near zero vectors should be excluded from the affine modeling stage; this can be justified because affine modeling should only be required if the camera is indeed non-stationary. The second modification is that the peak height, which is treated as a measure of confidence, is used to weight the contribution of each phase correlation vector to the estimate $\hat{\mathbf{x}}$ of the affine model parameters. In other words, $\hat{\mathbf{x}} = (\mathbf{A}^{\mathrm{T}}\mathbf{W}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{W}\mathbf{b}$, where $\mathbf{W}$ is a diagonal matrix with diagonal elements $0 \le W_{ii} \le 1$ corresponding to the phase correlation peak height associated with each vector. This reduces the effect of spurious vectors on the affine model, with the aim of obtaining reliable estimates of the affine parameters in a single step, rather than following a multi-stage process in which the initial parameter estimates are used to determine outliers before re-estimating the model.

The affine modeling stage is shown in the context of the entire algorithm in Fig. 3. The error between the position predicted by the affine model and the position obtained using the measured phase correlation vector is computed for each block. For block j, the error is:

$$\begin{aligned} \varepsilon(j) &= |\varepsilon_x(j)| + |\varepsilon_y(j)| \\ &= \left|\begin{bmatrix} x(j) & -y(j) & 1 & 0 \end{bmatrix}\hat{\mathbf{x}} - x'(j)\right| + \\ &\quad \left|\begin{bmatrix} y(j) & x(j) & 0 & 1 \end{bmatrix}\hat{\mathbf{x}} - y'(j)\right|, \end{aligned} \quad (7)$$

where j belongs to the set of "non-zero" phase correlation blocks. Each error $\varepsilon(j)$ is weighted by the corresponding phase correlation peak height, i.e., the confidence measure.

V. Local Deviation

The weighted errors are scaled by a local deviation measure, which gives priority to vectors of large magnitude that differ from the surrounding vectors. The calculation of the local deviation measure is summarized in Fig. 4.

If each image is divided into R x S partially overlapping phase correlation blocks, then X_0 (Y_0) is an R x S image representing the x-component (y-component) of each phase correlation vector, and H is an image of peak heights.

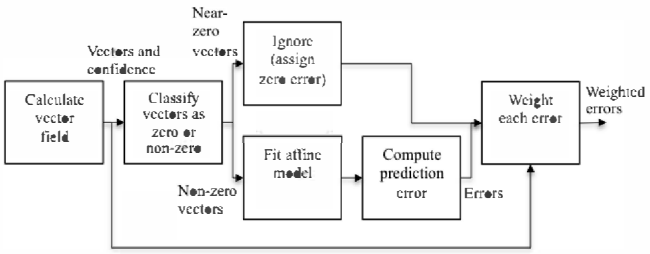

Figure 3. Overview of algorithm

First, the local mean of peak height, $\overline{H}$, is obtained by filtering H using an L x L box (averaging) filter. (As this filter is separable, it can be implemented efficiently as two 1D filtering operations.) To compute the weighted local mean of each vector component, X_0 (Y_0) is multiplied by H on an array (pixel-by-pixel) basis, the result is filtered using the box filter, and this is divided by $\overline{H}$ (again on an array basis) to produce $\overline{X_H}$ ($\overline{Y_H}$). The deviation of the vector from the local mean is then defined as $D = |X_0 - \overline{X_H}| + |Y_0 - \overline{Y_H}|$. This is compared to the deviation from the zero vector, $|X_0| + |Y_0|$, and the minimum is selected, so that lower priority is given to vectors that differ from their surrounding vectors but are small in magnitude. The measure is normalized so that the maximum value for each vector field (i.e., video frame) is 1.

Thus, the estimated level of threat for each block depends on how much the phase correlation motion vector differs from the camera motion model, the confidence in the motion vector estimate (peak height), how much it deviates from vectors in the immediate neighborhood, and the magnitude.

VI. Results

The algorithm was tested using actual encounter data provided by the National Research Council's Flight Research Laboratory. This was recorded using a Bell 205 helicopter as a surrogate UAV experimental platform. A Prosilica GC2450 monochrome camera (2448 x 2050 @ 15 fps) was mounted on an anti-vibration mount. A block size of 256 x 256 was used on the uncompressed video, yielding 18 x 15 correlation surfaces for each frame. Six different sequences were tested. Simulations were also performed on images subsampled by 4 along each dimension; in this case a block size of 64x64 was used to keep the number of correlation surfaces constant. The accompanying video shows three of these sequences after processing, in the case that no subsampling was applied. The output video was subsampled and compressed for display purposes after all processing had been completed.

For the initial evaluation of performance, the weighted errors from the affine modeling stage were multiplied by the local deviation measure to obtain an estimate of the level of threat for each block. This was scaled by a fixed scaling factor of 32 and mapped to the red color difference channel (Cr) of the output video using the color encoding defined in ITU-R BT.601 [18]. Values exceeding the maximum permitted level for Cr were clipped. The strength of the red

signal was intended to represent the estimated level of threat, thereby highlighting regions of interest to an operator. The phase correlation vectors were displayed at the center of each block for diagnostic purposes.

Since the phase correlation blocks overlapped, the threat measure for each block was displayed over the central (128x128) portion of each block. In some cases, this meant that the highlighted area was offset slightly from the actual target, even though the target was contained within the corresponding block. This display issue can be addressed by upsampling the threat measure using bilinear interpolation.

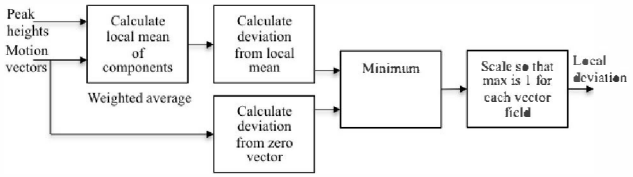

Figure 4. Calculation of local deviation

Example output frames are shown in Fig. 5 for a number of sequences. The same parameters were used in all cases.

The output frames were selected to illustrate typical performance and highlight some of the issues. Fig. 5 (d) shows that occasionally, when the target is not present, some regions are still highlighted, but the faintness of the red channel indicates that the estimated level of threat is low. A similar effect is seen in Fig. 5 (i), where in addition to the correct target, a small region in the sky is highlighted. This phenomenon mostly occurs in the relatively uniform regions of the sky, where the lack of detail makes motion estimation difficult. While the resulting vectors in these regions tend towards zero, they may not fall below the defined threshold for "zero or near zero vectors". Furthermore, scaling directly by the peak height (as opposed to some nonlinear function of this) may not best represent the relative "confidence" level of a vector estimated from a region with very little detail compared to that derived from a block containing strong features. Fig. 5 shows that the approach can cope with significant camera rotation and handle large changes in target size and motion.

For quantitative evaluation of performance, ground truth was generated for a subset of the sequences using a semi-automatic approach. Using the full resolution input, a rectangular region containing the target (if present) was manually identified for each frame. Within this region, an initial segmentation of the target and background was obtained by grey-level thresholding, and morphological opening was used to remove isolated pixels incorrectly classified as belonging to the target. The segmentation was verified and modified manually as required. The resulting output was a full-resolution binary image for each frame indicating the pixels belonging to the target. If the input to the algorithm had been downsampled, then the ground truth was downsampled in a similar manner and all non-zero pixels assigned to the target. Thus, performance was evaluated as if the sequence had been recorded at the lower resolution.

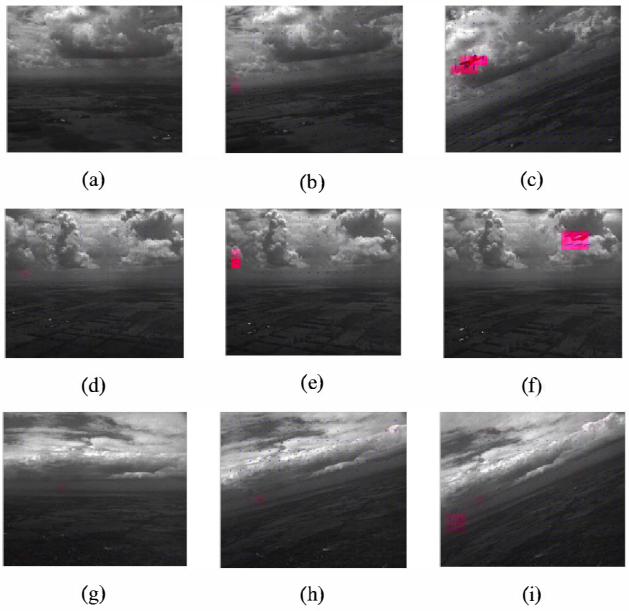

Figure 5. Example outputs: (a)-(c) are from sequence NRC04, (d)-(f) are from NRC05, and (g)-(i) are from NRC01

The estimated threat level was thresholded in order to generate detection decisions. For detecting conflicting traffic in sense and avoid, it is important to minimize false alarms (or false positives) as well as misses (false negatives) [1]. Thus, three measures were used to quantify performance [19]:

- Detection rate = $N_{tp}/(N_{tp} + N_{fn})$
- False positive rate = $N_{fp}/(N_{fp} + N_{tn})$
- Accuracy = $(N_{tp} + N_{tn})/N$

where N_{tp} was the number of true positives within each sequence, N_{fp} was the number of false positives, N_{fn} was the number of false negatives, N_{tn} was the number of true negatives, and $N = N_{tp} + N_{fp} + N_{fn} + N_{tn}$. Because the false negative rate is equal to (1 – detection rate), maximizing the detection rate implies minimizing the false negative rate or misses. The delay between the frame in which the target first appeared and when it was first detected was also measured.

Because each pixel was contained within four overlapping phase correlation blocks (except at the image borders), the target could be present in multiple phase correlation blocks. However, due to the windowing function, the strength of the target signal depended on its proximity to the block's center. For the purpose of measuring N_{tp} and N_{fn} over a test sequence, N_{tp} was incremented if the target was present in a frame and at least one phase correlation block that overlapped the target exceeded the detection threshold. If the target overlapped any phase correlation blocks in a frame, but none of these exceeded the detection threshold, then N_{fn} was incremented. "Overlap" was defined as the block containing the equivalent of at least 1.5 target pixels (based on Johnson's criteria for detection [20]), with the ground truth being weighted by the windowing function.

Results are given in Table 1 to Table 3. For each sequence, performance is compared for the subsampled

versus full-resolution input. As expected, the effect of subsampling is most pronounced in NRC01, as the target is distant and therefore extends over few pixels. In NRC04 and NRC05, the false negatives tend to occur as the target enters or leaves the field of view and is only partly visible. There are several reasons: the adverse effect of windowing when the target is close to a block edge, the target being present in fewer overlapping blocks, and the apparent change of shape as the target enters or leaves. With a threshold of 0.05 and no subsampling, the detection rate ranges from 96.8% to 98.5%, with a false positive rate of 0.8% to 0.9%.

TABLE I. RESULTS FOR NRC04

Performance	With subsampling by 4			Without subsampling		
	Threshold			Threshold		
	0.05	*0.1*	*0.2*	*0.05*	*0.1*	*0.2*
Detection rate	96.8%	91.9%	87.1%	96.8%	88.7%	87.1%
False positive rate	2.8%	0.7%	0.05%	0.8%	0.06%	0.02%
Accuracy	97.2%	99.3%	99.93%	99.2%	99.9%	99.96%
Detection delay (frames)	1	3	7	1	4	7

TABLE II. RESULTS FOR NRC05

Performance	With subsampling by 4			Without subsampling		
	Threshold			Threshold		
	0.05	*0.1*	*0.2*	*0.05*	*0.1*	*0.2*
Detection rate	93.2%	93.2%	93.2%	97.7%	97.7%	97.7%
False positive rate	2.4%	0.7%	0.04%	0.9%	0.1%	0.02%
Accuracy	97.6%	99.3%	99.96%	99.1%	99.9%	99.98%
Detection delay (frames)	1	1	1	0	0	0

TABLE III. RESULTS FOR NRC01

Performance	With subsampling by 4			Without subsampling		
	Threshold			Threshold		
	0.05	*0.1*	*0.2*	*0.05*	*0.1*	*0.2*
Detection rate	75.4%	73.9%	66.2%	98.5%	92.3%	63.1%
False positive rate	1.0%	0.4%	0.05%	0.8%	0.2%	0.05%
Accuracy	98.9%	99.6%	99.8%	99.2%	99.8%	99.8%
Detection delay (frames)	1	1	1	0	3	3

VII. CONCLUSION

The method presented is intended to flag regions of interest where the motion differs significantly from the background motion, for detecting intruder aircraft in sense and avoid. The block-based phase correlation approach has a number of advantages, including: a parallel processing structure that makes it suitable for real-time implementation on an FPGA; robustness to noise and changes in illumination; and the ability to cope with targets of varying size and speed, as well as regions lacking strong features. Initial results indicate that the requirements for low false negative and false positive rates can be achieved for various encounter geometries using a single set of algorithm parameters. However, a fundamental assumption is that the motion of the intruder aircraft differs from the background motion. When this is not the case (e.g., an aircraft is approaching head on) supplementary detection methods must be used, such as measuring the rate of object expansion. Future work will also involve developing the real-time hardware implementation, which will enable further testing and refinement of the algorithm to ensure robust performance.

ACKNOWLEDGMENT

The authors would like to thank the Flight Test Laboratory at the National Research Council's Institute for Aerospace Research in Ottawa for providing test video.

REFERENCES

[1] S. B. Hottman, K. R. Hansen, and M. Berry, "Literature review on detect, sense, and avoid technology for unmanned aircraft systems", Tech. Report DOT/FAA/AR-08/41, US Dept. of Transport, 2009.

[2] P. Angelov, Ed, *Sense and avoid in UAS: research and applications.* Chichester, UK: Wiley, 2012.

[3] B. Tippetts, S. Fowers, K. Lillywhite, D.-J. Lee, and J. Archibald, "Implementation of a feature detection and tracking algorithm for real-time applications", in *Proc. 3rd Int. Conf. on Advances in Visual Computing*, Part I, pp. 682-691, 2007.

[4] F. Aubépart and N. Franceschini, "Bio-inspired optic flow sensors based on FPGA: Application to micro-air-vehicles", *Microprocessors and Microsystems*, vol. 31, pp. 408–419, Sept. 2007.

[5] R. J. Carnie, R. A. Walker, and P. I. Corke, "Computer vision based collision avoidance for UAVs", in *Proc. 11th Australian Int. Aerospace Congress*, Melborne, Australia, 2005.

[6] M. Shah, H. Asad, and A. Basharat, "Detection and tracking of objects from multiple airborne cameras," SPIE Newsroom, 2006.

[7] I. Cohen and G. Medioni, "Detection and tracking of objects in airborne video imagery", in *Proc. of Workshop on Interpretation of Visual Motion*, 1998.

[8] K. Nordberg, P. Doherty, G. Farnebäck, P.-E. Forssén, G. Granlund, A. Moe, and J. Wiklund, "Vision for a UAV helicopter", in *Proc. of IROS'02, Workshop on Aerial Robotics*, Lausanne, Switzerland.

[9] L. Ng, P. Hubbard, P. Pace, K. Ellis, and S. O'Young, "Detection techniques for sense and avoid in unmanned air vehicles", presented at the Unmanned Systems Canada Conference, Montreal, 2010.

[10] Y.-S. Chen, Y.-P. Hung, and C.-S. Fuh, "Fast block matching algorithm based on the winner-update strategy", *IEEE Trans. on Image Processing*, vol. 10, no. 8, pp. 1212-1222, Aug. 2001.

[11] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision", in *Int. Joint Conf. on Artificial Intelligence*, pp. 674-679, 1981.

[12] C. Tomasi and T. Kanade, "Detection and tracking of point features", Carnegie Mellon Univ. Tech. Report CMU-CS-91-132, Apr. 1991.

[13] B. K. P. Horn and B. G. Schunck, "Determining optical flow", *Artificial Intelligence*, vol. 17, pp. 185-203, Aug. 1981.

[14] R. Evans and E. Turkbeyler, "Visual MTI for UAV systems", in *4th EMRS DTC Technical Conf.*, Edinburgh, July 2007.

[15] G. A. Thomas, "Television motion measurement for DATV and other applications", Technical Report BBC RD 1987/11.

[16] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd ed., New Jersey: Pearson, 2008, ch. 4.

[17] L. Hill and T. Vlachos, "On the estimation of global motion using phase correlation for broadcast applications", in *Proc. of 7th Int. Conf. on Image Processing and Its Applications,* vol. 2, pp. 721–725, 1999.

[18] International Telecommunication Union Radiocommunication Sector, "Studio encoding parameters of digital television for standard 4:3 and wide screen 16:9 aspect ratios", Rec. BT.601-7 (03/11), 2011.

[19] T. Ellis, "Performance metrics and methods for tracking in surveillance", in *3rd IEEE Workshop on Performance Evaluation of Tracking and Surveillance*, 2002.

[20] J. Johnson, "Analysis of Imaging Forming Systems", in *Proc. of the Image Intensifier Symposium*, Ft. Belvoir, VA, pp. 244 – 273, 1958.