Airborne Vision-based Collision-Detection System

John Lai[*]
Australian Research Centre for Aerospace Automation (ARCAA)
Queensland University of Technology
GPO Box 2434, Brisbane Queensland 4001
js.lai@qut.edu.au

Luis Mejias
Australian Research Centre for Aerospace Automation (ARCAA)
Queensland University of Technology
GPO Box 2434, Brisbane Queensland 4001
luis.mejias@qut.edu.au

Jason J. Ford
Australian Research Centre for Aerospace Automation (ARCAA)
Queensland University of Technology
GPO Box 2434, Brisbane Queensland 4001
j2.ford@qut.edu.au

Abstract

Machine vision represents a particularly attractive solution for sensing and detecting potential collision-course targets due to the relatively low cost, size, weight, and power requirements of vision sensors (as opposed to radar and TCAS). This paper describes the development and evaluation of a real-time vision-based collision detection system suitable for fixed-wing aerial robotics. Using two fixed-wing UAVs to recreate various collision-course scenarios, we were able to capture highly realistic vision (from an onboard camera perspective) of the moments leading up to a collision. This type of image data is extremely scarce and was invaluable in evaluating the detection performance of two candidate target

[*]Corresponding author; Tel.: +61 7 3138 9808; Fax: +61 7 3138 1516.

detection approaches. Based on the collected data, our detection approaches were able to detect targets at distances ranging from 400m to about 900m. These distances (with some assumptions about closing speeds and aircraft trajectories) translate to an advanced warning of between 8-10 seconds ahead of impact, which approaches the 12.5 second response time recommended for human pilots. We overcame the challenge of achieving real-time computational speeds by exploiting the parallel processing architectures of graphics processing units found on commercially-off-the-shelf graphics devices. Our chosen GPU device suitable for integration onto UAV platforms can be expected to handle real-time processing of 1024 by 768 pixel image frames at a rate of approximately 30Hz. Flight trials using manned Cessna aircraft where all processing is performed onboard will be conducted in the near future, followed by further experiments with fully autonomous UAV platforms.

1 Introduction

The problem of unmanned aerial vehicle (UAV) collision avoidance or 'sense-and-avoid' has been identified as one of the most significant challenges facing the integration of UAVs into the national airspace (Unmanned Aircraft Systems Roadmap 2007–2032, 2007; DeGarmo, 2004). The full potential of UAVs can never be realised unless the sense-and-avoid issue is adequately addressed. To this end, machine vision has emerged as a promising means of addressing the 'sense' and 'detect' aspects of collision avoidance, which demands the ability to automatically detect and track targets in naturally lit, high noise environments.

Machine vision represents a particularly attractive solution for sensing and detecting potential collision-course targets due to the relatively low cost, size, weight, and power requirements of the sensors involved (Maroney et al., 2007). Man-in-the-loop (MITL) solutions have been prototyped and demonstrated (Bryner, 2006). Furthermore, an automated detection system developed by Defense Research Associates was implemented and flight tested, where the detection approach is based on target pixel movement (or energy flow) in the image frame (Utt et al., 2005; Utt et al., 2004). Recently, the problem of aircraft detection using passive vision has been tackled using 1) a combination of morphological filtering and support vector machine classification (Dey et al., 2009; Geyer et al., 2009), and 2) a machine learning approach based on AdaBoost that exploits a modified version of the Viola and Jones object detector (Petridis et al., 2008).

There are many hurdles that must be overcome in the use of machine vision for target detection and tracking. We need to contend with not only the inherent noise of imaging sensors, but also with noise introduced by changing and unpredictable ambient conditions. The desire to overcome these challenges has driven the development of specialized image filtering and processing techniques that are optimized for the type of dim-target characteristics that are experienced in near collision events between fixed-wing aircraft or fixed-wing aerial robots.

Over the last three decades, a two-stage processing paradigm has emerged for the simultaneous detection and tracking of dim, sub-pixel sized targets (Gandhi et al., 2006; Gandhi et al., 2003; Arnold et al., 1993; Barniv, 1985). These two stages are: 1) an image pre-processing stage that, within each frame, highlights potential targets with attributes of interest; and 2) a subsequent temporal filtering stage that exploits target dynamics across frames. The latter temporal filtering stage is often based on a track-before-detect processing concept where target information is collected and collated over a period of time before the detection decision is made.

Generally, the goal of the image pre-processing stage is to enhance potential target features whilst suppressing background noise and clutter. There is an abundance of techniques and algorithms available which may be considered for this image processing role. In particular, non-linear spatial techniques such as median subtraction filters (Deshpande et al., 1999) have been widely discussed in the literature. Another non-linear image filtering approach that has received much attention over the last decade has its basis in mathematical morphology (Dougherty and Lotufo, 2003). Numerous morphology-based filters have been proposed for the detection of small targets in infrared (IR) images (Zhu et al., 2000; JiCheng et al., 1996; Tom et al., 1993). Specific implementations of the morphological filtering approach include the Hit-or-Miss filter (Schaefer and Casasent, 1995), Close-Minus-Open filter (Casasent and Ye, 1997), and the Top-Hat filter (Braga-Neto et al., 2004). Although a large proportion of research has focused on IR images, there are recent examples of morphological filters being incorporated into target detection algorithms operating on visual spectrum images (Carnie et al., 2006; Gandhi et al., 2006; Gandhi et al., 2003). Moreover, a sign of the increasing popularity of morphological filters for small target detection is evident in the host of studies undertaken into the issue of parameter design (Zeng et al., 2006; Yu et al., 2003). Finally, there have been efforts made to compare existing techniques with the morphology-based filters (Gandhi et al., 2006; Warren, 2002; Tom et al., 1993; Barnett et al., 1993), with the median filtering technique often featuring in the comparison studies.

On the other hand, the temporal filtering stage that follows the image pre-processing is designed to extract image features that possess target-like temporal behaviour. For this role, there are two particular filtering approaches that have received much attention in the literature: Viterbi based approaches and Bayesian based approaches. The Viterbi algorithm has formed the basis of the temporal filtering stage in numerous track-before-detect algorithms (Davey et al., 2008; Gandhi et al., 2006; Tonissen and Evans, 1996; Arnold et al., 1993; Barniv, 1985). The popularity of the Viterbi algorithm is in part due to its utility in the context of tracking where, under a number of assumptions, it is able to efficiently determine the optimal target track within a data sequence (Forney, 1973). Some analysis of the Viterbi algorithm's detection and tracking performance have been conducted (Johnston and Krishnamurthy, 2000; Tonissen and Evans, 1996; Barniv and Kella, 1987), and modifications that enhance the algorithm's tracking performance in the presence of non-Gaussian clutter noise have been proposed (Arnold et al., 1993). An alternative temporal filter design for track-before-detect algorithms is based on Bayesian filtering (Davey et al., 2008; Bruno, 2004; Bruno and Moura, 2001; Bruno and Moura, 1999). Advances in this filtering approach have considered the relaxation of typical white Gaussian noise assumptions and spatially correlated clutter (Bruno and Moura, 1999). Moreover, the modeling of clutter has been expanded to encompass a variety of Gaussian and non-Gaussian, correlated and uncorrelated clutter types, and the Bayesian algorithm is extended to accommodate multiple targets that may feature randomly varying amplitudes or intensities (Bruno and Moura, 2001). Finally, some comparison between the Viterbi and Bayesian approaches has been made at the theoretical level (Bruno and Moura, 2001), as well as on the practical level via Monte Carlo simulation trials (Davey et al., 2008).

A survey of potential technologies for unmanned aerial vehicle (UAV) sense-and-avoid concluded that the visual/pixel based technology offered the best chances for regulator approval (Karhoff et al., 2006). It is interesting to note that some studies have shown that it is actually difficult to detect and avoid a collision using the human visual system (Limitations of the See-and-Avoid Principle, 1991). To date, public domain hardware implementation of vision-based sense-and-avoid systems have been limited to a small number. Arguably, the most significant developments have been made by (Utt et al., 2005), where a combination of field programmable gate array chips and microprocessors using multiple sensors were tested in a twin-engine Aero Comander aircraft. A challenge that faces any vision-based sense-and-avoid system is the requirement of real-time operation. Motivated by this fact, we exploit the capabilities of data-parallel arithmetic architectures such as Graphics Processing Units (GPUs), which have proven to be very capable parallel processing devices that can outperform current CPUs by up to an order of magnitude (Owens et al., 2005).

The key contributions of this paper are: 1) demonstration of the coordinated flight of fixed-wing UAVs performing collision-course scenarios for collection of suitable test image data; 2) application of HMM and Viterbi-based target detection approaches to a computer vision sense-and-avoid problem; 3) analysis of the target detection approaches in terms of maximum detection range; and 4) implementation of the target detection approaches using GPU-based hardware and the demonstration of real-time detection capabilities. The use of fixed-wing UAVs for data collection presents unique challenges, particularly in relation to the synchronisation of aircraft flights to ensure a realistic scenario is replicated, and to maximise the target aircraft's time spent in the camera field of view. Furthermore, we highlight that we are particularly interested in determining the performance limits of passive machine vision for detecting collision-course aircraft, and hence our focus on the maximum detection range of our candidate detection algorithms. We acknowledge that the detection range will be influenced by various environmental conditions (such as weather and lighting); however, a detailed characterisation of the quality of the collected data is beyond the scope of this paper. Finally, we also provide some insight into the impact of image jitter on the performance of our candidate target detection approaches, as it has been previously shown that detection performance can be quite sensitive to jitter effects (Utt et al., 2005).

This paper is structured as follows. Section 2 introduces the basic characteristics of collision behaviour that make vision-based collision detection a difficult problem. Section 3 separately introduces the HMM and Viterbi based detection approaches examined in this paper. Section 4 evaluates the performance of the proposed collision detection system in three key areas: 1) the resilience of the detection algorithms to undesired camera motion (image jitter); 2) the target detection capability of the detection algorithms; and 3) the real-time processing capacity of a GPU-based hardware implementation. Section 5 describes some of the lessons learnt and future work planned.

2 Basic Characteristics of Potential Collision-Course Objects

It may be possible to detect collision course objects based on their physical appearance or the dynamics that they exhibit. The physical attributes of collision course objects, such as colour, brightness, shape, and size, depend largely on ambient lighting and atmospheric conditions, as well as the distance to the object. Sophisticated models have been devised in an attempt to describe and predict these observed target attributes as a function of distance, taking into account the effects of haze, atmospheric scattering, and potential defocus blur due to poor or mismatched optics (Geyer et al., 2009). However, it is difficult to accurately model the complex and unpredictable nature of lighting and atmospheric conditions, so we choose to instead exploit some simple but reliable characteristics of collision-course objects which are sufficient for our purposes of determining the maximum detection range of our candidate algorithms. More specifically, we elect to exploit the typical size, shape, and dynamics of a collision-course object on the image plane of a camera. We do not claim that any of these characteristics alone can uniquely identify a collision-course object, but we believe a combination of the three has the ability to eliminate the majority of non-genuine targets. Given the 12.5 second reaction time recommended for human pilots (FAA Advisory Circular: Pilots' role in collision avoidance, 1983), a collision-course object must be detected at a distance of over 1 kilometer (assuming a closing velocity of 100 m/s) to avoid a collision[1]. At this distance, aircraft and other objects of similar dimensions may take up an area anywhere from a few pixels to less than one pixel on the image plane[2]. It can be argued that a few pixels cannot really define any sort of 'shape', but at least it can be deduced that in the context of early detection, collision-course objects will tend to be small, point-like features, becoming smaller and dimmer the earlier that detection is required. Of all the characteristics of collision-course objects, their somewhat unique dynamics is perhaps the most suitable attribute to exploit. Objects on a collision course appear at the output of a fixed onboard vision sensor as relatively stationary features on the image plane (Limitations of the See-and-Avoid Principle, 1991). Features that are moving rapidly across the image plane do not correspond to collision-course objects.

The typical output from an computer vision sensor is a sequence of images (i.e. a video stream). Detecting collision-course objects is then a matter of searching for objects within the image sequence that possess the above characteristics; that is, dim sub-pixel size targets that are slowly moving in the image frame. These target properties correspond to the two stage detection paradigm that is described in the next section: morphological filtering to detect pin-like targets, and temporal filtering to detect persistent or almost stationary features.

We are aware of existing guidelines and requirements (Standard specification for design and performance of an airborne sense-and-avoid system, F2411-04, 2004; Office of the Secretary of Defense, 2004; Ebdon and Regan, 2004; FAA Advisory Circular: Pilots' role in collision avoidance, 1983) stipulating functional capabilities that a sense-and-avoid system must satisfy to gain regulatory approval. These are not inconsistent with our earlier characterisation of collision-course targets; they are merely specifications at a higher functional level which we will consider at the next stage of our research once we have determined the performance limits of a vision-based detection approach. For a detailed analysis of current regulatory requirements see (Geyer et al., 2008).

[1]Even though an automated system is likely to require less time to recognise threats and take evasive action, it can be argued that in the interests of safety, it is best to detect targets as early (i.e. as far away) as possible.

[2]Depending on camera resolution, field of view etc.

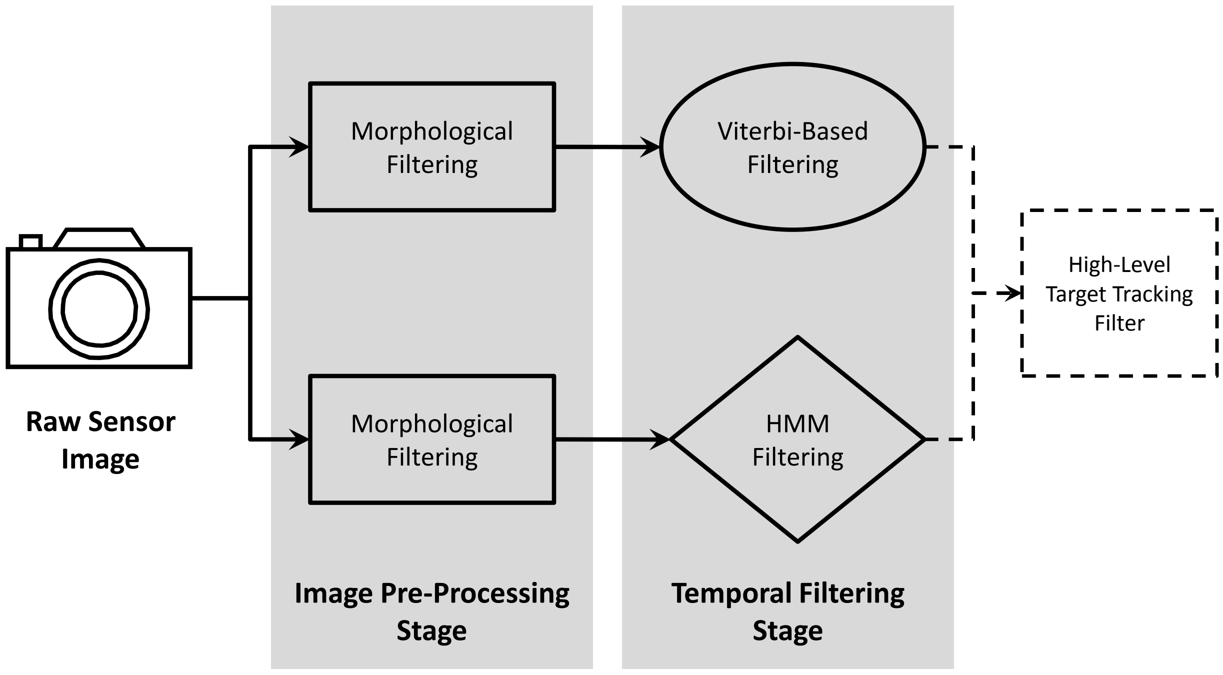


Figure 1: Candidate detection algorithms.

3 Detection Algorithms

The two candidate detection algorithms that will be considered in this paper are designed to process the sensor measurements in two stages: 1) image pre-processing, followed by 2) track-before-detect temporal filtering. Both algorithms will share a common 'Close-Minus-Open' morphological image pre-processing stage, but one approach will be coupled to a hidden Markov model temporal filter, whereas the other approach will be coupled to a Viterbi-based temporal filter, as illustrated in Figure 1. We highlight that after detection has occurred the position estimate output of our candidate detection algorithms could then be passed to a high-level target tracking filter (such as an extended Kalman filter). This has the potential to improve tracking performance, the analysis of which will be considered in future investigations.

3.1 Morphological Image Pre-Processing

This paper considers an image pre-processing technique that exploits grayscale morphological operations in order to process discrete 2D image data quantized to a finite number of intensity or grayscale levels, such as might be expected from the output of an electro-optical sensor. In particular, the Close-Minus-Open (CMO) morphological filter used is based on image morphology operations known as *top-hat* and *bottom-hat* transformations (Gonzalez et al., 2004). The effect of the top-hat transformation is to identify positively contrasting (brighter than background) features within an image that are smaller than a certain size (the cut-off size is specified through filtering kernels known as structuring elements), while the bottom-hat transformation performs a similar function but instead targets negatively contrasting (darker than background) features. It can be shown that summing the top-hat and bottom-hat transformations of the same image, which defines the Close-Minus-Open filtering operation, simultaneously identifies both positively and negatively contrasting features. This combination of morphological operations has been referred to elsewhere in the literature as a self-complementary top-hat filtering approach (Soille, 2003).

In this paper, the CMO filter is configured to serve as a powerful tool in the identification of small point-like features within the measurement image. For performance and computational reasons, the CMO filter implemented in this paper exploits a directional decomposition technique (Casasent and Ye, 1997). More specifically, the minimum response from a pair of CMO filters using orthogonal 1D structuring elements is used. Here, one CMO filter operates exclusively in the vertical direction, while the other operates exclusively in the horizontal direction. The vertical and horizontal structuring elements of the CMO morphological pre-processing filter are given by $s_v = [1, 1, 1, 1, 1]^{'}$ and $s_h = [1, 1, 1, 1, 1]$, respectively. An example of the output after CMO morphological pre-processing is illustrated in Fig. 4b.

3.2 Temporal Filtering

In many machine vision based detection problems, the existence of a target in a 3D volume of space must be determined from observations of a projection of the target space onto a 2D image plane. Here, target detection can be viewed as evaluating the likelihood of 2 alternate hypotheses, where H_1 denotes the hypothesis that there is a single target present in the camera field of view, and H_2 denotes the hypothesis that there is no target present.

The temporal filtering approaches implemented in this paper will assume that under hypothesis H_1, the projected target motion resides on a 2D plane fixed in space that is represented by the set of discrete 2D grid points $\{(i,j)|\, 1 \le i \le N_v, 1 \le j \le N_h\}$, with vertical and horizontal resolutions N_v and N_h respectively. Let $N = N_v \times N_h$ denote the total number of grid points. The measurements are provided by an imaging sensor whose field of view is represented by a 2D grid of image pixels locations aligned with the target space and denoted $\{(p,q)|\, 1 \le p \le N_v, 1 \le q \le N_h\}$.

3.2.1 Hidden Markov Model Filtering

It will be assumed that, when present, the target is located within a particular pixel of the image frame at each time instant. Thus, each pixel (i,j) represents a unique *state* of the HMM in the target detection problem. For notational convenience, the columns of the image frame are stacked to form a vector of pixel locations. In this way, each state may be referenced by a single index, in the sense that if the target is at pixel location (i,j), this corresponds to it being in the state $m = [(j-1)N_v + i]$.

Let $\mathbf{x}_k$ denote the state (target location) at time k. Between consecutive image frames the target may move to different pixel locations; that is, the target can transition between the states. The likelihood of state transitions can be described by the HMM's *transition probabilities* $\mathbf{A}^{mn} = P(\mathbf{x}_{k+1} = \text{state } m|\, \mathbf{x}_k = \text{state } n)$ for $1 \le m, n \le N$, which is the probability of moving from any one pixel position (state) n to any other pixel position (state) m. The transition probabilities can therefore be used to describe the expected mean target motion. For example, in the case of slow moving targets low probabilities tend to be assigned for transitions between distant pixels. Moreover, *initial probabilities* $\boldsymbol{\pi}^m = P(\mathbf{x}_1 = \text{state } m)$ for $1 \le m \le N$ are used to specify the probability that the target is initially located in state m. Finally, to complete the parameterisation of the HMM, there are the *measurement probabilities* $\mathbf{B}^m(\mathbf{Y}_k) = P(\mathbf{Y}_k|\, \mathbf{x}_k = \text{state } m)$ for $1 \le m \le N$ that are used to specify the probability of obtaining the observed image measurement $\mathbf{Y}_k \in R^{N_v \times N_h}$, given that the target is actually in pixel location (state) m (see (Elliott et al., 1995) for more details about the parameterisation of HMMs). A possible approach to estimate the probabilistic matrices $\mathbf{A}$ and $\mathbf{B}$ is discussed later on in the implementation of a HMM filter bank.

HMM Detection Strategy The HMM filtering approach performs temporal integration of the input measurements by recursively propagating $\boldsymbol{\alpha}_k^i$, an unnormalised probabilistic estimate of the target state $\mathbf{x}_k^i$, over time. This is achieved via the forward part of the forward-backward procedure (Rabiner, 1989), which can be decomposed into two stages: initialisation and recursion.

For $1 \le m \le N$

1. Initialisation: Let $\boldsymbol{\alpha}_k^m$ denote the probability $P\left(\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_k, \mathbf{x}_k = \text{state } m\right)$. Then $\boldsymbol{\alpha}_1^m = \boldsymbol{\pi}^m \mathbf{B}^m\left(\mathbf{Y}_1\right)$.

2. Recursion: At time $k > 1$, set $\boldsymbol{\alpha}_k^m = \left[\sum_{n=1}^{N} \boldsymbol{\alpha}_{k-1}^n \mathbf{A}^{mn}\right] \mathbf{B}^m\left(\mathbf{Y}_k\right)$.

The forward procedure filtering result is closely related to the two probabalistic measures that facilitate the detection of targets: 1) the probability of measurements up to time k assuming H_1, given by

$$P\left(\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_k \middle| H_1\right) = \sum_{m-1}^{N} \boldsymbol{\alpha}_k^m, \tag{1}$$

and 2) the conditional mean filtered estimate of the target state m given measurements up to time k and assuming H_1, given by

$$\begin{aligned} \hat{\mathbf{x}}_k^m &= E\left[\mathbf{x}_k = \text{state } m \middle| \mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_k, H_1\right] \\ &= \frac{\boldsymbol{\alpha}_k^m}{\sum_{n=1}^{N} \boldsymbol{\alpha}_k^n}, \end{aligned} \tag{2}$$

where $E\left[.|.\right]$ denotes the mathematical conditional expectation operation (Billingsley, 1995). The probability $P\left(\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_k \middle| H_1\right)$ may be interpreted as an indicator of target presence (following the probabilistic distance results of (Xie et al., 2005)), and the conditional mean estimate can be regarded as an indicator of likely target locations.

In the interest of computational efficiency, the conditional mean estimate is evaluated directly from the following expression (Elliott et al., 1995):

$$\hat{\mathbf{x}}_k = N_k \mathbf{B}_k\left(\mathbf{Y}_k\right) \mathbf{A} \hat{\mathbf{x}}_{k-1}, \tag{3}$$

where N_k is a scalar normalisation factor; $\mathbf{B}_k\left(\mathbf{Y}_k\right)$ is a $N \times N$ matrix where the main diagonal is occupied by the values of $\mathbf{B}^m\left(\mathbf{Y}_k\right)$ for $1 \leq m \leq N$ and all other elements are zero; $\mathbf{A}$ is a $N \times N$ matrix with elements $\mathbf{A}^{mn}$; and $\hat{\mathbf{x}}_k$ is a $N \times 1$ vector consisting of elements $\hat{\mathbf{x}}_k^m$ for $1 \leq m \leq N$ that are equivalent to those given in (3). Moreover, note the following relationship between the normalisation factor N_k and the probability of measurements up to time k assuming H_1:

$$P\left(\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_k \middle| H_1\right) = \prod_{l=1}^{k} \frac{1}{N_l}. \tag{4}$$

For the HMM filtering approach, let η_k, the test statistic for declaring the presence of a target, be given by the following exponentially weighted moving average filter with a window length of L:

$$\eta_k = \left(\frac{L-1}{L}\right)\eta_{k-1} + \left(\frac{1}{L}\right)\log\left(\frac{1}{N_k}\right). \tag{5}$$

In the later performance evaluation studies, we found a window length of $L = 10$ produced good detection results. A shorter window length may give earlier detections, but this is likely to be at the expense of increased false alarms. When η_k exceeds a predefined threshold, the HMM detection algorithm considers the target to be present and located at state

$$\gamma_k = \arg\max_m \left(\hat{\mathbf{x}}_k^m\right) \tag{6}$$

at time k. The definition of η_k and γ_k is motivated by the filtering quantities discussed earlier.

HMM Filter Bank Previous studies have shown that a multiple filter approach (filter bank) provides superior detection performance compared with a standard single filter implementation under a range of target speed and signal-to-noise ratio scenarios (Lai et al., 2008). The superior performance of the filter bank approach can be attributed largely to the collective ability of multiple filter models to provide a more precise description of the target dynamics (however, this more precise modelling of the target dynamics can have drawbacks when the modelling assumptions are violated; see Section 4.1). In this paper, a HMM filter bank consisting of four filters is implemented (Lai et al., 2008). Each HMM filter in the bank uses the same pre-processed image data, but otherwise operates independently of all other filters. The filter bank approach is less well characterised than the standard single HMM filter, and its application has not been prevalent in the context of dim-target detection from imaging sensors.

The transition probability parameters of each filter in the HMM filter bank are designed to handle target motion where the target either remains stationary or moves to an adjacent neighbouring pixel between consecutive image frames. In practice when the system is liable to the effects of image jitter, we allow for the possibility of larger inter-frame target motion. For example, Figure 2 illustrates a 3-by-3 patch of the possible transitions as seen in the image plane (which could handle target motion within 1 pixel per frame). The possibility of larger motion can be handled by larger patch sizes (for example, a 5-by-5 patch could handle target motion within 2 pixels per frame). These type of target motions correspond to transition probability matrices that only have non-zero probabilities for self-transitions and transitions to states nearby in the image plane (all other transitions have zero probability).

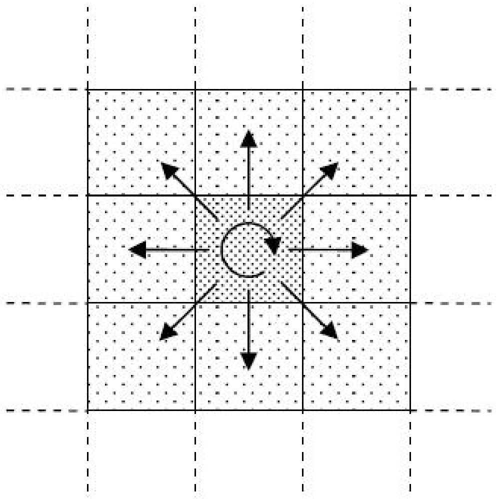


Figure 2: Transition patch of size 3-by-3 corresponding to only 9 non-zero transition probabilities.

An ad hoc approach to assigning the specific probability values within the patch has been used in this paper; better performance might be possible by designing the parameters according to relative entropy rate based strategies (Lai and Ford, 2010).

Furthermore, note that the implemented HMM filter exploits the following probabilistic relationship between target location $\mathbf{x}_k$ and the pre-processed measurements $\mathbf{Y}_k$:

$$\mathbf{B}^m(\mathbf{Y}_k) = \frac{P(\mathbf{Y}_k^m | \mathbf{x}_k = \text{state } m)}{P(\mathbf{Y}_k^m | \mathbf{x}_k \neq \text{state } m)}, \tag{7}$$

for $1 \leq m \leq N$. Note the computational advantage that (7) affords, given that $P(\mathbf{Y}_k^m | \mathbf{x}_k = \text{state } m)$ and $P(\mathbf{Y}_k^m | \mathbf{x}_k \neq \text{state } m)$ can each be determined on a single-pixel basis (rather than requiring the probability of a whole image (Lai et al., 2008)).

To construct the measurement probability matrix $\mathbf{B}_k(\mathbf{Y}_k)$, estimates of the probabilities $P(\mathbf{Y}_k^m | \mathbf{x}_k \neq \text{state } m)$ and $P(\mathbf{Y}_k^m | \mathbf{x}_k = \text{state } m)$ are required. The former describes the prior knowledge about the distribution of pixel values in the absence of a target (i.e. the noise and clutter distribution), while the latter captures the prior knowledge about the distribution of values at pixels containing a target. The required probabilities for $\mathbf{B}_k(\mathbf{Y}_k)$ are trained directly from sample data. The probability $P(\mathbf{Y}_k^m | \mathbf{x}_k \neq \text{state } m)$ is estimated as the average frequency that each pixel value resulted from a non-target location; one way this can be calculated is by sampling pixel values from image sequences without a target. Using a similar procedure, $P(\mathbf{Y}_k^m | \mathbf{x}_k = \text{state } m)$ is estimated as the average frequency that each pixel value measurement resulted from a target location, and this can be calculated based on image sequences containing a target. We acknowledge that in using an empirical approach for estimating the measurement probabilities, the amount of data available will influence the quality of the estimates. In general, estimates of $P\left(\mathbf{Y}_k^m \middle| \mathbf{x}_k \neq \text{state } m\right)$ tend to be easier to obtain, due to the relative abundance of non-target image data compared with those containing targets. From our experiences, probabilities estimated from at least 100 image frames have provided reasonable performance.

Note that in the HMM filter bank, detection events may be triggered by any filter. In particular, if a target is present and this presence is declared in more than one filter, then look to the dominate filter (the one with the highest γ_k) to provide the estimate of target position.

Remark: Strictly speaking, the right hand side of (7) is proportional to $\mathbf{B}^m\left(\mathbf{Y}_k\right)$ (the applicable scaling factor may be absorbed by N_k in the implementation of (3)). Moreover, this relationship only holds under the following assumptions: 1) the statistical properties of pixel values within an image are spatially independent, and 2) individual pixels do not allow the opportunity of perfect detection, in the sense that $P\left(\mathbf{Y}_k^m \middle| \mathbf{x}_k \neq \text{state } m\right) > 0$ whenever $P\left(\mathbf{Y}_k^m \middle| \mathbf{x}_k = \text{state } m\right) > 0$. Admittedly, the presence of extended (multi-pixel) targets or spatially correlated noise would violate the above assumptions, but has been found to only have moderate impact on performance.

3.2.2 Viterbi-Based Filtering

A Viterbi-based temporal filtering approach that is based on a dynamic programming algorithm (Carnie et al., 2006; Gandhi et al., 2006) was also examined. In this approach, pre-processed image frames are integrated over time along possible target trajectories in order to improve the signal-to-noise ratio. The output is an image, with large pixel values indicating likely target locations.

The motion of the target is modelled in terms of discrete velocity cells, where each cell (u, v) encompasses a range of possible target velocities. Assuming constant target velocity (i.e. no transitions between velocity cells) and a velocity cell resolution of one pixel/frame, is can be shown that if the target is at any particular pixel (i, j) at time k, there exists a neighbourhood of four connected pixels within which the target will be located at time $k+1$ (Tonissen and Evans, 1996). This neighbourhood of four connected pixel is denoted by Q^{ij} and collectively referred to as the *forward state transitions.* By symmetry, if the target is at any particular pixel (i, j) at time k, there exists a neighbourhood of four connected pixels within which the target was located at time $k-1$. This neighbourhood of four connected pixels is denoted by $\bar{Q}^{ij}$ and collectively referred to as the *backward state transitions.* Hence, for each velocity cell (u, v) there exists a unique set of forward state transitions Q_{uv}^{ij} and backward state transition $\bar{Q}_{uv}^{ij}$ corresponding to each particular pixel (i, j). This model for target dynamics can be modified to allow for transitions covering more than four pixels at a time by adjusting the resolution of the velocity cells. However, previous analysis (Tonissen and Evans, 1996) showed that better performance is achieved for smaller numbers of possible transitions, with four transition cell possibilities considered to be a reasonable choice for slow, non-manoeuvring targets (Gandhi et al., 2006).

Viterbi-based Detection Strategy The Viterbi-based algorithm performs temporal integration of the input measurements by recursively generating a set of intermediate images $\mathbf{D}$ for each velocity cell (u, v) that is considered. This process can be divided into two stages: initialisation and recursion.

For all (u, v), $1 \leq i \leq N_v$, and $1 \leq j \leq N_h$

1. Initialisation: Let $\mathbf{D}_k^{ij}(u, v)$ denote the ijth pixel of the intermediate image frame at time k for velocity cell (u, v). Then $\mathbf{D}_1^{ij}(u, v) = 0$.

2. Recursion: At time $k > 1$, set

$$\mathbf{D}_k^{ij}(u, v) = \left[(1 - \beta)\,\mathbf{Y}_k^{ij}\right] + \beta . \max_{(i', j') \in \bar{Q}_{uv}^{ij}} \left[\mathbf{D}_{k-1}^{i'j'}(u, v)\right],$$

   where $\mathbf{Y}_k^{ij}$ is the ijth pixel of the pre-processed image at time k, and β represents a memory factor that can vary between zero and one.

At any time k when a detection decision is required, take the maximum output across corresponding pixels of the intermediate image frames belonging to each velocity cell:

$$\mathbf{D}_{max,k}^{ij} = \max_{(u,v)} \left[\mathbf{D}_k^{ij}(u, v)\right], \tag{8}$$

for $1 \leq i \leq N_v$ and $1 \leq j \leq N_h$. This final image $\mathbf{D}_{max,k}$ that consolidates target information from across all velocity cells then serves as the basis for declaring detections.

For the Viterbi-based filtering approach, a test statistic λ_k for declaring the presence of a target at time k is given by

$$\lambda_k = \max_{ij} \left(\mathbf{D}_{max,k}^{ij}\right). \tag{9}$$

When λ_k exceeds a predefined threshold, the Viterbi-based algorithm considers a target to be present and located at state

$$\varsigma_k = \arg\max_{ij} \left(\mathbf{D}^{ij}_{max,k} \right) \tag{10}$$

at time k. The definition of λ_k and ς_k follow from the interpretation of the Viterbi-based algorithm's output as an image, where the pixel values correspond to target signal strength.

Viterbi-Based Filter Implementation The Viterbi-based filter implemented here mirrors those described in the existing literature (Carnie et al., 2006; Gandhi et al., 2006), where four velocity cells are used to detect targets that move with constant velocity in any direction, but are limited to a maximum speed of 1 pixel per frame (this is analogous to the 4 filters in the HMM filter bank). Non-maximal suppression is applied to the output of the filter to reduce an undesirable "dilation" effect where pixels in the neighbourhood of the target also attain significantly large values (Gandhi et al., 2006).

We highlight that the selection of memory factor β has been investigated previously (Carnie et al., 2006; Gandhi et al., 2006). Larger β values give more weighting to past measurements, and can reduce the incidence of false alarms at the expense of a delay in detection. In our implementation of the Viterbi-based filter, we let $\beta = 0.75$, which has been demonstrated to be a reasonable choice for the memory factor (Carnie et al., 2006).

4 Collision-Detection System Evaluation

We begin by evaluating our two candidate target detection approaches in terms of robustness to image jitter and target detection performance. We then discuss our proposed hardware implementation of the detection system, and evaluate the capacity of this system to support real-time execution of the candidate detection algorithms. The candidate detection algorithms are:

- CMO image pre-processing with HMM temporal filtering (patch size 5-by-5); and
- CMO image pre-processing with Viterbi-based temporal filtering.

To facilitate the evaluation of the detection algorithms, we will consider two types of signal-to-noise-ratio (SNR) quantities: 1) a target distinctness SNR (TDSNR), and 2) a false-alarm distinctness SNR (FDSNR).

The TDSNR provides a quantitative measure of the detection capability of the algorithm, and is defined as:

$$\text{TDSNR} = 20\,\log_{10}\left(\frac{P^T}{P^N}\right), \tag{11}$$

where P^T is the average target pixel intensity and P^N is the average non-target pixel intensity at the filtering output. In general, the more conspicuous the target is at the output of the filter, the higher the TDSNR value. On the other hand, FDSNR measures the tendency of the algorithm to produce false-alarms, and is defined as:

$$\text{FDSNR} = 20\,\log_{10}\left(\frac{P^F}{P^N}\right), \tag{12}$$

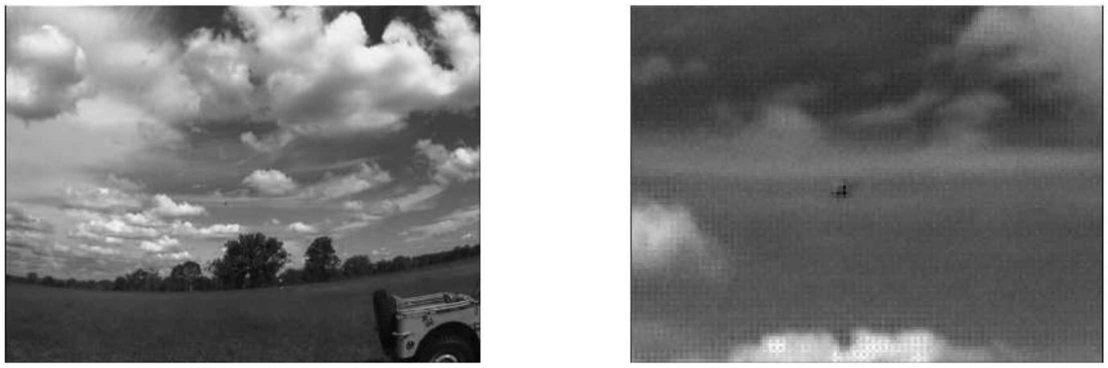
where P^F is the average of the highest non-target pixel intensity and P^N is the average non-target pixel intensity at the filtering output. Strong filter responses away from the true target location will tend to increase the FDSNR value. We highlight that TDSNR and FDSNR are calculated directly from the output of the filter, and are not based on the SNR quality of the input image measurements. For convenience, we will let ΔDSNR denote the difference between the two SNR metrics:

$$\begin{aligned} \Delta\text{DSNR} &= \text{TDSNR} - \text{FDSNR} \\ &= 20\,\log_{10}\left(\frac{P^T}{P^F}\right). \end{aligned} \tag{13}$$

4.1 Impact of Image Jitter

In this paper, we consider image jitter as the apparent motion of the image background between two consecutive frames of an image sequence. This apparent motion is caused by displacement of the camera relative to the objects in the scene, and gives the illusion that objects in an image are moving when in fact they are stationary in the environment. Image jitter is inherently present in all image data recorded on moving platforms. Passive motion-dampening devices or actively stabilised camera mounts may serve to reduce the effects of image jitter; however, jitter cannot be completely eliminated as no platform can be perfectly stable. In many cases, the effects of jitter can be quite severe, particularly for small UAV platforms that are more susceptible to unpredictable disturbances caused by wind gusts and air turbulence.

In addition to the physical mechanisms that can be employed to reduce jitter effects whilst images are being recorded, there are ego-motion compensation techniques that can be used after the image is captured to produce a jitter-corrected version of the raw image. Ego-motion compensation techniques typically rely on estimating the amount of camera displacement either indirectly via gyroscopes, accelerometers, and other



(a) (b)

Figure 3: (a) Sample image from jitter characterisation test data set; (b) Region of interest containing target.

inertial sensors (Jorge and Jorge, 2004), or directly through tracking salient features (Jung and Sukhatme, 2005).

In this study, we are concerned with the inherent robustness of the detection algorithms to jitter affected data that have not undergone ego-motion compensation. Characterising this inherent robustness of the detection algorithms is important for two main reasons. Firstly, particularly in an airborne environment, it would not be uncommon to encounter 'blue-sky' conditions where the background is more or less uniform, rendering feature-based compensation techniques much less effective, if not useless. Moreover, no compensation technique is perfect, and hence it would be highly desirable for any practical detection algorithm to at least possess some level of resilience to image jitter in the absence of ego-motion compensation. We will characterise the jitter performance of the two candidate detection algorithms by applying them to some test image sequences containing varying degrees of jitter. The filtering output of the candidate detection algorithms will then be evaluated in terms of the two types of SNR quantities, TDSNR and FDSNR, defined earlier.

4.1.1 Test Data

Figure 3a provides a sample of the type of data that was used to characterise the detection algorithms. The typical target size and background contrast can be seen in a close-up region of interest depicted in Fig. 3b. Another close-up sample is provided in Fig. 4a. Here, the target occupies around 6-10 pixels, is quite distinct, and may be considered fairly easy to detect. The relatively high target signal-to-noise ratio does not diminish the significance and practical relevance of the performance characterisation because investigating the impact of image jitter on algorithm performance is the main interest of this study, as opposed to investigating dim target detection performance (this is considered separately in the next subsection).

In the data sets, a range of jitter characteristics has been artificially introduced, ranging from low to moderate to extreme. A low amount of jitter is defined as involving apparent inter-frame background motion of between 0 and 1 pixels per frame. A moderate amount of jitter is defined as involving apparent inter-frame motion of between 1 and 3 pixels per frame. Finally, an extreme amount of jitter is defined as involving apparent inter-frame motion is greater than 4 pixels per frame.

4.1.2 Jitter Test Results

Tables 1 and 2 illustrate the typical performance of the HMM and Viterbi-based temporal filtering approaches under various jitter scenarios. In the presence of a low level of jitter both detection approaches were able to successfully track the target (successful tracking implies that the target is able to be located within two pixels of its true position). Figure 4 (c) and (d) illustrate the typical output of the HMM and Viterbi-based filter, respectively, under low jitter conditions. We highlight that in this case the target in the HMM filter output is considerably more distinct than in the Viterib-based filter output, and this is reflected in the larger ΔDSNR value for the HMM filter. However, under moderate jitter conditions, there were periods when the HMM filter failed to track the target successfully. This suggests that the HMM filter is more sensitive to the effects of jitter. Part of this sensitivity may be attributed to the jitter dynamics not being explicitly modelled in the HMM filter. Moreover, we highlight that the HMM algorithm exploits extra target dynamic behaviour information (i.e. modelling assumptions) that is not used in the Viterbi-based filter. This extra information improves baseline performance of the HMM filter, but also make the filter more sensitive to how valid these assumptions are, some of which are violated by image jitter. Hence, the impact of jitter is more pronounced on the 'optimised' performance of the HMM filter compared to the more 'ad hoc' Viterbi-based filter. Finally, in the presence of extreme jitter, both filtering approaches demonstrated poor tracking capabilities and exhibited correspondingly low ΔDSNR values.

Overall, the ΔDSNR values seem to provide a reasonable indication of the detection algorithms' jitter tracking performance. Using as a baseline the value from the low jitter scenario, which we denote by ΔDSNR_0, the results suggest that as a rough rule-of-thumb successful tracking can be accomplished under a particular jitter scenario x when:

$$\Delta\text{DSNR}_x > 0.5\ \Delta\text{DSNR}_0, \tag{14}$$

Table 1: Jitter performance of HMM temporal filtering

		SNR Value (dB)		
Jitter Level	**Tracking Outcome**	**TDSNR**	**FDSNR**	**ΔDSNR**
Low	Success	72.84	40.71	32.13
Moderate	Success Period	71.12	40.93	30.19
	Fail Period	-58.61	38.10	-96.71
Extreme	Fail	-57.60	35.61	-93.21

Table 2: Jitter performance of Viterbi-based temporal filtering

		SNR Value (dB)		
Jitter Level	**Tracking Outcome**	**TDSNR**	**FDSNR**	**ΔDSNR**
Low	Success	36.20	33.28	2.92
Moderate	Success	35.33	32.30	3.03
Extreme	Fail	30.00	29.60	0.40

where ΔDSNR_x is the ΔDSNR value corresponding to jitter scenario x.

Although neither the HMM nor Viterbi-based approaches handle extreme jitter, we note that if the target signal-to-noise ratio is high enough it may be possible to detect potential collision-course targets from the output of the morphological image pre-processing stage alone.

4.2 Target Detection Performance

In a potential collision scenario it is desirable to identify the other aircraft as early as possible to ensure enough time is available to plan and carry out appropriate actions. For instance, evasive manoeuvres may take a period of time to execute as the aircraft cannot be expected to respond instantaneously to control inputs. In general, increasing the range at which the threat can be detected will lead to earlier detection and hence more time to react. Thus, detection range may be considered a suitable statistic for characterising the detection performance of the candidate filtering algorithms.

In this study, we will firstly quantify detection performance in terms of the maximum range at which consistent detection occurs. We define consistent detection to have occurred when a simple threshold on the output of the filter allows the target to be detected in 10 successive frames with zero false-alarms. A consistent detection, however, does not preclude the existence of strong filter responses at non-target locations which are undesirable as they may be mistaken for the true target. Hence, ΔDSNR values (averaged across the

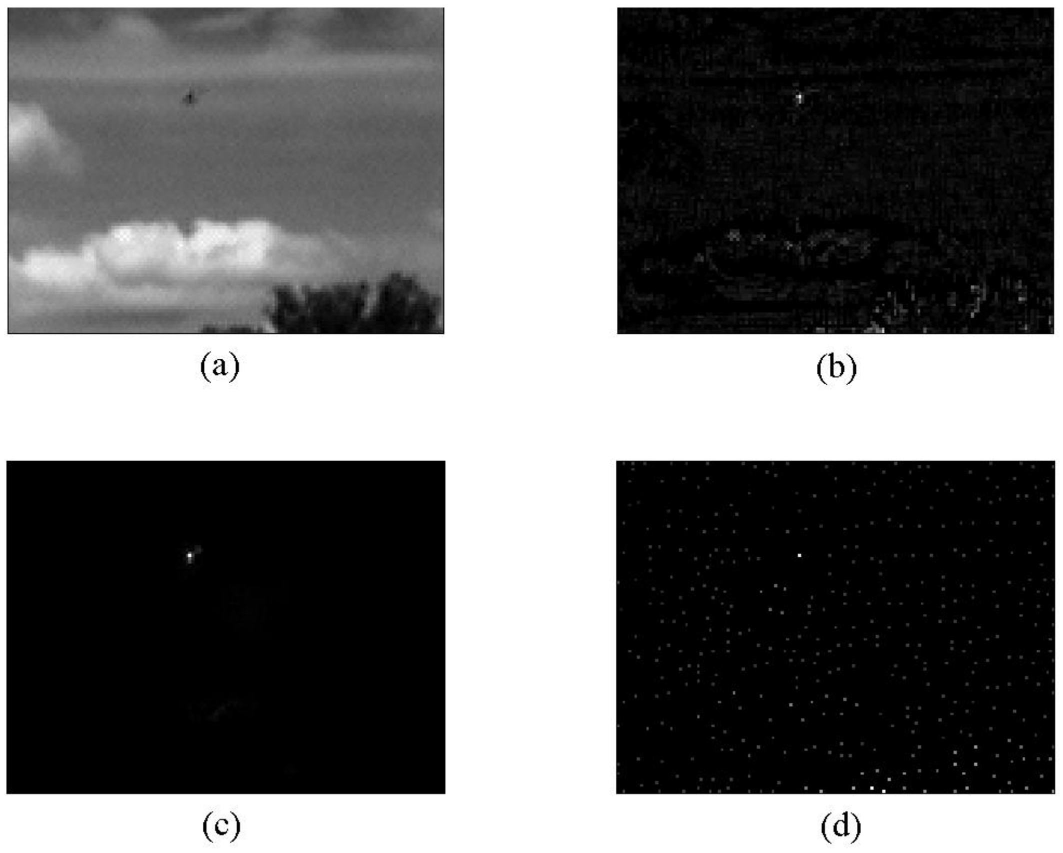


Figure 4: Sample of (a) test data frame; (b) CMO image pre-processing output; (c) HMM filter bank output (dominant member filter); and (d) Viterbi-based filter output, after processing 41 frames.

10 consistent detection frames) will be presented together with the detection range statistics to provide an indication of the detection confidence.

We highlight that special experiments involving two fixed-wing UAVs flying near collision-course trajectories were conducted to collect suitable test data for offline post-processing by the detection algorithms. We elaborate on the details of these data collection experiments in the following paragraphs.

4.2.1 Test Data

Image sequences depicting aircraft on collision-course naturally serve as ideal test cases for evaluating collision detection algorithms; however, due to the inherent risk of flying aircraft on converging paths, this type of image data is extremely scarce. This motivated us to conduct our own special flight experiments to collect suitable test data.

Two fixed-wing UAVs were deployed to collect suitable test data: 1) a Flamingo UAV (Silvertone UAV, www.silvertone.com) measuring 2.9m from nose to tail with a wingspan of 4m; and 2) a Boomerang 60 model airplane (Phoenix Models) measuring 1.5m from nose to tail with a wingspan of 2.1m. The Flamingo was powered by a 26cc 2-stroke Zenoah engine driving a 16 by 6 inch propeller. The powerplant for the Boomerang was an O.S. 90 FX engine driving a 15 by 8 inch propeller.

The avionics payload of the Flamingo included a MicroPilot® MP2128g flight controller, Microhard radio modems, an Atlantic Inertial SI IMU04 inertial measurement unit (IMU), a separate NovAtel OEMV-1 GPS device (in addition to the standard GPS capability provided by the MicroPilot® board), and an extensively customised PC104 mission flight computer. In contrast, the Boomerang only possessed a basic setup that featured a MicroPilot® MP2028g flight controller and Microhard radio modems.

In our experiments, the Flamingo served as the image data acquisition platform and was further equipped with a fixed non-stabilised Basler Scout Series scA1300-32fm/fc camera fitted with a Computar H0514-MP lens that offered 51.4 degree horizontal, 39.5 degree vertical, and 62.3 degree diagonal angles of view. The camera could be turned on and off remotely from the ground control station, and was configured to record 1024 by 768 pixel resolution image data at a rate of 15Hz with a constant shutter speed to maintain consistent lighting in the image frames. A solid-state hard-disk was used to store the recorded image data, as opposed to conventional mechanical disk drives which may be susceptible to vibrations during flight.

Figure 5 shows our UAV platforms (the Boomerang is in front of the Flamingo) configured for data collection. We highlight the streamlined pod just forward of the Flamingo's wing that houses the camera along with the IMU. Our goal was to use the Flamingo, denoted as the camera aircraft, to record images of the Boomerang, which acted as the target aircraft, whilst both aircraft were flown autonomously along preset waypoints that defined near collision-course paths (adequate height and time separation was maintained at all times for safety reasons). In this way we could then capture realistic examples of what would be observed by an aircraft in the moments leading up to a collision. In particular, we sought image data that depicted the target aircraft gradually emerging from a state of being imperceptible to the naked eye to being clearly distinguishable (as opposed to data where the target aircraft suddenly enters into the field of view). We considered this type of data to be ideal for evaluating the detection range capability of the candidate filtering algorithms. The captured image data was timestamped during flight so that it could be later correlated with inertial and GPS-based position logs from both aircraft in order to estimate the detection range.

We conducted numerous flights at an altitude of approximately 450m above mean sea level over agricultural

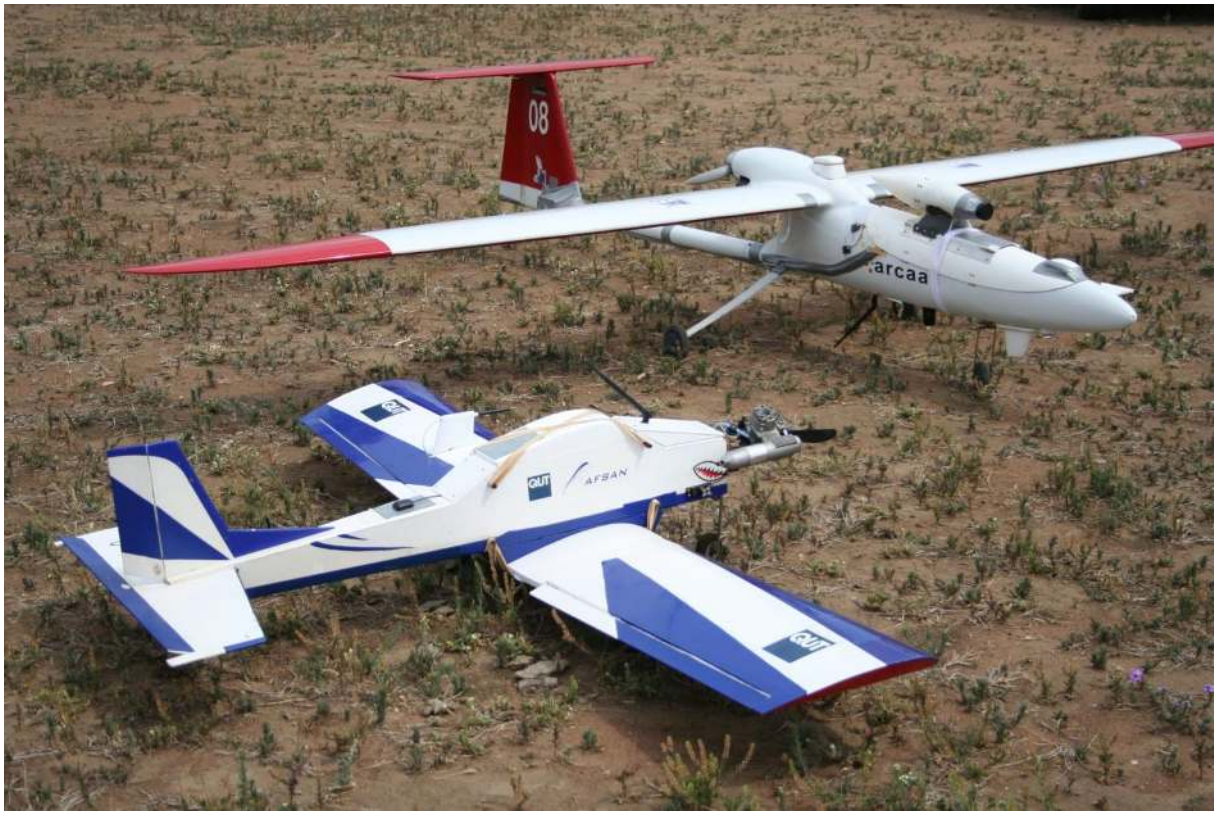


Figure 5: UAV platforms used to collected test image data. The Boomerang is in front of the Flamingo

and livestock fields in the town of Kingaroy, about 210km north-west of the city of Brisbane, Australia. Our testing site covered a 4 by 4 kilometer area centered on an unsealed rural airstrip. Apart from obtaining permission from the landowner, no additional approvals or waivers were required as our flight operations fell under existing civil aviation safety authority regulations ((Civil Aviation Safety Regulations 1998, 2009); specifically, Part 101, subpart 101.F). Figure 6 illustrates our general approach to recreating a head-on collision scenario (not to scale). Typically after take-off, both aircraft were placed in holding patterns until they were ready to be released to follow predefined waypoints that brought the aircraft onto converging paths (a similar strategy was used in the simulation of other collision geometries). Apart from take-offs and landings, the aircraft were flown autonomously throughout the collection of data. Figure 7 illustrates the aircraft trajectories in an actual head-on engagement scenario that was enacted.

One of the key challenges of this approach involved timing the release of the aircraft from their respective holding patterns in a way that maximised the time spent by the target aircraft in the field of view of the

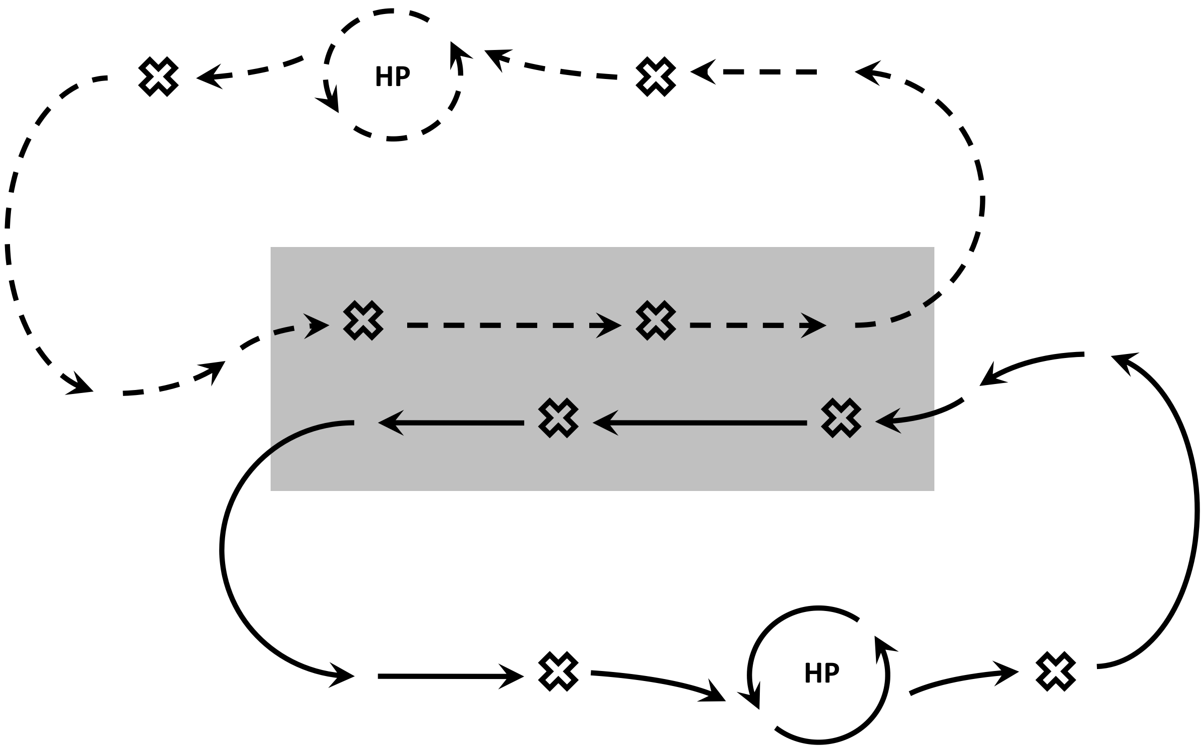
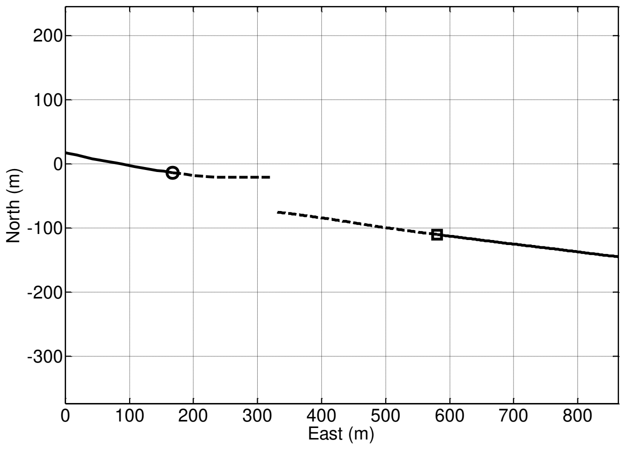
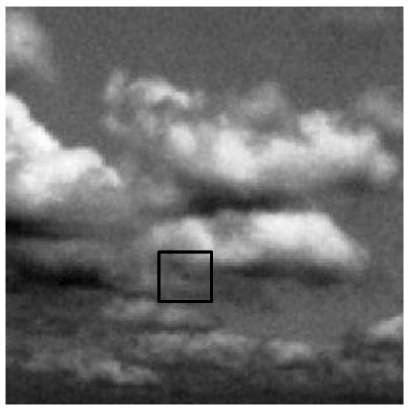


Figure 6: Typical flight plan for enacting a head-on collision scenario (not to scale). After exiting from their respective holding patterns (HP), the paths of the camera aircraft (solid line) and the target aircraft (dashed line) followed predefined waypoints (×) before converging at the designated area for capturing data (shaded region). The aircraft are approximately 2km apart at the start of their passing runs.

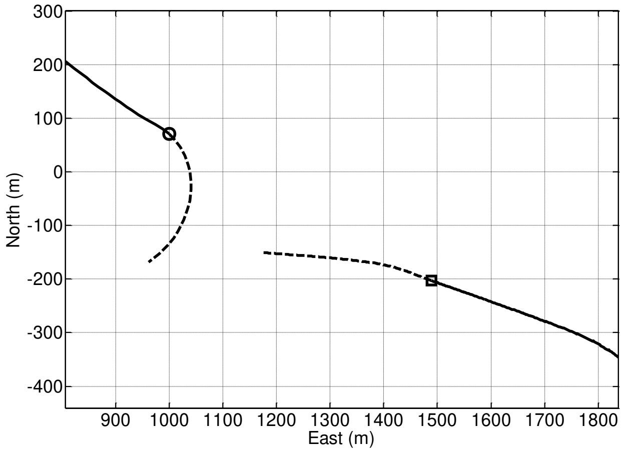
camera aircraft. In the case of the head-on collision scenario, this meant having both aircraft simultaneously in the shaded region of Figure 6 for as long as possible. However, it was difficult to perform this synchronisation, and some of the data collected was not considered suitable for analysis. In spite of this, we were able to locate useful image sequences from three separate engagement scenarios featuring two different types of collision geometries. In two out of the three scenarios, the camera aircraft and the target aircraft are converging head-on; and in one scenario the aircraft are converging at right-angles to each other. Figures 7 to 9 illustrate overhead views of the collision geometries flown in each of the three engagement scenarios; also shown is a zoomed in sample of the typical vision obtained of the target aircraft for each scenario. The aircraft paths are plotted in a local East, North, Up (ENU) Cartesian coordinate system relative to a runway used for take-offs and landings (not shown).

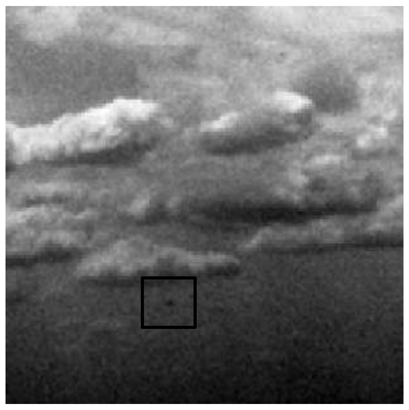




(a) (b)

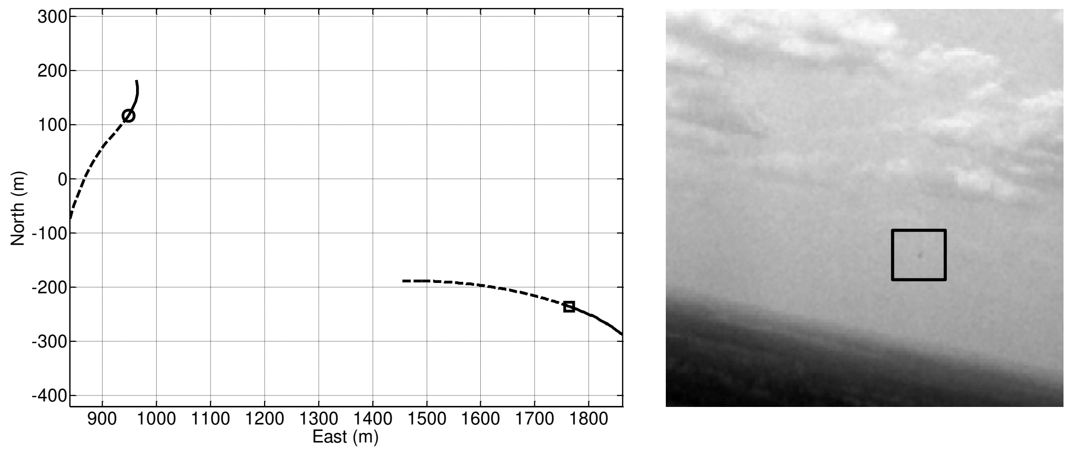
Figure 7: Engagement scenario 1. (a) Camera aircraft (□) and target aircraft (◯) positions leading up to point of detection (solid line) and shortly after detection (dashed line); (b) sample image frame containing target aircraft.





(a) (b)

Figure 8: Engagement scenario 2. (a) Camera aircraft (□) and target aircraft (◯) positions leading up to point of detection (solid line) and shortly after detection (dashed line); (b) sample image frame containing target aircraft.



(a) (b)

Figure 9: Engagement scenario 3. (a) Camera aircraft (□) and target aircraft (◯) positions leading up to point of detection (solid line) and shortly after detection (dashed line); (b) sample image frame containing target aircraft.

Analysis of the data in all three scenarios revealed the presence of extreme jitter magnitudes exceeding 4 pixels per frame. This level of jitter was not completely unexpected; the usual jitter anticipated from the use of a non-stabilised camera was exacerbated by the relatively lightweight UAV platforms (compared with manned aircraft), which are more liable to external disturbances such as wind gusts and air turbulence.

Given the presence of extreme jitter and in light of the results from the earlier jitter performance characterisation, we considered it appropriate to undertake some level of jitter compensation of the image data before applying the detection algorithms. As this paper is not primarily concerned with the subject of camera-motion compensation, we implemented a basic image correction procedure involving a combination optical flow (Lucas and Kanade, 1981) and template matching (Brunelli, 2009) techniques. Although this image correction procedure only accounted for translational motion, it was sufficient to reduce the overall effect of jitter to a moderate level. Image sequences with this moderate level of 'residual' jitter were then processed by the detection algorithms.

4.2.2 Detection Results

Table 3 illustrates the detection performance of the HMM and Viterbi-based filtering approaches in each of the three engagement scenarios. The detection ranges are calculated based on the camera and target aircraft GPS position information corresponding to the image frame that detection is declared (given that no unusual satellite geometries were encountered during the collection of the data, we estimate the 1-σ 3D position error of the GPS positions to be approximately 17 meters, based on standard values for HDOP, VDOP, and UERE (Parkinson and Spilker, 1996)).

Table 3: Target detection performance of HMM and Viterbi-based temporal filtering

		HMM Filter		**Viterbi-based Filter**	
Scenario	**Geometry**	**Detection Range (m)**	**ΔDSNR**	**Detection Range (m)**	**ΔDSNR**
1	Head-On	412	38.04	425	1.61
2	Head-On	562	31.94	564	1.23
3	Right-Angle	881	19.55	893	2.83

In all three scenarios detection occurred a few frames earlier for the Viterbi-based algorithm when compared with the HMM algorithm. This is reflected in the slightly greater detection ranges for the Viterbi-based approach. The approximate positions of the aircraft at the time of detection are marked on the flight paths given in Figures 7(a), 8(a), and 9(a) (due to the small differences in detection ranges, the aircraft positions corresponding to when detection is declared are not separately marked for the two algorithms). Overall, we obtained detection ranges that are generally consistent with the results reported in an earlier study (Carnie et al., 2006), which applied a detection algorithm similar to the Viterbi-based approach discussed in this paper. The earlier study attempted to detect a manned Cessna aircraft using image data captured by a fixed ground-based camera, and reported detection ranges out to about 6km. This range is roughly in proportion to our results in Table 3, considering that a Cessna aircraft is around 6-7 times the size of our Boomerang target aircraft. While we do not claim that a linear relationship exists between target size and detection distance, this comparison of detection range results reinforces the intuitive notion that larger targets should be able to be detected at greater ranges. We plan to conduct further experiments using full-scale Cessna aircraft in order to better understand the relationship between target size and detection range.

Furthermore, Table 3 shows the ΔDSNR values corresponding to the two target detection approaches. The ΔDSNR values for the HMM filter are consistently higher than the values for the Viterbi-based filter. This suggests that the HMM filter may be more effective at suppressing non-target responses (whilst maintaining a strong response at the true target location), and as a consequence, we may expect in general a lower incidence of false-alarms for the HMM filter than the Viterbi-based filter. We may also interpret the ΔDSNR values in terms of the empirical SNR criterion (14) established earlier for successful tracking. For the HMM filtering approach, the SNR criterion is easily satisfied, with the ΔDSNR values in all three scenarios clearly greater than $16.065 = 0.5\times32.13$. On the other hand, the results for the Viterbi-based filtering approach are mixed. Scenario 1 is a borderline case, and in Scenario 2 the ΔDSNR value is just below the threshold of $1.46 = 0.5\times2.92$. This suggests that the consistent detections established by the Viterbi-based filter in scenarios 1 and 2 are perhaps less reliable, in the sense that the filter may possibly be on the verge of losing track of the target.

We conjecture that residual jitter effects present in the image sequences after compensation (after all, no compensation technique is perfect) may be partly responsible for the difference in detection range performance. This is because residual jitter would tend to affect the HMM filter slightly more so than the Viterbi-based filter in view of our results from the earlier jitter performance analysis. We also highlight the significantly larger detection ranges reported for the right-angle collision geometry compared with the head-on cases. One possible explanation for this is that the target aircraft tends to appear as a larger object from the right-angle perspective (cross-section area approximately 0.25m^2) than from head-on (cross-section area approximately 0.1m^2).

We have used detection range as a metric for comparing the performance of the two candidate target detection algorithms. However, from an operational point of view it is the time-to-impact from the point of detection that is perhaps more informative, rather than the absolute distance to the conflicting aircraft. This is because, analogous to the time needed by human pilots to respond to a collision situation, an automated system also requires an interval of time to plan and carry out appropriate actions. Depending on the closing speed of the aircraft involved, a particular detection range may or may not provide sufficient time to accommodate both processing needs and aircraft response lag.

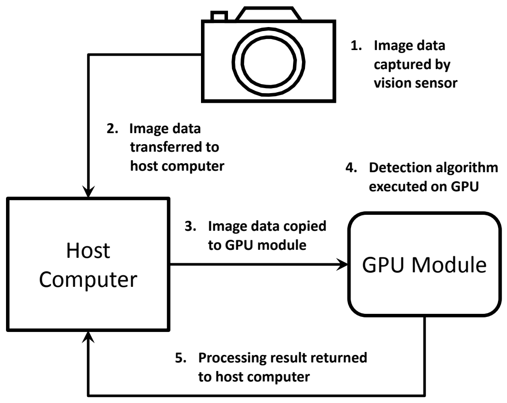
The Federal Aviation Administration has issued an advisory circular that provides guidance on the time required for a pilot to recognise an approaching aircraft and execute an evasive manoeuvre (FAA Advisory Circular: Pilots' role in collision avoidance, 1983). According to the advisory circular, as a general rule, a conflicting aircraft must be detected at least 12.5 seconds prior to the time of impact. This motivates us to undertake some further analysis to gauge approximately how much time ahead of impact the system was able to detect the target. We chose to conduct our analysis on the worst-case scenarios i.e. the head-on geometries of scenarios 1 and 2. In our study we projected, from the point of detection, hypothetical trajectories for each aircraft corresponding to the shortest possible collision-course path (this path is defined by a straight line drawn between the camera and target aircraft positions at the point of detection). Each aircraft's average speed in the last 5 seconds leading up to the point of detection was used as the speed travelled along the projected trajectories. Under these circumstances, the time-to-impact at the point of detection in Scenario 1 was estimated to be around 8 seconds (based on a combined closing speed of 51m/s), and for Scenario 2 the time was around 10 seconds (based on a combined closing speed of 53m/s). It is clear that these times are below the recommended 12.5 seconds; however, our results must be considered in the appropriate context.

The advisory circular information pertains to human pilots, with the 12.5 seconds broken down into the time taken to perform various sub-tasks. Over half of the time is allocated to the tasks of recognising the existence of the collision situation, making a decision to perform an avoidance manoeuvre, and manipulating the aircraft controls to execute the manoeuvre. An automated system must also carry out similar tasks, but there is potential for the tasks to be completed much more quickly given the rapidly improving processing capacities of computing hardware and the near instantaneous actuation of flight controls. Hence, we expect an equivalent safe detection time ahead of impact for automated systems to be less than 12.5 seconds.

4.3 Proposed System Hardware Configuration and Performance

Image processing is traditionally a very computationally intensive task. Hence, we considered it a challenge to achieve real-time performance for our proposed vision-based collision detection system. The increasing complexity of image processing algorithms is pushing at the processing limits of modern CPU-based computing architectures, even with the use of optimized computer vision libraries. Highly specialised hardware such as field programmable gate arrays (FPGAs) and dedicated digital signal processors have been used to overcome the limitations of the CPU; however, such hardware are typically not readily accessible. In this paper, we investigate the potential of widely available commercial-off-the-shelf (COTS) Graphics Processing Units (GPUs) for running our candidate target detection algorithms. The parallel processing (single-instruction, multiple-data (SIMD)) capability of GPUs is ideally suited to computer vision tasks, which often require the same calculations to be performed repeatedly on each individual pixel (or group of pixels) over an entire image.

Figure 10a illustrates the high-level hardware architecture of our collision detection system. Particular attention was paid to the interaction between hardware components to enhance processing speed. The key design strategies that we followed included: 1) the GPU module should carry out all the computationally intensive image processing tasks; 2) memory transfers between the host computer and the GPU module should be minimized; and 3) the host computer should be kept free to perform other non-image related tasks. By adhering to these strategies, we produced a system whereby each image frame captured by the vision sensor was directly copied from the host computer to the GPU module for processing. This allowed the entire detection algorithm to be executed on the GPU module and ensured that the GPU was utilized


1. Image data captured by vision sensor
2. Image data transferred to host computer
3. Image data copied to GPU module
4. Detection algorithm executed on GPU
5. Processing result returned to host computer
Host Computer
GPU Module

(a)

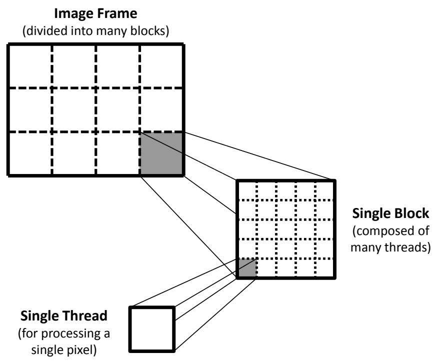


Image Frame
(divided into many blocks)
Single Block
(composed of many threads)
Single Thread
(for processing a single pixel)

(b)

Figure 10: (a) High-level system architecture; (b) GPU image processing architecture.

efficiently (in addition, offloading all the image processing to the GPU module eliminated time consuming memory transfers that would otherwise be required if the processing was shared between the host and GPU). Once processing is completed on the GPU, only one additional memory transfer was needed to return the processing result (for example, a detection probability) to the host computer for further analysis. This entire process is repeated at the capture of the next image frame.

Recently, the parallel processing capabilities of GPUs have been exploited by other authors for tracking applications (Ohmer and Redding, 2008). Figure 10b illustrates the general way in which we exploited the GPU for processing images. At a high level, this involved dividing the image frame into blocks, with each block containing a number of threads. We aimed to allocate one thread to each pixel of the image frame so that operations could be carried out simultaneously on all pixels, greatly enhancing the processing speed.

Overall, as a minimum standard, the system was required to handle real-time processing of image data (1024 by 768 pixel image frames, encoded at 8 bits per pixel) captured at a rate of 30Hz. As a first step to characterizing the performance of a GPU based system, we bench-tested an implementation of the system architecture in Figure 10a using the following desktop computer components:

- Host Computer: Intel Pentium IV 3.2GHz CPU (with Hyper-threading); 1Gb SDRAM at 666MHz; Linux Ubuntu 32 Bit operating system

- GPU Module: NVIDIA GeForce GTX 280, 1Gb GDDR3 RAM (GV-N28-1GH-B)

The NVIDIA GeForce GTX 280 has 30 multiprocessors and a compute capability of 1.3 (NVIDIA CUDA Compute Unified Device Architecture: Programming Guide Version 2.0, 2008). Programming access to the GPU module was facilitated by NVIDIA's Compute Unified Device Architecture (CUDA) framework (NVIDIA CUDA Compute Unified Device Architecture: Programming Guide Version 2.0, 2008). Using this configuration we were able to achieve an average frame rate of approximately 150Hz (or an average of 7.47ms to process each frame) for the Viterbi-based detection algorithm, and a frame rate of approximately 130Hz (or an average of 6.51ms to process each frame) for the HMM algorithm (these processing times include file input/output operations). It is clear that the processing rates for both algorithms are well in excess of our earlier 30Hz requirement.

Our bench-test with desktop components has demonstrated the enormous potential of GPUs for realising real-time performance in vision-based systems. Motivated by these encouraging results, we have implemented a flight-ready system architecture using the following components:

- Host Computer: Intel Core 2 Duo Merom 800MHz CPU; 2Gb SDRAM at 800MHz; Linux Debian 32 Bit operating system
- GPU Module: NVIDIA GeForce 9600 GT, 512Mb GDDR3 RAM (GV-N96TGR-512I)

Figure 11 illustrates the physical hardware layout of the system, which is currently being integrated on a custom-modified Cessna 172 aircraft, as shown in Figure 12.

The NVIDIA GeForce 9600 GT was selected as it offered a good balance between processing performance, power consumption, and volume requirements. Specifically, it was the highest performing (in terms of the number of multiprocessors) GPU device that did not require an external power connection to supplement supply from the PCI bus. Figure 11 illustrates the GeForce 9600 GT (right) alongside the host computer. The GPU has 8 multiprocessors, a compute capability of 1.1, and consumes only 59 Watts of power (low-power version).

From experience, our implementation of the detection algorithms on the GPU scales roughly linearly with the number of multiprocessors. Hence, we anticipate processing rates on our flight hardware configuration to be approximately four times less than those reported for the bench-tests. This translates to a frame rate of around 37Hz for the Viterbi-based algorithm and a frame rate of around 32Hz for the HMM approach, which both still satisfy our 30Hz requirement. We highlight that there is still scope for further improvement in processing speeds, as we have yet to exploit advanced GPU code optimisation techniques (such as pipelining and dynamic memory allocation methods).

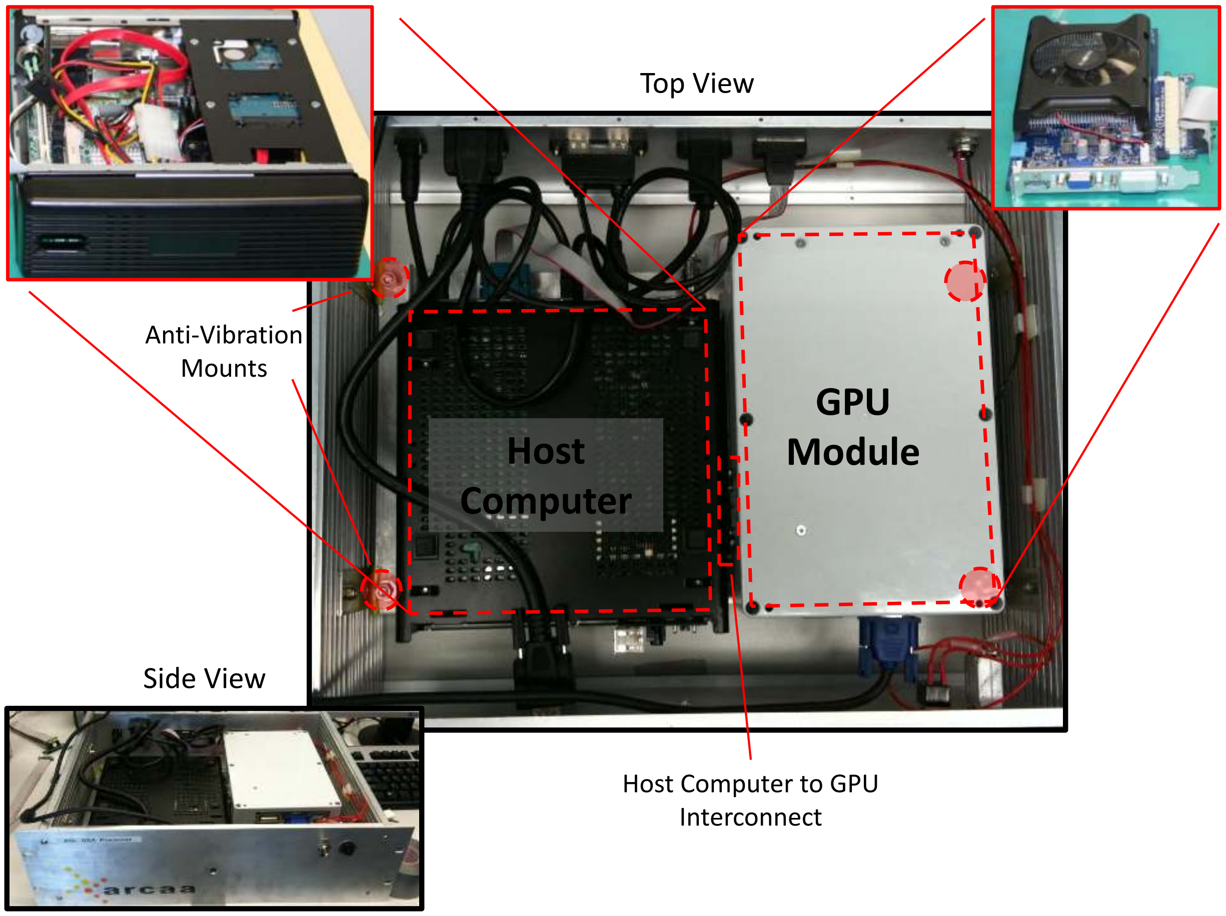


Figure 11: Flight-ready hardware configuration.

Figure 12: Cessna 172 aircraft.

5 Lessons Learnt and Future Work

Our long term goal extends beyond the ability to simply detect potential collision threats. We aim to establish a well-refined target detection capability, before progressively evolving this into a fully autonomous closed-loop collision avoidance system. This closed-loop system will include decision-making and control modules so that the aircraft may automatically and actively respond to any detected threats. In the near term, our plans are to conduct further testing and refinement of the detection system, as discussed in the following sections.

5.1 Hardware Implementation and Evaluation

Implementation of the flight hardware configuration is complete. The next step is to verify the anticipated processing speed of the hardware under realistic operating conditions via several flight trials using full-scale manned Cessna aircraft and UAV platforms. Should the actual speed fall below our expectations or our minimum requirements, we may consider employing advanced code optimisation techniques to improve processing speed. Alternatively, we may choose to upgrade to a more powerful GPU device at the expense of increased energy consumption.

5.2 Target Detection

Based on the detection range and image jitter performance results, neither the HMM nor the Viterbi-based algorithm stood out as the decidedly better detection approach. Admittedly, only three data sets were used in the analysis of detection range; hence, we intend to collect more data so that a more comprehensive performance analysis can be conducted. Our next planned flight experiments will be carried out using manned Cessna aircraft, which we anticipate may alleviate the synchronisation issues we encountered with UAV platforms (see next section) and allow higher quality data to be collected in larger quantities. A custom designed bracket has been manufactured that allows a camera to be mounted near the intersection of the wing and the supporting strut, as shown in Figure 13.

Moreover, our image jitter analysis results combined with the inter-frame motion observed in the collected image data suggests that jitter compensation is necessary for our target detection algorithms to perform effectively on a UAV platform using an unstablised camera. In addition to the simple image-based compensation approach used in this paper, we are testing more advanced image correction techniques based on aircraft attitude and inertial measurements. If our compensation techniques prove to be inadequate, we may

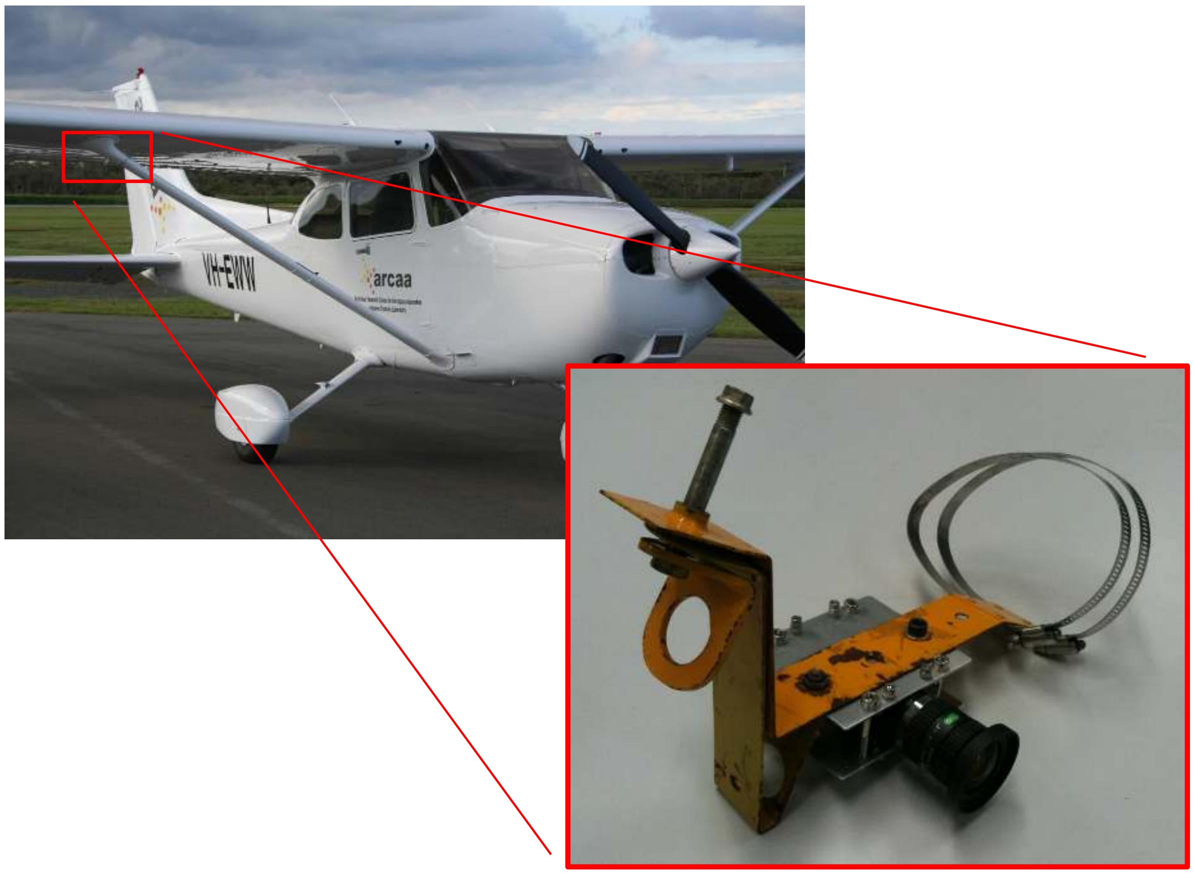


Figure 13: Camera mounting bracket for Cessna aircraft.

consider the use of a stabilised camera and/or increasing the data capture rate to further mitigate jitter effects in the captured data.

5.3 Data Collection

We encountered many challenges in the collection of suitable test data, particularly in our flight trials that simulated collision-course scenarios. These experiments required the aircraft flights to be synchronised in time so that the target aircraft is kept in the field of view of the camera aircraft for as long as possible. We found it particularly difficult to achieve a precise sychronisation of the flights. On reflection, we have identified several factors that have contributed to this: 1) The inability to colocate the ground control stations of each aircraft due to the range limits of aircraft communication links (our suboptimal approach to overcoming this was to station personnel at separate ground control stations and to coordinate the release of the aircraft from their holding patterns via hand-held radios); 2) The use of fixed-wing platforms which are more difficult to position accurately as they must be kept in constant motion (aircraft positions were monitored by sight and through the MicroPilot® Horizon software); 3) The inability to make adjustments to flight paths once the aircraft are released from their holding patterns. This is due to the aircraft being operated in an autonomous mode.

In the short term, we will switch to manned Cessna platforms in an attempt to improve the synchronisation of flights in future data collection experiments. In the long term, we are considering upgrades to antennas and cabling to improve signal gain so that the UAV ground control stations can be colocated. This will allow the flights to be better coordinated. It may also be possible to improve flight synchronisation by introducing more flexibility into the flight plan; for example, by adding a small segment of manual control just after the aircraft's release from the holding pattern so that adjustments and corrections can be made to improve timing if necessary (remotely piloting the aircraft for the entire experiment is not considered feasible as it is too difficult to maintain a consistent altitude for an extended period of time). An alternative is to develop new software that will allow both UAV platforms to be controlled by a single control station. This will open up the opportunity to have the aircraft flights automatically synchronised without human intervention. Overall, the field and operational experiences gained from our flight trials will be invaluable in fine-tuning future data collection experiments.

6 Conclusion

This paper proposes a vision-based collision detection system for aerial robotics. It considers two target detection approaches and a GPU-based hardware implementation. Special flight experiments were conducted to collect suitable test data for evaluating the detection capabilities of a HMM-based detection algorithm and a Viterbi-based algorithm. The detection algorithms were able to detect targets at distances ranging from 400m to around 900m (depending on the collision geometry). Based on aircraft closing speeds leading up to the point of detection and a hypothetical extrapolation of aircraft trajectories, these detection distances would provide an advanced warning about 8 to 10 seconds prior to impact. These results are not too far from the 12.5 second response time recommended for human pilots, and are an encouraging sign for vision-based collision detection approaches. We exploit the parallel processing capabilities of GPUs to enable our detection algorithms to execute in real time, and anticipate our flight-ready hardware configuration to achieve frame rates of approximately 30Hz, based on earlier bench-testing results.

Acknowledgments

This research is part of the Smart Skies Project and is supported, in part, by the Queensland State Government Smart State Funding Scheme. The authors also gratefully acknowledge the contribution of Scott McNamara and Angus Thomson for support with the GPU experimentations.

References

Arnold, J., Shaw, S. W., and Pasternack, H. (1993). Efficient target tracking using dynamic programming. *IEEE Transactions on Aerospace and Electronic Systems*, 29:44–56.

Barnett, J. T., Billard, B. D., and Lee, C. (1993). Nonlinear morphological processors for point-target detection versus an adaptive linear spatial filter: A performance comparison. In *Proceedings of Signal and Data Processing of Small Targets*, Orlando, FL.

Barniv, Y. (1985). Dynamic programming solution for detecting dim moving targets. *IEEE Transactions on Aerospace and Electronic Systems*, AES-21:144–156.

Barniv, Y. and Kella, O. (1987). Dynamic programming solution for detecting dim moving targets part II: Analysis. *IEEE Transactions on Aerospace and Electronic Systems*, AES-23:776–788.

Billingsley, P. (1995). *Probability and Measure*. New York: Wiley, 3rd edition.

Braga-Neto, U., Choudhary, M., and Goutsias, J. (2004). Automatic target detection and tracking in forward-looking infrared image sequences using morphological connected operators. *Journal of Electronic Imaging*, 13:802–813.

Brunelli, R. (2009). *Template matching techniques in computer vision: Theory and practice*. Chichester: Wiley.

Bruno, M. G. S. (2004). Bayesian methods for multiaspect target tracking in image sequences. *IEEE Transactions on Signal Processing*, 52:1848–1861.

Bruno, M. G. S. and Moura, J. M. F. (1999). The optimal 2D multiframe detector/tracker. *International Journal of Electronics and Communications (AEU)*, 53:1–17.

Bruno, M. G. S. and Moura, J. M. F. (2001). Multiframe detector/tracker: optimal performance. *IEEE Transactions on Aerospace and Electronic Systems*, 37:925–945.

Bryner, M. (2006). Developing sense & avoid for all classes of UAS. Technical report, Defense Research Associates.

Carnie, R., Walker, R., and Corke, P. (2006). Image processing algorithms for UAV "sense and avoid". In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Orlando, FL.

Casasent, D. and Ye, A. (1997). Detection filters and algorithm fusion for ATR. *IEEE Transactions on Image Processing*, 6:114–125.

Civil Aviation Safety Regulations 1998 (2009). Technical report, Civil Aviation Safety Authority, Australian Government. SR 1998 No.237.

Davey, S. J., Rutten, M. G., and Cheung, B. (2008). A comparison of detection performance for several track-before-detect algorithms. *EURASIP Journal on Advances in Signal Processing*, 2008:1–10.

DeGarmo, M. T. (2004). Issues concerning integration of unmanned aerial vehicles in civil airspace. Technical report, The MITRE Corporation. MP 04W0000323.

Deshpande, S. D., Er, M. H., Venkateswarlu, R., and Chan, P. (1999). Max-mean and max-median filters for detection of small targets. In *Proceedings of Signal and Data Processing of Small Targets*, Denver, CO.

Dey, D., Geyer, C., Singh, S., and Digioia, M. (2009). Passive, long range detection of aircraft: Towards a field deployable sense and avoid system. In *Proceedings of Field and Service Robotics*, Cambridge, MA.

Dougherty, E. R. and Lotufo, R. A. (2003). *Hands-on Morphological Image Processing*. Bellingham, WA: SPIE Optical Engineering Press.

Ebdon, M. D. and Regan, J. (2004). Sense-and-avoid requirement for remotely operated aircraft (ROA). Technical report, Air Combat Command, Langley Air Force Base.

Elliott, R. J., Aggoun, L., and Moore, J. B. (1995). *Hidden Markov Models: Estimation and Control*. Berlin: Springer-Verlag.

FAA Advisory Circular: Pilots' role in collision avoidance (1983). Technical report, Federal Aviation Administration, U.S. Department of Transportation. AC 90-48C.

Forney, G. D. J. (1973). The Viterbi algorithm. *Proceedings of the IEEE*, 61:268–278.

Gandhi, T., Yang, M.-T., Kasturi, R., Camps, O., Coraor, L., and McCandless, J. (2003). Detection of obstacles in the flight path of an aircraft. *IEEE Transactions on Aerospace and Electronic Systems*, 39:176–191.

Gandhi, T., Yang, M.-T., Kasturi, R., Camps, O., Coraor, L., and McCandless, J. (2006). Performance characterisation of the dynamic programming obstacle detection algorithm. *IEEE Transactions on Image Processing*, 15:1202–1214.

Geyer, C., Dey, D., and Singh, S. (2009). Prototype sense-and-avoid system for UAVs. Technical report, Robotics Institute, Carnegie Mellon University. CMU-RI-TR-09-09.

Geyer, C., Singh, S., and Chamberlain, L. (2008). Avoiding collisions between aircraft: State of the art and requirements for UAVs operating in civilian airspace. Technical report, Robotics Institute, Carnegie Mellon University. CMU-RI-TR-08-03.

Gonzalez, R. C., Woods, R. E., and Eddins, S. L. (2004). *Digital Image Processing using MATLAB*, chapter Morphological Image Processing, pages 334–377. Upper Saddle River, NJ: Pearson Prentice Hall.

JiCheng, L., ZhengKang, S., and Tao, L. (1996). Detection of spot target in infrared clutter with morphological filter. In *Proceedings of National Aerospace and Electronics Conference*, Dayton, OH.

Johnston, L. A. and Krishnamurthy, V. (2000). Performance analysis of a track before detect dynamic programming algorithm. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Istanbul.

Jorge, L. and Jorge, D. (2004). Inertial sensed ego-motion for 3D vision. *Journal of Robotic Systems*, 21:3–12.

Jung, B. and Sukhatme, G. S. (2005). Real-time motion tracking from a mobile robot. Technical report, Center for Robotics and Embedded Systems - University of Southern California. CRES-05-008.

Karhoff, B., Limb, J., Oravsky, S., and Shephard, A. (2006). Eyes in the domestic sky: An assessment of sense and avoid technology for the army's "warrior" unmanned aerial vehicle. In *Proceedings of IEEE Systems and Information Engineering Design Symposium (SIEDS)*, Charlottesville, VA.

Lai, J. and Ford, J. J. (2010). Relative entropy rate based multiple hidden Markov model approximation. *IEEE Transactions on Signal Processing*, 58:165–174.

Lai, J., Ford, J. J., O'Shea, P., and Walker, R. (2008). Hidden Markov model filter banks for dim target detection from image sequences. In *Proceedings of Digital Image Computing: Techniques and Applications (DICTA)*, Canberra.

Limitations of the See-and-Avoid Principle (1991). Technical report, Australian Transport Safety Bureau, Australian Government.

Lucas, B. D. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, Vancouver.

Maroney, D. R., Boiling, R. H., Heffron, M. E., and Flathers, G. W. (2007). Experimental platforms for evaluating sensor technology for UAS collision avoidance. In *Proceedings of Digital Avionics Systems Conference (DASC)*, Dallas, TX.

NVIDIA CUDA Compute Unified Device Architecture: Programming Guide Version 2.0 (2008). Retrieved February 10, 2010, from http://developer.nvidia.com/object/cuda_docs.html.

Office of the Secretary of Defense (2004). Airspace integration plan for unmanned aviation. Technical Report.

Ohmer, J. F. and Redding, N. (2008). GPU-accelerated KLT tracking with monte-carlo-based feature reselection. In *Proceedings of Digital Image Computing: Techniques and Applications (DICTA)*, Canberra.

Owens, J. D., Luebke, D., Govindaraju, N., Harris, M., Kruger, J., Lefohn, A. E., and Purcell, T. J. (2005). State-of-the-art-reports: A survey of general-purpose computation on graphics hardware. In *Proceedings of European Association for Computer Graphics Conference (Eurographics)*, Dublin.

Parkinson, B. W. and Spilker, Jr., J. J., editors (1996). *Global Positioning System: Theory and Applications*, volume 2, chapter GPS Error Analysis, pages 478–483. American Institute of Aeronautics and Astronautics.

Petridis, S., Geyer, C., and Singh, S. (2008). *Lecture Notes in Computer Science*, volume 5008, chapter Learning to Detect Aircraft at Low Resolutions, pages 474–483. Berlin/Heidelberg: Springer.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–286.

Schaefer, R. and Casasent, D. (1995). Nonlinear optical hit-miss transform for detection. *Applied Optics*, 34:3869–3882.

Soille, P. (2003). *Morphological Image Analysis: Principles and Applications*, chapter Opening and Closing, pages 105–137. Berlin: Springer, 2nd edition.

Standard specification for design and performance of an airborne sense-and-avoid system, F2411-04 (2004). ASTM International, West Conshohocken, PA.

Tom, V. T., Peli, T., Leung, M., and Bondaryk, J. E. (1993). Morphology-based algorithm for point target detection in infrared backgrounds. In *Proceeding of Signal and Data Processing of Small Targets*, Orlando, FL.

Tonissen, S. M. and Evans, R. J. (1996). Performance of dynamic programming techniques for track-before-detect. *IEEE Transactions on Aerospace and Electronic Systems*, 32:1440–1451.

Unmanned Aircraft Systems Roadmap 2007–2032 (2007). Technical report, Office of the Secretary of Defense, Department of Defense.

Utt, J., McCalmont, J., and Deschenes, M. (2004). Test and integration of a detect and avoid system. In *Proceedings of AIAA Infotech*, Chicago, IL.

Utt, J., McCalmont, J., and Deschenes, M. (2005). Development of a sense and avoid system. In *Proceedings of AIAA Infotech*, Arlington, VA.

Warren, R. C. (2002). A bayesian track-before-detect algorithm for ir point target detection. Technical report, Defence Science and Technology Organisation - Aeronautical and Maritime Research Laboratory.

Xie, L., Ugrinovskii, V. A., and Petersen, I. R. (2005). Probabilistic distances between finite-state finite-alphabet hidden Markov models. *IEEE Transactions on Automatic Control*, 50:505–511.

Yu, N., Wu, H., Wu, C., and Li, Y. (2003). Automatic target detection by optimal morphological filters. *Journal of Computer Science and Technology*, 18:29–40.

Zeng, M., Li, J., and Peng, Z. (2006). The design of top-hat morphological filter and application to infrared target detection. *Infrared Physics & Technology*, 48:67–76.

Zhu, Z., Li, Z., Liang, H., Song, B., and Pan, A. (2000). Gray-scale morphological filter for small-target detection. In *Proceedings of SPIE International Symposium on Optical Science and Technology: Infrared Technology and Applications*, San Diego, CA.