2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

# Probabilistic Target Detection by Camera-Equipped UAVs

Andrew Symington, Sonia Waharte, Simon Julier, Niki Trigoni

*Abstract*— **This paper is motivated by the real world problem of search and rescue by unmanned aerial vehicles (UAVs). We consider the problem of tracking a static target from a bird's-eye view camera mounted to the underside of a quadrotor UAV. We begin by proposing a target detection algorithm, which we then execute on a collection of video frames acquired from four different experiments. We show how the efficacy of the target detection algorithm changes as a function of altitude. We summarise this efficacy into a table which we denote the *observation model*. We then run the target detection algorithm on a sequence of video frames and use parameters from the observation model to update a recursive Bayesian estimator. The estimator keeps track of the probability that a target is currently in view of the camera, which we refer to more simply as *target presence*. Between each target detection event the UAV changes position and so the sensing region changes. Under certain assumptions regarding the movement of the UAV, the proportion of new information may be approximated to a value, which we then use to weight the prior in each iteration of the estimator. Through a series of experiments we show how the value of the prior for unseen regions, the altitude of the UAV and the camera sampling rate affect the accuracy of the estimator. Our results indicate that there is no single optimal sampling rate for all tested scenarios. We also show how the prior may be used as a mechanism for tuning the estimator according to whether a high false positive or high false negative probability is preferable.**

*Index Terms*— **UAV, vision, target detection, sensing**

## I. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) are typically commissioned for tasks that are perceived as too dull or dangerous for a human pilot to carry out [13]. In the past, research involving UAVs was constrained by large and expensive flight platforms which offered greater payloads. However, recent advances in embedded computing and sensors have made small, low-cost autonomous systems accessible to the broader research community.

In this paper we consider a single UAV with a bird's-eye view video camera, which it uses to sense the world beneath it. Computer vision is an active research field that has already seen various applications to UAVs, such as navigation [12], stabilisation and localisation [11], feature tracking [7], SLAM [5], collision avoidance [15] and autonomous landing [3]. The work in this paper was inspired by Mondragon *et al* [7] who use vision-based target tracking to estimate the velocity of the UAV relative to a global visual reference frame. In this work we adopt a similar vision-based algorithm, except we use it to track whether or not a particular target is in view of the camera. We then measure the effect of the altitude of the UAV and the frame rate of the camera on the accuracy of the tracker over time.

We begin by proposing a *target detection algorithm* that acts as a binary classifier or, put more simply, it determines whether or not a target object exists in a video frame. One would expect that the efficacy of such a classifier varies according to the physical appearance of the target, camera resolution, lighting conditions, etc. However, in our work we will assume that these factors remain constant and the efficacy is simply a function of the UAV's altitude: the further the camera from the target, the less information is available to the target detection algorithm, which causes it to miss targets. To this end, we run the target detection algorithm on a series of video frames and tabulate its efficacy as a function of altitude, which we denote the *observation model*.

The target detection algorithm treats each frame independently and so we therefore introduce a recursive Bayesian estimator to fuse a series of noise-affected detection events (observations) over time. The estimator takes this series to track the probability of *target presence* (whether the target is in view of the camera), taking into account the efficacy of the target detection algorithm at the current altitude. The values it uses to quantify this efficacy are drawn directly from the observation model. Our estimator also takes into account the fact that the camera view changes over time. Under certain assumptions regarding the movement of the UAV, the proportion of new information is given by the sampling rate of the detector. We therefore introduce a term that decays the estimate according to the proportion of new information, which we call the *exploration ratio*, that is added between successive observations.

The remainder of this paper is organised as follows. In Section II we describe the series of experiments that were conducted to obtain the video data used in this paper and show the post-processing steps that we followed to label the images with a ground truth. In Section III we begin by describing the target detection algorithm. We then measure the efficacy of this algorithm against a set of images in order to construct the observation model. In Section IV we firstly show the methodology behind the calculation of the exploration ratio for our UAV platform and then introduce the recursive Bayesian estimator. In Section V we first measure the accuracy of our estimator by comparing the probability of target presence against the ground truth over time, varying the UAV altitude, sampling rate and prior. We then comment

This work was supported by the SUAAVE project. Further information about this project can be found at http://www.suaave.org.

Andrew Symington, Niki Trigoni and Sonia Waharte are with the Oxford University Computing Laboratory, Wolfson Building, Oxford, OX1 3QD, UK. niki.trigoni@comlab.ox.ac.uk

Simon Julier is with the Department of Computer Science, University College London, Malet Place, London, WC1E 6BT, UK. s.julier@cs.ucl.ac.uk

on our findings. Section VI concludes this paper.

## II. DATA ACQUISITION AND PREPARATION

In this section we discuss how our video data was acquired and then prepared for use by the target detection algorithm.

### A. Acquisition

In order to obtain video data used in this paper we fixed a *FlyCamOne2* video camera to the underside of an Ascending Technologies Hummingbird quadrotor UAV. A number of targets were positioned in a $20m \times 20m$ grid on a flat grass field. We flew the UAV over the grid at fixed altitudes of 5m, 10m, 15m and 20m. In addition to capturing video data at 27 frames per second, the UAV also recorded GPS and inertial data at around 10Hz. We found that the human target depicted in Fig. 2 consistently yielded a sufficient amount of information to train an effective target detection algorithm, so we used it as the target in our model.

### B. Preparation

The first objective of data preparation is to isolate a set of example video frames that we will use to train the target detection algorithm. Each of these video frames must contain an unoccluded example image of the target. We draw a rectangle around the target and only the information within that rectangle is used to train the target detection algorithm. For our data set we used ten such frames for each altitude. These frames constitute the *training set*, while the remainder constitute the *evaluation set*.

The second objective is to label each frame in the evaluation set with a *ground truth*. The ground truth is effectively a binary flag which tell us whether the frame contains either (i) some or all of the target, or (ii) none of the target at all. The efficacy of the target detection algorithm will be measured by comparing the result of the detector against the ground truth, for all frames in the evaluation set.

## III. TARGET DETECTION

This section begins by describing the target detection algorithm that we used. Our goal is not to present a novel and provably superior algorithm, but rather to leverage existing techniques to create a realistic system, which we can use to generate meaningful results. In essence, one could use an alternate method such as Viola and Jones's [14] boosting to achieve exactly the same outcome, perhaps with a greater accuracy. However, regardless of the algorithm that is chosen, it will act as a binary classifier for unseen images. It the second part of this section we show how to measure the efficacy of the target detection algorithm and summarise it in the form of an observation model, which we will then use in the next section to update our belief of target presence.

### A. The target detection algorithm

For our application we require a target detection algorithm that determines whether or not a single image contains an instance of the target. In order to achieve this we use Bay, Tuytelaars and Van Gool's [1] Speeded-up Robust Features

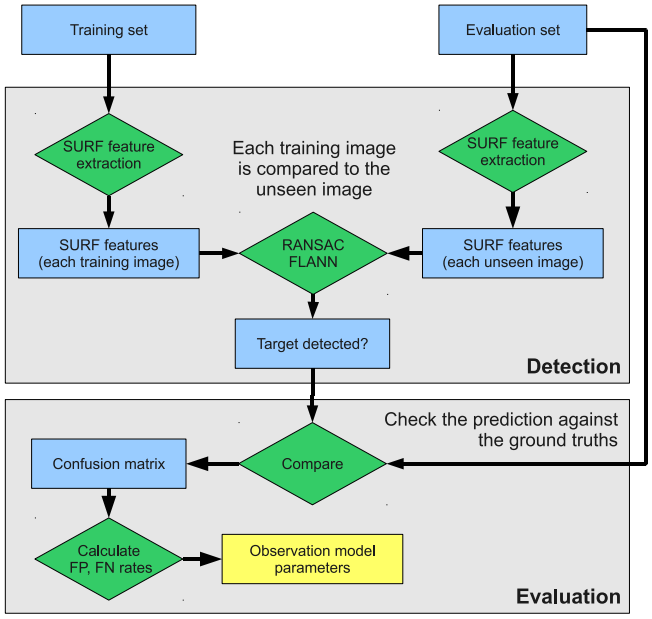


Fig. 1. This diagram provides a high-level summary of the target detection algorithm and the method by which it was evaluated. Recall that the video frames are split into a training set and evaluation set. For each frame in the evaluation set, the target detection algorithm loops through the template example and determines whether it contains the target object: the algorithm uses SURF keypoint matching with FLANN to find the most likely homographic projection of the training image into the evaluation image. The result of the target detection algorithm is compared to the ground truth label to form the observation model.

(SURF)[1]. The features produced by SURF are, essentially, keypoints on a 2D image that are robust to *slight* changes in scale, rotation and perspective. Each SURF keypoint has an associated multidimensional descriptor[2] that characterises the grayscale image gradient in the region surrounding the keypoint. The similarity between two keypoints is calculated by measuring the distance between their descriptors with the $n$-dimensional Euclidean metric. The Hessian determinant threshold governs the sensitivity of the SURF detector and, hence, the number of features that are returned. We determined empirically that a threshold of 500 culled many of the weaker (and often background) keypoints, while maintaining an acceptable number of keypoints on objects at high altitudes.

Recall that in the previous section we divided all the video frames in our data set into a training set and an evaluation set. For clarity, let us assume that we are working with the data from one altitude only. Now, assume that we are given some arbitrary image which might or might not contain the target, or part of a target. We refer to this as the *unseen image*. The target detection algorithm simply loops over all images in the training set and attempts to locate each one of these images within the unseen image. If a location is

[1]The algorithm that we implemented is based on the find_obj.cpp sample code in the OpenCV pre-2.0 distribution.

[2]Usually a 64 or 128 double vector, depending on the required resolution.

found for any of the training images the algorithm returns a *positive detection* event, which signals that the target was found. Otherwise, a *negative detection* event is returned.

What remains to be explained is how the training image is located in an unseen image. This is where the SURF features are used. The detection algorithm begins by calculating the SURF keypoints for both images. The aim is to find a *correspondence* between the keypoints in the training image and the keypoints in the unseen image. In order to do so the Fast Library for Approximate Nearest Neighbors (FLANN) [8] algorithm is used. This algorithm provides a fast approximation to $k$ nearest-neighbour classification. The result being that each keypoint in the unseen image is paired with its closest keypoint in the training image. Recall that the similarity of two keypoints is calculated by measuring the distance between their two associated SURF descriptors.

In the next stage of the detection phase we remove all of the weak correspondences. Weak correspondences usually occur when some keypoint located on the background clutter in the unseen image is incorrectly paired with a keypoint in the training image; FLANN always maps to the nearest neighbour, regardless of the distance to it. The intuitive way to do this would be to threshold the correspondences based on the distance between each pair of keypoints. In practice, this heuristic fails. A superior approach involves thresholding based on the distance ratio between each kepyoint in the unseen image and its two closest matching neighbours. That is, if the ratio of the distances between the two closest matches is greater than some threshold we cull the correspondence — the idea being that if one keypoint in the unseen image maps to two keypoints in the template image with equal strength, it is unlikely that it describes some particular feature uniquely [6]. Rather, it is more likely that the feature is background clutter being arbitrarily mapped to close neighbours in descriptor space. Ramisa *et al* [10] discuss the selection of this threshold for a variety of keypoint detectors. Their research shows that good results are typically obtained for SURF using a threshold between 0.6 and 0.8. Through experimentation we found that a value of 0.6 was best for our application: many background keypoints were discarded, while meaningful keypoints were preserved.

Finally, the correspondence set is passed to the RANdom SAmple Consensus (RANSAC) [4] algorithm, which determines the most likely projection of the template image into the unseen image, given the presence of some statistical outliers. Figure 2 illustrates this process – it shows the correspondence set as a collection of white lines connecting the template image keypoints to the unseen image keypoints. The projection is shown as a bounding polygon in the scene. If the RANSAC algorithm finds a projection and it has greater than five correspondences we assume that the scene contains the target. Through experimentation we found that if the threshold is set any lower the target detection algorithm returns significantly more false detections at lower altitudes. Conversely, if we set the threshold any higher, the target detection shows significantly more false negatives occur at higher altitudes.

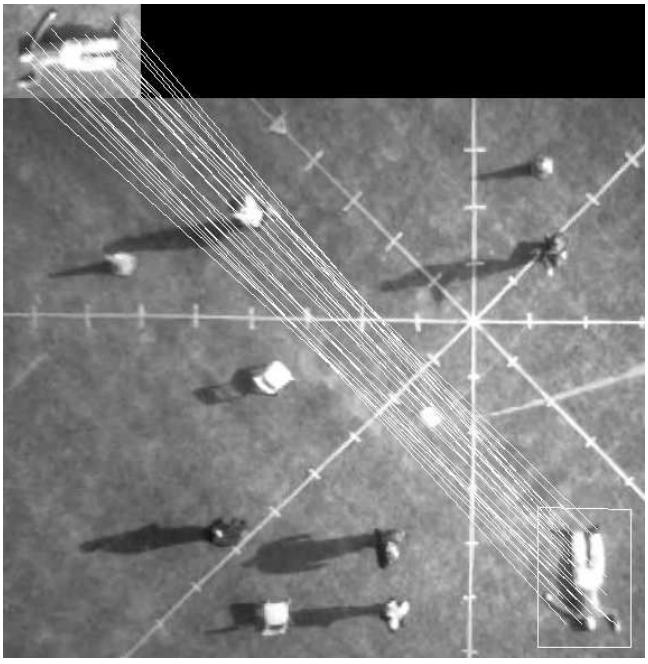

Fig. 2. At the top left of this figure is an example training image and beneath it is an unseen image containing the target. SURF keypoints for both the target image and scene are calculated. FLANN is used to find mappings from keypoints in the unseen image to keypoints in the template image (shown as lines in the figure). For clarity, we have not drawn any weak mappings – those which have a distance ratio above the threshold. The RANSAC algorithm uses the mappings to calculate the most likely projection of the training image into the unseen image (shown as a polygon).

### *B. Observation model*

The performance of a machine learning algorithm is measured by counting the number of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN). For a binary classifier these values are typically expressed in the form of a 2x2 confusion matrix [9].

To evaluate the performance of the target detection algorithm we executed it on every frame in the evaluation set and compared the detection result to the ground truth. If the target detection algorithm agreed with the ground truth, we incremented the TP and TN count for positive and negative detection events respectively. If the target detection algorithm detected a target incorrectly, the false positive count was incremented. On the other hand, if the target detection algorithm failed to detect a target that was there, the false negative count was incremented. We repeated this process for all four altitudes and the resultant confusion matrix is listed in Table I.

The observation model is derived directly from the confusion matrix. In essence, the observation model is simply the false positive probability $\alpha_h$ and false negative probability $\beta_h$ as a function of the UAV's altitude $h$. We calculate values for $\alpha_h$ and $\beta_h$ using Eqn. 1 and Eqn. 2 respectively. The

values for our data set are also listed in Table I .

$$\alpha_h = \frac{FP}{(FP+TN)} \quad (1)$$

$$\beta_h = \frac{FN}{(FN+TP)} \quad (2)$$

## IV. RECURSIVE BAYESIAN ESTIMATOR

In this section we firstly describe how to measure the amount of new information that is introduced between two observations, which we call the exploration ratio. We then present the recursive Bayesian estimator, which uses both the exploration ratio and the observation model for a given altitude to maintain a best estimate of target presence.

TABLE I
CONFUSION MATRICES AND OBSERVATION MODEL

| Altitude | Truth | Detected | Not Detected | $\alpha_h$ | $\beta_h$ |
|---|---|---|---|---|---|
| 5m | Present | 88 | 24 | 0.24569 | 0.21428 |
| | Absent | 685 | 2103 | | |
| 10m | Present | 100 | 25 | 0.06286 | 0.20000 |
| | Absent | 87 | 1296 | | |
| 15m | Present | 516 | 258 | 0.03107 | 0.33333 |
| | Absent | 38 | 1185 | | |
| 20m | Present | 571 | 302 | 0.00130 | 0.34593 |
| | Absent | 2 | 1526 | | |

### A. Calculating the exploration ratio

The role of the exploration ratio is to measure the proportion of new information that is introduced at each observation, resulting from the movement of the UAV, as a function of the UAV's altitude and the sampling rate of the sensor. To simplify the calculation of the exploration ratio we will make the following assumption regarding the movement of the UAV: it always moves at a constant speed in a single direction. Although one could use a more accurate method that takes into account the attitude and velocity of the UAV, for this initial study we use a simpler approximation.

Let $x_h$ and $y_h$ be the length and width of the camera sensing region at some altitude $h$, all of which are given in meters. Both $x_h$ and $y_h$ are related to one another according to the aspect ratio of the camera. In Fig. 3 we show the camera sensor coverage after the UAV displaces some distance $d$ as a result of moving in the specified direction. The new, shared and lost areas are clearly marked in the figure. The exploration ratio $e$ is simply the ratio of new area to entire observation area. In order to calculate this ratio we'll first need a value for $d$. The value for $d$ is calculated by dividing the constant velocity $v$ of the UAV (in meters per second) by the sampling rate $r$ (in frames per second). Eqn. 3 shows the full equation for calculating $e$.

$$e = \frac{dx_h}{y_h x_h} \quad (3)$$

$$= \frac{v}{r y_h} \quad (4)$$

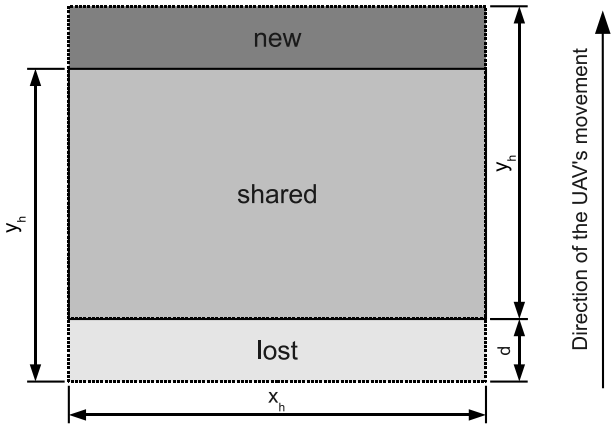


Fig. 3. The coverage or observation region of the video camera sensor is given by a rectangle of length $x$ and width $y$. Between each observation the UAV moves some small distance $d$ and the observation region changes accordingly. The ratio of new area $(d \times x_h)$ over the total sensing area $(x_h \times y_h)$ is referred to as the exploration ratio and it varies as a function of altitude and sampling rate.

We determined the $x_h$ and $y_h$ value for each altitude in our video data set using the one meter interval ticks on the star-shaped calibration pattern that we laid on the ground. The calibration pattern is clearly visible in the sample frame shown in Fig. 2. We then measured the average velocity of the UAV by integrating the acceleration readings from the inertial data to determine a reasonable value for $v$, which turned out to be slightly over one meter per second. Finally, we calculated the $e$ value for all combinations of the four altitudes (5m, 10m, 15m and 20m) and sampling rates (1 ,5 and 10 frames per second) that we chose to test. The result of our calculations are listed in Table II.

TABLE II
EXPLORATION RATIO FOR VARIOUS ALTITUDES AND SAMPLING RATES

| Altitude $h$ | $y_h$ | $r = 1$ FPS | $r = 5$ FPS | $r = 10$ FPS |
|---|---|---|---|---|
| 5m | 4m | 0.250000 | 0.050000 | 0.025000 |
| 10m | 7m | 0.142857 | 0.028571 | 0.014286 |
| 15m | 10m | 0.100000 | 0.020000 | 0.010000 |
| 20m | 13m | 0.076923 | 0.015385 | 0.007692 |

### B. The recursive Bayesian estimator

The role of the recursive Bayesian estimator is to take a series of observations and maintain the the probability of *target presence*. Moreover, the estimator takes into account the fact that the camera view changes over time and also that there is some error associated with the observation, both of which vary with altitude.

The observation model parameters are the probability of false positive and the probability of false negative, we defined earlier as $\alpha_h$ and $\beta_h$ respectively for some altitude $h$. Let us assume that the camera sensor has an *observation region* $\mathcal{O}(k^t)$ which is visible from the camera when the UAV is located at position $k^t$ at time $t$. Let $x_T$ represent the target.

We use Chung's [2] error model, where $d^t = 0$ and $d^t = 1$ denote negative and positive target detection events:

$$Pr_h(d^t = 1|x_T \in \mathcal{O}\left(k^t\right)) = 1 - \beta_h$$
$$Pr_h(d^t = 0|x_T \in \mathcal{O}\left(k^t\right)) = \beta_h$$
$$Pr_h(d^t = 0|x_T \notin \mathcal{O}\left(k^t\right)) = 1 - \alpha_h$$
$$Pr_h(d^t = 1|x_T \notin \mathcal{O}\left(k^t\right)) = \alpha_h$$

Let $d^t$ be the $t^{th}$ observation, $D^t$ be the set of $t$ observations and let $x_T = 1$ be the event that a target exists in a particular frame. The probability that the target is present in the frame at time $t$ is computed using Bayes rule:

$$Pr(x_T = 1|D^t) = \frac{Pr(d^t|x_T = 1)Pr(x_T = 1|D^{t-1})}{Pr(d^t|D^{t-1})} \quad (5)$$

The update equation for the recursive Bayesian estimator is conditional on whether a positive or negative detection event is encountered. Eqn. 6 shows this update equation.

$$P_t = \begin{cases} \frac{(1-\beta_h)P_{t-1}}{(1-\beta_h)P_{t-1}+\alpha_h(1-P_{t-1})}, & \text{if } d^t = 1 \\ \frac{\beta_h P_{t-1}}{\beta_h P_{t-1}+(1-\alpha_h)(1-P_{t-1})}, & \text{if } d^t = 0 \end{cases} \quad (6)$$

So far the estimator has implicitly assumed that the camera view does not change between observations. However, since the UAV is moving this is not the case. We therefore introduce a state transition term to the estimator that takes into account that a portion of the current frame contains new information. The probability $P_0$ represents our prior belief of target presence for some unexplored region. In each iteration the refactored prior, given in Eqn. 7, is a weighted combination of the previous step's posterior and $P_0$.

$$P_{t-1} \leftarrow eP_0 + (1 - e)P_{t-1} \quad (7)$$

This weighted update equation causes the estimate to converge exponentially to $P_0$ over time. This is useful for two reasons. Firstly, it takes into account the fact that the camera changes position over time and, hence, objects may appear and disappear from view. Secondly, it provides a method of ensuring that the estimate never converges to zero or one after a series of positive or negative detection events, which happens as a result of the limited storage capacity of floating point data types.

## V. EXPERIMENTS AND RESULTS

We conducted a series of experiments in order to measure the effect of altitude, sampling rate and prior on the performance of the estimator. We used real streams of video frames, taken from four different altitudes (5m, 10m, 15m and 20m)[3]. For example, Fig. 4 shows the result of running the Bayesian estimator on a video stream taken from an altitude of 5m. If after an observation the posterior probability of the estimator exceeds 0.5 (see the dashed line in the bottom three graphs in Fig. 4) we consider this to be a positive detection event. By comparing the ground truth with the estimator's predictions, we can measure the probability of the detector making a false positive prediction and that of making a false negative prediction. We use these two metrics to assess the estimator's accuracy[4].

The graphs in Fig. 5 show the accuracy of the estimator when the altitude is 5m and 20m, and for different prior ($P_0$) values and sampling rates. Our first observation is that as the prior increases, so the false positive probability increases, whereas the false negative probability decreases. Our second observation is that the lower the sampling rate, the higher the exploration ratio, and thus the higher the impact of prior on the false positive and false negative probabilities. These two observations held for all four altitudes tested, although not all graphs are included for space reasons.

Our third observation relates to the effect of altitude on the two estimator metrics. When the altitude changes the estimator uses a new set of parameters from the observation model and a different exploration ratio. The effect of the exploration ratio is relatively straightforward and we discussed it in the previous paragraph. However, the effect of the observation model parameters is less obvious, despite there being a general trend in the parameters themselves — in Table I we see that $\alpha_h$ and and $\beta_h$ decrease and increase respectively with altitude. Moreover, any trend that might exist may be further obfuscated by the fact that different video sequences were used to evaluate the four different altitudes. Therefore, we cannot draw any conclusive evidence from our results that suggest a trend based on altitude.

Finally, our experimental results show that that there is no best sampling rate for all scenarios. The optimal sampling rate depends on the altitude, on the prior, as well as on whether the application is more interested in reducing false positives or false negatives.

## VI. CONCLUSION AND FUTURE WORK

In this paper we use video data to train a target detection algorithm and measure parameters for an observation model that describes its efficacy. We then implemented a recursive Bayesian estimator to fuse a series of detections over time, taking into account the observation model and exploration ratio associated with the altitude at which the observations occur. Finally, we conducted a series of experiments to test the impact of the prior, altitude and sampling rate on the performance of the estimator, compared to the ground truth.

While our results show that sampling rate has a significant effect on the estimator's performance, it is clear that there is no optimal sampling rate that fits all scenarios. The prior should be chosen in conjunction with the application requirements — in the case of search and rescue one would seek to minimize the false negative probability, while for situations where there are energy or resource constraints one would seek to minimize the false positive probability.

In future work we plan to run the full estimator online. We also plan to conduct a detailed study of how altitude affects the performance of the estimator.

[3]Recall that the accuracy of the target detection algorithm depends on the altitude, as shown in Table I.

[4]These probabilities must not be confused with $\alpha_h$ and $\beta_h$, which measure the accuracy of the target detection algorithm.

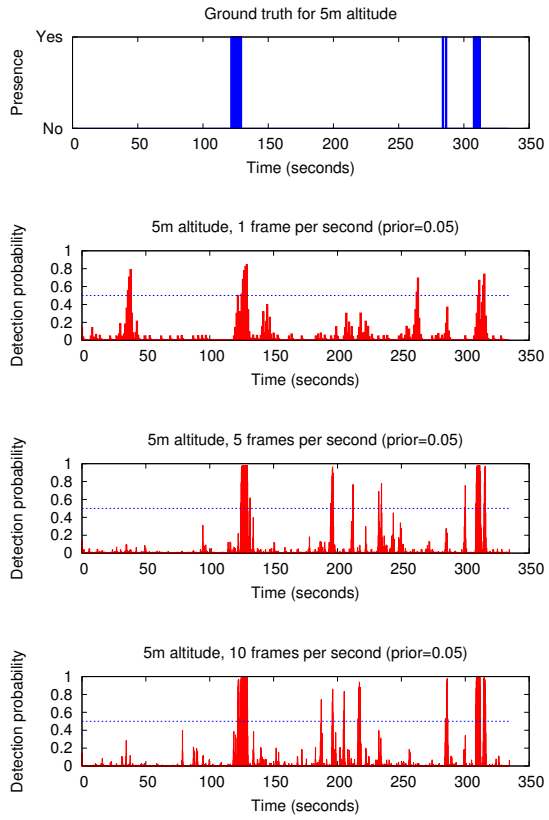


Fig. 4. The top graph shows the ground truth for the video data captured at 5m. The three graphs below show the evolution of the probability of target presence for the same altitude and period for 1 FPS, 5 FPS and 10 FPS. The dashed line at 0.5 is the threshold for a positive detection.

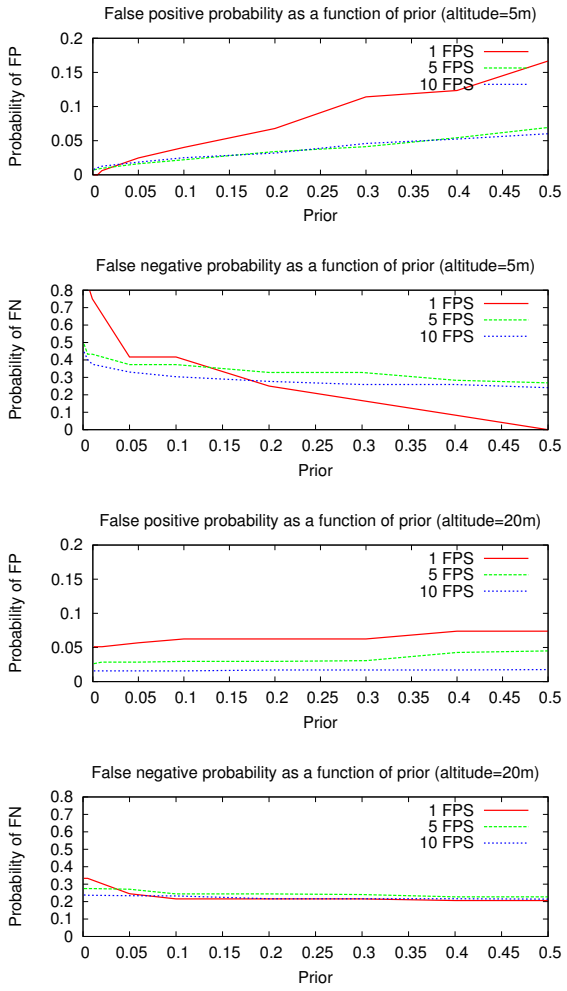


Fig. 5. The four graphs above show how the value of the prior affects the performance of the estimator, in terms of the false positive and false negative probabilities for various sampling rates. The top and bottom two graphs were generated from 5m and 20m data respectively.

## VII. ACKNOWLEDGMENTS

This research was supported by the Sensing Unmanned Autonomous Aerial Vehicles (SUAAVE) project under grants EP/F064217/1, EP/F064179/1 and EP/F06358X/1. Specifically, we'd like to thank Stephen Hailes, Renzo de Nardi, Graeme McPhillips, Mohib Wallizada and Dietmar Backes for assisting with the acquisition of the video data.

## REFERENCES

[1] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, 2008.

[2] T. Chung and J. Burdick, "A Decision-Making framework for control strategies in probabilistic search," in *IEEE International Conference on Robotics and Automation, 2007*, 2007, pp. 4386–4393.

[3] Y. Fan, S. Haiqing, and W. Hong, "A vision-based algorithm for landing unmanned aerial vehicles," in *Computer Science and Software Engineering, 2008 Intl. Conf. on*, vol. 1, Dec. 2008, pp. 993–996.

[4] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[5] T. Lemaire, S. Lacroix, and J. Sola, "A practical 3d bearing-only slam algorithm," in *Intelligent Robots and Systems, 2005 IEEE/RSJ International Conference on*, Aug. 2005, pp. 2449–2454.

[6] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Intl. Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[7] I. Mondragon, P. Campoy, J. Correa, and L. Mejias, "Visual model feature tracking for uav control," in *Intelligent Signal Processing. 2007 IEEE International Symposium on*, Oct. 2007, pp. 1–6.

[8] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *International Conference on Computer Vision Theory and Applications (VISAPP'09)*, Lisboa, Portugal, May 2009.

[9] F. Provost and R. Kohavi, "On applied research in machine learning," in *Machine learning*, 1998, pp. 127–132.

[10] A. Ramisa, S. Vasudevan, D. Aldavert, R. Toledo, and R. Lopez de Mantaras, "Evaluation of the sift object recognition method in mobile robots," in *Proceeding of the 2009 conference on Artificial Intelligence Research and Development*. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2009, pp. 9–18.

[11] H. Romero, R. Benosman, and R. Lozano, "Stabilization and location of a four rotor helicopter applying vision," in *American Control Conference, 2006*, June 2006, pp. 6 pp.–.

[12] B. Sinopoli, M. Micheli, G. Donato, and T. Koo, "Vision based navigation for an unmanned aerial vehicle," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 2, 2001, pp. 1757–1764 vol.2.

[13] P. van Blyenburgh, "UAVs: an overview," *Air & Space Europe*, vol. 1, no. 5-, pp. 43 – 47, 1999.

[14] P. Viola and M. J. Jones, "Robust real-time face detection," *Intl. Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, May 2004.

[15] J.-C. Zufferey and D. Floreano, "Toward 30-gram autonomous indoor aircraft: Vision-based obstacle avoidance and altitude control," in *IEEE International Conference on Robotics and Automation, 2005*, April 2005, pp. 2594–2599.