Autonomous Altitude Estimation Of A UAV Using A Single Onboard Camera

Anoop Cherian† Jon Andersh† Vassilios Morellas† Nikolaos Papanikolopoulos† Bernard Mettler⋆
†{ cherian, jander, morellas, npapas }@cs.umn.edu ⋆mettler@aem.umn.edu
†Department of Computer Science ⋆Department of Aerospace Engineering
University of Minnesota, Minneapolis, MN 55455

***Abstract*— Autonomous estimation of the altitude of an Unmanned Aerial Vehicle (UAV) is extremely important when dealing with flight maneuvers like landing, steady flight, etc. Vision based techniques for solving this problem have been underutilized. In this paper, we propose a new algorithm to estimate the altitude of a UAV from top-down aerial images taken from a single on-board camera. We use a semi-supervised machine learning approach to solve the problem. The basic idea of our technique is to learn the mapping between the texture information contained in an image to a possible altitude value. We learn an over complete sparse basis set from a corpus of unlabeled images capturing the texture variations. This is followed by regression of this basis set against a training set of altitudes. Finally, a spatio-temporal Markov Random Field is modeled over the altitudes in test images, which is maximized over the posterior distribution using the MAP estimate by solving a quadratic optimization problem with L1 regularity constraints. The method is evaluated in a laboratory setting with a real helicopter and is found to provide promising results with sufficiently fast turnaround time.**

I. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) have been an active area of research in the recent years. UAVs have been found to be an ideal platform for a number of civilian and military tasks like visual surveillance, inspection, firefighting, policing civil disturbances or reconnaissance support in natural disasters. The ability of UAVs to fly at low speeds, hover or fly laterally and perform maneuvers in narrow spaces facilitate them for these tasks. One of the most important tasks in achieving UAV autonomy is autonomous navigation, which needs good altitude estimation techniques. The main surge in mini-UAV designs these days have been on optimizing and miniaturizing the hardware and putting multiple functionalities into the same device. On-board cameras are indispensible components of a UAV, enabling it for environment monitoring, tracking etc. Compared to other sensors, (e.g. laser), video cameras are quite light and less power hungry. In this paper, we investigate the idea of estimating the altitude of a UAV from the images taken from a single on board camera using machine learning techniques.

Vision based control of an autonomous helicopter has been investigated quite thoroughly in the previous years. Different camera systems and arrangements have been tried. A downward-looking camera with a standard lens has been investigated in [11], [4], [15], but the state estimation of their approach is relative to the specifics of a given landing pad. In [9], a multi view geometry based approach to build a digital map of the ground is suggested. They use aerial image sequences taken from a side looking helicopter camera, with the assumption that there are uniquely recognizable features in the vicinity of the UAV to correlate the images in the sequence. An application of omni directional cameras for vision based navigation is described in [14], but the environment over which this is used seems very restrictive. A reinforcement learning strategy for performing various flight maneuvers have been investigated in [10], but they do not use any vision based techniques.

To the best of our knowledge, this is the first time that the problem of altitude estimation of a UAV has been studied exclusively and a machine learning framework being suggested. The motivation for this research stems from the recent developments in the area of 3D reconstruction using monocular cues. In [2], [1] and [3] Saxena et. al. proposes an algorithm for building a depth map from a single image. The algorithm uses a Markov Random Field (MRF) based supervised learning to build a model of the variation of depth at each pixel in a given image against a set of feature vectors computed from those pixels. But we found that their method cannot be applied to our problem due to the following reasons: (i) we have top-down aerial views, (ii) there is little structure compared to images taken on ground and (iii) we assume that the ground plane is flat; thus needing to compute only a single altitude from the entire image. We also assume that the UAV does not make sudden changes in altitude such that the deviation of altitude from one image to its preceding images is smooth. We incorporate this information also into our model to refine the predicted altitude. To account for issue (ii), we suggest a semi-supervised learning method for learning a sparse overcomplete basis from a corpus of possible terrain images. This is in lines of the Self-Taught Learning strategy proposed in [12]. Self-taught learning is a kind of transfer learning which is based on the assumption that any image consists of some basic ingredients like edges, textures, etc and thus learning a sparse overcomplete bases over a random set of images provide a powerful representation system to model any given image as a sparse linear combination of these bases. But our approach is not transfer learning and we use only aerial images of terrains where the UAV will fly. Later, we do supervised regression over this basis using the altitudes we have from a given training set.

Finally, we introduce a novel spatio-temporal MRF model to estimate the altitude of a patch in the image to the altitude of other patches in the same image and patches across images in the earlier time frames. The MRF model is later solved for the Maximum A Posteriori (MAP) estimate of the altitude.

The rest of the document is organized as follows: We begin with an overview of our motivation for using texture based techniques for altitude estimation in Section II, which precedes a discussion on computing the feature vectors. In Section III, we propose a probability model for the problem and optimization techniques for solving it. Section IV discusses our experiments and we conclude in section V.

II. Feature Vector

Given a video of altitude variations taken using a fixed focal length moving camera, humans will not have much of a difficulty in inferring the altitudes across frames. For example, we can easily say if an image was taken too close to the ground or far away or how much is the relative difference in altitudes between two given images. This is not only attributed to our prior knowledge about the environment, but also to our capability for using monocular cues such as texture variations, known object sizes, haze, focus/de-focus, etc in the inference. Texture gradients capture the distribution of the direction of the edges. It is a valuable source of depth cues and has been used quite effectively in papers like [2], [1] for 3D reconstruction.

When dealing with aerial images taken from a UAV, we have to face some more issues that cannot be adequately captured by texture variations alone. For example, most of the images are too noisy, have a variety of illumination differences, or are often blurred by the motion of the UAV. Moreover, aerial images lack structure compared to images taken on ground. For example, in ground images, we can probably assume that there is a ground plane, all objects stand on the ground, etc. But aerial images with top-down views look like random patches and application of conventional filters like autocorrelation filters, fourier/wavelets based filters, texture gradient filters like Nevatia-Babu, Laws masks filters, etc cannot effectively capture the texture variations to the respective altitude variations. Fig. 1 shows a few sample images that we will be working with in this paper. They were taken in our laboratory setting and the altitude at which each image was taken is also mentioned. Note the variation in texture as the altitude increases.

The motivation for our approach to solve this problem stems from the recent developments in sparse coding for compressed sensing [5], where information is encoded using a sparse overcomplete basis which effectively captures higher level information in the data, leading to a close to perfect reconstruction. The method was found to be robust to noise and relatively immune to illumination variations. In sparse coding, only a very few vectors from the basis set are needed to reconstruct a given image patch. Thus, a regression of this active set against altitudes provide a good representative relationship between altitude variations against the texture differences. Also, we would like to reduce the computational

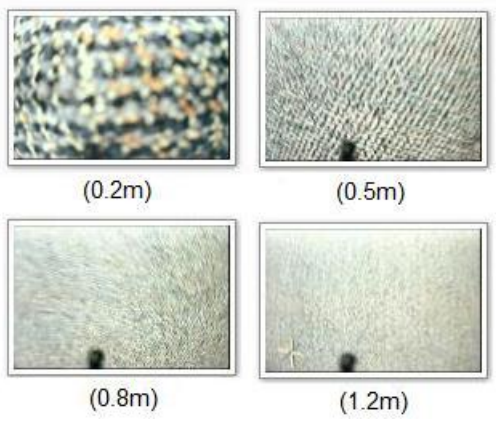

Fig. 1. Sample images of the top-down aerial views from an onboard camera of a UAV in the laboratory setting. The altitude at which each subimage was taken is also shown.

time for feature extraction and at the same time not compromising on the generality of the representation. We felt, conventional approaches using filter banks might not adhere to this requirement. For example, in [2], a filter bank of 510 dimensions is suggested. This increases the feature extraction time as well as the altitude prediction time. Fast turn-around time is a critical aspect in our application.

In [12], an efficient framework for learning such a sparse overcomplete basis is suggested, which is later used for object classification. Our problem is different from their approach, in that we do not learn basis from completely random images, but from aerial images of various terrains. Thus our philosophy is closer to a semi-supervised learning [17] setting, although we use their model to learn the basis. In order to prove the generality of our approach to arbitrary scenarios, we used random aerial images of various terrains from the internet to build our basis set. A few sample images that we used for this purpose are shown in Fig. 2.

Fig. 2. Random aerial images downloaded from the internet for learning the basis set.

Given a large corpus of image patches $I = \{I_1, ..., I_N\}$, each patch is vectorized as a k dimensional input vector y. The goal of sparse coding is to represent these vectors as a sparse approximate weighted linear combination of n basis vectors. That is, for the i^{th} input vector $y^i \epsilon \mathbf{R}^k$,

$$y^i \approx \sum_{j=1}^{n} b_j a_j^i = Ba^i \tag{1}$$

where $b_1, b_2, ..., b_n \epsilon \mathbf{R}^k$ are the basis vectors and $a^i \epsilon \mathbf{R}^n$ is a sparse vector of coefficients. Unlike similar methods such as PCA, the basis set B that we use here can be overcomplete

$(n > k)$, and can represent nonlinear features of y. To find the optimal B and a^i's, we solve the following optimization problem as formulated by [12]:

$$\min_{b,a} \sum_i \|y^i - \sum_j a_j^i b_j\|_2^2 \; + \; \beta\|a^i\|_1 \tag{2}$$
$$s.t. \; \|b_j\|_2 \; \leq 1, \; \forall \; j \; \epsilon \; \{1, ..., n\}$$

The optimization objective of (2) balances two terms: (i) the first quadratic term encourages each input y^i to be reconstructed well, as a weighted linear combination of the basis b_j with the corresponding weights given by the activations a_j^i, and (ii) it encourages the activations to have low L_1 norm, which encourages a^i to be sparse. The optimization problem is convex over each subset of variables a and b, but is not jointly convex. More specifically, the problem on activations a is an $L1$ constrained least squares problem, where as the one on the basis b is an $L2$ regularized least squares problem. The paper [7] provides an algorithm to solve these two sub-problems efficiently. Fig. 3 shows a basis set learnt using the above algorithm using random aerial images downloaded from the internet.

Once a sparse basis set $B \; \epsilon \; \Re^{k \times n}$ is obtained, we can construct a feature vector f for a given vectorized image patch p of dimension k by computing the activations on the basis that will produce this patch. That is,

$$\min_f \|p - \sum_j f_j b_j\|_2^2 + \beta\|f\|_1 \tag{3}$$

The feature vectors f from all the patches p in a given image are stacked up to form the feature vector set F that is used in the following sections.

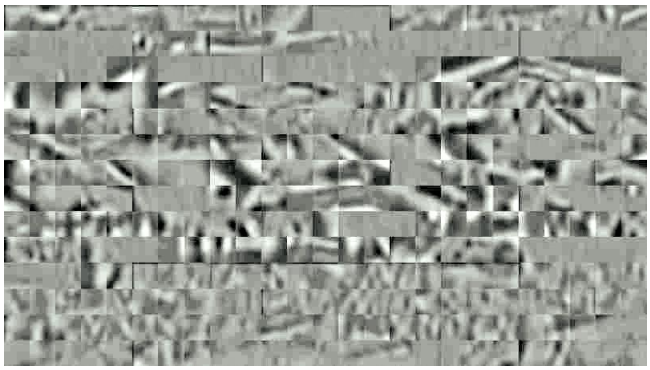

Fig. 3. 350 basis vectors each of size 16x16 learned using 50000 patches from random internet aerial images.

III. The Inference Model

Now that we have a comprehensive representation of the texture of an image as a linear combination of the basis, a supervised learning algorithm modeled on a spatio-temporal Gaussian Markov Random Field (MRF) [13] is deployed to estimate the posterior distribution of the altitude for every pixel block in the image. We model the posterior distribution of altitude d given the feature vectors set F and parameters σ and θ as:

$$P(d|F; \sigma, \theta) = \frac{1}{Z} exp(-E_{\sigma,\theta}(d, F)) \tag{4}$$

where

$$E_{\sigma,\theta}(d, F) = \sum_{i=1}^{n} \frac{(d^i - F_i'\theta)^2}{\sigma_a^2} + \sum_{j=1}^{T} \sum_{i=1}^{n} \frac{(d_{t-j}^i - d^i)^2}{\sigma_j^2}$$
$$+\beta \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} |d^i - d^j| \tag{5}$$

Here, Z is a normalization constant, $E_{\sigma,\theta}(d, F)$ defines the Gibbs energy function and F_i is the feature vector at pixel block i of the image computed using (3). The first term in (5) models the raw altitude at the pixel block i in terms of feature vectors F_i through the regressor θ. As it is apparent, we use a linear relationship between F_i and d. Linear least squares regression over the training set is used to find the vector θ.

We assume that the altitude of the UAV will not change abruptly, but rather smoothly. Thus the altitude predicted from one image frame to the next frame should not have drastic deviations in the predicted altitudes. This is formulated by the second term in (5), which constrains the altitude at pixel block i at time t, to be smooth to the pixel block i of the image frame at time $t - j$. The third term in Eq. (5) captures the relationship between the altitude predicted at pixel block i to all other blocks in the same image. Since we are working with aerial images with top-down view, and since we assume that the ground is flat, there will be only a single altitude value for the whole image. Thus we would like to constrain the altitudes predicted from different pixel blocks of one frame to be as equal as possible. That is, if d_i and d_j represent the altitudes at any two pixel blocks in the same frame, then $(d_i - d_j)$ should be as close to zero as possible. Fig. 4 shows a schema of the MRF setup we are assuming in our computations.

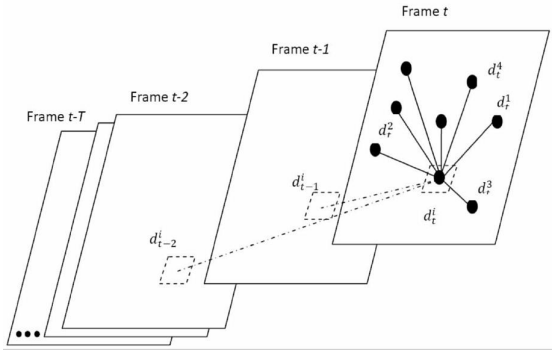

Fig. 4. The MRF model we assume. The schema represents the dependency of the altitude at pixel block i to other pixel blocks as modeled by the spatio-temporal MRF. In the schema we show frames at times t, $t-1$,...,$t-T$. Each solid black dot d_t^i shows the altitude at patch i at time t. The solid lines show the equality constraints we impose for the altitude predictions and dotted lines capture the smoothness across frames.

The parameter σ_a ensures smoothness of our raw predictions of altitude. From our experiments, we found that there

was heteroskedasticity in the data, i.e., the variance of the prediction error varied with the texture of the image. In order to accomodate for this variation, a separate σ_a parameter was estimated for each image by projecting the individual feature vectors F_i in the image patches to a regression hyperplane S that captures the expected error. That is, a hyperplane S is first estimated by linear least squares over $E[(d^i - F_i\theta)^2] = S^T F_i$ from the training data. Later, given the feature vectors F_i from a test image, we calculate $\sigma_a = E[\|S^T F_i\|]$, i.e. by averaging over the individual feature vectors projected on to S.

A different strategy was used to find the σ_j parameters. These parameters capture the variance of the altitude estimates across frames. We assume that the altitude variations are smooth and also they are consistent across all the patches in a single image. We estimate the hyperplane parameter S_j from the Eigen analysis of $(d^i - d^i_{t-j})$, where d_i is the true altitude from the training data and d^i_{t-j} represents the estimated altitude of ith pixel block at the j^{th} earlier time step from the current step t. The covariance matrix is formed by stacking up the predicted error across patches in a column and the variations across frames in rows. Finally, $\sigma_j = \|S_j^T d^i_{t-j}\|$ is computed. Note that in the Eigen analysis, we assume that the error in d^i_{t-j} is Normal distributed and $d^i = E(d^i_{t-j})$ for varying j. The parameter β in the model controls how much the predicted altitudes in a frame d^i and d^j are close and is computed experimentally through cross validation.

Once the parameters are estimated from the training data, given a test frame, we use the Maximum A Posteriori (MAP) over d to estimate the altitude d over the entire image. The MAP problem can be stated as follows.

$$\min_d \; E_{\sigma,\theta}(d, F). \tag{6}$$

To solve this problem, first (5) is reformulated as an L1 and L2 constrained optimization problem through the following linear algebra transformations. Let D is an $n \times 1$ vector built by stacking up all the altitude values d over the entire image. Let Y is also an $n \times 1$ vector created by stacking up the corresponding $F_i'\theta$ values. Further, let E and I be $n^2 \times n$ matrices defined as follows:

$$E = \begin{pmatrix} 1_n & 0 & 0 & 0 & 0 & 0 \\ 0 & 1_n & 0 & 0 & 0 & 0 \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ 0 & 0 & 0 & 0 & 0 & 1_n \end{pmatrix}, \; I = \begin{pmatrix} I_{n\times n} \\ I_{n\times n} \\ . \\ . \\ . \\ I_{n\times n} \end{pmatrix},$$

where 1_n is a vector of n ones and $I_{n\times n}$ denotes an $n \times n$ identity matrix. Thus (6) in terms of (5) can be rewritten as

$$\min_D \frac{\|D - Y)\|_2^2}{\sigma_a^2} + \sum_{j=1}^{T} \frac{\|D - D_j\|_2^2}{\sigma_j^2} + \beta\|(E - I)D\|_1 \tag{7}$$

Assuming $X = (E - I)D$, this becomes a standard L1 and L2 constrained quadratic optimization problem, which can be solved efficiently by a modified version of the Feature-Sign-Search algorithm depicted in [7]. The basic idea of this algorithm is that if we can infer the correct signs of the elements in the vector X, then the L1 minimization problem can be converted to a standard L2 minimization problem which can be solved efficiently using conventional optimization techniques.

IV. Experiments and Results

A. UAV System

The experiments in this paper were conducted in a laboratory setting using a Blade CX2 helicopter from E-Flite [8]. Fig. 5 shows the helicopter with the position and orientation of the video camera. We used an Eyecam 2.4 GHz Color Micro Wireless Video Camera System [6] onboard that captures NTSC video at 250K pixels and transmits it using a frequency set in the 2.5 GHz spectrum. To track the position of the helicopter during test flights, 6 Vicon high-resolution MX-40 grayscale cameras [16] with a resolution of 2352×1728 pixels was deployed along the perimeter of the lab as shown in Fig. 6, giving an experimental area of $4.5m \times 4.5m \times 2m$. The Vicon cameras send information every 7ms through high speed Ethernet cables to a central router, which collates the data, providing it to a PC running the ViconIQ software recording the true altitude.

Fig. 5. The Blade CX2 helicopter that was used for the experiments. It is a coaxial helicopter with a rotor diameter of 34.5 cms, a height of 18.3 cms and weighs approximately 220g (with battery).

B. Learning Setup

To learn the basis set, we collected approximately 250 random aerial images (640×480) from the internet. Each image was converted to gray scale, which preceds segregating the images into patches of size 10×10, vectorizing them, and later using (2) to build the basis. Fig. 7 shows the plot of the mean absolute error in altitude prediction against various choices of the basis set size. Trading-off between the accuracy of altitude prediction (as seen from Fig. 7) and the computational time, we fixed on 200 basis vectors for our experiments. The parameters of the MRF model were then

Fig. 6. The experimental setup for tracking the position of the helicopter. 6 high-resolution Vicon cameras (only three shown in the picture) were placed along the perimeter of the lab that tracks the reflective markers on the helicopter (seen with bright dots in the picture) determining the position.

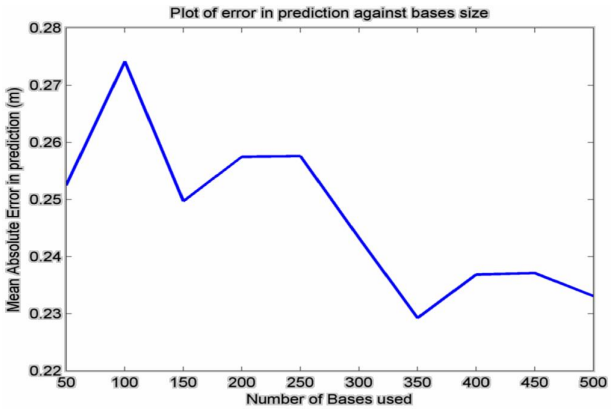

Fig. 7. A plot of the mean absolute error of prediction (y axis in m) against the number of basis vectors used (x axis). Each basis vector is of size 100×1.

$Experiment$	$Min\ Error(m)$	$Max\ Error(m)$	$Mean\ Error(m)$	$Std.\ Dev.$
Without boxes	0.0	1.13	0.35	0.26
With boxes	0.0	1.56	0.48	0.36

TABLE I

A SUMMARY OF THE VARIOUS FLIGHT EXPERIMENTS.

estimated by regressing the basis on patches from a training set of 200 images taken using the camera on the helicopter. These images were of the kind shown in Fig. 1. Finally the model was tested on pre-recorded lab flight sessions by minimizing (7). Each test image in the video sequence was first applied with Wiener filters to account for the motion blur and later convoluted with Gaussian filters for smoothing. To improve the speed of prediction, we used only the middle 100×100 section of each image.

All the algorithms were implemented in Matlab. Testing the framework took less than 0.5 seconds per image on a PC with a 2Ghz Pentium processor and 2GB RAM. Fig. 8 plots the true and predicted altitudes for various experimental UAV flight sessions. As is seen from the plots, predicted altitude gives a very good approximation to the true altitude most of the times. The prediction error increases as true altitude of the UAV goes high. This is expected, as at higher altitudes, the ground is seen as almost textureless. Thus our algorithm is mostly applicable to low altitude situations, like landing or low altitude flight, unless we have a more powerful camera onboard.

To evaluate the robustness of our algorithm with non-flat surfaces and across varying ground textures, we placed boxes of varying textures in the experimental area and the helicopter made to fly over them. Fig. 9 shows the results of this experiment. As is seen, our algorithm performs well in finding the altitude variations due to the presence of the obstacles. We repeated the experiments with boxes of high illumination and with non-textured surfaces, the outputs of which are shown in Fig. 10. As expected, in this case, the algorithm gets confused between textureless surfaces to textures at high altitudes and performs poorly. A summary of the various experiments showing the minimum, maximum, mean and the standard deviations of the errrors respectively is given in Table I.

V. Conclusion and Future Work

In this paper, we investigated the possibility of predicting the altitude of a UAV from ground looking image sequences taken from a single onboard camera. We found that sparse coding can effectively capture the texture in an image. Supervised regression using this sparse basis against a training set of altitudes provided a good prediction setup. Later, a spatio-temporal MRF was modeled and its MAP estimate with respect to the altitude was computed. The effectiveness of our approach was substantiated through laboratory experiments. We found that as the altitude of the UAV goes over a certain height, the images become textureless (which depends on the specific environment though) and our algorithm performs poorly. Thus our mechanism is most suited for situations where the UAV flies at low altitudes and at low speeds. Also, since the prediction is based on the texture variations in the image, the mechanism performs poorly when the ground surface is textureless. A way to improve our algorithm in such a situation will be to incorporate information from other sensors like ultrasonic/infra-red as priors to the MRF model. Another direction to carry forward this work would be to extend our framework to forward-looking onboard cameras. Such a setup could enable not only altitude estimation but also visual navigation and obstacle avoidance. These are topics for future research.

VI. Acknowledgments

This material is based upon work supported in part by, the U.S. Army Research Laboratory and the U.S. Army Research Office under contract #911NF-08-1-0463 (Proposal 55111-CI), and the National Science Foundation through grants #CNS-324864, #CNS-0420836, #IIP-0443945, #IIP-0726109, #CNS-0708344, and #CNS-0821474.

References

[1] A. Saxena, M. Sun, and A. Ng. 3-d scene structure from a single still image. *In ICCV workshop on 3D Representation for Recognition (3dRR-07)*, 2007.

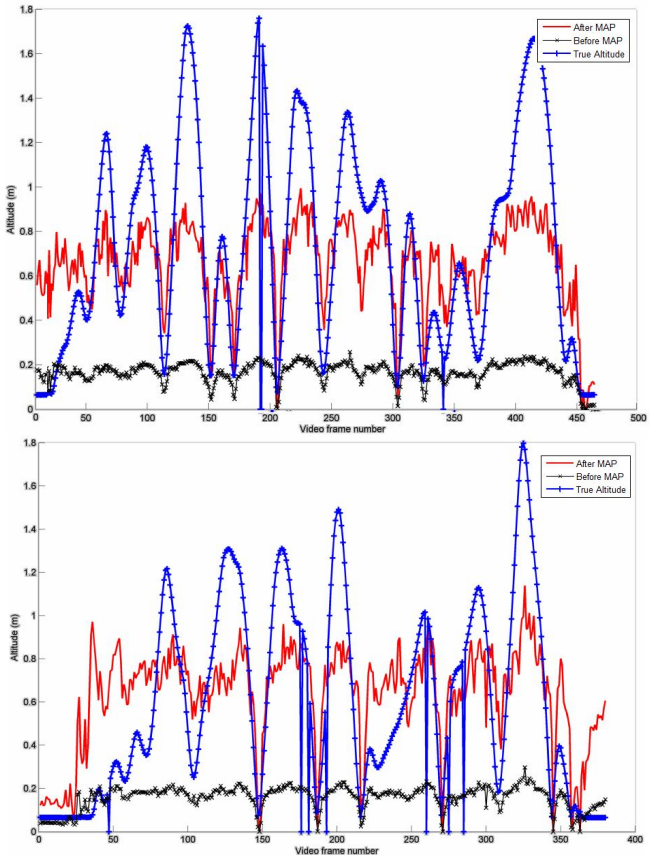

Fig. 8. Prediction of altitude for various flight sessions. The X axis shows the video frame number and Y axis gives the altitude. The blue curve shows the true altitude, the black curve shows the raw output using the linear prediction before MAP and red curve shows the final output after the MAP estimation. The helicopter was made to fly up and down which accounts for the variations in the blue curve.

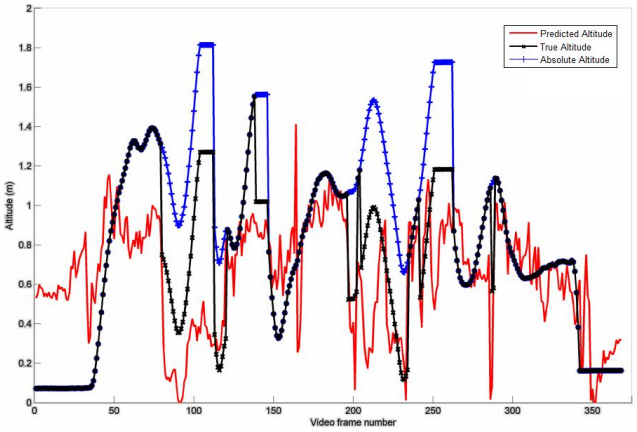

Fig. 9. The prediction accuracy when there were altitude variations of the ground. We kept a few boxes on the floor and the helicopter was made to fly over it. The plot shows the absolute altitude of the helicopter from the ground, the true altitude of the helicopter subtracting off the height of the boxes (whenever it flew over it) and the predicted altitude. As is seen, our altitude prediction is very close to the green curve most of the times.

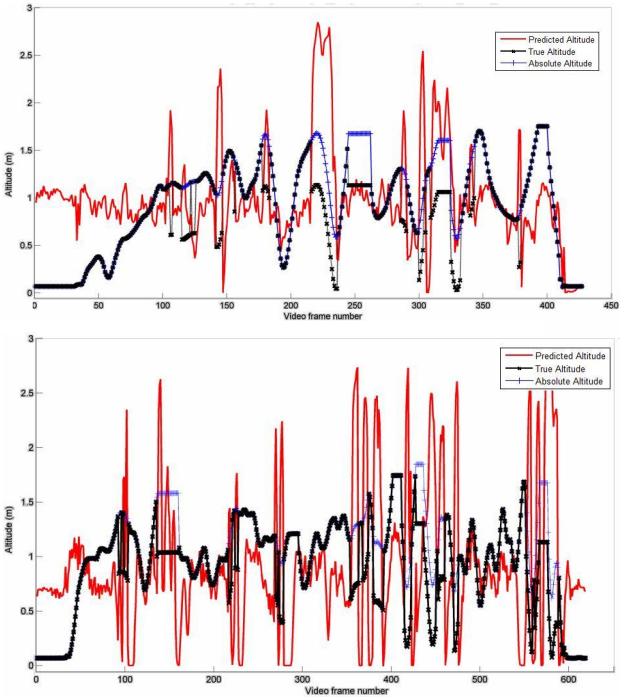

Fig. 10. Predition accuracy for illumination (former) and non textured surfaces. The plot shows the absolute altitude from ground, the true altitude from the boxes and the predicted altitude. Note the spikes in the predicted altitude where the algorithm gets confused with non-textured surfaces and high altitude surfaces.

[2] A. Saxena, S. Chung, and A. Ng. Learning depth from single monocular images. *NIPS*, 2005.

[3] A. Saxena, S. Chung, and A. Ng. 3-d depth reconstruction from a single still image. *IJCV*, Aug 2007.

[4] C. Sharp, O. Shakernia, and S. Sastry. A vision system for landing an unmanned aerial vehicle. *In Proceedings of IEEE International Conference on Robotics and Automation*, 2001.

[5] D. Donoho. Compressed sensing. *Technical Report, Department of Statistics, Stanford University*, 2004.

[6] Draganfly Innovations Inc. *Eyecam 2.4GHz Wireless Ultra Miniature Camera System: Installation Guide and Manual*, 2008.

[7] H. Lee, A. Battle, R. Rajat, and A. Ng. Efficient sparse coding algorithms. *NIPS 19*, 2007.

[8] Horizon Hobby, Inc. *Blade CX2 Manual*, 2006.

[9] M. Sanfourche, G. Besnerais, and S. Foliguet. Height estimation using ariel side looking image sequences. *ISPRS Archivs, Vol.XXXIV, Part3/W8, Munich*, 2003.

[10] P. Abbeel, A. Coates, M. Quigley, and A. Ng. An application of reinforcement learning to aerobatic helicopter flight. *In NIPS 19*, 2007.

[11] P. Garcia-Padro, G. Sukhatme, and J. Montgomery. Towards vision-based safe landing for an autonomous helicopter. *Robotics and Autonomous Systems*, 2000.

[12] R. Raina, A. Battle, H. Lee, B. Packer, and A. Ng. Self-taught learning: Transfer learning from unlabeled data. *In Proceedings of the 24th International Conference on Machine Learning*, 2007.

[13] S. Geman, and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6:721-741*, 1984.

[14] S. Hrabar, and G. Sukhatme. Omnidirectional vision for an autonomous helicopter. *In Proceedings of IEEE International Conference on Robotics and Automation*, 2004.

[15] S. Saripalli, J. Montgomery, and G. Sukhatme. Vision based autonomous landing of an unmanned aerial vehicle. *IEEE International Conference on Robotics and Automation*, 2002.

[16] Vicon. *Vicon MX Systems*, October 2006.

[17] X. Zhu. Semi-supervised learning literature survey. *Technical Report 1530, Department of Computer Sciences, University of Wisconsin, Madison*, 2005.