Technical Report
MECSE-21-2005

Heaven and Earth: How to tell the difference

Terry Cornall and Greg Egan

Heaven and Earth: How to tell the difference.

Terry Cornall[1], Greg Egan[2]

[1] *CTIE, Electrical and Computer Systems Eng., Monash University, Wellington Rd, Clayton, Victoria, 3168, Australia*
[2] *CTIE, Electrical and Computer Systems Eng., Monash University, Wellington Rd, Clayton, Victoria, 3168, Australia*

Summary: This paper discusses two methods for segmenting an image into sky and ground and three metrics for measuring the 'fitness' of the segmentation for the purpose of measuring the horizon angle and position. The reason for doing this is to use the horizon as a reference for stabilising the flight of an Unmanned Air Vehicle. (UAV). Results of applying the methods to video captured by a UAV are presented.

Keywords: UAV, unmanned aircraft, sky, ground, image processing, computer vision, horizon detection and tracking.

Introduction

Why do the authors want to classify parts of the image as sky and ground? To find the horizon. What use is that? The displacement and angle of the horizon in the video frame can inform us about the attitude of the camera and hence of the aircraft. This is ongoing work being done by the authors as well as other groups. [1] [9][10][11][12][13][14][15][16]

For our purposes, we need an algorithm that can segment a video frame into ground and sky parts. We need also to be able to measure how good the segmentation is. Because the equipment is to me flown in a small UAV it need low computational intensity and not need a lot of memory in order to keep the computing device as small and light as possible. The method also needs to be fast, as a frame processing rate of at least 5 frames per second is required.

For the authors' purposes, a good segmentation has clearly defined sky and ground classes with little or no overlap and a well defined interface, the horizon. A circularly shaped view is required for our work because it makes the measurement of the horizon angle simpler, given the average coordinates (centroids) of the classes. [1]

It is almost an everyday observation that the sky is bluer and/or brighter than the ground. On closer inspection this is not always true, of course. There are grey skies and blue skies with white clouds, and red sunsets and so on. Asked if the ground is ever blue, most of us would answer 'Not often'. Anyone experienced with walking in Australia's eucalyptus forests and staring at the mountains on the far horizon might answer differently, (the area around Sydney on NSW isn't called the Blue Mountains for nothing), and there are blue gravel roads and blue tar roads and blue lakes and other exceptions. Nonetheless, it is often the case that the sky is bluer and/or brighter than the ground. Other researchers have also considered this question and discussions can be found in [4][5][6][7][8].

Even grey or white skies have a profile such that the blue component is greater in the sky than in the ground, as evidenced by *Fig. 1* and *Fig. 2*. The profiles are drawn from the RGB values of pixels in a column up the centre of the image, so the sudden jump in the profiles show the position of the horizon in each case. The first figure of a clear blue sky shows, not surprisingly, that the blue value is higher than the red and green values above the horizon and the red dotted line in the blue profile shows the average blue value is below the blue value for all the pixels in the sample that are above the horizon.

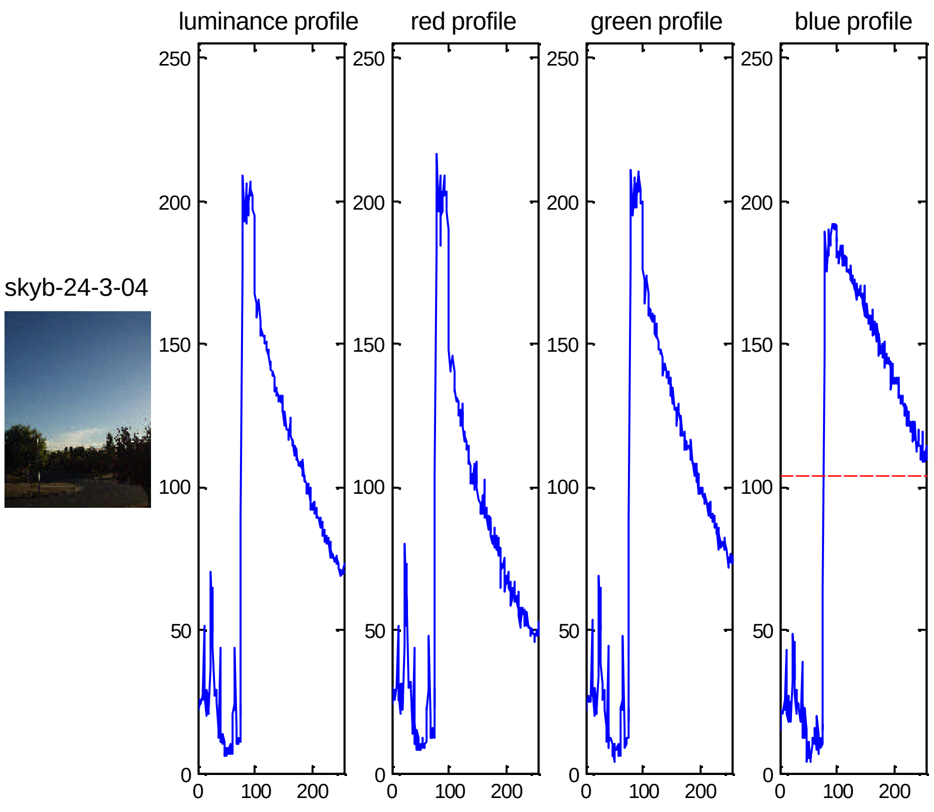

Fig. 1 Clear sky and RGB profiles

Fig. 2 on the other hand shows that the RGB profiles are all quite similar for an overcast sky. It does confirm that the blue component is above average above the horizon, indicating that the sky is merely brighter than the ground though not perceptually bluer. However, if we simply ignore the red and green components in both cases, we can discriminate between ground and sky by using the average blue value as a threshold.

It isn't always that simple, of course. Both these figures have relatively dark ground sections and this is not always the case. Note also the values in the blue profile on the right edge of *Fig. 1* showing a drop of blueness with increasing elevation. This can be a problem. Strangely enough, *Fig. 2* doesn't show the same drop, implying that lightly overcast skies are a better candidate than clear blue ones for this discriminator.

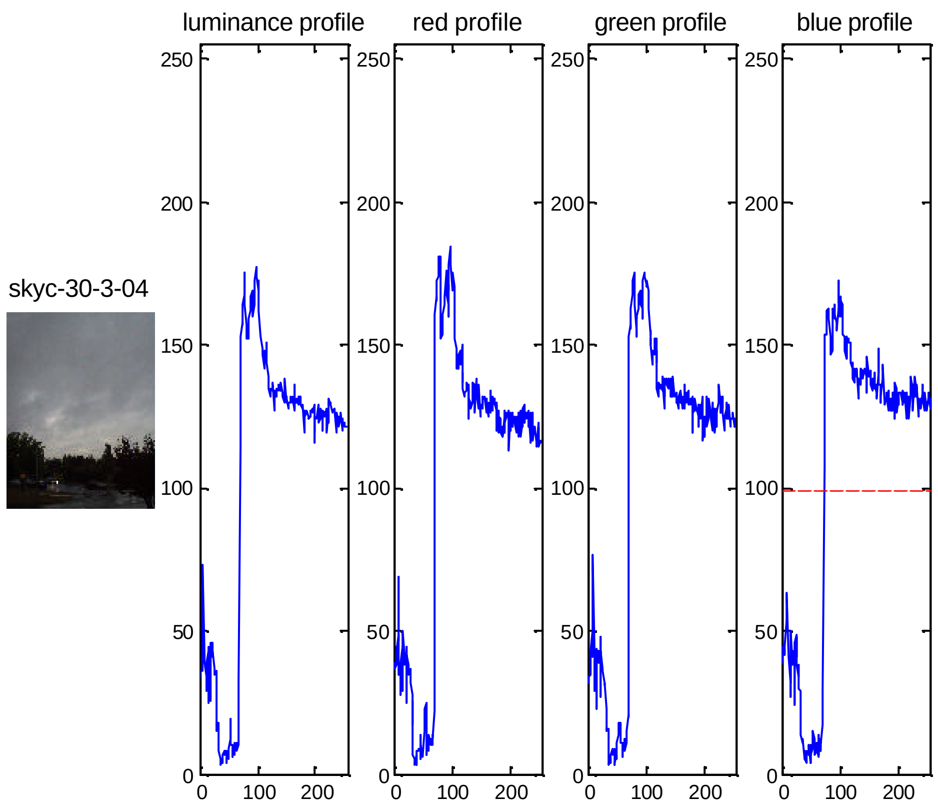

Fig. 2 Cloudy sky and RGB profiles

There are factors such as white or bright objects on the ground that complicate things, such as snow. There are even blue objects, such as lakes and seas, on the ‘ground’. Sometimes the skies are almost as dark as the ground, such as during thunderstorms. Breaks in the cloud can even allow sunlight through to make the ground locally brighter than the sky. These factors must all be taken into account because it is fairly clear that any decision made on a mistaken assumption of the position of the ground and sky could easily be disastrous for a flying vehicle. It may not in the end be possible to always decide on the basis of a visible-light image just where the sky and ground are, but on the other hand, it often is. What is needed is a simple method that works more often than not, and most importantly, a measurable value or values that allows us to decide if the decision is trustworthy.

Method one: Otsu thresholding using blue only.

Otsu’s algorithm works on a one-dimensional histogram to produce a threshold value that segments the histogram values into two classes in a manner that is claimed to be optimal for class separation. The algorithm first counts the number of pixels in the image that have a particular value of blue, usually using a coarse quantiser so there aren’t too many levels in the histogram. Then, for as many levels are represented in the histogram it calculates the value of a metric d_b^2 that would be generated if that level of the blue were used as a threshold to classify all the pixels with lower blue levels as belonging to class one, and the rest as being in class two.

$$d_b^2 = (u_2 - u_1)^2 (w_1 w_2)$$

Eqn. 1

u_1 and u_2 are the mean values of blue for each class and w_1 and w_2 are the number of pixels in or the probability of belonging to the class. The first term maximises class separation and the second term tends to equalise the size of the classes because $w_1 w_2$ is a maximum when $w_1 = w_2$. The metric is called d_b^2 because it is a measure of the variance between the classes. Over all, maximising d_b^2 maximises the difference between the blue values of the pixels belonging to the different classes. The threshold that generates the greatest value of d_b^2 is the level that creates classes with the largest interclass variance [2]. *Fig. 4* shows an example of the threshold chosen by the Otsu method for a given histogram and it can be clearly seen that the threshold has been automatically chosen to fall in the space between two large clusters in the values of blue.

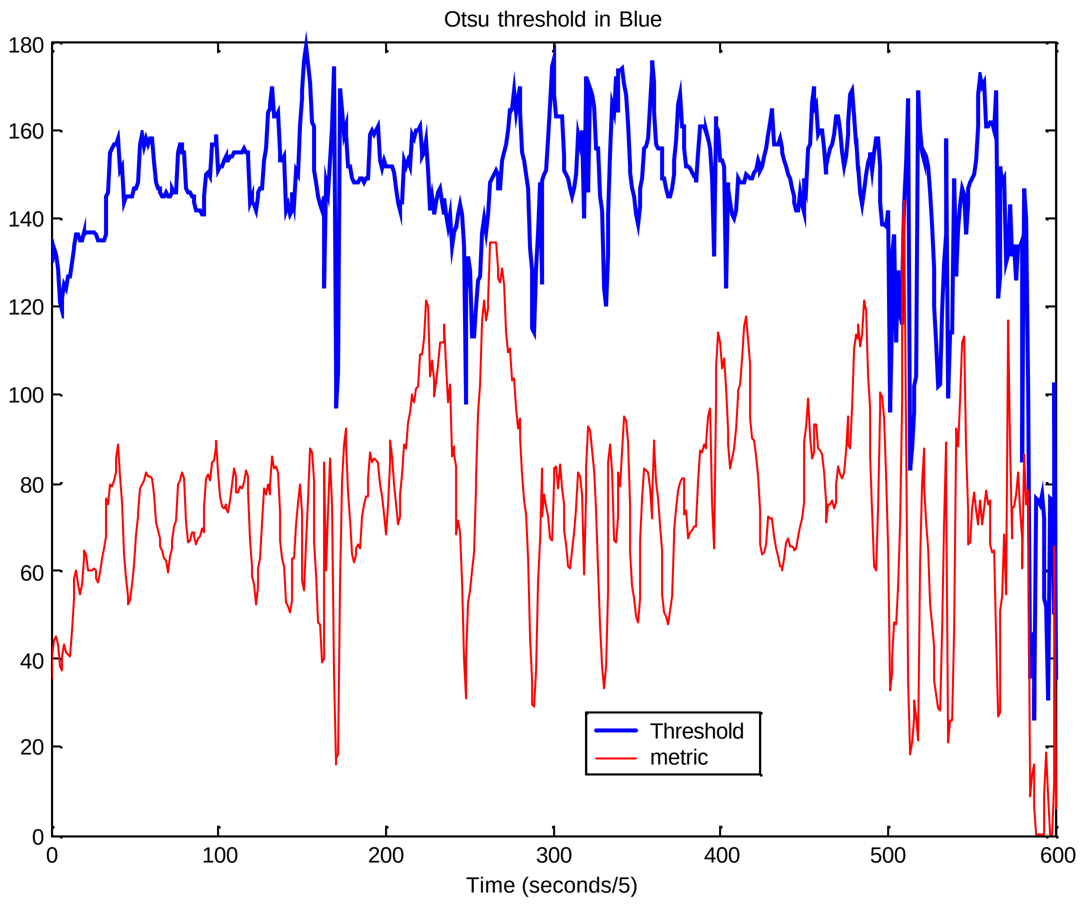

Fig. 3 Threshold for binarisation from Otsu method and the d_b^2 metric

Fig. 3 shows the results of applying Otsu's histogram analysis method to the blue component of the aerial video footage in the Grampians2 video. The software used in this case was Matlab, making use of the graythresh.m function that implements Otsu's algorithm. The resulting threshold for segmentation is shown as well as the d_b^2 metric (scaled to be visible in the same graph) that the algorithm uses. The graph shows that the threshold, for that particular video sequence, is centred on about a blue level 150 out of a maximum of 255, with deviations usually limited to +/- 20 or so. By observing the segmented video, the authors have determined that d_b^2 is a good indicator of unreliable segmentation, marked in the graph by the sudden drops in the metric below about 40 with this scaling. The video of the segmentation of the Grampians2 video sequence using Otsu's method can be seen in the file "*sky-ground-otsu-only-grampians2.avi*" *Fig. 4* shows a snapshot from that video showing the original image, the segmented image, the blue histogram and the threshold determined by Otsu's algorithm as well as a graph of the d_b^2 metric for all the processed frames up to that point in time. In this sequence, where the sky is clear and pale blue near the horizon and the ground is well lit without too much in the way of lightly colored objects on it, the segmentation using just the histogram analysis works reasonably well. The histogram of the blue component of the image is clearly bi-modal with distinct clusters, in this frame and in the majority of other frames. Most of the untrustworthy frames are due to video telemetry corruption, or when the horizon goes out of view. Note that there is a lake in the view in this sequence however, and it can clearly be seen from the accompanying video how that causes misclassification errors which are not indicated by the d_b^2 metric.

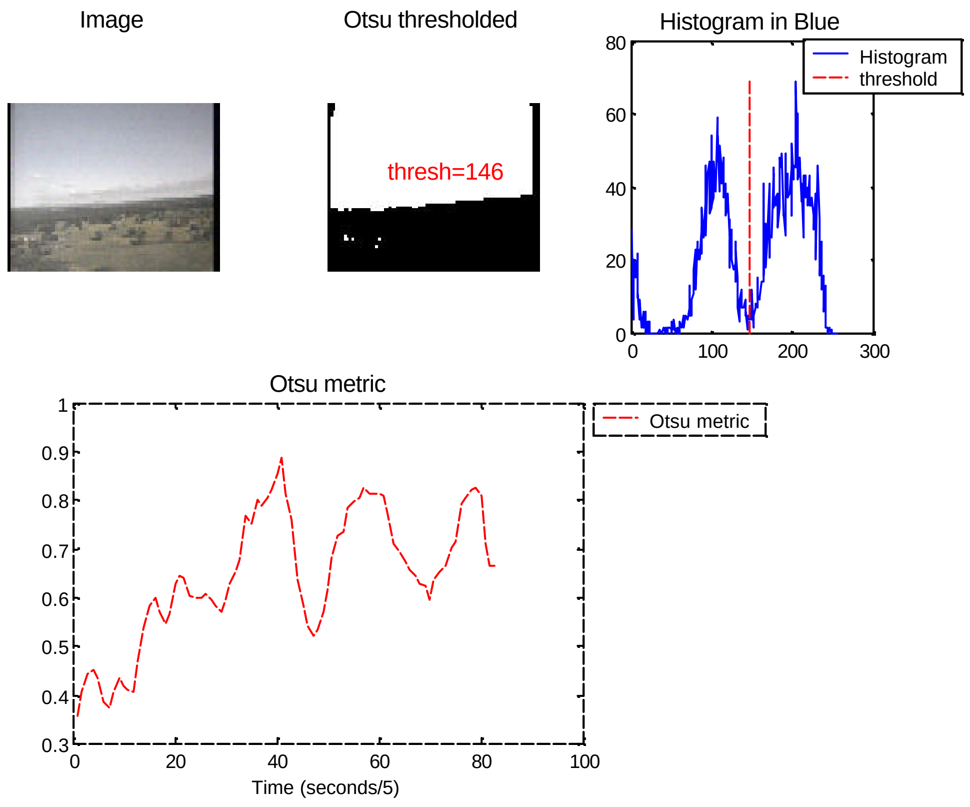

Fig. 4 Snapshot of Grampians2 video segmented using Otsu's method

Fig. 5 shows a snapshot from a different sequence called Varms101. In this case the day was much more clouded and it was later in the day with a bright horizon. *Fig. 5* is singled out as one example of how the segmentation can fail but be detected. Note that the value of the d_b^2 metric (this time scaled to have a maximum value of about 1) is very low at the time of this snapshot, indicating that the segmentation cannot be relied upon. The video "*sky-ground-otsu-only-varms101.avi*" shows the entire segmented sequence using Otsu's method.

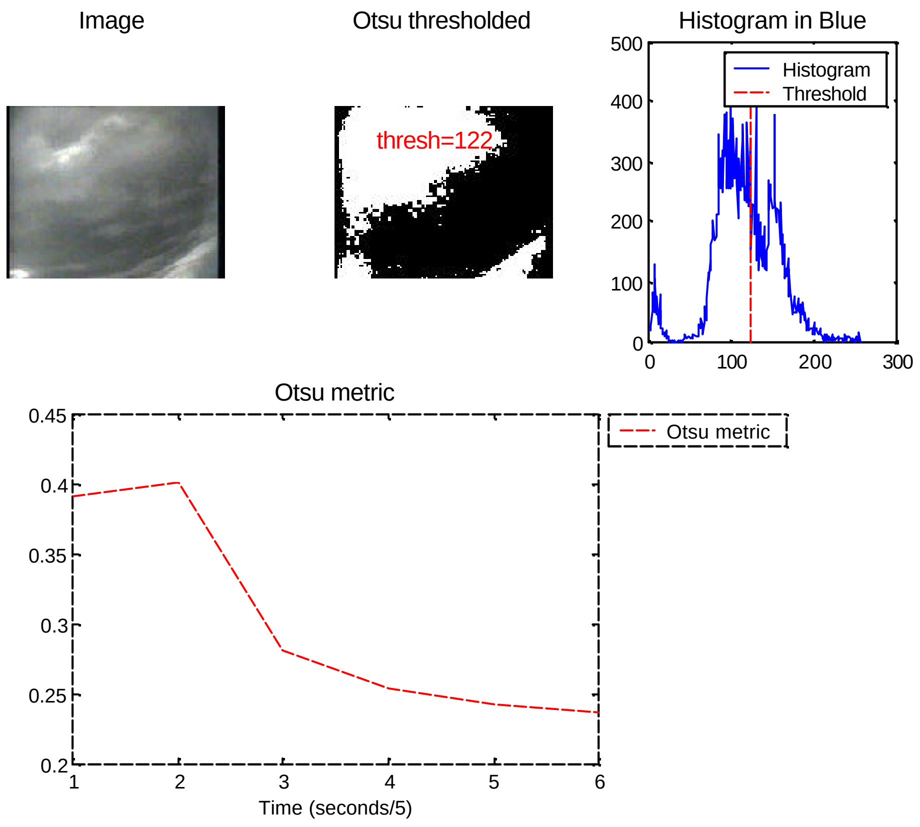

Fig. 5 Snapshot of cloud misclassified as ground by Otsu's method. Varms101 video.

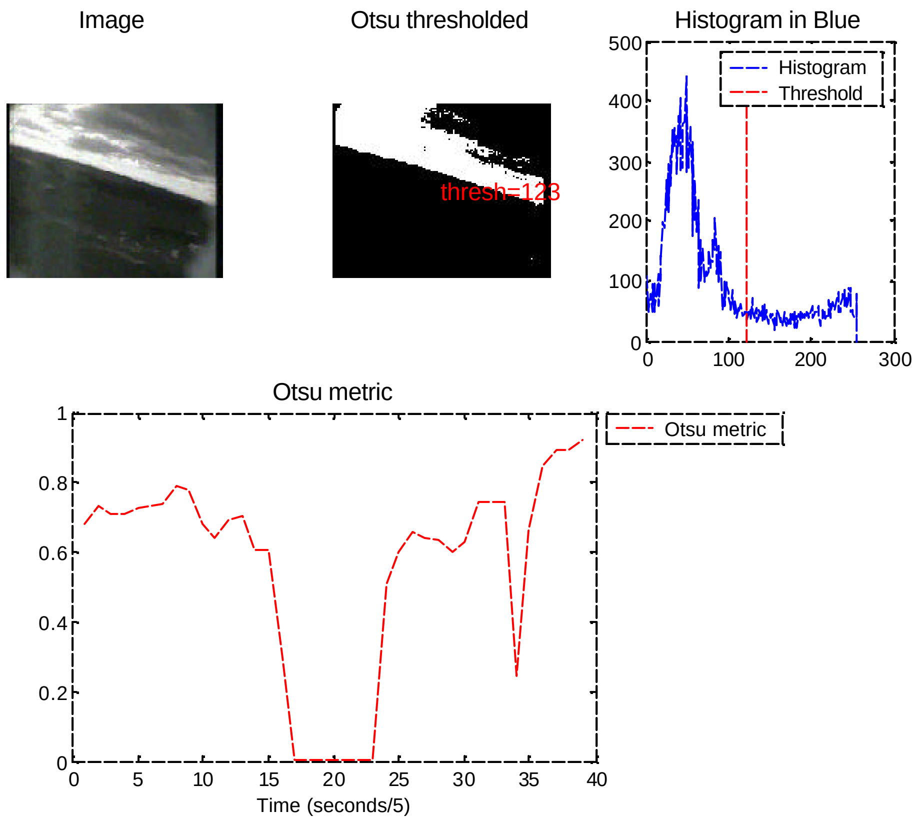

Fig. 6 Another bad example, with metric score above mean. Varms101 video.

Unfortunately, in this sequence there are other examples of failed classification that are not well indicated by the d_b^2 metric. *Fig. 6* shows one such where the histogram shows that there is not a clear distinction between sky and ground and the threshold selected by Otsu's algorithm has resulted in cloud misclassified as ground, but with a high confidence score of about 0.9. This is near the mean for the whole sequence. This indicates that the d_b^2 metric by itself is not an adequate indicator of reliability. This is not surprising, as the d_b^2 metric does not take the spatial grouping of the pixels into account at all, merely their blue value, so in marginal cases such as presented in the Varms101 video, misclassification will occur undetected.

Metric2.5

This metric (so called because it is a minor change from the second metric the authors considered) is measured using the statistics of the classified pixels in a manner similar to d_b^2 but operating on their spatial coordinates not heir blue values. It combines the mean coordinates u_1 and u_2 of the two classes with the populations n_1 and n_2 of each class and the radius of the circular viewport, R. If a rectangular viewport is being used, the method behind metric2.5 still applies but there would need to be adjustments made to account for the asymmetries introduced by the corners of the view for different angles of the horizon. With a circular viewport, these adjustments are not required. As the authors are using a circular viewport to facilitate the horizon angle calculations anyway, for reasons explained in [1], this does not impose any extra computational burden. The formula for metric2.5 is very similar to that used by Otsu's algorithm and the method could be seen as an extension of that work. The major differences are that it is applied to a two-dimensional variable, the spatial coordinate, and the product of population terms is de-emphasised:

$$m = \frac{(u_1 - u_2)^2 (n_1 n_2)^{1/3}}{3R^3}$$

Eqn. 2

The exponent of 1/3 applied to the product of the populations is to reduce its weight compared to the separation of the classes given by $(u_1 - u_2)^2$. The $3R^3$ in the denominator is a normalising factor to bring the value down to near 1. The effect of the population product is to increase the metric in favour of classes that have similar sized populations as the product is a maximum when the populations are the same, because $n_1 + n_2$ is a constant. The $(u_1 - u_2)^2$ term containing the class average coordinates, or centroids, increases as the class separation increases, which favours segmentations that have the classes well separated in space. This could lead to segmentations where there is one large class with a centrally located average coordinate and one small class with its average coordinate right on the circumference, but the term containing the populations discourages this by decreasing as the class sizes become dissimilar. The authors are concerned by the exponential terms as they lead to increased computational load and it would be reasonable to experiment with alternative means of altering the weights of each term. For a constant viewport size, the $3R^3$ is a constant.

Fig. 7 shows the comparison for metric2.5 and the d_b^2 metric, for the varms101 sequence, as well as their product. The file "*metric2p5-on-otsu-varms101.avi*" shows the segmented sequence.

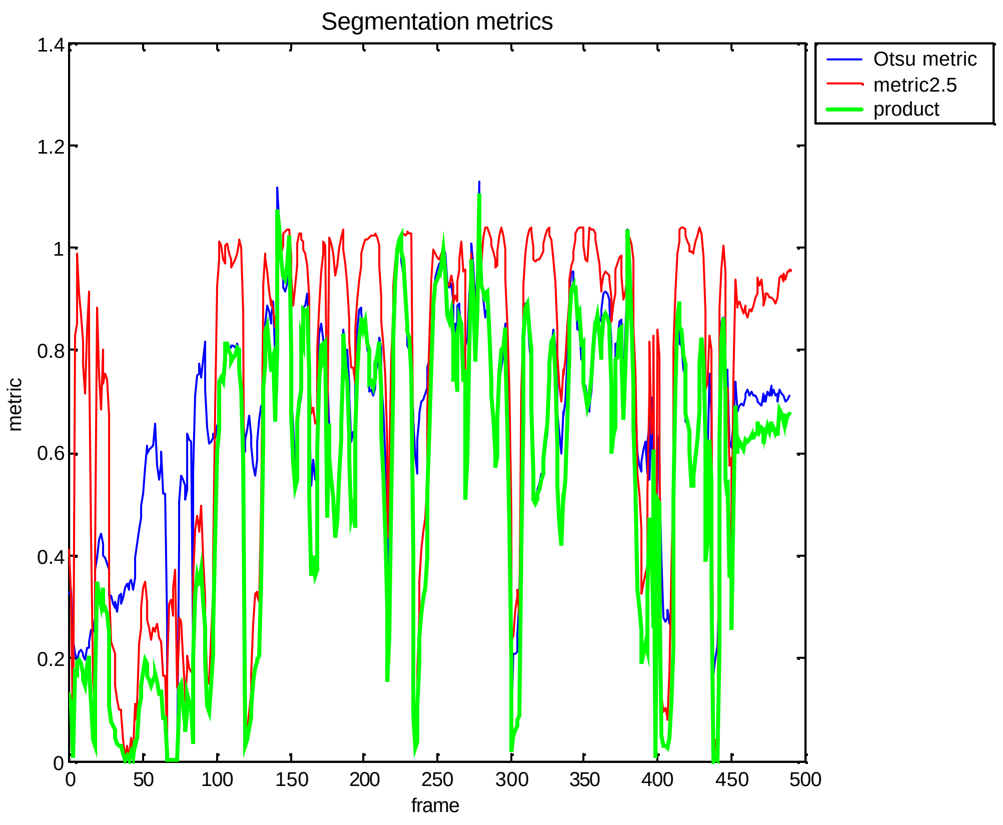

Fig. 7 Metric2.5 applied to Otsu thresholded video

Again, it is possible to find frames in this sequence where misclassified pixels are not indicated by a low value for metric2.5. *Fig. 8* shows a frame from the beginning of the sequence as an example. The classes are distinct and similarly sized so metric2.5 has a relatively high value, even though the pixels classified as ground are in fact due to dark clouds. Note however that in this case the d_b^2 metric has a relatively low value. The combined use of these two metrics will be better than either alone, using their product. (This can be seen in the file "*metric2p5-on-otsu-product-grampians2-badmarked.avi*" which marks segmented frames as 'bad' if the product of the metrics falls below a pre-determined threshold). It can be seen from *Fig. 7* that the two metrics tend to agree on disastrously untrustworthy classification such as that in the region of frame 50 and that metric 2.5 will pick up errors such as that near frames 120 and 230 that the Otsu metric may miss.

Fig. 9 shows the value and images for frame 120 where metric2.5 has a low value as it has detected that the classes are not clustered nicely into two distinct areas and the Otsu metric has a reasonably large value indicating that the histogram is bi-modal. This means that there is good contrast between the areas of the image that have a high blue value and the areas that don't. This is one case where this contrast in color value does not indicate the sky and ground parts of an image, but it can be rejected on the basis of the spatial arrangement of the classes.

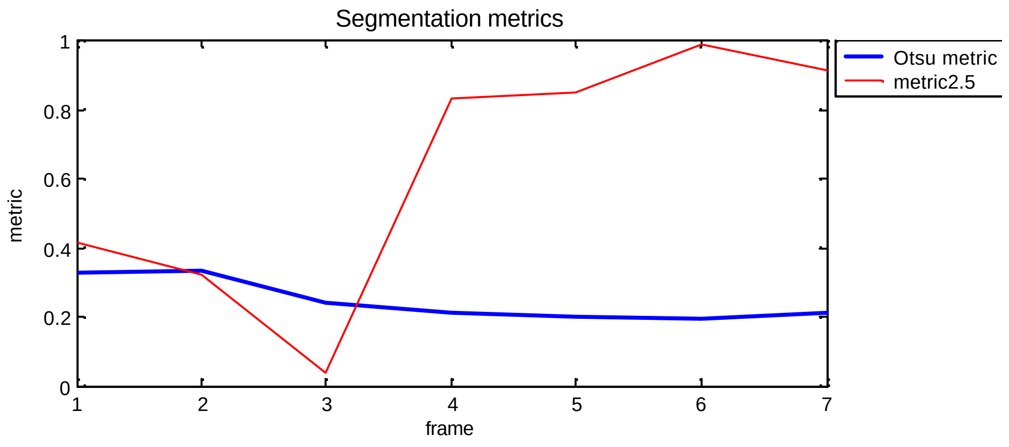

Fig. 8 Metric2.5 showing high value for misclassified frame.

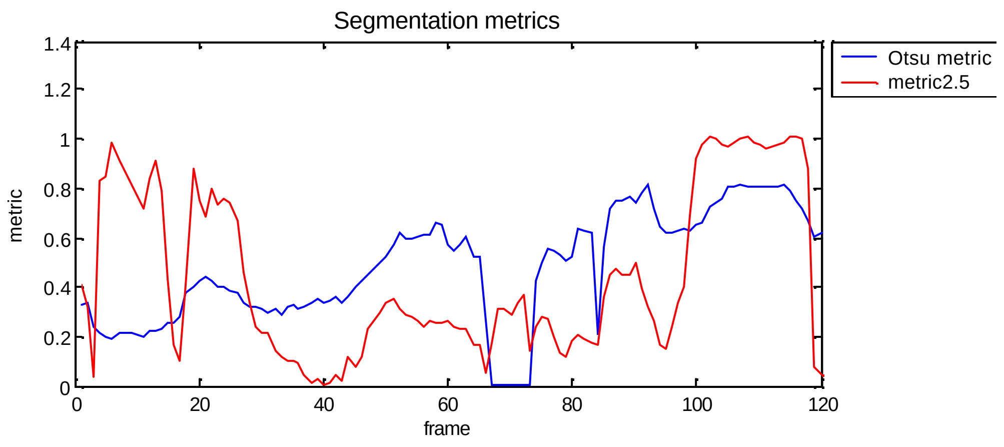

Fig. 9 Example where metric2.5 has low value and Otsu metric has high

Method 2 Kmeans clustering using blue, spatial position and texture.

A problem with the Otsu and similar color thresholding methods is that they do not take the spatial arrangement of the pixels into account. This can and does lead to pixels that are just above the upper ground 'blueness' threshold being classified as sky despite being much closer to the average ground spatial coordinates that they are to the average sky spatial coordinates. Whilst this can sometimes be detected by methods such as the metric2.5 discussed above, it would be better to avoid the problem in the first place. A method that also used the spatial Euclidean proximity of a pixel to other pixels that have been classified as either ground or sky would naturally produce more coherent segmentation classes. One such is the k-means classifier, and the implementation used in this work was from a book due to Sing-Tze Bow [3] and written as a Matlab function by Frank Dellaert of Georgia Tech. [17]

Essentially the k-means method starts with the approximate mean positions of k classes (where k=2 in this case) It then classifies pixels as belonging to the class whose mean position (centroid) is closest to the position of the pixel. Having done that, each class's centroid is re-calculated and the classification of pixels is done again. This procedure is repeated until the class centroids change by less than some small amount from one iteration to the next. For the next frame, it is reasonable to initially use the centroids as found for the last frame. This reduces the number of iterations compared to random initial centroids, but might cause errors due to persistent local maxima.

A disadvantage of the k-means classifier is that it is more computationally intensive than the histogram analysis method of Otsu. This is because it must consider not one attribute of the pixel, its blue component measurement but three; blue, x position and y position. Worse, it must consider these three measurements for every pixel whereas Otsu's method considered the *histogram* of the blue component, which has many fewer datapoints than there are pixels. Thus k-means applied to an NxM sized image requires O(IxKxNxM) operations where I is the number of iterations needed and K is the number of classes, whereas Otsu needed only O(HxH) where H is the number of values that blue can take, which is independent of the image size. Quantising the blue value so that the histogram has only 16 bins, for example, reduces the number of operations to order 16 squared. For k-means, reducing the spatial dimensions of the image to the point where it requires a similar number of operations means a great loss of resolution.

Fig. 10 shows an example where the k-means classifier (first image on second row) has a better result of correctly segmenting out sky and ground pixels than Otsu's classifier (second image first row) on a challenging frame with dark skies and ground. This is not to say that the k-means classifier always works perfectly, but observation shows that it works at least as well as the Otsu method and often better. The video "*kmeans-bluemeans-and-otsu-varms101.avi*" shows the entire sequence.

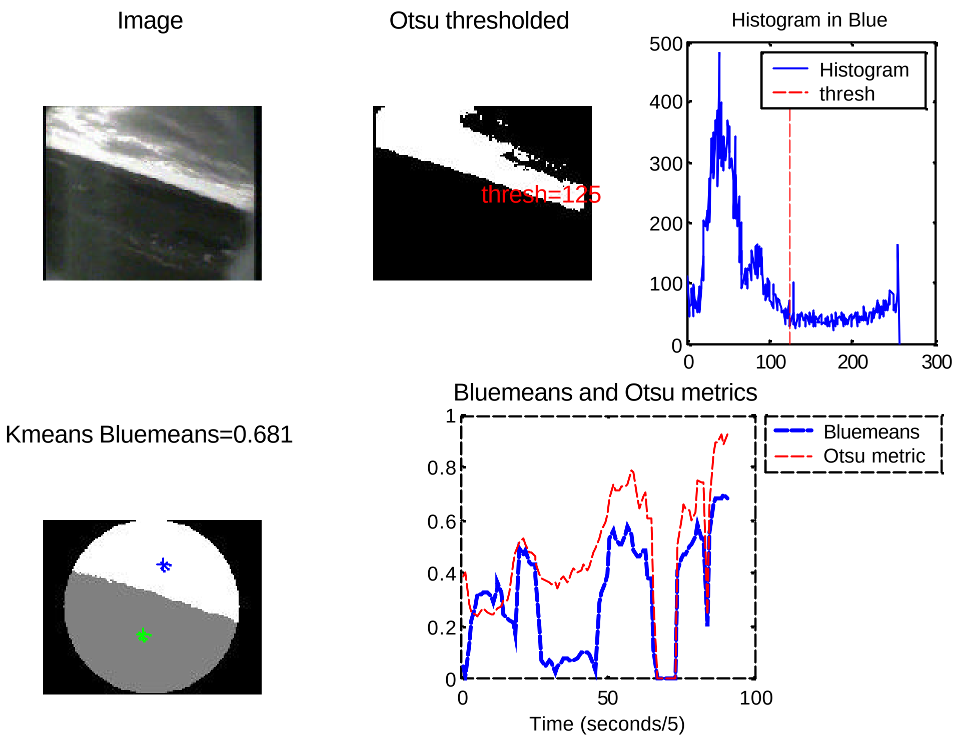

Fig. 10 K-means classifier compared to Otsu's, showing better result.

The k-means segmentation method can produce a spurious segmentation that satisfies the spatial criteria used for metric2.5 even when operating on a low contrast image, as shown in *Fig. 11* where the segmented image (first on second row) shows two distinct classes that have apparently nothing to do with the original image. Thus criteria not relying solely on spatial arrangement of the classes are required to measure the 'confidence', or lack of it, that should be associated with the segmentation. We must keep in mind that the d_b^2 metric is not automatically available in this process, but a related measure that is simple to calculate uses the difference in the mean blue component of the resulting classes. The mean values in each dimension (blue, x, y) are be accumulated as part of the algorithm. Should the absolute value of this difference be small, then low confidence should be placed on the segmentation. It may in fact be a 'good' segmentation, in that the position of the horizon is in accordance with what a human viewer might arrive at but if the original image was of very low contrast and the resulting mean blue values of the classes are too similar, then it is best to err on the side of caution and treat the result as being dubious. Another reason for the k-means method to arrive at spurious classification is if the image contains a view of only ground or sky, or even noise from corrupted telemetry. The classifier is designed to produce two classes using the spatial arrangement of the blue component of the image and it will do so, but in these circumstances the interface between these classes will not represent the position of the horizon, no matter how well defined and spatially separated the classes are. On the other hand, if the classification is correct and trustworthy, then there should be considerable difference in the mean blue value of the classes and so a metric based on that difference will usefully indicate how well or otherwise the algorithm worked. The equation used for bluemeans is simply:

$$bm = \frac{1.5|u_1 - u_2|}{255}$$

Eqn. 3

where u_1 and u_2 are the mean values of the blue component of each class and the constants are to give the measure a convenient value near 1. This gives a measure that is a minimum of zero when the means are the same and approaches one when they are most different. It is simple to calculate and judges the worth of the segmentation based on how well the classes are separated in the blue dimension. It is assumed that they are spatially disjoint because the k-means method naturally does that. If it proves necessary another measure like metric2.5 can be used to quantify how well the classes are arranged spatially. So if a segmentation arrives at two classes that are well separated in the blue and spatial dimensions, this may be a good result. Otherwise it definitely isn't. Another look at *Fig. 11*, this time at the graph labelled Bluemeans and Otsu metrics shows a very low value (compared to 1) for both metrics for this result, indicating lack of trustworthiness.

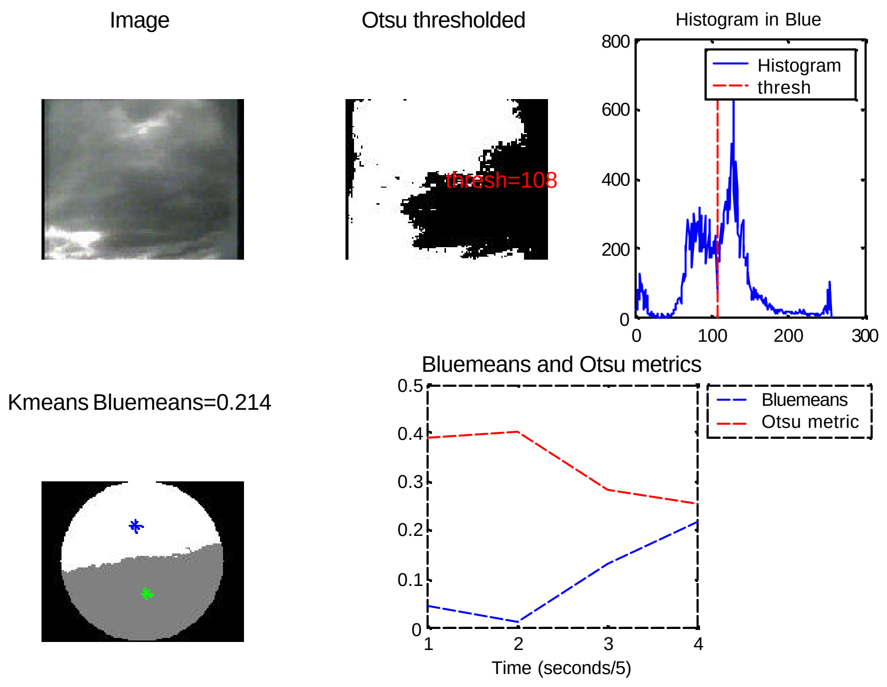

Fig. 11 K-means clustering showing a poor classification result

Fig. 12 shows the results of using the bluemeans metric on the k-means clustering algorithm applied to the whole Varms101 video, which was quite a challenging sequence because of dark clouds. The d_b^2 metric that Otsu uses is supplied for comparison. Both measures have been scaled to be visible in the same graph and approach a value of 1 for a good result and 0 for a poor result.

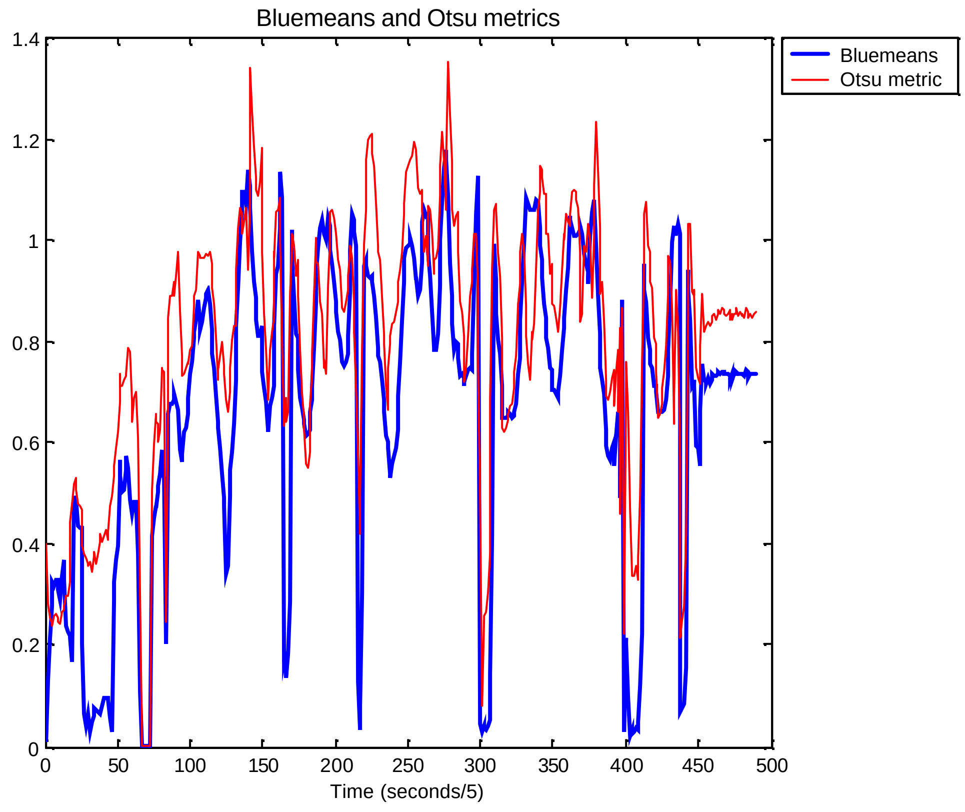

Fig. 12 Bluemeans fitness metric applied to K-means clustering using blue and spatial coordinates. Otsu d_b^2 metric supplied for comparison.

What is most significant are the changes in each measure, particularly where each sharply dips below their respective means indicating an untrustworthy segmentation result. ***Fig. 12*** shows that the two measures are largely in accord regarding untrustworthiness as they both have sharp dips at the same place, and these dips align with the decisions that the authors have made based on observing the segmentation results. The dips in the bluemeans measure are more distinct than those in the d_b^2 metric. The entire sequence is shown in the file "*kmeans-bluemeans-and-otsu-varms101.avi*"
The bluemeans metric is not a perfect indicator however. There are cases where a segmentation does not have a good spatial arrangement and yet does have a relatively high confidence score according to the bluemeans indicator. This implies, previous discussion notwithstanding, that it will be necessary to test the spatial arrangement of the classes.

Conclusion

One cannot always simply use the relative values of the blue component of an RGB visible-light image to determine the position of the ground and the sky, but one often can and it would be useful to know how to do so robustly and simply. Otsu thresholding works well for clear or light overcast skies and isn't very expensive to calculate, using the histogram of the blue component of the RGB image. The d_b^2 metric used in Otsu's algorithm chooses a segmentation threshold so that the decided classification of sky and ground has the classes maximally separated in the value of the blue component of the classes respective pixels. A low value of d_b^2 can indicate bad segmentation in

some circumstances such as low contrast in the image, but fails to detect it in others because it doesn't take the spatial arrangement of classes into account.
Metric2.5, which is an extension of Otsu's d_b^2 metric that takes the spatial arrangement of the classes into account is a useful adjunct to the d_b^2 metric as it can detect segmentation results that do not accord with the model of spatially well separated non-overlapping sky and ground classes.

K-means clustering using blue and spatial coordinates of the pixels works better than Otsu's method in that it produces results where the sky and ground classes are more spatially disjoint, but is much more computationally expensive. The Bluemeans metric is useful for quantifying the confidence of the K-means segmentation, which can produce incorrect but distinct and non-overlapping classes in images of low contrast.

References

[1] Terry Cornall and Greg Egan. "Measuring Horizon Angle from Video on a Small Unmanned Air Vehicle" ICARA 2004, Palmeston Nth N.Z. Dec 13-15.

[2] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," IEEE Transactions on Systems, Man, and Cybernetics, vol. 9, no. 1, pp. 62-66, 1979.

[3] Sing-Tze Bow, "Pattern Recognition and Image Preprocessing" 2nd ed Publisher :New York : Marcel Dekker, 2002 **ISBN:** 0-8247-0659-5

[4] D. Jud, D. MacAdam and D. Wyzsecki, "Spectral distribution of typical daylight as a function of correlated color temperature", Journal of the Optical Society of America, 1964, volume 54, number 8 pages 1031 to 1040

[5] "Spatial distribution of daylight - CIE standard overcast sky and clear sky", CIE (International Commission on Illumination) technical report, 1996, number S003

[6] A. J. Preetham, Peter Shirley and Brian Smits, "A practical analytic model for daylight", Proceedings of the 26th annual conference on Computer Graphics and Interactive Techniques, 1999, pages 91 to 100, isbn = 0-201-48560-5

[7] Jiebo Luo and Stephen Etz, "A Physics-Motivated Approach to Detecting Sky in Photographs", Proceedings of the 16th International Conference on Pattern Recognition, August 2002, Quebec City, number 26, pages 155 to 158

[8] Tomoyuki Nishita, Takao Sirai, Katsumi Tadamura and Eihachiro Nakamae, "Display of the Earth Taking into Account Atmospheric Scattering", Proceedings of the 20th annual conference on Computer Graphics and Interactive Techniques, 1993, pages 175 to 182

[9] S. Ettinger, P. Ifju and M. C. Nechyba, "Vision-Guided Flight for Micro Air Vehicles", http://mil.ufl.edu/~nechyba/mav/index.html#vision1, 2002, September.

[10] S. Ettinger, M.~Nechyba, P.~Ifju and M.~Waszak, "Vision-guided flight stability and control for micro air vehicles", International Conference on Intelligent Robots and Systems (IEEE/RSJ). Sept, 2002, volume 3, number 30, pages 2134 to 2140,

[11] Luc Fety, Michel Terre and Xavier Noreve, "Image Processing for the Detection of the Horizon and Device for the Implemention Thereof", Thompson TRT Defense, 1991, United States Patent, number = 5,214,720,

[12] G. Stange, S. Stowe, J.S. Chahl and A.~Massaro}, "Anisotropic imaging in the dragonfly median ocellus: a matched filter for horizon detection.", Journal of Comparative Physiology A.", 2002", VOLUME 188, PAGES 455 to 467

[13] S. Todorovic, M. C. Nechyba and P. G. Ifju, "Sky/Ground Modeling for Autonomous MAV Flight", Proc. IEEE Int. Conf. on Robotics and Automation., May, 2003, Tiawan.

[14] Centre for MAV Research, University of Florida, "Horizon Detection and Tracking", http://www.mil.ufl.edu/mav/research/vision/horizontracking, 2002, September.

[15] G. Barrows, J. S. Chahl and M. V. Srinivasan, "Biomimetic Visual Sensing and Flight Control", Bristol UAV Conference, April, 2002, Bristol, UK.

[16] T. Netter and N. Franceschini, "A Robotic Aircraft that Follows Terrain using a Neuromorphic Eye", (IEEE/RSJ) International Conference on Robots and Systems", October, 2002 Lausanne.

[17] Frank Dellaert. College of Computing, Georgia Tech. “kmeans.m” Matlab function http://www.cc.gatech.edu/~dellaert/html/software.html http://www.cc.gatech.edu/~dellaert/