Available at www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

Pattern Recognition Letters 24 (2003) 2345–2356

Pattern Recognition Letters

www.elsevier.com/locate/patrec

Evaluation of global image thresholding for change detection

Paul L. Rosin *, Efstathios Ioannidis

Department of Computer Science, Cardiff University, Newport Road, Cardiff CF24 3XF, UK

Received 3 January 2003

Abstract

The objective of this paper is to develop an approach for efficiently and quantitatively evaluating thresholding algorithms for change detection in a surveillance environment. Previous evaluation in the literature has either been subjective or small scale, in part due to the difficulties and/or the time and effort involved in determining appropriate ground truth. In comparison, our automated approach enables us to carry out a more thorough evaluation, and we test the performance of eight different thresholding algorithms using more than 4000 images with two different texture environments.

Keywords: Threshold selection; Surveillance; Change detection; Performance evaluation; Ground truth

1. Introduction

The rapid increase in use of video cameras for remote surveillance has been a significant development in the security industry. Unfortunately the task of distinguishing abnormal events from everyday activity requires the continued vigilance of security personnel. Since many video surveillance installations employ a large number of cameras, this task places a heavy burden on the operator which inevitably results in loss of concentration and therefore poor performance. One solution is to automate change detection and object tracking. This information can then be made available to alert the operator in the event that an illegal or abnormal activity is detected. Change detection is the very first step required towards a solution to this problem. Frame differencing and/or background subtraction followed by thresholding is a commonly used method for change detection if the images are co-registered. A surveillance camera that records at 30 frames per second does not have to deal with rapid changes in the lighting and illumination conditions. Therefore we can often assume a nearly constant environment between adjacent frames. Otherwise there are many algorithms of varying sophistication in the literature (Stauffer and Grimson, 1999; Ren et al., 2003) that can be used to generate and maintain background images, and so simple image differencing can be still be employed.

Thresholding is a fundamental technique applied in many image processing applications. Many relatively simple and computationally

* Corresponding author. Tel.: +44-2920875585; fax: +44-2920874598.
E-mail address: paul.rosin@cs.cf.ac.uk (P.L. Rosin).

0167-8655/03/$ - see front matter
doi:10.1016/S0167-8655(03)00060-6

effective algorithms have been developed and used for change detection in video sequences. However very little has been done on evaluating these algorithms using long sequences of realistic data, and unfortunately there is no widely accepted evaluation test framework. Consequently, when new thresholding algorithms are presented in the literature they often lack the extensive testing that is necessary to enable easy comparison with previous solutions.

The main problem of evaluation over a very large sample space is that it takes up a lot of time, which makes the process impractical in many cases. As a rule of thumb one can use the time relation (Forstner, 1996)

$$\text{theory:implementation:testing} = 1{:}10{:}100.$$

This does not include the cost of getting the ground truth which can be even more laborious. In the last few years there has been considerable interest in techniques for evaluation of computer vision algorithms (Bowyer and Phillips, 1998; MVA, 1997). Unfortunately, the majority require ground truth, which makes large scale evaluation impractical. Some alternatives exist; for instance, Levine and Nazif (1985) suggested assessing the quality of image segmentation without reference to a ground truth image, but by measuring for each region its internal homogeneity and its contrast along its boundaries. In a similar vein, Kitchen and Rosenfeld (1981) assessed the quality of thresholded edge maps using edge continuity and thinness. While appealing, the problem is that these criteria do not always reflect good results (Venkatesh and Rosin, 1995).

In most cases ground truth is essential for performing a quantitative analysis of an algorithm's results. There are three main approaches to generating ground truth. The first uses synthetic data; example applications are ellipse fitting (Fitzgibbon et al., 1999), edge detection (Venkatesh and Kitchen, 1992), corner detection (Zheng et al., 1999), and optic flow (Barron et al., 1994). This method enables ground truth to be easily provided—the problem is that the synthetic data will probably not faithfully represent the full range of real data. Alternatively, real image data can be manually annotated, e.g. to mark edge and non-edge pixels (Dougherty and Bowyer, 1998), or to specify an ideal image segmentation (Hoover et al., 1996). Now we have the opposite problem: the image data is good, but the ground truth is dubious. Since manual mark-up is tedious and time consuming large volumes of ground truthed data are likely to have errors. In addition, the process has become subjective, and different annotators often give different ground truth (Kadonaga and Abe, 1995). A third approach avoids explicitly determining a ground truth dataset, and relies instead on evaluating the algorithms' outputs by a human panel (Heath et al., 1997). Two disadvantages are the time consuming nature of the exercise (more images need to be viewed), and the difficulty in incorporating additional algorithms into the evaluation results at a later date (unless the same panel is reconvened).

The following give some examples of the evaluation of thresholding algorithms in the literature. Zhao et al. (2000) experimented on applying seven different thresholding algorithms to several blueprint images. Their testing, although methodical, was limited; they were subjective, the results were produced only by observation of the resulting images. Since no automatic method was used to quantify the performance of the algorithms, the evaluation is not easily repeatable.

Leung and Lam (1996) proposed a methodology for evaluation of iterative thresholding algorithms. Most of the measures relate to the stability of the iterative process which restricts their generality and usefulness. Actual evaluation of the correctness of the results required ground truth which they obtained by restricting the majority of the testing to synthetic data. Only a few real life images were tested and the evaluation of the correctness of the thresholding algorithms applied to them was subjective.

In (Sahoo et al., 1988) previous work done by Weszka (1978) and Fu and Mui (1981) was updated, providing a well-presented taxonomy of different thresholding algorithms with limited testing of results (only three images). In addition to region uniformity they used a shape measurement (based on the image gradient) to classify the performance of the algorithms. It is interesting to note that optimising thresholding according

to these two measures produced very different results.

In an extensive set of tests Sezgin and Sankur (submitted for publication) compared 41 thresholding algorithms. In addition to measuring region uniformity and pixel misclassification they measured shape distortion. The binarised image was compared to a ground truth image or to an edge map of the original image. Like Levine and Kitchen's work the latter approach avoids the need for ground truth, but again its validity is questionable.

While the above approaches performed low level evaluation of thresholding algorithms, the work by Trier and Jain (1995) took a goal directed approach, and assessed 11 thresholding algorithms by their effectiveness for document image analysis. As with so many evaluation techniques this required ground truth to be collected, which they described as "extremely tedious", thus limiting its wider application.

In this paper we propose a framework for testing thresholding algorithms at a low level as well as for use in a specific application, namely change detection. The approach is scalable as it can easily cope with a large number of thresholding algorithms and, more importantly, many images. The importance of testing with large amounts of data was recently highlighted by Forbes and Draper (2000) who showed that otherwise quite misleading evaluation results could be produced. In this case we obtained results on eight different thresholding algorithms which were applied to seven different sequences totalling over 4000 images. The following section goes through the thresholding algorithms that we put on test. In Section 3 the proposed evaluation framework is analysed, and Section 4 presents the results of the experiments.

2. Image thresholding algorithms

There are many thresholding algorithms published in the literature, and selecting an appropriate one can be a difficult task. The problem is that different algorithms typically produce different results since they make different assumptions about the image content. For instance, some require the two classes to have not too dissimilar sizes, others model the class distributions as Normals, etc. Our choice was based on algorithms that are widely known or offer an alternative method of thresholding calculation. In addition, they did not require parameters, and were straightforward to implement thereby ensuring that our coding was likely to be accurate.

- The Ridler and Calvard (1978) algorithm uses an iterative clustering approach. An initial estimate of the threshold is made (e.g. mean image intensity). Pixels above and below the threshold are assigned to the white and black classes respectively. The threshold is iteratively re-estimated as the mean of the two class means.
- The Tsai (1985) algorithm determines the threshold so that the first three moments of the input image are preserved in the output image.
- The Otsu (1979) algorithm is based on discriminant analysis and uses the zeroth- and the first-order cumulative moments of the histogram for calculating the value of the thresholding level.
- The Kapur et al. (1985) algorithm uses the entropy of the image. It considers the thresholding image as two classes of events with each class characterised by a pdf. The method then maximises the sum of the entropy of the two pdfs to converge to a single threshold value.
- Two approaches as described by Parker (1996) were implemented that use the entropy of the intensity histogram according to two definitions (by Huang and Wang (1995) and Yager (1979)) of fuzziness.
- The Rosin (2001) algorithm fits a straight line from the peak of the intensity histogram to the last non-empty bin. The point of maximum deviation between the line and the histogram curve will usually be located at a corner which is selected as the threshold value.
- The Normal fitting algorithm fits a Normal distribution to the intensity histogram. Since the Normal is likely to be asymmetrically truncated an iterative fitting method (Press et al., 1988) was used.

3. Evaluation methodology

To evaluate the thresholding algorithms for change detection we need to set up an environment that will enable us to obtain the necessary testing sequences on which the algorithms will be applied. Ground truth data is essential to provide a reference point to test the correctness of the thresholding results. Finally we will need to apply analytical methods to quantify and classify the results of the comparison.

3.1. Evaluation methodology

Our approach to evaluation is based on the following principles. Performance will be evaluated quantitatively by comparing the thresholded result against a ground truth image which specifies the true areas of change. Since we want to be able to apply the approach to large amounts of data this requires the ground truth data to be generated automatically, since otherwise it becomes impractically slow and laborious. One solution is to use synthetic data, but since this is rarely truly indicative of real-life data the evaluation task would become compromised. Our key idea is to keep with real images—as realistic as possible—but to control the contents of the scene in some way without substantially altering the nature of the image. The idea then is to distinguish areas of change by some properties such as number of regions, colour, size, shape, position, etc. These properties need to be sufficient to enable the moving object to be automatically found, but will not be used by the thresholding algorithms themselves (so that the evaluation is not compromised).

Having generated a set of ground truthed data this easily enables a larger secondary set of image data which shares the same ground truth to be generated. For instance, the images could be modified by adding noise, changing contrast, simulating occlusion, etc. Alternatively, once detected, the moving object could be cut out and pasted into new video sequences. Although the resulting images would not be totally realistic (e.g. shadows and illumination of the foreground object would be wrong) the data could still be useful for testing purposes.

3.2. Establishing the testing data and ground truth

As an example, in this paper we present the results of tracking a single moving object of a specific shape; namely a ball in a room. We used the ball for convenience as its projection in the image is constrained to a circle, allowing us to use a simple circle detection algorithm to establish the ground truth. Thus, the high level information that specified that change corresponded to a single round object enabled the ground truth to be automatically generated. Thresholding was applied to the difference image, and was therefore unbiased by the set-up for ground truth collection. Using two alternate floor textures (one of a single colour and one multi-coloured) seven sequences totalling over 4000 images (of size 768×576) were taken. Figs. 1 and 2 show a sample image, the difference image, and the results of thresholding.

A large amount of noise and blockiness was noticed in the initial experiments, causing difficulties in establishing the ground truth. This arose mainly from noise introduced during the digitisation process, and the high level of JPEG compression applied to the original image sequences. In particular, when background subtraction was applied to reveal the moving ball, variations in the compression in the pairs of images produced significant responses in the difference images around edges. Rather than improve the quality of the images it was decided that although they provided a greater challenge to the tracking algorithm the variations in the difference image provided a more demanding task to the thresholding algorithms. This was preferable since most thresholding algorithms work well on simple images, generally producing similar results. It is only on more difficult images that the results really diverge. To minimise the effects of noise and compression artifacts during tracking (but not during the later thresholding stage) a mask image was created from a background image containing no moving objects. The edges were extracted (using the Sobel detector), thresholded, and dilated to further reduce side effects.

To track the ball, edge detection was applied to the image sequence and artifacts removed by applying the mask. Circle detection was then

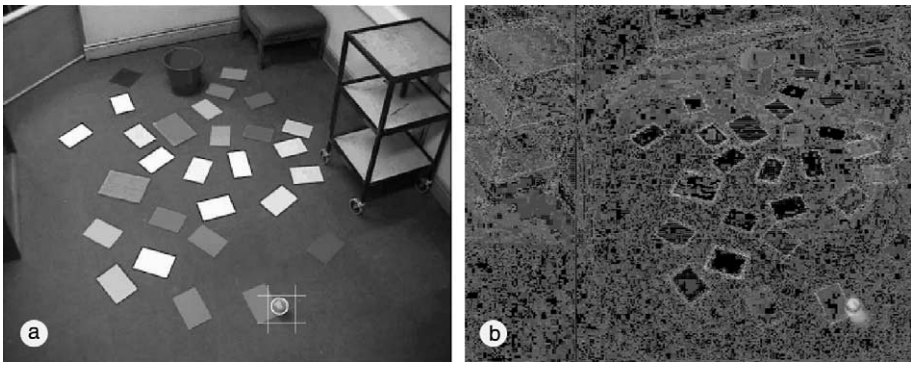

Fig. 1. (a) Sample image showing room with background texture on floor and tracked ball and (b) difference image before thresholding.

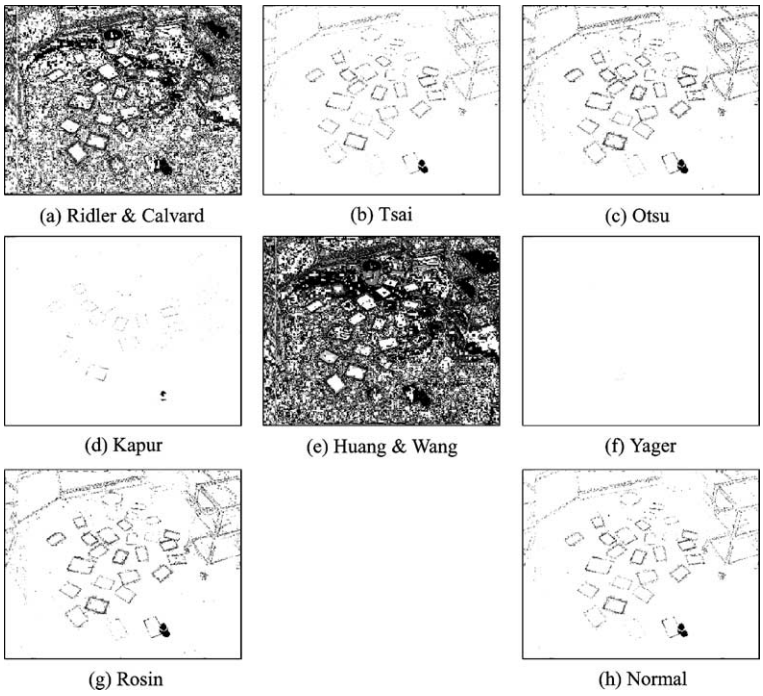

Fig. 2. Results of thresholding difference image with various algorithms.

straightforward. After initialisation in the first frame by hand, successive positions were found by searching in a window centred on the position of the ball for a new position and radius that maximised the mean edge magnitude over the circle's perimeter. The result of the circle detection was then used to position a disk representing the ball on a blank image thus producing the ground truth. This new sequence of ground truth images was visually inspected for discrepancies.

The resulting sequences were mostly very close to the expected result. Occasionally the tracker failed due to the multiple textures on the ball and the strong shadows which cause significant

distractions from the true boundaries of the ball. A possible solution would be to use a more sophisticated tracker, but given that the ball's movements were erratic (it was a toy that moved autonomously) improving the tracking would not be trivial and was not considered necessary; it was sufficient to simply to re-initialise the circle detection algorithm at those specific frames.

3.3. Analysis protocol

There are many different ways of evaluating the performance of algorithms, starting from analysing individual pixels at the lowest level, to higher levels which consider the overall effectiveness of the application that the thresholding is embedded within. Our initial approach is to measure the correctness of the algorithms at the pixel level which is independent of a specific application. At a goal directed level we continued by evaluating the effectiveness of the results for change detection.

3.3.1. Pixel based evaluation

The results of the low level pixel based comparison between the ground truth and the thresholded image for each frame of the sequence were based on the following values:

- True positives (TP): i.e. number of change pixels correctly detected.
- False positives (FP): i.e. number of no-change pixels incorrectly flagged as change by the algorithm.
- True negatives (TN): i.e. number of no-change pixels correct detected.
- False negatives (FN): i.e. number of change pixels incorrectly flagged as no-change by the algorithm.

From these four quantities the following measures were used:

- The percentage correct classification: $PCC = (\mathrm{TP} + \mathrm{TN})/(\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN})$.
- The Jaccard coefficient (Sneath and Sokal, 1973): $\mathrm{TP}/(\mathrm{TP} + \mathrm{FP} + \mathrm{FN})$.
- The Yule coefficient (Sneath and Sokal, 1973): $|(\mathrm{TP}/(\mathrm{TP} + \mathrm{FP})) + (\mathrm{TN}/(\mathrm{TN} + \mathrm{FN})) - 1|$.

The reason all three alternative measures were considered is that the seemingly simple task of quantifying the similarity between two binary images is surprisingly tricky. The most obvious approach is to combine all four values to form the PCC, and this is the most widespread method in computer vision for assessing a classifier's performance. However, it tends to give misleading estimates when the amount of change is small compared to the overall image. So, in our case where the amount of change represents less than 4% of the image, very high ratings (e.g. 96%) can be achieved simply by thresholding out everything and classifying the complete image as background. The error incurred by completely missing the ball is relatively small. In the area of remote sensing the limitations of PCC are well known (Congalton, 1991; Yuan and Elvidge, 1998), and alternatives such as the kappa coefficient (Cohen, 1960) and its various refinements are often used, but they have their own problems (Stehman, 1997).

The discipline of taxonomy also considers a range of assessment criteria. In particular, the Yule and Jaccard coefficients overcome our problem with the PCC to some degree by minimising or eliminating the effect of the expected large volume of true negatives. Note that the Yule coefficient cannot be applied when the algorithm correctly detected no change in the image (since one denominator becomes zero).

Within the sequences there are frames in which no change occurs. These were analysed separately so that we monitor the effects of noise and compression artifacts when no real activity exists in the sequence.

3.3.2. Goal based evaluation

At a higher level of analysis the surveillance system's goals will be to detect and possibly track moving objects. In this case small spurious blobs of brief duration are relatively unimportant as they can easily be filtered out. Thus, only the effectiveness of the detection of the main regions in the thresholded image should be considered in the assessment. A direct approach would be to simply determine the rate of success of the tracker. Instead, to avoid the rating being tied into the

tracker's mode of operation and settings we take an intermediate tack. Since we have control over the environment we can assume that there exists a single moving object in the scene. Therefore the automated goal directed testing took the form of checking the validity of the largest blob in the thresholding image. By filtering out all but this single blob, the spatial aspects of the image are taken into consideration. For instance, if thresholding produced many isolated small groups of pixels detected in the ball region this would receive a good TP score even though the results may be useless for tracking. In addition, a conservative size constraint was included to filter out tiny and enormous blobs, the allowable range of areas being $[20, 1000]$ pixels, enabling empty frames to be discarded.

Two aspects of the blob were measured—its area and compactness. The first computes the absolute area error which is given by subtracting the blob's area from the expected ground truth area. The second uses the knowledge that the moving object is round. Compactness is defined by the square of the perimeter of a region divided by its area, and is minimised in the case of the circle. Small values for both measures indicate good performance. Such an approach is similar to the *ultimate measurement accuracy* of Zhang (1996) in which image segmentation algorithms were assessed according to region area in the thresholded image, although in Zhang's case only extremely simple synthetic images were used.

In conclusion, we have described an evaluation system that is scalable and can be easily used for testing any thresholding algorithm. The results produced can be configured for different forms of evaluation, e.g. a variety of pixel based and/or goal based tests.

4. Results analysis

In this section we present the results obtained following the testing protocol. In Table 1, the test results for the seven sequences are listed, separated into the uniform and textured floor sequences. The results showed that most of the algorithms operate better under the less textured environment (no floor texture)—which is to be expected as the ball is then more distinct from the background.

Kapur was the best performing algorithm both quantitatively and qualitatively. It received substantially higher scores than the others, and on visual inspection we found it be the only one to avoid most of the ball shadow while retaining most of the ball. The Normal fitting, Tsai, and Rosin algorithms seemed to suffer more from the shadow and the compression/edge noise. In our opinion, on a high quality sequence with no artifacts, any of the top four algorithms would be suitable for change detection. The Ridler and Calvard, and Otsu algorithms performed significantly worse, and in general did very badly.

The Yager fuzziness algorithm achieved the best overall calculated performance according to the numerical scores, though it was proved, through visual inspection and analysis of the TP results, to actually perform the worst. About 40% of the time it would correctly detect less than 3% of the ball, and for 55% it would not detect any ball pixels (e.g. Fig. 2f). Furthermore, at best, it only detected 50% of the ball pixels. The reason that the measures do not sufficiently penalise this bad behaviour is due to the very small size of the ball, and so over-thresholding is assigned high scores. This was the main discrepancy regarding the evaluation measures, and the numerical ratings of the other algorithms match the visual judgements much better.

In Table 2, the results when no change occurred in the scene are shown. Thus, in this case the threshold determination task degenerates to ideally classifying all pixels as non-change. This situation is potentially problematic for some algorithms, but in fact causes little difference in the relative rankings of the algorithms currently under consideration.

We continued by performing evaluation at a goal based level. In Table 3 the compactness results applied on the sequences are shown. We note that the same four algorithms are in the top four ranking with some change in the order: Tsai, Rosin, Normal fitting, and Kapur. The reason for this change lies on the fact that the ball is multi-textured, therefore algorithms that have a less conservative approach to the thresholding value will be favoured over the others.

Table 1
Average and median scores for thresholding algorithms. Larger values indicate better performance

Algorithm	Measure	Uniform texture floor		Multi-texture floor	
		Avg. (1–4)	Med. (1–4)	Avg. (5–7)	Med. (5–7)
Ridler and Calvard	PCC	0.5527	0.5532	0.5799	0.5915
	Jaccard	0.0019	0.0017	0.0018	0.0015
	Yule	0.0018	0.0017	0.0017	0.0015
Tsai	PCC	0.9832	0.9839	0.9784	0.9770
	Jaccard	0.0437	0.0381	0.0267	0.0209
	Yule	0.0441	0.0384	0.0268	0.0209
Otsu	PCC	0.9022	0.9040	0.9559	0.9540
	Jaccard	0.0106	0.0063	0.0150	0.0112
	Yule	0.0105	0.0063	0.0149	0.0111
Kapur	PCC	0.9992	0.9992	0.9983	0.9983
	Jaccard	0.3557	0.3432	0.1543	0.1274
	Yule	0.5865	0.5645	0.2292	0.1664
Huang and Wang	PCC	0.2934	0.1348	0.5294	0.5860
	Jaccard	0.0013	0.0012	0.0016	0.0014
	Yule	0.0013	0.0012	0.0015	0.0014
Yager	PCC	0.9992	0.9993	0.9992	0.9994
	Jaccard	0.1179	0.1008	0.0472	0.0257
	Yule	0.8883	0.9779	0.5430	0.6214
Normal	PCC	0.9774	0.9775	0.9681	0.9684
	Jaccard	0.0317	0.0310	0.0182	0.0163
	Yule	0.0318	0.0310	0.0181	0.0161
Rosin	PCC	0.9891	0.9892	0.9814	0.9809
	Jaccard	0.0592	0.0523	0.0282	0.0246
	Yule	0.0604	0.0530	0.0283	0.0246

Table 2
PCC scores for sequences where no change occurs (larger values indicate better performance)

Algorithm	Ridler	Tsai	Otsu	Kapur	Huang	Yager	Normal	Rosin
Average	0.5452	0.9493	0.8264	0.9977	0.2041	0.9999	0.9721	0.9893
Median	0.5541	0.9368	0.8015	0.9984	0.1329	0.9999	0.9714	0.9894

Table 3
Compactness scores (smaller numbers represent better performance)

Algorithm	Ridler	Tsai	Otsu	Kapur	Huang	Yager	Normal	Rosin
Average	2144.49	175.90	1075.45	493.17	1019.72	616.86	220.09	212.90
Median	2074.00	187.00	952.00	522.00	562.00	628.00	172.00	235.00

Table 4 gives the area error. Note again that the Yager fuzziness method shows the best performance, but that is because 65% of the frames were not included in the analysis as no blob was detected. From this test we can see that the ranking of the best four algorithms is the same as at pixel level testing. One thing to note, there is little difference between the average and median values for

Table 4
Absolute area error (smaller numbers represent better performance)

Algorithm	Ridler	Tsai	Otsu	Kapur	Huang	Yager	Normal	Rosin
Average	292.74	69.66	319.53	24.42	198.51	24.75	109.49	42.09
Median	253.60	30.50	327.30	22.20	100.90	22.90	31.70	30.40

Kapur, but a more substantial difference for the other three. This suggests that sometimes the latter algorithms are unstable.

The experimental framework does not directly provide a means to exactly pinpoint the contributing factors to the ranking of the various algorithms since each thresholding algorithm is essentially treated as a black box. The closest we could get would be to systematically vary the scene and object parameters (e.g. illumination, background complexity, object size, colour, shape) and infer the factors from the scores. We have not carried out such experiments in this paper, but can make the following more general comments. Many traditional thresholding algorithms have difficulty with images that have primarily unimodal intensity distributions, such as encountered in this dataset, although Rosin's and Kapur's algorithms were shown to cope particularly well (Rosin, 2001). In that same study Ridler and Calvard's algorithm was generally found to perform well, but had occasional difficulty with images containing small foreground objects. Although many fuzzy thresholding schemes exist and have been shown to be successful in other contexts (Jawahar et al., 2000), it seems that in this case the two fuzzy measures of image similarity were not appropriate.

5. Further testing

To further test the thresholding algorithms we applied them to the ground truthed data made available by Prati et al. (2001). They provided 112 images from an indoor sequence containing a moving person along with a manual segmentation into foreground (human), shadow, and background. Although close examination of the ground

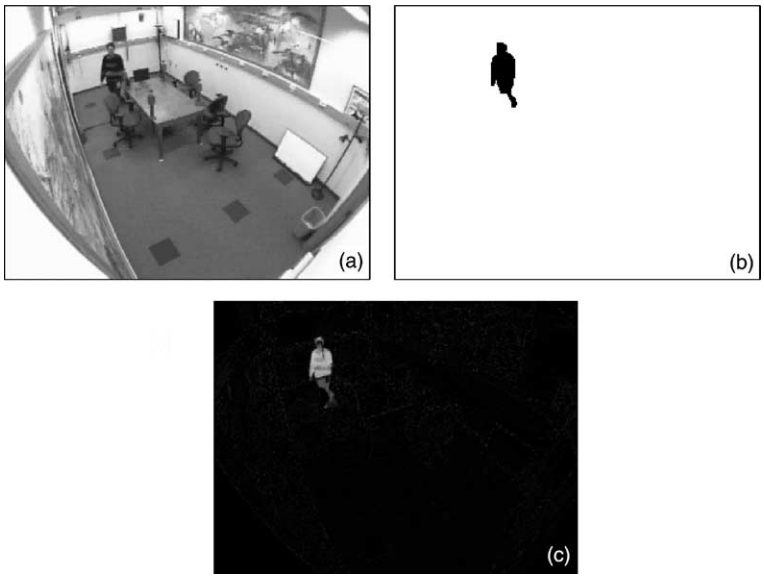

Fig. 3. Data from the intelligent room sequence. (a) Frame 218, (b) foreground and background ground truth and (c) difference image before thresholding.

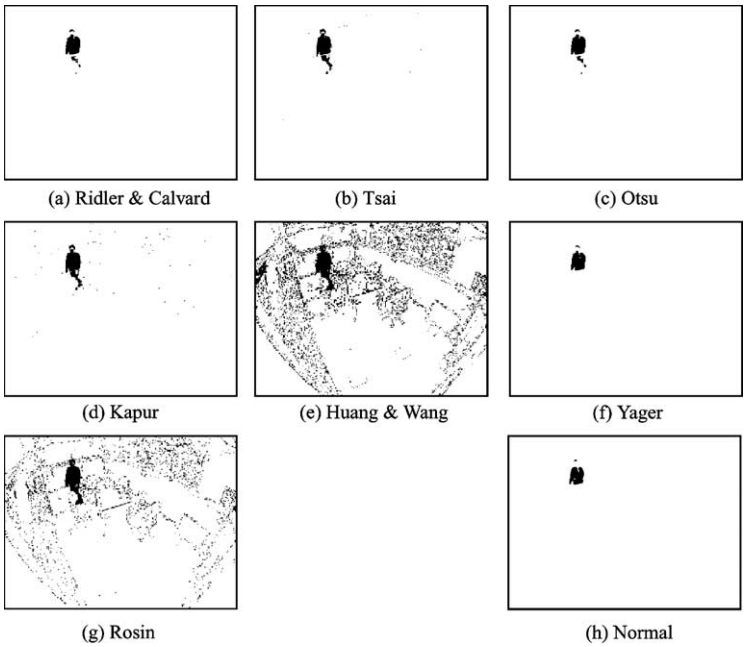

Fig. 4. Results of thresholding the difference image from frame 218 of the intelligent room sequence with various algorithms.

truth shows some errors in delineation this still provides a useful test set.

Fig. 3a and b shows an image from the sequence, and its associated ground truth (we use only foreground and background). Differencing with the first frame in the sequence (Fig. 3c) shows that there is considerable variation in contrast between the foreground and background, which makes the thresholding task rather harder. This is demonstrated in Fig. 4, where both the Yager and Normal algorithms have extracted the high contrast midsection of the person rather than the full body.

The results averaged over all images are listed in Table 5. Rosin's algorithm fared worse compared

Table 5
Mean results of thresholding the intelligent room sequence with various algorithms

Measure	Algorithm							
	Ridler	Tsai	Otsu	Kapur	Huang	Yager	Normal	Rosin
PCC	0.9928	0.9932	0.9928	0.9940	0.9119	0.9896	0.9929	0.9643
Jaccard	0.6258	0.6430	0.6295	0.6731	0.1963	0.4489	0.6119	0.2626
Yule	0.9395	0.8539	0.9381	0.8423	0.2380	0.9638	0.9127	0.2751

Table 6
Standard deviation of results of thresholding the intelligent room sequence

Measure	Algorithm							
	Ridler	Tsai	Otsu	Kapur	Huang	Yager	Normal	Rosin
PCC	0.0069	0.0060	0.0070	0.0054	0.0419	0.0101	0.0068	0.0078
Jaccard	0.1291	0.1349	0.1294	0.1215	0.2170	0.2521	0.1814	0.1310
Yule	0.1460	0.1600	0.1473	0.0757	0.3031	0.0923	0.1236	0.1465

to the previous test set. This is probably due to the bulk of background pixels around edges that give rise to higher differences than the remainder of the background, so that the result appears underthresholded (e.g. Fig. 4g). Tsai and Kapur's algorithms performed well again, while the Ridler and Calvard and Otsu algorithms performed at a similar level. The standard deviations of the scores (Table 6) highlights the variability (i.e. the unreliability) of the fuzzy based methods (Huang and Wang and Yager).

6. Conclusion

We have proposed an evaluation framework for testing thresholding algorithms in the context of surveillance applications. Its advantage over schemes using synthetic data is that it provides more realistic data, while compared to manual generation of ground truth it enables huge amounts of ground truthed data to be automatically generated in an efficient, simple, and objective manner.

As an example of the application of the methodology, tests were performed on a large scale (over 4000 images) providing a quantitative and thorough evaluation of eight different thresholding algorithms. The results showed that four of the algorithms had an acceptable performance, with Kapur's showing the best performance behaviour. A comparison with a different (manually) ground truthed dataset containing only 112 images showed a similar assessment for most of the algorithms.

We used pixel based numerical methods to capture the performance of the thresholding algorithms, and found that they can give misleading rankings. This led us to consider testing at a goal based level, which enabled these effects to be identified. In the remaining cases the algorithm performances were similar with the results obtained by pixel level testing. Furthermore, an indication of an algorithm's stability can be determined by looking at the standard deviation of its scores over the dataset, or by comparing its mean and median scores. Currently at the goal directed level we use compactness and area error measures, but we believe that more measures can be used to expand the testing.

In additional future work we plan to experiment under different or variable environments, e.g. illumination change, variable noise levels, different proportions of change (e.g. larger/smaller balls), etc. and study the effects of image pre- and post-filtering, e.g. median filtering after thresholding.

Acknowledgements

We would like to thank City University for providing the image sequences. This project is funded by EPSRC grant number: GR/M59594.

References

Barron, J., Fleet, D., Beauchemin, S., 1994. Performance of optical flow techniques. Int. J. Computer Vision 12 (1), 43–77.

Bowyer, K., Phillips, P., 1998. Empirical Evaluation Techniques in Computer Vision. CS Press.

Cohen, J., 1960. A coefficient of agreement for nominal scales. Educ. Psychol. Measurement 20, 37–40.

Congalton, R., 1991. A review of assessing the accuracy of classifications of remotely sensed data. Remote Sensing Environ. 37, 35–46.

Dougherty, S., Bowyer, K., 1998. Objective evaluation of edge detectors using a formally defined framework. In: Empirical Evaluation Techniques in Computer Vision. pp. 211–234.

Fitzgibbon, A., Pilu, M., Fisher, R., 1999. Direct least square fitting of ellipses. IEEE Trans. PAMI 21 (5), 476–480.

Forbes, L., Draper, B., 2000. Inconsistencies in edge detector evaluation. In: Conf. Computer Vision Pattern Recognition. pp. II:398–404.

Forstner, W., 1996. 10 pros and cons against performance characterisation of vision algorithms. In: ECCV Workshop on Performance Characteristics of Vision Algorithms.

Fu, K., Mui, J., 1981. A survey of image segmentation. Pattern Recognition 13 (1), 3–16.

Heath, M., Sarkar, S., Sanocki, T., Bowyer, K., 1997. Robust visual method for assessing the relative performance of edge detection algorithms. IEEE Trans. PAMI 19 (12), 1338–1359.

Hoover, A., Jean-Baptiste, G., Jiang, X., Flynn, P., Bunke, H., Goldgof, D., Bowyer, K., Eggert, D., Fitzgibbon, A., Fisher, R., 1996. An experimental comparison of range image segmentation algorithms. IEEE Trans. PAMI 18 (7), 673–689.

Huang, L., Wang, M., 1995. Image thresholding by minimizing the measures of fuzziness. Pattern Recognition 28, 41–51.

Jawahar, C., Biswas, P., Ray, A., 2000. Analysis of fuzzy thresholding schemes. Pattern Recognition 33 (8), 1339–1349.

Kadonaga, T., Abe, K., 1995. Comparison of methods for detecting corner points from digital curves. In: Int. Workshop on Graphics Recognition. pp. 3–12.

Kapur, J., Sahoo, P., Wong, A., 1985. A new method for gray-level picture thresholding using the entropy of the histogram. Comput. Vision Graphics Image Process. 29 (3), 273–285.

Kitchen, L., Rosenfeld, A., 1981. Edge evaluation using local edge coherence. IEEE Trans. Systems Man Cybernet. 11, 597–605.

Leung, C., Lam, F., 1996. Performance analysis for a class of iterative image thresholding algorithms. Pattern Recognition 29 (9), 1523–1530.

Levine, M., Nazif, A., 1985. Dynamic measurement of computer generated image segmentations. IEEE Trans. PAMI 7 (2), 155–164.

MVA, 1997. Special issue on performance evaluation. Machine Vision Applicat. 9 (5/7).

Otsu, N., 1979. A threshold selection method from gray-level histograms. IEEE Trans. Systems Man Cybernet. 9, 62–66.

Parker, J., 1996. Algorithms for Image Processing and Computer Vision. John Wiley & Sons.

Prati, A., Cucchiara, R., Mikic, I., Trivedi, M., 2001. Analysis and detection of shadows in video streams: A comparative evaluation. In: Conf. Computer Vision Pattern Recognition. pp. II:571–576.

Press, W., Flannery, B., Teukolsky, S., Vettering, W., 1988. Numerical Recipes in C. Cambridge University Press.

Ren, Y., Chua, C., Ho, Y., 2003. Statistical background modeling for non-stationary camera. Pattern Recognition Lett. 24 (1–3), 183–196.

Ridler, T., Calvard, S., 1978. Picture thresholding using an iterative selection method. IEEE Trans. Systems Man Cybernet. 8, 629–632.

Rosin, P., 2001. Unimodal thresholding. Pattern Recognition 34 (11), 2083–2096.

Sahoo, P., Soltani, S., Wong, A., Chen, Y., 1988. A survey of thresholding techniques. Comput. Vision Graphics Image Process. 41, 233–260.

Sezgin, M., Sankur, B., 2001. Image thresholding techniques: Quantitative performance evaluation. submitted for publication.

Sneath, P., Sokal, R., 1973. Numerical Taxonomy. The principle and practice of numerical classification. W.H. Freeman.

Stauffer, C., Grimson, W., 1999. Adaptive background mixture models for real-time tracking. In: Conf. Computer Vision Pattern Recognition. pp. II:246–252.

Stehman, S., 1997. Selecting and interpreting measures of thematic classification accuracy. Remote Sensing Environ. 62, 77–89.

Trier, O., Jain, A., 1995. Goal-directed evaluation of binarization methods. IEEE Trans. PAMI, 1191–1201.

Tsai, W., 1985. Moment-preserving thresholding. Comput. Vision Graphics Image Process. 29, 377–393.

Venkatesh, S., Kitchen, L., 1992. Edge evaluation using necessary components. CVGIP: GMIP 54 (1), 23–30.

Venkatesh, S., Rosin, P., 1995. Dynamic threshold determination by local and global edge evaluation. Comput. Vision Graphics Image Process. 57 (2), 146–160.

Weszka, J., 1978. A survey of threshold selection techniques. Comput. Graphics Image Process. 7 (2), 259–265.

Yager, R., 1979. On the measure of fuzziness and negation. Part I: Membership in the unit interval. Int. J. General Systems 5, 221–229.

Yuan, D., Elvidge, C., 1998. NALC land cover change detection pilot study: Washington DC area experiments. Remote Sensing Environ. 66 (2), 166–178.

Zhang, Y., 1996. A survey of evaluation methods for image segmentation. Pattern Recognition 29, 1335–1346.

Zhao, M., Yang, Y., Yan, H., 2000. An adaptive thresholding method for binarization of blueprint images. Pattern Recognition Lett. 21 (10), 927–943.

Zheng, Z., Wang, H., Teoh, E., 1999. Analysis of gray level corner detection. Pattern Recognition Lett. 20 (2), 149–162.