Defence Science Journal, Vol. 61, No. 1, January 2011, pp. 51-56, DOI: 10.14429/dsj.61.479

Fast Change Detection

Aniruddha Bose and Kunal Ray

Proof and Experimental Establishment, Chandipur–756 025
E-mail: a.bose@pxe.drdo.in, kunalrayind@gmail.com

ABSTRACT

Automated detection of changes in a sequence of images captured under fixed background and steady light condition is an often required operation having widespread application. Possible application areas can be as diverse as from military to atmospheric science, from medicine to video surveillance, etc. There are many approaches to the problem of detecting changes; the more reliable one tries to make these more complex and computationally expensive these become, requiring sophisticated algorithms and specialised hardware. However, often one needs to use simple and computationally cheap procedures to be used with cots hardware when the problem scenario has static features with transient change in features associated to some small part of the image or field of view. A super-pixel-based change detection algorithm has been descried here that is basically a modification of the image differencing technique. The procedure has been seen to detect even a small transient change in intensity at a frame rate of as high as fifty frames per second using cots hardware.

Keywords: Image processing, automated change detection, algorithm, fast algorithm, image differencing technique

1. INTRODUCTION

Detecting transient changes in features associated with very small part of an image frame in a sequence of images is a necessity in diverse areas. Among the most important applications include remote sensing[1-4], industrial inspection[5], video surveillance[6,7] and semi automated medical diagnosis[8,9]. Other than thesewe have applications of change detection in underwater sensing[10], and even in driving assistance[11,12].

As change can be of many types, so is its detection, which can be done following many different approaches; most common among these uses image differencing, principal component analysis, object classification and comparison, change vector analysis[1,13] etc. Typically, through detection of changes one strives to ascertain the evolution of specific features or parameters over time. Simple change detection techniques only intimate of actual occurrence of change, while more sophisticated techniques provide detailed information about the location(s) of the features(s) undergoing change as well as qualitative or quantitative information about the relevant feature(s). Typically except for very simple cases, which are unrealistic, or for very simple techniques, change detection is quite expensive in terms of time as well as computation. For simple techniques which work directly on image data, like simple image differencing, the output is not very reliable except for artificially simplified cases or for highly controlled scenarios. To improve reliability, one slowly moves from using direct image data to single or multi-step-derived information like information generated after segmentation or object classification. Recently, object-based change detection has come to the forefront of interest in the field of remote sensing[14] as reliability of such methods have been seen to be among the best possible offline procedures. In these types of methods, one needs to segment the image and identify objects of interest; subsequent steps are then carried out w.r.t. the objects or some of their features only. In contrast with the traditional pixel-based approach, these kind of object-based approaches have the benefit of having much more information like object texture, shape, compactness, etc. to aid in decision-making steps, which makes them much more reliable. These object-based techniques, though very useful for offline processes, are impractical when the computational cost becomes important. For scenarios in which many of the features of the events are known and are almost static (though not controlled), changes are expected to be associated to a very small (< 0.001) part of image and the response time of detection algorithm must be fraction of inter-arrival time of image frames, algorithms having low computational cost yet moderate-to-high reliability are the answer.

A simple super pixel-based change detection algorithm that is very fast but is more reliable than traditional image differencing techniques is discussed. This technique can be generalised for any sized image and can be very easily implemented in hardware to provide change detection as a pre-processing step at TV rate.

2. METHODOLOGY

Most of the simple techniques, which work directly on image data, are based on exploitation of intensity data of the pixels. A common approach in intensity-based change

Received 7 July 2010, Revised 30 November 2010, Published Online 6 January 2011

detection problems is some variations of image differencing techniques where the choice is based on minimisation of computation cost and time. These are primarily used for the so called motion-still segmentation algorithms[7], shot change detection algorithms[6], etc. which are widely used for video data compression. These algorithms are primarily pixel-based, as even pixel level mistakes in reconstruction will be noticeable. These algorithms should be reliable and fast. However, there are many applications like vehicle tracking[12] where speed is a very important criterion.

The problem scenario in the current case is described as follows:

- Projectiles fitted with flash charge are fired over sea during low tide.
- The shells explode coming in proximity with ground generating a predictable sized flash event.
- The event is observed through cameras staring in the approximate direction of the event.
- The background is primarily featureless and slowly varying.
- The flash event always occurs at an approximately fixed distance and so flash size can be predicted with little error.
- The requirement is automatic detection of a flash event.
- The exploding projectile still moves quite fast and so the frame rate of the camera should be high.

A similar requirement is also there in applications where the detection by the camera is used to cue some event. A typical example of such a case can be using a camera to provide trigger to optical tracker, radar, etc. in place of a flash detector. A flash detector is extremely fast but may be difficult to set up and may not be very reliable for remote observation. In such a scenario one needs to use a high frame rate camera with suitable lens and a fast algorithm to process the output.

One can formulate the problem by considering a sequence of gray scale images having a resolution of 2^m x 2^n where m, n is positive integer. Let $I_k(i,j)$ $(0<i<m, 0<j<n)$ is the intensity of the i,jth pixel in the kth frame of the sequence. If one performs an image subtraction between the kth and $k+1$th frame then the i,jth pixel of the difference image, $Ä_{k,k+1}(i,j)$, is given by

$$\Delta_{k,k+1}(i,j) = I_{k+1}(i,j) - I_k(i,j) \tag{1}$$

Collecting the values for all pixels, one gets the difference image. Ideally the quantity on the left should be zero when there is no change in the images through the sequence, and the difference image should be a matrix with all elements zero. However, for real image sequences, the matrix will almost never have all zero values. This is due to various fluctuations of pixel-level attributes which may not be controllable. As most of these fluctuations are random in nature and their magnitudes are usually very small, their effects can be eliminated if one uses a threshold or if one clubs a number of pixels together to form larger super pixels and use their properties instead of those of single pixels. In that case also, threshold will be required; however the threshold in the latter case should be globally more stable than the threshold required in case of individual pixels. This is logical as the fluctuations mentioned earlier are expected to be random in nature.

A super pixel-based image differencing algorithm has been discussed that could keep up with a frame rate of up to 50 fps for an image size of 2^m x 2^n where $m=n=9$ and give reliable result.

The procedure can be broken up in following steps:

Step 1. Generation of super pixel dimensions and allocation of data structure: The authors have combined a 2^4x2^4 block of pixels into one super pixel. The number of such blocks along the horizontal and vertical directions can be found out by dividing the relevant number of pixels by 16. Two numbers of two-dimensional arrays of unsigned integers have been used to represent two sets of values for the boxes; one for each of the two images under comparison. Both the data structures are initialised to zeros. The particular size of the pixel boxes should be chosen judiciously; too small a box will approach the classical image differencing situation and too large a box will fail to detect any change other than catastrophic ones.

Step 2. Setting up the access mechanisms for the images under comparison: The images under comparison are gray scale images referenced by unsigned character pointers. To improve the performance, the authors have type casted this unsigned character pointers to double word (WIN32 data type DWORD). This has allowed to move every four contiguous pixel values at a time between memory and CPU resulting in substantial reduction in number of memory accesses which is an important aspect of the optimisation in the algorithm.

Step 3. Processing on actual image data: To achieve an efficient performance level, bitwise operators have been used. The procedure can be understood in terms of following steps:

- To access the pixels there are nested 'for' loops which traverse the pixel location in vertical and horizontal directions. Usage of DWORD type casting reduces number of memory accessing to one-fourth of actual pixels of the image.
- To generate the indices for the pixel boxes the current value of the pixel location counter was right shifted by two bits (divide by 2^2). Let the image resolution be 640 across by 512 pixels down. So the counters run up to 160 and 128, respectively. Let the first counter be at a value of 8 and the second one is at a value of 29. These are actually referencing pixels around the locations 32 and 116, respectively which should go to the box with indices 2 and 7 (divide them by 16). The same values are generated by right shifting the counter values by two bits.
- Next step should be the actual extraction of the pixel data. DWORD pointer as mentioned earlier is

used to access every four contiguous pixels. To access the information bitwise ANDing were applied between the content pointed to by the DWORD pointer and a suitable mask patterns to extract the desired byte corresponding to one of the four pixels. The result was then right shifted by suitable number to extract the byte. For example, for the first pixel, the mask O x FCOOOOOO was used and the result was then shifted to right by 24 bits to extract the first byte; similarly for the second pixel we use the mask O x FCOOOO was used and shifted right by 16, and so on. The outputs of each step were added to generate an accumulation of pixel values and were assigned to the appropriate boxes. Selection of mask as O X FC rejects the lowest two of the significant bits which are considered to be noisy implying an overall input SNR of 18dB per pixel. This inference is based on actual scenario of interest and may require modification for different scenarios of image capture. The procedure can also be executed in reverse order with some changes in implementation.

- After data was extracted for all the four pixels and has been summed up, the accumulated value was sent to the appropriate box as mentioned earlier where it was added to the existing value in the box.
- In this way, the entire image was scanned.

Step 4. Checking for Change: Afterwards the values accumulated in the boxes were compared between the current frame and the previous frame using nested 'for' loops. To save computation time, once the comparison has been made for a box pair, the accumulated value in the box for current frame was transferred to the corresponding box for previous frame within the same nested 'for' loops. A difference more than a pre-set threshold is considered as valid change. Depending on the requirement, the processing can be stopped as soon as a valid change is detected; or the locations of all the valid change points may be saved in an array for further processing – like detection of shape, contour, etc.

3. RESULTS & DISCUSSION

The system was tested with various types of backgrounds. Figures 1 and 2 show a part of input images captures under ambient sunlight and the detected event is an explosion in air. The input image has a size of 512 x 512 and the explosion patch and its reflection have an approximate size of 17 x 40. To illustrate the output of our algorithm, the procedure was run on the each input image to generate $2^5 \times 2^5$ numbers of rectangular sub-images of $2^4 \times 2^4$ pixels size, where each of these pixels were set to the average gray scale value of that sub-image. Figures 3 and 4 show the respective modified input images. Figure 5 illustrates the difference image. To improve visualisation contrast stretching has been applied to produce Figs 3-5.

Looking at the figures, the boxes are clearly identifiable.

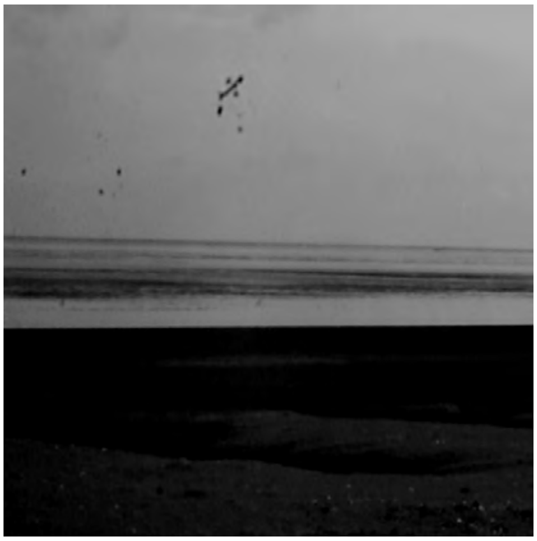


Figure 1. The background.

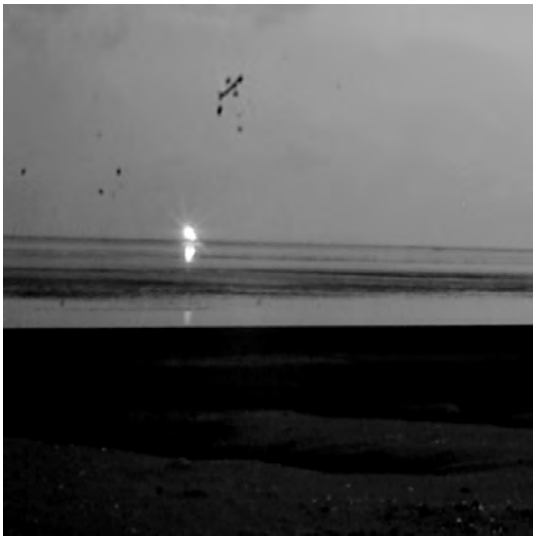


Figure 2. The event.

The apparent darkness in the second image is due to the bigger maximum for normalisation in it. Clearly the event can be detected based on the processed image.

Since the technique is based on comparison of intensity values, the choice of threshold becomes quite important and decides the reliability of outcome. However, through experiment it has been found that the intensity of the flash patch is quite high compared to the rest of the image and so any threshold value more than the average intensity of the scene is adequate to ensure reliability. Repeated application of this algorithm throughout the year in different ambient light conditions can help to generate suitable methodology to automatically determine the optimal value of threshold on any particular operating situation.

Often threshold selection is based on a study of image histogram[15]. However, the histogram-based image analysis works well when the image sequences are in general unimodal

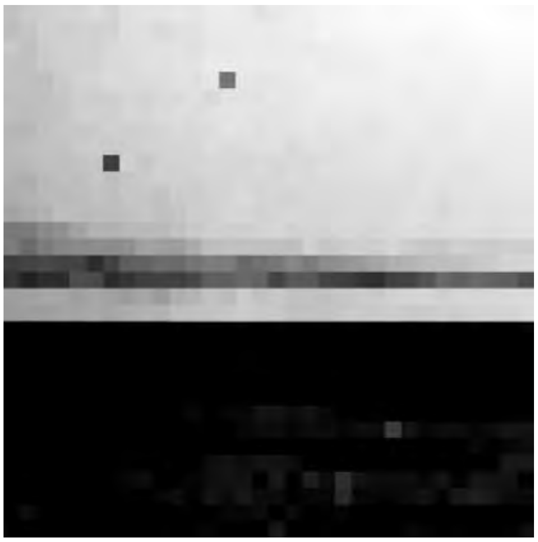


Figure 3. Processed view of Fig 1.

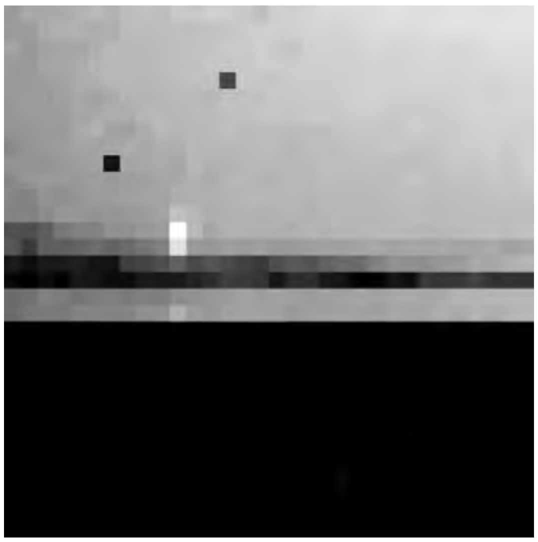


Figure 4. Processed view of Fig. 2.

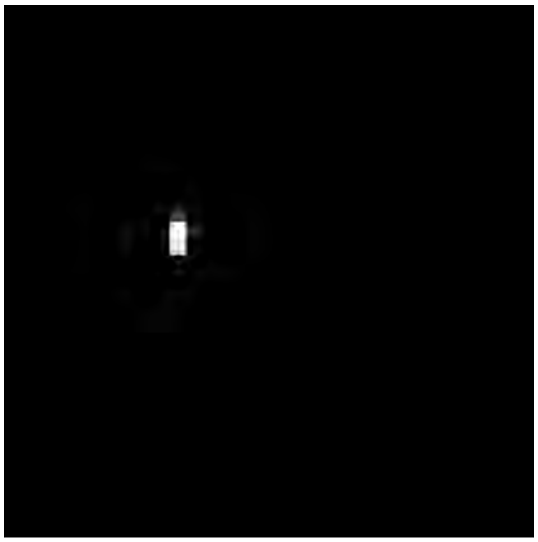
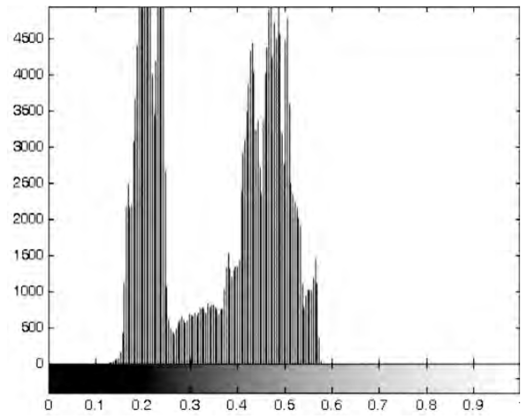


Figure 5. The difference image between Figs 3 and 4.

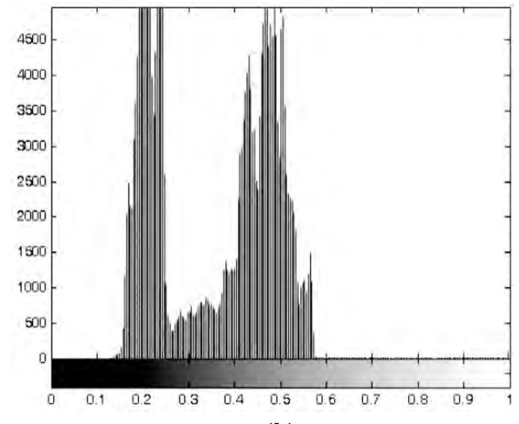
and becomes bimodal in case of presence of any event. The small bright spot of image in Fig. 2 is too little to modify the histogram of image of Fig. 1. Looking at the Figs 6(a) and 6(b) one cen see that these are almost identical and so histogram-based threshold estimators cannot work.

To get an estimate of the speed of execution, the algorithm was run for image sequences of different length starting from 100 frames to 700 frames, and the time taken was checked as per the computer clock. The image resolution was 640 pixels by 512 pixels. A plot of mean time of execution against image sequence length shown in Fig. 7(a) indicates the performance of the algorithm. The application software has been developed using Microsoft Visual C++ 6.0. The time estimates were found by checking the PC clock before and after running the algorithm. A commercial desktop PC with 3.00 GHz Pentium Core 2 Duo processor with 3 GB RAM and running Windows XP service pack 3 (upgraded from SP2) was used.

To assess the improvement in performanece due to usage of DWORD type casting, the effect of using LPSTR to store the image data in the same algorithm on the same sequence of frames was tested. An LPSTR is a pointer to an 8-bit character, so instead of one memory access to transfer data of four pixels to CPU for subsequent



(a)

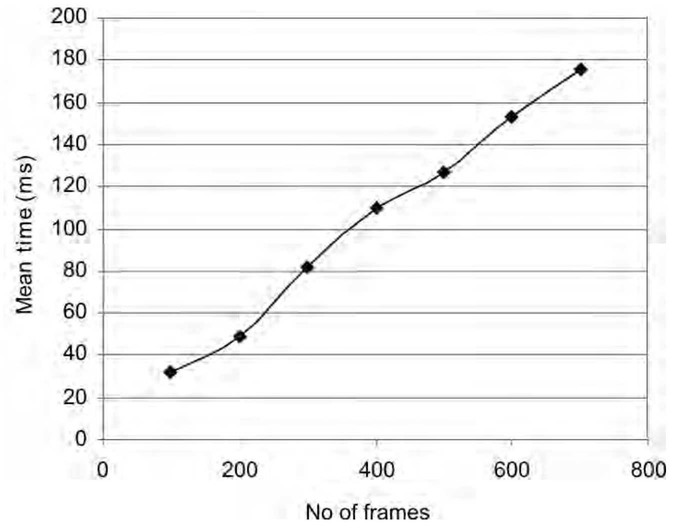


(b)

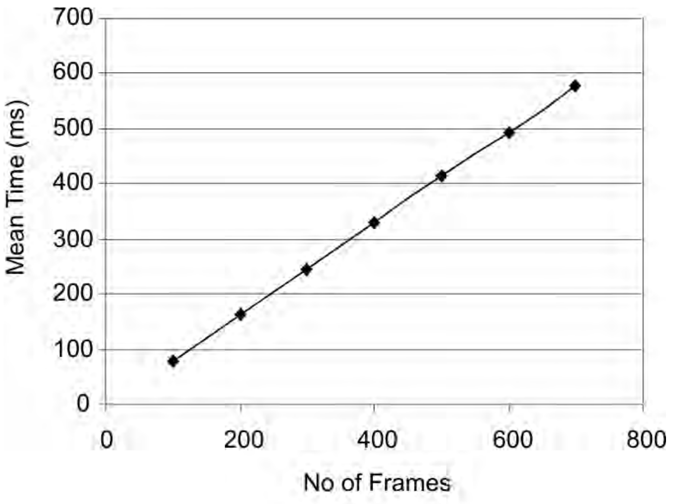
Figure 6.(a) Histogram of Fig. 1, (b) Histogram of Fig. 2.

bitwise AND operation to separate each pixel value, one is actually accessing memory location four times to get the data for the four pixels. The finding of the exercise is illustrated in Fig. 7(b).

One can see by looking at Figs 7(a) and 7(b) that the time of execution has increased approximately by three times.



(a)



(b)

Figure 7. (a) Plot of mean time of execution against no. of frames processed, (b). Plot of time of execution against no of frames processed (second case).

4. CONCLUSIONS

An innovative approach for online detection of any intensity-based change associated with small part of the image in a sequence captured at a frame rate of 50 fps has been discussed. Essentially the objective was that processing time should be much less than the inter-frame interval. The algorithm developed has been in use for inhouse developed software based on photogrammetry-based measurement technique. It is seen that thoughtfully chosen data structure, data type casting, usage of appropriate operations, sequencing of activities to achieve optimised performance, has remarkably improved the performance of the algorithm. Figure 7(a) clearly shows that the algorithmic approach used can allow much higher frame rate in our type of usage. Similarly this approach may be used to support complex image processing tasks at 50 image frames/s.

ACKNOWLEDGEMENTS

The authors wish to acknowledge Maj Gen P. Mathur, Director, and Mr T.K. Biswal, Sc F for rendering support and encouragement.

REFERENCES

1. Fung F; and Le Drew E. Application of principal components analysis to change detection. Photogram. Engg. & Remote Sen., **53**(12), 1987, 1649-658.
2. Wang, Fangju, A. Knowledge-based vision system for detecting land changes at urban fringers. *IEEE Trans. Geosci. Remote Sen.*, **31**(1), 1993.
3. Xu, L; He, Z; Zhang, S. & Guo, Y., The comparative study of three methods of remote sensing change detection. *In* 17th IEEE International Conference on Geoinformatics, 2009.
4. Dai, X & Khorram, S. The effects of image misregsitration on the accuracy of remotely sensed change detection. *IEEE Trans. on Geosci. Remote Sen.*, **36**(5), 1998.
5. Choi, S.H.; Yun, J.P.; Seo, B.; Park, Y.; & Kim, S.W. Real-time defects detection algorithm for high-speed steel bar in coil, *World Acad. of Sci. Engg. Techno.*, **25**, 2007.
6. Kim, W. & Kim, J. An adaptive shot change detection algorithm and Its implementation on portable multimedia player. *IEEE Trans. Consumer Electro.*, **55**(2), 2009.
7. Voran, Stephen & Wolf, Stephen. Motion-still segmentation algorithm for VTC/VT objective quality assessment. Analogue Interface Performance Specification for Digital Video Teleconferencing/Video Telephony Service, National Telecommunications and Information Administration Institute for Telecommunication Sciences. Document No. T1Q1.5/91-110, 22 January 1991.
8. Dumskyj, M.J. Aldington, S.J., Dore, C.J.; & Kohner, E.M. The accurate assessment of changes in retinal vessel diameter using multiple frame electrocardiograph synchronised fundus photography. *Current Eye Res.*, **15**(6), 652-32, 1996.
9. Lemieux, L.; Wieshmann, U.; Moran, N.; Fish, D.; & Shorvon, S. The detection and significance of subtle changes in mixed-signal brain lesions by serial MRI scan matching and spatial normalisation, *Medical Image Ana.*, **2**(3), 227-42, 1998.
10. Edgington, D.; Dirk, W.; Salamy, K.; Koch, C.; Risi, M.; & Sherlock, R. Automated event detection in underwater video. *In* Proceedings of MTS/IEEE Oceans 2003 Conference, 2003.
11. Fang, C.Y.; Chen, S.W. & Fuh, C.S. Automatic change detection of driving environments in a vision-based

driver assistance system. *IEEE Trans. Neural Networks*, **14**(3), 646-57, 2003.
12. Kan, W.Y.; Krogmeier, J.V. & Doerschuk, P.C. Model-based vehicle tracking from image sequences with an application to road surveillance, *Optical Engineering*, **35**(6), 1723-729, 1996.
13. Johnson, R.D. & Kasischke E.S. Change vector analysis: a technique for multispectral monitoring of land cover and condition. *Int. J. of Remote Sen.* **19**(3): 411-26, 1998.
14. Wang, W.; Zhao, Z. & Zhu, H. Object-oriented change detection method based on multi-scale and multi-feature fusion. *In* IEEE Urban Remote Sensing Joint Event, 2009.
15. Nicolas, Coudray; Jean-Luc, Buessler & Jean-Philippe, Urban. Robust threshold estimation for images with unimodal histograms. *Pattern Recog. Lett.* **31**, 1010-019.

Contributors

Mr Aniruddha Bose obtained his BSc (Physics) from Visva Bharati University, West Bengal in 1987, and MSc (Computer Science) from JK Institute of Applied Physics & Technology, Allahabad University in 1989. He is working as a Scientist since 1990 at Proof and Experimental Establishment, Chandipur, Orissa, India.

Mr Kunal Ray obtained his BSc (Physics) from Serampore College, West Bengal, in 2001, and MSc (Physics) from University of Calcutta, West Bengal, in 2003. He is working as a Scientist since 2004 at Proof and Experimental Establishment, Chandipur, Orissa, India.