Chapter 2
Pixel-Based Change Detection Methods

Abstract In this chapter, we consider pixel-based change detection methods. First, we provide well-known methods in the literature. Then, we propose two novel pixel-based change detection methods.

Keywords Pixel-based change detection · Image differencing · Automated thresholding · Percentile · Otsu's method · Kapur's algorithm · Image rationing · Image regression · Least-squares · Change vector analysis (CVA) · Median filtering · Background formation · Fuzzy logic · Fuzzy xor

2.1 Image Differencing

In this technique, images of the same area, obtained from times t_1 and t_2, are subtracted pixelwise. Mathematically, the difference image is

$$I_d(x, y) = I_1(x, y) - I_2(x, y), \tag{2.1}$$

where I_1 and I_2 are the images obtained from t_1 and t_2, (x, y) are the coordinates of the pixels. The resulting image, I_d, represents the intensity difference of I_1 from I_2. This technique works only if images are registered.

To interpret the difference image, we need to recall the quotation from Singh [1]: "The basic premise in using remote sensing data for change detection is that changes in land cover must result in changes in radiance values and changes in radiance due to land cover change must be large with respect to radiance changes caused by other factors." Based on this principle, we can expect that intensity differences due to land cover change resides at the tails of the difference distribution of the image. Assuming that changes due to land cover are less than changes by other factors, we expect that most of the difference is distributed around the mean. We can illustrate the difference distribution as in Fig. 2.1.

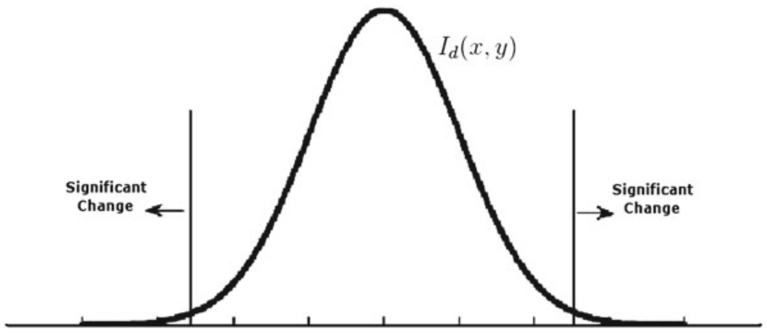


Fig. 2.1 Distribution of a difference function. Significant changes are expected at the tails of the distribution

For a zero mean difference distribution, we can normalize I_2 as

$$\tilde{I}_2(x, y) = \frac{\sigma_1}{\sigma_2}(I_2(x, y) - \mu_2) + \mu_1, \tag{2.2}$$

where $\tilde{I}_2$ is the normalized form of I_2. μ_1, σ_1 and μ_2, σ_2 are the mean and the standard deviation of I_1 and I_2, respectively. After normalization, the mean and standard deviation of the two images are equalized. Hence, the difference image will have zero mean. Now, we can update Eqn. 2.1 as

$$I_d(x, y) = |I_1(x, y) - \tilde{I}_2(x, y)|. \tag{2.3}$$

To detect the change, we can apply simple thresholding to $I_d(x, y)$ as

$$T(x, y) = \begin{cases} 1, & I_d(x, y) \geq \tau \\ 0, & \text{otherwise,} \end{cases} \tag{2.4}$$

where the threshold τ is often determined empirically.

Since the threshold value in Eqn. 2.4 is important, various automated threshold selection algorithms are proposed. Most of the time, the performance of these algorithms is scene dependent due to the assumptions they are based on. Rosin and Ioannidis [2] investigated the performance of several automated thresholding algorithms using a large set of difference images calculated from an automatically created ground truth database. They give results based on several measures for a complete evaluation. In this study, we benefit from three different threshold selection methods. These are percentile thresholding, Otsu's method [3] and Kapur's algorithm [4].

We will briefly explain these thresholding methods next. To this end, we make some assumptions about I_d as follows. I_d is a grayscale image which is represented by N_g gray levels, $\{1, 2, \ldots, N_g\}$. The number of pixels at level i is denoted by n_i and the total number of pixels is N.

The first thresholding method is based on the percentile [5]. It is a statistics of ordinal scale data. Assume that A is a sorted array of pixel values of I_d in ascending order. Rank of the Pth percentile of I_d is given by

$$R = \text{ceil}\left(\frac{P}{100} \times N\right) \tag{2.5}$$

where the ceil function rounds its argument to the nearest greater integer. Pth percentile is found by indexing A using that rank.

The second thresholding method is proposed by Otsu. It uses measures of class separability in finding an optimal threshold value. Relative frequencies of pixel values at level i are given by

$$p_i = \frac{n_i}{N}, \quad p_i \geq 0, \quad \sum_{i=1}^{N_g} p_i = 1. \tag{2.6}$$

A threshold value at gray level k divides the histogram of I_d into two classes. Each class has its own probability of occurrence (total probability of its samples) and own mean value. Evaluation function of the Otsu's method is the between-class variance given by

$$\sigma_b^2 = \frac{[\mu_{I_d}\omega(k) - \mu_\omega]^2}{\omega(k)[1-\omega(k)]}, \tag{2.7}$$

where μ_{I_d} is the mean of I_d; $\omega(k)$ is the probability of the class which includes gray levels up to k and μ_ω is the mean of the class ω. The optimal threshold value k^* maximizes

$$\sigma_b^2(k^*) = \max_{1\leq k\leq N_g} \sigma_b^2(k). \tag{2.8}$$

The last thresholding method we use is Kapur's algorithm. Similar to Otsu's method, it divides the image histogram into two classes. It then utilizes the sum of the entropy of these two classes as an evaluation function. The value which maximizes this sum is taken as the optimal threshold value. For two classes A and B, Shannon entropy of these classes are defined as

$$H(A) = -\sum_{i=1}^{k} \frac{p_i}{\omega(k)} \ln \frac{p_i}{\omega(k)}, \tag{2.9}$$

$$H(B) = -\sum_{i=k+1}^{N_g} \frac{p_i}{[1-\omega(k)]} \ln \frac{p_i}{[1-\omega(k)]}, \tag{2.10}$$

where the histogram is divided at gray level k. The optimal threshold value k^* maximizes the sum $\phi(k) = H(A) + H(B)$ such that

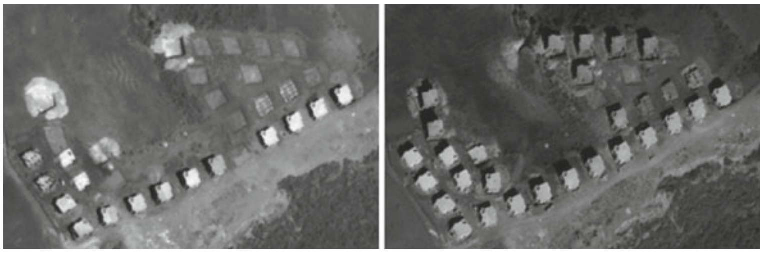


Fig. 2.2 Images taken at two different times from a developing region of Adana

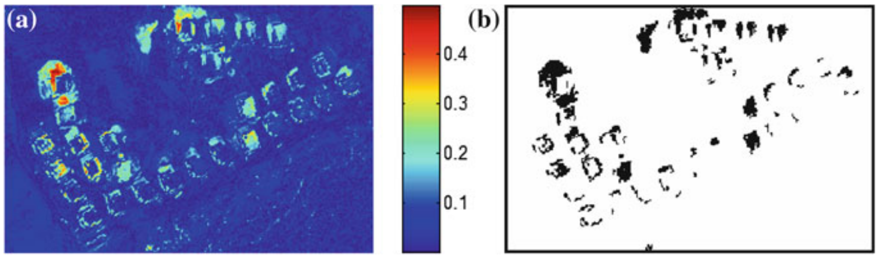


Fig. 2.3 Image differencing applied to the Adana image set. **a** The difference image **b** Thresholded version

$$\phi(k^*) = \max_{1 \leq k \leq N_g} \phi(k). \tag{2.11}$$

To explain different change detection methods, we pick the Adana test image set given in Fig. 2.2. The two images, taken in different times, in this set represent a region with construction activity. These images are registered. Therefore, they can be used for pixelwise change detection methods. The difference between these two images is clearly seen. We will use this image set in the following sections also.

The difference image obtained from the Adana image set is as in Fig. 2.3a. This image is color coded with the color scale given next to it. We also provide the thresholding result in Fig. 2.3b. In thresholding, we benefit from Kapur's method. As can be seen, the thresholded image provides sufficient information about the changed regions in the image.

Griffiths [6] used image differencing for detecting the change in urban areas. He used Landsat TM data (with 30 m resolution), SPOT XS multispectral data (with 20 m resolution), and SPOT panchromatic data (with 10 m resolution) in his study. He proposed using an urban mask to find changes in urban areas using image differencing. Griffiths indicated that the mixture of buildings, streets, and small gardens in urban areas produce a highly textured appearance compared with the much smoother texture of arable fields. He used a standard deviation filter to quantify the texture. The urban mask is multiplied by the difference image to eliminate non-urban areas. Furthermore, he refined the results based on a previous study. In this technique, changes that occur far from the urban areas are assumed to be non-urban change. This is because new urban development generally occurs at the periphery of existing urban areas. Griffiths presented his results for each technique by visual interpretation.

Saksa et al. [7] used image differencing for detecting clear cut areas in boreal forests. They tested three methods using Landsat satellite imagery and aerial photographs as: pixel-by-pixel differencing and segmentation, pixel block-level differencing and thresholding, pre-segmentation and unsupervised classification. In the first method, they found the difference image. Then, they used a segmentation algorithm to delineate the clear cut areas. In the second method, they included neighboring pixels into the calculation of the difference image. Therefore, negative effects of misregistration are reduced in the resulting image. In the third method, they first segmented the images. Then, they obtained a segment-level image difference. They labeled clear cut areas by using an unsupervised classification algorithm. Saksa et al. concluded that, predelineated segments or pixel blocks should be used for image differencing in order to decrease the amount of misinterpreted small areas.

Lu et al. [8] compared 10 binary change detection methods to detect land cover change in Amazon tropical regions. They used Landsat TM (Thematic Mapper) data in their study. In addition to band differencing, they tested a modified version of image differencing where pixels are accepted as changed when majority of the bands indicate change. For six-band Landsat TM data, if four of the bands indicate change then the pixel value is labeled as changed. They reported that the difference of Landsat TM band 5, modified image differencing, and principal component differencing produced best results.

2.2 Image Rationing

Similar to image differencing, images are compared pixelwise in this method. Therefore, images must be registered beforehand. The ratio image, used in this method, is calculated by

$$I_r(x, y) = \frac{I_1(x, y)}{\tilde{I}_2(x, y)}. \tag{2.12}$$

In Eqn. 2.12, the I_r image takes values in the range $[0, \infty)$. If the intensity values are equal, it takes the value 1. To normalize the value of I_r, we can benefit from the arctangent function as

$$I_r(x, y) = \arctan\left(\frac{I_1(x, y)}{\tilde{I}_2(x, y)}\right) - \frac{\pi}{4}. \tag{2.13}$$

Now, ratio image takes values in the range $[-\pi/4, \pi/4]$. To threshold I_r, we can benefit from the same methods as we did in the previous section. In Fig. 2.4a, we provide the I_r image obtained from the Adana test image set. We provide the thresholded version of this image in Fig. 2.4b. As in the previous section, we used

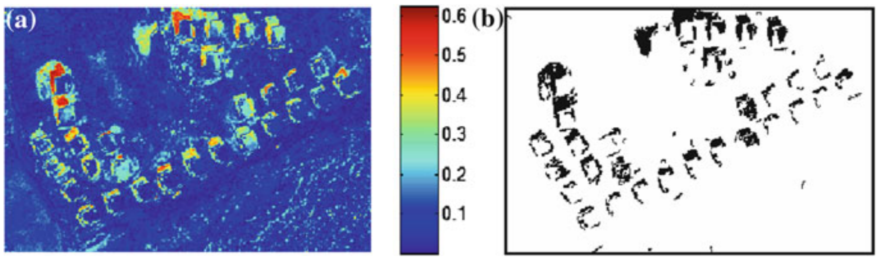


Fig. 2.4 Image ratio applied to the Adana image set. **a** The ratio image **b** Thresholded Version

Kapur's method to obtain the optimal threshold value. As can be seen, the thresholded image provides sufficient information on the changed regions in the image.

2.3 Image Regression

In image regression, the I_2 image (obtained from t_2) is assumed to be a linear function of the I_1 image (obtained from t_1). Under this assumption, we can find an estimate of I_2 by using least-squares regression as

$$\hat{I}_2(x, y) = aI_1(x, y) + b. \tag{2.14}$$

To estimate the parameters a and b, we define the squared error between the measured data and predicted data (for each pixel) as

$$e^2 = (I_2(x, y) - \hat{I}_2(x, y))^2 = (I_2(x, y) - aI_1(x, y) - b)^2. \tag{2.15}$$

The sum of the squared error becomes

$$S = \sum_{n=1}^{N} e^2 = \sum_{n=1}^{N} (I_2(x_n, y_n) - aI_1(x_n, y_n) - b)^2. \tag{2.16}$$

Here, we assume that we have N observations. We want to find the parameters a and b to minimize the sum of the squared error S. Therefore, we first calculate the partial derivatives of S with respect to a and b as

$$\frac{\partial S}{\partial b} = -2 \sum_{n=1}^{N} (I_2(x_n, y_n) - aI_1(x_n, y_n) - b), \tag{2.17}$$

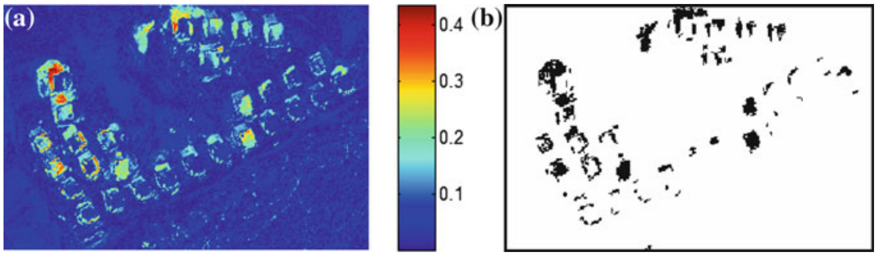


Fig. 2.5 Image difference after regression applied to the Adana image set. **a** The difference image **b** Threshold Version

$$\frac{\partial S}{\partial a} = -2\sum_{n=1}^{N}[(I_2(x_n, y_n) - aI_1(x_n, y_n) - b)I_1(x_n, y_n)]. \tag{2.18}$$

By equating Eqn. 2.17 and 2.18 to zero, we obtain two equations with two unknowns as

$$0 = \sum_{n=1}^{N} I_2(x_n, y_n) - \sum_{n=1}^{N} aI_1(x_n, y_n) - \sum_{n=1}^{N} b, \tag{2.19}$$

$$0 = \sum_{n=1}^{N} I_2(x_n, y_n)I_1(x_n, y_n) - \sum_{n=1}^{N} aI_1(x_n, y_n)^2 - \sum_{n=1}^{N} bI_1(x_n, y_n). \tag{2.20}$$

Solving these equations, we obtain

$$a = \frac{n\sum_{n=1}^{N} I_2(x_n, y_n)I_1(x_n, y_n) - \sum_{n=1}^{N} I_2(x_n, y_n)\sum_{n=1}^{N} I_1(x_n, y_n)}{n\sum_{n=1}^{N} I_1(x_n, y_n)^2 - (\sum_{n=1}^{N} I_1(x_n, y_n))^2}, \tag{2.21}$$

$$b = \frac{\sum_{n=1}^{N} I_2(x_n, y_n) - a\sum_{n=1}^{N} I_1(x_n, y_n)}{N}. \tag{2.22}$$

We manually picked the observations (for $n = 1, \ldots, N$) from I_1 and I_2 (from the unchanged areas). When we subtract I_2 from $\hat{I}_2$ as $I_d(x, y) = I_2(x, y) - \hat{I}_2(x, y)$, we expect to find changes originating from land cover. When we apply this method to the normalized I_2 ($\tilde{I}_2$), we further eliminate the insignificant changes that still remain after normalization. Consequently, this technique gives slightly better performance compared to image differencing.

We provide the difference image obtained by image regression using our Adana image test set in Fig. 2.5a. We provide the thresholded version of this image in Fig. 2.5b. As in the previous sections, we benefit from Kapur's method in threshold selection. As can be seen, the change map obtained is similar to image differencing.

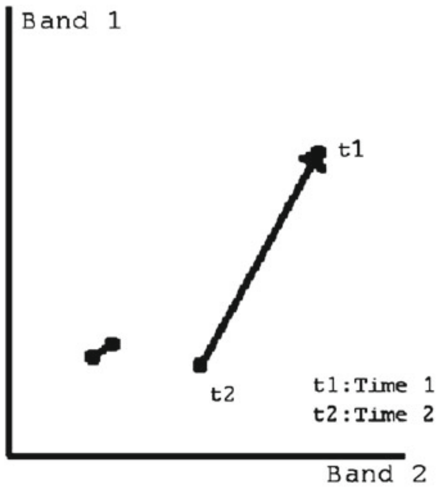


Fig. 2.6 Unchanged and changed pixel vectors in a 2-D spectral space

2.4 Change Vector Analysis

Change Vector Analysis (CVA) is a technique where multiple image bands can be analyzed simultaneously. As its name suggests, CVA does not only function as a change detection method, but also helps analyzing and classifying the change. In CVA, pixel values are vectors of spectral bands. Change vectors (CV) are calculated by subtracting vectors pixelwise as in image differencing. The magnitude and direction of the change vectors are used for change analysis. In Fig. 2.6, a changed pixel and an unchanged pixel are given in a two-band spectral space.

The change vector magnitude can indicate the degree of change. Thus, it can be used for change and no-change classification. Under ideal conditions, such as perfect image registration and normalization, unchanged pixel magnitudes must be equal to zero. However, this is not the case in practice. Therefore, thresholding must be applied to the change magnitude. While the change vector magnitude behaves like a multi-band version of the image differencing, the change direction gives us additional information about the change type. This is often more valuable than the amount of change, since in most applications we are interested in a specific change type.

In practice, the number of CV directions are uncountable. Therefore, it is necessary to quantize the CV space and assign directions accordingly. A simple quantization of CV directions can be achieved by dividing the space by its main axes. In Fig. 2.7, a 2D CV space is quantized into four subsets (quadrants) by the axis of band 1 and band 2. For three band images, subsets can be octants. CVs can be assigned to subsets via signs of their components.

As mentioned earlier, CV directions can be used in classifying the change. By using subsets, we can determine 2^n classes of change for an n dimensional space. CVA

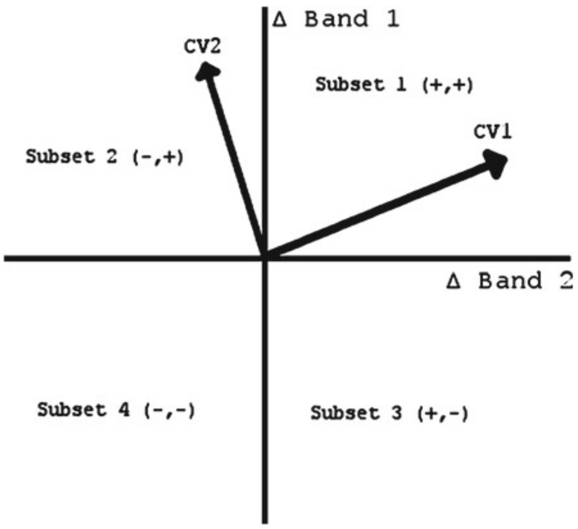


Fig. 2.7 Change vector space is divided into four subsets

can also be applied to transformed data such as Kauth-Thomas Transformation (KTT) (explained in Sect. 3.2) rather than to raw data. In the KTT space, a simultaneous increase in the greenness feature and decrease in the brightness feature indicates gain of vegetation. Therefore, in the change vector space of KTT bands, we can assign this change class (change toward vegetation) to the subsets where greenness is positive and brightness is negative.

CVA is used first by Malila [9] for change detection. He used the KTT with CVA and reported results for change of forestation. He used change directions to distinguish changes due to harvesting and regrowth. Johnson et al. [10] provided a comprehensive investigation of CVA. They provided the details to the implementation of CVA after a functional description. They reported that, CVA can be used in applications which require a full-dimensional data processing and analysis technique. They also found CVA to be useful for applications in which: the changes of interest and their spectral manifestation are not well-known a priori; changes of interest are known or thought to have high spectral variability; changes in both land cover type and condition may be of interest.

In Fig. 2.8a, we provide the change vector magnitude image which can be used for change and no-change classification. In Fig. 2.8b the change vector magnitude image is thresholded by Kapur's algorithm. This thresholded image also provides sufficient information on changed regions.

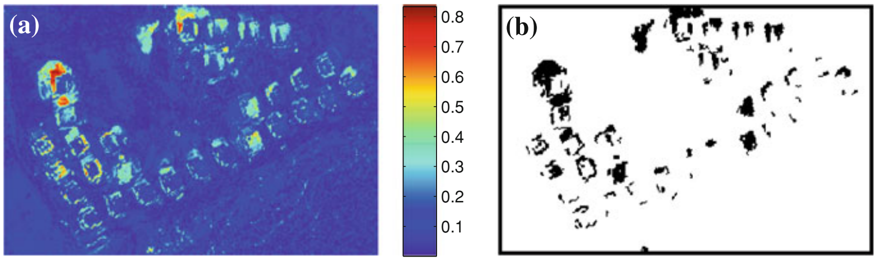


Fig. 2.8 CVA (in terms of magnitude value) applied to the Adana image set. **a** The magnitude value **b** Threshold Version

2.5 Median Filtering-Based Background Formation

In this method, we detect changes by subtracting the multi-temporal images from a reference image (background). This technique is widely used in video processing for detecting and analyzing motion. Although well known in video signal processing community, this is the first time median filtering is used for change detection in satellite images.

For this method, several background formation techniques are proposed in the literature. In this study, we use temporal median filtering for background formation. It is defined as

$$I_{bg}(x, y) = \text{Med}(I_1(x, y), I_2(x, y), \ldots, I_N(x, y)), \tag{2.23}$$

where I_{bg} is the background image and $(I_1, I_2, \ldots, I_N)$ are the images from times $(t_1, t_2, \ldots, t_N)$.

Parameters of the median operator are pixel values from the same spatial location (x, y). Median is calculated by choosing the middle element of the sorted array of its parameters. This leads to the removal of outliers (impulsive or salt and pepper noise) from the input pixel set. This characteristic of the median filter helps us to find a pixel value for every spatial location which is equal or approximately equal to the majority of the elements of the temporal pixel set. We expect to find the change by subtracting each image from the background, thus enchanting deviations from the median value.

We provide our multi-temporal images for background formation in Fig. 2.9. We also provide the median filtering result in Fig. 2.10.

Using the extracted background image, we obtain four difference images as in Fig. 2.11. We also provide the thresholded version of these images in Fig. 2.12. As in the previous sections, we used Kapur's algorithm in finding the threshold value. As can be seen in Fig. 2.12, majority of the significant changes are gathered in the first and last images in the series. These correspond to the two extremes of the time interval.

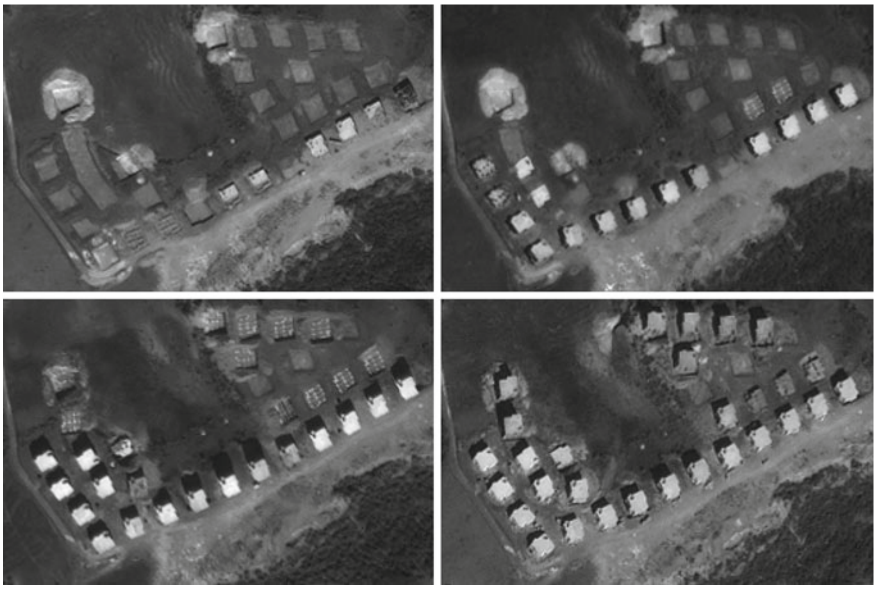
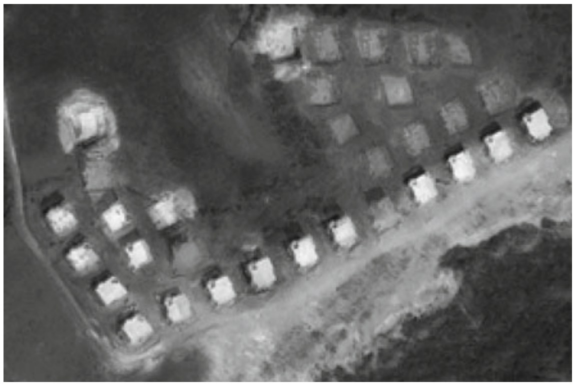


Fig. 2.9 Multi-temporal images used in background formation

Fig. 2.10 Background image generated by median filtering



2.6 Pixelwise Fuzzy XOR Operator

The last method for pixelwise change detection is a novel contribution to the community. In this method, the binary XOR operation is taken as a benchmark. Its fuzzy version is used for change detection. Our rationale here is as follows. Assume that we have two binary images (composed of only ones and zeros) and we want to detect the changed pixels in these. Each pixel $p(x, y)$ in a binary image B is valued according to a characteristic function β_B, which could also be called as the "whiteness" function defined as

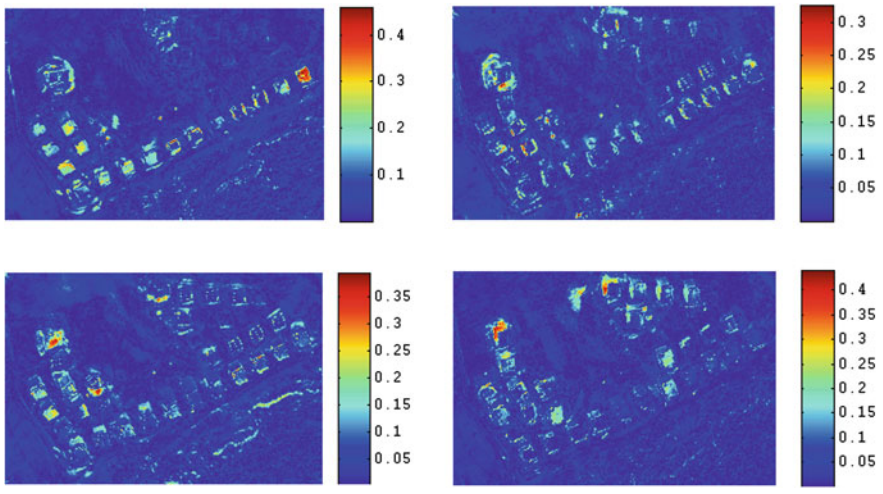


Fig. 2.11 Difference images for each sample generated by subtracting each sample from the background image

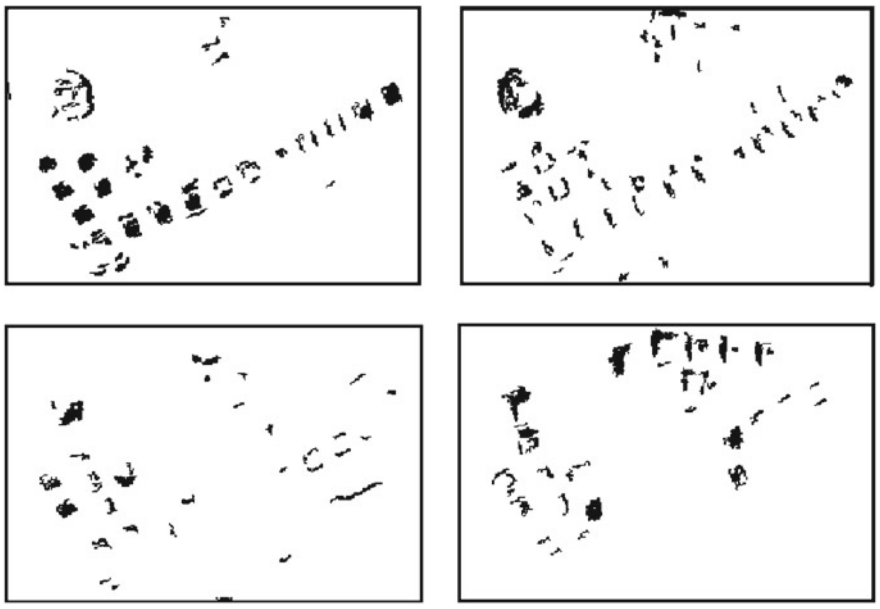


Fig. 2.12 Difference images are thresholded by Kapur's algorithm

$$p(x, y) = \beta_{\mathrm{B}}(x, y) = \begin{cases} 1, & \text{if } B(x, y) \text{ is white} \\ 0, & \text{otherwise.} \end{cases} \tag{2.24}$$

Between two pixels p_1 and p_2, at the same (x, y) coordinates of the two binary images B_1 and B_2, the existence of a change can only mean that either "p_1 is white and p_2 is not" or "p_1 is not white and p_2 is." This directly implies the XOR operation in binary logic. Hence the obvious solution to the change detection problem is XOR-ing the two binary images pixelwise as

$$C(x, y) = B_1(x, y) \oplus B_2(x, y). \tag{2.25}$$

This operation gives '0' for pixels having the same value in both images, and gives '1' for pixels having different values. Therefore, white pixels in the resulting binary image $C(x, y)$ represent the changed regions.

Unfortunately, this method cannot be applied to panchromatic or multispectral satellite imagery (having pixel values in a certain range). In order to perform a similar method on satellite imagery, we propose a fuzzy representation for these. We also benefit from the combination of fuzzy and crisp (binary) operations.

Panchromatic images are composed of pixels with values $p(x, y)$ in a certain range. Normalizing these values and mapping them to the range [0, 1] effectively translates the image into a fuzzy set, whose elements (pixels) have membership grades in proportion to their "whiteness." The membership grade $g(x, y)$ of each pixel $p(x, y)$ in the grayscale image G is thus defined by the fuzzy membership function μ_G as

$$g(x, y) = \mu_G(x, y) = \begin{cases} 1.00, & \text{if } G(x, y) \text{ is pure white} \\ \ldots & \ldots \\ 0.50, & \text{if } G(x, y) \text{ is gray} \\ \ldots & \ldots \\ 0.00, & \text{if } G(x, y) \text{ is pure black.} \end{cases} \tag{2.26}$$

Comparison of two binary images involves the crisp question "Are these two pixels different?." Whereas a fuzzy comparison of two panchromatic images involves the fuzzy question "How different are these two pixels?." Also the question of "Above what amount of difference shall the two pixels be labeled as changed?." The amount of difference between gray level values in the image domain directly corresponds to the difference between the degrees of membership in the fuzzy domain. For this particular application, the fuzzy complement (NOT) operation, defined as

$$\bar{g}(x, y) = \mu_{\bar{G}}(x, y) = 1 - g(x, y) \tag{2.27}$$

and the algebraic representation of the fuzzy intersection (AND) operation, defined as the multiplication of membership functions

$$\mu_{G_1 \cap G_2} = \mu_{G_1}(x, y)\mu_{G_2}(x, y) = g_1(x, y)g_2(x, y) \tag{2.28}$$

were used to obtain a fuzzy difference metric [11].

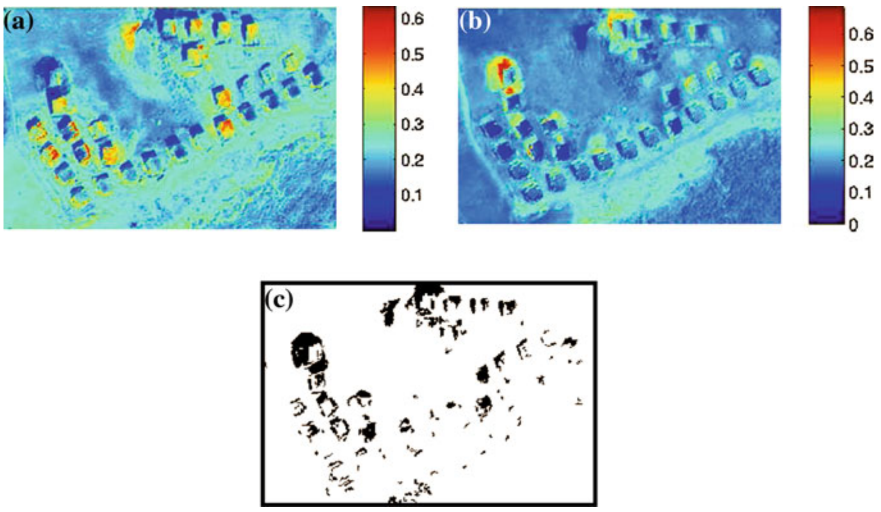


Fig. 2.13 Fuzzy XOR applied to the Adana image set. **a** Fuzzy AND $(g_1(x, y)\bar{g}_2(x, y))$ **b** Fuzzy AND $(\bar{g}_1(x, y)g_2(x, y))$ **c** Thresholded version

In a manner similar to the binary case, the measure of change between two pixels p_1 and p_2 is given by the degree of truth of the following statement: either "p_1 is lighter AND p_2 is darker" OR "p_1 is darker AND p_2 is lighter"; which can be rephrased as, either "p_1 has a high membership grade AND p_2 has a low membership grade" OR "p_1 has a low membership grade AND p_2 has a high membership grade."

Considering that "having a low membership grade" is the opposite of "having a high membership grade," the former statement's degree of truth is the complement of the latter's, and the degree of truth in "having a high membership grade" is equivalent to the membership grade $g(x, y)$ itself. Consequently, the above fuzzy rule can be formulated as

$$C(x, y) = \mu_{(G_1 \cap \bar{G}_2) \cup (\bar{G}_1 \cap G_2)}(x, y) = (g_1(x, y)\bar{g}_2(x, y)) \cup (\bar{g}_1(x, y)g_2(x, y)). \tag{2.29}$$

The fuzzy value $C(x, y)$ represents the measure of change between two images at the coordinate (x, y). The decision of a significant change can be made by means of applying an appropriate threshold and converting $C(x, y)$ to a crisp YES/NO value. Experiments have shown that, the results from the two fuzzy AND operations are distributed in a way that automatically indicates an appropriate threshold for defuzzification. More explicitly, threshold values are obtained for both fuzzy AND operations from $\tau = \text{argmax}(H_a) + 2\sigma_a$. Here, H_a is the histogram of the corresponding fuzzy AND operation and σ_a is the standard deviation of the corresponding fuzzy AND operation. In fact, applying this threshold and converting the fuzzy AND results to a crisp binary value, and then combining them with the binary OR operator yielded better results in detecting changed regions in satellite images. Therefore, the proposed method was eventually established as an ensemble of both fuzzy and binary logic operations.

We provide the images obtained by fuzzy AND operations using our Adana image set in Fig. 2.13a and Fig. 2.13b. We provide the thresholded version after finding the $C(x, y)$ in Fig. 2.13c. As can be seen, the resulting image shows the changed region fairly well.

References

1. Singh, A.: Review article: Digital change detection techniques using remotely-sensed data. Int. J. Remote Sens. **10**(6), 989–1003 (1989)
2. Rosin, P.L., Ioannidis, E.: Evaluation of global image thresholding for change detection. Pattern Recognition Lett. **24**(14), 2345–2356 (2003)
3. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. Syst. Man Cybern. **9**(1), 62–66 (1979)
4. Kapur, J.N., Sahoo, P.K., Wong, A.K.C.: A new method for gray-level picture thresholding using the entropy of the histogram. Comput. Vis. Graph. Image Process. **29**(3), 273–285 (1985)
5. Devore, J.: Probability and Statistics for Engineering and Sciences. 6 edn. Thompson (2004)
6. Griffiths, G.H.: Monitoring urban change from Landsat TM and Spot satellite imagery by image differencing. In: Proceedings of the 1988 International Geoscience and Remote Sensing Symposium, vol. 1, (1988)
7. Saksa, T., Uuttera, J., Kolstrom, T., Lehikoinen, M., Pekkarinen, A., Sarvi, V.: Clear-cut detection in boreal forest aided by remote sensing. Scandinavian J. For. Res. **18**(6), 537–546 (2003)
8. Lu, D., Mausel, P., Batistella, M., Moran, E.: Land-cover binary change detection methods for use in the moist tropical region of the Amazon: A comparative study. Int. J. Remote Sens. **26**(1), 101–114 (2005)
9. Malila, W.A.: Change vector analysis: An approach for detecting forest changes with Landsat. In: LARS Symposia, p. 385 (1980)
10. Johnson, R.D., Kasischke, E.S.: Change vector analysis: A technique for the multispectral monitoring of land cover and condition. Int. J. Remote Sens. **19**(3), 411–426 (1998)
11. Klir, G.J., Yuan, B.: Fuzzy Sets and Fuzzy Logic Theory and Applications. Prentice Hall, New York (1995)