Chapter 7
Pixel Fusion

Abstract. The subject of this chapter is image fusion techniques which rely on simple pixel-by-pixel operations. The techniques include the basic arithmetic operations, logic operations and probabilistic operations as well as slightly more complicated mathematical operations. The image values include pixel gray-levels, feature map values and decision map labels. Although more sophisticated techniques are available, the simple pixel operations are still widely used in many image fusion applications.

7.1 Introduction

In this chapter we consider fusion techniques which rely on simple pixel operations on the input image values. We assume the input images are spatially and temporally aligned, semantically equivalent and radiometrically calibrated. We start with the image fusion of K input images $I_1, I_2, \ldots, I_K$ using a simple arithmetic addition operator.

7.2 Addition

Addition which is probably the simplest fusion operation. It works by estimating the average intensity value of the input images $I_k, k \in \{1, 2, \ldots, K\}$, on a pixel-by-pixel basis. If $\widetilde{I}(m,n)$ denotes the fused image at the pixel (m,n), then

$$\widetilde{I}(m,n) = \frac{1}{K} \sum_{k=1}^{K} I_k(m,n) \; . \tag{7.1}$$

Although extremely simple, (7.1) is widely used if the input images are of the same modality.

The technique assumes semantic alignment and requires very accurate spatial and radiometric alignment. The technique has the advantage of suppressing any noise which is present in the input images. The following example illustrates how the pixel addition technique reduces image noise in a video sequence.

Example 7.1. Video Noise Averaging [3]. We consider an efficient method for video denoising. Although we can apply static image denoising methods to the case of image sequences we can do much better by including temporal information (inter-frame methods). This temporal information is crucial since our perception is very sensitive to temporal distortions like edge displacement: the disregard of temporal information may lead to inconsistencies in the result.

The input to the denoising algorithm is a video sequence of $M \times N$ images $I_k, k \in \{1,2,\ldots\}$. We partition each image I_k into a disjoint set of horizontal lines $L_k^{(i)}$. For each line $L_k^{(i)}$ we consider the family of lines which are close to $L_k^{(i)}$ in the same image and in the neighouring images. We warp each of these lines so they match with $L_k^{(i)}$. Let $\phi(L_l^{(j)})$ denote the warped version of the line $L_l^{(j)}$ onto the line $L_k^{(i)}$. We then obtain a denoised version of $L_k^{(i)}$ by performing an average of the lines $\phi(L_l^{(j)})$.

The pixel average technique has the disadvantage that it tends to suppress salient image features producing a low contrast image with a "washed-out" appearance. This effect can be alleviated, to some extent, by using a linear weighted average of the input images:

$$\widetilde{I}(m,n) = \sum_{k=1}^{K} w_k I_k(m,n) \Big/ \sum_{k=1}^{K} w_k \,, \tag{7.2}$$

where w_k are pre-selected scalars which are chosen so that each input image contributes an "optimal" amount towards the fused image. For instance, when fusing thermal and electro-optical sensors we may assign larger weights to the warmer or the cooler pixels of the thermal image or we may assign larger weights to those pixels whose intensities are much different from its neighbors. In some applications we estimate the weights w_k using the expectation-maximization (EM) algorithm (see Ex. 7.2).

Instead of pre-selecting the weights w_k we may allow the weights to vary automatically according to the amount of information contained in I_k. One method of defining the weights w_k is to use the method of principal component analysis (PCA) (Sect. 9.2).

However, notwithstanding how the weights are chosen, pixel averaging will tend to reduce the contrast of an object if in one image the object appears with a certain contrast and in another image the object appears with the opposite contrast.

7.2.1 Robust Averaging

Instead of using the arithmetic mean we may use robust equivalents which are robust against outliers. Two such operators are the median operator and the trimmed mean operator:

Median operator

$$\widetilde{I}(x,y) = \text{med}_k \left(I_k(x,y) \right) \ .$$

Trimmed mean operator

$$\widetilde{I}(x,y) = \frac{1}{K-2\alpha} \sum_{k=\alpha+1}^{K-\alpha} I_{(k)}(x,y) \ ,$$

where $I_{(k)}(x,y) = I_l(x,y)$ if $I_l(x,y)$ is the lth largest gray-level at (x,y) and α is a small constant. We often set $\alpha = \lfloor K/20 \rfloor$.

7.3 Subtraction

Subtraction is the complement to addition and is used as a simple fusion operator in change detection algorithms. These algorithms apply the subtraction operator pixel-by-pixel to generate a signed difference image D:

$$D(x,y) = I_1(x,y) - I_2(x,y) \ ,$$

where I_1 and I_2 are two input images which have been carefully aligned. The difference image is then thresholded to create a change map $B(x,y)$, where

$$B(x,y) = \begin{cases} 1 \text{ if } |D(x,y)| > t \ , \\ 0 \text{ otherwise} \ . \end{cases}$$

The threshold t may be constant over the image D or it may vary from pixel to pixel. The following example illustrates the Bayesian approach to change detection

Example 7.2. Unsupervised Change Detection [2, 4]. Given a difference image we write it as a one-dimensional vector $\mathbf{D} = (D(1), D(2), \ldots, D(M))^T$. We assume the probability density of the difference values, $P(D)$, can be modeled as a mixture of $K = 2$ components: one component corresponding to the class c_1 of "change" pixels and the other component corresponding to the class c_2 of "no-change" pixels:

$$P(|D) = P(c_1)p(D|c_1) + P(c_2)p(D|c_2)\,,$$
$$= \sum_{k=1}^{K} W_k p(c_k|D)\,,$$

where $W_k = P(c_k)$ is the *a priori* probability of the class $c_k, k \in \{1,2\}$. The standard approach for finding the *a posteriori* probability $p(c_k|D(m))$ is the expectation-maximization (EM) algorithm. We assume the likelihood $p(D(m)|c_k)$ is Gaussian:

$$p(D(m)|c_k) = \frac{1}{\sigma_k\sqrt{2\pi}} \exp -\frac{1}{2}\left(\frac{D(m)-\mu_k}{\sigma_k}\right)^2,$$

where μ_k and σ_k are, respectively, the mean and standard deviation of the kth Gaussian distribution. Then the EM algorithm iteratively updates the *a posteriori* probability distribution $p(D(m)|c_k)$ that $D(m)$ was generated by the kth mixture component, the *a priori* class probabilities W_k, and the Gaussian parameters μ_k and σ_k. Each iteration t consists of two steps:

E-step. Update the *a posteriori* probability $p(D(m)|c_k)$:

$$p^{(t+1)}(c_k|D(m)) = W_k^{(t)} p(D(m)|\mu_k^{(t)}, \sigma_k^{(t)}) \Big/ \sum_{h=1}^{K} W_h^{(t)} p(D(m)|\mu_h^{(t)}, \sigma_h^{(t)})\,.$$

M-step. Update the maximum likelihood estimates of the parameters $W_k^{(t)}, \mu_k^{(t)}$ and $\sigma_k^{(t)}$ for each component $k, k \in \{1,2,\ldots,K\}$:

$$W_k^{(t+1)} = \frac{1}{M}\sum_{m=1}^{M} p^{(t+1)}(c_k|D(m))\,,$$
$$\mu_k^{(t+1)} = \frac{\sum_{m=1}^{M} p^{(t+1)}(c_k|D(m))D(m)}{\sum_{m=1}^{M} p^{(t+1)}(c_k|D(m))}\,,$$
$$(\sigma_k^2)^{(t+1)} = \frac{\sum_{m=1}^{M} p^{(t+1)}(c_k|D(m))\left(D(m)-\mu_k^{(t+1)}\right)^2}{\sum_{m=1}^{M} p^{(t+1)}(c_k|D(m))}\,.$$

After several iterations the *a posteriori* probabilities $p^{(t)}(c_k|D(m))$ and the parameters $W_k^{(t)}$, $\mu_k^{(t)}$ and $\sigma_k^{(t)}$ converge to their final values. The $D(m)$ are then assigned to the class c_k with maximum *a posteriori* probability:

$$c_{opt} = \arg\max_k P(c_k|D(m))\,,$$
$$= \arg\max_k (P(c_k)p(D(m)|c_k))\,.$$

We may generalize the above procedure by assuming generalized Gaussian likelihoods [2].

Example 7.3. Mixture of Generalized Gaussian Distributions [2]. Ref. [2] suggests that a better model for the likelihood $p(D(m)|c_k)$ is a generalized Gaussian distribution:

$$p(D(m)|\mu_k,\sigma_k,\alpha_k) = \frac{\lambda_1(\alpha_k)}{\sigma_k}\exp\left(-\lambda_2(\alpha_k)\left|\frac{D(m)-\mu_k}{\sigma_k}\right|^{\alpha_k}\right),$$

where

$$\lambda_1(\alpha_k) = \frac{\alpha_k\Gamma(3/\alpha_k)^{1/2}}{2\Gamma(1/\alpha_k)^{3/2}},$$
$$\lambda_2(\alpha_k) = \left(\frac{\Gamma(3/\alpha_k)}{\Gamma(1/\alpha_k}\right)^{\alpha_k/2}.$$

The advantage of using the generalized Gaussian distribution is that by changing α_k we may change the shape of $p(D(m)|\mu_k,\sigma_k,\alpha_k)$. For example, $p(D(m)|\mu_k,\sigma_k,\alpha_k)$ assumes, respectively, the form of an impulsive, Laplacian, Gaussian and uniform distribution as α_k adopts the values $0,1,2$ and ∞.

The EM algorithm [6] for the generalized Gaussian model is the same as the standard EM algorithm given above apart from an addition to the M-step, where we update the shape parameter α_k. In order to update α_k we first update the kurtosis of the distribution:

$$\kappa_k^{(t+1)} = \frac{\sum_{m=1}^{M} p^{(t)}(c_k|D(m))\left(D(m)-\mu_k^{(t+1)}\right)^4}{\left(\sigma_k^{(t+1)}\right)^4\sum_{m=1}^{M} p^{(t)}\left(c_k|D(m)\right)} - 3,$$

and then calculate $\alpha_k^{(t+1)}$ using the following relationship:

$$\kappa_k^{(t+1)} = \frac{\Gamma\left(5/\alpha_k^{(t+1)}\right)\Gamma\left(1/\alpha_k^{(t+1)}\right)}{\Gamma\left(3/\alpha_k^{(t+1)}\right)^2} - 3.$$

Apart from the EM algorithm, Chapt 12 contains a review of many formulas and algorithms used to threshold D.

The difference image is sensitive to noise and variations in illuminations. In general, therefore, difference images are only used if the input images were captured with the same sensor under similar conditions, i. e. the photometric transformation between corresponding pixel gray-levels values should be close to identity.

In the next two sections we consider the multiplication and division operators. In general, these operations are much less widely used than the addition and the subtraction operations.

7.4 Multiplication

Multiplication and division are not widely used as image fusion operators. However one important image fusion application where multiplication is used is Brovey pan-sharpening.

Example 7.4. Brovey Pan Sharpening [11]. The Brovey transform is a simple method for combining multi-spectral image with a panchromatic image. The technique is limited to three spectral bands which we identify with the R, G and B channels. The transform is defined as follows

$$\begin{pmatrix} R_{brovey} \\ G_{brovey} \\ B_{brovey} \end{pmatrix} = \begin{pmatrix} R \\ G \\ B \end{pmatrix} + (P-I) \begin{pmatrix} R/P \\ G/P \\ B/P \end{pmatrix} ,$$

where $I = (R+G+B)/3$ and P denotes the panchromatic image.

7.5 Division

The following example illustrates shadow detection by computing a ratio map R.

Example 7.5. Shadow Detection in Color Aerial Images [5, 10]. Shadow detection algorithms are very important in many image fusion algorithms. Generally these algorithms work by selecting a region which is darker than its neighboring regions but has similar chromatic properties. For *RGB* color aerial images we may detect shadows as follows. We transform the *RGB* input color image into a intensity-hue-saturation (*IHS*) color space (16.1–16.3). Then at each pixel (m,n) we form a ratio map $R_e(m,n)$ by comparing the hue of the pixel to the intensity of the pixel. The value $R_e(m,n)$ measures the likelihood of the pixel (m,n) being in shadow.

In [5] the ratio map is defined as follows (assuming 24-bit *RGB* input picture):

$$R_e(m,n) = \text{round}\left(\frac{H_e(m,n)}{I_e(m,n)+1}\right) ,$$

where

$$I_e(m,n) = \frac{R(m,n)+G(m,n)+B(m,n)}{3} ,$$

$$H_e(m,n) = \frac{255(\tan^{-1}(H(m,n)+\pi)}{2\pi} .$$

A shadow map is then formed by thresholding R_e:

$$S_e(m,n) = \begin{cases} 1 \text{ if } R_e(m,n) > T \ , \\ 0 \text{ otherwise} \ , \end{cases}$$

where $S_e(m,n) = 1$ denotes a shadow pixel at (m,n).

7.6 Feature Map Fusion

In feature map fusion we fuse together the feature maps $F_k, k \in \{1,2,\ldots,K\}$. The following example illustrates the fusion of multiple feature maps which are semantically equivalent. The fusion operator used is a simple arithmetic average operator applied separately to each pixel.

Example 7.6. Fusion of Multiple Edge Maps. Given an input image I we perform edge extraction using several edge operators e. g. Sobel, Canny and zero-crossing. The operators all measure the same phenomena ("presence of an edge") and are therefore semantically equivalent. The feature maps still require radiometric calibration onto a common scale. If $F_{sobel}(m,n)$, $F_{canny}(m,n)$ and $F_{zero}(m,n)$ denote the feature maps after calibration, then we may fuse the maps together using a simple arithmetic mean operator:

$$\widetilde{F}(m,n) = \frac{1}{3}(F_{sobel}(m,n) + F_{canny}(m,n) + F_{zero}(m,n)) \ .$$

The following example illustrates the fusion of multiple feature maps which do not measure the same phenomena but which have been made semantically equivalent by transforming them into probabilistic, or likelihood, maps.

Example 7.7. Multi-Feature Infra-red Target Detection in an Input Image [12]. We continue with Ex 5.3. We consider the detection of a small target in an infra-red input image I. At each pixel (m,n) in I we test for the presence of a target by extracting K features $F_k, k \in \{1,2,\ldots,K\}$. The features do not measure the same phenomena. However, according to the theory of infra-red target detection, they are all related to the presence of an infra-red target. We make the F_k semantically equivalent by transforming $F_k(m,n)$ into a probability $p_k(m,n)$ which measures the probability, or likelihood, of an infra-red target being present at (m,n). Let $\widetilde{p}(m,n)$ be the fused probability, or likelihood, that an infra-red target is present at (m,n). Then, methods for fusing the $p_k(m,n)$ include:

Mean

$$\widetilde{p}(m,n) = \frac{1}{K}\sum_{k=1}^{K} p_k(m,n) \; .$$

Product

$$\widetilde{p}(m,n) = \prod_{k=1}^{K} p_k(m,n) \; .$$

Minimum

$$\widetilde{p}(m,n) = \min_k p_k(m,n) \; .$$

Median

$$\widetilde{p}(m,n) = \text{median}(p_k(m,n)) \; .$$

Maximum

$$\widetilde{p}(m,n) = \max_k p_k(m,n) \; .$$

Another method for feature map fusion is rank fusion. The following example illustrates rank fusion for face recognition.

Example 7.8. Face Recognition Using Rank Fusion [7]. Given an input image I we extract several different features $F_k, k \in \{1,2,\ldots,K\}$. *Note*: In this example, the features F_k refer to the entire image I and not just to a pixel (x,y).

We make the F_k semantically equivalent by transforming each F_k into a multiple set of L likelihoods $p_k(l), l \in \{1,2,\ldots,L\}$, where $p_k(l)$ is the probability that the feature F_k belongs to the lth individual (i. e. belongs to class l). If the $p_k(l)$ are reliable we may fuse them together using any of the operators discussed in Ex. 7.7. However, in many cases we can only rely on the rank of $p_k(l)$ and not on the actual value of $p_k(l)$. In this case we transfer each $p_k(l)$ into a rank $r_k(l)$, where

$$r_k(l) = r \qquad \text{if } p_k(l) \text{ is } r\text{th largest likelihood} \; .$$

The optimum classification of the input image I is then

$$l^* = \arg\min_l \big(\widetilde{r}(l)\big) = \arg\min_l \big(\sum_{k=1}^{K} r_k(l)\big) \; , \tag{7.3}$$

where $\widetilde{r}$ denotes the sum of the ranks for class l:

$$\widetilde{r}(l) = \sum_{k=1}^{K} r_k(l) \; .$$

A simple numerical example illustrating the technique is as follows: We have three features: face matching (F_1), ear matching (F_2) and signal matching (F_3). The ranks obtained for each feature are:

$$\begin{array}{llll} r_1(1)=3, & r_1(2)=1, & r_1(3)=4, & r_1(4)=2\,, \\ r_2(1)=2, & r_2(2)=1, & r_2(3)=4, & r_2(4)=5\,, \\ r_3(1)=1, & r_3(2)=2, & r_3(3)=3, & r_3(4)=4\,. \end{array}$$

The fused ranks are:

$$\widetilde{r}_1=6, \qquad \widetilde{r}(2)=4, \qquad \widetilde{r}(3)=11, \qquad \widetilde{r}(4)=11\,.$$

The optimal classification is $l^*=\arg\min_l(\widetilde{r}(l))=\arg\min(6,4,11,11)=2$.

7.7 Decision Fusion

In decision fusion we fuse together a set of decision images, or label maps, $D_k, k \in \{1,2,\ldots,K\}$. The D_k are themselves obtained by performing a decision procedure on all pixels (m,n) in the input image I_k. For each pixel (m,n), $D_k(m,n)$ is a label l which may be any identifying name or symbol. We shall find it convenient to associate each label l with an integer chosen from $l \in \{1,2,\ldots,L\}$. It should, however, be emphasized, that in general different labels have different meaning and this must be taken into account when the D_k are fused together.

We shall start by considering the case when the $D_k, k \in \{1,2,\ldots,K\}$, are semantically equivalent, i. e. a label l in D_h has the same semantic interpretation as the label l in $D_k, h \neq k$. Then in Sects. 7.7.3 and 7.7.4 we consider the more complicated case, when the D_k are no longer semantically equivalent.

The simplest way of fusing D_k which are semantically equivalent is to fuse the D_k using the majority-vote rule:

$$\widetilde{D}(m,n)=l \qquad \text{if} \qquad \sum_{k=1}^{K}\delta\left(D_k(m,n),l\right) \geq \frac{1}{2}\,,$$

or the weighted majority-vote rule:

$$\widetilde{D}(m,n)=l \qquad \text{if} \qquad \sum_{k=1}^{K} w_k\,\delta\left(D_k(m,n),l\right) \geq \sum_{k=1}^{K} w_k/2\,, \tag{7.4}$$

where

$$\delta(a,b)=\begin{cases} 1 \text{ if } a=b\,, \\ 0 \text{ otherwise}\,. \end{cases}$$

In some applications the weights in (7.4) may be obtained from the D_k themselves (see e. g. Ex. 10.7). In other cases we may obtain the weights using the expectation-maximization (EM) algorithm (see Ex. 7.2 and Chapt. 21).

The majority-vote and weighted majority-vote rules are widely used for decision fusion. They are simple to implement and robust against noise and outliers (see Fig. 7.1).

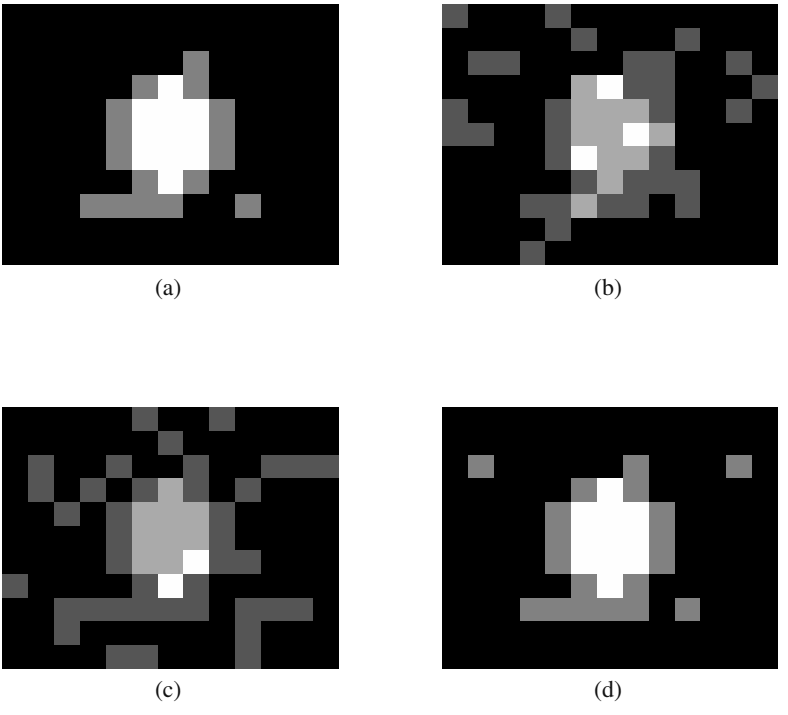

Fig. 7.1 **(a)** Shows a decision map D_1 in which the structures are contiguous. **(b)** and **(c)** Show decision maps D_2 and D_3. These are the same as D_1 and with additive noise. **(d)** Shows the decision map $\widetilde{D}$ obtained by majority-vote fusion of D_1, D_2 and D_3. In this case, the fused map maintains the contiguous nature of the original input maps.

The majority-vote and the weighted majority-vote rules do not, however, take into account pixel-to-pixel correlations. In some cases this may lead to a fragmentation of structures which are contiguous in the input images (see Fig. 7.2). To prevent the fragmentation we must include the effect of pixel-to-pixel correlations. One way of doing this is to use a Markov random field which is discussed in Chapt. 17. Alternatively, if the fragmentation arises because the D_k are not perfectly aligned, then we may use the shape-based averaging algorithm.

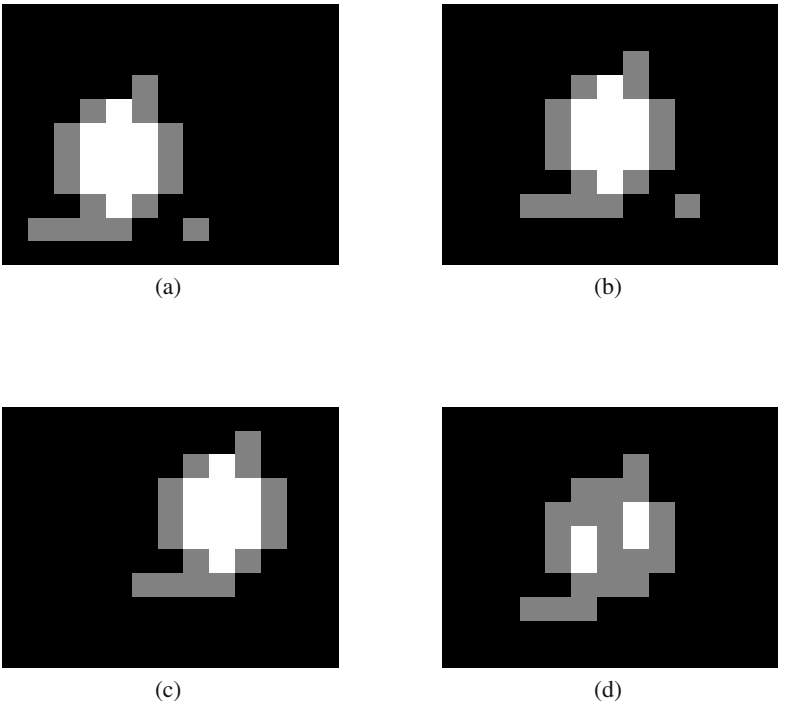

Fig. 7.2 **(a)** Shows a decision map D_1 in which the structures are contiguous. **(b)** and **(c)** Show decision maps D_2 and D_3. These are the same as D_1 but are slightly displaced up and down, left and right. **(d)** Shows the decision map $\widetilde{D}$ obtained by majority-vote fusion of D_1, D_2 and D_3. In this case, the fused map does not maintain the contiguous nature of the input maps.

7.7.1 Shape-Based Averaging

Shape-based averaging [8] was introduced specifically to address the above fragmentation problem. The basis of the algorithm is the *signed* distance transform which assigns to each pixel in the decision map its signed distance from the nearest "feature" pixel. If we regard any pixel with a label l as a feature pixel, then we may decompose a decision map $D_k(m,n)$ into L signed distance transforms $s_k(m,n|l), l \in \{1,2,\ldots,L\}$. Let $d_k(m,n|l)$ be the smallest Euclidean distance from (m,n) to a pixel with a label l and let $d_k(m,n|\widetilde{l})$ be the smallest Euclidean distance from (m,n) to a pixel with a label not equal to l, then the signed distance map $s_k(m,n|l)$ is defined as:

$$s_k(m,n|l) = d_k(m,n|l) - d_k(m,n|\widetilde{l}) \,. \tag{7.5}$$

According to (7.5) $s_k(m,n|l)$is negative if the pixel (m,n) lies inside the structure with label l, is positive if (m,n)lies outside the structure and is zero if, and only

if, (m,n) lies on the perimeter of the structure (see Ex. 7.9). For each label $l, l \in \{1,2,\ldots,L\}$, we calculate a mean signed distance map, $\bar{s}(m,n|l)$, by averaging the $s_k(m,n|l)$ over all k:

$$\bar{s}(m,n|l) = \frac{1}{K}\sum_{k=1}^{K} s_k(m,n|l) \; .$$

The value of the fused decision map $\widetilde{D}(m,n)$ (Fig. 7.3) is then defined as the label l which has the minimum $\bar{s}(m,n|l)$ value:

$$\widetilde{D}(m,n) = \arg\min_{l} \bar{s}(m,n|l) \; .$$

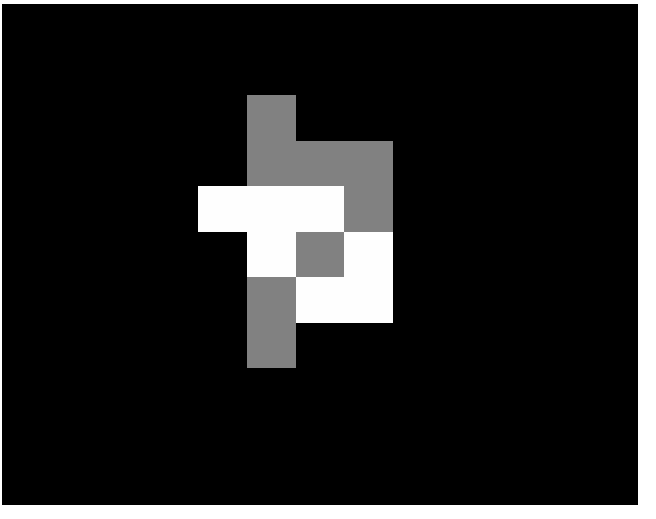

Fig. 7.3 Shows the decision map $\widetilde{D}$ which is obtained by shape-based averaging the decision maps D_1, D_2 and D_3 which appear in Fig. 7.2(a)-(c). Observe how shape-based averaging helps to preserve the contiguous nature of the input images (cf. Fig. 7.2(d)).

Example 7.9. Signed Distance Transform. Consider the following one-dimensional image D wth three labels A, B and C:

$$D = (\, A\ A\ B\ C\ C\ C\ B\ C\ B\ A\,)^T \; .$$

The corresponding distance transforms $d(i|l=C)$, $d(i|\widetilde{l}=C)$ are:

$$d(i|l=C) = (\,3\ 2\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 2\,)^T \; ,$$
$$d(i|\widetilde{l}=C) = (\,0\ 0\ 0\ 1\ 2\ 1\ 0\ 1\ 0\ 0\,)^T \; ,$$

and the signed distance transform $s(i|l=C) = d(i|l=C) - d(i|\widetilde{l}=C)$ is

$$s(i|l=C) = (\,3\;2\;1\;-1\;-2\;-1\;1\;-1\;1\;2\,)^T\,.$$

7.7.2 Similarity

Decision maps are often fused together by measuring their similarity. This is often used in pattern recognition problems. Given two decision maps D_1 and D_2, we declare D_1 and D_2 to represent the same visual scene or object if the similarity measure $S(D_1, D_2)$ is greater than some threshold T.

Example 7.10. Face Recognition Using a Local Binary Pattern [1]. A direct method for performing face recognition is to compare a given test image B with a collection of training images $A_k, k \in \{1,2,\ldots,K\}$, which belong to K different individuals. In order to measure the similarity $S(B,A_k)$ we must ensure that the test image B and the training images $A_k, k \in \{1,2,\ldots,K\}$, are radiometrically calibrated. One way of doing this is to use the local binary pattern (LBP) operator (Sect. 3.4) to convert B into a decision map D_B and A_k into a decision map D_k.

7.7.3 Label Permutation

We now consider decision fusion when the $D_k, k \in \{1,2,\ldots,K\}$, are not semantically equivalent. In many cases we may assume that, to a good approximation, there is an unknown one-to-one correspondence between the labels in the different D_k. In other words we assume that each label p in D_k corresponds to a single label q in D_h and vice versa. In this case, we may simply solve the label correspondence problem by permuting the labels $p, p \in \{1,2,\ldots,L_k\}$, in D_k until the overall similarity

$$\sum_{k=1}^{K}\sum_{h=1}^{K} S(\pi_k(D_k), \pi_h(D_h))\,,$$

is a maximum, where $\pi_k(D_k)$ denotes a permutation of the labels in D_k (see Ex. 5.5).

A convenient similarity measure for this purpose is the normalized mutual information NMI [9]:

$$NMI(D_k, D_h) = \sum_{p=1}^{L_k}\sum_{q=1}^{L_h} \widetilde{M}_{p,q} \log \frac{\widetilde{M}_{p,q}}{\widetilde{M}_p \widetilde{N}_q} \Bigg/ \sqrt{\sum_{p=1}^{L_k} \widetilde{M}_p \log(\widetilde{M}_p) \sum_{q=1}^{L_h} \widetilde{N}_q \log(\widetilde{N}_q)}$$

where $\widetilde{M}_p$ is the relative number of pixels in D_k with a label $p, p \in \{1,2,\ldots,L_k\}$, $\widetilde{N}_q$ is the relative number of pixels in D_h with a label $q, q \in \{1,2,\ldots,L_h\}$, and $M_{p,q}$ is the relative number of pixels which jointly have a label p in D_k and have a label q in D_q [1]. If $\widetilde{\pi}_k$ denotes the optimal permutation for D_k, then:

$$(\widetilde{\pi}_1, \widetilde{\pi}_2, \ldots, \widetilde{\pi}_K) = \arg \max_{\pi_1,\pi_2,\ldots,\pi_K} \sum_{k=1}^{K} \sum_{h=1}^{K} NMI(\pi_k(D_k), \pi_h(D_h)) \,. \tag{7.6}$$

Eq. (7.6) represents a difficult combinatorial optimization problem. However, greedy search techniques, including simulated annealing and genetic algorithm, may give an approximate solution in an acceptable time. Given the (approximate) optimal permutation $\widetilde{\pi}_k, k \in \{1,2,\ldots,K\}$, we may find $\widetilde{D}$ by applying the majority-vote rule to $\widetilde{\pi}_k(D_k)$:

$$\widetilde{D}(m,n) = l \qquad \text{if} \qquad \sum_{k=1}^{K} \delta\left(\widetilde{\pi}_k(D_k(m,n)), l\right) \geq \frac{K}{2} \,,$$

where

$$\delta(a,b) = \begin{cases} 1 \text{ if } a = b \,, \\ 0 \text{ otherwise} \,. \end{cases}$$

7.7.4 Co-associative Matrix

In this section we consider the fusion of K decision maps D_k when we do not, or cannot, solve the label correspondence problem: We suppose the D_k are of size $M \times N$ and are spatially aligned. We transform the D_k into a common representational format by converting them into co-associative matrices $A_k, k \in \{1,2,\ldots,K\}$:

$$A_k(i,j) = \begin{cases} 1 \text{ if } D_k(m_i,n_i) = D_k(m_j,n_j) \,, \\ 0 \text{ otherwise} \,. \end{cases}$$

Let $\widetilde{A}$ denote the result of fusing the A_k together. Then we define $\widetilde{D}$ as the decision map which corresponds to $\widetilde{A}$. This is illustrated in the next example.

Example 7.11. Mean Co-Association Matrix. The arithmetic mean is the simplest method for fusing co-association matrices:

$$\widetilde{A}(i,j) = \frac{1}{K} \sum_{k=1}^{K} A_k(i,j) \,.$$

[1] The relative number of pixels is a probability. Thus $\widetilde{M}_p = M_p/M$ and $\sum_p \widetilde{M}_p = 1$, where M_p is the number of pixels in D_k with a label p and M is the total number of pixels in D_k. Similarly, $\widetilde{N}_q = N_q/M$ and $\widetilde{M}_{p,q} = M_{p,q}/M$.

Given $\widetilde{A}$ we now search for a decision map $\widetilde{D}$ whose co-association matrix closely approximates $\widetilde{A}$. To do this we use spectral cluster algorithms. These algorithms use greedy search techniques and require an estimate of the number of clusters $\widetilde{L}$. A simple estimate of $\widetilde{L}$ [13] is:

$$\widetilde{L} = \min_{L} \Big(\sum_{i=1}^{L} \widetilde{\nu}_i > \alpha \sum_{i=1}^{MN} \widetilde{\nu}_i \Big) ,$$

where $\widetilde{\nu}_i$ is the ith largest eigenvalue of $\widetilde{A}$ and α is some fraction close to one. A reasonable value for α is $\alpha = 0.8$.

In Chapt. 20 we describe another method for fusing the $D_k, k \in \{1, 2, \ldots, K\}$, which does not require solving the label correspondence problem and which does not use the co-association matrix.

7.8 Software

CLUSTERPACK. A matlab toolbox for spectral clustering. Authors: A. Strehl and J. Ghosh [9].

SPECTRAL CLUSTERING TOOLBOX. A matlab toolbox for spectral clustering. Authors: Deepak Verma and M. Meila. The toolbox may be used to cluster the mean co-associative matrix.

References

1. Ahonen, T., Hadid, A., Pietikainen, M.: Face description with local binary patterns: application to face recognition. IEEE Trans. Patt. Anal. Mach. Intell. 28, 2037–2041 (2006)
2. Bazi, Y., Bruzzone, L., Melgani, F.: Image thresholding based on the em algorithm and the generalized Gaussian distribution. Patt. Recogn. 40, 619–634 (2007)
3. Bertalmio, M., Caselles, V., Pardo, A.: Movie denoising by average of warped lines. IEEE Trans. Image Process. 16, 2333–2347 (2007)
4. Bruzzone, L., Prieto, D.F.: Automatic analysis of the difference image for unsupervised change detection. IEEE Trans. Geosci. Remote Sens. 38, 1171–1182 (2000)
5. Chung, K.-L., Lin, Y.-R., Huang, Y.-H.: Efficient shadow detection of color aerial images based on successive thresholding scheme. IEEE Trans. Geosci. Remote Sens. 47, 671–682 (2009)
6. Hesse, C.W., Holtackers, D., Heskes, T.: On the use of mixtures of Gaussians and mixtures of generalized exponentials for modelling and classification of biomedical signals. In: IEEE Benelux EMBS Symposium (2006)
7. Monwar, M.M., Gavrilova, M.L.: Multimodal biometric system using rank-level fusion approaches. IEEE Trans. Syst. Man Cybernetics 39B, 867–878 (2009)
8. Rohlfing, T., Maurer Jr., C.R.: Shape-based averaging. IEEE Trans. Image Process. 16, 153–161 (2007)

9. Strehl, A., Ghosh, J.: Cluster ensembles - a knowledge reuse framework for combining multiple partitions. J. Mach. Learn. Res. 3, 583–617 (2002)
10. Tsai, V.J.D.: A comparative study on shadow compensation of color aerial images in invariant color models. IEEE Trans. Geosci. Remote Sens. 44, 16671–16671 (2006)
11. Tu, T.-M., Su, S.-C., Shyu, H.-C., Huang, P.S.: A new look at IHS-like image fusion methods. Inf. Fusion 2, 177–186 (2001)
12. Wang, Z., Gao, C., Tian, J., Lia, J., Chen, X.: Multi-feature distance map based feature detection of small infra-red targets with small contrast in image sequences. In: Proc. SPIE, vol. 5985 (2005)
13. Wang, X., Yang, C., Zhou, J.: Spectral aggregation for clustering ensemble. In: Proc. Int. Conf. Patt. Recog. (2008)