Chapter 4
Spatial Alignment

Abstract. The subject of this chapter is spatial alignment. In image fusion this is defined as the process of geometrically aligning two or more images of the same scene acquired at different times (multi-temporal fusion), or with different sensors (multi-modal fusion), or from different viewpoints (multi-view fusion). It is a crucial pre-processing operation in image fusion and its accuracy is a major factor in determining the quality of the output image. In order to keep our discussion focused we shall concentrate on the image registration of two input images, A and B, which we define as finding the transformation T which "optimally" maps spatial locations in the image B to the corresponding spatial locations in the image A.

4.1 Introduction

Let A and B denote two digital input images which we assume are derived from the same scene. The images will naturally have limited fields of view which will most likely be different. However, as the two images A and B are derived from the same scene we expect a relation to exist between the spatial locations in A and the spatial locations in B. If (u,v) denotes a pixel location in the reference image A and (x,y) denotes a pixel location in the floating image B, then the transformation T represents a mapping of every pixel location (x,y) in B into the corresponding location (u',v') in A [1]:

$$\begin{pmatrix} u' \\ v' \end{pmatrix} = T \begin{pmatrix} x \\ y \end{pmatrix} .$$

In general, the location (u',v') does not correspond to a pixel location in A. Let B' be the corresponding tranformed B image. The image B' is only defined at the points (u',v'), where by definition, $B'(u',v') = B(x,y)$. In order to convert $B'(u',v')$ into a digital image which is defined at the same pixel locations as A we apply an interpolation/resampling operation to $B'(u',v')$:

[1] The reader should note the subtle difference between (x,y) and (u',v'): (x,y) represents a discrete pixel location in B while (u',v') represents the corresponding floating spatial location in A. In general (u',v') does *not* correspond to a pixel location in A.

$$\widetilde{B}(u,v) \equiv R\left(B'(u',v')\right) , \tag{4.1}$$

where R is an appropriate resample/interpolation operator. In practice (4.1) is implemented by using the inverse transformation T^{-1} which maps pixels in A to their corresponding locations in B. The following example illustrates the concept of a nearest neighbor resample/interpolation operator.

Example 4.1. Nearest neighbour resample/interpolation algorithm. The simplest resample/interpolation algorithm is the nearest neighbor algorithm. Let (u,v) denote a pixel location in A. Suppose the corresponding location in B is $(x',y') = T^{-1}(u,v)$. In general (x',y') will not fall on a pixel location in B. Let $P_k = (x_k,y_k), k \in \{0,1,2,3\}$, denote the four pixel locations in B which surround the point (x',y') where P_0 is the point nearest to (x',y') (Fig. 4.1), then the nearest neighbor gray-level is $\widetilde{B}(u,v)$, where

$$\widetilde{B}(u,v) \equiv R\left(B'(u',v')\right) = B(x_0,y_0) .$$

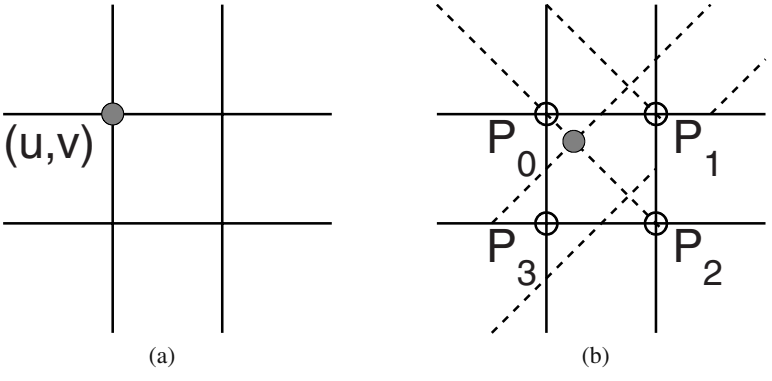

Fig. 4.1 Shows nearest neighbor interpolation. **(a)** Shows the reference image A and its grid lines as full-lines. A pixel location (u,v) is shown by a filled circle at the intersection of two grid-lines. **(b)** Shows the floating image B and its grid lines as full lines. Also shown (by dashed lines) are the inverse transformed grid lines of A. The filled circle shows the location of the inverse transformed point $(x',y') = T^{-1}(u,v)$.

4.2 Pairwise Transformation

The (pairwise) transformation

$$\begin{pmatrix} u' \\ v' \end{pmatrix} = T \begin{pmatrix} x \\ y \end{pmatrix} ,$$

Table 4.1 Spatial Transformations $T : (u', v')^T = T\left((x, y)^T\right)$

Name	Formula
Translation	$u' = x + a_1, v' = y + a_2.$
Similarity	$u' = a_1 x + a_2 y + a_3,\ v' = -a_2 x + a_1 y + a_4.$
Affine	$u' = a_1 x + a_2 y + a_3,\ v' = a_4 x + a_5 y + a_6.$
Perspective	$u' = (a_1 x + a_2 y + a_3)/(a_7 x + a_8 y + 1),$
	$v' = (a_4 x + a_5 y + a_6)/(a_7 x + a_8 y + 1).$
Polynomial	$u' = \sum a_{ij} x^i y^j,\ v' = \sum b_{ij} x^i y^j.$

is a mathematical relationship that maps a spatial location $(x, y)^T$ in one image to a new location, $(u', v')^T$, in another image. The choice of transformation is always a compromise between a smooth distortion and a distortion which achieves a good match. One way to ensure smoothness is to assume a low-order parametric form for the transformation [8, 23] such as that given in Table 4.1. In most applications the transformation T is chosen on the grounds of mathematical convenience. However, sometimes, we may have information regarding the physical processes which govern the formation of the pictures. In this case we may use physical arguments in order to *derive* the transformation T.

In many applications, the images also undergo local deformations. In this case, we cannot describe the alignment of two images using a single low-order transformation. In this case we use a composite transformation T, which consists of a low-order global transformation T_G and a local transformation T_L:

$$\begin{pmatrix} u' \\ v' \end{pmatrix} = T \begin{pmatrix} x \\ y \end{pmatrix} = T_G \begin{pmatrix} x \\ y \end{pmatrix} + T_L \begin{pmatrix} x \\ y \end{pmatrix} ,$$

where the parameters of T_L change with (x, y).

The thin-plate spline (TPS) is often used to model the composite transformation T.

4.2.1 Thin-Plate Splines

Mathematically, the TPS model for the composite transformation T is:

$$u' = a_1 + a_2 x + a_3 y + \sum_{m=1}^{M} \alpha_m r_m^2 \ln r_m^2 ,$$

$$v' = a_4 + a_5 x + a_6 y + \sum_{m=1}^{M} \beta_m r_m^2 \ln r_m^2 ,$$

where $(x_m, y_m), m \in \{1, 2, \ldots, M\}$, is a set of known anchor points and $r_m^2 = (x - x_m)^2 + (y - y_m)^2 + d^2$. Apart from the parameter d, the transformation T has six parameters, $a_1, a_2, \ldots, a_6$, corresponding to the global affine transformation T_G and

$2N$ parameters $(\alpha_m, \beta_m), m \in \{1, 2, \ldots, M\}$, corresponding to the local transformation T_L, and which satisfy the following constraints:

$$\sum_{m=1}^{M} \alpha_m = 0 = \sum_{m=1}^{M} \beta_m \,,$$
$$\sum_{m=1}^{M} x_m \alpha_m = 0 = \sum_{m=1}^{M} x_m \beta_m \,,$$
$$\sum_{m=1}^{N} y_m \alpha_m = 0 = \sum_{m=1}^{M} y_m \beta_m \,.$$

The TPS coefficients can be calculated using a least square solution [12]:

$$\begin{pmatrix} \alpha_1 & \beta_1 \\ \alpha_2 & \beta_2 \\ \vdots & \vdots \\ \alpha_M & \beta_M \\ a_2 & b_2 \\ a_3 & b_3 \\ a_1 & b_1 \end{pmatrix} = \begin{pmatrix} 0 & U(r_{12}) & \ldots & U(r_{1M}) & 1 & x_1 & y_1 \\ U(r_{21}) & 0 & \ldots & U(r_{2M}) & 1 & x_2 & y_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ U(r_{M1}) & U(r_{M2}) & \ldots & 0 & 1 & x_M & y_M \\ 1 & 1 & \ldots & 1 & 0 & 0 & 0 \\ x_1 & x_2 & \ldots & x_M & 0 & 0 & 0 \\ y_1 & y_2 & \ldots & y_M & 0 & 0 & 0 \end{pmatrix}^{-1} \begin{pmatrix} u'_1 & v'_1 \\ u'_2 & v'_2 \\ \vdots & \vdots \\ u'_M & v'_M \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix},$$

where $r_{ij}^2 = (x_i - u'_j)^2 + (y_j - v'_j)^2 + d^2$ and $U(r) = r^2 \ln r^2$. For further details concerning the estimation of the TPS parameters see [22].The following example illustrates the use of a TPS to model the warping of a fingerprint.

Example 4.2. Fingerprint Warping Using a Thin-plate spline [18]. The performance of a fingerprint matching system is affected by the nonlinear deformations introduced in the fingerprint during image acquisition This nonlinear deformation is represented using a global affine transformation T_G and a local transformation T_L. We use a TPS function [22] to represent the composite transformation:

$$u' = a_1 + a_2 x + a_3 y + \sum_{m=1}^{M} \alpha_m r_m^2 \ln r_m^2 \,,$$
$$v' = a_4 + a_5 x + a_6 y + \sum_{m=1}^{M} \beta_m r_m^2 \ln r_m^2 \,,$$

where $(x_m, y_m), m \in \{1, 2, \ldots, M\}$, is a set of known anchor points and $r_m^2 = (x - x_m)^2 + (y - y_m)^2 + d^2$.

4.3 Hierarchical Registration

The simplest approach to register two images A and B and to calculate the transformation T is to decompose T into numerous local affine registrations of small sub-images. The idea is to reduce the complexity of the registration process using a hierarchical strategy (see Fig. 4.2).

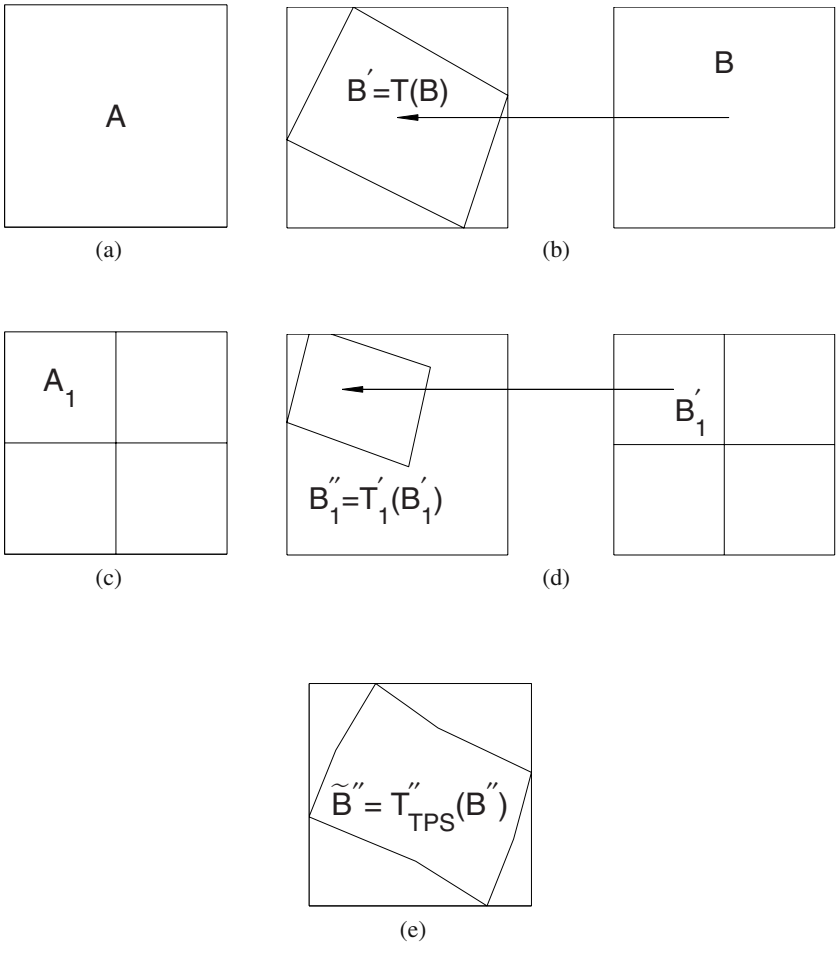

Fig. 4.2 Shows two stages in hierarchical registration process of the floating image B with the reference image A. **(a)** Shows the reference image A. **(b)** Shows the input floating image B and the transformed floating image $B' = T(B)$ after the first stage in global registration. **(c)** Shows the reference image divided into 4 quadrants. **(d)** Shows the transformed image B' divided into four quadrants and the transformed quadrants $B''_k = T'(B'_k)$. (e) Shows the composite image $\widetilde{B}'' = T''_{TPS}(B'')$ formed by applying the TPS transformation to the transformed quadrants $B''_k, k \in \{1,2,\ldots,4\}$.

In the hierarchical approach [1, 13] we progressively subdivide the input images A and B into smaller sub-images which are separately registered by maximizing an appropriate image similarity measure. We separately model each sub-image registrations with an affine transformation. Then the final composite transformation T is found by assimilating all the sub-image transformations using a TPS interpolation algorithm.

The following example explains the basic steps in the hierarchical registration scheme of Likar-Pernus [13].

Example 4.3. Likar-Pernus Hierarchical Registration Scheme [13]. In the Likar-Pernus algorithm we progressively subdivide the two input images A and B. We automatically register the sub-images, and then apply the thin-plate splines interpolation between the centers of registered sub-images. The steps for the first two hierarchical levels are:

1. Register B to the reference image A using an affine transformation T. Let $B' = T(B)$ be the transformed image.
2. Separately partition the images A and B' into four sub-images A_1, A_2, A_3, A_4 and B'_1, B'_2, B'_3, B'_4 of identical size. Each corresponding sub-image pair (A_k, B'_k) is independently registered by using an affine transformation T'_k. Let $B''_k = T'_k(B'_k)$ be the corresponding transformed sub-image.
3. Assimilate the four transformed sub-images $B''_1, B''_2, B''_3, B''_4$ into a single transformed image $\widetilde{B}''$ as follows: The coordinates of the centers of the four registered sub-images $B''_1, B''_2, B''_3, B''_4$ form four point pairs, which are the inputs to the thin-plate splines algorithm. The result is a transformed image $\widetilde{B}'' = T''_{TPS}(B'')$.
4. Separately partition the registered images A and $\widetilde{B}''$ into 16 sub-images $A_1, A_2, \ldots, A_{16}$ and $\widetilde{B}''_1, \widetilde{B}''_2, \ldots, \widetilde{B}''_{16}$ of identical size. Each corresponding sub-image pair $(A_k, \widetilde{B}''_k)$ is registered using an affine transformation T''_k. Let B''' be the corresponding transformed image.
5. Assimilate the 16 transformed sub-images $B'''_1, B'''_2, \ldots, B'''_{16}$ into a single transformed image $\widetilde{B}'''$ as follows: The coordinates of the centers of the 16 registered sub-images (A_k, B'''_k) form 16 point pairs, which are the inputs to the thin-plate splines algorithm. The result is a transformed image $\widetilde{B}''' = T'''_{TPS}(B''')$.

Fig. 4.2 is a graphical description of the hierarchical procedure. *Note*: The registration of finer details is preceded by registration and smooth interpolation obtained at a more global scale.

In principle, the hierarchical decomposition of the images A and B may be continued until the sub-images contain only one pixel. However, in practice, we stop the decomposition process much earlier. The reasons for this are twofold: (1) The algorithm is sensitive to the accuracy of the sub-image registrations: a misregistration at a given hierarchical level can propagate to the lower hierarchical levels. (2)

The probability of a misregistration increases as we move to the lower hierarchical levels. This is because, in general, the reliability of an image similarity measure decreases with the size of the image patch (Chapt 14).

One way to prevent registration errors propagating down the hierarchy is to *adaptively* stop the hierarchical sub-division before the image patches become so small they are effectively "structureless". For this purpose we may use the following test: An image patch (containing K pixels, $(x_k, y_k), k \in \{1, 2, \ldots, K\}$, with gray-levels g_k) is said to be structureless if $\rho > \tau$, where ρ is Moran's autocorrelation coefficient [1]:

$$\rho = \frac{K \sum_{h,k} w_{hk} (g_h - \bar{g})(g_k - \bar{g})}{\sigma \sum_{h,k} w_{hk} \sum_h (g_h - \bar{g})^2} ,$$

τ is a given threshold and $w_{hk} = 1/\sqrt{(x_k - x_h)^2 + (y_k - y_h)^2}$ is the inverse Euclidean distance between (x_k, y_k) and (x_h, y_h).

4.4 Mosaic Image

Thus far we have considered the problem of registering a pair of images. In some applications we are interested in building a single panoramic or "mosaic" image $\widetilde{I}$ from multiple images $I_k, k \in \{1, 2, \ldots, K\}$. To do this we need to find functions T_k which transform each input image I_k onto the image $\widetilde{I}$.

Building a mosaic image from a sequence of partial views is a powerful means of obtaining a broader view of a scene than is available with a single view. Research on automated mosaic construction is ongoing with a wide range of different applications.

Example 4.4. Mosaic Fingerprint Image [9]. Fingerprint-based verification systems have gained immense popularity due to the high level of uniqueness attributed to fingerprints and the availability of compact solid-state fingerprint sensors. However, the solid-state sensors sense only a limited portion of the fingerprint pattern and this may limit the accuracy of the user verification. To deal with this problem we may construct a mosaic fingerprint image from multiple fingerprint impressions.

Example 4.5. Mosaic Image of the Retina [3, 4]. One area in which mosaic images are particularly valuable is in the diagnosis and treatment of diseases of the retina. A seamless mosaic image which is formed from multiple fundus camera images aids in the diagnosis and provides a means for monitoring the progression of different diseases. It may also be used as a spatial map of the retina during surgical treatment.

At first sight we may assume that the ability to spatially align a pair of images is sufficient to solve the problem of forming a mosaic of the entire scene from multiple partial views. Theoretically, if one image can be established as an "anchor image" I_0 on which to base the mosaic image I, then the transformation of each remaining image onto this anchor may be estimated using pairwise registration. The mosaic image I^* is then formed by "stitching" together the transformed images $T_m(I_m)$. Unfortunately, in practice, this approach may not work for the following reasons:

Non-Overlap. Some images may not overlap with the anchor image at all. This makes a direct computation of the transformation impossible. In other cases, images may have insufficient overlap with the anchor image to compute a stable transformation. The straightforward solution is to compose transformations using an "intermediate" image. This is problematic, however, since repeated application of the transformation will often magnify the registration error.

Inconsistent T_m. The transformations T_m may be mutually inconsistent. This may happen even if all the image-to-anchor transformations have been accurately estimated. The reason for this is as follows: Although each image may individually register accurately with the anchor image and the non-anchor images may register accurately with each other, this does not ensure that the transformations onto the anchor image are *mutually consistent*.

One approach to solving this problem is to constrain the transformations so that they are all mutually consistent.

Example 4.6. Transformation Constraints in a Mosaic Image. Given a sequence of N images $I_1, I_2, \ldots, I_N$, we estimate $N(N-1)$ pairwise transformations

$$\begin{pmatrix} u' \\ v' \end{pmatrix} = T_{ij} \begin{pmatrix} x \\ y \end{pmatrix} ,$$

where $(u', v')^T$ and $(x, y)^T$ denote, respectively, the coordinates of corresponding points in I_i and I_j. The T_{ij} must satisfy the following relationships:

$$\begin{aligned} T_{ik} &= T_{ij} \circ T_{jk} , \\ T_{ij} &= T_{ji}^{-1} , \end{aligned}$$

where $T_{ij} \circ T_{jK}$ denotes the application of T_{jk} followed by the application of T_{ij}.

For an affine transformation, the transformation T_{ij} can be written in matrix form as $\begin{pmatrix} u' \\ v' \end{pmatrix} = A_{ij} \begin{pmatrix} x \\ y \end{pmatrix} + B_{ij}$. In this case, the above relationships become

$$\begin{aligned} A_{ik} &= A_{ij} A_{jk} , \\ B_{ik} &= A_{ij} B_{jk} + B_{ij} . \end{aligned}$$

4.4.1 Stitching

In "stitching" together the transformed images $T_m(I_m)$ our aim is to produce a visually plausible mosaic image $\widetilde{I}$ in which, I^* is geometrically and photometrically as similar as possible to the input images $T_m(I_m)$ and the seams between the stitched images are invisible. The stitching algorithms which are commonly used nowadays fall into two types:

Optimal Seam Algorithms. These algorithms search for a curve in the overlap region on which the differences between the $T_m(I_m)$ are minimized. Then each image is copied to the corresponding side of the seam.

Transition Smoothing Algorithms. These algorithms minimize seam artifacts by smoothing the transition region (a small region which is spatially near to the seam) [24].

The following example describes the transition smoothing, or feathering, of two input images I_1 and I_2.

Example 4.7. Feathering [24]. In feathering, the mosaic image $\widetilde{I}$ is a weighted combination of the input images I_1, I_2, where the weighting coefficients vary as a function of the distance from the seam. In general, feathering works well as long as there is no significant misalignment. However, when the misalignments are significant, the mosaic image displays artifacts such as double edges. A modification of feathering which is less sensitive to misalignment errors, is to stitch the derivatives of the input images instead of the images themselves. Let $\partial I_1/\partial x$, $\partial I_1/\partial y$, $\partial I_2/\partial x$ and $\partial I_2/\partial y$ be the derivatives of the input images. If F_x and F_y denote the derivative images formed by feathering $\partial I_1/\partial x$ and $\partial I_2/\partial x$ and $\partial I_1/\partial y$ and $\partial I_2/\partial y$, then we choose the final mosaic image $\widetilde{I}$ to be the image whose derivatives $\partial\widetilde{I}/\partial x$ and $\partial\widetilde{I}/\partial y$ are closest to F_x and F_y.

4.5 Image Similarity Measures

In order to be able to register two images, a measure has to be defined to numerically quantify the goodness of fit between the images, namely the similarity measure. The choice of the appropriate similarity measure is crucial for a successful image registration procedure, so the decisive criterion is the type of images to be registered. Therefore depending on the type of the modalities used to acquire the images, the user can choose between several similarity measures (see Chapt. 14). In this chapter we shall concentrate on the mutual information similarity measure. This has been found to be the most successful especially when the input images are heterogeneous, i. e. they were captured with different sensors or with different spectral bands or with different spatial resolutions [7].

4.6 Mutual Information

Given a reference image A and a spatially aligned and resampled image B [2], the mutual information of A and B is defined as

$$MI(A,B) = \int\int p_{AB}(a,b)\log_2 \frac{p_{AB}(a,b)}{p_A(a)p_B(b)}dxdy\,, \tag{4.2}$$

where $p_A(a)$ is the probability a pixel (x,y) in A has a gray-level a, $p_B(b)$ is the probability a pixel (x,y) in B has a gray-level b and $p_{AB}(a,b)$ is the joint probability a pixel (x,y) in A has a gray-level a and the same pixel in B has a gray-level b.

4.6.1 Normalized Mutual Information

The integral in (4.2) is taken over the pixels which are common to both A and B. As a result, $MI(A,B)$ may vary if the number of pixels which are common to A and B changes. In general the variations in $MI(A,B)$ are small but they may lead to inaccuracies in a spatial alignment algorithm. To avoid these inaccuracies, we often use a normalized mutual information similarity measure in place of $MI(A,B)$. Four commonly used normalized MI measures are [10]:

$$NMI(A,B) = \begin{cases} \dfrac{MI(A,B)}{H(A)+H(B)} \\ \dfrac{MI(A,B)}{\min(H(A),H(B))}\,, \\ \dfrac{MI(A,B)}{H(A,B)}\,, \\ \dfrac{MI(A,B)}{\sqrt{H(A)H(B)}}\,, \end{cases}$$

where

$$\begin{aligned} H(A) &= -\int\int p_A(a)\log_2 p_A(a)dxdy\,, \\ H(B) &= -\int\int p_B(b)\log_2 p_B(b)dxdy\,, \\ H(A,B) &= -\int\int p_{AB}(a,b)\log_2 p_{AB}(a,b)dxdy\,, \end{aligned}$$

and the integration is performed over the overlap region of the images A and B.

[2] *Note*: In this section B denotes the spatially aligned and re-sampled floating image.

4.6.2 Calculation

The most common approach to calculate the mutual information $MI(A,B)$ and the normalized mutual information $NMI(A,B)$ is to calculate the entropies $H(A)$, $H(B)$ and $H(A,B)$ using the marginal probabilities $p_A(a)$ and $p_B(b)$ and the joint probability $p_{AB}(a,b)$. Since, the marginal probabilities may be derived from $p_{AB}(a,b)$:

$$p_A(a) = \int p_{AB}(a,b)db\,, \qquad p_B(b) = \int p_{AB}(a,b)da\,,$$

we need only consider the calculation of $p_{AB}(a,b)$.

4.6.3 Histogram

The most straightforward way to calculate the joint probability distribution $p_{AB}(a,b)$ is to use a discrete histogram H_{AB} as follows: We quantize the gray-levels in A and B into P and Q bins respectively Then we approximate $p_{AB}(a,b)$ using the two-dimensional histogram $H_{AB} = (h_{AB}(1,1), h_{AB}(1,2), \ldots, h_{AB}(P,Q))^T$, where $h_{AB}(p,q)$ is the number of pixels whose gray-levels in A fall in the pth bin and whose gray-levels in B fall in the qth bin.

In this case, the formula for the mutual information, is

$$MI(A,B) = \sum_{(p,q)} h_{AB}(p,q)\log_2\left(\frac{h_{AB}(p,q)}{h_A(p)h_B(q)}\right) \Big/ \sum_{(p,q)} h_{AB}(p,q)\,,$$

where

$$h_A(p) = \sum_q h_{AB}(p,q) \qquad \text{and} \qquad h_B(q) = \sum_p h_{AB}(p,q)\,.$$

Although widely used, the histogram method suffers from several drawbacks: It yields a discontinuous density estimate and there is no principled method for choosing the size and placement of the bins. For example, if the bin width is too small, the density estimate is noisy while if the bin width is too large the density estimate is too smooth. Legg *et al.* [14] recommends using Sturges' rule for the optimal bin width:

$$w = \frac{r}{1+\log_2(K)}\,,$$

where r is the range of gray-level values, K is the number of elements in the input image. In this case the optimal number of bins is r/w.

A partial solution to these problems is to calculate $p_{AB}(a,b)$ using the method of Parzen windows.

4.6.4 Parzen Windows

Instead of using discrete histogram bins to calculate the joint probability distribution $p_{AB}(a,b)$, we use continuous bins. This is known as kernel, or Parzen-window,

density estimation [19, 20] and is a generalization of histogram binning. If A and B each contain K pixels with gray-levels $a_k, b_k, k \in \{1, 2, \ldots, K\}$, then the estimated joint density $p_{AB}(a,b)$ is given by

$$p_{AB}(a,b) = \frac{1}{K^2 \sigma_A \sigma_B} \sum_{k=1}^{K} \sum_{l=1}^{K} H\left(\frac{a-a_k}{\sigma_A}\right) H\left(\frac{b-b_l}{\sigma_B}\right),$$

where $H(x)$ denotes a kernel function which satisfies $\int_x H(x)dx = 1$. In general a density estimate $p(x)$ is more sensitive to the choice of the bandwidth σ and less sensitive to the choice of the kernel $H(x)$. For this reason we often use a zero-mean Gaussian function with standard deviation σ for the kernel $H(x)$. Table 4.2 lists several schemes which are commonly used to calculate the optimal bandwidth σ.

Table 4.2 Methods for Calculating Optimum One-Dimensional Bandwidth σ

Name	Description
Rule-of-Thumb	Suppose the input data (consisting of N measurements $a_i, i \in \{1, 2, \ldots, N\}$), is generated by a given parametric density function, e. g. a Gaussian function. In this case $\sigma = 1.06\hat{\sigma}N^{-1/5}$, where $\hat{\sigma}$ is the sample standard deviation. Robust versions of this bandwidth are available: $\sigma = 1.06\min(\hat{\sigma}, \hat{Q}/1.34)N^{-1/5}$ and $\sigma = 1.06\hat{s}N^{-1/5}$, where $\hat{Q}$ is the sample interquartile distance and $\hat{s} = \mathrm{med}_j\lvert a_j - \mathrm{med}_i a_i\rvert$.
Cross-Validation (CV)	Use a CV procedure to directly minimize the MISE or the AMISE. CV variants include least square, biased and smoothed CV [11].
Plug-in	Minimize the AMISE using a second bandwidth known as the *pilot* bandwidth Σ. In the *solve-the-equation plug-in* method we write L as a function of the kernel bandwidth σ [11].

MISE is the mean integrated square error and is defined as $\mathrm{MISE}(p, \hat{p}_\sigma) = \int \left(p(a) - \hat{p}_\sigma(a)\right)^2 da$, where $\hat{p}_\sigma(a)$ is the kernel approximation to $p(a)$. AMISE denotes the asymptotic MISE and represents a large number approximation of the MISE.

4.6.5 Iso-intensity Lines

Iso-intensity lines [17] is a new scheme developed specifically for calculating the joint probability density $p_{AB}(a,b)$. Suppose the gray-levels in A and B are quantized, respectively, into P and Q bins. For each pixel location (m,n) we estimate the gray-level values G_1, G_2, G_3, G_4 of its four neighbors which lie at a horizontal or vertical distance of half a pixel from (m,n). We divide the square defined by these neighbors into a pair of triangles (see Fig. 4.3). Within the triangle we suppose the gray-level values vary linearly as follows:

$$A(m+\delta x, n+\delta y) = a_A x + b_A y + c_A ,$$
$$B(m+\delta x, n+\delta y) = a_B x + b_B y + c_B ,$$

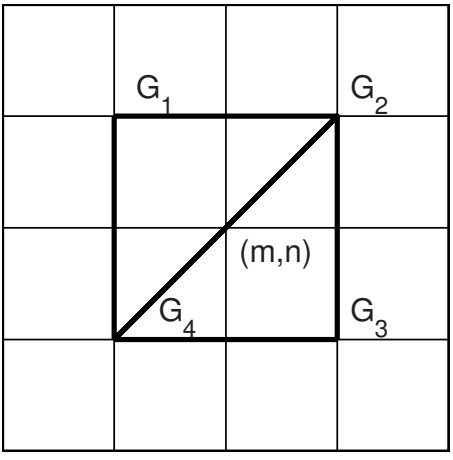

Fig. 4.3 Shows the pixel (m,n) with the gray-levels $G_k, k \in \{1,2,3,4\}$

where $A(m+\delta x, n+\delta y)$ and $B(m+\delta x, n+\delta y)$ denote, respectively, the gray-level of a point $(m+\delta x, n+\delta y)$ in the triangle and $-0.5 \leq \delta x, \delta y \leq 0.5$. To calculate the joint distribution of the two images A and B we sequentially consider the PQ different gray-level pairs denoted as (α,β). For each pixel (m,n) we see whether the pair of corresponding triangles contains a point $(m+\delta x, n+\delta y)$ which has a gray-level value α in A and β in B. Such a point $(m+\delta x, n+\delta y)$ contributes a vote to the entry (α,β) in $p_{AB}(a,b)$.

4.7 Partial Volume Interpolation

The histogram, Parzen and iso-intensity line algorithms all assume the images A and B are spatially aligned and if necessary image interpolation has been performed. The partial volume interpolation (PVI) is an alternative technique which does not assume spatial alignment or image interpolation [16]. It works as follows.

Suppose T represents a mapping of the pixel (x,y) in B into the corresponding location (u',v') in A. In general (u',v') will not correspond to a pixel location in A. Suppose $Q_k = (u_k, v_k), k \in \{0,1,2,3\}$, are the four pixel locations in A which surround (u',v'). (Fig. 4.4). Then if $A(u_k,v_k)$ has a quantized gray-level α_k and $B(x,y)$ has a quantized gray-level β, then $H_{AB}(\alpha_k,\beta)$ receives a fractional vote equal to

$$r_k^{-1} / \sum_{h=0}^{3} r_h^{-1} ,$$

where $r_k = \sqrt{(u_k-u')^2 + (v_k-v')^2}$.

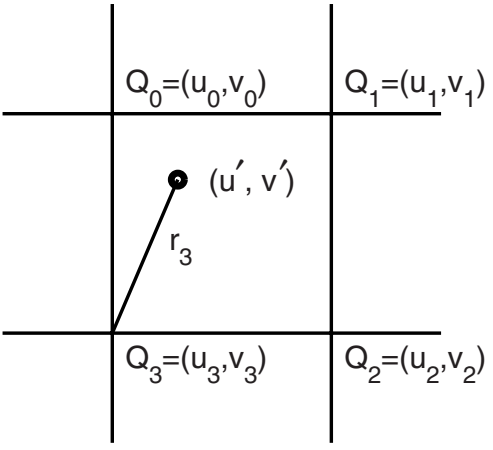

Fig. 4.4 Shows the four pixel locations $Q_k = (u_k, v_k), k \in \{0, 1, \ldots, 3\}$, in A which surround the transformed point (u', v'). Also shown is the Euclidean distance $r_3 = \sqrt{(u_3 - u')^2 + (v_3 - v')^2}$ from Q_3 to (u', v').

4.8 Artifacts

The success of the mutual information algorithms image registration lie in their inherent simplicity. It makes few assumptions regarding the relationship that exists between different images. It only assumes a *statistical* dependence. The idea is that although different sensors may produce very different images, since they are imaging the same underlying scene, there will exist some inherent mutual information between the images. When the images (or image patches) are spatially aligned, then the mutual information is maximal. To be an effective similarity measure, however, we require the mutual information to fall monotonically to zero as we move away from perfect alignment. In practice, the MI does not fall monotonically to zero.

These artifacts are due to inaccuracies in estimating the marginal densities $p_A(a)$ and $p_B(b)$ and the joint density $p_{AB}(a, b)$. The artifacts are of two types:

Interpolation effects [21]**.** Initially, when the images are aligned the pixel locations of A and B coincide. Therefore no interpolation is needed when estimating the joint intensity histogram. At the same time the dispersion of the histogram is minimal when the images are registered and therefore the joint entropy is minimal. By translating the floating image B with an integer number of the pixel dimension, the grid points of the two images will again be aligned avoiding the need for interpolation, but the dispersion of the joint histogram is increasing due to misregistration, reducing the MI accordingly. For all other translations, corresponding to some fraction of pixel dimension, the pixel locations of the images do not coincide anymore and therefore interpolation is required to estimate intensity values between pixel locations of the reference image. As a consequence the joint histogram is not only dispersed because of the image content and a possible

misregistration, but it also contains an additional dispersion induced by the interpolation method. More dispersion implies a higher joint entropy value, which in turn decreases the MI of the reference image between the pixel location. The MI is found to vary as shown in Fig.4.5. An effective way to reduce the interpolation effects is to use nearest neighbor interpolation with jittered sampling [21]. We jitter the coordinates of each pixel which is to be interpolated by adding a normally distributed random offset (zero mean and standard deviation of one-half).

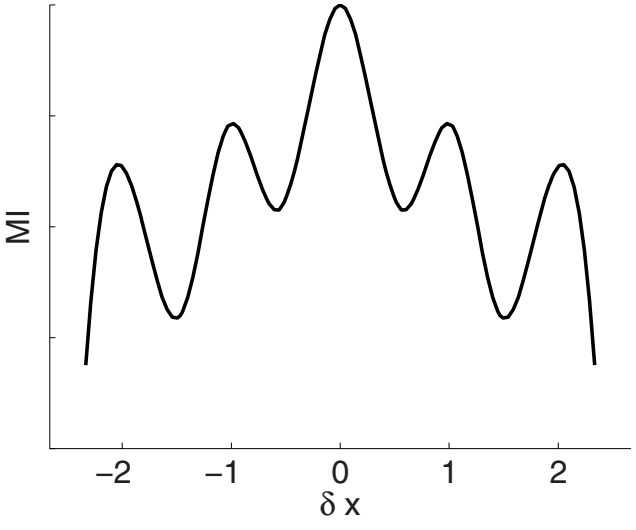

Fig. 4.5 Shows the typical interpolation artifacts as a function of the relative displacement δx between the reference image and the floating image. *Note*: PVI interpolation curves are often concave in shape.

Small size effects. Sometimes an image patches with a low structural content may appear. This often occurs when we register the two images using a hierarchical matching algorithm. These structureless patches often lead to inconsistent local registrations due to a low MI response. If two signals are independent then their MI reaches its minimum possible value of zero. We might expect, therefore, that by shifting a structureless sub-image around its initial position, the similarity measure will have a small response. Surprisingly this is not true. The MI starts to increase as soon as a structureless sub-image overlaps a region of higher structural content. One explanation for this phenomena is the following [1]: The number of samples required to obtain a consistent estimate of the marginal entropies $H(A)$ and $H(B)$ is much less than the number of samples required to obtain a consistent of the joint entropy $H(A,B)$.

4.9 Software

KDE, KDE2D. Automatic data-driven bandwidth selection functions. Available from the matlab central depository. Author: Zdravko Botev [2].

MATLAB IMAGE PROCESSING TOOLBOX. Matlab image processing toolbox. The toolbox contains m-files for performing image registration, re-sampling and interpolation.

THIN PLATE SPLINES. Suite of matlab m-files for performing thin plate spline interpolation. Available from matlab central depository. Author: Bing Jian.

4.10 Further Reading

The subject of image registration has been intensely investigated for many years. A modern review of image warping is [8]. Specific references on the use of mutual information for image registration are [15, 16]. The calculation of mutual information has been considered by many authors including [5, 6].

References

1. Andronache, A., von Siebenthal, M., Szekely, G., Cattin, P.: Non-rigid registration of multi-modal images using both mutual information and cross-correlation. Med. Imag. Anal. 12, 3–15 (2006)
2. Botev, Z.I.: A novel nonparametric density estimator. Technical Report. The University of Queensland
3. Can, A., Stewart, C.V., Roysam, B., Tanenbaum, H.L.: A feature-based technique for joint, linear estimation of higher-order image-to-mosaic transformations: mosaicing the curved human retina. IEEE Trans Patt. Anal. Mach. Intell. 24, 412–419 (2002)
4. Can, A., Stewart, C.V., Roysam, B., Tanenbaum, H.L.: A feature-based, robust, hierarchical algorithm for registering pairs of images of the curved human retina. IEEE Trans. Patt. Anal. Mach. Intell. 24, 347–364 (2002)
5. Darbellay, G.A.: An estimator for the mutual information based on a criterion for independence. Comp. Stats. Data Anal. 32, 1–17 (1999)
6. Darbellay, G.A., Vajda, I.: Estimation of the information by an adaptive partitioning of the observation space. IEEE Trans. Inf. Theory 45, 1315–1321 (1999)
7. Fransens, R., Strecha, C., van Gool, L.: Multimodal and multiband image registration using mutual information. In: Proc. ESA-EUSC (2004)
8. Glasbey, C.A., Mardia, K.V.: A review of image warping methods. J. Appl. Stat. 25, 155–171 (1998)
9. Jain, A., Ross, A.: Fingerprint mosaicking. In: IEEE Int. Conf. ICASSP (2002)
10. Hossny, M., Nahavandi, S., Creighton, D.: Comments on Information measure for performance of image fusion. Elect. Lett. 44, 1066–1067 (2008)
11. Jones, M.C., Marron, J.S., Sheather, S.J.: A brief survey of bandwidth selection for density estimation. J. Am. Stat. Assoc. 91, 401–407 (1996)
12. Likar, B., Pernus, F.: Registration of serial transverse sections of muscle fibers. Cytometry 37, 93–106 (1999)

13. Likar, B., Pernus, F.: A hierarchical approach to elastic registration based on mutual information. Image Vis. Comp. 19, 33–44 (2001)
14. Legg, P.A., Rosin, P.L., Marshall, D., Morgan, J.E.: Improving accuracy and efficiency of registration by mutual information using Sturges' histogram rule. In: Proc. Med. Image Understand. Anal., pp. 26–30 (2007)
15. Maes, F., Vandermeulen, D., Suetens, P.: Medical image registration using mutual information. Proc. IEEE 91, 1699–1722 (2003)
16. Pluim, J.P.W., Maintz, J.B.A., Viergever, M.A.: Mutual information based registration of medical images: a survey. IEEE Trans. Med. Imag. 22, 986–1004 (2003)
17. Rajwade, A., Banerjee, A., Rangarajan, A.: Probability density estimation using isocontours and isosurfaces: application to information theoretic image registration. IEEE Trans. Patt. Anal. Mach. Intell. (2009)
18. Ross, A., Dass, S.C., Jain, A.K.: A deformable model for fingerprint matching. Patt. Recogn. 38, 95–103 (2005)
19. Scott, D.W.: Multivariate Density Estimation. Wiley, Chichester (1992)
20. Silverman, B.: Density Estimation for Statistical Data Analysis. Chapman and Hall, Boca Raton (1986)
21. Tsao, J.: Interpolation artifacts in multimodality image registration based on maximization of mutual information. IEEE Trans. Med. Imag. 22, 854–864 (2003)
22. Zagorchev, L., Goshtasby, A.: A comparative study of transformation functions for nonrigid image registration. IEEE Trans. Image Process. 15, 529–538 (2006)
23. Zitova, B., Flusser, J.: Image registration: A survey. Image Vis. Comput. 21, 977–1000 (2003)
24. Zomet, A., Levin, A., Peleg, S., Weiss, Y.: Seamless image stitching by minimizing false edges. IEEE Trans. Image Process. 15, 969–977 (2006)