

Chapter 2

PRELIMINARIES AND REVIEWS

In this chapter, we first present some mathematical preliminaries on various topics needed for the development of the book, beginning with a glossary of notations. We then review related literature on (i) face recognition under illumination and/or pose variations and (ii) face recognition from multiple still images or video sequences (including visual tracking).

2.1 Preliminaries

We begin by introducing some notations commonly used throughout the book and then present basic introductions to several relevant research topics, including Lambertian illumination model, subspace analysis, kernel method, regression, state space time series, and particle filter.

2.1.1 Notation

We denote a scalar by a , a vector by \mathbf{a} , and a matrix with p rows and q columns by $\mathbf{A}_{p \times q}$, and a block matrix by \mathcal{A} . The matrix transpose is denoted by \mathbf{A}^T , the pseudo-inverse by \mathbf{A}^\dagger . The matrix L_r -norm is denoted by $\|\cdot\|_r$.

The following special notations are introduced for brevity, convenience, and emphasis of special structure.

- Concatenation notations: \Rightarrow and \Downarrow .
 \Rightarrow and \Downarrow mean horizontal and vertical concatenations, respectively. For example, we can represent an $n \times 1$ vector $\mathbf{a}_{n \times 1}$ and its transpose by

$$\mathbf{a} = [a_1, a_2, \dots, a_n]^T = [\Downarrow_{i=1}^n a_i], \quad \mathbf{a}^T = [a_1, a_2, \dots, a_n] = [\Rightarrow_{i=1}^n a_i].$$

Similarly, we can represent a matrix $\mathbf{A}_{p \times q}$ by

$$\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_q] = [\Rightarrow_{i=1}^q \mathbf{a}_i]$$

where each \mathbf{a}_i is a $p \times 1$ vector.

We can use \Rightarrow and \Downarrow to concatenate matrices to form a block matrix \mathcal{A} . For instance, given a collection of matrices $\{\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_n\}$ of size $p \times q$, we construct a block matrix of size $p \times nq$ as

$$\mathcal{A}_{p \times nq} = [\Rightarrow_{i=1}^n \mathbf{A}_i].$$

Given a collection of matrices $\{\mathbf{A}_{11}, \mathbf{A}_{12}, \dots, \mathbf{A}_{1n}, \dots, \mathbf{A}_{mn}\}$ of size $p \times q$, we construct a block matrix of size $pm \times qn$ as

$$\mathcal{A}_{mp \times nq} = [\Downarrow_{i=1}^m [\Rightarrow_{j=1}^n \mathbf{A}_{ij}]].$$

and a block matrix of size $p \times qmn$

$$\mathcal{A}_{p \times qmn} = [\Rightarrow_{i=1}^m [\Rightarrow_{j=1}^n \mathbf{A}_{ij}]].$$

- Kronecker (tensor) product: \otimes .

It is defined as

$$\mathbf{A}_{p \times q} \otimes \mathbf{B}_{m \times n} = [\Downarrow_{i=1}^p [\Rightarrow_{j=1}^q a_{ij} \mathbf{B}]]_{pm \times qn}.$$

Note that the two matrices can be of different sizes.

- Hadamard (element-wise) product: \circ .

It is defined as

$$\mathbf{A}_{m \times n} \circ \mathbf{B}_{m \times n} = [\Downarrow_{i=1}^m [\Rightarrow_{j=1}^n a_{ij} b_{ij}]]_{m \times n}.$$

Note that the two matrices must be of identical size.

- Vectorization operator: $vec(\cdot)$.

It converts a $p \times q$ matrix to a $pq \times 1$ vector by arranging all the elements of the matrix according to a fixed order, say a lexicographic order. In other words,

$$vec(\mathbf{A})_{pq \times 1} = [\Downarrow_{i=1}^p [\Downarrow_{j=1}^q a_{ij}]].$$

- Gram matrix. The dot product matrix (or Gram matrix) of two matrices $\mathbf{A}_{p \times q} = [\Rightarrow_{i=1}^q \mathbf{a}_i]$ and $\mathbf{B}_{p \times q} = [\Rightarrow_{j=1}^q \mathbf{b}_j]$ is given as

$$\mathbf{A}_{p \times q}^\top \mathbf{B}_{p \times q} = [\Downarrow_{i=1}^q [\Rightarrow_{j=1}^q \mathbf{a}_i^\top \mathbf{b}_j]].$$

- Identity matrix \mathbf{I}_m of size $m \times m$.
- Diagonal matrix $\mathbf{D}[d_1, d_2, \dots, d_m]$ of size $m \times m$ whose diagonal elements are $\{d_1, d_2, \dots, d_m\}$.
- Normal distribution $\mathcal{N}(\mathbf{x}; \mu, \Sigma)$ with mean μ and covariance matrix Σ .

2.1.2 Lambertian illumination model

Many illumination models have been proposed in the literature. In the face recognition community, a Lambertian imaging model with a varying albedo field is mostly used. In the Lambertian illumination model, an image pixel h is represented as

$$h = \max(\rho \mathbf{n}^T \mathbf{s}, 0) = \max(\mathbf{t}^T \mathbf{s}, 0), \quad (2.1)$$

where ρ is the albedo at the pixel, $\mathbf{n} = [\hat{a}, \hat{b}, \hat{c}]^T$ is the unit surface normal vector at the pixel, $\mathbf{t}_{3 \times 1} = \rho \mathbf{n}$ is the product of albedo and surface normal, and \mathbf{s} (a 3×1 unit vector multiplied by its intensity) specifies a distant illuminant. If the pixel is not directly facing the light source, it satisfies $\mathbf{t}^T \mathbf{s} < 0$ or $h = 0$. This is called an *attached shadow*. Another kind of shadow is *cast shadow*. Cast shadow is generated at a certain pixel when the light source is blocked by other pixels. This is related to the geometry of the object. Figure 2.1 gives an illustration of the Lambertian illumination model.

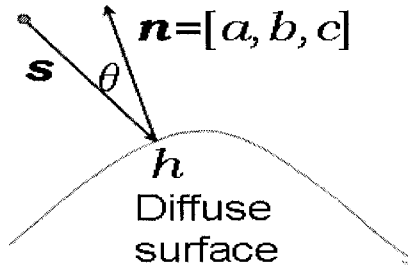


Figure 2.1. Illustration of Lambertian illumination model.

An image \mathbf{h} is a collection of d pixels $\{h_i, i = 1, \dots, d\}$. By stacking all the pixels into a column vector, we have

$$\begin{aligned} \mathbf{h}_{d \times 1} &= [\downarrow_{i=1}^d h_i] = [\downarrow_{i=1}^d \max(\mathbf{t}_i^T \mathbf{s}, 0)] \\ &= \max(\mathbf{T}_{d \times 3} \mathbf{s}_{3 \times 1}, 0), \end{aligned} \quad (2.2)$$

where the \mathbf{T} matrix encodes the ‘product’ of the albedo and the surface normal for all pixels. We call the \mathbf{T} matrix as the *object-specific albedo-shape* matrix.

Another useful parameterization of the Lambertian model uses surface shape gradients. Suppose that $p_{(x,y)}$, $q_{(x,y)}$ are the shape gradients, i.e., partial derivatives of the depth map $z_{(x,y)}$.

$$h = \rho \cos(\theta) = \rho \frac{1 + pP_s + qQ_s}{\sqrt{1 + p^2 + q^2} \sqrt{1 + P_s^2 + Q_s^2}} \quad (2.3)$$

where θ is the angle between the outward normal to the surface $\vec{n} = (p, q, 1)$ and the negative illumination vector $-\vec{L} = (P_s, Q_s, 1)$ which represents the direction opposite to the distant light source.

2.1.3 Subspace analysis

Subspace analysis is often used in pattern recognition, signal processing and computer vision problems as an efficient method for both dimensionality reduction and finding the direction of the projection with certain properties. For example, in the context of face recognition [29], one attempts to find some basis vectors in that space serving as directions of projection, and hopefully the projected data are clustered according to their class labels.

The basic framework of subspace analysis is as follows. Suppose we have a d -dimensional random vector \mathbf{x} where d is very large, we attempt to find m basis vectors ($m < d$) forming a projection matrix $\mathbf{U}_{d \times m}$, such that the new representation \mathbf{z} defined below satisfies certain properties.

$$\mathbf{z}_{m \times 1} = \mathbf{U}_{d \times m}^T \mathbf{x}_{d \times 1}.$$

Different properties give rise to different kinds of analysis methods. Two popular methods are principal component analysis (PCA), linear discriminant analysis (LDA) and independent component analysis (ICA).

- PCA [12], an unsupervised method, decomposes the available data into uncorrelated directions, along which there exist the maximum variations. In other words, it tries to minimize the reconstruction error $\|\mathbf{U}\mathbf{U}^T - \mathbf{X}\|_2$, where $\mathbf{X} = [\Rightarrow_{n=1}^N \mathbf{x}_n]$ encodes the training data set.

$$\mathbf{U} = \arg \min_{\mathbf{U}} \|\mathbf{U}\mathbf{U}^T - \mathbf{X}\|_2.$$

To this end, a total scatter matrix $\hat{\Sigma} = \mathbf{X}\mathbf{X}^T$ is defined and the optimal matrix \mathbf{U} is formed by the eigenvectors corresponding to the m largest eigenvalues of $\hat{\Sigma}$.

- LDA [7], a supervised method, exploits the class label information and attempts to maximize the between-class scatter while minimizing the within-class scatter. In LDA [7], two scatter matrices are defined: between-class scatter matrix Σ_B and within-class scatter matrix Σ_W [7]. The following cost function $J(\mathbf{U})$ is maximized.

$$\mathbf{U} = \arg \max_{\mathbf{U}} \frac{\|\mathbf{U}^T \Sigma_B \mathbf{U}\|_2}{\|\mathbf{U}^T \Sigma_W \mathbf{U}\|_2}.$$

- ICA [43, 243], an unsupervised method, finds the projection directions along which the data are statistically independent. Often, a contrast function that measures the independence is minimized.

2.1.4 Kernel method

PCA and LDA are linear methods that utilize first- and second-order statistics. Therefore, their effectiveness is diminished when confronted with highly nonlinear data structures. To enhance their modeling capability, kernel PCA (KPCA) [272] and kernel LDA (KLDA) [263, 268] have been proposed in the literature. These kernel methods enhance the modeling capability by nonlinearly mapping the data from the original space to a very high dimensional feature space, the so-called RKHS. The nonlinear mapping enables implicit characterization of higher-order statistics. The key idea of kernel methods is to avoid the explicit knowledge of the mapping function by evaluating the dot product in the feature space using a kernel function.

In the core of kernel methods lies a kernel function. Let \mathcal{E} be a set. A two variable function $k(\alpha, \beta)$ on $\mathcal{E} \times \mathcal{E}$ is a *reproducing kernel* if for any finite point set $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$ and for any corresponding real numbers $\{c_1, c_2, \dots, c_n\}$, the following condition holds:

$$\sum_{i=1}^n \sum_{j=1}^n c_i c_j k(\alpha_i, \alpha_j) \geq 0.$$

The most widely used kernel functions in the literature [18, 19, 21] are defined on a vector space, that is $\mathcal{E} = \mathcal{R}^p$. For example, two popular kernel functions based on vector inputs are the radial basis function (RBF) and the polynomial kernels. Their definitions are as follows [18, 19, 21]. $\forall \mathbf{x}, \mathbf{y} \in \mathcal{R}^p$,

$$k(\mathbf{x}, \mathbf{y}) = \exp\{-\theta^{-1} \|\mathbf{x} - \mathbf{y}\|^2\}, \quad k(\mathbf{x}, \mathbf{y}) = \{\mathbf{x}^\top \mathbf{y} + \theta\}^d.$$

The kernel function k can be interpreted as a dot product between two vectors in a very high-dimensional space, i.e., the RKHS \mathcal{H}_k . In other words, there exists a nonlinear mapping function $\phi : \mathcal{R}^p \rightarrow \mathcal{H}_k = \mathcal{R}^f$, where $f > p$ and f could even be infinite, such that

$$k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^\top \phi(\mathbf{y}).$$

This is the so-called ‘kernel trick’, which is also illustrated in Figure 2.2.

Given a set of training data $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$, the Gram matrix characterizes complete information for the kernel method. The Gram matrix $\mathbf{K} = [k(\alpha_i, \alpha_j)]$ is an $n \times n$ matrix whose ij^{th} element equals to $k(\alpha_i, \alpha_j)$.

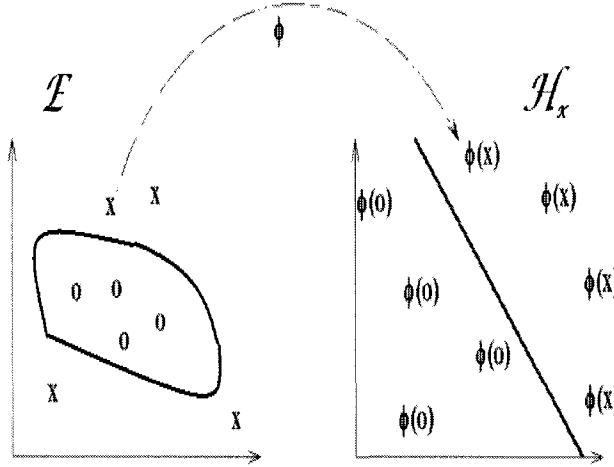


Figure 2.2. Illustration of ‘kernel trick’: The nonlinear decision boundary in the original space \mathcal{E} becomes linear in the RKHS \mathcal{H}_k .

2.1.5 Regression

Regression finds the solution to the following minimizing problem:

$$\hat{g}(x) = \arg \min_{g \in \mathcal{G}} \mathcal{E}_{p(x,y)} \{L(y(x), g(x))\}, \quad (2.4)$$

where \mathcal{G} is the set of allowed output functions, $\mathcal{E}_{p(x,y)}$ takes the expectation under the distribution $p(x, y)$, and the $L(\circ, \circ)$ function is the loss function that penalizes the deviation of the regressor output $g(x)$ from the true output $y(x)$. We assume that $x \in \mathcal{R}^d$ and $y(x) \in \mathcal{R}^q$.

In practice, it is impossible to compute the expectation since the distribution $p(x, y)$ is unknown. Given a set of training examples $\{(x_n, y(x_n)); n = 1, 2, \dots, N\}$, the cost function $\mathcal{E}_{p(x,y)} L(y(x), g(x))$ is approximated as the training error $J(g) = N^{-1} \sum_{n=1}^N L(y(x_n), g(x_n))$.

If the number of samples N is infinitely large, the above approximation is exact by the law of the large number. Unfortunately, a practical value of N is never large enough, especially when dealing with image data and high-dimensional output parameter. A more severe problem is *overfitting*: given a limited number of training examples, the function $g(x)$ can be easily and *arbitrarily* constructed to yield a zero training error. Therefore, additional regularization constraints are used. The combined cost function is given as

(ignoring the scaling factor N^{-1})

$$J(\mathbf{g}) = \sum_{n=1}^N L(y(\mathbf{x}_n), \mathbf{g}(\mathbf{x}_n)) + \lambda R(\mathbf{g}),$$

where $\lambda > 0$ is the *regularization coefficient* that controls the degree of regularization and $R(\mathbf{g})$ is the regularization term. Regularization often imposes certain smoothness on the output function or reflects the belief of some prior knowledge of the output.

Popular regression algorithms are data-driven in the sense that the output function are directly related to the training data inputs. Examples of data-driven regressors include nonparametric kernel regression (NPR), linear methods and their nonlinear kernel variants such as kernel ridge regression (KRR), support vector regression (SVR), etc.

Nonparametric kernel regression (NPR)

Nonparametric kernel regression (NPR) [9] is a smoothed version of k -nearest-neighbor (k NN) regression. The k NN regressor approximates the conditional mean, an optimal estimate in the L^2 sense. NPR takes the following form:

$$\mathbf{g}(\mathbf{x}) = \frac{\sum_{n=1}^N h_{\sigma}(\mathbf{x}; \mathbf{x}_n) y(\mathbf{x}_n)}{\sum_{n=1}^N h_{\sigma}(\mathbf{x}; \mathbf{x}_n)},$$

where $h_{\sigma}(\circ; \mathbf{x}_n)$ is a kernel function. The most widely used kernel function is the RBF kernel

$$h_{\sigma}(\mathbf{x}; \mathbf{x}_n) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_n\|^2}{2\sigma^2}\right).$$

The RBF kernel has a noncompact support. Other kernel functions with compact supports such as the Epanechnikov kernel can be used too.

In general, when confronted with the scenario of image based regression, NPR, albeit smooth, tends to overfit the data and to yield a low bias and a high variance.

Kernel ridge regression (KRR)

Kernel ridge regression (KRR) [9] assumes that the multiple-output regression function takes a linear form:

$$\mathbf{g}(\mathbf{x}) = \sum_{n=1}^N \alpha_n k(\mathbf{x}; \mathbf{x}_n),$$

where $k(\mathbf{x}; \mathbf{x}_n)$ is a reproducing kernel function and α_n is a $q \times 1$ vector that weights the kernel function. The solution to the multiple-output KRR is

$$\mathbf{g}(\mathbf{x}) = \mathbf{Y}(\mathbf{K} + \lambda \mathbf{I})^{-1} \kappa(\mathbf{x}),$$

where $\mathbf{Y}_{q \times N} = [\Rightarrow_{i=1}^N \mathbf{y}(\mathbf{x}_i)]$ is the training output matrix, $\mathbf{K}_{N \times N} = [k(\mathbf{x}_i; \mathbf{x}_j)]$ is the Gram matrix for the training data, and

$$\kappa(\mathbf{x})_{N \times 1} = [\Downarrow_{n=1}^N k(\mathbf{x}; \mathbf{x}_n)].$$

In general, when a linear kernel is used, KRR tends to underfit the data and yields a high bias and a low variance. Using the nonlinear kernel function often gives enhanced performance. One computational difficulty of KRR lies in inverting the $N \times N$ matrix $\mathbf{K} + \lambda \mathbf{I}$.

Support vector regression (SVR)

Support vector regression (SVR) [21] is a robust regression method. Its current formulation works for *single output* data, i.e. $q = 1$. SVR minimizes the following cost function

$$\frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{n=1}^N |y(\mathbf{x}_n) - g(\mathbf{x}_n)|_\epsilon,$$

where $|\cdot|_\epsilon$ is an ϵ -insensitive function, $g(\mathbf{x}) = \sum_{n=1}^N w_n k(\mathbf{x}; \mathbf{x}_n)$ with $k(\mathbf{x}; \mathbf{x}_n)$ being a reproducing kernel function and w_n its weight, and $\mathbf{w} = [\Downarrow_{n=1}^N w_n]$. Because some of the coefficients w_n , which can be found through a quadratic programming procedure, are zero-valued, the samples \mathbf{x}_n associated with nonzero weights are called support vectors.

SVR strikes a good balance between bias and variance tradeoff and hence is very robust. Unfortunately, directly applying SVR to the multiple-output regression problem is difficult.

2.1.6 State space time series model and particle filter

State space time series models [17] are widely employed to represent video data. Two important components of state space modeling are state transition and observation models whose most general forms can be defined as follows:

$$\text{State transition model: } \theta_t = \mathbf{f}_t(\theta_{t-1}, \mathbf{u}_t), \quad (2.5)$$

$$\text{Observation model: } \mathbf{y}_t = \mathbf{g}_t(\theta_t, \mathbf{v}_t), \quad (2.6)$$

where \mathbf{u}_t is the system noise, $\mathbf{f}_t(\cdot, \cdot)$ characterizes the kinematics, \mathbf{v}_t is the observation noise, and $\mathbf{g}_t(\cdot, \cdot)$ models the observer.

The key quantity that characterizes the time series is the posterior distribution $p(\theta_{1:s} | \mathbf{y}_{1:t})$. Depending on the relation between s and t , we solve three different problems: *filtering* if $s = t$, *prediction* if $s > t$, and *smoothing* if $s < t$. In face tracking and recognition problems, we mostly consider the filtering problem and how to solve it.

General particle filter algorithm

Given the state transition model in (2.5) characterized by the state transition probability $p(\theta_t|\theta_{t-1})$ and the observation model in (2.6) characterized by the likelihood function $p(y_t|\theta_t)$, the problem is reduced to computing the posterior probability $p(\theta_t|y_{1:t})$. The nonlinearity and non-Normality in (2.5) and (2.6) make the standard Kalman filter [1] ineffective. The particle filter is a means of approximating the posterior distribution $p(\theta_t|y_{1:t})$ by a set of weighted particles $\mathcal{S}_t = \{\theta_t^{(j)}, w_t^{(j)}\}_{j=1}^J$ with $\sum_{j=1}^J w_t^{(j)} = 1$. It can be shown [248] that \mathcal{S}_t is *properly weighted* with respect to $p(\theta_t|y_{1:t})$ in the sense that, for every bounded function $h(\cdot)$,

$$\lim_{J \rightarrow \infty} \sum_{j=1}^J w_t^{(j)} h(\theta_t^{(j)}) = \mathbb{E}_p[h(\theta_t)]. \quad (2.7)$$

Given $\mathcal{S}_{t-1} = \{\theta_{t-1}^{(j)}, w_{t-1}^{(j)}\}_{j=1}^J$ which is properly weighted with respect to $p(\theta_{t-1}|y_{1:t-1})$, we first resample \mathcal{S}_{t-1} to reach a new set of samples with equal weights $\{\theta_{t-1}^{(j)}, 1\}_{j=1}^J$. We then draw samples $\{u_t^{(j)}\}_{j=1}^J$ for u_t and propagate $\theta_{t-1}^{(j)}$ to $\theta_t^{(j)}$ by (2.5). The new weight is updated as

$$w_t \propto p(y_t|\theta_t) \quad (2.8)$$

The complete algorithm is summarized in Figure 2.3. This algorithm was first introduced to the vision community by Isard and Blake [183] (called the CONDENSATION algorithm) to deal with a contour tracking problem.

```

Initialize a sample set  $\mathcal{S}_0 = \{\theta_0^{(j)}, 1\}_{j=1}^J$  according to prior distribution  $p(\theta_0)$ .
For  $t = 1, 2, \dots$ 
  For  $j = 1, 2, \dots, J$ 
    Resample  $\mathcal{S}_{t-1} = \{\theta_{t-1}^{(j)}, w_{t-1}^{(j)}\}$  to obtain a new sample  $(\theta_{t-1}^{(j)}, 1)$ .
    Predict the sample by drawing  $u_t^{(j)}$  for  $u_t$  and computing  $\theta_t^{(j)} = f_t(\theta_{t-1}^{(j)}, u_t^{(j)})$ .
    Compute the transformed image  $z_t^{(j)} = \mathcal{T}\{y_t; \theta_t^{(j)}\}$ .
    Update the weight using  $w_t^{(j)} = p(y_t|\theta_t^{(j)}) = p(z_t^{(j)}|\theta_t^{(j)})$ .
  End
  Normalize the weight using  $w_t^{(j)} = w_t^{(j)} / \sum_{j=1}^J w_t^{(j)}$ .
End

```

Figure 2.3. The general particle filter algorithm.

Often in time series, we are interested in deriving the best estimate $\hat{\theta}_t$ given the observations up to now. For example, the state estimate $\hat{\theta}_t$ can either be the minimum mean square error (MMSE) estimate,

$$\hat{\theta}_t = \theta_t^{mmse} = \mathbb{E}[\theta_t|y_{1:t}] \approx J^{-1} \sum_{j=1}^J w_t^{(j)} \theta_t^{(j)}, \quad (2.9)$$

where E the expectation operator, the maximum a posteriori (MAP) estimate,

$$\hat{\theta}_t = \theta_t^{map} = \arg \max_{\theta_t} p(\theta_t | y_{1:t}) \approx \arg \max_{\theta_t} w_t^{(j)}, \quad (2.10)$$

or other forms based on $p(\theta_t | y_{1:t})$.

Variations of particle filter

Sequential Importance Sampling (SIS) [237, 248] draws particles from a *proposal distribution* $q(\theta_t | \theta_{t-1}, y_{1:t})$ and then for each particle a proper weight is assigned as follows:

$$w_t \propto p(y_t | \theta_t) p(\theta_t | \theta_{t-1}) / q(\theta_t | \theta_{t-1}, y_{1:t}). \quad (2.11)$$

Selection of the proposal distribution $q(\theta_t | \theta_{t-1}, y_{1:t})$ is usually dependent on the application. We here focus on the extensions used in the vision literature. In the ICONDENSATION algorithm [184] which fuses low-level and high-level visual cues in the conventional CONDENSATION algorithm [183], the proposal distribution, a fixed Gaussian distribution for low-level color cue, is used to predict the particle configurations, then the posterior distribution of the high-level shape cue is approximated using SIS. It is interesting to note that two different cues can be combined together into one state vector to yield a robust tracker, using the co-inference algorithm [198] and the approach proposed in [196]. In the chapter on visual tracking, we also use a prediction scheme but our prediction is based on the same visual cue i.e. the appearance in the image, and it is directly used in the state transition model rather than used as a proposal distribution.

2.2 Reviews

In this section, we review two important topics in unconstrained face recognition: (i) face recognition under illumination, pose, and/or aging variations and (ii) face recognition from multiple still images or video sequences (including visual tracking).

2.2.1 Face recognition under illumination, pose and/or aging variations

We first characterize the three factors of illumination, pose, and identity in the context of face recognition under illumination and pose variations. We then address approaches on face recognition under illumination and/or pose variances. We also give a brief review of facial aging.

Identity, illumination, and pose

Three factors are involved in face recognition under pose and illumination variations, namely illumination, pose, and identity. Using the human face

images as examples, we now address issues involved in each of the three factors by fixing the other two.

- *Illumination.* Various illumination models are available in the literature, ranging from models for highly specular objects such as mirrors to models for matte objects. Mostly objects belong to the latter category and are described by Lambertian reflectance models for their simplicity. Early shape from shading approaches [10] assumed a constant albedo field. However, this assumption is violated at locations such as eyes and mouth edges. For the human face, the Lambertian reflectance model with a varying albedo field provides a reasonable approximation [75, 84, 144, 168, 95]. The Phong illumination model also has application [72]. Later, we adopt the Lambertian reflectance model with a varying albedo field to model the effect of illumination.

- *Pose.* The issue of pose essentially amounts to a correspondence problem. If dense correspondences across poses are available and if a Lambertian reflectance model is further assumed, a rank-1 constraint is implied because theoretically, a 3D model can be recovered and used to render novel poses. However, recovering a 3D model from 2D images is a difficult task. There are two types of approaches for recovering 3D models from 2D images: model-based and image-based. Model-based approaches [72, 215, 224, 226] require explicit knowledge of prior 3D models, while image-based approaches [190, 194, 218, 219, 221] do not use prior 3D models. In general, model-based approaches [72, 215, 224, 226] register the 2D face image to 3D models that are given beforehand. In [215, 226], a generative face model is deformed through bundle adjustment to fit 2D images. In [224], a generative face model is used to regularize the 3D model recovered using the Structure from motion (SfM) algorithm. In [72], 3D morphable models are constructed based on many prior 3D models. There are mainly three types of image-based approaches: SfM [190, 194], visual hull [218, 221], and light field rendering [219, 216] methods. The SfM approach [190] works with sparse correspondence and does not reliably recover the 3D model amenable for practical use. The visual hull methods [218, 221] assume that the shape of the object is convex, which is not always satisfied by the human face, and also require accurate calibration information. The light field rendering methods [219, 216] relax the requirement of calibration by a fine quantization of the pose space and recover a novel view by sampling the captured data that form the so-called light field. Later in 5, we propose an image-based method with no prior 3D models used. It handles a given set of views through an analysis analogous to the light field concept. However, no novel poses are rendered.

- *Identity.* One straightforward method to describe the identity is through discrete class labels. However, using this discrete description it is impossible to establish a link between objects used in the training and testing stages in terms of the identity. An alternative way is to associate the labels with continuous-valued features, which are regarded as an identity signatures. One good example is to use subspace encoding [50, 64], where linear generalization is assumed to incorporate the fact that all human faces are similar. Once the subspace basis are learned from the training set, they are used to characterize the gallery/probe set, thus enabling the required generalization capability.

Face recognition under illumination variation

Face recognition under illumination variation is a very challenging problem. The key is to successfully separate the illumination source from the observed appearance. Once separated, what remains is illuminant-invariant and appropriate for recognition. In addition to illumination variation, various issues embedded in the recognition setting make recognition even more difficult. We follow the FERET recognition protocol introduced in [60]. Assuming the availability of the following three sets, namely one training set, one gallery set, and one probe set, the recognition algorithm learns from the training set the characteristic features, associates descriptive features with the objects in the gallery set, and determines the identity for the objects in the probe set. Different recognition settings can be formed in terms of identity and illumination overlaps among the training, gallery, and probe sets. The most difficult setting, which is the focus of Chapter 4, is obviously the one in which there is no overlap at all among the three sets in terms of both identity and illumination, except the identity overlap between the gallery and probe sets. In this setting, generalizations from known illumination to unknown illumination and from known identities to unknown identities are particularly desired.

Existing approaches can be grouped into three streams: subspace methods, reflectance-model methods, and 3D-model-based methods. (i) The first approach is very popular for the recognition problem. After removing the first three eigenvectors, PCA was reported to be more robust to illumination variation than the ordinary PCA or the ‘Eigenface’ approach [64]. The ‘Fisherface’ approach [44, 77] used LDA to handle illumination variations. In general, subspace learning methods are able to capture the generic face space and thus recognize new objects not present in the training set. The disadvantage is that subspace learning is actually tuned to the lighting conditions of the training set; therefore if the illumination conditions are not similar among the training, gallery, and probe sets, recognition performance may not be acceptable. (ii) The second approach [144, 75, 161, 84, 90, 93] employs a Lambertian reflectance model with a varying albedo field, mostly ignoring both attached and cast shad-

ows. The main disadvantage of this approach is the lack of generalization from known objects to unknown objects, with the exception of [84, 90]. In [84], Shashua and Raviv used an ideal-class assumption. All objects belonging to the ideal class are assumed to have the same shape. The work of Zhang and Samarasinghe [90] utilized the regularity of the harmonic image exemplars to perform face recognition under varying light. (iii) The third approach employs 3D models. The ‘Eigenhead’ approach [71] assumes that the 3D geometry (or 3D depth information) of any face lies in a linear space spanned by the 3D geometry of the training ensemble and uses a constant albedo field. The morphable model approach [72] is based on a synthesis-and-analysis strategy. It is able to handle both illumination and pose variations with illumination directions specified. The weakness of the 3D model approaches is that they require 3D models and complicated fitting algorithms.

Face recognition under pose variation

As mentioned earlier, pose variation essentially amounts to a correspondence problem. Unfortunately, finding correspondences is a very difficult task and, therefore there exists no subspace based on an appearance representation when confronted with pose variation. Approaches to face recognition under pose variation [75, 76, 81] avoid the correspondence problem by sampling the continuous pose space into a set of poses, *v.i.z.* storing multiple images at different poses for each person at least in the training set. In [81], view-based ‘Eigenfaces’ are learned from the training set and used for recognition. In [75], a denser sampling is used to cover the pose space. However, as [75] uses object-specific images, appearances belonging to a novel object (*i.e.* not in the training set) cannot be handled. In [76], the concept of light field [219] is used to characterize the continuous pose space. ‘Eigen’ light fields are learnt from the training set. However, the implementation of [76] still discretizes the pose space and recognition can be based on probe images at poses in the discretized set. One should note that the light field is not related to variation in illumination.

Face recognition under illumination and pose variations

Approaches to handling both illumination and pose variations include [72, 77, 88, 89, 94]. The approach [72] uses 3D morphable models to characterize the human faces. Both geometry and texture are linearly spanned by those of the training ensemble consisting of 3D prior models. It is able to handle both illumination and pose variations. Its only weakness is a complicated fitting algorithm. Recently, a fitting algorithm more efficient than suggested in [72] is proposed in [83]. In [77], the Fisher light field is proposed to handle both illumination and pose variations, where the light field is used to cover the pose variation and the LDA to cover the illumination variation. Since discriminant analysis is just a statistical analysis tool which minimizes the within-class scatter while

maximizing the between-class scatter and has no relationship with any physical illumination model, it is doubtful if discriminant analysis is able to generalize to new lighting conditions. Instead, this generalization may be inferior because discriminant analysis tends to overly tune to the lighting conditions in the training set. The ‘Tensorface’ approach [88, 89] uses a multilinear analysis to handle various factors such as identity, illumination, pose, and expression. The factors of identity and illumination are suitable for linear analysis, as evidenced by the ‘Eigenface’ approach (assuming a fixed illumination and a fixed pose) and the subspace induced by the Lambertian model, respectively. However, the factor of expression is arguably amenable for linear analysis and the factor of pose is not amenable for linear analysis. In [94], preliminary results are reported by first warping the albedo and surface normal fields at the desired pose and then performing recognition as usual.

Facial aging

While studying the role played by these external factors in affecting face recognition is crucial, it is important to study the role played by natural phenomenon such as facial aging. Aging effects on human faces manifest in different forms in different ages. While aging effects are manifested more in terms of changes in the cranium’s shape during one’s younger years, they are manifested more in terms of wrinkles and other skin artifacts during one’s older years. Here, we provide a brief overview of the literature on facial aging.

Pittenger and Shaw [140] characterized the growth of human faces as a visco-elastic event and proposed shear & strain transformations to model the changes in the shape of face profiles due to growth. They studied the effects of shear and strain transformations on the perceived age. O’Toole et al. [139] applied a standard facial caricaturing algorithm to three dimensional models of faces and reported an increase in the perceived age of faces when facial creases were exaggerated into wrinkles and a decrease when such creases were de-emphasized.

Lanitis et al. [136] proposed a method for simulating aging effects on face images. On a database of age progressive images of individuals each under 30 years of age, they used a combined shape-intensity model to represent faces. They modeled age as a quadratic function of the PCA coefficients extracted from the model parameters. They reported results on experiments such as estimating the age of an individual from his/her face image; simulating aging effects on face images etc. In [137], Lanitis *et al.* used a similar framework as defined in [136] on a similar data set and evaluated the performance of three age classifiers: the first was a quadratic function of the model parameters; the second was based on the distribution of model parameters; and the third was based on supervised and unsupervised neural networks trained on the model parameters.

Tiddeman et al. [141] developed a model for aging face images by transforming facial textures. Face images were represented in terms of 2D shape vectors and pixel intensities. They developed prototype faces by averaging the 2D shape vectors and pixel intensities across a set of face images under each age group. To age a face image, they superimposed the difference in 2D shape vectors and pixel intensities of the prototype faces on to the face image. Further, they simulated wrinkles in face images by employing locally weighted wavelet functions at different scales and orientations and thereby enhanced the edge amplitudes. Their experimental evaluation reported significant increase in the perceived age of subjects. Wu et.al. [142] came up with a dynamic model to simulate wrinkles in 3D facial animation and skin aging. They represented skin deformations as plastic-visco-elastic processes and generated permanent wrinkles through a simulation of inelastic skin deformations. Givens et.al. [135] analyzed the role of various co-variables such as age, gender, expression, facial hair etc in affecting recognition and noted that older faces were easily recognized by three face recognition algorithms.

2.2.2 Face recognition from multiple stills or videos

In this section, we review related literature on face recognition from multiple still images, or from video sequences. Since visual tracking is an integrated part in video-based recognition, we also briefly review the literature on tracking in the end.

Three properties

It is obvious that multiple still images or a video sequence can be regarded as a single still image in a degenerate manner [103, 107, 117, 119, 120]. More specifically, suppose that we have a single-still-image-based FR algorithm \mathcal{A} (or the base algorithm) by some means, we can construct a recognition algorithm based on multiple still images or a video sequence by combining multiple base algorithms denoted by \mathcal{A}_i 's. Each \mathcal{A}_i takes a different single image as input, coming from the multiple still images or video sequences. The combination rule can be ad hoc chosen to be additive, multiplicative, and so on.

However, the fused algorithms completely neglect additional properties possessed by multiple still images or video sequences, which are not present in a still image. In particular, three properties manifest themselves, which motivated various approaches recently proposed in the literature.

- 1 [P1: **Set of observations**]. This property is directly utilized by the fused algorithms. One main disadvantage may be the *ad hoc* nature of the combination rule. However, theoretical analysis based on a set of observations can be principally derived. For example, a set of observations can be sum-

marized using quantities like matrix, probability density function, manifold, etc. Hence, corresponding knowledge can be utilized to match two sets.

- 2 [P2: **Temporal continuity/Dynamics**]. Successive frames in the video sequences are continuous in the temporal dimension. Such continuity, coming from facial expression, geometric continuity related to head and/or camera movement, or photometric continuity related to changes in illumination, provides an additional constraint for modeling face appearance. In particular, temporal continuity can be further characterized using kinematics. For example, facial expression and head movement when an individual participates certain activity result in structured changes in face appearance. Modeling of such a structured change (or dynamics) further regularizes FR.
- 3 [P3: **3D model**]. This means that we are able to reconstruct a 3D model from a group of still images and a video sequence. Recognition can then be based on the 3D model. Using the 3D model provides possible invariance to pose and illumination.

Clearly, the first and third properties are shared by multiple still images and video sequences. The second property is solely possessed by video sequences.

Below, we review various face recognition approaches utilizing these properties in one or more ways. Generally speaking, the newly designed algorithms are better in terms of recognition performance, computational efficiency, etc.

P1: Approaches utilizing set of observations

Four rules of summarizing a set of observations have been presented. In general, different data representations are utilized to describe multiple observations and corresponding distance functions based on the presentations are invoked for recognition.

One image or several images

Algorithms designed by representing a set of observations into one image or several images and then applying the combination rules are essentially still-image-based and hence are not reviewed here.

Matrix

Yamaguchi *et al.* [121] proposed the so-called *Mutual Subspace Method* (MSM) method. In this method, the matrix representation is used and the similarity function between two matrices is defined as the angle between two subspaces of the matrices (also referred to as principal angle or canonical correlation coefficient). Suppose that the columns of X and Y represent two subspaces U_X and U_Y , the principle angle θ between the two subspaces is defined as

$$\cos(\theta) = \max_{u \in U_X} \max_{v \in U_Y} \frac{u^T v}{\sqrt{u^T u} \sqrt{v^T v}}. \quad (2.12)$$

It can be shown that the principle angle θ is equal to the largest singular value of the matrix $U_X^T U_Y$ where U_X and U_Y are orthogonal matrices encoding the column bases of the X and Y matrices, respectively.

In general, the leading singular values of the matrices $U_X^T U_Y$ defines a series of principal angles $\{\theta_k\}$'s.

$$\cos(\theta_k) = \max_{u \in U_X} \max_{v \in U_Y} \frac{u^T v}{\sqrt{u^T u} \sqrt{v^T v}} \quad (2.13)$$

subject to:

$$u^T u_i = 0, v^T v_i = 0, i = 1, 2, \dots, k-1. \quad (2.14)$$

Yamaguchi *et al.* [121] recorded a database of 101 individuals posing variation in facial expression and pose. They discovered that the MSM method is more robust to noisy input image or face normalization error than the still-image-based method that is referred to as conventional subspace method (CSM) in [121]. The similarity function of the MSM method is more stable and consistent than that of the CSM method.

Wolf and Shashua [275] extended the computation of the principal angles into the RKHS. Kernel principal angles between two matrices X and Y are then based on their 'kernelized' versions $\phi(X)$ and $\phi(Y)$. A 'kernelized' matrix $\phi(X)$ of $X = [x_1, x_2, \dots, x_n]$ is defined as $\phi(X) = [\phi(x_1), \phi(x_2), \dots, \phi(x_n)]$. The key is to evaluate the matrix $U_{\phi(X)}^T U_{\phi(Y)}$ defined in RKHS. In [275], Wolf and Shashua showed the computation using the 'kernel trick'.

Another contribution of Wolf and Shashua [275] is that they further proposed a positive kernel function taking matrix as input. Given such a kernel function, it can be readily plugged into a classification scheme such as a support vector machine (SVM) [18]. Face recognition using multiple still images, coming from a tracked sequence, were studied and the proposed kernel principal angles slightly outperforms other non-kernel versions.

Zhou [278] systematically investigated the kernel functions taking matrix as input (also referred to as matrix kernels). More specifically, the following two functions are kernel functions.

$$k_{\bullet}(X, Y) = \text{tr}(X^T Y), \quad k_{\star}(X, Y) = \det(X^T Y), \quad (2.15)$$

where tr and \det are matrix trace and determinant. They are called as matrix trace and determinant kernels. Using them as building blocks, Zhou [278] constructed more kernels based on the column basis matrix, the 'kernelized' matrix, and the column basis matrix of the 'kernelized' matrix.

$$k_{U_{\bullet}}(X, Y) = \text{tr}(U_X^T U_Y), \quad k_{U_{\star}}(X, Y) = \det(U_X^T U_Y), \quad (2.16)$$

$$k_{\phi_{\bullet}}(X, Y) = \text{tr}(\phi(X)^T \phi(Y)), \quad k_{\phi_{\star}}(X, Y) = \det(\phi(X)^T \phi(Y)), \quad (2.17)$$

$$k_{\mathbf{U}_{\phi\bullet}}(\mathbf{X}, \mathbf{Y}) = \text{tr}(\mathbf{U}_{\phi(\mathbf{X})}^T \mathbf{U}_{\phi(\mathbf{Y})}), \quad k_{\mathbf{U}_{\phi\star}}(\mathbf{X}, \mathbf{Y}) = \det(\mathbf{U}_{\phi(\mathbf{X})}^T \mathbf{U}_{\phi(\mathbf{Y})}). \quad (2.18)$$

Probability density function (PDF)

Shakhnarovich *et al.* [118] used multivariate normal density for summarizing face appearances and the Kullback-Leibler (KL) divergence or relative entropy for recognition. The KL divergence between two Gaussian densities $p_1 = \mathcal{N}(\mu_1, \Sigma_1)$ and $p_2 = \mathcal{N}(\mu_2, \Sigma_2)$ can be explicitly computed as

$$\begin{aligned} KL(p_1||p_2) &= \int_{\mathbf{x}} p_1(\mathbf{x}) \log \frac{p_1(\mathbf{x})}{p_2(\mathbf{x})} d\mathbf{x} \\ &= \frac{1}{2} \log \left(\frac{|\Sigma_2|}{|\Sigma_1|} \right) + (\mu_1 - \mu_2)^T \Sigma_2^{-1} (\mu_1 - \mu_2) \\ &\quad + \frac{1}{2} \text{tr}(\Sigma_1 \Sigma_2^{-1}) - \frac{d}{2}, \end{aligned} \quad (2.19)$$

where d is the dimensionality of the data. One disadvantage of the KL divergence is that it is asymmetric. To make it symmetric, they used $KL(p_1||p_2) + KL(p_2||p_1)$. Shakhnoarovich *et al.* [118] achieved better performance than the MSM approach by Yamaguchi *et al.* [121] on a dataset including 29 subjects.

Other than the KL divergence, probabilistic distance measures such as Chernoff distance and Bhattacharyya distance can also be used. The Chernoff distance is defined and computed in the case of normal density as:

$$\begin{aligned} J_C(p_1, p_2) &= -\log \left\{ \int_{\mathbf{x}} p_1^{\alpha_2}(\mathbf{x}) p_2^{\alpha_1}(\mathbf{x}) d\mathbf{x} \right\} \\ &= \frac{1}{2} \alpha_1 \alpha_2 (\mu_1 - \mu_2)^T [\alpha_1 \Sigma_1 + \alpha_2 \Sigma_2]^{-1} (\mu_1 - \mu_2) \\ &\quad + \frac{1}{2} \log \frac{|\alpha_1 \Sigma_1 + \alpha_2 \Sigma_2|}{|\Sigma_1|^{\alpha_1} |\Sigma_2|^{\alpha_2}}, \end{aligned} \quad (2.20)$$

where $\alpha_1 > 0$, $\alpha_2 > 0$ and $\alpha_1 + \alpha_2 = 1$. When $\alpha_1 = \alpha_2 = 1/2$, the Chernoff distance reduces to the Bhattacharyya distance.

In [265], Jebara and Kondon proposed probability product kernel function

$$k(p_1, p_2) = \int_{\mathbf{x}} p_1^r(\mathbf{x}) p_2^r(\mathbf{x}) d\mathbf{x}, \quad r > 0. \quad (2.21)$$

When $r = 1/2$, the kernel function k reduces to the so-called Bhattacharyya kernel since it is related to the Bhattacharyya distance. When $r = 1$, the kernel function k reduces to the so-called expected likelihood kernel. In practice, we can simply use the kernel function k as a similarity function.

However, the Gaussian assumption can be ineffective when modeling the nonlinear face appearance manifold. In [277] (also Chapter 7), Zhou and Chelappa modeled the nonlinearity through a different approach: kernel methods.

Since a nonlinear function is used, albeit in an implicit fashion, Zhou and Chelappa [277] investigated their uses in a different space. To be specific, analytic expressions for probabilistic distances that account for nonlinearity or high-order statistical characteristics of the data are derived. On a dataset involving subjects presenting appearances with pose and illumination variations, the probabilistic distance measures performed better than their non-kernel counterparts.

Recently, Arandjelović and Cipolla [98] used resistor-average distance (RAD) for video-based recognition.

$$RAD(p_1, p_2) = (KL(p_1||p_2)^{-1} + KL(p_2||p_1)^{-1})^{-1}. \quad (2.22)$$

Further, computation of the RAD was conducted on the RKHS to absorb non-linearity of face manifold. Some robust techniques such as synthesizing images to account for small localization errors and RANSAC algorithms to reject outliers were introduced to achieve improved performance. They [99] further extended their work to use the symmetric KL divergence distance between two mixture-of-Gaussian densities.

Manifold

Fitzgibbon and Zisserman [104] proposed to compute a joint manifold distance to cluster appearances. A manifold is captured by subspace analysis which is fully specified by a mean and a set of basis vectors. For example, a manifold \mathcal{P} can be represented as

$$\mathcal{P} = \{\mathbf{m}_p + \mathbf{B}_p \mathbf{u} | \mathbf{u} \in \mathcal{U}\} \quad (2.23)$$

where \mathbf{m}_p is the mean and \mathbf{B}_p encodes the basis vectors. In addition, the authors invoked affine transformation to overcome geometric deformation. The joint manifold distance between \mathcal{P} and \mathcal{Q} is defined as

$$d(\mathcal{P}, \mathcal{Q}) = \min_{\mathbf{u}, \mathbf{v}, \mathbf{a}, \mathbf{b}} \|T(\mathbf{m}_p + \mathbf{B}_p \mathbf{u}, \mathbf{a}) - T(\mathbf{m}_q + \mathbf{B}_q \mathbf{v}, \mathbf{b})\|^2 + E(\mathbf{a}) + E(\mathbf{b}) + E(\mathbf{u}) + E(\mathbf{v}), \quad (2.24)$$

where $T(\mathbf{x}, \mathbf{a})$ transforms image \mathbf{x} using the affine parameter \mathbf{a} and $E(\mathbf{a})$ is the prior cost incurred by invoking the parameter \mathbf{a} .

In experiments, Fitzgibbon and Zisserman [104] performed automatic clustering of faces in feature-length movies. To reduce the lighting effect, the face images are high-pass filtered before subject to a clustering step. The authors reported that sequence-to-sequence matching presents a dramatic computational speedup when compared with pairwise image-to-image matching.

Identity surface is a manifold, proposed by Li *et al.* in [114], that depicts face appearances presented in multiple poses. The pose is parameterized by yaw α and tilt θ . Face image at (α, θ) is first fitted to a 3D point distribution model and an active appearance model. After pose estimation, the face appearance is

warped to a canonical view to provide a pose-free representation from which a nonlinear discriminatory feature vector is extracted. Suppose that the feature vector is denoted by \mathbf{f} , the function $\mathbf{f}(\alpha, \theta)$ defines the identity surface that is pose-parameterized. In practice, since only a discrete set of views are available, the identity surface is approximated by piece-wise planes. The manifold distance between two manifolds $\mathcal{P} = \{\mathbf{f}_p(\alpha, \theta)\}$ and $\mathcal{Q} = \{\mathbf{f}_q(\alpha, \theta)\}$ is defined as

$$d(\mathcal{Q}, \mathcal{P}) = \int_{\alpha} \int_{\theta} w(\alpha, \theta) d(\mathbf{f}_q(\alpha, \theta), \mathbf{f}_p(\alpha, \theta)) d\alpha d\theta. \quad (2.25)$$

where $w(\alpha, \theta)$ is a weight function.

A video sequence corresponds to a trajectory traced out in the identity surface. Suppose that video frames sample the pose space at $\{\alpha_j, \theta_j\}$, the following distance $\sum_j w_j d(\mathbf{f}_q(\alpha_j, \theta_j), \mathbf{f}_p(\alpha_j, \theta_j))$ is used for video-based FR. In the experiments, 12 subjects were involved and a 100% recognition accuracy was achieved.

P2: Approaches utilizing temporal continuity/dynamics

Simultaneous tracking and recognition is an approach proposed by Zhou *et al.* [129] that systematically studied how to incorporate temporal continuity in video-based recognition. Zhou *et al.* modeled two tasks involved, namely tracking and recognition, in a probabilistic framework. A time series model is used, with the state vector (n_t, θ_t) where n_t is the identity variable and θ_t is the tracking parameter, and the observation \mathbf{y}_t (i.e. the video frame). The time series model is fully specified by the state transition probability $p(n_t, \theta_t | n_{t-1}, \theta_{t-1})$ and the observational likelihood $p(\mathbf{y}_t | \theta_t, n_t)$. This is covered in detail in Chapter 10.

In the work of Zhou *et al.* [129], in addition to the case that the gallery consists of one still image per individual, they also extended the approach to handle video sequence in the gallery set. Representative exemplars are learned from the gallery video sequences to depict individuals. Then simultaneous tracking and recognition was invoked to handle video sequences in the probe set. Li and Chellappa [112] also proposed an approach somewhat similar to [129]. In [112], only tracking was implemented using SIS and recognition scores were subsequently derived based on tracking results.

Lee *et al.* [109] performed video-based face recognition using probabilistic appearance manifolds. The main motivation is to model appearances under pose variation, i.e., a generic appearance manifold consists of several pose manifolds. Since each pose manifold is represented using a linear subspace, the overall appearance manifold is approximated by piecewise linear subspaces. The learning procedure is based on face exemplars extracted from a video sequence. K-means clustering is first applied and then for each cluster principal component analysis is used for subspace characterization.

In addition, the transition probabilities between pose manifolds are also learned. The temporal continuity is directly captured by the transition probabilities. In general, the transition probabilities between neighboring poses (such as frontal pose to left pose) are higher than those between far-apart poses (such as left pose to right pose). Recognition also reduces to computing posterior distribution.

In experiments, Lee *et al.* compared three methods that use temporal information differently: the proposed method with learned transition matrix, the proposed method with uniform transition matrix (meaning that temporal continuity is lost), and majority voting. The proposed method with learned transition matrix achieved a significantly better performance than the other two methods. Recently, Lee and Kriegman [110] extended [109] by learning the appearance manifold from a testing video sequence in an online fashion.

Liu and Chen [115] used adaptive hidden Markov model (HMM) to depict the dynamics. HMM is a statistical tool to model time series. Usually, the HMM is denoted by $\lambda = (A, B, \pi)$, where A is the state transition probability matrix, B is the observation PDF, and π is the initial state distribution. Given a probe video sequence Y , its identity is determined as

$$\hat{n} = \arg \max_{1,2,\dots,N} p(Y|\lambda_n), \quad (2.26)$$

where $p(Y|\lambda_n)$ is the likelihood of observing the video sequence Y given the model λ_n . In addition, when certain conditions hold, HMM λ_n was adapted to accommodate the appearance changes in the probe video sequence that results in improved modeling over time. Experimental results on various datasets demonstrated the advantages of using the adaptive HMM.

Aggarwal *et al.* [100] proposed a system identification approach for video-based FR. The face sequence is treated as a first-order auto-regressive and moving averaging (ARMA) random process.

$$\theta_{t+1} = A\theta_t + \mathbf{v}_t, \quad y_t = C\theta_t + \mathbf{w}_t, \quad (2.27)$$

where $\mathbf{v}_t \sim \mathcal{N}(0, Q)$ and $\mathbf{w}_t \sim \mathcal{N}(0, R)$. System identification is equivalent to estimating the parameters A , C , Q , and R from the observations $\{y_1, y_2, \dots, y_T\}$. Once system is identified or each video sequence is associated with its parameters, video-to-video recognition uses various distance metrics constructed based on the parameters. Promising experimental results (over 90%) were reported when significant pose and expression variations are present in the video sequences.

Facial expression analysis is also related to temporal continuity/dynamics, but not directly related to FR. Examples of expression analysis include [45, 62]. A review of face expression analysis is beyond the scope of this chapter.

P3: Approaches utilizing 3D model

There is a large body of literature on SfM. However, the current SfM algorithms do not reliably reconstruct the 3D face model. There are three difficulties in the SfM algorithm. The first lies in the ill-posed nature of the perspective camera model that results in instability of the SfM solution. The second is that the face model is not a truly rigid model especially when facial expressions and other deformations are present. The final difficulty is related to the input to the SfM algorithm. This is usually a sparse set of feature points provided by a tracking algorithm that itself has many flaws. Interpolation from a sparse set of feature points to a dense set is very inaccurate.

To relieve the first difficulty, orthographic and paraperspective models are used to approximate the perspective camera model. Under such approximate models, the ill-posed problem becomes well-posed. In Tomasi and Kanade [194], the orthographic model was used and a matrix factorization principle was discovered. The factorization principle was extended to the paraperspective camera model in Poelman and Kanade [223]. Factorization under uncertainty was considered in [175, 182].

The second difficulty is often resolved by imposing a subspace constraint on the face model. Bregler *et al.* [176] proposed to regularize the nonrigid face model by using the linear constrain. It was shown that factorization can be still be obtained. Brand [175] considered such factorization under uncertainty. Xiao *et al.* [228] discovered a closed form solution to nonrigid shape and motion recovery.

Interpolation from a sparse set to a dense depth map is always a difficult task. To overcome this, a dense face model is used instead of interpolation. However, the dense face model is only a generic model and hence may not be appropriate for a specific individual. Bundle adjustment [215, 226] is a method that adjust the generic model directly to accommodate the video observation. Roy-Chowdhury and Chellappa [224] took a different approach for combining the 3D face model recovered from the SfM algorithm with the generic face model. Jebara and Pentland [108] regularized the SfM using a parameterized face model built from a training set.

The SfM algorithm mainly recovers the geometric component of the face model, i.e., the depth value of every pixel. Its photometric component is naively set to the appearance in one reference video frame. Image-based rendering method, on the other hand, directly recovers the photometric component of the 3D model. Light field rendering [216, 219] in fact bypasses the stage of recovering the photometric of the 3D model but rather recovers the novel views directly. The light field rendering methods [216, 219] relax the requirement of calibration by a fine quantization of the pose space and recover a novel view by sampling the captured data that form the so-called light field. The ‘eigen’ light field approach developed by Gross *et al.* [76] assumes a subspace assumption

of the light field. In Zhou and Chellappa [97] (described in detail in Chapter 5), the light field subspace and the illumination subspace are combined to arrive at a bilinear analysis. Another line of research relates to 3D model recovery using the visual hull methods [218, 221]. But, the visual hull method assumes that the shape of the object is convex, which is not always satisfied by the human face, and also requires accurate calibration information. Direct use of visual hull for FR is not found in the literature.

To characterize both the geometric and photometric components of the 3D face model, Blanz and Vetter [72] fitted a 3D morphable model to a single still image. The 3D morphable model uses a linear combination of dense 3D models and texture maps. In principle, the 3D morphable model can be fitted to multiple images. The 3D morphable model can be thought of as an extension of 2D active appearance model [212] to 3D, but the 3D morphable model uses dense 3D models. Xiao *et al.* [229] proposed to combine a linear combination of 3D sparse model and a 2D appearance model.

Although there is significant interest in recovering the 3D model, directly performing FR using the 3D model is a recent trend [101, 116, 102]. Blanz and Vetter [72] implicitly did so by using the combining coefficients for recognition. Beumier and Acheroy [101] conducted matching based on 2D sections of the facial surface. Mavridis *et al.* [116] used 3D+color camera to perform face recognition. Bronstein *et al.* [102] used a 3D face model for compensating the effect of facial expression in face recognition. However, the above approaches use the 3D range data as input. Because in this chapter we are mainly interested in face recognition from multiple still images or video sequence, a thorough review of face recognition based on 3D range data is beyond its scope.

Future approaches from multiple stills or videos

Thus far, we reviewed the approaches that utilize the three properties. Although they usually achieved good recognition performance, they have their own assumptions or limitations. For example, the Gaussian distribution used in Shakhnoarovich *et al.* [118] is easily violated by pose and illumination variations. The HMM used in Liu and Chen [115] poses a strong constraint on the change of face appearance that is not satisfied by video sequences that contain an arbitrarily moving face.

In this section, we forecast possible new approaches. These new approaches either arise from new representation for more than one still image or extend the capability of the existing approaches.

New representation

In the matrix representation, multiple observations are encoded using a matrix. In other words, each observation is an image that is ‘vectorized’. The ‘vectorization’ operator ignores the spatial relationship of the pixels.

To fully characterize the spatial relationship, a tensor can be used in lieu of matrix. Here, a tensor is understood as a 3-D array. Tensor representation is used in Vasilescu and Terzopoulos [88] to learn a generic model of the face appearance for all humans, at different views, and under different illuminating conditions, etc. However, comparing two tensors has not been investigated in the literature.

In principle, the PDF representation is very general. But in the experiment, a certain parametric form is assumed as in Shakhnoarovich *et al.* [118]. Other PDF forms can be employed. The key is to find an appropriate density that can model the face appearance. The same problem appears in the manifold description. With advances in manifold modeling, FR based on manifold can be improved too.

Using the training set

The training set is usually used to provide a generic model of face appearances of all humans, while the images in the gallery set is related to an individualized model of face appearance belonging to the same person. If there are enough observations, one can build an accurate model of the face for each individual in the gallery set and hence the knowledge of the training set is not necessary. If the number of images is not sufficiently large, one should combine the knowledge of a generic model with the individualized model to describe the identity signature.

3D model comparison

As mentioned earlier, comparison between two 3D models has not been fully investigated yet. In particular, direct comparison of the geometric component of the 3D model is rather difficult because it is nontrivial to recover the 3D model in the first place and the correspondence between two 3D models cannot be easily established.

Current approaches [229] warp the model to the frontal view and use the frontal 2D face appearance for recognition. However, these approaches are very sensitive to illumination variation. Generalized photometric stereo [95] can be incorporated into these approaches for a more accurate model.

The most sophisticated 3D model is to use a statistical description. In other words, both the geometric component g and the texture component f have their distributions, say $p(g)$ and $p(f)$, respectively. Such distributions can be learned from multiple still images/video sequence. Probabilistic matching can then be applied for FR.

Utilizing more than one property

Most of the approaches reviewed earlier utilize only one of the three properties. However, these properties are not overlapping in the sense that more than one property can be unified to achieve further improvements.

Probabilistic identity characterization [131] proposed in Chapter 11 is an example of integrating the properties $P1$ and $P2$, where FR from multiple still images and FR from video sequences are unified in one framework.

Statistical 3D model is a combination of properties $P1$ and $P3$, where the PDF part of property $P1$ is used.

Visual tracking

Roughly speaking, previous work on visual tracking can be divided into two groups with no clearly defined boundaries: deterministic tracking and stochastic tracking. Our approach combines the merits of both stochastic and deterministic tracking approaches in a unified framework using a particle filter. We give below a brief review of both approaches.

Deterministic approaches usually reduce to an optimization problem, e.g., minimizing an appropriate cost function. The definition of the cost function is a key issue. A common choice in the literature is the sum of squared distance (SSD) used in many optical flow approaches [180]. In fact, using SSD is equivalent to using a model where the noise obeys an iid Gaussian distribution; therefore this case can also be viewed as stochastic tracking. A gradient descent algorithm is most commonly used to find the minimum. Very often, only a local minimum can be reached. In [180], the cost function is defined as the SSD between the observation and a fixed template, and the motion is parameterized as affine. Hence the task is to find the affine parameter minimizing the cost function. Using a Taylor series expansion and keeping only the first-order terms, a linear prediction equation is obtained. It has been shown that for the affine case, the system matrix can be computed efficiently since a fixed template is used. Mean shift [178] is an alternative deterministic approach to visual tracking, where the cost function is derived from the color histogram.

Stochastic tracking approaches often reduce to an estimation problem, e.g., estimating the state for a time series state space model. Early works [171, 177] used the Kalman filter or its variants [1]. However, this restricts the type of model that can be used. Recently sequential Monte Carlo (SMC) algorithms [6, 179, 245, 248], which can model nonlinear/non-Gaussian cases, have gained prevalence in the tracking literature due in part to the CONDENSATION algorithm [183]. Stochastic tracking improves robustness over its deterministic counterpart by its capability for escaping the local minimum since the searching directions are for the most part random even though they are governed by a deterministic state transition model. Toyama and Blake [195] proposed a probabilistic paradigm for tracking with the following properties: Exemplars are learned from the raw training data and embedded in a mixture density; the kinematics is also learned; the likelihood measurement is constructed on a metric space. However, as far as computational load is concerned, stochastic algorithms in general are more intense. Note that the stochastic approaches can often be formulated as optimization problems.



<http://www.springer.com/978-0-387-26407-3>

Unconstrained Face Recognition

Zhou, S.K.; Chellappa, R.; Zhao, W.

2006, XII, 244 p. 20 illus., Hardcover

ISBN: 978-0-387-26407-3