Chapter 14

Recognition

R. Szeliski, *Computer Vision: Algorithms and Applications*, Texts in Computer Science,
DOI 10.1007/978-1-84882-935-0_14, © Springer-Verlag London Limited 2011

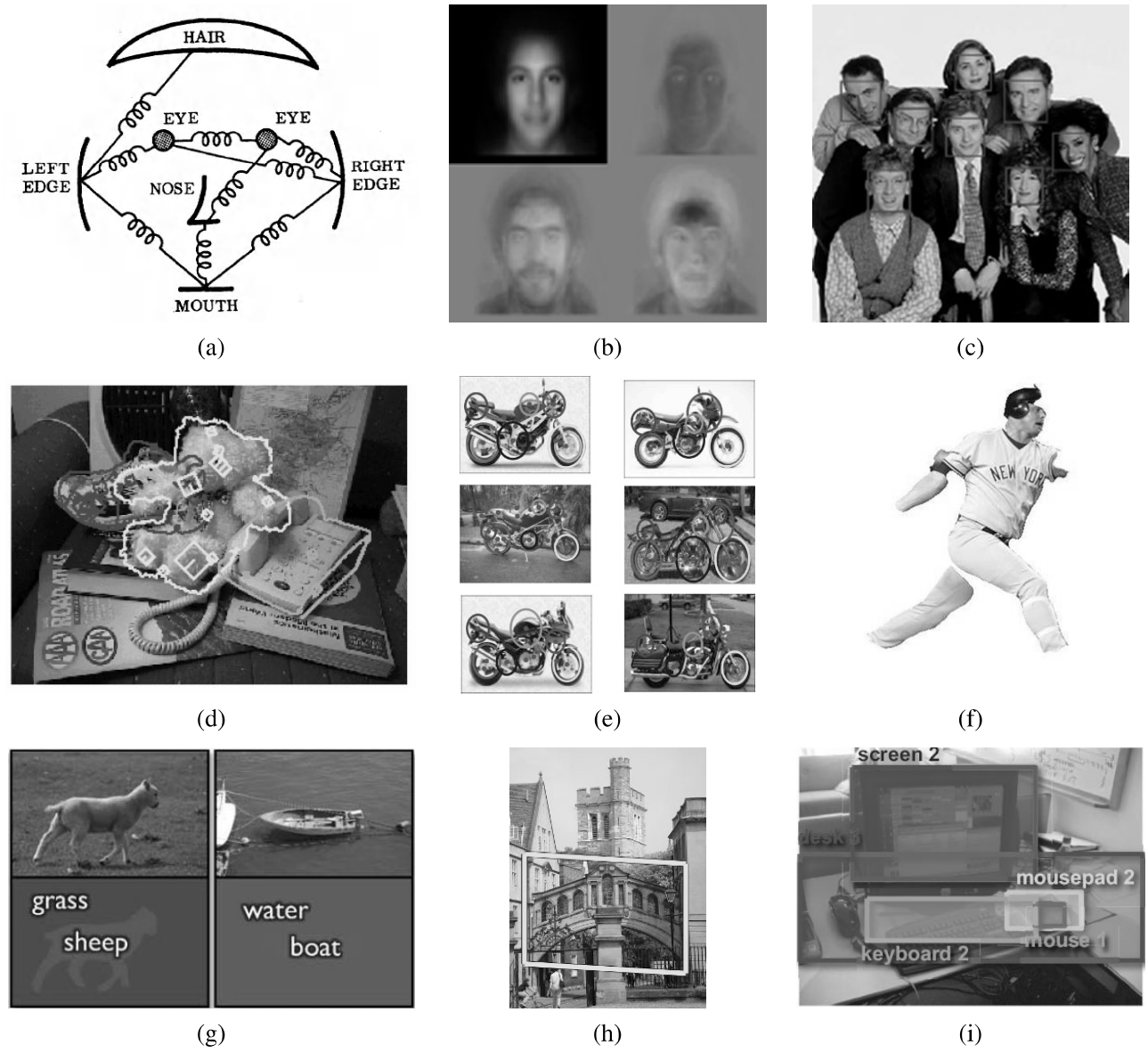


Figure 14.1 Recognition: face recognition with (a) pictorial structures (Fischler and Elschlager 1973) © 1973 IEEE and (b) eigenfaces (Turk and Pentland 1991b); (c) real-time face detection (Viola and Jones 2004) © 2004 Springer; (d) instance (known object) recognition (Lowe 1999) © 1999 IEEE; (e) feature-based recognition (Fergus, Perona, and Zisserman 2007); (f) region-based recognition (Mori, Ren, Efros *et al.* 2004) © 2004 IEEE; (g) simultaneous recognition and segmentation (Shotton, Winn, Rother *et al.* 2009) © 2009 Springer; (h) location recognition (Philbin, Chum, Isard *et al.* 2007) © 2007 IEEE; (i) using context (Russell, Torralba, Liu *et al.* 2007).

Of all the visual tasks we might ask a computer to perform, analyzing a scene and recognizing all of the constituent objects remains the most challenging. While computers excel at accurately reconstructing the 3D shape of a scene from images taken from different views, they cannot name all the objects and animals present in a picture, even at the level of a two-year-old child. There is not even any consensus among researchers on when this level of performance might be achieved.

Why is recognition so hard? The real world is made of a jumble of objects, which all occlude one another and appear in different poses. Furthermore, the variability intrinsic within a class (e.g., dogs), due to complex non-rigid articulation and extreme variations in shape and appearance (e.g., between different breeds), makes it unlikely that we can simply perform exhaustive matching against a database of exemplars.[1]

The recognition problem can be broken down along several axes. For example, if we know what we are looking for, the problem is one of *object detection* (Section 14.1), which involves quickly scanning an image to determine where a match may occur (Figure 14.1c). If we have a specific rigid object we are trying to recognize (*instance recognition*, Section 14.3), we can search for characteristic feature points (Section 4.1) and verify that they align in a geometrically plausible way (Section 14.3.1) (Figure 14.1d).

The most challenging version of recognition is general *category* (or *class*) recognition (Section 14.4), which may involve recognizing instances of extremely varied classes such as animals or furniture. Some techniques rely purely on the presence of features (known as a "bag of words" model—see Section 14.4.1), their relative positions (*part-based models* (Section 14.4.2)), Figure 14.1e, while others involve segmenting the image into semantically meaningful regions (Section 14.4.3) (Figure 14.1f). In many instances, recognition depends heavily on the *context* of surrounding objects and scene elements (Section 14.5). Woven into all of these techniques is the topic of *learning* (Section 14.5.1), since hand-crafting specific object recognizers seems like a futile approach given the complexity of the problem.

Given the extremely rich and complex nature of this topic, this chapter is structured to build from simpler concepts to more complex ones. We begin with a discussion of face and object detection (Section 14.1), where we introduce a number of machine-learning techniques such as boosting, neural networks, and support vector machines. Next, we study face recognition (Section 14.2), which is one of the more widely known applications of recognition. This topic serves as an introduction to subspace (PCA) models and Bayesian approaches to recognition and classification. We then present techniques for instance recognition (Section 14.3), building upon earlier topics in this book, such as feature detection, matching, and geometric alignment (Section 14.3.1). We introduce topics from the information and document retrieval communities, such as frequency vectors, feature quantization, and inverted indices (Section 14.3.2). We also present applications of location recognition (Section 14.3.3).

In the second half of the chapter, we address the most challenging variant of recognition, namely the problem of category recognition (Section 14.4). This includes approaches that use bags of features (Section 14.4.1), parts (Section 14.4.2), and segmentation (Section 14.4.3). We show how such techniques can be used to automate photo editing tasks, such as 3D modeling, scene completion, and creating collages (Section 14.4.4). Next, we discuss the role that context can play in both individual object recognition and more holistic scene under-

[1] However, some recent research suggests that direct image matching may be feasible for large enough databases (Russell, Torralba, Liu *et al.* 2007; Malisiewicz and Efros 2008; Torralba, Freeman, and Fergus 2008).

standing (Section 14.5). We close this chapter with a discussion of databases and test sets for constructing and evaluating recognition systems (Section 14.6).

While there is no comprehensive reference on object recognition, an excellent set of notes can be found in the ICCV 2009 short course (Fei-Fei, Fergus, and Torralba 2009), Antonio Torralba's more comprehensive MIT course (Torralba 2008), and two recent collections of papers (Ponce, Hebert, Schmid *et al.* 2006; Dickinson, Leonardis, Schiele *et al.* 2007) and a survey on object categorization (Pinz 2005). An evaluation of some of the best performing recognition algorithms can be found on the PASCAL Visual Object Classes (VOC) Challenge Web site at http://pascallin.ecs.soton.ac.uk/challenges/VOC/.

14.1 Object detection

If we are given an image to analyze, such as the group portrait in Figure 14.2, we could try to apply a recognition algorithm to every possible sub-window in this image. Such algorithms are likely to be both slow and error-prone. Instead, it is more effective to construct special-purpose *detectors*, whose job it is to rapidly find likely regions where particular objects might occur.

We begin this section with face detectors, which are some of the more successful examples of recognition. For example, such algorithms are built into most of today's digital cameras to enhance auto-focus and into video conferencing systems to control pan-tilt heads. We then look at pedestrian detectors, as an example of more general methods for object detection. Such detectors can be used in automotive safety applications, e.g., detecting pedestrians and other cars from moving vehicles (Leibe, Cornelis, Cornelis *et al.* 2007).

14.1.1 Face detection

Before face recognition can be applied to a general image, the locations and sizes of any faces must first be found (Figures 14.1c and 14.2). In principle, we could apply a face recognition algorithm at every pixel and scale (Moghaddam and Pentland 1997) but such a process would be too slow in practice.

Over the years, a wide variety of fast face detection algorithms have been developed. Yang, Kriegman, and Ahuja (2002) provide a comprehensive survey of earlier work in this field; Yang's ICPR 2004 tutorial[2] and the Torralba (2007) short course provide more recent reviews.[3]

According to the taxonomy of Yang, Kriegman, and Ahuja (2002), face detection techniques can be classified as feature-based, template-based, or appearance-based. Feature-based techniques attempt to find the locations of distinctive image features such as the eyes, nose, and mouth, and then verify whether these features are in a plausible geometrical arrangement. These techniques include some of the early approaches to face recognition (Fischler and Elschlager 1973; Kanade 1977; Yuille 1991), as well as more recent approaches based on modular eigenspaces (Moghaddam and Pentland 1997), local filter jets (Leung, Burl, and Perona 1995; Penev and Atick 1996; Wiskott, Fellous, Krüger *et al.* 1997), support

[2] http://vision.ai.uiuc.edu/mhyang/face-detection-survey.html.

[3] An alternative approach to detecting faces is to look for regions of skin color in the image (Forsyth and Fleck 1999; Jones and Rehg 2001). See Exercise 2.8 for some additional discussion and references.

Figure 14.2 Face detection results produced by Rowley, Baluja, and Kanade (1998a) © 1998 IEEE. Can you find the one false positive (a box around a non-face) among the 57 true positive results?

vector machines (Heisele, Ho, Wu *et al.* 2003; Heisele, Serre, and Poggio 2007), and boosting (Schneiderman and Kanade 2004).

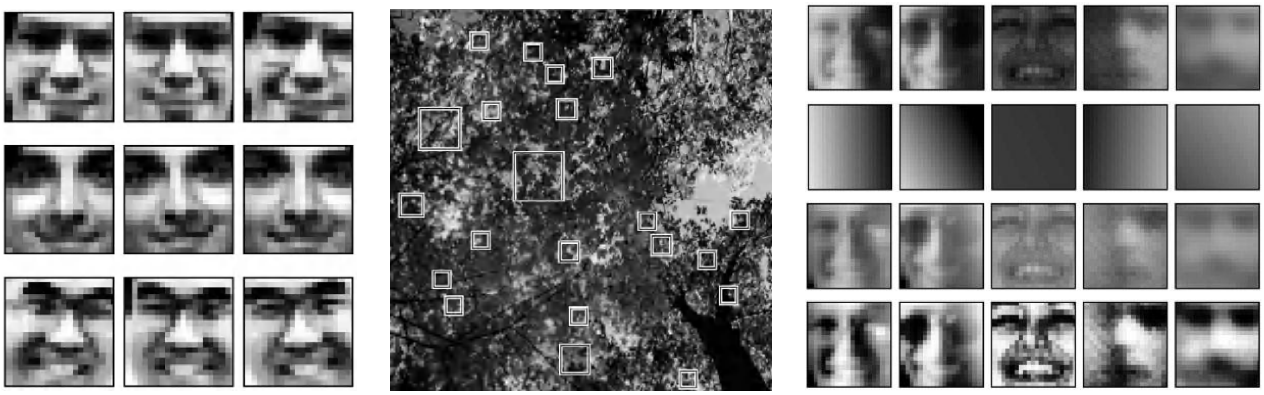
Template-based approaches, such as active appearance models (AAMs) (Section 14.2.2), can deal with a wide range of pose and expression variability. Typically, they require good initialization near a real face and are therefore not suitable as fast face detectors.

Appearance-based approaches scan over small overlapping rectangular patches of the image searching for likely face candidates, which can then be refined using a *cascade* of more expensive but selective detection algorithms (Sung and Poggio 1998; Rowley, Baluja, and Kanade 1998a; Romdhani, Torr, Schölkopf *et al.* 2001; Fleuret and Geman 2001; Viola and Jones 2004). In order to deal with scale variation, the image is usually converted into a sub-octave pyramid and a separate scan is performed on each level. Most appearance-based approaches today rely heavily on training classifiers using sets of labeled face and non-face patches.

Sung and Poggio (1998) and Rowley, Baluja, and Kanade (1998a) present two of the earliest appearance-based face detectors and introduce a number of innovations that are widely used in later work by others.

To start with, both systems collect a set of labeled face patches (Figure 14.2) as well as a set of patches taken from images that are known not to contain faces, such as aerial images or vegetation (Figure 14.3b). The collected face images are augmented by artificially mirroring, rotating, scaling, and translating the images by small amounts to make the face detectors less sensitive to such effects (Figure 14.3a).

After an initial set of training images has been collected, some optional pre-processing can be performed, such as subtracting an average gradient (linear function) from the image to compensate for global shading effects and using histogram equalization to compensate for



(a) (b) (c)

Figure 14.3 Pre-processing stages for face detector training (Rowley, Baluja, and Kanade 1998a) © 1998 IEEE: (a) artificially mirroring, rotating, scaling, and translating training images for greater variability; (b) using images without faces (looking up at a tree) to generate non-face examples; (c) pre-processing the patches by subtracting a best fit linear function (constant gradient) and histogram equalizing.

varying camera contrast (Figure 14.3c).

Clustering and PCA. Once the face and non-face patterns have been pre-processed, Sung and Poggio (1998) cluster each of these datasets into six separate clusters using k-means and then fit PCA subspaces to each of the resulting 12 clusters (Figure 14.4). At detection time, the DIFS and DFFS metrics first developed by Moghaddam and Pentland (1997) (see Figure 14.14 and (14.14)) are used to produce 24 Mahalanobis distance measurements (two per cluster). The resulting 24 measurements are input to a multi-layer perceptron (MLP), which is a neural network with alternating layers of weighted summations and sigmoidal non-linearities trained using the "backpropagation" algorithm (Rumelhart, Hinton, and Williams 1986).

Neural networks. Instead of first clustering the data and computing Mahalanobis distances to the cluster centers, Rowley, Baluja, and Kanade (1998a) apply a neural network (MLP) directly to the 20×20 pixel patches of gray-level intensities, using a variety of differently sized hand-crafted "receptive fields" to capture both large-scale and smaller scale structure (Figure 14.5). The resulting neural network directly outputs the likelihood of a face at the center of every overlapping patch in a multi-resolution pyramid. Since several overlapping patches (in both space and resolution) may fire near a face, an additional merging network is used to merge overlapping detections. The authors also experiment with training several networks and merging their outputs. Figure 14.2 shows a sample result from their face detector.

To make the detector run faster, a separate network operating on 30×30 patches is trained to detect both faces and faces shifted by ± 5 pixels. This network is evaluated at every 10th pixel in the image (horizontally and vertically) and the results of this "coarse" or "sloppy" detector are used to select regions on which to run the slower single-pixel overlap technique. To deal with in-plane rotations of faces, Rowley, Baluja, and Kanade (1998b) train a *router*