Chapter 15

Object Recognition

An object recognition system finds objects in the real world from an image of the world, using *object models* which are known a priori. This task is surprisingly difficult. Humans perform object recognition effortlessly and instantaneously. Algorithmic description of this task for implementation on machines has been very difficult. In this chapter we will discuss different steps in object recognition and introduce some techniques that have been used for object recognition in many applications. We will discuss the different types of recognition tasks that a vision system may need to perform. We will analyze the complexity of these tasks and present approaches useful in different phases of the recognition task.

The object recognition problem can be defined as a labeling problem based on models of known objects. Formally, given an image containing one or more objects of interest (and background) and a set of labels corresponding to a set of models *known* to the system, the system should assign correct labels to regions, or a set of regions, in the image. The object recognition problem is closely tied to the segmentation problem: without at least a partial recognition of objects, segmentation cannot be done, and without segmentation, object recognition is not possible.

In this chapter, we discuss basic aspects of object recognition. We present the architecture and main components of object recognition and discuss their role in object recognition systems of varying complexity.

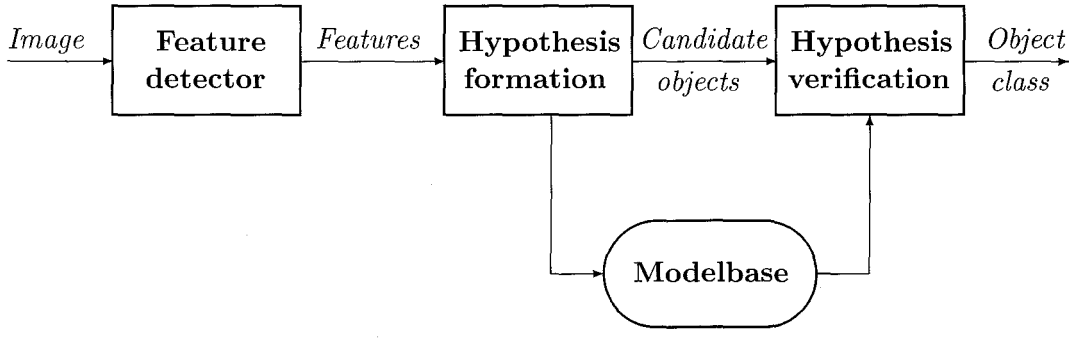


Figure 15.1: Different components of an object recognition system are shown.

15.1 System Component

An object recognition system must have the following components to perform the task:

- Model database (also called modelbase)
- Feature detector
- Hypothesizer
- Hypothesis verifier

A block diagram showing interactions and information flow among different components of the system is given in Figure 15.1.

The model database contains all the models known to the system. The information in the model database depends on the approach used for the recognition. It can vary from a qualitative or functional description to precise geometric surface information. In many cases, the models of objects are abstract feature vectors, as discussed later in this section. A feature is some attribute of the object that is considered important in describing and recognizing the object in relation to other objects. Size, color, and shape are some commonly used features.

The feature detector applies operators to images and identifies locations of features that help in forming object hypotheses. The features used by a

system depend on the types of objects to be recognized and the organization of the model database. Using the detected features in the image, the hypothesizer assigns likelihoods to objects present in the scene. This step is used to reduce the search space for the recognizer using certain features. The modelbase is organized using some type of indexing scheme to facilitate elimination of unlikely object candidates from possible consideration. The verifier then uses object models to verify the hypotheses and refines the likelihood of objects. The system then selects the object with the highest likelihood, based on all the evidence, as the correct object.

All object recognition systems use models either explicitly or implicitly and employ feature detectors based on these object models. The hypothesis formation and verification components vary in their importance in different approaches to object recognition. Some systems use only hypothesis formation and then select the object with highest likelihood as the correct object. Pattern classification approaches are a good example of this approach. Many artificial intelligence systems, on the other hand, rely little on the hypothesis formation and do more work in the verification phases. In fact, one of the classical approaches, template matching, bypasses the hypothesis formation stage entirely.

An object recognition system must select appropriate tools and techniques for the steps discussed above. Many factors must be considered in the selection of appropriate methods for a particular application. The central issues that should be considered in designing an object recognition system are:

- *Object or model representation:* How should objects be represented in the model database? What are the important attributes or features of objects that must be captured in these models? For some objects, geometric descriptions may be available and may also be efficient, while for another class one may have to rely on generic or functional features. The representation of an object should capture all relevant information without any redundancies and should organize this information in a form that allows easy access by different components of the object recognition system.

- *Feature extraction:* Which features should be detected, and how can they be detected reliably? Most features can be computed in two-dimensional images but they are related to three-dimensional characteristics of objects. Due to the nature of the image formation process,

some features are easy to compute reliably while others are very difficult. Feature detection issues were discussed in many chapters in this book.

- *Feature-model matching:* How can features in images be matched to models in the database? In most object recognition tasks, there are many features and numerous objects. An exhaustive matching approach will solve the recognition problem but may be too slow to be useful. Effectiveness of features and efficiency of a matching technique must be considered in developing a matching approach.

- *Hypotheses formation:* How can a set of likely objects based on the feature matching be selected, and how can probabilities be assigned to each possible object? The hypothesis formation step is basically a heuristic to reduce the size of the search space. This step uses knowledge of the application domain to assign some kind of probability or confidence measure to different objects in the domain. This measure reflects the likelihood of the presence of objects based on the detected features.

- *Object verification:* How can object models be used to select the most likely object from the set of probable objects in a given image? The presence of each likely object can be verified by using their models. One must examine each plausible hypothesis to verify the presence of the object or ignore it. If the models are geometric, it is easy to precisely verify objects using camera location and other scene parameters. In other cases, it may not be possible to verify a hypothesis.

Depending on the complexity of the problem, one or more modules in Figure 15.1 may become trivial. For example, pattern recognition-based object recognition systems do not use any feature-model matching or object verification; they directly assign probabilities to objects and select the object with the highest probability.

15.2 Complexity of Object Recognition

As we studied in earlier chapters in this book, images of scenes depend on illumination, camera parameters, and camera location. Since an object must

be recognized from images of a scene containing multiple entities, the complexity of object recognition depends on several factors. A qualitative way to consider the complexity of the object recognition task would consider the following factors:

- *Scene constancy:* The scene complexity will depend on whether the images are acquired in similar conditions (illumination, background, camera parameters, and viewpoint) as the models. As seen in earlier chapters, scene conditions affect images of the same object dramatically. Under different scene conditions, the performance of different feature detectors will be significantly different. The nature of the background, other objects, and illumination must be considered to determine what kind of features can be efficiently and reliably detected.

- *Image-models spaces:* In some applications, images may be obtained such that three-dimensional objects can be considered two-dimensional. The models in such cases can be represented using two-dimensional characteristics. If models are three-dimensional and perspective effects cannot be ignored, then the situation becomes more complex. In this case, the features are detected in two-dimensional image space, while the models of objects may be in three-dimensional space. Thus, the same three-dimensional feature may appear as a different feature in an image. This may also happen in dynamic images due to the motion of objects.

- *Number of objects in the model database:* If the number of objects is very small, one may not need the hypothesis formation stage. A sequential exhaustive matching may be acceptable. Hypothesis formation becomes important for a large number of objects. The amount of effort spent in selecting appropriate features for object recognition also increases rapidly with an increase in the number of objects.

- *Number of objects in an image and possibility of occlusion:* If there is only one object in an image, it may be completely visible. With an increase in the number of objects in the image, the probability of occlusion increases. Occlusion is a serious problem in many basic image

computations. Occlusion results in the absence of expected features and the generation of unexpected features. Occlusion should also be considered in the hypothesis verification stage. Generally, the difficulty in the recognition task increases with the number of objects in an image. Difficulties in image segmentation are due to the presence of multiple occluding objects in images.

The object recognition task is affected by several factors. We classify the object recognition problem into the following classes.

Two-dimensional

In many applications, images are acquired from a distance sufficient to consider the projection to be orthographic. If the objects are always in one stable position in the scene, then they can be considered two-dimensional. In these applications, one can use a two-dimensional modelbase. There are two possible cases:

- Objects will not be occluded, as in remote sensing and many industrial applications.
- Objects may be occluded by other objects of interest or be partially visible, as in the bin of parts problem.

In some cases, though the objects may be far away, they may appear in different positions resulting in multiple stable views. In such cases also, the problem may be considered inherently as two-dimensional object recognition.

Three-dimensional

If the images of objects can be obtained from arbitrary viewpoints, then an object may appear very different in its two views. For object recognition using three-dimensional models, the perspective effect and viewpoint of the image have to be considered. The fact that the models are three-dimensional and the images contain only two-dimensional information affects object recognition approaches. Again, the two factors to be considered are whether objects are separated from other objects or not.

For three-dimensional cases, one should consider the information used in the object recognition task. Two different cases are:

- *Intensity:* There is no surface information available explicitly in intensity images. Using intensity values, features corresponding to the three-dimensional structure of objects should be recognized.
- 2.5-*dimensional images:* In many applications, surface representations with viewer-centered coordinates are available, or can be computed, from images. This information can be used in object recognition. Range images are also 2.5-dimensional. These images give the distance to different points in an image from a particular view point.

Segmented

The images have been segmented to separate objects from the background. As discussed in Chapter 3 on segmentation, object recognition and segmentation problems are closely linked in most cases. In some applications, it is possible to segment out an object easily. In cases when the objects have not been segmented, the recognition problem is closely linked with the segmentation problem.

15.3 Object Representation

Images represent a scene from a camera's perspective. It appears natural to represent objects in a camera-centric, or viewer-centered, coordinate system. Another possibility is to represent objects in an object-centered coordinate system. Of course, one may represent objects in a world coordinate system also. Since it is easy to transform from one coordinate system to another using their relative positions, the central issue in selecting the proper coordinate system to represent objects is the ease of representation to allow the most efficient representation for feature detection and subsequent processes.

A representation allows certain operations to be efficient at the cost of other operations. Representations for object recognition are no exception. Designers must consider the parameters in their design problems to select

the best representation for the task. The following are commonly used representations in object recognition.

15.3.1 Observer-Centered Representations

If objects usually appear in a relatively few stable positions with respect to the camera, then they can be represented efficiently in an observer-centered coordinate system. If a camera is located at a fixed position and objects move such that they present only some aspects to the camera, then one can represent objects based on only those views. If the camera is far away from objects, as in remote sensing, then three-dimensionality of objects can be ignored. In such cases, the objects can be represented only by a limited set of views—in fact, only one view in most cases. Finally, if the objects in a domain of applications are significantly different from each other, then observer-centered representations may be enough.

Observer-centered representations are defined in image space. These representations capture characteristics and details of the images of objects in their relative camera positions.

One of the earliest and most rigorous approaches for object recognition is based on characterizing objects using a feature vector. This feature vector captures essential characteristics that help in distinguishing objects in a domain of application. The features selected in this approach are usually global features of the images of objects. These features are selected either based on the experience of a designer or by analyzing the efficacy of a feature in grouping together objects of the same class while discriminating it from the members of other classes. Many feature selection techniques have been developed in pattern classification. These techniques study the probabilistic distribution of features of known objects from different classes and use these distributions to determine whether a feature has sufficient discrimination power for classification.

In Figure 15.2 we show a two-dimensional version of a feature space. An object is represented as a point in this space. It is possible that different features have different importance and that their units are different. These problems are usually solved by assigning different weights to the features and by normalizing the features.

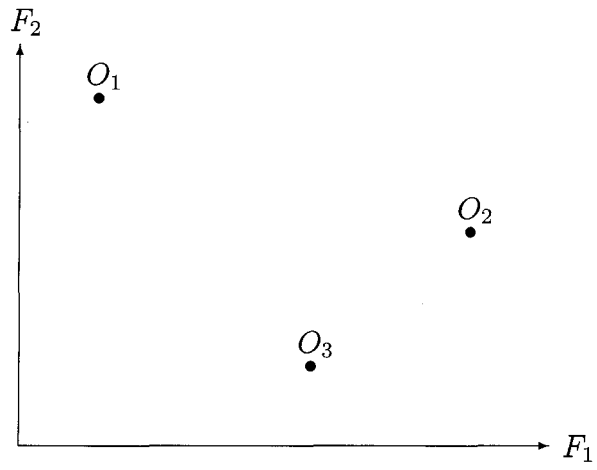


Figure 15.2: Two-dimensional feature space for object recognition. Each object in this space is a point. Features must be normalized to have uniform units so that one may define a distance measure for the feature space.

Most so-called approaches for two-dimensional object recognition in the literature are the approaches based on the image features of objects. These approaches try to partition an image into several local features and then represent an object as image features and relations among them. This representation of objects allows partial matching also. In the presence of occlusion in images, this representation is more powerful than feature space. In Figure 15.3 we show local features for an object and how they will be represented.

15.3.2 Object-Centered Representations

An object-centered representation uses description of objects in a coordinate system attached to objects. This description is usually based on three-dimensional features or description of objects.

Object-centered representations are independent of the camera parameters and location. Thus, to make them useful for object recognition, the representation should have enough information to produce object images or object features in images for a known camera and viewpoint. This requirement suggests that object-centered representations should capture aspects of the geometry of objects explicitly. Some commonly used object-centered representations are discussed here.

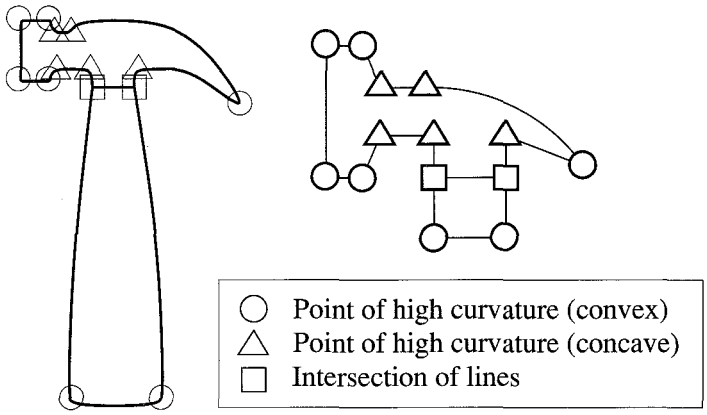


Figure 15.3: In (a) an object is shown with its prominent local features highlighted. A graph representation of the object is shown in (b). This representation is used for object recognition using a graph matching approach.

Constructive Solid Geometry

A CSG representation of an object uses simple volumetric primitives, such as blocks, cones, cylinders, and spheres, and a set of boolean operations: union, intersection, and difference. Since arbitrarily curved objects cannot be represented using just a few chosen primitives, CSG approaches are not very useful in object recognition. These representations are used in object representation in CAD/CAM applications. In Figure 15.4, a CSG representation for a simple object is shown.

Spatial Occupancy

An object in three-dimensional space may be represented by using nonoverlapping subregions of the three-dimensional space occupied by an object. In addition to simple occupancy, one may consider representing other properties of objects at points in space. There are many variants of this representation such as voxel representation, octree, and tetrahedral cell decomposition. In Figure 15.5, we show a voxel representation of an object.

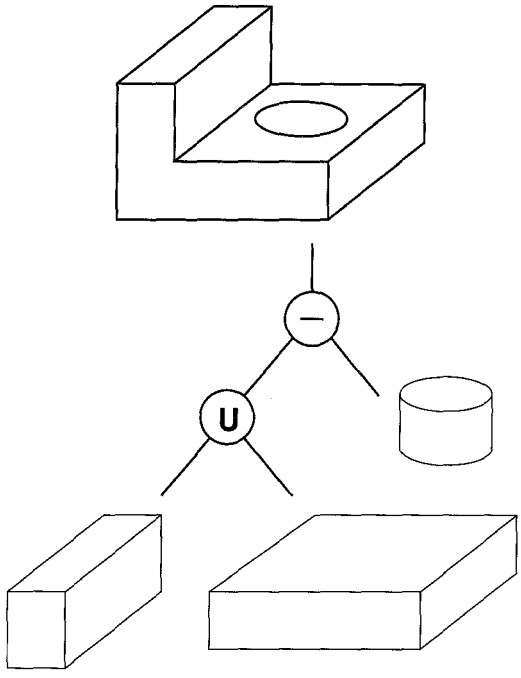


Figure 15.4: A CSG representation of an object uses some basic primitives and operations among them to represent an object. Here we show an object and its CSG representation.

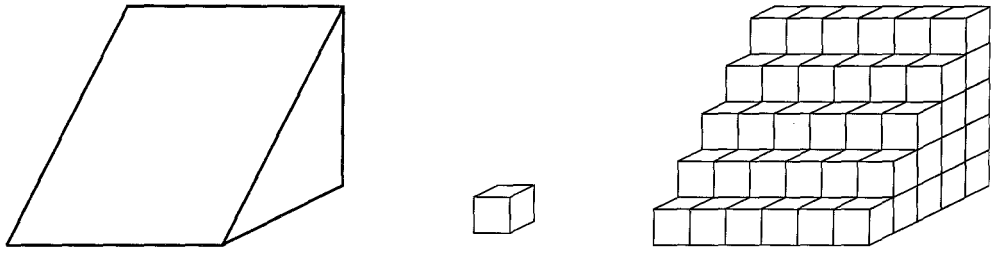


Figure 15.5: A voxel representation of an object.

A spatial occupancy representation contains a detailed description of an object, but it is a very low-level description. This type of representation must be processed to find specific features of objects to enable the hypothesis formation process.

Multiple-View Representation

Since objects must be recognized from images, one may represent a three-dimensional object using several views obtained either from regularly spaced viewpoints in space or from some strategically selected viewpoints. For a limited set of objects, one may consider arbitrarily many views of the object and then represent each view in an observer-centered representation.

A three-dimensional object can be represented using its aspect graph. An aspect graph represents all stable views of an object. Thus, an aspect graph is obtained by partitioning the view-space into areas in which the object has stable views. The aspect graph for an object represents a relationship among all the stable views. In Figure 15.6 we show a simple object and its aspect graph.

Surface-Boundary Representation

A solid object can be represented by defining the surfaces that bound the object. The bounding surfaces can be represented using one of several methods popular in computer graphics. These representations vary from triangular patches to nonuniform rational B-splines (NURBS). Some of these representations were discussed in Chapter 13.

Sweep Representations: Generalized Cylinders

Object shapes can be represented by a three-dimensional space curve that acts as the spine or axis of the cylinder, a two-dimensional cross-sectional figure, and a sweeping rule that defines how the cross section is to be swept along the space curve. The cross section can vary smoothly along the axis. This representation is shown in Figure 15.7.

For many industrial and other objects, the cross section of objects varies smoothly along an axis in space, and in such cases this representation is satisfactory. For arbitrarily shaped objects, this condition is usually not satisfied, making this representation unsuitable.

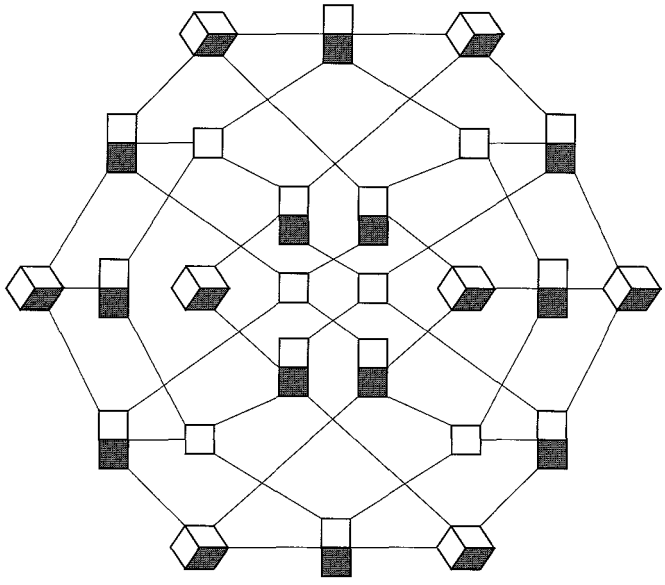


Figure 15.6: An object and its aspect graph. Each node in the aspect graph represents a stable view. The branches show how one can go from one stable view to other stable views through accidental views.

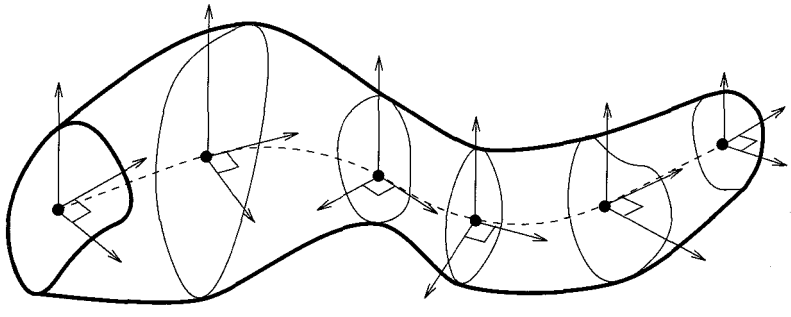


Figure 15.7: An object and its generalized cylinder representation. Note the axis of the cylinder is shown as a dashed line, the coordinate axes are drawn with respect to the cylinder's central axis, and the cross sections at each point are orthogonal to the cylinder's central axis.

15.4 Feature Detection

Many types of features are used for object recognition. Most features are based on either regions or boundaries in an image. It is assumed that a region or a closed boundary corresponds to an entity that is either an object or a part of an object. Some of the commonly used features are as follows.

Global Features

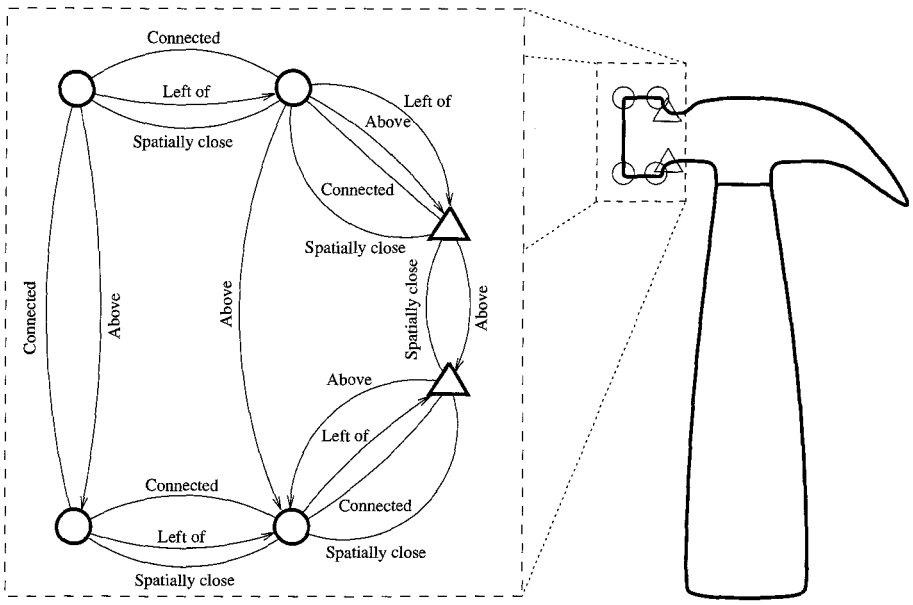
Global features usually are some characteristics of regions in images such as area (size), perimeter, Fourier descriptors, and moments. Global features can be obtained either for a region by considering all points within a region, or only for those points on the boundary of a region. In each case, the intent is to find descriptors that are obtained by considering all points, their locations, intensity characteristics, and spatial relations. These features were discussed at different places in the book.

Local Features

Local features are usually on the boundary of an object or represent a distinguishable small area of a region. Curvature and related properties are commonly used as local features. The curvature may be the curvature on a boundary or may be computed on a surface. The surface may be an intensity surface or a surface in 2.5-dimensional space. High curvature points are commonly called corners and play an important role in object recognition. Local features can contain a specific shape of a small boundary segment or a surface patch. Some commonly used local features are *curvature, boundary segments,* and *corners.*

Relational Features

Relational features are based on the relative positions of different entities, either regions, closed contours, or local features. These features usually include distance between features and relative orientation measurements. These features are very useful in defining composite objects using many regions or local features in images. In most cases, the relative position of entities is what defines objects. The exact same feature, in slightly different relationships, may represent entirely different objects.



Number of Regions	2
Area	42574 pixels
Perimeter	1343 pixels

Figure 15.8: An object and its partial representation using multiple local and global features.

In Figure 15.8, an object and its description using features are shown. Both local and global features can be used to describe an object. The relations among objects can be used to form composite features.

15.5 Recognition Strategies

Object recognition is the sequence of steps that must be performed after appropriate features have been detected. As discussed earlier, based on the detected features in an image, one must formulate hypotheses about possible objects in the image. These hypotheses must be verified using models of objects. Not all object recognition techniques require strong hypothesis formation and verification steps. Most recognition strategies have evolved to

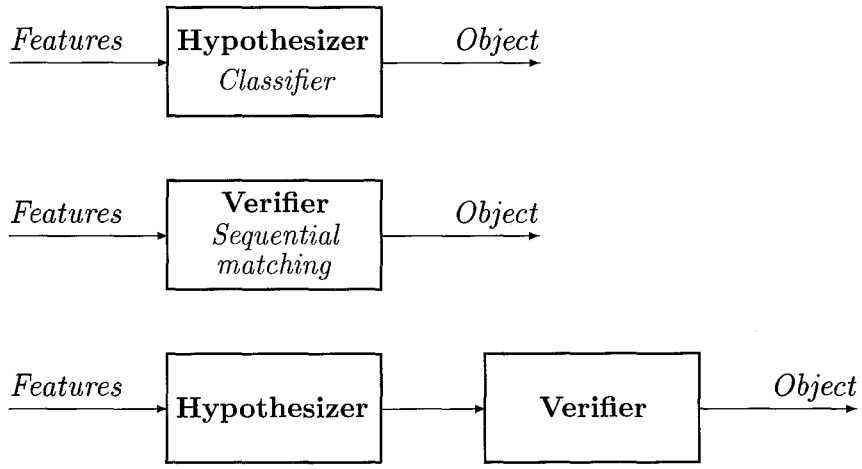


Figure 15.9: Depending on the complexity of the problem, a recognition strategy may need to use either or both the hypothesis formation and verification steps.

combine these two steps in varying amounts. As shown in Figure 15.9, one may use three different possible combinations of these two steps. Even in these, the application contest, characterized by the factors discussed earlier in this section, determines how one or both steps are implemented. In the following, we discuss a few basic recognition strategies used for recognizing objects in different situations.

15.5.1 Classification

The basic idea in classification is to recognize objects based on features. Pattern recognition approaches fall in this category, and their potential has been demonstrated in many applications. Neural net-based approaches also fall in this class. Some commonly used classification techniques are discussed briefly here. All techniques in this class assume that N features have been detected in images and that these features have been normalized so that they can be represented in the same metric space. We will briefly discuss techniques to normalize these features after classification. In the following discussion, it will be assumed that the features for an object can be represented as a point in the *N-dimensional* feature space defined for that particular object recognition task.

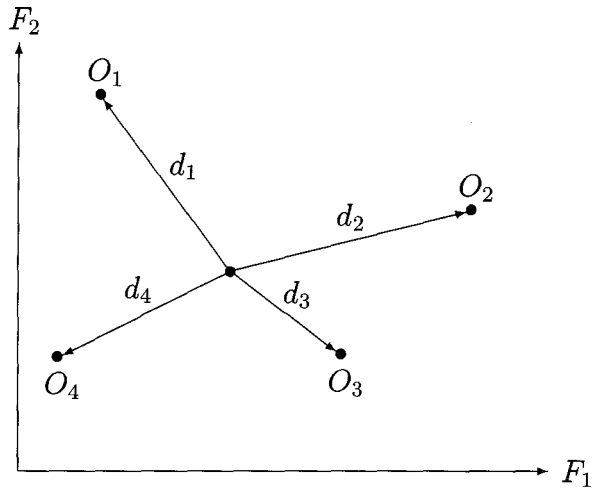


Figure 15.10: The prototypes of each class are represented as points in the feature space. An unknown object is assigned to the closest class by using a distance measure in this space.

Nearest Neighbor Classifiers

Suppose that a model object (ideal feature values) for each class is known and is represented for class i as f_{ij}, $j = 1, \ldots, N$, $i = 1, \ldots, M$ where M is the number of object classes. Now suppose that we detect and measure features of the unknown object U and represent them as u_j, $j = 1, \ldots, N$. For a 2-dimensional feature space, this situation is shown in Figure 15.10. To decide the class of the object, we measure its similarity with each class by computing its distance from the points representing each class in the feature space and assign it to the nearest class. The distance may be either Euclidean or any weighted combination of features. In general, we compute the distance d_j of the unknown object from class j as given by

$$d_i = \left[\sum_{j=1}^{N} (u_j - f_{ij})^2 \right]^{1/2}, \tag{15.1}$$

then the object is assigned to the class R such that

$$d_R = \min_{i=1}^{M} [d_i]. \tag{15.2}$$

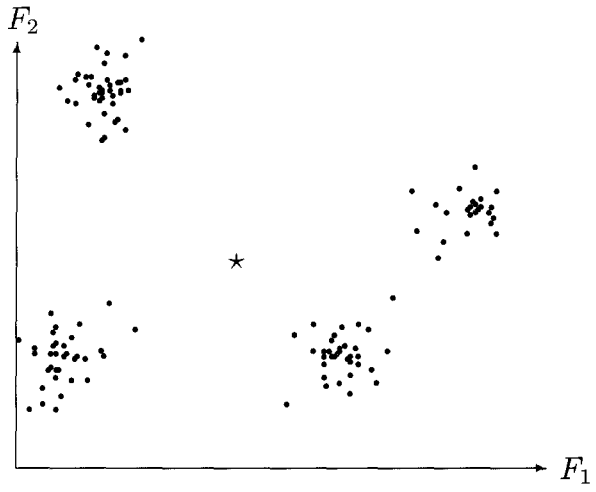


Figure 15.11: All known objects of each class are represented as points in the feature space. Each class is thus represented by a cluster of points in the feature space. Either the centroid of the cluster representing the class or the closest point of each class is considered the prototype for classification.

In the above, the distance to a class was computed by considering distance to the feature point representing a prototype object. In practice, it may be difficult to find a prototype object. Many objects may be known to belong to a class. In this case, one must consider feature values for all known objects of a class. This situation is shown in Figure 15.11. Two common approaches in such a situation are

1. Consider the centroid of the cluster as the prototype object's feature point, and compute the distance to this.

2. Consider the distance to the closest point of each class.

Bayesian Classifier

A Bayesian approach has been used for recognizing objects when the distribution of objects is not as straightforward as shown in the cases above. In general, there is a significant overlap in feature values of different objects.

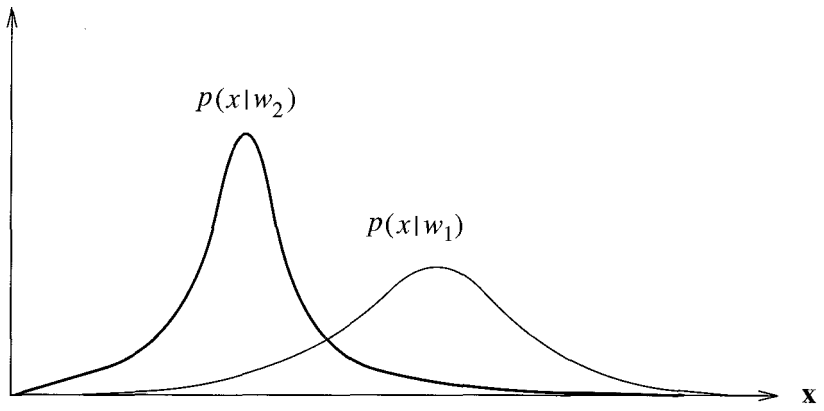


Figure 15.12: The conditional density function for $p(x|w_j)$. This shows the probability of the feature values for each class.

Thus, as shown for the one-dimensional feature space in Figure 15.12, several objects can have same feature value. For an observation in the feature space, multiple-object classes are equally good candidates. To make a decision in such a case, one may use a Bayesian approach to decision making.

In the Bayesian approach, probabilistic knowledge about the features for objects and the frequency of the objects is used. Suppose that we know that the probability of objects of class j is $P(w_j)$. This means that a priori we know that the probability that an object of class j will appear is $P(w_j)$, and hence in absence of any other knowledge we can minimize the probability of error by assigning the unknown object to the class for which $P(w_j)$ is maximum.

Decisions about the class of an object are usually made based on feature observations. Suppose that the probability $p(x|w_j)$ is given and is as shown in Figure 15.12. The conditional probability $p(x|w_j)$ tells us that based on the probabilistic information provided, we know that if the feature value is observed to be x, then the probability that the object belongs to class j is $p(x|w_j)$. Based on this knowledge, we can compute the a posteriori probability $p(w_j|x)$ for the object. The a posteriori probability is the probability that, for the given information and observations, the unknown object belongs to class j. Using Bayes' rule, this probability is given as:

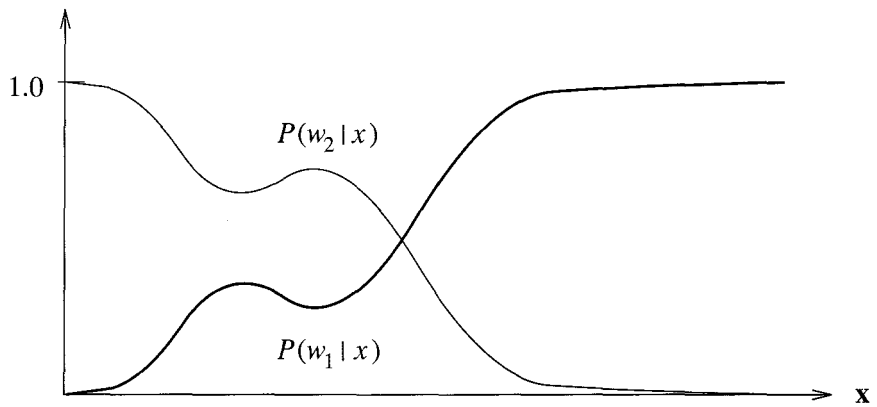


Figure 15.13: A posteriori probabilities for two different values of a priori probabilities for objects.

$$P(w_j|x) = \frac{p(x|w_j)P(w_j)}{p(x)} \tag{15.3}$$

where

$$p(x) = \sum_{j=1}^{N} p(x|w_j)P(w_j). \tag{15.4}$$

The unknown object should be assigned to the class with the highest a posteriori probability $P(w_j|x)$. As can be seen from the above equations, and as shown in Figure 15.13, a posteriori probability depends on prior knowledge about the objects. If a priori probability of the object changes, so will the result.

We discussed the Bayesian approach above for one feature. It can be easily extended to multiple features by considering conditional density functions for multiple features.

Off-Line Computations

The above classification approaches consider the feature space, and then, based on the knowledge of the feature characteristics of objects, a method is used to partition the feature space so that a class decision is assigned to each point in the feature space. To assign a class to each point in the feature space, all computations are done before the recognition of unknown objects begins.

This is called *off-line computation.* These off-line computations reduce the computations at the run time. The recognition process can be effectively converted to a *look-up table* and hence can be implemented very quickly.

Neural Nets

Neural nets have been proposed for object recognition tasks. Neural nets implement a classification approach. Their attraction lies in their ability to partition the feature space using nonlinear boundaries for classes. These boundaries are obtained by using training of the net. During the training phase, many instances of objects to be recognized are shown. If the training set is carefully selected to represent all objects encountered later during the recognition phase, then the net may learn the classification boundaries in its feature space. During the recognition phase, the net works like any other classifier.

The most attractive feature of neural nets is their ability to use nonlinear classification boundaries and learning abilities. The most serious limitations have been the inability to introduce known facts about the application domain and difficulty in debugging their performance.

15.5.2 Matching

Classification approaches use effective features and knowledge of the application. In many applications, a priori knowledge about the feature probabilities and the class probabilities is not available or not enough data is available to design a classifier. In such cases one may use direct matching of the model to the unknown object and select the best-matching model to classify the object. These approaches consider each model in sequence and fit the model to image data to determine the similarity of the model to the image component. This is usually done after the segmentation has been done. In the following we discuss basic matching approaches.

Feature Matching

Suppose that each object class is represented by its features. As above, let us assume that the jth feature's value for the ith class is denoted by f_{ij}. For an unknown object the features are denoted by u_j. The similarity of the object

with the ith class is given by

$$S_i = \sum_{j=1}^{N} w_j s_j \tag{15.5}$$

where w_j is the weight for the jth feature. The weight is selected based on the relative importance of the feature. The similarity value of the jth feature is s_j. This could be the absolute difference, normalized difference, or any other distance measure. The most common method is to use

$$s_j = |u_j - f_{ij}| \tag{15.6}$$

and to account for normalization in the weight used with the feature.

The object is labeled as belonging to class k if S_k is the highest similarity value. Note that in this approach, we use features that may be local or global. We do not use any relations among the features.

Symbolic Matching

An object could be represented not only by its features but also by the relations among features. The relations among features may be spatial or some other type. An object in such cases may be represented as a graph. As shown in Figure 15.8, each node of the graph represents a feature, and arcs connecting nodes represent relations among the objects. The object recognition problem then is considered as a graph matching problem.

A graph matching problem can be defined as follows. Given two graphs G_1 and G_2 containing nodes N_{ij}, where i and j denote the graph number and the node number, respectively, the relations among nodes j and k is represented by R_{ijk}. Define a similarity measure for the graphs that considers the similarities of all nodes and functions.

In most applications of machine vision, objects to be recognized may be partially visible. A recognition system must recognize objects from their partial views. Recognition techniques that use global features and must have all features present are not suitable in these applications. In a way, the partial-view object recognition problem is similar to the graph embedding problem studied in graph theory. The problem in object recognition becomes different when we start considering the *similarity* of nodes and relations among them.

We discuss this type of matching in more detail later, in the section on verification.

15.5.3 Feature Indexing

If the number of objects is very large and the problem cannot be solved using feature space partitioning, then indexing techniques become attractive. The symbolic matching approach discussed above is a sequential approach and requires that the unknown object be compared with all objects. This sequential nature of the approach makes it unsuitable with a number of objects. In such a case, one should be able to use a hypothesizer that reduces the search space significantly. The next step is to compare the models of each object in the reduced set with the image to recognize the object.

Feature indexing approaches use features of objects to structure the modelbase. When a feature from the indexing set is detected in an image, this feature is used to reduce the search space. More than one feature from the indexing set may be detected and used to reduce the search space and in turn reduce the total time spent on object recognition.

The features in the indexing set must be determined using the knowledge of the modelbase. If such knowledge is not available, a learning scheme should be used. This scheme will analyze the frequency of each feature from the feature set and, based on the frequency of features, form the indexing set, which will be used for structuring the database.

In the indexed database, in addition to the names of the objects and their models, information about the orientation and pose of the object in which the indexing feature appears should always be kept. This information helps in the verification stage.

Once the candidate object set has been formed, the verification phase should be used for selecting the best object candidate.

15.6 Verification

Suppose that we are given an image of an object and we need to find how many times and where this object appears in an image. Such a problem is essentially a verification, rather than an object recognition, problem. Obviously a verification algorithm can be used to exhaustively verify the presence of each model from a large modelbase, but such an exhaustive approach will not be a very effective method. A verification approach is desirable if one, or at most a few, objects are possible candidates. There are many approaches for verification. Here we discuss some commonly used approaches.

15.6.1 Template Matching

Suppose that we have a template $g[i,j]$ and we wish to detect its instances in an image $f[i,j]$. An obvious thing to do is to place the template at a location in an image and to detect its presence at that point by comparing intensity values in the template with the corresponding values in the image. Since it is rare that intensity values will match exactly, we require a measure of dissimilarity between the intensity values of the template and the corresponding values of the image. Several measures may be defined:

$$\max_{[i,j] \in R} |f - g| \tag{15.7}$$

$$\sum_{[i,j] \in R} |f - g| \tag{15.8}$$

$$\sum_{[i,j] \in R} (f - g)^2 \tag{15.9}$$

where R is the region of the template.

The sum of the squared errors is the most popular measure. In the case of template matching, this measure can be computed indirectly and computational cost can be reduced. We can simplify:

$$\sum_{[i,j] \in R} (f - g)^2 = \sum_{[i,j] \in R} f^2 + \sum_{[i,j] \in R} g^2 - 2 \sum_{[i,j] \in R} fg. \tag{15.10}$$

Now if we assume that f and g are fixed, then $\sum fg$ gives a measure of mismatch. A reasonable strategy for obtaining all locations and instances of the template is to shift the template and use the match measure at every point in the image. Thus, for an $m \times n$ template, we compute

$$M[i,j] = \sum_{k=1}^{m} \sum_{l=1}^{n} g[k,l]\, f[i+k, j+l] \tag{15.11}$$

where k and l are the displacements with respect to the template in the image.[1]

Our aim will be to find the locations that are local maxima and are above a certain threshold value. However, a minor problem in the above computation

[1]This operation is called the *cross-correlation* between f and g.

was introduced when we assumed that f and g are constant. When applying this computation to images, the template g is constant, but the value of f will be varying. The value of M will then depend on f and hence will not give a correct indication of the match at different locations. This problem can be solved by using normalized cross-correlation. The match measure M then can be computed using

$$C_{fg}[i,j] = \sum_{k=1}^{m}\sum_{l=1}^{n} g[k,l]\, f[i+k, j+l] \tag{15.12}$$

$$M[i,j] = \frac{C_{fg}[i,j]}{\{\sum_{k=1}^{m}\sum_{l=1}^{n} f^2[i+k, j+l]\}^{1/2}}. \tag{15.13}$$

It can be shown that M takes maximum value for $[i,j]$ at which $g = cf$. In Figure 15.14, we show an image, a template, and the result of the above computation. Notice that at the location of the template, we get local maxima.

The above computations can be simplified significantly in binary images. Template matching approaches have been quite popular in optical computing: frequency domain characteristics of convolution are used to simplify the computation.

A major limitation of template matching is that it only works for translation of the template. In case of rotation or size changes, it is ineffective. It also fails in case of only partial views of objects.

15.6.2 Morphological Approach

Morphological approaches can also be used to detect the presence and location of templates. For binary images, using the structuring element as the template and then *opening* the image will result in all locations where the template fits in. For gray images, one may use gray-image morphology. These results are shown for a template in Figure 15.15.

15.6.3 Symbolic

As discussed above, if both models of objects and the unknown object are represented as graphs, then some approach must be used for matching graphical representations. Here we define the basic concepts behind these approaches.

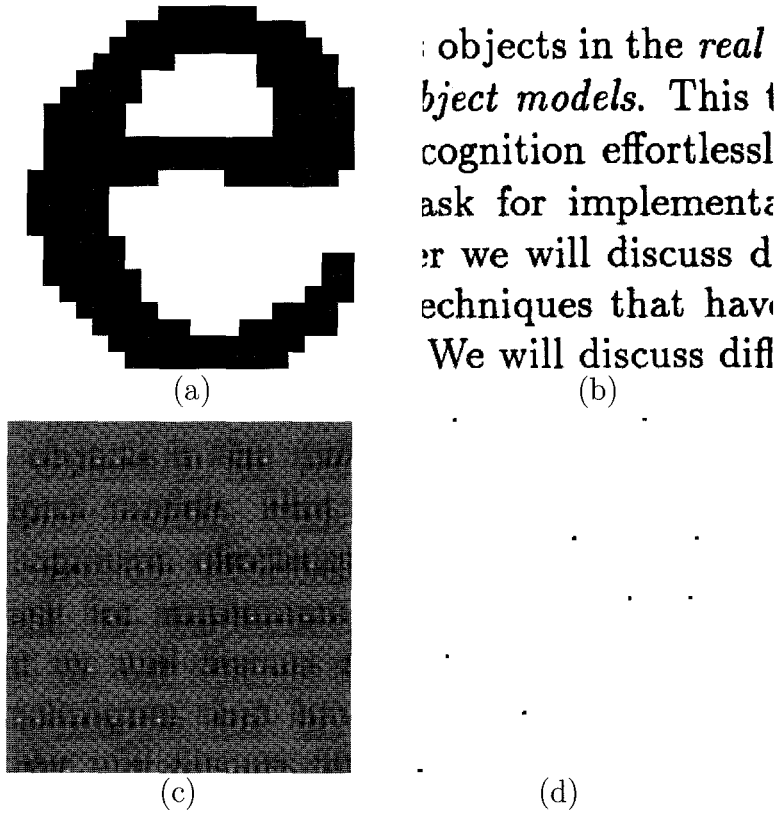
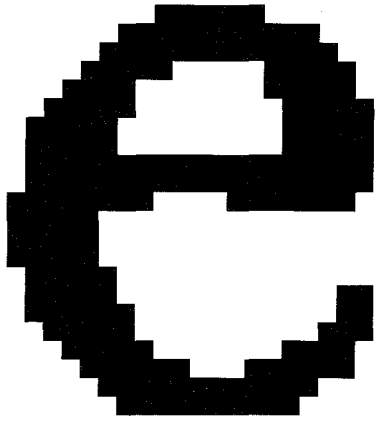


Figure 15.14: A template (a), an image (b), the result of the template matching computations discussed above (c), and the thresholded result to find the match locations (d), $T = 240$.

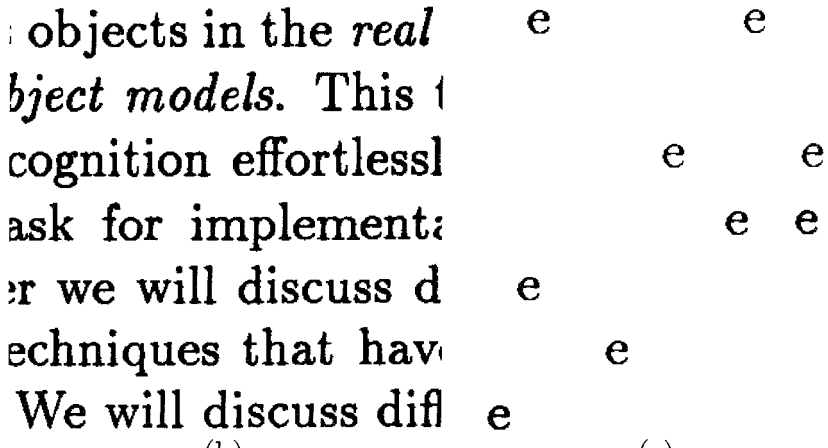
Graph Isomorphism

Given two graphs (V_1, E_1) and (V_2, E_2), find a 1:1 and onto mapping (an isomorphism) f between V_1 and V_2 such that for $\theta_1, \theta_2 \in V_1, V_2$, $f(\theta_1) = \theta_2$ and for each edge of E_1 connecting any pair of nodes θ_1 and $\theta_1' \in V_1$, there is an edge of E_2 connecting $f(\theta_1)$ and $f(\theta_1')$.

Graph isomorphism can be used only in cases of completely visible objects. If an object is partially visible, or a 2.5-dimensional description is to be matched with a 3-dimensional description, then graph embedding, or subgraph isomorphisms, can be used.



(a)



(b) (c)

Figure 15.15: A structuring element (a), an image (b), and the result of the morphological opening (c).

Subgraph Isomorphisms

Find isomorphisms between a graph (V_1, E_1) and subgraphs of another graph (V_2, E_2).

A problem with these approaches for matching is that the graph isomorphism is an NP problem. For any reasonable object description, the time required for matching will be prohibitive. Fortunately, we can use more information than that used by graph isomorphism algorithms. This informa-

tion is available in terms of the properties of nodes. Many heuristics have been proposed to solve the graph matching problem. These heuristics should consider:

- Variability in properties and relations
- Absence of properties or relations
- The fact that a model is *an abstraction* of a class of objects
- The fact that instances may contain extra information.

One way to formulate the similarity is to consider the arcs in the graph as springs connecting two masses at the nodes. The quality of the match is then a function of the goodness of fit of the templates locally and the amount of energy needed to stretch the springs to force the unknown onto the modelence data.

$$\begin{aligned} C = & \sum_{d \in R_1} \text{template cost}(d, F(d)) \\ & + \sum_{(d,e) \in R_2} \text{spring cost}(F(d), F(e)) \\ & + \sum_{c \in R_3} \text{missing cost}(c) \end{aligned} \tag{15.14}$$

where $R_1 = \{\text{found in model}\}$, $R_2 = \{\text{found in model } x \text{ found in unknown}\}$, and $R_3 = \{\text{missing in model}\} \cup \{\text{missing in unknown}\}$. This function represents a very general formulation. *Template cost*, *spring cost*, and *missing cost* can take many different forms. Applications will determine the exact form of these functions.

15.6.4 Analogical Methods

A measure of similarity between two curves can be obtained by comparing them on the same frame of reference, as shown in Figure 15.16, and directly measuring the difference between them at every point. Notice that in Figure 15.16 the difference is measured at every point along the x axis. The difference will always be measured along some axis. The total difference is either

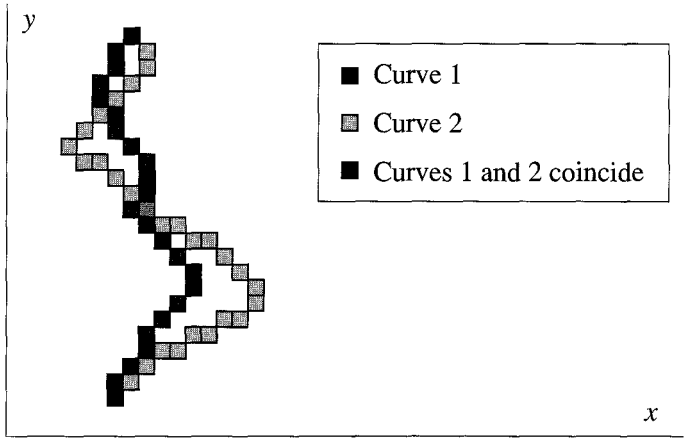


Figure 15.16: Matching of two entities by directly measuring the errors between them.

the sum of absolute errors or the sum of squared errors. If exact registration is not given, some variation of correlation-based methods must be used.

For recognizing objects using three-dimensional models, one may use rendering techniques from computer graphics to find their appearence in an image and then try to compare with the original image to verify the presence of an object. Since the parameters required to render objects are usually unknown, usually one tries to consider some prominent features on three-dimensional models and to detect them and match them to verify the model's instance in an image. This has resulted in development of theories that try to study three-dimensional surface characteristics of objects and their projections to determine *invariants* that can be used in object recognition. Invariants are usually features or characteristics in images that are relatively insensitive to an object's orientation and scene illumination. Such features are very useful in detecting three-dimensional objects from their two-dimensional projections.

Further Reading

Object recognition has been one of the most important topics in machine vision. In one form or another, it has attracted significant attention. Many approaches have been developed for pattern classification. These approaches

are very useful in many applications of machine vision. For an excellent introduction to pattern classification, see [70]. Some very good survey papers on object recognition are by Chin and Dyer [57], Binford [34], and Besl and Jain [27].

Many object recognition systems are built upon low-level vision modules which operate upon images to derive depth measurements. These measurements are often incomplete and unreliable and thus adversely affect the performance of higher-level recognition modules. In contrast to this approach, Lowe describes a system in which bottom-up description of images is designed to generate viewpoint-invariant groupings of image features [158]. Brooks' ACRONYM system is a domain-independent model-based interpretation system which uses generalized cylinders for the description of model and scene objects [49, 50]. Some later work along this line was performed under the SUCCESSOR project and is given in [33].

Most object recognition research has considered a small set of objects. If a very large number of objects are to be recognized, the recognition task will be dominated by hypothesis and test approaches. The hypothesis phase will require organization of models indexed by features so that, based on observed features, a small set of likely objects can be selected. Later these selected models may be used to recognize objects by verifying which object from this set is present in the given image. Such approaches are given in Knoll and Jain [143], Ettinger [75], Grimson [93], Lamdan and Wolfson [151].

In many industrial applications, detailed geometric models of objects are available. These models can be used for generating recognition strategies, including feature selection, for three-dimensional objects. CAD-based object recognition is being studied at several places now [34, 99, 32, 221, 178]. An important step in the recognition of three-dimensional objects is to consider their possible two-dimensional projections to determine effective features and recognition strategy. Classification of infinite two-dimensional projection views of objects into topologically equivalent classes, called aspect graphs, was introduced by Koenderink and Van Doorn [144, 145]. Their application to recognition is described by Chakravarty and Freeman [56]. Gigus and Malik [88] developed an algorithm for generating aspect graphs. Recently algorithms have been designed for computing aspect graphs for curved objects also [73, 149, 226].

Ikeuchi and Kanade [120] describe a novel system in which the object and sensor models are automatically compiled into a visual recognition strategy.

The system extracts from the models those features that are useful for recognition and determines the control sequence that must be applied to handle different object appearances. An alternative to this kind of approach is presented by neural network approaches for object recognition. Object recognition is one of the most researched areas in neural networks. Most research in neural networks, however, has addressed only limited two-dimensional objects.

Exercises

15.1 List the major components of an object recognition system. Discuss their role in the recognition task.

15.2 Stereotyping is a phenomenon often criticized in society. Object recognition tasks, however, are dependent on stereotyping. Explain how stereotyping plays an important role in object recognition, particularly its role in relating modelbase and set of features.

15.3 What factors would you consider in selecting an appropriate representation for the modelbase? Discuss the advantages and disadvantages of object-centered and observer-centered representations.

15.4 What is an aspect graph? Develop a generalized aspect graph that is based on image features and their relationships for an object. Where can you use such an aspect graph?

15.5 What is feature space? How can you recognize objects using feature space?

15.6 Compare classical pattern recognition approaches based on Bayesian approaches with neural net approaches by considering the feature space, classification approaches, and object models used by both of these approaches.

15.7 One of the most attractive features of neural nets is their ability to learn. How is their ability to learn used in object recognition? What kind of model is prepared by a neural net? How can you introduce your knowledge about objects in neural nets?

15.8 Where do you use matching in object recognition? What is a symbolic matching approach?

15.9 What is feature indexing? How does it improve object recognition?

15.10 Discuss template matching. In which type of applications would you use template matching? What are the major limitations of template matching? How can you overcome these limitations?

15.11 Sketch the aspect graph of a four-faced trihedral polyhedron with triangular faces.

15.12 A template g is matched with an image f, both shown below, using the normalized cross-correlation method. Find:

a. The cross-correlation C_{fg}.

b. $\sum\sum f^2$.

c. The normalized cross-correlation $M[i, j]$.

$f =$

0	0	0	0	0	0	0	0
0	2	4	2	0	0	0	0
0	0	2	0	0	0	0	0
0	0	2	0	0	0	2	0
0	0	0	0	0	0	2	0
1	2	1	0	0	2	4	2
0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0

$g =$

1	2	1
0	1	0
0	1	0

Computer Projects

15.1 Implement an object recognition system to recognize objects from their partial views. The objects in an image are from a given set of about 10 objects that are commonly found in an office scene. Select only objects that are more or less two-dimensional (coins, keys, sticky pads, business cards, etc.). Consider the camera to be mounted about 8 feet

above the desk. Test your system by considering many random images in which these objects appear in different ways.

15.2 Continuing the above example, now consider that the objects are three-dimensional (mouse, stapler, etc.), and redesign and reimplement a prototype object recognition system. This system should recognize three-dimensional objects from their partial views.

15.3 Now assume that you have a large number of objects in your modelbase. Redesign your system to perform the object recognition task efficiently for a large number of objects.