Interdisciplinary Information Sciences Vol. 17, No. 2 (2011) 49–129
©Graduate School of Information Sciences, Tohoku University
ISSN 1340-9050 print/1347-6157 online
DOI 10.4036/iis.2011.49

Introduction to Pattern Recognition and Computer Vision

Koichiro DEGUCHI*

Graduate School of Information Sciences, Tohoku University, Sendai 980-8579, Japan

Received May 24, 2010; final version accepted February 8, 2011

These lecture notes are composed for a half-year course lecture to introduce the pattern recognition and computer vision for under-graduate or graduate students. They consist of three parts: Pattern Recognition, 3D Object Recognition, and Image Matching, all of which also have three chapters.

These notes focus on the orthodox and theoretical approaches to the image processing, the computer vision and the robot vision problems, which students and young researchers must learn and follow as an introductory course for these fields. The topics selected here are main subjects of the transdisciplinary applications of computer science, mathematics, optics, cognitive science, artificial intelligence, and other academic fields. Those topics do not cover total and advanced fields of pattern recognition and computer vision, but include specially interesting fruits of modern researches in the fields.

This work is intended by the author and the editors of IIS Journal to contribute to introduce new comers to the fascinating research areas of the pattern recognition and the computer vision.

KEYWORDS: pattern recognition, computer vision, object recognition, geometry of image, image matching, projective invariants, matching by graph-theory

Contents

* Corresponding author. E-mail: kodeg@fractal.is.tohoku.ac.jp

Part I. Pattern Recognition

1. Introduction to Pattern Recognition

1.1 What is a "Pattern"?

In these lecture notes we treat problems of computer recognition of images. We begin first with the strategy to recognize two-dimensional figure shapes, such as hand-written or printed characters.

The recognition of characters and figures is called the *Pattern Recognition*.

A *pattern* is defined as a combination of (1) a set of *features* which describe, respectively, partial or whole shapes of the pattern and (2) their *spatial* distributions. Here, the space includes one-dimension, two-dimension, three-dimension and higher ones in general. In one-dimensional space, a voice which is a series of phonemes and a music which is a series of sounds in time-sequence are typical examples. Characters shown in Fig. 1.1 are typical examples in two-dimensional space. In those patterns, "lines," "arcs," "end points," "T-junctions," "rounds" and "holes" are the features. Their distributions, that is, their relative placements on a plane, define a character pattern. Other than those patterns, gray-level images, color images, and geological data are also two-dimensional patterns.

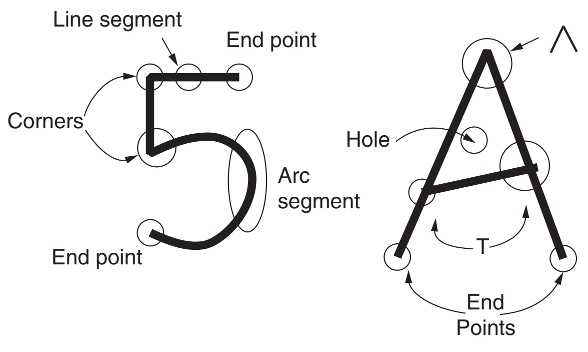


Fig. 1.1. Character patterns. The *spatial distribution* of *features* defines the *pattern*.

We may consider three or higher dimensional patterns, in which some features distribute in the three or higher dimensional space. Sometimes, we consider a higher dimensional pattern by adding the time dimension for a temporal change of the pattern.

Therefore, the pattern recognition is to extract both informations of (1) "kinds of features" and (2) "types of distribution structures." That is, the pattern recognition consists of two steps: the first step is the feature extraction and the second step is the structure analysis of the feature allocations. Through these two steps, the pattern recognition determines which given class the pattern belongs to.

This class, in general, has ambiguity to some extent. The members of the class are assumed to have their own respective variations. For example, a hand-written character allows to have infinite variety of its form even it is defined as some character. This is also the case for printed characters, because we have many "fonts" for a character, and it may have noise and smear.

Pattern recognition must cope with these conditions as a common problem to read out the characters correctly.

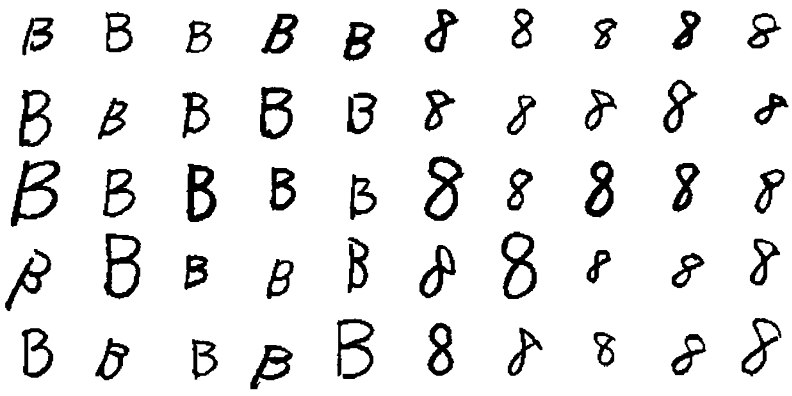


Fig. 1.2. Examples of hand-written character patterns of "B" and "8."

Figure 1.2 shows examples of hand-written character patterns of "B" and "8." All of them are either "B" or "8," and have a wide variety of forms. This means that, to read them out, we must ignore small and local differences among the members of each character class and, on the other hand, capture the global features commonly found within each respective class.

Considering these points, the most important issue is to identify which features are local and which features are global for the pattern recognition. For these examples, a global feature of "having two holes at top and bottom" is common to all characters in both of these two classes and is not a useful feature for the recognition of the character class.

In the following sections, the strategies and the methods to select the proper features for the pattern recognition are examined.

1.2 Strategy and Method of the Pattern Recognition

The methods of the pattern recognition are characterized with next two strategic aspects. One is the strategy to combine the feature extraction and the structural analysis and to build up the total process. The other is how to treat the above mentioned infinite variations of the pattern. Of course, these two are mutually dependent.

Table 1.1 shows three typical methods and their strategies. The descriptions of the features of the pattern and their distribution are the basic frames of the methods, and the treatment of the infinite variations of patterns is the third one.

Table 1.1. Methods of pattern recognition.

Approach	Feature description	Structure description	Treatment of variations
Statistical method	Feature vectors	Feature space	Statistical treatment
Structural method	Symbols	Formal grammar	Inherent freedom
Graph matching	Nodes and	Semantic graph	Partial matching

Given a pattern, firstly, the feature extraction is carried out. This is done in image processing. Typically for a character pattern, lines and curves are extracted from an image, and their end points, corners, and crossings are extracted and localized. The column of the "Feature extraction" in Table 1.1 shows the description of the features extracted from an image.

The next problem is how to describe the localizations of the extracted features and their relative distributions. This problem includes the treatments of the above mentioned infinite variations.

Therefore, the essential key to the pattern recognition methods is the description of the structures of the spatial distribution of the features. In this lecture, we first present typical conventional method of the "statistical pattern recognition" and, then, "structural pattern recognition" in the following chapters.

We also introduce a modern method employing the graph theory in the final chapter of these lecture notes.

2. Statitical Pattern Recognition

2.1 Features and Their Distributions

In the statistical pattern recognition, the features of a pattern are expressed as numerical values, which are called *feature values*. One pattern may have several feature values, which are expressed with a vector (*feature vector*) as an array of the feature values. The variation of a pattern is considered to be a statistical perturbation of the feature vector from a typical pattern.

Let us go back to the examples of the recognition of hand-written characters of "B" and "8" in Fig. 1.2. The problem is to discriminate patterns of "B" from "8." A feature which tells "the pattern of "B" consists of two rings at top and bottom" is of no use for the recognition because it is common to both of pattern classes. The major difference between these two patterns other than this feature is found as,

(I) the left-most side edge of "B" looks linear, while that of "8" looks curved [1],
which is shown in Fig. 2.1.

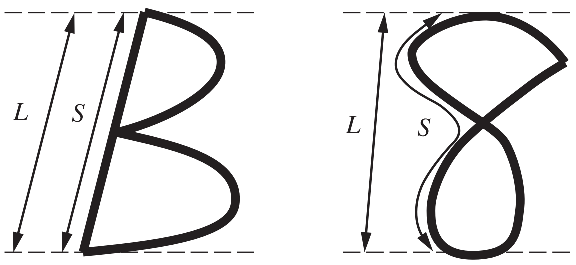


Fig. 2.1. Lengths to compose a feature value to discriminate "B" from "8."

It should be noted that the left edge of "B" is not always strictly a straight line. That is, to utilize this feature to discriminate the patterns, we need to (1) express the "straightness" as a numerical value, and (2) give a threshold boundary for this value to determine whether the edge is linear or curved.

As shown in Fig. 2.1, let the height of a given character be denoted by L, and the total length along the left side edge from the top to the bottom by S. Then, we define the *straightness* as

$$x = \frac{L}{S} \tag{2.1}$$

We can consider

$$\begin{aligned} x \approx 1 &\longrightarrow \text{linear} \\ x \approx 0 &\longrightarrow \text{curved} \end{aligned} \tag{2.2}$$

Now, the problem is: What value is the proper boundary between them?

We obtained 88 hand-written samples for both of the characters "B" and "8," and examined the distributions of the value of x for both of them. They are shown in Fig. 2.2.

Let us denote the probability density of x with respect to the character "B" with $p(x|B)$. (Here, the probability density $p(x)$ is defined as the value which satisfies that the probability of the value of x drops between x and $x + dx$ will be $p(x)dx$.) This means, under the condition of "B," the probability of x to be between x and $x + dx$ is given as $p(x|B)dx$.

Similarly, $p(x|8)$ is defined for "8." Figure 2.2 can be considered as the graphs of $p(x|B)$ and $p(x|8)$ obtained empirically. We will not discuss further whether the number of samples of 88 is sufficient to obtain their correct graphs or not here [2].

The probability for the case where the value of x is below X is called the *empirical cumulative distribution*. That is, for example of the character "B," it is defined as

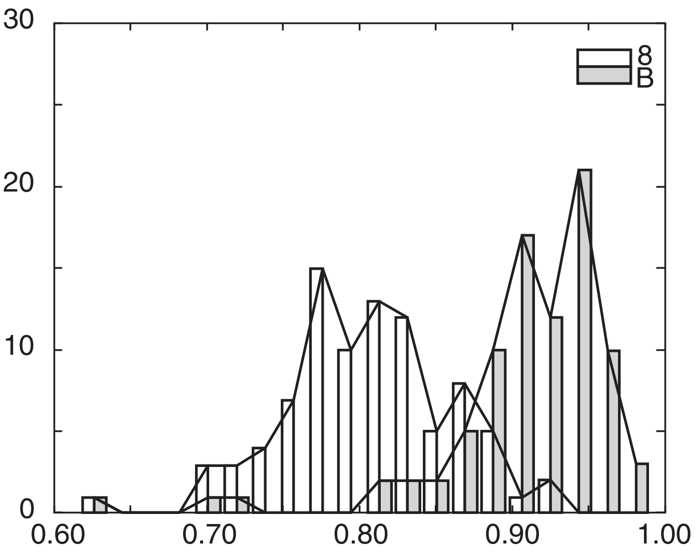


Fig. 2.2. Distributions of x for 88 hand-written samples for both of the characters "B" and "8."

$$F(X|B) = \int_{-\infty}^{X} p(x|B)dx \tag{2.3}$$

Also $F(X|8)$ is similarly defined. Figure 2.3 shows $F(X|B)$ and $F(X|8)$ for $p(x|B)$ and $p(x|8)$ in Fig. 2.2, respectively.
The empirical cumulative distribution is a monotonically increasing function from 0 to 1 according to the increase of X.

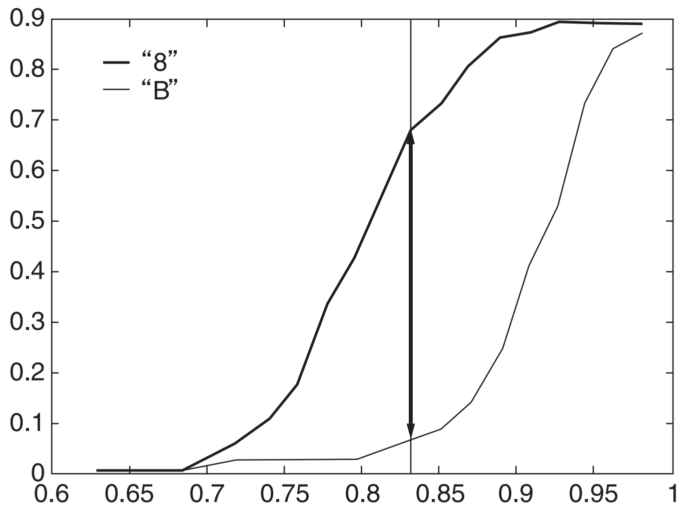


Fig. 2.3. Empirical cumulative distributions of the straightness x of "B" and "8." The arrow at the center indicates the value at which the difference between both distributions becomes maximum.

2.2 Decision Based on the Distribution of Feature Values

Let us consider a decision of "B" or "8" for a given character pattern by reading out the value of the straightness x. We model $p(x|B)$ and $p(x|8)$ with simpler forms as shown in Fig. 2.4 to make the problem be easily understood. If the regions of x where $p(x|B)$ and $p(x|8)$ have non-zero values are not overlapping, that is, those domains are completely separated, we can select the boundary value $x = \tau$ at an in-between point of both regions. If a read-out of x becomes smaller than τ, the given character pattern is decided as "8," and if larger, it is "B."

But, in our case, the samples have an overlapping region as shown in the figure. This means that some values of x are assigned to both of "B" and "8."

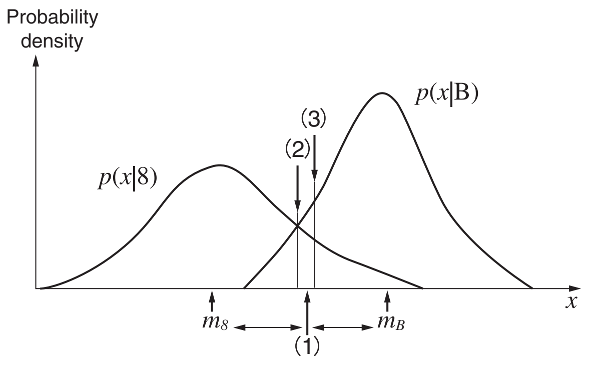


Fig. 2.4. Decision process based on the distribution of feature values.

For such cases, we have some strategies what value of x should be the boundary of τ, as listed in the following.

1. Denote the mean values of x with respect to "B" and "8" with m_8 and m_B, respectively. Then, τ is given as

$$\tau = \frac{1}{2}(m_B + m_8) \tag{2.4}$$

   Because both mean values can be considered to correspond to typical patterns of respective characters, the boundary should be selected as the furthest value from these two typical values. This decision is agreeable but too intuitive. It has no background of the statistical theory.
2. The boundary τ should be selected as satisfying $p(\tau|8) = p(\tau|B)$.
   In general, in the region right of this τ, it holds that $p(x|8) < p(x|B)$, and in the left region, $p(x|8) > p(x|B)$. As the result, this is the decision according that which of $p(x|8)$ and $p(x|B)$ is larger at x.
3. τ should satisfy $F(\tau|8) + F(\tau|B) = 1$.
   $1 - F(\tau|8)$ is the total portion of the case where a given "B" pattern is decided to be "8," and $F(\tau|B)$ is that of the reverse case. Therefore, the τ which makes $1 - F(\tau|8) = F(\tau|B)$, that is, $F(\tau|8) + F(\tau|B) = 1$, gives the boundary which makes probabilities of false decisions of "B" and "8" be equal.
4. The boundary τ should make $F(\tau|8) - F(\tau|B) \rightarrow$ Max.
   Also consider the total portions of false decisions. The sum of $1 - F(\tau|8) + F(\tau|B)$ is the total sum of false decisions. So that, this τ makes the error decision be the smallest.

The positions of the boundary τ for these cases of 1, 2, and 3 are shown at (1), (2), and (3), respectively, in Fig. 2.4. The boundary τ for the case 4 was already shown in Fig. 2.3 with the cumulative distributions of the sample character patterns.

As shown in this figure, we should note that, if we intend to reduce the decision error for "8," it in turn increases the decision error for "B," and vice versa. The cases from (1) to (4) stand for respective policies how to compromise between these two decision errors. In the case (4), where the total decision error becomes minimum, the decision error for "B" was 7% and that for "8" was 32%, and the total error was 39% as shown in Fig. 2.3. This is the minimum total error for the sample character patterns, which have the distributions of Fig. 2.2. These distributions tell us that it would be hard to reduce both decision errors simultaneously under 20%.

2.3 Feature Vectors and Feature Space

We have described some strategies of the statistical pattern recognition approach employing a feature value x and its boundary for the decision. As shown in the previous section, when $p(x|B)$ and $p(x|8)$ have mutual overlapping, basically we could not improve the performance further by the selection of the value τ.

Here, we intend to add new features for the recognition.

Observing sample character patterns of "B" and "8," we find another difference between these two classes than the previous (I). That is,

(II) The width of the top ring is wider than that of the bottom ring for "8," and inversely the bottom is wider than the top for "B."

Then, we define the ratio of t and b shown in Fig. 2.5 as

$$x_2 = \frac{b}{t} \tag{2.5}$$

and call it *top-to-bottom ratio*.

We employ it as the combination with the above straightness x_1 and define a *feature vector* $\boldsymbol{x} = (x_1, x_2)$.

For every sample pattern, we put a point in xy-plane with x_1 as the x-coordinate and x_2 as y-coordinate. Such a space whose coordinates are given with respective feature values is called a *feature space* (in the case we are considering here, the space is a plane).

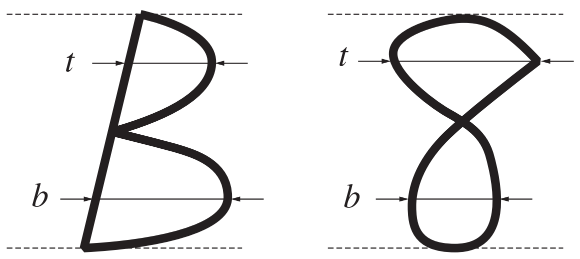


Fig. 2.5. The second feature for the recognition of "B" and "8."

For each sample of above 88 character patterns of "B" and "8," we measured feature values of the vector of $\boldsymbol{x} = (x_1, x_2)$ and obtained the plots in the feature space as shown in Fig. 2.6. Points of "B" are plotted with symbols of "+," and "8" with "◇" in the plane.

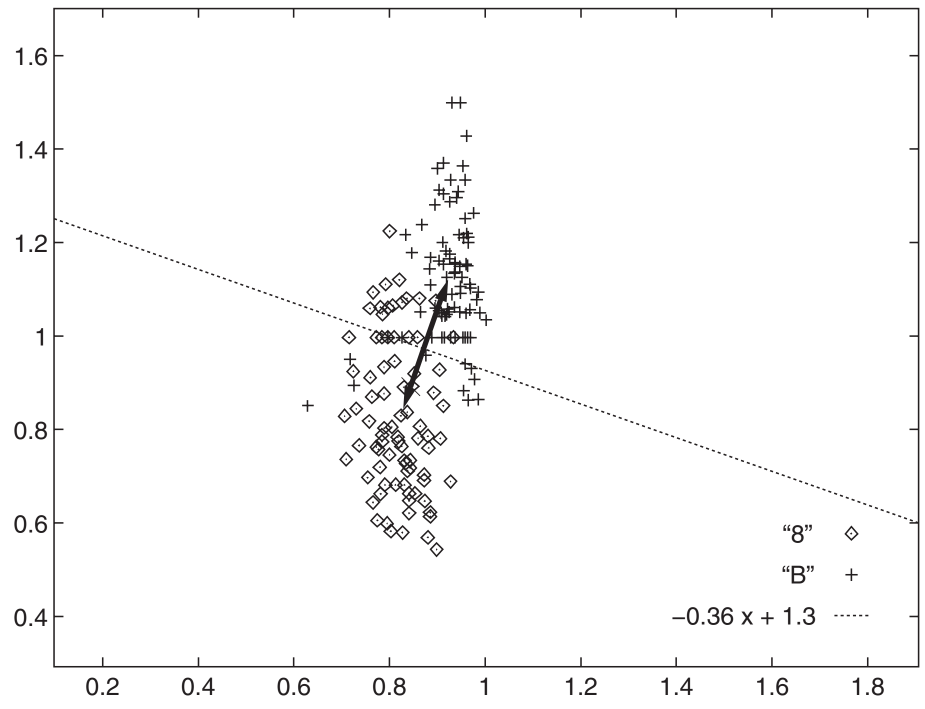


Fig. 2.6. Distribution of the feature vectors for hand-written sample characters of "B" and "8" (88 samples for both respective characters) in the feature space. The mean vectors of each respective characters and the perpendicular bisector between these two mean vectors are overlaid.

Now, the problem to decide a pattern to be "B" or "8" becomes to put a proper boundary surface (in our case here, it will be a boundary curve on the plane). This surface is called *decision boundary*.

An example of generally acceptable selection of the decision boundary is given as follows.

The mean vectors of each distributions of "B" and "8" in the feature space can be considered to correspond to the most typical patterns for each respective characters. This means that we can decide the class of pattern by observing which typical mean vector the given pattern is closer to.

Let us denote those mean vectors with $\boldsymbol{m}_B$ and $\boldsymbol{m}_8$, respectively. In Fig. 2.6, those vectors were given as $\boldsymbol{m}_8 = (0.816, 0.827)$ and $\boldsymbol{m}_B = (0.921, 1.121)$.

When we are given a new sample pattern, we extract $\boldsymbol{x}$ from the pattern, and examine

$$\|\boldsymbol{x} - \boldsymbol{m}_8\| \lesseqgtr \|\boldsymbol{x} - \boldsymbol{m}_B\| \tag{2.6}$$

This means that we select the boundary line as the perpendicular bisector between $\boldsymbol{m}_8$ and $\boldsymbol{m}_B$. This boundary is shown in Fig. 2.6. The ratios of correct and false decisions using this boundary line are listed in the Table 2.1.

The addition of the new features improved the decision performances. But, we may have more efficient decision boundary other than the perpendicular bisector. In the next section we consider the selection of the decision boundary more deeply by employing models of probability densities of features.

Table 2.1. Results of the decision using the perpendicular bisector as the decision boundary.

	Correct	Error
8	80.68%	19.32%
B	89.77%	10.23%

2.4 Models of the Feature Values' Probability Density

2.4.1 Independent identical normal distributions for feature values

We consider to model the probability density of the occurrences of feature vectors $\boldsymbol{x}$ for the character patterns of "B" and "8." The most common and general modeling is to assume the normal distributions.

If the occurrences of the feature values of x_1 and x_2 are independent, we have

$$\begin{aligned} p(\boldsymbol{x}|B) &= p(x_1, x_2|B) &= p(x_1|B)p(x_2|B) \\ p(\boldsymbol{x}|8) &= p(x_1, x_2|8) &= p(x_1|8)p(x_2|8) \end{aligned} \tag{2.7}$$

Moreover, if we can assume that both of the probability densities $p(x_1|B)$ and $p(x_2|B)$ for "B" are of the normal distributions with the respective means of m_i^B $(i = 1, 2)$ and the common variance σ^2, that is,

$$p(x_i|B) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x_i - m_i^B)^2}{2\sigma^2}\right] \quad (i = 1, 2)$$

(We have similar assumptions for "8.") Then,

$$p(\boldsymbol{x}|B) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{1}{2\sigma^2} \|\boldsymbol{x} - \boldsymbol{m}_B\|^2\right) \tag{2.8}$$

(We have also a similar probability density for "8.")

Now, if we make decision with respect to $p(\boldsymbol{x}|8) \lessgtr p(\boldsymbol{x}|B)$, from (2.8), this is sufficient to evaluate $\|\boldsymbol{x} - \boldsymbol{m}_8\| \lessgtr \|\boldsymbol{x} - \boldsymbol{m}_B\|$. This is, in turn, just the minimum distance based decision of (2.6) to cach of the mean vectors.

That is to say, the minimum distance based decision substantially assumes that the occurrences of every feature values is independent and has the normal distributions with an identical variance and different means. For the example patterns of hand-written characters of "B" and "8," this assumption is nothing but to consider that the variances of the distributions of the straightness x_1 and the top-to-bottom ratio x_2 are same. It is un-natural for these examples.

2.4.2 Individual variances for distributions of feature values

Next, we assume that the variances are different for each features but common for different characters. That is, variances of x_1^B and x_1^8 are same. This is the case for x_2^B and x_2^8. We denote these variances with σ_1^2 and σ_2^2, respectively. Also we assume that the occurrences of every feature values are independent.

Then, we have a covariance matrix of $\boldsymbol{x}$ as

$$\Sigma = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}$$

and using this matrix, the probability density is expressed as

$$p(\boldsymbol{x}|B) = \frac{1}{2\pi|\Sigma|^2} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{m}_B)\Sigma^{-1}(\boldsymbol{x} - \boldsymbol{m}_B)^\top\right) \tag{2.9}$$

(We have similar form for "8.")

Now, to make a decision according to $p(\boldsymbol{x}|8) \lessgtr p(\boldsymbol{x}|B)$ is, from (2.9), to have a boundary which satisfies

$$\begin{aligned} &-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{m}_8)\Sigma^{-1}(\boldsymbol{x} - \boldsymbol{m}_8)^\top \\ &= -\frac{1}{2}(\boldsymbol{x} - \boldsymbol{m}_B)\Sigma^{-1}(\boldsymbol{x} - \boldsymbol{m}_B)^\top \end{aligned} \tag{2.10}$$

that is a line having the equation

$$\boldsymbol{x}\Sigma^{-1}(\boldsymbol{m}_B - \boldsymbol{m}_8)^\top = \frac{1}{2}(\mathbf{m}_B + \boldsymbol{m}_8)\Sigma^{-1}(\boldsymbol{m}_B - \boldsymbol{m}_8)^\top \tag{2.11}$$

This is just the straight line passing through a point of $\frac{1}{2}(\boldsymbol{m}_B + \boldsymbol{m}_8)$ and being normal to the vector of $\Sigma^{-1}(\boldsymbol{m}_B - \boldsymbol{m}_8)$ in the feature space.

Here, we assumed that the variances of the straightness and the top-to-bottom ratio were different but they were common for "B" and "8." The boundary for our example patterns is shown in Fig. 2.7. We had decision errors shown in Table 2.2, as the results, by employing this boundary.

Table 2.2. Decision errors with individual variances for distributions of feature values.

	Correct	Error
8	92.05%	7.95%
B	94.32%	5.68%

2.4.3 General normal distributions

We consider a more general case, where $\boldsymbol{x}$ of "B" has a normal distribution with the covariance matrix of Σ_B and the mean of $\boldsymbol{m}_B$, and $\boldsymbol{x}$ of "8" with the covariance matrix of Σ_8 and the mean of $\boldsymbol{m}_8$, respectively.

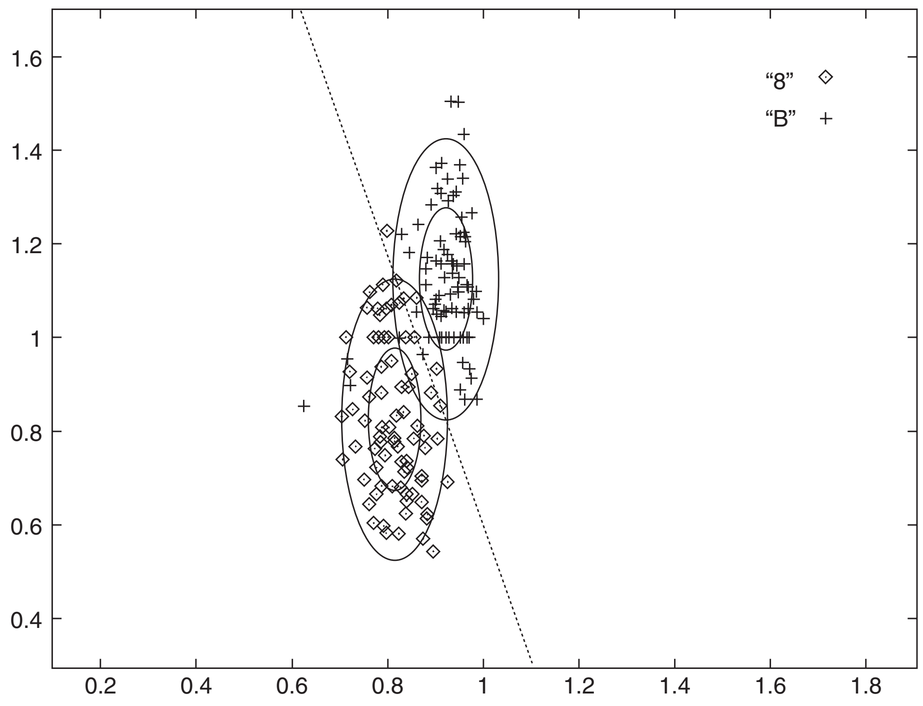


Fig. 2.7. The boundary for example patterns of "B" and "8" assuming individual variances for distributions of feature values.

For this case the decision according to $p(\boldsymbol{x}|8) \lesseqgtr p(\boldsymbol{x}|B)$ has the boundary on which these probabilities are equal, that is,

$$(\boldsymbol{x} - \boldsymbol{m}_8)^\top \Sigma_8^{-1}(\boldsymbol{x} - \boldsymbol{m}_8) - \frac{1}{2}\log|\Sigma_8|$$
$$= (\boldsymbol{x} - \boldsymbol{m}_B)^\top \Sigma_B^{-1}(\boldsymbol{x} - \boldsymbol{m}_B) - \frac{1}{2}\log|\Sigma_B| \qquad (2.12)$$

This is categorized into the so-called conic curve, which is either of straight line pair, ellipse, parabola, or hyperbola, in the feature space (it is a 2-dimensional plane for our example).

In this equation, $\|(\boldsymbol{x} - \boldsymbol{m}_B)^\top \Sigma_B^{-1}(\boldsymbol{x} - \mathbf{m}_B)\|^{1/2}$ is the distance between $\boldsymbol{x}$ and its mean $\boldsymbol{m}_B$ corrected with the covariance Σ_B. This is a distance from the mean point and is measured with respect to the probability. That is, the feature points having the same probability are considered to be at the same distance from its mean. This is called the *Mahalanobis distance*.

The decisions described until here are summarized in Fig. 2.8. Their probabilities of the occurrence of $\boldsymbol{x}$ were modeled according to their assumptions, respectively, in the plane of (x_1, x_2) as shown in the figure, and the shown respective decision boundaries were obtained.

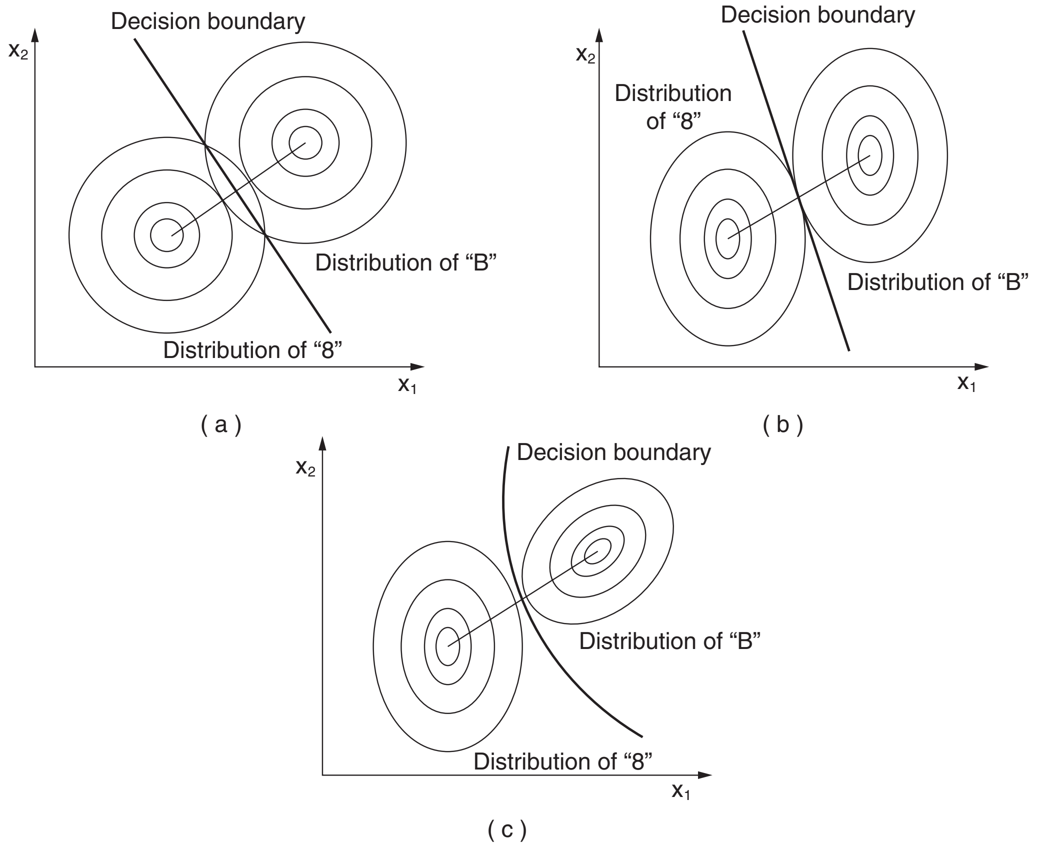


Fig. 2.8. The decision boundaries for the assumptions of (a) independent and identical normal distribution, (b) individual variances for features, and (c) general normal distribution.

2.5 Likelihood and the Bayes' Law

In the previous chapter, the probabilities of occurrences of features such as $p(x_1|B)$, $p(x_1|8)$, $p(x_2|B)$, and $p(x_1|8)$ were introduced, and recognition based on these probabilities was considered.

They express the distributions of, for example, the straightness x_1 for sets of given samples of hand-written characters, "B" and "8." These distributions were actually obtained by measuring x_1 by gathering a large number of hand-written character of "B" and "8."

This means that the probability density is given with the condition where characters of "B" or "8" are actually given. It is a *conditional* probability density. Therefore, the value of $p(x_1|B)$ expresses how likely the value of x_1 will be when the given character is "B." Also, $p(x_1|8)$ expresses how likely x_1 will be when the character is "8." So, the direct comparison between $p(x_1|8)$ and $p(x_1|B)$ would not make sense, because their preconditions are different.

We are firstly given a value of x_1, and the character will be either "B" or "8." Then we should discriminate these two characters from their probabilities. Basically, the sum of the probabilities of all the occurrences must be 1. But,

$$p(x_1|8) + p(x_1|B) \tag{2.13}$$

will not necessarily be 1.

Given x_i, the value of $p(x_i|8)$ is particularly called the *likelihood* for "8," and $p(x_i|B)$ the likelihood for "B."

Now, what value must be compared to make the decision of choosing the character to be "B" or "8" when we are given a feature value? We must evaluate the probabilities of "B" and "8" under the condition that a value of x_1 is given, that is to evaluate

$$p(8|x_1) \lesseqgtr p(B|x_1) \tag{2.14}$$

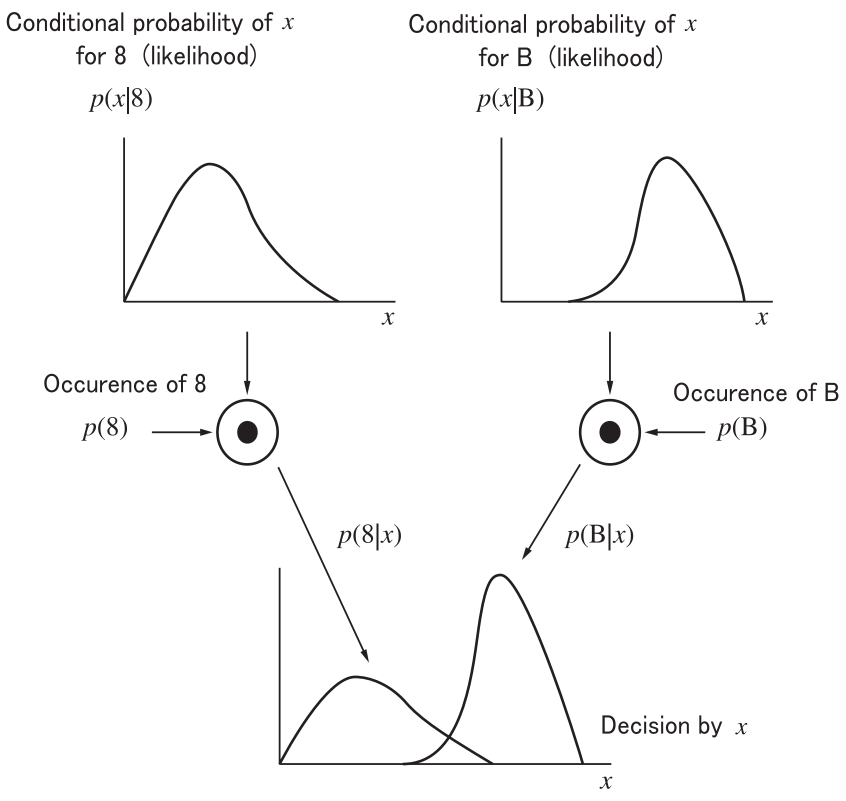


Fig. 2.9. Probability of the occurrence of x and its likelihood.

Actually, we have a relation between these two types of conditional probability, which is called the *Bayes' law*, as

$$\begin{aligned} p(8|x_1) &= \frac{p(x_1|8)p(8)}{p(x_1)} \\ p(B|x_1) &= \frac{p(x_1|B)p(B)}{p(x_1)} \end{aligned} \tag{2.15}$$

Therefore, we must correct the comparisons in the previous section to the comparison of (2.14), that is,

$$p(x_1|8)p(8) \lesseqgtr p(x_1|B)p(B) \tag{2.16}$$

where $p(x_1)$ in both equations of (2.15) is common and can be omitted. This suggests that we must basically consider the probabilities of the occurrences of the characters of "B" and "8" themselves. This process is outlined in Fig. 2.9. If the character "8" appears very seldom, for example, the value of the conditional probability of $p(x_1|8)$ has no practical meaning for the determination.

2.6 Feature Space, Principal Component Analysis and Eigenspace

In the previous sections, we first demonstrated that one feature value is not substantially sufficient to discriminate patterns when it has overlapping regions for two or more classes. Generally, any feature value has such regions for character patterns. This means that discrimination of character patterns by using a single feature value can achieve a limited capability. Then, we employed one more feature value, and improved the performance. It is because the overlappings were not common for all features.

This may suggest that, if we employ more features, we obtain better performance. The additional new feature must add new information about the differences between the classes. We selected such a second feature for the patterns of "B" and "8." However, the improvement will not be so easy generally, when we must discriminate between large number of classes, even though we select many types of feature values. Basically, if the performance depends highly on the skillful selection of the feature values, such a discrimination is not said to be a good method.

Therefore, a good feature value must detect a specific kind of feature of patterns independently from other feature values. This may reduce the number of feature values for an efficient discrimination. It must have common values for patterns within a class, and, at the same time, it must have a large difference for other class of patterns.

To select the good feature values some methods to evaluate its feasibility have been proposed. One is the *principal component analysis (PCA)* from the field of statistics, and another is the *eigenspace method* from the field of pattern recognition.

First, many features are extracted from a pattern. Let us assume to extract M features, and compose a feature vector $\boldsymbol{x} = (x_1, x_2, \cdots, x_M)^\top$. A given pattern corresponds to a point in the M-dimensional feature space.

The first requirement of independency of the feature values is that the pattern points are distributed widely and uniformly in the feature space. Actually and usually, the points stay within local and rather narrow potion of the space. However, we have some mechanisms to reproduce a new and lower dimensional feature vectors by combining some features, which are necessary and sufficient for the recognition. In some character recognition techniques, for example, more than three hundred feature values are listed casually and randomly, then about ten feature values are reproduced by the eigenspace method into a sufficiently efficient feature vector.

2.7 Eigenspace Method

Here, the solution by the eigenspace method is presented.

Figure 2.10 shows the schematics of the eigenspace method. Here, N samples having M dimensional feature vectors distribute in M-dimensional space. Then, we express the N samples in the linear summation of K ($K < N, M$) normal vectors, which are called the "eigenvectors." The N linear summations must have minimum errors as their average from their original sample vectors. That is, denoting

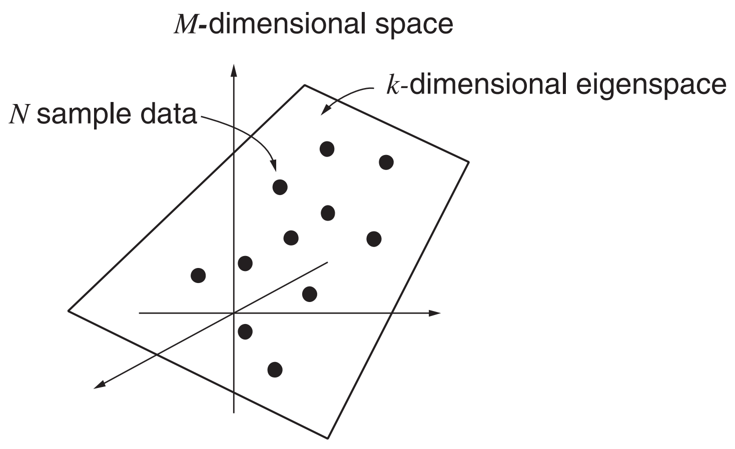


Fig. 2.10. Sample data and their eigenspace.

$$\hat{\boldsymbol{x}}_i = a_{i1}\boldsymbol{e}_1 + a_{i2}\boldsymbol{e}_2 + \cdots + a_{ik}\boldsymbol{e}_K \tag{2.17}$$

then, we select $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_K\}$ so as to be

$$J = \sum_{i=1}^{N} |\hat{\boldsymbol{x}}_i - \boldsymbol{x}_i|^2 \longrightarrow \text{minimum} \tag{2.18}$$

where the next equation must hold for all i,

$$|\boldsymbol{e}_i| = 1, \boldsymbol{e}_i^\top \boldsymbol{e}_j = \begin{cases} 1 : i = j \\ 0 : i \neq j \end{cases} \tag{2.19}$$

We must consider the set of normal vectors which are obtained from the covariance matrix of N samples of $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}$, that is the matrix given as

$$Q \equiv \frac{1}{n} XX^\top \tag{2.20}$$

where $X = [\boldsymbol{x}_1\, \boldsymbol{x}_2 \cdots \boldsymbol{x}_N]$, and the eigen values of the matrix of

$$Q\boldsymbol{e}_j = \lambda_j \boldsymbol{e}_j \tag{2.21}$$

Then, we have a series of the eigenvectors $\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_K$ in an order of the largest eigen values of $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_K$.

These eigenvectors are orthonormal with each other, and the coefficient of a_j for the $\boldsymbol{x}$ in (2.17) is given as

$$a_j = \boldsymbol{e}_j^\top \boldsymbol{x} \tag{2.22}$$

The individual sample of $\boldsymbol{x}$ is expanded by weights a_j as

$$\hat{\boldsymbol{x}} = \sum_{j=1}^{K} a_j \boldsymbol{e}_j \tag{2.23}$$

where this is called as the *Karhunen-Loève expansion*.

The eigenspace is defined as a space spanned by $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_K\}$. This space is given as a K-dimensional subspace in the M-dimensional space and including all the samples mostly within it. The original M-dimensional sample vector $\boldsymbol{x}$ can be expressed as a $K(\ll M)$ dimensional vector with the coefficients of $\hat{\boldsymbol{x}}$, as $\boldsymbol{a} = [a_1, a_2, \cdots, a_K]^\top$.

This subspace is known to reduce efficiently the dimensionality of samples without loss of information which the sample set originally had.

2.8 Economic Gain

Up to here, we discussed the performance of the pattern discrimination based on the probabilities of correct and false decisions. But, we have many cases where we must evaluate effectiveness of the correct decision by different criteria, or cases where the amounts of loss are largely different, even for the same rate of false decision.

Here, we introduce an evaluation of amount of effectiveness or gain for the decision, which is called the *economic gain*.

We consider a simple case of mechanical piece inspection. We are given a mechanical piece shown in Fig. 2.11, and decide it is good or bad according to the measurement of the size x. The sizes x of good pieces and bad ones distribute with probabilities $p(x|g)$ and $p(x|b)$, respectively, where g and b stand "good" and "bad," also respectively.

If these two distributions do not have overlapping area as shown in (a), we can decide the piece to be good or bad depending on whether the size x is larger or smaller than a value $x = \tau$ which is set between the two distributions. If these two have overlapping area as shown in (b), it is necessary to have some decision error, which is the same as with previous examples.

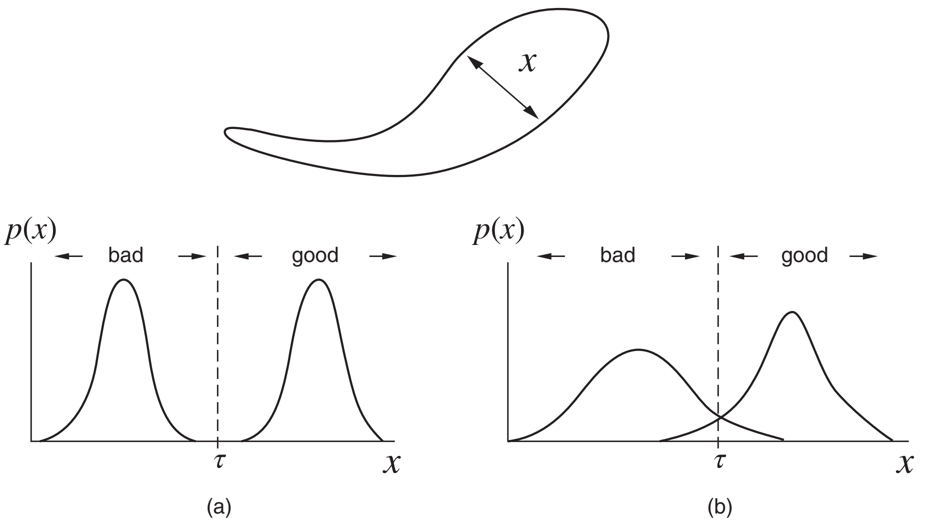


Fig. 2.11. An example of a decision for mechanical piece inspections according to a characteristic size x. (a) The case where we can make certain decisions of good or bad pieces. (b) The case we have some decision error.

Now, we assume the production cost of this piece to be 500Yen, and selling price to be 510Yen. Then, we have four types of *gains* according to the cases where the piece is *substantially* good or bad and the cases where it is *considered* as good or bad, as shown in Table 2.3, which is called an *economic gain matrix*.

This matrix tells that, if we sell a good piece as a good one, we gain 10Yen. On the other hand, if we decide a piece to be *bad* and scrap it while it is actually good or bad, we have a loss (or a negative gain) of production cost of 500Yen. A big problem may occur if we sell a bad piece as a good one. We must take the full responsibility for a possible accident. The matrix shows that it may cost 5,000,000Yen.

This matrix lists gains of $e(b,b), e(g,b), \cdots$, where $e(g,b)$, for example, is the gain when we make a decision for a good piece (g) to be bad (b).

Now, we introduce a boundary value $x = \tau$ for the decision, and then we have probabilities for the results of this decision. The set of those probabilities are listed in Table 2.5, which is called a *contingency table*.

In the table, $P(b,g)$, for example, is the probability of the decision for a bad piece to be a good one, and

$$P(b,b) + P(b,g) + P(g,b) + P(g,g) = 1 \tag{2.24}$$

Rewriting these probabilities into the conditional probabilities of the decisions under the states where the pieces were substantially good and bad, respectively, we have

Table 2.3. An example of the economic gain matrix for the good-bad decision of production piece shapes.

		Decision	
		bad	good
State	bad	–500 Yen	–5,000,000 Yen
	good	–500 Yen	10 Yen

Table 2.4. General form of an economic gain matrix.

		Decision	
		bad	good
State	bad	$e\ (b,\ b)$	$e\ (b,\ g)$
	good	$e\ (g,\ b)$	$e\ (g,\ g)$

Table 2.5. Contingency table.

		Decision	
		bad	good
State	bad	$P(b,\ b)$	$P(b,\ g)$
	good	$P(g,\ b)$	$P(g,\ g)$

$$
\begin{aligned}
P(g|g) &= \frac{P(g,g)}{P(g,g)+P(g,b)} \\
P(b|g) &= \frac{P(g,b)}{P(g,g)+P(g,b)} \\
P(g|b) &= \frac{P(b,g)}{P(b,g)+P(b,b)} \\
P(b|b) &= \frac{P(b,b)}{P(b,g)+P(b,b)}
\end{aligned}
$$

From these, we have $P(g|g) = 1 - P(b|g)$ and $P(b|b) = 1 - P(g|b)$. Also we have the probabilities for the cases where the production pieces are good and bad as $P(g) = P(g,g) + P(g,b)$ and $P(b) = P(b,g) + P(b,b)$, respectively.

Then, from these forms of probabilities, we obtain the "expectation of the gain" E as

$$
\begin{aligned}
E &= P(g,g)e(g,g) + P(g,b)e(g,b) + P(b,g)e(b,g) + P(b,b)e(b,b) \\
&= \big\{(1 - P(b|g))e(g,g) + P(b|g)e(g,b)\big\}P(g) \\
&\quad \big\{P(b|g)e(b,g) + (1 - P(b|g))e(b,b)\big\}P(b) \\
&= \alpha P(b|g) + \beta P(g|b) + \gamma
\end{aligned}
\tag{2.25}
$$

where α, β, and γ are constants independent of the choice of the boundary τ.

The probability $P(b|g)$ in the above form represents the probability of the decision for good pieces to be bad ones, and is called the *error of the first kind* or the *false detection*. On the other hand, $P(g|b)$ is the probability of the decision for bad pieces to be good ones, and is called the *error of the second kind* or the *misdetection*.

Generally, these two probabilities are incompatible. That is, reducing the error of the first kind, in turn, increases the error of the second kinds, and vice versa. Then, we make a compromise between them at the point where the E of (2.25) will be maximized.

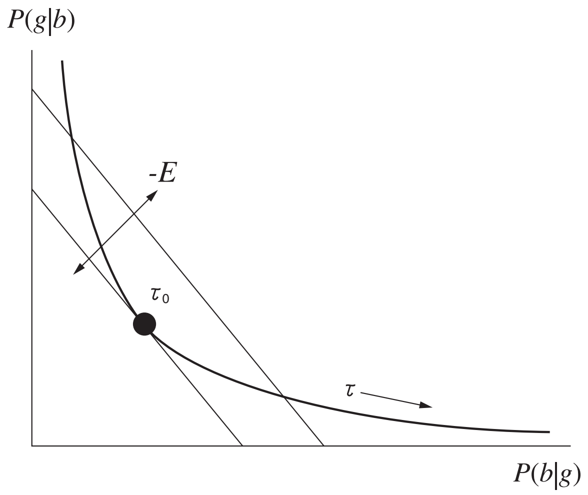


Fig. 2.12. Plots of the economic gain with respect to the boundary value (characteristic curve). Choosing $\tau = \tau_0$ for the boundary value, we make the economic value maximum (The loss will be minimum.)

By setting a value for the boundary τ, the values of $P(b|g)$ and $P(g|b)$ are determined, and then, the gain E will be given from (2.25). Usually, according to the value of τ, we have a plot in the $(P(b|g), P(g|b))$-plane, and by changing the value of τ, we have a curve just as shown in Fig. 2.12. This curve is called the *characteristic curve* [3].

From a pair of $P(b|g)$ and $P(g|b)$ on this curve, we have an expectation of the gain E by (2.25). Then, maximizing E just corresponds to the choice of their pair at the point on the curve where the $E = \alpha P(b|g) + \beta P(g|b) + \gamma$ becomes a tangent line to the curve. The value of τ which produces such $P(b|g)$ and $P(g|b)$ is the most reasonable boundary.

2.9 Clustering

2.9.1 Classes and clusters of feature vectors

The discussion up to here assumes that we are given a set of sample patterns which is known to belong to their respective classes, first, and then we design a classification mechanism for unknown patterns. This is called the "supervised" design.

On the other hand, we may have a problem where a plenty number of sample patterns are given, and we decide what types of classes must be prepared for these patterns. To solve this problem, we first examine how the sample patterns form clusters in the feature space. This process is called the *clustering*.

For example, as shown in Fig. 2.13, a set of two-dimensional feature vectors is distributed in the feature space. We have different strategies for the design of the classifier according to the available a priori knowledge about the object patterns.

It includes the knowledge that, within a cluster, the feature vectors form some special type of distribution, for example, a normal distribution. Another kind of knowledge is that, the type of distribution is not known, but the number of clusters is know a priory.

Commonly, within a class, some of feature values have always similar values. This fact is also one of the knowledge on the feature vectors. Clustering utilizes this knowledge.

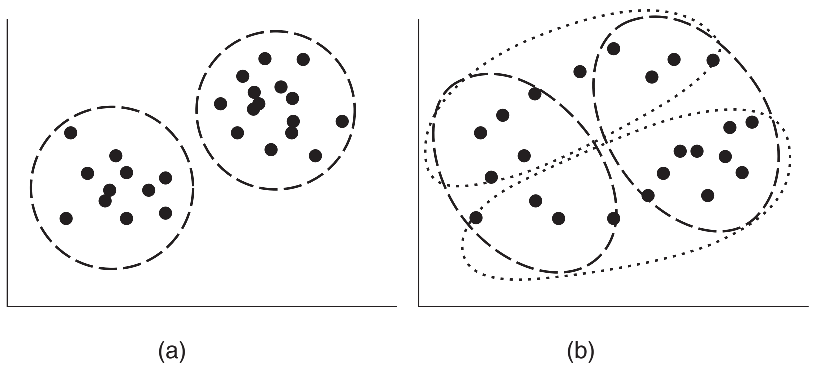


Fig. 2.13. Schematics of the clustering. We must clusters for a distribution of given feature vectors. (a) Example where obvious clusters are formed. (b) Example where clusters are not obvious.

2.9.2 Thresholding

The most typical and simple example of clustering is the binarization of images, where a set of gray level pixels are divided into two classes of black and white pixels. It converts a given gray scale image into a black-and-white binary image.

The key problem for the binarization is to find a proper boundary gray level for dividing to extract some region of interest out of the background in an image. For example, we must extract line and curve components beforehand for the character recognition.

To convert the gray level $x_{i,j}$ at a pixel (i, j) into either of binary values of 0 or 1 referring a threshold value τ is called the *thresholding*. Thresholding is basically assuming that the gray levels in an image are originally distributed at or around either of two specific values, and they were contaminated by noise. The proper boundary τ between these two specific values is not known and to be determined from the gray level distribution.

If this assumption holds, the gray level histogram forms two separated peaks and the threshold can be determined as the midpoint between those two peaks. On the other hand, if the two specific gray level values of X_1 and X_2 are mutually near-by, and there exists some noise, the histogram forms a shape of a mountain having two peaks just as shown in Fig. 2.14. For this case, the valley point of those two peaks is most likely to be considered as the threshold.

However, such clear peaks and the simple valley are seldom formed, and usually the histograms become uneven with many small notches. The idea against such an uneven histogram has been the modeling of the histogram with some probability distribution and setting the threshold onto the ideal model. The first model is that the gray level histogram should be composed with two normal distributions. It assumes that originally every pixel's gray level value should be either of two gray level values, but it suffered from natural noise and reading error to distribute with some spans. This distribution would be considered as the normal distribution because the noise and errors are most commonly modeled with a normal distribution [4].

That is, let the distribution be denoted with the mean of m and the variance of σ^2, as $N(m, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{(x-m)^2/2\sigma^2}$. Then, the gray level distribution is modeled as the composition of two normal distribution with the respective contributions of θ and $1 - \theta$ as

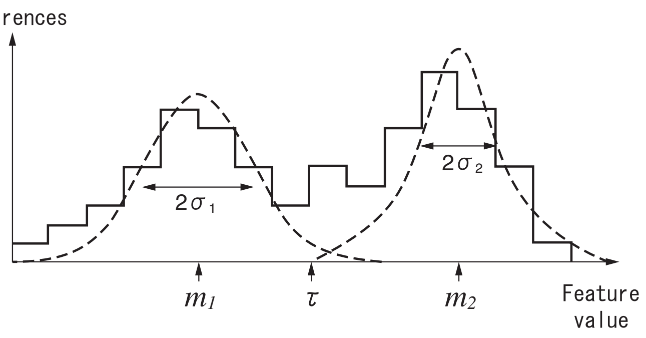


Fig. 2.14. An example of the gray level histogram and the model fitting.

$$
\begin{aligned}
p(x) &= \theta N(m_1, \sigma_1^2) + (1-\theta) N(m_2, \sigma_2^2) \\
&= \theta \frac{1}{\sqrt{2\pi}\sigma_1} e^{(x-m_1)^2/\sigma_1^2} + (1-\theta) \frac{1}{\sqrt{2\pi}\sigma_2} e^{(x-m_2)^2/\sigma_2^2}
\end{aligned}
\tag{2.26}
$$

where m_1 and m_2 are the mean points of two respective peaks, and σ_1^2 and σ_2^2 are their own respective variances (see Fig. 2.14). The total mean and variance are given as

$$
m = \int_{-\infty}^{\infty} xp(x)dx = \theta m_1 + (1-\theta) m_2 \tag{2.27}
$$

$$
\begin{aligned}
\sigma^2 &= \int_{-\infty}^{\infty} x^2 p(x)dx \\
&= \theta\sigma_1^2 + (1-\theta)\sigma_2^2 + \theta(1-\theta)(m_1 - m_2)^2
\end{aligned}
\tag{2.28}
$$

Then, by computing m and σ^2 from given data, we fit (2.26) to the gray-level histogram $p(x)$ with the constraints of (2.27) and (2.28). Commonly, the least square fitting is employed, but it has not reasonable basis for the histogram fittings. Moreover, m_1, m_2, σ_1, σ_2, and θ with the least square errors cannot be given analytically but by an iterative numerical computation. However, for the case where the histogram has two peaks, usually we have a stable threshold value.

We have another approach where the histogram is considered as a combination of two distributions, similarly, and employ the "discriminant criterion" for the distributions [5]. In this approach, also we basically employ the same assumption and express each individual distributions with two parameters as were for the normal distributions. But, we do not employ an explicit function form as in (2.26) for the distributions.

The discriminant criterion evaluates the goodness of the division of a given distribution into two new ones by the ratio of the between-class variance and the within-classes variance. We consider digital images, whose gray levels are quantized into integer values from 1 to L. When we set the threshold k for the division of given gray-level histogram into two classes, the discriminant criterion for this division is given as follows, and we can evaluate the optimality of the division.

Let the integer set of gray-levels of an image be $S = [1, 2, \ldots, L]$ and the proportion ratio of the pixels having a gray-level $i \in S$ be P_i. Then, the mean and the variance of the total gray-levels in an image are given as

$$m = \sum_{i=1}^{L} iP_i \tag{2.29}$$

$$\sigma^2 = \sum_{i=1}^{L} (i-m)^2 P_i \tag{2.30}$$

We divide the pixels into two classes C_1 and C_2 with the boundary level of k. Denoting

$$\omega(k) = \sum_{i=1}^{k} P_i \tag{2.31}$$

$$m(k) = \sum_{i=1}^{k} iP_i, \tag{2.32}$$

the proportions of the classes C_1, and C_2 are given, respectively, as

$$\begin{aligned} \omega_1 &= \omega(k) \\ \omega_2 &= 1 - \omega(k) \end{aligned} \tag{2.33}$$

The means within respective classes of C_1 and C_2 are given as

$$\begin{aligned} m_1 &= \frac{\sum_{i=1}^{k} iP_i}{\omega_1} = \frac{m(k)}{\omega(k)} \\ m_2 &= \frac{\sum_{i=k+1}^{L} iP_i}{\omega_2} = \frac{(m - m(k))}{1 - \omega(k)} \end{aligned} \tag{2.34}$$

and the variances within the classes are

$$\begin{aligned} \sigma_1^2 &= \frac{\sum_{i=1}^{k} (i - m_1)^2 P_i}{\omega_1} \\ \sigma_2^2 &= \frac{\sum_{i=k+1}^{L} (i - m_2)^2 P_i}{\omega_2} \end{aligned} \tag{2.35}$$

where it holds that

$$\omega_1 m_1 + \omega_2 m_2 = m, \text{ and } \omega_1 + \omega_2 = 1 \tag{2.36}$$

We call the average of two variances within two divided classes with the "within-class variance," and the variance of this average with the "between-class variance." These are expressed with σ_w^2, σ_b^2 as

$$\sigma_w^2 = \omega_1 \sigma_1^2 + \omega_2 \sigma_2^2 \tag{2.37}$$

$$\begin{aligned} \sigma_b^2 &= \omega_1 (m_1 - m)^2 + \omega_2 (m_2 - m)^2 \\ &= \omega_1 \omega_2 (m_1 - m_2)^2 \end{aligned} \tag{2.38}$$

The smaller the within-class variance is, the closer are the distributions of the individual classes. And the larger between-class variance, the better is the separation of the classes. Then, we evaluate the discrimination with the criterion of

$$\lambda = \frac{\sigma_b^2}{\sigma_w^2} \tag{2.39}$$

Actually, we reduce the computation by replacing it with

$$\eta = \frac{\sigma_b^2}{\sigma^2} = \frac{\lambda}{\lambda + 1} \tag{2.40}$$

because the σ is constant irrespective to k, and σ_b can be obtained from each average as (2.38). When $\lambda \geq 0$, η is monotonically increasing with respect to λ. It gives equivalent evaluation of λ for its minimization, resulting in the evaluation only with the between-class variance. If the variance of the total data is infinite, this means that the maximizing the between-class variance will be the optimal discrimination. We express the criterion as the function of k, again, as

$$\eta(k) = \frac{\sigma_b^2(k)}{\sigma^2} \quad \left(\text{where } \sigma_b^2(k) = \frac{(m\omega(k) - m(k))^2}{\omega(k)(1 - \omega(k))} \right) \tag{2.41}$$

and the optimal threshold k_{opt} is given as

$$k_{opt} = \arg\max_k \ \sigma_b^2(k) \tag{2.42}$$

Moreover in the actual computation, $\omega(k)$ and $m(k)$ can be simplified as a recurrence equations with respect to k, that is,

$$\omega(k) = \omega(k-1) + P_k \tag{2.43}$$

$$m(k) = m(k-1) + kP_k \tag{2.44}$$

and they can be utilized for the evaluation of (2.41).

Examples of image binarization with this criterion are shown in Fig. 2.15.

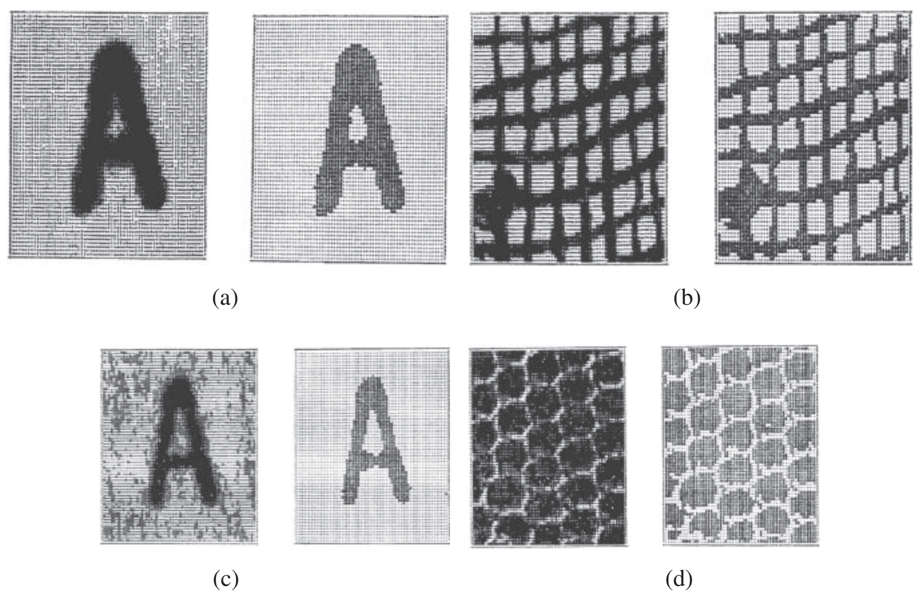


Fig. 2.15. Examples of image binarization by employing the discriminant criterion. Four gray-level images of (a)–(d) were converted into binary images shown on their respective right hand sides.

3. Structural Pattern Recognition

3.1 Structures of Patterns

3.1.1 Hierarchical structure of patterns

It was noted that the pattern recognition consisted of two steps of feature extraction and the structural analysis of the features. In this chapter, we further consider the spatial structure of patterns.

In Fig. 3.1, a simple 3D scene is shown. In this figure, a "tree structure" is also shown, which describes the scene [6].

The highest branching in this tree structure tells that the image of "SceneA" consists of two components of "ObjectsB" and "BackgroundC." At the next layer, the left hand branching tells that this ObjectsB consists of "ObjectD" and "ObjectE," and the right hand branching tells that the BackgroundC consists of "FloorF" and "WallG." Then, the lowest layer shows that the ObjectD has "FaceH" and "TriangleI," and the ObjectE has "FaceJ," "FaceK," and "FaceL."

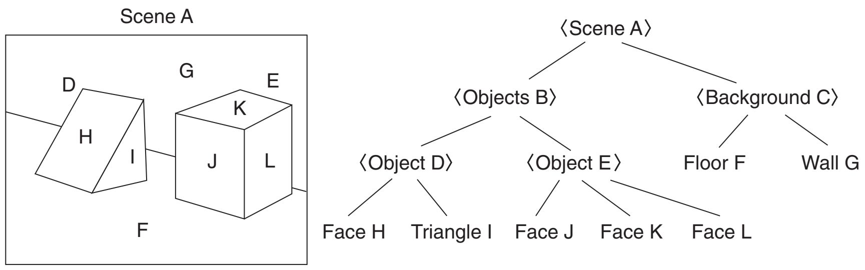


Fig. 3.1. An example of the tree structure description of the patterns.

In this tree structure representation, the actual visible components in the image are from FaceH to WallG, which are at the leaves of the tree. They are called *terminals*. Other symbols from SceneA to ObjectE, which are enclosed with < and >, are introduced for the tree structure representation, and called *non-terminals*.

This tree structure representation is common to that of the structure of a sentence as shown in Fig. 3.2.

Also in this representation, "Sentence," "SubjectPhrase," "VerbPhrase," "Verb" and "Object" are non-terminals introduced to describe the structure of sentences. Only the words of "a," "student," "reads," "the," and "book" actually appear in the sentence. These words are also called terminals.

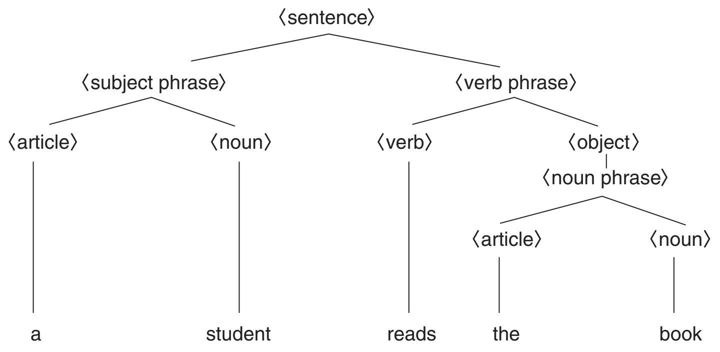


Fig. 3.2. A tree structure representation of a sentence.

A set of rules prescribing such the structure of sentences is called the *grammar*. The process of analysis by grammatical decomposition of "sentence → clauses → phrases → words" based on the grammatical rules is called the *syntax analysis* or *parsing*. That is, given a sentence, the syntax analysis is to examine whether, from the series of words in the sentence, such a hierarchical tree structure as shown in Fig. 3.2 can be composed.

A word or a series of words can be a "Verb" or a "Verb Phrase" only when it has some specific sequential relation to other words in the sentence. This means that, in the tree structure in Fig. 3.2, the branching at a node is permitted only with some limited combination of branches. For example, <SubjectPhrase> can be branching with <article> and <noun>, which means that <SubjectPhrase> can be decomposed into <article> and <noun> in this grammar. The set of such possible branchings is just the main part of the grammar.

Starting from a given sequence of words, if we can compose a tree structure which has only permitted branchings, the sequence can be considered to have a formal style of a sentence. Regardless if it has substantial meaning or not, the sequence is a formal sentence.

Comparing the Figs. 3.1 and 3.2 leads to the view point described here, where the pattern recognition and the scene analysis can be treated with the same frame-work of the grammatical analysis. This approach is called the *syntactic pattern recognition*.

The most worthy characteristic of the syntactic analysis is that, in spite that the grammar itself consists of a finite number of rules, an infinite variation of sentences can be generated based on the grammar. The essential device which enables to generate infinite variations of sentences from a finite number of rules is the *recursive definition* of the rules. The work of this device is described later with actual examples. This capability is also significantly fascinating for pattern recognition.

In the syntax analysis of a given sentence, first, it is examined to have a structure of a sentence and its tree structure is composed at the same time. Then, from the tree structure, the analysis whether it has some special meaning or not is examined. For the image pattern, the same approach can be applied to obtain the meaning of the scene.

We discuss its details in the next section.

3.2 Pattern Recognition by the Syntax Analysis

3.2.1 What is the grammar?

We already have grammars for natural languages, such as Japanese, English and so on. Sentences appearing in those natural languages have too large variations, and as the results they need complicated rules having many exceptions. On the other hand, along with the developments of high-level programming languages which have rich power of expressions, so-called the "formal grammars" were established. They rule formally the structure of sentences of program codes.

The formal grammar G is defined with quadruple of sets as $G = (V_N, V_T, P, S)$. The component sets are given as,

V_N : Set of non-terminal symbols introduced to describe the grammar

V_T : Set of terminals

P : Set of "rewriting rules"

$S \subset V_N$: Set of "start symbols"

where examples of V_N and V_T were shown at the beginning of this section.

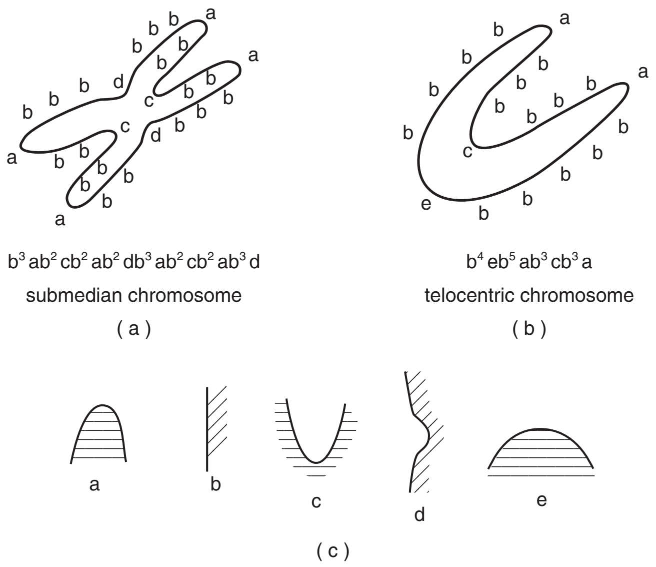


Fig. 3.3. An application of the syntactic approach for the pattern analysis of chromosomes. Two types of chromosomes of (a) and (b) are discriminated from their contour shapes. The given contour is converted into a sequence of local shape symbols shown in (c), and applied with the grammatical rules. b^3, for example, stands for three continuous b, and so on.

The terminals and non-terminals are denoted, usually, with some symbols. The rewriting rules are a set of rules which allow, in this grammar, to convert a partial sequence of such symbols α into another sequence symbol β.

An analysis whether a sequence of terminal symbols, which is just a sentence, satisfies the grammar or not is called the syntax analysis. This is equivalent to find a set of sequences of rewritings from one of the start symbol to the given sentence applying rewriting rules registered in the grammar. If we can generate the sentence with such the process from a start symbol, it is said to satisfy the grammar.

The features which compose a given pattern can be considered to be terminal symbols. For those features, a grammar describes their specific spatial allocations to compose a member of a pattern class. Given a pattern, it is converted into a sequence of features, then the grammar is applied to examine whether it is ruled by the grammar or not, and finally decided to belong to the class because it is generated from a start symbol by applying the rewriting rules.

As was mentioned above, the grammar which consists of a finite number of non-terminals, terminals, and rewriting rules specifies infinite variations of sentences. It is devised by the mechanism which allows recursive definition in the rewriting rules.

We show some examples of this mechanism.

Figures 3.3, 3.4, and 3.5 show a historical example which demonstrated the effectiveness of the syntactic approach to the pattern recognition [7]. (a) and (b) in Fig. 3.3 show two typical shape classes of chromosome ((a) submedian chromosome and (b) telocentric chromosome). A decision is to be made which class a give pattern must belong to or not belong to both. First, along with the contour, local shapes are picked up as one of the shape features of a, b, $\cdots$, and

$G_s = (V_N, V_T, P_s, \text{<s-mc>})$: Grammar of submedian chromosome

$G_t = (V_N, V_T, P_t, \text{<t-mc>})$: Grammar of telocentric chromosome

P_s : <s-mc> ⟶ <arm pair> d <arm pair> d

P_t : <t-mc> ⟶ <bottom> <arm pair>

<arm pair> ⟶ <arm> <right part>

<arm pair> ⟶ <left part> <arm>

<right part> ⟶ c <arm>

<left part> ⟶ <arm> c

<bottom> ⟶ b <bottom>

<bottom> ⟶ <bottom> b

<bottom> ⟶ e

<arm> ⟶ b <arm>

<arm> ⟶ <arm> b

<arm> ⟶ a

Fig. 3.4. The grammar and its rules for the chromosome shape analysis employing the syntactic approach.

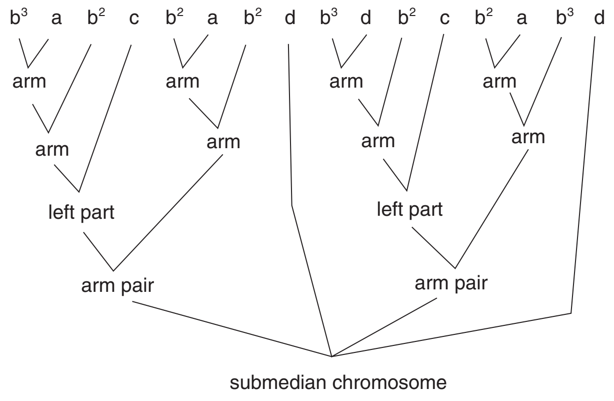


Fig. 3.5. A result of the syntax analysis of the chromosome shape. The contour shape given in Fig. 3.3(a) was discriminated as a submedian chromosome by the grammar G_s.

e in (c), and we obtain a sequence of symbols. The sequences of the symbols for the given contour shapes are shown in (a) and (b), respectively. The sequences are considered as “sentences.”

Figure 3.4 shows the both grammars corresponding to the submedian chromosome and the telocentric chromosome. The non-terminals in these grammars are enclosed with < and >, which are introduced to describe these grammars themselves.

The first line of the rewriting rules, for example, tells that the contour shape of the submedian chromosome <s-mc> is defined as the composition of <arm pair> *d* <armpair> *d*, where *d* is a local ∧-shaped curve shown in (c). That is, <s-mc> can be rewritten into <arm pair> *d* <arm pair> *d*. Then, <arm pair> is defined as <arm><right pair> or <left pair><arm>, and <arm> is defined as either of *b* <arm>, <arm> *b*, or *a*. Both of *a* and *b* are also local fragment curves shown in (c).

Each of <right pair> and <left pair> are defined, respectively, and finally described as sequences of local fragment curves from *a* to *e*.

Applying repeatedly the rewriting rules listed here, and from the start symbol <s-mc>, if we can generate a sequence of symbols such as shown in (a), it must be discriminated as a submedian chromosome. If we success to generate a sequence starting from <t-cc>, then it is a telocentric chromosome. If the both processes fail, the pattern belongs to none of those two classes.

The actual application of those rewriting rules is shown in Fig. 3.5. This figure demonstrates that the pattern (a) is decided to belong to the class of submedian chromosome and the tree structure was obtained for this pattern.

3.2.2 Types of grammars and parsers

Here we denote the rewriting rule with $\alpha \longrightarrow \beta$. Then the formal grammars are classified into next three types according to the constraints on the grammar. (If we count one more which has no constraint, we have totally four types.)

Table 3.1. Correspondences among the types of grammar, classes of languages, and available automata for the syntactic analysis parser.

Type of grammar	Language set	Parsing automaton
Type 0	Recursively enumerable set	Turing machine
	Recursive set	
Type I	Context-sensitive language	Stack automaton
Type II	Context-free language	Non-deterministic push-down automaton
	Deterministic lamguage	Deteministic push-down automaton
	LR-language	
	LL-language	
Type III	Regular set	Finite automaton
	⋮	

Type I: *Context sensitive grammar.* Denoting the lengths (numbers of composing symbols) of α and β with $|\alpha|$ and $|\beta|$, respectively, they must satisfy $|\alpha| \leq |\beta|$. This means that no symbols disappear by the rewritings.

Type II: *Context free grammar.* $\alpha \in V_N$ must holds. That is, a non-terminal symbol is rewritten into a sequence of words, and this rewriting must not depend on the pre-positioning or post-positioning words of the rewritten symbol. This is just the meaning of "context free." On the other hand, the Type I has no such limitation, so that it is "context sensitive."

Type III: If $A, B \in V_N$ and $a \in V_T$, only the rewritings of $\alpha = A$, and $\beta = aB$ or $\beta = a$ are allowed. That is, always one terminal symbol must appear by the rewriting.

These types of grammars have inclusive relations in order, that is, the Type III is the most strict and included in the Type II, which is next most strict and is included in the Type I. The condition for the Type III satisfies those of the Type II and the Type I, and so on. On the other hand the syntax analyzers or the parsers become simpler in the order of I, II and III.

The syntactic pattern recognition was a product of the combination of linguistics and the developments of programming languages, the computation theories, and artificial intelligence, whose research was rapidly accelerated in the period of 1960–70. From the view point of computational theory, the syntactic analysis can be automated by help of the theory of automaton. The correspondence between the types of grammar and the available automata for the syntactic analysis is shown in Table 3.1.

3.3 Automaton

A sentence ruled by the Type III grammar can be analyzed with a deterministic or non-deterministic finite-state automaton. The automaton is a machine which has internal states, and according to an input it transits the state. In the correspondence to the grammar, states are interpreted as non-terminals, and the rewriting rule of $A \longrightarrow aB$ as the transition from the state A to the state B with the input a. The context of "finite-state" means that the number of states is finite. The term of "deterministic or non-deterministic" means whether the state transition is uniquely defined for an input or not. (This will be seen in the following example.)

Now, let us see an example.

Figure 3.6 shows a simple optical mechanism to read out numbers. As shown in (a), number characters are passing above a row of five photo sensors from A to E. Every sensor outputs 0 or 1 at a clock period according that a spot of the pattern just on the sensor is white or black. At the time of $t = 1, 2, \cdots$, the photo sensors detect sets of black and white patterns shown in (b). Let us assign symbols from a to g for every set as shown in the figure, that is, for example, at $t = 1$ the sensors receive black and white pattern of a as shown.

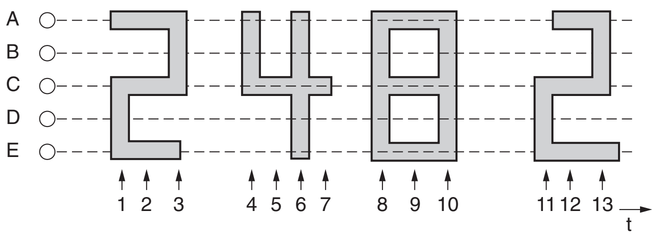
As a character is passing over the sensors, a sequence of symbols is output, which is interpreted as a sentence. We intend to read out the patterns shown in (b) as "2," "4," and "8," respectively.

For the character "2," the symbol a appears once or twice, and several times of b, then once or twice of c.

For "4," once or twice of d, several times of e, once or twice of f, and then also several times of e appear.

For "8," once or twice of f, several times of b, then again once or twice of f appear.

To read out those sequences, each automaton in Fig. 3.7 is composed, which corresponds to the rewriting rules of grammars. Finite state automata can be expressed with state transition graphs in the figure. In those graphs, the nodes are the states, and the directed and labeled arcs indicate that, when the machine is at some state and receives an input of



(a)

	Symbol	a	b	c	d	e	f	g
Optical sensor	A	●	●	●	●	○	●	○
	B	○	○	●	●	○	●	○
	C	●	●	●	●	●	●	●
	D	●	○	○	○	○	●	●
	E	●	●	●	○	○	●	●

(b)

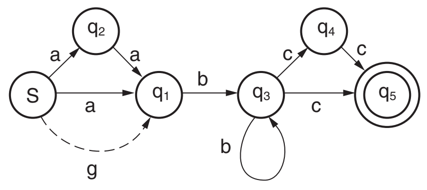
Fig. 3.6. A simple optical character reader, and sets of detected signal patterns.

one of the labels of the arcs starting from the node corresponding to the current state, then the state is transmitted to the state directed by the arc. Starting from special state *S*, through a series of state transitions according to a sequence of input symbols which are read out, and then, if we can reach one of the final states shown with double circles, the sequence is concluded as "2," "4," or "8," respectively. For this case, it is said that the grammar or the automaton accepts the input. On the way of the transition, if no arc corresponding to the input is found, they do not accept the input, and the character is not ruled by this grammar.

At every step of transition on the way, if only one arc is selected uniquely as an acceptable input, the automaton is said to be *deterministic*. If we have several candidate arcs and we must adopt all of them at this step, it is said to be *non-deterministic*. For the non-deterministic cases, one of the candidates will be selected after some future inputs as the result.

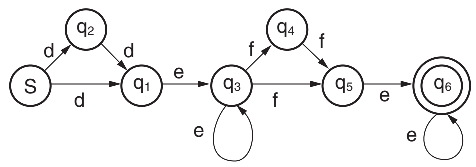
All of the automata shown in Fig. 3.7 are non-deterministic.

P_2:	$S \rightarrow aQ_1$	$Q_3 \rightarrow bQ_3$
	$S \rightarrow aQ_2$	$Q_3 \rightarrow cQ_4$
	$Q_1 \rightarrow bQ_3$	$Q_3 \rightarrow cQ_5$
	$Q_2 \rightarrow aQ_1$	$Q_4 \rightarrow cQ_5$



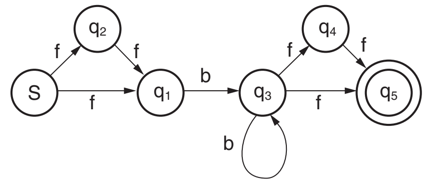
(a)

P_4:	$S \rightarrow dQ_1$	$Q_3 \rightarrow fQ_4$
	$S \rightarrow dQ_2$	$Q_3 \rightarrow fQ_5$
	$Q_1 \rightarrow eQ_3$	$Q_5 \rightarrow eQ_6$
	$Q_2 \rightarrow dQ_1$	$Q_4 \rightarrow fQ_5$
	$Q_3 \rightarrow eQ_3$	$Q_5 \rightarrow eQ_6$



(b)

P_8:	$S \rightarrow fQ_1$	$Q_3 \rightarrow bQ_3$
	$S \rightarrow fQ_2$	$Q_3 \rightarrow fQ_4$
	$Q_1 \rightarrow bQ_3$	$Q_3 \rightarrow fQ_5$
	$Q_2 \rightarrow fQ_1$	$Q_4 \rightarrow fQ_4$



(c)

Fig. 3.7. The rewriting rules of grammars to read out the characters of (a) "2," (b) "4," and (c) "8" with photo sensors, and their respective expressions of "non-deterministic finite state automata."

For the pattern recognition, the most significant advantage of those mechanisms is easiness of modification of grammars and automata adapting to the patterns to be recognized. For example, if we need to read the rightmost pattern in Fig. 3.6(a) as also “2,” we only need to add an arc of transition just indicated with a dotted curve and a symbol *g* in Fig. 3.7(a).

3.4 Grammatical Inference

The capability of a grammar, that is, what kind of characteristics of the grammar rules can describe what the extent patterns, has been studied deeply along with the developments of programming languages. As a result, wide variety of patterns can be discriminated by the syntactic analysis, and many of its effective applications have been developed. Actually, the initial automatic postal code recognition systems installed the syntactic analysis method.

However, successful examples were limited up to the cases only when their sophisticated grammars work well, and it has become common knowledge that the composition of such an excellent grammar is not easy. Example based generation of the grammar had been intensively studied, where sample patterns were shown first, then the grammar ruling those examples was automatically generated. But, almost all of them have failed. The grammar which accepts all the example patterns can be generated easily, but usually it cannot be effective.

Some successful approach can only deal with one-dimensional sequence of symbols. To convert a two-dimensional image pattern into one-dimensional sequence of symbols, the common strategies are, as was the example of the chromosome shape analysis, to extract features along with the contour of the shape, or to scan the image with raster-fusion, for example. On the other hand, two-dimensional grammars, such as shown in Fig. 3.8 have been designed, where two-dimensional relations between image features were introduced in rules. But, they have not achieved successful results.

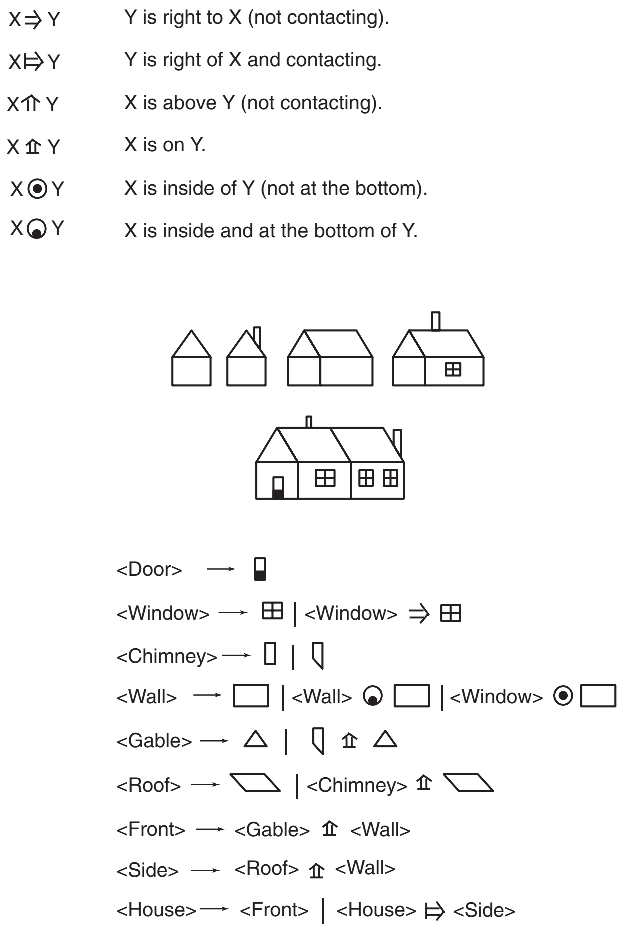

X ⇒ Y — Y is right to X (not contacting).
X ⤇ Y — Y is right of X and contacting.
X ⇑ Y — X is above Y (not contacting).
X ⇮ Y — X is on Y.
X ◉ Y — X is inside of Y (not at the bottom).
X ◉ Y — X is inside and at the bottom of Y.
<Door> →
<Window> → | <Window> ⇒
<Chimney> → |
<Wall> → | <Wall> | <Window>
<Gable> → | ⇮
<Roof> → | <Chimney> ⇮
<Front> → <Gable> ⇮ <Wall>
<Side> → <Roof> ⇮ <Wall>
<House> → <Front> | <House> ⤇ <Side>

Fig. 3.8. An example of two-dimensional grammar. Introducing such the two-dimensional relations as “left-hand-side,” “above” and so on, it was intended to rule the shapes of “houses.”

Part II. 3D Object Recognition

4. Image of 3D Solid Shape

4.1 Scene Analysis

4.1.1 Blocks world

From this chapter we discuss the problem to understand 3D scene from an image.

First, we limit the 3D world to a simple one. In this world, surfaces of all the objects are planes. That is, we assume that all the objects are polyhedra such as cubes, parallelepipeds, pyramids and so on, and there are not spheres, pillars and other objects having curved surfaces. Then, crossings of surfaces will be straight lines, and no curved lines exist in this world. Then the projections of the scene of this world on an image only consist of straight line segments.

Here, we constrain ourselves to the objects in our world so that, (1) edges are generated by meeting two surface planes, which are projected on an image as straight line segments. (2) Points at which three or more surface planes meet will be vertices. We apply one more limitation that (3) more than three surfaces do not meet at a point. That is, the vertices are the points where just three surface planes are meeting. The above examples of cubes, parallelepipeds, and triangular pyramids are members of objects in this world, but quadratic pyramids are not.

These constraints seem too rigid. However, because, for example, a quadratic pyramid can be interpreted as a pair of triangular pyramids attached to each other, they allow such objects as quadratic pyramids to exist in this world. It only means that almost all objects in this world are built up with rather simple shaped objects. A curved surface also can be interpreted as a patchwork of many small flat surfaces.

These polyhedral objects are just like blocks, then, such the world is called a *blocks world*. Figure 4.1 shows an example of image of a projection of a scene of the blocks world.

The scene of this world only consists of straight line segments of edges which are crossings of two planar surfaces. Its scene analysis is just an interpretation of a line drawing picture [8].

4.1.2 Labelling of lines in a scene

In Fig. 4.2, we show other example objects. (a) just looks like an L-shaped object, and it may not be interpreted as other shapes. (b) shows a cube attached with another small cube at its vertex. And, it also looks a cube which has concave hole at one of its vertices. We could not discriminate these two from this line drawing image, but there seems not to exist other shape than these two. In such a line drawing image, information about 3D scene structure is essentially ambiguous, and there may exist several interpretations. However, the number of possible interpretations is smaller than it would be expected.

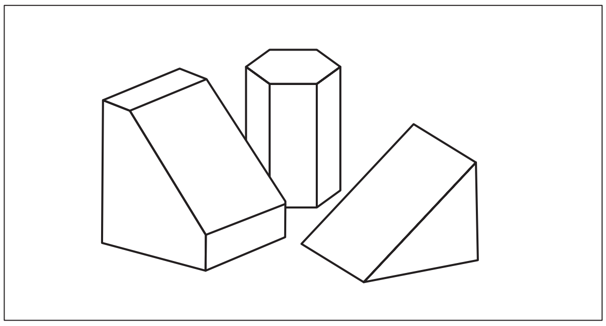


Fig. 4.1. A scene of the blocks world.

That is to say, there must exist some rules which make the line drawing of (c) be felt unnatural as an image of 3D scene. This section is intended to discuss and derive such rules.

We described that, in the scenery of our blocks world, all the lines were the boundaries of two planar faces of objects, that is, straight line segments of edges. We have two groups of edges; only one of the two side faces is seen or both of them are seen. (We have not such a case where an edge line is seen but none of face is seen.)

That is, actually, edges in line drawing scenes such as Figs. 4.1 and 4.2 are of three types of (a), (b) and (c) in Fig. 4.3. For a edge based analysis of line drawings, it is convenient to consider two types for (c); one is for the case where right side of the edge is the object face and left side is the background space, and the other is for the case where left side is the object face and right side is the background space. When we trace the edges, we are on either of those

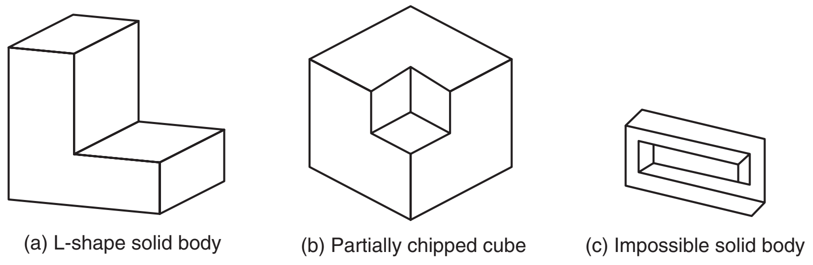


Fig. 4.2. Line drawings of (a) real existing object, (b) ambiguous object which has not unique interpretation for its form, and (c) false and non-existing object.

four types of edges from (a) to (d) in this figure. We denote these edges by symbols of +, −, —▶— and —◀—, respectively, as shown in the figure. (The reason why we need to discriminate —▶— and —◀— will be shown in the following example.)

Those symbols are called "(edge) labels," and the selection of labels for the respective edges in the line drawings is "labeling". The example of the labeling for Fig. 4.2(a) is shown in Fig. 4.4.

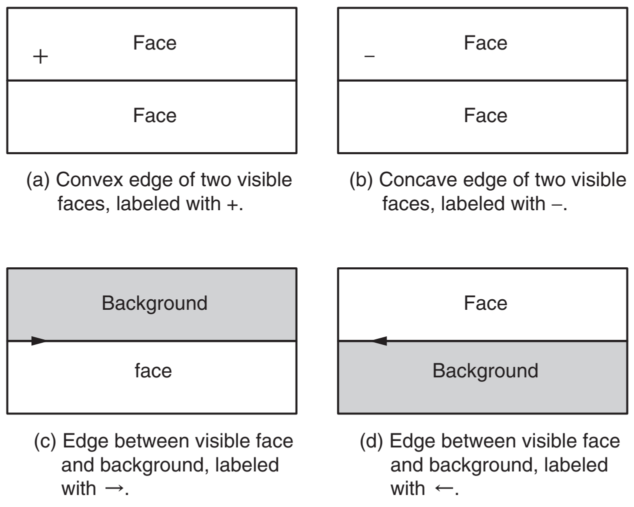

Face
+
Face
(a) Convex edge of two visible faces, labeled with +.
Face
−
Face
(b) Concave edge of two visible faces, labeled with −.
Background
face
(c) Edge between visible face and background, labeled with →.
Face
Background
(d) Edge between visible face and background, labeled with ←.

Fig. 4.3. Appearances of edges on a blocks world scene image.

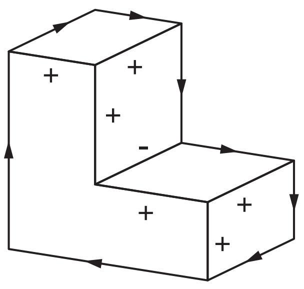


Fig. 4.4. An example of the labeling for line drawings.

4.1.3 Junction catalog and interpretation of line drawings

As was mentioned above, edges are crossings of two faces, and a point where three faces meet becomes a vertex. The vertex is also a point where different two or three edges meet, in the line drawings. In the blocks world, more than three faces do not meet at a point, then no more than three lines meet at a point in line drawings.

In the line drawings, when two lines meet and form an L-shape vertex, which is called *L-type vertex*, we may have four kinds of labeling for each line (note that we must discriminate —▶— and —◀—), and we may consider $4 \times 4 = 16$ combinations of labeling for this vertex as shown in Fig. 4.5. However, only smaller number of combinations than these 16 can appear actually.

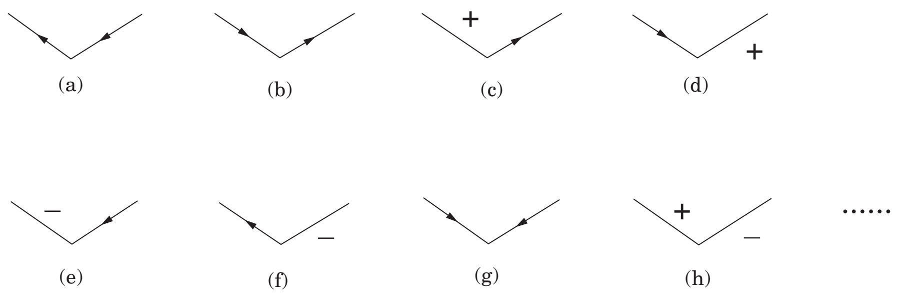


Fig. 4.5. L-type labeling. We may consider 16 labelings as a combinations of each line labeling. But only 6 labelings from (a) to (f) appear in the actual line drawings.

This is because: A vertex in the blocks world is a point where three edges meet in 3D space. If only two of those edges appear just as an L-type vertex as in Fig. 4.5, one of them is occluded by the body, that is, at least one of two edges must be labeled with —▶— or —◀—. As the result, only 6 combinations from (a) to (f) in Fig. 4.5 actually exist.

Next, we examine the vertices where all three edges appear in the line drawings. We can classify the appearances of the crossings into three types of *Arrow*, *Fork*, and *T* as shown in Fig. 4.6. That is, we discriminate them according that the angles between out-most edges are larger than, equal to, or smaller than rectangle. We overlook finer measurements of the angles. These three appearances of angles are essentially different from each other according to the viewing position relative to the faces meeting at this vertex.

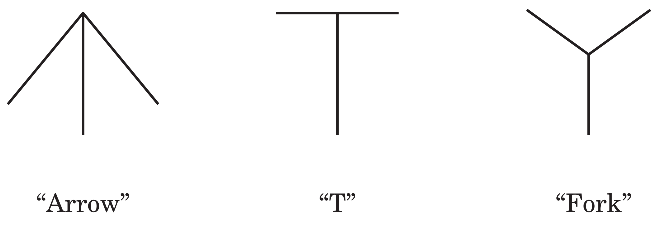


Fig. 4.6. Three types of vertices at which three lines meet.

For the *Arrow* vertices, if three of edges may have all possible labels independently, the total of $4^3 = 64$ types may exist. But, the number of actually existing labeling is only three. For the *Fork*, also only three can actually exist out of 64 (if we take the rotational symmetries into account, the total number of combinations of labeling becomes 24). For the *T*, the number of actual existing labelings is 4. All of these actually existing labelings are shown in Fig. 4.7.

We explain the reason why these numbers of the actual appearances are so limited.

In the blocks world, a vertex of an object is a point where three faces meet in 3D space. Three planar surfaces meeting at the point are schematically expressed as in Fig. 4.8, and they divide the 3D space into 8 portions. (In the figure, planar surfaces are shown to cross mutually in rectangular angle. But, this angle is not essential.) The meeting

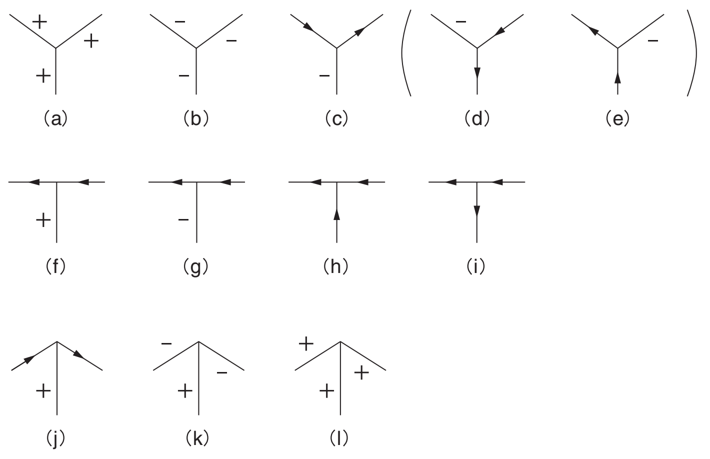


Fig. 4.7. Actually existing vertices of *Fork*, *Arrow*, and *T* types. ((d) and (e) are also registered when we count the different direction ones.)

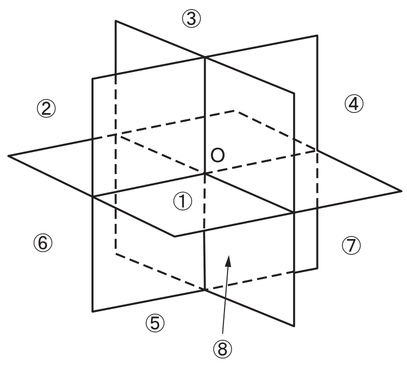


Fig. 4.8. Crossing of three planar surfaces meeting at a point. A vertex of a polyhedron is characterized according that which of 1/8 portions are occupied by the object.

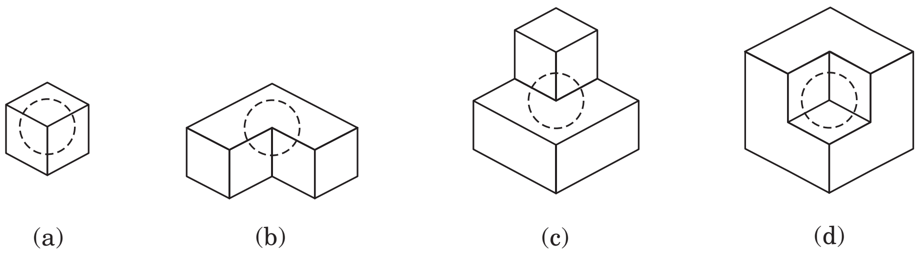


Fig. 4.9. The centre point of *O* in Fig. 4.8 will become a vertex only in these four cases.

point, which is denoted by *O*, can become an actual vertex only in the four cases shown in Fig. 4.9. This is when one, three, five, or seven of the 1/8 portions are occupied by the object. In other cases, the center *O* will not be a vertex but be a mere point in a plane or an edge.

Whether these four types of vertices will be seen as either of "Arrow," "Fork," or "T" is determined with the position of eye. That is, it depends on which of the 1/8 portions of Fig. 4.8 the eye position belongs to. Whether a planar surface can be seen or not is determined with which side of the surface it is seen from. For the object shown in Fig. 4.9(a), for example, we have 7 different viewpoints, because one of 1/8 portions is occupied by the object itself and cannot be a viewing position. The appearances of this vertex viewed from each respective 1/8 portion are shown in Fig. 4.10. They are finally grouped into 3, because (b), (d), and (e) are essentially same, and (c), (f), and (g) are also same.

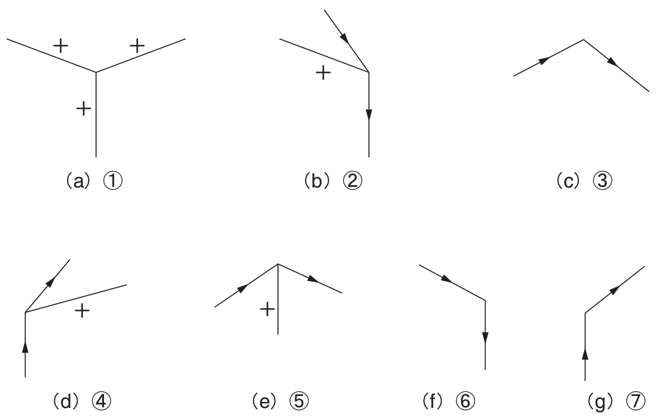


Fig. 4.10. The appearances of the vertex of Fig. 4.9(a). When the 8-th 1/8 portion is occupied by an object and the viewing position is placed in one of rest portions, the vertex will be seen as shown in this figure.

In the case where three 1/8 portions are occupied by an object, as shown in Fig. 4.9(b), the appearance of the vertex also depends on which of the rest of 5 portions the viewing position belongs to. They are shown in Fig. 4.11.

The rests in Fig. 4.9(c) and (d) are two cases where 5 and 7 portions of 1/8 spaces are occupied, respectively. For the former case, 3 views are existing, and for the latter, only one view is existing. They all are shown in Fig. 4.12.

To reduce duplicating views, only 6 views remain for a vertex produced with 3 planes, all of which are shown in Fig. 4.7. (If we separately count (d) and (e) which are rotations of (c), the total number is 8.)

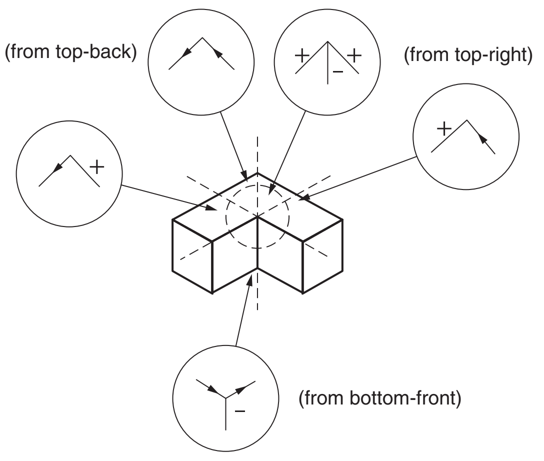
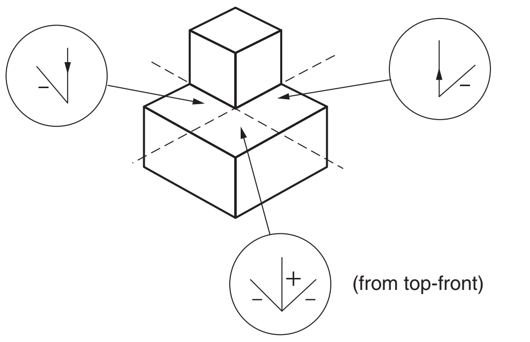
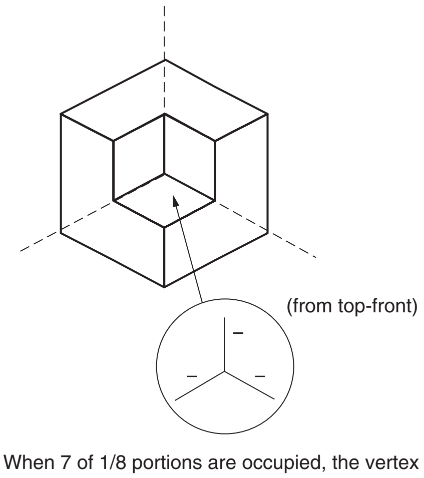


Fig. 4.11. When three of 1/8 portions, 6th, 7th and 8th ones, are occupied by the object, the viewing position can be placed in rest of 5 portions, and the vertex will be seen from respective portions as shown in this figure.



(a) When 5 of 1/8 portions are occupied, the vertex is viewed as shown from the rest 3 portions, respectively.



(b) When 7 of 1/8 portions are occupied, the vertex is viewed as shown from top-front portion.

Fig. 4.12. Other appearances of this vertex in the rest two cases.

We must consider further cases where a vertex is appearing on the image but actually it is not exist in 3D space. This is such case of the point as B or C in Fig. 4.13, where a part of an edge is just behind of another surface of a body. Three edges meet at B and C, But two of them are of the edge of the covering body, and it is different cases of the points A and D. These two edges are originally parts of a continuous straight line. That is, these vertices are categorized into type T, and the vertical segment is behind of horizontal two edges. Then, labels of these two are left-arrows (—◀—), and we have only 4 different views shown in Fig. 4.14 characterized with the different labels of the vertical edge.

To summarise the discussion here, we only have total 16 labelings of vertices shown in Figs. 4.5 and 4.7 (if we count additional two in Fig. 4.7, it becomes 18) are possibly appearing in an image viewing an actual 3D space, notwithstanding that the numbers of combination labelings are 16 for type L, 64 for type Arrow and T, and 24 for type Fork, which are totally 168 combinations.

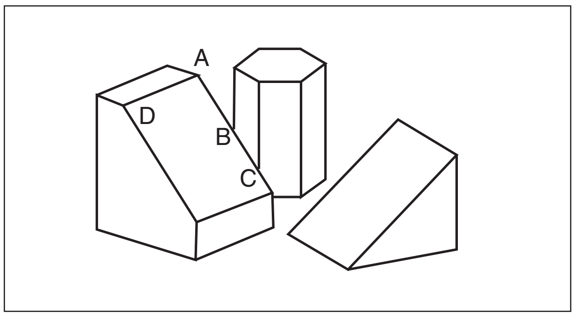


Fig. 4.13. Type T vertices. The vertex points B and C are pseudo ones produced when a part of an edge is covered by another body surface. These are different cases of the point A or D.

This fact was found by D. A. Huffman and M. B. Clowes, independently but almost at the same time of early year of 1970’s [9]. Then, these list of actually existing vertices are called the *Huffman-Clows Junction Catalog*. Of course, this is the fact in a very limited blocks world. However, we may relax the limitation step by step and extend the world to actual one.

4.2 Interpretation of Line Drawings with the Junction Catalog

4.2.1 Labeling rules and its 3D interpretations

Only the vertices listed in the Huffman-Clowes Junction Catalog appear in a line drawing which is a projection of the 3D world. This is the basic rule, so that let us call it the “Rule 0.” Now, we discuss what can be derived from this rule.

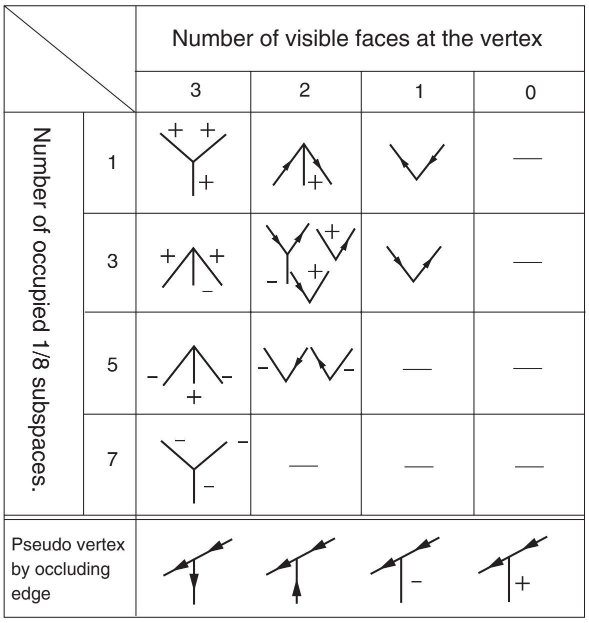

Number of visible faces at the vertex
3
2
1
0
Number of occupied 1/8 subspaces.
1
3
5
7
Pseudo vertex by occluding edge

Fig. 4.14. Summary of actually existing vertex labelings including 4 of the type T vertices.

First, as was described, each respective edge has either label of “+,” “−,” “—▶—,” or “—◀—.” Every edge does not change its label without meeting a vertex. A label “+” of an edge will not be “−” at a midway if it is edge of a polyhedral body. Then, the edge should have same label at its both ends of vertices. Let us call this the “Rule I.”

We have one more rule on the boundary of a body. The out-most contour loop of edges is the boundary between the body and the background. Following this contour clockwise, for example, all the edges on the loop should be labeled with “—▶—.” We call this the “Rule II.”

For the line drawing of Fig. 4.15(a), starting to apply the Rule II, we are going, then, to label all the edges by applying the Rule I. We should not contradict with the Rule 0.

The process of this labeling is shown in Fig. 4.15(b), (c), and (d). First, edges on the out-most contour are given labels of “—▶—” as in (b). Then, at the vertex A on the contour, for example, edges which are not labeled yet are given proper labels which do not contradict with the Rule 0 as in (c). For this vertex, we have no other labeling in the Junction

Catalog than "+," it was uniquely determined. If the vertex at the other end of this edge has edges which have not been labeled yet (as the vertex B in (c)), we proceed to label these edges holding the Rule 0 with other edges. Finally, for this line drawing, we have unique labeling as shown in Fig. 4.15(d). This result agrees with our natural recognition.

Other examples are shown in Fig. 4.16. Structures of a hole of the box and a complicated combination of bodies are correctly interpreted by these labelings.

4.2.2 Ambiguity of labeling

Figure 4.17 shows examples which are unable for the labeling.

For the line drawing (a), we have not labeling for the vertex P in the Junction Catalog. This is because we cannot produce such the body as (a) only with planar surfaces, and it is not a polyhedron. Also in (b), at the vertex P, we have impossible labeling. If this body is just like a box, at the vertex A four planes meet and the vertex A has a wrong labeling. This results in the deadlock for labeling to P.

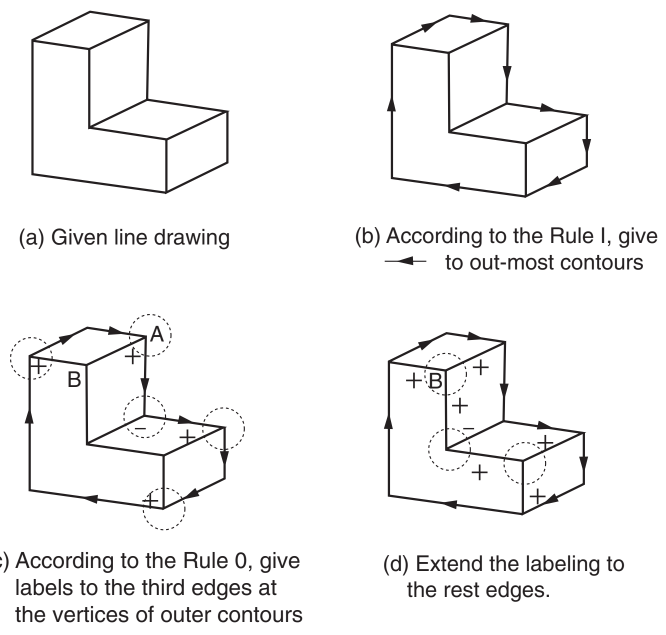

(a) Given line drawing
(b) According to the Rule I, give ⟵ to out-most contours
A
B
+
−
(c) According to the Rule 0, give labels to the third edges at the vertices of outer contours
(d) Extend the labeling to the rest edges.

Fig. 4.15. An example of interpretation of line drawing with the Junction Catalog.

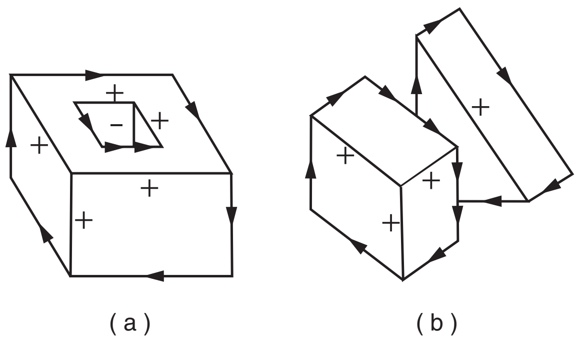


Fig. 4.16. Examples of complicated 3D scene interpretations. By the same process of Fig. 4.15, these line drawings had correct labelings.

These examples show that the interpretation discussed here is not always applicable for bodies out of the blocks world. In the blocks world, the theory of Junction Catalog completely describes the properties of the line drawings of the 3D bodies, and always the interpretation based on the theory should be applicable for every line drawing images. But, it is constructed with too simplified rules as shown here, we have also some examples which outwit this scheme.

For an example, as shown in Fig. 4.18(a) and (b), we have such two interpretations. (a) is a reasonable and proper interpretation, but (b) does not actually exist. We cannot exclude such a false interpretation by the theory.

We have another type of impossible interpretations such as shown in (c). The objects shown here cannot exist in the real world. They originate from the contradiction to the rule that an edge produced by a crossing of two planes must be

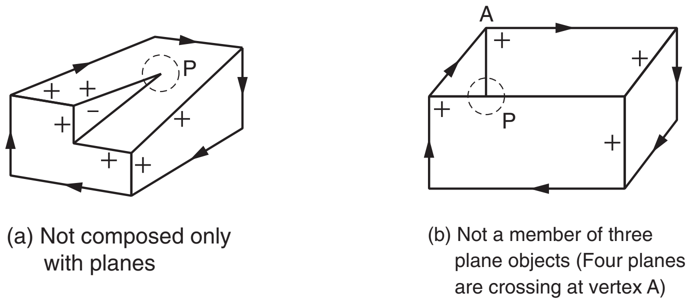

P
A
P
(a) Not composed only with planes
(b) Not a member of three plane objects (Four planes are crossing at vertex A)

Fig. 4.17. Examples of impossible labelings. In both cases, at *P*, possible labelings are not found in the Junction Catalog.

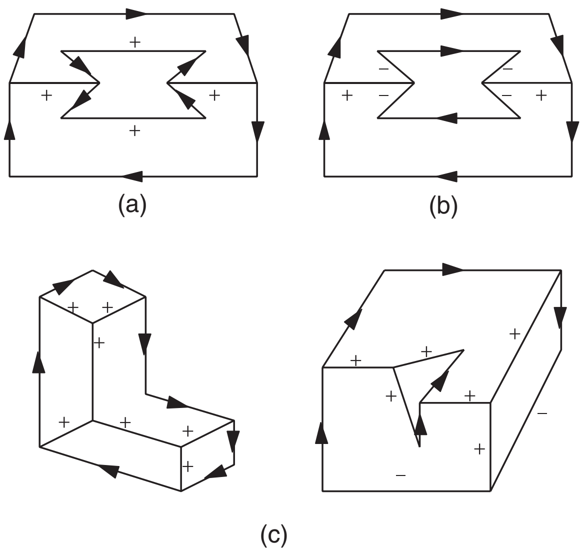


Fig. 4.18. An example having two possible labelings of (a) and (b), and (c) other ones having no possible interpretation.

straight. Even if an edge disappears on the midway as in (c), this rule must hold. But, the naive and simple theory cannot demolish the argument of outliers of the strict rules.

Our theory only consider whether the angle between edges at a vertex be acute or obtuse. To make an accurate interpretation on a appearance of an existing polyhedron, we must employ finer measurements for angles and lengths and apply more strict constraints.

However, even with the incomplete theory, the interpretation based on the Junction Catalog covers and applicable for wide variety of line drawings. The discovery of the capability of such simple rules for the interpretation of images was a great step for the progress of image recognitions.

4.3 Philosophy of the Line Drawing Interpretations

Before introducing fine measurements, let us discuss the feasibility of the line drawing interpretation by relaxing the restriction of the blocks world and reviewing the rules from another stand point of view.

The line drawing interpretation was initially introduced to computer vision society by L. Roberts (1965) [10]. He considered a solid body, whose line drawing is, for example, shown in Fig. 4.19(1), as a set of simple convex polyhedra shown in (2), which were called “primitive polyhedra.” Then, he examined the processes to decompose the solid body into primitive polyhedra and reversely reconstruct it from the primitive polyhedra as shown in (3).

Roberts suggested line segment cues for finding composition primitives by searching such line structures as shown in Fig. 4.20 in the image. The searching is carried out in the order of (a), (b), (c), and (d). This search is accomplished by matching in phase, regardless of orientation and scale. Then, identifying convex vertices and grouping them, we can cut down the primitive polyhedra from a solid body. This is just to utilize the constraints on the appearances of primitive polyhedra, which is common to the concept of the Junction Catalog.

Roberts’ decomposition of a solid body into primitives became a basic strategy of the “Constructive description approaches” and attracted a great deal of attention again later.

A. Guzman (1968) [11] followed this work. His objective was also decomposing such the line drawing as shown in Fig. 4.21 into primitive polyhedra. He examined appearances of convex vertices in more detail, and found next facts: If all the vertices in a line drawing are convex, where their appearances are classified as shown in Fig. 4.22, surfaces linked with an arc in the figure are belonging to same solid object. In the line drawing, a polygon having a continuous contour edge is a image of a surface of a solid object. Then, by making up linked polygonal surface into a group, a solid object (or its part) can be composed.

For example, in the line drawing in Fig. 4.23, the surfaces A, B, and C are mutually linked and found to compose an object. Also D and E compose another object. The relation between these two objects cannot be resolved from this line drawing, but it can be easily guessed that this is a scene of a stack of two rectangular parallelepipeds.

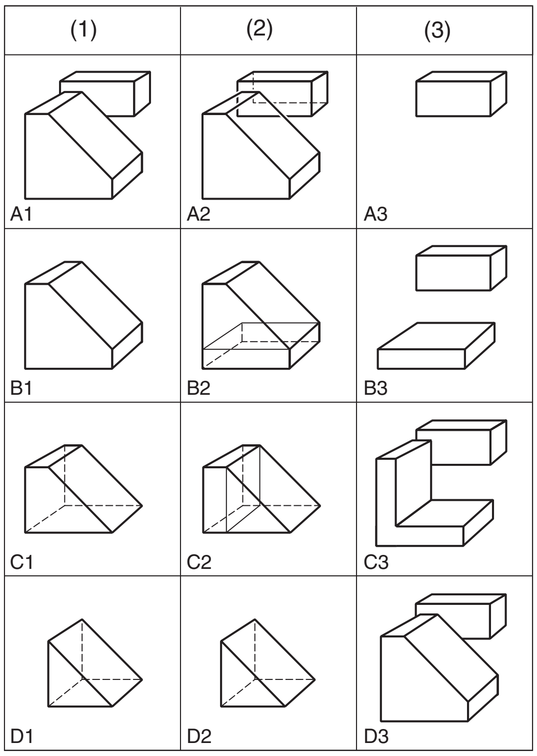


Fig. 4.19. Roberts' decomposition and reconstruction of solid bodies. (1) Decomposition into primitive polyhedra, (2) Finding composition primitives, and (3) Reconstruction of the original solid.

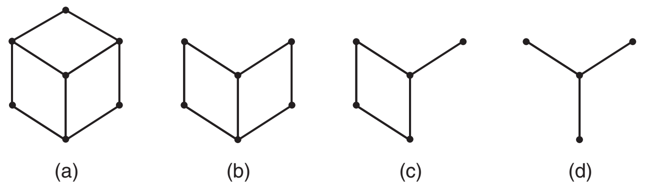


Fig. 4.20. Roberts' line segment cues to find composition primitives in line drawing.

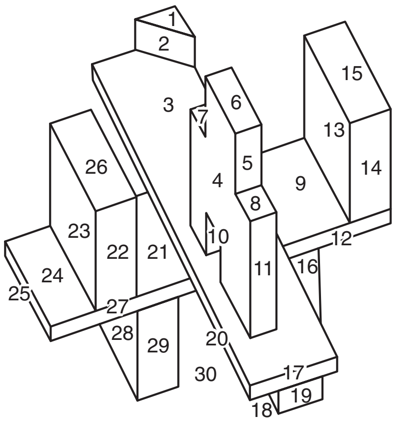


Fig. 4.21. An example of line drawing which Guzman examined.

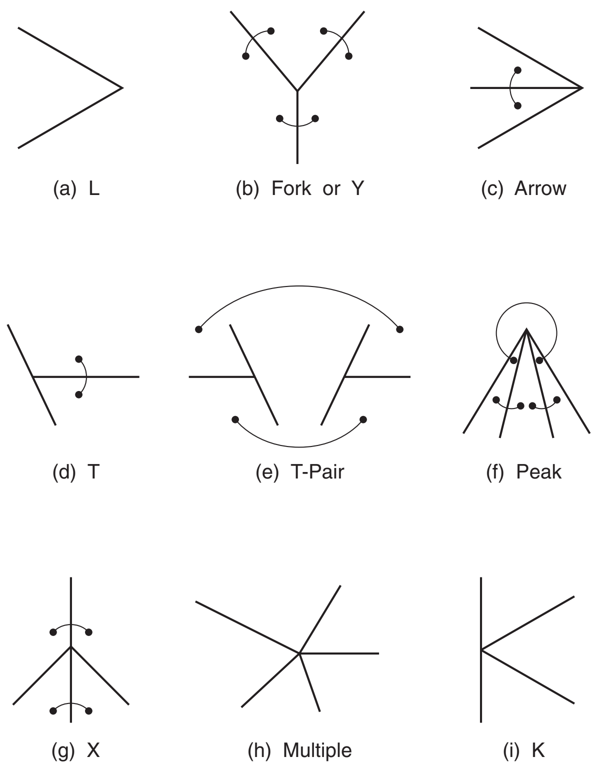


Fig. 4.22. Guzman's classification of vertices and surface links. Surfaces linked by an arc, which are marked with ●, are belonging to same solid object.

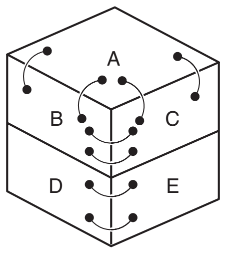


Fig. 4.23. A solid object interpreted with the Guzman's theory.

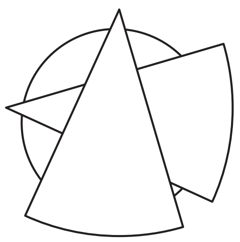


Fig. 4.24. Line drawing which can be interpreted as a stack of three solid bodies by an extension of the idea of Guzman's link.

The basic idea common to them is: from local features of a line drawing, such as properties of appearances of vertices, by propagating their information to whole image keeping mutual consistencies, the global interpretation of the object's 3D structures can be obtained. This idea was succeeded by the concept of the Huffman-Clowes Junction Catalog. Actually, the Guzman's link is not always accepted. But, it was proved that, for a vertices isolating in the blocks world where only three surfaces meet at a vertex, the link rightly held. Later, it is extended to the interpretation, for example, of the line drawing of Fig. 4.24 to be a stack of three solid bodies.

4.4 Extensions of Line Drawing Interpretations

Now let us review how the Huffman-Clowes Junction Catalog and the interpretation of line drawings based on it have been extended.

The limitations of their ideas mainly came from the next two: Objects were only of the blocks world where all the surfaces of bodies are plane. And the rules are provided only for the phases of relation between edges and vertices, that is, lengths of edges or angles of vertices are very roughly described.

Here, we discuss, if we relax the first limitation, what kinds of interpretations can be enabled and what kinds of difficulties might be introduced. On the second limitation, we will discuss in a later chapter.

D. Waltz (1975) [12] discussed more realistic application of the theory of blocks world. He considered such actual line drawings as shown in Fig. 4.25 including other lines than edges which were crossings of planar surfaces. They were lines of boundaries of shadows, cracks, and dividing lines between two stacking bodies shown in Fig. 4.26.

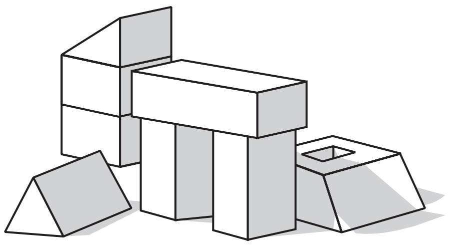


Fig. 4.25. Line drawing image including shadow regions which was treated by Waltz.

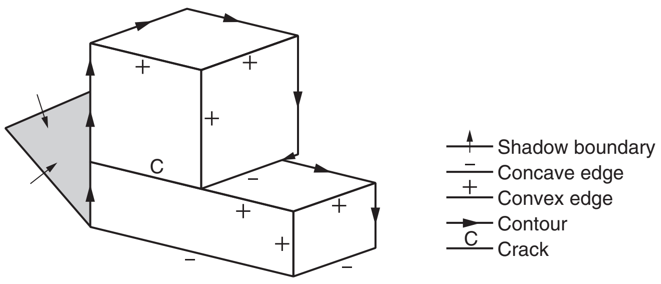


Fig. 4.26. Waltz's labeling for a line drawing having shadow and cracks.

To describe them, he claimed to add new line labels for a line drawing shown in Fig. 4.27, the resulting total labels were classified as shown in Fig. 4.28.

It is noted that, at a vertex of a block where only three surfaces meet may have four edge lines as shown Fig. 4.29. In this figure, area A is supposed to be the background. If area D is a shadow region, only the labeling shown in this figure is accepted, because the edge between C and D must be concave and, then, as the result, the edge between B and C must be convex.

As shown here, Waltz succeeded to give proper interpretation to the line drawing of Fig. 4.26 as well as to specify which edges are boundaries of the body and where is shadows.

This approach of extension to accept more actual cases, however, cannot be a good strategy. If we employ more labels to fit the actual situations, because we have a combination of labeled edge at a vertex, the variety of vertices grows extra rapidly in its number. This is called a "combinatorial explosion." But, Waltz pointed out a very important fact that, for the growth of the combinatorial number, the number of actually existing vertices will be orders of 0.06% or 0.0001% of the combinations as shown in Table 4.1.

Waltz showed that only unique possible interpretation for the line drawing of Fig. 4.26 could exist under the constraint that all the possible vertices must be mutually compatible.

Other approaches of extension have been proposed such as, for example, an extension to the "Origami world" where not only polyhedra but also bodies made of thin planes as an origami exist, and a world containing cylinders, cones and simple curved surfaces. In the former case, there would exist vertices at which four surfaces meet, then additional registration of acceptable types of vertices are needed in the Junction Catalog.

If we permit curved surfaces, we will have not only enormous number of vertex types but also loss of the consistency of edges. This consistency rules that an edge must not change its label on the way between vertices nor disappear without reaching at a vertex, which play important role in the interpretation of line drawings. Then, our basic concept of interpretation by propagating local properties of vertices to whole line drawing image keeping constraints of the Junction Catalog will not hold anymore. This requires almost entire reconstruction of our framework.

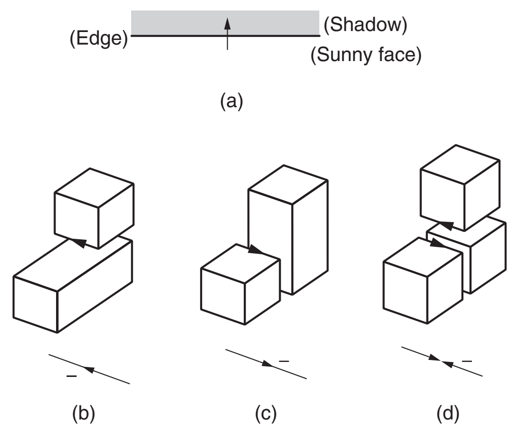


Fig. 4.27. New line labels added for Waltz's interpretations.

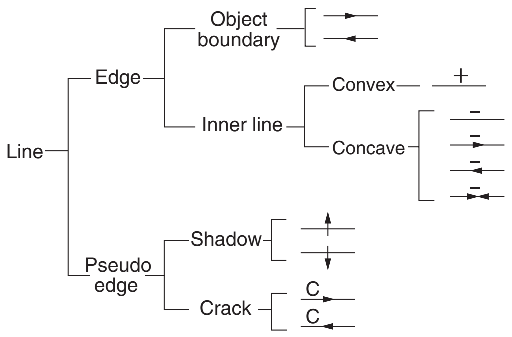


Fig. 4.28. Classification of Waltz's line labelings.

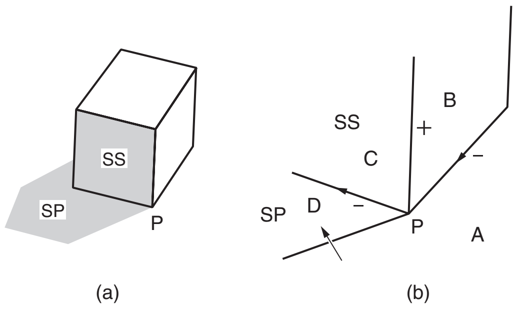


Fig. 4.29. (a) A vertex where four lines are meeting will exist by considering shadows (at P). (b) Only unique interpretation is possible if the regions *SS* and *SP* were known to be in shadow regions.

Table 4.1. The ratios of numbers of actually existing labels for vertices in Fig. 4.22 to the combinatorial numbers.

Vertex type	Number of labeling combination	Number of possible labeling (approx.)	Rate
L	2.50×10^{3}	80	3.20%
Fork	1.25×10^{5}	500	4.0×10^{-1} %
T	1.25×10^{5}	500	4.0×10^{-1} %
Arrow	1.25×10^{5}	70	5.6×10^{-2} %
T-Pair	6.25×10^{6}	300	4.8×10^{-3} %
K	6.25×10^{6}	100	1.6×10^{-3} %
X	6.25×10^{6}	100	1.6×10^{-3} %
Multiple	6.25×10^{6}	100	1.6×10^{-3} %
Peak	6.25×10^{6}	10	1.6×10^{-4} %

5. Geometry of Image of Space (I)

5.1 How is an Object Seen?

5.1.1 Representation of image of a 3D plane

The scene interpretation discussed in the previous chapter is based on the rough relationships between lines. It is noted that even such rough relations lead to plenty of 3D information about an object. However, to obtain much finer properties of the shape we need geometrical measurements such as line lengths and angles. Here, let us discuss additional rules how the measurements affect the appearances of objects, and how to reconstruct more correct original 3D shapes from their images considering the measurements.

First, we extend the previous theory of lines to that of planes. Indeed, lines in a line drawing are just the boundaries of planes, that is, in this sense, they are descriptions of planes.

As was described, lines in a line drawing image of polyhedra are of a contour of an object to background or a crossing of two planar surfaces. The former one is recognized as a chain of out-most boundaries. The latter is generated when two planes cross each other, so that it inherently succeeds the properties of the crossing planes. Therefore, it is important to characterize the geometry of lines to discuss on the image of 3D planes.

From here, we need to have some notions of 3D objects and their mathematical expressions.

First, we introduce a measure to express the amount of slant of a plane. It is the measure of *gradients*. It is defined as the relative angle of a plane to the image plane. The image of a scene is the projection of 3D space on the image plane as show in Fig. 5.1. On the image plane, x-y axes are placed and an image point is located with the x-y coordinates.

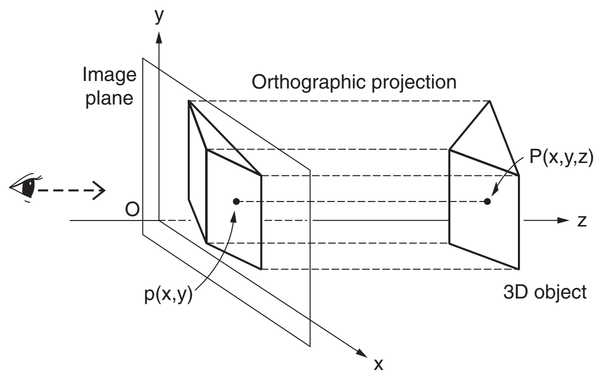


Fig. 5.1. An object in 3D space and its image (in the case of *orthographic projection*).

For the simplicity, the view point is assumed to be far from the object in comparison with the size of the object. Then, a point at (x, y, z) in 3D space is projected at (x, y) on the image plane. This is called the *orthographic projection*, which is commonly used in mechanical drafting. Projection of straight lines in 3D space will be straight lines on the image.

When the object is nearer or its size is bigger, we must employ more sophisticated projections. We will discuss about those projections in the next chapter.

Now, we consider a plane in 3D space. It can be considered as a surface of a polyhedron.

Generally, a plane in 3D space is expressed in the coordinates system of Fig. 5.1 as an equation of

$$z = px + qy + K \tag{5.1}$$

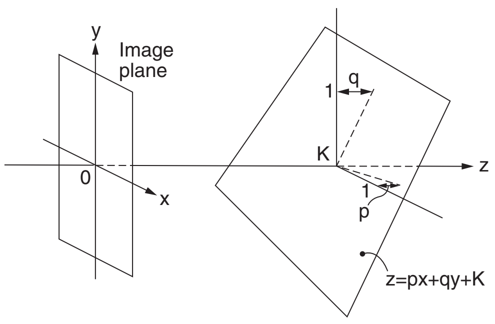
In this equation, the coefficients p and q express the amounts of inclinations in x and y directions, respectively, because z value increases by p on the plane if x value, for example, increases by 1 (See Fig. 5.2(a)). That is, the pair of coefficients (p, q) of (5.1) defines the orientation of the plane in space. Then, this (p, q) is called the *gradient* of the plane. The plane with $(p, q) = (0, 0)$ is just parallel with the image plane and normal to the line of sight (z-axis).

The vector $(p, q, -1)$ in this (x, y, z) coordinate space becomes normal to the plane having the gradient of (p, q) (See Fig. 5.2(b)).

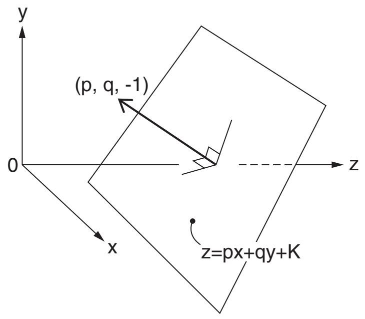
A. Mackworth (1973) [13] showed that the image of the line which was crossing of two planes was constrained with the gradients of those planes. This is explained in the following subsection.

5.1.2 Gradient space and a crossing line of planes

In an image, we have a line L which is the image of the crossing line M of planes P_1 and P_2 (Fig. 5.3(a)). The line L



(a) Plane in 3D space and its gradient (p, q).



(b) Vector (p, q, –1) becomes normal to the plane $z = px + qy + K$.

Fig. 5.2. Expression of a plane in space.

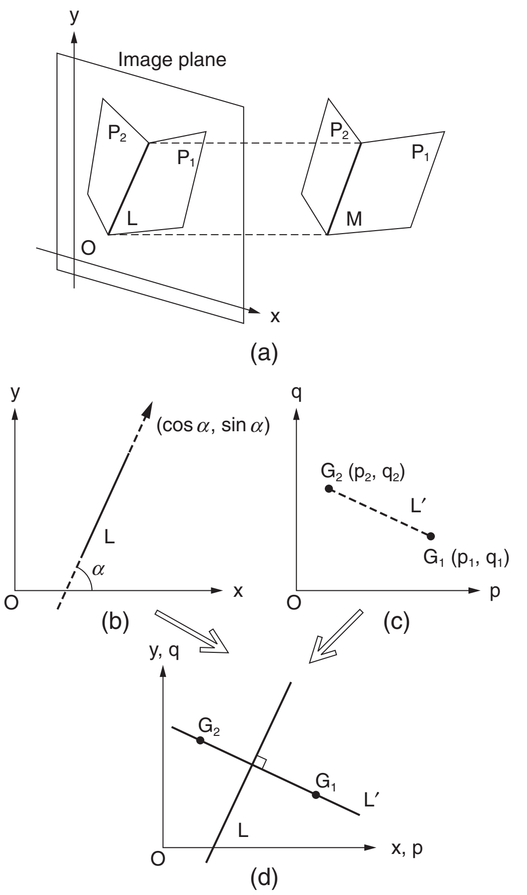


Fig. 5.3. (a) Crossing line M of two planes P_1 and P_2 in 3D space and its image L. (b) Image of a crossing line of two planes, and (c) a line passing through each of gradients of planes meet at right angles (d).

has an angle α to the x axis in the image as shown in Fig. 5.3(b). This means that the orientation of the line L is expressed with a vector $(\cos\alpha, \sin\alpha)$.

The original 3D line M must have the orientation of $(\cos\alpha, \sin\alpha, \beta)$, where β is unknown, because its orientation has the same angle in x plane and only has ambiguity in depth (z) direction.

We denote the gradients of these two planes P_1 and P_2 by $G_1 = (p_1, q_1)$ and $G_2 = (p_2, q_2)$, respectively. Now, we consider a 2D space spanned by p and q axes, in which G_1 and G_2 are points in this space as shown in Fig. 5.3(c). This space represented by (p, q) is called the *gradient space*. A line L′ in the gradient space and passing through both of G_1 and G_2 has an orientation vector $(p_1 - p_2, q_1 - q_2)$ in this space.

Let us go back to consider about the 3D crossing line M again. It lies on both planes of P_1 and P_2, so that it has right angles to both of normal vectors of the two planes. This implies that both inner products of the orientation vector of M and each respective normal vectors of these planes become 0. It is expressed with mathematical equations as

$$\begin{aligned}(p_1, q_1, -1)\cdot(\cos\alpha.\sin\alpha, \beta) = p_1\cos\alpha + q_1\sin\alpha - \beta = 0\\ (p_2, q_2, -1)\cdot(\cos\alpha.\sin\alpha, \beta) = p_2\cos\alpha + q_2\sin\alpha - \beta = 0\end{aligned} \tag{5.2}$$

Eliminating unknown variable β from these two, we have

$$(p_1 - p_2)\cos\alpha + (q_1 - q_2)\sin\alpha = 0 \tag{5.3}$$

which expresses that the vector $(\cos\alpha, \sin\alpha)$ and $(p_1 - p_2, q_1 - q_2)$ make right angles.

This $(\cos\alpha, \sin\alpha)$ is the orientation of the line L which is the image of the crossing line of planes, and $(p_1 - p_2, q_1 - q_2)$ is that of L′ which is passing through the gradients of planes. Therefore, in other words, as shown in Fig. 5.3(d), when we overlay the image plane and the gradient space, we have such the property that

A line which is an image of crossing line of two 3D planes meets at right angles with the line passing through both gradients of the planes.

Reversely, it also holds that

Given an image of crossing line of two 3D planes, both of the gradients of those planes locate on a line perpendicular to the given line in the gradient space.

Let us assume that two planes A and B are crossing in 3D space. As shown in Fig. 5.4, if we have an image of this crossing line parallel to y-axis, and if the gradient of the plane A on one side of this line is given as $G_A = (p_A, q_A)$, then the gradient $G_B = (p_B, q_B)$ of other side plane B must be on a line parallel to x-axis and passing through G_A. That is, $q_B = q_A$. Figure 5.4 shows possible relationships between G_A and G_B for some crossing angles of those planes.

Only from this relation, gradients of planes cannot be determined uniquely from the image. However, this fact is so powerful as will be discussed in the next sections.

5.2 Line Drawing Interpretation by Plane Gradients

The relation between the plane gradients and the image of crossing line, described up to here, is called the *Mackworth's Law*. This law is powerful also for interpretation of line drawings. In Fig. 5.5, an example of the interpretation by this law is shown.

We assume that an image of (a) is given and all of angles in the image between edge lines of crossing of two of the surface planes A, B, and C are 120 degree. Then, in the gradient space, as shown in (b), the gradient G_B must be located in the relative orientation to G_A as is perpendicular to the crossing line. If G_A and G_B are given keeping this relation, G_C is determined from G_A, G_B and the orientation of their crossing line as shown in the figure. As the results, G_A, G_B, and G_C form regular triangles in the gradient space.

The 3D spatial form of such the three surfaces is not unique. For example, in the figure (c), (1) is a cubic vertex whose three surface planes have right angles to each other. (2) is more sharp vertex, (3) is more obtuse one, and (4) is of a slanting vertex. (In the gradient space, it should be noted that the further from the origin means the steeper the plane is.) That is, all of these solid objects look as the figure (a), and we cannot know more about the shape of the vertex. But, this fact still limit a small range of variety of shapes.

We examine one more example presented by Mackworth in Fig. 5.6.

We assume to be given a line drawing of (a) and that the background plane A is parallel to the image plane. That is, the gradient of A is at the origin $(0, 0)$ in the gradient space. We also assume the surface B is crossing to A on the line 1. Then, the gradient of B locates, as shown in (b), on the line perpendicular to the line 1 and passing through G_A. If B further has crossing with the background plane A on the line 2, the gradient G_B must lie also on the line perpendicular to the line 2 and passing through G_A as shown in (c). But, these two are not compatible. Therefore, the line 2 is not the crossing of the planes A and B, and we have conclusion that the plane B floats above the plane A.

Moreover, the surface C has crossings with both of A and B on lines of 3 and 4, respectively. Then, the gradient G_C of C lies on both of line perpendicular to 3 and passing through G_B and line perpendicular to 4 and passing through G_A. This results in that its position in the gradient space is determines uniquely as shown in (d).

Here, the edges 2 and 5 in this figure are crossings of unseen planes with the surfaces B and C, respectively. If the number of the unseen planes is only one, it (named as plane D) must have the gradient lying on both lines perpendicular to 2 and 5, respectively, passing through G_B and G_C. It concludes that G_D will be determined uniquely from G_B and G_C.

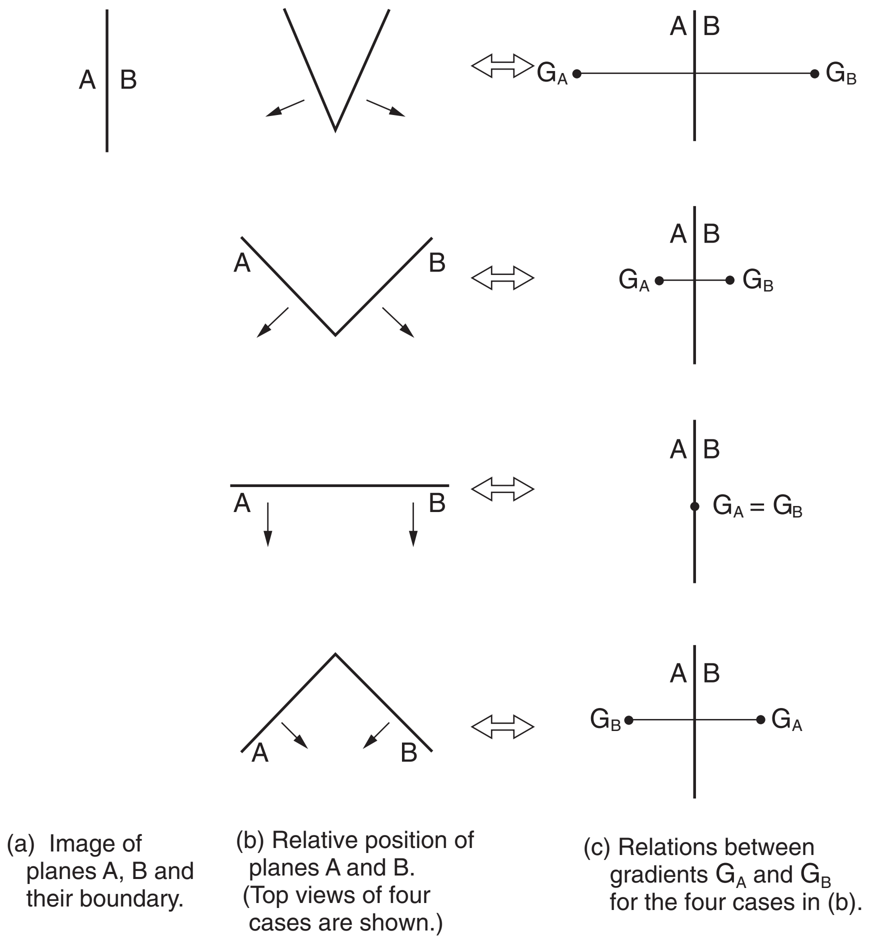

A
B
A
B
A
B
A
B
A
B
G_A
G_B
A
B
G_A
G_B
A
B
$G_A = G_B$
A
B
G_B
G_A
(a) Image of planes A, B and their boundary.
(b) Relative position of planes A and B. (Top views of four cases are shown.)
(c) Relations between gradients G_A and G_B for the four cases in (b).

Fig. 5.4. A crossing line of two planes, angles between those planes, and the relations of gradients of those planes corresponding to the crossing angles.

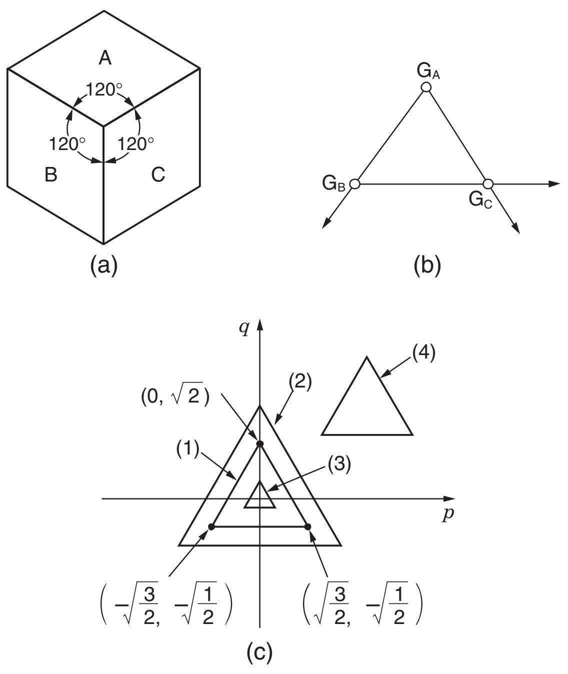
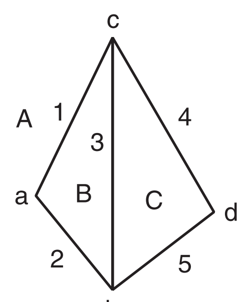
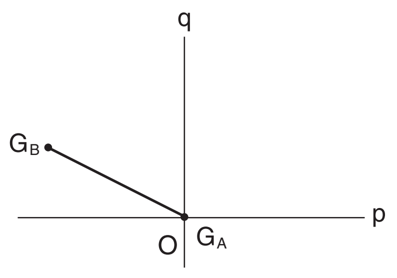

A
120°
120°
120°
B
C
(a)
G_A
G_B
G_C
(b)
q
p
$(0, \sqrt{2})$
(1)
(2)
(3)
(4)
$\left(-\sqrt{\frac{3}{2}}, -\sqrt{\frac{1}{2}}\right)$
$\left(\sqrt{\frac{3}{2}}, -\sqrt{\frac{1}{2}}\right)$
(c)

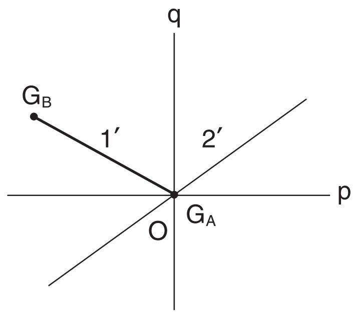
Fig. 5.5. Determination of gradients of planes by edge line orientations. From an image of (a), relations of gradients of planes are determined as (b), and (c) possible polyhedra having the sets of gradients of planes which satisfy the relations.



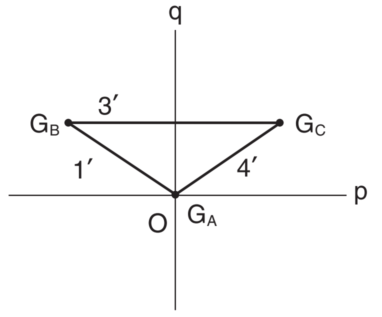
(a) A given line drawing. The gradient G_A of plane A is assumed to be (0, 0).



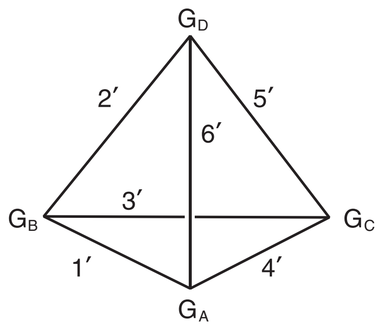
(b) Relative position of G_B to G_A.



(c) The relation between G_B and G_A shows that edge 2 is not the boundary of A and B.

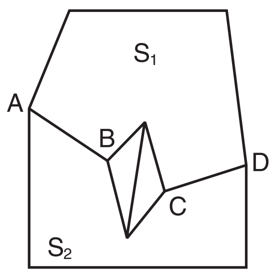


(d) Relative position of G_C to G_B and G_A.

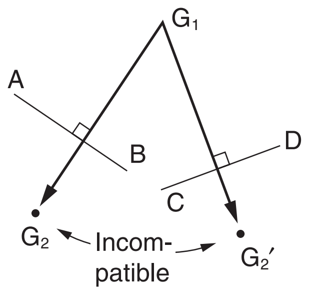


(e) Position of gradient G_D of unseen plane D.

Fig. 5.6. Interpretation of polyhedra by Mackworth.



(a)



(b)

Fig. 5.7. Polyhedron in (a) is an impossible one, because the position of the gradient G_2 cannot be determined.

Moreover, if such the G_D lies on the line passing through G_A and perpendicular to the image of the edge between a and d, then this unseen plane D has crossing with the plane A on the line passing through a and d (It is unseen.). It is proven that this object has only one plane D in its hidden side. That is, it is concluded that this image is of an object of a triangular pyramid protruding from the plane A, and one of whose side planes is hidden. It is amazing that the interpretation describes properties of unseen hidden planes.

In the previous chapter, we had an example of line drawing of an object for which the labeling in the interpretation with the Junction Catalog can be possible consistently, but actually its 3D shape cannot exist. The theory in this chapter excludes such an interpretation. It can be proven immediately that objects such as shown in Fig. 5.7, for example, cannot be interpreted as possible ones. In (a), denoting the gradients of the upper surface S_1 with G_1 and the side surface S_2 with G_2, because these two surfaces are crossing on the edge AB, the G_2 must lie in the gradient space at the point shown in (b) relatively to G_1. On the other hand, S_1 and S_2 are also crossing on the edge CD, and the gradient of S_2 must lie at G_2'. These two are gradient of a flat surface, and must be at the same point. So that, such a surface must not exist.

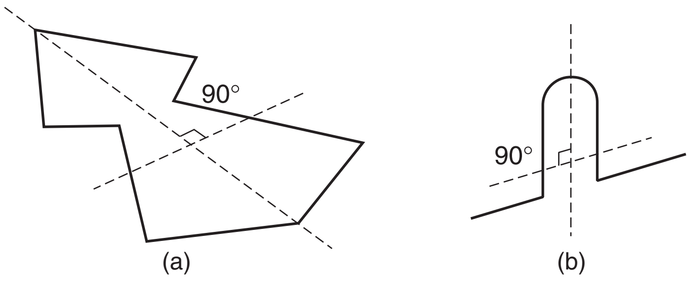


Fig. 5.8. Skew symmetry. Artificial objects usually have two perpendicular axes of symmetry in their surfaces.

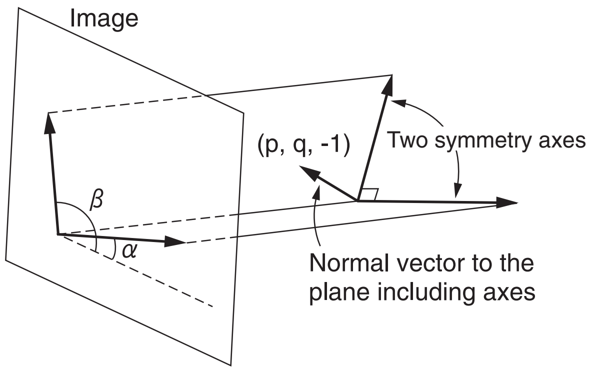


Fig. 5.9. Relation between spatial axes of symmetry and those in an image.

5.3 Skew Symmetry

Another line drawing interpretation based on the gradient space is the *skew symmetry* introduced by T. Kanade (1979) [14].

Almost all objects have some axes of symmetry in nature. Especially, artificial ones usually have one or more axes which must be perpendicular to the main axes. Such examples are shown in Fig. 5.8. Those two axes are perpendicular when they are seen in front of the objects, but from oblique direction it will be skewed.

Now, let us denote the gradient of the plane on which those two axes lie with $G_P = (p, q)$. If the two axes in the figure (a) have slant angles α and β, respectively, as shown in Fig. 5.9, then we can recover the spatial direction of this plane, that is $G_P = (p, q)$, up to some ambiguity from the angles of α and β. Actually, we can imagine the direction of the plane from the line drawing of Fig. 5.8.

These relations are mathematically formalized as follows.

In the image, directions of two axes are expressed with vectors of $(\cos\alpha, \sin\alpha)$ and $(\cos\beta, \sin\beta)$, respectively. As was explained, the vector which is projected to $(\cos\alpha, \sin\alpha)$ in the image has a form of $(\cos\alpha, \sin\alpha, \omega)$ in original 3D space, where ω is unknown value.

At the same time, a plane having a gradient $G_P = (p, q)$ has the normal vector of $(p, q, -1)$ in 3D space. The two skew symmetry axes lie on this plane, so that these two axes are perpendicular to the normal vector in 3D space. This means, first,

$$(p, q, -1) \cdot (\cos\alpha, \sin\alpha, \omega) = 0 \tag{5.4}$$

and we have

$$\omega = p\cos\alpha + q\sin\alpha \tag{5.5}$$

In the same way, for the axis having the angle β, its spatial direction is determined as $(\cos\beta, \sin\beta, p\cos\beta + q\sin\beta)$.

The two spatial vectors of the skew symmetry axes are also perpendicular to each other. That is,

$$\begin{aligned} \boldsymbol{a} &= (\cos\alpha, \sin\alpha, p\cos\alpha + q\sin\alpha) \\ \boldsymbol{b} &= (\cos\beta, \sin\beta, p\cos\beta + q\sin\beta) \end{aligned} \tag{5.6}$$

are perpendicular to each other, then

$$\boldsymbol{a} \cdot \boldsymbol{b} = 0$$

Rewriting this, we have

$$\cos(\alpha - \beta) + (p\cos\alpha + q\sin\alpha)(p\cos\beta + q\sin\beta) = 0 \tag{5.7}$$

This constrains the gradient (p, q) of the plane. To make it simple to understand, introducing a variable $\lambda = (\alpha + \beta)/2$ and rotating the p and q axes in the gradient space by an angle λ, we have new coordinates of

$$\begin{aligned} p' &= p\cos\lambda + q\sin\lambda \\ q' &= -p\sin\lambda + q\cos\lambda \end{aligned} \tag{5.8}$$

and the constraint of

$$(p')^2\cos(\gamma/2) - (q')^2\sin(\gamma/2) = -\cos\gamma \tag{5.9}$$

where $\gamma = \alpha - \beta$. This is a form of a hyperbolic curve in the (p, q) gradient space, shown in Fig. 5.10.

The discussion of the skew symmetry above can be applied as follows, for example. The object shown in Fig. 5.11(a) is an image of a cube. Each of three visible surfaces of this object have each respective pairs of symmetry axes shown with broken lines. Then, as was described above, the gradient G_1 of the surface 1 must lie on the hyperbolic curve S_1 as shown in the figure (b). Similarly, the gradients G_1 of the surface 2 and G_3 of the surface 3 also must lie on the hyperbolic curves S_2 and S_3, respectively.

On the other hand, we have already the relation between the gradients of two planes and their crossing edge, which says that the line passing through those two gradients and the edge are perpendicular to each other in the gradient space, the positions of G_1, G_2, and G_3 form a regular triangle.

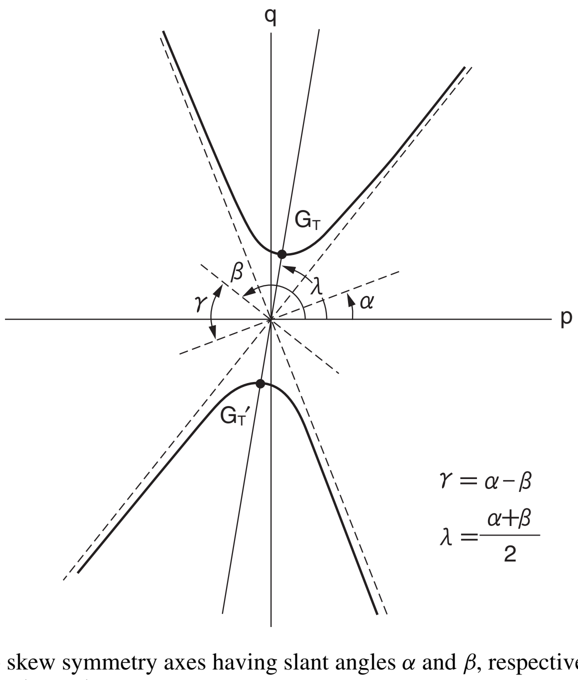


Fig. 5.10. When the images of two skew symmetry axes having slant angles α and β, respectively, the gradient of the plane lie on a hyperbolic curve in (p, q) space shown here.

The sets of G_1, G_2, and G_3 which satisfy these two constrains simultaneously are only two shown (b). One is just of the cubic which is originally assumed to be in (a). The other is of the concave dent on a wall. Both are actually possible for the interpretation of the line drawing of (a), and the gradients of their composing surfaces are determined uniquely for respective cases. That is, with the images of symmetric axes, we can completely determine actual 3D shapes only from their line drawing.

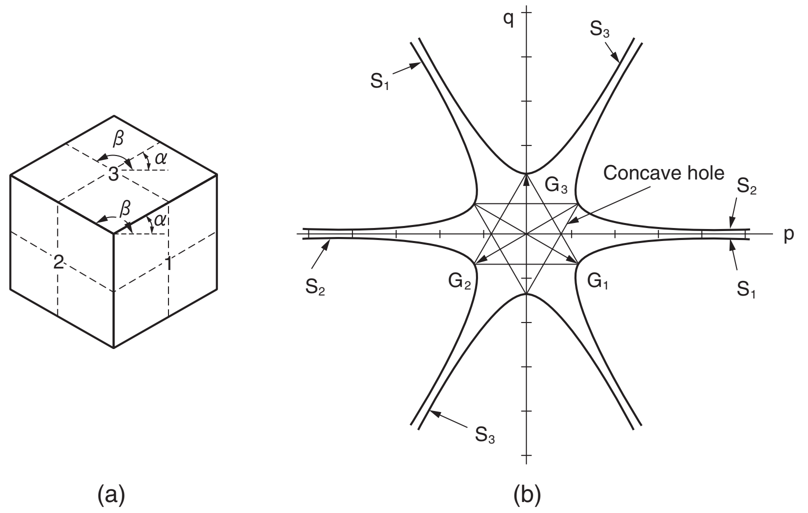


Fig. 5.11. Determination of spatial directions of object surfaces employing the skew symmetry concept. (a) When each of the surfaces 1, 2, and 3 have respect skew symmetry axes, (b) by the constraints from the skew symmetry and the relation between the surface crossing edge and their gradients. The gradients G_1, G_2, and G_3 of those three surfaces are determined.

As was seen here, applying a combination of some different constraints coming from different theories is common method to interpret image as will be shown also in the later chapters. Maybe in our perception, a consistent combination of fragments of local informations will be searched to make possible and correct interpretation for a given images.

6. Geometry of Image of Space (II)

6.1 Orthogonal and Perspective Projections

6.1.1 Camera optics of imaging

In a camera, an image of 3D space in front is formed as a projection through a lens onto a 2D plane. When the shape of an object in 3D space is given, its image is completely determined with the law of geometric optics. Our problem here is that we must reconstruct the law at the stand point to understand the 3D world from the image formations. In other words, we intend to equip full mechanism how the 3D spatial features are projected on a image.

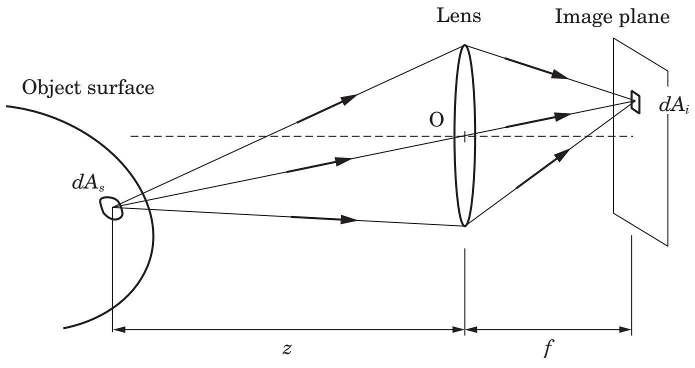


Fig. 6.1. Physical model of camera optics and imaging.

As shown in Fig. 6.1, light starting from an object surface reaches, through a lens, to the image plane, which consists of 2D arrangement of image sensors in general digital camera, and produces electric signals. A light bundle emitting or reflecting at a small region dA_s on the surface is concentrated by the lens into a pixel region dA_i on the image plane.

By simplifying this physical model and focusing on the positional relations between these two small regions of dA_s and dA_i, we have a model of the *pin-hole camera* (Fig. 6.2(a)). This name came from that these two points should lie on a line passing through the lens center, which is called the *optical center*, as shown in Fig. 6.1. This is equivalent to an actual pin-hole camera which substitutes the lens shown in Fig. 6.1 with a screen having a small pin hole at the center. It models only the geometrical relations between object points and their image points.

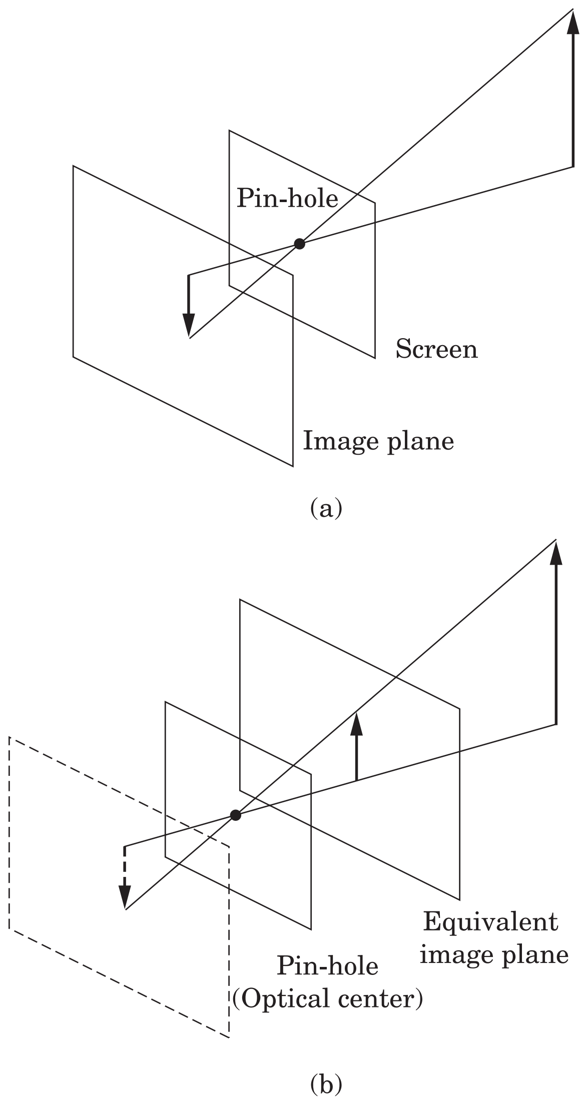


Fig. 6.2. Pin-hole camera and its central projection model.

A spatial line from an object point to its image point passing through the optical center is called a line of sight. In actual cameras, we have a series of combination lens and have not such a physically simple optics. But, we can always consider such an equivalent simple lens and its optical center.

6.1.2 Perspective transform

In the pin-hole camera modeled as in Fig. 6.2(a), images are formed up-side-down. So, it is convenient to convert the placement of the image plane as shown in Fig. 6.2(b). In this new model, the image of spatial objects is formed as seen from the optical center and perspectively through a transparent image plane. Therefore, this model is called the central projection or the *perspective projection*.

To formulate the imaging of the perspective projection, we introduce the 3D coordinates shown in Fig. 6.3, whose origin locates at the optical center. Its z-axis is normal to the image plane and called the *optical axis* of the camera. The x-y plane, therefore, will be parallel to the image plane. Such the system is called the *camera coordinates*. The length from the optical center O_c to the image plane is just the focal length of the lens. We denote it by f, then the image plane is expressed as $z = f$.

A spatial point at (x, y, z) in this coordinate system form an image at $(f\frac{x}{z}, f\frac{y}{z}, f)$. Then, we call the next essential transform of this imaging as the *perspective transform*:

$$(x, y, z) \longrightarrow \left(f\frac{x}{z}, f\frac{y}{z} \right) \tag{6.1}$$

where the coordinates in the left hand side are in the camera coordinates of a spatial point, and those in the right hand side represent the position of its image.

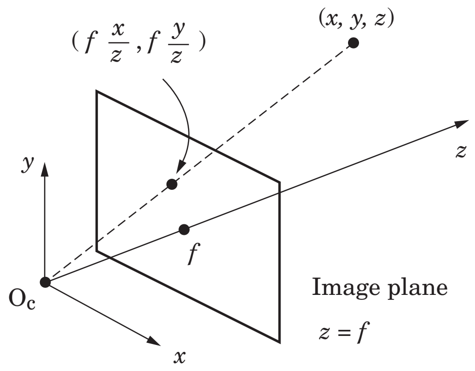


Fig. 6.3. 3D coordinates to expressing the camera image formation (Camera coordinates).

This transform looks simple at the first glance. However, many of interesting results are derived from this transform. This says that, typically, the size of image decreases inversely proportional to the distance to the object, which makes a distortion of the image shape of the object. This distortion, in turn, has a possibility to recover some informations about the 3D object shape from its image. We will discuss the utilization of this property to reconstruction of the original shape in later chapters. Here, we simplify the imaging model further, and discuss the fundamental aspects how the object shape reflect on its images.

6.1.3 Orthographic projection model

When an object is placed far from the camera, that is, the depth of the object in z direction is much smaller than its distance from the camera, then, the value of $1/z$ can be considered as constant on the object. The projection of a point (x, y, z) in this case can be expressed, denoting the image coordinates with (X, Y), as

$$\begin{aligned} X &= cx \\ Y &= cy \end{aligned} \tag{6.2}$$

where c is a constant determined by the optical system.

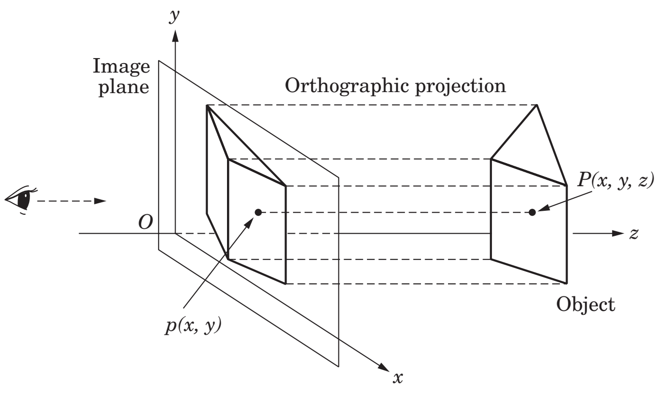


Fig. 6.4. Orthographic (or parallel) projection model.

This is nothing but the imaging model introduced in the previous chapter, and called the *orthographic projection*, or the *parallel projection*. Although already we showed the relation between an object and its image in Fig. 5.1, we show the schematics of the orthographic projection modeling in Fig. 6.4. It means that we ignore the perspectivity, but the structure of the object can be easily understood.

In this chapter, let us discuss how the 3D shape structures reflect on the image properties in the cases of the orthographic projection, and, inversely, how to utilize such the image properties for the extraction of 3D properties of an object. After that, we will extend and modify the theories for the case of the perspective projection.

6.2 Planes and Their Gradients: Revisited

6.2.1 Expression of the slant of a plane

Let us revisit the theory of the gradients of planes and their images by the orthographic projection. In the orthographic projection of (6.2), the value of c is not important for the analysis and set to 1 for the simplicity. Then, a point $P(x, y, z)$ in 3D space is projected as

$$\begin{aligned} X &= x \\ Y &= y \\ Z &= f \end{aligned} \tag{6.3}$$

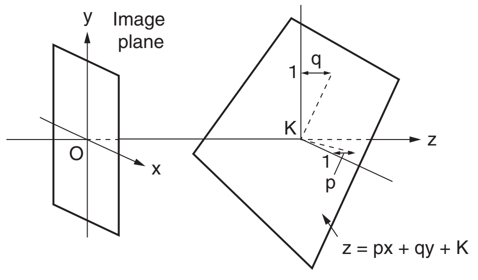
where this Z represents the position of the image plane. Because the image is irrespective to the position of the image plane, this value f does not make sense.

In the previous chapter, we introduced a value of the gradient, which represent the slanting amount of a plane with respect to our line of sight, as shown in Fig. 6.5(a).

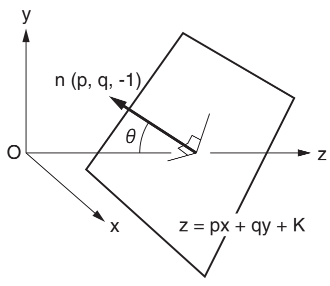
That is, when we express a spatial plane in the camera coordinates as

$$z = px + qy + K \tag{6.4}$$

we defined the pair of (p, q) as the gradient of the plane. The vector $\boldsymbol{n} = (p, q, -1)$ constructed with this gradient is normal to the plane in the 3D space, as shown in the figure (b).



(a) A spatial plane and its gradient (p, q)



(b) Vector $n(p, q, -1)$ is perpendicular to the plane $z = px + qy + K$, and called the normal vector

Fig. 6.5. Representation of slanting amount of a plane.

Denoting the angle between our line of sight, whose direction is $(0, 0, -1)$, and $\boldsymbol{n}$ with θ, then we have

$$\cos\theta = \frac{1}{\sqrt{p^2 + q^2 + 1}} \tag{6.5}$$

We have other representations of the slant of spatial plane.

A commonly used one is to represent how much angle the plane inclines toward a specific direction. This direction is represented with an angle and called a *tilt* and the inclining angle is called a *slant*. Actually, imagine an initial plane parallel to the image plane and it inclines around an axis on the plane. The direction angle of this axis is just the tilt. When it is of the direction of x-axis, the angle is set to 0, and the direction of this axis is measured in a counter-clockwise with an angle of τ. The angle of the slant of the plane around this axis is measured also in a counter-clockwise with an angle of σ, whose value is 0 when the plane is also parallel to the image plane. The relation between these (τ, σ) and the gradient (p, q) is shown in Fig. 6.6.

The other method to represent the inclination of a spatial plane is: First, we set an axis in the plane according to the pose of the figures in the plane. Then, we have an angle of this axis to the image plane. The second angle represents how much the plane inclines around this angle. This representation is shown in Fig. 6.7. For example, we assume a planar curve in the space as shown in 6.8(a). With this curve, the length L between both end points of P and Q and the height H from the line of PQ are given. The image of the curve with the orthographic projection will be shown in (b).

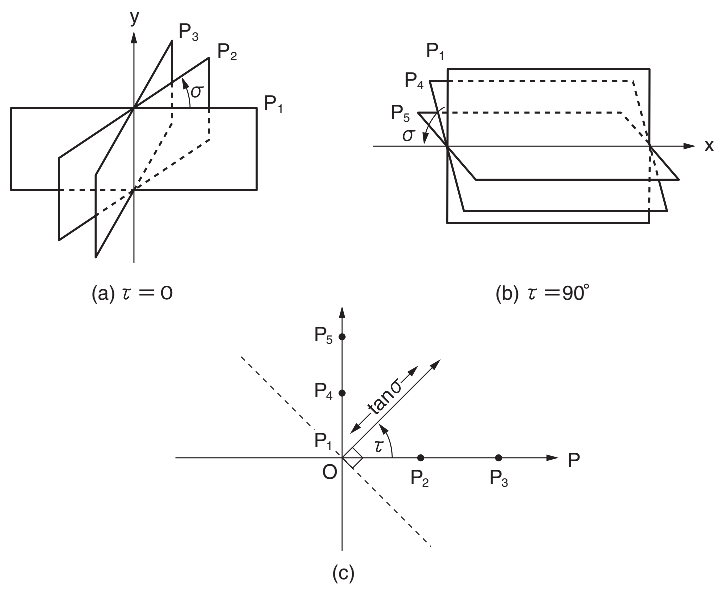


Fig. 6.6. Representation of an inclination of a spatial plane with the tilt and slant. Planes shown in (a) and (b) having tilts τ and slants σ have the gradients of (p, q) shown in (c).

We derive the 3D pose of this curve, that is just the inclination of the plane on which the curve lies. For this purpose, the third representation is convenient.

We select the line PQ as the axis on the plane on which the curve lies. Let the angle between PQ and the image plane be θ, and the rotation angle of this plane around this axis be ϕ. Then, the length L and the height H are simply projected to the length l and height h on the image plane as

$$\begin{aligned} l &= L\cos\theta \\ h &= H\sin\phi \end{aligned} \tag{6.6}$$

If we know the values of L and H, we obtain (θ, ϕ) from the equation of (6.6).

Figure 6.9 shows a synthetic image of a house-plant composed with a simple model. Each curves are leaves. The flower-pod is omitted from the image. The leaves come out at the edge of the round flower-pod. The shapes of all leaves are of the same planar curve.

Then, from each of the length l between two end point of the leaf and the height h on the image, its spatial pose can be determined. That is, the spatial poses of each leaves and spatial arrangements of all the leaves can be determined from their image of Fig. 6.9. Figure 6.10 shows the production of a top view which should be seen as is of the house plant from those spatial poses of the leaves.

When the direction of the axis is set with the angle ω to the x axis, the relation between the angles of (θ, ϕ) and the gradient (p, q) will be

$$\begin{aligned} \tan\theta &= p\cos\omega + q\sin\omega \\ \tan\phi &= -p\sin\omega + q\cos\omega \end{aligned} \tag{6.7}$$

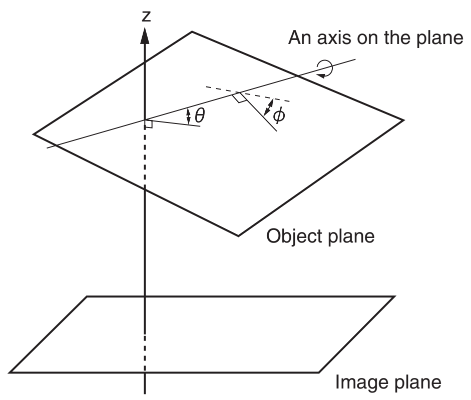


Fig. 6.7. By selecting an axis on the plane with respect to the pose of figure on it, the amount of the inclination angle of the plane can be represented.

6.2.2 Angles on a spatial plane

In the previous chapter, the gradient space, which is spanned by the gradient (p, q), was introduced, and the Mackworth's law was derived as its application. Here, we discuss another interpretation of an image by employing a spatial angle and its projection on the image.

Now, we assume spatial lines crossing with an angle A as shown in Fig. 6.11, where the value of A is known. On its image, the images of these two lines are crossing with an angle α. In other words, the image projection of an angle A is just the angle α.

The relation between A and α is determined by the inclination of the plane on which the angle of A lies. So, let us derive the inclination of the plane by reading the image angle of α.

Let us assume that we are given images of a horse, whose body, front legs, and head are known to be on a plane and the angles between them are 90 degree and 45 degree, respectively, as shown in Fig. 6.12. Those images are (a), (b), and (c), for example. We consider the possible spatial directions of his body, represented with a vector V, for these images.

We express the inclination of the plane including this horse with (θ, ϕ) of the third method described in the previous section. From the property of the orthographic projection, about the V, only its angle to the depth direction is unknown. This angle is defined as shown in Fig. 6.13. This plane is assumed to be rotated around the axis of V by ϕ.

Then, an angle A on the plane is projected on the image as an angle α, which is given as

$$\tan\alpha = \frac{\sin\phi\tan A}{\cos\theta + \cos\phi\tan A} \tag{6.8}$$

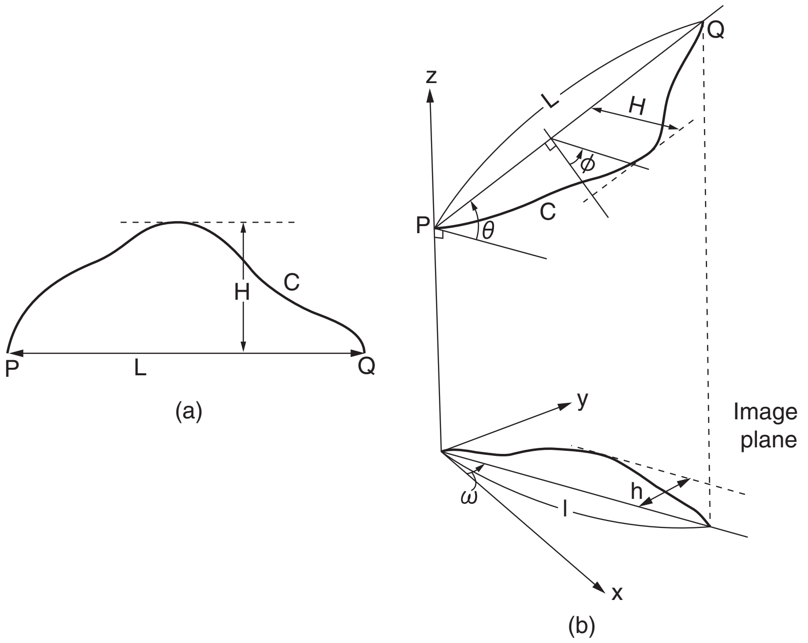


Fig. 6.8. (a) A planar curve C. (b) The projection of the curve C onto the x-y image plane.

Because the front legs and head of the horse, whose spatial angles to body direction V are 90 and 45 degrees, respectively, have respective angles in the image as shown in Fig. 6.12(a), the pose of this horse has restrictions shown in Fig. 6.14(a) and (b). The plot shown in this figure is the area allowing up to 5 degree error for the reading of the image. If we are given a value of the angle θ of the direction of V, the possible rotation angle of the plane including horse's body and front legs around V to obtain the image will be calculated by (6.8). This possible rotation angle is nothing but the ϕ. The possible pair of (θ, ϕ) is not unique and shown in the figure with respect to θ.

Similarly, in the Figure (b), the possible pair of (θ, ϕ) to obtain the given image angle between the body and head, which is 45 degree in the space, is also plotted with the same allowance of the reading error.

The body, the front legs, and the head lie on a plane, so that the pose of the horse should satisfy both of the restrictions. That is, the pair (θ, ϕ) in the common area of (a) and (b) is the totally possible one pose of the horse. This area is shown in (c). In other words, the horse should not have a pose other than these possible ones.

Figures 6.14(d), (e), and (f) also show the possible pose of the horse when the image of Fig. 6.12(b) is given, and (g), (h), and (i) are for the image of (c).

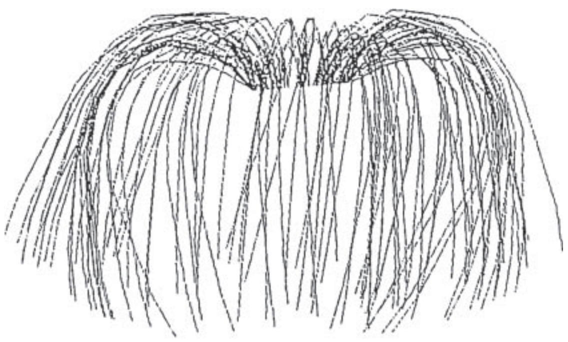


Fig. 6.9. A synthetic image of a house plant (a foliage plant in a flower-pod). All the leaves have same shape with different spatial poses.

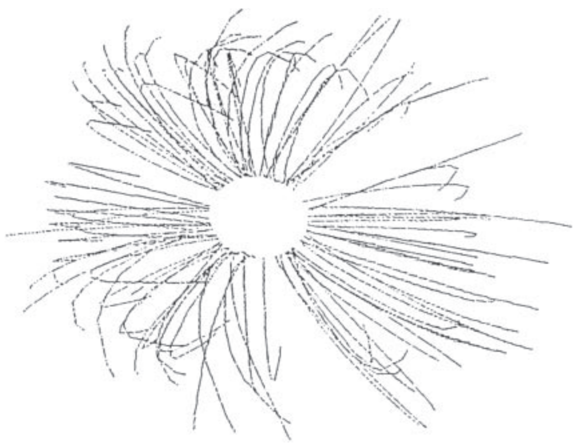


Fig. 6.10. The production of a top view which should be seen as is of the house plant from those spatial poses of the leaves derived from the side view of Fig. 6.9.

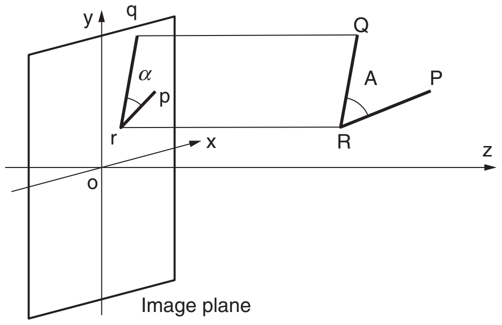


Fig. 6.11. A spatial angle and its image by the orthographic projection.

As shown here, from the relation between spatial angle and its image angle introduces restriction on the spatial pose of the object. Of course, it does not always lead to a unique solution, but usually it may take place strong constraint.

6.2.3 Gradients and projection of angles

We formulate the constraints of the projection of angles in connection with the gradients of planes. Let the gradient of a plane including $\angle SOT$ of angle A be (p, q), as shown in Fig. 6.15. The normal vector of this plane is $\boldsymbol{n} = (p, q, -1)$. We assume that this angle A is projected on an image as $\angle sot$ by the orthographic projection as shown in the figure.

The angle of os to x-axis is denoted with β and the angle of ot with γ. Then, we have $\alpha = \gamma - \beta$.

Now, setting lengths of os and ot to be 1, for the simplicity, the vectors os and ot are given, respectively, as

$$\begin{aligned} \vec{\mathrm{os}} &= (\cos\beta, \sin\beta, 0) \\ \vec{\mathrm{ot}} &= (\cos\gamma, \sin\gamma, 0) \end{aligned} \tag{6.9}$$

Because the orthographic projections of $\vec{\mathrm{OS}}$ and $\vec{\mathrm{OT}}$ are $\vec{\mathrm{os}}$ and $\vec{\mathrm{ot}}$, respectively, we have

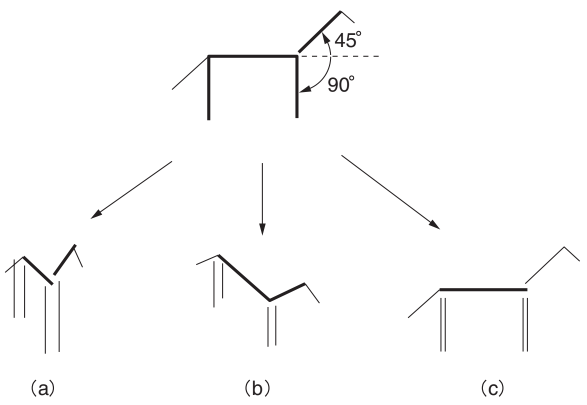


Fig. 6.12. Images of a "Horse." When the horse looks as (a), (b), and (c), for example, its respective possible poses are limited, and they can be estimated only from these images.

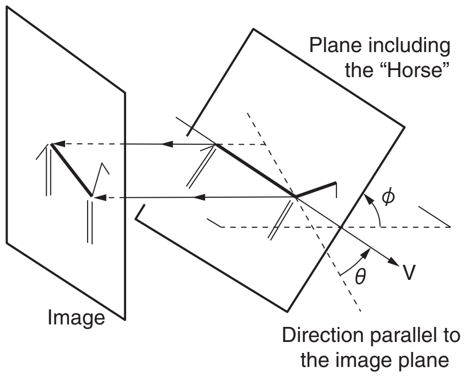

Plane including the "Horse"
ϕ
θ
V
Image
Direction parallel to the image plane

Fig. 6.13. Spatial parameters to determine the pose of the "Horse."

$$\begin{aligned}\vec{\mathrm{OS}} &= (\cos\beta, \sin\beta, \omega_s)\\ \vec{\mathrm{OT}} &= (\cos\gamma, \sin\gamma, \omega_t)\end{aligned} \tag{6.10}$$

Here, values of ω_s and ω_t are unknown, but, from the fact that both of the vectors $\vec{\mathrm{OS}}$ and $\vec{\mathrm{OT}}$ are perpendicular to $\boldsymbol{n}$, we also have

$$\begin{aligned}\omega_s &= p\cos\beta + q\sin\beta\\ \omega_t &= p\cos\gamma + q\sin\gamma\end{aligned}$$

From $\angle SOT = A$, we have one more equation

$$\vec{\mathrm{OS}}\cdot\vec{\mathrm{OT}} = |\vec{\mathrm{OS}}|\,|\vec{\mathrm{OT}}|\cos A, \tag{6.11}$$

that is,

$$\cos\beta\cos\gamma + \sin\beta\sin\gamma + \omega_s\omega_t = \sqrt{1+\omega_s^2}\cdot\sqrt{1+\omega_t^2}\cdot\cos A \tag{6.12}$$

On the other hand, from $\angle sot = \alpha$ and $\cos\beta\cos\gamma + \sin\beta\sin\gamma = \cos\alpha$, next relation holds that

$$\cos\alpha + \omega_s\omega_t = \sqrt{1+\omega_s^2}\cdot\sqrt{1+\omega_t^2}\cdot\cos A \tag{6.13}$$

This is the final relation between the spatial angle A and its orthographic projection image of α.

When $A = 90$ degree, the right hand side of (6.13) becomes 0, and

$$\cos\alpha + \omega_s\omega_t = \cos(\beta-\gamma) + (p\cos\beta + q\sin\beta)(p\cos\gamma + q\sin\gamma) = 0 \tag{6.14}$$

which is the same equation as (5.7). This is just the skew symmetry discussed at the end of the previous chapter.

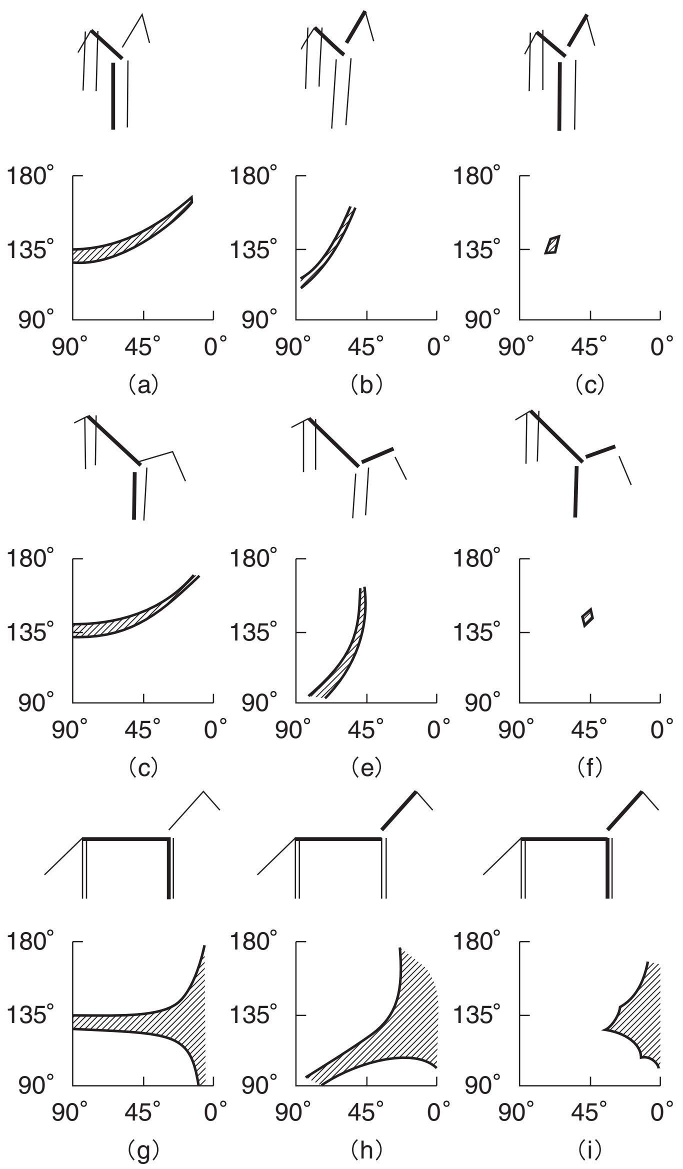


Fig. 6.14. When the "Horse" looks as the top figures in top images, which are the same one shown in Fig. 6.12(a), from the relation between its body and front legs, we have the possible pose (θ, ϕ) is restricted within the region shown in (a), where allowance of up to 5 degree of reading error is included. At the same time, from the relation between its body and head, the region is restricted as shown in (b). Then, commonly possible pose is restricted as (c). From the figure in middle, which is just the Fig. 6.12(b), we have also the pose restricted as shown in (d), (e), and (f). From the bottom figure, which is also just the Fig. 6.12(c), we have (g), (h), and (i). As shown in these figures, the possible poses of the "Horse" are restricted within respective small regions for all cases.

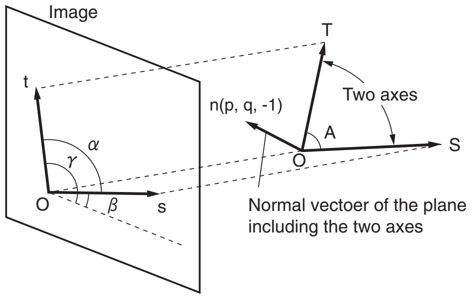


Fig. 6.15. Two axes in the space, angle between them, and their image projections.

Part III. Image Matching

7. Perspective Projection Images

7.1 Projections of Line and Plane and Vanishing Point and Vanishing Line

We examined imaging system by simplifying the projection optics, so far, as the orthographic projection. Now, we employ a more realistic model of the *perspective projection* shown in Fig. 7.1, and discuss its properties. Here, we describe some important properties of the relation between object 3D shapes and their projection images.

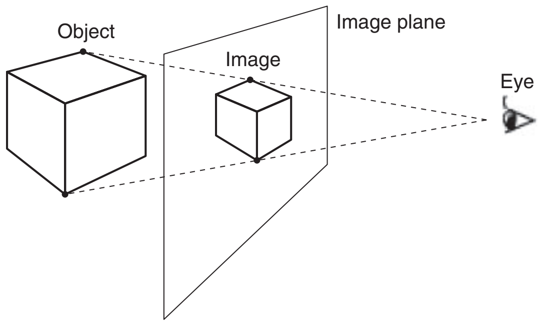


Fig. 7.1. Perspective projection.

In the simple orthographic projection, the distance to the object does not affect on its image. On the other hand, in the perspective projection, the image is deformed depending on the distance even if it is seen from the same direction, as is shown in Fig. 7.2.

Let us recall and show the imaging system of the perspective projection in Fig. 7.3, which was already shown in Chapter 6. In the coordinate system to describe the perspective projection, we denote by O the origin of coordinate system, which is the viewing point. The image plane is $z = f$ where f is the focal length of the lens.

By the perspective projection, a spatial point $\mathrm{P}(x, y, z)$ will be projected on the image at

$$\begin{aligned} X &= f\frac{x}{z} \\ Y &= f\frac{y}{z} \end{aligned} \tag{7.1}$$

This transform of the coordinates generates the vanishing point and the vanishing line as shown in Fig. 7.4. This is the most important property of the perspective projection.

The vanishing points are generated as follows. Consider a line in 3D space which passes the point $\mathrm{P}_0(x_0, y_0, z_0)$ and has the direction presented by the vector $\Delta\boldsymbol{t} = (\Delta x, \Delta y, \Delta z)$. Positional vector $\boldsymbol{P}_t$ of a point P_t on this line is expressed with a parameter t as

$$\boldsymbol{P}_t = (x_0, y_0, z_0) + t \cdot (\Delta x, \Delta y, \Delta z) \tag{7.2}$$

see Fig. 7.5. Then, its projection $\boldsymbol{p}_t$ on the image is given by (7.1) as

$$\boldsymbol{p}_t = \left(f\frac{x_0 + t\Delta x}{z_0 + t\Delta z},\ f\frac{y_0 + t\Delta y}{z_0 + t\Delta z} \right) \tag{7.3}$$

Here, the point $\boldsymbol{P}_\infty$ with $t \to \infty$ is the point at infinity on this spatial line, and its image point $\boldsymbol{p}_\infty$ is just the vanishing point. Therefore, we have

$$\boldsymbol{p}_\infty = \lim_{t\to\infty} \boldsymbol{p}_t = \lim_{t\to\infty} \left(f\frac{x_0/t + \Delta x}{z_0/t + \Delta z},\ f\frac{y_0/t + \Delta y}{z_0/t + \Delta z} \right) = \left(f\frac{\Delta x}{\Delta z},\ f\frac{\Delta y}{\Delta z} \right) \tag{7.4}$$

This gives the image coordinate of the vanishing point of the line of (7.2).

It should be noted, first, that the position of the vanishing point $\boldsymbol{p}_\infty$ does not depend on the 3D coordinates of $\mathrm{P}_0(x_0, y_0, z_0)$ which is a point on the spatial line. That is, it is determined only by the direction $\Delta\boldsymbol{t} = (\Delta x, \Delta y, \Delta z)$ of the spatial line independent of its position. The point $\boldsymbol{p}_\infty$ of (7.4) is common for all the spatial lines having the same direction $\Delta\boldsymbol{t} = (\Delta x, \Delta y, \Delta z)$ regardless of their positions. The lines having the same direction are parallel lines. So that, in other words, *parallel lines vanish at one point.*

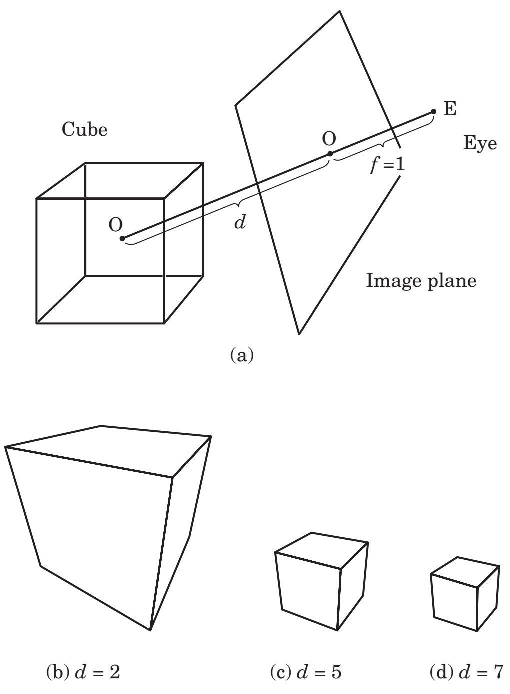


Fig. 7.2. Difference of appearance of image of a object depending on its distance. Images of a cubic object of (a) at the distance of (b) $d = 2$, (c) $d = 5$, and (d) $d = 7$, for $f = 1$.

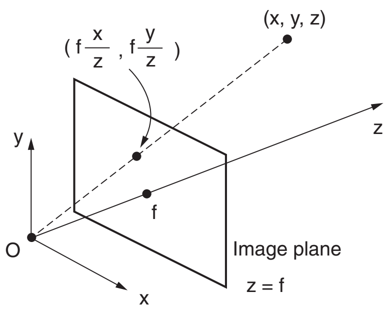


Fig. 7.3. Optical imaging system of perspective projection.

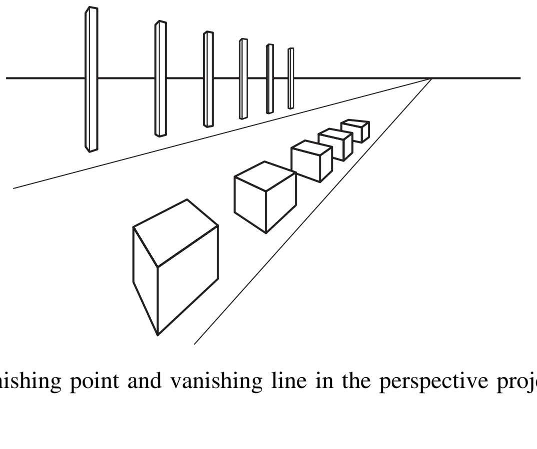


Fig. 7.4. Vanishing point and vanishing line in the perspective projection image.

Furthermore, (7.4) suggests one more important property of the vanishing point. Inversely, from the position of the vanishing point, the spatial direction of the original line can be recovered.

Next we discuss on the vanishing lines and their formulations. As shown in Fig. 7.6, we denote the gradient of a spatial plane S with $\boldsymbol{G}_S = (p_S, q_S)$. The normal vector of this plane is given as $\boldsymbol{n}_S = (p_S, q_S, -1)$.

Now, we consider a line l lying on this plane and having a direction of $\Delta\boldsymbol{t} = (\Delta x, \Delta y, \Delta z)$. This direction vector $\Delta\boldsymbol{t}$ is perpendicular to $\boldsymbol{n}_S$. Then,

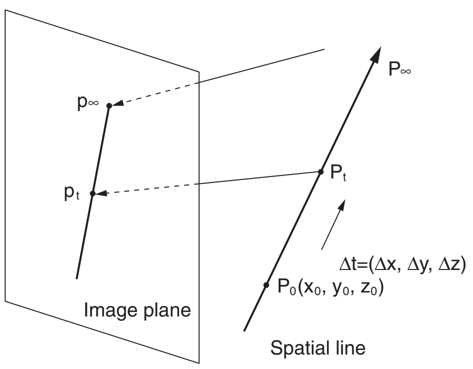


Fig. 7.5. A spatial line and its image. The image of the point P_∞ at the infinity on the spatial line is just the vanishing point p_∞.

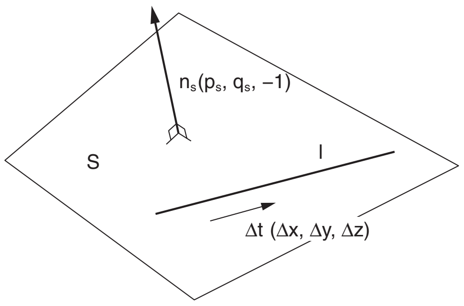


Fig. 7.6. A spatial plane S, its normal vector $\boldsymbol{n}_S$, and a line l on this plane.

$$\boldsymbol{n}_S \cdot \Delta \boldsymbol{t} = p_S \Delta x + q_S \Delta y - \Delta z = 0, \tag{7.5}$$

and we have

$$p_S \frac{\Delta x}{\Delta z} + q_S \frac{\Delta y}{\Delta z} = 1 \tag{7.6}$$

On the other hand, from (7.4), $(f\Delta x/\Delta z, f\Delta y/\Delta z)$ is just the vanishing point of this line. It means that equation (7.6) can be rewritten as

$$\boldsymbol{G}_S \cdot \boldsymbol{p}_\infty = f \tag{7.7}$$

Because the vanishing line is the set of all the vanishing points of lines on the plane, all the points of $\boldsymbol{p}_\infty$ which satisfy (7.7) will form the vanishing line.

Drawing this line, we have next configuration: Given $\boldsymbol{G}_S = (p_S, q_S)$ and setting

$$r = \frac{f}{|\boldsymbol{G}_S|} = \frac{f}{\sqrt{p_S^2 + q_S^2}} \tag{7.8}$$

as shown in Fig. 7.7, on the image plane, $\boldsymbol{p}_\infty$ locates on the line which is crossing perpendicularly with the line passing the origin O and $\boldsymbol{G}_S$ at the distance of r from O. It means that this line is nothing but the vanishing line.

Conversely, we can derive the gradient of a plane from its vanishing line. When we are given a line

$$ax + by = c \tag{7.9}$$

on the image as a vanishing line, the gradient of the plane whose vanishing line becomes this line will be

$$\boldsymbol{G}_S = \left(p_S = f\frac{a}{c},\ q_S = f\frac{b}{c} \right) \tag{7.10}$$

Another important property of the vanishing line is, similarly to the vanishing point, that it is determined only by the gradient of the plane and independent of its position. Planes which have same gradient, those are parallel planes, generate the same vanishing line.

The vanishing line is also obtained by giving two sets of parallel lines on a plane other than from its gradient. They generate two respective vanishing points, and the line connecting those points will be the vanishing line. Denoting two respective directions of those sets of parallel lines with $\Delta \boldsymbol{t}_1$ and $\Delta \boldsymbol{t}_2$, respectively, the gradient of plane on which those parallel lines lie is given by

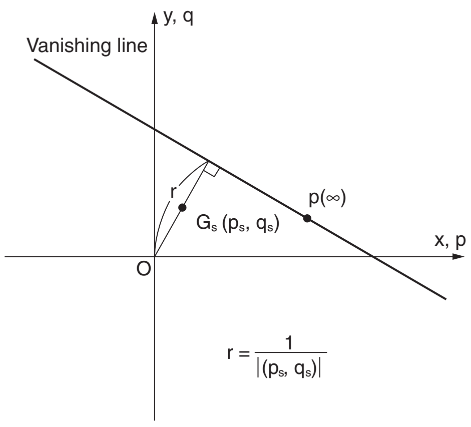


Fig. 7.7. Relation between the gradient $\boldsymbol{G}_S$ of a plane and its vanishing line.

$$(p_S, q_S, -1) = c\Delta \boldsymbol{t}_1 \times \Delta \boldsymbol{t}_2 \tag{7.11}$$

because the normal vector of the plane is perpendicular to both of $\Delta \boldsymbol{t}_1$ and $\Delta \boldsymbol{t}_2$. (In this equation, c is a constant to make the third component of the right hand side vector be -1, and $\times$ means the outer product.) Then, by combining the equations from (7.7) to (7.10), we obtain the vanishing line from $\Delta \boldsymbol{t}_1$ and $\Delta \boldsymbol{t}_2$.

7.2 Determination of the Slant of a Plane

From deformations of a planar figure in its perspective image, the slant of the spatial plane on which the figure lies can be determined. This is the same as the case of the orthogonal projection. However, this time more versatile and sophisticated properties of the deformations can be available for the determination.

In the previous section, we described that

1. the vanishing points and the vanishing lines are specially appear in the perspective images, and they express spatial directions of the lines and planes projected on the image, and
2. all the properties concerning on their directions can be derived from the positions of vanishing points and vanishing lines.

That is, the determination of the slant of a spatial plane from figures lying on it is equivalent to the direct or indirect determination of the position of the vanishing line of the plane in the image.

For example, because an image of a planar grid pattern shown in Fig. 7.8 consists of two sets of parallel lines, two vanishing points of those respective sets of lines are shown, and a line passing through those points are determined on the image. This line is nothing but the vanishing line of the plane on which the grid pattern lies.

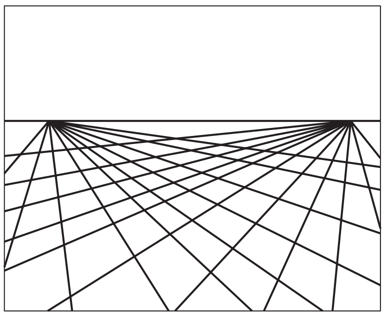


Fig. 7.8. From an image of a grid pattern lying on a spatial plane, the vanishing points and the vanishing line can be determined, and then, the slant of the plane can be determined.

This story holds in the general cases where we have two sets of parallel lines lying on a plane. It means that, if we have an image of a parallelogram on a spatial plane, we can determine the slant of the plane as shown in Fig. 7.9. We denote the coordinates of the four vertices of the parallelogram on the image with (x_i, y_i) $(i = 1, \cdots, 4)$ as shown. If we express the line passing through (x_1, y_1) and (x_2, y_2) with $a_1x + b_1y = c_1$, then we have

$$a_1 = y_1 - y_2, \ b_1 = -(x_1 - x_2), \ c_1 = x_2y_1 - x_1y_2 \tag{7.12}$$

Similarly, if we express the line passing through (x_3, y_3) and (x_4, y_4) with $a_2x + b_2y = c_2$, we have

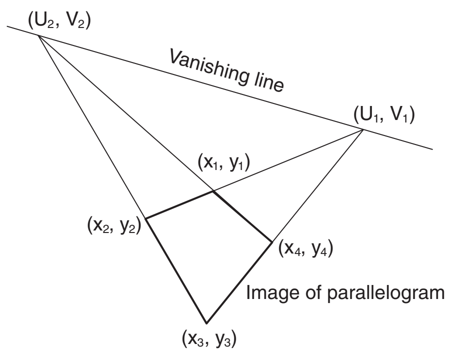


Fig. 7.9. From an image of a parallelogram, the slant of plane on which the figure is lying can be determined.

$$a_2 = y_3 - y_4,\ b_2 = -(x_3 - x_4),\ c_2 = x_4y_3 - x_3y_4 \tag{7.13}$$

The crossing point of these two lines is just the vanishing point of these two spatially parallel lines. Let us denote this point with (U_1, V_1), then

$$U_1 = \frac{b_2c_1 - b_1c_2}{a_1b_2 - a_2b_1}, \quad V_1 = -\frac{a_2c_1 - a_1c_2}{a_1b_2 - a_2b_1} \tag{7.14}$$

We have one more vanishing point (U_2, V_2) from another set of parallel edges similarly given by

$$U_2 = \frac{b_4c_3 - b_3c_4}{a_3b_4 - a_4b_3}, \quad V_2 = -\frac{a_4c_3 - a_3c_4}{a_3b_4 - a_4b_3} \tag{7.15}$$

where

$$\begin{aligned} a_3 &= y_1 - y_4, \quad b_3 = -(x_1 - x_4), \quad c_3 = x_4y_1 - x_1y_4 \\ a_4 &= y_2 - y_3, \quad b_4 = -(x_2 - x_3), \quad c_4 = x_3y_2 - x_2y_3 \end{aligned} \tag{7.16}$$

The vanishing line of the spatial plane on which the parallelogram lies is given as the line connecting these (U_1, V_1) and (U_2, V_2) as

$$(V_1 - V_2)x - (U_1 - U_2)y = U_2V_1 - U_1V_2 \tag{7.17}$$

Therefore, from the discussion in the previous section, the spatial slant of a plane on which a parallelogram having images of four vertices at (x_i, y_i) $(i = 1, \cdots, 4)$ lies is given as

$$p_S = f\frac{-V_1 + V_2}{U_1V_2 - U_2V_1}, \quad q_S = f\frac{U_1 + U_2}{U_1V_2 - U_2V_1} \tag{7.18}$$

7.3 Slant of Plane and Scale Reduction of a Pattern

In the previous chapters, we have described about the tilt angle τ of a spatial plane. It is the angle of the direction in which the plane has the largest slant. So that, for the image shown in Fig. 7.10, if the pattern can be assumed to distribute uniformly on the plane, the tilt angle indicates the direction in which the density of pattern on the image is mostly changing. As shown in the figure, equivalently, the tilt angle τ is given as is perpendicular to the direction in which the image pattern density is mostly uniform.

Figure 7.11 shows the cross section plane which includes the view point O and the tilt angle direction and perpendicular to the image plane. Here, the image plane is parallel to the xy-plane as is in the descriptions so far. Then, this cross section includes z-axis, which is the line of sight, as shown in the figure. The crossing point of z-axis and the image plane is at $(0, 0, f)$, where f is the focal length, and this point is just the center of image plane. Let us denote this point with I_0. The image of an object point P on this plane π locates at the crossing point P′ of the line OP and the image plane $z = f$ as shown in the figure. The angle between the object plane π and the image plane on this cross section is the slant angle σ.

As was previously described, a vanishing line is just the crossing line between the image plane and a plane passing the viewpoint O and being parallel to the object plane π. Therefore, in this cross section, the vanishing line locates at the point V_L. The angle between OV_L and the image plane is also σ. Then, we have

$$\mathrm{I}_0\mathrm{V}_L = f/\tan\sigma \tag{7.19}$$

This tells that the distance from the image center to the vanishing line on the image plane is given as the right hand side of this equation. In the first section of this chapter, we derived that, when the gradient of the plane π is (p_S, q_S), this distance was given as $f/|(p_S, q_S)|$. Then, from the relation between the gradient shown in Fig. 6.6 and σ, this (7.19) also can be obtained.

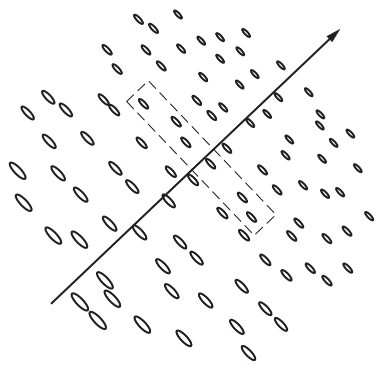


Fig. 7.10. When a pattern distributes uniformly on a spatially slant plane, its tilt angle is given for the direction in which the change of image pattern density is maximal.

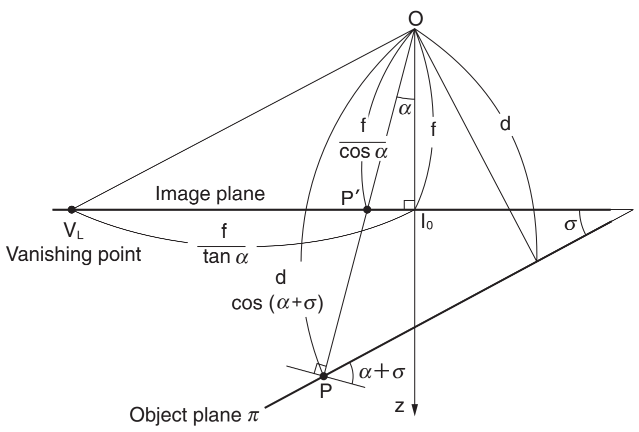


Fig. 7.11. The image projection of a point on a spatial plane described on the cross section which includes the viewing point and is perpendicular to the tilt direction and image plane. The figure shows parameters for the description of the projection.

From this figure, we can read out the shrinkage ratios of lengths at the point P on the object plane π and its image at P′ on the image plane. These ratios are expressed as a function of the angle α between OI_0 and OP as

$$\text{Ratio in the parallel direction to the vanishing line: } w(\alpha) = \frac{f}{d} \cdot \frac{\cos(\alpha + \sigma)}{\cos\alpha}$$

$$\text{Ratio in the vertical direction to the vanishing line: } h(\alpha) = \frac{f}{d} \cdot \left(\frac{\cos(\alpha + \sigma)}{\cos\alpha}\right)^2$$

where d is the distance from O to the object plane π. Then the shrinkage ratio of areas is given with the multiplication of them as

$$s(\alpha) = \frac{f^2}{d^2} \cdot \left(\frac{\cos(\alpha + \sigma)}{\cos\alpha}\right)^3$$

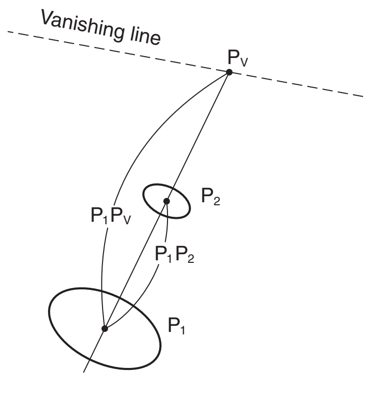


Fig. 7.12. Relation between points and a vanishing line.

Because the spatial distance d cannot be read out only from images, we cannot apply this relation directly to the scene analysis. But, we can utilize this relation by considering the shrinkage ratio of areas in the image. As shown in Fig. 7.12, when we denote the ratios at the two points of P_1 and P_2 with s_1 and s_2, respectively, because the shrinkage ratio at the vanishing point P_V of the line connecting P_1 and P_2 on the image becomes 0, we have

$$P_1P_V = \frac{s_1^{1/3}}{s_1^{1/3} - s_2^{1/3}} \cdot P_1P_2 \tag{7.20}$$

If there exist two figures at P_1 and P_2, whose original areas on the spatial plane π are the same, s_1 and s_2 in the above equation can be replaced with the respective areas on the image. This implies that we can obtain the vanishing point of the line passing through those two points on the image. This vanishing point is nothing but a point on the vanishing line of the plane π.

Figure 7.13 shows an image of a plane on which circles having a same area are distributing. From this image, we first extracted an arbitrary pair of circles, then located the vanishing point for this pair by applying above equation of (7.20), and plotted the points which must be on the vanishing line on the image next by next. The vanishing line of this plane just came to appear as shown in the figure.

7.4 Invariants for the Perspective Projection — Cross Ratio

By the perspective projection, we have much more complex deformations of figures than those by the orthogonal projection. The most significant difference is that the deformation depends on the position of the figure in the image. For example, lines with the same length and direction may have different lengths on the image depending on their positions. While the perspective projection gives the most realistic image because it follows the real physical optics of imaging, the original shapes of 3D objects and measures of lengths and areas become hard to guess from their images.

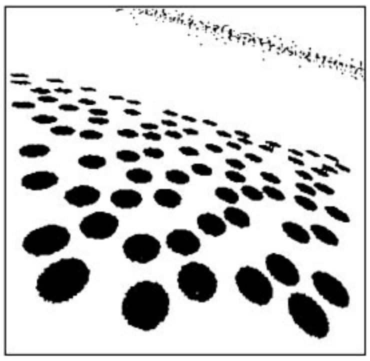


Fig. 7.13. Derivation of the vanishing line of an object plane from images of circles distributing on the plane.

Now, we investigate what kind of measures will not vary by the perspective projection. Such measures are called *invariants*. Typical one of the invariants for the perspective projection is the *cross ratio*.

Let us assume that, as is shown in Fig. 7.14, a line L in the space is projected on the image as a line L', and the four points A, B, C, D on L are projected to A′, B′, C′, D′ on L', respectively. The viewing point O, lines L and L', therefore all the points A, $\cdots$, D, A′, $\cdots$, D′ lie on a plane in the space. Arrangement of these lines and points on the plane are given as shown in Fig. 7.15.

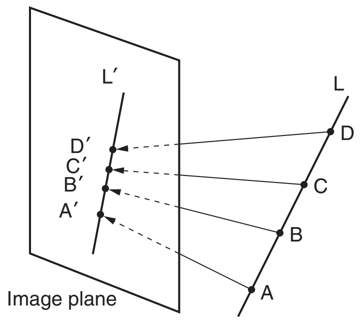


Fig. 7.14. Image of a spatial line and points on it.

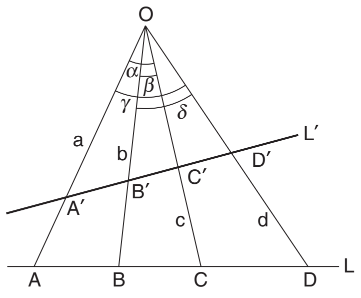


Fig. 7.15. Arrangements of a spatial line and points on it and their images.

In this figure, we set OA $= a$, OB $= b$, OC $= c$, and OD $= d$, and moreover, $\angle$AOC $= \alpha$, $\angle$BOC $= \beta$, $\angle$AOD $= \gamma$, and $\angle$BOD $= \delta$. Because

$$\begin{aligned}\mathrm{AC}/\mathrm{BC} &= \triangle\mathrm{OAC}/\triangle\mathrm{OBC}\\ \triangle\mathrm{OAC} &= \frac{1}{2}ac\sin\alpha\\ \triangle\mathrm{OBC} &= \frac{1}{2}bc\sin\beta\end{aligned} \tag{7.21}$$

then

$$\mathrm{AC}/\mathrm{BC} = a\sin\alpha/b\sin\beta \tag{7.22}$$

and similarly

$$\mathrm{AD}/\mathrm{BD} = a\sin\gamma/b\sin\delta \tag{7.23}$$

Therefore, we have

$$\frac{\mathrm{AC}}{\mathrm{BC}}\bigg/\frac{\mathrm{AD}}{\mathrm{BD}} = \frac{\sin\alpha}{\sin\beta}\bigg/\frac{\sin\gamma}{\sin\delta} \tag{7.24}$$

and also

$$\frac{\mathrm{A'C'}}{\mathrm{B'C'}}\bigg/\frac{\mathrm{A'D'}}{\mathrm{B'D'}} = \frac{\sin\alpha}{\sin\beta}\bigg/\frac{\sin\gamma}{\sin\delta} \tag{7.25}$$

From these equations, the left hand sides of the last two equations become equal. It implies that, by setting

$$R(\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D}) = \frac{\mathrm{AC}\cdot\mathrm{BD}}{\mathrm{BC}\cdot\mathrm{AD}} \tag{7.26}$$

we finally have

$$R(\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D}) = R(\mathrm{A'},\mathrm{B'},\mathrm{C'},\mathrm{D'}) \tag{7.27}$$

This R is called the “cross ratio of the four points A, B, C, and D.” That is, the equation (7.27) implies that “the cross ratio is invariant for the perspective projection.”

Here, it is important that the cross ratio is defined by four points on a line. If we are given a spatial line and three points on it, whose spatial and image positions are known, then the image position of the fourth point is determined, given its relative position to the three points. Conversely, given the image position of the fourth point, its spatial position is determined.

For example, let us imagine an image of rails and crossties which are equipped at constant intervals, as shown in Fig. 7.16(a). For the simplicity, the interval is assumed to be 1. The image is assumed to be given as (b), where images of points A, B and C on the rails are given on the image as A′B′ $= 1$ and A′C′ $= 20/11$.

We give the number of the intervals for the crosstie point from A, just as the first for B, the second for C, and so on, and denote the n-th crosstie point with D_n. Then, we have a problem to determine the position of the image D'_n of the point D_n. The answer is given as follows: First, we have

$$R(\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D}_n) = \frac{\mathrm{AC}\cdot\mathrm{BD}_n}{\mathrm{BC}\cdot\mathrm{AD}_n} = \frac{2\cdot(n-1)}{1\cdot n} \tag{7.28}$$

Then, by setting $\mathrm{A'D}'_n = d_n$, we rewrite it as

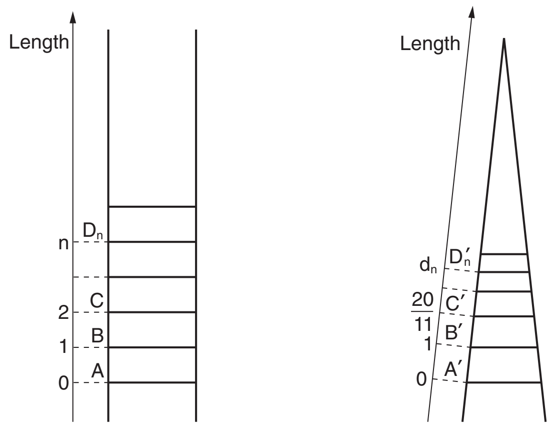


Fig. 7.16. (a) Rails and crossties, and (b) their image.

$$R(\mathrm{A}',\mathrm{B}',\mathrm{C}',\mathrm{D}'_n) = \frac{\mathrm{A}'\mathrm{C}' \cdot \mathrm{B}'\mathrm{D}'_n}{\mathrm{B}'\mathrm{C}' \cdot \mathrm{A}'\mathrm{D}'_n} = \frac{(20/11)\cdot(d_n - 1)}{(9/11)\cdot d_n} \tag{7.29}$$

Therefore, from

$$R(\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D}_n) = R(\mathrm{A}',\mathrm{B}',\mathrm{C}',\mathrm{D}'_n) \tag{7.30}$$

we have

$$d_n = 10n/(n+9) \tag{7.31}$$

For example, $d_3 = 10 \cdot 3/(3+9) = 2.5$ and $d_4 = 10 \cdot 4/(4+9) = 40/13$.

Specially, we are interesting in the position of the vanishing point of rails. This point is given by setting $n \to \infty$. Then, we have the vanishing point as

$$\lim_{n\to\infty} d_n = \lim_{n\to\infty} \frac{10n}{n+9} = \lim_{n\to\infty} \frac{10}{1+9/n} = 10 \tag{7.32}$$

It should be noted that, while we assumed the positions of A, B, C to be at the same intervals in above example, we have no constraints for the positions of these three points. Only we must have their relative positions on the rail to determine its vanishing point.

If we have three points P, Q, R on a spatial line L as shown in Fig. 7.17, whose intervals on the line are known, denoting the point of infinity on L with ∞, the cross ration of P, Q, R and ∞ is given as

$$R(\infty,\mathrm{P},\mathrm{Q},\mathrm{R}) = \frac{\mathrm{P}-\mathrm{R}}{\mathrm{P}-\mathrm{Q}} \tag{7.33}$$

where P − R and P − Q are intervals between P and R and P and Q, respectively. So that, if we also have the images p, q, r of the points P, Q, R, because $R(\mathrm{x},\mathrm{p},\mathrm{q},\mathrm{r}) = \frac{\mathrm{x}-\mathrm{q}}{\mathrm{x}-\mathrm{r}} \cdot \frac{\mathrm{p}-\mathrm{r}}{\mathrm{p}-\mathrm{q}}$, we can determine the vanishing point $\mathrm{x} = \mathrm{x}_\infty$ of L from the next equation

$$\frac{\mathrm{P}-\mathrm{R}}{\mathrm{P}-\mathrm{Q}} = \frac{\mathrm{x}_\infty - \mathrm{q}}{\mathrm{x}_\infty - \mathrm{r}} \cdot \frac{\mathrm{p}-\mathrm{r}}{\mathrm{p}-\mathrm{q}} \tag{7.34}$$

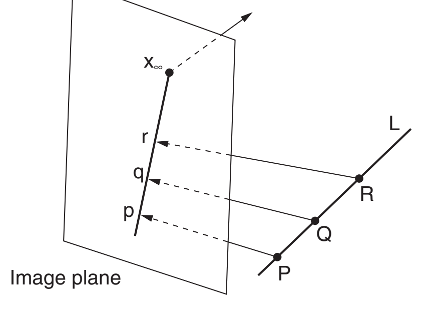


Fig. 7.17. If positions of three points on a spatial line L are known, the vanishing point of the line one the image can be derived.

Then, we have

$$x_\infty = \frac{r(p-q)(P-R) - q(p-r)(P-Q)}{(p-q)(P-R) - (p-r)(P-Q)} \tag{7.35}$$

This means that, if we have three points on a line and their intervals are known, we can determine the vanishing point of the line.

Previously in this chapter, we treated with a problem to determine the spatial slant of a plane from the image of a parallelogram on it. We utilized two sets of its parallel edges. Extending their image lines, we obtained two vanishing points and determined the vanishing line connecting them. This problem also can be solved utilizing the notion of the cross ratio [15].

As shown in Fig. 7.18, we are given a parallelogram ABCD in space and its image A′B′C′D′. From the geometry of parallelogram, denoting the crossing point of two diagonal lines AC and BD with P, we have AP = PC and BP = PD. The image P′ of the point P is given on the image also as the crossing point of two diagonal lines A′C′ and B′D′. Therefore, each respective vanishing points of two diagonal lines A′C′ and B′D′ are obtained from the knowledge that the three image points A′, P′ and C′ are originally spacing with equal intervals, and it holds also for B′, P′ and D′. Also, we obtain the vanishing line of the plane on which the parallelogram lies by connecting these two vanishing points, and finally determine the spatial slant of the plane.

By this method, we obtain two vanishing points much more stably than the method to extend edges. It is because, when the spatial plane is nearly parallel to the image plane, images of two corresponding edges of parallelogram become nearly parallel, and the determination of crossing point of two nearly parallel lines becomes too unstable.

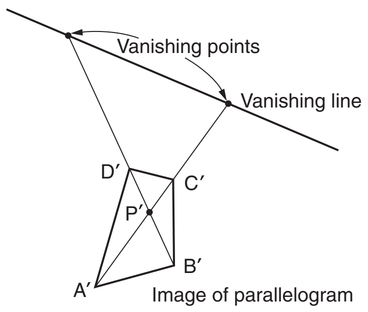


Fig. 7.18. The slant of a plane on which a parallelogram lies can be determined by applying the cross ratio to its diagonal lines and obtaining two vanishing points.

8. Matching by Invariances

8.1 Model-Based Matching

To search and find a specific object in a given image is one of the main computer vision problems [16]. The problem, where, first, some models of object shapes are given, then, we decide which models are included in an observed image or matched to some parts of the image, is called the *model-based matching*.

The simplest model-based matching is the following one. As shown in Fig. 8.1, for example, first, the models are given as images as (a). Then, the models in (a) are overlaid on observed images (b) or (c) by shifting and rotating their positions next by next. They are compared with the overlaid parts to decide whether they are matched or not, or to find the most matching positions. In this sense, this method is called the *image matching*. However, this method only works when both of size and pose of the object in model image are just the same as those in the observed image. Usually, the object is observed as an image in a wide variety of conditions and from different directions and distances, so that the method is of small use in itself for general cases. The method must be better if it accepts deformations of the object.

To develop the methods having ability to accept the deformation of the model in some extent, the next two are major problems to be considered: How to incorporate the deformation, and how to reduce the computational cost. It should be noted about the computational cost that the time from when the observed image is given to the completion of the matching is important. The time to analyze and prepare the models is not counted. We may have possibility to reduce this “real time” for matching by achieving thorough preparations of models. This means that it is just the main concept of the model-based matching.

Now, basically as shown in Fig. 8.2, the matching consists of next steps.

1. Pattern features are extracted from both of model and observed images.
2. For some choice of the model features, candidate set of corresponding features in the observed image are selected. Then, we assume the best geometrical transform such as a shift, a magnification or reduction, a rotation, and a collapse to overlap each corresponding pair of the selected features.

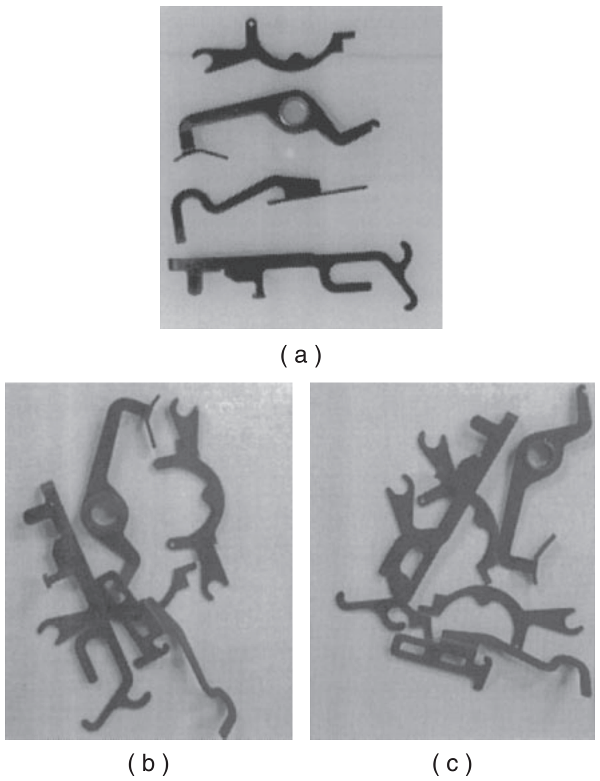
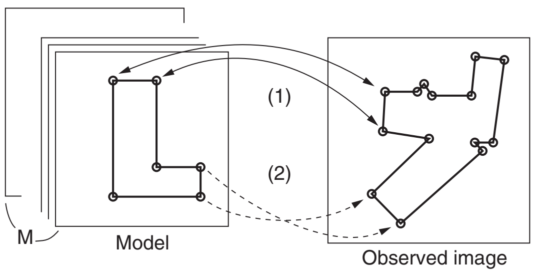


Fig. 8.1. A model-based matching, where given model shapes in (a) are searched in images of (b) or (c).



Select one out of M models. Then;
(1) Choose k feature points in the model and assume their respective correspondences in the observed image. Then, determine the transform by which the correspondeces hold.
(2) Check whether, by this transform, the correspondences of the rest points also hold or not.
If all the correspondences are successful, matching has been done.
If not, try another choice of k points in the model. If all the choices fail, select another model.

Fig. 8.2. Steps of the model-based matching.

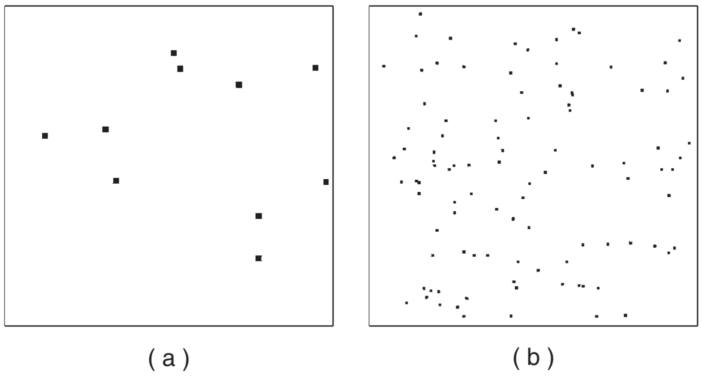


Fig. 8.3. Where is the model point pattern (a) in the observed point pattern image (b)?

3. The assumed transform is applied to all the model features, and whether the consistent matching to the features in the observed image can be achieved or not is checked.

If the complete matching between the model and the observed features holds, we conclude that the observed image matches to the model image. If not, the selection of the above second step is repeated.

Let us analyze computational costs of these steps. We assume that a model has n feature points while an observed image has m observed feature points. Then, the problem is, just as shown in Fig. 8.3, to find the location of the set of point pattern (a) in the point pattern image (b).

We may have several cases of conditions for this problem, each of which gives different difficulty level, such that;

1. Pattern (a) is placed within pattern (b) in its original form. (Its position is unknown.)
2. In (b), (a) is placed with magnified or reduced size. (Its size is unknown)
3. Moreover, in (b), (a) is rotated. (Its rotation angle is unknown.)
4. Moreover, in (b), (a) is collapsed with some unknown manner. (Elastic deformation is a typical example.)
5. Moreover, in (b), (a) is transformed with more complex rules (Perspective transform is a typical example.).

The matching becomes more difficult in this order. This difficulty is expressed with the number of feature points in the model pattern needed to define and identify the "geometrical transform" in the second step of above mentioned matching process.

Now, let us assume that k points are needed to define the transform.

For example, for Case 1, where only the position of model pattern in the observed pattern is unknown and its original shape and pose are kept, then, if any one point of the model pattern is corresponded to some one of the observed feature point, the total position of the model pattern in the observed image can be identified. That is, for this case, $k = 1$. If additionally the size is unknown, as is Case 2, two point correspondence totally needed, and $k = 2$. We consider later what kind of the transform needs how many number of k in more detail.

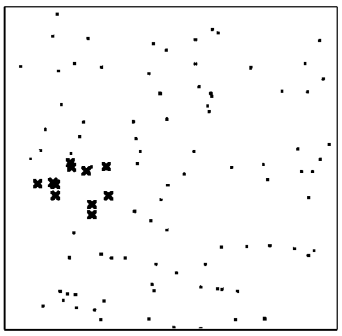


Fig. 8.4. Result of the matching for patterns in Fig. 8.3.

Here, we assume that we can define and identify a transform if any arbitrary k points in the model pattern have their own respective correspondences in the observed pattern. The total number of choices of k candidate of the corresponding points from m observed feature points is given as $m \times (m-1) \times \cdots\cdots (m-k+1)$. For these each selections, we need to check whether the rest $(n-k)$ model points make their own respective correspondences within rest $(m-k)$ observed feature points or not. Then, total computational cost will be estimated as an order of $m \times (m-1) \times \cdots\cdots (m-k) \times (n-k)$. For the patterns of Fig. 8.3, we are given the numbers of the points on the model $n = 10$ and the observed pattern $m = 100$. The position and the size of the model (a) in the observed image are unknown. For this case, $k = 2$, and the computational cost becomes an order of $100 \times 99 \times 98 \times 8$.

Figure 8.4 is the result of the matching for patterns in Fig. 8.3, which shows where the model pattern lies in the observed image. (For one point of the model, two candidates on the observed points are marked as its correspondence and cannot be determined which is true one.)

8.2 Invariants for Geometrical Transform

As previously described, the matching is based on the idea that a positional relation between or among feature points on the model pattern maintains also between or among corresponding points on the observed image.

We are treating with the cases where the model placed not only with a positional shift but also with a transform of rotation, magnification, reduction and so on. Therefore, for the decision whether the positional relation is maintained or not, it must be convenient to express the relations with some measures which are invariant to such the transforms. Just as introduced in the previous chapter, those measures are called "invariants." [17]

For example, in the simplest case where the model pattern has only a parallel shift in the observed image, first, we choose one base point and express positions of all other points in their relative positions to the base point. These relative coordinates are invariant to the parallel shift, because the base point will have parallel shift together. In this case, if we establish correct correspondence of the base point to a point in the observed image, we can make other correspondences by comparing the relative coordinates of the feature points.

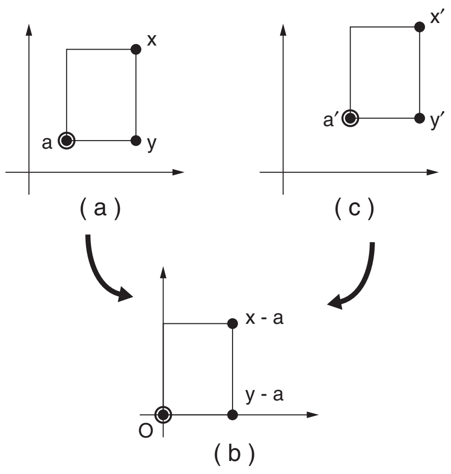


Fig. 8.5. Invariants for the parallel shift of the model pattern.

As shown in Fig. 8.5, for the feature points $\boldsymbol{a}, \boldsymbol{x}, \boldsymbol{y}, \cdots$ on the model pattern of (a), we chose $\boldsymbol{a}$ for the base point, and express other point as $\boldsymbol{x}-\boldsymbol{a}, \boldsymbol{y}-\boldsymbol{a}, \cdots$ as (b). We call these coordinates the "relative coordinates with the *basis* $\boldsymbol{a}$."

When also the feature points $\boldsymbol{a}', \boldsymbol{x}', \boldsymbol{y}', \cdots$ on the observed image as shown in (c) are given, and we express them as the relative coordinates with the basis of $\boldsymbol{a}'$, if these two patterns are same and $\boldsymbol{a}$ corresponds $\boldsymbol{a}'$, then, the relative coordinates of all other points will match.

That is, such the relative coordinates do not change by a parallel shift. We describe this as "the relative coordinates with one basis point are invariant for parallel shift."

Next, we assume the case where the pattern has parallel shift, rotation, and magnification or reduction, simultaneously. This is just a *similarity transform*.

For this case, as shown in Fig. 8.6, we choose two point $\boldsymbol{a}$ and $\boldsymbol{b}$ within the pattern, then, we transform the model pattern so that the point $\boldsymbol{a}$ lies just on the coordinate origin O and $\boldsymbol{b}$ lies on 1 on the x-axis by a parallel shift, a rotation, and a magnification or reduction. This transform is schematically shown in the figure as from (a) to (b). We assume that the coordinates of a point $\boldsymbol{x}$ becomes (α, β) by this transform.

Also for the observed pattern shown as (c), we choose two points of $\boldsymbol{a}'$ and $\boldsymbol{b}'$ and apply the same transform. If the patterns (a) and (c) are similar and $\boldsymbol{a}$ corresponds to $\boldsymbol{a}'$ and $\boldsymbol{b}$ corresponds to $\boldsymbol{b}'$, then, the new relative coordinates of a point $\boldsymbol{x}'$, which corresponds to $\boldsymbol{x}$, will become (α, β). That is, in other words, "the relative coordinates with a basis pair of $(\boldsymbol{a}, \boldsymbol{b})$ are invariants for the similarity transform."

Next, let us consider invariant coordinates for more general transforms. In addition to the above mentioned parallel shift, rotation, and magnification or reduction, we allow a special expansion only in a specific direction. This transform is observed when a figure is seen at a sidelong glance. It is called the *affine transform*. The affine transform can be expected to approximate almost all kinds of deformations of figures effectively.

For the affine transform, the invariants are constructed as follows [18]. As is shown in Fig. 8.7, we choose three feature points $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$, for example, as the basis. Then we set the relative coordinate system for which $\boldsymbol{b}-\boldsymbol{a} \equiv (1,0), \boldsymbol{c}-\boldsymbol{a} \equiv (0,1)$ will hold, where $\boldsymbol{a}$ corresponds to the origin $\mathrm{O}(0,0)$, $\boldsymbol{b}$ to $(1,0)$, and $\boldsymbol{c}$ to $(0,1)$. The relative coordinates (α, β) for another each point z will be given as to hold

$$z = \alpha(\boldsymbol{b}-\boldsymbol{a}) + \beta(\boldsymbol{c}-\boldsymbol{a}) + \boldsymbol{a}$$

For all the respective feature points, such the coordinates (α, β) are determined uniquely.

Because the point $\boldsymbol{x}$ in the figure (a) is expressed as $\boldsymbol{x} = (\boldsymbol{b}-\boldsymbol{a}) + (\boldsymbol{c}-\boldsymbol{a}) + \boldsymbol{a}$, its relative coordinates are $(1,1)$ as shown in (b). For the pattern in (c), if $\boldsymbol{a}'$ corresponds to $\boldsymbol{a}$, $\boldsymbol{b}'$ to $\boldsymbol{b}$, and $\boldsymbol{c}'$ to $\boldsymbol{c}$, the point $\boldsymbol{x}'$ which has same relative position to those points will have the same relative coordinates $(1,1)$, and those of z' will be (α, β). That is, this coordinate system (α, β) is invariant for the affine transform. For this case, the basis is three point set of $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$.

For other kind geometrical transforms to be expected, some respective invariants have been proposed. In general, for the more complex transform, the more number of points for the basis will be needed. For the perspective transform introduced in the previous chapter, also an invariant coordinate system based on the notion of the cross ratio has been proposed. Its detail is described later in this chapter.

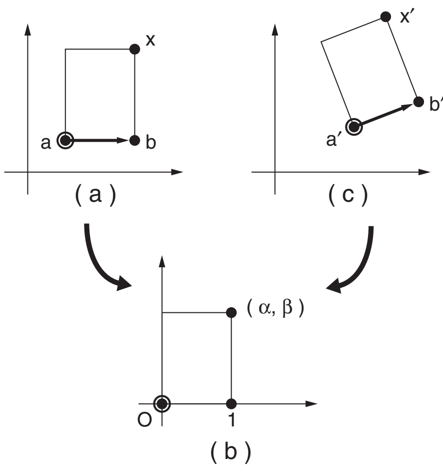


Fig. 8.6. Invariants for the similarity transform.

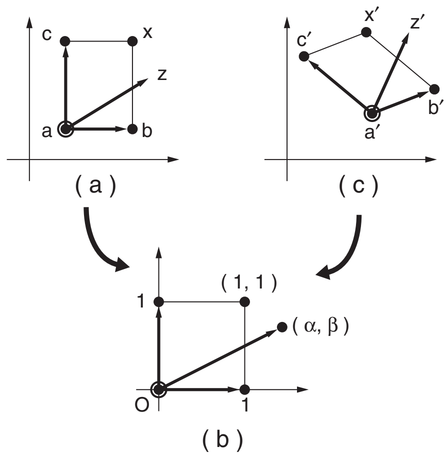


Fig. 8.7. Construction of invariant coordinates for the affine transform. For a point z, with the three point basis of $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ and setting $(\boldsymbol{u} = \boldsymbol{b} - \boldsymbol{a}, \boldsymbol{v} = \boldsymbol{c} - \boldsymbol{a})$ to be unit vectors in two coordinate axes, its relative coordinate (α, β) will be an invariant to the affine transform.

8.3 Geometric Hashing

8.3.1 Matching using invariants and hash-table

The geometric hashing [19] is an algorithm for speed-up the solution of such the general point matching problem as shown in Fig. 8.3, in which the above mentioned notion of "invariant" is used for point correspondences. The "hash table" is used for efficient searching of the corresponding candidates. It first constructs a data base on the geometric structure of the model points in the preprocessing stage to accelerate the searching speed. It is a framework for making the second and third steps of above mentioned general matching methodology effective.

In the geometrical matching employing the invariants, first we should pick up the basis from both of the model pattern and the observed image. Only when those basis correspond to each other, the matching of the invariant coordinates works meaningfully. But, it is not always possible to choose the properly meaningful basis from the beginning. Then, we must take trials of the choice next by next. The number of the possible choices in the model is, for example of the affine transform shown in Fig. 8.3, because we choose three out of ten feature points, $10 \times 9 \times 8$. On the other hand, for the observed image, it becomes $100 \times 99 \times 98$, because we choose also three out of 100 feature points. These make the number of the trials for correspondences become their products. In each trial, we check the coincidences of the invariant coordinates of the rest points. It results in, as the worst case, the total number of the comparisons of $(10 \times 9 \times 8 \times 7) \times (100 \times 99 \times 98 \times 97)$ if we check all the possibilities.

The geometric hashing provides an idea to reduce this computational cost by a complete preparation of the matching in advance. We are given the models beforehand, then we extract sufficient information from them, and we can reduce the computational cost after receiving the observed image. For the example above, the number of comparison will be reduced at $1/(11 \times 10 \times 9 \times 8)$.

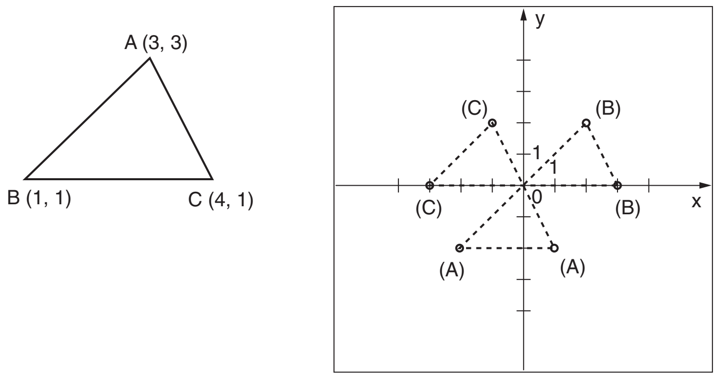


Fig. 8.8. Invariant coordinates data base for the triangle ABC and the transform of parallel shift only.

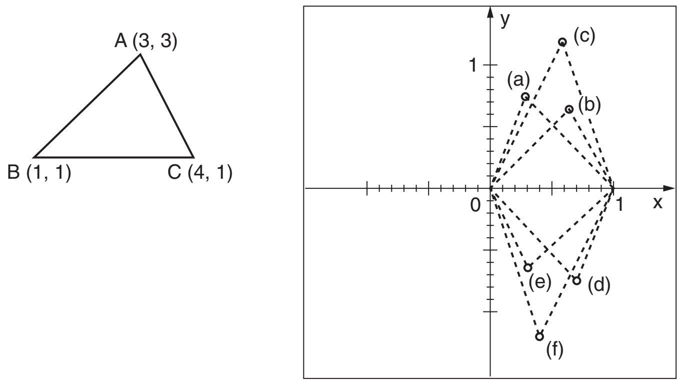


Fig. 8.9. Invariant coordinates data base for the triangle ABC and the similarity transform.

The geometric hashing consists of two stages: The first one is the *Preprocessing stage*, where a data base for the comparison is constructed from the models, and the second one is the *Recognition stage*, where what kind of transform must be applied to a model is decided referring to the data base to fit the structure of feature points in the observed image.

In this method, because the Preprocessing stage can be processed beforehand as its name, how much we can reduce the computational cost of the second Recognition stage is its practical key point.

8.3.2 Feature point data base on invariant coordinates

In the Preprocessing stage, feature points are extracted from models at the beginning, then, the data base for the comparison is constructed by referring the invariant coordinates of those points.

For example, let us assume the transform consists of only parallel shift. We have a model figure of triangle ABC as shown in the left side of Fig. 8.8, coordinates of whose vertices are also shown in the figure.

For the *parallel shift*, one point is sufficient to be the basis. We choose it from the vertices A, B, and C, that is, we have three choices. For example, if we choose A as the basis, we have invariant coordinates of B and C as shown in the figure with marks of "(A)." Also the coordinates shown with marks of "(B)" and "(C)" are the invariant coordinates of two other vertices when we choose B and C, respectively, as the basis. These coordinates are all of the components of the data base for this triangle model.

The invariant coordinates data base for the same triangle ABC and the *similarity transform* is shown in Fig. 8.9. Two points are needed for the basis. The invariant coordinates marked "(a)" are the relative coordinates of C when A and B are chosen as the basis, and the rests are:

(b):	the relative coordinates of	C	with basis of	A and B
(c):	-	A	-	B and C,
(d):	-	B	-	C and A,
(e):	-	C	-	B and A,
(f):	-	A	-	C and B,
(g):	-	B	-	A and C.

8.3.3 Construction of data base on hash table

The hash table is a kind of data structures. Figure 8.10 is an example of the two dimensional hash table, where small square *bins*, which are also called buckets, are placed at every grid points and store data within them.

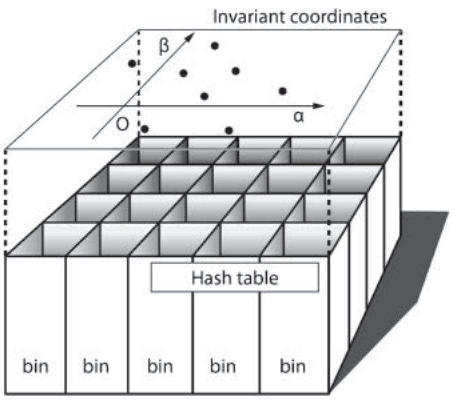


Fig. 8.10. 2-D hash table.

Two dimensional coordinates (α, β) are quantized into small width square grids, and each bin on the hash table corresponds to a grid point. Here, the coordinates (α, β) are the invariant relative coordinates determined according to the transform of models to the observed image. That is, a data in a set of $(d_1, d_2, \ldots)$ is referred with the coordinates (α, β).

As was mentioned above, first, in the Preprocessing stage, after extraction of feature points from models, then, a data based for the comparison is constructed with their invariant coordinates.

It was also mentioned that, to define invariant coordinates for the affine transform, a basis consisting of three points $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ was needed. The basis points are chosen from all the three point combinations within all the feature points in a model. For every basis, relative positions of other points than the basis are represented as invariant coordinates, and written in the hash table as followings: For example, we assume that we are given a set of feature points $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}, \boldsymbol{e}, \cdots$ and chose $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ as a set of basis. We also assume that, with this basis, the point $\boldsymbol{d}$ is represented with invariant coordinates (α, β). Then, into the bin corresponding to the coordinates (α, β), the basis name of $B_m = (\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ is written to state that "the model has a feature point at this relative coordinates when the set of $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ is chosen as a basis." All the model points other than the basis are registered in the hash table by writing their basis into bins corresponding to their respective invariant coordinates.

If we have only one model, this is sufficient for the necessary information. If we have multiple models to compare, name of the model should be additionally written in. Then, choosing another basis, for example, the set of $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{d})$, similar writings are carried out.

Figure 8.11 shows these processes to write data about the feature points of models into bins on the hash table.

We repeat this preprocessing for all other models. Therefore, if we have M models, all of which have m feature points, and the basis is composed by k points, then,

- for a set of basis, data are written into $m - k$ position bins on the hash table,
- the number of choices of points of the basis set for a model is $m(m-1)(m-2)\cdots(m-k+1)$,
- we have M models.

As the result, the total number of points whose data must be written into bins on the hash table will be

$$Mm(m-1)\cdots(m-k) \sim Mm^{k+1}$$

Of course, some data may be written into same bin.

8.3.4 Voting in the recognition stage

To identify the model which mostly fits to the observed image, so called a voting is carried out. Before the voting we begin with choice of a basis similarly as the data base composition process, as,

1. Extract feature points from the observed image.
2. From those points, choose a set of basis B_{im}, (If the affine transform is assumed, for example, it is composed by three points.) and obtain relative coordinates for all the rest points to the basis points.
3. By using those relative coordinates, refer the hash table. If, for example, for a point on the observed image, the bin referred by its relative coordinates (α, β) has some data about some models, that is, if we have a data of $(\alpha, \beta) \longrightarrow (M_1, B_1)$, where M_1 is the model number and B_1 is the number specifying the basis, then we vote one to the set of (M_1, B_1) which is a combination of (*model, basis*).
4. When all the votings for all the points in the observed image other than the basis point are completed, find the pair of (*model, basis*) which obtain the maximum number of votes.
5. If the maximum number of the vote is sufficiently large (the threshold is determined beforehand), then go to the

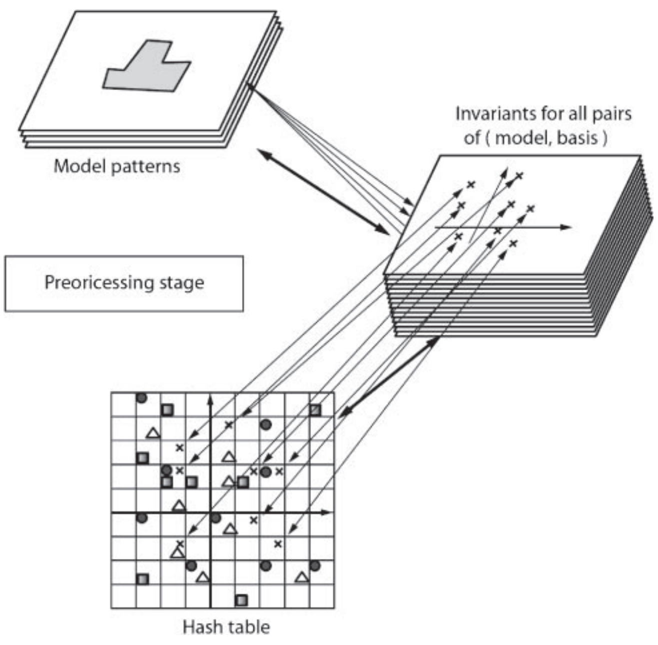


Fig. 8.11. Writing invariant coordinates data into bins of the hash table.

next step with the pair of (*model, basis*) $= (M_{max}, B_{max})$. If not, go back to 2., and choose new set of the basis B_{im} for the observed image.

6. Calculate the parameters of the transform f which makes $B_{im} = f(B_{max})$, that is, the basis B_{max} coincides to the basis B_{im} of the observed image.
7. Apply the transform f to all the rest points of the model M_{max}, and compare the results with the observed image to verify the validity of the model and the basis. (This step is called the *Verification*.)

For the case where the number of feature points in the observed image is n and the basis consists of k points,

- we survey on the hash table at $n - k$ points for a set of basis, and
- we have totally $n(n-1)(n-2)\cdots(n-k+1)$ choices for the basis set.

Therefore, we must take $n(n-1)\cdots(n-k) \sim n^{k+1}$ searches, for the worst case, on the hash table to find one model pattern.

The verification stage is just the matching stage in the conventional pattern matching strategies. The voting on the hash table can be considered as a process or a filtering to (largely) reduce the number of candidates for the matching.

By the way, for the case to find out a pattern by a simple matching technique, as was mentioned at the beginning of this chapter, with the same condition,

- we have $m(m-1)(m-2)\cdots(m-k+1)$ choices for the basis of a model, and,
- $n(n-1)(n-2)\cdots(n-k+1)$ choices for the basis of the observed image, then,
- we assume a transform from a chosen pair of bases, apply it to the rest $(m-k)$ points of the model, and verify whether they are consistently matched, and,
- repeat these processes for all the M models.

The total number of comparison needed here is of the order of $Mm^{k+1}n^k$. This implies that the computational cost is largely reduced by the hash table technique.

Some additional ideas to reduce the cost furthermore can be incorporated. For example, to avoid such a choice of basis in the observed image that can never correspond to the basis of models, to eliminate the writing into bin of (β, α) if same data are written into bin of (α, β) for the affine transform because of its symmetrical property, and so on.

The total process flow is schematized with block diagrams in Fig. 8.12.

8.3.5 Pattern matching by the geometric hashing

Let us follow the process of a simple pattern matching by the geometric hashing.

Figure 8.13(a) shows a model of an air plane. Sharp vertices of its contour are extracted as feature points. The problem is to find this pattern in an observed image of (b). Here, the transform is assumed to be only a simple parallel shift.

Figure 8.14 shows the composed data base. In the data base, for example, symbols of $\bigcirc$ show, if the feature point $\bigcirc$ in the model pattern of Fig. 8.13(a) is placed at the origin O, the places where other feature points will be. Also $\triangle$'s show the places where feature points will be if the $\triangle$ in the model is placed at the origin. For other feature points in the model, the relative positions are registered in the data base with the same manners.

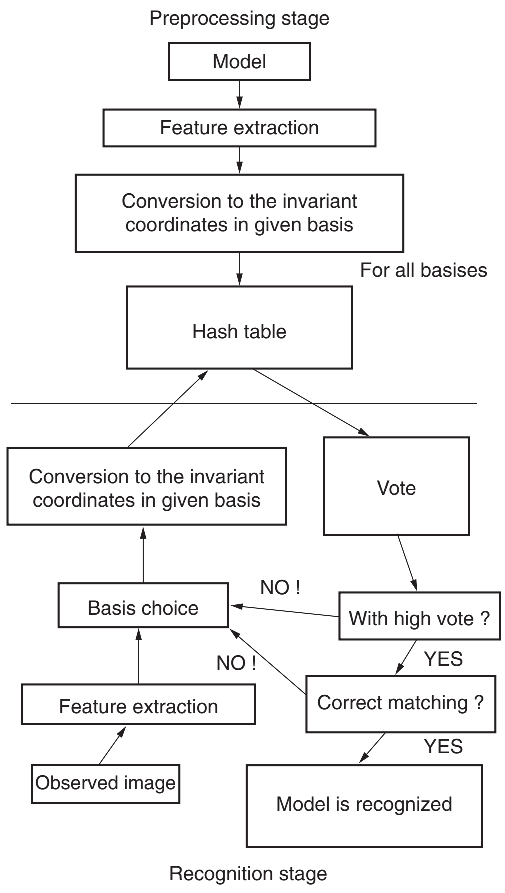


Fig. 8.12. Total process of the geometric hashing.

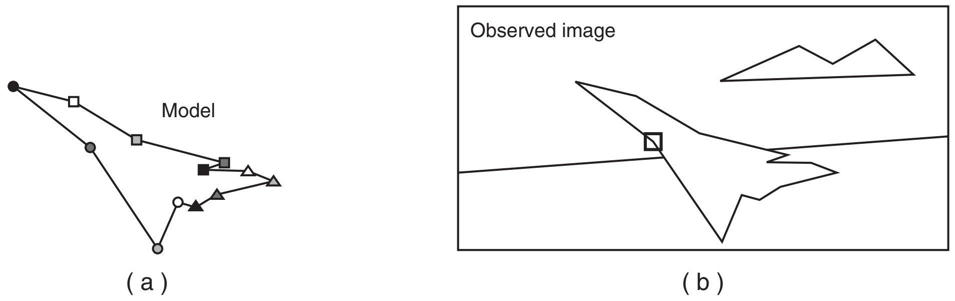


Fig. 8.13. (a) A model of an air plane. This pattern will be found in an observed image (b).

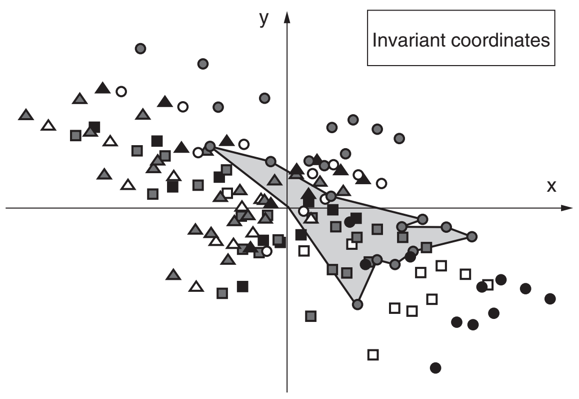


Fig. 8.14. Data base of the invariant coordinates of model feature points in Fig. 8.13(a), for which the respective selected bases are shown with the same symbols in the model. When a point marked with □ in Fig. 8.13(b) is chosen as the basis, on the coordinate plane, an air plane shape shown here is overlaid and the positions of its vertices are registered in the data base.

The processes up to here are of the Preprocessing stage. This data base is registered into bins which are corresponding to small divided cells of the invariant coordinate plane. Those bins compose the hash table.

Now, we are given an observed image shown in Fig. 8.13(b), and step into the Recognition stage. We also extract feature points at vertices of contour edges in the figure. From them we choose randomly a point as the basis, and represent the positions of other points with the relative coordinates to this basis point.

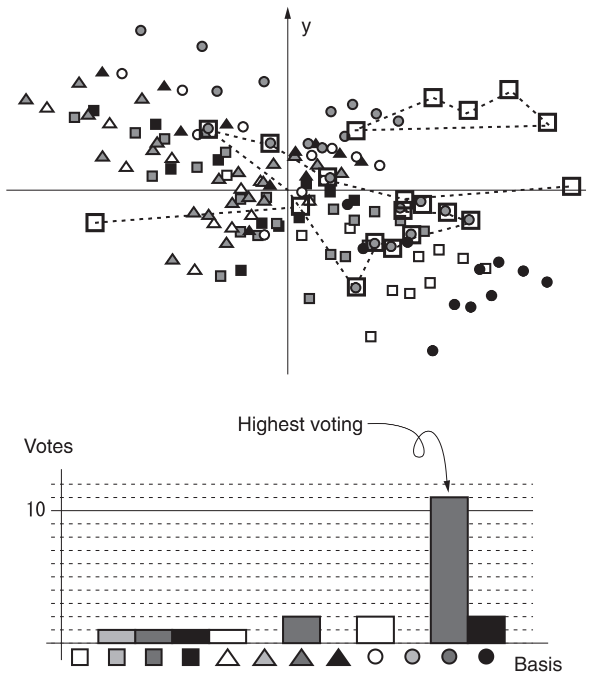


Fig. 8.15. The voting on the hash table and its results.

Next, by using these relative coordinates, the bins on the hash table of Fig. 8.14 are referred. If we choose the point marked with □ in Fig. 8.13(b), for example, in the observed image as the basis, the relative coordinates of all other feature points in the observed image should be placed at respective positions where marked with □'s in the invariant coordinate plane, as shown in the upper figure of Fig. 8.15. Then, if the bins positioned at places marked with these □'s have the symbol of ○, for example, the ○ is voted with the number of bins which contain ○. If two or more basis symbols are written in the bin, the votes are casted for every basis.

After completing this process for any one of feature points in the observed image, the votes for the bases are compared. In the example case here, the basis having the symbol of ● obtained the highest voting, as shown in lower place of Fig. 8.15. The number of votes is sufficient because total votes for all the other symbols than this basis's one is smaller than 11, which is the number obtained by ●. Therefore, in the Recognition stage, as a result, we have a conclusion that the air plane model is found as the point of □ in the observed image is just corresponding to the point of ● in the model.

8.4 Perspective Invariants

8.4.1 Invariant coordinates with cross ratio

At the last section of the previous chapter, we introduced the cross ratio which is invariant for the perspective transform. It is defined with four points A, B, C, D on a line in space as $R(\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D}) = (\mathrm{AC}\cdot\mathrm{BD})/(\mathrm{BC}\cdot\mathrm{AD})$, and reserves its value with their images of those points. For the perspective transform of a plane, we can compose an invariant coordinate system by introducing this notion of the cross ratio.

Figure 8.16(a) shows a perspective projection of the xy plane in (b). In both figures, the origin O of the xy plane, x and y axes, and points of U_x and U_y, which are points at $x = 1$ and $y = 1$ on the respective axes, and the infinite points of I_x and I_y in the x and y directions, respectively, and their projections are shown.

The projection of U(1, 1) will be given as the crossing point of $\mathrm{I}_x\mathrm{U}_y$ and $\mathrm{I}_y\mathrm{U}_x$ as shown in the figure.

The position of the projection of a general point $\mathrm{P}(x, y)$ can be obtained as followings. It is derived from the facts that the perspective projection of a line forms a line and that on both the lines the cross ratio will be reserved.

For a given point $\mathrm{P}(x, y)$, let us denote the crossing point of $\mathrm{I}_y\mathrm{P}$ and the projection of x axis with P_x. Then, from the figure (b), it holds that

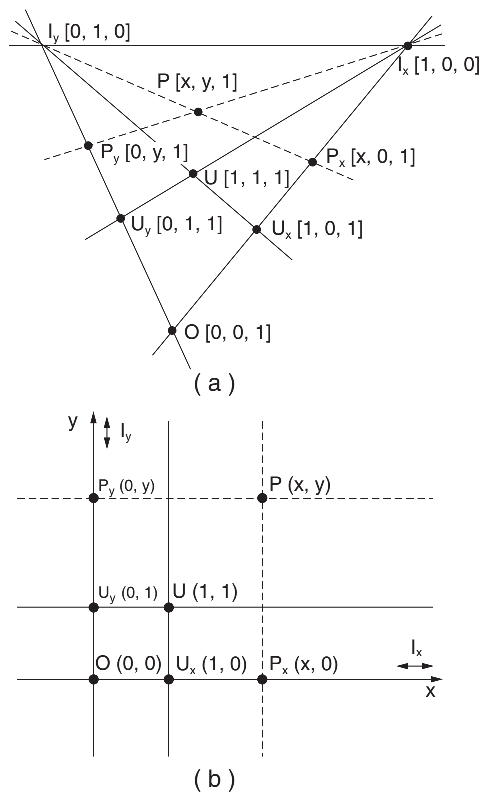


Fig. 8.16. (a) Perspective projection image of *xy* plane of (b). The position of the projection of a point on the *xy* plane of (b) can be obtained by the correspondences of lines and the cross ratio on the lines.

$$R(\mathrm{I}_x, \mathrm{O}, \mathrm{U}_x, \mathrm{P}_x) = x \tag{8.1}$$

Similarly, by denoting the crossing point of $\mathrm{I}_x\mathrm{P}$ and the projection of y axis with P_y, it also holds that

$$R(\mathrm{I}_y, \mathrm{O}, \mathrm{U}_y, \mathrm{P}_y) = y \tag{8.2}$$

The positions of the two points of P_x and P_y are determined from above two equations, and then, as the crossing point of $\mathrm{I}_x\mathrm{P}_y$ and $\mathrm{I}_y\mathrm{P}_x$, the position of the projection of P will be given.

8.4.2 Perspective invariants on angles

In the above example of perspective projection, the transform of a point coordinates needed correspondences of five points. This number of five is just the number of points needed to compose the basis of the perspective transform, while one for the parallel shift, two for the similarity transform, and three for the affine transform [20]. That is, the more complex transforms are assumed, the more number of points are needed to compose the basis.

When the number of points for a basis increases, the number of point data to be written on the hash table increases rapidly, and huge amount of memory space becomes necessary. This is because this number of points is proportional to $m(m-1)\cdots(m-k+1)$ where k is the number of points to compose a basis. However, it should be noted that the computational cost in the Recognition stage will not increase. It is proportional to the number of feature points. This is a great merit of the geometric hashing.

In the following, we show a use of the cross ration for its wider application to construct new invariants in perspective transform.

The cross ratio was originally introduced to describe the theory of relations among angles and their opposite edge lengths as shown in Fig. 8.17.

The theory tells that by defining

$$R(\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}) = \frac{\mathrm{AC}\cdot\mathrm{BD}}{\mathrm{BC}\cdot\mathrm{AD}} \tag{8.3}$$

the value of $R(\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D})$ is invariant regardless the placements of the line L. This came from that the right hand side of the equation of

$$\frac{\mathrm{AC}}{\mathrm{BC}} \Big/ \frac{\mathrm{AD}}{\mathrm{BD}} = \frac{\sin\alpha}{\sin\beta} \Big/ \frac{\sin\gamma}{\sin\delta} \tag{8.4}$$

was independent of the placement of L.

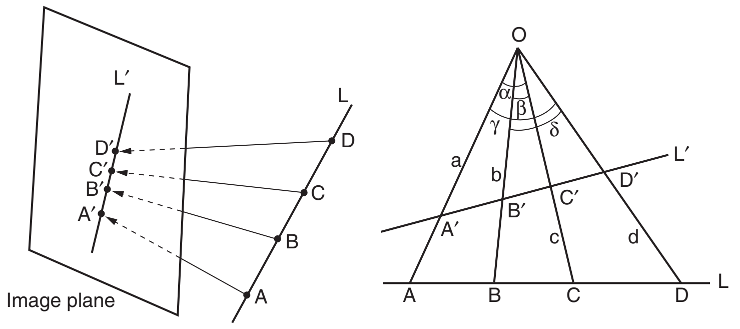


Fig. 8.17. The cross ratio of the point O and points A, B, C, D on a line *L*.

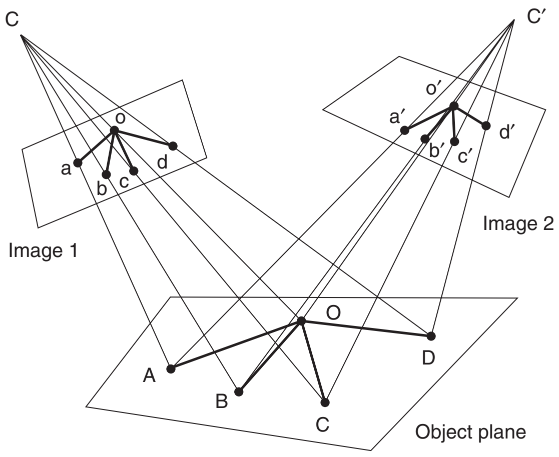
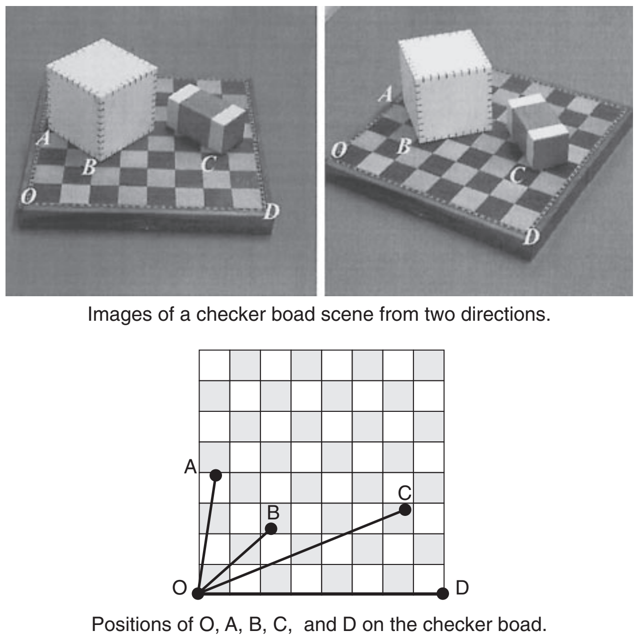


Fig. 8.18. Cross ratio of angles on perspective projection images.



A
B
C
O
D
Images of a checker boad scene from two directions.
A
B
C
O
D
Positions of O, A, B, C, and D on the checker boad.

Fig. 8.19. Examples of cross ratios of angles on perspective projection images. The cross ratio of angles composed with the center O and four points of A, B, C, and D is invariant for different viewing points.

This means that for the five points on a plane, which are O, A, B, C, and D, the cross ratio of the right hand side of (8.4) describing with the angles spanned by the central point O and the edges of OA, OB, OC, and OD is also reserved for the perspective projection. That is, as shown in Fig. 8.18, for a set of five points on a plane, a new invariant is composed with angles with respect to one point among them.

When these given five points on a plane are viewed from other two points in space and projected on respective two image planes as shown in the figure, on those two image plane, the two sets of respective five points form their cross ratios on their own angles. Those cross ratios of angles with the centers of O and its projections of o and o′ and the sets of edges of OA, OB, OC, OD, and their projections of oa, ob, oc, od and o′a′, o′b′, o′c′, o′d′ are of the same value of $\frac{\sin\alpha}{\sin\beta} / \frac{\sin\gamma}{\sin\delta}$ and will be invariant.

Two images of a checker board viewed from two different points, for example, are shown in Fig. 8.19, both on which images of five points O, A, B, C, and D on the checker board are shown. On these three planes, which are the original checker board and its tow images, the cross ratios of angles were read. They were all 1.43. This confirms the invariance of the cross ratio of angles.

9. Image Matching

9.1 Knowledge Based Image Recognition and Matching

9.1.1 Graph representation of an image

In Section 3, we introduced the representation of an image by graphs for the structural pattern recognitions. This representation focused on the hierarchical structure of patterns. We show again a typical example of scene image having the hierarchical structure, which is represented in a graph, in Fig. 9.1. Here, in the graph representation, relations between component objects are more explicitly presented. These relations labeled in the graph are not only on their positional relations and inclusions but also on their functional relations. Such a graph structure is called the *semantic network*. (In the following, words of "network" and "graph" are used with the same meaning.)

In this section, we show the framework of so-called the "knowledge based image recognition and matching" employing the semantic networks.

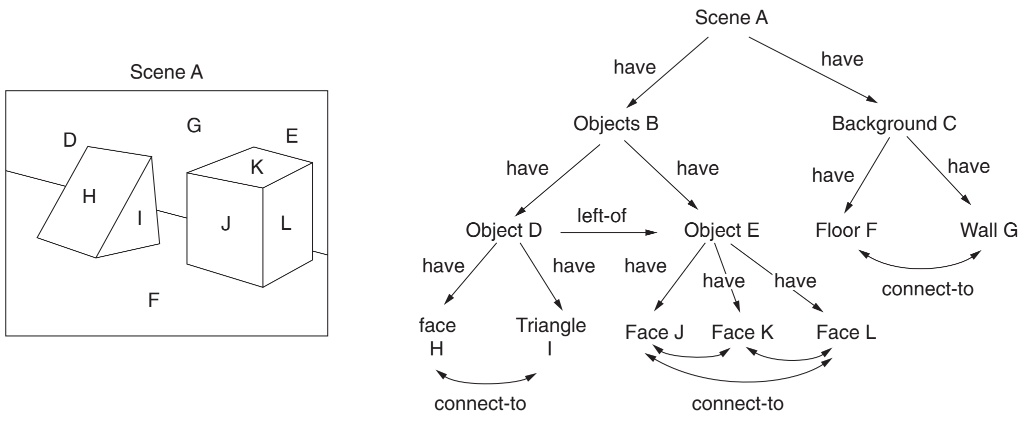


Fig. 9.1. An example of the semantic network representing a scene.

The semantic network represents objects and their mutual relations by a graph having nodes and arcs. Usually, a node represents an object, and an arc represents a relation between two objects, and it may have a label or a direction. In a graph, if two nodes X and Y are connected with an arc having a label *L* and a direction from X to Y, it may present the proposition "For X, Y has a relation of *L*."

An arc having no label and no direction simply means that the two nodes have some relation. Therefore, we may have many different network representations for the same meaning or the same relation. For example, to represent the scene in Fig. 9.2(a), we may not only have a simple network of (b) where only positional relation of two objects is represented with an arc, but (c), in which properties of the objects are also represented as arcs with labels, and (d), in which arcs are representing only existences of some relations between objects and all the objects, properties, and relations are explicitly represented with nodes. In the last representation of (c), other than real objects themselves, their properties and relations are considered to be objects.

Of course, how highly explicit representation we should have, in practical use, is depending on how much efficiently we can utilize the labels attached on arcs. If we have not proper manner to interpret the labels, the "semantics" of the network cannot be correctly read out.

The effectiveness of the semantic network for image representations comes from its property that, in the network, objects having special mutual relation or common concepts are combined directly with arcs. That is, spatial relations or states of objects are represented by simple and explicit ways.

Another more important property of the semantic network is that it is also useful to represent our knowledge about objects. Therefore, the representation of images with a semantic network is convenient to realize the matching between "knowledge" and images.

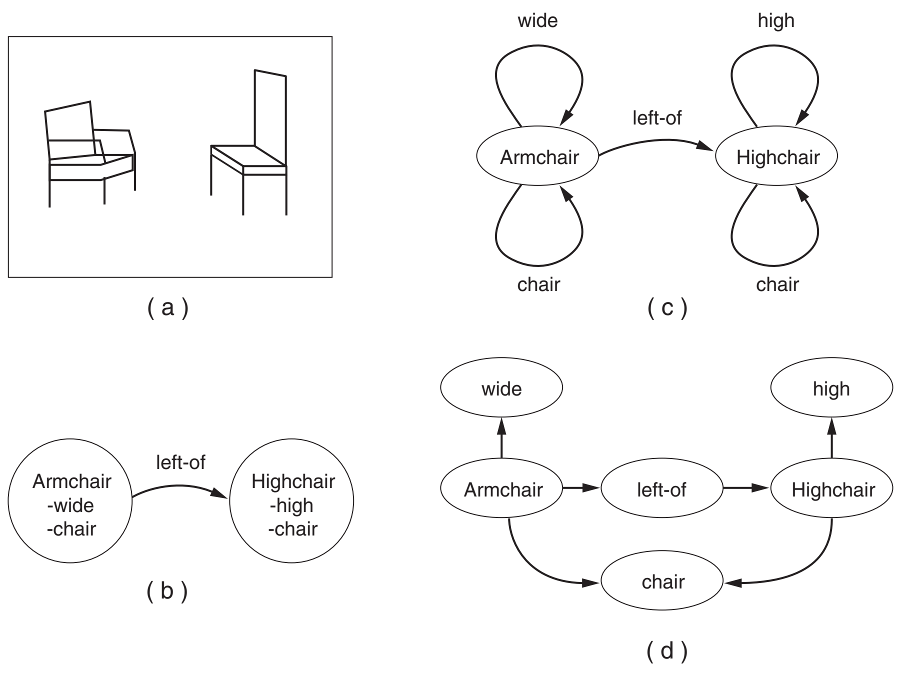


Fig. 9.2. Different networks all of which equivalently represent the same scene. (a) Scene. (b) A simple network. (c) Another network in which object properties are represented by labeled arcs. (d) More structural network in which all of objects, properties and relations are represented as nodes explicitly.

9.1.2 Graph matching of knowledge and images

Now, let us consider the matching process when such semantic networks on the knowledge and images are given.

We have a graph $X = (V_x, E_x)$ to match, where V_x is a set of nodes and E_x is a set of arcs. ("V" and "E" came from "vertex" and "edge," respectively, in the drawing form of graphs.) We are also given another input graph $Y = (V_y, E_y)$ to be matched, where similarly V_y is a set of nodes and E_y is a set of arcs. Between components of both sets of nodes V_x and V_y and also between components of both sets of arcs E_x and E_y, if one-to-one correspondences hold, and, at the same time, arcs connecting corresponding nodes are also corresponding, then this pair of graphs X and Y are called the *graph isomorphism*. That is, X and Y have same structure as graphs.

For example, if both a "knowledge" and an image have same meaning of an object, the semantic network representing the knowledge and that representing the image must be the graph isomorphism. Then, additionally, it must be checked whether if the corresponding nodes and arcs and labels on both of them have same meaning or equivalent.

If the complete isomorphism does not hold, we have many cases where the graphs are partially matching. The important and meaningful cases are where one graph is just a part of other graph. In general, the knowledge is incorporated with sufficient and wide variety of components. On the other hand, in almost cases, the image is a snap shot and an instance of the object or the event and incorporated with an incomplete information. That is, in a knowledge data base, about an object, all of its properties are listed up and stored, while from an image only small part of properties of the object can be extracted.

For example, Fig. 9.3(a) shows a shape of a hammer, and (b) shows the semantic network of the hammer in which all of the features possibly appeared on the image of hammer and their relations are listed up in the graph structure. Not all the nodes and arcs in (b) can appear on an image.

Now, to confirm if the given image of (a) is of a hammer, the network extracted from the image must be checked up whether if it is a part of the semantic network of (b) and how wide overlap is has.

Even if we have a partial matching of graphs, we have still a problem whether if those matching are of meaningful and important nodes and arcs. In fact, there exit both of essential nodes and arcs for practical cases, and also we have rather unimportant ones in the semantic network. Therefore, every correspondences of nodes and arcs must be evaluates with some respective weights. and the weighted sum of the evaluation must meet for a total matching in usual practical cases. However, the best matching by this evaluation or the maximizing the evaluation requires, in general, huge computational costs, and it is not obtained easily.

For such the matching problems, usually some strategy is worked out from human empirical knowledge (it is called "heuristics"), and the prioritized matching candidates which may be nearly optimal are searched. Decomposing the problem into some small subproblems may also be a good and efficient strategy. Many practical matching algorithms which do not always find the best matching but give sufficiently good solutions have been proposed in the literatures.

Hereafter, we show algorithms, for the case where one of the graphs must be a part of other one, to find the best matching in the sense that the number of correspondences between nodes becomes maximum.

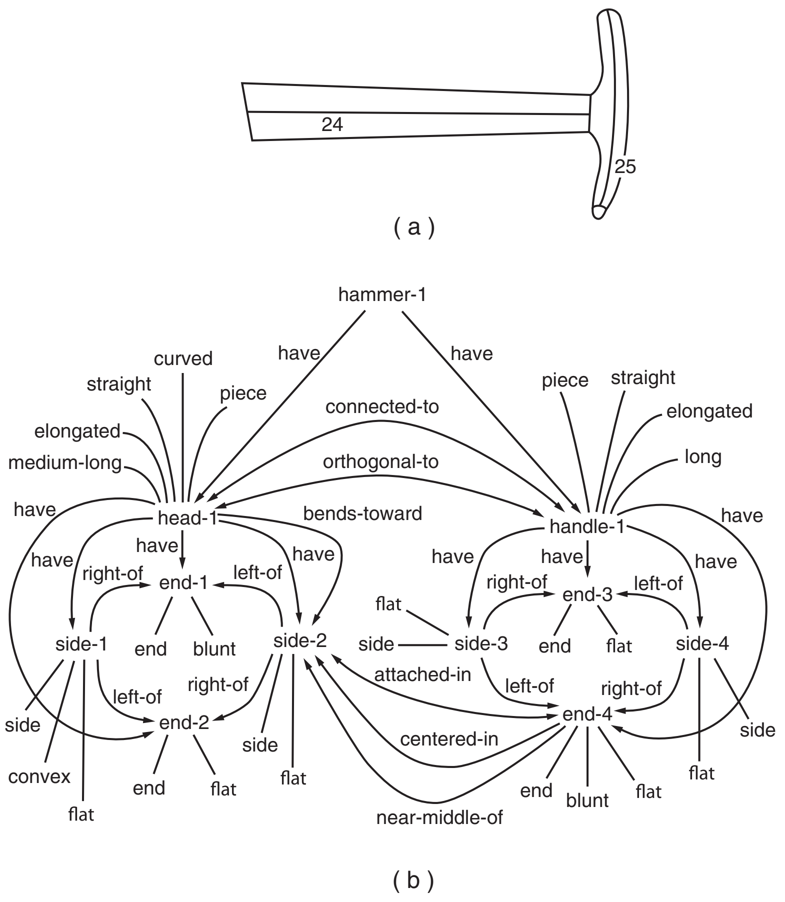


Fig. 9.3. (a) Shape of a hammer. (b) Semantic network of the shape of hammer in which all of the features possibly appeared on the image of hammer and their relations are listed up.

9.2 Graph Matching by Search Tree

Figure 9.4 shows an example of graph matching by employing a *search tree*. A graph shown in (a) partially matches with (b). The problem is to find the best set of correspondences between nodes in both graphs.

Arcs in these graphs are assumed to have no directions nor labels, for the simplicity. The nodes shown with △'s have common property and can be mutually matched. Similarly, it holds between nodes having □'s and ○'s, respectively.

In this algorithm, the search tree is composed as follows. First, we pick up the node A. The node A in the graph (a) can correspond to nodes of A′ or A″ in (b). So, we make a layer named as "Selection for A″ and place two branches of the selections A : A′ and A : A″ from the root of tree shown in (c).

Because in (a) we have a constraint that the arc going out from the node A must connect to only nodes having the property of □, for the selection of A, the correspondence of A : A′ allows B : B′, B : B″, or B : B‴ in the next layer. For the A : A″, only B : B″ or B : B‴ is allowed as possible correspondences. These selections are added to the tree as the branches from the selections of *A*.

By placing branches of possible matching of nodes of □'s which can be connected to the nodes of ○'s in the next layer of the tree, the tree (c) is fully composed.

In this tree, a path from the root to a leaf expresses a set of correspondences, so that the number of branches on the path is just the number of corresponding nodes in the matching. The longer this path is, the better the matching must be in this algorithm.

Also in this tree, paths having three branches are the longest ones. They are four sets of {A : A′, B : B′, C : C′}, {A : A′, B : B′, C : C″}, {A : A′, B : B‴, C : C′}, and {A : A″, B : B‴, C : C′}. We cannot identify which one is the best among them, because all three nodes in (a) have their respective correspondences.

This algorithm is simple and easily understandable. However, it has some weak points.

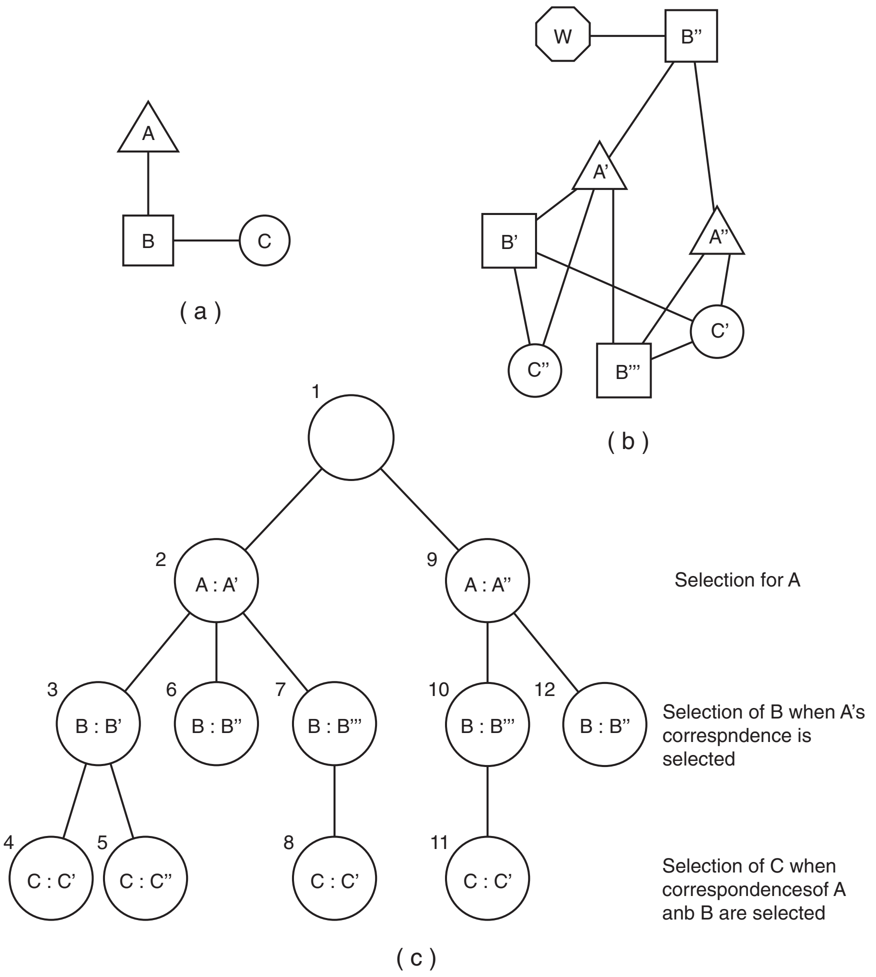


Fig. 9.4. Graph matching by search tree to find the best partial correspondences of the nodes in graph (a) to nodes in larger graph (b). (c) is their search tree.

In the above description, first, we listed up all selections for A, then checked out the selections for B, and so on. From the view point of the structure of the tree, we examined all the branches at the same layer in its breadth direction, then we proceeded to the next layer. Such the searching process is called the *breadth-first search*.

On the other hand, we have another strategy for the searching process. As in the order of the number attached at every branching point in (c), The possible matching is searched in the depth direction such as to select one A, then possible B and C. If we reach the limit before matching is completed, we go back one layer and select another branch of possibility. That is, we proceed in depth direction first in the tree, and it is called the *depth-first search*.

Another strategy is the *best-first search*, which is efficiently applicable for the case where, as was mentioned above, every correspondence between nodes has some weight evaluating the goodness for matching. The correspondence having the higher score is the more prioritized for the search. Also many smart algorithms to obtain the best matching incorporating the heuristics with the minimum computational cost have been proposed in the literatures.

If all the possible matchings need to be searched out, the total computational costs of the breadth and depth searches are same. However, we have many cases where at least one good candidate of matching is sufficient to be searched, we have a choice of better algorithms. But it must be noted that the practical computational cost highly depends on the structure of a given graph and hard to estimate beforehand.

Another improvement of the algorithms is the proper selection of the order of nodes to make correspondences. In the above example, the correspondences are checked in the order of A, B, and C. This order is known to effect on the total efficiency of the matching. Moreover, if no corresponding node for A of (a) exists in (b), for example, this tree search cannot go ahead even when B and C have their proper respective correspondences in (b). Usually, an observed image has some inevitable noise, and the graph extracted from such the image must have some wrong nodes and arcs. To treat with those graphs, special care for the evaluation of the matching other than the number of successful correspondences must be needed.

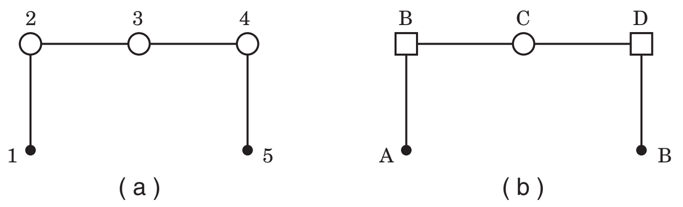


Fig. 9.5. Another example of two graphs to be examined their matching.

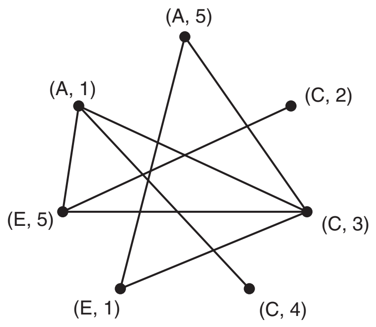


Fig. 9.6. Match graph for the graphs in Fig. 9.5.

9.3 Matching by Match Graph

9.3.1 Match graph

Here we show another example graphs to be matched in Fig. 9.5. For the graphs of (a) and (b), we find correspondences between member nodes of each graphs. Similar to the previous example, nodes labeled with ○ correspond to ○, □ to □, and △ to △, respectively in other graph. Now, however, we take account that some wrong information may be read out from images, and we do not limit within only these correspondences. Actually, for this example, these two graphs does not completely match including node labels.

If we select the best match by accounting the number of corresponding pairs of nodes, the algorithm described here is more effective.

First, for the graphs (a) and (b), we list up all the pairs of nodes which are correctly matched because of their common properties. That is, (A,1), (A,5), (C,2), (C,3), (C,4), (E,1), and (E,5) are the pairs to have correct possible correspondences. Then, we built a new graph having those pairs as its nodes. This new graph is shown in Fig. 9.6.

In this graph, node pairs which are mutually compatible are connected with arcs. For example, pairs of (A,1) and (A,5) are incompatible, because, if A corresponds to 1, then A cannot correspond to 5. So, between such the nodes of (A,1) and (A,5), the graph has no arc. On the other hand, when A corresponds to 1, it is possible that C corresponds to 3, and the graph has an arc between (A,1) and (C,3).

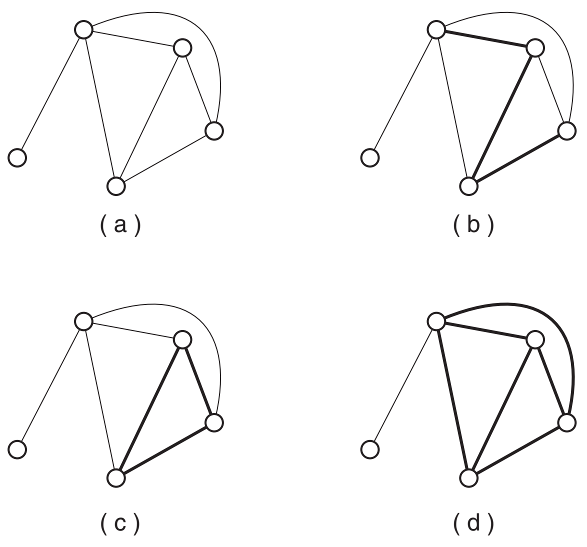


Fig. 9.7. For a graph (a), (b) an example of the connecting partial graph, (c) that of the complete partial graph (clique), and (d) that of the maximal clique.

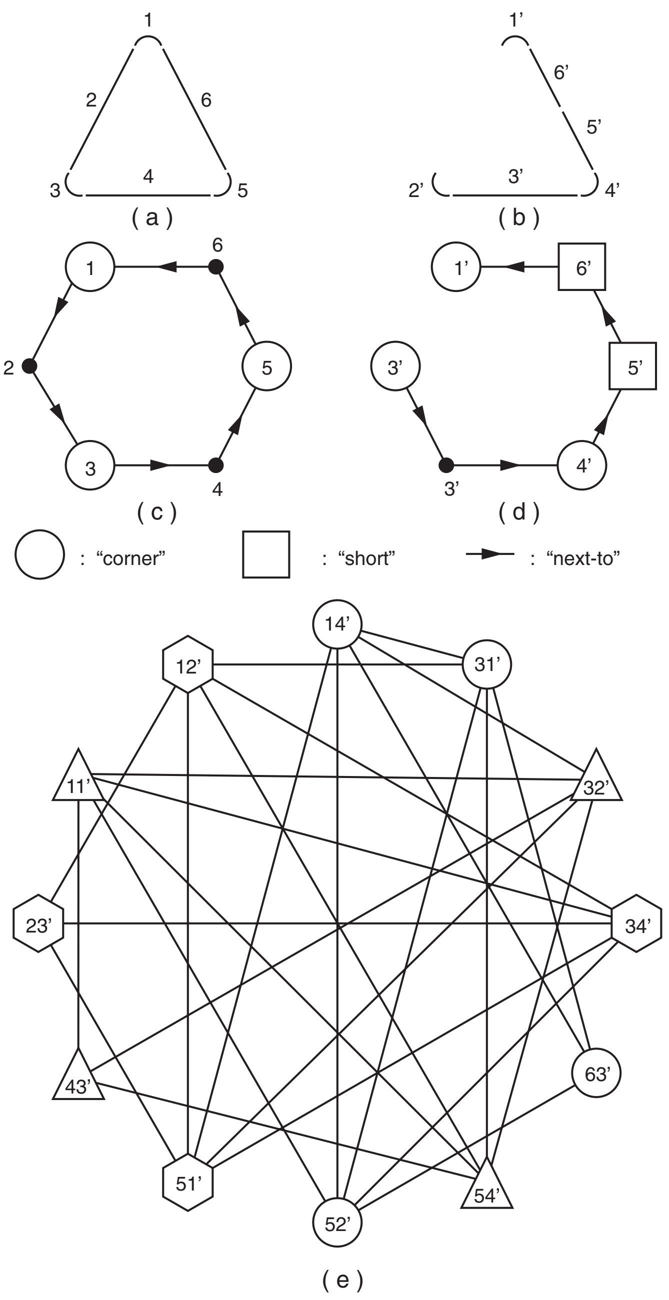


Fig. 9.8. Another example of pattern matching by clique method. (a) A standard pattern, and (b) an input pattern. (c) and (d) are their respective graph representations. (e) is the match-graph for these two graphs. Nodes of the largest three maximal cliques are shown in ○s, △s and hexagons, respectively.

In Fig. 9.6, we show the result in which all the compatible pairs are connected with arcs. This graph is called the *match graph* or the *association graph*. We can derive the best matching from this graph.

Before describing the procedures, we list up the definitions on a part of a graph.

9.3.2 Partial graph and cliques

For a given graph, a part of the graph is called the *partial graph* or *subgraph*.

We are given a graph shown in Fig. 9.7(a).

The partial graph of which all the nodes are connected to any other nodes through links of arcs is called the *connecting partial graph*. Its example is shown in the figure (b).

The connecting partial graph in which all the nodes have direct link of arcs between them is called the *complete partial graph* or the *clique*. An example is shown in (c).

When any node is added to a clique from its outside, and if it can no longer be the clique, it is called the *maximal clique*. Also an example is shown in (d).

Among the maximal cliques in a graph, one which has the largest number of nodes is called the *largest maximal clique*.

9.3.3 Matching by clique method

Let us apply the above definitions to the match graph.

The clique in the match graph represents a set of corresponding node pairs which are compatible in each other.

Therefore, the largest maximal clique of the match-graph represents the set of pairs of compatible correspondences and the number of its member is the largest. That is, *the largest maximal clique gives the best set of corresponding pairs of nodes for two graphs*.

The largest maximal clique for the graphs in Fig. 9.6 is the set of correspondences of {(A,1), (C,3), (E,5)} or {(A,5), (C,3), (E,1)}. From the observation of original two graphs in Fig. 9.5, both of these two matching are considered to be the best ones.

Another example is shown in Fig. 9.8.

We examine the matching and the correspondences between a standard pattern of Fig. 9.8(a) and an input pattern (b). (c) and (d) are each respective graph representations of these patterns. (e) is the match graph for these two graphs.

Three largest maximal cliques are found in the match-graph. Nodes of the largest maximal cliques are shown in ○'s, △'s and hexagons, respectively. It tells that three best matchings are inherently existing between these original two graphs, all of which have four pairs of correspondences. This is evident from the original patterns of (a) and (b) in this example.

The algorithm finding the largest maximal clique is understandable explicitly. However, its computational cost grows much faster than the polynomial order of the number of graph nodes. It is classified into the NP complete problems.

NOTES and REFERENCES

[1] This idea came from the article of Duda, R. O., Elements of pattern recognition, in *Adaptive, Learning and Pattern Recognition Systems — Theory and Applications* (Mendel, J. M. and Fu, K. S., Eds.), pp. 3–33, Mathematics in Science and Engineering Vol. 66, 1970.

[2] This problem is subject to a field of the statistics called the experimental design. Refer to a textbook of statistics for more details.

[3] The discussion here can be applied for the case of a communication where a signal of 0 or 1 is received from a noisy channel and it must be decided as 0 or 1. For this case, we also set some boundary to determine it. In the field of communication, from this meaning, such the curve in Fig. 2.12 is called the "receiver operating characteristic (ROC)" or "ROC curve."

[4] Chow, C. K. and Kaneko, T., Automatic boundary detection of the left-venticule from cineangiograms, *Computers and Biomedical Research*, **5**: 388–410 (1972).

[5] Otsu, N.: A threshold selection method from gray-level histograms, *IEEE Trans. Syst. Man, Cyber.*, **SMC-9**: 62–66 (1979).

[6] This figure is up-side-down of a "tree." Such the branching structure is called a tree structure. The up-most point is called the root, the branching points are called nodes, the line between nodes branches, and the edges leaves, analogous to a tree.

[7] Ledley, R. S., High-speed automatic analysis of biomedical pictures, *Science*, **146**: 216–223 (1964).

[8] The theories of 3D line drawing scene analysis described in the following were discussed in detail in: Barrow, H. G. and Tenenbaum, J. M., Interpreting line drawings as three-dimensional surfaces, *Artificial Intelligence*, **17**: 75–116 (1981).

[9] Huffman, D. A., Impossible objects as nonsense sentences, in *Machine Intelligence*, **6**: 295–323 (1971), Edinburgh University Press; and Clowes, M. B., On seeing things, *Artificial Intelligence*, **2**: 79–116 (1979).

[10] Roberts, L., Machine perception of three-dimensional solids, in *Optical and Electro-Optical Information Processing*, MIT Press (1965).

[11] Guzman, A., Decomposition of a visual scene into three-dimensional bodies, *AFIPS Proceedings Fall Joint Comp. Conf.*, **33**: 291–304 (1968).

[12] Waltz, D., Understanding line drawings of scenes with shadows, in *The Psychology of Computer Vision* (Winston, P., Ed.), pp. 19–91, McGraw-Hill (1975).

[13] Mackworth, A., Interpreting pictures of polyhedral scenes, *Artificial Intelligence*, **4**: 121–137 (1973).

[14] Kanade, T., Recovery of the three-dimensional shape of an object from a single view, *Artificial Intelligence*, **17**: 409–460 (1981).

[15] Haralick, R. M., Determining camera parameters from the perspective projection of a rectangle, *Pattern Recognition*, **22**: 225–230 (1989).

[16] Grimson, W. E. L., *Object Recognition by Computer*, MIT Press (1990).

[17] Forsyth, D., *et al.*, Invariant descriptors for 3-D object recognition and pose, *IEEE Trans. Pattern Analysis Machine Intelligence*, **PAMI-13**: 971–991 (1991).

[18] Lamdan, Y., Schwaltz, J. T., and Wolfson, H. J., Affine invariant model-based object recognition, *IEEE Trans. Robotics and Automation*, **RA-6**: 578–589 (1990).

[19] Lamdan, Y. and Wolfson, H. J., Geometric hashing: A general and efficient model-based recognition scheme, in *Proc. Second International Conference on Computer Vision*, pp. 238–249 (1988).

[20] If the object is planar, four points are sufficient.

Index