Image-based pattern recognition principles

Outline

- Introduction
- 2D Matched Filter
- Image Registration
- Bayes Statistical Classifier
- Neural Networks
- Syntactic Recognition
- Face Recognition

Introduction

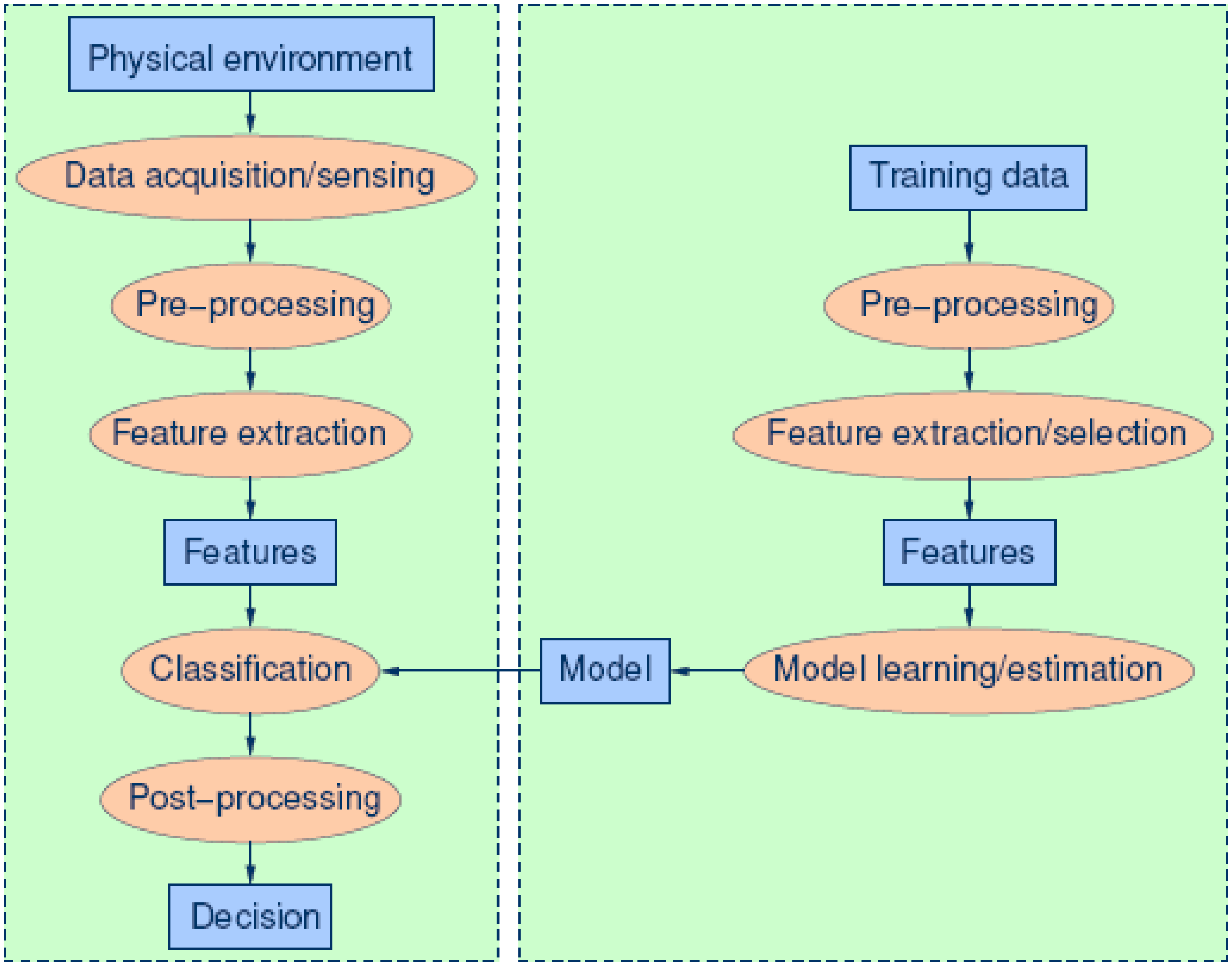
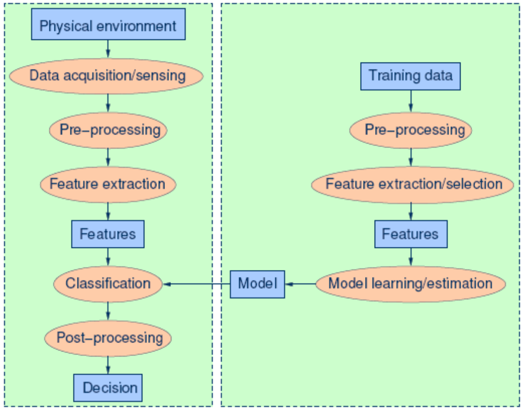


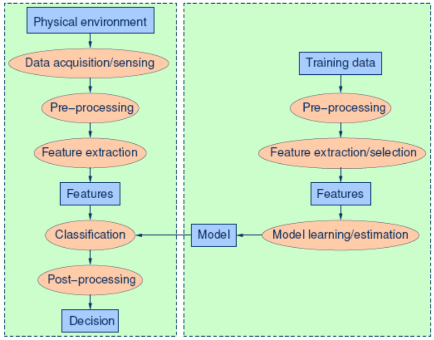
Fig.1 Basic components of a pattern recognition system[8]

Introduction

- **Data acquisition and sensing**
- **Pre-processing**
  - Removal of noise in data.
  - Isolation of patterns of interest from the background.
- **Feature extraction**
  - Finding a new representation in terms of features.

    (Better for further processing)





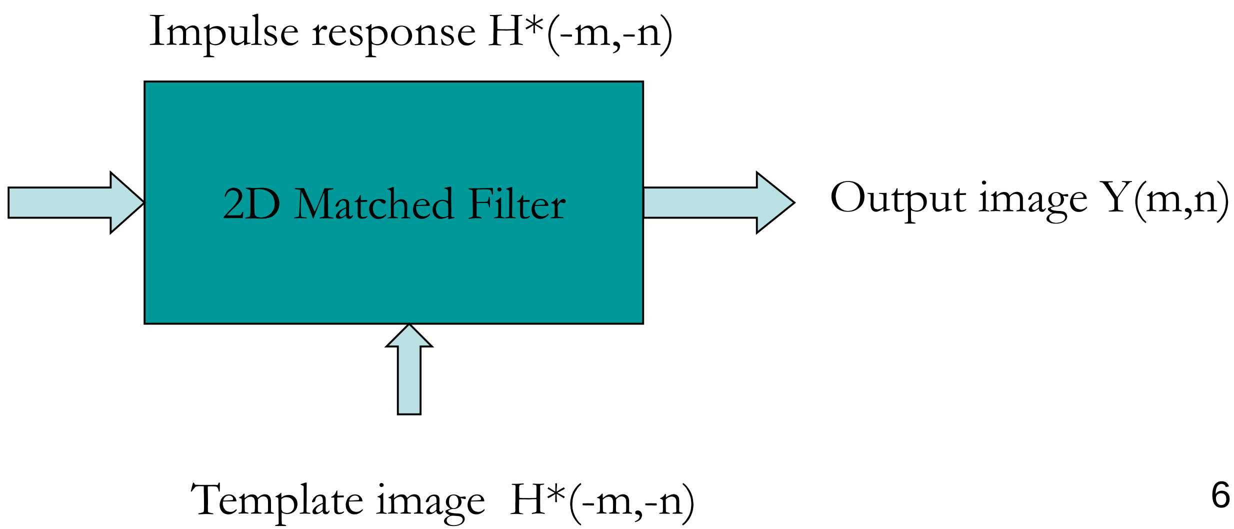
Introduction

- **Model learning and estimation**
  - Learning a mapping between features and pattern groups.
- **Classification**
  - Using learned models to assign a pattern to a predefined category
- **Post-processing**
  - Evaluation of confidence in decisions.
  - Exploitation of context to improve performances.

2D Matched Filter

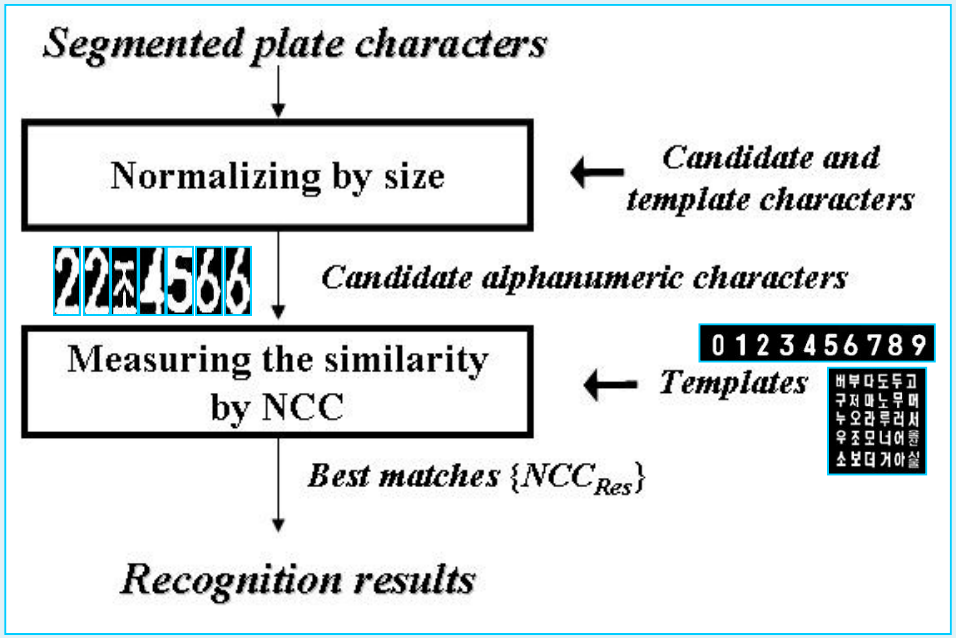
- **Functionality**
  - Degrading the noise effect.
  - Computing the similarity of two objects.
    (Template matching for images)
- **Functional block**

Input image I(m,n)



Algorithm description

Character recognition



The proposed character recognition scheme

(a)

(b)

Template used for pattern matching: *(a)* 10 prototypes for the Korean plate numbers and *(b)* 30 prototypes for the Korean plate characters.

$$NCC_{ct} = \frac{\sum_{x=0}^{m-1} \sum_{y=0}^{n-1} (c - \overline{c}) \left(t - \overline{t}\right)}{\sqrt{\sum_{x=0}^{m-1} \sum_{y=0}^{n-1} (c - \overline{c})^2 \left(t - \overline{t}\right)^2}} \tag{5.1}$$

where NCC_{ct} is the correlation coefficient. The candidate plate character recognition process is based on the value of the correlation coefficient. If the value of the correlation coefficient execeeds a threshold set by the user, then the similarity measure is large enough and the input character can be assumed to present. Finally, a box on the target character

Processing example

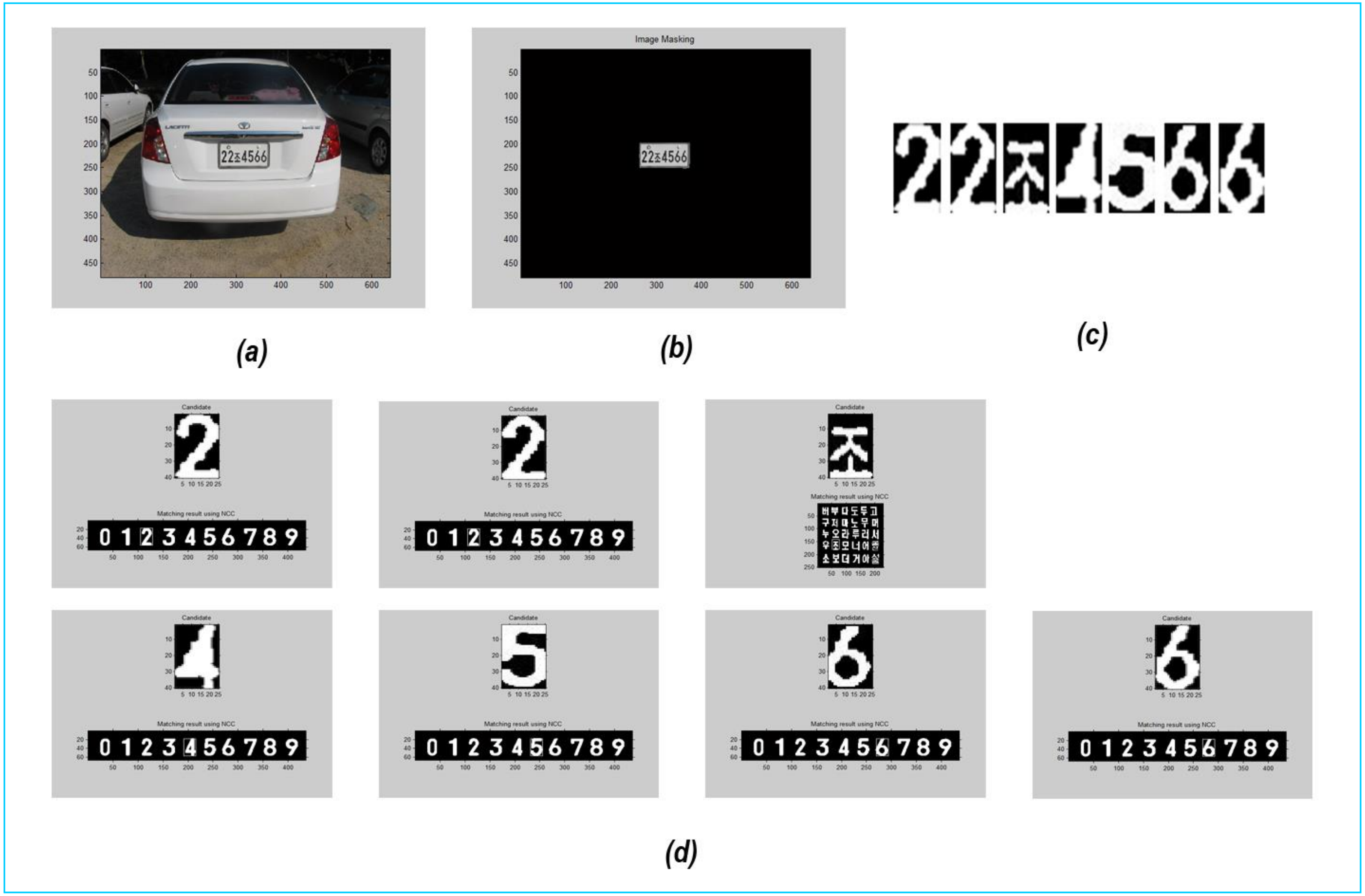


Illustration of LP segmentation and alphanumeric characters recognition: *(a)* an LP image, *(b)* extracted candidate region, *(c)* equal-sized candidate alphanumeric characters and *(d)* candidate alphanumeric characters measuring the similarity by NCC.

Processing example

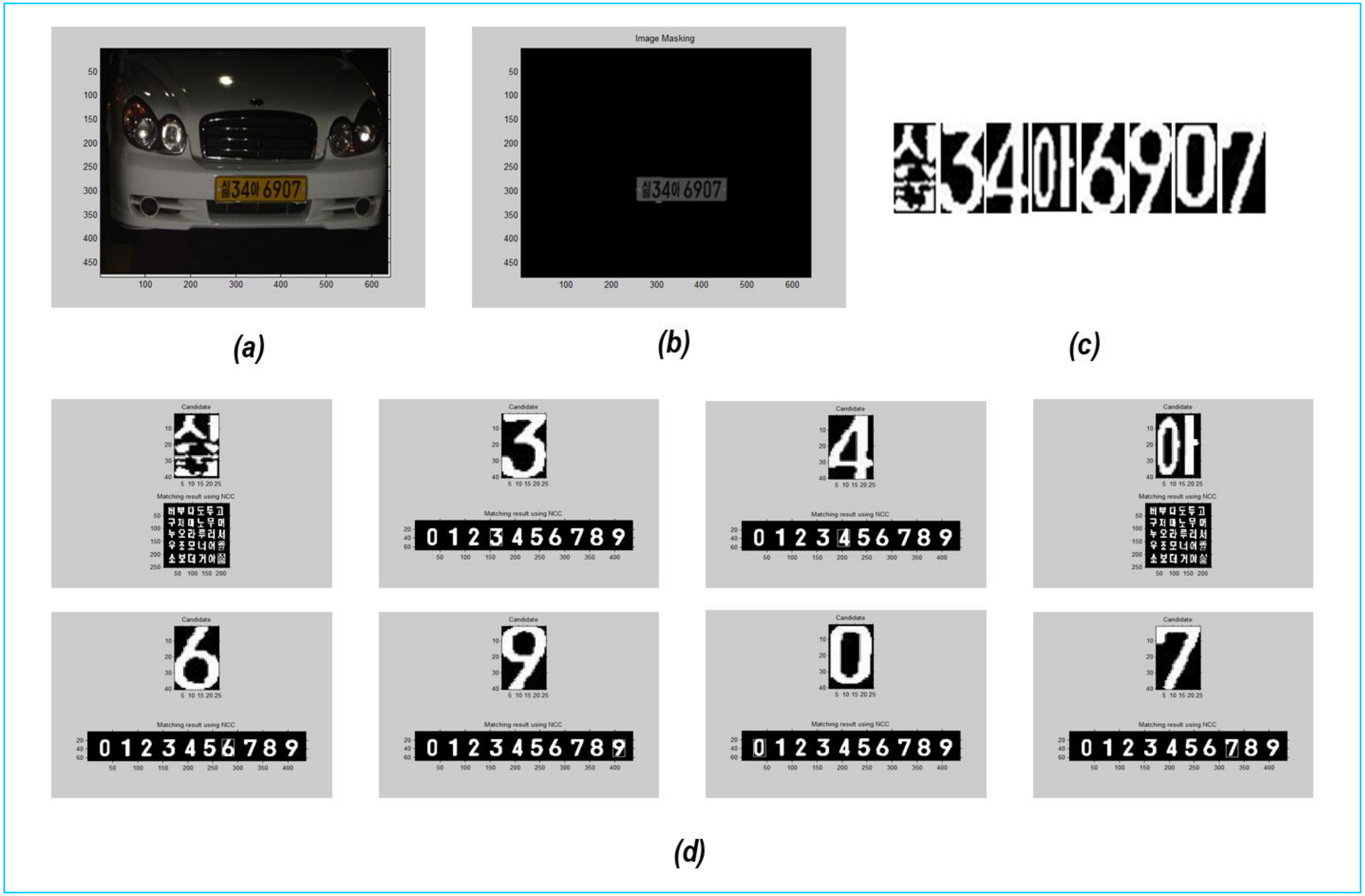


Illustration of LP segmentation and alphanumeric characters recognition: *(a)* an LP image, *(b)* extracted candidate region, *(c)* equal-sized candidate alphanumeric characters and *(d)* candidate alphanumeric characters measuring the similarity by NCC.

Processing example

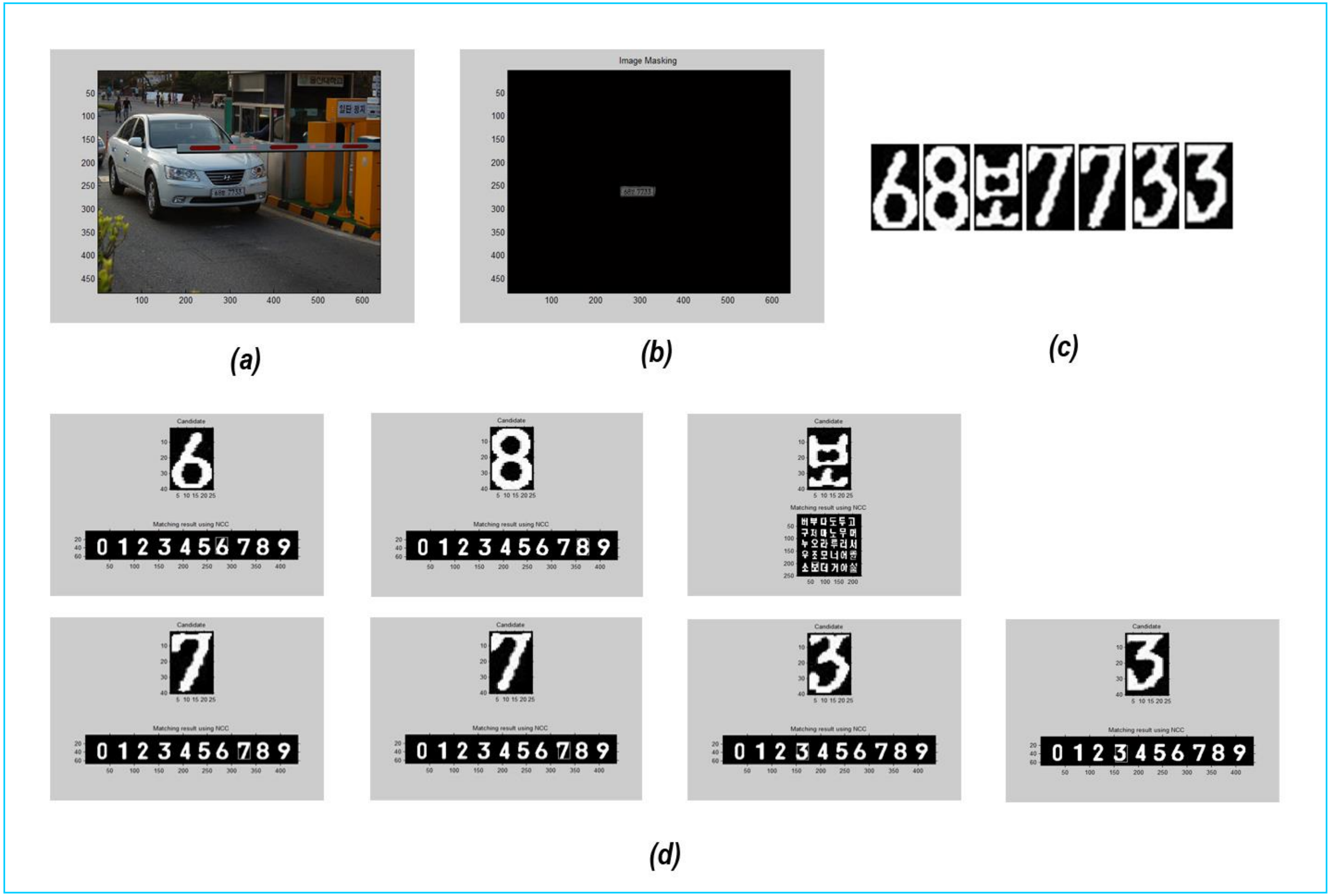


Illustration of LP segmentation and alphanumeric characters recognition: (a) an LP image, (b) extracted candidate region, (c) equal-sized candidate alphanumeric characters and (d) candidate alphanumeric characters measuring the similarity by NCC.

CHAIN CODE

Freeman Chain Code

- Two processes include in this phase:
  - **Boundary extraction:** For image segmentation is performed by finding boundaries between objects
  - Steps are:
    - Mark potential edge points by finding discontinuities in features
    - Threshold the results
    - Merge edge segments into boundaries via edge linking
  - **Character segmentation**

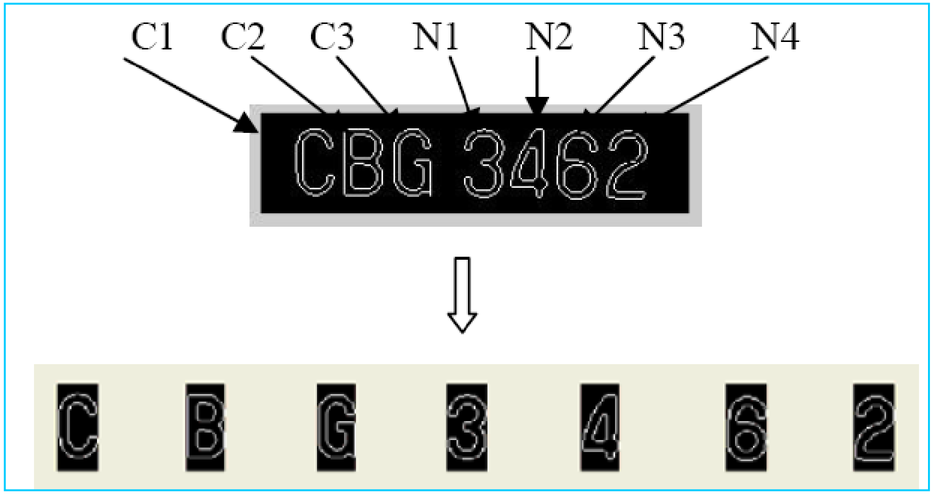


Fig. 4. Boundary image and segmented character regions

Chain code derivation

- The algorithm for extracting chain codes for 8-connected boundaries is as follows:
  - Find the pixel in the object that has the leftmost value in the topmost row
  - Define a variable *dir* = *7*

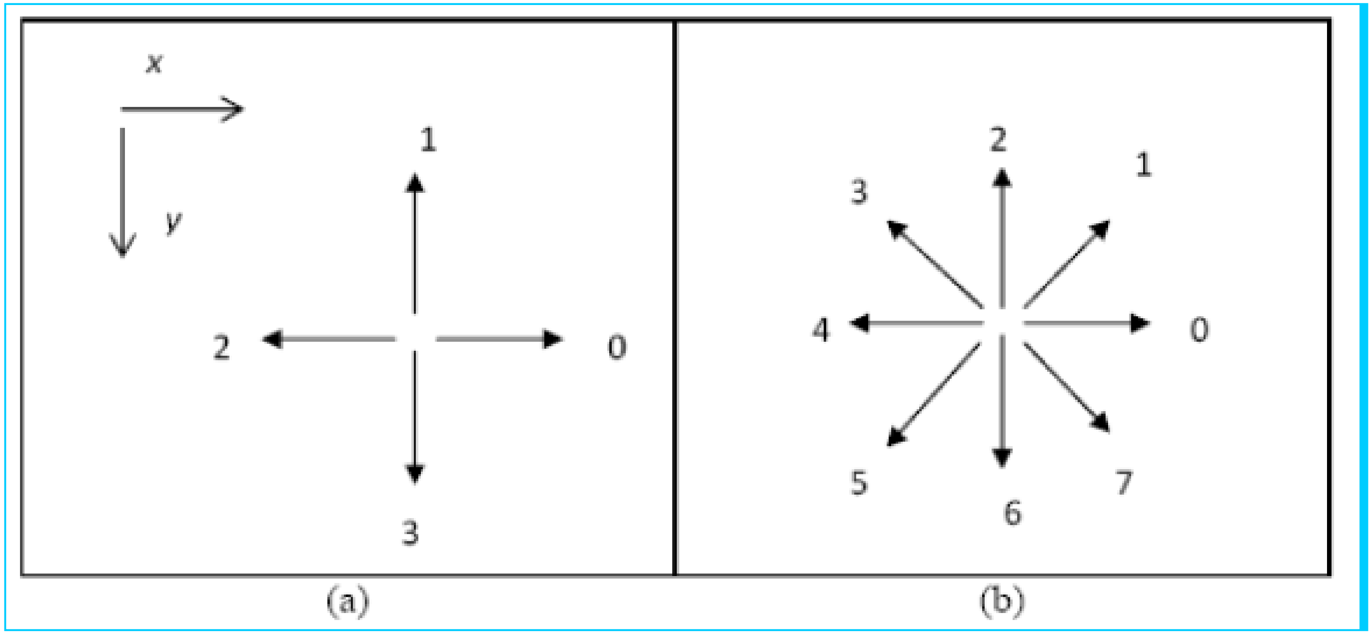


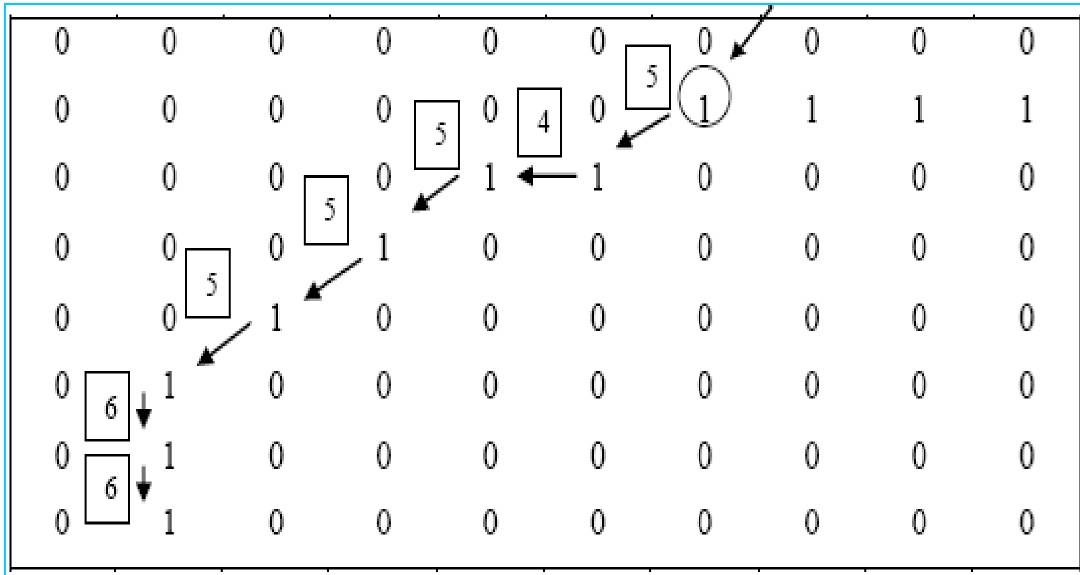
Fig. 5. Direction numbers for (a) 4-directional and (b) 8-directional chain code

Cont.

- Traverse the 3x3 neighborhood of the current pixel in a counter-clockwise direction
  - **Even**(0,2,4,6): dir + 7 % 8
  - **Odd** (1,3,5,7): dir + 6 % 8

dir	0	1	2	3	4	5	6	7
dir + 7 (mod 8)	7	0	1	2	3	4	5	6
dir + 6 (mod 8)	6	7	0	1	2	3	4	5

- First foreground pixel will be new boundary element. Stop when the current boundary element P_n is equal to the second element P_1 and the previous boundary pixel P_{n-1} is equal to first boundary element P_0



Columns 1 through 19

5 4 4 5 4 5 4 5 5 6 6 5 6 5 6 6 6 6 6

Columns 20 through 38

6 6 6 6 6 6 6 6 6 6 6 6 6 6 7 6 6 6 6

Columns 39 through 57

6 7 6 7 7 7 7 7 0 0 7 0 0 0 0 0 0 1 0

Columns 58 through 76

1 0 1 1 1 2 1 2 2 1 2 2 2 3 4 4 4 4 5

Columns 77 through 95

6 6 6 5 5 5 5 4 4 4 4 4 3 3 3 2 3 2 2

Columns 96 through 114

2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

Columns 115 through 133

2 1 1 1 1 0 0 0 7 0 7 7 7 6 7 6 7 0 0

Columns 134 through 152

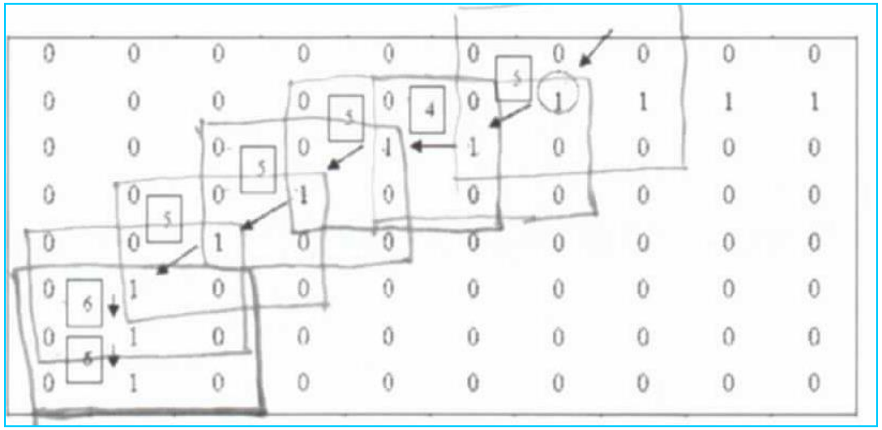
0 1 2 2 2 2 3 2 2 3 3 3 3 4 3 4 4 4 3

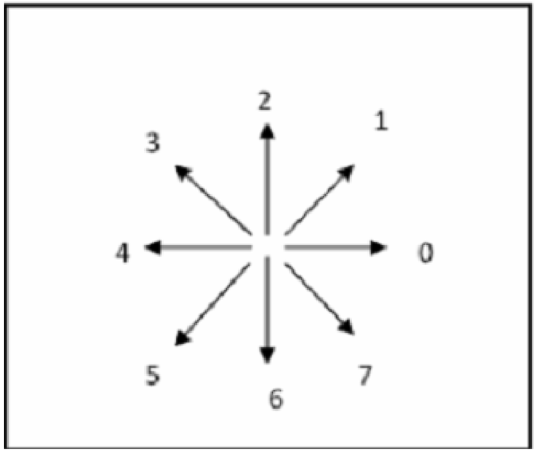
Columns 153 through 154

4 4

Fig. 7. The initial location and the direction to derive chain codes. And Fig. 8. the chain code extracted from the boundary image of character "C"

Cont.





dir = 7

(0, 2, 4, 6) even : dir + 7 % 8

(1, 3, 5, 7) odd : dir + 6 % 8

5 ← 7 + 6 % 8 = 5

4 ← 5 + 6 % 8 = 3

5 ← 4 + 7 % 8 = 3

5 ← 5 + 6 % 8 = 3

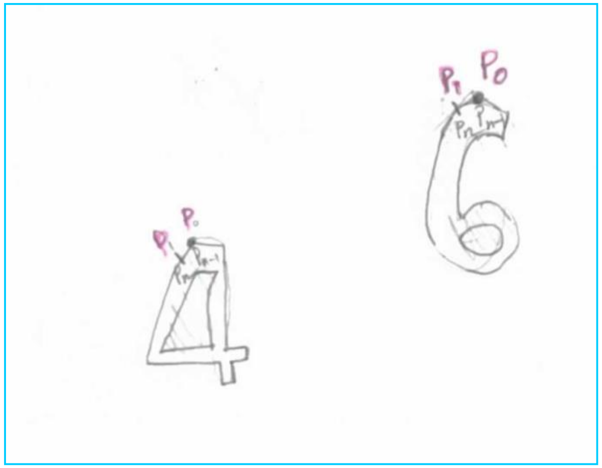
5 ← 5 + 6 % 8 = 3

6 ← 5 + 6 % 8 = 3

6 ← 6 + 7 % 8 = 5

Recognition

- **Character recognition has been done by using**
  - The list of chain codes derived for each character
  - Calculating the total number of each code direction contained in the list of chain code *(how many orders coordinates observe is important part in the rate of recognition)*
  - Total number of each code direction is used as a guide to recognize the characters



Eight-direction code vector

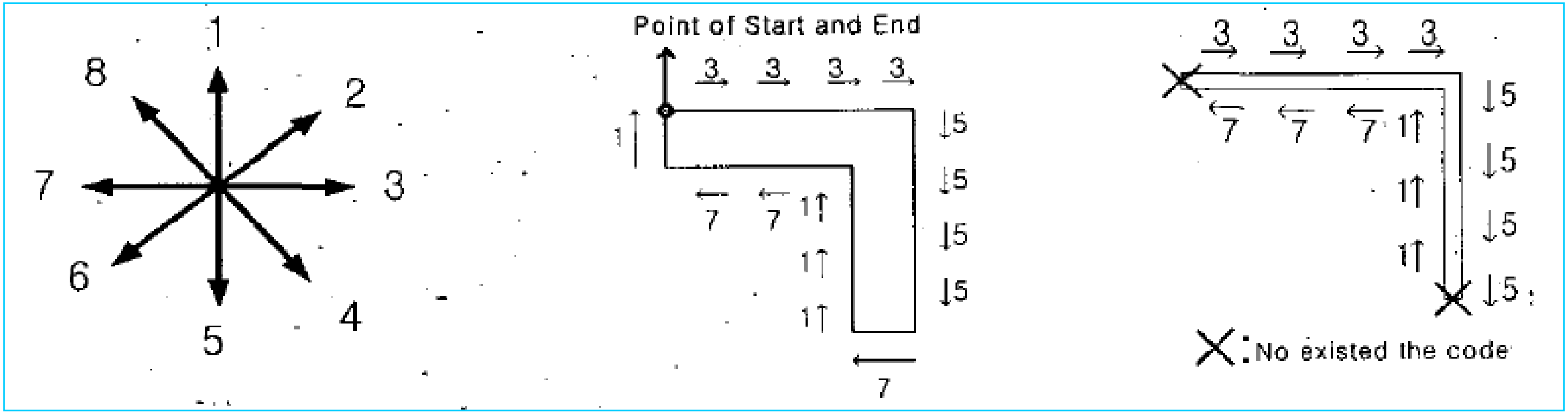


Fig. 6. Formation of closed region by eight-direction code vector : (a) eight-direction code vector , (b) pattern formation of closed region, and (c) no existed code vector

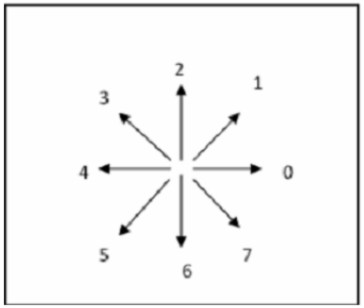


Image Registration

- What is Image Registration?
  - ◆ Aligning images correctly to make systems have
    better performance.
- Misregistration between images
  - ◆ Translational differences
  - ◆ Scale differences
  - ◆ Rotational differences

Image Registration : Detecting Translational Parameter

- **Spatial domain approach**

  -Normalized 2D matched filter

  - The highest output value is the best translational

    position.

- **Frequency domain approach**

  -Phase correlation method

Image Registration : Detecting Translational Parameter

- **Phase correlation method**
  - $F_2(x, y) = F_1(x-x0, y-y0)$

$$\Big\Downarrow \text{ F.T.}$$

$$F_2(w_x, w_y) = F_1(w_x, w_y)\, exp\{-i(w_x x0 + w_y y0)\}$$

  - Cross-power spectrum

$$\Big\Downarrow \text{ I.F.T.}$$

Image Registration : Detecting Scale and Rotational Parameter

- **Detecting rotational parameter**

F.T.

❶ Taking magnitudes both sides

❷ Representing in polar form

Image Registration : Detecting Scale and Rotational Parameter

- Detecting scale parameter

$$F_2(x, y) = F_1(ax, by)$$

$\Big\Downarrow$ F.T.

$\Big\Downarrow$ Frequency variables to a logarithmic scale

Bayes Statistical Classifiers

- **Consideration**
  - Randomness of patterns
- **Decision criterion**

Pattern x is labeled as class w_i if

L_{ij} : Misclassification loss function

$p(\mathbf{x}/w_i)$: P.d.f. of a particular pattern x comes from class w_i

$P(w_i)$: Probability of occurrence of class w_i

Bayes Statistical Classifiers

- Decision criterion :

  Given L_{ij} is symmetrical function

  - Posterior probability decision rule

$d_j(\boldsymbol{x})$: decision functions

Pattern $\boldsymbol{x}$ classifies to class j if $d_j(\boldsymbol{x})$ yields the largest value

Bayes Statistical Classifiers

- **Advantages**
  - Optimization in minimizing the total avarage loss

    in miscalssification.
- **Disadvantages**
  - Both $P(w_j)$ and $p(\mathbf{x}/w_j)$ must be known in advance.

    Estimation is required.

    Performance highly depends on the assumption of

    the distributions ($P(w_j)$ and $p(\mathbf{x}/w_j)$)

Neural Networks

- **What is Neural Networks?**
  - Ideas stem from the operation of human neural
    networks.
  - Networks of interconnected nonlinear computing
    elements called nurons.

Neural Networks

- Perceptron : two classes model

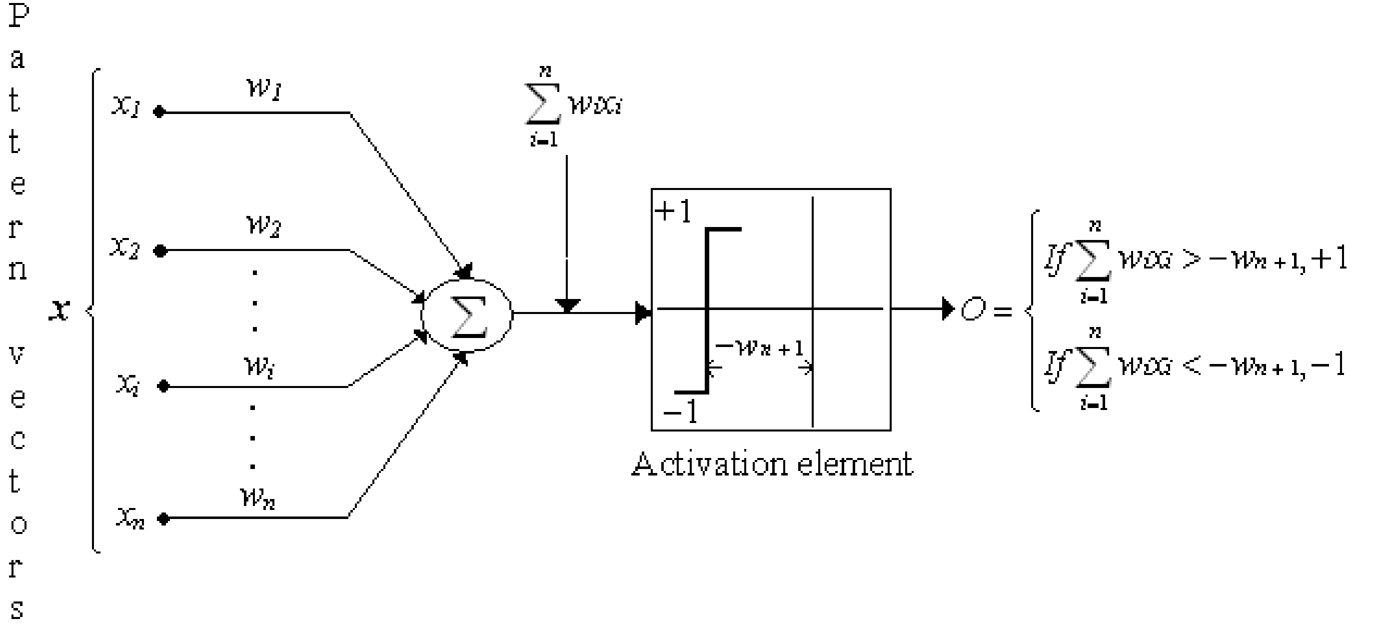


Fig.2 Structure of perceptron

Neural Networks : Multilayer Feedforward Neural Networks

- Basic structure

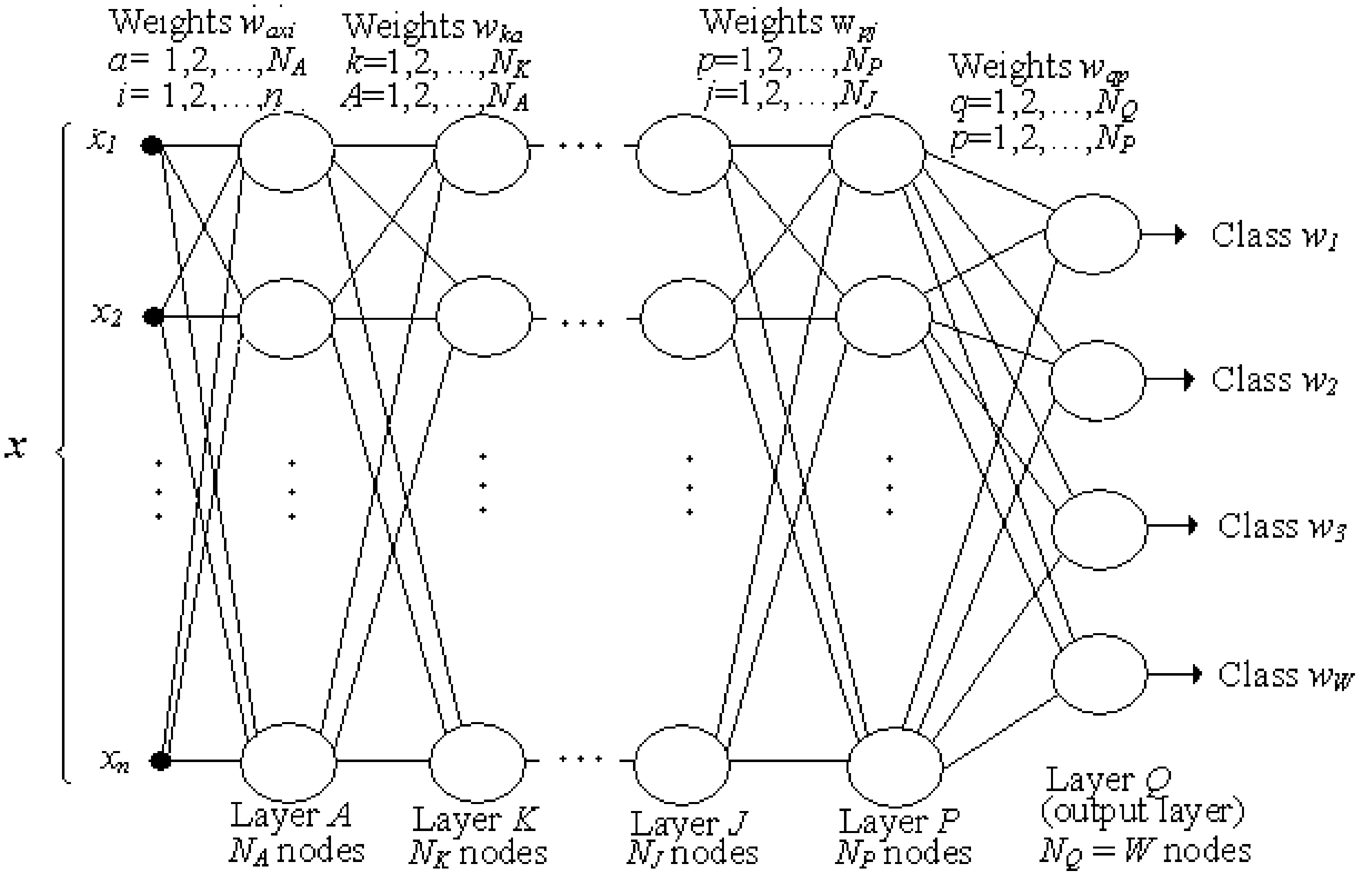


Fig.3 Structure of multilayer feedforward neural networks

Neural Networks : Multilayer Feedforward Neural Networks

- **Training algorithm : back propagation**
  - Sigmoid activation function

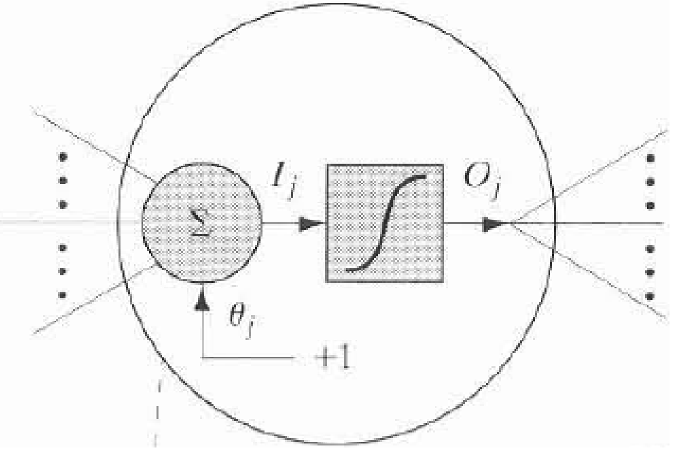


Fig.4 Blowup of a neuron[1]

Neural Networks : Multilayer Feedforward Neural Networks

1. Initialization

Assigning an arbitrary set of weights throughout the network (not equally).

2. Iterative step

a. Computing O_j for each node by using training vector, then generating the error terms for output δ_q, where

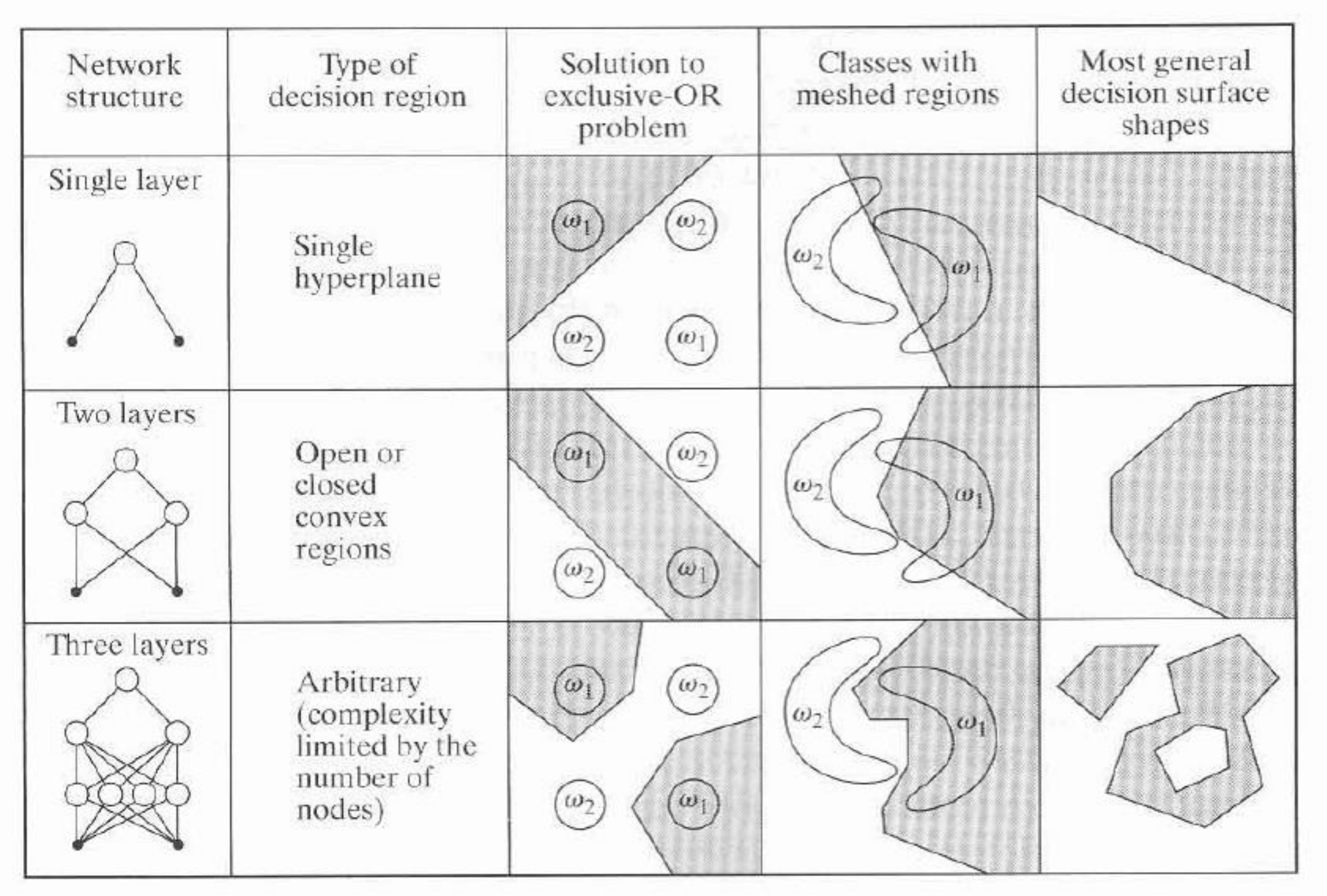
, r_q is the desired response.

b. Backward passing appropriate error signal is passed to each node and the corresponding weight changes are made.

Neural Networks

- **Decision surface complexity**

Table2 : Decision surface complexity of multilayer feedforward neural networks[1]



Network structure	Type of decision region	Solution to exclusive-OR problem	Classes with meshed regions	Most general decision surface shapes
Single layer	Single hyperplane	ω_1 ω_2 ω_2 ω_1	ω_2 ω_1	
Two layers	Open or closed convex regions	ω_1 ω_2 ω_2 ω_1	ω_2 ω_1	
Three layers	Arbitrary (complexity limited by the number of nodes)	ω_1 ω_2 ω_2 ω_1	ω_2 ω_1	

Syntactic Recognition

- Concerning the structural relation.
- Patterns represent in combinations of primitives.

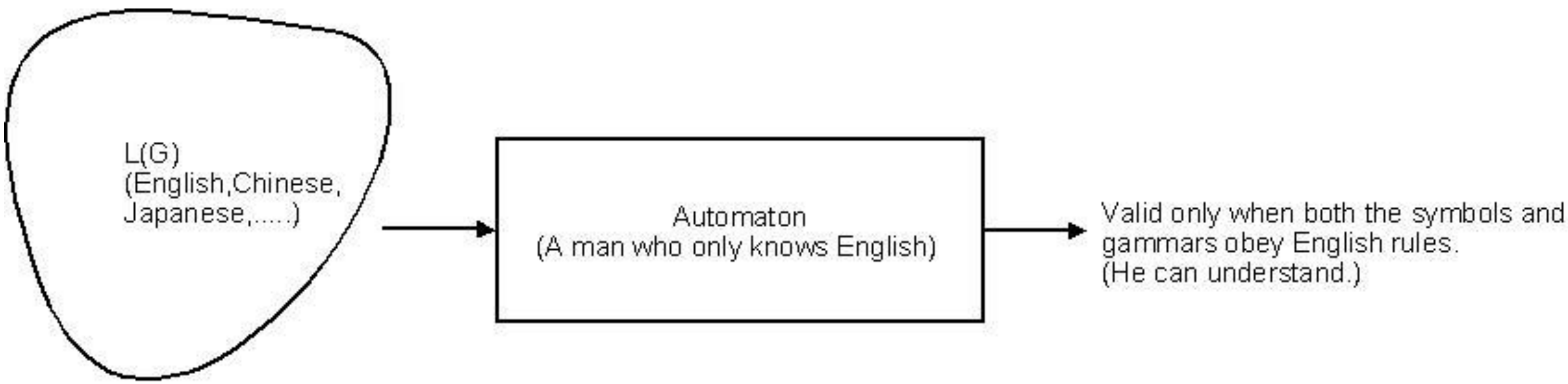


Fig.5 Conceptual diagram of syntactic recognition

Syntactic Recognition : String Case

- Input to the automata are unknown sentences

  generated by the corresponding grammars

  respectively.

  - The grammar $G = (N, \Sigma, P, S)$

    N is a finite set of variables called *nonterminals*,

    Σ is a finite set of constants called *terminals*,

    P is a set of rewriting rules called *productions*, and

    S in N is called the *starting symbol*

Syntactic Recognition : String Case

- An example

  $N=\{A,B,S\},\Sigma=\{a,b,c\}$

  $P=\{S\rightarrow aA,\ A\rightarrow bA,\ A\rightarrow bB,\ B\rightarrow C\}$

  $S\rightarrow aA\rightarrow abA\rightarrow abbA\rightarrow \ldots. \rightarrow abbbbbc$

  $L(G)=\{ab^nc|n\geqq 1\}$

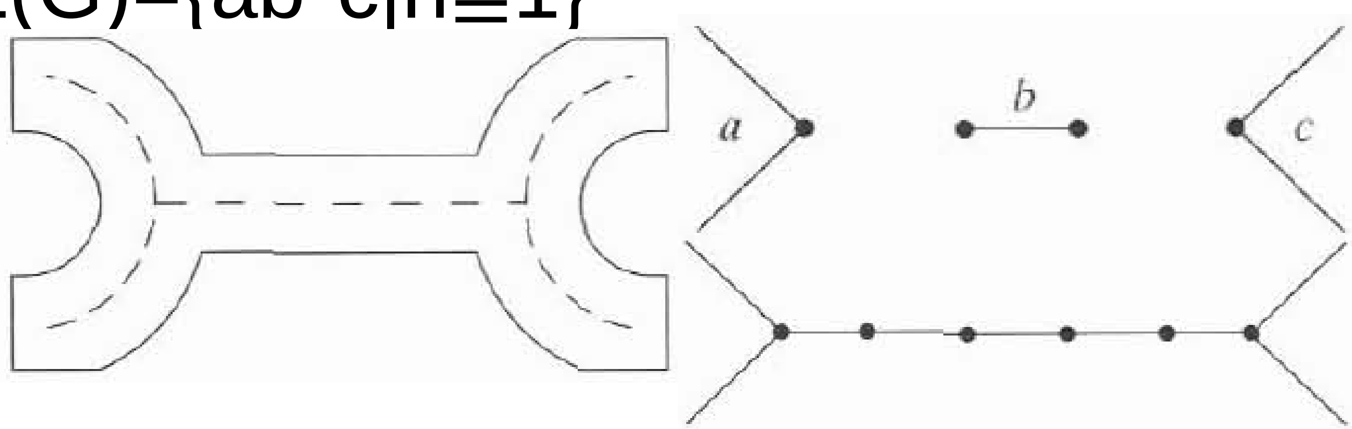


Fig.6 An example of string language[1]

Syntactic Recognition : String Case

- The finite automata $A_f = (Q, \Sigma, \delta, q_0, F)$

  Q is a finite, nonempty set of *states*,

  Σ is a finite input *alphabet*,

  δ is a *mapping* from $Q \times \Sigma$ into the collection of all

  subsets of Q,

  q_0 is the *starting state*, and

  F is a set of *final states*.

Syntactic Recognition : String Case

- A simple automaton

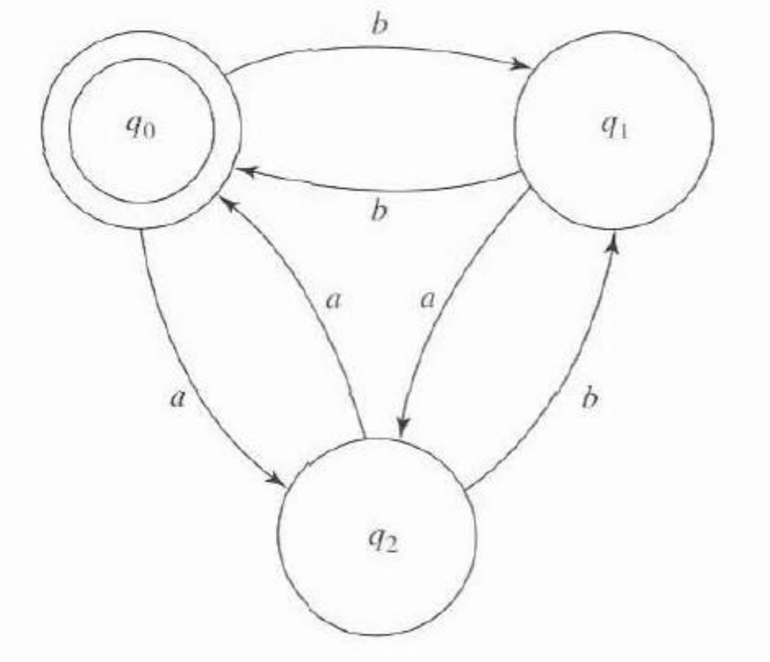


Fig.7 State machine of the automaton[1]

$A_f = (Q, \Sigma, \delta, q0, F)$

$Q=\{q_0,q_1,q_2\}$

$\Sigma=\{a, b\}$

$F=q_0$

$\delta(q_0,a)=\{q_2\}$

$\delta(q_0,b)=\{q_1\}$

$\delta(q_1,a)=\{q_2\}$

$\delta(q_1,b)=\{q_0\}$

$\delta(q_2,a)=\{q_0\}$

$\delta(q_2,b)=\{q_1\}$

Invalid input string : bababbb

Valid input string : aaabbbb

Syntactic Recognition : String Case

- Conversion between regular grammar and
  corresponding automaton states.

$G = (N, \Sigma, P, S)$ ⟹ $Af = (Q, \Sigma, \delta, q_0, F)$

$X_0 \equiv S$ $Q=\{q_0, q_1, \ldots, q_n, q_{n+1}\}$

$N=\{X_0 \sim X_n\}$

The mappings in δ are obtained by using the following two rules, for a in Σ ,and each i and j ,with $0 \leqq i \leqq n$, $0 \leqq j \leqq n$,

1.If $X_i \rightarrow aX_j$ is in P, then $\delta(q_i, a)$ contains q_j.

2.If $X_i \rightarrow a$ is in P, then $\delta(q_i, a)$ contains q_{n+1}.

Syntactic Recognition : String Case

- Grammars are not known in advance, we need
  to learn the automata from sample patterns.
- An unknown grammar *G* and a finite sets of samples R^+

$h(z, R^+, k) = \{w \mid zw \text{ in } R^+, |w| \leqq k\}$, z belongs to Σ^*

$Q = \{q \mid q = h(z, R+, k) \text{ for } z \text{ in } \Sigma^*\}$

$\delta(q, a) = \{q' \text{ in } Q \mid q' = h(za, R^+, k), \text{ with } q = h(z, R^+, k)\}$

$q_0 = h(\lambda, R^+, k)$

$F = \{q \mid q \text{ in } Q, \lambda \text{ in } q\}$

Syntactic Recognition : String Case

- An example of learning automaton structure

  from a given sample set

  $R^+ = \{a, ab, abb\}$ (k=1)

- Determining h(z, R+, k)

λ is a empty string set

z= λ $h(\lambda, R^+, 1) = \{w \mid \lambda w \text{ in } R^+, |w| \leqq 1\}$
$= \{a\}$
$= q_0$

z= a $h(a, R^+, 1) = \{w \mid aw \text{ in } R^+, |w| \leqq 1\}$
$= \{\lambda, b\}$
$= q_1$

z= ab $h(ab, R^+, 1) = \{w \mid abw \text{ in } R^+, |w| \leqq 1\}$
$= \{\lambda, b\}$
$= q_1$

z= abb $h(abb, R^+, 1) = \{w \mid abbw \text{ in } R^+, |w| \leqq 1\}$
$= \{\lambda\}$
$= q_2$

Syntactic Recognition : String Case

❷ Obtaining mapping function

$Q=\{q_0,q_1,q_2,q_3\}$,q3 denotes empty set state

$h(\lambda,R+,1) = q0$, $z=\lambda$
$\delta(q_0,a)= h(\lambda a,R^+,1)= h(a,R^+,1)=q_1$
$\delta(q_0,b)= h(\lambda b,R^+,1)= h(b,R^+,1)=q_3$

$h(a,R^+,1)= h(ab,R^+,1) =q_1$
$\delta(q_1,a)= h(aa,R^+,1)= h(aba,R^+,1)=q_3$

$\delta(q_1,b) \supseteq h(ab,R^+,1)=q_1$ $\delta(q_1,b) \supseteq h(abb,R^+,1)=q_2$
$\delta(q_1,b) =\{q_1,q_2\}$

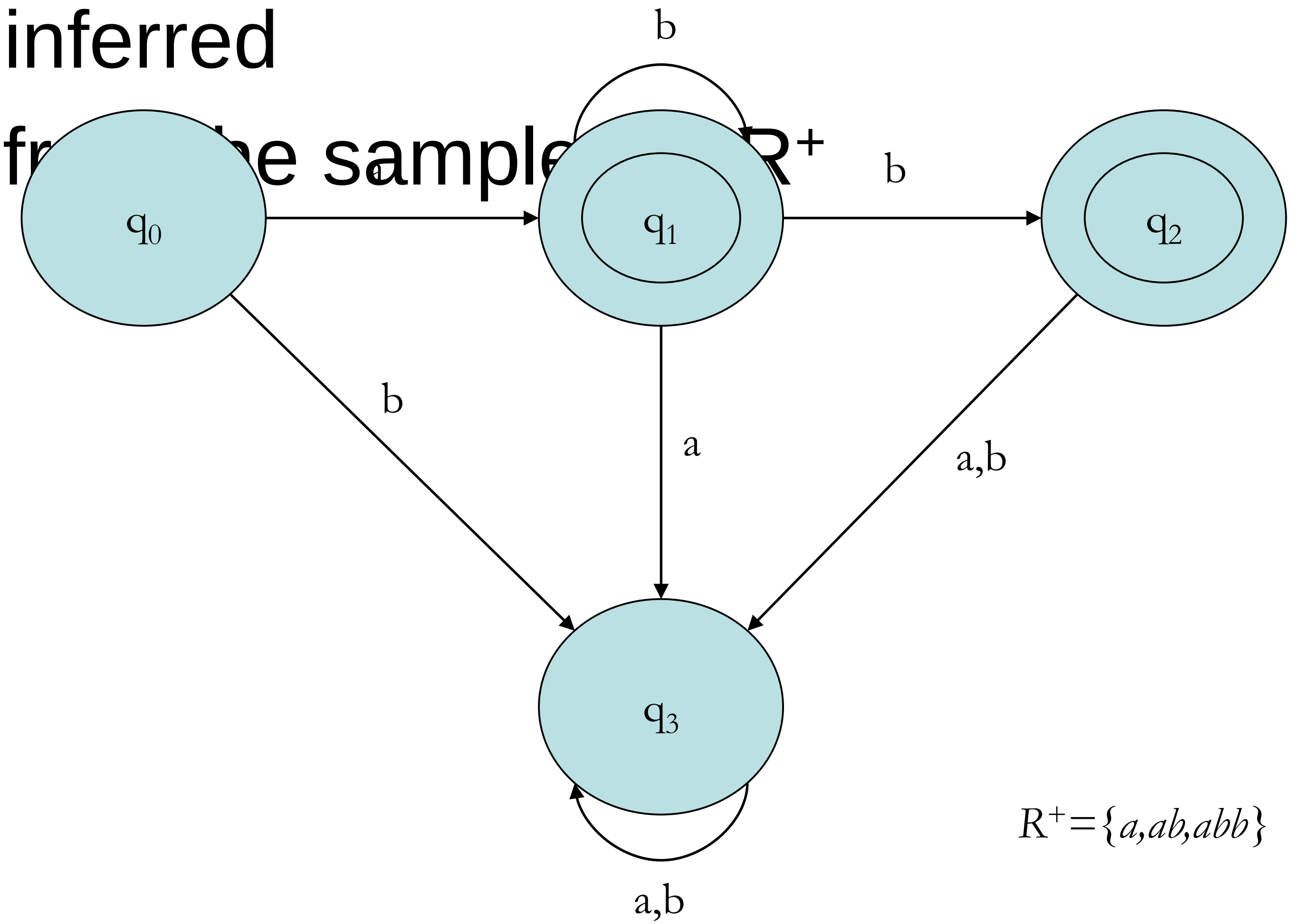
$\delta(q_2,a) =\delta(q_2,b) =\delta(q_3,a) =\delta(q_3,b) =q_3$

❸ Obtaining final state F

$q1=\{\lambda, b\}$ $q2=\{\lambda\}$
$F=\{q1, q2\}$

Syntactic Recognition : String Case

- State diagram for the finite automaton inferred
  from the sample R^+



$R^+=\{a,ab,abb\}$

Syntactic Recognition : String Case

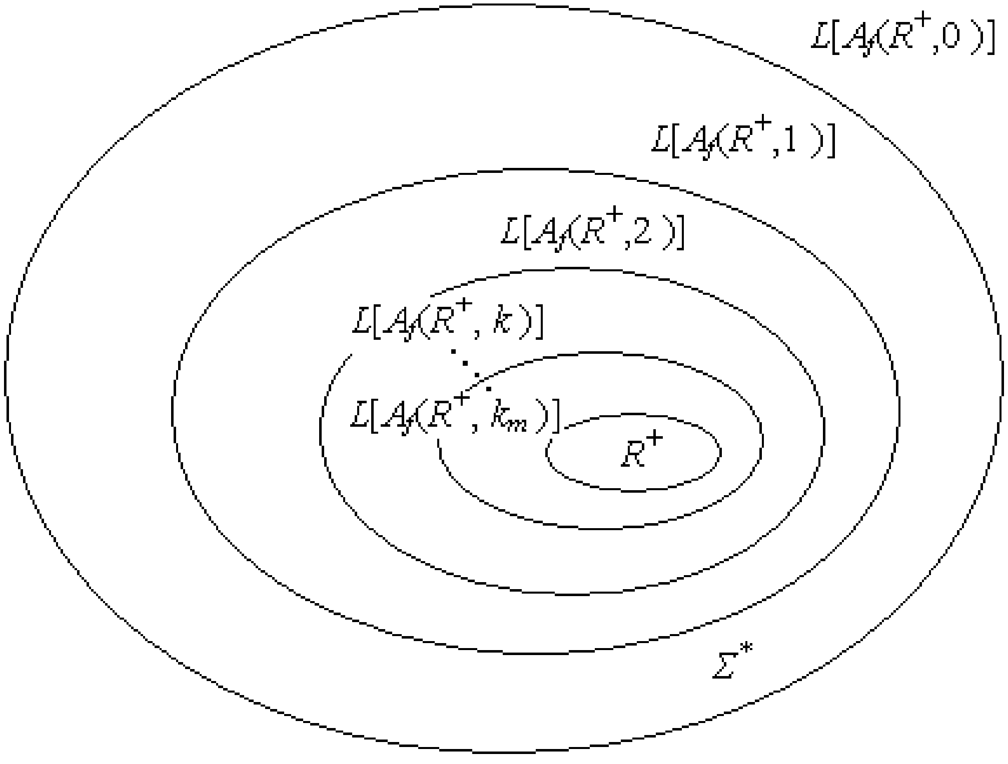


Fig.8 Graphic relation between k and $L[A_f(R^+, k + 1)]$

Face Recognition

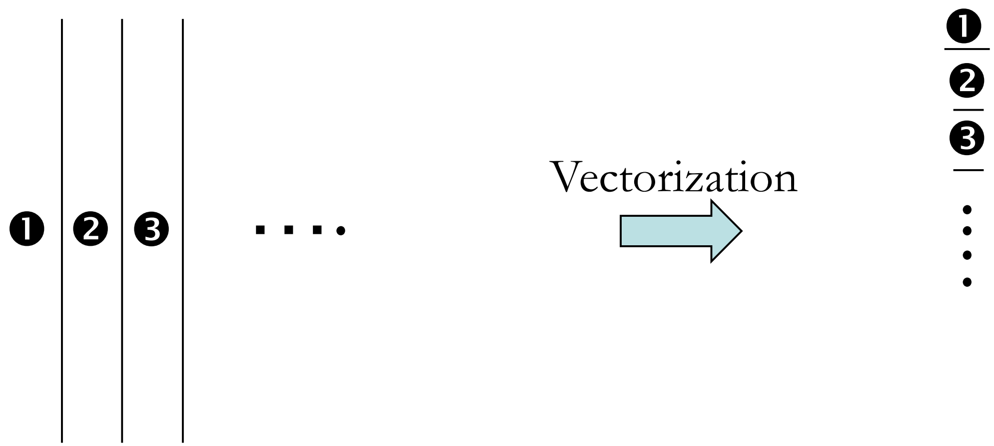
- User-friendly pattern recognition application
- Weakness of face recognition
  - Illumination problems
  - Pose problems(profile or frontal view)

Fig.9 Examples of illumination problems[9]

Face Recognition : Eigenspace-Based Approach

- Eigenspace-based approach
  - ◆ A holistic approach
  - ◆ Reducing the high dimensionality problem , and large computational complexity.

A face image of size 200×180



Face Recognition : Standard Eigenspace-Based Approach

- Standard Eigenspace-based approach
  - Given a set of training face images, computing the eigenvectors of the distribution of face images within the entire image space .(PCA method)

Size of $N^2 \times N^2$

Size of $N^2 \times M$

Length of N^2

Γ_n : face vectors

Ψ : Mean vector

C : Covariance matrix of training set

M : Number of training face images

Face Recognition : Standard Eigenspace-Based Approach

- C is too big! We can reduce the eigenvlue value problems from order of $N^2 \times N^2$ to $M \times M$ using the following analysis.

v_i : eigenvectors of $A^T A$

μ_i : eigenvalues of $A^T A$ and C

Av_i : eigenvectors of C

Face Recognition : Standard Eigenspace-Based Approach

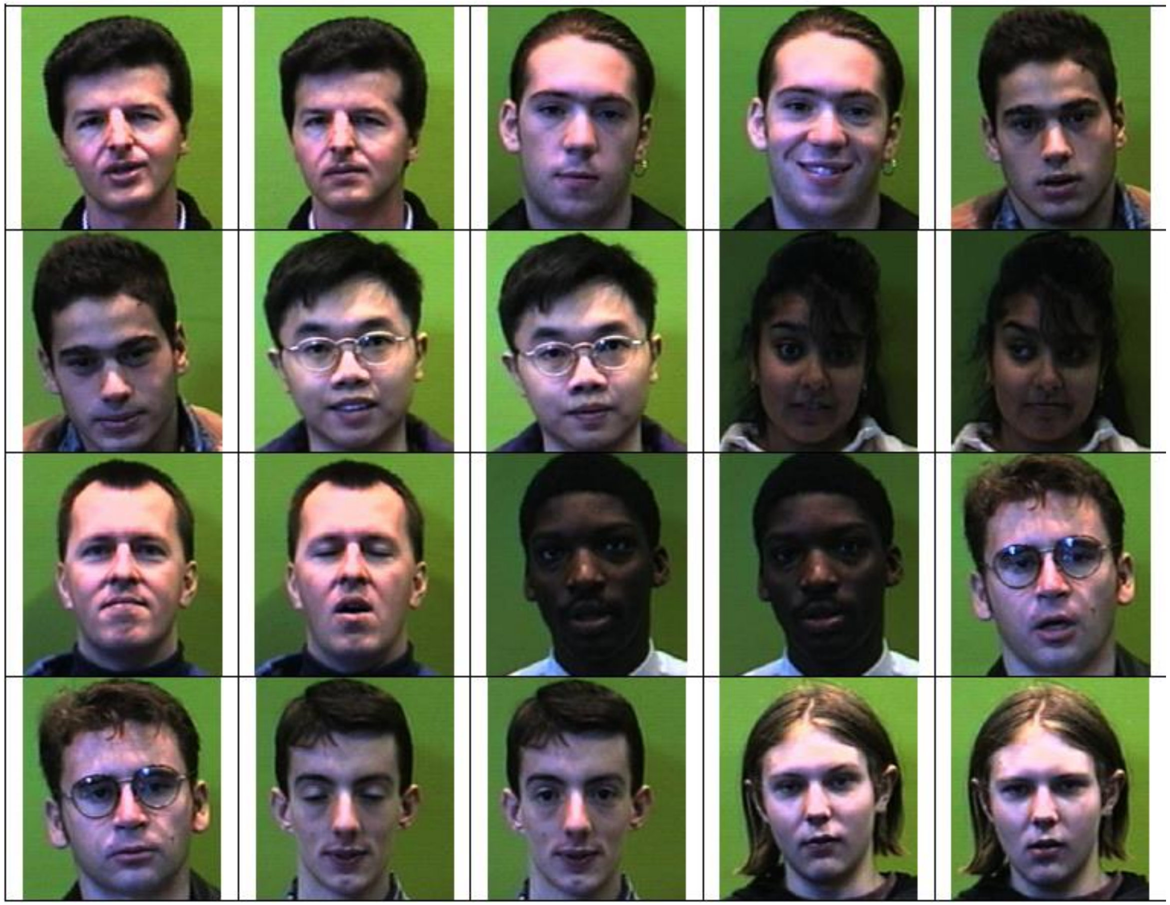


Fig.10 Training set

Fig.11 Mean face

Face Recognition : FLD Eigenspace-Based Approach

- Mathematical Expression

  ❶ Selecting projection unitary vector $\boldsymbol{u}$ s.t. $\Upsilon(\boldsymbol{u})$ to be maximized

S_b : Measuring the separation between the individual class means respect to the global mean face

S_w : Measuring the separation between vectors of each class respect to their own class mean

  ❷ Using Lagrange multiplier and set $\boldsymbol{u}^T S_w \boldsymbol{u}=1$ be the constraint condition

: Generalized eigenvalue problem