

TR-88-003

AN AUTOREGRESSIVE-MOVING AVERAGE MODEL FOR
SHAPE ANALYSIS

書 考 參
——
借 外 不

中研院資訊所圖書室



3 0330 03 000078 5

0078

TR-88-003

An Autoregressive-Moving Average Model for
Shape Analysis

Jun S. Huang and Shih P. Lu

Computer Vision Laboratory
Institute of Information Science
Academia Sinica
Taipei, Taiwan, Rep. of China

April 1988

ABSTRACT:

An new approach based on the stochastic model, Autoregressive – Moving Average (ARMA) model, to recognize the shape of patterns is proposed. In general, the dimension of the feature vector is fixed, thus all the feature clusters of the patterns are in the same space. So the number of the clusters of the patterns in the feature space is limited. In this paper, we use an algorithm to determine the order of the ARMA model dynamically, so the order can be treated as a feature. This increases the degree of freedom of the feature space. In our experiment, some patterns can be easily classified. This is because the order is determined dynamically, so some classes may contain only one pattern. It is easy to recognize those patterns whose orders belong to the classes containing just one pattern. Finally, three decision functions are used to investigate the recognition rate of our new approach — Autoregressive—Moving Average model. The results indicate the new method is reliable.

I. Introduction:

Pattern recognition is an area including broad application domain, such Optical Character Recognition(OCR)[1], and Object Recognition[2]. The research on pattern recognition has been progressing for a long time. Thousands of paper have been published. In convention, the methodologies can be divided into Syntactic and Statistical pattern recognition. In this paper, we propose a statistical algorithm to classify shape patterns.

In general, the first step in pattern recognition is feature extraction. The feature extraction of a pattern can have definite goals. While analyzing a pattern, we hope the features selected may be used to reconstruct the original pattern, and can preserve information as much as possible. In recognition, we hope the features extracted can obviously distinguish one from the other patterns, and they are orientation and size invariant and also not noise sensitive. In past, the research of statistical work on feature extraction has been devoted on three classes [3]:

- 1) Linear and nonlinear transformation to map patterns to lower-dimensional spaces for pattern recognition or enhance the class separability.
- 2) Feature evaluation criteria that bound the Bayes error probability and transformations that are optimum with respect to such criteria.
- 3) Search procedures for suboptimal selection of a subset from a given set of measured or derived features.

The content of this paper belongs to the first class.

In the first class, many transformation techniques have been proposed. In the early approaches, some algorithms tried to extract the features from the brightness function $f(x,y)$ of an image. Typical methods are moments and Fourier descriptors, etc. In general, the number of patterns the system learns can not be too large. Otherwise the clusters of pattern will overlap in the features space, and this will decrease the recognition rate. The early research [14] on OCR, only 10 to 36 characters the system can recognize. It is easy to understand that the characters set of a successful OCR algorithm should at least have more

than 92 characters(52 alphabetic, 10 numeric, 30 special characters). Seven or more moment invariants seems impossible to classify 92 characters (clusters) exactly. In recent publications, Dudani[5] uses two set of seven moment invariants as features to recognize aircraft. Although he gives some excited result, the system just handles six types of aircraft. In [4], Pavalidis says moment is not a popular technique any more. But in image processing, Tsai[9] has successfully thresholded an image by using moment method. In general, moments are noise sensitive and hence are not suitable for recognition task.

A more promising technique is Fourier Transform (FT). It provides a much more application domain than moment such as handprinted character recognition [2]. Pattern representation and reconstruction using Fourier descriptors still has many problems, such as variability in the different starting point[6], and the dimensionality of features, etc.

In this paper, we propose an approach based on stochastic model – Autoregressive moving average model for shape analysis. In literature[7,8], circular autoregressive (CAR) model proposed to analyze and classify shape patterns. In both approaches, the model order is fixed, which is questionable, and a set of coefficients, for example 15, is calculated by linear regression (LR) or maximum likelihood (ML) for each pattern. Although three recognition procedure are proposed in [8]. In fact, because of the order of model limited the number of patterns to be recognized by the system is not large. If we determine the order of the model from the boundary of the pattern, and the order can be treated as a feature of the pattern. This will increase much more degree of freedom in defining a classification space. And the classification space is not a fixed dimensional space any more. Apart from the dynamic model order, to improve the accuracy of a model we use Autoregressive and Moving Average (ARMA) model to fit the shape pattern. By adding moving average model we can avoid the cyclic (or preiodic) phenomenon of the AR model and compensate the approximation error in defining a model.

The following sections, we will describe the mathematical model about the AR, and ARMA model, and the solution algorithm for dynamical model of ARMA function is

described. In section IV, we demonstrate the result of our experiments on shape recognition by our new method: ARMA model.

II. Mathematical Model:

What is the time series? What is the stochastic process? In [13], Box and Jenkins define the terms as follow.

A time series is a set of observations generated sequentially in time. A statistical phenomenon that evolves in time according to probabilistic laws and we use a model to describe the probability structure of a sequence observations is called stochastic process. The objective of the statistical investigation is to infer the properties of the population from those observations. So, for example, we can make a forecast by inferring the probability distribution of the observation in the future from the past value.

A stochastic process is called stationary process if the model is based on the assumption that the process is in a particular state of statistical equilibrium about a constant mean level.

There are three important practical applications that can be handled by these models very usefully.

1. Forecasting:

If we analysis the observations occurred in the past, and we can forecast its value in future time. Usually it is applied in the applications of economic and business planning and production planning, etc.

2. Estimation of transfer function:

In industry, we can achieve better control of existing plants to improve the design of new plants.

3. Design of discrete control system:

In stochastic models, the Autoregressive model is extremely useful in the representation of certain practically occurring series. It is defined as following:

$$z_t = \alpha_1 z_{t-1} + \alpha_2 z_{t-2} + \alpha_3 z_{t-3} + \dots + \alpha_p z_{t-p} + a_t$$

where the z_t 's are obserations and a_t 's are the random error, and p and α_i $i=1..p$ are unknown, to be estimated, coefficients.

It is called an Autoregressive (AR) process of order p.

Another kind of the model is finite Moving Average process. In this process, z_t depends linearly on a finite number q of previous a_t 's. Thus

$$z_t = a_t - \beta_1 a_{t-1} - \beta_2 a_{t-2} - \dots - \beta_q a_{t-q}$$

We call it Moving Average (MA) process of order q.

To achieve the flexibility in fitting the time series, mixed Autoregressive and Moving Average is investigated. Thus,

$$z_t = \alpha_1 z_{t-1} + \alpha_2 z_{t-2} + \alpha_3 z_{t-3} + \dots + \alpha_p z_{t-p} + a_t - \beta_1 a_{t-1} - \beta_2 a_{t-2} - \dots - \beta_q a_{t-q}$$

We call it ARMA with order (p,q), in abbreviation ARMA(p,q).

The models described above are all stationary process, that is, a set of observations, said $z_{t_1}, z_{t_2}, \dots, z_{t_m}$, the joint probability distribution associated with these observations at any time t_1, t_2, \dots, t_m , is as same as that associated with another set of observations observed at time $t_1+k, t_2+k, \dots, t_m+k$, for a constant k. For example, Let $m=1$, the assumption of stationary process implies that the probability distribution $p(z_t)$ is the same for any time t. So, the stationary process has a constant mean:

$$\mu = E[z_t] = \int_{-\infty}^{\infty} z_t p(z_t) dz,$$

And the variance is also a constant:

$$\sigma^2 = E[(z_t - \mu)^2] = \int_{-\infty}^{\infty} (z_t - \mu)^2 p(z) dz$$

III. The algorithm to solve dynamic order of ARMA

We have seen the mathematical model of ARMA. How to determine the order of the model dynamically? An algorithm proposed by Franke in statistical journal [10] called Levinson–Durbin recursion for autoregressive moving average processes is invoked. After the order λ is determined, we give model coefficients some initial values, and then use the subroutine ZXSSQ of IMSL mathematical package to calculate the true coefficient values. The purpose of ZXSSQ is to minimize the sum of squares of function in N variables using a finite difference Levenberg–Marquardt algorithm. And we invoke subroutine FLIKAM, proposed in journal Applied Statistics [11] to calculate sum of square for ZXSSQ λ and approximate the final estimated coefficients. In the following paragraph we will describe the detail of λ computing algorithm.

Let $y(n)$ be observations.

Let $c(k)$ be the covariance of $y(n)$ and $y(n+k)$, then we first determine pure autoregressive using Levinson–Durbin algorithm.

$$\begin{aligned}
 1) \quad \sigma_{0,0}^2 &= c(0); \\
 2) \quad \alpha_{p,0}(p) &= - \left\{ c(p) + \sum_{k=1}^{p-1} \alpha_{p-1,0}(k) c(p-k) \right\} / \sigma_{p-1,0}^2 \\
 \alpha_{p,0}(k) &= \alpha_{p-1,0}(k) + \alpha_{p,0}(p) \alpha_{p-1,0}(p-k), \quad (k=1, \dots, p-1) \\
 \sigma_{p,0}^2 &= \left\{ 1 - \alpha_{p,0}^2(p) \right\} \sigma_{p-1,0}^2 \quad (3.1)
 \end{aligned}$$

In [10], Franke proposed a theorem to fit the part of moving average part. Here we reprint the recursive relation of the coefficients.

$$\begin{aligned}
 \text{i)} \quad \sigma_{0,0}^2 &= \nu_{0,0}^2 = c(0) \\
 \text{ii)} \quad &\text{for all } q \geq 1
 \end{aligned}$$

$$\sigma_{p,q}^2 = \sigma_{0,q-1}^2 - \sigma^2 h^2(q), \quad \nu_{0,q}^2 = \nu_{0,q-1}^2 - \sigma^2 h^2(q-1)$$

$$\beta_{0,q}(k) = h(k), \delta_{0,q}(q+1-k) = h(k-1), (k=1, \dots, q); \quad (3.2)$$

iii) for all $p \geq 1$ $\alpha_{p,0}(k)$ ($k=1, \dots, p$), $\sigma_{p,0}^2$ satisfy the Levinson-Durbin recursion eq (3.1) and $\gamma_{p,0}(k) = \alpha_{p,0}(k)$ ($k=1, \dots, p$), $\nu_{p,0}^2 = \sigma_{p,0}^2$; (3.3)

iv) if, for $p, q \geq 1$, $y(n)$ is not an autoregressive-moving average process of order $(p-1, q-1)$, then

$$\alpha_{p,q}(p) = - \left\{ c(p) + \sum_{k=1}^{p-1} \gamma_{p-1,q}(p-k)c(k) - \sigma_{p-1,q}^2 \sum_{k=1}^q \delta_{p-1,q}(q+1-k)h(k) \right\} \nu_{p-1,q}^{-2}$$

$$\gamma_{p,q}(p) = - \left\{ c(p) + \sum_{k=1}^{p-1} \alpha_{p-1,q-1}(k)c(p-k) - \sigma_{p-1,q-1}^2 \sum_{k=1}^{q-1} \beta_{p-1,q-1}(k)h(k-p) \right\} / (\sigma_{p-1,q-1}^2 - \sigma^2)^{-1}$$

$$\sigma_{p,q}^2 = \sigma_{p-1,q}^2 - \alpha_{p,q}^2(p) \nu_{p-1,q}^2, \quad \nu_{p,q}^2 = \nu_{p-1,q}^2 - \gamma_{p,q}^2(p) (\sigma_{p-1,q-1}^2 - \sigma^2),$$

$$\alpha_{p,q}(k) = \alpha_{p-1,q}(k) + \alpha_{p,q}(p) \gamma_{p-1,q}(p-k) \quad (k=1, \dots, p-1)$$

$$\gamma_{p,q}(k) = \gamma_{p-1,q-1}(k) + \gamma_{p,q}(p) \alpha_{p-1,q-1}(k) \quad (k=1, \dots, p-1)$$

$$\beta_{p,q}(k) = \beta_{p-1,q}(k) + \alpha_{p,q}(p) \delta_{p-1,q}(q+1-k) \quad (k=1, \dots, q)$$

$$\delta_{p,q}(k) = \delta_{p-1,q-1}(k) + \gamma_{p,q}(p) \beta_{p-1,q-1}(k) \quad (k=1, \dots, q-1)$$

$$\delta_{p,q}(q) = \gamma_{p,q}(p) \quad (3.4)$$

We summarize the algorithm in the following:

Step 1.

Calculate the autocovariance of $y(n)$

$$\hat{c}_y(k) = T^{-1} \sum_{j=1}^{T-k} y(j)y(j+k) \quad (k=0 \dots N)$$

Using Levinson-Durbin algorithm i.e. eq (3.1).

Autoregressive estimates $\alpha_{p,0}(k)$ and $\sigma_{p,0}^2$ are calculated. Then choose an order h to minimize the criterion function $AIC(p)$. Here we define the range of h , $5 \leq h < 20$.

$$\text{and } \text{AIC}(p) = \log \hat{\sigma}_{p,0}^2 + 2p/T \quad (p \leq N)$$

After h is chosen, calculate the residuals

$$\hat{\epsilon}(k) = y(k) + \sum_{j=1}^h \hat{\alpha}_{h,0}(j) y(k-j), \quad k=0,1,\dots,T$$

Step 2:

Estimate the cross-covariance of $y(n)$ and $\hat{\epsilon}(n)$ by

$$\hat{c}_{ye}(k) = T^{-1} \sum_{j=h+1}^{T-k} y(k+j) \hat{\epsilon}(j) \quad (k = 0 \dots Q)$$

$$\text{Let } \hat{\sigma}_{ye}^2 = c_{ye}(0), \quad \hat{h}(k) = \hat{c}_{ye}(k) / \sigma_{ye}^2$$

Using eq (3,2), (3,3), (3,4) calculate the estimates

$$\hat{\alpha}_{p,q}(k), \quad (k = 1 \dots p), \quad \hat{\beta}_{p,q}(k), \quad (k = 1 \dots q),$$

Then \hat{p}, \hat{q} are chosen by minimize the criterion function

$$\text{BIC}(p,q) = \log \sigma_{p,q}^2 + (p+q)T^{-1} \log T \quad (p \leq P, q \leq Q)$$

Here we define $Q = 10$.

Let $\hat{\alpha}(k)$ ($k=1 \dots \hat{p}$), $\hat{\beta}(k)$, ($k = 1 \dots \hat{q}$) be the coefficients of order (\hat{p}, \hat{q}) , then calculate $\tilde{\epsilon}$

$$\tilde{\epsilon}(k) = y(k) + \sum_{j=1}^{\hat{p}} \hat{\alpha}(j) y(k-j) - \sum_{j=1}^{\hat{q}} \hat{\beta}(j) \tilde{\epsilon}(k-j)$$

$$\tilde{\epsilon}(k) = y(k) = 0 \quad (k < 0)$$

using $\tilde{\epsilon}$ instead $\hat{\epsilon}$, step 2 repeated once.

Step 3:

Let p, q be the order determined in step 2, and $\alpha(k)$, $k=1 \dots p$, $\beta(k)$, $k=1 \dots q$ be the coefficients estimated in step 2. Using $\tilde{\alpha}, \tilde{\beta}$ to be the initial value, calculate the final estimates $\tilde{\alpha}, \tilde{\beta}$ by calling ZXSSQ of IMSL and sum of square by calling FILKAM.

IV. Experiment

In previous section, we described the mathematical model of ARMA process. The observation of ARMA is one-dimensional scalar data. We should map two dimensional image pattern into one dimensional data. The most reliable information of two-dimensional image is the silhouettes in a thresholded image. The most useful information is the shape of a pattern. For a binary image, the contour is the most important feature. We can trace the contour of the pattern easily and then calculate the center point, and the vectors between the centroid and boundary points. Although there four method is proposed in [7], we believe among them, method 2 is the most reliable. We use vectors as data input to our computing algorithm.

For the reason that the number of vectors affects the order determined by the algorithm described in last section, not all of the vectors are treated as the input data. Here we use the algorithm of vector selection in [8]. The vector is selected where the boundary point intersects with the N radius vectors from centroid. For example, in Fig 1. we select the vectors. Then got one-dimensional time series.

How to choose N ? In [7], N is set 64. In coding the shape of a pattern, 64 vectors will not lose too much information. But in recognition, rotation and scaling must be considered. 64 seems too less, and the quantity of input data of the algorithm described in previous section should be large, so that the result will be more reliable. It is a tradeoff to increase the number of vectors and to avoid the errors caused by noises. Thus we use the property:

$$r(0) = r(N), r(1) = r(N+1) ..$$

$$\text{that is, } r(i) = r(N+i).$$

Then we can trace the boundary twice or three times. Experimentally, we let N be 120, that is, we calculate each vector from the centroid by three degree apart. And we trace boundary twice. This can give us more stable result. If we trace the boundary once, sometimes the orders of the same shape calculated twice will different.

The results of our experiment are shown in Table I. Most of the variances of the coefficients are very small, and this means that the final estimates are very stable.

One point must be reminded. For all convex shape of pattern, the number of vectors is 240; but for a nonconvex shape it is not exact, and the number of the vectors of a nonconvex shape will change. For example, if a contour tracer traces such a shape shown in Fig. 2(a), it may generate a sequence vectors,

$$\dots \overline{OP}, \overline{OQ}, \overline{OR}, \dots$$

But next time, the shape is rotated. It will generate the following sequence vectors,

$$\dots \overline{OP}, \overline{OR}, \dots$$

i.e. Fig. 2(b) and will lose \overline{OQ} , because the vector \overline{OQ} does not intersect with the vector $i2\pi/N$ from centroid. Thus the number of vectors of with nonconvex shape may be variant. So is the order of such pattern. For this reason, in Table I, shape \blacksquare have two order, 9 and 10. We will consider both cases.

After features extraction, how to determine what is the input pattern? In [15], Cash et.al. proposed 4 decision functions. They are the followings:

1.) Euclid distance:

$$D_E = \sqrt{\sum_{i=1}^n (F_{L_i} - F_i)^2}$$

Where F_{L_i} is the mean of the feature of the training sample.

F_i is the feature of the input pattern.

Choose a minimum distance between the feature of input pattern and the center point of the cluster of the training samples in the feature space.

So far, it seems that Euclidean distance function is a most popular and simplest decision function for recognizing patterns. In fact, Euclidean distance function is the simplest method, but in certain cases, we doubt the reliability of Euclidean function.

2.) Weighted Euclid distance:

$$D_w = \sqrt{\sum_{i=1}^n w_i (F_{L_i} - F_i)^2}$$

Where w_i is the i -th weighted factor.

What value is the weighted factor assigned? Intuitively, the feature which has a small variance is more reliable. Therefore, it seems reasonable:

$$w_i = \frac{1}{\sigma_i}$$

σ_i is the standard deviation.

3.) Cross Correlation:

$$R = \frac{\sum_{i=1}^n F_{L_i} F_i}{\sqrt{\sum_{i=1}^n F_{L_i}^2 * \sum_{i=1}^n F_i^2}}$$

This measure function is also used broadly. This function is a normalized cross correlation so that the input pattern is assigned to the one that the value of R is closest to 1.

4.) The Mahalanobis Distance:

$$D_H = \sum_{i=1}^n \left[\frac{(F_{L_i} - F_i)^2}{\sigma_{L_i}^2} \right]$$

The Mahalanobis function is a weighted distance between the input feature and the mean of the mean of the training samples. This is similar to the weighted Euclid distance.

In [8], Susan et.al. also proposed three recognition decision function, they are 1) feature weighting method, 2) rotated coordinate system method, and 3) the hyperplane method.

1.) Feature Weighting function:

$$D_F = \left(\prod_{j=1}^m \sigma_j \right)^{2/m} * \left[\sum_{i=1}^m \left(\frac{F_i - F_{L_i}}{\sigma_i} \right)^2 + m \right]$$

where σ_i is the variance of i -th coefficient of feature vector q_k , and m is the order of the model.

2.) Rotated Coordinate System Method:

$$D_R = \left(\prod_{j=1}^m \sqrt{\lambda_j} \right)^{2/m} * \left[\left[\sum_{i=1}^m \frac{[(F_i - \bar{F}_{L_i}) * c_i]^2}{\lambda_i c_i} \right] + m \right]$$

where the λ_i is the eigenvalue of covariance matrix of the feature vector of the training samples, and c_i is the element of the eigenvector.

3.) Hyperplane Method:

Hyperplane divides the pattern space into unique convex polyhedral regions for each class. For each of c class, the linear discriminant function can be defined as following.

$$g_i(p) > g_j(p) \quad \text{for all } j \neq i \text{ if } p \in \text{class } i.$$

A least-square method uses all c training sets to find the discriminant function which best satisfy the conditions:

$$g_i(p) = 1 \quad \text{for } p \in \text{class } i$$

$$g_j(p) = 0 \quad \text{for all } j \neq i$$

function g_i can be defined as

$$g_i(p) = y^T \kappa^{-1} M_i,$$

where y is feature vector of input pattern.

$$y = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \cdot \\ \cdot \\ \theta_m \end{bmatrix}$$

M_i is the class i mean vector, and

$$\kappa = \sum_{i=1}^c \kappa_i$$

where κ_i is the correlation function for class i

$$\kappa_i = E_i(y y^T).$$

The decision rule is

$$p \in \text{class } i \text{ if } y^T \kappa^{-1} M_i > y^T \kappa^{-1} M_j \\ \text{for all } j = i, 2, \dots, c, j \neq i.$$

All of these decision functions listed above, need a number of training samples to calculate the mean and the variance of each coefficient of learning samples to get a reliable center point of the cluster of the training pattern. For the flexibility, a technique of mean and variance calculation proposed in [12] is used. So the system can learn pattern any time. Here we reprint the equations.

Let the N be the number of the learned patterns.

\bar{X}_n , S_n be the mean and variance of N pattern respectively.

\bar{X}_{n+1} , S_{n+1} be the new mean and variance, after a new pattern P is added.

Then,

$$\bar{X}_{n+1} = \frac{N}{N+1} \bar{X}_n + \frac{P}{N+1} \\ S_{n+1} = S_n + \frac{N}{N+1} (P - \bar{X}_n)^2$$

The system can revise the mean and variance any time and immediately.

In our experiments, 12 patterns are chosen to be tested. They are classified by the model order, and the patterns are classified into the same class if they has the same model order. Table I shows the mean and the variance of each coefficients. The number of the training set of patterns are variant. It depends on the variance of the coefficient. If the variance is large, more samples are trained. All training samples are rotated and scaled with various degree.

Here we chose three decision functions to discriminate the feature vectors. They are Feature Weighting function proposed in [8], Euclid distance, and Cross correlation proposed

in [15]. Why them? There are two reasons:

- 1.) They are easy to implement, and need less computation time than the others.
- 2.) They are dissimilar to each other.

Table II shows the recognition rate of each pattern. It is easy to recognize those patterns whose orders belong to the class containing just one pattern. After analysing all of the error cases. We make the following conclusions:

1. The distribution of the clusters of two, or three or more patterns are too close, so that the decision function makes a wrong decision. Such case is showed in Table III. In Table III (a), the input pattern is \star , and it is easy to find out that the distances between input pattern and the pattern \mathbf{T} are very close. In construct, Table III (b) is an opposite case. If such case is occurred, an auxiliary recognition function can be applied to choose a candidate. This problem can be sloved by learning more samples.
2. The order of input pattern is computed by the algorithm to a wrong order, so that the pattern falls into another class. It does not compare with the coefficients of its original pattern. In our experiment, we summarize the following three cases of this phenomenon.
 - i) The condition of Fig. 2 occurs, and the system did not train such case at training time. There are two possible occurrences. (1) The probability of wrong order is very low, so the system could not detect from the training samples. (2) The noise affects the order. Beause the order is determined from data, so it is sensitive in the number of the vectors.
 - ii) The variation of the order of MA part. We can approximate the pattern with a higher MA order. In our experiment, just find one pattern that is, pattern \star . The order of pattern \star is ARMA(6,0), seldom the pattern is classified to ARMA(6,1). Table IV shows the

coefficients of ARMA(6,1).

3. In some class the variances of the coefficients of the pattern are very large. So the sum of variances is also very large. It affects the decision function. For example, in Table V (a) shows the means and variances of coefficients of pattern Ψ , this maybe we train a wrong sample. After retraining, shown in Table V (b), the variance is reduced, and the recognition rate increases immediately.

V. Summary

In this paper, we present a new approach based on Autoregressive and Moving Average model to recognize shape of the pattern, and an algorithm to determine the order of ARMA model dynamically. By our algorithm, the dimension of the feature vector is not fixed any more. Twelve patterns is tested, and three decision functions is used to investigate the recognition rate. After this experiment, we find that Feature Weighting function works more stably than the other two, and the pattern with a convex shape has a higher recognition rate than the one with a nonconvex shape. Feature extraction using ARMA model with dynamic order can reduce the recognition burden. Because the number of patterns with the same order to compare will be less than the other model such as Fourier Descriptors, etc.

Reference:

- [1.] Eric Persoon, King Sun Fu, "Shape Discrimination Using Fourier Descriptors", IEEE Trans. on System, Man, and Cybernetics, Vol. SMC-7, No. 3, Mar. 1977
- [2.] King Sun Fu, "Syntactic Pattern Recognition and Applications", Prentice-Hall 1983
- [3.] Leveen Kanal, "Pattern in Pattern Recognition", IEEE Trans. on Information Theory, Vol IT-20, No. 6, Nov. 1974
- [4.] Theodosios Pavlidis, "Algorithms for Shape Analysis of Contours and Waveforms", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, No. 4, Jul. 1980
- [5.] S. A. Dudani, K. J. Breeding, R. B. McGhee, "Aircraft Identification by Moment Invariants", IEEE Trans. on Computers, Vol C-26, No. 1, Jan. 1977
- [6.] Charles T. Zahn, Ralph Z. Roskies, "Fourier Descriptors for Plane Closed Curves", IEEE Trans. on Computers, Vol. C-21, No. 3, Mar. 1972
- [7.] R. L. Kashyap, R. Chellappa, "Stochastic Models for Closed Boundary Analysis: Representation and Reconstruction", IEEE Trans. on Information Theory, Vol. IT-27, No. 5, Sep. 1981
- [8.] Susan R. Dubois, Filson H. Glanz, "An Autoregressive Model Approach to Two-Dimensional Shape Classification", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-8, No. 1, Jan. 1986
- [9.] Wen-Hsiang Tsai, "Moment-Preserving Thersholding: A new Approach", Computer Vision, Graphics and Image Processing, Vol 29, 1885
- [10.] J. Franke, "A Levinson-Durbin Recursion for Autoregressive-Moving average processes", Biometrika 72, 3, 1985
- [11.] G. Mèlard, "A Fast Algorithm for the Exact Likelihood of

Autoregressive—Moving Average Models", Applied Statistics, 1984

- [12.] J. S. Huang "A Reliable Method for Finger—print Recognition", Technical report, Computer Vision Lab. Academia Sinica, Taipei, Taiwan, R.O.C. 1987
- [13.] George E. P. Box, wilym M. Jenkins, "Time Series Analysis Forecasting and Control", Holden—Day rev. ed. 1976
- [14.] M. K. Hu, "Visual Pattern Recognition by moment invariants", IRE Trans. Inform. Theory, IT—8, pp. 179—187, 1962
- [15.] Glenn L. Cash, Mehdi Hatamian, "Optical Character Recognition by the Method of Moments", Computer Vision, Graphics, and Image Processing, Vol. 39, pp 291—310, 1987

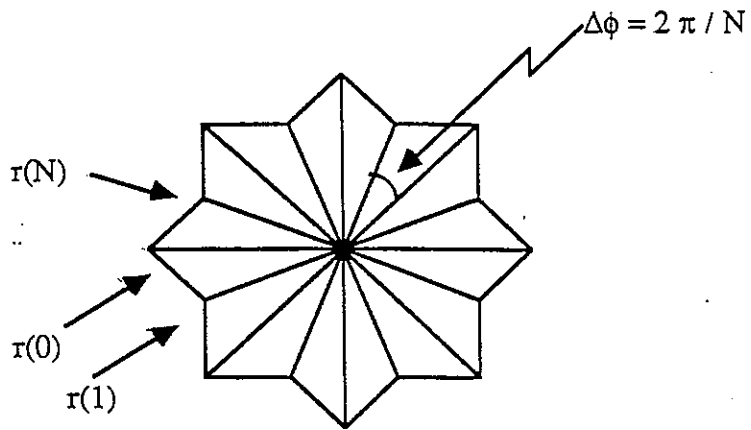


Fig 1. Vectors selection on a shape of pattern.

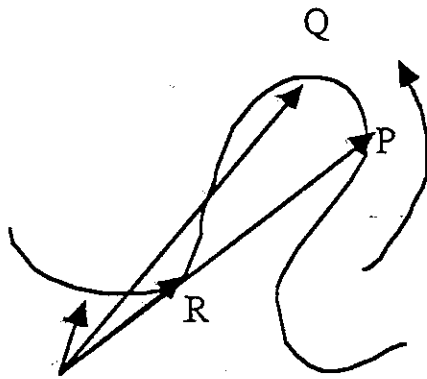


Fig 2 (a). A patial contour of pattern.

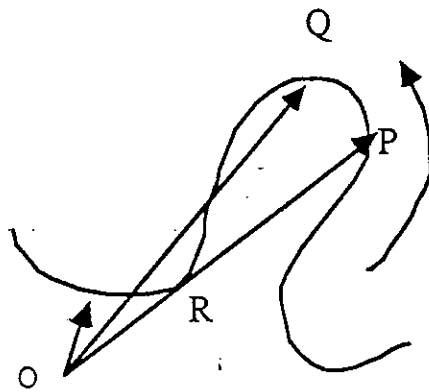


Fig 2 (b).

A patial contour of pattern after rotated.

Lose vector \overline{OR}

★

ARMA(5, 0)

	Mean	Variance
1	0.548880	0.052816
2	-1.180682	0.021081
3	-1.364043	0.039577
4	-0.787912	0.007784
5	1.798710	0.038185
Sum of Var		0.026512 Training : 9

T

ARMA(5, 0)

	Mean	Variance
1	-0.288765	0.922094
2	-1.010653	0.010627
3	-1.045731	0.030057
4	-0.505891	0.015576
5	1.446901	0.019795
Sum of Var		0.039050 Training 19

V

ARMA(5, 0)

	Mean	Variance
1	-1.610439	0.088016
2	0.732723	0.347813
3	0.003007	0.726831
4	-0.283206	0.854388
5	0.164244	0.153035
Sum of Var		0.310998 Training 14

W

ARMA(5, 0)

	Mean	Variance
1	-1.664049	0.233096
2	0.566553	0.709413
3	0.305581	0.816608
4	-0.260440	1.158054
5	0.058515	0.205986
Sum of Var		0.553040 Training 26

Class A: ARMA(5,0)

Table I The classes of the classification of the training set.

ARMA(6, 0) —		
Mean	Variance	
1	0.445190	0.322218
2	-0.797620	0.030530
3	-1.124964	0.121688
4	-0.966168	0.098362
5	-0.441238	0.018096
6	1.683133	0.198389
Sum of Var 0.086632 Training 8		

ARMA(6, 0) —		
Mean	Variance	
1	2.722213	3.367947
2	0.317246	0.034172
3	0.119145	0.013555
4	-0.137230	0.023774
5	-0.301525	0.057988
6	-3.546528	3.604422
Sum of Var 0.140681 Training 23		

ARMA(6, 0) —		
Mean	Variance	
1	-1.228174	0.211649
2	-0.799583	0.004482
3	-0.672452	0.008261
4	-0.493207	0.008465
5	-0.245768	0.007089
6	1.298239	0.026091
Sum of Var 0.015187 Training 16		

ARMA(6, 0) —		
Mean	Variance	
1	2.539408	4.215354
2	0.667374	0.142986
3	0.220033	0.045774
4	-0.187552	0.111405
5	-0.594261	0.298251
6	-3.146580	4.676288
Sum of Var 0.403047 Training 13		

ARMA(6, 0) —		
Mean	Variance	
1	3.358508	5.626099
2	0.280219	0.037964
3	0.086659	0.006083
4	-0.133274	0.007241
5	-0.320491	0.039293
6	-4.223699	5.940062
Sum of Var 0.194007 Training 21		

Class B: ARMA(6,0)

Table 1.

(Continued).

	Mean	ARMA(7, 0) Variance
1	-0.434488	0.957934
2	-0.706496	0.006965
3	-0.708110	0.012502
4	-0.626381	0.024305
5	-0.565838	0.049182
6	-0.361710	0.015944
7	1.640734	0.069058
Sum of Var 0.037776 Training 14		

Class C: ARMA(7,0)

	Mean	ARMA(8, 0) Variance
1	0.715309	2.319125
2	-1.741698	1.744731
3	-1.182193	0.702284
4	-0.298739	0.090347
5	0.648069	0.105761
6	1.491397	0.576681
7	1.590033	1.964084
8	-2.219599	2.326059
Sum of Var 0.719143 Training 8		

Class D: ARMA(8,0)

	Mean	ARMA(9, 0) Variance
1	1.208763	2.463917
2	0.060551	0.062775
3	-0.698029	0.136676
4	-1.207470	0.438961
5	-1.312453	0.770789
6	-1.006077	0.403766
7	-0.420677	0.090661
8	0.272176	0.017663
9	2.139301	1.586275
Sum of Var 0.268842 Training 9		

Class E: ARMA(9,0)

	Mean	ARMA(10, 0) Variance
1	1.931679	1.644185
2	-0.002121	0.006528
3	-0.601152	0.038638
4	-1.108928	0.203131
5	-1.451197	0.501685
6	-1.377268	0.485426
7	-0.937727	0.197473
8	-0.407609	0.042511
9	0.213846	0.008070
10	2.793010	1.632728
Sum of Var 0.136642 Training 6		

Class F: ARMA(10,0)

Table 1.

(Continued)













class	pattern	recognition rate	number of test	recognize	miss
A		92.86	28	26	2
		92.73	55	51	4
		100.00	40	40	0
		100.00	28	28	0
B		100.00	32	32	0
		91.67	36	33	3
		88.24	34	30	4
		100.00	30	30	0
		100.00	32	32	0
C		100.00	30	30	0
D		100.00	34	34	0
E		100.00	25	25	0

Table II (a.) The recognition rate using Weighted Feature function.













class	pattern	recognition rate	number of test	recognize	miss
A		92.86	28	26	2
		85.45	55	47	8
		87.50	40	35	5
		100.00	28	28	0
B		100.00	32	32	0
		61.11	36	22	14
		82.35	34	28	6
		100.00	30	30	0
		100.00	32	32	0
C		100.00	30	30	0
D		100.00	34	34	0
E		100.00	25	25	0

Table II (b.) The recognition rate using Euclid distance function.













class	pattern	recognition rate	number of test	recognize	miss
A		96.43	28	27	1
		81.82	55	45	10
		87.50	40	35	5
		82.14	28	23	5
B		100.00	32	32	0
		83.33	36	30	6
		85.29	34	29	5
		100.00	30	30	0
		100.00	32	32	0
C		100.00	30	30	0
D		100.00	34	34	0
E		100.00	25	25	0

Table II (c.) The recognition rate using Cross Correlation function.

input pattern : ★
order : ARMA(5,0)
feature vector : 0.195765 -0.944032 -1.083878
 -0.663209 1.508246

distance calculated by feature weighting function

★ : 0.329706
T : 0.292863
H : 20.727790
Y : 23.323088

Table III (a). Two clusters are too close to make an wrong decision

input pattern : T
order : ARMA(5,0)
feature vector : 0.564584 -1.187487 -1.412959
 -0.730302 1.799313

distance calculated by feature weighting function

★ : 0.145649
T : 0.187457
H : 26.827008
Y : 32.458173

Table III (b). Two clusters are too close to make an wrong decision


input pattern : 
 order : ARMA(6,1)
 feature vector AR : -0.700510 -0.780307 0.318456
 0.199838 -0.007714 -0.023481
 MA coeficient : -0.286593

Table IV. Variation on MA order.

— pattern ∇ ARMA(5, 0) —

	Mean	Variance
1	-1.305520	0.624432
2	-0.142057	3.628530
3	0.940712	5.066145
4	-0.744696	5.427259
5	0.258233	1.126892
Sum of Var 2.340299 Training 10		

Table V (a). A training wrong sample.

— pattern ∇ ARMA(5, 0) —

	Mean	Variance
1	-1.692980	0.090668
2	0.892005	0.455900
3	-0.115434	0.954488
4	-0.231200	0.982637
5	0.153618	0.167591
Sum of Var 0.365214 Training 9		

Table V(b) After retraining.