doi:10.11835/j.issn.1671-8224.2014.03.02

To cite this article: PU Ling-min, HU Hong-mei. Codebook design using improved particle swarm optimization based on selection probability of artificial bee colony algorithm [J]. J Chongqing Univ: Eng Ed [ISSN 1671-8224], 2014, 13(3): 90-98.

Codebook design using improved particle swarm optimization based on selection probability of artificial bee colony algorithm *

PU Ling-min†, HU Hong-mei‡

Chien-shiung Institute of Technology, Taicang 215411, P. R. China

Received 2 April 2014; received in revised form 5 May 2014

Abstract: In the paper, a new selection probability inspired by artificial bee colony algorithm is introduced into standard particle swarm optimization by improving the global extremum updating condition to enhance the capability of its overall situation search. The experiment result shows that the new scheme is more valuable and effective than other schemes in the convergence of codebook design and the performance of codebook, and it can avoid the premature phenomenon of the particles.

Keywords: vector quantization; codebook design; particle swarm optimization; artificial bee colony algorithm

Introduction

Codebook design is the key technique of vector quantization and the performance of the codebook is directly relevant with the effect of vector quantization. The goal to study codebook design algorithms is to search for a more efficient global optimization algorithm or a nearly global best codebook to increase the performance of codebooks and decrease the total compute complexity.

In 1980, Linde et al. [1] first proposed an efficient and simple vector quantization codebook design algorithm, called LBG (Linde, Buzo and Gray) codebook design algorithm. The algorithm is easy to carry out and has the strict theory basis, but its computation is large and easy to converge into the local minimum. To overcome the problems, different improved algorithms appeared, such as simulated annealing (SA) [2], tabu search (TS) [3], neural network [4], genetics algorithm (GA) and ant colony optimization (ACO) [5]. Some experiments show that these algorithms can improve the performance of codebooks from different extents, but they have some flaws. For example, the performance of SA algorithm is related to the initial value and sensitive to parameters; TS algorithm has a slow convergence speed and a long running time; neural network is subject to the initial network weight, which tends to converge to different
local minima with different weights; GA algorithm has more complex programming and spends more time to search for satisfactory solutions; and ACO is easy to be influenced by initial distribution points and obstacles, and ants fall into infeasible points in the complex environment easily.

Particle swarm optimization (PSO) is a new evolutionary computing method that was developed by Eberhart and Kennedy [6] in 1995 through the simulation of simplified social models of bird flocks. Due to its simple concept and easy implementation, it has been successfully applied to function optimization, data mining and other fields. Its basic idea is used to search the optimal solution in the complex space through collaboration and information sharing among individual groups. The flow chart of standard PSO is shown in Fig. 1.

In PSO, the individual is called a particle. Each particle represents a feasible solution of the optimization problem and evaluates its fitness value through the objective function. In the search space, particles fly at a certain speed which is determined and adjusted dynamically by the particle itself and flying experience of other particles. The velocity of each particle decides the direction and distance of its flight. The particle searches by following the current optimal maximum. The following describes the standard PSO algorithm [7-8].

Suppose that the number of swarms is \( N \) and the search space is \( M \)-dimensional, then the position and the velocity of the \( i \)-th particle is respectively represented as \( X_i(x_{i1}, x_{i2}, x_{i3}, ..., x_{im}) \) and \( V_i(v_{i1}, v_{i2}, v_{i3}, ..., v_{im}) \). The best previous position of this particle is denoted as \( P_i(p_{i1}, p_{i2}, p_{i3}, ..., p_{im}) \) called individual optimum and the best previous position discovered by the whole swarm is denoted as \( P_g(p_{g1}, p_{g2}, p_{g3}, ..., p_{gm}) \) called global optimum [9].

In each iteration, the particle updates its velocity and position according to Eqs. (1) and (2).

\[
\begin{align*}
    v_{ij}(t+1) &= wv_{ij}(t) + c_1r_1(p_{ij}(t) - x_{ij}(t)) + \\
    &+ c_2r_2(p_{gj}(t) - x_{ij}(t)), \\
    x_{ij}(t+1) &= x_{ij}(t) + v_{ij}(t+1),
\end{align*}
\]

where \( j \) is the \( j \)-th dimension of the particle, \( i \) is the \( i \)-th particle, \( t \) is the \( t \)-th iteration, \( c_1 \) and \( c_2 \) are the acceleration coefficients and their values are usually from 0 to 2, \( w \) is the inertia weight, and \( r1 \) and \( r2 \) are the random number with uniform distribution \( U(0,1) \).

### 2 Codebook design scheme based on PSO

2.1 Codebook design scheme

Suppose \( X = \{x_0, x_1, x_2, ..., x_{N-1} \mid x \in \mathbb{R}^d \} \) is the
training sequence composed of $N$ training vector $x_i$. Codebook design process is to seek the best clustering scheme that can divide training vectors into $M$ partitions according to Eq. (3).

$$R_j = \{ x_i \mid d(x_i, y_j) = \min_{0 \leq i \leq M-1} d(x_i, y_j), x \in X \},$$

$$j \in \{0, 1, 2, \ldots, M\},$$

where $y_j$ is the $j$-th codeword and $R_j$ is the $j$-th cell. The purpose of codebook design scheme is to minimize the total distortion measure among all training vectors and their own relative partitions.

$$D(Y) = \frac{1}{M} \sum_{j=0}^{M-1} \min_{l \neq j, k \neq l} d_y,$$

(4)

where $d_y$ is the distortion measure between the training vector $x_i$ and the codeword $y_j$, which can be attained by the most commonly used mean square error (MSE) as

$$d_y = \sum_{l=0}^{k-1} (x_i - y_j)^2,$$

(5)

where $k$ is the dimension of training vectors.

Vector quantization codebook design process is essentially a kind of high dimensional data clustering. And the individual behaviour of particles is a clustering process. The basic idea of particles clustering scheme is to pick up a training vector randomly from training vectors that are not clustered, and then choose a cell to put it down according to the transition probabilities between the training vector and all cells until all training vectors are completely clustered. Every particle will repeat it on this way. After a period of time, $m$ particles will construct $m$ codebooks. This shows the parallel trait of PSO. If the termination condition has achieved, the best codebook outputs. Otherwise, the training vectors will be re-clustered until the better codebook generates.

2.2 Codebook design scheme based on PSO

In the algorithm, the code method based on clustering centroids is adopted. That is, the position of each particle is composed of $N$ cluster centroids. In addition, the velocities and fitnesses of particles are considered. Because the dimension of training vectors is $d$, and the position $X$ of the particle is $N \times d$ dimension variable, so is the velocity $V$. Meanwhile, each particle has its fitness value.

When the initial position $\{X_1, X_2, X_3, \ldots, X_N\}$ is determined, namely the clustering centroid is determined, then the cluster partition of training vectors sequence $S = \{S_1, S_2, S_3, \ldots, S_M\}$ is done according to the nearest neighbor rule as Eq. (6).

$$\|S_j - X_j\| = \min_{l=1, 2, \ldots, N} \|S_j - X_l\|,$$

(6)

Then the dispersion measure of the cluster $j$ is achieved.

$$J_j = \sum_{S \in S_j} d(S_i, X_j),$$

(7)

Then the individual fitness $f$ can be calculated by Eq. (8).

$$f_i = L \sum_{j=1}^{M} J_j,$$

(8)

where $\sum_{j=1}^{M} J_j$ is the total dispersion measure that is the total distortion value between the training vector in the cluster and the cluster centroid, and $L$ is the adjusting constant according to the specific situation. The smaller the total dispersion measure is, the more the individual fitness is.
Steps of codebook design based on PSO are as follows.

**Step 1** Initial parameters. In the beginning, each particle is put into a cluster. Then the cell centroids can be calculated according to the nearest neighbor, which is used to be the initial positions of particles. At the same time, the fitnesses of particles are got, and the velocities of particle are initialized. After repeating $N$ times, the $N$ initial particle swarms appear.

**Step 2** For each particle, comparing its current fitness with the best fitness it got. If it is better, the best position $P_{\text{best}}$ will be updated based on Eq. (9).

$$P_i(t+1) = \begin{cases} P_i(t), & \text{if } f(X_i(t+1)) \leq f(P_i(t)), \\ X_i(t+1), & \text{others}, \end{cases}$$

(9)

where $i$ is the iteration time and $i$ is the $i$-th particle.

**Step 3** For each particle, comparing its current fitness with the fitness that all particles go through the best global position $G_{\text{best}}$. If it is better, update the $G_{\text{best}}$ as follows.

$$P_g(t) \in \{P_0(t), P_1(t), \ldots, P_N(t)\} \mid f(P_g(t)) = \max \{f(P_0(t)), f(P_1(t)), \ldots, f(P_N(t))\},$$

(10)

**Step 4** Adjust the velocity and position of each particle according to Eqs. (1) and (2).

**Step 5** Optimize the new individual by LBG algorithm. The final codebook will be attained until the termination condition (namely, a better codebook has been formed or maximum number of iterations has been reached) is achieved. Otherwise, re-estimate the centroids in terms of LBG optimization rule.

### 2.3 Improved codebook design scheme based on PSO

The nature of PSO is to guide the particle to search the next iteration by self-information, the individual optimum information and the global optimum information. As it approaches the optimum, its speed is smaller, which would lead to a strong convergence and fall into local minima easily.

In the standard PSO algorithm, particles update the individual optimum and global optimum only when they find the better solution, which will make the search neighbourhood of particles smaller and bring the premature convergence. In addition, empty cells are more prone to occur.

To avoid the flaw, an improved algorithm that changes the global optimum updating condition inspired by artificial bee colony algorithm (ABC) [10-12] is presented in the paper. The basic idea is as follows.

If the fitness value of the new particle is larger than the current global optimum, the system will accept the particle; otherwise, the system will decide whether it is accepted according to the selection probability $p_i$ attained by calculating Eq. (11).

$$p_i = 0.1 + 0.9 \times \frac{f_i}{\max f}.$$  

(11)

where $\max f$ is the best fitness in all particles and the 0.1 and 0.9 are empirical value inspired by ABC algorithm. If $p_i$ is larger than the threshold $r$ which is a random value in $(0, 1)$, the new individual is accept. Otherwise, it is abandoned. The goal of the threshold $r$ is to make the random perturbation in each iteration, which could enlarge the global search scope by comparing its value with the selection probability to decide whether to accept the current solution. With the increase of the number of iterations or particles, the global search ability and convergence performance are better.

Moreover, some empty cells occur while re-clustering the new individuals. Then, many training vectors farthest from their cell centroids are taken and put into the empty cell nearest from them. Repeat this
process until all empty cells disappear.

The above-mentioned method can make the total distortion measure smaller and avoid the premature convergence.

3 Experimental result

3.1 Experimental process

The flow chart of isolated word speech recognition diagram based on Discrete Hidden Markov Model (DHMM) [13-14] using PSO is shown in Fig. 2. The experimental process is as follows [15].

1) Preprocessing the speech signal by sampling and anti-aliasing filtering and so on.

2) Feature extraction of recognized speech signal. Feature parameters of speech signal are divided from frame to frame and each frame is considered as a vector. Finally, the speech feature parameters are got into vector sequences. The zero-crossing with peak amplitude (ZCPA) is adopted for feature extraction. In this paper, 1024-dimension vector sequences of speech signal are extracted.

3) Codebook design based on PSO. In this paper, the codebook size and dimension is 128 and 4, respectively. All the training data are divided into 128 clusters which have their own label from 0 to 127. After iterative calculation, vector quantizer can be attained.

4) 1024-dimensional speech feature parameters are partitioned into well trained vector quantizer. 4-dimensional speech feature parameters are taken as a vector and finally 256 vectors form. By quantizing 256 vectors, code labels from 0 to 127 can be attained according to the principle of the nearest neighbour.

5) Training code labels of the training speech into DHMM model. Speech recognition system adopts DHMM model without leapfrog from left to right. Each word model includes five states. The training method uses the classical Baum-Welch algorithm. Finally, each word is trained as one parameter in DHMM model.

6) The matching probability between each word and its DHMM model parameter is calculated by using Viterbi algorithm. The model corresponding to the maximum probability is the recognition result.

Fig. 2 Isolated word speech recognition diagram based on DHMM (Discrete Hidden Markov Model) where PSO is particle swarm optimization
3.2 Experimental parameter setup

The experimental data are isolated word speech database with 10 words, 20 words, 30 words, 40 words and 50 words, respectively, which are taken in different Gauss white noise environments by pronouncing each word 3 times by 16 persons. Finally, 9 speech data are used to be the training speech corpus, and 7 speech data are used to be the test corpus. The signal/noise ratio (SNR) of speech signal in five cases are 15 dB, 20 dB, 25 dB, 30 dB and clean.

PSO algorithm contains some important parameters including the population size N, the dimension of particles D, the most iteration time Tmax, the acceleration coefficients c1 and c2, the inertia weight w and the fitness function f.

In the experiment, these parameters are set by empirical value and search features of particles. N = 30, D = 512, Tmax = 10, and c1 = c2 = 1.8.

In addition, w is set by

\[ w = w_{\text{max}} - T \times \frac{w_{\text{max}} - w_{\text{min}}}{T_{\text{max}}} \], \tag{12} \]

where \( w_{\text{max}} \) is the most the inertia weight and is taken as 0.9, \( w_{\text{min}} \) is the lest the inertia weight and is taken as 0.4, and T is the iteration time. The way of setting w can make the global convergence ability of the beginning better. Then with the increase of the number of iterations, w decreases and the local convergence ability becomes stronger.

The fitness function f can reflect the true fitness of all particles within populations, according to which, the optimum solution can be chosen. In the experiment, the fitness function f can be attained by Eq. (8).

3.3 Experimental result

The recognition result is judged by the recognition ratio between the number of words identified correctly in the test database and the total number of words in all test databases. Tables 1, 2 and 3 show the comparison of recognition results in different conditions.

Table 1 Comparison of recognition results based on DHMM (Discrete Hidden Markov Model) improved by two PSO (particle swarm optimization) algorithms where SNR is signal/noise ratio

<table>
<thead>
<tr>
<th>Word size</th>
<th>Algorithm</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
<th>Clean</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>PSO</td>
<td>86.2%</td>
<td>86.3%</td>
<td>85.5%</td>
<td>86.8%</td>
<td>89.7%</td>
</tr>
<tr>
<td></td>
<td>Improved PSO</td>
<td>85.7%</td>
<td>89.3%</td>
<td>87.2%</td>
<td>84.4%</td>
<td>89.9%</td>
</tr>
<tr>
<td>20</td>
<td>PSO</td>
<td>78.3%</td>
<td>78.9%</td>
<td>81.4%</td>
<td>86.3%</td>
<td>85.8%</td>
</tr>
<tr>
<td></td>
<td>Improved PSO</td>
<td>82.3%</td>
<td>82.7%</td>
<td>83.1%</td>
<td>84.4%</td>
<td>84.7%</td>
</tr>
<tr>
<td>30</td>
<td>PSO</td>
<td>80.7%</td>
<td>81.4%</td>
<td>85.1%</td>
<td>83.6%</td>
<td>84.3%</td>
</tr>
<tr>
<td></td>
<td>Improved PSO</td>
<td>81.2%</td>
<td>84.2%</td>
<td>82.7%</td>
<td>85.1%</td>
<td>82.9%</td>
</tr>
<tr>
<td>40</td>
<td>PSO</td>
<td>83.5%</td>
<td>77.3%</td>
<td>77.9%</td>
<td>80.8%</td>
<td>83.1%</td>
</tr>
<tr>
<td></td>
<td>Improved PSO</td>
<td>78.3%</td>
<td>79.6%</td>
<td>81.7%</td>
<td>82.9%</td>
<td>84.4%</td>
</tr>
<tr>
<td>50</td>
<td>PSO</td>
<td>72.6%</td>
<td>76.3%</td>
<td>81.4%</td>
<td>78.1%</td>
<td>81.6%</td>
</tr>
<tr>
<td></td>
<td>Improved PSO</td>
<td>76.2%</td>
<td>77.6%</td>
<td>78.7%</td>
<td>83.2%</td>
<td>82.8%</td>
</tr>
</tbody>
</table>

Table 2 Average recognition result in difference words based on two algorithms where PSO is particle swarm optimization

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Word size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10</td>
</tr>
<tr>
<td>PSO</td>
<td>86.9%</td>
</tr>
<tr>
<td>Improved PSO</td>
<td>87.3%</td>
</tr>
</tbody>
</table>

Table 3 Average recognition result in different SNRs (signal/noise ratios) based on two algorithms where PSO is particle swarm optimization

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>SNR/dB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>15</td>
</tr>
<tr>
<td>PSO</td>
<td>80.2%</td>
</tr>
<tr>
<td>Improved PSO</td>
<td>80.7%</td>
</tr>
</tbody>
</table>
Table 1 shows that the recognition ratio of improved PSO algorithm based on DHMM is higher in various degree than standard PSO in most cases, which can explain that the improved PSO can solve the problem of premature convergence appearing in standard PSO.

Table 2 shows that the average recognition ratio of improved PSO increases about 0.2% to 1.7%, and Table 3 shows that the average recognition result of improved PSO is slightly higher than standard PSO in difference SNRs.

Based on the comparison results of their average recognition ratio, it can be seen that the improved PSO is more suitable for codebook design than the standard PSO, because it can better find the global optimal solution. At the same time, the experiment results prove that the improved PSO can improve the effectiveness of DHMM in isolated word speech recognition system.

In addition, we can input a speech signal of 40 s and sampling rate of 8KHZ16 bit in PCM format, where 30 s signal is used as the training speech and others are used as the test speech. The algorithm uses the speech data in wav format, so their level amplitudes are normalized. We designed the codebook of 1024 size on the sample data of speech signal by using the standard PSO and improved PSO algorithm, respectively, reflecting the global searching ability of two algorithms. The initial codebook is generated in a random selection and the processing method of empty cell is the LBG algorithm.

Fig. 3 shows the sample data of the initial speech signal. Figs. 4 and 5 show the result of codebook designed by using the standard PSO and the improved PSO. From Figs. 3, 4 and 5, it can be seen that the codebook performance of the improved PSO is better than that of the standard PSO. Through auditioning the reconstructed speech signal, we can find that the performance of speech reconstruction using standard PSO is noisy, not distinct and unstable. By contrast, the definition and comprehension of speech reconstruction using improved PSO is better. So the codebook design ability of the improved PSO algorithm is stronger than that of the standard PSO algorithm and the improved PSO algorithm can better improve the performance of the codebook.
4 Conclusions

We proposed a new global extremum updating condition in standard PSO by changing the selection probability to avoid the existing problems of standard PSO for codebook design. The improved method can increase the global search capabilities of particle swarms, can avoid the premature convergence of particles, and is more effective in vector quantization.

Reference


