Object Tracking with an Evolutionary Particle Filter Based on Self-Adaptive Multi-Features Fusion

Zhang Xiaowei¹,², Liu Hong¹,²,* and Sun Xiaohong¹,²

1 School of Information Science and Engineering, Shandong Normal University, Jinan, China
2 Shandong Provincial Key Laboratory for Novel Distributed Computer Software Technology, Jinan, China
* Corresponding author E-mail: lhsdcn@jn-public.sd.cninfo.net

Received 20 Jun 2012; Accepted 5 Nov 2012
DOI: 10.5772/54869
© 2013 Zhang and Liu; licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract Particle filter algorithms are widely used for object tracking in video sequences, but the standard particle filter algorithm cannot solve the validity of particles ideally. To solve the problems of particle degeneration and sample impoverishment in a particle filter tracking algorithm, an improved object tracking algorithm is proposed, which combines a multi-feature fusion method and a genetic evolution mechanism. The algorithm dynamically computes the feature's fusion weight by the discriminability of each vision feature and then constructs the important density function based on selecting a feature's fusion method adaptively. Moreover, a self-adaptive genetic evolutionary mechanism is introduced into the particle resampling process and makes the particle become an agent with the ability of dynamic self-adaption. With self-adaptive crossover and mutation operators, the evolution system produces a large number of new particles, which can better approximate the true state of the tracking object. The experimental results show that the proposed object tracking algorithm surpasses the conventional particle filter on both robustness and accuracy, even though the tracking object is very challenging regarding illumination variation, structural deformation, the interference of similar targets and occlusion.

Keywords Particle Filter, Self-Adaptive, Multi-Features Integration, Resampling, Genetic Evolution

1. Introduction

Visual object tracking in video sequences is a central concern within the field of computer vision. Reliable visual tracking is indispensable in many emerging vision applications, such as automatic video surveillance [1, 2], human computer interface [3, 4], video compression [5, 6] and robotics [7, 8]. However, the task of robust tracking is very challenging regarding illumination variation, background clutter, fast motion, occlusion, structural deformation, real-time restriction, etc.

In order to solve the above problems, a variety of algorithms have been proposed. These tracking
algorithms can be divided into two categories. The first category is a deterministic method. This method finds the local maximum of probability distribution in the direction of gradient. The Mean-Shift [9, 10] is a typical example. It is usually quicker and more accurate than the probabilistic multi-hypothesis tracking algorithm, but it may run into trouble when complete occlusion occurs or if similar targets are presented in background. The modified algorithm of Mean-Shift is called the Continuously Adaptive Mean-Shift (Referred to as CamShift). The second category is a probabilistic method that involves target estimation. The preventative method is a particle filter, which is a multi-hypothesis tracking algorithm under the Bayesian framework [11, 12, and 13]. Moreover, due to the particle filter’s non-Gaussian, nonlinear assumption and multiple hypothesis properties, it has been successfully applied to visual tracking systems, in which the system state and measurement models are usually nonlinear. Although the particle filter is widely used, room for improvement still exists. In visual tracking, there are three main factors that affect the performance of the particle filter, namely, the reliable observation model, the accurate motion model and sample impoverishment, which are usually regarded as key research points.

Based on the advantages and shortcomings of these tracking algorithms, a variety of algorithms have been proposed. Maggio and Cavallaro [14] used semi-overlapping colour histograms to improve the sensitivity to anisotropic scale and rotations changes, which could increase the efficiency of the particle filter. However, this method performs ineffectively when occlusion occurs because the particles lack diversity. Wang and Yang [15] proposed the CamShift guided, particle filter visual tracking method. In this method, CamShift helps improve the sampling efficiency of the particle filter in both scale space and position. However, this method relies uniquely on colour information to perform the tracking, so the tracker does not have enough information to determine the position of the target as a result of the interference of a similar target and illumination variation. Kristanetal [16] tried to construct a two-stage dynamic model, which included a liberal model and a conservative model. The method improved the performance of the particle filter due to the two-stage dynamic model’s ability to actively adapt to the target’s motion during tracking. However, all of the above mentioned methods have not solved the inherent sample impoverishment of the particle filter.

Sample impoverishment is brought about by resampling, which is introduced to avoid particle degradation in the particle filter. The target tracking will fail over time when sample impoverishment occurs in the complex environment. Zhao and Li [17] proposed a particle filter based on Particle Swarm Optimization resampling for vision tracking. The method uses a PSO algorithm to search the sample area around the last object position depending on current observation, so it is able to improve the sample impoverishment to a certain extent. However, the PSO algorithm is a large calculation and has changes of particle variance distribution. Parketal [18] proposed a new evolutionary particle filter to prevent sample impoverishment by the advantages of the evolutionary algorithm in a particle filter. They rigorously account for the change of the object distribution caused by genetic operators such as crossover and mutation. Their work is only a theoretical proof and is not used in practical applications.

In this paper, an object tracking algorithm with an evolutionary particle filter, based on self-adaptive multi-features fusion is proposed. The main contributions of this study lie in the following. Firstly, because using a single cue for tracking is insufficient to deal with a wide variety of environmental conditions, the method of colour cues integrated with texture cues is proposed. Secondly, we applied an adaptive cue-integration method. In principle, when the colour cue is more reliable, its weight will become higher than the texture cue. When the colour cue is less reliable, it is compensated by the texture cue. Thirdly, the mechanism of genetic evolution is introduced for resampling in the particle filter. The crossover operator and mutation operator can be dynamically calculated based on adaptive multi-feature integration to produce the optimal particle and maintain the diversity of particles. All of these operations can improve the effectiveness and accuracy of target tracking.

The rest of the paper is organized as follows. Section 2 presents the colour cue and the texture cue and describes adaptive multi-feature integration. The particle filter is introduced for video target tracking in Section 3. Section 4 describes an evolutionary particle filter with GA. In Section 5 we illustrate experimental results of the proposed algorithm. Finally, Section 6 presents the conclusion of the whole paper.

2. Adaptive multi-features integration

Video object tracking is the extraction of candidate targets, which match the target’s template, so the feature extraction is the most fundamental and crucial step for a tracking algorithm. This section describes the visual cues, which are utilized in tracking an object of interest by combining colour and texture information.

2.1 Colour cue

Colour features have been widely applied, because of their advantages, which include the invariant of rotation and scale, the insensitivity of structural deformation and simple calculation. However, they may run into trouble when similar objects are presented in the background or when illumination variation occurs. Therefore, in order to reduce the effect of light intensity for tracking, we use
HSV colour space to establish a colour model, which does not take into account the brightness component of V. The colour histogram is constructed as:

$$p_a(y) = C_h \sum_{i=1}^{N} K \left( \frac{y-x_i}{h} \right)^2 \delta \left[ b_i(x_i) - u \right]$$

(1)

where function $b_i(x_i) \in \{1,2,...,N_1 \}$ maps the pixel location to the corresponding histogram bin, $K(\cdot)$ is the Epanechnikov kernel profile with radius $h$:

$$K(r) = \begin{cases} 1-r^2, & r < 1 \\ 0, & \text{other} \end{cases}$$

(2)

$\delta$ is the Kronecker delta function and $C_h$ is a normalization constant:

$$C_h = \frac{1}{h} \sum_{i=1}^{N} K \left( \frac{y-x_i}{h} \right)^2$$

(3)

The Similarity between object template $q(y_0)$ and candidate target $p(y) = \{p_a(y)\}_{a=1}^{N}$ can be measured by Bhattacharyya distance $d$:

$$d = \sqrt{1 - \rho(p_a(y), q(y_0))}$$

(4)

where $\rho(p_a(y), q(y_0))$ is the Bhattacharyya coefficient:

$$\rho(p_a(y), q(y_0)) = \sum_{a=1}^{N} p_a(y)q(y_0)$$

(5)

### 2.2 Texture cue

Local binary patterns (LBP) are a valid description of a local texture operator and have been widely applied in the fields of texture classification and pattern recognition, because of their strong ability to classify high computational efficiency and invariance of dull grey. LBP describe the image of a texture cue based on the binary mode results from comparing the area of the image of each pixel with its neighbour’s pixels within the domain of grey values. An LBP operator is shown in Fig. 1. The LBP is given by:

$$LBP_{p,R}(x,y) = \sum_{p=0}^{P-1} s(g_p - g_0)2^p$$

(6)

where $P$ indicates target area of pixels, $R$ represents the target pixel radius, $g_0$ is the grey at the centre pixel $(x,y)$ and $g_p$ is the grey value of the adjacent pixels around centre point $g_0$ in the region. The function $s(x)$ can be defined as:

$$s(x) = \begin{cases} 1, & x \geq T \\ 0, & x < T \end{cases}$$

(7)

However, the LBP may construct a sparse histogram, so the histograms lose statistical significance, because the LBP operator produces a more binary mode, but actually the number of pixels in the target area is relatively small. So a LBP unified mode [15] can be proposed, which is defined as:

$$LBP^{un2}_{p,R} = \sum_{p=0}^{P-1} s(g_p - g_0)2^p, U(LBP_{p,R}) \leq 2$$

(8)

where function $U(LBP_{p,R})$ calculates the hop variables in the sequence of the binary model, where the binary can be changed from “0” to “1” or from “1” to “0”. For example, the hop variables of texture patterns “11001011” and “01010011” are 4 and 6. The function $U(LBP_{p,R})$ can be described as:

$$U(LBP_{p,R}) = \sum_{p=1}^{P-1} s(g_p - g_c) - s(g_{p-1} - g_c)$$

(9)

By introducing a unified mode in the calculation process of the LBP histogram, the histogram level number is only assigned for the unified mode, and all of the non-uniform patterns are placed in a common histogram level number. For an LBP operator with 8 neighbouring pixels, the LBP series of the histogram is reduced to 59 from 256 and contains 58 unified modes and 1 non-unified mode.

### 2.3 Features fusion

Multi-feature fusion strategy includes weighted fusion, multiplicative fusion, elections and the minimum and maximum rules. So far, in these integration strategies, weighted fusion and multiplicative fusion are the most popular integration policy. From the Bayesian view, the result of multiplicative fusion is optimal, but its premise condition is the independence of each feature and this assumption can cause tracking failure because of being more sensitive to noise. In contrast, the weighted fusion cannot enlarge the noise, but each feature can be given a certain weight for weighted fusion. However, weighted fusion cannot improve the credibility of tracking, so it is not beneficial for long and accurate tracking.
In this paper, we combine the advantages of multiplicative fusion and weighted fusion, thus adaptively select the fusion strategy. Multiplicative fusion can be used when the visual features have a higher degree of confidence, and when degradation of the visual features occurs, the weighted fusion can be switched to produce a more stable likelihood function.

Therefore, we dynamically select the multi-features fusion strategy by whether degradation occurs, which can be measured according to whether the particle filter is needed for sampling. When the particles need resampling, the multiplicative integration Eq. (10) can be selected; otherwise, the weighted fusion Eq. (11) can be selected.

\[
\hat{\omega}_i = \left(\omega_i^a\right)^{\alpha} \left(\omega_i^b\right)^{\beta} \quad (10)
\]

\[
\hat{\omega}_i = \alpha \omega_i^a + \beta \omega_i^b \quad (11)
\]

where \( \alpha = \omega_i^a \left(\omega_i^a + \omega_i^b\right), \beta = 1 - \alpha \), which can be adaptively adjusted according to the distinction of colour cue and texture cue.

3. Particle filter algorithm

The particle filter is used to estimate the state of a nonlinear dynamic system sequentially in time. Generally speaking, the particle filter is based on a system of model and measurement time-dependent equations:

\[
x_k = f(x_{k-1}, w_k) \quad (12)
\]

\[
z_k = h(x_k, v_k) \quad (13)
\]

Eq. (12) is the system update equation, which represents the estimation of the state of the system from time \( k-1 \) to time \( k \). State \( x_k \) depends on the previous state \( x_{k-1} \) of the system and a stochastic error \( w_{k-1} \), which represents the uncertainty of the state update. Since \( w_{k-1} \) is a random variable of known statistics, the equation implicitly defines a probability density function \( p(x_k | x_{k-1}) \). Eq. (13) is the measurement equation, defining the dependency of the measure \( z_k \) on the current unknown value of the state \( x_k \) and the error term \( v_k \). Since \( v_k \) is a stochastic variable, this equation also implicitly defines a probability density function \( p(z_k | x_k) \).

The target dynamics are assumed to be represented as a temporal Markov chain:

\[
p(x_{0:k} | z_{1:k}) = \sum_{i=1}^{N} \omega_i^k \delta(x_{0:k} - x_{0:k}^i) \quad (14)
\]

The weight \( \omega_i^k \) is approximated as :

\[
\omega_i^k \approx \frac{p(x_{0:k} | z_{1:k})}{q(x_{0:k} | z_{1:k})} \quad (15)
\]

where important density function \( q(x_{0:k} | z_{1:k}) \) can be represented as:

\[
q(x_{0:k} | z_{1:k}) = q(x_k | x_{0:k-1}, z_{1:k}) q(x_{0:k-1} | z_{1:k-1}) \quad (16)
\]

According to the Bays rule, the posterior density is given by:

\[
p(x_{0:k} | z_{1:k}) = \frac{p(z_k | x_k) p(x_k | x_{k-1}) p(x_{0:k-1} | z_{1:k-1})}{p(z_k | x_k) p(x_k | x_{k-1}) p(x_{0:k-1} | z_{1:k-1})} \quad (17)
\]

So the weight \( \omega_i^k \) can be simplified as:

\[
\omega_i^k \propto \omega_i^{k-1} p(z_k | x_k^i) \quad (18)
\]

Then, the weight \( \omega_i^k \) can be normalized by:

\[
\omega_i^k = \omega_i^k \sum_{i=1}^{N} \omega_i^{k-1} \quad (19)
\]

and the posterior density can be represented as:

If a large number of particles can be sampled, the posterior density would be accurate, but this is impractical in real-time object tracking, so the limited particles from sampling in an accurate area are important in vision tracking.

4. Adaptive evolutionary particle filter tracking

To solve the problems of particle degeneration and particle shortage in the particle filter tracking algorithm, an evolutionary mechanism can be introduced. The particle is defined as a smart particle, which survives in a certain environment with a specific purpose and evolutionary behaviour. We improve the diversity of particle samples by using selection, crossover and mutation operation of the genetic algorithm and control the crossover and mutation operators for adaptive resampling based on a dynamic adaptive strategy, so that the performance of the particle filter is improved.

4.1 The genetic manipulation

In order to ensure a high speed of calculation and avoid the trouble caused by binary encoding and decoding, this section uses decimal encoding particles i.e., the particle selection, crossover and mutation operations are carried out on the basis of decimals.

1. Selection:

The fitness of intelligent individuals in the population can be determined according to the degree of similarity between the states of the particles and the target template.
The more similarity, the greater the weight of the particle and the particle can be selected by its higher probability. We calculate the fitness of each intelligent particle by using fitness function \( \alpha_k = f(P_k^m) \), and normalize the weight of the smart particle with \( \omega_k = \alpha_k \sum_{i=1}^{N} \alpha_i \). The weight of particle \( \hat{W}_k^i \) can be accumulated in the smart particle swarm, \( \hat{W}_k^i = \sum_{i=1}^{j} \omega_i \), \( i = 1,2,...,N \) and then a random number \( R \) in the range \([0,1]\) can be generated.

We follow these methods to determine whether an intelligent individual can be a new particle individual as the next generation in the group.

If \( R < \hat{W}_k^i \), select the first particle individual \( P_k^1, P_k^{i+1} = P_k^1 \);

If \( \hat{W}_k^{i-1} < R \leq \hat{W}_k^i \), select the particle individual \( P_k^i, P_k^{i+1} = P_k^i \).

2. Crossover:

In order to increase the diversity of individual particles in the population and to avoid falling into the local optimal solution, we randomly select two individuals from the population for crossover. The new particle individuals \( \{P_k^m, P_k^a\} \) can be generated according to crossover probability \( \omega_k \) and a random number \( R \) in the range \([0,1]\), when \( R < \omega_k \) from the population of the two individual particles \( \{P_k^m, P_k^a\} \).

\[
\hat{P}_k^m = \alpha P_k^m + (1-\alpha)P_k^a \quad (20)
\]

\[
\hat{P}_k^a = \beta P_k^a + (1-\beta)P_k^m \quad (21)
\]

where \( \alpha, \beta \) can be seen as weight coefficients:

\[
\alpha = f(P_k^m)/\left( f(P_k^m) + f(P_k^a) \right) \\
\beta = f(P_k^a)/\left( f(P_k^m) + f(P_k^a) \right) \quad (22)
\]

The new particle individuals can be accepted if they are consistent with crossover criteria. The crossover criteria can be represented as:

If \( f(\hat{P}_k^m) > \max\{f(P_k^m), f(P_k^a)\} \), then accept the particle \( \hat{P}_k^m \); otherwise, accept an individual particle according to the probability, which is described as \( f(\hat{P}_k^m) / \max\{f(P_k^m), f(P_k^a)\} \).

3. Mutation:

To bring new samples to the particle population, a new particle individual \( \hat{P}_k^i \) can be generated according to mutation probability \( \omega_m \) and a random number \( R \) in the range \([0,1]\). When \( R < \omega_m \). The new particle individual can be described following Eq. (23):

\[
\hat{P}_k^i = P_k^i + \eta \cdot N(0, \Sigma) \quad (23)
\]

The new particle individual can be accepted if consistent with mutation criteria. The mutation criteria can be represented as:

If \( f(\hat{P}_k^i) > f(P_k^i) \), then accept the particle \( \hat{P}_k^i \); otherwise, accept an individual particle according to the probability, which is described as \( f(\hat{P}_k^i) / f(P_k^i) \).

4.2 Adaptive evolution strategy

The performance of GA can be affected by the crossover probability \( P_c \) and the mutation probability \( P_m \) to a large extent and incorrect parameters can cause premature convergence. The larger \( P_c \), i.e., the greater the number of new individuals can be generated. At the same time, the destruction possibility of the genetic pattern is larger, so high-fitness individuals will be destroyed. With \( P_c \) too small, the search process will be slow. In the case of \( P_m \), being too small, it is difficult to produce a new individual. If it is too large the algorithm becomes a purely random search. Therefore, the design of both crossover and mutation operators should not be large enough to destroy the population of the best individuals, but also must be able to produce some good individuals.

In this paper, the crossover operator and the mutation operator can be dynamically computed according to evolutionary strategy and adapted as fitness changes. For the low-fitness individuals a crossover operation with a higher probability can be used. The crossover operation can also be conducted for high-fitness individuals under the condition that the best individuals cannot be destroyed.

Srinivas [19] proposed an adaptive genetic algorithm (S-AGA). \( P_c \) and \( P_m \) are adjusted as follows:

\[
P_c = \begin{cases} 
\frac{k_1(f_{\max} - f^*)}{(f_{\max} - f_{avg})} & f^* \geq f_{avg} \\
\frac{k_3}{f_{\max} - f_{avg}} & f^* < f_{avg} 
\end{cases} \quad (24)
\]

\[
P_m = \begin{cases} 
\frac{k_1(f_{\max} - f^*)}{(f_{\max} - f_{avg})} & f^* \geq f_{avg} \\
\frac{k_3}{f_{\max} - f_{avg}} & f^* < f_{avg} 
\end{cases} \quad (25)
\]

where \( k_1 = k_3 = 1, k_2 = k_4 = 0.5 \), \( f_{\max} \) is the maximum fitness in the particle population, \( f^* \) is the larger fitness of the crossover individuals, \( f_{avg} \) is the average fitness of the population and \( f^* \) is the fitness of the mutation individual.
In S-AGA, the low-fitness individual particles adapt the constant high probability to crossover operation, and the high-fitness individual adaptively can use low probability to crossover operation. This approach can effectively protect the best individuals from being destroyed, but it is unfavourable for the new generation of the best individuals. The best individual is not necessarily a global optimal solution, once the algorithm goes to a local optimum value it is difficult to find.

The paper [18] proposed adaptive genetic algorithm (W-AGA). \( P_c \) and \( P_m \) are adjusted as follows:

\[
P_c = \begin{cases} 
\frac{(p_{c1} - p_{c2})(f' - f_{avg})}{(f_{max} - f_{avg})}, & f' \geq f_{avg} \\
p_{c1}, & f' < f_{avg} 
\end{cases}
\]

(26)

\[
P_m = \begin{cases} 
\frac{(p_{m1} - p_{m2})(f_{max} - f)}{(f_{max} - f_{avg})}, & f \geq f_{avg} \\
p_{m1}, & f < f_{avg} 
\end{cases}
\]

(27)

where \( p_{c1} = 1.0, \ p_{c2} = 0.6, \ p_{m1} = 0.1, \ p_{m2} = 0.01 \). The algorithm makes the individual particles not be in a standstill state, however, when the mutation probability can achieve a greater value for a high-fitness individual, the excellent individuals will more probably be damaged.

In order to solve particle premature convergence and the problem of not benefitting from producing the best individual, this paper proposes a dynamic adaptive genetic algorithm for a particle filter. The algorithm dynamically adjusts the crossover operator and the mutation operator for different particles in particle sampling. It ensures that the new particle individuals have higher fitness, under the condition that the best individuals cannot be destroyed.

When we design the crossover operator, for low-fitness individual particles a constant high probability can be used in the crossover. For high-fitness individual particles the crossover operator \( P_c \) can be determined by using Eq. (28), which can keep a certain probability of crossover to find the optimal individual particle.

\[
P_c = \begin{cases} 
\frac{(p_{c1} - p_{c2})(f' - f_{avg})}{(f_{max} - f_{avg})}, & f' \geq f_{avg} \\
p_{c1}, & f' < f_{avg} 
\end{cases}
\]

(28)

where \( p_{c1} \) and \( p_{c2} \) represent the range of \( P_c \) and \( C_1 \) is a constant, which can adjust the changes in \( P_c \) when fitness is higher than the average fitness.

When we design the mutation operator, a constant high probability can be used for the mutation for low-fitness individual particles, which are more likely to produce an outstanding individual particle. For high-fitness individual particles, the mutation operator \( P_m \) can be adjusted by using Eq. (29), which uses the pattern of exponential changes. Compared to Eq. (27), it is more conducive in protecting the best individuals from damage.

\[
P_m = \begin{cases} 
\frac{(p_{m1} - p_{m2})(f_{max} - f)}{(f_{max} - f_{avg})}, & f \geq f_{avg} \\
p_{m1}, & f < f_{avg} 
\end{cases}
\]

(29)

where \( p_{m1} \) and \( p_{m2} \) represent the range of \( P_m \) and \( C_2 \) is a constant that can adjust the changes of \( P_m \) when fitness is higher than the average fitness.

![Figure 2. Crossover and mutation operator curve](image)

Fig. 2 compares the S-AGA, the W-AGA algorithm and the proposed method with the adaptive changes in crossover probability and mutation probability for individual particles. From the diagram, we can see the superiority of the improved algorithm that uses nonlinear adjustment of the crossover probability and the mutation probability. In a crossover operation, the high-fitness individuals maintain a high probability of crossover to produce a greater number of outstanding individuals, which is more in line with the natural laws of evolution. In the mutation operation, the low-fitness individuals have a higher probability of mutation, so produce better individual particles. While high-fitness individuals utilize the small probability of mutation to protect the elite individuals from damage.

4.3 Real-time and stability analysis of the algorithm

Considering the particle number of the particle filter is huge, GA will bring about much computation and then the algorithm will not work well. Therefore, the algorithm only selects the effective particles with typical visual features to improve the real-time performance and stability of the algorithm. The visual features, which involve colour cue and texture cue in the proposed algorithm, have been introduced in Section 2, by which the tracking target and
background can be distinguished well. Each visual feature has a corresponding set of specific particles and the weight of a particle can be computed by the proposed multi-feature fusion strategy. The fitness of GA particles uses the normalized weight of particles. Then the stability of the algorithm can be ensured by the excellent particles, which can be obtained through the self-adaptive genetic evolutionary mechanism. Finally, the algorithm’s real-time ability can be guaranteed by selecting a limited number of effective particles. The algorithm’s real-time ability is verified in Table 1 and the stability is shown in Section 5.

| No processing | CamShift | Classic PF | Proposed algorithm |
|---------------|----------|------------|---------------------|
| 84.48         | 62.30    | 34.69      | 47.81               |

**Table 1.** Frames Per Second

4.4 Adaptive evolutionary particle filter tracking algorithm

In this algorithm, the importance of the weight of each particle is seen as the fitness function of the particle, which utilizes features such as adaptive integration. The standard particle filter resampling is improved by an adaptive genetic algorithm based on selection, crossover and mutation, to solve the problems of particle degeneration and particle shortage. It is described in Algorithm 1.

Algorithm 1. Adaptive evolutionary particle filter tracking algorithm

Step 1: Particle initialization

Generate a particle group by the priori probability \( p(x_0) \) and assign the weights with \( 1/N \) for all particles.

Step 2: Update the importance weights

Firstly, set \( k := k + 1, \) sampling \( x^i_k = \{ x_k \mid x_{0:k-1}, z_{0:k} \} \), \( i = 1,2,\ldots,N \). Secondly, calculate the weight of color cue for each particle \( w^i_k \), calculate the weight of texture cue for each particle \( w^i_k \), and dynamically select the fusion method to update the importance weights:

\[
 w^i_k = w^i_{k-1} p(z_k \mid x_{k-1}) = w^i_{k-1} \omega^i_k
\]

Thirdly, normalize the importance weights:

\[
 \overline{w}^i_k = \frac{w^i_k}{\sum_{i=1}^{N} w^i_k}
\]

Step 3: Resampling based on a dynamic adaptive genetic algorithm.

If \( N_{\text{eff}} = \frac{1}{\sum_{i=1}^{N} (\overline{w}^i_k)^2} < N_{\text{threshold}} \), then particles are resampled based on a dynamically adaptive genetic algorithm. Then calculate normalized weights of each particle:

\[
 \hat{w}^i_k = \frac{1}{\sum_{i=1}^{N} \overline{w}^i_k}
\]

Step 4: Output

The state estimation:

\[
 \hat{x}_k = \frac{1}{\sum_{i=1}^{N} \hat{w}^i_k x^i_k}
\]

The variance estimation:

\[
 P_\sigma = \sum_{i=1}^{N} \hat{w}^i_k (x^i_k - \hat{x}_k)(x^i_k - \hat{x}_k)^T
\]

Step 5: The next time state prediction

\[
 x^i_{k+1} = f(x^i_k, w^i_k), i = 1,2,\ldots,N
\]

Step 6: If the end, then exit this algorithm; otherwise return to Step 2.

5. Experimental Results and Analysis

In this section the performance of the proposed algorithm is compared with other trackers from a number of aspects. All the experiments are carried out on 640*480 pixel sequences on a PC with a 2.8GHz Pentium 4 CPU and 1GB memory. Algorithms are tested with the video sequences, whose information is shown in Table 2.

| Sequence | Frame size | Sequence characteristics | Total frames |
|----------|------------|--------------------------|--------------|
| S1       | 640*480   | Illumination variation   | 623          |
| S2       | 640*480   | Interference of similar target and occlusion | 631          |
| S3       | 640*480   | Shape change             | 653          |

**Table 2.** The video sequences used in our experiments.

5.1 Human face tracking with illumination variation

In this experiment human face tracking can be tested with illumination variation in public test video sequences S1. When we only use colour information for a particle filter, the tracking will fail. This is because the target colour changes dramatically when the illumination changes. Our algorithm considers the texture cue and uses a genetic evolution operation, so we can still track the target very well. When the colour cue becomes invalid, the texture cue plays a leading role and the tracker can track the target reliably in the whole sequence. The genetic evolution operation will ensure an optimal particle for tracking is produced. Fig. 3 shows the tracking result of a CamShift guided particle filter algorithm and Fig. 4 shows the tracking result of the algorithm proposed in this paper.
From Fig. 5, we can see that the tracking result of S1 in both the X direction and the Y direction. In the 540th frame, the CamShift guided particle filter using the colour cue deviates greatly from the true location. However, the evolutionary particle filter based on self-adaptive multi-features fusion can achieve a stable track.

**Figure 3.** The human tracking result of S1 with CamShift guided particle filter algorithm.

**Figure 4.** The human tracking result of S1 with the proposed algorithm of this paper.

**Figure 5.** The tracking result of S1 in X and Y direction.
5.2 Interference of similar target and occlusion for human face tracking

In this experiment, when the phenomenon of interference of a similar target and occlusion occurs the human face tracking can be tested in the public test video sequences S2. The conventional particle filter will lead the tracking to fail due to the interference of a similar target and occlusion because of the problems of particle degeneration and particle shortage. We can make the particle become an agent with dynamic self-adaption abilities by introducing evolutionary behaviours such as selection, crossover and mutation.

Figure 6. The human tracking result of S1 with CamShift guided particle filter algorithm.

Figure 7. The human tracking result of S2 with the proposed algorithm of this paper.

Figure 8. The tracking result of S2 in X and Y direction.

Fig. 6 shows the changes in the tracking process with a CamShift guided particle filter algorithm. The tracking performance is becoming increasingly unstable over time because it only utilizes the colour cue. When the tracking target is sheltered by a similar target in the 578th frame, the algorithm falls into a local extreme value leading tracking to fail. From Fig. 7, we can see that the proposed method can track accurately and stably, even tracking objects when occlusion occurs.

From Fig. 8, we can see that the tracking result of S2 in both the X direction and the Y direction. From the 82nd frame to 176th frame, the object tracking failure of the CamShift guided particle filter algorithm occurs because of the interference of a similar background. After the 578th frame, the tracking target is lost when it is interfered with by a dynamically similar target in the X direction. On the contrary, the proposed algorithm has a higher robustness when interference of similar targets
and occlusion occurs. This verifies the efficiency of the algorithm.

5.3 The object tracking with structural deformation

The experiment is mainly used to validate the tracking performance, when structural deformation of the tracking target occurs. Human tracking can be tested in the public test video sequences S2. The CamShift guided particle filter algorithm will deviate from and even lose the tracking target with structural deformation. This is because the algorithm only utilizes the colour cue and has the problem of particle degeneration. The experimental result of the tracking algorithm is shown in Fig. 9. In the 122nd frame, the tracking area deviates from the tracking target and even fails to track in the 560th frame. Different from the CamShift guided particle filter algorithm, the tracking method of this paper fuses the colour and texture cues and introduces a self-adaptive dynamic genetic evolution mechanism for resampling to solve the degeneration and shortage of particles. Fig. 10 shows the results of the proposed tracking algorithm.

Fig. 11 shows the tracking result of structural deformation in both the X direction and the Y direction. In the long process of target tracking, the structure morphology will be changed over time so tracking fails. By combining the colour cue with the texture cue and using the resampling of the self-adaptive genetic evolution the performance of particle filter can be changed to obtain higher robustness.

![Figure 9](image9.png)  
**Figure 9.** The human tracking result of S3 with CamShift guided particle filter algorithm.

![Figure 10](image10.png)  
**Figure 10.** The human tracking result of S3 with the proposed algorithm of this paper.

![Figure 11](image11.png)  
**Figure 11.** The tracking result of S3 in X and Y direction.
6. Conclusion

To counteract the problems of existing target tracking algorithms, this paper proposes an object tracking algorithm with an evolutionary particle filter based on self-adaptive multi-feature fusion. The algorithm constructs the importance function that approaches the posterior probability distribution by fusing the colour cue and the texture cue adaptively. The crossover operator and the mutation operator can be dynamically calculated based on the mechanism of genetic evolution for resampling in the particle filter. The results of the experiment show that the proposed object tracking algorithm has more powerful anti-interference ability and higher accuracy for tracking.

7. Acknowledgments

This paper is supported by the National Natural Science Foundation of China (No.60970004, No. 61272094) and the Ph.D. Programs Foundation of Ministry of Education of China (No. 20093704110002), National Science Foundation of Shandong Province (No. Z2008G02, ZR2010QL01) and Shandong Provincial Key Laboratory Project.

8. References

[1] J. Zhu, Y. Lao, Y. F. Zheng (2010), Object tracking in structured environments for video surveillance applications, IEEE Transactions on Circuits and Systems for Video Technology 20(2) 223-235.
[2] C. M. Huang, L. C. Fu (2011), Multi-target visual tracking based effective surveillance with cooperation of multiple active cameras, IEEE Transactions on Systems, Man and Cybernetics, PartB: Cybernetics 41(1) 234-247.
[3] T. Darrell, G. Godon, M. Harville, & J. Woodall (2000), Integrate person tracking using stereo, colour, and pattern detection. International Journal on Computer Vision 175-185.
[4] C. Colombo,A. D. Bimbo & A. Valli (2003). Visual capture and understanding of hand pointing actions in a 3-D environment. IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics 677-686.
[5] W. E. Vieux, K. Schwerdt, & J. L. Crowley (1999). Face tracking and coding for video compression. In International conference on computer vision systems 151-160
[6] B. Menser & M. Brunig (2000), Face detection and tracking for video coding applications. In Conference record of the thirty-fourth asilomar conference on signals, systems and computers (pp.49-53).

[7] H. J. Bohme, T. Wilhelm, J. Key, C. Schauer, C. Schroter, H. M. Gross (2003). An approach to multi-modal human-machine interaction for intelligent service robots. Robotics and Autonomous Systems 83-96.
[8] R. Muffoz Salinas, E. Aguirre, M. Garcia-Silvente, & M. Gomez (2005). A multi-agent system architecture for mobile robot navigation based on fuzzy and visual behaviors. Robotica, 23 689-699.
[9] D. Comaniciu, V. Ramesh & P. Meer (2000). Real-time tracking of non-rigid objects using mean shift[C]/, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. Hilton Head Island 142-149.
[10] D. Comaniciu, V. Ramesh & P. Meer (2003). Kernel-Based Object Tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 25(5) 564-577.
[11] M. S. Arulampalam, S. Maskell, N. Gordon & T. Clapp (2002). A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking. IEEE Transactions of Signal Processing1 174-188.
[12] K. Nummioaro & E. Koller-Meier (2003). An adaptive color-based particle filter [J]. Image and Vision Computing 21(1) 99-110.
[13] Z. Khan & T. Balch (2005). MCMC based particle filtering for tracking a variable number of interacting targets, IEEE Transactions on Pattern Analysis and Machine Intelligence 27(11) 1805-1819.
[14] E. Maggio, & A. Cavallaro (2005). Hybrid particle filter and meanshift tracker with adaptive transition model, Proceedings of IEEE signal processing society international conference on acoustics, speech, and signal processing (ICASSP) 221-224.
[15] W. Zhaowen & Y. Xiaokang (2009). Cam-Shift guided particle filter for visual tracking [J].Pattern Recognition Letters 30(4) 407-413.
[16] M. Kristan, S. Kovacic, A. Leonardis & J. Pers (2010). A two-stage dynamic model for visual tracking, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 40(6) 1505-1520.
[17] Z. Jing & L. Zhiyuan (2010). Particle filter based on Particle Swarm Optimization resampling for vision tracking, Expert Systems with Applications 37 8910-8914.
[18] S. Park, J. P. Hwang, E. Kim & H. J. Kang (2009). A new evolutionary particle filter for the prevention of sample impoverishment, IEEE Transactions on Evolutionary Computation 13(4) 801-809.
[19] M. Srinivas & L. M. Patnaik (1994) Adaptive probabilities of crossover and mutation in genetic algorithms, IEEE Transactions on Systems Man and Cybernetics 24(4) 656-667.