Hindawi
Computational Intelligence and Neuroscience
Volume 2020, Article ID 8839725, 13 pages
https://doi.org/10.1155/2020/8839725

Research Article

Tracking Objects Based on Multiple Particle Filters for Multipart Combined Moving Directions Information

Ngo Duong Ha[1,2] **Ikuko Shimizu,**[3] **and Pham The Bao**[4]

[1]*Faculty of Mathematics and Computer Science, University of Science, Vietnam National University-Ho Chi Minh City, Ho Chi Minh, Vietnam*
[2]*Information Technology Faculty, Ho Chi Minh City University of Food Industry, Ho Chi Minh, Vietnam*
[3]*Tokyo University of Agriculture and Technology, Tokyo, Japan*
[4]*Information Science Faculty, Sai Gon University, Ho Chi Minh, Vietnam*

Correspondence should be addressed to Pham The Bao; ptbao@sgu.edu.vn

Received 11 September 2020; Revised 7 October 2020; Accepted 25 November 2020; Published 17 December 2020

Academic Editor: Silvia Conforto

Object tracking is an important procedure in the computer vision field as it estimates the position, size, and state of an object along the video's timeline. Although many algorithms were proposed with high accuracy, object tracking in diverse contexts is still a challenging problem. The paper presents some methods to track the movement of two types of objects: arbitrary objects and humans. Both problems estimate the state density function of an object using particle filters. For the videos of a static or relatively static camera, we adjusted the state transition model by integrating the movement direction of the object. Also, we propose that partitioning the object needs tracking. To track the human, we partitioned the human into N parts and, then, tracked each part. During tracking, if a part deviated from the object, it was corrected by centering rotation, and the part was, then, combined with other parts.

1. Introduction

The object tracking in videos is a technique that has many applications in many fields. For example, in the biomedical field [1, 2], the object-tracking technique is applied to automatically track cells while they are born, duplicated, or as they move and die. Another example is the application of the technique in autopilot systems, where it is used to observe and track the vehicles around the driving car [3, 4] or footballer tracking [5–7]. A highly accurate vehicle-tracking program is an indispensable necessity for safety. Moreover, the tracking technology is usually combined with the identification and recognition systems to create a complete tactic for real-life applications.

Tracking objects in video is difficult due to many challenges that all are needed to be considered and solved. The first challenge is that we do not know, in advance, the object that we need to track. There may be no information about that object. In the absence of information, the object description for the program script must be highly general. Another challenge is that the tracked object has heterogeneity in colors, which vary by each part of the object. For example, to track human movement, the head is characterized by the hair color (black and yellow), while the body and legs are described by the color of the shirt and pants that the person is wearing. Because of the challenges and difficulties mentioned above, no comprehensive tracking algorithm can be adopted for all problems.

In this paper, we present the method to modify the state model according to the direction of predicting that an object appears in the same direction of motion with a higher probability. In addition, we explore the effectivity to track partially obscured objects by tracking its visible sections. To do this, we divided the object into multiple sections and tracked these sections independently. When some parts obscure the object, our approach should still successfully track the object movement. We also present the experimental particle filter model and present a suggestion for integrating information on the direction of the object's movement, the N-particle filter model, to track each part then combines.

The rest of the paper is organized as follows: The most relevant work that motivated this paper is reviewed in Section 2. Section 3 describes, in detail, our method multiple particle filters for multipart combined moving directions information. Section 4 summarizes the results from our method. Section 5 is the discussion of our paper.

2. Related Work

The correlation filters approach is a powerful tool in digital signal processing [8, 9]. This algorithm class utilizes the properties of Fourier transform of turning convolution in the spatial domain into function multiplication in the Fourier domain [10–13]. The original idea of the correlation filter was to solve the problem of locating an object in an image. In other words, if the object of interest appears in the image, its position including the axis coordinates is determined. The tool to solve this problem is the average synthetic exact filter [10]. The next, correlation filter, is the Total Minimum Output Error, studied by Bolme et al. [13]. This tracking method is very powerful and can cope with situations such as changing light and changing the size and shape of objects.

Aviden et al. [14] at Misubishi Electric Research Labs considered object tracking as the binary classification problem to distinguish background pixels and tracking-object pixels using the AdaBoost technique. The method's idea was training weak classification functions to classify the background and object and, then, combine them to form a strong classification based on the Adaboost mechanism. But, the author realized that if an object is not in the rectangle form, pixels in the containing object rectangle but not in the object will be labeled as belonging to the object. These pixels are considered alien elements, while AdaBoost is sensitive to alien elements [15]. In addition, some other limitations of the approach are as follows: it has not solved the completely obstructed object's situation in a long time and the featured space used in the algorithm does not yet utilize the spatial information of the image.

The approach based on random process filtering has been studied for a long time in the field of mathematical statistics, and there have been many discovered impressive results [16–18]. Most of the algorithms based on this approach are based on the Bayes optimal solution for the hidden Markov filtering problem [19–21]. That means building a hidden Markov model plays a key role, and the model is more accurate; in fact, more Bayesian solutions accurately estimate the state of the object. The work in [20] uses the featured color histogram to construct a particle filter to track objects. The work in [22] uses gentle AdaBoost to construct an updated observation model over time.

Recently, the Siamese network-based trackers have received significant attention for their well-balanced tracking accuracy and efficiency. These trackers formulate visual tracking as a cross-correlation problem and are expected to better leverage the merits of deep networks from end-to-end learning [23–30]. Bhat et al. [31] proposed a gradient-guided method to update the template. Li et al. [32] developed a discriminative model-prediction architecture for tracking.

3. Methodology

Problem 1. Highlight that first frame coordinates $(x_1, y_1, \omega_1, h_1)$ are given and we need to infer object coordinates $(x_k, y_k, \omega_k, h_k)$ in the subsequent frames.

Filtering the state $(x_k, y_k, \omega_k, h_k)$ of the object in the next frames, we rely on the hidden Markov model theory with the construction of two models: state transitions and observations. The state transitions model in the studies is quite similar and are all Gaussian motion, and the main difference in the algorithms depends on the observed model. Using particle filters allows us to better handle color clutter in the background, as well as tracking completely obstructed objects.

The principle of the particle filter according to the Figure 1, including 3 steps:

Measurement: calculating the samples' weights based on the observation at time n

Resampling: resampling or fine tuning (based on the threshold of the return weights) the samples to remove or adjust the samples in which the object positions are overmismatched at the current time

Prediction: predicting the object status at time $n+1$ based on the likelihoods from time n to the previous states

Based on the particle filter operating mechanism in Figure 1, we present the approximate results of posterior density function $p(x_k|y_{1:k})$ at time k.

From the posterior distribution at time $k-1$, $p(x_k|y_{1:k-1})$, we calculate the prior distribution for the time k (without observing y_k) by using Chapman–Kolmogorov equality [16], $p(x_k|y_{1:k-1}) = \int p(x_k|x_{k-1}).p(x_{k-1}|y_{1:k-1})\, dx_{k-1}$, where $p(x_k|x_{k-1})$ already exists in the state transition model and $p(x_{k-1}|y_{1:k-1})$ is the posterior root of step $k-1$.

After observing y_k, we update the prior density function at the predicted step at the level k: $p(x_k|y_{1:k}) = (p(y_k|x_k)^* p(x_k|y_{1:k-1})/p(y_k|y_{1:k-1}))$, where $p(y_k|x_k)$ already exists in the observation model and $p(x_k|y_{1:k-1})$ is an a priori at the time k calculated in the previous step.

As a result, we obtain a weighted pattern representing the posterior density function at the time k: $p(x_k|y_{1:k}) \sim \{x_k^l, (1/Z)\omega_{k-1}^{l\,*} p(y_k|x_k^l)\}_{l=1}^{L}$.

When an object is in motion, it usually moves in a specific trajectory. Therefore, to predict the object's location, we propose integrating the direction of motion, which will be discussed in detail in Section 3.1. In addition, different parts of an object carry their distinctive characteristics of shape, color, light absorption, and reflection capacity. If we use a particle filter, it can yield false tracking results. Besides, if a part of the object has the same color and brightness level as any other object in the frame, the tracking may be distorted. To fix this problem, we propose to divide the object into many parts, each of which will have the same properties. We, then, track the movement of each part with the constraints that these parts move in the same direction and maintain similar area and shape.

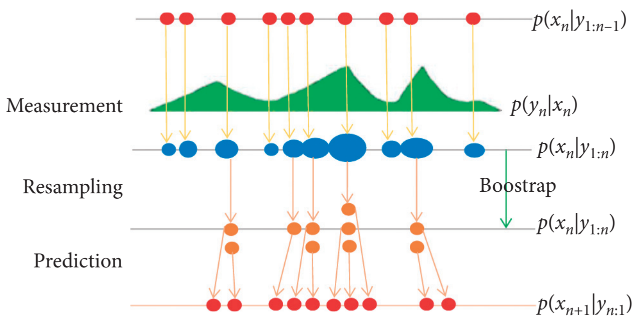

FIGURE 1: Demonstrate the operating mechanism of a particle filter.

3.1. Moving Direction Information. With the videos filtered out from the dataset in which the camera was relatively stable, we modified the state transition model in the hidden Markov model by integrating the direction of the object movement. This means that instead of using the Gaussian motion state model, we projected these Gauss functions into several different directions with different ratios before we made a new pattern. Because each object moves in a specific trajectory, the direction of the object's motion will remain constant for a certain period of time. Specifically, we consider the direction of motion as a separate component. At each assessment, we will update the direction of motion. We use this direction of motion to impact the particle filter at the prediction step, with the purpose that the particle filter will predict the object appearing in the same direction with higher probability.

$\overrightarrow{v}$ is rotated an angle θ by multiplying the rotation matrix by $\overrightarrow{v}$ as follows:

$$\overrightarrow{v_\theta} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} . \overrightarrow{v}, \tag{1}$$

where $\theta = \{0, \pi/4, -\pi/4, \pi/2, -\pi/2, 3\pi/4, -3\pi/4, \pi\}$.

Figure 2 depicts the probability distribution in the direction of the object's appearance, in which the red direction (in the direction of a_1) is the predicted movement direction of the object, $a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8$ are real numbers in [0, 1], and $\sum_{i=1}^{8} a_i = 1$. In the prediction step, we translate $N_s \cdot a_1$ particle by $\overrightarrow{v}$, $N_s \cdot a_2$ particle by $\overrightarrow{v_{(\pi/4)}}$,$N_s \cdot a_3$ particle by $\overrightarrow{v_{(-\pi/4)}}$, $N_s \cdot a_4$ particle by $\overrightarrow{v_{(\pi/2)}}$, $N_s \cdot a_5$ particle by $\overrightarrow{v_{(-\pi/2)}}$, $N_s \cdot a_6$ particle by $\overrightarrow{v_{(3\pi/4)}}$, $N_s \cdot a_7$ particle by $\overrightarrow{v_{(-3\pi/4)}}$, and $N_s \cdot a_8$ particle by $\overrightarrow{v_\pi}$.

We propose the particle filter algorithm to integrate the direction of motion in Algorithm 1.

3.2. Multiple Particle Filters Model

3.2.1. Multiparts of an Object. While considering the problem where the object shapes are less changing, if the object includes many parts with dissimilarity about the colors and contrast, using 1 particle filter for tracking will lead to incorrect tracking. Therefore, an object needs to separate each area with similar color, grayscale, and contrast into n parts, each part being tracked separately. In this way, the parts which affected by the environment and other object

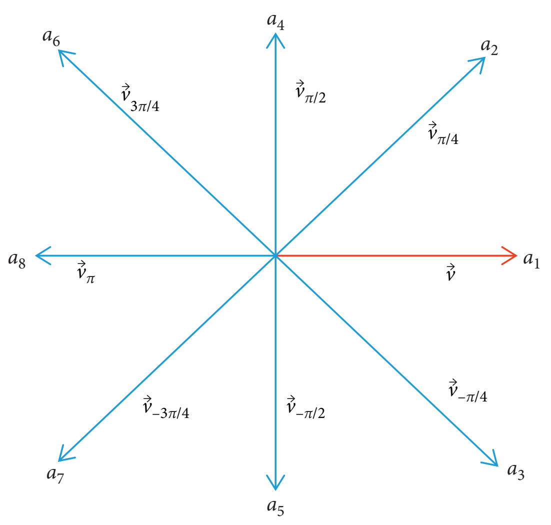

FIGURE 2: Distribute probabilities for moving directions.

artifacts will cause incorrect position identification which will need to be adjusted. For example, a human object can be normally represented by a 3-partition structure as illustrated in Figure 3. This structure divides the human object based on the gray color changes among the black head, the white shirt body, and the black pants legs. The resulting human object will be divided into 3 parts with a border represented by a different gray level, each part using a particle filter to track and combine based on the best feature matching part.

3.2.2. Build Model. The adjustment of deviated areas should take two steps:

(i) The center of the similar areas changed, which allowed the incorrect position of the similar areas to be adjusted accordingly to the correct position of the object

(ii) Size ratio of similar area allowed similar areas to scale the height to the height of the original object

Thus, when one part of the object is obscured or similar to another, we can restore and track enough.

3.2.3. Fine Tuning Parts. Once the anterior root of the object is defined, the object is defined into n parts in a structure H, as shown in Figure 3. We, then, used one particle filter to track each part $S_1, S_2, \ldots, S_n$. At each time data point, each particle filter in the tracking area can diverge from each other. Therefore, a modification model is needed to correct this issue. We used the collected assessment data of each part to evaluate which tracked part behaves the best. We kept this best-tracked part fixed and applied the rotation algorithm to $n-1$ other parts using the fixed part as the origin. The adjustment of N-particle filters when tracking an object in the frame time k is described below.

Step 1: we calculate the rotation from the center of each section with the remaining $n-1$ based on frame 0.

```
Input: Particles pf, Observation image, motion direction $\vec{v}$
Output: New particles represent $p((x, y, \omega, h)|\text{observed image})$, estimating the state of the object in the observed image.
Step 1:
  Translate $N_s \cdot a_1$ particles in pf by $\vec{v}$
  Translate $N_s \cdot a_2$ particles in pf by $\overrightarrow{v_{(\pi/4)}}$
  Translate $N_s \cdot a_3$ particles in pf by $\overrightarrow{v_{(-\pi/4)}}$
  Translate $N_s \cdot a_4$ particles in pf by $\overrightarrow{v_{(\pi/2)}}$
  Translate $N_s \cdot a_5$ particles in pf by $\overrightarrow{v_{(-\pi/2)}}$
  Translate $N_s \cdot a_6$ particles in pf by $\overrightarrow{v_{(3\pi/4)}}$
  Translate $N_s \cdot a_7$ particles in pf by $\overrightarrow{v_{(-3\pi/4)}}$
  Translate $N_s \cdot a_8$ particles in pf by $\overrightarrow{v_{\pi}}$
Step 2:
  for $i = 1$ to $N_s$ do
  Beginfor
    /* (with $(x_i, y_i, \omega_i, h_i)$ is $i$ th particle) */
    Get $x_{\text{new}} \sim x_i + N(0, \sigma_x^2)$
    Get $y_{\text{new}} \sim y_i + N(0, \sigma_y^2)$
    Get $\omega_{\text{new}} \sim \omega_i + N(0, \sigma_\omega^2)$
    Calculate $h_{\text{new}} = \eta * \omega_{\text{new}}$
    Calculate likelihood $= p(\text{observated image}|(x_{\text{new}}, y_{\text{new}}, \omega_{\text{new}}, h_{\text{new}})$ by Algorithm 8
    Update weight of $i$ th particle: $\text{weight}_i = \text{weight}_i * \text{likelihood}$
  Endfor
Step 3:
Calculate the total $\text{sw} = \sum_{j=1}^{N_s} \text{weight}_i + \varepsilon$
  for $i = 1$ to $N_s$ do
  Beginfor
    Standardize weight $\text{weight}_i = (\text{weight}_i + \varepsilon/\text{sw})$
  Endfor
Step 4: Calculate $N_{\text{eff}} = (1/\sum_{j=1}^{N_s} \text{weight}_j^2)$
Step 5:
  if $N_{\text{eff}} < N_s/2$ then
    $\{(x_i, y_i, \omega_i, h_i), \text{weight}_i\}_{i=1}^{N_s} = \text{RESAMPLE}(\{(x_i, y_i, \omega_i, h_i), \text{weight}_i\}_{i=1}^{N_s})$
Endif
Step 6: Estimate the state of the object in the $k$th image by calculating the average of the new set of particles
EstimatedStatus $= \sum_{i=1}^{N_s} \text{weight}_i * \text{particle}[i]$
```

Algorithm 1: Particle filter integrated motion direction.

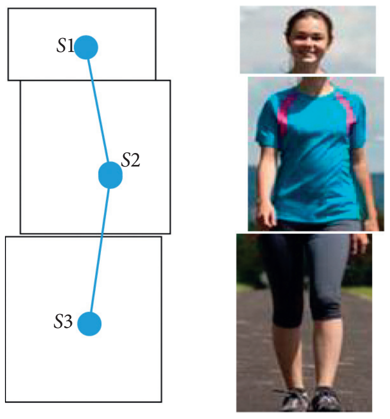

Figure 3: Some structure II partition objects.

$$\theta_{ij}^0 = \left(S_i^0, S_j^0\right), \text{where } i, j = \overline{1, n}. \quad (2)$$

For example, the angle of rotation from the center of S_2^0 compared to the rest of S_1^0, S_3^0 is the angle $\theta_{12}^0, \theta_{32}^0$ according to Figure 4(a).

Step 2 : we suppose that $S_1^k, S_2^k, \ldots, S_n^k$ are estimates and $S_1^0, S_2^0, \ldots, S_n^0$ are parts of the object in the original image. The distance is calculated as $\text{distance}_i = \|\text{HOG}(S_i^k) - \text{HOG}(S_i^0)\|^2$, $i = \overline{1, n}$.

However, because the division of parts may not be equal, to determine which is best, we multiply the coefficient by each distance and, then, compare $K_1 * \text{distance}_1, K_2 * \text{distance}_2, \ldots, K_n * \text{distance}_n$. The smallest value is considered the best estimate. The best estimate is placed at the kth frame as $S_{\min}^k$.

Step 3 : when the best part $(S_{\min}^k)$ was selected, we performed the center rotation of the remaining $n-1$ relative to the best part $(S_{\min}^k)$ with the rotation angle defined. The result is the new center coordinates of the $n-1$ part.

find the new center coordinates of the part S_i^k with $i = 1, (n-1)$ by calculating the center rotation of the section S_i^k compared to the center of the part $S_{\min}^k$ with the angle of rotation $(\theta_{i\min}^k - \theta_{i\min}^0)$ as follows:

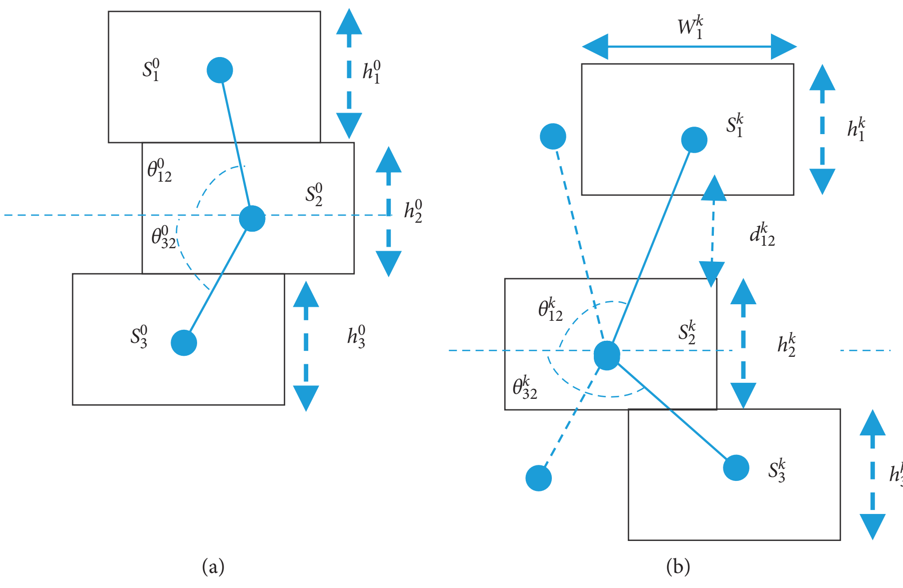

FIGURE 4: (a) The angle $\theta_{12}^0, \theta_{32}^0$ is the angle between the center S_2^0 and the center S_1^0, S_3^0 based on Frame 0, and h_1^0, h_2^0, h_3^0 are the heights of S_1^0, S_2^0, S_3^0. (b) In the kth frame, the parts S_1^k, S_3^k have been skewed from S_2^k with angle $\theta_{12}^k, \theta_{32}^k$, and h_1^k, h_2^k, h_3^k are the heights of S_1^k, S_2^k, S_3^k.

$$\overrightarrow{v_{S_{i'}^k}} = \begin{pmatrix} \cos\left(\theta_{i\min}^k - \theta_{i\min}^0\right) & -\sin\left(\theta_{i\min}^k - \theta_{i\min}^0\right) \\ \sin\left(\theta_{i\min}^k - \theta_{i\min}^0\right) & \cos\left(\theta_{i\min}^k - \theta_{i\min}^0\right) \end{pmatrix} . \overrightarrow{v}_{S_i^k}, \tag{3}$$

where $\overrightarrow{v}_{S_i^k}$ is the center coordinate of the part S_i^k and $\overrightarrow{v_{S_{i'}^k}}$ is the new center coordinate of the part S_i^k after performing the rotation.

For example, according to Figure 4(b), the new center coordinates of S_1^k, S_3^k parts are found compared to S_2^k centers with the rotation angle $(\theta_{12}^k - \theta_{12}^0)$ and $(\theta_{32}^k - \theta_{32}^0)$ as

$$\overrightarrow{v_{S_{1'}^k}} = \begin{pmatrix} \cos\left(\theta_{12}^k - \theta_{12}^0\right) & -\sin\left(\theta_{12}^k - \theta_{12}^0\right) \\ \sin\left(\theta_{12}^k - \theta_{12}^0\right) & \cos\left(\theta_{12}^k - \theta_{12}^0\right) \end{pmatrix} . \overrightarrow{v}_{S_1^k},$$

$$\overrightarrow{v_{S_{3'}^k}} = \begin{pmatrix} \cos\left(\theta_{32}^k - \theta_{32}^0\right) & -\sin\left(\theta_{32}^k - \theta_{32}^0\right) \\ \sin\left(\theta_{32}^k - \theta_{32}^0\right) & \cos\left(\theta_{32}^k - \theta_{32}^0\right) \end{pmatrix} . \overrightarrow{v}_{S_3^k}. \tag{4}$$

Step 4 : translating the particles progress of part S_i^k with $i = \overline{1, (n-1)}$ which differs from the best part $(S_{\min}^k)$ to the position of n-1 new part $(S_{i'}^k)$ is created out through Step 3.

Step 5: the sizes of $S_1^k, S_2^k, \ldots, S_n^k$ parts are scaled according to the ratio of $S_1^0, S_2^0, \ldots, S_n^0$. That is, we will calculate h_i^k and w_i^k with $i = \overline{1, (n-1)}$ of the sections $S_1^k, S_2^k, \ldots, S_n^k$.

First, we find the horizontal dimensions of each S_i^k part as follows: $w_i^k = \sum_{i=1}^n w_i^k / n$ with $i = \overline{1, (n-1)}$.

Next, we find the height dimensions of each S_i^k part as follows: $h_i^k = (w_i^k * h_i^0 / w_i^0)$.

Step 6 : translating the particles progress of part S_i^k with $i = \overline{1, (n-1)}$ compared to the best part (S_2^k) with distance $d_{i\min}^k$.

For example, as shown in Figure 4(b), the particles of S_1^k are translated towards the best part (S_2^k) with about d_{12}^k.

We propose the multiparticle filter algorithm in Algorithms 2 and 3.

4. Experiment

4.1. Environment. Installation environment: we experiment on computers using the Windows 10 Pro 64 bit, RAM 8 GB, Chip Intel Core (TM) 5i-3210M CPU @ 2.5GHz; Matlab programming language version R2016a.

4.2. Data Set. In 2013, Wu et al. [33] gathered many video sources related to the track and proceeded to create ground truth for these videos to form the TB-100 dataset. Because the TB-100 is a compilation of data from many sources, the context of the videos is also very different and diverse in attributes such as the type of objects to track, color or black-and-white videos, and still or dynamic cameras. The video datasets used to support the findings of this study have been deposited in http://www.visual-tracking.net.

Challenges in the dataset include the following:

IV- illumination variation: the brightness of the subject varies significantly

SV- scale variation: the ratio of the rectangle containing the first image object to the current image is out of range $[(1/t_s), t_s], t_s > 1\, (t_s = 2)$

OCC- occlusion: the object is partially or completely obscured

```
Input: pf particles sample set (based on Algorithm 4), $k$ th image (where $k$ starts from the second image)
Output: The new set of particles represents $p((x, y, \omega, h)|\text{image}\,k$, estimate the state of the object in the $k$ th image.
Step 1:
  for $i = 1$ to $N_s$ do
  Beginfor
    /*(with $(x_i, y_i, \omega_i, h_i)$ is the $i$ th particle)*/
    Get $x_{\text{new}} \sim x_i + N(0, \sigma_x^2)$
    Get $y_{\text{new}} \sim y_i + N(0, \sigma_y^2)$
    Get $\omega_{\text{new}} \sim \omega_i + N(0, \sigma_\omega^2)$
    Calculate $h_{\text{new}} = \eta * \omega_{\text{new}}$
    Calculate likelihood = $p(\text{image2}|(x_{\text{new}}, y_{\text{new}}, \omega_{\text{new}}, h_{\text{new}})$ by Algorithm 8
    Update weight for $i$ th particle: $\text{weight}_i = \text{weight}_i * \text{likelihood}$
  Endfor
Step 2:
  Calculate sum $\text{sw} = \sum_{j=1}^{N_s} \text{weight}_i + \varepsilon$
  for $i = 1$ to $N_s$ do
  Beginfor
    Standardize weight: $\text{weight}_i = (\text{weight}_i + \varepsilon/\text{sw})$
  Endfor
Step 3: Calculate $N_{\text{eff}} = (1/\sum_{j=1}^{N_s} \text{weight}_j^2)$
Step 4:
  if $N_{\text{eff}} < N_s/2$ then
    $\{(x_i, y_i, \omega_i, h_i), \text{weight}_i\}_{i=1}^{N_s} = \text{RESAMPLE}(\{(x_i, y_i, \omega_i, h_i), \text{weight}_i\}_{i=1}^{N_s})$ use boostrap
  Endif
Step 5: Estimate the state of the object in the $k$ th image by calculating the average of the new set of particles
EstimatedStatus = $\sum_{i=1}^{N_s} \text{weight}_i * \text{particle}[i]$
```

Algorithm 2: Particle filter for random processes $(x_n, y_n, \omega_n, h_n)$.

```
Step 1: Initialize $N$-particles set $pf_1, pf_2, \ldots, pf_n$
Step 2:
  for $i = 1$ to $n$ do
  Beginfor
    Take $D_i$ patern for part $i$ according to Algorithm 4.
    Train strong classification $F_i = \text{gentleAdaboost}(D_i)$ according to Algorithm 5.
  Endfor
Step 3:
  while The video is not over do
  Beginwhile
    Get observation photos obs
    for $i = 1$ to $n$ do
    Beginfor
      Use particle filter to estimate the $i$th state according to Algorithm 1 or Algorithm 2
        $S_i^* = (x_i^{\text{est}}, x_i^{\text{est}}, x_i^{\text{est}}, x_i^{\text{est}}) = \text{particlefilter}(obs, pf_i)$
      Measure the distance: $\text{distance}_i = \|\text{HOG}(S_i^*) - \text{HOG}(S_i^0)\|^2$
    Endfor
    Choose the best part $i_0 = \arg\min(K_1 * \text{distance}_1, K_2 * \text{distance}_2, \ldots, K_n * \text{distance}_n)$
    Using center $i_0$ part and structure H perform rotation of the remaining $n-1$ centers.
    Tranlate $n-1$ other particle set $i_0$ to $n-1$ has just been centered on $i_0$.
    Scale the particle sets according to the structure ratio H.
    Translating $n-1$ set of other particles $i_0$ toward the part $i_0$.
    for $i = 1$ to $n$ do
    Beginfor
        Take a new sample $D_i^*$ based on the new section rotated from $i_0$.
        Update strong classification $F_i$ according to Algorithm 6
    Endfor
  Endwhile
```

Algorithm 3: (MultiPart).

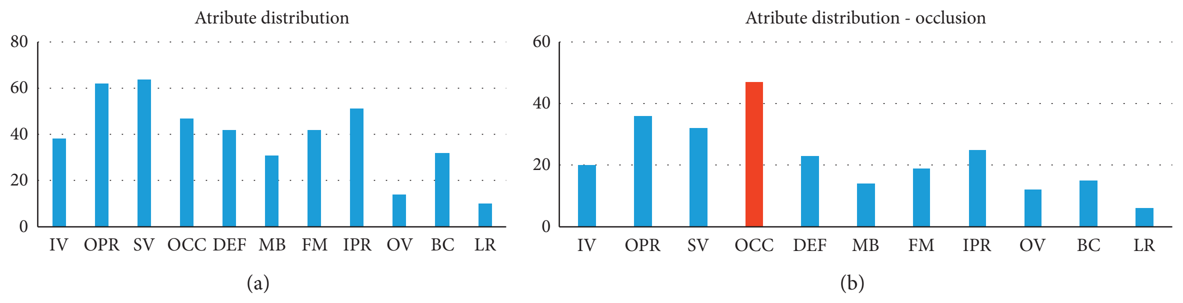

Figure 5: (a) Distribution of the properties throughout the data set. (b) Distribution of attributes in videos that have OCC attributes.

DEF- deformation: nonsolid objects that change shape

MB- motion blur: the subject is blurry due to camera movement

FM- fast motion: groundtruth motion is greater than t_m pixels ($t_m = 20$)

IPR- in-plane rotation: objects rotate in the image domain

OPR- out-of-plane rotation: object out of the image domain

OV- out of view: part of the object out of the image domain

BC- background clutters: the background near the object has the same color or line as the object

LR- low resolution: the number of pixels in the rectangle that contains the object (considering ground truth) is less than t_r ($t_r = 400$)

The abovementioned challenges are distributed in the data set, which is shown in Figure 5.

4.3. Evaluating. We use the evaluation criteria presented at the site [33] to evaluate the tracking algorithm.

Method 1 (R_1): evaluation based on the Euclid distance (precision plot): we measure the distance Euclid d from the estimated center of the algorithm to the actual center of the object (ground truth), if d is less than or equal to a threshold t_0. The view is successful according to Figure 6(a).

Method 2 (R_2): evaluation based on levels of overlap (success plot): the number of overlapping points is defined as $= (|r_t \cap r_a|/|r_t \cup r_a|)$, in which r_t is the bounding rectangle determined by the algorithm and r_a is the ground truth rectangle according to Figure 6(b).

We calculate the ratio of R_1, R_2 by (the number of successful images/the total number of images of *images*).

4.4. Result. The results of tracking people with the camera do not fluctuate much, the rotation angle is conserved, and the proportions on the body of people and people are not too small.

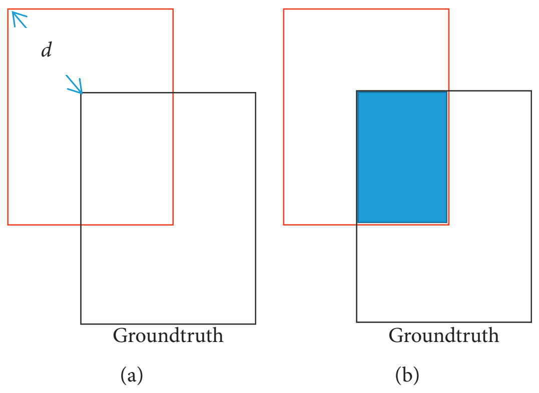

Figure 6: (a) Euclid distance measurement. (b) Measure of the level of overlap.

We named Program 1 as MultiPart3 using the MultiPart algorithm by dividing the object into 3 parts in a ratio of 1 : 5 : 3; Program 2 is MultiPart3_direction using the MultiPart algorithm to calculate the direction of moving objects by dividing the object into 3 parts in a ratio of 1 : 5 : 3; Program 3 is MultiPart2 using the MultiPart algorithm by dividing the object into 2 parts in a ratio of 5 : 3; Program 4 is MultiPart2_direction using the MultiPart algorithm to calculate the direction of object movement by dividing the object into 2 parts in a ratio of 5 : 3. The abovementioned five programs compared with DiMP algorithms [31] and GradNet [32] are shown in Table 1 and Figure 7.

The MultiPart2 algorithm uses 2 particle filters in a ratio of 5 : 3 to track, allowing a large portion of the head (head and body) to be more informative, less changing over time, and “denser” than the leg. The average accuracy result ($R1 = 92.2\%$, $R2 = 87.9\%$) is slightly larger than that of the GradNet algorithm ($R1 = 85.9\%$, $R2 = 86.3\%$). With the abovementioned results, we can see that the tracking part has much information for good average results compared with the object tracking. However, for data (Dancer và Dancer2) that have a tracking object who wears a skirt or long skirt covering feet, tracking using 2 particle filters at a ratio of 5 : 3 gives a low result compared to an object trace. By tracking, the object intact in this case is best.

For the videos mentioned above, the MultiPart3 algorithm divides 3 parts in a 1 : 5 : 3 ratio, after each image has adjustments of the parts according to the rotation technique

Table 1: Algorithm results of DiMP, GradNet, MultiPart3, MultiPart3_direction, MultiPart2, and MultiPart2_direction.

Video	DiMP (%)		GradNet (%)		MultiPart2 (%)		MultiPart2_direction (%)		MultiPart3 (%)		MultiPart3 direction (%)	
	*R*1	*R*2	*R*1	*R*2	*R*1	*R*2	*R*1	*R*2	*R*1	*R*2	*R*1	*R*2
Crossing	100	100	100	100	100	98.3	100	93.3	100	100	100	100
Dancer	93.3	100	92.4	98.2	95.1	88.9	91.1	88.4	83.1	86.2	64.9	74.2
Dancer2	97.3	99.3	100	100	77.3	59.0	47.3	57.3	100	98.7	96.0	96.7
David3	90.9	90.9	100	100	73.0	81.4	64.3	67.9	80.6	84.5	92.5	92.5
Human8	100	100	8.6	7.0	100	89.1	100	99.2	100	81.3	100	89.8
Walking	100	99.8	100	99.3	100	98.5	100	98.8	99.8	99.0	100	99.3
Walking2	42.2	39.6	100	100	100	100	100	100	98.8	96.0	100	100
	89.1	89.9	85.9	86.4	92.2	87.9	86.1	86.4	94.6	92.2	93.3	93.2

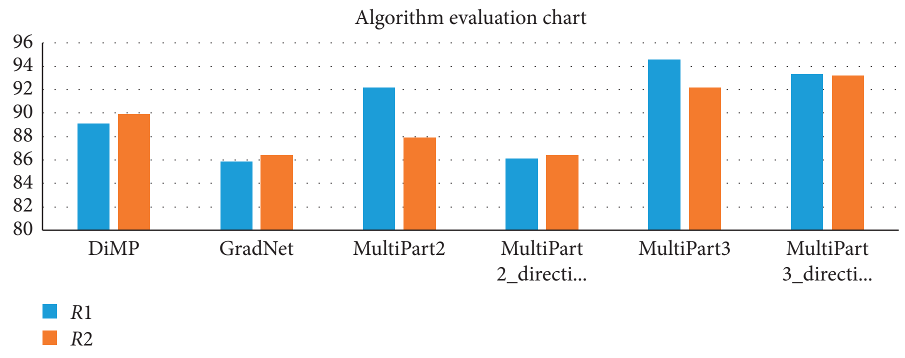

FIGURE 7: Algorithm evaluation chart.

TABLE 2: Notation table.

Symbol	Explanation
$(x_k, y_k, \omega_k, h_k)$	The state of the object in the frame k
η	The ratio between the weight/height of the object at the first frame
ε	Epsilon
Δx	Dirac delta function at x
Δ	Kronecker function
$N(0, \sigma_x^2)$	Gaussian transition function
HOG	Histogram of oriented gradients
N_{eff}	Effective sample size

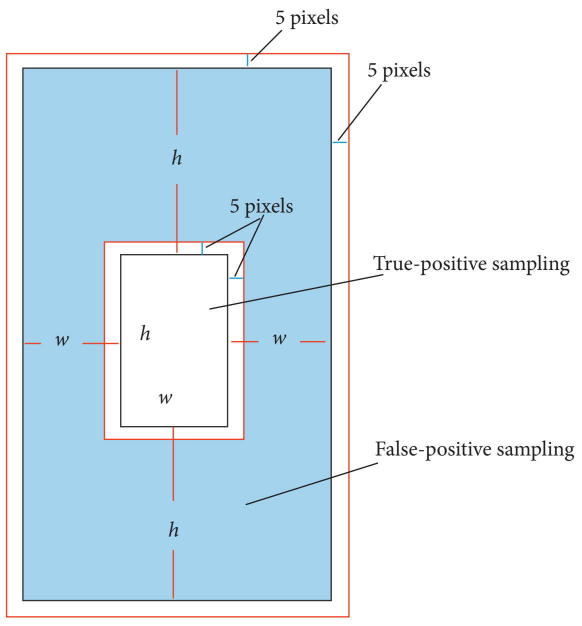

FIGURE 8: Limitations for false-positive and true-positive sampling [19].

based on the structure H to bring the wrong parts to the right area of ground truth. The results of the MultiPart3 algorithm ($R1 = 94.6\%$ and $R2 = 92.2\%$) are better than those of DiMP ($R1 = 89.1\%$, $R2 = 89.9\%$) and GradNet ($R1 = 85.9\%$, $R2 = 86.3\%$). In addition, the algorithm MultiPart3_direction is the combination of the object movement direction for average accuracy results ($R1 = 93.3\%$, $R2 = 93.2\%$), and this result is better than DiMP and GradNet algorithm's result. The MultiPart3_direction algorithm approximates the average accuracy result with the MultiPart3 algorithm because the dancer data have the human object that jumps up and down suddenly and the refined data take a number of frames so that the determination of the motion direction is wrong.

5. Conclusions

This paper presented several methods for object tracking in the videos mainly related to particle filters. To solve the general problem, we built a hidden Markov model and applied particle filters. For tracking human videos in normal condition where the human scale is preserved, we used 3 particle filters to track each part of the body or track the part of the body containing the most information, which will, then, infer to the whole body. Experimental results show dividing the object into $(n + m)$ parts, even when n parts of objects are partially obscured and the remaining m parts are tracked normally and do not affect the tracking of the subject in the video.

The development direction of this paper is to change the observation model. We found that the gentle Adaboost training process is time consuming. However, the algorithms using correlation filters have the advantage of being fast and highly accurate. For future studies, we suggest integrating the correlation filters into the observation model to shorten the execution time. In addition, we are planning to study parts of the traced objects in parallel to shorten the execution time.

Appendix

Table 2 describes the notations.

A. Sample to Train Gentle Adaboost

Figure 8 describes the random true-positive sampling with the bordering error at ±5 pixels and false-positive sampling which does not include any particular object.

```
Input: Image at initial time, position and object size $(x_0, y_0, \omega_0, h_0)$
Output: Data set $D$ size $N_+ + N_-$, including $N_+$ positive patern, $N_-$ negative patern.
Step 1: Initialize a feature stack to hold feature vectors.
Step 2: Get the Haar-like characteristics [34] of the area $(x_0, y_0, \omega_0, h_0)$, $v$ = getHaarlike(the area $(x_0, y_0, \omega_0, h_0)$)
        Put $v$ on stack features, features.push ($v$)
Step 3:
  for $i = 1$ to $N_+ - 1$ do
  Beginfor
    Randomly draw rectangular area $S$ from the image at a position of ±5 pixels from the object
    Get the Haarlike feature vector over the region $S$, $v = getHaarlike(S)$
    Put vector $v$ in the stack features, features.push(v)
  Endfor
Step 4:
  for $i = 1$ to $N_- \_1$ do
  Beginfor
    Randomly take the area of $S$ rectangle in the green part of Figure 8
    Get the Haarlike feature vector over the region $S$, $v = getHaarlike(S)$.
    Put vector $v$ in the stack features, features.push ($v$)
  endfor
```

Algorithm 4: Sample to train gentle Adaboost.

B. Create Gentle Adaboost

```
Input: Training set $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$ with $y_i \in \{1, -1\}$ is the label of $x_i$
Output: Strong classification function $F$
Step 1: Initialize coefficients $\omega_1 = \omega_2 = \cdots = \omega_N = (1/N)$ for training set
Step 2:
  for $t = 1$ to $s$ do
  Beginfor
    for $j = 1$ to $m$ do
    Beginfor
      Training weak classification $h_j$, mean that calculate $(a, k, \theta, b)$ according to the formula
    Endfor
    Select the weak classification with the lowest error, set to $f_t$
    Update strong classification $F(x) = F(x) + f_t(x)$
    Update weight $\omega_i = \omega_i \times \exp(-y_i f_t(x_i))$
  Endfor
```

Algorithm 5: Gentle Adaboost [35].

C. Update Gentle Adaboost

```
Input: Old strong classification $F$ is a $4 \times s$ matrix, new $D$ data set
Output: A new strong classification $F^*$
Step 1:
  Initialize the weight set $(w_1, w_2, \ldots, w_N)$ consisting of $N$ equal numbers, that equal $1/N$, where $N$ is the number of elements in the data set.
  Initialize matrix $F^*$ size $4 \times s$ to contain weak classifications
  Initializes the chosen stack to save the position of the weak classification that have been selected
Step 2:
  for $i = 1$ to $s$-$T$ do
  Beginfor
```

Algorithm 6: Continued.

```
        Initialize the loss stack to store error values
        for $j = 1$ to $s$ do
        Beginfor
          if $-(j \in \text{chosen})$ then
            Calculate the error of classification $j$ according to the following formula:
              $J_j = \sum_{m=1}^{N} w_m \cdot (y_m - a_j \cdot (D[m][k_j] > \theta_j) - b_j)^2$
            Put $J_j$ into loss, loss.push($J_j$)
          Endif
        Endfor
            Select the classification with the smallest error value, $j_0 = \arg\min(\text{loss})$
        Put $j_0$ into loss, loss.push ($j_0$)
        for $j = 1$ to $s$ do
        Beginfor
          Update weight $w_j = w_j \times \exp(-y_j \cdot (a_{j_0}(D[j][k_{j_0}] > \theta_{j_0}) + b_{j_0}))$
        Endfor
      Endfor
    Step 3: freshly train weak classification $T$ on data set $D^*$ by Algorithm 5
    Step 4: combine $s$-$T$ weak classification in Step 2 and $T$ weak classification in Step 3 to create a new strong classification $F^*$.
```

Algorithm 6: Update gentle Adaboost.

D. Calculate the Classification Point

```
Input: Strong classification $F$ is a $4 \times s$ matrix (find strong classification based on Algorithm 5), Haar-like $x$ feature vector of area to be calculated
Output: Classification point conf
Step 1: Assign point conf = 0
Step 2:
  for $i = 1$ to $s$ do
  Beginfor
    Get $a, b, \theta, k$ are $i$ th classification
    Update conf = conf + $a \times \delta(x[k] > \theta) + b$
  Endfor
```

Algorithm 7: Calculate the Classification point.

E. Calculate the Likelihood of the Image on the Hypothesis of the State of the Object

```
Input: Observation image, hypothesis$(x, y, \omega, h)$, vector $\text{HOG}_1$ of object in the first image, strong classification $F$.
Output: Likelihood $p$(observated image| hypothesis$(x, y, \omega, h)$)
Step 1: Extract image area in rectangle $(x, y, \omega, h)$, $h$ = crop(image, $(x, y, \omega, h)$)
Step 2: Resize image $h$ for the size of the object in the first image
Step 3:
  Extract feature HOG on $h$, $\text{HOG}_2$ = getHOG($h$)
  Extract feature Haar-like on $h$, $\text{Haar}_2$ = getHaarlike($h$)
Step 4: Calculate classification point conf = $F(\text{Haar}_2)$ by Algorithm 7
Step 5: Calculate likelihood = $\exp(\alpha * \text{conf} - \gamma \|\text{HOG}_1 - \text{HOG}_2\|^2)$
```

Algorithm 8: Calculate the likelihood of the image on the hypothesis of the state of the object.

Data Availability

Public data were used to research. The [TB-100] data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] F. Boukari and S. Makrogiannis, "Spatio-temporal level-set based cell segmentation in time-lapse image sequences," *International Symposium on Visual Computing*, Springer, Berlin, Germany, 2014.

[2] O. Dzyubachyk, W. A van Cappellen, J. Essers, W. J. Niessen, and E. Meijering, Niessen and E. Meijering, "Advanced level-set-based cell tracking in time-lapse fluorescence microscopy," *IEEE Transactions on Medical Imaging*, vol. 29, no. 3, 2010.

[3] N. Seenouvong, U. Watchareeruetai, C. Nuthong, K. Khongsomboon, and N. Ohnishi, "A computer vision based vehicle detection and counting system," in *Proceedings of the International Conference on Knowledge and Smart Technology (KST)*, Bangkok, Thailand, October 2016.

[4] R. H. Pena-Gonzalez and M. A. Nuño-Maganda, "Computer vision based real-time vehicle tracking and classification system," in *Proceedings of the International Midwest Symposium on Circuits and Systems (MWSCAS)*, College Station, TX, USA, August 2014.

[5] J. Komorowski, G. Kurzejamski, and G. Sarwas, "BallTrack: football ball tracking for real-time CCTV systems," in *Proceedings of the International Conference on Machine Vision Applications (MVA)*, Tokyo, Japan, May 2019.

[6] M. Xu, O. James, and G. Jones, "Tracking football players with multiple cameras," in *Proceedings of the International Conference on Image Processing*, Varzim, Portugal, August 2004.

[7] T. Seidl, M. Völker, N. Witt et al., "Evaluating the indoor football tracking accuracy of a radio-based real-time locating system," in *Proceedings of the International Symposium on Computer Science in Sports (ISCSS)*, Loughborough, UK, September 2015.

[8] K. Kapinchev, A. Bradu, F. Barnes, and P. Adrian, "GPU implementation of cross-correlation for image generation in real time," in *Proceedings of the International Conference on Signal Processing and Communication Systems (ICSPCS)*, Cairns, Australia, December 2015.

[9] C. Wang, L. Zhang, J. Yuan, and L. Xie, "Kernel cross-correlator," in *Proceedings of the Thirty-second AAAI Conference on Artificial Intelligence*, pp. 4179–4186, Association for the Advancement of Artificial Intelligence, New Orleans, LA, USA, Feburary 2018.

[10] D. S. Bolme, B. A. Draper, and J. Ross Beveridge, "Average of synthetic exact filters," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, June 2009.

[11] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proceedings of the Computer Vision-ECCV 2012. European Conference on Computer Vision*, pp. 702–715, Florence, Italy, October 2012.

[12] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.

[13] D. S. Bolme, J. Ross, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Regconition*, pp. 2544–2550, San Francisco, CA, USA, June 2010.

[14] S. Avidan, "Ensemble tracking," in *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence*, New Jerssey, NJ, USA, December 2007.

[15] Y. Freund, "An adaptive version of the boost by majority algorithm," *Machine Learning*, vol. 43, no. 3, pp. 293–318, 2001.

[16] Z. Chen, "Bayesian filtering: from kalman filters to particle filters, and beyond," *Statistics: A Journal of Theoretical and Applied Statistics*, vol. 182, no. 1, 2003.

[17] A. Doucet and A. M. Johansen, "A tutorial on particle filtering and smoothing: fifteen years later," 2009.

[18] D. Jurafsky and H. M. James, "Hidden Markov models," *Speech and Language Processing*, Pearson, London, UK, 2017.

[19] D. A. Klein, D. Schulz, S. Frintrop, and A. B. Cremers, "Adaptive real-time video-tracking for arbitrary objects," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, MO, USA, November 2010.

[20] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *Proceedings of the Computer Vision - ECCV 2002. European Conference on Computer Vision*, pp. 661–675, Copenhagen, Denmark, May 2002.

[21] C. Yang, D. Ramani, and L. Davis, "Fast multiple object tracking via a hierarchical particle filter," *Proceedings of the Tenth IEEE International Conference on Computer Vision*, pp. 212–219, Beijing, China, October 2005.

[22] O. Samualssonn, "Video tracking algorithm for unmanned aerial vehicle surveillance," M. Sc. thesis, KTH Electrical Engineering, Stockholm, Sweden, 2012.

[23] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-convolutional siamese networks for object tracking," in *Proceedings of the ECCV*, Amsterdam, Netherlands, October 2016.

[24] J. Valmadre, L. Bertinetto, J. F. Henriques, A. Vedaldi, and P. H. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proceedings of the CVPR*, Honolulu, Hawaii, July 2017.

[25] Q. Wang, J. Gao, J. Xing, M. Zhang, and W. Hu. Dcfnet, "Discriminant correlation filters network for visual tracking," 2017, http://arxiv.org/abs/1704.04057.

[26] Q. Wang, Z. Teng, J. Xing, J. Gao, W. Hu, and S. Maybank, "Learning attentions: residual attentional siamese network for high performance online visual tracking," in *Proceedings of the CVPR*, Salt Lake City, UT, USA, June 2018.

[27] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, "Distractor-aware siamese networks for visual object tracking," in *Proceedings of the ECCV*, Munich, Germany, September 2018.

[28] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proceedings of the CVPR*, Salt Lake City, UT, USA, June 2018.

[29] R. Tao, E. Gavves, and A. W. M. Smeulders, "Siamese instance search for tracking," in *Proceedings of the CVPR*, Las Vegas, NV, USA, July 2016.

[30] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang, "Learning dynamic siamese network for visual object tracking," in *Proceedings of the ICCV*, Venice, Italy, October 2017.

[31] G. Bhat, M. Danelljan, L. Van Gool, and T. Radu, "Learning discriminative model prediction for tracking," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Seoul, Korea, November 2019.

[32] P. Li, B. Chen, W. Ouyang, D. Wang, X. Yang, and H. Lu, "GradNet: gradient-guided network for visual object tracking," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Seoul, Korea, November 2019.

[33] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: a benchmark," in *Proceedings of the Computer Vision and Pattern Recognition*, Portland, OR, USA, June 2013.

[34] V. Paul and M. J. Jones, "Robust real-time object detection," in *Proceedings of the International Journal of Computer Vision*, Vancouver, Canada, July 2001.

[35] A. J. Ferreira and M. A. T. Figueiredo, *Boosting Algorithms: A Review of Methods, Theory,and Applications*, Springer US, New York, NY, USA, 2012.