Stacked Multi-layer Self-Organizing Map for Background Modeling

Zhenjie Zhao, Xuebo Zhang, *Member, IEEE,* and Yongchun Fang, *Senior Member, IEEE*

***Abstract*—In this paper, a new background modeling method called *Stacked Multi-layer Self-Organizing Map Background Model* (SMSOM-BM) is proposed, which presents several merits such as strong representative ability for complex scenarios, easy to use, and so on. In order to enhance the representative ability of the background model and make the parameters learned automatically, the recently developed idea of Representative Learning (or Deep Learning) is elegantly employed to extend the existing single-layer Self-Organizing Map Background Model to a multi-layer one (namely, the proposed SMSOM-BM). As a consequence, SMSOM-BM gains several merits including strong representative ability to learn background model of challenging scenarios, and automatic determination for most network parameters. More specifically, every pixel is modeled by a *Stacked Multi-layer Self-Organizing Map* (SMSOM), and spatial consistency is considered at each layer. By introducing a novel over-layer filtering process, we can train the background model layer by layer in an efficient manner. Furthermore, for real-time performance consideration, we have implemented the proposed method using NVIDIA CUDA platform. Comparative experimental results show superior performance of the proposed approach.**

.

***Index Terms*—Background modeling, representative learning, self-organizing map.**

I. Introduction

BACKGROUND modeling is a critical component for motion detection tasks and it is essential for most of modern video surveillance applications. Given one scenario $\mathcal{S}$, *background modeling* aims at constructing a proper model to represent it even under such challenging cases as dynamic background, camera jitter, and so on. With this model, we can find the *changing region* of a new input image by comparing this image with the background model through some proper metrics. In general, this process is called *foreground detection* or *background subtraction*, and the output result is a binary image where the part with 'white' color is utilized to stand for the changing region (*foreground*) and the 'black' part stands for the unchanging region (*background*). Using this binary image, a lot of computer vision problems can be solved in an easier way, such as object tracking, three-dimensional (3D) reconstruction, object recognition, and so on.

This work is supported in part by National Natural Science Foundation of China under Grant 61203333 and 61325017, in part by Specialized Research Fund for the Doctoral Program of Higher Education of China under Grant 20120031120040, and in part by Tianjin Natural Science Foundation under Grant 13JCQNJC03200.

Z. Zhao, X. Zhang(✉) and Y. Fang are with Institute of Robotics and Automation System and also Tianjin Key Lab of Intelligent Robotics, Nankai University, Tianjin, China, 300071

E-mail: zhangxb@robot.nankai.edu.cn

In general, background modeling methods can be classified as the following three categories:

1. Statistical methods, which try to model a distribution of the background [1], [2], [3], [4], [5], [6]. Typical approaches include Gaussian Mixture Model (GMM) [2] and Kernel Density Estimation (KDE) [4]. An excellent survey on this field can be found in [7]. In [8], background is modeled as a fuzzy GMM, and the work in [9] further extends this model by introducing spatial and temporal consistency constraints. In [10], background model is obtained by a bidirectional analysis based on probability framework, and with some time latency, the output accuracy is improved;

2. Dimension reduction methods, which try to describe the background model in a low dimensional subspace [11], [12], [13], [14], [15], [16], such as Principle Component Analysis (PCA) [12], Independent Component Analysis (ICA) [14]. The authors in [13] formulate background modeling as a Robust Principle Component Analysis (RPCA) problem, which can be successfully solved by some effective optimization algorithms. In [15] and [16], RPCA is reformulated by the Bayesian framework, and more prior information like the sparsity of the background can be added elegantly. Recent advances in this field can be found in [17] and references therein. In [18], the background is modeled by compressed images, and thanks to recent advances on compressive sensing, effective ways of foreground detection are available;

3. Heuristic methods or intelligent methods [19], [20], [21], [22], [23], [24], [25], which combine some intuitive rules [20], or use some general nonlinear model like Artificial Neural Network (ANN) [21] to fulfill the foreground detection task.

Though many background modeling methods have been proposed over the years [26], [27], [28], [29], [30], [31], obtaining a general model which can describe a diversity of scenarios is still a challenging task. What's more, tuning parameters for different scenarios is also sophisticated and time-consuming. In this paper, we try to formulate background modeling as a supervised machine learning problem to deal with this issue. In other words, given a sequence of training images that can be labeled as background, our goal is to find a proper model to represent these data. In addition, we should also have an effective way to train this model, at the same time, the parameters should be learned automatically. Whenever a new input image comes, the model should be able to predict the background region and the foreground region correctly.

From previous analysis, we know that, in order to get accurate background subtraction results, a background model which has a strong representative ability is required. In this paper, we propose a *Stacked Multi-layer Self-Organizing Map*

(SMSOM) model, which is organized by stacking the existing Self-Organizing Map (SOM) (see [32]) elegantly with a newly proposed over-layer deep filter (see III-B), and it is trained layer-wise under the supervised representative learning framework. Furthermore, by modeling every pixel as a SMSOM, we obtain the *Stacked Multi-layer Self-Organizing Map Background Model* (SMSOM-BM) for the whole scenario. After training this model with background images, we can use it to fulfill the foreground detection task. Experiment results compared with mainstream approaches show the proposed model can deal with very challenging scenarios. The main contribution of this paper can be summarized as follows:

1. A new *Stacked Multi-layer Self-Organizing Map* (SMSOM) architecture is proposed, and by modeling every pixel as a SMSOM, we obtain a novel background model, SMSOM-BM. The underlying idea of this new model is inspired by the existing SOM and the recently developed deep learning approaches, and thus the newly proposed method gains some inherent advantages such as strong representative ability than SOM, low complexity, and so on;

2. An effective way of training SMSOM is proposed. We introduce an over-layer deep filter between two successive layers of SMSOM, which can filter out the data that cannot be represented well by the current layer, and send these data to the next layer. This process helps us train a deep SMSOM efficiently;

3. Most of the network parameters are learned automatically, and only two parameters of learning rates are needed, which makes the proposed method much easier to use.

The rest of this paper is organized as follows. Section II gives a brief review of some related works. The proposed *Stacked Multi-layer Self-Organizing Map Background Model* is presented in section III. Some implementation details, together with experiments are shown in section IV. Section V concludes the paper.

II. Related Works

Over the years, numerous background modeling methods have been proposed, and a thorough review is apparently beyond the scope of this paper. Readers can refer to [33] for an overview of this topic. In this section, we will first present some background modeling methods on neural networks. And then, some related works that directly motivate this paper will be given.

In [34], each pixel is modeled as an Adaptive Resonant Theory (ART) network. By introducing some new mechanisms (ART+), such as neural merging process, ART+ can get better performance than ART and some traditional methods. In [35], background modeling is formulated as a 3-layer neural network (multi-layer perceptrons, MLPs). The input data can be seen as some hand-designed features by considering spatio-temporal constraints, and the output is represented as *background* and *foreground* labels. By using only one neural network to model every pixel position, computation cost and memory requirement are reduced. In [22], [23], the authors modeled every pixel (transformed to HSV values) as a SOM [32]. The training process of SOM is accomplished by finding the activated node called *winner* which has the smallest distance with the model vectors. Once the distance is smaller than a threshold, the activated node with its neighbors are updated by the current pixel. One novel characteristic in [22] and [23] is that the neighbors of the *winner* node may contain SOM model nodes of the current pixel's neighbor pixels, and with that mechanism, the method can model spatial consistency of background images quite well. In order to lower the computational load in [22], the authors in [36] map each pixel to only one neuron node, and satisfactory experiment results are reported. The work in [37] extends the one-to-one SOM [36] to adjust the threshold parameters automatically by fuzzy approaches. More specifically, the threshold parameters are inferred on-line by the pre-designed fuzzy rules. Experiment results show that the fuzzy method is comparable to the one whose parameters are adjusted by humans off-line. In [38], a kind of neural networks called Retinotopic Self-Organizing Map (RESOM) is introduced to fulfill the background modeling task. More specifically, different color channels (red, blue, green) are modeled independently as different Self-Organizing Retinotopic Maps (SOMRs), and foreground extraction is done by combining the weights of different SOMRs.

While SOM [22], [23] successfully deals with such challenging scenarios as dynamic background, we believe that multi-layer ones could further improve the results with stronger representative ability. What's more, learning the threshold parameters of SOM Background Model automatically is also an important work, which can make the method much easier to use. We also try to deal with automatic parameters tuning in this paper.

On the other hand, some recent advances on Representative Learning [39], [40], also known as Deep Learning, show that a more accurate representative model can be learned by using deep network architectures, compared with shallow network ones. Some successful deep networks include Stacked Denoising Auto Encoder (SDAE) [41], Convolutional Neural Networks (CNN) and Deep Belief Network (DBN) [42]. Motivated by some key concepts in deep learning, namely, layer-wise training, pre-training, fine-tuning, we try to build a multi-layer SOM to model the background images. SOM is stacked with one another, and the training process of this multi-layer SOM model is layer-wise. After training the background model, it will be utilized to fulfill the foreground detection task. In addition, the background model is still updated during the foreground detection phase because test images contain both background region and foreground region, which can make the background model adapt to the scenario changes, as most of background modeling methods do. These two phases, training phase and on-line update phase can be seen as the pre-training and fine-tuning phases in typical deep networks. Pre-training initializes SMSOM as a good approximation of the training background images, while fine-tuning can make SMSOM to adapt the background changes on-line.

Another related work which tries to build a hierarchical SOM is called Growing Hierarchical Self-Organizing Map (GHSOM) [43], which is originally used for high dimensional data analysis. In [44], GHSOM is utilized to image segmen-

tation by modeling every pixel position as a GHSOM without spatial constraints. GHSOM tries to separate the training data into different layer SOM to model the hierarchical information in the data with a growing strategy, which means if one layer cannot represent the data well, another layer should be created. Our method has a similar *growing* characteristic but with a different strategy, more specifically, the structure of our model is fixed and the *growing* in our model can be seen as the learning process of parameters layer by layer. On the other hand, as we are trying to model background images, spatial constraints between neighbor pixels should be considered. In summary, our *Stacked Multi-layer Self-Organizing Map Background Model* is different from GHSOM at least in three points: 1. our model's structure is fixed, on the contrary, GHSOM's structure is determined by the training data, which may lead to over-fit problems easily *when it is applied to foreground detection task*; 2. we have a *fine-tuning* phase, which means that the model is updated on-line with testing data and this phase can make the model adapt to the scenario changes; 3. in our background model, every pixel corresponds to one *Stacked Multi-layer Self-Organizing Map* (SMSOM); in order to model the spatial relationship of the image, every pixel's SOM cooperates with each other on each layer, while the spatial constraint is not taken into account in GHSOM.

III. Stacked Multi-layer SOM Background Model

In this section, we will describe the proposed *Stacked Multi-layer Self-Organizing Map Background Model* (referred as SMSOM-BM) in detail. In general, SMSOM-BM should be trained using some background images first, and then, based on this pre-trained model, when a new test image comes, foreground detection can be conducted. To make the algorithm adaptive to background changes during foreground detection, the background model should be updated using testing images on-line, this process is generally called *background maintenance*. The whole flow diagram is shown in **Figure 1**. This section is organized as follows: the model structure is proposed in subsection III-A, while the training process, the detection process, and the maintenance process are presented in subsection III-B, III-C, III-D, respectively.

A. Model structure

The proposed SMSOM-BM consists of one *input layer* and several *computational layers* for every pixel position of one image, and the input data can be thought of as each pixel's three channel intensity values. The number of computation layer can be arbitrarily chosen. For example, if it is chosen as one layer, then SMSOM-BM with one layer becomes similar with existing background models in [22], [23], but foreground detection and background maintenance strategies are different . These improvements can drop off the manual tuning threshold parameters in [22], [23] and make the model more robust and easier to use. Details can be found in subsection III-C and III-D. Another important component is the over-layer deep filter $\mathcal{F}$, which can filter out the data that cannot be represented well by the current layer, and send these data to the next layer. Details can be found in subsection III-B. With

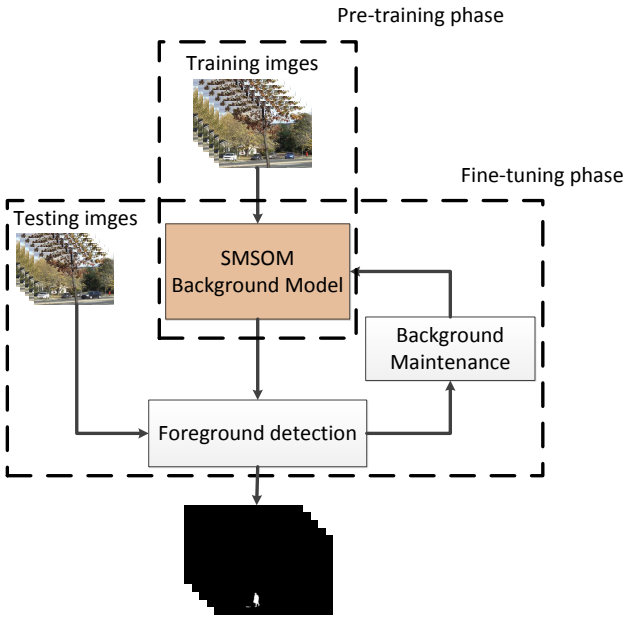

Fig. 1. Foreground detection by SMSOM-BM

$\mathcal{F}$, we divide the training data into different groups, where each computational layer handles one group. In this way, each layer is utilized to model a part of features for the background, and the hierarchical information is captured by our layer-wise training mechanism. Moreover, computational nodes at the same layer cooperate with each other to model the spatial constraints of images. Details can be found in subsection III-B. For clarity, we will refer SMSOM as one pixel's model, and SMSOM-BM as the whole background model consisting of all the single pixel's models.

Assuming that each pixel of the image (training images or testing images) has three values (transformed from RGB values to HSV values), namely, $h_i,\ s_i,\ v_i$, where $0 \leq h_i \leq 2\pi,\ 0 \leq s_i \leq 1,\ 0 \leq v_i \leq 1$, with i being the pixel position. For every computational layer, each input pixel i ($h_i,\ s_i,\ v_i$) is associated with 9 computational nodes in that layer, and each node is denoted by $node_q^{(l)}$ (the superscript l stands for the node belongs to layer l, the subscript q is the index of the node, $q = 1,\ 2, \ldots,\ 9$). These 9 nodes form the lth layer of SMSOM, and we refer it as *layer l of pixel i*, or *layer l* for simplicity. Each input value has a connectional weight with $node_q^{(l)}$, for example, the weights of $h_i,\ s_i$ and v_i are $\omega_{h_i q}^{(l)}$, $\omega_{s_i q}^{(l)}$ and $\omega_{v_i q}^{(l)}$, respectively, where $q = 1,\ 2, \ldots,\ 9$.

As an example, assuming that we have an image that only has two pixels. Each pixel is associated with one SMSOM model described previously, and these two SMSOM models compose the SMSOM-BM, as shown in **Figure 2**. For each layer of a L-layer SMSOM, we adopt a 2 dimensional SOM. Roughly speaking, $a_q^{(l)}$ or $b_q^{(l)}$ is the computational node. For convenience, we define the 8 *neighbors* of one computational node when different pixels are connected with each other at the same layer, for instance, the 8 neighbors of $a_6^{(1)}$ are $a_2^{(1)},\ a_3^{(1)},\ b_1^{(1)},\ b_4^{(1)},\ b_7^{(1)},\ a_9^{(1)},\ a_8^{(1)},\ a_5^{(1)}$, and the 8 neighbors of $a_5^{(2)}$ are $a_1^{(2)}, a_2^{(2)}, a_3^{(2)}, a_6^{(2)},\ a_9^{(2)},\ a_8^{(2)},\ a_7^{(2)},\ a_4^{(2)}$.

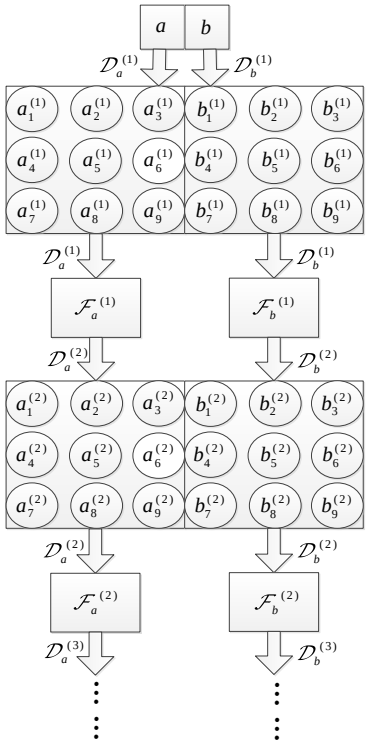

Fig. 2. Structure of SMSOM Background Model

$\mathcal{F}_i^{(l)}$ is the over-layer deep filter for layer l, which can help us to prepare the input data for layer $l+1$, for instance, $D_a^{(1)}$ is the input data sequence for pixel a, and $D_a^{(1)}$ is filtered by $\mathcal{F}_a^{(1)}$ to produce the input data sequence $D_a^{(2)}$ for layer 2 of pixel a, see subsection III-B for more details.

B. Training process

The training process is more sophisticated than the detection and maintenance processes, and we divide it into 4 parts: training data, competing process, cooperating process and over-layer deep filtering process, which are presented in detail below.

1) Training data: Assuming the total training data set for pixel i is a data sequence:

$$\mathcal{D}_i = \{I_i(1),\ I_i(2), \ldots,\ I_i(N)\}, \tag{1}$$

where $I_i(t) = (h_i(t),\ s_i(t),\ v_i(t))$ stands for the HSV values of the image $I(t)$ at pixel position i, t denotes the index of the training image sequence, $t = 1,\ 2, \ldots, N$, and the cardinality of $\mathcal{D}_i$ is N. Each SMSOM at position i, uses its corresponding pixel values $\mathcal{D}_i$ as the input data. In addition, for pixel position i, each layer l of the SMSOM has a different number of training data $D_i^{(l)}$, which is denoted as $n_i^{(l)}$, where $l = 1, 2, \ldots,\ L$, with L being a prescribed maximum layer; the over-layer deep filer (see subsection III-B-4) selects the data for the layer l from the layer $l-1$. For example, in **Figure 2**, the training data for pixel a is $\mathcal{D}_a = \{I_a(1),\ I_a(2), \ldots,\ I_a(N)\}$. After training layer 1 of pixel a, some data in $\mathcal{D}_a$ may not be represented well by layer 1, therefore, we introduce an over-layer deep filter here to filter out these data and put them into layer 2. More specifically, the input data sequence for layer 1, $\mathcal{D}_a^{(1)}$ (here, $\mathcal{D}_a^{(1)} = \mathcal{D}_a$), is filtered by $\mathcal{F}_a^{(1)}$ to produce the input data sequence for layer 2, $\mathcal{D}_a^{(2)}$. This process is done recursively until a prescribed maximum layer L is reached. Therefore, according to the notation shown previously, we have $\mathcal{D}_a = \mathcal{D}_a^{(1)} \supseteq \mathcal{D}_a^{(2)} \supseteq \cdots \supseteq \mathcal{D}_a^{(L)}$, $N = n_a^{(1)} \geq n_a^{(2)} \geq \cdots \geq n_a^{(L)}$.

2) Competing process: From previous descriptions, the input data sequence for layer l of pixel i is $\mathcal{D}_i^{(l)}$, and it is utilized to train layer l. The training process contains two phases: competing process and cooperating process. More specifically, every element $I_i^{(l)}(t)$ in $\mathcal{D}_i^{(l)}$ is used to accomplish the competing process and cooperating process one by one, which is called the *training process of (layer l, pixel i)*. The competing process is explained as follows.

For each pixel $I_i^{(l)}(t) \in \mathcal{D}_i^{(l)}$, where $t = 1,\ 2, \ldots,\ n_i^{(l)}$(the superscript l denotes that these data are from the input of layer l), we calculate the distance in the same way as [22], [23]:

$$\begin{aligned} &d_{iq}^{(l)}(t) = \\ &||\big(v_i^{(l)}(t)s_i^{(l)}(t)\cos h_i^{(l)}(t),\ v_i^{(l)}(t)s_i^{(l)}(t)\sin h_i^{(l)}(t),\ v_i^{(l)}(t)\big) \\ &-\big(\omega_{v_iq}^{(l)}(t)\omega_{s_iq}^{(l)}(t)\cos\omega_{h_iq}^{(l)}(t),\ \omega_{v_iq}^{(l)}(t)\omega_{s_iq}^{(l)}(t)\sin\omega_{h_iq}^{(l)}(t), \\ &\omega_{v_iq}^{(l)}(t)\big)||_2^2. \end{aligned} \tag{2}$$

The node which has the minimum value of $d_{iq}^{(l)}(t)$ ($q = 1,\ 2, \ldots,\ 9$) is called the *winner*, and the index of the *winner* will be used in the subsequent cooperating process, namely,

$$q_i^{(l)}(t)^* = \arg\min_q\{d_{iq}^{(l)}(t)\}. \tag{3}$$

In this paper, we call the node with the maximum of $d_{iq}^{(l)}(t)$ ($q = 1, 2, \ldots, 9$) the *loser*, which is a general concept and explained in detail in the following ***Remark 1***.

Remark 1. Given K points $\{\mathbf{x}_1,\ \mathbf{x}_2, \ldots,\ \mathbf{x}_K\}$, $\mathbf{x}_k \in \mathbb{R}^m$, which can be classified as the same class, and the distance set between a test point $\mathbf{x}_{test} \in \mathbb{R}^m$ and these K points is $\{\mathbf{d}_1,\ \mathbf{d}_2, \ldots,\ \mathbf{d}_K\}$, where $\mathbf{d}_k = d(\mathbf{x}_k,\ \mathbf{x}_{test})$ is a proper distance metric, such as Euclidean distance, or formula (2). We call the point $\mathbf{x}_{k^*}$ which has the maximum distance $\mathbf{d}_{k^*}$ the *loser*, namely, $k^* = \arg\max_k\{\mathbf{d}_k\}$, and this maximum distance $\mathbf{d}_{k^*}$ is called the *loser's threshold*. *Loser's threshold* describes the level of dissimilarity between the test point $\mathbf{x}_{test}$ and the K points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K\}$, and it plays a very important role in the newly proposed over-layer deep filter.

3) Cooperating process: The *winner* node shows that the current input pixel i can be thought of as a good approximation of the background, thus pixel i should be used to updated the weight vector of the *winner* node. Inspired by [22], [23], we also use the current pixel to update the *winner* node with index $q_i^{(l)}(t)^*$, together with its 8-*neighbors* (see subsection III-A). The update formula is as follows:

$$\omega_{p_iq}^{(l)}(t+1) = (1-\alpha_{train})\cdot\rho_q\cdot\omega_{p_iq}^{(l)}(t)+\alpha_{train}\cdot\rho_q\cdot p_i^{(l)}(t), \tag{4}$$

where $p_i^{(l)}(t) \in \{h_i^{(l)}(t),\ s_i^{(l)}(t),\ v_i^{(l)}(t)\}$, α_{train} is a parameter needed to be tuned manually, and it reveals the prior changing frequency of the training background images from frame to frame, subscript q is the index of neighbor computational

nodes or the *winner* node itself, $\rho_q = \exp(-\frac{||q-q^*||^2}{2\sigma^2})$ is a Gaussian kernel value from node q to the *winner* node q^*, where $||q-q^*||$ is the chessboard distance between q and q^* (if q is the *winner* node, the distance is 0), and σ^2 denotes the variance of the Gaussian kernel.

4) Over-layer deep filtering process: After *training process of (layer l, pixel i)*, the weights of layer l are obtained, namely, $\omega_{h_iq}^{(l)},\ \omega_{s_iq}^{(l)},\ \omega_{v_iq}^{(l)}$ ($q = 1,\ 2, \ldots,\ 9$). These weights serve as the model of SMSOM at layer l. And then, with the input data sequence $\mathcal{D}_i^{(l)}$ for layer l, we calculate the average distances $\mathbb{E}[d_{iq}^{(l)}]$ as:

$$\mathbb{E}[d_{iq}^{(l)}] = \frac{1}{n_i^{(l)}} \sum_{t=1}^{n_i^{(l)}} d_{iq}^{(l)}(t), \tag{5}$$

where $q = 1,\ 2, \ldots,\ 9$, and we assume every element of the input data sequence has the same probability in the background distribution. The largest value in this set $\{\mathbb{E}[d_{i1}^{(l)}], \mathbb{E}[d_{i2}^{(l)}], \ldots, \mathbb{E}[d_{i9}^{(l)}]\}$, namely, the *loser's threshold*, can be calculated, which is denoted by $\tau_i^{(l)}$. Loosely speaking, this *loser's threshold* $\tau_i^{(l)}$ will serve as the over-layer deep filter $\mathcal{F}_i^{(l)}$ of layer l for pixel i.

Once $\mathcal{F}_i^{(l)}$ is determined, we can prepare the input data for layer $l+1$. For each element $I_i^{(l)}(t) \in \mathcal{D}_i^{(l)}$, the *winner*'s distance $d_{iq^*}^{(l)}(t)$ will be calculated, and if $d_{iq^*}^{(l)}(t) \le \tau_i^{(l)}$, $I_i^{(l)}(t)$ will stay at layer l, otherwise, we call it is filtered out by layer l, and forms one element of the input data for layer $l+1$. Finally, the total input data of layer $l+1$ can be denoted by $\mathcal{D}_i^{(l+1)} = \{I_i^{(l+1)}(1),\ I_i^{(l+1)}(2), \ldots,\ I_i^{(l+1)}(n_i^{(l+1)})\}$. It should be noted that, through this over-layer filtering process, only the background data which are not matched very well at the lower layer will enter the next layer for further model learning and training. Hence, the input data for higher layers will be less and less, which makes SMSOM be trained in a very efficient manner. The filtering process from layer l to layer $l+1$ is summarized in **Algorithm 1**.

Algorithm 1 Over-layer filtering process of SMSOM

Input: $\mathcal{D}_i^{(l)},\ \tau_i^{(l)},\ \{\omega_{h_iq}^{(l)},\ \omega_{s_iq}^{(l)},\ \omega_{v_iq}^{(l)}\}$, where $q = 1,\ 2, \ldots,\ 9$
Output: $\mathcal{D}_i^{(l+1)}$

```
Initialize D_i^(l+1) = ∅;
for each I_i^(l)(t) ∈ D_i^(l), t = 1, 2, n_i^(l) do
    calculate the distance d_iq*^(l)(t) by formula (2), (3);
    if d_iq*^(l)(t) > τ_i^(l) then
        D_i^(l+1) = D_i^(l+1) ∪ {I_i^(l)(t)};
    else
        continue;
    end if
end for
```

C. Detection process

After training the SMSOM-BM, our task now is to utilize this model to classify $I_i = \{h_i,\ s_i,\ v_i\}$ as *foreground* or *background* (*foreground* and *background* will be used to stand for two labels). The process is, firstly, we calculate the *winner*'s distance $d_{iq^*}^{(1)}$ of the SMSOM model at layer 1, if $d_{iq^*}^{(1)} \le \tau_i^{(1)}$, then it is classified as *background* and the detection process of I_i is over. On the other hand, if $d_{iq^*}^{(1)} > \tau_i^{(1)}$, then I_i will be put into layer 2 to find whether it is *background*. The same process is done recursively until a prescribed maximum layer L is reached. If no match is found in the SMSOM model of pixel i, we classify I_i as *foreground*.

The detection process is briefly summarized in **Algorithm 2**. In **Algorithm 2**, we define *DETECT*(I_i, *layer l of pixel i*) as the process explained above, namely, for input I_i, calculate the *winner*'s distance at *layer l of pixel i*, and if $d_{iq^*}^{(l)} \le \tau_i^{(l)}$, we classify it as *background*, and return **true**, otherwise, put it into layer $l+1$, and return **false**.

Algorithm 2 Pixel classification by SMSOM

Input: test data I_i, SMSOM model $\{\omega_{h_iq}^{(l)},\ \omega_{s_iq}^{(l)},\ \omega_{v_iq}^{(l)}\}$, where $q = 1,\ 2, \ldots,\ 9, l = 1,\ 2, \ldots,\ L$, *loser's threshold* $\tau_i^{(l)}$, $l = 1,\ 2, \ldots,\ L$
Output: I_i's label L_i: *foreground* or *background*

```
allocate a temporary boolean variable tempLabel;
for each l ∈ {1, 2, ..., L} do
    tempLabel = DETECT(I_i, layer l of pixel i);
    if tempLabel == true then
        L_i = background;
        break;
    else if l == L then
        L_i = foreground;
    else
        continue;
    end if
end for
```

D. Maintenance process

If I_i is classified as *background* at layer l, we should use I_i to update the SMSOM model at layer l. This process is similar to the cooperating process in III-B, but with a smaller learning rate α_{test}, which reflects a prior that background images do not change too fast during foreground detection. The update formula is shown as follows:

$$\omega_{p_iq}^{(l)}(t+1) = (1-\alpha_{test}) \cdot \rho_q \cdot \omega_{p_iq}^{(l)}(t) + \alpha_{test} \cdot \rho_q \cdot p_i, \tag{6}$$

where $p_i \in I_i = \{h_i,\ s_i,\ v_i\}$. Readers should notice that other layers except layer l will not be updated, and the SMSOM model of different pixels may be updated at different layers. If I_i is classified as *foreground*, no update is done for the SMSOM model of pixel i.

The whole process of our foreground detection method is summarized in **Algorithm 3**.

IV. Implementation details and Experiments

From a theoretical point of view, more layers of SOM lead to better background modeling accuracy and thus better foreground detection performance; meanwhile, more layers

Algorithm 3 Foreground detection by SMSOM-BM

1: Use images $\{I_{train}(1),\ I_{train}(2), \ldots,\ I_{train}(N)\}$ to train SMSOM-BM;
2: **for** every testing image $I_{test}(k)$, where $k \in \{1,\ 2,\ \ldots\}$ **do**
3: Use **Algorithm 2** to classify the label (*foreground, background*) of every pixel of $I_{test}(k)$;
4: Maintain SMSOM-BM according to section III-D;
5: **end for**

make the algorithm more time-consuming. In our implementation[1] and experiments, by finding a trade-off between real-time running speed and foreground detection performance, we adopt a 3-layer SMSOM model, namely, $L = 3$. The total parameters of SMSOM-BM contain: 1) the weights $\omega_{iq}^{(l)}$ $(q = 1,\ 2, \ldots,\ 9,\ l = 1,\ 2,\ 3)$, which are learned and updated automatically using the proposed approach, i is the pixel position; 2) the learning rates α_{train} and α_{test}, which are the only parameters needed to be tuned manually. In this paper, we adopt $\alpha_{train} = 1$, $\alpha_{test} = 0.03$ for all experiments.

The weights are initialized as follows:

$$\begin{cases} \omega_{h_iq}^{(l)} = h_i(1) \\ \omega_{s_iq}^{(l)} = s_i(1) \\ \omega_{v_iq}^{(l)} = v_i(1) \end{cases}, \tag{7}$$

for $q = 1,\ 2, \ldots,\ 9,\ l = 1,\ 2,\ 3,$ and $h_i(1),\ s_i(1),\ v_i(1)$ are the HSV values at pixel i of the first frame.

To demonstrate the effectiveness of the proposed method, we conduct several experiments using a new dataset CDnet2012, which can be downloaded from [45]. CDnet2012 contains several very challenging scenarios for motion detection. These scenarios can be classified into 6 categories, namely, *baseline*, *dynamic background*, *camera jitter*, *intermittent object motion*, *shadow* and *thermal* (*intermittent object motion* and *shadow* are not used), while each category contains several videos of different scenarios. To comparatively show the performance of the proposed approach, we report the results of three state of the art algorithms, namely, GMM [2], KDE [4], SOBS [22] (the results of these methods are also downloaded from [45][2]). SOBS can be seen as an one-layer SMSOM-BM with manually tuning threshold parameters. To achieve real-time performance, we have implemented the proposed method using NVIDIA CUDA platform. Every pixel's SMSOM model is handled by a thread, and all threads run parallelly. **Table I** summarizes the runtime performance on a standard PC with 2.67GHz Intel CPU, 6GB RAM, and a NVIDIA GeForce GTX 260 GPU, which presents that good real-time performance can be achieved for most scenarios. However, since the currently used GPU is a low level old product, much faster performance can be achieved using a moderate or high level GPU such as GeForce GTX TITAN (the number of GTX TITAN's Stream Processor is more than 10 times than that of GTX 260[3]).

[1]The code can be found at: http://zhaozj89.github.io/SMSOM/

[2]The results in [45] are post-processed by a 5×5 median filter; therefore, for a fair comparison, the results of our method are also post-processed by a 5×5 median filter

[3]http://www.geforce.com/hardware/desktop-gpus

TABLE I
RUNTIME PERFORMANCE

	image size	frames per second (fps)
baseline		
highway	320×240	31.7
office	360×240	26.6
pedestrians	360×240	27.2
PETS2006	720×576	7.0
dynamic background		
boats	320×240	29.9
canoe	320×240	29.5
fall	720×480	7.2
fountain01	432×288	20.1
fountain02	432×288	21.1
overpass	320×240	32.1
camera jitter		
badminton	720×480	7.7
boulevard	352×240	28.7
sidewalk	352×240	26.6
traffic	320×240	34.0
thermal		
corridor	320×240	36.0
diningRoom	320×240	36.6
lakeSide	320×240	35.4
library	320×240	36.1
park	352×288	23.1

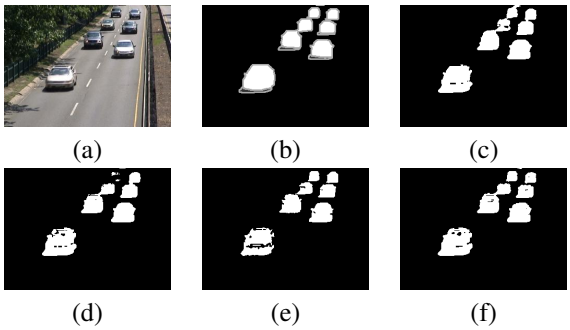

Fig. 3. Results on *highway*: (a) original image, (b) ground truth, (c) output of SMSOM-BM, (d) output of GMM, (e) output of SOBS, (f) output of KDE

A. Qualitative evaluation

Because of the limited space, we only present some typical results of the proposed method from different categories. In this subsection, we will present some qualitative results in a comparative way, while the quantitative evaluation will be provided in subsection IV-B.

1) highway: highway is a video from the category *baseline*, which is a typical scenario in visual surveillance applications. As shown in **Figure 3(a)**, cars running on the highway should be detected as foreground, and there are some small motion parts in the background, namely, the waving trees, and this is a mild challenge for motion detection algorithms. From **Figure 3**, we see that the proposed method successfully detects most of moving cars as foreground perfectly except some very small black holes in the foreground regions. The results of GMM, KDE and SOBS are also shown in **Figure 3**, which also present very good detected foreground regions. *Baseline* is a relatively easy category, so all of the evaluated algorithms can achieve acceptable results. Readers can see the results below for some more challenging scenarios.

2) fountain: fountain is a scenario from the *dynamic background* category, which is a very challenging video. As shown

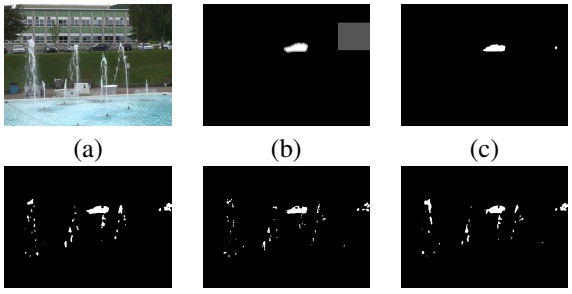

Fig. 4. Results on *fountain*: (a) original image, (b) ground truth, (c) output of SMSOM-BM, (d) output of GMM, (e) output of SOBS, (f) output of KDE

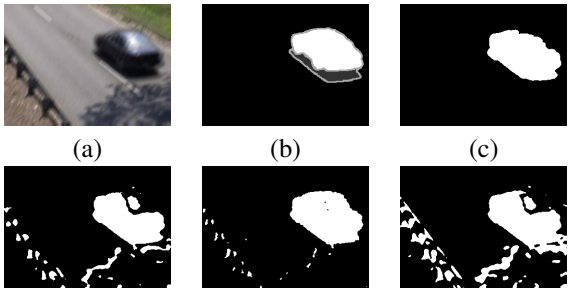

Fig. 5. Results on *traffic*: (a) original image, (b) ground truth, (c) output of SMSOM-BM, (d) output of GMM, (e) output of SOBS, (f) output of KDE

in **Figure 4(a)**, the fountain in the middle of the image moves regularly, and it should be detected as background during foreground detection phase, while moving cars on the road which are not contained in the training images should be detected as foreground. After the pre-training phase, the proposed method has the ability to 'learn' the particular motion pattern of the fountain, therefore, it will not be classified as foreground during the foreground detection phase. On the other hand, when the true foreground region comes (cars), this region will be detected as foreground since the proposed SMSOM-BM do not 'learn' such motion pattern during the training phase. **Figure 4** presents the detection results of different algorithms. The gray region in the ground truth image (**Figure 4(b)**) is not the region of interest (Non-ROI), and it could be omitted when evaluating different algorithms. In **Figure 4** we can see that the proposed method can detect the moving car without any fountain regions, which shows that SMSOM-BM has a very strong learning and representative ability to describe dynamic background.

3) traffic: traffic is also a very challenging video which belongs to *camera jitter* category. Because of the unstable camera, the whole background vibrates strongly, thus the background is prone to be wrongly detected as foreground. Compared to *founatain*, *traffic* is more challenging because the whole background moves irregularly due to the camera jittering. Even in such a challenging scenario, we find that SMSOM-BM is also able to learn this strongly irregular motion pattern in the background quite well, which shows the superior representative ability of SMSOM-BM. **Figure 5** presents the comparative results using different algorithms. The proposed method can detect the moving car perfectly without being affected by the vibrating background.

4) lakeSide: In some particular visual surveillance applications, infrared cameras (IR-cameras) are usually utilized to capture thermal images. Motion detection on thermal images is a challenging task because these videos contain some typical thermal artifacts, such as heat stamps, heat reflections and camouflage effects [46]. *lakeSide* is such a typical scenario, which belongs to the *thermal* category and contains the challenges mentioned above. In this case, it is very likely for a motion detection algorithm to omit some foreground parts because the appearance of the foreground is quite similar with that of the background. In **Figure 6(a)**, human beings should be detected as foreground during foreground detection,

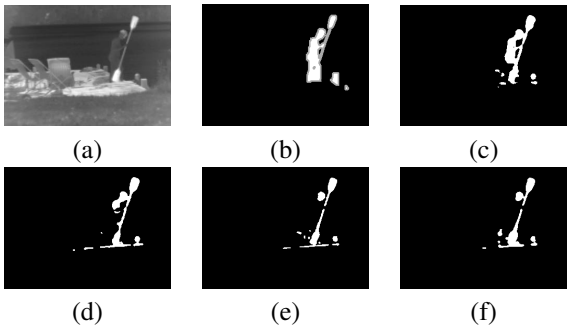

Fig. 6. Results on *lakeSide*: (a) original image, (b) ground truth, (c) output of SMSOM-BM, (d) output of GMM, (e) output of SOBS, (f) output of KDE

although their appearance is quite similar with the background. In **Figure 6**, it is shown that the proposed SMSOM-BM can detect most of the foreground regions except some small black parts, and it outperforms GMM, SOBS and KDE in terms of foreground completeness and accuracy.

B. Quantitative evaluation

The ground truth of CDnet2012 is provided in the form of pixel labels, which can be utilized to evaluate a motion detection algorithm at pixel level. With these pixel labels, *true positive* (TP), *true negative* (TN), *false positive* (FP) and *false negative* (FN) can be calculated, and then we use some carefully chosen metrics to evaluate the proposed method more completely. More specifically, 3 metrics which are commonly utilized to evaluate a new proposed algorithm are adopted in this paper, namely, Recall (Re), Precision (Pr) and F-measure.

Recall represents the percentage of detected foreground region compared with the whole foreground region in the ground truth, which is obtained as follows:

$$\text{Recall (Re)}=\frac{\text{TP}}{\text{TP}+\text{FN}}. \tag{8}$$

A big Recall value is expected, which shows that a motion detection algorithm can detect most of the foreground region. On the contrary, a small Recall value shows that parts of the foreground region are classified as background wrongly.

Precision denotes the percentage of the detected true foreground region in the ground truth compared with the total detected foreground region, which is derived as follows:

$$\text{Precision (Pr)}=\frac{\text{TP}}{\text{TP}+\text{FP}}. \tag{9}$$

More specifically, although a motion detection algorithm may detect all the foreground regions correctly in the ground truth, Precision value will still be small if some moving background regions are detected as foreground wrongly.

F-measure is the harmonic mean of Recall and Precision, namely,

$$\text{F-measure}= (\frac{1/\text{Pr}+1/\text{Re}}{2})^{-1}, \tag{10}$$

and a big F-measure presents a good balance between Recall and Precision. Therefore, F-measure shows the overall performance of a motion detection algorithm, and it is the most important metric.

Recall, Precision and F-measure together can give a relatively objective quantitative evaluation of a motion detection method. To be more convincing, we use all the videos for each category to evaluate the proposed method (termed as **SMSOM-BM** in the Tables II-VII). Experiment results are shown in the rest of this subsection. It should be noticed that, from the descriptions in section III, we know that SMSOM-BM needs foreground-free training images to learn the threshold τ. Therefore, for some videos that do not contain training images, we set $\tau = 0.06$ for foreground detection. However, due to the lack of training process, these coarsely tuned parameters cannot provide a good initial estimate of SMSOM-BM, which leads to performance decreasing for these cases. For a relatively fair comparison, we also report the metric values without considering these scenarios in the subsequent analysis (termed as **SMSOM-BM with training** in the Tables II-VII).

1) baseline: baseline contains 4 videos of different scenarios. We present the Recall, Precision and F-measure values of this category in **Table II**.

The SMSOM-BM results show the means of these three metrics for all the contained scenarios by the proposed method. The SMSOM-BM with training results give means of these 3 metrics only for those scenarios with training images (in this category, the scenario *PETS2006* and *highway* do not contain training images). These metrics are utilized to compare with other motion detection algorithms, as **Table III** shows.

As mentioned previously, *baseline* is a relatively easy category, therefore, all presented methods can achieve high Precision, Recall and F-measure values and the trained SMSOM-BM model achieves superior overall performance.

2) dynamic background: dynamic background category contains 6 different scenarios, namely, *boats*, *canoe*, *fall*, *fountain01*, *fountain02*, *overpass*, and all videos contain foreground-free training images. The results of the 3 metrics are listed in **Table IV**.

The comparison results of the average Precision, Recall and F-measure values are listed in **Table V**. This category seems more challenging than *baseline* because the 3 metric values of all the methods are relatively lower than these in *baseline*. Although the Recall values of KDE, GMM and SOBS are higher than SMSOM-BM, they have much lower Precision values than SMSOM-BM, which means that these methods seem not to deal with dynamic background well, and most of the moving background regions are detected as foreground wrongly. Moreover, the proposed method can also achieve the highest F-measure value in this category.

TABLE II
BASELINE RESULTS OF THE PROPOSED METHOD

	Pr	Re	F-measure
highway	0.9324	0.9612	0.9466
office	**0.9812**	0.9407	**0.9605**
pedestrians	0.9216	0.9349	0.9282
PETS2006	0.8083	**0.9959**	0.8717
SMSOM-BM	0.9109	0.9457	0.9268
SMSOM-BM with training	0.9514	0.9378	0.9444

TABLE III
PRECISION, RECALL, F-MEASURE COMPARISON RESULTS OF BASELINE

	Pr	Re	F-measure
SOBS	0.9313	0.9193	0.9251
KDE	0.8969	0.9223	0.9092
GMM	0.8180	0.8461	0.8245
SMSOM-BM	0.9109	**0.9458**	0.9268
SMSOM-BM with training	**0.9514**	0.9378	**0.9444**

3) camera jitter: camera jitter contains 4 videos, *badminton*, *boulevard*, *sidewalk*, *traffic*. **Table VI** shows metric values for these scenarios by using the proposed method. *Badminton* and *sidewalk* do not contain foreground-free training images, so we list another row of metric values that these scenarios are not considered. **Table VII** presents the comparison result of different methods. The 3 metric values of GMM and KDE are relatively lower than those for the *dynamic background* category, which shows that *camera jitter* seems to be more challenging than *dynamic background*. In such challenging case, SMSOM-BM achieves the highest Precision and F-measure values if training images are available.

4) thermal: thermal contains 5 scenarios, namely, *corridor*, *diningRoom*, *lakeSide*, *library* and *park*, and all videos contain foreground-free training images. We list the 3 metric values for all scenarios using SMSOM-BM in **Table VIII**. The comparison results are shown in **Table IX**, from which we can see that SMSOM-BM presents very attractive performance, and SMSOM-BM can achieve the highest Recall and F-measure values.

TABLE IV
DYNAMIC BACKGROUND RESULTS OF THE PROPOSED METHOD

	Pr	Re	F-measure
boats	0.9750	0.4446	0.6107
canoe	0.9573	0.4736	0.6337
fall	0.9373	0.6435	0.7631
foutain01	0.7342	0.5220	0.6102
fountain02	0.9640	**0.7320**	**0.8321**
overpass	**0.9758**	0.4308	0.5977
SMSOM-BM	0.9239	0.5410	0.6754

TABLE V
F-MEASURE COMPARISON RESULTS OF DYNAMIC BACKGROUND

	Pr	Re	F-measure
SOBS	0.5856	**0.8798**	0.6439
KDE	0.5732	0.8012	0.5961
GMM	0.5989	0.8344	0.6330
SMSOM-BM	**0.9239**	0.5410	**0.6754**

TABLE VI
CAMERA JITTER OF THE PROPOSED METHOD

	Pr	Re	F-measure
badminton	0.5688	**0.9866**	0.7216
boulevard	0.6602	0.7209	0.6892
sidewalk	0.2272	0.7234	0.3458
traffic	**0.7558**	0.8005	**0.775**
SMSOM-BM	0.5530	0.8079	0.6335
SMSOM-BM with training	0.7080	0.7602	0.7321

TABLE VII
F-MEASURE COMPARISON RESULTS OF CAMERA JITTER

	Pr	Re	F-measure
SOBS	0.6399	0.8007	0.7086
KDE	0.4862	0.7375	0.5720
GMM	0.5126	0.7334	0.5969
SMSOM-BM	0.5530	**0.8079**	0.6335
SMSOM-BM with training	**0.7080**	0.7602	**0.7321**

TABLE VIII
THERMAL OF THE PROPOSED METHOD

	Pr	Re	F-measure
corridor	0.7502	0.8847	0.8119
diningRoom	0.9055	0.8477	0.8757
lakeSide	0.8719	0.5162	0.6484
library	**0.9371**	**0.9555**	**0.9462**
park	0.9098	0.4383	0.5916
SMSOM-BM	0.8717	0.7719	0.7931

TABLE IX
F-MEASURE COMPARISON RESULTS OF THERMAL

	Pr	Re	F-measure
SOBS	0.8754	0.5888	0.6834
KDE	**0.8974**	0.6725	0.7423
GMM	0.8652	0.5691	0.6621
SMSOM-BM	0.8717	**0.7719**	**0.7931**

V. CONCLUSION

A *Stacked Multi-layer Self-Organizing Map* (SMSOM) is proposed in this paper, and we also introduce a novel over-layer deep filter which can be utilized to train this model very efficiently. By modeling every pixel as a SMSOM, we construct a *Stacked Multi-layer Self-Organizing Map Background Model* (SMSOM-BM) which is further adopted to fulfill the foreground detection task. We also conduct several comparative experiments to show the superior performance of the method in both qualitative and quantitative ways. What' more, a CUDA implementation makes this method more practical in real applications.

REFERENCES

[1] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: real-time tracking of the human body," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 780–785, 1997.

[2] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Pro. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, 1999, pp. 246–252.

[3] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction," in *Proc. of the 17th Int. Conf. on Pattern Recognition*, 2004, pp. 28–31.

[4] A. Elgammal, D. Harwood, and L. Davis, "Nonparametric model for background subtraction," in *European Conf. Comp. Vision (ECCV)*, 2000, pp. 751–767.

[5] L. Li, W. Huang, I.-H. Gu, and Q. Tian, "Statistical modeling of complex backgrounds for foreground object detection," *IEEE Trans. Image Process.*, vol. 13, no. 11, pp. 1459–1472, 2004.

[6] A. Lanza and L. Di Stefano, "Statistical change detection by the pool adjacent violators algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 9, pp. 1894–1910, 2011.

[7] T. Bouwmans, F. El Baf, and B. Vachon, "Statistical background modeling for foreground detection: A survey," *Handbook of Pattern Recognition and Computer Vision*, vol. 1, no. 3, pp. 219–237, Nov. 2008.

[8] F. Baf, T. Bouwmans, and B. Vachon, "Type-2 fuzzy mixture of gaussians model: Application to background modeling," in *Advances in Visual Computing*, 2008, pp. 772–781.

[9] Z. Zhao, T. Bouwmans, X. Zhang, and Y. Fang, "A fuzzy background modeling approach for motion detection in dynamic backgrounds," in *Int. Conf. on Multimedia and Signal Processing*, 2012, pp. 177–185.

[10] A. Shimada, H. Nagahara, and R.-I. Taniguchi, "Background modeling based on bidirectional analysis," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2013, pp. 1979–1986.

[11] T. Bouwmans, "Subspace learning for background modeling: A survey," *Recent Patents on Computer Science*, vol. 4, no. 3, pp. 223–234, Nov. 2009.

[12] N. Oliver, B. Rosario, and A. Pentland, "A bayesian computer vision system for modeling human interactions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 831–843, 2000.

[13] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. ACM*, vol. 58, no. 3, pp. 11:1–11:37, Jun. 2011.

[14] M. Yamazaki, G. Xu, and Y.-W. Chen, "Detection of moving objects by independent component analysis," in *Asian Conference on Computer Vision*, 2006.

[15] X. Ding, L. He, and L. Carin, "Bayesian robust principal component analysis," *IEEE Trans. Image Process.*, vol. 20, no. 12, pp. 3419–3430, 2011.

[16] S. Babacan, M. Luessi, R. Molina, and A. Katsaggelos, "Sparse bayesian methods for low-rank matrix estimation," *IEEE Trans. on Signal Process.*, vol. 60, no. 8, pp. 3964–3977, 2012.

[17] T. Bouwmans and E. Zahzah, "Robust PCA via principal component pursuit: A review for a comparative evaluation in video surveillance," *Special Isssue on Background Models Challenge, Computer Vision and Image Understanding (CVIU),* 2014.

[18] V. Cevher, A. Sankaranarayanan, M. F. Duarte, D. Reddy, and R. G. Baraniuk, "Compressive sensing for background subtraction," in *European Conf. Comp. Vision (ECCV)*, 2008, pp. 155–168.

[19] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground background segmentation using codebook model," *Real-Time Imaging*, vol. 11, no. 3, pp. 172 – 185, 2005.

[20] O. Barnich and M. Van Droogenbroeck, "Vibe: A universal background subtraction algorithm for video sequences," *IEEE Trans. Image Process.*, vol. 20, no. 6, pp. 1709–1724, 2011.

[21] D. Culibrk, O. Marques, D. Socek, H. Kalva, and B. Furht, "Neural network approach to background modeling for video object segmentation," *IEEE Trans. Neural Netw.*, vol. 18, no. 6, pp. 1614–1627, 2007.

[22] L. Maddalena and A. Petrosino, "A self-organizing approach to background subtraction for visual surveillance applications," *IEEE Trans. Image Process.*, vol. 17, no. 7, pp. 1168–1177, Jul. 2008.

[23] L. Maddalena and A. Petrosino, "The SOBS algorithm: What are the limits?" in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 21–26.

[24] M. Hofmann, P. Tiefenbacher, and G. Rigoll, "Background segmentation with feedback: The pixel-based adaptive segmenter," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 38–43.

[25] M. Van Droogenbroeck and O. Paquot, "Background subtraction: Experiments and improvements for vibe," in *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 32–37.

[26] R. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: a systematic survey," *IEEE Trans. Image Process.*, vol. 14, no. 3, pp. 294–307, 2005.

[27] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: principles and practice of background maintenance," in *Proc. IEEE Int. Conf. Computer Vision*, 1999, pp. 255–261.

[28] H. Woo, Y. M. Jung, J.-G. Kim, and J. K. Seo, "Environmentally robust motion detection for video surveillance," *IEEE Trans. Image Process*, vol. 19, no. 11, pp. 2838–2848, 2010.

[29] R. Caseiro, J. Henriques, P. Martins, and J. Batista, "A nonparametric riemannian framework on tensor field with application to foreground

segmentation," in *Proc. IEEE Int. Conf. Computer Vision*, 2011, pp. 1–8.
[30] A. Schick, M. Bauml, and R. Stiefelhagen, "Improving foreground segmentations with probabilistic superpixel markov random fields," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 27–31.
[31] S. Brutzer, B. Höferlin, and G. Heidemann, "Evaluation of background subtraction techniques for video surveillance," in *Pro. IEEE Conf. Computer Vision and Pattern Recognition*, 2011, pp. 1937–1944.
[32] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.
[33] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview," *Computer Science Review*, vol. 11-12, pp. 31–66, May. 2014.
[34] R. M. Luque, E. Domínguez, E. J. Palomo, and J. Muñoz, "An ART-type network approach for video object detection." in *European Symposium on Artificial Neural Networks-Computational Intelligence and Machine Learning (ESANN)*, 2010.
[35] N. Rai, S. Chourasia, and A. Sethi, "An efficient neural network based background subtraction method," in *Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012)*, ser. Advances in Intelligent Systems and Computing, 2013, vol. 201, pp. 453–460.
[36] M. Chacon, G. Sergio, and V. Javier, "Simplified som-neural model for video segmentation of moving objects," in *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*, June 2009, pp. 474–480.
[37] M. Chacon-Murguia and S. Gonzalez-Duarte, "An adaptive neural-fuzzy approach for object detection in dynamic backgrounds for surveillance systems," *IEEE Trans. Ind. Electron.*, vol. 59, no. 8, pp. 3286–3298, Aug 2012.
[38] J. Ramirez-Quintana and M. Chacon-Murguia, "Self-organizing retinotopic maps applied to background modeling for dynamic object segmentation in video sequences," in *Neural Networks (IJCNN), The 2013 International Joint Conference on*, Aug 2013, pp. 1–8.
[39] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.
[40] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, Jan. 2009.
[41] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, Dec. 2010.
[42] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, pp. 504–507, Jul. 2006.
[43] A. Rauber, D. Merkl, and M. Dittenbach, "The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data," *IEEE Trans. Neural Netw.*, vol. 13, no. 6, pp. 1331–1341, Nov. 2002.
[44] E. Palomo, E. Domínguez, R. Luque, and J. Muñoz, "Image hierarchical segmentation based on a ghsom," in *Neural Information Processing*, ser. Lecture Notes in Computer Science, C. Leung, M. Lee, and J. Chan, Eds. Springer Berlin Heidelberg, 2009, vol. 5863, pp. 743–750.
[45] http://www.changedetection.net/.
[46] N. Goyette, P. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection.net: A new change detection benchmark dataset," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2012, pp. 1–8.

Zhenjie Zhao received the bachelor degree in Automation from Nankai University (with honor). He is now a master candidate at Institute of Robotics and Automatic Information System of Nankai University. His research interests are image processing, computer vision, and machine learning.

Xuebo Zhang Biography text here.

Yongchun Fang Biography text here.