IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 24, 2020, accepted August 27, 2020, date of publication August 31, 2020, date of current version September 14, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3020619

Video Snow Removal Based on Self-Adaptation Snow Detection and Patch-Based Gaussian Mixture Model

BIN YANG[1,2], ZHENHONG JIA[1,2], JIE YANG[3], (Member, IEEE), AND NIKOLA K. KASABOV[4], (Fellow, IEEE)

[1]College of Information Science and Engineering, Xinjiang University, Urumqi 830046, China
[2]Key Laboratory of Signal Detection and Processing, Xinjiang University, Urumqi 830046, China
[3]Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, Shanghai 200400, China
[4]Knowledge Engineering and Discovery Research Institute, Auckland University of Technology, Auckland 1020, New Zealand

Corresponding author: Zhenhong Jia (jzhh9009@sohu.com)

This work was supported in part by the National Science Foundation of China under Grant U1803261, and in part by the International Science and Technology Cooperation Project of the Ministry of Education of the People's Republic of China under Grant DICE 2016–2196.

ABSTRACT Video desnowing has become a challenging research topic in computer vision in recent years. Existing methods cannot remove most of the snow in heavy snow scenes and will cause the deformation of moving objects when used for snowy videos that include moving objects. These methods have poor generalizability, exhibiting poor performance when removing snow from videos with different resolutions. In this paper, we propose a new video snow removal method based on self-adaptation snow detection and a patch-based Gaussian mixture model (VSRSG). First, an optical flow estimation method and a support vector machine (SVM) are used to detect snowflakes, and a self-adaptation threshold process is used to remove dense snowflakes in the snowflake detection map to obtain a sparse snowflake detection map. Then, a patch-based Gaussian mixture model (PBGMM), which can remove moving objects and both sparse and dense snowflakes from videos and restore a clear video background, is applied for background modeling. A Markov random field (MRF) and self-adaptation threshold processing are used to extract sparse snowflakes and moving objects and combine them with the background to form an input video without dense snowflakes. Finally, a similar block matching method is employed to fill in the detected snowflake pixels with the information from adjacent frames to remove the sparse snowflakes in the near range. This method can also remove snowflakes in front of moving objects. Experiments show that the proposed method can simultaneously remove sparse snowflakes, dense snowflakes and snowflakes in front of moving objects and outperforms other state-of-the-art methods.

INDEX TERMS Video desnowing, self-adaptation snowflake detection, patch-based Gaussian mixture model, low-rank background modeling, moving foreground detection.

I. INTRODUCTION

With the development of computer vision technology, outdoor vision systems have been increasingly and widely used in military, traffic and safety applications. Video-based computer vision technology applications, such as target tracking, target recognition, and behavior analysis, often require moving target detection as a first step, so research on moving target detection in videos has become very important. However, the development of moving object detection also faces main challenges, such as adverse weather conditions. Adverse weather will seriously reduce the accuracy of moving object detection [43], [44]. For example, gale weather will lead to camera shaking and dynamic backgrounds (such as shaking leaves and waving water surfaces), fog and dust will reduce the overall visibility of the video, and snowflakes and raindrops will be misjudged by the moving object detection algorithm as moving objects. Therefore, it is necessary to build an outdoor vision system that eliminates the impact of various adverse weather conditions on video moving object detection.

In research on eliminating the influence of adverse weather on video, most attention has been focused on rain removal from videos [45]–[49], and few papers have discussed snow

The associate editor coordinating the review of this manuscript and approving it for publication was Qiangqiang Yuan.

removal. As a common product of adverse weather, snow and rain have certain similarities in that they both have high intensity and fall at a rapid rate. Many researchers have used video rain removal algorithms for video desnowing [9], [11]. However, video rain removal methods are not directly suitable for video desnowing. For example, raindrops have the same size, and the falling direction and speed of all raindrops are generally the same [1]. Compared with rain, snow falls in different directions and the shapes are not uniform; these characteristics make snow removal more challenging than rain removal. Therefore, removing snowflakes from video images is also of great research significance.

In previous research, some methods have relied on the size and intensity characteristics of snow for snow detection via the frame difference method [6], [8], [10]. The snow pixels detected in the current frame are replaced with the median or mean of the pixels in the adjacent frame to achieve snow removal. These methods can be suitable for videos with light snow but often fail on videos with heavy snow. In videos with heavy snow, the same pixel in several adjacent frames might be covered by snow at all times, which can lead to erroneous detection. Because this method considers only the characteristics of snow and does not consider the characteristics of moving objects, it does not work on snow video that includes moving objects because the positional changes of the moving objects in the adjacent frames can be mistaken for snow. To remove snow from videos that contain moving objects, some researchers applied prior knowledge of the video background and the moving objects [11], [12]. These methods modeled the free-snow background using a low-rank matrix and detected the moving objects by applying the continuity of the moving object in the spatiotemporal domain. These methods consider the characteristics of moving objects but not those of the snow, which may explain why these methods do not perform well for heavy snow. When working with a heavy snow scene, these methods caused deformations in moving objects and the misjudgment of sparse snowflakes in the near range as a moving object, resulting in incomplete snow removal. At the same time, the above methods do not consider videos with different resolutions, as it is often difficult to obtain a better snow removal effect for these videos.

In general, the above methods have a limited scope of application. As existing methods set only uniform pixel sizes or weights for snowflake detection and removal, the above algorithms are not effective for snow removal when dealing with videos with heavy snow. When the existing algorithms remove snow from videos containing moving objects, some snowflakes may be misjudged as moving objects, which will result in their incomplete removal and the deformation of moving objects. Moreover, the generalizability of the algorithm worsens, and it is difficult to achieve a good snow removal effect when removing snow from videos with high or low resolution. Therefore, it is necessary to find snowflake removal methods suitable for videos with heavy snow scenes, moving objects and different resolutions.

To address the above problems, this paper proposes a new video snow removal method based on self-adaptation snow detection and a patch-based Gaussian mixture model, called VSRSG. In this paper, snowflakes are divided into sparse snowflakes with large sizes in the near range and dense snowflakes with small sizes in the far range [11]. The input video is decomposed into a low-rank background, dense snowflakes in the far range and a moving foreground, which includes moving objects and sparse snowflakes in the near range. First, an optical flow estimation method and a support vector machine (SVM) are employed for preliminary snowflake detection to obtain a snowflake detection map. To enable our method to deal with different resolution videos, we conduct dilation operations on the snowflake detection map and then use a self-adaptation threshold to remove dense snowflakes with smaller sizes from the snowflake detection map and keep only sparse snowflakes. This detection process can also detect snowflakes in front of moving objects. Then, a patch-based Gaussian mixture model (PBGMM) is applied for background modeling. This model will restore a clear video background without dense snowflakes, sparse snowflakes and moving objects. A Markov random field (MRF) is used for moving foreground detection. To eliminate the influence of dense snowflakes on detection, self-adaptation threshold processing is added to remove the detected dense snowflakes. We paste the detected moving objects and sparse snowflakes back into the clear background. At this time, the new video includes only a low-rank background, moving objects and sparse snowflakes. Finally, the target frame in the new input video is decomposed into disjointed blocks. A similar block matching algorithm is used to find similar blocks in adjacent frames, and then the snowflake pixels in the sparse snowflake detection map of the target frame are filled with the information of these blocks to remove sparse snowflakes. Experimental results show that our method can reconstruct clear snow-free backgrounds and effectively remove snowflakes from videos. The method proposed in this paper is different from previous methods in the following aspects.

1) In this paper, snowy videos with different resolutions are considered. Setting a uniform size for snowflake detection and removal will lead to poor performance. To achieve excellent performance on snow videos with different resolutions, we use self-adaptation threshold processing in both the snowflake and foreground detection processes to eliminate the influence of dense snowflakes and dynamic backgrounds (such as leaf shaking) for sparse snowflake detection and moving foreground detection.

2) Moving objects and sparse snowflakes are both defined as moving foregrounds. The existing algorithms separate snowflakes from the moving foreground after moving foreground detection, but misjudgment always occurs when the snowflakes are separated from the moving objects in the moving foreground, resulting in the incomplete removal of snowflakes. Therefore, the proposed method does not separate snowflakes from the moving foreground, so we paste

them back onto the generated low-rank background. At this point, the new video contains only the clear background and the moving foreground. Then, the sparse snowflakes in the moving foreground are removed according to the detected sparse snowflake map and the new video. At the same time, the snowflakes in front of the moving object are removed.

The paper is organized as follows. Section II introduces related work. Section III describes the proposed method. Section IV discusses the experimental results, and Section V concludes the paper.

II. RELATED WORK

Some researchers have conducted single-image snow removal research based on the shape and brightness characteristics of snowflakes. Sun *et al.* [13] proposed a snow removal method based on fuzzy connectivity that solves the fuzzy problem of various shapes and rapidly falling snowflakes by selecting multiple seed points for fuzzy growth and uses the H component of the hue-saturation-intensity (HIS) color space to remove snow. Pei *et al.* [16] proposed a single-image snow removal method that removes snowflakes based on the characteristics of snow saturation and visibility. Some researchers employed guided filters to remove snowflakes from images. Xu *et al.* [14] proposed the use of a guide filter to remove snowflakes in an image. This algorithm uses an original image as both the input image and the guide map of the guide filter so that the guide filter can retain the edge structure of the object and remove snowflakes in the image. Rajderkar and Mohod [15] decomposed an image into a low-frequency component and high-frequency component using a bilateral filter. Using dictionary learning and sparse coding, the high-frequency component is decomposed into a snow component and a non-snow component, and the geometric component of the high-frequency component is fused with the low-frequency component of the image to obtain an image without snow. Ding *et al.* [17] designed a guided smoothing filter to remove snow. Wang *et al.* [18] proposed a single-image snow removal method based on guided filter and dictionary learning and divided the image into a high-frequency component and low-frequency component using a guide filter. These researchers designed a three-layer decomposition algorithm to extract the details in the high-frequency component and finally added the details with the low-frequency component to obtain the image without snow. Some researchers have also used sparse expressions to remove snowflakes in images. The algorithm developed by Huang *et al.* [21] is divided into two modules. The first module uses the sparse regularization method to reconstruct potential snow-free images. The second module proposes a self-correcting mechanism to seek better reconstruction of snow-free images by means of time-varying inertial weight particle swarm optimization. Accordingly, with the development of deep learning algorithms, some researchers have also used deep learning network structures to remove snow from a single image. Liu *et al.* [19] proposed a learning-based multistage snow removal network for the removal of snow from a single image. They designed a semitransparent recovery (TR) module to restore the pixels blocked by snowflakes and then used a residual generation model to generate a complete snow-free image recovery diagram based on the area not covered by snowflakes and the area recovered by the TR module. Li *et al.* [20] proposed a deep learning snow removal algorithm based on a generative adversarial network. Li *et al.* [39] proposed a new snow removal method based on a physical model and a superimposed dense network. A physics-based snowflake model was derived from a snowflake imaging process. Based on this model, a snow dataset was synthesized, and an end-to-end stacked deep learning network was designed to detect and remove snowflakes.

In video image desnowing algorithms, some researchers used the frame difference method to detect and remove snowflakes. Zhou *et al.* [6] proposed a method for the detection and removal of snow using an improved symmetrical frame difference method. Huiying and Xuejing [8] proposed a video desnowing method based on the frame difference method and added region and direction angle constraints to further distinguish the snow region. Finally, snow was removed using the five-frame difference method. Yang *et al.* [10] proposed a method based on the five-frame difference method that detects snow particles via an improved continuous five frame difference and removes snow using gradient minimization. Some researchers have detected and removed snowflakes according to their direction and brightness characteristics. Shen *et al.* [4] proposed a new snow detection method in which several filters are used to detect snowflakes in a video, and a matting algorithm is used to obtain the α value of the snowflakes. Bossu *et al.* [5] proposed a method that selects potential snowflakes in a video based on the selection rules derived from their photometry and size. Tian *et al.* [12] proposed a stereoscopic video desnowing method. These researchers synthesized a given left-view frame by warping the right-view frame, the previous frame, and the next frame. This method also eliminated outliers by investigating the geometrical distribution of snow pixels and removed snow by applying a selective nonlocal means filter. Some researchers used prior knowledge of the background to remove snowflakes. Kim *et al.* [9] proposed a video desnowing method based on temporal correlation and low-rank matrix completion. This method can handle videos captured by a moving camera. Ren *et al.* [11] divided snow into sparse and dense categories and modeled them separately in a matrix decomposition framework. Moving objects were detected with MRFs, and a group sparsity term was designed to filter snow pixels within them. Tian *et al.* [12] proposed a method for video desnowing based on low-rank decomposition, snow-free background reconstruction by global low-rank decomposition and removal of snow in front of moving objects by local low-rank decomposition.

The method proposed in this paper differs from previous algorithms in the following aspects:

1) Snow videos with different resolutions are considered. To remove snow from videos with different resolutions, we add self-adaptation threshold processing in sparse snowflake detection and moving foreground detection to help remove the detected dense snowflakes and the dynamic background while ensuring the integrity of the sparse snowflakes and moving foreground.

2) Sparse snowflakes are not separated from moving objects during moving foreground detection. We paste the detected moving foreground into a clear background image, use the block matching algorithm to find similar blocks between the current frame image and the adjacent frames, and then use the information of adjacent frames to fill the snowflake pixels in the current frame and simultaneously remove the snowflakes in front of the moving object.

III. PROPOSED METHOD

In this section, we propose a novel video desnowing method. Fig. 1 shows the snow removal process of our method. First, an optical flow estimation method, an SVM and self-adaptation processing are applied to detect sparse snowflakes in the video. A PBGMM is employed for background modeling to remove dense snowflakes. Then, the sparse snowflakes and moving objects in the video are extracted by using an MRF and adaptive threshold processing, which are combined with the background to form the new input video without dense snowflakes. Finally, the sparse snowflakes and the snowflakes in front of the moving objects are removed by using a similar block matching method based on the sparse snowflake detection map and the information from the adjacent frames.

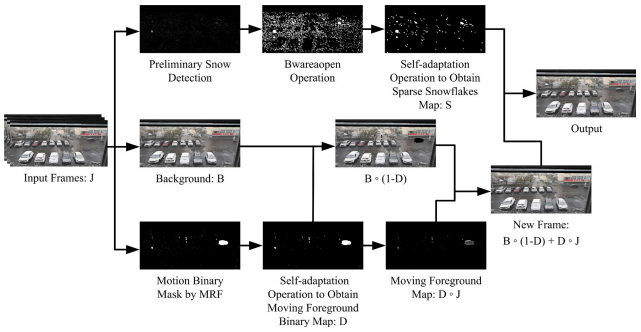

FIGURE 1. Overview of the proposed method.

A. PROBLEM MODEL

Videos are generally a 3-level multidimensional array. If the multidimensional data are reshaped into a one-dimensional vector or a two-dimensional matrix, the inherent dependence structure between multiple factors will inevitably be destroyed, which may cause information loss. Therefore, we use a tensor to decompose videos. The tensor-based principal component analysis (RPCA) method has been successfully used in the field of moving object detection in recent years, and its performance is better than that of the previous latest moving object detection technology [50]–[55]. The existing tensor-based RPCA moving object detection method decomposes a video into a low-rank component which consists of true data and a sparse component which consists of moving sparse outliers, or decomposes the video into low-rank components, sparse components and noise components. However, none of the above methods consider the presence of snowflakes in the video. Snowflakes are different from general video noise (such as Gaussian noise) and dynamic background disturbances. They have the characteristics of moving objects, which will seriously reduce the accuracy of moving object detection algorithms. In addition, due to the excellent performance of the tensor-based RPCA method, this method has also been successfully applied to the field of image and video denoising [56], [57]. Therefore, we use the tensor-based RPCA model to decompose the video and further restrict the snowflakes to achieve snow removal.

An input video can be represented as a three-dimensional tensor $L \in \mathbb{N}^{h\times w\times n}$, where h, w and n represent the height, width and number of frames of the video, respectively. The input video L can be divided into three layers:

$$L = B + M + S \tag{1}$$

where $B, M, S \in \mathbb{N}^{h\times w\times n}$, B is the background, M is the moving object, and S is the sparse snowflake. In this paper, dense snowflakes are not considered in the modeling process; those snowflakes will be removed in the background modeling process.

B. SPARSE SNOWFLAKE DETECTION

Snowflakes in a video can be divided into dense snowflakes and sparse snowflakes [11]. In this paper, the snowflake detection process includes only the detection of sparse snowflakes and not dense snowflakes. The detection of dense snowflakes is not helpful for the following desnowing process because it increases the runtime of the whole method. Therefore, we remove the dense snowflakes from the final snowflake detection map and retain only the sparse snowflakes. First, the optical flow estimation method is applied to detect the motion field of the snow removal target frame. Then, an SVM is employed to classify the detected motion and achieve preliminary snowflake detection. Finally, self-adaptation thresholding is added to remove the influence of dense snowflakes, background noise and moving object on sparse snowflake detection.

The optical flow field estimation algorithm can be applied to motion detection between video frames [9], [29], [38]. Sparse snowflakes move quickly and generally appear in at most three consecutive video images. Therefore, we use the previous frame and next frame of the target frame to help complete snowflake detection. The optical flow motion field of the target frame can be estimated by contorting the adjacent frame as the target frame image, which can be completed by the algorithm given in [38]. We contort the previous frame J_{z-1} and the image of the next frame J_{z+1} into the

current frame:

$$\overline{J}_z^{pre} = J_{z-1}(x + \delta_z^{pre}(x)) \tag{2}$$

$$\overline{J}_z^{next} = J_{z+1}(x + \delta_z^{next}(x)) \tag{3}$$

where $\bar{J}_z^{pre}$ is the target frame contorted by the previous frame J_{z-1}; $\bar{J}_z^{next}$ is the target frame contorted by the next frame J_{z+1}; $\delta_z^{pre}(x)$ represents the optical flow vector of pixel x from J_z to J_{z-1}; $\delta_z^{next}(x)$ represents the optical flow vector of pixel x from J_{z+1} to J_z. To select the pixels that are more similar to the pixel for J_z (x), the binary tag $t(x)$ is exploited to label each pixel and can be represented as follows:

$$\overline{J}_z = \begin{cases} \overline{J}_z^{pre}(x), & t(x) = 0 \\ \overline{J}_z^{next}(x), & t(x) = 1 \end{cases} \tag{4}$$

Let T represent the tag map, which is composed of the labels of all the pixels in the current frame. A pixel in the target frame is more similar to a pixel of the contorted target frame of the previous frame than that of the next frame when $t(x) = 0$. A pixel of the target frame is more similar to the pixel of the contorted target frame of the next frame than that of the previous frame when $t(x) = 1$. Through the value of the binary label $t(x)$ of each pixel, we select the pixels in the corresponding previous and next contorted frames and combine them into a contorted frame with the highest similarity to the target frame. We determine T by minimizing the tag cost as follows:

$$C(T) = C_d(T) + \tau C_s(T) \tag{5}$$

where $\tau = 50$ is a regularization parameter. $C_d(T)$ is a measure of the data cost of the contorted frame with the current frame:

$$C_d(T) = \sum_x (J_z(x) - \overline{J_k}(x))^2 \tag{6}$$

$C_s(T)$ is the smoothness cost to constrain the tags of neighboring pixels to be the same, which is defined as follows:

$$C_s(T) = \sum_x \sum_{y \in N_x} (\mathrm{t(x)} \oplus \mathrm{t(y)}) \tag{7}$$

where N_x is the set of the 4-connected neighbors to x, and "$\oplus$" is the exclusive-or operator. The initial snowflake detection map obtained by the optical flow estimation algorithm can be expressed as follows:

$$S_0(x) = \max\left\{J_z(x) - \overline{J}_z(x), 0\right\} \tag{8}$$

where the brightness of a snowflake pixel is always higher than that of a background pixel, and the pixel with a difference less than 0 will be judged as 0.

There is still considerable background noise information and information of other moving objects in the snowflake detection map obtained by the optical flow estimation method. To separate snowflakes from the rest of the information, we use the sparse representation technique [31] to represent snowflake map. First, suppose that there are $h \times w$ pixels in snowflake detection map S, and a block of size r is selected around each pixel to construct a two-dimensional matrix Ψ of size $r \times (h \times w)$. Each column of the matrix Ψ represents each selected block. Then, we construct an overcomplete dictionary $Q \in \mathbb{N}^{r \times e}$ composed of e R-dimension basis vectors with by algorithm given in [32]. Through the linear combination of the base vector of the overcomplete dictionary Q to represent every block to reconstruct the matrix Ψ:

$$\|\Psi - QA\| \tag{9}$$

where A is the coefficient matrix of size $e \times (h \times w)$. Assuming that A should be a sparse matrix, we can find the optimal coefficient matrix A^* [33] by solving the optimization problem as follows:

$$A^* = \underset{A \in \mathbb{N}^{e \times (h \times w)}}{\arg\min} \left\{ \frac{1}{2} \|\Psi - QA\|_2^2 + \rho \|A\|_1 \right\} \tag{10}$$

where ρ is set to 0.15.

After using the sparse representation to represent the detected snowflake map, the basis vectors in the overcomplete dictionary Q are classified into snowflake components and the other components. To more accurately separate the snowflake vector from all the vectors, first, we use the kernel regression method [24] to analyze the block structure, which is based on singular value decomposition to find the kernel that best matches the intensity distribution in each block. We can analyze the structural shape and orientation of the kernel for each block by applying singular value decomposition.

From the rotational angle of the obtained kernel, we used an SVM classifier to separate the snowflake vector [35]. The angular component eigenvectors of falling snowflakes are used to help the SVM separate snowflake vectors from noise vectors. To improve SVM classification, we use 3072 effective basis vectors extracted from the synthetic snow image as positive samples and 3072 noise vectors without snow as negative samples to train the SVM. This training process needs to be trained offline only once and can be directly invoked each time it is needed. After the classification, most of the noise vector in Q is substituted for the zero vector. We use $\hat{Q}$ to represent the newly classified dictionary. Using the new dictionary $\hat{Q}$ and coefficient matrix A^*, we can obtain a new matrix $\hat{\Psi}$:

$$\widehat{\Psi} = \widehat{Q}A^* \tag{11}$$

We restore the column vectors in matrix $\hat{\Psi}$ to the form of blocks. Then each block is arranged in an overlapping manner according to the arrangement of its central pixels in the snowflake detection map, and the value of each pixel in the snowflake detection map is set to the average value of the overlapping pixels to obtain a new snowflake detection map $\hat{S}$. Then, we transform the new snowflake detection map $\hat{S}$ into a binary sparse snowflake detection map U by

setting the threshold value $\phi = 3$:

$$U(x) = \begin{cases} 1, & \text{if } \widehat{S}(x) > \phi \\ 0, & others \end{cases} \tag{12}$$

Fig. 2(a) shows the input frame of the video, and Fig. 2(b) shows the snowflake detection map generated by the above process. As shown in Fig.2(b), there are dense snowflakes, background noise information and moving object information in the snowflake map. To remove other noise information, we dilate the snowflake detection map $\hat{S}$. As shown in Fig.2 (c), sparse snowflakes, dense snowflakes and background noise are clearly shown in the dilated snowflake map. We set the area threshold of the connected domain to remove the unnecessary information in the detection map. However, different videos always have different resolutions. If we set a uniform threshold to remove dense snowflakes, background noise information, and some moving object information, when performing snowflake detection on a high-resolution video, during the process of removing moving object information, the sparse snowflake information may be removed if the threshold is too small. Similarly, when performing snowflake detection on low-resolution video, the dense snowflake and background noise information may not be removed if the threshold is too large. To address the above problem, we set a self-adaptation threshold for sparse snowflake detection for both high-resolution and low-resolution videos. We set a maximum threshold to remove the moving object information as follows:

$$C_\varphi = \begin{cases} 1, & if\ C^\alpha < v_{\max} \\ 0, & others \end{cases} \tag{13}$$

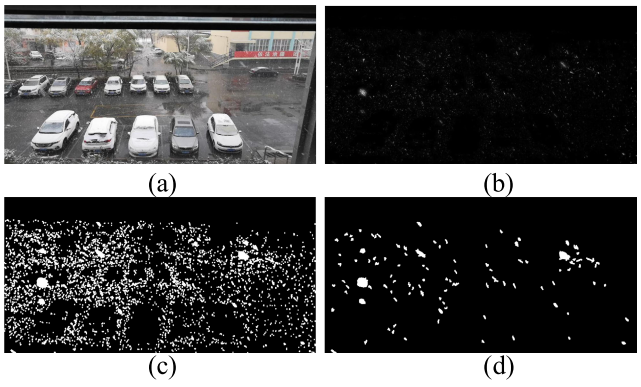

FIGURE 2. Sparse snowflake detection on a frame in the "car" video: (a) input frame; (b) snowflake detection by optical flow and an SVM; (c) dilation operation; (d) final sparse snowflake map.

where C^α takes each connection domain in the snowflake detection map; C_φ represents each pixel in the connected domain C^α in the snowflake detection map $\hat{S}$; v_{max} represents the threshold for determining the maximum area of of snowflake pixels; and the value of v_{max} is 0.25% of the total number of pixels of the image. Similarly, to detect only sparse snowflakes, we also set a minimum threshold to filter out the detected dense snowflakes:

$$C_\varphi = \begin{cases} 1, & if\ C^\alpha > v_{\min} \\ 0, & others \end{cases} \tag{14}$$

where $v_{\min}$ represents the threshold for determining the minimum area of snowflake pixels, and the value of $v_{\min}$ is 0.004% of the total number of pixels of the image. This process is used to eliminate the influence of dense snowflakes and background noise (such as shaking leaf) on the detection of sparse snowflakes. It is worth mentioning that the snowflake detection process can also detect snowflakes in front of moving objects. As shown in Fig. 2 (d), we can obtain the final sparse snowflake detection map, and the formula is expressed as follows:

$$S = U(C_\varphi) = \begin{cases} 1, & if\ v_{\min} < C^\alpha < v_{\max} \\ 0, & others \end{cases} \tag{15}$$

C. BACKGROUND MODEL

The purpose of the background modeling process is to establish a clear background without snowflakes and moving objects. This paper assumes that all the videos to be processed were shot by static cameras and that the brightness of video background B does not change suddenly in a short period of time. Each frame in the video is linearly related to the remaining frames, which can form a low-rank matrix. This process can be represented by the decomposition of the low-rank matrix:

$$B = UV^\perp \tag{16}$$

where $U \in \mathbb{N}^{m\times r}$ and $V \in \mathbb{N}^{m\times r}$; "$\perp$" represents the transpose of the matrix. We impose the following constraint on B:

$$rank(B) \le q \tag{17}$$

where q is a constant that constrains the complexity of the background model.

To address the above background modeling problem, we use the PBGMM to model the background layer [3]. Background modeling based on a PBGMM is a background representation method based on the statistical information of pixel samples. The background is represented by statistical information such as the probability density of a large number of sample pixel values over a long period of time, and then the target pixel is judged by the statistical difference, so the complex background can be modeled. The background of the video is a three-dimensional tensor, and we define the matrix f to represent the whole video:

$$f : \mathbb{N}^{h\times w\times n} \to \mathbb{N}^{p^2\times m_p} \tag{18}$$

where p represents the size of the patch and m_p represents the total number of patches in the entire video. A patch of a size $p \times p$ and a moving step of 1 is used to slide on the video to obtain pixel information, and each time the information of p^2 pixels in the continuous n-frame image of the video is

obtained. The video information obtained each time by the patch is represented as a single column of matrix f. In this way, matrix f represents a matrix composed of $p^2 \times m_p$ pixels that constitute the entire video. After decomposing the video into a matrix, the PBGMM can be defined as:

$$\sum_{k=1}^{K} \pi_k G(f(B)_m|\mu, \varepsilon_k) \tag{19}$$

where K represents the number of Gaussian models; $\pi_k > 0$ represents the mixing coefficient of the Gaussian models and $\sum_{k=1}^{K} \pi_k = 1$; $G(\cdot|, \mu, \varepsilon_k)$ represents the Gaussian distribution with mean $\mu = 0$ and covariance matrix $\varepsilon \in \mathbb{N}^{p^2 \times p^2}$. $f(\cdot)_m$ is the m-th column of matrix f. Finally, the problem of background modeling can be defined as a PBGMM with a parameter $\Theta = \{U, V, \pi, \varepsilon\}$:

$$\min_{\Theta} - \sum\nolimits_{m=1}^{m_p} \log \sum\nolimits_{k=1}^{K} \pi_k G(f(B)_m|\mu, \varepsilon_k) \tag{20}$$

The expectation maximization (EM) algorithm can be exploited to solve the above model problems. The step of steady rise in the EM algorithm can reliably find the optimal convergence value [25]. The EM algorithm estimates the values of model parameters according to the observed data. Then it estimates the value of a missing data point according to the parameter value estimated in the previous step, estimates the parameter value again according to the estimated missing data value plus the previously observed data, and then iterates repeatedly until convergence.

In the E step of the EM algorithm, we introduce a latent variable h_{mk}, where $h_{mk} \in \{0, 1\}$ and $\sum_{k=1}^{K} h_{mk} = 1$ that represents the assignment $f(B)_m$ as a specific component of the mixture model. The posterior probability of component k given $f(B)_m$ can be represented by the following:

$$\lambda_{mk} = E(h_{mk}) = \frac{\pi_k G(f(B)_m|\mu, \varepsilon_k)}{\sum_k \pi_k G(f(B)_m|\mu, \varepsilon_k)} \tag{21}$$

The M step of the EM algorithm, minimizes the parameters in Θ:

$$\min_{\Theta} \sum_{m=1}^{m_p} \sum_{k=1}^{K} \lambda_{mk} \left(\frac{1}{2} f(B)_m^{\perp} \varepsilon_k^{-1} f(B)_m + \frac{1}{2} \log |\varepsilon_k| - \log \pi_k \right) \tag{22}$$

The algorithm is repeated iteratively for the above E and M steps until it finally converges. The problem of updating π and ε can be completed by the following equations:

$$\Sigma_k = \sum\nolimits_{m=1}^{m_p} \lambda_{mk} \tag{23}$$

$$\pi_k = \frac{\Sigma_k}{\Sigma} \tag{24}$$

$$\varepsilon_k = \frac{1}{\Sigma_k} \sum\nolimits_{m=1}^{m_p} \lambda_{mk} f(B)_m^{\perp} f(B)_m \tag{25}$$

To achieve the optimal effect of the low rank matrix decomposition problem, the alternating direction method of multipliers (ADMM) [27] is used to update U and V. Fig. 3 shows the clear background obtained by the PBGMM.

(a) (b)

FIGURE 3. Background modeled by the PBGMM of the "car" video: (a) input frames; (b) clear background.

D. MOVING FOREGROUND DETECTION

The algorithms proposed by previous researchers all sought to separate snowflakes from moving objects [11], [12] when detecting moving objects to reduce the number of instances where snowflakes are misjudged as moving objects. However, these algorithms have difficulty obtaining appropriate separation conditions between moving objects and snowflakes to distinguish them. The research concept of this paper is different from that of these previous studies. In this paper, the moving objects and the sparse snowflakes in the video are both regarded as moving foreground; we detect both moving objects and sparse snowflakes simultaneously. We use a binary three-dimensional tensor $D \epsilon \mathbb{N}^{h \times w \times n}$ to represent the detection of the moving foreground:

$$D_{i,j,z} = \begin{cases} 0, & others \\ 1, & motion\ foreground \end{cases} \tag{26}$$

where $D_{i,j,z}$ represents the pixel value of pixel (i, j) in frame z.

The MRF exhibits good performance in detecting moving foreground [11]. Therefore, this paper uses the MRF to detect the foreground motion. The binary graph D obtained for moving foreground detection can be understood as an MRF energy minimization problem, which can be solved by the graph cut optimization algorithm [22].

The moving foreground detected by an MRF is shown in Fig.4(b). The binary map includes moving objects, sparse snowflakes and dense snowflakes. To remove dense snowflakes in binary map, we use equation (14) to add self-adaptation threshold processing to remove dense snowflakes with smaller areas, where the value of the self-adaptation threshold v_{min} is set to 0.0013%. A binary map of moving foreground without dense snowflakes is shown in Fig.4(c).

After moving foreground detection, the new input video can be expressed as follows:

$$L_1 = D \circ L + \widetilde{D} \circ B \tag{27}$$

where $\tilde{D}$ satisfies $\tilde{D} + D = 1$;the operator "$\circ$" indicates that the corresponding pixel is multiplied. The new video can be represented as a combination of a snow-free background and a moving foreground (including moving objects and sparse snowflakes) detected by the MRF. The dense snowflakes are removed by self-adaptation thresholding. Fig.4 shows the moving foreground detection procedure and the results of

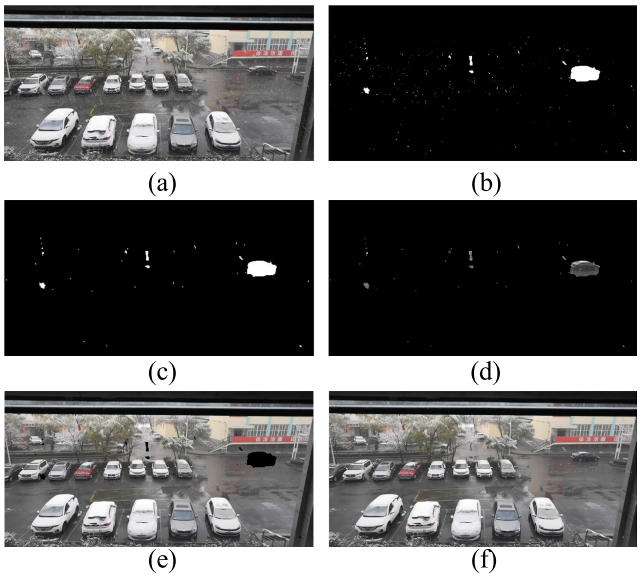

FIGURE 4. The moving foreground detection process of the "car" video: (a) the input frame; (b) moving foreground detection binary mask from the MRF; (c) self-adaptation threshold processing of the binary mask; (d) multiplication of the binary mask by the input frame; (e) multiplication of the background image by the inverse of the binary mask; (f) the new frame without dense snow.

pasting the detected moving objects and sparse snowflakes into the background to generate a video without dense snowflakes.

E. SPARSE SNOWFLAKE REMOVAL

In Section III B, the sparse snowflake detection diagram is obtained. In Section III C, the dense snowflakes are removed to restore a clear static background. In Section III D, we detect both moving objects and sparse snowflakes and paste them to a static background. Now, we have a new video that includes a static background, moving objects, and sparse snowflakes.

In this section, our task is to remove the sparse snowflakes in the video based on the sparse snowflake detection map to obtain the final video without snow. First, we divide the snowflake target frame into disjointed blocks. Then, we use the information of adjacent frames to fill in the pixels in the target frame. Different from the method from [9], to achieve excellent snow removal performance for snow videos with different resolutions, we set the block area to 2% of the single frame pixel of the video. During the process of filling information, considering the impact of heavy snow on the video, we decided to use the information of six adjacent frames to fill the information in the target frame. For each block γ in the target frame, we search for the most similar block in each of the six frames J_{z-3}, J_{z-2}, J_{z-1}, J_{z+1}, J_{z+2} and J_{z+3} before and after target frame J_z. The similarity between blocks is determined by the squared differences in the snow-free pixels between blocks. Then the $6l$ blocks that are similar to block γ in six frames are stored as columns of matrix Γ:

$$\Gamma = [\gamma, \gamma_1, \gamma_2, \cdots, \gamma_{6l}] \tag{28}$$

we also define the binary snowflake mask matrix Θ corresponding to Γ:

$$\Theta = [o, o_1, o_2, \cdots, o_{6l}] \tag{29}$$

where each column of the matrix is composed of binary snowflake detection blocks corresponding to the block in Γ.

VSRSG Algorithm

Input: video $L \in \mathbb{N}^{h\times w\times n}$; subspace rank q; the parameters of the PBGMM; the threshold of snow detection: v_{min} and v_{max}

Initialization: Initialize U, V

1 Obtain sparse Snow Map S by equation (2)-(15)
2 **while** not converge **do**
3 Evaluate λ_{mk} by equation (21)
4 Minimization parameter Θ by equation(22)
5 Update U, V by ADMM
6 **end while**
7 Motion foreground detection by MRF and self-adaptation threshold processing
8 Obtain the new input video L_1 without dense snow by equation (27)
9 Remove the sparse snow and snow in front of moving object by equation (28)-(31)

Output: The final desnowed video

We can use the low-rank compensation technique to find a filled matrix X from matrix Γ and determine that matrix X is the minimized kernel norm $\|X\|_*$ problem, which has the following constraints:

$$\tilde{\Theta} \circ X = \tilde{\Theta} \circ \Gamma \tag{30}$$

$$\Theta \circ X \leq \Theta \circ \Gamma \tag{31}$$

where $\tilde{\Theta}$ satisfies $\tilde{\Theta} + \Theta = 1$, which means that matrix $\tilde{\Theta}$ and matrix Θ are added to obtain a matrix of 1; the operator "$\circ$" indicates that the corresponding pixel is multiplied. Equation (30) indicates that the matrix X should retain the snow-free pixels in matrix Γ. Equation (31) indicates that the brightness of the snow-free pixels will be lower than that of snow pixels. We use the EM algorithm in [23], [36] to solve the optimization problem with the above constraints to obtain the filled matrix X. Then we replace the snowflake element of each block in each frame with the element at the corresponding position in matrix $\ddot{\mathrm{X}}$ to obtain the final video without snow. Fig. 5 shows how to remove sparse snowflakes by using the sparse snowflake detection map and the new video frames

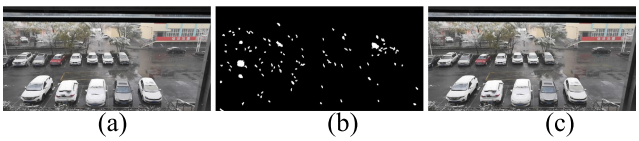

FIGURE 5. Snowflake removal for the new frame without dense snowflake: (a) the new input frame; (b) sparse snow detection map; (c) the final desnowed frame.

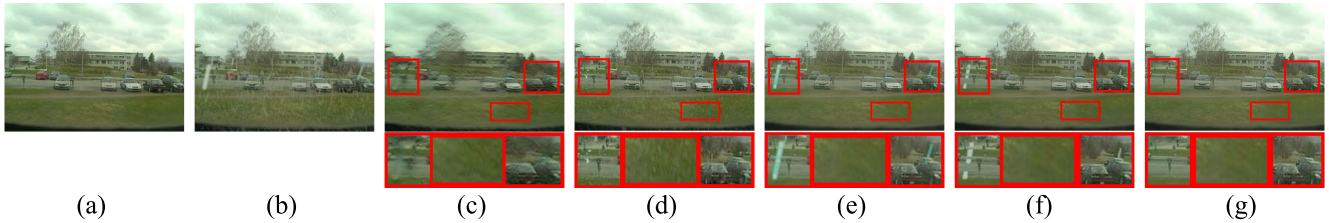

FIGURE 6. Comparison of the snow removal effects for the synthetic video called "parking": (a) the ground truth; (b) input; (c) Yang's method [37]; (d) Kim's method [9]; (e) Wei's method [3]; (f) Li's method [2]; (g) the VSRSG algorithm.

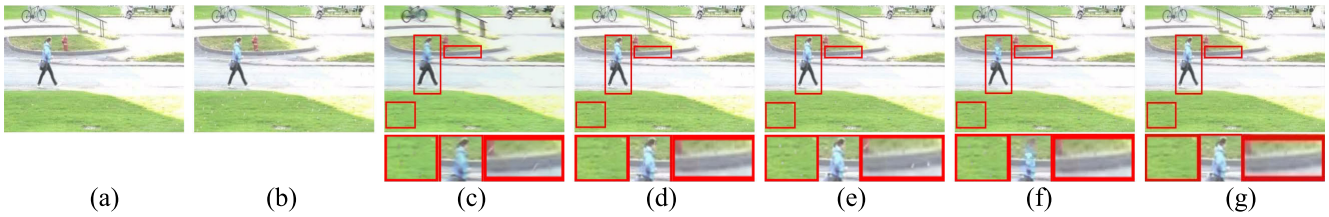

FIGURE 7. Comparison of the snow removal effects for the synthetic video called "pedestrians": (a) the ground truth; (b) input; (c) Yang's method [37]; (d) Kim's method [9]; (e) Wei's method [3]; (f) Li's method [2]; (g) the VSRSG algorithm.

without dense snowflakes. The *VSRSG* algorithm shows the whole process of our snow removal algorithm in this paper.

IV. EXPERIMENTS

To evaluate the performance of our proposed model, we show the results of our VSRSG algorithm and comparison methods on various real and synthetic snowy videos, including synthetic videos and real snow videos. All the videos shown in this work were captured by static cameras. The selected contrast areas are marked with yellow or red rectangles to facilitate observation. The comparison methods include those of Yang *et al.* [37], Kim *et al.* [9], Wei *et al.* [3] and Li *et al.* [2]. The algorithm of Yang *et al.* [37] was published in *Transactions on Pattern Analysis and Machine Intelligence* (TPAMI) in 2019. This algorithm uses a multitask deep network to perform single image rain removal. To make the algorithm suitable for image snow removal, we use the Snow100K^2 [19] dataset to train the network and then use it for experimental comparison. The algorithm of Kim *et al.* [9] was published in *Transactions on Image Processing* (TIP) in 2015 and exhibits good performance in addressing light snow scenes and moving camera snow removal. The paper by Wei *et al.* [3] was presented at the *International Conference on Computer Vision* (ICCV) in 2017, and the paper by Li *et al.* [2] was presented at the *Conference on Computer Vision and Pattern Recognition* (CVPR) in 2018. Both of these methods achieve a notably good video rain removal effect.

A. SNOW REMOVAL FOR SYNTHETIC VIDEOS

In this section, we discuss the effect of our method and the comparison methods on synthetic video desnowing. The two selected original snow-free videos were taken from the changedetection.net (CDnet) dataset [28],[1] and the resolution of both video datasets is 360 × 240 pixels. We add different degrees of snow scenes to the two videos, and use our method and the comparison methods to remove snow. The synthetic video has the original input video as the real background. We can evaluate the performance of all the methods based on three evaluation criteria: the peak signal-to-noise ratio (PSNR) [40], the feature similarity index measure (FSIM) [41] and the structural similarity index measure (SSIM) [42].

Fig. 6 shows the comparisons of the desnowing effects for the synthetic video called "parking". It can be seen from Fig. 6(b) that the synthetic snowflakes we added to the scene of "parking" include both sparse and dense snowflakes. It can be seen from the marked part in the picture that Yang's method [37] can remove most of the snowflakes in the image but blurs the image. Kim's method [14] can remove sparse snowflakes, but dense snowflakes in the far range are not removed. Wei's method [3] removes dense snowflakes but does not remove sparse snowflakes and produces color changes in sparse snowflakes. Li's method [2] removes dense snowflakes, but some sparse snowflakes remain. Our method achieves a better snow removal effect than the other methods, and our method can remove both dense and sparse snowflakes.

Fig. 7 shows the snow removal effect for the synthetic snow video called "pedestrians". As shown in the figure, Yang's method [37] can remove a portion of the snowflakes, but it causes image blur and reduces the brightness of the image. Kim's method [9] does not cause moving object deformations, but snowflakes in very bright areas are not removed. Wei's method [3] does not cause moving object deformations. However, as shown in the marked part of the image, some

[1] http://www.changedetection.net

FIGURE 8. Comparison of the snow removal effects for the real video called "house": (a) input; (b) Yang's method [37]; (c) Kim's method [9]; (d) Wei's method [3]; (e) Li's method [2]; (f) the VSRSG algorithm.

FIGURE 9. Comparison of the snow removal effects for the real video called "wuxi": (a) input; (b) Yang's method [37]; (c) Kim's method [9]; (d) Wei's method [3]; (e) Li's method [2]; (f) the VSRSG algorithm.

snowflakes have not been removed. Li's method [2] removes most of the snowflakes in the image, but it causes serious moving object deformations. The performance of our method, VSRSG, is better than those of other methods. Our method effectively removes the snowflakes in the image without causing moving object deformations.

Table 1 shows the PSNR, FSIM and SSIM, which are used to evaluate the snow removal effects of different methods on the synthetic videos "parking" and "pedestrians". In Table 1, we can see that although the differences between the evaluation results of our method and the comparison methods are not large, and, in general, the snow removal results of our method are higher than those of other comparative methods in terms of the three evaluation indexes. This section shows the snow removal effect of the proposed method in this paper and the comparison methods on synthetic snow videos. The proposed algorithm is slightly better than other comparison algorithms in terms of both the visual effect and evaluation indexes. To further confirm the snow removal effect of the proposed method, we remove snow from a real snow scene in Section IV B.

TABLE 1. Comparison of the PSNR, FSIM and SSIM for two synthetic snow videos.

	parking			pedestrians		
	PSNR	FSIM	SSIM	PSNR	FSIM	SSIM
Yang [37]	25.4264	0.8504	0.7366	27.7871	0.9834	0.8410
Kim [14]	29.7615	0.9292	0.8540	38.9786	0.9342	0.9829
Wei [5]	29.4948	0.9496	0.9140	39.0564	0.9854	0.9820
Li [4]	30.0452	0.9536	0.9127	37.4123	0.9832	0.9784
VSRSG	**31.2343**	**0.9590**	**0.9237**	**39.4550**	**0.9893**	**0.9898**

B. SNOW REMOVAL ON REAL SNOW VIDEO

In this section, we use our method and the comparison methods to remove snow from real snow scenes. All the videos were taken by static cameras. The videos in Fig. 8 and Fig. 10 were obtained from the video website New Horizon[2] and the video in Fig. 9 was obtained from YouTube.[3] The videos in Fig. 11 and Fig. 12 were captured by the authors. The videos in Fig. 13 and Fig. 14 were obtained from the common snow video called "traffic".[4]

Figs. 8-10 show real snow videos without moving objects. The resolution of the video called "house" shown in Fig.8 is 608 × 342 pixels, the resolution of the video called "wuxi" shown in Fig.9 is 1080 × 606 pixels, and the resolution of the video "forest" shown in Fig.10 is 608× 342 pixels. As seen from the marked parts in the three images, the snow removal effect of Yang's method [37] is poor, which causes the image to be blurred and creates artifacts in the image. Kim's method [9] can remove sparse snowflakes from the video, but it cannot remove dense snowflakes. Wei's method [3] can effectively remove dense snowflakes, but it cannot cleanly remove sparse snowflakes. Moreover, this method will cause image color changes. Li's method [2] can remove dense snowflakes in the video, but it is not as good as the previous three algorithms in removing sparse snowflakes. The VSRSG method proposed in this paper achieve a better snow removal effect than the other four methods and can remove sparse snowflakes and dense snowflakes without causing color changes.

Fig. 11 and Fig. 12 show the snow removal results of the 16th and 43rd frames, respectively, of the video called "car". The resolution of the video "car" is 1080 × 606 pixels. It can be seen from the figure that Yang's method [37] removes only a portion of the snowflakes and misinterprets the words on the wall as snowflakes. Kim's method [9] removes sparse

[2]http://xsj.699pic.com/
[3]https://www.youtube.com/watch?v=d5ifLTweKA8
[4]http://www.cs.columbia.edu/CAVE/projects/camera_rain/

FIGURE 10. Comparison of the snow removal effects for the real video called "forest": (a) input; (b) Yang's method [37]; (c) Kim's method [9]; (d) Wei's method [3]; (e) Li's method [2]; (f) the VSRSG algorithm.

FIGURE 11. Comparison of the snow removal effects for frame 16 of the real video called "car": (a) input; (b) Yang's method [37]; (c) Kim's method [9]; (d) Wei's method [3]; (e) Li's method [2]; (f) the VSRSG algorithm.

FIGURE 12. Comparison of the snow removal effects for frame 43 of the real video called "car": (a) input; (b) Yang's method [37]; (c) Kim's method [9]; (d) Wei's method [3]; (e) Li's method [2]; (f) the VSRSG algorithm.

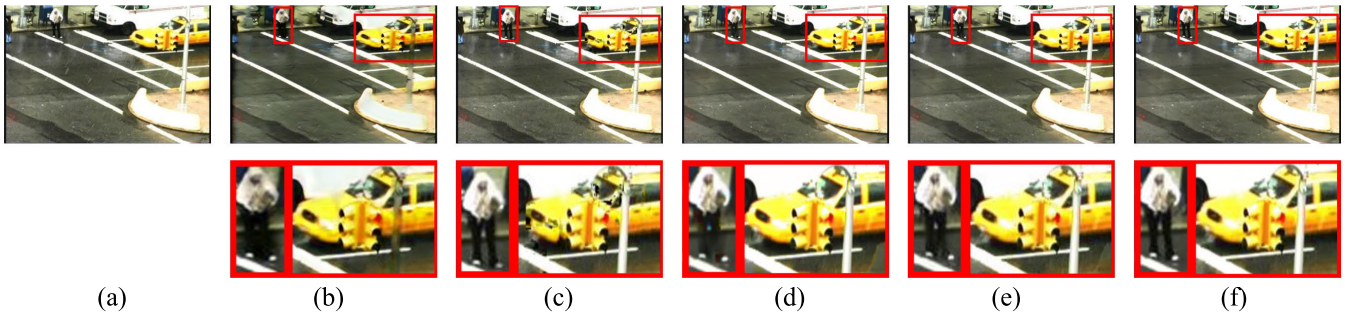

FIGURE 13. Comparison of the snow removal effects for frame 70 of the real video called "traffic": (a) input; (b) Yang's method [37]; (c) Kim's method [9]; (d) Wei's method [3]; (e) Li's method [2]; (f) the VSRSG algorithm.

snowflakes and snowflakes in front of moving objects, but this method still cannot remove dense snowflakes. Wei's method [3] removes dense snowflakes, but some sparse snowflakes remain in the image, and the algorithm does not remove snowflakes in front of moving objects. Li's method [2] also removes dense snowflakes, but large sparse snowflakes remain in the figure. This method not only fails to remove snowflakes in front of moving objects but also deforms the moving objects. The snow removal effect of our method is better than that of the other methods. Our method can remove sparse and dense snowflakes, and can remove snowflakes in front of moving objects without causing moving objects deformation.

Fig. 13 and Fig. 14 show the snow removal results of the 70th and 98th frames, respectively, of the video called "traffic". The resolution of the "traffic" video is 360×240 pixels. Yang's method [37] can remove most snowflakes, but it mistakes the utility pole with traffic lights as snowflakes, resulting in the removal of the pixel information of the utility pole. Kim's method [9] shows a better snow removal effect than Yang's method, but it deforms the moving objects. Wei's method [3] can remove most of the snowflakes, but it can

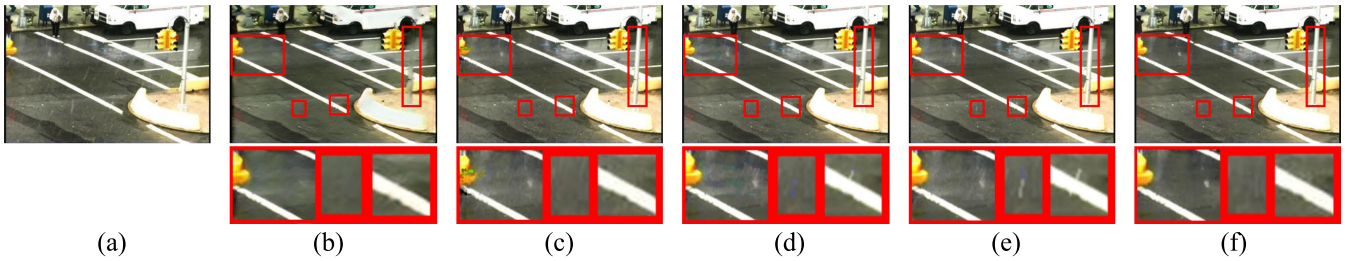

FIGURE 14. Comparison of the snow removal effects for frame 98 of the real video called "traffic": (a) input; (b) Yang's method [37]; (c) Kim's method [9]; (d) Wei's method [3]; (e) Li's method [2]; (f) the VSRSG algorithm.

be seen from the area marked in the image that this method causes a color change in the area around the moving objects. Li's method [2] does not deform the moving objects, but a few snowflakes are not removed. The snow removal results of our method are better than those of the other methods. The proposed method does not cause deformations or color changes in the moving objects while removing snowflakes.

C. COMPUTATION COMPLEXITY ANALYSIS

In this part, we will talk about the computation complexity of the proposed method. It can be seen from Section III that the proposed method includes sparse snowflake detection, background model, moving foreground detection and sparse snowflake removal. Therefore, the computation complexity of the proposed method is the sum of computation complexity of these different processes. Let h represents the high of the video, w represents the width of the video, n represents the number of frames of the video, d_b represents the number of iterations of the background model process, d_m represents the number of iterations of the moving foreground detection process. The computation complexity of the entire proposed method and each process is shown in Table 2. It can be seen from Table 2 that the computation complexity of our method is mainly affected by the background model and moving foreground detection process.

TABLE 2. Computation complexity of each process.

Process	Complexity
Sparse snowflake detection	$O(h\times w\times n)$
Background model	$O(3\times h\times w\times n\times d_b)$
Moving foreground detection	$O(3\times h\times w\times n\times d_m)$
Sparse snowflake removal	$O(6\times h\times w\times n)$
The VSRSG	$O(3\times h\times w\times n\times d_b+3\times h\times w\times n\times d_m)$

We also discuss the runtime of the proposed method. Table 3 shows the comparison of the runtimes of our method and the three traditional methods. The resolutions of the test videos are different. The experimental environment was a PC with an i7-7700 CPU and 32 GB RAM. It can be seen from the table that our method has the shortest runtime when removing snow from videos that do not contain moving objects. The proposed method has a higher runtime than Kim's method [9] but is still lower than those of Wei *et al.* [3] and Li *et al.* [2] methods when removing snow from a video that contains moving objects. The proposed method and Wei *et al.* [3] and Li *et al.* [2] methods both perform low-rank background reconstruction on all the frames of the video. Low-rank background modeling can restore a clear background image, but it will dramatically increase the runtime of the algorithm when processing a high-resolution video. The runtime of Kim's method [9] on the frame from Fig. 11 is the shortest, mainly because the method does not use low-rank background reconstruction for all the frames of the video. Regardless of the synthetic video snow removal effect, real video snow removal effect, the effect of our method is better than that of the other comparison algorithms.

TABLE 3. Comparison of the runtimes of the four competing methods on three 100-frame videos. The unit is seconds.

Dataset	Fig.6	Fig.8	Fig.11
Frame size	360×240	608×342	1080×540
Kim[9]	1934	2497	**5096**
Wei[3]	478	876	7158
Li[2]	1165	2510	7949
VSRSG	**343**	**498**	5513

V. CONCLUSION

In this paper, we propose a new video snow removal method called VSRSG. This method considers videos with heavy snow, moving objects and different resolutions. We use self-adaptation sparse snowflake detection based on an optical flow estimation method and an SVM, and use a PBGMM for background modeling. Then, an MRF and self-adaptation threshold are used to extract moving objects and sparse snowflakes in a video, which are combined with the background to form a new input video without dense snowflakes. Finally, sparse snowflakes and snowflakes in front of moving objects are removed by using a similar block matching method based on the sparse snowflake detection map and the information of adjacent frames. The experiments show that our method can achieve good performance in most snow videos. In future research, we will consider removing snow from videos captured by moving cameras.

REFERENCES

[1] K. Garg and S. K. Nayar, "Detection and removal of rain from videos," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2004, p. 1.

[2] M. Li, Q. Xie, Q. Zhao, W. Wei, S. Gu, J. Tao, and D. Meng, "Video rain streak removal by multiscale convolutional sparse coding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6644–6653.

[3] W. Wei, L. Yi, Q. Xie, Q. Zhao, D. Meng, and Z. Xu, "Should we encode rain streaks in video as deterministic or stochastic?" in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2516–2525.

[4] Y. Shen, L. Ma, H. Liu, Y. Bao, and Z. Chen, "Detecting and extracting natural snow from videos," *Inf. Process. Lett.*, vol. 110, no. 24, pp. 1124–1130, Nov. 2010.

[5] J. Bossu, N. Hautière, and J.-P. Tarel, "Rain or snow detection in image sequences through use of a histogram of orientation of streaks," *Int. J. Comput. Vis.*, vol. 93, no. 3, pp. 348–367, Jul. 2011.

[6] X. Zhou, Y. Piao, and G. Shi, "Restoration technology of the video under snow," in *Proc. Int. Conf. Comput. Sci. Electron. Technol. (ICCSET)*. Atlantis Press, 2015, pp. 387–390.

[7] J.-H. Kim, J.-Y. Sim, and C.-S. Kim, "Stereo video deraining and desnowing based on spatiotemporal frame warping," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 5432–5436.

[8] D. Huiying and Z. Xuejing, "Detection and removal of rain and snow from videos based on frame difference method," in *Proc. 27th Chin. Control Decis. Conf. (CCDC)*, May 2015, pp. 5139–5143.

[9] J.-H. Kim, J.-Y. Sim, and C.-S. Kim, "Video deraining and desnowing using temporal correlation and low-rank matrix completion," *IEEE Trans. Image Process.*, vol. 24, no. 9, pp. 2658–2670, Sep. 2015.

[10] T. Yang, V. Nsabimana, B. Wang, Y. Sun, X. Cheng, H. Dong, Y. Qin, B. Zhang, and F. Ingrabire, "Snow fluff detection and removal from video images," in *Proc. 43rd Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Oct. 2017, pp. 3840–3844.

[11] W. Ren, J. Tian, Z. Han, A. Chan, and Y. Tang, "Video desnowing and deraining based on matrix decomposition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4210–4219.

[12] J. Tian, Z. Han, W. Ren, X. Chen, and Y. Tang, "Snowflake removal for videos via global and local low-rank decomposition," *IEEE Trans. Multimedia*, vol. 20, no. 10, pp. 2659–2669, Oct. 2018.

[13] Y. Sun, X. Duan, H. Zhang, and Z. Yu, "A removal algorithm of rain and snow from images based on fuzzy connectedness," in *Proc. Int. Conf. Comput. Appl. Syst. Model. (ICCASM)*, Oct. 2010. pp. V5-478–V5-481.

[14] J. Xu, W. Zhao, P. Liu, and X. Tang, "Removing rain and snow in a single image using guided filter," in *Proc. IEEE Int. Conf. Comput. Sci. Autom. Eng. (CSAE)*, May 2012, pp. 304–307.

[15] D. Rajderkar and P. S. Mohod, "Removing snow from an image via image decomposition," in *Proc. IEEE Int. Conf. Emerg. Trends Comput., Commun. Nanotechnol. (ICECCN)*, Mar. 2013, pp. 576–579.

[16] S.-C. Pei, Y.-T. Tsai, and C.-Y. Lee, "Removing rain and snow in a single image using saturation and visibility features," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops (ICMEW)*, Jul. 2014, pp. 1–6.

[17] X. Ding, L. Chen, X. Zheng, Y. Huang, and D. Zeng, "Single image rain and snow removal via guided l0 smoothing filter," *Multimedia Tools Appl.*, vol. 75, no. 5, pp. 2697–2712, Mar. 2016.

[18] Y. Wang, S. Liu, C. Chen, and B. Zeng, "A hierarchical approach for rain or snow removing in a single color image," *IEEE Trans. Image Process.*, vol. 26, no. 8, pp. 3936–3950, Aug. 2017.

[19] Y.-F. Liu, D.-W. Jaw, S.-C. Huang, and J.-N. Hwang, "DesnowNet: Context-aware deep network for snow removal," *IEEE Trans. Image Process.*, vol. 27, no. 6, pp. 3064–3073, Jun. 2018.

[20] Z. Li, J. Zhang, Z. Fang, B. Huang, X. Jiang, Y. Gao, and J.-N. Hwang, "Single image snow removal via composition generative adversarial networks," *IEEE Access*, vol. 7, pp. 25016–25025, 2019.

[21] S.-C. Huang, D.-W. Jaw, B.-H. Chen, and S.-Y. Kuo, "Single image snow removal using sparse representation and particle swarm optimizer," *ACM Trans. Intell. Syst. Technol.*, vol. 11, no. 2, pp. 1–15, Mar. 2020.

[22] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 2, pp. 147–159, Feb. 2004.

[23] N. Srebro and T. Jaakkola, "Weighted low-rank approximations," in *Proc. 20th Int. Conf. Mach. Learn. (ICML)*, 2003, pp. 720–727.

[24] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection.Net: A new change detection benchmark dataset," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2012, pp. 1–8.

[25] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Stat. Soc. B, Methodol.*, vol. 39, no. 1, pp. 1–22, 1977.

[26] H. Yong, D. Meng, W. Zuo, and L. Zhang, "Robust online matrix factorization for dynamic background subtraction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 7, pp. 1726–1740, Jul. 2018.

[27] S. Boyd, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2010.

[28] D. Meng, Q. Zhao, and Z. Xu, "Improve robustness of sparse PCA by L1-norm maximization," *Pattern Recognit.*, vol. 45, no. 1, pp. 487–497, Jan. 2012.

[29] A. Bruhn, J. Weickert, and C. Schnörr, "Lucas/Kanade meets horn/schunck: Combining local and global optic flow methods," *Int. J. Comput. Vis.*, vol. 61, no. 3, pp. 211–231, 2005.

[30] L. Xu, J. Jia, and Y. Matsushita, "Motion detail preserving optical flow estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 9, pp. 1744–1757, Sep. 2012.

[31] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Springer, 2010.

[32] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *J. Mach. Learn. Res.*, vol. 11, pp. 19–60, 2010.

[33] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.

[34] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *IEEE Trans. Image Process.*, vol. 16, no. 2, pp. 349–366, Feb. 2007.

[35] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27, Apr. 2011.

[36] S. Zhang, W. Wang, J. Ford, and F. Makedon, "Learning from incomplete ratings using non-negative matrix factorization," in *Proc. SIAM Int. Conf. Data Mining*, Apr. 2006, pp. 549–553.

[37] W. Yang, R. T. Tan, J. Feng, Z. Guo, S. Yan, and J. Liu, "Joint rain detection and removal from a single image with contextualized deep networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 6, pp. 1377–1393, Jun. 2020.

[38] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Proc. Eur. Conf. Comput. Vis.*, 2004, pp. 25–36.

[39] P. Li, M. Yun, J. Tian, Y. Tang, G. Wang, and C. Wu, "Stacked dense networks for single-image snow removal," *Neurocomputing*, vol. 367, pp. 152–163, Nov. 2019.

[40] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of PSNR in image/video quality assessment," *Electron. Lett.*, vol. 44, no. 13, pp. 800–801, Jun. 2008.

[41] L. Zhang, L. Zhang, X. Mou, and D. Zhang, "FSIM: A feature similarity index for image quality assessment," *IEEE Trans. Image Process.*, vol. 20, no. 8, pp. 2378–2386, Aug. 2011.

[42] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[43] B. Garcia-Garcia, T. Bouwmans, and A. J. R. Silva, "Background subtraction in real applications: Challenges, current models and future directions," *Comput. Sci. Rev.*, vol. 35, Feb. 2020, Art. no. 100204.

[44] T. Bouwmans, F. Porikli, B. Höferlin, and A. Vacavant, *Background Modeling and Foreground Detection for Video Surveillance*. Boca Raton, FL, USA: CRC Press, 2014.

[45] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, L.-J. Deng, and Y. Wang, "FastDeRain: A novel video rain streak removal method using directional gradient priors," *IEEE Trans. Image Process.*, vol. 28, no. 4, pp. 2089–2102, Apr. 2019.

[46] J. Liu, W. Yang, S. Yang, and Z. Guo, "D3R-net: Dynamic routing residue recurrent network for video rain removal," *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 699–712, Feb. 2019.

[47] J. Chen, C.-H. Tan, J. Hou, L.-P. Chau, and H. Li, "Robust video content alignment and compensation for rain removal in a CNN framework," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6286–6295.

[48] W. Yang, R. T. Tan, S. Wang, and J. Liu, "Self-learning video rain streak removal: When cyclic consistency meets temporal correspondence," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1720–1729.
[49] W. Yang, J. Liu, and J. Feng, "Frame-consistent recurrent video deraining with dual-level flow," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1661–1670.
[50] N. Vaswani, T. Bouwmans, S. Javed, and P. Narayanamurthy, "Robust subspace learning: Robust PCA, robust subspace tracking, and robust subspace recovery," *IEEE Signal Process. Mag.*, vol. 35, no. 4, pp. 32–55, Jul. 2018.
[51] N. Vaswani, Y. Chi, and T. Bouwmans, "Rethinking PCA for modern data sets: Theory, algorithms, and applications [Scanning the Issue]," *Proc. IEEE*, vol. 106, no. 8, pp. 1274–1276, Aug. 2018.
[52] T. Bouwmans, S. Javed, H. Zhang, Z. Lin, and R. Otazo, "On the applications of robust PCA in image and video processing," *Proc. IEEE*, vol. 106, no. 8, pp. 1427–1457, Aug. 2018.
[53] A. Sobral, S. Javed, S. K. Jung, T. Bouwmans, and E.-H. Zahzah, "Online stochastic tensor decomposition for background subtraction in multispectral video sequences," in *Proc. IEEE Int. Conf. Comput. Vis. Workshop (ICCVW)*, Dec. 2015, pp. 106–113.
[54] S. Javed, T. Bouwmans, and S. K. Jung, "Stochastic decomposition into low rank and sparse tensor for robust background subtraction," in *Proc. 6th Int. Conf. Imag. Crime Prevention Detection (ICDP)*, 2015, pp. 1–6.
[55] Y. Sun, J. Yang, Y. Long, Z. Shang, and W. An, "Infrared patch-tensor model with weighted tensor nuclear norm for small target detection in a single frame," *IEEE Access*, vol. 6, pp. 76140–76152, 2018.
[56] A. Ravindran, M. Baburaj, and S. N. George, "Video inpainting based on re-weighted tensor decomposition," in *Proc. 2nd Int. Conf. Comput. Vis. Image Process.* Singapore: Springer, 2018, pp. 265–276.
[57] W. Dong, T. Huang, G. Shi, Y. Ma, and X. Li, "Robust tensor approximation with Laplacian scale mixture modeling for multiframe image and video denoising," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 6, pp. 1435–1448, Dec. 2018.

BIN YANG received the bachelor's degree in electronic science and technology from the School of Optoelectronics Technology, Chengdu University of Information Technology, China, in 2018. He is currently pursuing the master's degree with the School of Information Science and Engineering, Xinjiang University, China.

His research interests include image and video denoising.

ZHENHONG JIA received the B.S. degree from Beijing Normal University, Beijing, China, in 1987, and the M.S. and Ph.D. degrees from Shanghai Jiao Tong University, Shanghai, China, in 1987 and 1995, respectively.

He is currently a Professor with the Autonomous University Key Laboratory of Signal and Information Processing, Xinjiang University, China. His research interests include image processing, and photoelectric information detection and sensors.

JIE YANG (Member, IEEE) received the Ph.D. degree from the Department of Computer Science, Hamburg University, Germany, in 1994.

He is currently a Professor with the Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, China. His major research interests include object detection and recognition, data fusion and data mining, and medical image processing.

NIKOLA K. KASABOV (Fellow, IEEE) received the M.S. degree in computing and electrical engineering and the Ph.D. degree in mathematical sciences from the Technical University of Sofia, Sofia, Bulgaria, in 1971 and 1975, respectively.

He is currently the Director and the Founder of the Knowledge Engineering and Discovery Research Institute and a Professor of knowledge engineering with the School of Computing and Mathematical Sciences, Auckland University of Technology, Auckland, New Zealand. He has published more than 650 works in his research areas. His major research interests include information science, computational intelligence, neural networks, bioinformatics, neuroinformatics, and speech and image processing.