Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 6660869, 16 pages
https://doi.org/10.1155/2021/6660869

Research Article

Motion-Compensated Frame Interpolation Using Cellular Automata-Based Motion Vector Smoothing

Ran Li,[1] **Ying Yin**,[2] **Fengyuan Sun**,[3] **Yanling Li**,[1] **and Lei You**[1]

[1]*School of Computer and Information Technology, Xinyang Normal University, Xinyang 464000, China*
[2]*College of Teacher Education, Xinyang Normal University, Xinyang 464000, China*
[3]*Guangxi Key Laboratory of Wireless Wideband Communication and Signal Processing, Guilin University of Electronic Technology, Guilin 541004, China*

Correspondence should be addressed to Lei You; leiyou@xynu.edu.cn

Received 16 December 2020; Revised 27 January 2021; Accepted 2 February 2021; Published 22 February 2021

Academic Editor: Zhili Zhou

Motion-Compensated Frame Interpolation (MCFI) is one of the common temporal-domain tamper operations, and it is used to produce faked video frames for improving the visual qualities of video sequences. The instability of temporal symmetry results in many incorrect Motion Vectors (MVs) for Bidirectional Motion Estimation (BME) in MCFI. The existing Motion Vector Smoothing (MVS) works often oversmooth or revise correct MVs as wrong ones. To overcome this problem, we propose a Cellular Automata-based MVS (CA-MVS) algorithm to smooth the Motion Vector Field (MVF) output by BME. In our work, a cellular automaton is constructed to deduce MV outliers according to a defined local evolution rule. By performing CA-based evolution in a loop iteration, we gradually expose MV outliers and reduce incorrect MVs resulting from oversmoothing as many as possible. Experimental results show the proposed algorithm can improve the accuracy of BME and provide better objective and subjective interpolation qualities when compared with the traditional MVS algorithms.

1. Introduction

Motion-Compensated Frame Interpolation (MCFI) [1] is one of the common temporal-domain tamper operations, and it produces several new video frames between two neighboring video frames along motion trajectories of objects. It can be used to increase the frame rate or temporal resolution of a video sequence, so it is also called Motion-Compensated Frame Rate Upconversion (MC-FRUC) [2, 3]. MCFI is a key step for many video applications, e.g., in the low bit-rate video coding, it is used to remove the temporal redundancy [4]; in the slow replay, it is used to improve movement details in a short time interval [5]; in Liquid Crystal Display (LCD), it is used to reduce the hold-type motion blur [6], etc. Therefore, as a fundamental technique, MCFI has been keeping a high research value since it was born.

Motion Estimation (ME) and Motion-Compensated Interpolation (MCI) are the main parts in MCFI. ME is for predicting the Motion Vector Field (MVF) between the neighboring frames, and MCI is for interpolating a new frame by using the MVF output of ME. The performance of MCFI heavily depends on the prediction accuracy of ME, so many works focus on improving ME performance, and they can be classified into two types: Unidirectional ME (UME) [7] and Bidirectional ME (BME) [8]. Block Matching Algorithm (BMA) [9] is the core of UME and BME, and it is performed in order to produce block-based MVF according to the execution mechanisms of UME and BME. UME predicts the motion trajectory of each block from next frame and previous frame, then determines the MVs of blocks on these motion trajectories in the intermediate frame. The MVs produced by UME are close to reality, but for some blocks in the intermediate frame, there could be no MV or multiple MVs, thus introducing holes or overlaps. That is why many works abandon the use of UME and turn to BME. BME directly predicts the MVF of intermediate frame according to the assumption of temporal symmetry of translational motion, so each block has a unique MV, making the estimated MVF free from overlaps and holes. BME has been popular due to its straightforward implementation, though it

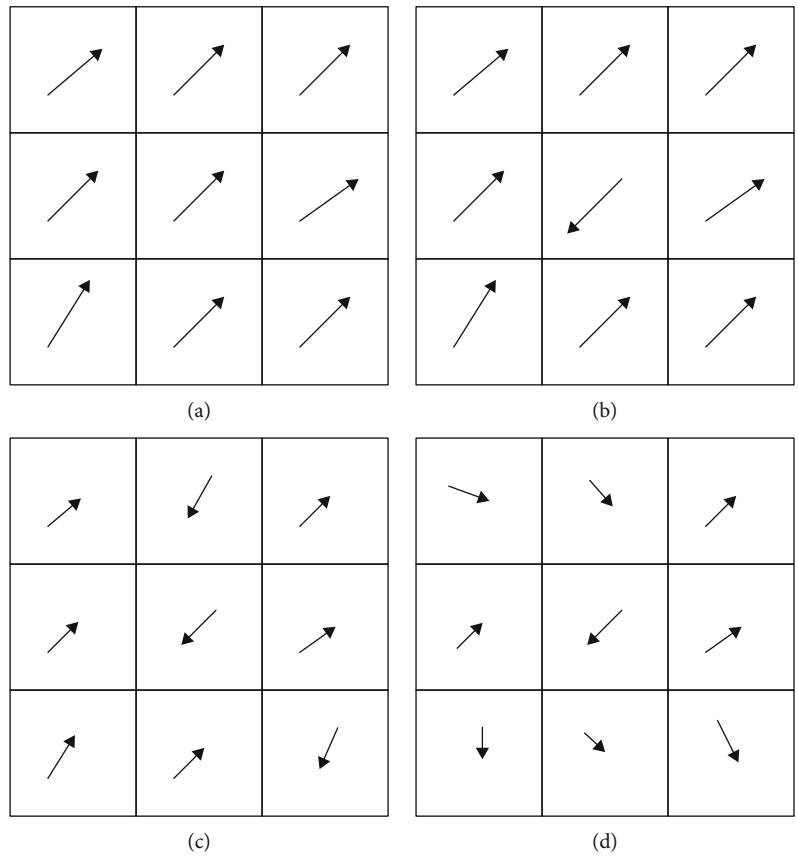


Figure 1: Illustrations on the distribution of MV outliers: (a) normal; (b) single outlier; (c) outliers in adjacent MVs; (d) various adjacent MVs.

usually produces inevitable MV outliers because the translational motion cannot describe some complex movements, e.g., rotation and deformation, and the flat region contains many structural-similar patches. To reduce MV outliers produced by BME, Motion Vector Smoothing (MVS) [10] becomes an essential postprocessing step, and in some works, the BME combined with MVS is called a true ME [11–13]. From the above, it can be seen that a high-efficiency MVS is a key to improving the prediction accuracy of BME.

A typical MVS method is Median Filter (MF) [14], which replaces the MV of a block with the mean of MVs of its adjacent blocks. When an MV outlier occurs in a flat region, MF is not to adjust this outlier along with the major direction of neighboring MVs but to produce a new MV biased toward the major direction. That keeps MVF output by MF from being a high spatial correction in the flat region. A more popular way is to use Vector-Median Filter (VMF) [15], whose effectiveness is much more evident in the motion field with a high spatial correction when compared with MF. Beyond that, VMF can also reduce impulse noise while preserving contours. In the fields including edges and textures, spatial coherence is limited by the fact that the adjacent MVs do not necessarily align with a direction, so MF and VMF present a poor ability to correct MV outliers. To suppress MV outliers in the nonflat regions, Weighed VMF (WVMF) [16] is a good way. By adaptively controlling weights, WVMF relies not only on spatial coherence but also on the measure of the matching success. Though each MV in MVF can be smoothed by any one of MF, VMF, and WVMF, not all the MVs are outliers, thus introducing computation redundancy, even oversmoothing, i.e., revising the right MV as the wrong one. To prevent oversmoothing, many works add outlier detection before filtering MVF, e.g., Yoon et al. [17] regard an MV as an outlier if its absolute horizontal or vertical component value is larger than the average of those of its neighboring MVs; Kim and Sunwoo [18] detect an MV outlier by comparing the absolute difference of the current MV with the mean MV of its neighboring MVs. However, these methods cannot identify outliers accurately, especially that some obvious outliers are often omitted. Another way of suppressing oversmoothing is to refine MVF by imposing spatial-temporal smoothness constraint upon BMA, e.g., Huang et al. [19] designed a Spatial MVS (S-MVS), which uses Markov Random Field (MRF) to model spatial smoothness constraint of MVF, and refine each MV by performing BMA with MRF-based penalty term; Yoo et al. [20] proposed a Temporal MVS (T-MVS), which selects a reliable MV from the temporal-neighbor MVs along the forward and backward directions. With the spatial-temporal smoothness constraint, these methods make MVF closer to reality but cost lots of computations. So, both correction capability and computational complexity being considered, a more effective MVS can be realized by combining outlier detection and spatial-temporal smoothing. In this work, we try to provide a

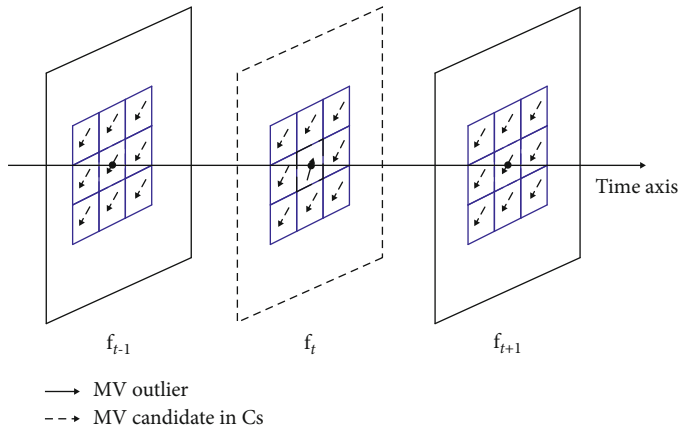


Figure 2: Illustrations on the spatial-temporal MV candidates.

solution to make better use of the advantages of outlier detection and spatial-temporal smoothing.

In this paper, we first perform BME to generate the initial MVF of the intermediate frame; then, a Cellular Automata- (CA-) based MVS (CA-MVS) algorithm is proposed to filter the MVF output by BME. The proposed CA-MVS algorithm is a combination of outlier detection and spatial-temporal smoothing, in which CA is the key to a trade-off between oversmoothing and computational redundancy. Compared with the existing works, the main contributions of our work are summarized as follows:

(i) MV outliers are found in MVF by measuring the angle between each MV and the mean of its neighboring MVs. A harsh threshold on the angle is set in order to pick out the evident outliers

(ii) According to the positions of outliers, the outlier map is generated, and input to CA. By the evolution rule of CA, some hidden outliers are found. After several iterations, the distribution of MV outliers tends to be stable, resulting in a good balance between oversmoothing and computational redundancy

(iii) Spatial-temporal coherence is exploited to correct MV outliers. For any MV outlier, VMF is first performed on its neighboring MVs to obtain the spatially predictive MV. Then, we refine this outlier by using the temporally neighboring MV candidates along the above-mentioned spatially predictive MV

Experimental results show that the proposed CA-MVS algorithm can improve the prediction accuracy of BME and provide better objective and subjective interpolation qualities than the traditional MVS methods.

2. Background

2.1. Motion Vector Smoothing (MVS). BME suffers from a limitation concerning the fidelity of the predicted MVF: MV outliers are introduced for the instability of temporal symmetry assumption. This drawback can be effectively overcome by performing Motion Vector Smoothing (MVS)

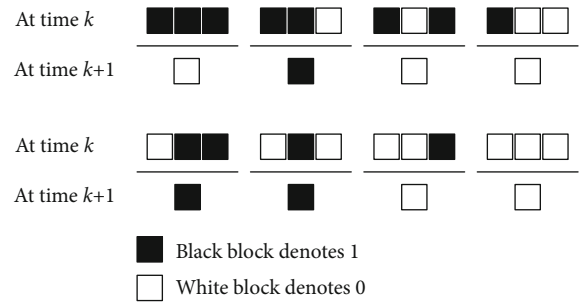


Figure 3: One-dimensional cellular automaton with code 76.

in the further stage. As shown in Figure 1(a), due to the spatial coherence of MVF, a true MV is similar in value and direction to its adjacent MVs. An example of an MV outlier is illustrated in Figure 1(b), an MV is detected as an outlier once it is different from its adjacent MVs in value or direction. In this case, MF or VMF can be used to correct this outlier. Suppose v_0 is an MV outlier, and $\mathbf{v}_1 - \mathbf{v}_8$ are its adjacent MVs, MF is performed as follows:

$$\bar{\mathbf{v}}_0 = \frac{1}{8}\sum_{i=1}^{8}\mathbf{v}_i. \tag{1}$$

$\bar{\mathbf{v}}_0$ is the corrected MV of $\mathbf{v}_0$, at the geometric center of $\mathbf{v}_1 - \mathbf{v}_8$. As shown in Figure 1(c), if there are outliers in $\mathbf{v}_1 - \mathbf{v}_8$, the mean MV cannot align at the main direction. An alternative solution is VMF, which outputs the median MV $\hat{\mathbf{v}}_0$ of $\mathbf{v}_1 - \mathbf{v}_8$ as the corrected MV, i.e.,

$$\hat{\mathbf{v}}_0 = \begin{bmatrix} \text{med}\{v_{1,\text{h}},\cdots,v_{i,\text{h}},\cdots,v_{8,\text{h}}\} \\ \text{med}\{v_{1,\text{v}},\cdots,v_{i,\text{v}},\cdots,v_{8,\text{v}}\} \end{bmatrix}, \tag{2}$$

in which med$\{\cdot\}$ produces the median value of input set, $v_{i,\text{h}}$ is the horizontal component of the ith adjacent MV $\mathbf{v}_i$, and $v_{i,\text{v}}$ is the vertical component of the ith adjacent

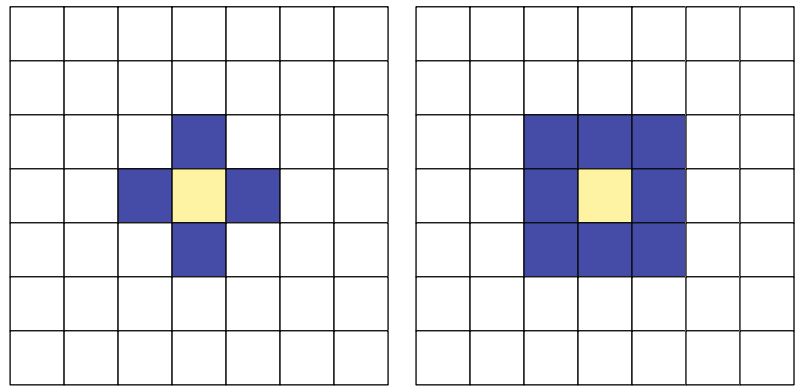


Figure 4: Examples of neighbors set: (a) von Neumann neighborhood; (b) Moore neighborhood.

MV $\mathbf{v}_i$. The median MV $\hat{\mathbf{v}}_0$ satisfies the following property:

$$\hat{\mathbf{v}}_0 \in \{\mathbf{v}_1, \cdots, \mathbf{v}_i, \cdots, \mathbf{v}_8\},$$
$$\sum_{i=1}^{8} \|\hat{\mathbf{v}}_0 - \mathbf{v}_i\| \leq \sum_{i=1}^{8} \|\mathbf{v}_j - \mathbf{v}_i\|, \quad j = 1, 2, \cdots, 8, \tag{3}$$

in which $\|\bullet\|$ is the norm. Equation (3) indicates that the sum of distances between the median MV and adjacent MVs is smaller than the one between any MV in $\mathbf{v}_1 - \mathbf{v}_8$ and its adjacent MVs. This property forces $\hat{\mathbf{v}}_0$ to be biased toward the main direction of $\mathbf{v}_1 - \mathbf{v}_8$. A drawback of VMF lies in the lack of turning parameters. As shown in Figure 1(d), if the adjacent MVs vary in value and direction, the median MV is unreliable since we are confident in the validity of any MV. To overcome such a drawback, WVMF has been introduced based on VMF and is defined as

$$\hat{\mathbf{v}}_0 \in \{\mathbf{v}_1, \cdots, \mathbf{v}_i, \cdots, \mathbf{v}_8\},$$
$$\sum_{i=1}^{8} w_i \|\hat{\mathbf{v}}_0 - \mathbf{v}_i\| \leq \sum_{i=1}^{8} w_i \|\mathbf{v}_j - \mathbf{v}_i\|, \quad j = 1, 2, \cdots, 8, \tag{4}$$

in which w_i is the weight coefficient corresponding to $\mathbf{v}_i$. The fixed weights are first considered; then, the definition is extended to the case of adaptively varying weights. The more reliable $\mathbf{v}_i$ is, the higher the corresponding weight w_i is. Therefore, the weighted-median MV is close to the reliable MVs.

MF, VMF, and WVMF exploit the spatial coherence of MVF to smooth MV, so their performances degrade when the spatial coherence is limited. In addition to spatial coherence, temporal coherence can also be used to construct the smoothness constraint into BMA. As shown in Figure 2, by assuming that MVs remain stable in a local region along time axis, the temporally and spatially neighboring MVs are combined in a candidate set $\mathbf{C}_S$. MV outlier can be refined by searching more reliable MV candidates in $\mathbf{C}_S$. To measure the reliability of MV candidates, the Sum of Bidirectional Absolute Difference (SBAD) is defined by

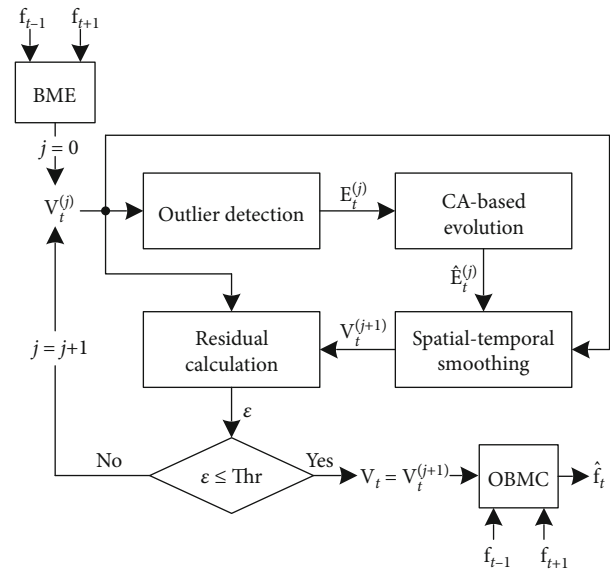


Figure 5: Framework of the proposed CA-MVS algorithm.

$$\mathrm{SBAD}_{\mathbf{B}}(\mathbf{v}_c) = \sum_{(x,y)\in\mathbf{B}} \left| f_{t-1}(x - v_{c,\mathrm{h}}, y - v_{c,\mathrm{v}}) - f_{t+1}(x + v_{c,\mathrm{h}}, y + v_{c,\mathrm{v}}) \right|, \tag{5}$$

in which $\mathbf{v}_c$ is the MV candidate, $\mathbf{B}$ is the pixel position set of the current block in the intermediate frame, $v_{c,\mathrm{h}}$ and $v_{c,\mathrm{v}}$ are horizontal and vertical components of $\mathbf{v}_c$, respectively, and $f_{t-1}(x, y)$ and $f_{t+1}(x, y)$ represent the pixel value at the position (x, y) in the previous frame and next frame, respectively. By the SBAD criterion, the MV outlier $\mathbf{v}_0$ can be smoothed as follows:

$$\tilde{\mathbf{v}}_0 = \arg \min_{\mathbf{v}_c \in \mathbf{C}_S} \mathrm{SBAD}_{\mathbf{B}}(\mathbf{v}_c), \tag{6}$$

in which $\tilde{\mathbf{v}}_0$ is the refined MV of $\mathbf{v}_0$. Combined with SBAD, MVS can fully exploit the temporal coherence of MVF. For the above-mentioned MVS algorithms, an inevitable defect is

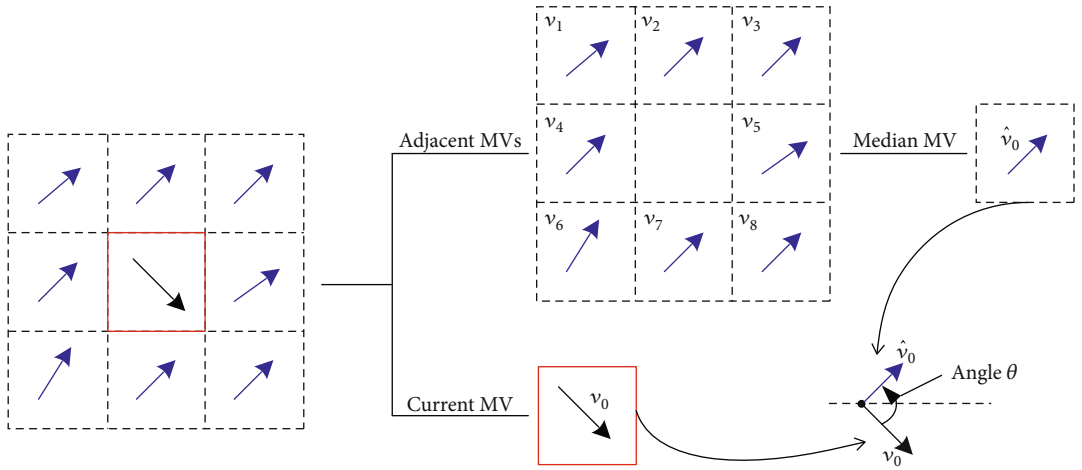


Figure 6: Illustration on angle computing.

oversmoothing, i.e., revising true MV as a wrong one, which not only reduces the fidelity of MVF but also introduces invalid computations. To overcome oversmoothing which results from the unstable spatial-temporal coherence, we need a proper mechanism of outlier detection to correctly recognize the MV outlier.

2.2. Cellular Automata (CA). Cellular Automata (CA) are discrete dynamical systems with a simple structure to investigate self-organization in statistical mechanics, and they were originally introduced by von Neumann and Ulam as a possible idealization of biological systems [21]. It is when a computer game "Life," an application of a two-dimensional cellular automaton, became successful that CA began to attract researchers' attention [22]. Then, Stephen Wolfram improved the theory of CA by an in-depth and comprehensive study on the elementary CA [23–25]. It has been widely used in a variety of fields such as sociology, graphics, and physics [26].

The construction of a cellular automaton can be represented by the following formula:

$$\mathrm{CA} = (\Omega, \mathbf{S}, \mathbf{\Lambda}, d, g), \tag{7}$$

which shows that each point in a d-dimensional lattice Ω, called a cell, can take any one from a finite state set $\mathbf{S}$, and the states of the cells of a lattice are updated according to a local evolution rule g, i.e.,

$$S_i^{k+1} = g\left(S_i^k, S_{i\pm r}^k\right), \quad r \in \mathbf{\Lambda}, \tag{8}$$

which denotes that the state S_i^{k+1} of a cell at time $k+1$ depends on its own state S_i^k at time k, and the states $S_{i\pm r}^k$ of cells in its neighbors set $\boldsymbol{\Lambda}$ at time k. All cells in the lattice are updated synchronously, and the state of the lattice advances in discrete time steps. An example of the one-dimensional cellular automaton with code 76 is illustrated in Figure 3. All cells are arranged in a line. Each cell has binary states 1 or 0, and its left and right cells construct its neighbors set. The eight possible states of three adjacent cells are given at time k; then, the central cell of the three takes its state at the next time $k+1$ by a defined rule. The time evolution of the complete cellular automaton is obtained by simultaneous application of the rule at each cell for each time step. In a two-dimensional case, different definitions of neighbors set are possible, among which von Neumann neighborhood and Moore neighborhood are common. As shown in Figure 4(a), four cells, the cells above and below each cell and the two on its right and left, are called von Neumann neighborhood of this cell. Moore neighborhood is illustrated in Figure 4(b), and it is an enlargement of von Neumann neighborhood containing the diagonal cells.

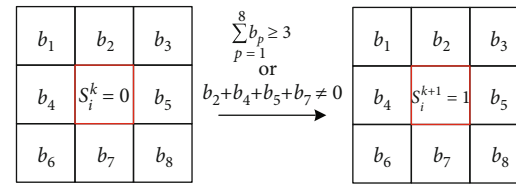


Figure 7: Local evolution rule of the constructed cellular automaton.

Table 1: Truth table on CA-based evolution.

S_i^k	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	S_i^{k+1}
1	×	×	×	×	×	×	×	×	1
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	1	0
0	0	0	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0	1	0
0	0	0	1	0	0	1	0	0	0
0	1	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	1	0	0	0
0	1	0	1	0	0	0	0	0	0
0	Else								1

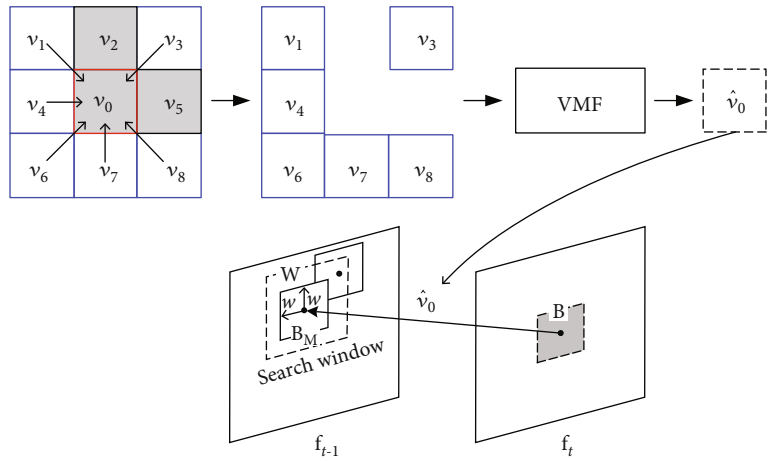


Figure 8: Illustration on spatial-temporal smoothing. Gray blocks denote MV outliers, and white blocks denote correct MVs.

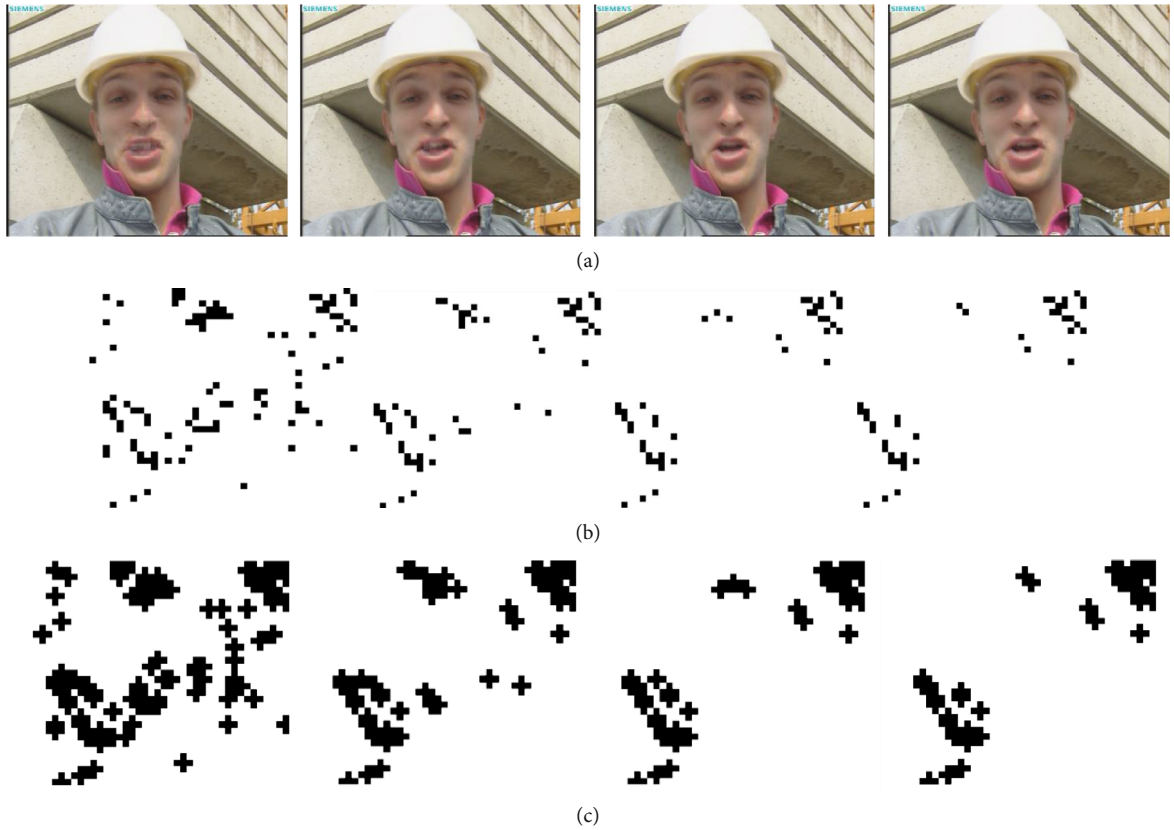
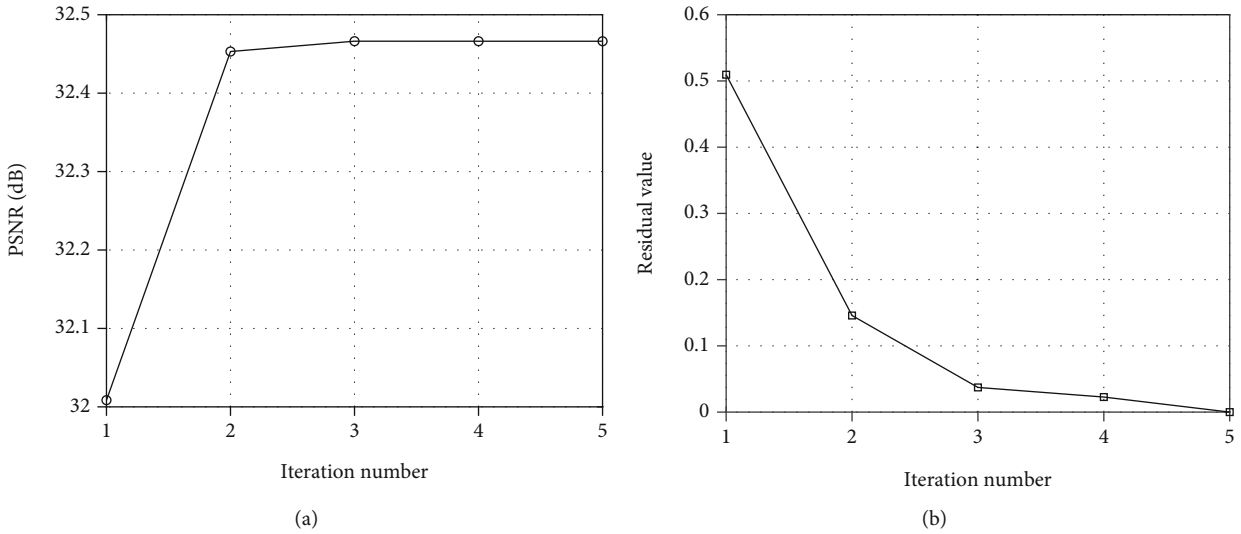


Figure 9: Interpolated frame and the corresponding outlier map at each iteration in the CA-MVS algorithm for the 82-rd frame of *Foreman* sequence. In each subfigure, from left to right: results after the first, second, third, and fourth iteration. For the outlier map, the black block denotes outlier, and the white block denotes nonoutlier. (a) Interpolated results; (b) outlier maps after detecting; (c) outlier maps after CA-based evolution.

By properly modeling objective behaviors as evolution rules, CA can simulate the running of some physical systems. Particularly in two dimensions, CA have been extensively used to exploit the statistics and latent features of images [27, 28]. The MVF of video frame is a two-dimensional lattice, and there exists inherent coherence between adjacent MVs, enabling CA to deduce the statistical mechanics of MV outliers in MVF. Motivated by the rule evaluation of CA, we attempt to construct a two-dimensional cellular automaton to model a universal law analogous to the variation of MV outliers. The cellular automaton controls the evaluation of the outlier map and helps MVS to get a good trade-off between oversmoothing and computational complexity.



(a) (b)

Figure 10: PSNR and residual values per iteration in the CA-MVS algorithm for the 82-rd frame of *Foreman* sequence: (a) PSNR curve; (b) residual curve.

3. Proposed CA-Based MVS (CA-MVS) Algorithm

3.1. Framework Overview. Figure 5 shows the framework of the proposed CA-MVS algorithm. BME is first performed to produce the initial MVF $\mathbf{V}_t^{(0)}$ from the intermediate frame $\mathbf{f}_t$ to the previous frame $\mathbf{f}_{t-1}$. According to the assumption of temporal symmetry, we also get the initial MVF $-\mathbf{V}_t^{(0)}$ from the intermediate frame $\mathbf{f}_t$ to the next frame $\mathbf{f}_{t+1}$. Then, $\mathbf{V}_t^{(0)}$ is input into the proposed CA-MVS algorithm, which executes a loop of outlier detection, CA-based evolution, spatial-temporal smoothing, and residual calculation. In the jth iteration, we detect the MV outliers in $\mathbf{V}_t^{(j)}$, and the binary map $\mathbf{E}_t^{(j)}$ is generated to reveal the distribution of outliers. To avoid wrong detection results, outlier detection is limited strictly, so the outlier map $\mathbf{E}_t^{(j)}$ only reflects the evident MV outliers. Based on $\mathbf{E}_t^{(j)}$, CA-based evolution is performed to find the hidden outliers. The constructed cellular automaton uses the propagation effect of outliers to model the local evolution rule and deduces $\widehat{\mathbf{E}}_t^{(j)}$ from $\mathbf{E}_t^{(j)}$ to expose some hidden outliers. The spatial-temporal smoothing is used to correct any outlier pointed by $\widehat{\mathbf{E}}_t^{(j)}$ in $\mathbf{V}_t^{(j)}$ and outputs the MVF $\mathbf{V}_t^{(j+1)}$ of $\mathbf{f}_t$ at the $(j+1)$th iteration. We select the less spatial-temporal neighboring MVs to smooth MV outliers in order to suppress oversmoothing. To decide whether to quit the loop, we calculate the residual ε between $\mathbf{V}_t^{(j+1)}$ and $\mathbf{V}_t^{(j)}$ as follows:

$$\varepsilon = \frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} \sum_{l=1}^{2} \left| V_t^{(j+1)}[m, n, l] - V_t^{(j)}[m, n, l] \right|, \quad (9)$$

in which $M \times N$ denotes the spatial size of $\mathbf{V}_t^{(j)}$, (m, n) denotes the spatial position of each MV in $\mathbf{V}_t^{(j)}$, and l denotes the horizontal or vertical component of MV, i.e., $l = 1$ means the horizontal component, and $l = 2$ means the vertical component. The similarity between $\mathbf{V}_t^{(j+1)}$ and $\mathbf{V}_t^{(j)}$ is measured by the residual ε, and a threshold Thr is set to determine whether to perform the next iteration. Once ε is less than or equal to Thr, we make the loop exit and output $\mathbf{V}_t^{(j+1)}$ as the final MVF $\mathbf{V}_t$ of $\mathbf{f}_t$. After several iterations, by the evolution of CA, the number of outliers tends to be stable, preventing oversmoothing introduced by redundant computations. According to $\mathbf{V}_t$, the Overlapped Block Motion Compensation (OBMC) [29] is finally performed on $\mathbf{f}_{t-1}$ and $\mathbf{f}_{t+1}$ to interpolate the intermediate frame $\mathbf{f}_t$. Outlier detection, CA-based evolution, and spatial-temporal smoothing are the important parts of the proposed CA-MVS algorithm, and the following describes them in detail.

3.2. Outlier Detection. An evident MV outlier shows its large angle with respect to one of its adjacent MVs, so we use the angle between every two adjacent MVs to detect the outliers in MVF. A 3×3 window is used to scan all MVs in MVF from left to right and from top to bottom, and as shown in Figure 6, in the MV window, $\mathbf{v}_0$ is the MV to be detected in the red block, and $\mathbf{v}_1 - \mathbf{v}_8$ are eight adjacent MVs of $\mathbf{v}_0$, which are marked in blue. It costs many computations to compute all angles between $\mathbf{v}_0$ and $\mathbf{v}_1 - \mathbf{v}_8$. In order to reduce computations, we first get the median MV $\widehat{\mathbf{v}}_0$ of $\mathbf{v}_1 - \mathbf{v}_8$ according to Equation (2), then compute the angle θ between $\mathbf{v}_0$ and $\widehat{\mathbf{v}}_0$ as follows:

$$\theta = \arccos\left(\frac{\widehat{\mathbf{v}}_0 \cdot \mathbf{v}_0}{\|\widehat{\mathbf{v}}_0\|_2 \|\mathbf{v}_0\|_2}\right), \quad \theta \in [-\pi, \pi], \quad (10)$$

in which $\arccos(\cdot)$ is the arc-cosine function. $\widehat{\mathbf{v}}_0$ represents the main direction of $\mathbf{v}_1 - \mathbf{v}_8$, so a large θ means that $\mathbf{v}_0$ deviates far from its adjacent MVs. Based on this experience, we regard $\mathbf{v}_0$ as an evident outlier if $|\theta|$ is larger than $\pi/2$. After detecting all MVs in $\mathbf{V}_t^{(j)}$, the outlier map $\mathbf{E}_t^{(j)}$ is generated, in which outlier and nonoutlier are marked with 1 and 0, respectively. This

Figure 11: Visual results on the 14th interpolated frame of the *Foreman* sequence with different MVS algorithms: (a) MF; (b) VMF; (c) WVMF; (d) S-MVS; (e) T-MVS; (f) CA-MVS.

strict criterion makes the exposed outliers more reliable, but it also excludes some real outliers. These hidden outliers lead to the trade-off between oversmoothing and computations: if the hidden outliers are omitted, computations invested to smooth outlier are not many, but they can mislead the correction of exposed outliers, resulting in oversmoothing. Therefore, we add CA-based evolution to keep a good balance between oversmoothing and computations.

3.3. CA-Based Evolution. To ensure the accuracy of detecting outliers, the evident outliers are only marked with 1 in $\mathbf{E}_t^{(j)}$. When we use $\mathbf{E}_t^{(j)}$ to decide whether to smooth any MV in $\mathbf{V}_t^{(j)}$, those hidden outliers cannot be corrected, and the exposed outliers would also be modified incorrectly once some outliers hide in its adjacent MVs. It is necessary to find these hidden outliers in $\mathbf{V}_t^{(j)}$ in order to solve the above problem of oversmoothing. Motivated by the CA theory, we construct a cellular automaton to model the interaction between outliers in $\mathbf{E}_t^{(j)}$ and deduce the hidden outliers from the exposed ones according to the defined local evolution rule. $\mathbf{E}_t^{(j)}$ is a 2-dimensional lattice, and its element is regarded as a cell. Each cell has two states 0 and 1, denoting nonoutliers and outlier, respectively. Due to the locally stationary statistics of MVF, outliers propagate in its neighborhood. In a 3×3 window, this propagation effect can be enhanced when an outlier is closer to the center or more outliers occur in the window. Based on this experience, as shown in Figure 7, we define a local evolution rule for the constructed cellular

(a) (b) (c) (d) (e) (f)

FIGURE 12: Visual results on the 58th interpolated frame of the *Mobile* sequence with different MVS algorithms: (a) MF; (b) VMF; (c) WVMF; (d) S-MVS; (e) T-MVS; (f) CA-MVS.

automaton. Suppose the state S_i^k of the current cell is 0, and the states of its adjacent cells are $b_1 - b_8$, which have two possible values 0 and 1. The state S_i^{k+1} of the current cell at time step $k+1$ is specified as follows:

$$S_i^{k+1} = g\left(S_i^k, \{b_p\}_{p=1,2,\cdots,8}\right) = \begin{cases} 1, & \sum_{p=1}^{8} b_p \geq 3, \\ 1, & b_2 + b_4 + b_5 + b_7 \neq 0, \\ 0, & \text{else.} \end{cases} \tag{11}$$

According to Equation (11), the current MV can be identified as an outlier if outliers occur in its von Neumann neighborhood or more than three outliers occur in its Moore neighborhood. The states at each cell in $\mathbf{E}_t^{(j)}$ are updated simultaneously, and these updated states form the new outlier map $\widehat{\mathbf{E}}_t^{(j)}$. To speed up the CA-based evolution, we summarize the truth table on the above defined local rule, as shown in Table 1. According to Table 1, the state of the cell at a given time is obtained immediately depending on the logical combination of its neighbors' states at the previous time step. The CA-based evolution propagates the outlier in a local region and deduces the moderate number of hidden outliers.

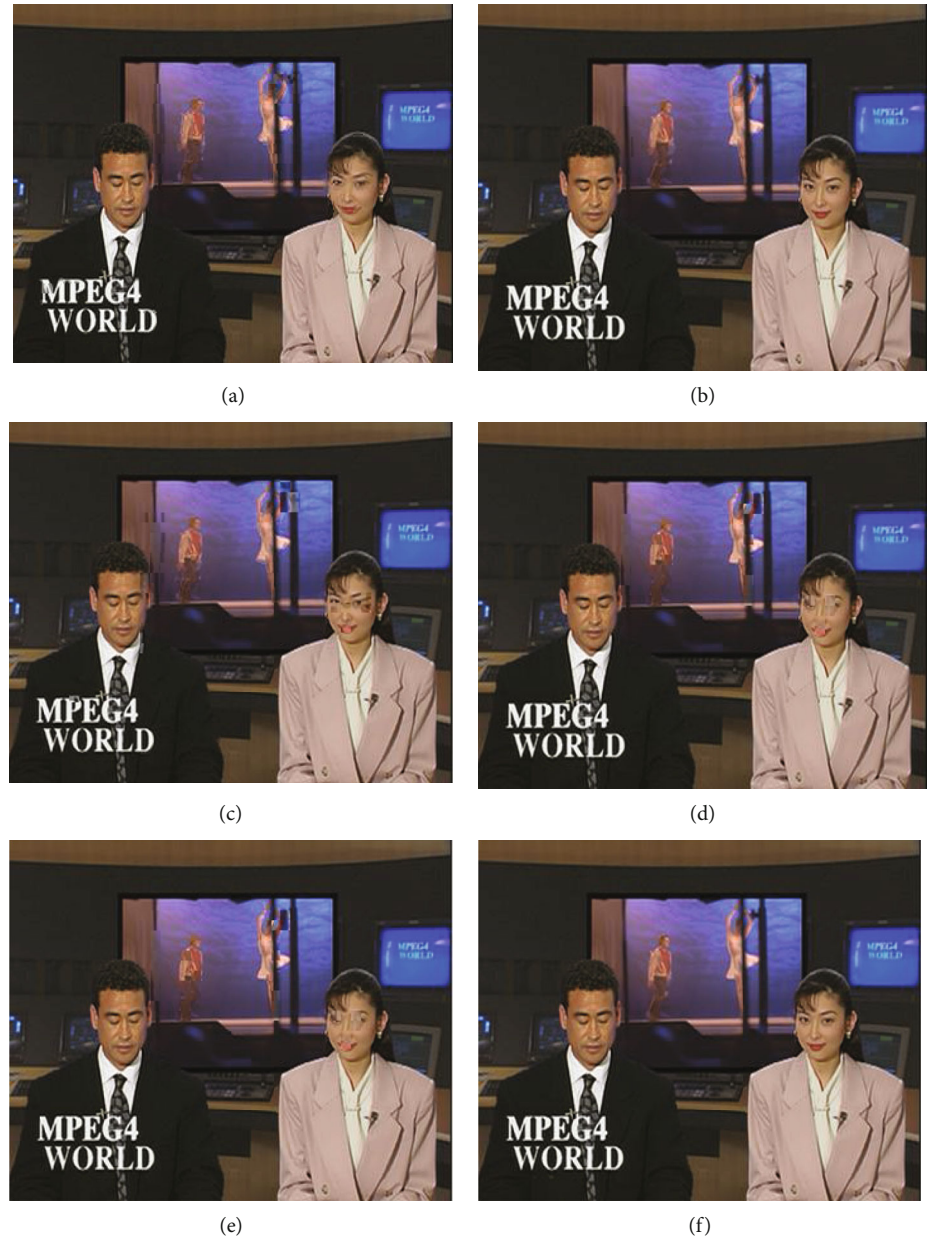


Figure 13: Visual results on the 22-rd interpolated frame of the *News* sequence with different MVS algorithms: (a) MF; (b) VMF; (c) WVMF; (d) S-MVS; (e) T-MVS; (f) CA-MVS.

Compared with $\mathbf{E}_t^{(j)}$, $\widehat{\mathbf{E}}_t^{(j)}$ can smooth more outliers, and at the same time, the increased computations can achieve some alleviation of oversmoothing, bringing a good balance between oversmoothing and computational complexity.

3.4. Spatial-Temporal Smoothing. Any outlier pointed by $\widehat{\mathbf{E}}_t^{(j)}$ is corrected by the proposed spatial-temporal smoothing. To improve the correction capability, the existing works are trying to construct a large-scale candidate set by exploiting the spatial-temporal coherence of MVF. However, the more candidate MVs there are, the more computations are invested, particularly the higher the probability of outliers occurring in the candidate set is. We combine VMF into the construction of candidate MV set in the spatial-temporal neighborhood, which can simplify the set of candidate MVs while providing a robust capability to correct outliers. The flow of spatial-temporal smoothing is illustrated in Figure 8. Suppose $\mathbf{v}_0$ is the MV outlier to be smoothed, and $\mathbf{v}_1 - \mathbf{v}_8$ are its adjacent MVs, in which $\mathbf{v}_2$ and $\mathbf{v}_5$ are MV outliers. First, we input the correct MVs in $\mathbf{v}_1 - \mathbf{v}_8$ into VMF and generate their median MV $\widehat{\mathbf{v}}_0$. For the current block $\mathbf{B}$, we find the matching block $\mathbf{B}_{\mathrm{M}}$ in $\mathbf{f}_{t-1}$ along $\widehat{\mathbf{v}}_0$ and open a search window $\mathbf{W}$ centered $\mathbf{B}_{\mathrm{M}}$. We set the radius w of $\mathbf{W}$ to be 1 and collect the relative displacements from $\mathbf{B}$ to all search points in $\mathbf{W}$ as the candidate MV set $\mathbf{C}_{\mathrm{S}}$. The SBAD value of each MV candidate is computed according to Equation (5) and

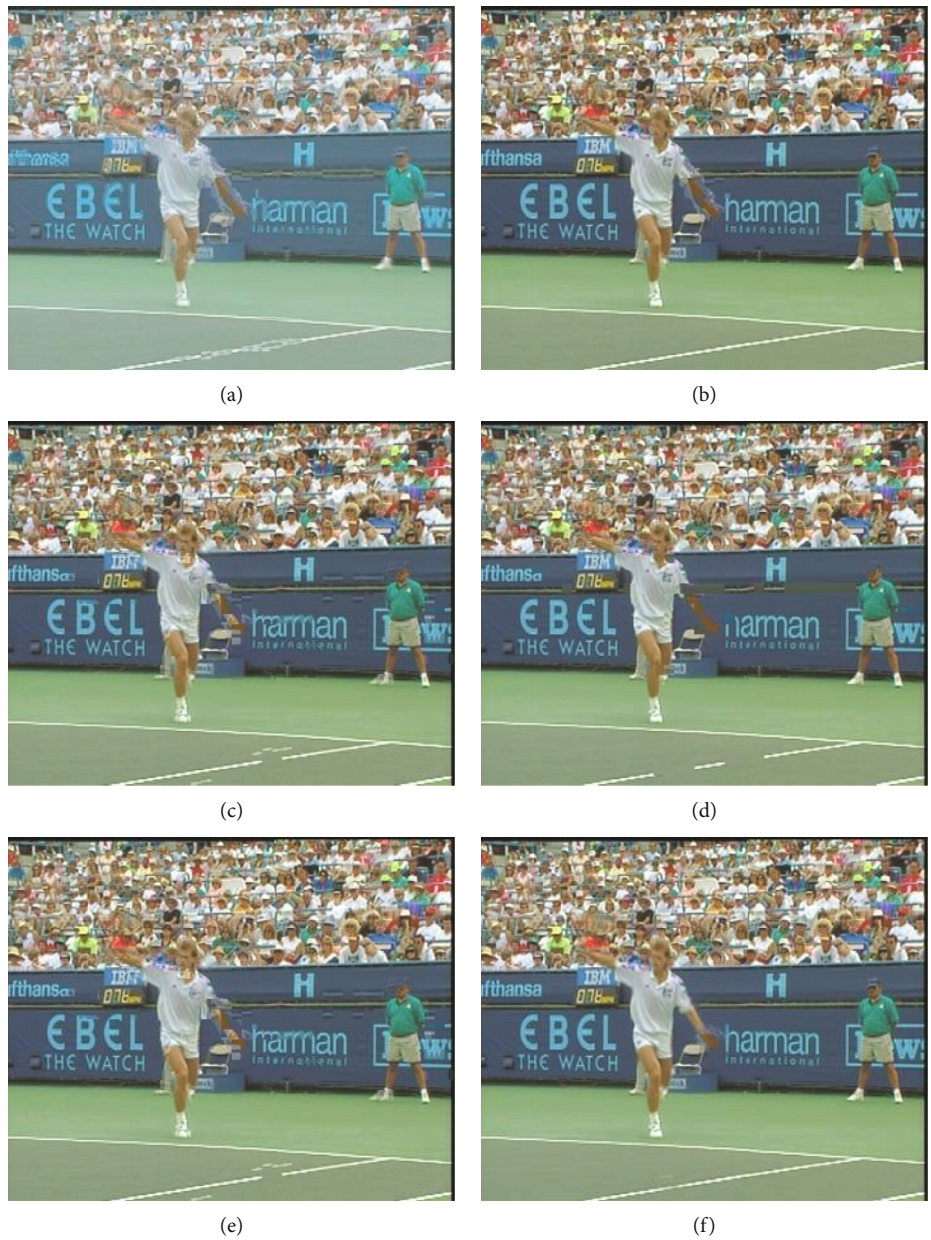


Figure 14: Visual results on the 30th interpolated frame of the *Stefan* sequence with different MVS algorithms: (a) MF; (b) VMF; (c) WVMF; (d) S-MVS; (e) T-MVS; (f) CA-MVS.

selects the candidate with the smallest SBAD value as the corrected MV by Equation (6). After scanning all outliers in $\widehat{\mathbf{E}}_t^{(j)}$, $\mathbf{V}_t^{(j)}$ is updated as $\mathbf{V}_t^{(j+1)}$. Due to a small radius of the search window, the number of candidates in $\mathbf{C}_S$ is limited, but these candidates have a strong spatial-temporal coherence with $\mathbf{v}_0$, and in that way, a good correction capability can be ensured.

4. Experimental Results

In this section, the performance of the proposed CA-MVS algorithm is evaluated by testing it on different video sequences and comparing the results with those obtained by the traditional MVS algorithms including MF [14], VMF [15], WVMF [16], S-MVS [19], and T-MVS [20]. An MCFI algorithm is also combined by BME, CA-MVS, and OBMC and compared with the recent state-of-the-art MCFI algorithms [1, 3, 7, 8] from objective perspectives. All test sequences used for experiments are in the standard CIF (352×288) formats and 30 frame/s. To evaluate the quality of the interpolated frames, we remove the first 50 even frames of each test sequence, then use various MCFI algorithms to recover these even frames from the 51 first odd frame. In the proposed CA-MVS algorithm, the block size is set to be 8×8, and how to set the threshold Thr will be discussed in Subsection 4.1. The comparing algorithms keep their original parameter settings. All experiments are conducted on a Windows machine with an Intel Core i7 3.40 GHz CPU and a

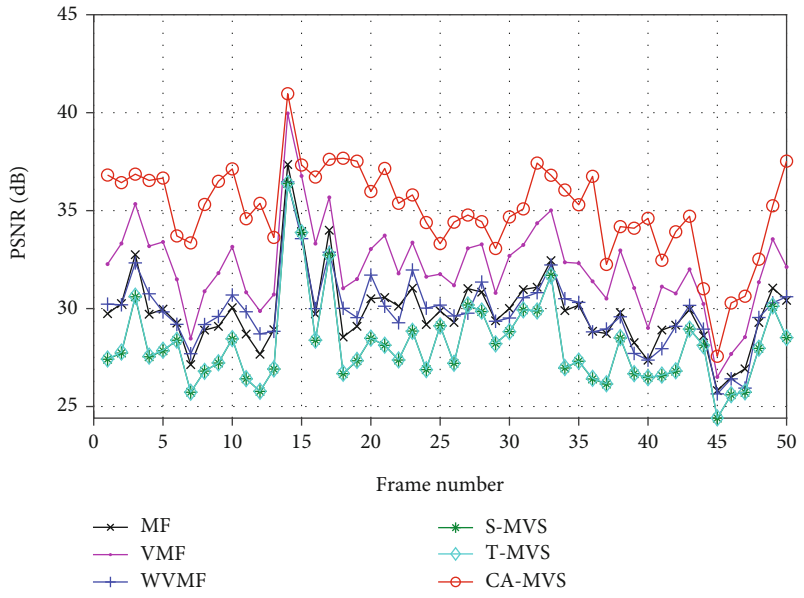
Table 2: Average PSNRs (dB) of the interpolated frames recovered by CA-MVS and traditional MVS algorithms.

Test sequence	MF [14]	VMF [15]	WVMF [16]	S-MVS [19]	T-MVS [20]	CA-MVS
Akiyo	40.15	46.09	39.16	37.32	37.34	46.83
Coastguard	26.99	30.77	28.50	27.46	27.52	32.52
Container	35.79	44.49	35.58	38.95	38.93	44.91
Flower	24.80	29.55	26.31	26.22	26.28	31.68
Football	21.35	21.83	21.20	21.42	21.32	22.86
Foreman	29.82	32.11	29.88	28.16	28.17	34.73
Hall	31.19	35.29	30.51	31.76	31.78	35.48
Mobile	19.06	22.54	20.45	19.20	19.35	26.56
Mother & daughter	37.21	41.56	36.48	38.33	38.34	42.53
News	32.34	36.35	31.20	31.78	31.84	37.20
Paris	27.88	31.95	28.63	28.19	28.20	33.52
Stefan	23.45	26.60	23.60	23.72	23.62	28.52
Tennis	25.62	28.17	26.38	26.52	26.43	30.05
Average	28.90	32.87	29.07	29.15	29.16	34.41

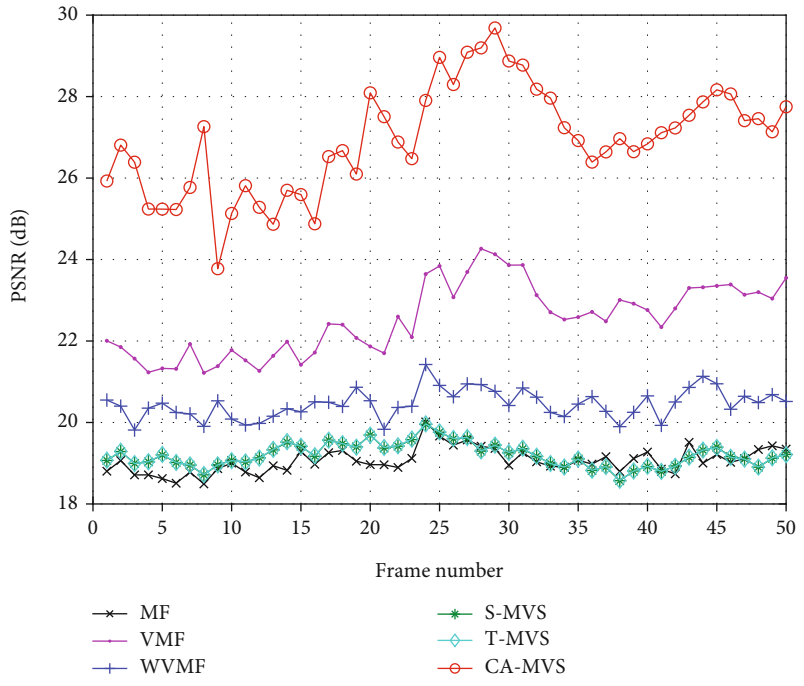
memory of 8 GB. All algorithms are implemented in MATLAB.

4.1. Effect of CA on Performance. We evaluate the effect of CA on the performance improvement by presenting the variations of interpolated frames and outlier maps in the loop iterations as shown in Figure 9. In the first iteration, there are many outliers in the outlier map after detection, then by CA-based evolution, many hidden outliers are exposed, and finally, we smooth these outliers and get the corresponding interpolated frame as shown in the left of Figure 9(a), from which it can be seen that obvious mismatches occur around the mouth region. In the second iteration, by outlier detection, the outliers are reduced significantly, but outliers around the mouth cannot be detected. Thanks to the CA-based evolution, these outliers hidden around the mouth are exposed again. The interpolated frame after the second iteration is shown in the second column of Figure 9(a), and we can see that the mismatches around the mouth are suppressed effectively. In the third iteration, after detecting, some outliers disappear as a result of the last spatial-temporal smoothing; then, the CA-based evolution propagates outliers around the detected outliers. After correcting these outliers, the interpolated frame is shown in the third column of Figure 9(a), and it can be seen that mismatches are removed. In the fourth iteration, the distribution of outliers tends to be stable, and the quality of the interpolated frame as shown in the right of Figure 9(a) is a little different from that after the third iteration. Figure 9 indicates that the bad effects from outliers in the interpolated frame are reduced step by step owing to the implementation of CA-based evolution. Figure 10(a) shows the varying PSNR values of the interpolated frame as the number of iteration times increases, and it can be seen that the PSNR value increases gradually at each iteration and tends to stabilize after the third one, indicating that CA-based evolution can effectively improve the interpolated quality. We cannot observe the quality degradation in both objective and subjective results due to oversmoothing, which presents that outlier propagation controlled by CA prevents oversmoothing. From Figure 10(b), we can find that the PSNR value has little change when residual value ε is smaller than 0.1 for the 82-rd frame of the *Foreman* sequence. Limited by space, we cannot present the PSNR and residual curves for other sequences, but these results are similar to Figure 10, i.e., the PSNR value tends to be stable when ε is smaller than 0.1. Therefore, in order to remove the invalid iterations, we set the threshold Thr to be 0.1.

4.2. Subjective Evaluation. Figure 11 shows the visual results of the 14th interpolated frame of the *Foreman* sequence with different MVS algorithms. MF and VMF can produce clear background, but serious blocking artifacts occur around the nose. For WVMF, S-MVS, and T-MVS, there are many mismatches on the background, and the faces are deformed severely. The proposed CA-MVS algorithm provides a pleasant result, in which there are no blocking artifacts. Figure 12 shows the visual results on the 58th interpolated frame of the *Mobile* sequence using different MVS algorithms. MF, VMF, and WVMF make the numbers on the calendar blurry and recover the red ball wrongly. S-MVS and T-MVS produce more serious distortion: numbers on the calendar disappear, and the color distortion occurs around the red ball. The CA-MVS algorithm produces clear numbers, and no deformation is generated. Figure 13 shows the visual results of the 22-rd interpolated frame of the *News* sequence with different MVS algorithms. WVMF, S-MVS, and T-MVS produce serious blocking effects around the face of anchorwoman. MF and VMF generate a clear face, but some mismatches occur on the background. The CA-MVS algorithm provides a comfortable result. Figure 14 shows the visual results of the 30th interpolated frame of the *Stefan* sequence with different MVS algorithms. For the traditional MVS algorithms, there are serious ghost effects around the sportsman; however, the CA-MVS algorithm effectively suppresses the blurring and provides a satisfying result.



(a)



(b)

Figure 15: Continued.

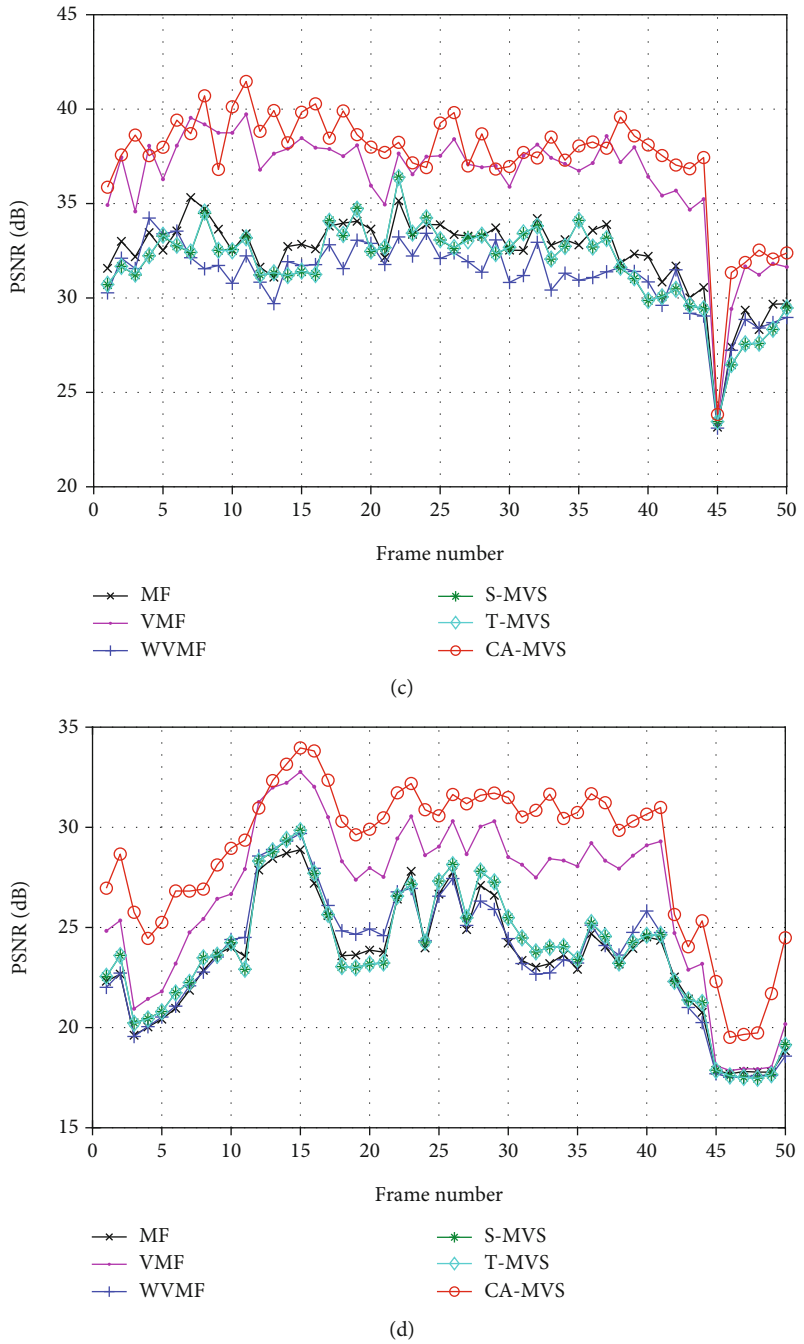


Figure 15: PSNRs of the interpolated frames constructed by different MVS algorithms: (a) *Foreman*; (b) *Mobile*; (c) *News*; (d) *Stefan*.

4.3. Objective Evaluation. The proposed CA-MVS and traditional MVS algorithms are, respectively, combined with BME and OBMC, and we get some different MCFI algorithms, which are used to interpolate the absent frames of test sequences. Table 2 presents the average PSNR values of these interpolated frames when using different MVS algorithms. It can be seen that the CA-MVS algorithm has obvious PSNR gains when compared with any of the other algorithms, e.g., for *Football* with complex motions, the CA-MVS algorithm is 1.03 dB higher than VMF, and for *News* with simple motions, the CA-MVS algorithm obtains, respectively, 6.00 dB, 5.42 dB, and 5.36 dB PSNR gains when compared with WVMF, S-MVS, and T-MVS. The last row of Table 2 lists the average PSNR values on all test sequences for various MVS algorithms, and it can be seen that the proposed CA-MVS algorithm obtains obvious PSNR improvements compared with other MVS algorithms. Figure 15 shows the PSNRs of individual interpolated frames on *Foreman*, *Mobile*, *News*, and *Stefan*. We can see that the proposed CA-MVS algorithm outperforms the traditional MVS algorithms in most cases, and especially for *Mobile* and *Stefan*, obvious PSNR improvements of the CA-MVS algorithm can also be obtained. Table 3 lists the average processing time of various MVS algorithms. It can be seen that the CA-MVS

Table 3: Average time (s/frame) comparison of CA-MVS and traditional MVS algorithms.

Method	Execution time (s/frame)
MF [14]	0.0464
VMF [15]	0.0548
WVMF [16]	0.3722
S-MVS [19]	3.1201
T-MVS [20]	3.1076
CA-MVS	0.3717

Table 4: Average PSNR (dB) comparison of the CA-MVS and some recent state-of-the-art MCFI algorithms.

Test sequence	[1]	[3]	[7]	[8]	CA-MVS
Coastguard	—	—	34.08	32.54	32.52
Container	—	43.64	42.27	44.71	44.91
Flower	31.27	29.78	31.76	32.22	31.68
Football	—	21.98	26.08	23.35	22.86
Foreman	33.63	—	32.86	34.96	34.73
Mobile	28.87	26.25	—	—	26.56
News	36.19	—	34.45	37.96	37.20
Stefan	29.38	—	25.46	28.91	28.52
Tennis	31.65	28.57	—	—	30.05

algorithm does not excessively introduce computations and only increases by about 0.3 s when compared with MF and VMF. The CA-MVS algorithm requires fewer computations than WVMF, S-MVS, and T-MVS. Table 4 presents the PSNR results of the CA-MVS algorithm and recent state-of-the-art MCFI algorithms [1, 3, 7, 8]. The results of [1, 3, 7, 8] are directly taken from the original researches. From Table 4, we can see that the CA-MVS algorithm obtains PSNR gains for some test sequences, e.g., for *Container*, the CA-MVS algorithm outperforms [3, 7, 8], and for *Football*, the CA-MVS algorithm obtains 0.88 dB PSNR gains over [3]. In most cases, the proposed CA-MVS algorithm has comparable PSNR results to those of the state-of-the-art MCFI algorithms. From the above, we can see that the CA-MVS algorithm can provide a good objective interpolation quality.

5. Conclusion

In this paper, the CA-MVS algorithm is proposed to reduce the incorrect MVs resulting from BME. To overcome oversmoothing, according to the CA theory, we construct a cellular automaton to model the interaction between outliers and define a logical evolution rule to accurately expose outliers step by step. In the CA-MVS algorithm, a loop iteration is executed. First, by the angles between every two adjacent MVs, the evident outliers are detected. Second, through CA-based evolution, we find the hidden outliers based on the information from the exposed outliers. Third, the spatial-temporal smoothing corrects each MV outlier by searching the reliable MV from the spatial-temporal neighboring MVs. Finally, we calculate the residual between the current and the previous MVFs to decide whether to make the loop exit. Experimental results show that, compared with the traditional algorithms, the CA-MVS algorithm can better improve the accuracy of BME and provide better subjective and objective interpolation qualities.

As the research in this paper is exploratory, there are many intriguing questions that future work should consider. First, the outlier detection is required to make it more accurate. Second, we will further improve the CA evolution rule to more efficiently find hidden outliers. Finally, the color information will be considered to be mixed into the CA evolution.

Data Availability

The experimental codes have been downloaded from Ran Li's homepage: http://www.scholat.com/liran358.

Conflicts of Interest

The authors declare no conflict of interest.

Acknowledgments

This work was funded in part by the Project of Science and Technology Department of Henan Province in China, under Grant no. 212102210106, in part by the National Natural Science Foundation of China, under Grant no. 31872704, in part by the Innovation Team Support Plan of University Science and Technology of Henan Province in China, under Grant no. 19IRTSTHN014, and in part by the Guangxi Key Laboratory of Wireless Wideband Communication and Signal Processing and China Ministry of Education Key Laboratory of Cognitive Radio and Information Processing.

References

[1] Q. Lu, N. Xu, and X. Fang, "Motion-compensated frame interpolation with multiframe-based occlusion handling," *Journal of Display Technology*, vol. 12, no. 1, pp. 45–54, 2016.

[2] X. Ding, G. Yang, R. Li, L. Zhang, Y. Li, and X. Sun, "Identification of motion-compensated frame rate up-conversion based on residual signals," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 7, pp. 1497–1512, 2018.

[3] X. Hoang van, "Statistical search range adaptation solution for effective frame rate up-conversion," *IET Image Processing*, vol. 12, no. 1, pp. 113–120, 2018.

[4] Y. Dar and A. M. Bruckstein, "Motion-compensated coding and frame-rate up-conversion: models and analysis," *IEEE Transactions on Image Processing*, vol. 24, no. 7, pp. 1–2066, 2015.

[5] F. Zhou, S. B. Kang, and M. F. Cohen, "Time-mapping using space-time saliency," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 23–28, Columbus, OH, USA, June 2014.

[6] S. H. Chan, T. X. Wu, and T. Q. Nguyen, "Comparison of two frame rate conversion schemes for reducing LCD motion blurs," *IEEE Signal Processing Letters*, vol. 17, no. 9, pp. 783–786, 2010.

[7] W. H. Lee, K. Choi, and J. B. Ra, "Frame rate up conversion based on variational image fusion," *IEEE Transactions on Image Processing*, vol. 23, no. 1, pp. 399–412, 2014.

[8] H. R. Kaviani and S. Shirani, "Frame rate up-conversion using optical flow and patch-based reconstruction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1581–1594, 2016.

[9] K. Bhattacharjee, S. Kumar, H. M. Pandey, M. Pant, D. Windridge, and A. Chaudhary, "An improved block matching algorithm for motion estimation in video sequences and application in robotics," *Computers & Electrical Engineering*, vol. 68, pp. 92–106, 2018.

[10] H. B. Yin, X. Z. Fang, H. Yang, S. Y. Yu, and X. K. Yang, "Motion vector smoothing for true motion estimation," in *2006 IEEE International Conference on Acoustics Speed and Signal Processing Proceedings*, Toulouse, France, May 2006.

[11] D. Y. Kim, H. Lim, and H. W. Park, "Iterative true motion estimation for motion-compensated frame interpolation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 3, pp. 445–454, 2013.

[12] S.-C. Tai, Y.-R. Chen, Z.-B. Huang, and C.-C. Wang, "A multipass true motion estimation scheme with motion vector propagation for frame rate up-conversion applications," *Journal of Display Technology*, vol. 4, no. 2, pp. 188–197, 2008.

[13] W. Xu, H. Yu, D. Lu, F. Liu, and Z. Liu, "A novel data reuse method to reduce demand on memory bandwidth and power consumption for true motion estimation," *IEEE Access*, vol. 6, pp. 10151–10159, 2018.

[14] S.-J. Kang, D.-G. Yoo, S.-K. Lee, and Y. H. Kim, "Design and implementation of median filter based adaptive motion vector smoothing for motion compensated frame rate up-conversion," in *2009 IEEE 13th International Symposium on Consumer Electronics*, Kyoto, Japan, May 2009.

[15] M. Barni, V. Cappellini, and A. Mecocci, "Fast vector median filter based on Euclidean norm approximation," *IEEE Signal Processing Letters*, vol. 1, no. 6, pp. 92–94, 1994.

[16] L. Alparone, M. Barni, F. Bartolini, and V. Cappellini, "Adaptively weighted vector-median filters for motion-fields smoothing," in *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, pp. 9–9, Atlanta, GA, USA, 1996.

[17] S. J. Yoon, H. H. Kim, and M. Kim, "Hierarchical extended bilateral motion estimation-based frame rate up-conversion using learning-based linear mapping," *IEEE Transactions on Image Processing*, vol. 27, no. 12, pp. 5918–5932, 2018.

[18] U. S. Kim and M. H. Sunwoo, "New frame rate up-conversion algorithms with low computational complexity," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 3, pp. 384–393, 2014.

[19] Y. L. Huang, F. C. Chen, and S. Y. Chien, "Algorithm and architecture design of multirate frame rate up-conversion for ultra-HD LCD systems," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 12, pp. 2739–2752, 2017.

[20] D.-G. Yoo, S.-J. Kang, and Y. H. Kim, "Direction-select motion estimation for motion-compensated frame rate up-conversion," *Journal of Display Technology*, vol. 9, no. 10, pp. 840–850, 2013.

[21] J. V. A. W. Neumann, *Theory of Self-Reproducing Automata*, University of Illinois Press, 1966.

[22] R. Wainwright, *Conway's Game of Life: Early Personal Recollections*, Game of Life Cellular Automata, 2010.

[23] S. Wolfram, "Statistical mechanics of cellular automata," *Reviews of Modern Physics*, vol. 55, no. 3, pp. 601–644, 1983.

[24] S. Wolfram, "Universality and complexity in cellular automata," *Physica D: Nonlinear Phenomena*, vol. 10, no. 1-2, pp. 1–35, 1984.

[25] S. Wolfram, "Cellular automata as models of complexity," *Nature*, vol. 311, no. 5985, pp. 419–424, 1984.

[26] U. Frisch, B. Hasslacher, and Y. Pomeau, "Lattice-gas automata for the Navier-Stokes equation," *Physical Review Letters*, vol. 56, no. 14, pp. 1505–1508, 1986.

[27] B. Priego, D. Souto, F. Bellas, and R. J. Duro, "Hyperspectral image segmentation through evolved cellular automata," *Pattern Recognition Letters*, vol. 34, no. 14, pp. 1648–1658, 2013.

[28] Y. Zhao, H. M. Guo, and S. A. Billings, "Application of totalistic cellular automata for noise filtering in image processing," *Journal of Cellular Automata*, vol. 7, no. 3, pp. 207–221, 2012.

[29] M. T. Orchard and G. J. Sullivan, "Overlapped block motion compensation: an estimation-theoretic approach," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 693–699, 1994.