

Chapter 9

Digital Video Stabilization in Static and Dynamic Scenes

Margarita N. Favorskaya, Lakhmi C. Jain
and Vladimir Buryachenko

Abstract The digital video stabilization is oriented on the removal of unintentional motions from video sequences caused by camera vibrations under external conditions, motion of robots stabilized platforms in a rugged landscape, a sea, oceans, or jitters during a non-professional hand-held shooting. The approaches for digital video stabilization in static and dynamic scenes are similar. However, objectively the analysis of dynamic scenes is needed in advanced intelligent methods. Several sequential stages include the choice of the key frames, the local and global motion estimations, the jitters compensation algorithm, the inpainting of frames boundaries, and the blurred frames restoration, for which the novel methods and algorithms were developed. The proposed application of fuzzy logic operators improves the separation results between the unwanted motion and the real motion of rigid objects. The corrective algorithm compensates the unwanted motion in frames; thereby the scene is aligned. The quality of stabilization in test video sequences was estimated by Peak Signal to Noise Ratio (PSNR) and Interframe Transformation Fidelity (ITF) metrics. During experiments, the PSNR and ITF estimations were received for six video sequences received from the static camera and eight video sequences received from the moving camera. The ITF estimations increase up on 3–4 dB or 15–20 % relative to the original video sequences.

Keywords Video stabilization · Video sequence · Motion estimation · Robust detectors · Fuzzy logic · Motion inpainting · Frame deblurring

M.N. Favorskaya (✉) · V. Buryachenko
Institute of Informatics and Telecommunications, Siberian State Aerospace University,
31 Krasnoyarsky Rabochy, Krasnoyarsk 660014, Russian Federation
e-mail: favorskaya@sibsau.ru

V. Buryachenko
e-mail: buryachenko@sibsau.ru

L.C. Jain
Faculty of Education, Science, Technology and Mathematics, University of Canberra,
Canberra, ACT 2601, Australia
e-mail: lakhmi.jain@unisa.edu.au

9.1 Introduction

The unintentional video camera motions are usual artifacts in non-professional hand-held shooting, surveillance tasks, or shooting by cameras, which are maintained on the mobile moving platforms in outdoor environment. As a result, the processing of non-stabilized original video sequence by the well-known conventional filters will not provide good segmentation, recognition, and surveillance of moving objects in static and dynamic scenes [1]. Also such technique is wide applied in video encoding tasks [2]. In this chapter, the novel methods of Digital Video Stabilization (DVS) for video surveillance task are developed for the static and dynamic scenes. The DVS algorithms as pseudo real-time and unreal-time applications are represented, depending from criteria of accuracy/computer speed. The novelty consists in the application of Takagi-Sugeno-Kang (TSK) model for improvement the motion vectors clustering, the decision procedure of key frames choice, and the reconstruction procedure of frame boundaries in static scenes and texture tiles from neighbor frames in dynamic scenes. Some often cases of objects surveillance are studied.

All variety of methods for videos stabilization techniques can be classified as mechanical, optical, electronic, and digital approaches. Historically, the mechanical stabilization based on a feedback from vibration sensors (gyros, accelerometers, etc.) were the first applied in video cameras [3]. Various control techniques are used for the stabilized platforms. Conventional design methods, modern synthesis tools such as linear quadratic regulator or linear quadratic Gaussian with loop transfer recovery, and fuzzy control systems can be used for these purposes [4]. The scope of such devices is wide but sometimes it is required the additional video stabilization, when a magnitude of vibrations has large values.

The optical image stabilization manipulates the images before their getting to the Charge-Coupled Device (CCD). The optical devices use prisms or lens of a moving assembly for tuning of light length way through camera lens systems. Vibrations occur the shifting of the lens group on a plane perpendicular to the optical axis in both horizontal and vertical directions. Two vibration-detecting sensors are used to detect the angle and speed of movement [3]. Usually for these purposes, the additional knowledge about physical motion of camera is required. Also the optical stabilization is not suitable for small sizes mobile cameras. Thus, the DVS became the most appropriate decision in modern compact video devices [5].

The electronic stabilization systems detect the camera jitters through their sensors, when the light hits the CCD. This responds by a slightly moving, and the image remains in the same position on the CCD. Such effect decreases a video quality because the pseudo-stabilized CCD area becomes smaller. Therefore, a digital zooming or oversized CCD is required. Electronic stabilization has the advantage against the optical stabilization by reducing of lens complexity and price.

The DVS approach is achieved by the synthesis of new imagery based on removal of unintentional motions between key frames and the reconstruction of frame boundaries after frame stabilization. The complexity of this task connects with a

separation the motion of objects from the unwanted camera jitters. In the case of static scenes, the proposed stabilization method does not consider specialties of camera motion trajectory. Such interpretation permits to edit static scenes faster and exactly and allows the pseudo real-time applications. However, such approach failures under the fast or changeable motions, which are often occurred in the dynamic scenes. In this case, it is needed to choice the criterion – accuracy or computer speed. According to the chosen criterion, some novel procedures are developed. In the pseudo real-time applications, the last 25–30 frames are analyzed. In the unreal-time applications, a whole video sequence is acquired, and then new frames are generated to compensate the warping between two successive original frames [6]. The most of stabilization techniques have a deal with the rigid objects. The non-rigid objects possess such specialties, which do not permit to generalize their processing.

The DVS algorithms ought to be robust to cluttered scene background, moving objects, and lighting change. One of such fast algorithms uses the feature-histogram building [7]. Various 2D and 3D stabilization algorithms are presented in [8–10]. A mosaic-based registration technique was described by Hansen in the pioneer research [11]. This system was based on a multi-resolution iterative procedure that estimated the affine motion parameters between levels of image Laplacian pyramid. The optical flow of local patches was computed by using a cross-correlation scheme.

The chapter is organized as follows. In Sect. 9.2, the problem statement of DVS for static and dynamic scenes is discussed. The description of the existing approaches for DVS is provided by Sect. 9.3. The main novel methods and algorithms suitable for static and dynamic scenes stabilization are detailed in Sects. 9.4 and 9.5, respectively. Section 9.6 presents a discussion of experimental results, involving experiments with stationary and moving video cameras. Conclusion and future development remarks are given in Sect. 9.7.

9.2 Problem Statement

The processing of video sequence occurs in the spatio-temporal domain. The DVS task is not the exclusion. Let an original video sequence $VS_{or}(FR)|_z$, where FR is a frame, z is a common number of frames, be a non-stabilized video sequence. Its transformation to the stabilized video sequence $VS_{st}(FR)|_z$ includes the sequential sub-transformations. For static scenes, they are represented by Eq. 9.1, where the operator O_{sf} selects a current set of frames $FS_{FR_t \dots FR_{t+n}}$ for current processing, $t \in \{0, 1, 2, \dots, z - 1\}$ is a number of frame, n is a number of selected frames, the operator O_{me} estimates an unwanted motion in a scene, the operator O_{mc} compensates an unwanted motion, the operator O_{mi} scales an area of stabilized frames.

$$VS_{or}(FR)|_z \xrightarrow{O_{sf}} FS_{FR_t \dots FR_{t+n}} \xrightarrow{O_{me}} FS_{FR_t \dots FR_{t+n}} \xrightarrow{O_{mc}} FS_{FR_t \dots FR_{t+n}} \xrightarrow{O_{mi}} VS_{st}(FR)|_z \quad (9.1)$$

As a result, the stabilized video sequence $VS_{st}(FR)|_z$ with the same number of frames will be created. Such fractional processing of frames is actual procedure for pseudo real-time applications. For non-real time applications in static scene, a finite set of frames $CS_{FR_0 \dots FR_{z-1}}$ may be used.

For dynamic scenes, Eq. 9.1 is extended by the additional operator O_{ss} , which divides an original video sequence in the relatively static scenes with non-essential changing for current video processing. Equation 9.2 provides the transformations for dynamic scenes, where SS_i is a set of scenes, $i \in \{0, 1, \dots, m-1\}$ is a number of scene, m is a total amount of scenes.

$$\begin{aligned} VS_{or}(FR)|_z &\xrightarrow{O_{ss}} SS_{FR_t \dots FR_{t+n}} \xrightarrow{O_{sf}} \\ FS_{FR_t \dots FR_{t+n}} &\xrightarrow{O_{me}} FS_{FR_t \dots FR_{t+n}} \xrightarrow{O_{mc}} FS_{FR_t \dots FR_{t+n}} \xrightarrow{O_{mi}} VS_{st}(FR)|_z \end{aligned} \quad (9.2)$$

Sometimes the blurred frames or frames with strong jutties appear in a video sequence. Their interpolation is executed by one of known methods, if the first and the last frames of the current scene have the appropriate quality for the DVS; otherwise the frames with high jutties are replaced by the interpolated frames. In this case Eq. 9.2 is replaced by Eq. 9.3, where O_{fi} is the operator of frames interpolation. Each of Eqs. 9.1–9.3 corresponds to the special task of computer vision.

$$\begin{aligned} VS_{or}(FR)|_z &\xrightarrow{O_{ss}} SS_{FR_t \dots FR_{t+n}} \xrightarrow{O_{sf}} \\ FS_{FR_t \dots FR_{t+n}} &\xrightarrow{O_{me}} FS_{FR_t \dots FR_{t+n}} \xrightarrow{O_{fi}} VS_{st}(FR)|_z \end{aligned} \quad (9.3)$$

9.3 Related Work

The DVS methods smooth and compensate the undesired motion of video camera and then restore frames by algorithms of digital video processing. Usually the DVS task is divided in three sub-tasks: a motion estimation, a motion compensation, and a motion inpainting. The motion estimation is a crucial aspect of video stabilization. A great variability of motion estimation methods, which were actively developed during last years, can be classified in two main categories [12]: the comparative and the gradient methods shown in Table 9.1. The dynamic textures are the special class of objects, usually background, which are characterized by an alternate motion (a motion of growth or water under a wind). In the DVS task, it is not required to estimate such motion, only to detect and ignore. The majority objects of interest are the rigid objects, and the main motion estimation methods are developed for their segmentation [13]. The non-rigid objects have a gaseous or liquid structure and do not require the exact estimations of their motion in a scene for the DVS [14]. Let us

Table 9.1 Classification of motion estimation methods

Groups of objects	Comparative methods	Gradient methods
Dynamic textures		Method of spatio-temporal fractal analysis Analysis based on autoregression functions
Rigid objects	Background subtraction* Block-matching algorithm* Density motion functions* Motion patterns	Edge points tracking Feature points tracking Optical flow Kurtosis estimations
Non-rigid objects	Background subtraction* Block-matching algorithm*	Feature points tracking Optical flow

High speed and less accurate methods are labeled by symbol ‘*’

notice that all motion estimation methods are calculated in the spatio-temporal domain of the original video sequences.

The efficient algorithm of statistical DVS was presented by Shakoor and Moattari [15]. The algorithm reduces the computational cost of Block-Matching Algorithm (BMA) by using a mean and a variance of pixels in each analyzed block with predetermined sizes (usually 16×16 pixels). According to such approach, the best block should not have a uniform area and should belong to a background without moving objects.

The basic BMA is suitable for motion analysis of rigid objects under the assumption that shift, rotation, and scale changing between frames are non-significant and can be neglected. First, a previous frame is divided on non-crossed blocks with similar sizes, which are defined by an intensity function $I_t(x, y)$, where (x, y) are coordinates, t is a discrete time instant. Second, for each block in the small neighborhood $-S_x < d_x < +S_x$ and $-S_y < d_y < +S_y$, the most similar block in a current frame $I_{t+1}(x + d_x, y + d_y)$ is searched, which is also divided on the non-crossed blocks with the same sizes as a previous frame. The similarity is determined by a minimization of the error functional $e_{(\cdot)}$ according to the applied metric. Usually three metrics are used such as Sum of Absolute Differences (SAD), Sum of Squared Differences (SSD), and Mean of Squared Differences (MSD) (Eq. 9.4), where n is a number of analyzed surrounding blocks.

$$\begin{aligned}
 e_{SAD}(d_x, d_y) &= \sum_{x=1}^N \sum_{y=1}^N |I_{t+1}(x, y) - I_t(x + d_x, y + d_y)| \\
 e_{SSD}(d_x, d_y) &= \sum_{x=1}^N \sum_{y=1}^N (I_{t+1}(x, y) - I_t(x + d_x, y + d_y))^2 \\
 e_{MSD}(d_x, d_y) &= \frac{1}{n \times n} \sum_{x=1}^N \sum_{y=1}^N (I_{t+1}(x, y) - I_t(x + d_x, y + d_y))^2
 \end{aligned}
 \tag{9.4}$$

To reduce the unnecessary computation, the partial distortion elimination was introduced in SAD-metric (pSAD) by Shakoor and Moattari [15]. Therefore, $e_{SAD}(d_x, d_y)$ is transformed in Eq. 9.5, where $k = 1, 2, \dots, N$, k is a k th partial SAD.

$$e_{pSAD}(d_x, d_y) = \sum_{x=1}^k \sum_{y=1}^N |I_{t+1}(x, y) - I_t(x + d_x, y + d_y)| \quad (9.5)$$

If an intermediate sum in k rows is larger than the minimum value of matching distortion, then the following computation is unnecessary. When the best block from four current estimated blocks is selected, a Local Motion Vector (LMV) is calculated only for the best block by use a full search strategy. This algorithm gives the main attention to search a stabilized macro-block in a frame, for which the LMVs and a Global Motion Vector (GMV) are calculated.

Methods from Table 9.1 include the steps for the LMVs and the GMVs definition. The GMVs are the basic for scene correction. For the GMVs estimations, a Speeded-Up Robust Features (SURF) tracking and a discrete Kalman filter were proposed in the researches [16, 17]. The SURF algorithm is used to obtain the stable feature points in the neighbor frames for global motion estimation by six parameters in 2D affine camera model [18]. The matching of SURF descriptors was done by the nearest neighbor distance ratio method. The Kalman filter estimated the process state at some time and then obtained the feedback measurements. The Kalman filter smoothed the estimated accumulated affine transformations by removing the high frequency components. During the motion compensation stage, the difference between the smoothed parameters and the estimated ones permitted to reconstruct the stabilized video sequence.

To account the temporal lighting variations, the generalized optical flow constraint under the non-uniform lighting change was used in the research [19]. Instead of using the traditional optical flow, the generalized optical flow constraint was applies in a local window of frame. The authors represented their performance of optical flow by Eq. 9.6, where $I_{t-1}(x, y)$ and $I_t(x, y)$ are intensity functions in a previous frame ($t - 1$) and a current frame t , respectively, u and v are velocity vector components along axes OX and OY, respectively, w is a constant for compensating the non-uniform lighting change.

$$\frac{\partial I_{t-1}(x, y)}{\partial x} u + \frac{\partial I_{t-1}(x, y)}{\partial y} v + I_{t-1}(x, y) \cdot w + I_{t-1}(x, y) - I_t(x, y) = 0 \quad (9.6)$$

Three parameters (u, v, w) are estimated by the iterative linear least squares method in a local window. The main idea of such iterative process is to move the block with the newly estimated motion vector and to compute the updated flow constraints in a recursive manner. Let us notice that the Lucas and Kanade optical flow computation method cannot provide the reliable motion estimation in the close homogeneous regions. Chang et al. [19] suggested the procedure for homogeneous regions detection by Eq. 9.7, where T is a user-defined threshold.

$$\sum_{x,y \in W_{ij}} \left(\left| \frac{\partial I_{t-1}(x,y)}{\partial x} \right| + \left| \frac{\partial I_{t-1}(x,y)}{\partial y} \right| \right) < T \quad (9.7)$$

Such homogeneous regions are skipped. Thereby, the approach can be called the sparse optical flow vectors estimation. The camera motion parameters are smoothed temporally to reduce the motion fluctuations by using a regularization method, which is considered the cost function for the penalty of data deviations and the cost function corresponding to the temporal motion smoothness constraint.

One of interesting approaches for fast and accurate global motion estimation is based on the Levenburg–Marquardt Algorithm (LMA) and several sub-sampling patterns with their combinations [20]. The LMA is a method based on the gradient descent for iteratively estimation of parameters of perspective model. The LMA is a highly expensive computationally method but it was substantially accelerated by using the sub-set selection methods based on the pixel-sub-sampling patterns. Partly the pixel sub-sampling patterns are similar to the BMA applied to the gradient frames.

One of the most popular approaches for motion compensation is based on the assumption that the GMVs have a high-frequency component, and the application of the low-pass filtering will free the original video sequence from the unwanted motion. In this case, the application of a first-order infinite impulse response of a low-pass filter integrates a differential motion in a scene and smoothes the global movement trajectory. Also a smoothing algorithm based on the smoothing absolute frame positions provides a successful stabilization performance [21, 22]. To other decisions, it may be concerned the application of discrete Fourier transform, Kalman filter, fuzzy systems, and fuzzy Kalman systems [23, 24]. Some adaptive filters with a smoothing factor and the adaptive procedures were proposed to remove the camera jitters [25, 26].

In the research of Puglisi and Battiato [27], the fast and the accurate block-based local motion estimator based only a translational motion together with a robust alignment algorithm using a voting are proposed. The collected information from the different spatial locations in a frame is applied to compute the GMVs through a voting strategy. The GMVs are related to a similarity motion model: two translations, one zoom factor, and one rotation. An integral projection-based error function is used in a search strategy. Instead of usual intensity function, the authors applied the gradient of integral projections that provides a high accuracy.

The fuzzy Kalman compensation of the GMVs in the log-polar plane was proposed by Kyriakoulis and Gasteratos [28]. Due to special features of the log-polar plane, each GMV was calculated as the average value of the four LMVs. Then the GMV displacements were imported in the fuzzy Kalman system. The fuzzy system was tested with several types of the Membership Functions (MFs), the different aggregation and the defuzzification methods.

For hand-held cameras and third-generation mobile phones, the unwanted motion is mainly caused by two independent motions: the camera motion (ego-motion) and the undesired hand jitter (high-frequency motion) [29]. The independent component

analysis searches the components that are both statistically independent and non-Gaussian. The authors assume that the estimated LMVs between the consecutive frames are the time-varying signals $x_j(t)$ and are a linear mixture of the ICs $s_i(t)$. The relative amplitude of each independent motion vector s_i at the estimated LMV is related to the selected frame region and can be defined as a weighting factor a_{ij} for each type of motion. The mixture $x_j, j = 2$ for two frame regions is represented by Eq. 9.8.

$$\begin{aligned}x_1 &= a_{11}s_1 + a_{12}s_2 \\x_2 &= a_{21}s_1 + a_{22}s_2\end{aligned}\tag{9.8}$$

Some original approaches can be found in the researches, dedicating to video stabilization by using a principal component analysis [30], an independent component analysis [31], a probabilistic global motion estimation based on Laplacian two-bit plane matching [32], wavelet transformations [33], a calculation of statistical functions, mean and variance of pixels in each block of the BMA [15], etc. An algorithm to estimate the global camera motion with Shift-Invariant Feature Transform (SIFT) features was proposed by Hu et al. [34]. These SIFT features have been proved to be affine invariant and used to remove the intentional camera motions.

Tanakian et al. [35] proposed the integrated system of the video stabilizer and the video encoder by using the BMA for the LMVs detection, the histogram analysis for the GMVs detection, and the low pass filtering for a Smooth Motion Vector (SMV) obtaining as the intentional motion correction. The authors suggested a low pass filtering to remove a high frequency component of intentional motion. They approximate the SMVs by the first-order auto-regression function (Eq. 9.9), where α is a smoothing factor, $0 \leq \alpha \leq 1$; n is a frame number.

$$|SMV_n| = \alpha|SMV_{n-1}| + (1 - \alpha)|GMV_n|\tag{9.9}$$

The authors proposed a rule to chose α value ($\alpha = 0.1$ or $\alpha = 0.95$) in dependence of the GMVs and the SMVs magnitudes in the previous frames. In their following research, a fuzzy system for tuning of smoothing factor α was suggested according to noise and camera motion acceleration [36]. The trapezoidal and triangular the MFs were used for adaptive filtering of horizontal and vertical motion components between $(n - 3)$, $(n - 2)$, $(n - 1)$, and n frames.

To product the full-frame stabilized video sequence with a good visibility is the final stage of the DVS. The direct pixel based on the full-frame video stabilization approach was proposed by Matsushita et al. [37]. This approach uses a technique of image mosaics by accumulating neighboring frames with natural stitching of multiple images and a motion deblurring method to reduce the motion blur caused by the original camera motion. The propagated motion field, based on a pyramidal version of the Lucas-Kanade optical flow computation, is used to help naturally fill up the missing image areas even for scene regions that are non-planar and dynamic. Also in order to sharpen the blurry frames by a novel interpolation-based deblurring

method was developed. The crucial idea is to transfer the blurry pixels in a current frame by the corresponding sharper pixels from neighbor frames. This method works well in most videos except the cases with a large area cover of moving object and impossibility of correct the GMVs detection.

The method of dual-tree complex wavelet transform for video stabilization was developed by Pang et al. [38]. This method considers the dependence between the phase changes of wavelet transform and shift invariant feature displacement in a spatial domain. The smoothness of motion jitters is achieved by the optimal Gaussian kernel filtering. This phase-based method is invariant to lighting changes, but has a high computational cost. Usually the technologies of image warping are used, which reduce an area of stabilized frame. The motion inpainting of frame boundaries (the reconstruction of frame boundaries) can be successfully applied for static scenes. However, this may be impossible in dynamic scenes, when the reconstruction data are absent in neighbor frames.

Liu et al. [8, 39] proposed a content-preserving warping based on two objectives: to displace all tracked feature coordinates to their regularized re-projected locations and, at the same time, to minimize the warping distortion in the content-rich regions with a minimum computational cost.

A novel method to stabilize video sequences based on a 3D perspective camera model without recovering the dense depth maps was proposed in the research of Zhang et al. [40]. By balancing the smoothness and similarity, a video stability was optimized related to rotation, zooming, and translation components with suitable weights. Based on a 3D perspective camera model, the depth relative motion (camera translation) and the depth irrelative motion (camera rotation and zooming) were separated. The corresponding SIFT features are constrained frame by frame according to the epipolar geometry theory with application of RANdom SAMple Consensus (RANSAC) algorithm.

The literature survey shows a great variability of existing stabilization methods. Often the authors solve the DVS task for a static scene and prefer to use more simple decisions in order to provide the pseudo real-time applications.

9.4 Video Stabilization in Complex Static Scenes

The DVS in static scenes is a particular case of the DVS in dynamic scenes. The both approaches have some common procedures and a similar logic of realization but the differences are also essential, especially on the stage of motion inpainting. For static scenes, it is required to find vectors of unwanted motion, which are enough uniform distributed in each original frame and have the similar magnitudes and directions.

The motion estimation of cluttered background is the task with a high computational cost. The main goal of the current research was not only to develop the novel algorithms for separation of motion vectors but also to design the fast algorithms in order to make the DVS in a static scene as a pseudo real-time

application. These issues are performed in Sect. 9.4.1. Motion compensation by the smoothing of global motion in a static scene and camera path estimation is situated in Sect. 9.4.2. Section 9.4.3 provides the static scene alignment with using two main techniques: the stabilized frame scaling with the reduced stabilized frame area and the frame borders restoration with the non-changeable frame sizes.

9.4.1 Motion Estimation in Static Scene

According to the problem statement representing in Sect. 9.2, the motion estimation in static scenes can be provided by fast comparative methods. The small displacements of objects in a scene between two sequential frames may be roughly performed as the parallel transitions from frame to frame. Usually a motion of objects in static scene satisfies the assumption that a motion is described by the almost continuous function. The proposed modification of the BMA based on a statistical model of background is concerned to fast realization of the basic BMA. Let us consider the enhanced statistical model of a background.

The enhanced statistical model of background in a static scene is based on the following parameters: an average of frames I_{med} (medium values), a mean value $\mu(x, y)$, and a variance $\sigma^2(x, y)$ for K frames, which are described by the intensity function $I_t(x, y)$, where x and y are coordinates of a current pixel, t is a number of frame at moment t . A mean value $\mu(x, y)$ and a variance $\sigma^2(x, y)$ for K frames are calculated by Eqs. 9.10–9.11, where $w_t(x, y)$ are the weighting coefficients.

$$\mu(x, y) = \left(\sum_{t=1}^K w_t(x, y) \cdot I_t(x, y) \right) / \left(\sum_{t=1}^K w_t(x, y) \right) \quad (9.10)$$

$$\sigma^2(x, y) = \frac{1}{K-1} \left(\sum_{t=1}^K (w_t(x, y))^2 \cdot (I_t(x, y) - \mu(x, y))^2 \right) / \left(\sum_{t=1}^K w_t(x, y) \right)^2 \quad (9.11)$$

Let us notice, that Eq. 9.11 gives a unbiased estimator of variance $\sigma^2(x, y)$ for small K value, $K < 30$.

The weighting coefficients are used for minimization of spikes, which are maximally removed from the average of frames I_{med} (a noisy compensation) and are normally distributed. Equation 9.12 provides the estimations for $w_t(x, y)$, where standard deviation σ_{ex} is calculated from K neighbor frames in a spatial domain. The use of weighting coefficients $w_t(x, y)$ in the statistical model of background permits to build a robust background model without any training video sequences.

$$w_t(x, y) = \frac{1}{\sqrt{2\pi\sigma_{ex}^2}} \exp\left(-\frac{(I_t(x, y) - I_{med}(x, y))^2}{2\sigma_{ex}^2}\right) \quad (9.12)$$

The statistical model of background (Eqs. 9.10–9.12) is recalculated periodically because of lighting or meteorological changes. The enhanced model of background separates the background and the foreground moving objects. If Eq. 9.13 is executed, then this pixel is a foreground pixel, and otherwise. Value λ is equaled to 3 according to the rule of three sigma: all values of normal distributed random variable lay in the interval $\pm 3\sigma$ with not less reliability than 99.7 %.

$$|\mu(x, y) - I(x, y)|^2 \geq \lambda^2 \sigma^2(x, y) \quad (9.13)$$

Such statistical model is especially effective, when a set of sequential frames without moving objects can be provided. This requirement is ordinary for the outdoor and the indoor surveillance tasks. The worse results are received, when the moving objects with a small area (less 5–8 % of frame area) appear in static scene. The most problematic case connects with the moving object with a large sizes, when only the probable regions with or without motion are determined.

The background subtraction and the BMA are the main comparative methods of fast motion estimation. The background subtraction is the simplest motion estimation technique. For each current frame, the intensity values and the color components of each pixel are compared with the corresponding values of pixels in an initial averaged (sampling) frame of video sequence. As a result, the binary masks of moving foreground objects in a scene will be received. Such method is a noise-dependent. Therefore, a median filter or the mathematical morphological operators are applied for binary masks improvement. The filter parameters determine the sensitivity and the reliability of background subtraction method. The simplicity and the high computational speed are its main advantages. However, shadows, dynamic background, lighting change, and camera inaccuracy make this approach the non-used in practice.

More appropriate decision connects with the BMA application for a set of sequential frames. Experiments show that 25 frames processing (near 1 s) is a good decision that provides a delay for receiving of stabilized video sequence in 2–4 s. In surveillance systems as the urban surveillance or computer vision in the outdoor environment, such results are satisfied. The hardware realization by CUDA technology will reduce the processing duration in times.

The proposed procedure for motion estimation includes three steps:

- The local motion estimations by fast BMA modification.
- The accuracy improvement by using the Takagi-Sugeno-Kang model.
- The global motion estimation in a frame.

Let us consider local motion estimation by the BMA. The basic BMA has various interpretations such as full search, pattern search, and recursive search, among others [12]. The full search strategy provides the best results with the highest

computational cost. There are known its modifications such as three step search, four step search, block based gradient descent search, diamond search algorithm, adaptive rood pattern search, etc., which were developed to reduce the computational cost with a non-essential quality loss. All these fast search algorithms are based on the assumption that a block distortion measure increases monotonically around the global minimum. However, the search process can be easily trapped in one of the local minimums, a lot of which are included in any video sequence as noises, lighting changes, or dynamic textures. To avoid such problem, the Markov model with three states was proposed by Chen et al. [41] to provide an acceptance probability of being able to jump out of a local minimum. Also many BMA modifications were developed for various cases of motion estimation, for example, a motion estimation in noisy video sequences [42], fast BMA [43], Gaussian mixture model [44], a motion estimation by using Lie operators [45], a bilinear deformable BMA [46], a fuzzy logic BMA [47], etc.

A motion vector $\mathbf{MV}(d_x, d_y)$, for which an error functional e according one of metrics (Eq. 9.4) has the minimum value, is considered as a displacement vector for the given block. It shows the displacement of the left top corner in the marked block from a previous frame ($t - 1$) to a current frame t . The proposed BMA modification uses the transparent masks for moving objects and the opaque mask for a static background, which often involves periodical motions of textons. Usually such motion is not interesting for estimation and ignored. In the case of static scene, all values of background pixels can be set to a constant value, for example, -1 . When a motion is detected, the transparent mask is put on a visual object. Therefore, the intensity function describing a moving object will be available for estimation especially during the overlapping of visual objects. Forcibly maintained to negative constant values, the opaque background masks permit to reduce BMA calculations due to only the analysis in a neighbor region. The procedure of basic BMA is reactivated periodically (with interval 1 s) as an additional search of other moving objects, appearing in static scene. As a result, a set of the LMVs fields will be built for the chosen frames; the LMVs field shows the motion vectors between two neighbor frames. Such LMVs field is enough chaotic, and the following procedure is to separate of the LVMs as “good” and as “bad” motion vectors.

The field of LMVs includes the motion vectors, which describe an unwanted camera motion and objects motion in a scene. For such clustering, a novel fast method for detection of unwanted camera motion was developed based on Takagi-Sugeno-Kang (TSK) model. A fuzzy zero-order TSK model is adopted to infer the quality index: four different output fuzzy sets are defined to describe the quality of the matching, named as excellent, good, medium, and bad [48]. A zero-order TSK model is very simple, compact, and computationally efficient model, which permits to use the adaptive techniques. These adaptive techniques customize the MFs in such manner that the input data are modeled by a fuzzy system in the best way. Also a quite complex data behavior can be interpreted by using the “IF-THEN” fuzzy rules. In our experiments, the triangle, trapezoidal, and S -shape MFs to partitioning the LMVs were used. A view of these MFs is represented in Fig. 9.1, parameters a and b of S -shape membership are fitted empirically. Our recommendations are to use

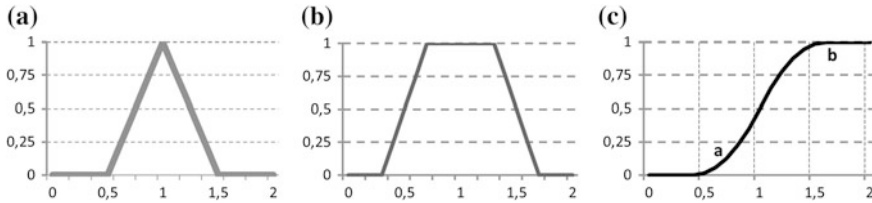


Fig. 9.1 A view of the MFs in the TSK model: **a** triangle, **b** trapezoidal, **c** S-shape

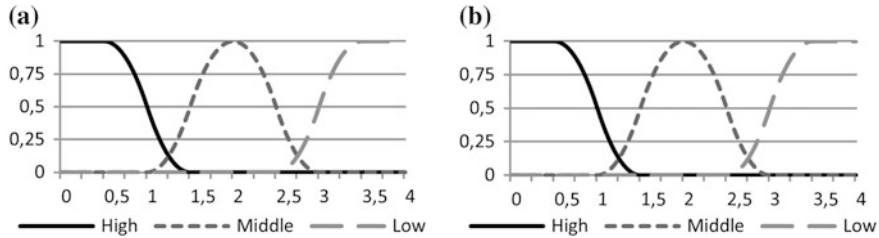


Fig. 9.2 A view of S-shape MFs: **a** for the non-noisy video sequence, **b** for the noisy video sequence

$a = 0.5$ and $b = 1.5$ for the non-noisy video sequence and $a = 0.75$ and $b = 1.75$ for the noisy video sequence. The recommended S-shape functions are situated in Fig. 9.2.

The inputs of fuzzy logic model are two error measures: an Euclidean distance e_i between expected and the real LMVs calculated in one of SAD, SSD, or MSD metrics (magnitude of vectors) $\mathbf{E}' = (e_1, e_2, \dots, e_i, \dots, e_n)$ and an angle between these LMVs c_i , $\mathbf{C}' = (c_1, c_2, \dots, c_i, \dots, c_n)$, where $i = 1 \dots n$. In this research, the similar approach from [49] to find error deviations d_i^e and d_i^c was used. The median values M_E and M_C of sets $\mathbf{E}' = (e_1, e_2, \dots, e_i, \dots, e_n)$ and $\mathbf{C}' = (c_1, c_2, \dots, c_i, \dots, c_n)$, $i = 1 \dots n$, respectively, are provided by Eq. 9.14.

$$d_i^e = e_i/M_E \quad d_i^c = c_i/M_C \tag{9.14}$$

Values of error deviations d_i^e and d_i^c from Eq. 9.14 are mapped in three different classes of accuracy: high, medium, and low. The lower values of error deviations are mapped to the best class, and otherwise. If the MFs are overlapped, then better class of the input fuzzy set is chosen.

The output of fuzzy logic model indicates a final reliability of estimation for a quality of the matching by using the TSK model. The quality index is a value in the range [0, 1]. It shows the quality of the LMVs, which are clustered in four classes: excellent, good, medium, and bad. The “IF–THEN” fuzzy rules defined for two inputs (error deviations d_i^e and d_i^c) are the following:

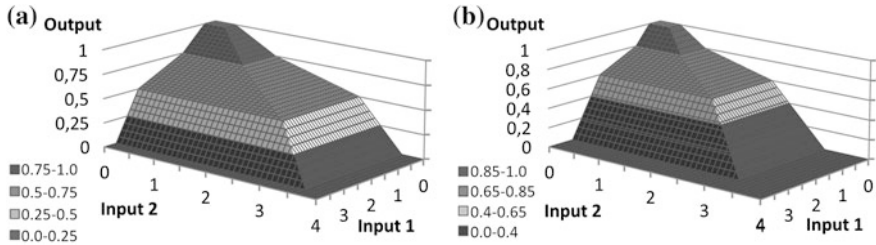


Fig. 9.3 The interpretation of the TSK model: **a** for the non-noisy video sequence, **b** for the noisy video sequence

- IF (both inputs = “high”) THEN (quality = “excellent”).
- IF ((one input = “high”) AND (other input = “medium”)) THEN (quality = “good”).
- IF (both inputs = “medium”) THEN (quality = “medium”).
- IF (at least one input = “low”) THEN (quality = “bad”).

Each of these four classes is mapped in a set of the constant values (1.0, 0.75, 0.5, 0.25, 0.0) [48]. During our experiments, the results for noisy video sequences were received with a set of the constant values (1.0, 0.85, 0.65, 0.4, 0.0). The TSK models for non-noisy and noisy video sequences with the sets of constant values (1.0, 0.75, 0.5, 0.25, 0.0) and (1.0, 0.85, 0.65, 0.4, 0.0), respectively, are show in Fig. 9.3. The TSK model permits to discriminate the LMVs with excellent and good quality and detect the best LMVs (with excellent and good values of indexes) in order to improve the final result.

Our following researches permitted to speed the LMVs calculation for both types of video sequences in the static scenes. Let us introduce the initial procedure, which will put an invisible grid on each frame adaptively to the frame sizes with 40–60 cells. The sizes of such grid are less than the frame sizes in order to reject the boundary areas of frame, which are more stressed to artifacts of instability. For five first frames in a scene, the LMVs estimations and their improvements by TSK model are calculated for all cells of this grid. For each cell, the information of reliable LMVs is accumulated under the condition, that 4–16 reliable LMVs are determined in a cell. According to the scene background, some of such cells are selected for the following analysis. Therefore, the LMVs of unwanted motion are calculated only in the selected cells, that permits to avoid the challenges of lighting change or moving foreground objects and reduce the number of analyzing cells in 1.5–3 times. Figure 9.4 provides such adaptive and fast technique for frame number 69 from video sequence “EllenPage_Juggling.avi”. Figure 9.5 illustrates several frames from the same video sequence with the imposed grid.

The selected cells include information only about unwanted motion that also increases an accuracy of the GMV in a frame. Let us consider the technique of the global motion estimation in static scene. The global motion caused by a camera movement is estimated for each frame by using a clustering model. On the one

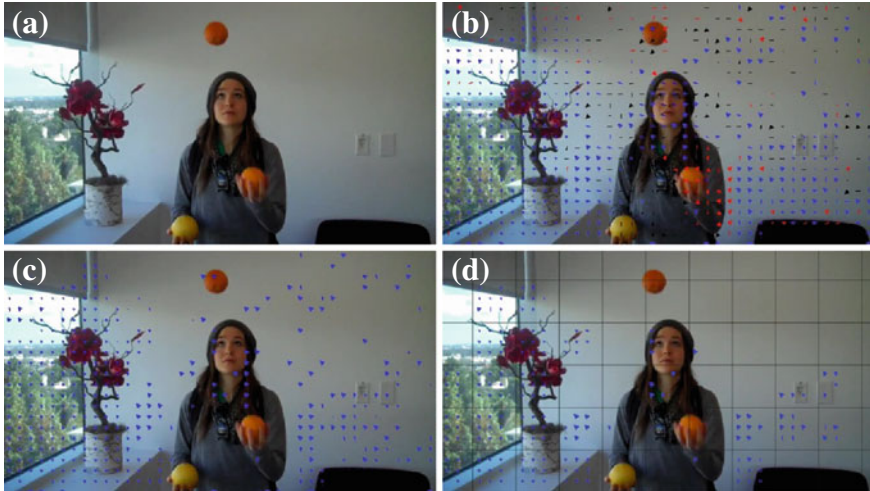


Fig. 9.4 The adaptive technique for LMVs estimation in static scene, video sequence “EllenPage_Juggling.avi”: **a** the initial frame 69; **b** all calculated LMVs; **c** the reliable LMVs based on the TSK model; **d** the reliable LMVs in the selected cells of imposed grid

hand, the LMVs of background are very similar in magnitudes and directions. On the other hand, they are essentially different from the motion vectors of foreground objects. The procedure, classifying the LMVs in two clusters – background and foreground, has the following steps:

- Step 1. The histogram H is built, which includes only valid LMVs with excellent and good values of indexes (near 30 % from all detected LMVs).
- Step 2. The LMVs are clustered by a criterion of the similar magnitudes.
- Step 3. The LMV with a maximum magnitude from background motion cluster is chosen as the GMV for a current frame.

The example of a histogram with valid LMVs is presented in Fig. 9.6.

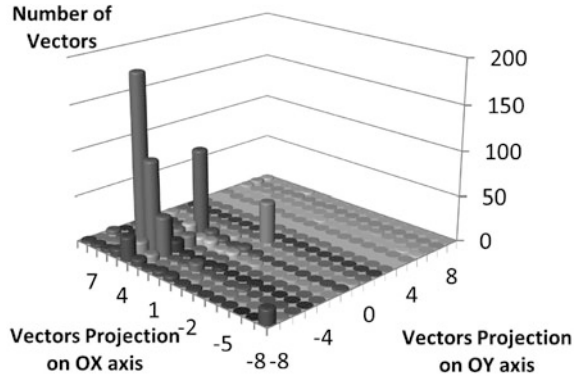
For the GMV detection, the TSK model can be also applied. The global motion includes two major components: a real motion (for example, a panning) and an unwanted motion caused by camera jitters. Usually an unwanted motion corresponds to a high frequency signal. Therefore, the low-frequency filtering can remove the unwanted motion.

The model, proposed in the research of Kyriakoulis and Gasteratos [28], was used to create a Smooth Motion Vector (SMV) calculated by Eq. (9.14). The low-pass filter of the first order requires the low computational cost and can be used in a real-time application.



Fig. 9.5 The LMVs estimation in static scene from video sequence “EllenPage_Juggling.avi”: all calculated LMVs are in the left column; the reliable LMVs based on the TSK model are in the middle column; the reliable LMVs in the selected cells of imposed grid are in the right column

Fig. 9.6 The example of a histogram with valid LMVs



9.4.2 Unwanted Motion Compensation

The unwanted motion compensation in static scenes is based on a smoothing of the GMVs. Any moving rigid object has the following states [50]:

- Appearance. The object is appearing in a scene.
- Mature. The object has been continuously tracked in some interval for approval that it is a foreground object, not a noise.
- Temporarily unavailable. The object temporarily loses its track because of being hidden, noised, or exited.
- Occlusion. The object is partially or totally hidden by other objects in a scene.
- Disappearance. The object may either already exit from a scene or be hidden by background objects such as buildings or trees.
- Reappearance. The object appears again after disappearing. The surveillance is restarted.
- Out of scene. The object has indeed moved away in a scene and its track considered terminated.

Only “appearance”, “mature”, “occlusion”, and “reappearance” are considered in the DVS task. For these purposes, the well-designed techniques are developed, including shadow compensation, lighting enhancement, and others challenges [51, 52].

The tuning procedure of a smoothing factor α , based on the analysis of previous 25 frames, was proposed. First, for k -sampling of frames, a global difference $GDiff_k$ is calculated by Eq. 9.15, where $|GMV_i|$ is a magnitude of global motion vector in frame I , $k > 25$.

$$GDiff_k = \sum_{i=k-25}^k \left| |GMV_i| - |GMV_{i-1}| \right| \tag{9.15}$$

Second, a value of α is chosen by Eq. 9.16, where $\alpha_{\max} = 0.95$ and $\alpha_{\min} = 0.5$ are maximum and minimum empirical values. In any case, the result from Eq. 9.16 is rounded to α_{\max} .

$$\alpha_k = \begin{cases} \frac{GDiff_k}{|GMV_{\max}|} \times \frac{\alpha_{\max}}{\alpha_{\min}} & \text{if } GDiff_k < |GMV_{\max}| \\ \alpha_{\max} & \text{if } GDiff_k \geq |GMV_{\max}| \end{cases} \quad (9.16)$$

The GMV in a frame is selected from the LMVs with maximum magnitude. It is easy may be done from a histogram with valid LMVs (Fig. 9.6). The total amount of LMVs with the similar magnitudes and directions are calculated. The LMV with the maximum value is considered as the GMV in a current frame.

9.4.3 Static Scene Alignment

In most algorithms for static scene stabilization, a motion inpainting is often considered as a scene alignment task [27]. At present time, two approaches of scene alignment are known. They are based on a frame scaling with reduction of frame sizes and a restoration of frame borders with conservation of frame sizes. For dynamic scenes, this is more complex task in comparison of static scenes.

For simple static scene, a scene alignment procedure can be simplified by a forcibly replacement of background from the statistical background model discussing in Sect. 9.4.1. In this case, only motion of moving objects ought to be compensated, and static scene alignment does not necessary. To make such “replaced” static scene more realistic, the procedures for rendering natural dynamic textures or other artifacts are required.

For each frame after calculation of smooth factor α , a module of smooth motion vector SMV_n is determined using Eq. 9.9. A magnitude of Undesirable Motion Vector (UMV) UMV_n is calculated by Eq. 9.17.

$$|UMV_n| = |GMV_n| - |SMV_n| \quad (9.17)$$

In the development of scene alignment method, a direction of vector SMV_n is normalized up to 8 directions with 45° step. For restoration of current frame, pixels are shifted on a value of Accumulated Motion Vector (AMV) AMV_n of unwanted motion by using Eq. 9.18. The stabilized location in a frame is determined from previous frames, beginning from a key frame, where m is a number of key frames in a video sequence.

$$AMV_n = \sum_{i=m}^n |UMV_i| \quad (9.18)$$

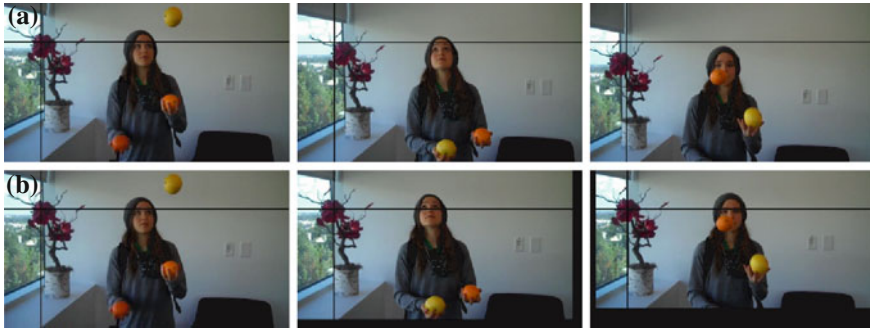


Fig. 9.7 The scaling resume for frames 540, 555, and 570 from video sequence “EllenPage_Juggling.avi”: **a** a non-stabilized video sequence, **b** the stabilized video sequence by changing of frame boundaries

The role of key frame is to represent the homogeneous regions in a following set of frames. The technique of key frame detection has been widely studied in [53, 54]. For static scenes, it is necessary to find the frames with the significant inter-frame difference. It means that the important motion changing occurs in a scene. Our recommendations connect with the limited power of such set, not more 25 frames. Then the following key frame ought to be chosen. The using of such technique permits to stabilize a video sequence with scaling. The sizes of such stabilized frames become less on 10–20 % relative the original video sequence that Fig. 9.7 demonstrates. Experiments show that the application of frame scaling for dynamic scenes may reduce the stabilized frame area up to 50–60 % that is not acceptable for the user.

All existing video inpainting algorithms can be broadly classified in two categories: Partial Differential Equation (PDE)-based methods and Texture Synthesis (TS)-based methods. The PDE-based image inpainting reconstructs the missing data spatially by extending the edges and filling the hole with smoothed color information by a diffusion process. In this case, a temporary nature of video sequence is ignored, and each frame is considered as an individual image. Such approach does not reproduce the texture and suffer from the blurred artifacts. This method is effective for restoring of small scratches or spots in archival footage.

The non-parametric sampling is an important class of TS-based methods, which uses the spatio-temporal patches extracted from the neighbor frames. The space-time patches called epitomes are created by a probabilistic learning of large number space-time patches taken from input video sequence [55].

The simple technique for static background frame borders restoration was applied in this research. Any static scene can be represented as two layers – foreground and background layers according to the affine model and as several layers – foreground and 2–3 background layers according to the perspective model of scene. The background layer (layers) is delivered by the statistical background model.

The foreground layer includes moving objects tracking by Kalman filter with following processing by the de-blocking filter with sizes 5×5 pixels to smooth the boundaries of the moving regions.

9.5 Video Stabilization Method in Dynamic Scenes

The analysis of dynamic scenes is the most complex issue in video processing. In literature, the researches devoting to DVS in dynamic scenes are sparsely represented. Before motion estimation, compensation, and inpainting, it is required to separate a video sequence on the relatively static scenes. The DVS task is solved for each separated scene, and then stabilized fragments ought to be collected in a whole stabilized video sequence. At this stage, the local smooth estimations of unwanted camera motion can be applied to smooth transitions between scenes. A valid scene transition does not require the additional processing. A sharp scene transition is needed in a future algorithm development or is executed by the user.

Section 9.5.1 provides a scene separation model. The recommended methods for motion estimation of background and moving objects are discussed in Sect. 9.5.2. The deblurring methods for the DVS task are proposed in Sect. 9.5.3. The unwanted motion compensation for dynamic scenes is represented in Sect. 9.5.4. At last, Sect. 9.5.5 includes the issues of motion inpainting in dynamic scenes by frames interpolation.

9.5.1 Scene Separation

For accurate DVS in dynamic scenes, a video sequence ought to be cut in separate relatively static scenes. Then the DVS task is partly added up to the DVS in static scenes discussed in Sect. 9.4, but with a dynamic background. The continuous outdoor shooting can be concerned to the complex cases, when the explicit cut of scenes is impossible, and the user's help is needed. All methods and algorithms of scenes cut can be divided in two categories. Methods from the first category use information from a service recording of video sequence, In other words, labels of beginning and ending of video scenes are located during a shooting by operator manually or by a video camera automatically in compliance with turn-on and turn-off modes of shooting [56].

Methods from the second category are based on the inner information of video sequence, and usually include two stages. On the first stage, frames histograms, configuration and number of feature points, color areas location, and other parameters are estimated [57]. On the second stage, the adaptive threshold values are determined, according to which a procedure of scenes cut is executed. The proposed algorithm, using the configuration of feature points, is realized cyclically from frame to frame and involves three steps:

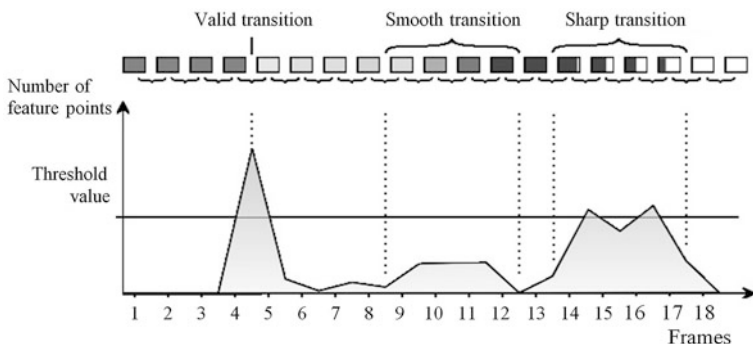


Fig. 9.8 Examples of scenes cut depending from a number of feature points, which have large displacement values

Step 1. The calculation of distance from each feature point to the center of frame j by Eq. 9.19, where x_{FP_i} , y_{FP_i} are coordinates of feature point i , x_{c_j} , y_{c_j} are coordinates of a center point of frame j .

$$D_{ij} = \sqrt{(x_{FP_{ij}} - x_{c_j})^2 + (y_{FP_{ij}} - y_{c_j})^2} \quad (9.19)$$

Step 2. The calculation of feature point displacement D_{ij} by using Eq. 9.20, where TH_j is a threshold for frame j for each feature point cyclically.

$$|D_{ij} - D_{ij-1}| < TH_j \quad (9.20)$$

Step 3. The calculation a number of feature points, which have large shifts in frame j by Eq. 9.21, where TH_{av} is an average threshold.

$$f(D, TH_j, j) = \text{count}(TH_j > TH_{av}) \quad (9.21)$$

If the function $f(\cdot)$ from Eq. 9.21 is in a local maximum in a current frame, then the previous and the following frames are the scene boundaries in video sequence. Such function displaces three types of dynamic scenes performance: valid transition, smooth transition, and sharp transition as Fig. 9.8 shows.

The smooth transition is the most complex case for DVS. Our recommendations connect with the sub-dividing of such transition in several sub-scenes for better visibility receiving.

9.5.2 Deblurring for Visual Objects with Complex Motion

The blur is a usual effect during a hand-held shooting. Unfortunately, the classical DVS approach based on the smoothness of camera motion leaves the blurriness artifacts untouched [39]. On the one hand, it is caused by camera jitters or both camera jitters and motion of objects. Such blur ought to be compensated before motion estimations, because the tracking of feature points or a motion field building are not reliable in the blurred frames. On the other hand, it may be assumed that only small amount of frames in the original video sequence are blurred frames. Such specialties permit to restore the original video sequence by a deblurring procedure more successful in comparison with other scopes of video restoration. The common strategy is to find the blurred frames, create the spatio-temporal blur kernel, and restore the blurred frames. Let us notice that a deblurring procedure can be applied only in the unreal-time applications.

The deblurring as a fundamental problem has been extensively studied in image processing and computer vision [58]. Its causes may be a high speed of moving objects or a directed high speed of moving camera. The most of existing methods are based only on the spatial blur kernel. The reasonable way of the classical approach is based on the single frames deblurring with following generation a temporal result. Through the non-aligned frames and a temporal coherence, even the classical multi-frame deblurring approaches are useless [59, 60].

The proposed approach removes the blurs caused only by a hand-held shooting. If a video sequence contains the blurring objects in a scene, then one of existing deblurring techniques may be applied, for example, the single image deblurring [61], the multi-image deblurring [62], the video deblurring by interpolation [37].

Let us suppose that an original video sequence is separated in scenes, and the following discussion will be concern to a single scene, which includes non-significant displacement describing by the simplest translation model because of neighbor frames similarity. For detection a blurred frame, the simple procedure of neighbor frames subtraction in each pixels with following total sum calculation of their absolute differences is used. Preliminary, the frames ought to be transformed to YUV-color space, and the brightness component Y is analyzed. A blur frame is characterized by a higher homogeneity distribution, and experiments show that such differences between normal and blur frames will be less than between two non-blurred frames in a non-stabilized video sequence.

A motion blurring due to camera jitters can be modeled as a spatial-temporal invariant convolution process described by Eq. 9.22, where f is a blurred frame, g is a non-blurred frame, p is the blur kernel (point spread function), n is the noise, and a $\text{sign} *$ means the convolution operator.

$$f = g * p + n \quad (9.22)$$

The recovery of frame g from a blurred frame f calls the image deconvolution problem. Two cases of image deconvolution: a non-blind deconvolution and a blind

deconvolution (the last one is actual for the DVS task), always varies in the different frames. In the case of a blind deconvolution, both a blur kernel p and a frame g are unknown. However, the prior assumptions of a kernel p and a frame g have to be made. The Gaussian-type optical blurring may be accepted, and also a frame g can be replaced by an adjacent non-blurred frame. In general, the motion blur kernel is expressed by Eq. 9.23, where C is a continuous curve of finite length in dimension R^2 denoting a camera trajectory, $v(x, y)$ is a speed function varying along a curve C . However, the estimation of camera trajectory and the definition of speed function is not a trivial task.

$$p = v(x, y)|_C \tag{9.23}$$

In particular case, when the blurred images are caused only by camera jitters, the simple procedure for blurred images detection is proposed. If small linear displacement in two neighbor frames is known, then the total sum of pixels differences will have less value between non-blurred and blurred frames or between two blurred frames in comparison with two non-blurred frames. The experimental results confirmed such assumption. Two variants of blurred frames restoration are possible, when a number of blurred frames is 1–2 and more. In the first case, the source for local blur kernel will be a decreased by the non-blurred frame to avoid the problems with shifts of boundaries. In the second case, the complex procedure of frames interpolation is required [63], which will replace the blurred frames by 2–5 interpolated frames. Let us specify the common expression for restoration of a single blurred frame. First, the local blur kernels are applied for each pixels as a weighed function $f_b(j, l)$ of a patch in a warping frame centered by pixel l . As a result, the deblur function $f_d(i, m)$ will be received in a spatial slicing window W_i , $i = n \times n$, $n = 7–11$ pixels centered by pixel m . Equation 9.24 provides the local patch deblurring, where $w(i, m, j, l)$ is a weight, determined by Eq. 9.25 as a Gaussian distributed value.

$$f_d(i, m) = \frac{\sum_{(j,l) \in W_{i,m}} w(i, m, j, l) f_b(j, l)}{\sum_{(j,l) \in W_{i,m}} w(i, m, j, l)} \tag{9.24}$$

$$w(i, m, j, l) = \exp\left(-\frac{(f_b(j, l) - f_d(i, m))^2}{2\sigma_{W_{i,m}}^2}\right) \tag{9.25}$$

Second, the deblurred frame is formed by the local deblurred patches, which can overlap each others. The more accurate approach connects with a patch-based texture synthesis [64]. To accelerate the deblurring process of overlapped patches, a sparse regular grid may be built, which also helps to avoid an over-smooth de-blurring effect caused by averaging of many patches.

Figure 9.7 shows the stabilization result by changing of frame boundaries in static scene. The similar effect is watched in dynamic scene. The using of

background model or/and a tile texture reconstruction of frame boundaries [65] permits to leave frame sizes non-changeable. More offensive problems appear, when a source of tile texture reconstruction is absent in the original video sequence. In this case, the texture synthesis using data from neighbor spatial regions can be applied to interpolate the missing pixels.

9.5.3 Motion Estimation of Background and Foreground

Usually the building of the background model without moving foreground objects is impossible. The local motion estimation can be done by using a multi-level motion model in affine or perspective scenes with depth. This is one of the main issues in the dynamic scene processing. For global motion estimation, a clustering procedure from Sect. 9.4.1 may be applied with more often re-calculations according to current camera speed. The motion estimation in a dynamic scene is implemented by a way including the following steps:

- The motion estimation of background.
- The motion estimation by feature tracking.
- The accurate estimation of moving objects by optical flow

The main idea of preliminary background motion estimation consists in detection such motion level (levels), which is owned to the background of dynamic scene with a high probability value. The basic model of multi-level motion is built on the following assumptions:

- Each pixel in a current frame is characterized by a motion vector, which connects it with a pixel in following frame.
- A set of various parametric motion levels exists. Each of levels uses the own probability model.
- A motion on each level is determined by a mixture Gaussian model.

In this model, a set of inner similar motion levels ml is determined, which correspond to the rigid objects situated on the different distances from a moving camera and regions rf in a frame. Let some motion structure $MS_i(x, y)$ in a point (x, y) of i th current frame corresponds to a level motion lm with θ_{lm} parameters. It means that in following $(i + 1)$ frame, a motion model $MS_{i+1}(x, y)$ will be shifted in a point $((x, y) + \mathbf{v}(x, y, \theta_{ml}))$ with an error of measurement $V_{xy, ml}$, which has value 1, if a structure owns to ml th motion level, and otherwise. Let us assume that any frame has a Gauss noise with standard deviation σ . Equation 9.26 provides a plausibility function, where $\boldsymbol{\theta} = (\theta_1, \dots, \theta_{rf})$.

$$L(V, \boldsymbol{\theta}) = - \sum_{xy, ml} V_{xy, ml} \frac{(MS_i(x, y) - MS_{i+1}(x + v_x(x, y, \boldsymbol{\theta}_{ml}), y + v_y(x, y, \boldsymbol{\theta}_{ml})))^2}{2\sigma^2} + C \quad (9.26)$$

The probabilities of motion values can be represented as the clustering maps of local motion estimations in each level and are provided by Eq. 9.27, where $V_{xy, ml}$ denotes a motion level. Each intensity level determines a motion level, and each motion level connects with own motion model of affine type usually.

$$P(V_{xy, ml} = 1 | MS_i, MS_{i+1}, \boldsymbol{\theta}) \quad (9.27)$$

For background estimation, not all from a set of motion levels can be chosen but only whose, which have better stabilization results; in other words the levels, where the moving objects are absent. After such pseudo-static levels extraction, a background motion for the LMVs estimation can be built. For each pixel, a Gaussian distribution $P(I_{xy, ml} | \boldsymbol{\theta})$ in RGB color space is determined by Eq. 9.28, where $I_{ml}(x, y)$ is the intensity value of pixel (x, y) on motion level ml , μ_i is a mean value in neighborhood, Σ_i is the covariance matrix, and $|\Sigma_i|$ is its determinant. Values μ_{ml} and Σ_{ml} are determined from a set of initial frames in dynamic scene.

$$P(I_{xy, ml} | \boldsymbol{\theta}) = \frac{1}{(2\pi)^{3/2} |\Sigma_{ml}^{1/2}|} \exp\left(-\frac{1}{2} (I_{ml}(x, y) - \mu_{ml})^T \sum_{ml}^{-1} (I_{ml}(x, y) - \mu_{ml})\right) \quad (9.28)$$

The updating of such background model is not required because a scene is dynamic, and the background model will recalculated with a high frequency in comparison with a static scene. In general, the efficiency of background estimation is determined by a camera speed [66]. More speed value of camera means a less-successful background motion approximation.

The core of the DVS is the LMVs estimations of objects motions, and more acceptable decision for dynamic scenes will be a feature tracking approach. Two main strategies the SIFT and the SURF algorithms were investigated in dynamic scenes. The SIFT algorithm detects and describes the distinctive features based on difference of Gaussians of an image at different scales [67]. It detects the robust features, and builds a key point descriptor (for each feature), which is invariant to translation, rotation and scale. Such technique provides the accurate interframe key point matching and includes four steps, as mentioned below:

- The Difference-of-Gaussian (DoG) scale-space construction.
- The stable feature detection.
- The gradient orientation and magnitude assignment.
- The extraction of feature descriptor.

The first three items can be regarded as the step of feature detection, and the fourth one as the step of feature description. The SIFT feature is invariant to translation, scaling, and rotation, while at the same time it is quite robust to lighting change. A number of key points are restricted according to data interpolation, removal of low-contrast feature points, and feature points with high edge responses, using a threshold value, affine or 3D projection parameters. The detection of orientation is based on the local image gradient directions. In the neighborhood of feature point, Eqs. 9.29–9.30 calculate the gradient magnitude $M(x, y, \sigma)$ and the orientation $\theta(x, y, \sigma)$, where $L(x, y, \sigma)$ is a Gaussian-smoothed image at required scale; σ is a standard deviation; (x, y) is coordinates of pixel [67].

$$M(x, y, \sigma) = \sqrt{(L(x+1, y, \sigma) + L(x-1, y, \sigma))^2 + (L(x, y+1, \sigma) + L(x, y-1, \sigma))^2} \quad (9.29)$$

$$\theta(x, y, \sigma) = \arctg \frac{L(x, y+1, \sigma) + L(x, y-1, \sigma)}{L(x+1, y, \sigma) + L(x-1, y, \sigma)} \quad (9.30)$$

Then the orientation histogram with 36 bins (10° on an each bin) is created by the gradient magnitudes and the Gaussian-weighted circles with $\sigma = 1.5 \sigma$. The orientations, corresponding to local peaks (more 80 %), are assigned with the feature points. During a step of descriptor extraction, the local gradient data is used to create the future point descriptors (a set of 16 histograms aligned in a 4×4 grid, each with 8 orientation bins). Thus, the resulting descriptor contains 128 elements.

However, it is very difficult to achieve the software-based real-time computing of SIFT features due to its computational complexity. At present, the hardware architectures are designed. One of such architectures is implemented in a fully parallel hardware based on Field Programmable Gate Array (FPGA) with the SIFT feature description by a high-performance fixed-point Digital Signal Processor (DSP) chip. Such FPGA + DSP hardware module designed by Zhong et al. can be directly driven by the output of a regular video camera [68]. The system is able to detect the SIFT features in the images with sizes 320×256 pixels within 10 ms and takes merely about 80 μ s per a SIFT feature descriptor.

The SURF descriptor is based on convolutions and uses the Hessian matrix-based measure for a distribution-based detector [69]. The Hessian matrixes in continuous and discrete variants are presented in Fig. 9.9.

The Hessian matrix $\mathbf{H}(P, \sigma)$ in a point P is determined by Eq. 9.31, where $L_{xx}(P, \sigma)$, $L_{xy}(P, \sigma)$, $L_{yx}(P, \sigma)$, and $L_{yy}(P, \sigma)$ are convolutions the second derivative of Gaussian $G(P)$ with a function describing a frame I_p in a point P along OX direction, diagonal in the first quadrant, OY direction, and diagonal in the second quadrant respectively.

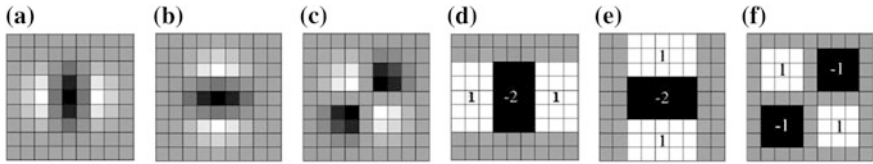


Fig. 9.9 A view of Hessian matrix: **a** a continuous variant along OX, **b** a continuous variant along OY, **c** a continuous variant along XY, **d** a discrete approximation along OX, **e** a discrete approximation along OY; **f** a discrete approximation along XY

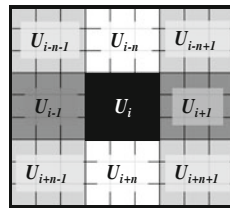


Fig. 9.10 A view of FELF detector

$$\mathbf{H}(P, \sigma) = \begin{bmatrix} L_{xx}(P, \sigma) & L_{xy}(P, \sigma) \\ L_{yx}(P, \sigma) & L_{yy}(P, \sigma) \end{bmatrix} \tag{9.31}$$

Equation 9.32 calculates the convolution along OX direction.

$$L_{xx}(P, \sigma) = \frac{\partial^2}{\partial x^2} G(P) * I_P \tag{9.32}$$

The SURF is based on the Haar wavelet response in the selected direction. It constructs a square region aligned to it, and extracts the SURF descriptor, which is invariant to rotation. The Haar wavelets are easy computed by integral images, if the window location in a point of interest is split up in 4×4 sub-regions. An underlying intensity pattern (first derivatives) of each sub-region is described by vector $\mathbf{V}_H = (\Sigma d_x, \Sigma d_y, \Sigma |d_x|, \Sigma |d_y|)$, where d_x and d_y are the Haar wavelet responses in horizontal and vertical directions, $|d_x|$ and $|d_y|$ are absolute values of corresponding responses. The overall vector will contain 64 elements. The resulting SURF descriptor is invariant to rotation, scaling, and lighting change. The SURF detector has a similar performance in comparison with other descriptors being at the same time faster.

In some applications, the DoGs detector (the Laplacian of Gaussian detector) is used, which shows the difference between two Gaussian smoothed images. Such approach is applied for the SIFT detector to build a scale space pyramid by sub-sampling images and convolving with differently sized kernels. Maxima and minima values are determined to find the response from the Difference of Gaussian

function in the 9-pixel neighborhood on the same scale level, at the scale level above, and the scale level below. The similar approach is used in a Finite Element Laplacian Feature (FELF) detector, where second order Gaussian smoothed image derivatives are used to compute the Hessian matrix [70]. Figure 9.10 provides a view of such detector.

The following feature points tracking by Lucas-Kanade algorithm [71] is a well-designed procedure, discussing in literature [12, 72].

The optical flow is a widely distributed method for accurate motion estimation in video sequences [73, 74]. It builds the motion vectors between two neighbor frames at time instants t and $(t + \delta t)$ in every pixel position with coordinates (x, y) . The intensity function $I(x, y, t)$ moves by δx , δy , and δt between two neighbor frames. Under the assumption that the intensity function, which describes a visual object, remains constant, the main equation of optical flow for motion estimation in video sequences, can be written in a view of Eq. 9.33.

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t) \quad (9.33)$$

On account of the motion is small enough, Eq. 9.33 may be performed by expanding function $I(x, y, t)$ in a Taylor series by Eq. 9.34, where H.O.T. means high order terms, which are small enough and ignored.

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t) + \frac{\partial I}{\partial x} \delta x + \frac{\partial I}{\partial y} \delta y + \frac{\partial I}{\partial t} \delta t + \text{H.O.T.} \quad (9.34)$$

From Eq. 9.34, the Eq. 9.35 follows, where v_x and v_y are the (x, y) speed components, $\delta x/\delta t$, $\delta y/\delta t$, $\delta t/\delta t$ are the partial derivatives in coordinates (x, y, t) .

$$\begin{aligned} \frac{\partial I}{\partial x} \delta x + \frac{\partial I}{\partial y} \delta y + \frac{\partial I}{\partial t} \delta t &= 0 \\ \frac{\partial I}{\partial x} \frac{\delta x}{\delta t} + \frac{\partial I}{\partial y} \frac{\delta y}{\delta t} + \frac{\partial I}{\partial t} \frac{\delta t}{\delta t} &= 0 \\ \frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t} &= 0 \end{aligned} \quad (9.35)$$

The Eq. 9.35 applied to the gradient of intensity function is represented as symmetric covariance 3D structure tensor \mathbf{J}_S [12]. The eigenvalues $\Lambda = \{\lambda_k\}$, ($k = 1, 2, 3$) in neighborhood 3×3 pixels are characterized the local intensity displacements along two spatial axes OX and OY in Euclid space and a temporal axis. The intensity maps $\lambda_1(I)$, $\lambda_2(I)$, $\lambda_3(I)$, based on eigenvalues $\lambda_1(x, y, t)$, $\lambda_2(x, y, t)$, $\lambda_3(x, y, t)$ of local 3D structure tensor, provide the motion estimations in three dimensions.

To other known estimations of optical flow, the Bouguet approach [75] and the Horn-Schunck approach [76] can be mentioned. The Bouguet approach implements a sparse iterative version of Lucas-Kanade tracking feature points in the pyramids.

In the classical Horn-Schunck method, the optical flow estimations are performed as a variational problem, when a vector field is defined as a minimum of certain energy functional J under the assumption that the frame noise and the optical flow derivatives have a Gaussian distribution. This functional includes two terms: a data term provided by the optical flow constraint and a regularity term based on the gradient of the optical flow. The Eq. 9.36 shows a common view of functional J , where a weight parameter α shows a smoothness degree in a regularity term, Ω is a vector field.

$$J = \int_{\Omega} (I_x v_x + I_y v_y + I_t)^2 dt + \alpha^2 (|\nabla v_x|^2 + |\nabla v_y|^2) \quad (9.36)$$

The optical flow separates the spatio-temporal set of pixels in a set of “moving” points and a set of “static” points. For more accurate separation of such sets especially in noisy (type “salt-pepper”) video sequence, additional procedures can be recommended based on Lorentzian estimator, Tukey’s bi-weight estimator, German-McClure estimator, or Leclerc estimator. For Gaussian noisy video sequences, the improvement can be achieved by applying the higher-order statistics. The optical flow values are caused by an additive noise, which is often modeled by a Gaussian distribution (hypothesis H_1) or by a true motion in each pixel (hypothesis H_0) in Eq. 9.37, where $v_k(\bar{r})$ is a flow estimation in a frame k , $z_k(\bar{r})$ is an additive Gaussian noise, and $u_k(\bar{r})$ is the lighting variation caused by true motion [77].

$$\begin{aligned} H_0 : v_k^0(\bar{r}) &= z_k(\bar{r}) \\ H_1 : v_k^1(\bar{r}) &= u_k(\bar{r}) + z_k(\bar{r}) \end{aligned} \quad (9.37)$$

The computer cost of pixels separation on moving (active) and static ones is reduced, if the Gaussianity in the optical flow estimations will be detected. The classical measure of Gaussianity is the kurtosis, which is equaled to 0 for a Gaussian random variable. The kurtosis is determined by Eq. 9.38, where y is a random value, $E[\cdot]$ is an expectation.

$$\text{kurt}(y) = E[y^4] - 3(E[y^2])^2 \quad (9.38)$$

The kurtosis values for active pixels are significantly higher than those for static pixels with noise-induced optical flow values. The binary mask, which demonstrates a pixel activity, can separate pixels on moving and static more accurate that permits better motion compensation caused by camera jitters.

9.5.4 *Unwanted Motion Compensation*

The unwanted motion compensation can be realized by different ways: a smoothing of the GMVs, camera path estimation, and one of surveillance mode – a frame retargeting. Let us consider these cases particularly.

The challenge of the GMVs smoothing is closely tied with a key frames extraction in dynamic scene. The intervals between key frames can have different values. If key frames are extracted, then the task is transformed for the GMVs smoothing in static scene, as this was considered in Sect. 9.4.1.

For dynamic scenes, the extraction of key frame is more difficult task in comparison to the static scene. The accurate inter-frame differences can be calculated as the correlation of RGB color channels, color histogram, moments of inertia, or descriptors based on feature points [78–80]. Also the difference between the current frame and the follow key frame ought to have a significant value. The mentioned characteristics have high computational cost, and do not propose a real-time application. In this research, a way for calculation of feature points was chosen to detect the key frames in dynamic scene. If a number of feature points is essentially changed in local regions of frame, then this frame is marked as a key frame.

The procedure of camera path estimation requires a pre-segmentation of background, feature descriptors extraction, and background objects recognition in adjacent key frames. The main idea is to find “good” corresponding points and track their displacements in all frames between key frames. As a result, two envelope curves from points with high and low ordinate values will be built because of jump camera jitters. Then the curve with middle ordinate value can be interpolated by linearization (if an angle of shooting does not change, and only transitions are available) or by bilinear or bi-cubic functions (if an angle of shooting changes, and transitions/rotations are available) under the assumption that the ideal trajectory of camera is a blend curve. Such points compulsory own to background. Several pairs of such corresponding points can be determined in adjacent key frames to increase a reliability of camera path estimation.

If displacements between real and interpolated trajectories are calculated for each frame, then the locations of moving foreground object may be recalculated by the displacement values under assumption that a motion is defined by the affine motion model. This approach is enough complex: to find the corresponding points with the following interpolation is a separate high computational task. It can be simplified, if the calibrated video camera will be used. However, the camera calibration for dynamic scenes is too difficult and continuous procedure.

A video retargeting is one of ways for the DVS, when the object of interest is held in the center of a frame in the stabilized position. Such surveillance can be realized by some techniques beginning from the feature points tracking to Kalman filter application. In this section, let us discuss the application of Kalman filter and particle filter for the DVS tasks. Also the interesting issue connects with the re-targeting of non-rigid objects.

The Kalman filter provides a recursive solution to the linear optimal filtering and applies in static and dynamic environment [81]. Feature evaluation by Kalman filter during a tracking process is under the following constrains [82]:

- The confidence estimations and discriminative ability of a feature has the Gaussian distribution.
- Features with higher discriminative ability should have larger confidence estimation, and vice versa.

First, the state of Kalman filter is represented as the combination of confidence estimations $W_t = \{w_t(1), w_t(2), \dots, w_t(N)\}$ and variation “speed” $\Delta W_t = W_t - W_{t-1}$ of each feature, where $w_t(i)$ is a confidence estimation of i th feature. Second, the measurements of the filter $S_t = \{S_t(1), S_t(2), \dots, S_t(N)\}$ are provided by a frame in a time instant t . The predicted equation and the measurement equation of Kalman filter are calculated by Eq. 9.39, where $I_{N \times N}$ is an identity matrix describing a targeting object, u_t is a displacement and v_t is a speed, u_t and v_t are both Gaussian noised functions.

$$\begin{cases} \begin{pmatrix} W_{t+1} \\ t+1 \end{pmatrix} = \begin{pmatrix} I_{N \times N} & I_{N \times N} \\ 0 & I_{N \times N} \end{pmatrix} \begin{pmatrix} W_t \\ \Delta W_t \end{pmatrix} + u_t \\ (S_t) = (I_{N \times N} \quad 0) \begin{pmatrix} W_t \\ \Delta W_t \end{pmatrix} + v_t \end{cases} \quad (9.39)$$

The particle filter is an estimation algorithm for implementing a recursive temporal Bayesian filter by Monte Carlo simulations. It represents a posterior state of moving object by a set of random samples with associated confidence estimations. The feature evaluation by using the particle filter can calculate the confidence estimations or discriminative abilities for non-Gaussian and non-linear distribution [83]. The core of this procedure is to define such feature set, each feature in which is seen as a particle. As a result, a weighted sample of particles at frame t , $\{(i, w_t(i))\}$, $i = 1, 2, \dots, N$, where i denotes the i th feature (particle), is created. The iterations of this process evaluate the temporal consistency even a variation of features is a non-linear and a non-Gaussian [84].

The challenges of non-rigid deformations, rotations, appearance, occlusions, and drifting are known as the template update problem or stability-plasticity dilemma. However, most of approaches such as robust learning algorithms [85], different learning paradigm [86], multiple different classifiers [87], or a conservative learning framework [88] are limited to a bounding-box-based representation.

To avoid of inaccuracy, the segmentation of non-rigid objects can be represented by the deformable parts model [89] or models obtained via the generalized Hough-transform [90, 91]. These methods need large amount of labeled training data, which cannot be provided during a tracking of unknown objects.

In the research of Godec et al. [92], a novel tracking-by-detection approach is proposed. It is based on the online Hough ferns and a couple of procedures: the

voting-based detection and back-projection with a rough GrabCut segmentation [93]. The randomized Hough ferns use simple pixel comparisons as splitting tests that allows the robustly detection of the non-rigid objects. The voting-based detection procedure has a valid geometric relation. The back-projection procedure roughly separates the object from the background pixel-wise. Such approach is very perspective for non-rigid natural objects and blurred objects segmentation.

The random ferns are such modification of random forest, which is based on independent flat test structures instead of tree-like structures [94, 95]. For S binary tests, the best matching class c for a given image sample \mathbf{v} is estimated by Eq. 9.40 assuming a uniform prior distribution over all classes.

$$\begin{aligned} c &= \arg \max_c P(c|\mathbf{v}) = \arg \max_c P(c|x_1, x_2, \dots, x_S) \\ &= \arg \max_c P(x_1, x_2, \dots, x_S|c) \end{aligned} \quad (9.40)$$

However, the joint features distribution over all tests cannot be modeled in practice. The features are grouped in several independent sets x_S . If a power of set is equaled 1, then a well-known Naïve-Bayes formulation is acceptable (Eq. 9.41).

$$P(x_1, x_2, \dots, x_S|c) = \prod_{c=1}^S P(x_S|c) \quad (9.41)$$

The Random Ferns is based on a semi-Naïve-Bayes formulation of Eq. 9.41 using larger feature sets and expressed by Eq. 9.42, where \tilde{x}_m denotes a set test and M is a number of used groups.

$$P(x_1, x_2, \dots, x_S|c) = \prod_{m=1}^M P(\tilde{x}_m|c) \quad (9.42)$$

Godec et al. show that Naïve formulation can be used for tracking, if $P(x_n - c)$ is modeled by using histograms instead of binary features [96]. However, binary tests can be also interpreted as a semi-Naïve formulation. For tracking of unknown objects, a node optimization is difficult because of the limited training data in several frames. A special tree-growing scheme was proposed by Saffari et al. [85]. The more complex statistics use the Hough-transform.

9.5.5 Motion Inpainting in Dynamic Scenes

The motion inpainting in dynamic scenes can be considered as the task of missing data restoration. Many interesting methods are designed for texture reconstruction, for example, image inpainting by contourlet transform [97], regularized image restoration [98], etc. More reasonable approach is the information extraction from

the sharp neighbor frames. In this research, two cases are applied: the frame boundaries restoration and the frames interpolation between key frames.

Frame boundaries restoration in dynamic scenes includes two cases, when the missing pixels can be taken from the previous frame and when it is impossible. First, the assumption is declared that the missing pixels can be founded. A pseudo-panoramic background may be built from neighbor stabilized frames, and it will be the source of tile texture reconstruction by a line or a field of textons with following inpainting of moving foreground objects in the reconstruction area [65]. Second, if the reconstruction information is absent (very sharp jitters respect to the whole video sequence or the sequential blurred frames), the decision is accepted about removal such frames and adding the interpolated frames under the assumption of smooth camera motion.

For the DVS task, it is enough to apply the simplest linear model of background and foreground objects motion. In this case, the point coordinates (x_n, y_n) in reconstruction frame n are calculated by Eq. 9.43.

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = (n - 1) \cdot \begin{bmatrix} x_i - x_{i-1} \\ y_i - y_{i-1} \end{bmatrix} \quad (9.43)$$

By using a normalized correlation function, the location of slicing window 11×11 or 15×15 pixels from stabilized area of a recoverable frame and a pseudo-panoramic background is determined. Such value shows the background place suitable for restoration of frame boundaries. Then a field of textons is replaced in a missing area of frame with corresponding stitching procedures [99, 100]. If a foreground moving object is in this part of scene, then its image is also restore according to linear motion model but with the own shifts. Let us notice that the proposed frame boundaries restoration is out from border of real-time application because the analysis of sequential frames can be required, and also the work with texture is usually a durable process.

The task of missing frames interpolation is a separate complex issue in digital video processing. In the current research, this is an additional aspect of motion inpainting in dynamic scenes and concerns only to the blurred frames. The goal is to improve a video sequence, when several sequential frames are blurred very strongly, and the procedure proposed in Sect. 9.5.2 is a non-useful processing.

In such particular case, the blurred frames are removed. Then very complex and computer high cost analysis is initiated for such pseudo-static scene. The main idea is to build the previous and the following trajectories of moving objects in a missing interval. The procedure is based on interpolation of two data types: background and foreground. In literature, the main attention gives consideration for interpolation of moving foreground objects in a scene. Three main approaches, among others, are used:

- The interpolation based on functions [101–104].
- The interpolation based on autoregressive modeling [105, 106].
- The interpolation using Markov random fields [107–112].

Objectively, if more blurred frames are in a video sequence, then the results of interpolation will be worse especially in the dynamic scenes.

9.6 Discussion of Experimental Results

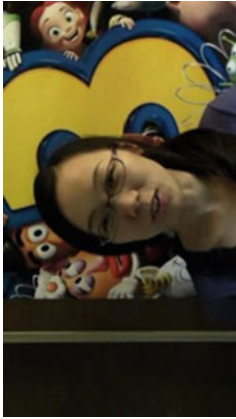

Two main directions of the experimental work were determined: the motion estimations based on the proposed methods (Sect. 9.6.1) and the stabilization estimations (Sect. 9.6.2). Six video sequences received from the static camera and eight video sequences received from the moving camera were used in this research. The titles, URL, and snapshots of some investigated static and dynamic scenes are presented in Tables 9.2 and 9.3, respectively.

All experiments were executed by the designed software tool “DVS Analyzer”, v. 2.04, which was developed in Laboratory of Image and Videos Processing (Department of Informatics and Computer Technique, Siberian State Aerospace University). The software tool “DVS Analyzer” has two modes: the pseudo real-time stabilization of video sequences, which are broadcasted from the surveillance cameras (the simplified processing) and the unreal-time stabilization of available video sequences (the intelligent processing). The architecture of the software tool includes the extended set of program modules, which can be designed or developed independently each from others. The Pre-processing Module, the Motion Estimation Module, the Motion Compensation Module, the Motion Inpainting Module, the Module of Quality Estimation, the Core Module, and the Interface Module are the main components of the software tool “DVS Analyzer”. Let us briefly discuss the functionality of each program module.

The Pre-processing Module involves various spatio-temporal filters such as the auto-contrast filter, the temporal 2D_cleaner filter, and the adaptive Gauss filter, which are applied to frames representing in color RGB-, HSV-, and YUV-spaces. Also this module divides a video sequence in the scenes. The Motion Estimation Module calculates the LMVs by the BMA and feature correspondences. It builds the GMVs in frames by using of fuzzy TSK model. The Motion Compensation Module determines the SMVs in each frame based on the GMVs. The Motion Inpainting Module realizes the frame stabilization by a re-calculating of an original frame according to the U MVs in scenes. The procedure for restoration of frame boundaries and the deblurring procedure are executed in this module. The Module of Quality Estimation provides the quality comparison of original and stabilized video sequences according to PSNR and ITF metrics. The Core Module controls and coordinates the work of all other Modules and includes program codec for a pre-processing and saving of the stabilized video sequences. The Interface Module displays the received results in file/monitor/printed version.

The software tool “DVS Analyzer”, v. 2.04 was designed in the Rapid Application Development Embracadero RAD Studio 2010. Some external software tools were used: the libraries “Video for Windows” for initial processing and “Alpha-Controls 2010”, v. 7.3 for enhanced user interface and a video codec “K-Lite Codec

Table 9.2 The description of investigated static scenes

Title, URL	Snapshot	Resolution	Frames	Motion type
"SANY0025_xvid.avi" http://cpl.cc.gatech.edu/projects/videostabilization/		640 × 360	445	Slow motion of camera, motion of large object
"If_juggle.avi" http://cpl.cc.gatech.edu/projects/videostabilization/		480 × 360	460	Stable camera, fast motion of small objects

(continued)

Table 9.2 (continued)


Title, URL	Snapshot	Resolution	Frames	Motion type
"akiyo.avi" http://see.xidian.edu.cn/vips/database_Video.html		352 × 288	300	Stable camera, motion of large object

Table 9.3 The description of investigated dynamic scenes

Title, URL	Snapshot	Resolution	Frames	Motion type
"Cat_orig.avi" http://www.youtube.com/watch?v=4XOCRIDA4w4		640 × 360	1582	Camera movement, fast motion of large object, lighting change
"Gleicher1.avi" http://cpl.cc.gatech.edu/projects/videostabilization		640 × 360	495	Camera movement, complex scene, low-contrast background
"Sam_1.avi" http://cpl.cc.gatech.edu/projects/videostabilization		640 × 360	278	Camera movement, fast motion of object, irregular motion

Pack”, v. 8.0. The experiments were performed on a computer with the following configuration: CPU Intel Core I5.760, 4 Gb RAM, Nvidia GeForce 460GTX, Windows 7 64 bit.

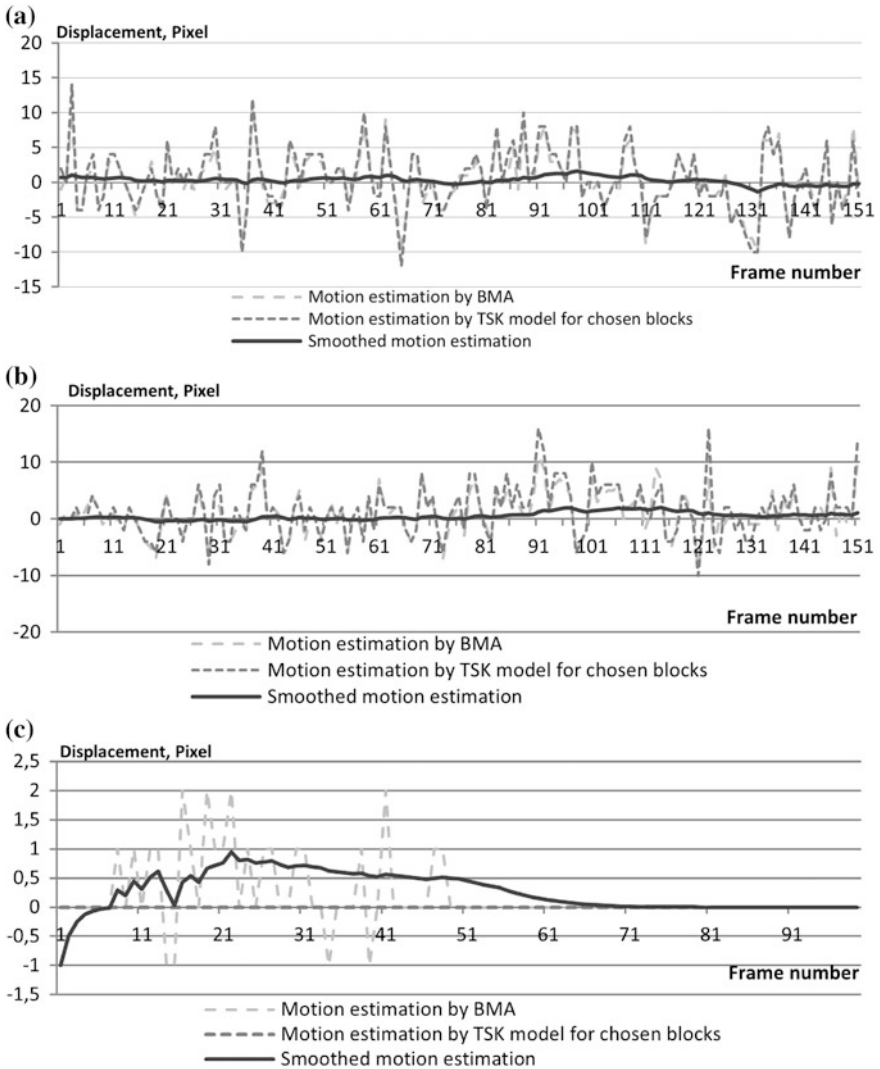


Fig. 9.11 Graphics of motion estimation and compensation results in static scenes: **a** “SANY0025_xvid.avi”, **b** “lf_juggle.avi”, **c** “akiyo.avi”

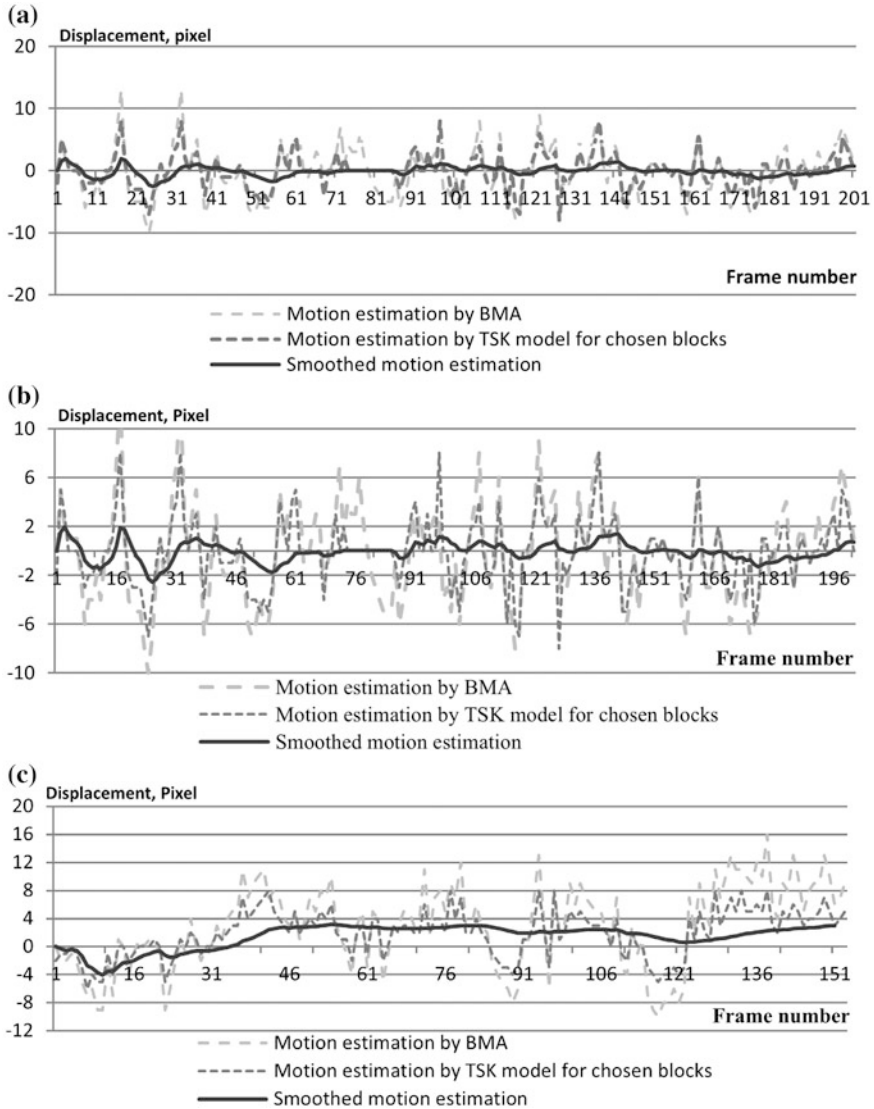


Fig. 9.12 Graphics of motion estimation and compensation results in dynamic scenes: a “Cat_orig.avi”, b “Gleicher4.avi”, c “Sam_1.avi”

9.6.1 Experimental Results for Motion Estimations

The experimental graphics for motion estimation and compensation are represented in Figs. 9.11 and 9.12 for static and dynamic scenes, respectively.

The motion estimations in static scenes by using the TSK model provide more accurate estimations of the global motion due to the fact that the motion of

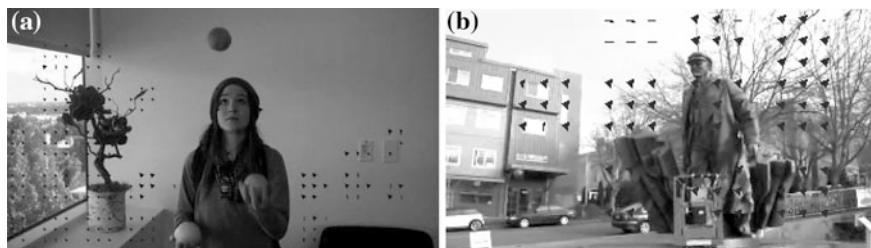


Fig. 9.13 The frames with “rejected” regions for motion estimations: **a** “EllenPage_juggling.avi”, **b** “Gleicher1.avi”

foreground objects is not considered. Such algorithmic specialty is well demonstrated in video sequences (“road_cars_krasnoyarsk_synthetic.avi”; “akiyo.avi”, “Butovo_synthetic.avi”). In original video sequences with static scenes, these differences are less and appear only in the GMV estimations. For video sequences including many small sizes objects (“lf_juggle.avi”, “Butovo_synthetic.avi”), the algorithm for motion estimations by using the TSK model shows the best results increasing ITF value up on 3 dB. For the dynamic scenes stabilization, the detection of the SMVs is the most important step. Such SMVs detection is executed based on the GMVs in a frame.

The non-contrast regions in frames decrease the quality of stabilization (“EllenPage_Juggling.avi”, “Gleicher4.avi”, “Gleicher1.avi”) and show the unpredicted results. Therefore, the non-contrast regions do not process in the most of frames. The frames with such “rejected” regions are represented in Fig. 9.13.

If the GMVs are estimated inaccurately, then the quality of following stabilized video sequence will decrease. The developed algorithm of motion estimation is non-sensitive to the large moving foreground moving in video sequences “EllenPage_Juggling.avi”, “Sam_1.avi”, “Cat_orig.avi”, “Cleicher3.avi”. The best global motion estimations were received for video sequence “EllenPage_Juggling.avi” due to ignoring the fast moving of foreground objects.

9.6.2 Experimental Results for Stabilization Estimations

The objective estimations of the DVS quality were calculated by Mean-Square Error (MSE) and PSNR metrics between a current frame I^{cur} and a key frame I^{key} , which are expressed in Eqs. 9.44–9.45, where I_{max} is a maximum of pixel intensity, m and n are the frame sizes along OX and OY axes.

$$MSE = \frac{1}{m \times n} \sum_{y=1}^m \sum_{x=1}^n (I^{cur}(x, y) - I^{key}(x, y))^2 \quad (9.44)$$

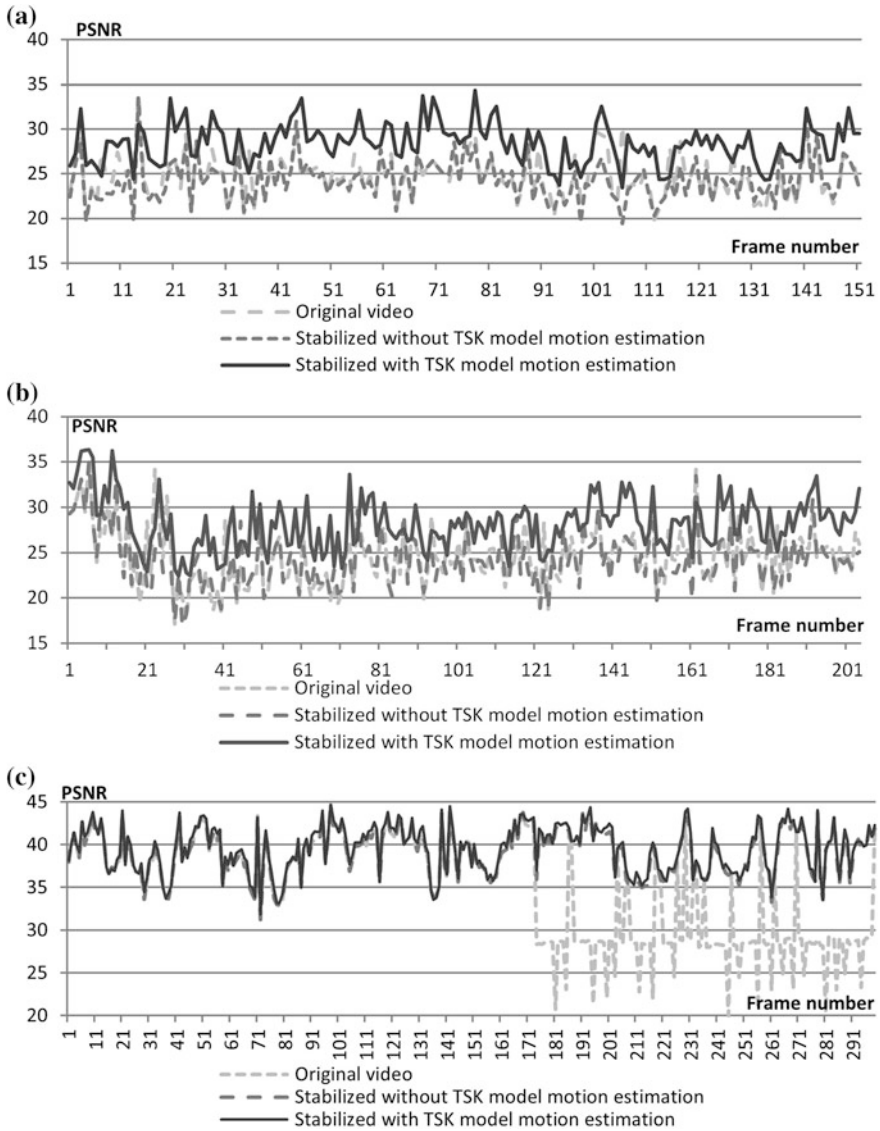


Fig. 9.14 Graphics of stabilization quality in static scenes: **a** “SANY0025_xvid.avi”, **b** “lf_juggle.avi”, **c** “akiyo.avi”

$$PSNR = 10\log_{10}\left(\frac{I_{\max}^2}{MSE}\right) \tag{9.45}$$

The PSNR metric is useful for estimations between neighbor frames. The quality of the ITF estimations provides the objective estimation in whole video sequence.

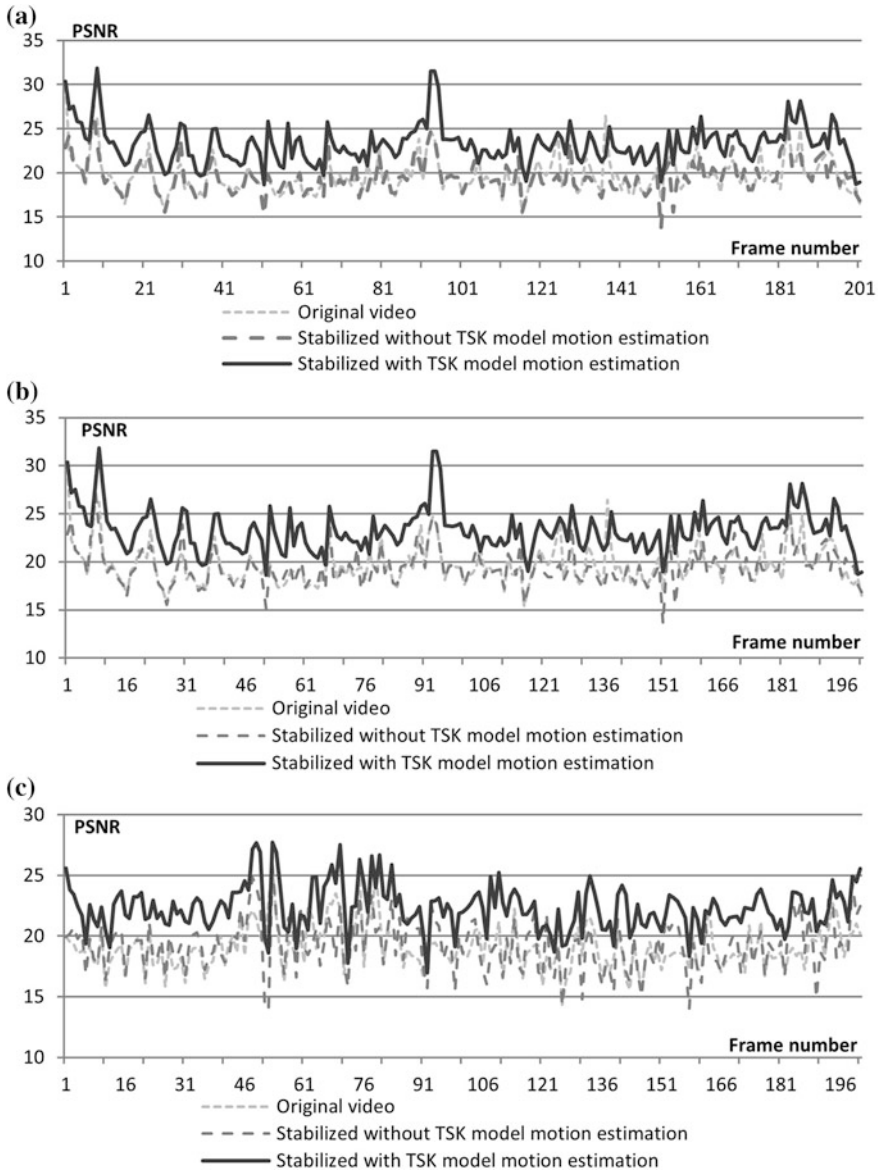


Fig. 9.15 Graphics of stabilization quality in dynamic scenes: **a** “Cat_orig.avi”, **b** “Gleicher4.avi”, **c** “Sam_1.avi”

The ITF of the stabilized video sequence is almost higher than the ITF of the original video sequence. This parameter is calculated by Eq. 9.46, where N_{fr} is a frame amount in a video sequence.

Table 9.4 ITF estimations for static scenes

Video sequence	ITF estimations (dB)		
	Original	Without TSK model	With TSK model
road_cars_krasnoyarsk.avi	22.70482	22.80707	25.91258
SANY0025_xvid.avi	20.5389	21.09076	23.79189
If_juggle.avi	24.30286	24.37177	28.06012
akiyo.avi	35.92952	39.14661	39.53257
EllenPage_Juggling.avi	24.65855	25.23049	28.58255
Butovo_synthetic.avi	22.26415	27.19789	27.20789

Table 9.5 ITF estimations for dynamic scenes

Video sequence	ITF estimations (dB)		
	Original	Without TSK model	With TSK model
Cat_orig.avi	25.07131	26.47094	28.14086
Gleicher4.avi	19.29703	19.50634	23.18371
Sam_1.avi	19.09737	19.28141	22.20112
Gleicher1.avi	18.86996	19.48223	22.78846
Gleicher2.avi	19.91954	20.36718	24.56673
Gleicher3.avi	16.55214	16.71899	20.12285
new_gleicher.avi	17.28921	17.81638	21.70575
yuna_long_original.avi	17.84131	18.94389	21.46971

$$ITF = \frac{1}{N_{fr}} \sum_{k=0}^{N_{fr}} PSNR_k \tag{9.46}$$

The experimental graphics for stabilization quality are represented in Figs. 9.14 and 9.15 for static and dynamic scenes respectively. As shown from Figs. 9.14 and 9.15, the PSNR estimations of the stabilized video sequences are always higher than the PSNR estimations of the original video sequences.

Tables 9.4 and 9.5 contain the ITF estimations for all (used in experiments) whole video sequences with static scenes and dynamic scenes, respectively.

As it seems from Tables 9.4 and 9.5, the video stabilization results are different for various video sequences because of varied foreground and background content, moving objects, lighting, noisy, and the shooting condition. The using of the TSK model provides the increment of the ITF estimations up on 3–4 dB or 15–20 %.

The stabilization and temporal results of “Deshaker”, “WarpStabilizer”, “Video Stabilization with Robust L1 Optimal Camera Paths”, and our developed “DVS Analyzer” are situated in Table 9.6.

Table 9.6 Comparison of stabilization algorithms for static and dynamic scenes

Video sequence	Algorithm							
	Deshaker		WarpStabilizer		Video Stabilization with robust L1 optimal camera paths		DVS analyzer	
	ITF (dB)	Time (s)	ITF (dB)	Time (s)	ITF (dB)	Time (s)	ITF (dB)	Time (s)
EllenPage_Juggling.avi	25.61	3.53	26.68	4.53	27.33	3.17	28.58	3.54
Gleicher4.avi	20.33	1.89	19.15	2.78	20.45	1.44	23.18	1.66
Sam_1.avi	20.09	1.22	20.27	2.65	20.58	1.01	22.20	1.23
road_cars_krasnoyarsk	22.31	1.45	21.48	2.15	25.2	1.29	25.91	0.24
SANY0025xvid.avi	23.53	1.33	22.7	1.87	22.74	1.34	23.79	0.17
lf_juggle.avi	26.65	1.22	24.41	1.64	26.15	1.18	28.06	0.15

The ITF estimations of the proposed software tool “DVS Analyzer” provides better results (at average 1–3 dB or 5–15 %) with the lower processing time relatively the existing software tools.

9.7 Conclusion and Future Development

In this chapter, the intelligent methods for digital video stabilization in static and dynamic scenes were developed. The extended literature review represents the state-of-art in the DVS till the present time. In this research, many novel reasonable methods were proposed and realized including the statistical background model, the scene separation in video sequences received from the moving camera, the fuzzy TSK model application, the detection of “good” regions in frames, which contain only camera jitters, frames boundaries restoration in static and dynamic scenes, etc. All methods and algorithms were realized by the designed software tool “DVS Analyzer”, v. 2.04.

For experiments, six video sequences received from the static camera and eight video sequences received from the moving camera were used. Graphics of the motion estimation and compensation and the stabilization quality demonstrate the improvements relative to the original video sequences. The PSNR and ITF metrics were used to estimate the received results. The stabilized results directly depend from varied foreground and background content, the moving objects, lighting, noisy, and the shooting condition for each video sequence. The ITF estimations increase up on 3–4 dB or 15–20 % relative to the original video sequences. The ITF estimations of the proposed software tool “DVS Analyzer” provides better results (at average 1–3 dB or 5–15 %) with the lower processing time relatively the existing software tools.

The future efforts are connected with the development of advanced motion inpainting methods and algorithms for the DVS task and also fast realization of algorithms without the essential accuracy reduction for pseudo real-time applications.

References

1. Battiato S, Lukac R (2008) Video stabilization techniques. In: Furht B (ed) *Encyclopedia of multimedia*. Springer Science+Business Media, New York
2. Chen H, Liang CK, Peng YC, Chang HA (2007) Integration of digital stabilizer with video codec for digital video cameras. *IEEE Trans Circuits Syst Video Technol* 17(7):801–813
3. Rawat P, Singhai J (2011) Review of motion estimation and video stabilization techniques for hand held mobile video. *Int J of Signal and Image Process* 2(2):159–168
4. Leghmizi S, Liu S (2011) A survey of fuzzy control for stabilized platforms. *Int J of Comput Sci Eng Surv (IJCSES)* 2(3):48–57
5. Battiato S, Bruna AR, Puglisi G (2010) A robust block based image/video registration approach for mobile imaging devices. *IEEE Trans Multimedia* 12(7):622–635
6. Bosco A, Bruna A, Battiato S, Bella G, Puglisi G (2008) Digital video stabilization through curve warping techniques. *J IEEE Trans Consum Electron* 54(2):220–224
7. Lee J, Park Y, Lee S, Paik J (2009) Statistical region selection for robust image stabilization using feature-histogram. In: *IEEE international conference on image processing (ICIP)*, pp 1553–1556
8. Liu F, Gleicher M, Jin H, Agarwala A (2009) Content-preserving warps for 3D video stabilization. *ACM Trans Graph (SIGGRAPH)* 28(3):44:1–44:9
9. Wang JM, Chou HP, Chen SW, Fuh CS (2009) Video stabilization for a hand-held camera based on 3D motion model. In: *IEEE international conference on image processing (ICIP)*, pp 3477–3481
10. Nestares O, Gat Y, Haussecker H, Kozinsev I (2010) Video stabilization to a global 3D frame of reference by fusing orientation sensor and image alignment data. In: *9th IEEE international symposium on mixed augmented reality (ISMAR)* 257–258
11. Hansen M, Anandan P, Dana K, van der Wal G, Burt PJ (1994) Real time scene stabilization and mosaic construction. *Image understanding Workshop, Defense Advanced Research Project Agency (DARPA)*, pp 457–465
12. Favorskaya M (2012) Motion estimation for object analysis and detection in videos. In: Kountchev R, Nakamatsu K (eds) *Advances in reasoning-based image processing, analysis and intelligent systems*. Springer, Berlin Heidelberg
13. Babagholami-Mohamadabadi B, Bagheri-Khaligh A, Hassanpour R (2012) Digital video stabilization using radon transform. *Digital Image Comput: Tech Appl (DICTA)* 12:1–8
14. Favorskaya M, Levitin K (2014) Early smoke detection in outdoor space by spatio-temporal clustering using a single video camera. In: Tweedale JW, Jain LC (eds) *Recent advances in knowledge-based paradigms and applications*. Springer, Berlin Heidelberg
15. Shakoer MH, Moattari M (2011) Statistical digital image stabilization. *J Eng Technol Res* 3(5):161–167
16. Zhou M, Asari VK (2011) A fast video stabilization system based on speeded-up robust features advances. In: Zhou M, Asari VK (eds) *Adv Vis Comput., LNCSSpringer*, Berlin Heidelberg
17. Ertürk S (2002) Real-time digital image stabilization using Kalman filters. *Real Time Image* 8(4):317–328
18. Pinto B, Anurenjan PR (2011) Video stabilization using speeded up robust features. In: *International conference on communications and signal processing (ICCSP)*, pp 527–531

19. Chang HC, Lai SH, Lu KR (2006) A robust real-time video stabilization algorithm. *J Vis Commun Image Represent* 17(3):659–673
20. Alzoubi H, Pan WD (2008) Fast and accurate global motion estimation algorithm using pixel subsampling. *Inf Sci* 178:3415–3425
21. Del Blanco CR, Jaureguizar F, Salgado L, Garcia N (2008) Automatic feature-based stabilization of video with intentional motion through a particle filter. In: del-Blanco CR, Jaureguizar F, Salgado L, García N (eds) *Advanced concepts for intelligent vision systems (ACIVS) LNCS*. Springer, Berlin Heidelberg
22. Yang J, Schonfeld D, Mohamed M (2009) Robust video stabilization based on particle filter tracking of projected camera motion. *IEEE Trans Circ Syst Video Technol* 19(7):945–954
23. Litvin A, Konrad J, Karl WC (2003) Probabilistic video stabilization using Kalman filtering and mosaicking. *Image and video communication and processing 2003*. SPIE-IS&T Electron Imaging SPIE 5022:663–674
24. Güllü MK, Ertürk S (2004) Membership function adaptive fuzzy filter for image sequence stabilization. *IEEE Trans Consum Electron* 50(1):1–7
25. Battiato S, Puglisi G, Bruna AR (2008) A robust video stabilization system by adaptive motion vectors filtering. In: *IEEE international conference on multimedia and expo*, pp 373–376
26. Tanakian MJ, Rezaei M, Mohanna F (2012) Digital video stabilizer by adaptive fuzzy filtering. *EURASIP J Image Video Process* 2012. SpringerOpen. doi:[10.1186/1687-5281-2012-21](https://doi.org/10.1186/1687-5281-2012-21)
27. Puglisi G, Battiato S (2011) A robust image alignment algorithm for video stabilization purposes. *IEEE Trans Circ Syst Video Technol* 21(10):1390–1400
28. Kyriakoulis N, Gasteratos A (2008) A recursive fuzzy system for efficient digital image stabilization. *J Adv Fuzzy Syst* 8:1–8
29. Amanatiadis AA, Andreadis I (2010) Digital image stabilization by independent component analysis. *IEEE Trans Instrum Measur* 59(7):1755–1763
30. Shen Y, Guturu P, Damarla T, Buckles BP, Namuduri KR (2009) Video stabilization using principal component analysis and scale invariant feature transform in particle filter framework. *IEEE Trans Consum Electron* 55(3):1714–1721
31. Tsai D, Lai S (2008) Defect detection in periodically patterned surfaces using independent component analysis. *Pattern Recogn* 41(9):2812–2832
32. Kim N, Lee H, Lee J (2008) Probabilistic global motion estimation based on Laplacian two-bit plane matching for fast digital image stabilization. *EURASIP J Adv Signal Process* 1–10
33. Pun CM, Lee MC (2003) Log-polar wavelet energy signatures for rotation and scale invariant texture classification. *IEEE Trans Pattern Anal Mach Intell* 25(5):590–603
34. Hu R, Shi R, Shen I, Chen W (2007) Video stabilization using scale invariant features. In: *IEEE 11th international conference on information visualization IV'07*, pp 871–877
35. Tanakian MJ, Rezaei M, Mohanna F (2010) Digital video stabilization system by adaptive motion vector validation and filtering. In: *International conference on communication engineering*, pp 165–183
36. Tanakian MJ, Rezaei M, Mohanna F (2011) Digital video stabilization system by adaptive fuzzy filtering. In: *19th European signal process conference*, pp 318–322
37. Matsushita Y, Ofek E, Ge W, Tang X, Shum HY (2006) Full-frame video stabilization with motion inpainting. *IEEE Trans Pattern Anal Mach Intell* 28(7):1150–1163
38. Pang D, Chen H, Halawa S (2010) Efficient video stabilization with dual-tree complex wavelet transform. *EE368 project report*
39. Liu F, Gleicher M, Wang J, Jin H, Agarwala A (2011) Subspace video stabilization. *ACM Trans Graph* 30(1):4:1–4:10
40. Zhang G, Hua W, Qin X, Shao Y, Bao H (2009) Video stabilization based on a 3D perspective camera model. *Vis Comput* 25(11):997–1008
41. Chen PH, Chen HM, Hung KJ, Fang WH, Shie MC, Lai F (2006) Markov model fuzzy-reasoning based algorithm for fast block motion estimation. *J Vis Commun Image Represent* 17(1):131–142

42. Benmoussat N, Belbachir MF, Benamar B (2007) Motion estimation and compensation from noisy image sequences: a new filtering scheme. *Image Vis Comput* 25(5):686–694
43. Liu X, Cong W (2010) Hybrid-template adaptive motion estimation algorithm based on block matching. *Int Conf Comput Commun Technol Agric Eng*. doi:[10.1109/CCTAE.2010.5543459](https://doi.org/10.1109/CCTAE.2010.5543459)
44. Boudlal A, Nsirir B, Aboutajdine D (2010) Modeling of video sequences by gaussian mixture: application in motion estimation by block matching method. *EURASIP J Adv Signal Process*. doi:[10.1155/2010/210937](https://doi.org/10.1155/2010/210937)
45. Jang SW, Pomplun M, Kim GY, Choi HI (2005) Adaptive robust estimation of affine parameters from block motion vectors. *Image Vis Comput* 23(14):1250–1263
46. Basarab A, Liebgott H, Morestin F, Lyshchik A, Higashi T, Asato R, Delachartre P (2008) A method for vector displacement estimation with ultrasound images and its application for thyroid nodular disease. *Med Image Anal* 12(3):259–274
47. Moreno-Garcia J, Rodriguez-Benitez L, Fernandez-Caballero A, Lopez MT (2010) Video sequence motion tracking by fuzzification techniques. *Appl Soft Comput* 10(1):318–331
48. Sugeno M (1985) *Industrial applications of fuzzy control*. Elsevier Science Inc., New York
49. Battiato S, Gallo G, Puglisi G, Scellato S (2009) Fuzzy-based motion estimation for video stabilization using SIFT interest points. In: *IS&T-SPIE electronic imaging symposium, digital photography V EI-7250*, pp 1–8
50. Pan JY, Hu B, Zhang JQ (2008) Robust and accurate object tracking under various types of occlusions. *IEEE Trans CSVT* 18:223–236
51. O’Hara S, Lui YM, Draper BA (2012) Using a product manifold distance for unsupervised action recognition. *Image Vis Comput* 30(3):206–216
52. Belardinelli A, Carbone A, Schneider WX (2013) Classification of multi-scale spatiotemporal energy features for video segmentation and dynamic objects prioritization. *Pattern Recogn Lett* 34(7):713–722
53. Panagiotakis C, Doulamis AD, Tziritas G (2009) Equivalent key frames selection based on iso-content principles. *IEEE Trans Circuits Syst Video Technol* 19(3):447–451
54. Chao GC, Tsai YP, Jeng SK (2010) Augmented keyframe. *J Vis Commun Image Represent* 21(7):682–692
55. Cheung V, Frey BJ, Jovic N (2005) Video epitomes. *IEEE Comput Soc Conf Comput Vis Pattern Recognit (CVPR)* 1:42–49
56. Yu G, Li Z, Suyu W, Lansun S (2009) A novel scene cut detection method in H.264/AVC compression domain. *Chin J Electron* 18(4):695–699
57. Jokovic J, Horevic D (2009) Scene cut detection in video by using combination of spatial-temporal video characteristics. In: *9th international conference on telecommunication in modern satellite, cable, and broadcasting services*, pp 479–482
58. Xiang S, Meng G, Wang Y, Pan C, Zhang C (2012) Image deblurring with matrix regression and gradient evolution. *Pattern Recogn* 45(6):2164–2179
59. Cho S, Wang J, Lee S (2011) Handling outliers in non-blind image deconvolution. In: *International conference on computer vision ICCV11*, pp 495–502
60. Xu YQ, Wang L, Hu XY, Peng SL (2012) Single-image blind deblurring for non-uniform camera-shake blur. In: *ACCV12*, vol 7726, pp 336–348
61. Whyte O, Sivic J, Zisserman A, Ponce J (2010) Non-uniform deblurring for shaken images. In: *Computer vision and pattern recognition CVPR 2010*, pp 491–498
62. Cai JF, Ji H, Liu C, Shen Z (2009) Blind motion deblurring using multiple images. *J Comput Phys* 228(14):5057–5071
63. Favorskaya M, Pyankov D, Popov A (2013) Motion estimations based on invariant moments for frames interpolation in stereovision. *Proc Comput Sci* 22:1102–1111
64. Kwatra V, Essa I, Bobick A, Kwatra N (2005) Texture optimization for example-based synthesis. *ACM Trans Graph* 24(3):795–802
65. Favorskaya M, Damov M, Zotin A (2013) Accurate spatio-temporal reconstruction of missing data in dynamic scenes. *Pattern Lett Recognit* 34(14):1694–1700

66. Lim T, Han B, Han JH (2012) Modeling and segmentation of floating foreground and background in videos. *Pattern Recognit* 45(4):1696–1706
67. Lowe D (2004) Distinctive image features from scale-invariant key points. *Int J Comput Vis* 60(2):91–110
68. Zhong S, Wang J, Yan L, Kang L, Cao Z (2013) A real-time embedded architecture for SIFT. *J Syst Archit* 59(1):16–29
69. Bay H, Ess A, Tuytelaars T, Van Gool L (2008) Speeded-up robust features (surf). *Comput Vis Image Underst* 110(3):346–359
70. Kerr D, Coleman SA, Scotney BW (2011) Finite element laplacian feature detector. In: IAPR, conference on machine vision and applications (MVA 2011), pp 381–384
71. Lucas B, Kanade T (1981) An iterative image registration technique with an application to stereo vision. In: 7th international joint conference on artificial intelligence (IJCAI), vol 2, pp 674–679
72. Liao B, Du M, Hu J (2010) Color optical flow estimation based on gradient fields with extended constraints. In: International conference on networking and information technology, pp 279–283
73. Lee KJ, Yun ID, Lee SU (2013) Adaptive large window correlation for optical flow estimation with discrete optimization. *Image Vis Comput* 31(9):631–639
74. Nayak NM, Zhu Y, Roy-Chowdhury AK (2013) Vector field analysis for multi-object behavior modeling. *Image Vis Comput* 31(6–7):460–472
75. Kim KS, Jang DS, Choi HI (2007) Real time face tracking with pyramidal Lucas-Kanade feature tracker. In: International conference on computational science and its applications ICCSA'07 Part I, pp 1074–1082
76. Horn BKP, Schunck BG (1981) Determining optical flow. *Artif Intell* 17:185–203
77. Briassouli A, Kompatsiaris I (2009) Robust temporal activity templates using higher order statistics. *IEEE Trans Image Process* 18(12):2756–2768
78. Kim HH, Kim YH (2010) Toward a conceptual framework of key-frame extraction and storyboard display for video summarization. *J Am Soc Inf Sci Technol* 61(5):927–939
79. Ejaz N, Baik SW (2011) Weighting low level frame difference features for key frame extraction using fuzzy comprehensive evaluation and indirect feedback relevance mechanism. *Int J Phys Sci* 6(14):3377–3388
80. Ejaz N, Tariq TB, Baik SW (2012) Adaptive key frame extraction for video summarization using an aggregation mechanism. *J Vis Commun Image Represent* 23(7):1031–1040
81. Brown RG, Hwang PYC (1992) Introduction to random signals and applied Kalman Filtering. Wiley, New York
82. Han ZJ, Ye QX, Jiao JB (2008) Online feature evaluation for object tracking using Kalman Filter. In: 19th International conference on pattern recognition ICPR 2008, pp 1–4
83. Wang J, Chen X, Gao W (2005) Online selecting discriminative tracking features using particle filter. In: IEEE international conference on computer vision and pattern recognition CVPR'2005, vol 2, pp 1037–1042
84. Han Z, Ye Q, Jiao J (2011) Combined feature evaluation for adaptive visual object tracking. *Comput Vis Image Underst* 115(1):69–80
85. Saffari A, Leistner C, Santner J, Godec M, Bischof H (2009) On-line random forests. In: IEEE 12th international conference on computer vision workshops (ICCV Workshops), pp 1393–1400
86. Grabner H, Leistner C, Bischof H (2008) Semi-supervised on-line boosting for robust tracking. In: 10th European conference on computer vision ECCV '08 Part I, pp 234–247
87. Santner J, Leistner C, Saffari A, Pock T, Bischof H (2010) PROST parallel robust online simple tracking. In: IEEE conference on computer vision and pattern recognition, pp 723–730
88. Kalal Z, Matas J, Mikolajczyk K (2010) P–N learning: bootstrapping binary classifiers by structural constraints. IEEE conference on computer vision and pattern recognition, pp 49–56
89. Felzenszwalb P, Girshick R, McAllester D, Ramanan D (2010) Object detection with discriminatively trained part based models. *IEEE Trans Pattern Anal Mach Intell* 32(9):1627–1645

90. Gall J, Lempitsky V (2009) Class-specific Hough forests for object detection. In: IEEE conference on computer vision and pattern recognition CVPR 2009, pp 1022–1029
91. Maji S, Malik J (2009) Object detection using a max-margin Hough transform. In: IEEE conference on computer vision and pattern recognition, CVPR 2009, pp 1038–1045
92. Godec M, Roth PM, Bischof H (2011) Hough-based tracking of non-rigid objects. In: IEEE international conference on computer vision and image understanding computer vision (ICCV 2011), pp 81–88
93. Rother C, Kolmogorov V, Blake A (2004) GrabCut: interactive foreground extraction using iterated graph cuts. *ACM Trans Graph* 23(3):309–314
94. Ozuysal M, Calonder M, Lepetit V, Fua P (2010) Fast keypoint recognition using random ferns. *IEEE Trans Pattern Anal Mach Intell* 32(3):448–461
95. Villamizar M, Moreno-Noguer F, Andrade-Cetto J, Sanfeliu A (2010) Efficient rotation invariant object detection using boosted Random Ferns. In: IEEE conference on computer vision and pattern recognition, CVPR 2010, pp 1038–1045
96. Godec M, Leistner C, Saffari A, Bischof H (2010) On-line random naive bayes for tracking. In: International conference on pattern recognition ICPR 2010, pp 3545–3548
97. Rodriguez-Sánchez R, García JA, Fdez-Valdivia J (2013) Image inpainting with nonsubsampling contourlet transform. *Pattern Recogn Lett* 34(13):1508–1518
98. Kim S, Shina J, Paik J (2003) Real-time iterative framework of regularized image restoration and its application to video enhancement. *Real-Time Imaging* 9(1):61–72
99. Szeliski R (2006) Image Alignment and Stitching: a tutorial. *Found Trends Comput Graph Vis* 2(1):1–104
100. Eden A, Uyttendaele M, Szeliski R (2006) Seamless image stitching of scenes with large motions and exposure differences. In: IEEE computer society conference on computer vision and pattern recognition, vol 2, pp 2498–2505
101. Ma LM, Wu ZM (2009) Approximation to the k-th derivatives by multiquadric quasi-interpolation method. *J Comput Appl Math* 231(2):925–932
102. Colonnese S, Randi R, Rinauro S, Scarano G (2010) Fast image interpolation using circular harmonic functions. *European workshop on visual information processing EUVIP 2010*, pp 114–118
103. Lee Y, Yoon J (2010) Nonlinear image upsampling method based on radial basis function interpolation. *IEEE Trans Image Process* 19(10):2682–2692
104. Gao Q, Wu Z, Zhang S (2011) Applying multiquadric quasi-interpolation for boundary detection. *Comput Math Appl* 62(12):4356–4361
105. Zhang X, Wu X (2008) Image interpolation by adaptive 2-D autoregressive modeling and soft-decision estimation. *IEEE Trans Image Process* 17(6):887–896
106. Takeda H, van Beek P, Milanfar P (2008) Spatio-temporal video interpolation and denoising using motion-assisted steering kernel MASK regression. In: International conference on image processing ICIP 2008, pp 637–640
107. Ceccarelli M (2007) A finite Markov random field approach to fast edge-preserving image recovery. *Image Vis Comput* 25(6):792–804
108. Li M, Nguyen TQ (2008) Markov random field model based edge-directed image interpolation. *IEEE Trans Image Process* 17(7):1121–1128
109. Nemirovsky S, Porat M (2009) On texture and image interpolation using Markov models. *Sig Process Image Commun* 24(3):139–157
110. Simonyan K, Vatolin D (2009) Edge-directed interpolation in a bayesian frame-work. In: British machine vision conference, vol 10, pp 1521–1527
111. Colonnese S, Rinauro S, Scarano G (2011) Markov random fields using complex line process: an application to Bayesian image restoration. *European workshop on visual information processing EUVIP 2011*, pp 30–35
112. Colonnese S, Rinauro S, Scarano G (2013) Bayesian image interpolation using Markov random fields driven by visually relevant image features. *Sig Process Image Commun* 28(8):967–983