Int. Journ. of Unconventional Computing, Vol. 12, pp. 169–187
Reprints available directly from the publisher
Photocopying permitted by license only

Motion Vector Estimation of Textureless Objects Exploiting Reaction-Diffusion Cellular Automata

MIHO USHIDA[1], ALEXANDRE SCHMID[2], TETSUYA ASAI[1,*],
KAZUYOSHI ISHIMURA[1] AND MASATO MOTOMURA[1]

[1]*Graduate School of Information Science and Technology, Hokkaido University, Kita 14, Nishi 9, Kita-ku, Sapporo 060-0814, Japan*
[2]*Microelectronic Systems Laboratory, Ecole Polytechnique Fédérale de Lausanne, Lausanne CH-1015, Switzerland*

Received: February 8, 2016. Accepted: February 15, 2016.

Conventional motion estimation algorithms extract motion vectors from image sequences based on the image's local-brightness differences in consecutive images. Therefore, motion vectors are extracted along the moving edges formed by moving objects over their background. However, in the case of "textureless" moving objects, motion vectors inside the objects cannot be detected because no brightness (texture) differences exist inside the object. Severe issues may incur in motion-related imaging applications because motion-vectors of vast (inner) regions of textureless objects can not be detected, although the inner part is moving with the object's edges. To solve this problem, we propose an unconventional image-processing algorithm that generates spatial textures based on object's edge information, allowing the detection of the textures motion. The model is represented by a 2-D crossbar array of a 1-D reaction-diffusion (RD) model where 1-D spatial patterns are created inside objects and aggregated to form textures. Computer simulations confirm the approach, showing the formation of textures over approaching objects, which may open applications in machine vision and automated decision systems.

Keywords: Texture generation, cellular automata, reaction-diffusion systems, unconventional image processing

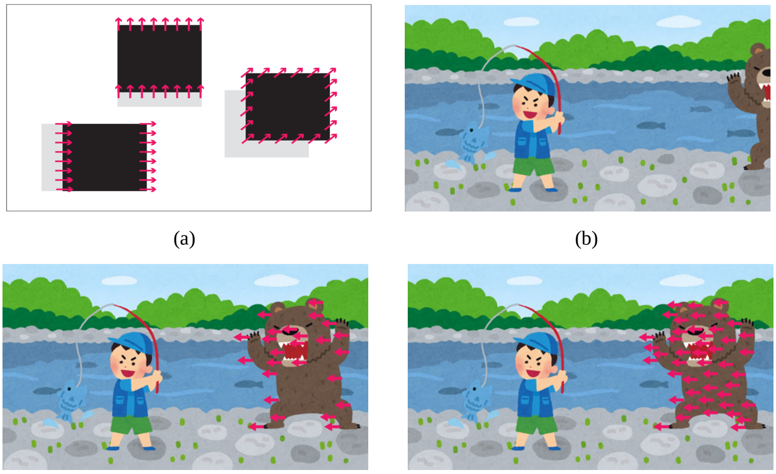
[*] Contact author: E-mail: asai@ist.hokudai.ac.jp

1 INTRODUCTION

Motion estimation has initially been developed as a video compression technique [1]. The limitations of the human visual system have been exploited to enhance performances, specifically accepting a certain level of insensitivity of human observers to spacial or temporal artifacts within a streaming video. Motion estimation has recently been applied in various machine-vision applications, such as anomaly detection, game interfaces, hand gesture user interfaces [2], and image stabilization [3]. This technology has attracted significant attention in recent years, [4–6].

The block-matching method is frequently used in motion vector estimation algorithms to determine the movement of an object in a video sequence [7]. Block-matching is intrinsically based on variations in brightness of identical or shifted regions (blocks) in consecutive images. This constraint becomes an issue when attempting to detect the motion of textureless objects; motion vectors cannot be detected, except at the object boundaries (Figure 1). For example, as illustrated in Figure 1(a), when a completely black board moves over an undetermined background, it is not possible to identify whether the moving object is a board or only its frame (boundary only) using the motion vectors extracted from classical algorithms. Considering the natural scene illustrated in Figure 1(b) at frame n and followed by Figure 1(c) at frame $n + 1$, motion vectors can only be identified from zones of the moving bear which have boundaries towards their surroundings, *i.e.*, the body and some features inside the head; over textureless zones such as the bear belly, it is not possible to identify whether the moving object is the entire object (belly) or only its boundary only using the motion vectors extracted from classical algorithms. As a result, the motion vector field is relatively sparse (Figure 1(c)), which results in a low number of important clues generally necessary for motion classification tasks. In order to extract dense motion vector fields, also the inside of moving objects should be identified as moving and accordingly assigned a motion vector. Specifically, this rule should also apply to textureless object which have an homogeneous luminance information over large zones located inside moving objects. This principle is illustrated in Figure 1(b) at frame n and followed by Figure 1(d) at frame $n + 1$ where a dense motion field of motion vectors is extracted, also inside homogeneous areas such as the bear's belly.

Objects without texture may be many in an image or video sequence, due to incorrect environmental conditions leading to over or under-exposed areas, low image precision encoding (8-bit, or lower), but in majority resulting from the nature of recorded objects themselves, *e.g.*, flat object surfaces (paper, sports ball which appears flat in 2D image acquisition, etc.), uniform backgrounds (wall, curtains, etc.). Generally, these conditions may be expected



(a) (b)

(c) (d)

FIGURE 1
Concept of motion vector extraction. (a) Applying classical motion vector extraction techniques, motion vectors (purple arrows) of textureless objects (black square) that can be detected by conventional motion-estimation algorithms (gray squares depict the initial location of the moving object). Concept of motion vector extraction from a natural scene, (b) frame n of a video sequence, (c) frame $n + 1$ including sparse vectors extraction using classical techniques, and (d) frame $n + 1$ including dense vectors extraction using the proposed reaction-diffusion based technique

in computer vision applications, where the quality of the acquired image can not always be guaranteed, in contrast to the initial movie video applications for which motion estimation algorithms were developed. Among computer vision applications, machine-vision applications have a high potential of growth, considering the expanding market of embedded systems which are equipped with low-cost cameras [8], *e.g.*, driven by cellular phones. In such applications, motion vector fields are used as input to subsequent processing such as classification or creation of information content (tagging) from acquired data, and enabling automated decision-taking processes [9]. Reliably determining the correct motion vector field of each moving object in a scene is a key factor. Application examples include processing of three-dimensional acquired video scenes, such as surveillance and multi-object tracking in busy environments such as city area and road safety monitoring, collision avoidance in mobile systems (automotive or new robotic applications), novel medical imaging aiming at supporting robotic assisted surgery or treatment (tumor irradiation), [10].

In order to be able to determine motion vectors covering the entire surface of a moving textureless object, we propose to assign texture to these objects.

A texture inside an object is expected to follow the object's movement and assist the process of motion estimation. The reaction-diffusion (RD) model is selected to assign texture to textureless objects. The RD is a well-known model for spatial pattern generation [11, 12]. It can be used to simulate the diffusion of chemical activators and inhibitors, and the interactions between activators and inhibitors. These dynamics can generate the stable striped or spotted patterns observed in nature, and covering the surface (skin or fur) of the bodies of animals, for example.

The operation principle, limitations and possible hardware implementation of a RD based pattern generation concept enabling detecting motion vectors on the boundary and the interior of moving objects, are discussed in the following. Section 2 presents and motivates the one-dimensional RD model used in this research. Section 3 presents simulation results, considering different conditions of the background. Finally, Section 4 discusses and summarizes the results.

2 ALGORITHMS AND METHODS

2.1 Reaction-diffusion in a two-dimensional discrete field

The RD process is defined in its fundamental form as operating over a continuous spacial domain. Following the procedure described in [13], the dynamics of the RD process is adapted to support the diffusion of activators and inhibitors in independent discrete fields, which are eventually convoluted into a 2D array of cells. The variation of concentration between activators, u, and inhibitors, v, determines each cell's state at the spacial location (x, y). The diffusion equations are integrated over time δt. Finally, the subsequent state of a cell is determined from the value of the sigmoid function of $(u - v)$. The dynamics of the diffusion process is expressed as follows.

$$
\begin{aligned}
\frac{\partial u(\mathbf{r}, t)}{\partial t} &= D_u \nabla^2 u(\mathbf{r}, t), \\
\frac{\partial v(\mathbf{r}, t)}{\partial t} &= D_v \nabla^2 v(\mathbf{r}, t),
\end{aligned} \tag{1}
$$

where D_u represents the diffusion coefficient of the activators and D_v represents the diffusion coefficient of the inhibitors, $\mathbf{r} = (x, y)$ represents the two-dimensional field, and t is time. The dynamics of the reaction process is expressed as follows.

$$
\begin{aligned}
u(\mathbf{r}, \delta t(n+1)) &= v(\mathbf{r}, \delta t(n+1)) \\
&= f(u(\mathbf{r}, \delta t \cdot n) - v(\mathbf{r}, \delta t \cdot n) - c),
\end{aligned} \tag{2}
$$

$$f(x) = \frac{1}{1+e^{-\beta x}},$$

where n represents the time step, β represents the measure of steepness of the function, and c represents an offset value. The sequential process of diffusion and reaction is named "one update" in the following.

The impulse response of a diffusion equation is expressed as a Gaussian. Consequently, a difference of Gaussian (DoG) represents the impulse response of $(u - v)$. The iterative application of the this DoG at each update governs the formation of striped or spotted patterns in a two-dimensional field, depending on the values of parameters c and (D_u/D_v).

The possibility of applying the RD algorithm as a preprocessing step of motion estimation in real-time embedded applications reflects into specific constraints that must be handled at the algorithmic and hardware levels. In this Section, we consider algorithmic adaptations that are required to support reaction-diffusion applied to consecutive real images extracted from a video streaming input source.

2.2 Reaction-diffusion applied to a video streaming input

The high-level process-flow of the proposed motion estimation system is first presented. We consider video acquisition typically using a commercial imager that consecutively samples image frames. In this study, a standard video acquisition rate is considered at 30 fps (frame per second), while high-speed video acquisition is considered at 1,000 fps. The resulting video stream is sent as successive frames to a preprocessing unit which achieves an iterative image filtering process aiming at creating textures inside textureless objects. In a real-time system, one new result must be delivered by this module at the video acquisition rate. The algorithms and hardware methods applied in this block are the core of this study. Finally, the result of two consecutive textured frames is processed using a classical block matching algorithm which searches the destination location of every block of 3 pixels by 3 pixels from the previous textured image within the eight possible first neighbors plus its unchanged location from the current textured frame. The final result consists of a field of vectors of amplitude limited to one pixel. Hence, as a direct impact, in this research, only a small amplitude of the motion of objects between two frames is tolerated, which translates into constraints in term of the usage of a high-speed camera or a slow motion of objects in the field of view. This constraint has been determined as a proof-of-concept acceptable simplification and does not constitute any intrinsic limitation to the generality of the proposed methods.

Reaction-diffusion algorithms used in still image applications are generally based on a Carthesian coordinate system that matches digital image acquisition, representation and storage, and hence, diffusion is only considered along two orthogonal axis, *i.e.*, four possible spacial directions within a two-dimensional diffusion field. Texture patterns are formed in consecutive applications of the RD algorithm, using the luminance information of objects as the origin of the first RD step (update), and then using the consecutive results as the origin of the next RD updates. Consequently, after stability is reached, striped or spotted patterns have formed around and inside initial objects, and do not evolve in any futher update. The process is non-linear, and is not algorithmically reversible. The dynamics of pattern formation in a one or two-dimensional array governed by RD is detailed in [13] and is not repeated here.

Additional techniques pertaining to the two-dimensional RD process may be considered, such as bilateral filtering of the source image to suppress noise, temporal amplification of the β factor of the sigmoid function, *i.e.*, gradual increase of β in consecutive updates, spacial amplification of β depending on eight surrounding pixels, the consecutive usage of spacial amplification of β followed by the usage of a stable value of β. While each of these technique has its own set of advantages over the others in terms of stability of the patterns, correct generation inside and outside objects, sensitivity to parameters, and hardware resources, none appears to exhibit a determinant benefit in all conditions.

Remarkably, the generation of patterns can be guided. Since the RD operation uses object edges to generates patterns, the presence or insertion of patterns may serve as a way to control the RD pattern generation process of the next frame. In synthetic images, this property can be used to control or limit the generation of patterns to the inside of a selected object, by inserting a striped background (*i.e.*, outside the object) of carefully selected spacial orientation and frequency, for instance, that will prevent the generation of any other pattern, as it corresponds to a stability state of the RD process.

Motion detection finds practical applications from processing consecutive images forming a video stream that are acquired from an image sensor. The modern market of image sensors is dominated by CMOS image sensors, mostly targeting the cheap consumer product market segment. It is thus important to consider non-idealities of the image sensors, which are briefly reviewed in the following, in order to assess their potential adverse effects to the image processing. All types of semiconductor image sensors suffer from systematic and random noise adverse effects resulting from pixel-level non-idealities. Such pixel-level random noise effects include dark-signal-non-uniformity (statistical variation of the photodetector dark current), photoresponse non-uniformity (random pixel gain error), pixel-response non-linearity, pixel temporal noise, and also offset-fixed-pattern-noise (spacial variation of pixel response). Guaranteeing that the random component of CMOS image sensor noise be limited below the quantization level of the imager analog-to-digital converter (ADC) is theoretically and technically possible within reasonable specifications, but is a process that requires trading off with other technical specifications, *e.g.*, frame rate, pixel size or ADC resolution, and which eventually severely impacts the product marketability. In addition, some contribution of the random pixel noise is content-dependent, *e.g.*, variable within dark versus bright zones of an image. In practical terms, even high-quality imagers result from a trade-off in their specifications that may advantage high-frame rate or high spacial resolution to the detriment of thorough pixel-noise suppression. Eventually, most imagers acquire images that are expected to be observed by humans whose eyes can not perceive random pixel noise that may even simultaneously alter many pixels by a few percent in their intensity. Also, this level of noise may not disrupt the majority of machine vision processing, or may be filtered using conventional methods, in case of necessity. The presence of significant residual pixel-noise in commercial imagers can be easily evidenced by subtraction of supposedly identical consecutive still images.

In contrast to human eye processing, two-dimensional RD processing of consecutive images in a video stream proves to be extremely sensitive to pixel noise, even at the lowest possible intensity, *i.e.*, equivalent to one LSB. Pixel-noise diffuses and is amplified by the RD process, which hence propagates additional unwanted effects to neighboring pixels.

The sensitivity of two-dimensional RD to different levels of noise intensities injected into a synthetic image consisting of a square with one point in its center, are presented as an illustrative example in Figure 2. The top-left image shows the stabilized patterns formed inside the square in absence of any pixel noise. All eight other images include pixel-noise at different intensity and different spacial locations, clearly evidencing the development of different patterns. Since pixel-noise is random, the stabilized pattern resulting from the RD process can not be predicted, and clearly depends on the initial object as well as noise.

The adverse effect of pixel-noise inducing unpredictable patterns in consecutive identical image zones is presented in Figure 3, [14]. The fingers only slightly move towards the left from frame 10 to frame 11, conforming the constraint of a high-speed video acquisition (1,000 fps) or a low-speed motion of the objects; in this case, a high-speed camera is used. The resulting motion vector field confirms the correct generation of motion vectors inside the fingers, also including regions of almost homogeneous intensity. The inlet named *finger* is a zoom into a dark border area of the small finger, and confirm the generation of the motion-vector field represented as arrows that generally point to the left. The unwanted effects of noise is observed inside objects

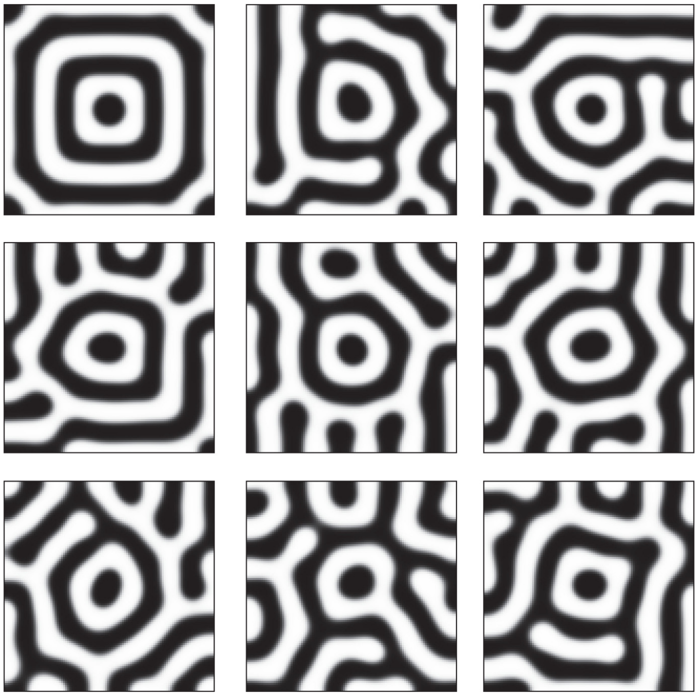


FIGURE 2
Sensitivity of two-dimensional RD. The image located at top-left is generated from an input without noise; the others are generated from an identical input (acquired image), however including different levels of noise

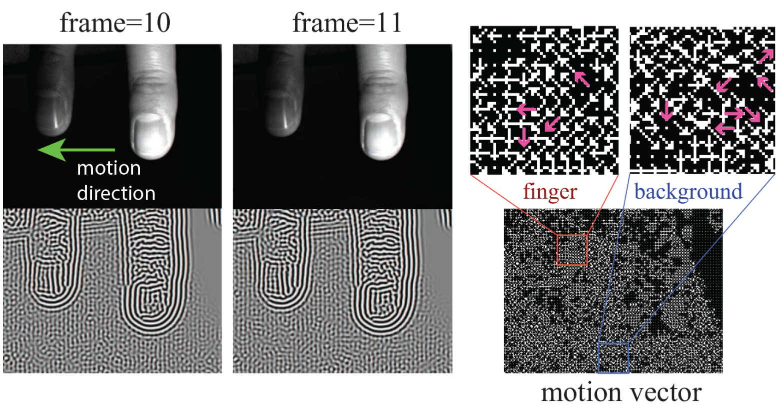


FIGURE 3
Snapshot of two-dimensional RD preprocessing (frame=10 and frame=11), and corresponding motion estimation (motion vector) showing globally correct results inside the finger (finger, red inlet) and globally incorrect results in the background (background, blue inlet); motion vectors are represented as arrows, some of which are highlighted in purple

(fingers), but can be best identified comparing the black background of the two textured images. Due to the presence of pixel-noise, the spotted patterns that develop in the background are different within the two consecutive frames resulting in the unwanted detection of an inhomogeneous motion-vector field by the block-matching algorithm, *i.e.*, consisting of motion vectors apparently pointing into random directions, which is observed in the inlet named *background*. Also, some blocks located inside the fingers are identified to move to the bottom, which is interpreted as incorrect in consideration of the rest of the motion-vector field.

The texture-generation algorithm based on the two-dimensional RD process can be improved taking benefit of the possibility of guiding the generation of patterns by overlaying an existing stabilized pattern over the new image that must be preprocessed. This feature is achieved by adding a weighted version of the pattern resulting from the RD of the previous frame, $n-1$, to the new frame, n, prior to processing to RD of frame n. The blending rate (BR) parameter determines the weight applied to each pixel of the pattern generated at frame $n-1$, and which is added to current frame, n, as a seed guiding the new RD pattern generation.

Motion vector estimation results achieved by adding this preprocessing technique are shown in Figure 4, presenting two examples, namely a finger moving to towards the left (Figure 4(a)), and a toy box approaching the camera from a frontal direction (Figure 4(b)). Two consecutive frames and their RD processing and motion vector extraction are shown in both cases. These examples contain the results of RD processing produced using an identical value of BR= 0.02. Comparing Figure 3 with Figure 4(a) confirms that motion-vector estimation errors in the background are significantly reduced. A high frame-rate, *e.g.*, 1,000 fps is observed to promote stability of the patterns both inside and outside textureless objects. Nevertheless, the analysis of Figure 4 (a) and (b) shows the necessity of adjusting parameter BR with respect to the presence or absence of textures in the images' moving objects and backgrounds. Consequently, the method improves the accuracy of motion vector generation, but proves extremely sensitive to parameter BR, which optimal value is strongly content-dependent. In addition, this method operates in time-domain and requires extra hardware consisting of a frame buffer storage.

Consequently, the presence of pixel-noise in commercial imagers constitutes a fundamental constraint which disqualifies the usage of two-dimensional RD for generating predictable patterns that are suitable to motion-vector extraction. Clearly, the algorithms must be adapted to support some level of imperfection of commercial image sensors, since expecting image sensors exempt from any random noise is not reasonable.

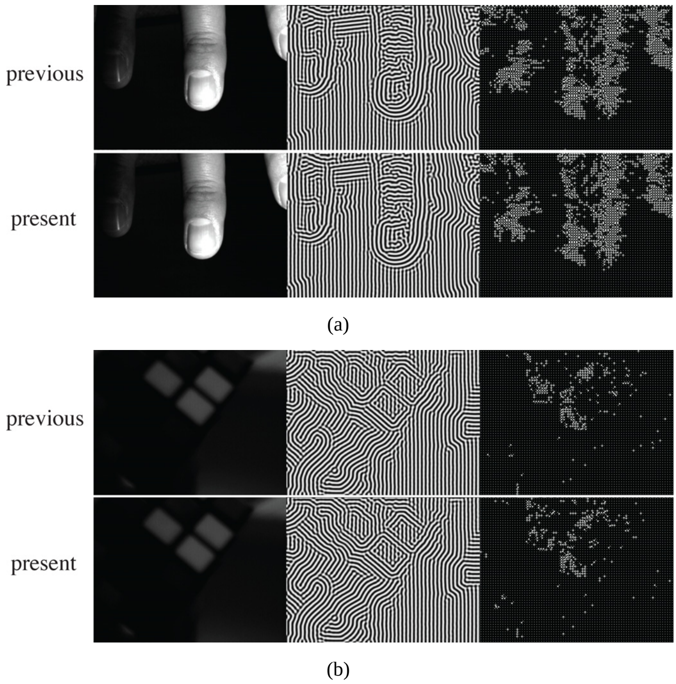


FIGURE 4
Snapshot of two-dimensional RD preprocessing and motion estimation, using processing by addition of the previous frame RD result. Left image: captured image, central image: RD image, and right image: motion vector field, similar to the enlarged vectors in Figure 3, right. Two examples are presented, and for each, the results of two consecutive frames are shown. (a) fingers moving towards the left, (b) a toy box approaching in front

2.3 One-dimensional reaction-diffusion

As observed in Figure 2, diffusion simultaneously occurs in all directions in a two-dimensional field, thereby creating complex patterns. A one-dimensional-based RD process for motion estimation is presented in the following as a method enabling the control of noise diffusion.

One-dimensional RD processing of an image prescribes that the RD process be independently applied to orthogonal image directions. In a two-dimensional RD process, diffusion occurs in a non-restricted two-dimensional diffusion field. In contrast, in a one-dimensional RD process, diffusion is limited to a two-way diffusion field. The equations governing the one-dimensional algorithm are Equation 1 and Equation 2, but in the one-dimensional case $\mathbf{r} = x$. This limitation also applies to pixel-noise components. As a result, the unwanted pixel-noise contribution is constrained in a more severe manner in a one-dimensional RD process, where options to

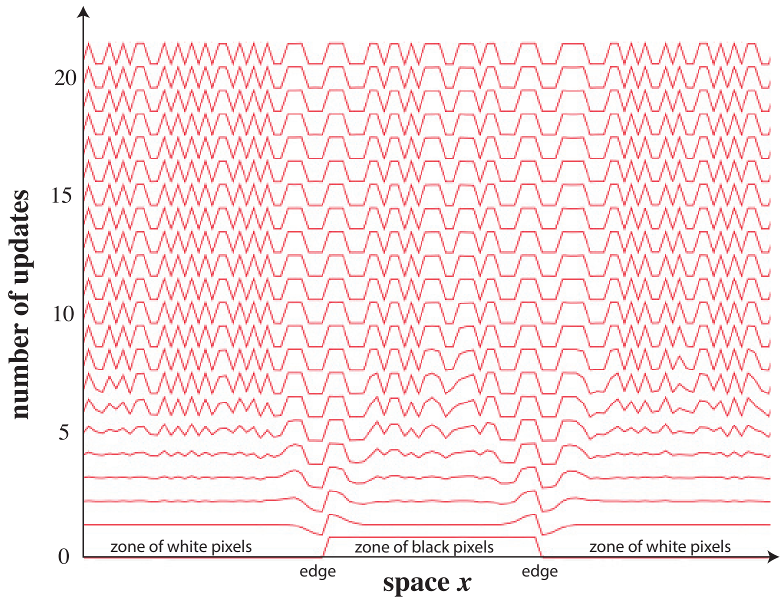


FIGURE 5
Development of a one-dimensional RD waveform without additional filter processing

create new unwanted stripes or spots are limited by the smaller number of potentially participating or available neighbors, and the existence of legitimate stripes, corresponding to neighbors which are not available to participate in the formation of an unwanted stripe. Intuitively, pixel-noise has less opportunity to diffuse in a zone that offers premises for developing an unwanted stripe or spot.

In order to support the one-dimensional RD method, each row of an image is independently processed by the algorithm. Since movement may occur into any direction, columns must also be considered following the same independence criterion. Eventually, the results may be aggregated when stability is reached.

Hence, one-dimensional RD has a significant merit in its inherent control and limitation of noise diffusion. A second merit lies in a less complex hardware implementation. In the following, we describe the one-dimensional process, its control of noise as the adapted version of RD to be used for motion-vector generation.

A one-dimensional RD process is shown in Figure 5, where the horizontal axis represents the pixels in one row of an image, while the vertical axis presents the update number which is considered until stability is observed reached, *i.e.*, at or prior to update 21. The image source of the row presents a black stripe located in the middle of the row over a white background including very light pixel-noise. Following RD theory [13], edges initiate the propagation of stripes, which is visible from update 1 through update 6. However, very clearly observable from update 4, noise impacts on the proper generation of further stripes, creating an inhomogeneous pattern including high-amplitude high spacial frequencies. In addition, in absence of noise, the one-dimensional RD process appears very robust and generates stable patterns which may in turn cause inconsistent conditions, specifically in regions where stripes generated by different object edges get in close vicinity without matching the spacial phase of each other. Expressed with more detail, a step function is expected to generate a stable pattern consisting of a square wave representing black and white pixels. The stable spacial frequency of the wave is entirely determined by the parameters of the system, and can be derived following the procedure explained in [13]. In a real image case where several edges are expected in the image, several waves are created and interfere. For example in Figure 5, two edges separate the central black zone with respect to the white surrounding background. Consequently, the waves of fixed spacial frequency meet in the middle of the of the black zone. Generally, the wave generated from the left-hand size and right-hand side edges are likely out of phase, creating an inconsistent condition where interference manifest itself as an oscillation of high spacial frequency. Following an identical process, waves propagating along the white zones are also likely to collide and interfere due to their phase difference and to the existence of a boundary condition that dictates continuity between the left-end and right-end of a row. This represents the most likely situation, unless synthetic images have been specifically prepared to avoid the issue. Consequently, an additional filtering step is required which aims at inhibiting the influence of noise, and also create a temporal break of stability to let the patterns reorganize in a way that cancels inconsistencies. An additional diffusion filter is inserted into the process flow; the diffusion has no subtraction or amplification. The filter is applied every third iteration of the RD process, starting from the first update, enabling the development of a regular pattern where the influence of noise has vanished, as presented in Figure 6.

The algorithm proposed in this paper processes a two-dimensional input image. The image is first reorganized into a one-dimensional format, namely consisting of x (horizontal, rows) and y (vertical, columns) arrangements. These arrangements are then repeatedly processed by one-dimensional RD. Finally, they are multiplied with each other. Subsequently, block matching may take place, and estimates the motion vectors from the preprocessing results of consecutive frames.

Simulation results obtained using the proposed algorithm after stability is reached are shown in Figure 7. The input image includes noise. Figure 7 (a)

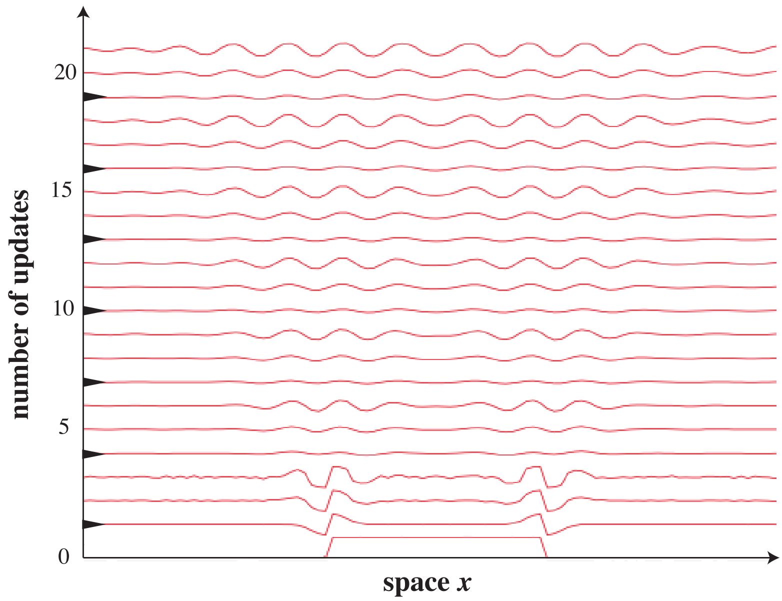


FIGURE 6
Development of a one-dimensional RD waveform with additional filter processing every third updates. An arrow on the vertical axis indicates the updates requiring filter application (strict diffusion), while all other updates include RD

shows the result without additional diffusion filter processing. Clearly, the generated texture is disordered by the influence of noise, in spite of using a one-dimensional RD process. Figure 7 (b) shows the result with additional diffusion filter processing. An ordered texture has been generated while noise has vanished, demonstrating a clear improvement over the result of Figure 7 (a).

3 SIMULATION RESULTS

The suitability of the proposed algorithm to generate stable and noiseless patterns is shown in Section 2. Section 3 discusses its applicability in real conditions, using computer simulations. In the following, simulation results considering various background conditions are explored and discussed.

3.1 Uniform background

Simulation results considering a uniform (black) background and a white square moving object are shown in Figure 8, at three instants, each separated by ten frames. Texture is generated from the boundary of the square.

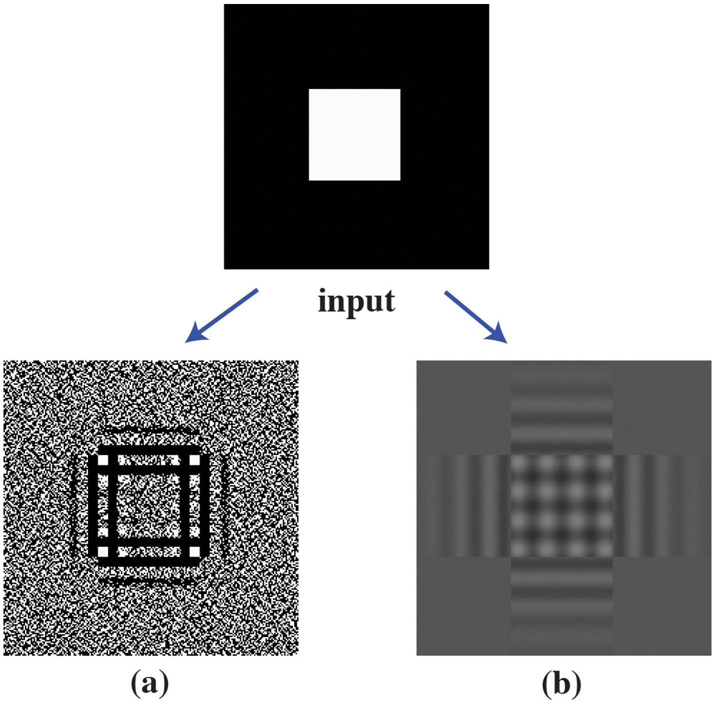


FIGURE 7
Difference in the texture generated by the proposed algorithm consisting of two one-dimensional RD processes, (a) without additional filter processing and (b) with additional filter processing

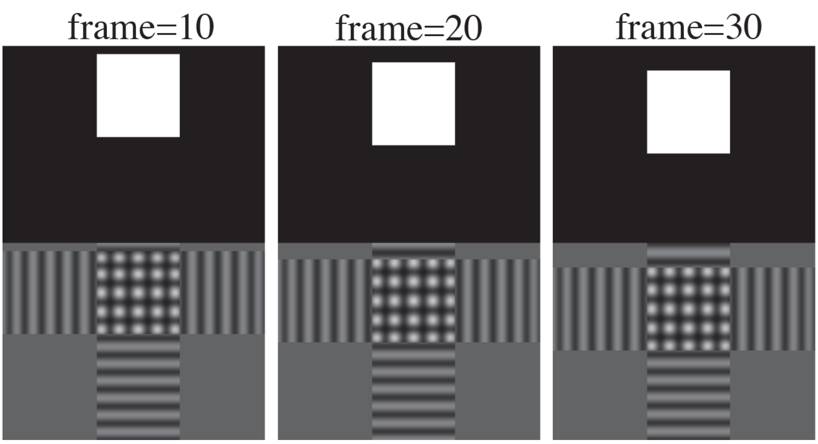


FIGURE 8
Snapshots in the case of a uniform background taken at frames no. 10, 20, and 30; the textureless object slowly moves into the bottom-right direction

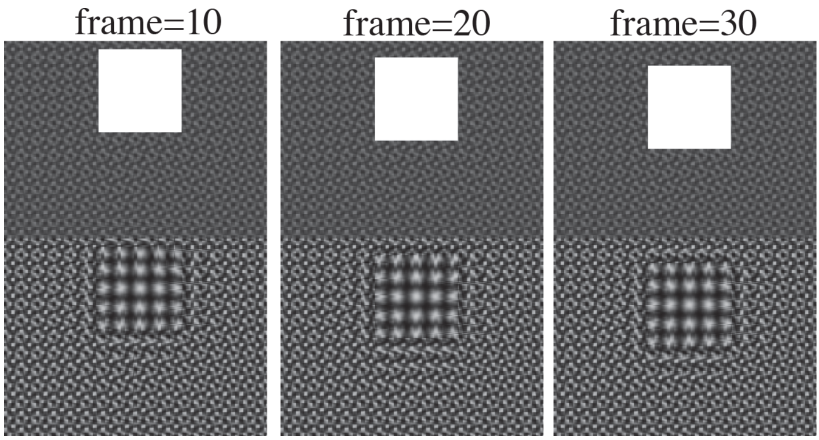


FIGURE 9
Snapshots in the case of an ideal stable background

However, the texture not only expands inside the object, but also outside the boundaries. This result appears trivial keeping in mind that the definition of the moving object, or similarly the interpretation of the inside and outside of an object are only possible with prior knowledge, under the conditions of uniform object over a uniform background. The generation of patterns outside the object leads to errors in motion detection.

3.2 Ideal background

Simulations conducted under the condition of a uniform object moving over a patterned background are considered. Simulation results are shown in Figure 9. An ideal background is defined from its capacity to block the generation of patterns, thereby enabling an algorithm to differentiate an object, or a moving object from its background. Consequently, an ideal background can be obtained from a any pattern that has developed into a stable state using the RD algorithm. As an example of an ideal background, a cross-stripe image is generated using the proposed algorithm until the generated texture is stable. Clearly, the expansion of texture outside the moving object is observed to be inhibited. Thus, under this type of ideal conditions, the proposed algorithm is expected to provide excellent inputs to the block-matching unit.

3.3 Real background

Texture generation simulations considering a real background are conducted to confirm the practicality of the proposed algorithm. Simulation results using a bookshelf as the real background over which a synthetic uniform, *i.e.*, textureless, black square is moving are shown in Figure 10. In this case, the

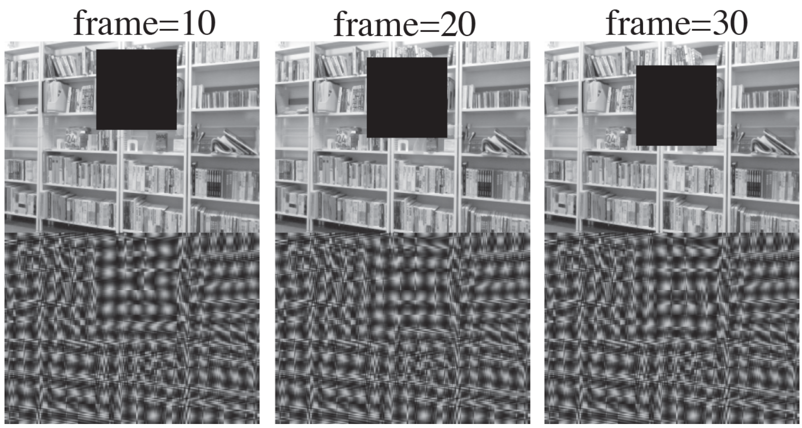


FIGURE 10
Snapshots in the case of a real background

outside expansion of texture is inhibited. Stable texture is generated inside the object boundaries, which is shown in the next Subsection.

3.4 Motion estimation

The optical flow function available from the OpenCV library is used as a motion estimation processing [17]. The results of motion estimation with a real background image are shown in Figure 11. The images located on the

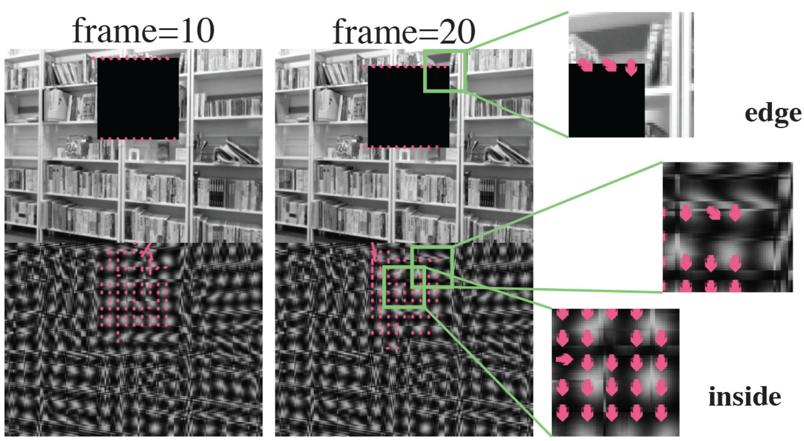


FIGURE 11
Snapshots of motion estimation considering a real background, at two frames. The upper images and *edge* inlet result from a classical motion vector estimation and yield in a sparse motion vector field located at the edges of the moving object. The lower images result from RD preprocessing prior to carry out motion vector extraction, and yield a dense motion vector field

upper position that are labeled *edge* and in which the bookshelf is easily distinguished present results obtained without RD preprocessing. Motion vectors are only detected on the boundary of the square. The images located on the lower position that are labeled *inside* and in which the bookshelf can not be distinguished present results obtained using the proposed RD preprocessing.

The motion vector field is detected on the boundary and extends inside the textureless moving object. Furthermore, the direction of the motion vectors is almost correct is all cases.

The proposed algorithm thus presents a significant improvement in the quality of the generated motion vector field, specifically also in the case of real world sequences that have a sufficient texturing level, and over which the textureless object may move.

From a theoretical point of view, the proposed RD-based texture generation inside moving objects is only suitable to objects translating in front of the camera, which may be a severe limitation of the method. Nevertheless, in practical terms, respecting the condition of high-speed image acquisition, or equivalently the condition of a low-speed moving object has been observed to yield very small differences in consecutive images. As a benefit, (moving) approaching objects as well as (moving) tilting objects have also been observed to be supported by the proposed method. The strict conditions of success can be expressed in terms of speed of approach or rotation over oneself, with respect to the camera acquisition rate and have not yet been studied.

The proposed one-dimensional algorithm applied to two-dimensional video sequences is suitable to an efficient hardware implementation. In prior research, the two-dimensional RD algorithm [15] has been implemented on digital hardware [16]. Based on this prior art, an efficient hardware implementation of the one-dimensional RD model can be considered.

4 SUMMARY AND DISCUSSION

In this study, a texture generation algorithm that uses RD processing is presented to enable motion estimation on the inside of the boundary of textureless objects by detecting the motion of the generated texture. The nature of the RD process that is applied to natural images and images acquired from imperfect cameras dictates the usage of a one-dimensional RD process, that is applied in two perpendicular directions in independent steps. The results obtained are finally combined and the textured images delivered to a motion vector estimation algorithm based on the optic flow detection. Simulation results emphasize the necessity of a suitable, *i.e.*, textured background to avoid the expansion of generated texture outside the moving object. Motion vectors could be reliably determined from texture generated inside moving objects under the condition of an appropriate background.

The reliable and correct determination of motion vector fields has not been considered in video processing prepared for human auditors, but it is expected to become a necessity in future machine vision based automated decision systems. The classification of patterns consisting of motion vectors has been carried out in simulation confirming the possibility of using the vector fields generated using the proposed method. In this case, cognitive classification has been achieved using a machine learning algorithm, [18]. A perceptron-based classifier has been used to assess whether an object moves to the right. Preliminary results using the real image situation confirm significant improvement of the time of convergence of the neural network using the RD texture generation, as a benefit of the increased number of motion vectors. Hence, the performance of machine learning decision-based vision systems appears increased by extending its input information content, in this case consisting of a dense motion vector field.

Finally, the proposed method has low memory requirements, and is thus suitable for a future integration into a hardware platform.

ACKNOWLEDGMENTS

This study was supported by a Grant-in-Aid for Scientific Research on Innovative Areas [2511001503] from the Ministry of Education, Culture, Sports, Science and Technology (MEXT) of Japan.

REFERENCES

[1] A. j. Tabatabai and Radu S. Jasinschi and T. Na Veen, Motion Estimation Methods for Video Compression A Review, *Journal of the Franklin Institute*, Vol. 335, No. 8, pp. 1411–144, 1998.

[2] M. Tanaka, Japan patent Kokai JP2014-52934A, 2014.

[3] Y. Takagi, Japan patent Kokai JP2012-15959A, 2012.

[4] X. Liyin, S. Xiuqin, Z. Shun, A review of motion estimation algorithms for video compression, 2010 International Conference on Computer Application and System Modeling (ICCASM), pp. V2-446-V2-450, 2010.

[5] M. Jakubowski, G. Pastuszak, Block–Based Motion Estimation Algorithms – A Survey, *Springer Opto–Electronics Review*, Vol. 21, No. 1, pp. 86–102, 2013.

[6] N. K. Parmar, M. H. Sunwoo, Recent Progress on Block-Based Motion Estimation Techniques, *IETE Technical Review*, Vol. 32, No. 5, pp. 356–363, 2015.

[7] L.-M. Po, W.-C. Ma, A novel four-step search algorithm for fast block motion estimation, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 6, No. 3, pp. 313–317, 1996.

[8] R. Fontaine, The State-of-the-Art of Mainstream CMOS Image Sensors, 2015 International Image Sensor Workshop (IISW), Introductory Paper, Vaals, The Netherlands, 2015.

[9] M. Mori, T. Itou, M. Ikebe, T. Asai, T. Kuroda, M. Motomura, FPGA-Based Design for Motion Vector Estimation Exploiting High-Speed Imaging and Its Application to Motion Classification with Neural Networks, *Journal of Signal Processing*, Vol. 18, No. 4, pp. 165–168, 2014.

[10] M. P. Vijaykumar, A. Kumar, S. Bhatia, Latest Trends, Applications and Innovations in Motion Estimation Research, *International Journal of Scientific & Engineering Research*, Vol. 2, No. 7, pp. 1–6, 2011.

[11] J. D. Murray, Mathematical Biology I: An Introduction (3rd Ed.), Chap. 7, p. 239, Springer, New York, 2002.

[12] M. Gerhardt, H. Schuster, A cellular automaton describing the formation of spatially ordered structures in chemical systems, *Physica D*, Vol. 36, pp. 209–221, 1989.

[13] Y. Suzuki, T. Takayama, I. Motoike, and T. Asai, Striped and spotted pattern generation on reaction diffusion cellular automata: Theory and LSI implementation, *Int. J. Unconv. Comput.*, vol. 3, pp. 1–13, 2007.

[14] M. Ushida, K. Ishimura, T. Asai, and M. Motomura, A reaction-diffusion algorithm for texture generation towards motion-vector estimation of textureless objects, 2015 RISP International Workshop on Nonlinear Circuits, Communications and Signal Processing, Universiti Teknologi Malaysia, Kuala Lumpur, Malaysia, pp. 361–364, 2015.

[15] K. Ishimura, K. Komuro, A. Schmid, T. Asai, and M. Motomura, Image steganography based on reaction diffusion models toward hardware implementation, *Nonlinear Theory and Its Applications*, vol. 5, no. 4, pp. 456–465, 2014.

[16] K. Ishimura, K. Komuro, A. Schmid, T. Asai, and M. Motomura, FPGA implementation of hardware-oriented reaction-diffusion cellular automata models, *Nonlinear Theory and Its Applications*, vol. 6, no. 2, pp. 252–262, 2015.

[17] G. Bradski, A. Kaehler, Learning OpenCV: Computer vision with the OpenCV library, O'Reilly Media, Inc., 2008.

[18] T. Itou, M. Mori, M. Ikebe, T. Asai, T. Kuroda, M. Motomura, A new architecture for feature extraction to perform machine learning by using motion vectors and its implementation in an FPGA, Proceedings of the 2015 RISP International Workshop on Nonlinear Circuits, Communications and Signal Processing, Kuala Lumpur, Malaysia, pp. 294–297, 2015.