2015 RISP International Workshop on Nonlinear Circuits,
Communications and Signal Processing (NCSP'15)
Kuala Lumpur, Malaysia, February 27 - March 2, 2015

A Reaction-Diffusion Algorithm for Texture Generation to Enable Motion Vector Estimation of Textureless Objects

Miho Ushida[1†], Kazuyoshi Ishimura[1], Alexandre Schmid[2], Tetsuya Asai[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, Swiss Federal Institute of Technology (EPFL)
Lausanne CH-1015 Switzerland
[†]Email: ushida@lalsie.ist.hokudai.ac.jp

Abstract

We propose a preprocessing technique using a reaction-diffusion (RD) model to enable motion vector estimation of textureless object surfaces. Because the motion of pixels between frames is relatively small in high-speed imaging, motion vectors of an object are detected by applying a classic block matching algorithm in small areas. Classic block matching methods only detect motion vectors for the boundaries of objects that have minimal texture information. Therefore, as a preprocessing step, we add an RD model that creates a texture on the object. As a result, the motion vector of an object's surface can be estimated. However, independent (of other frames) RD processing causes a flicker (noise) in the generated texture and background. Consequently, estimation errors occur. To resolve these problems, we propose a method to remove noise and improve preprocessing using an RD model.

1. Introduction

Motion vector estimation is an image processing technology applied in a wide range of fields; for example, MPEG motion compensation for moving image compression, dangerous object detection using an onboard camera in a vehicle [1], and so on.

The block matching method is frequently used in motion vector estimation algorithms. Using this method, calculation time increases as the search range between neighboring frames is extended. However, when high-speed imaging is used, the movement of an object between frames will decrease. Thus, it becomes possible to estimate motion vectors by searching over a small range. However, when motion vectors of a textureless object are estimated, motion vectors are estimated only for the boundary of the object; therefore, it is not possible to precisely estimate the motion vectors of the object's surface.

We focused on the dynamics of organic pattern generation, and designed a preprocessing technique using an RD model [2], to address the limitations of the block matching method. This novel algorithm executes the RD process for each frame sent from the camera module. Using this technique, self-organizing patterns form in the image areas that have moving textureless objects. By applying block matching to neighboring frames processed in this manner, it is possible to estimate motion vectors of not only the boundaries of an object, but also surfaces, because of the texture that has been generated.

2. Proposed algorithm

2.1 Reaction-diffusion model

We implemented texture generation as a preprocess, using an RD cellular automata (CA) model [2]. Dynamics of an RD model are typically described using partial differential equations; however, an RD-CA model discretizes space and simplifies ranges where pixels interact to the nearest four neighboring pixels. As shown in Fig. 1 (a), we can easily implement the dynamics of this model with the blurring filter kernel using the nearest four neighboring pixels, a temporary memory location that stores the blurring processing results, and a sigmoid function. As shown Fig. 1 (b), the blurring filter kernel of the nearest four neighboring pixels at a_c is described as

$$\text{kernel out}_{a_C} = \frac{(a_N + a_E + a_W + a_S + 4a_C)}{8} \qquad (1)$$

The filter kernel scans from the upper left of the image toward the right, one line at a time (Fig. 1 (c)). The blurring intensity of the entire image increases as a result of the scanning iterations. First, after eight iterations, the results are temporarily stored to generate self-organizing patterns. This corresponds to the diffusion of the activators. Subsequently, eight additional blurring iterations are performed. This corresponds to

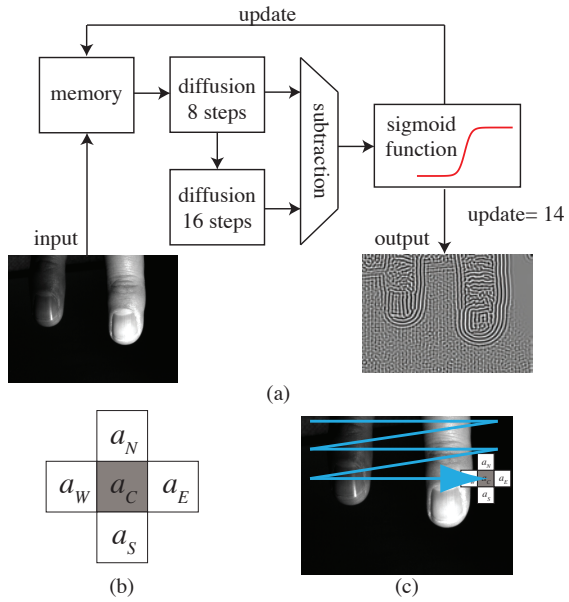

Figure 1: Reaction-diffusion module: (a) proposed algorithm for motion vector estimation, (b) a filter kernel, and (c) a kernel scan.

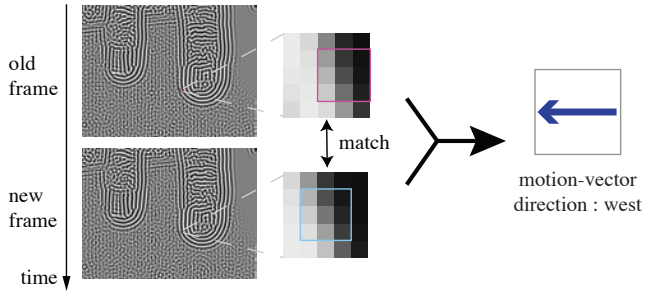

Figure 2: A motion vector estimation.

the diffusion of the inhibitors. After computing the difference of these values, amplifying it by a sigmoid function corresponds to the reaction. We define this series of processes as the update process. Textures are generated by repeating the update process for the obtained image (Fig. 1 (a)).

2.2 Block matching method

The block matching method estimates the motion vectors of each block by dividing an image into small, equally sized regions, and comparing the block with the same block from a previous frame. By exploiting high-speed imaging, it is possible to narrow block matching search ranges, because the motion of a pixel between neighboring frames will be relatively small.

We generate self-organizing patterns for each high-speed imaging frame using an RD model; subsequently, we estimate

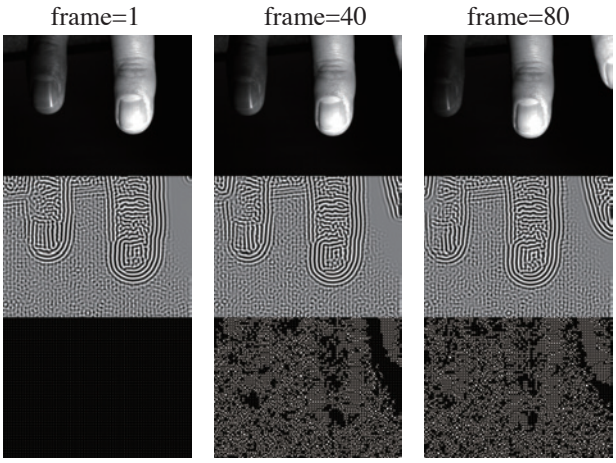

Figure 3: Snapshots of camera module outputs, self-organized patterns with RD, and motion vectors.

motion vectors by applying block matching (Fig. 2).

2.3 Simulation result

A snapshot that contains estimated motion vectors that were produced using the proposed algorithm is shown in Fig. 3. These show, from the top, a source image, the results of RD, and the results of motion vector estimation. As planned, texture was generated on the fingers, which were moving objects; as a result, we can confirm that the generated texture was largely maintained, although the objects moved. However, motion vectors were estimated in the entire image, although only the fingers moved; consequently, many estimation errors occurred. The texture flicker generated by RD, and the noise caused by light reflections, are assumed to be the causes of these estimation errors. To reduce these errors, we devised a method of removing noise before executing RD processing, and tested it using a simulation. We describe the solution and results in the following chapter.

3. Solution

3.1 Addition of old frame

We utilized high-speed imaging to inhibit the flicker of the generated texture, which was one of the causes of the estimation errors. In high-speed imaging, the motion of a pixel between frames is relatively small. RD processing was executed after the addition of an image obtained from RD processing for a previous frame, as an afterimage for the following frame (Fig. 4). Using parameter BR, we adjusted the degree to which we leave afterimages. As a result, it became possible to generate new self-organizing patterns (depending on patterns generated in the previous frame), and to inhibit flicker in the image.

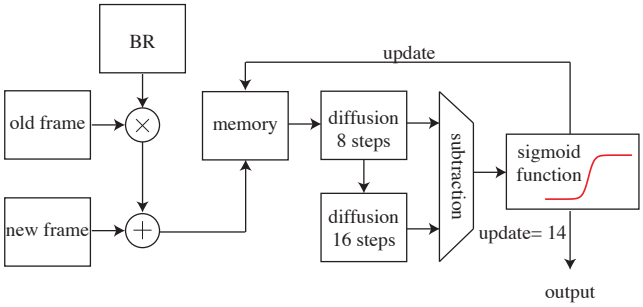

Figure 4: Algorithm with addition of previous frame processing

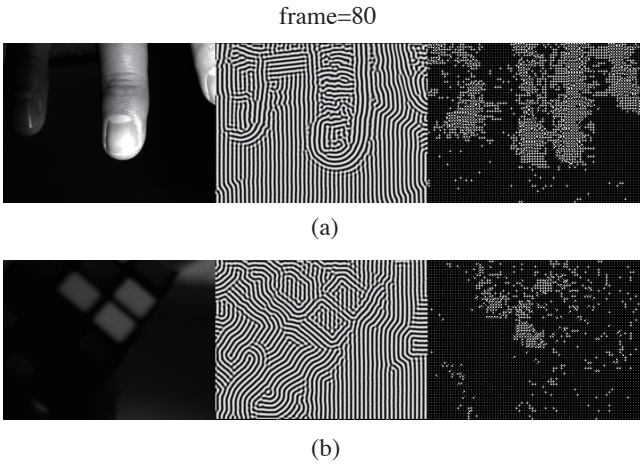

Figure 5: Snapshots from simulation with addition of previous frame processing: (a) fingers move toward the left, (b) a toy box approaches in front.

Motion vector estimation results achieved by adding this preprocessing technique are shown in Figs. 5 (a) and (b). These two figures contain the results of RD processing produced using the same parameter (specifying how much of the previous frame will be added). By comparing Fig. 3 with Fig. 5 (a), we can confirm that estimation errors in the background became lighter when the preprocessing technique was executed. However, as we can observe when comparing Fig. 5 (a) with (b), it is necessary to adjust the parameter to estimate motion vectors, depending on the presence or absence of textures in the image's moving objects and backgrounds.

3.2 β differentiation depending on variance

Subsequently, we employed variance to prevent estimation errors resulting from noise. Variance in textureless regions results in a value near zero. However, when noise appears in regions where texture did not exist originally, pixel values vary and the variance has a value that is not zero (Fig. 6). Using this information, it adjusts β, the inclination of a sigmoid

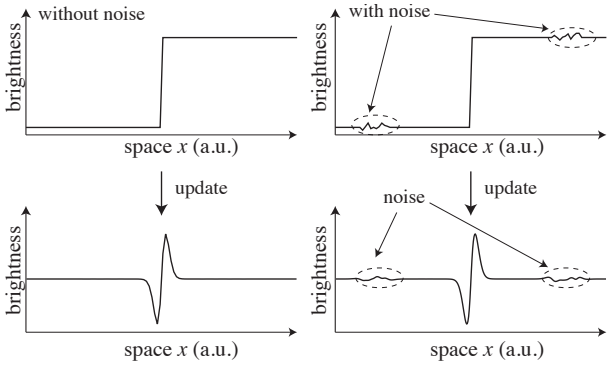

Figure 6: The relation between noise and variance.

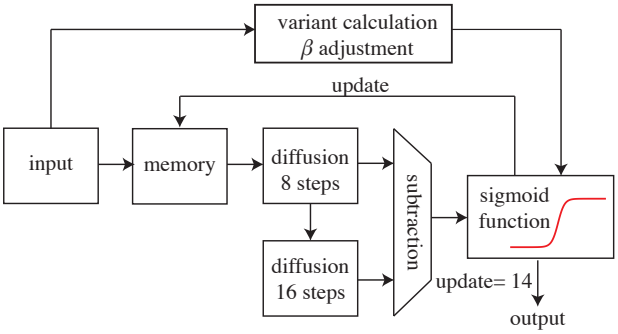

Figure 7: Algorithm with β differentiation processing.

function in RD processing, depending on the variance. We set the threshold value of the variance to 0.0005, decreased β in regions where the variance was smaller than the threshold value, and increased β in regions where the variance was larger. These steps were taken to curb the amplification of noise (Fig. 7). Results of motion vector estimation achieved by adding this processing are shown in Fig. 8. By comparing Fig. 5 (a) with Fig. 8 (a), we can confirm that motion vectors were only estimated on the fingers, and that we could prevent the extension of textures generated from background noise.

3.3 two-stage RD processing

In the processing described in the previous section, we could repress noise. However, because of the relatively small surface variance of the textureless object for which we originally attempted to generate a pattern, the texture generation was curbed, and motion vectors were only estimated for the boundary of the object. To address this problem, we devised the solution shown in Fig. 9. First, RD processing is executed using β depending on variance, as we proposed in the previous section, for each pixel a certain number of times. Noise is curbed here, and textureless regions in the image are smoothed. Using an obtained image as a source, RD processing is executed a certain number of times, with constant

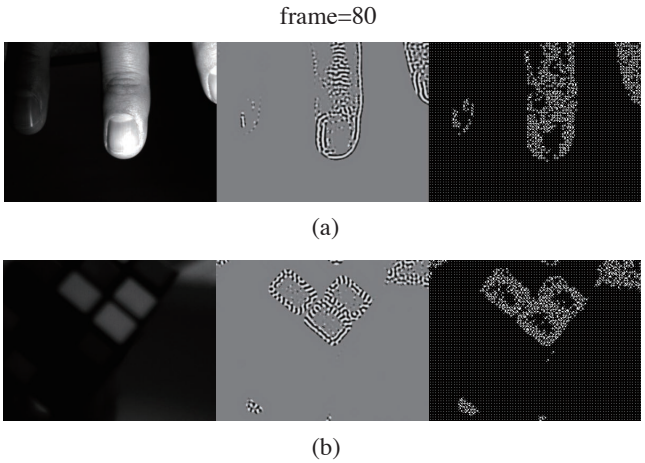

Figure 8: Snapshots from simulation with β differentiation processing: (a) fingers move toward the left, (b) a toy box approaches in front.

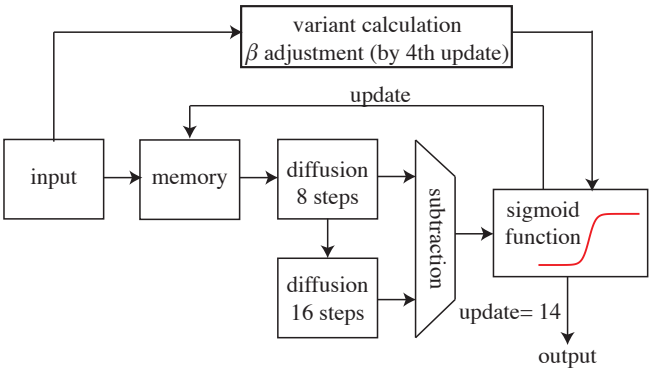

Figure 9: Algorithm with two-stage RD processing

β. As a result, the texture is expanded from the boundary of the object to the textureless surface, and the motion vector is estimated. Motion vector estimation results achieved using this processing technique are shown in Fig. 10. By comparing Fig. 8 (a) with Fig. 10 (a), we can confirm that a texture was generated on the surface of the finger nail (where texture was not generated by the process in the previous section), and that the motion vector was estimated. However, naturally, the texture also expands to the background outside of the object; therefore, we cannot conclude that the motion vector is precisely estimated. Results of a simulation where a textureless object moves in the background are shown in Fig. 10 (b).

4. Summary

In this study, we proposed a technique to generate textures on textureless objects using RD processing as a preprocessing step, and to estimate motion vectors not only for the boundary

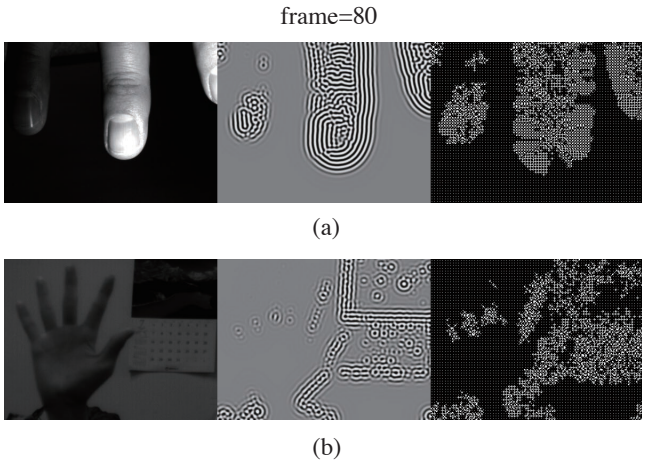

Figure 10: Snapshots from simulation with two-stage RD processing: (a) fingers move toward the left, (b) hand approaches in front.

of the moving object, but also its surface when we use block matching.

At first, when only RD processing is executed as a preprocessing step, estimation errors occurred because of flickers from the generated texture and noise caused by light reflections. To resolve these problems, we devised a method of inhibiting the texture flicker by adding RD results from a previous frame at a certain rate, and a method of reducing optical noise by differentiating β in a sigmoid function used in RD, depending on variance. We then tested the methods in a simulation. In the results, we could reduce texture flicker and noise. However, all of these methods required parameter adjustments depending on the presence or absence of textures on the background and moving objects; therefore, the success or failure of the motion vector estimation depended on these adjustments. In future work, we will establish more general parameters, devise enhanced preprocessing techniques, and perform additional simulations, so that we can more accurately generate self-organizing patterns from the surfaces of moving textureless objects and estimate motion vectors when various moving images are input.

References

[1] MORI, Masafumi, et al. “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).

[2] SUZUKI, Youhei, et al. “Striped and spotted pattern generation on reaction-diffusion cellular automata Theory and LSI implementation,” Int. J. Unconv. Comput., vol. 3, pp. 1–13, 2007