Fifth International Workshop on Computational Intelligence & Applications
IEEE SMC Hiroshima Chapter, Hiroshima University, Japan, November 10, 11 & 12, 2009

Real-Time Visal Motion Detection by Spatiotemporal Energy Model Implemented on GPU

Akitoshi Hanazawa
Kyushu Institute of Technology, LSSE, Department of Brain Science and Engineering
Hibikino 2-4, Wakamatsu, Kitakyushu, Fukuoka 808-0196, Japan
email: hanazawa@brain.kyutech.ac.jp

Abstract — **The aim of this study is to develop a real-time visual motion detection system by using physiologically meaningful image processing algorithm. Spatiotemporal energy model has been recognized as the most plausible algorithm corresponding to the jobs in motion detection performed by simple and complex cells existing in area V1 of cats or macaque monkeys. Because of the parallelism of the brain, this algorithm inherently has high parallel performance. Together with the locality, spatiotemporal Gabor filtering and succeeding energy extraction process fit with the architecture of present GPU (Graphic Processing Unit). Enabling real-time motion detection at each pixel location over the entire input image is fundamental in many applications as for instances in robotics vision and car-mounted camera. This system, moreover, is open for further expansion based on the physiological knowledge about mammalian visual system.**

I. INTRODUCTION

Application of the mathematical models of mammalian visual information processing on artificial image processing has been fairly attended but faintly executed because of their expensive computational cost. Pixelwise parallel processing, the fundamental policy of the visual system, causes the heavy load of serial processing systems. Spatial 2D Gabor filtering regarded as exemplifying such a model applicable to technology also has such difficulty in the segment requiring real–time system, such as robotics vision and car-mounted camera. In these situations in which cameras are moving around, detection of visual motion direction is crucial for the recognition of environmental 3D structures, objects, self-motion, and etc. Spatiotemporal 3D Gabor filter is a model of a type of front-end cells in the visual cortex, that is, simple cell sensitive to visual motion direction existing in area V1 of the visual cortex. This filter is more time consuming than 2D Gabor filter. Although it is possible to save processing time by some programming technique or simplification, it is useless for real-time system as far as applying for VGA resolution image sequences. Among several schemas that have possibility to overcome this limitation, GPGPU (General Purpose computation on GPU) technology looked attractive because of its balance between the number of GPU processing unit and their functional ability. Actually it is reported here that spatiotemporal energy model, that is, spatiotemporal 3D Gabor filtering followed by energy extraction process, is executable in real-time by implementing on GPU, meaning that an obstacle on the path through the border of vision research science and image processing technology is removed. The priority of this paper, therefore, is to show the potential of brain-like parallel image processing through GPGPU acceleration.

II. ALGORITHM

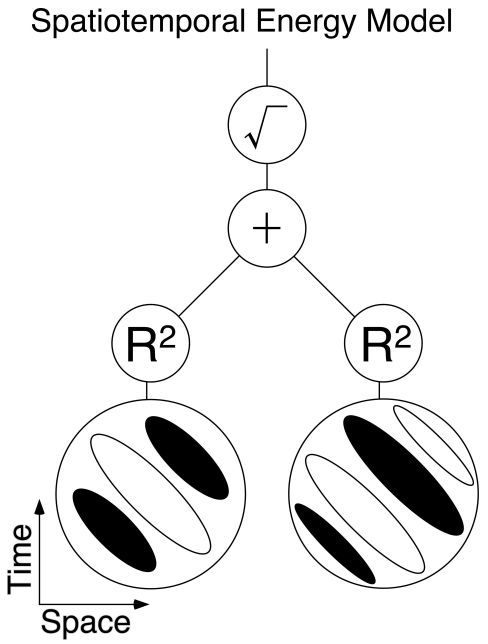

Fig. 1 Diagram of spatiotemporal energy model. Bottom circles represent a quadrature pair of spatiotemporal Gabor filters. Outputs of the filters are transformed into energy expression.

Spatiotemporal energy model shown in Fig. 1 has been proposed as the model of motion sensitive simple and complex cells in area V1 of the visual cortex [1]. For the visual system, 2D visual images are continuously supplied and constitute 3D continuum of visual information. Motion detection can be formulated as detecting the 3D orientation of edges in the spatiotemporal 3D space. This can be achieved by using some 3D convolutional filter followed by some position or phase invariant integration. The visual system employs 3D Gabor as convolution kernel (Fig. 2) for simple cells [2][3] and phase invariant energy expression for complex cells, which is shown in upper part of Fig. 1 where

outputs of the spatiotemporal filters in quadrature phase relationship are squared (R^2), added (+) and rooted ($\sqrt{}$) . Through this process, the output amplitude becomes stable and invariant for stimulus phases whereas those of the spatiotemporal filters are periodically modulated because of their phase dependency. High-speed implementation of this model has been attempted by using a specialized hardware [4].

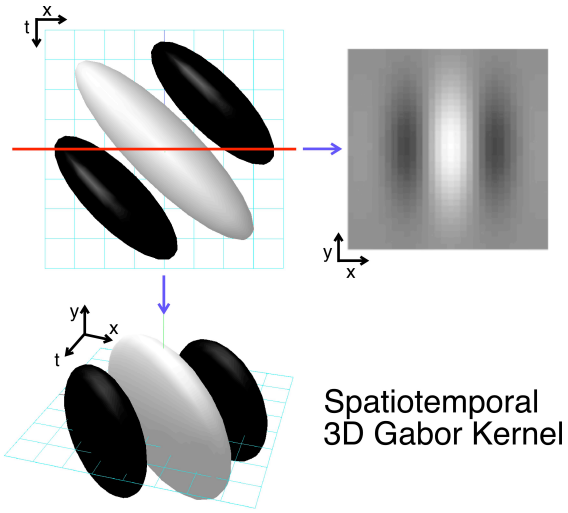

Fig. 2 Graphical representation of spatiotemporal 3D Gabor filter. Top view corresponds to spatiotemporal filter, and a cross section at a plane perpendicular to t axis (x-y plane) shows spatial Gabor filter.

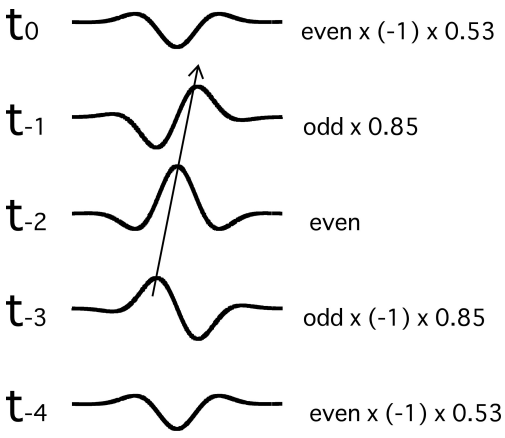

Fig. 3 Temporal summation across subsequent 5 frames. Each frame was filtered by even and odd spatial Gabor filters. Arranging phase shift between the neighboring frames produce the counterpart of spatiotemporal convolution in our algorithm. Recent five frames (t_0~t_{-4}) were used and a temporal window function (0.53, 0.85, 1, 0.85, 0.53) was applied.

In this study, spatial 2D Gabor filter was applied to each video frame, and then temporally summed. For each frame, even and odd kernel was used. The constituents of the temporal summation were recent 5 frames. To make sensitivity to moving or stationary edges along axis perpendicular to the filter orientation, the difference in the phase of Gabor kernel between two continuous frames was varied as -90, 0, 90 degree. For 0 degree difference, images filtered with the same phase were summed. For -90 or 90 degree, images filtered with even and odd ones were properly chosen and summed as shown in Fig. 3. The central part of a Gauss distribution was used as a temporal weight function. The resultant images were outputs of the spatiotemporal filters and inputs to the following process leading to the final output of the spatiotemporal energy model. Outputs for three motion direction, e.g. leftward, stationary, and rightward for horizontal axis, were compared, and the maximal response was regarded as detected motion direction at the filter location. This directional information was thrown into further competition among those from multiple resolutions described in the next section.

III. MULTI-RESORUTION

When using spatiotemporal energy model for motion detection, detectable motion speed is confined in proportion with the size of the filter. Multi-resolution approach is a prescription for the limitation. Decreasing the size of images is comparable to increasing the size of filter kernel. At high and low resolution, slow and fast motion can be detected, respectively. Here a VGA input fame was scaled into 3 lower resolutions, and motion detection was performed independently at each resolution (Fig. 4). This process consequently requires one more step that integrates the outputs of different resolutions for the final single output. This integration was just done by giving a priority to the fastest motion signal among signals form different resolutions at each pixel.

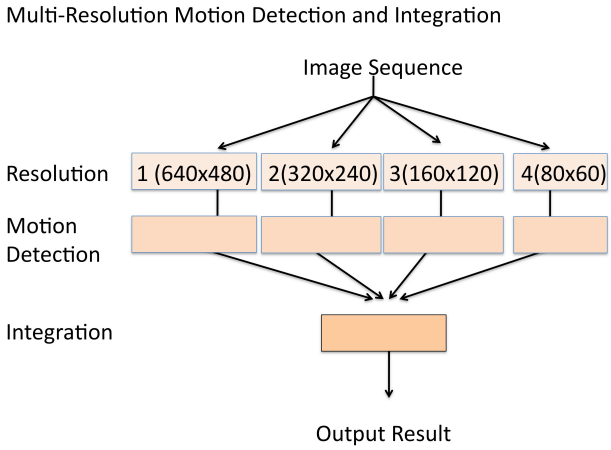

Fig. 4 Motion detection at 4 different resolutions. A single input image was scaled into 4 sizes and processed in parallel. Outputs form different resolutions were integrated for a single output.

These motion detection processes are summarized as shown in Fig. 5. Input image sequence is firstly transformed into gray scale images. Motion detections are independently performed at 4 different resolutions through spatial Gabor convolution, temporal summation of adjacent 5 frames for 3

motion directions, and combining the outputs from the quadrature pair into energy expression. The direction showing the largest response is regarded as detected motion direction at the resolution. Finally, the results at 4 resolutions are integrated for a single result of motion detection, and an output image sequence is produced to show the result visually.

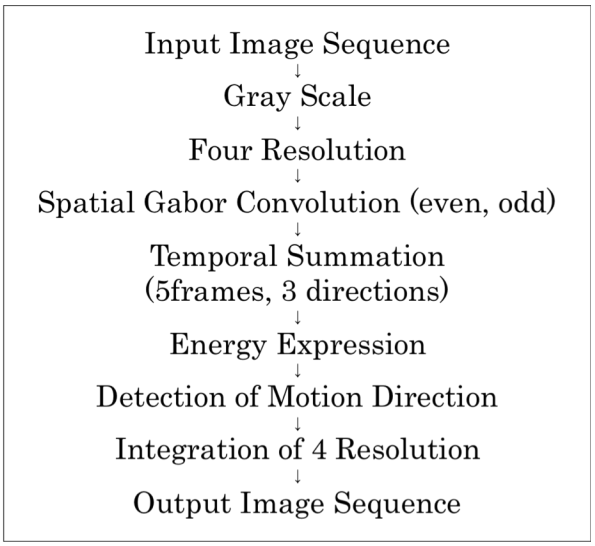

Fig. 5 A moment image in a driving movie.

An example of the result of motion detection through these processes is shown in Fig. 6, 7 and 8. The picture in Fig. 6 is a moment in a movie recorded by a vide camera attached on a car. For the calculation, the frame in Fig. 6 as well as two preceding frames and two succeeding frames, 5 frames in total, were used. Detected motion directions by offline calculation are indicated by colors in Fig. 7 for 4 different resolutions and in Fig. 8 for integrated result. Four colors were assigned for detected four motion directions, red for leftward, green for rightward, yellow for upward, blue for downward. White was assigned for stationary edges. The difference in motion speed is indicated by purity of the colors. If motion speed of an edge is slow, the edge is painted by whitish color, e.g. slow leftward motion by pink. Because of the expansive optical flow, the left part moving leftward is indicated by red. In the same manner, the rightmost power pole is indicated by green meaning rightward motion, and the bottom zebra crossing is indicated by blue meaning downward motion. Thus the distribution of colors in Fig. 7 and 8 tells you that the car is moving forward. For the highest resolution at upper left in Fig. 7, relatively slow motions around the central part of the image are detected. For lower resolutions, relatively fast motions around the peripheral part are detected. They are integrated into an image in Fig. 8. Our attempt is to earn this resultant image as continuous image sequence in real-time.

Fig. 6 A moment image in a driving movie.

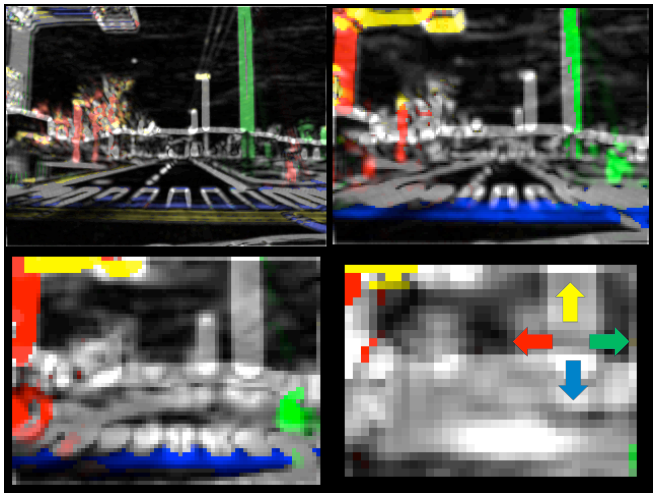

Fig. 7 Detected motion directions at 4 different resolutions shown by 4 colors. Upper left: 640x480, upper right: 320x240, lower left: 160x120, lower right: 80x60.

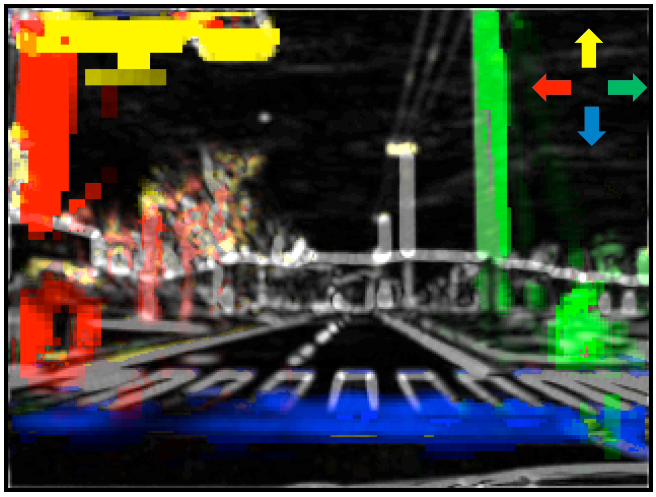

Fig. 8 Integrated motion directions shown by 4 colors.

IV. EXPERIMENTAL SETUPS AND METHODS

The specs of the computer used for measuring processing speed were as follows. CPU: Intel Core i7 975 EX, 4 cores, 3.33 GHz. GPU: nVIDIA GeForce GTX285, 240 cores, graphic 702 MHz, processor 1512 MHz. The computer had two GPU cards devoted to GPGPU calculation, and one more for graphics use only. These GPU units were controlled under a GPGPU environment named CUDA2.0 supplied by a GPU vendor nVIDIA. This environment enables us to use processors equipped for graphics as those for parallel processing, and consists of a driver for the video cards and C-language programming libraries. Program codes controlling the GPUs were integrated with those for the CPU in a project of Microsoft Visual Studio 2005. A PC camera supplied image frames through USB interface. Total processing time for each frame and processing time consumed for spatial convolution at each resolution were measured and averaged over 1000 frames. The sizes of the convolution filter were 8 x 8, 16 x 16, and 32 x 32 pixels containing carrier sine or cosine wave whose wavelengths were 4, 8, and 16 pixels, respectively (Fig. 9). Half height width of Gaussian window function was the same as the wavelength of the carrier wave. Phase shift of 90 degree corresponds to the 1/4 of the wavelength, that is, 1, 2, and 4 pixels, respectively.

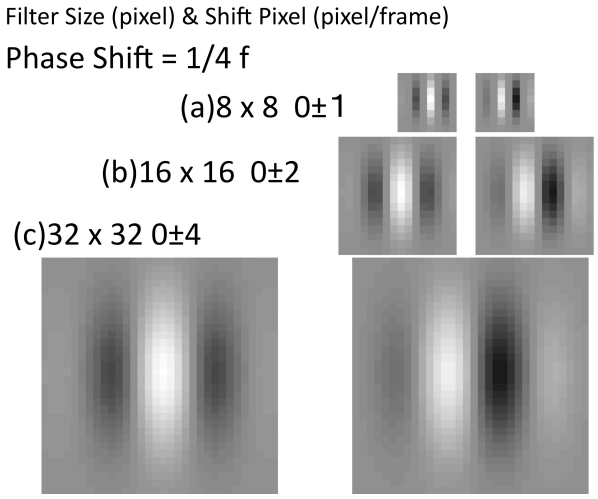

Fig. 9 Quadrature pairs (left: even, right odd) of spatial Gabor filter kernel. Sizes were varied. For each size, phase shift between adjacent temporal frames were 0, 1/4, or -1/4 f (=wavelength). Corresponding shift distances are indicated as pixel numbers after the filter sizes.

V. IMPLEMENTATION

To compare the performance of GPU with that without GPU, the same algorithm was executed under two different conditions on the same computer. For the condition without GPU, motion detection algorithm was implemented by using CPU only (Fig. 10). Threads were used to improve the performance. For the condition with GPU (Fig. 11), CPU threads were used only for sampling and displaying images, and for controlling two GPUs. The CPU sends a single original VGA frame to the GPU cards, and receives a single output frame from GPU. All the motion detection process of spatiotemporal energy model, multi-resolution scale change, and integration are implemented on the GPUs. One GPU was assigned for the motion detection along horizontal axis, leftward and rightward, and another for that along vertical axis, upward and downward.

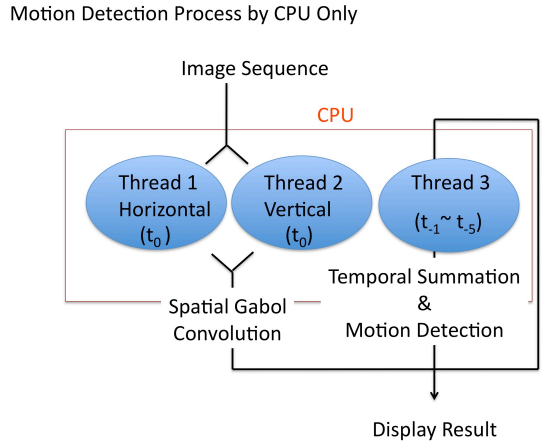

Fig. 10 Processing stream under CPU only condition.

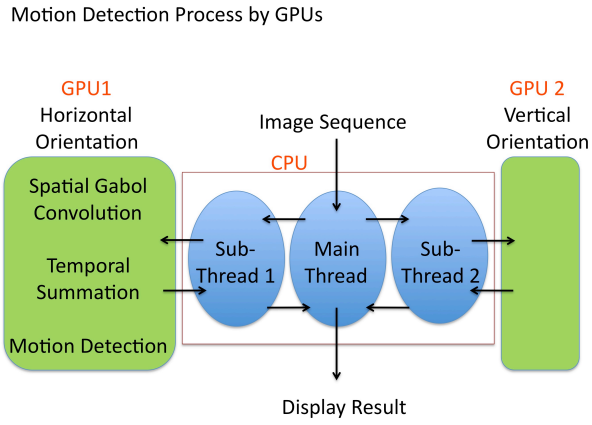

Fig. 11 Processing stream under GPU condition

VI. RESULTS

Processing times under CPU only condition are shown in Table 1 and as a graph in Fig. 12. Total processing times are about 4 times different between adjacent filter sizes. These differences are proportional to those in the filter size as area equivalent to the number of pixels contained in the filters. This means that, for CPU only condition, total processing time is proportional to the number of repetitive convolutional calculations. Actually, more than 95% of the total processing time is consumed by spatial convolution of the Gabor filter, and processing times of spatial convolution at each resolution are also about 3~4 times different between adjacent filter

sizes. Under this condition, processing time for 8 x 8 filter is about 0.5 sec for 1 frame or 2 FPS (frames per second), meaning that real-time processing is almost impossible. For larger filters, their temporal resolutions are far beyond reality.

CPU Only: Processing Time (ms)

	8 x 8	16 x 16	32 x 32
Total	422	1566	5673
640 x 480	309	1185	4428
320 x 240	75	281	986
160 x 120	18	62	185
80 x 60	3.9	12	22

Table 1 Processing times under CPU only condition. Total processing time and those consumed by spatial Gabor convolution at four different resolutions are shown for each filter size.

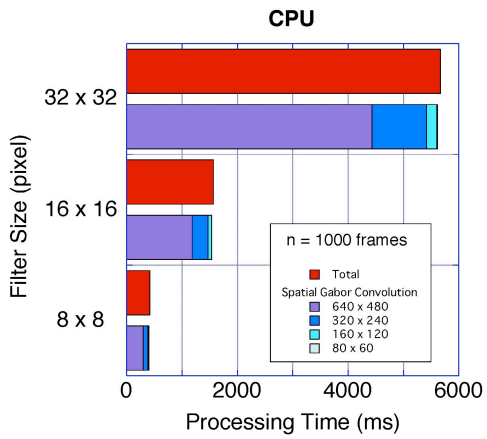

Fig. 12 Processing times under CPU only condition. Total processing time and those consumed by spatial Gabor convolution are indicated for each filter size.

For GPU condition, Table 2 and a graph in Fig. 13 show the results. Processing times are quite different from those under the former condition. Firstly, the amounts are quite smaller than the former condition. The smallest and middle size filter consumed less than 10 ms in total. Even the largest filter consumed 22 ms. Secondly, the differences between filter sizes are small. For 8 x 8 and 16 x 16, total processing times are almost the same. For 32 x 32, total processing time is twice that of 16 x 16. This difference is much smaller than those seen under CPU only condition. Focusing on spatial convolution at each resolution, the differences between 16 x 16 and 32 x 32 are large. Especially, the difference at 640 x 480 resolution is 5.3 times. The cause of this difference is now under investigation. Thirdly, the percentages of processing time consumed by convolutional calculation are smaller than those under the former condition. For 8 x 8, 16 x 16, and 32 x 32 filter, the percentages of the processing time consumed by spatial convolution were 32%, 40%, 78% of total ones, respectively. Decrease in processing time for convolution made the rest processing time, e.g. integration of the outputs of 4 different resolutions, standing out.

GPU: Processing Time (ms)

	8 x 8	16 x 16	32 x 32
Total	9.23	9.22	22.02
640 x 480	1.01	2.33	12.39
320 x 240	0.40	0.75	3.23
160 x 120	0.28	0.36	0.96
80 x 60	0.24	0.26	0.53

Table 2 Processing times under GPU only condition. Total processing time and those consumed by spatial Gabor convolution at four different resolutions are shown for each filter size.

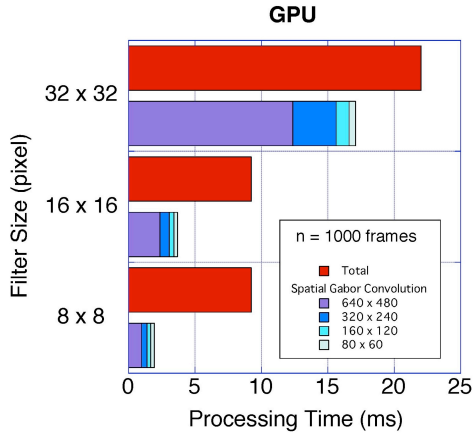

Fig. 13 Processing times under GPU condition. Total processing time and those consumed by spatial Gabor convolution are indicated for each filter size.

Temporal Resolution (FPS)

	8 x 8	16 x 16	32 x 32
CPU Only	2.37	0.64	0.18
GPU	108.4	108.5	45.4
Ratio	x 45.7	x 170	x 252

Table 3 Temporal resolutions and their ratio between CPU only and GPU conditions. Numbers show temporal resolution by FPS calculated from processing times. The bottom row shows multiplying factors for each filter size.

Overall, as shown in Table 3, the temporal resolutions for each filter size under CPU only condition are far less achieving real-time. On the other hand, the smallest and

middle size filter under GPU condition marked more then 100 FPS. Considering that the frame rate of ordinary PC camera is 30 FPS, this temporal resolution itself is more than three times the resolution required and has capacity of some post-process on top of the motion detection process. Even for the largest filter, the rate exceeds 30 FPS. The ratios of processing time between CPU only and GPU conditions are 46~252 times depending on the filter size. The computer with GPUs attached was about 1.5 times more expensive than that without GPUs. Together with the performances of these systems, which is 50~250 times different, the cost of the system with GPUs was 1/30~1/150 of that of the system without GPUs.

VII. CONCLUSION

Again the aim of this study is to open a path through the border of vision science and image processing technology. This should be beneficial for both fields like a tunnel under some mountain connecting two big cities. One of the obstacles has been processing speed. Nothing other than hyper parallel architecture can conquer this situation. Here we constructed this passage on a common parallel technology. Accessibility to GPGPU technology is comparable to ordinary high spec PCs. Temporal resolution of 100 FPS is enough for real-time usage, and has more capacity for expansive implementation of higher order receptive field functions such as surround suppression [5] or contextual modulation [6]. Spinning off the product of this physiological approach into some image processing system should become drive force for both image technology and vision science.

Acknowledgement This work was partially supported by a grant of Knowledge Cluster Initiative implemented by Ministry of Education, Culture, Sports, Science and Technology (MEXT).

REFERENCES

[1] E. H. Adelson, J. R Bergen. Spatiotemporal energy models for the perception of motion, *J.Opt. Soc. Am. A*, Vol. 2, pp. 284-299, 1985.

[2] G. C. DeAngelis, I. Ohzawa, R. D. Freeman. Spatiotemporal organization of simple-cell receptive fields in the cat's striate cortex. I. General characteristics and postnatal development, *J.Neurophysiol.*, Vol. 69, pp. 1091-1117, 1993.

[3] G. C. DeAngelis, I. Ohzawa, R. D. Freeman. Spatiotemporal organization of simple-cell receptive fields in the cat's striate cortex. II. Linearity of temporal and spatial summation, *J.Neurophysiol.*, Vol. 69, pp. 1118-1135, 1993.

[4] A. Spinei, D. Pellerin, D. Fernandes, J. Herault. Fast hardware implementation of Gabor filter based motion estimation, *Integrated Computer-Aided Engineering*, Vol. 7, pp. 67-77, 2000.

[5] N. Petkov, E. Subramanian. Motion detection, noise reduction, texture suppression, and contour enhancement by spatiotemporal Gabpr filters with surround inhibition, *Biological Cybernetics*, Vol. 97, pp. 423-439, 2007.

[6] K. Zipser, V. A. Lamme, P. H. Schiller, Contextual modulation in primary visual cortex, *J.Neurosci.*, Vol. 16, pp. 7376-7389, 1996.