computing@tanet.edu.te.ua
www.tanet.edu.te.ua/computing

ISSN 1727-6209
International Scientific Journal of Computing

A NEURAL NET FOR 2D-SLOPE AND SINUSOIDAL SHAPE DETECTION

A. Brückmann [1], F. Klefenz [2], A. Wünsche [3]

[1] Fraunhofer AEMT, Langewiesenerstr. 22, D-98693 Ilmenau, brueckma@idmt.fraunhofer.de, http://www.idmt.fraunhofer.de
[2] Fraunhofer AEMT, Langewiesenerstr. 22, D-98693 Ilmenau, klz@idmt.fraunhofer.de, http://www.idmt.fraunhofer.de
[3] Fraunhofer AEMT, Langewiesenerstr. 22, D-98693 Ilmenau, wuensche73@web.de, http://www.idmt.fraunhofer.de

Abstract: *2D-slope and sinusoidal shape detection are application specific tasks which are widely discussed in the literature. A neural network is presented which is able to learn a set of different slopes or a set of sinusoids of different frequencies and to detect test patterns after the training stage. The neural net is composed of input neurons, delay neurons and output neurons. The delay neurons form a set of tapped delay lines. Each delay line adapts to its specific signal propagation velocity. The signal propagation velocity vector field of the delay lines is learned by collectively tuning the signal propagation velocities. The neural net is fed with a set of spatiotemporal training patterns, such as bars of different slopes or sinusoids of different frequencies. After training, the net is tested with a random set of 2D-patterns. Unsupervised learning with a Boltzmann temperature term is assumed.*

Keywords: *Orientation selectivity, frequency selectivity, delay tuning*

1. INTRODUCTION

Orientation selectivity is discussed by demonstrating the detection of bars of different slopes e.g. in [1], [2], [3]. The structure of a neural net which detects bars or sinusoids is outlined in the second section. The elements of the neural net, like tapped delay lines and dendritic tree output neurons, are introduced. The matrix-like interconnectivity patterns between the tapped delay lines and the output neurons are given. The dynamics of the neural net are described in the third section. The fourth section explains how the net self-learns the patterns for different net sizes by unsupervised learning with a Boltzmann temperature term. In the fifth section the solution found by the neural net is compared to mathematically derived solutions which are computed by Hough transform space-time equations for straight lines and sinusoids.

2. THE STRUCTURE OF THE NEURAL NET

The neural net learns to detect a set of training patterns like bars or sinusoids. The neural net is trained with a set of n different bars or n sinusoids of different frequencies. The training and test patterns are 2D binary pixel images of size $n \times n$. A typical set of training patterns in an image of size 9×9 is displayed in fig. 3 and fig. 4. The neural net discriminates n different patterns after training.

The structure of the neural net is shown in fig. 1. It consists of n input neurons, $n \times n$ delay neurons and n output neurons. The neural net is composed of two structuring elements: tapped delay lines consisting of concatenating delay neurons, and dendritic tree output neurons. The dendritic tree output neurons are equidistantly interspaced and perpendicular to the parallel delay lines. Between two adjacent output neurons a segment of the signal conducting pathways is confined as displayed in fig. 2.

The spatiotemporal input patterns are transformed to a time and place code where the firing of an output neuron signals the presence of a bar or sinusoid at time t. Each firing output neuron i signals a bar with a specific slope or a sinusoid with a specific frequency at time t, i.e. output neuron n codes a bar with a slope of 45°. The output neurons are linearly aligned with ascending slopes from slope 0 (neuron #1) up to slope 45° (output neuron #n). The n output neurons form a feature vector consisting of the state of the output neurons at time t. The feature vector is recomputed every time step t. The neural net collectively tunes the signal propagation velocities of the delay lines.

3. DYNAMICS OF THE NEURAL NET

Spatiotemporal processing takes place by shifting the image into the net row by row. Each time step an image row is clamped to the input neurons. Each input neuron triggers a signal propagating through its dedicated delay line upon activation by a clamped input pixel. The neural net operates in parallel on *n* spatiotemporal input data streams. Coordinated volleys of neural discharges are happening just-in-time at certain layers where the volleys are summed up by the corresponding output neurons. Output neuron *k* sums up the synaptic activity in its layer at each time step *t*. A spatiotemporal input pattern, i.e. a bar with slope *a* gives rise to a characteristic activation wave front. This wave front propagates through the delay lines. Due to the specific signal propagation velocity field the activation wave front forms a planar wave front at a specific layer *k* and for a specific time *t*. The output neuron spikes upon registration of the planar wave front. The activation wave front rapidly dissolves before or behind that layer due to the different signal propagation velocities.

4. THE LEARNING PROCESS

The net learns to collectively tune the signal propagation velocities in the delay lines. The input layer feeds the subsequent layers with the spatiotemporal input patterns and triggers the signal propagation through the associated delay lines. The delays of the delay neurons are equal and are set for the sake of simplicity to 1 (the duration of a clock step). Each delay line consists of the signal conducting pathway as displayed in fig. 1 and fig. 2. Each pathway branches at a signal path bifurcation into a signal delay path and a direct path [fig. 2]. Both paths recombine at a signal junction [fig. 2]. The path selection and therefore the signal propagation velocity is regulated by two weights $w_{ij,delay}$ and $w_{ij,direct}$ [fig. 2]. By adjusting these weights according to the applied learning rule the signal propagation velocities in the delay lines are collectively tuned. The weights $w_{ij,delay}$ and $w_{ij,direct}$ are in the range [0, 1] and are initially set to 0.5. The input layer differs from the other subsequent layers in that the direct path weights $w_{1j,direct}$ are set to 1 clamping the paths of the input layer directly to the first output neuron. The synaptic interconnections to the output neurons are hardwired $w_{ij,hardwired} = 1$) [fig. 2].

The synaptic weights are trained with an unsupervised learning rule and a Boltzmann temperature function which decreases from a starting temperature T_{max} to a lower end temperature T_{min} in constant amounts δT. Each layer has its own Boltzmann temperature. The following learning rule applies for all subsequent layers. A random number in the range [0, 1] is computed for every signal bifurcation. The probability of the direct or delay signal path being taken at a signal bifurcation is computed by a Boltzmann temperature dependent term

$$P_{direct} = \frac{e^{w_{ij,direct}/T_{Boltz}}}{e^{w_{ij,direct}/T_{Boltz}} + e^{w_{ij,delay}/T_{Boltz}}} \tag{1}$$

$$P_{delay} = 1 - P_{direct} \tag{2}$$

The random value at each node is then compared with the probability of the direct path. If the random number is greater or equal than P_{direct} the delay path is activated. If the random number is less than P_{direct} the direct path is activated.

If an output neuron spikes in a layer the weights of the selected paths are collectively changed by $+\varepsilon$ and the other by $-\varepsilon$. The weights are updated by

$$w_{ij,select,new} = w_{ij,select,new} + (1 - w_{ij,select,old} \times \alpha) \tag{3}$$

and

$$w_{ij,deselect,new} = w_{ij,deselect,new} - w_{ij,deselect,old} \times \alpha \tag{4}$$

An output neuron only spikes if all selected signal paths are activated, because the thresholds of the output neurons are equally set to the number of signal paths minus 1. The thresholds of the output neurons can be adjusted to lower values (e.g. the output neuron spikes if more than *k* inputs are active). This could accelerate the learning process and be more robust to noise or defective structures i.e. complete loss of several delay lines etc. The two weights $w_{ij,delay}$ and $w_{ij,direct}$ in the signal bifurcation paths are always simultaneously changed. The weights $w_{ij,delay}$ are computed as $w_{ij,delay}=1-w_{ij,direct}$. The Boltzmann temperature is lowered when an output neuron spikes. If the Boltzmann temperature has reached its minimal value, the maximum of both weights converges to 1, the minimum to 0.

The learning is described for a neural net with 9 output neurons, 9 neural delay lines and 9 input neurons [fig. 1]. The nine bars displayed in fig. 3 are repeatedly presented to the net in the learning phase. Each bar is computed by the straight line equation $y=a\times x+b$, where *b* is set to zero (common origin). The minimal and maximal slopes a_{min} and a_{max} are set to 0 and 1 respectively. The nine slopes a_i are varied between the interval [0, 1] by subdividing it into 9 slope values.

The neural net is trained to recognize nine bars with different slopes $a_i=\{0,a_2,\ldots,1\}$. The alignment of the output neurons forms a one-dimensional feature axis with ascending slopes from output neuron #1 (horizontal bar detector) to output neuron #9 (diagonal bar detector).

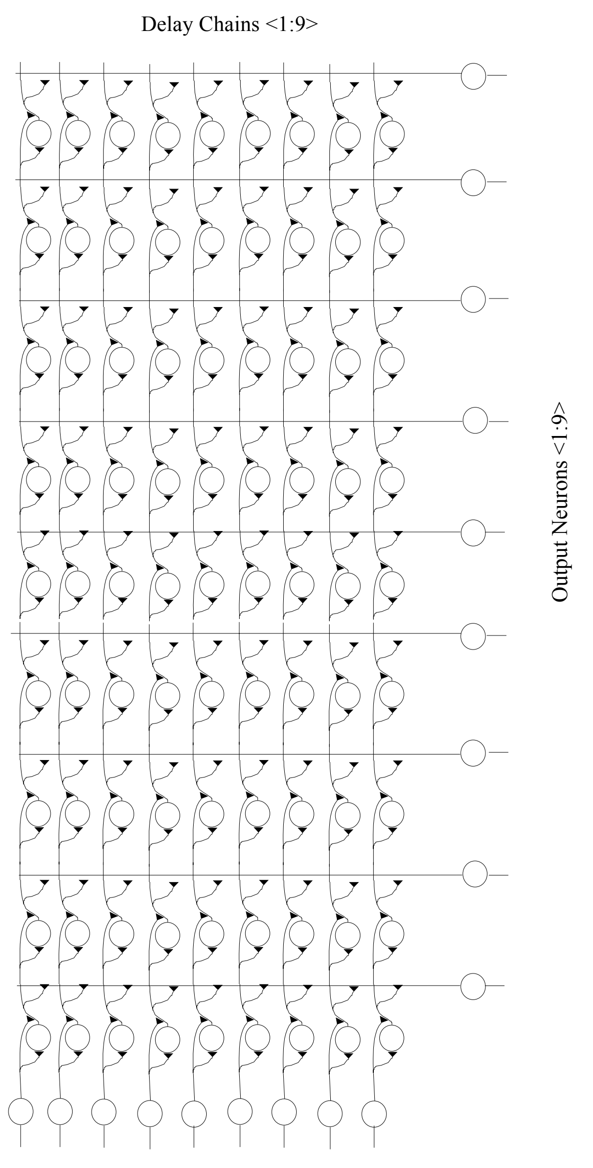

Fig. 1 – Structure of the neural net

For each layer the weight setting has to be learned. The learning of the weights is a time evolving process. Weights of the first layer settle first and converge to their 0 or 1 state. After the weights in the first layer have settled, the weights of the second layer begin to settle, then the weights of the subsequent layers, until the weights of the last layer settle. Subsequent learning in each layer depends on the preconditioned setting of the weights in the previous layers. Learning finishes when all weights converged to their final states 1 or 0.

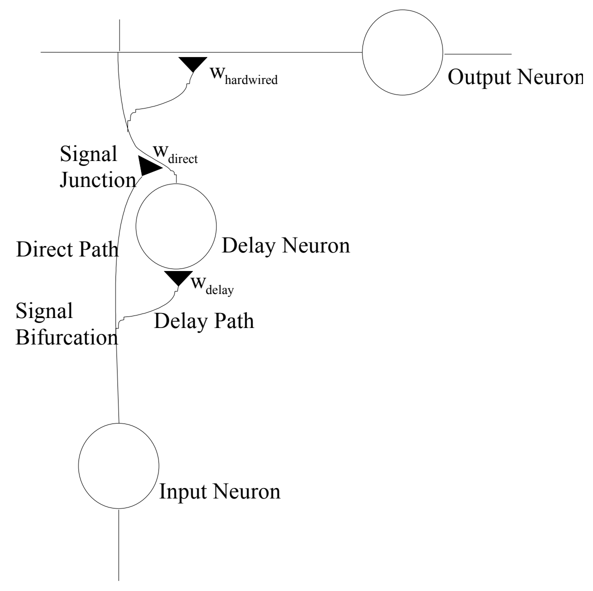

Fig. 2 – The signal paths and the synaptic interconnections

One image row with 9 input pixels is clamped to the input neurons every time step. The input layer is hardwired by setting the synaptic weights of the direct signal path to 1 and the delay path to 0, so that the first output neuron is directly interconnected to the latest image row. In the next time step output neuron #2 in the second layer receives direct inputs from image row(i+1) and the delayed inputs from image row(i). Output neuron #i in layer i is exposed both to the input stimuli from layer i–1 and the stored previous input stimuli from the intermediate delay neurons of the precedent layers. A spatiotemporal input pattern presented at output neuron i has been distorted by the previous layers, so that the output neuron #i learns a distorted input pattern j’ instead of the original input pattern j. Each layer can therefore be trained separately, substituting the spatiotemporal input pattern j by the distorted pattern j’ at layer i–1. The previous layers i–1 bend a specific spatiotemporal input j to a horizontal bar at layer i. Each output neuron is trained by its learning rule to learn a horizontal bar.

Various sets of spatiotemporal patterns can be learned. The learning of the weight settings depends on the training set and the size of the neural net. In learning the weight settings the neural net is able to detect patterns, like bars or sinusoids. The net has been trained with bars of different slopes and in a separate run with sinusoids of different frequencies. The training patterns are displayed in fig. 3 and fig. 4. The neural net has been trained for neural net sizes of 3×3 up to 16×16. Each net with n output neurons is trained with a set of n distinct training patterns. A weight setting for a neural net of size 9×9 trained with bars is displayed in tab. 1. The table is associated with the topology of the neural net in fig. 1 and reads from left to right starting with the weight setting w_{ij} of delay line 1 up to the weight setting w_{9j} of delay line 9. The table reads from bottom to top with the weight w_{i1} from layer 1 up to the weight w_{i9} from layer 9. Tab. 2 displays the weight setting for a net of size 9×9 trained with sinusoids. The weight setting for sinusoids differs from bars; see tab. 1 and tab. 2 for comparison. The velocity vector field is monotonic with descending velocities from delay line 1 to delay line 9 either for bars as for sinusoids. The neural net correctly detected bars and sinusoids for all examined net sizes.

5. COMPARING THE NEURAL NET SOLUTION TO A MATHEMATICAL SOLUTION

The neural net architecture as displayed in fig. 1 is topologically identical to the layered architecture of a parallel Hough transform ASIC as described in [4] and shares the same functional elements if the symbols of fig. 1 are translated into the equivalent electronic gate-level description. The Hough transform is a standard pattern recognition tool for finding simple patterns like straight lines, circles

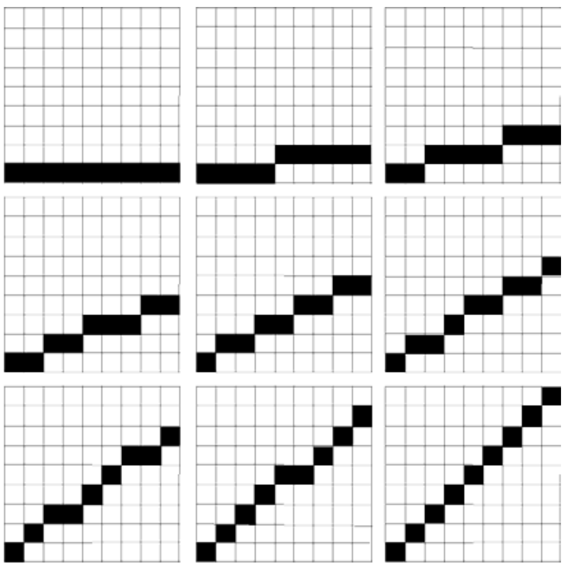
Fig. 3 – The training set of nine bars, x-axis horizontal, y-axis vertical

and trigonometric functions in images [5],[6]. The Hough transform is analytically expressible for straight lines, circles and trigonometric functions by their corresponding coordinate transform equations; read [7] for reference, where an algorithm for a parallel execution of the Hough transform in a pixel grid for real-time detection of circles in multi-wire drift chambers is explained in detail.

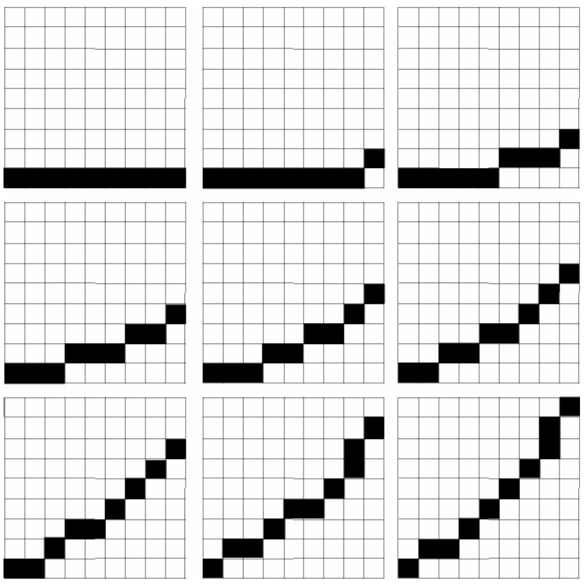
Fig. 4 – The training set of nine sinusoids, x-axis vertical, y-axis horizontal

To compute the Hough transform for straight lines, the straight line equation $y=ax+b$ is rewritten in the form $b=-x_i \times a+y_j$. The Hough transform is executed by generating for each input pixel i,j with input coordinates x_i,y_j the corresponding straight lines $b(a)$ with slope $-x_i$ and offset y_j in the a,b coordinate system. The input data is processed by the parallel Hough transform in the same way as the neural net does. The straight line generation $b(a)$ is parallelized for a complete input row $\{x_0,\ldots,x_n\}$ and serialized in y; $y=y_0,\ldots,y_n$. For all set pixels i in an input image row j, the straight lines $b=-x_i \times a$ with fixed slope $-x_i$ for each position $\{x_0,\ldots,x_n\}$ are generated. The slopes $-x_i$ are expressed as a run length coded local slope sequence, e. g. the diagonal line is run length coded as {1,1,1,1,…,1}. The velocity vector field for the parallel Hough transform execution consists of the set of all run length coded local slope sequences $-x_i$.

Comparing the velocity vector fields of the parallel Hough transform and the neural net shows whether they converge to the same or a similar solution. The velocity vector field of the neural net

Table 1. Weight setting in a net of size 9×9, trained with bars

w_{1j}	w_{2j}	w_{3j}	w_{4j}	w_{5j}	w_{6j}	w_{7j}	w_{8j}	w_{9j}
1 0	1 0	1 0	1 0	1 0	0 1	0 1	0 1	0 1
1 0	1 0	1 0	0 1	0 1	1 0	1 0	0 1	0 1
1 0	1 0	0 1	1 0	1 0	0 1	0 1	0 1	0 1
1 0	1 0	1 0	1 0	0 1	1 0	0 1	1 0	0 1
1 0	0 1	1 0	0 1	1 0	0 1	0 1	0 1	0 1
1 0	1 0	1 0	1 0	0 1	0 1	1 0	0 1	0 1
1 0	1 0	0 1	0 1	1 0	1 0	0 1	0 1	0 1
1 0	1 0	1 0	1 0	0 1	0 1	0 1	0 1	0 1
1 0	1 0	1 0	1 0	1 0	1 0	1 0	1 0	1 0

Table 2. Weight setting in a net of size 9×9, trained with sinusoids

w_{1j}	w_{2j}	w_{3j}	w_{4j}	w_{5j}	w_{6j}	w_{7j}	w_{8j}	w_{9j}
1 0	1 0	1 0	1 0	1 0	0 1	0 1	0 1	0 1
1 0	0 1	1 0	1 0	0 1	1 0	1 0	1 0	0 1
1 0	1 0	1 0	0 1	1 0	0 1	0 1	0 1	0 1
1 0	1 0	0 1	1 0	0 1	1 0	0 1	0 1	0 1
1 0	1 0	1 0	1 0	1 0	0 1	1 0	0 1	0 1
1 0	1 0	1 0	0 1	0 1	1 0	0 1	0 1	0 1
1 0	1 0	1 0	1 0	1 0	0 1	0 1	0 1	0 1
1 0	1 0	1 0	1 0	1 0	1 0	1 0	1 0	0 1
1 0	1 0	1 0	1 0	1 0	1 0	1 0	1 0	1 0

is defined by the weight setting. The weight setting defines the signal propagation through the delay lines. If the net is configured with the weights setting of tab. 1, we can single step through the net to follow its function. All direct path segments are selected for delay line #1 [tab. 1]. The signal passes through delay line #1 instantaneously. Delay line #2 is configured with the first 4 segments set to direct path, the fifth set to delay path, and the next four to direct path. The signal propagation is blocked at the fifth delay neuron at time t, and is propagated through the next for segments at time t+1. The path length of a non-interrupted signal path is counted by the number of output neurons which are simultaneously activated between a path start delay neuron and the next path stop delay neuron. The signal propagation for each delay line is run length coded by serially writing the path lengths for each delay line. For delay line #2 the signal propagation code reads as {4 5}, for delay line #3 it is {2 4 3}; see tab. 1. The velocity vector field of the neural net consists of the set of the path length sequences of the delay lines #1 to #*n*.

By direct comparison of fig. 3 and tab. 1 it is obvious that each delay line learns the local slope x_i of its associated bar expressed as a run length coded local slope sequence. The local slope sequence {2 4 3} of bar #3 in fig. 3 is the same as the path length sequence {2 4 3} for delay line #3 as shown in tab. 1. Each column *j* (delay line #*j*) codes the run length coded local slopes x_i of its associated bar in a net of size 9×9; compare tab. 1 and fig. 3.

6. SUMMARY

A neural net with velocity tuneable delay lines self-learns to detect bars of different slopes and sinusoids of different frequencies depending on the applied training set. Self-learning has been examined for different sizes of the neural net. The neural net executes a coordinate transform which maps the spa-tiotemporal input patterns to a feature vector. The weight settings are either analytically derived by the Hough transform equations or are self-learned by the neural net.

7. REFERENCES

[1] R. Shapley, D. McLaughlin, M. Shelley. Orientation selectivity, M. A. Arbib (Editor). *The Handbook of Brain Theory and Neural Networks*. MIT Press. 2003. p. 831 – 835

[2] G. Blasdel. Orientation selectivity, preference and continuity in monkey striate cortex, *Journal of Neuroscience* 12 (8) (1992). p. 3139 – 3161

[3] D. H. Hubel, T. N. Wiesel, M. P. Stryker. Anatomical demonstration of orientation columns in macaque monkey, *Journal of Comparative Neurology* 177 (1978). p. 361 – 380

[4] A. Epstein, G. U. Paul, B. Vettermann, C. Boulin, F. Klefenz. A parallel systolic array ASIC for real time execution of the Hough-transform, E. S. Peris, A. F. Soria, V. G. Millan (Editors). *Proceedings of the 12th IEEE International Congress on Real Time for Nuclear and Plasma Sciences*, Valencia, 2001. p. 68 – 72

[5] D. H. Ballard. Generalizing the Hough transform to detect arbitrary shapes, *Pattern Recognition* 13 (1981). p. 111 – 122

[6] P. V. C. Hough. Method and means for recognizing complex patterns, *US Patent 3069654* (1962)

[7] F. Klefenz, K. H. Noffz, R. Zoz, W. Conen, R. Männer. R. (1993) Track recognition in 4 microseconds by a systolic trigger processor using a parallel Hough transform, *IEEE Tr. Nucl. Sci.* 40 (4) (1993). p. 688 – 691

__Andreas Brueckmann__ was born in 1978 in Hessisch Lichtenau, Germany. He studies Computer Science at the Technical University of Ilmenau, Germany. In addition he is involved in the development of music recognition systems and sematic audio analysis at Fraunhofer IDMT Ilmenau, Germany. His areas of interests are sports, making music and conjuring tricks.

__Dr. Frank Klefenz__ was born in 1961 in Heidelberg, Germany. He received a diploma in physics in 1988 and a PhD in physics in 1992. He developed several systolic array computers in FPGA as second level triggers for CERN, Geneva. He devised a parallel Hough transform ASIC. He is currently at Fraunhofer IDMT leading a group of research scientists in the fields of music recognition. He holds *several patents in that field. His areas of interests are saxofone playing and adobe constructions.*

***Andreas Wünsche** was born in 1973 in Hohenmölsen, Germany. He received a diploma in telecommunications from the University of Applied Sciences in Leipzig in 2000. He joined Fraunhofer IDMT and specialized in developing a Hubel-Wiesel neural net simulator. He now is at Siemens VDO in Wetzlar. His areas of interests are home recording and drumming.*