Simulation of pulsed laser material processing controlled by an extended self-organising Kohonen feature map

Gábor J. Tóth[1], **Tamás Szakács and András Lőrincz**

Department of Photophysics, Institute of Isotopes, The Hungarian Academy of Sciences,

Budapest, Konkoly-Thege utca 29-33, P.O.Box 77, Hungary H-1525

[1] Eötvös Loránd University, Budapest, Rákóczi út 5, Hungary H-1088

Abstract

We have simulated the control of laser material processing. The process to control was surface polishing. In the simulations we assumed pulsed ultraviolet laser beams of 100 nanosecond duration. The controller was an artificial neural network (ANN): an extended self-organising Kohonen feature map. The controller was trained – tuned – on examples of laser evaporation of one dimensional 'surfaces' by using a Widrow-Hoff type Delta-rule for error correction. Strength of the approach lies in the speed of parallel computing and the adaptive properties of the controller in a changing environment, like drift in laser properties, optical components, etc. Results show more than an order of magnitude improvement in surface roughness after an ANN designed laser shot onto a large surface. Restrictions of the model are discussed.

Keywords: Neural control, laser processing.

PACS numbers: 02.90.+p 44.90.+c 81.90.+c

Correspondence : András Lőrincz

FAX : (36-1) 156-5045

1 Introduction

Though computer technology has been developing at a remarkable rate, it is well known that traditional computers are surpassed in the field of pattern recognition, associations and the like by the human brain. This is why artificial neural networks (ANNs) are in the focus of research interest. ANNs try to model some aspects of human information processing. The idea gains its strength through the possibility of extensive parallel processing with simple neuron-like units thereby offering high speed and excellent fault tolerance properties and, through the working mechanism, the advantage that tuning (training) takes place with the help of examples. This means adaptivity to the network as long as learning is not stopped.

Both the evolution of ANN paradigms and their applications are striking. The wide range of applications includes pattern recognition [1], associative memories [2] and control systems [3].

The adaptive properties of ANNs may be advantageous in control tasks when the parameters or the equation of the object to be controlled are not known in detail. The application we present here is of that type. The goal is the laser polishing of rough surfaces. We assume a pulsed laser producing pulses of 100 nanoseconds duration. In the simulations we envisage an optical system that detects surface roughness with the help of some interferometric method [4]. The surface data are then plugged into the control system that designs the laser field distribution along the surface. Tailoring of the output of

the pulsed laser is the task of another optical system. The design of the appropriate optics is not part of this paper. We should like to mention, however, that fast interferometric methods [4] and spatial light modulators [5] are capable of the required performance. The result of the laser shot is detected through the same interferometric procedure and the ANN is adjusted by an error correction rule in the training phase.

The application assumes fast neural network processors that are already coming on to the market. At present we are simulating these ANN's on traditional machines instead of real parallel computing. The result is that our software – that runs on a 25 MHz IBM PC 386 machine equipped with a coprocessor – could generate laser fields of 200 nm spatial resolution for 1 mm^2 surface in about 7000 seconds. The 200 nm resolution is about the edge of present day optics. For comparison let us consider excimer lasers that are capable of producing high energy 100 nanosecond pulses. The lasing of excimer lasers takes place in the ultraviolet and penetration is not deeper than 1 μm for most metallic and semiconducting material. These lasers are capable of firing at a 50 Hz rate. One would like to keep this rate as that allows reasonable material machining speeds. The energy of the pulses is in the order of 200 mJ. Short penetration depth and 100 nanosecond heat diffusion allow the micromachining of about 5 mm^2 per shot at a depth of 100 μm assuming a decrease of laser energy of an order of magnitude in the optical system. This area is still large for a vibrationless mechanical and optical system that should deliver the laser energy to the surface. Taking a conservative estimate of machining to be 1 mm^2/s the computing speed should be accelerated by almost four orders of magnitude. This figure is not unrealistic, however, since present day commercial ANN simulator PC cards could increase our computing speed by almost three orders of magnitude allowing a 0.1 mm^2/s

machining speed with conventional computers. VLSI ANN chips can readily achieve the required performance [6].

There is another point apart from speed that makes ANN favourable for the laser machining task and that is its adaptive properties. Laser machining requires a laser, light guiding optics, and optical devices for detection. All of these may change by aging. This makes adaptivity very attractive and ANN's are designed to meet this challenge.

In this paper we present an approach that uses an extension of Kohonen's self organising network architecture for the formation of topographically correct feature maps [7]. In the problem of input-output mapping an extended version of the original Kohonen network has already been applied for learning visuomotor coordination [8]. The Kohonen network performs optimal topology-conserving vector digitisation in a self-organising fashion. The network contains a second layer of neurons which produce the output of the network.

2 Description of the problem

Our approach to the problem was to simulate the network and to make an appropriate laser-matter interaction model. Several simplifications were built into this model to allow the simulation to be run within reasonable time limits. In the 100 nanosecond time region

considered here, simplifications are believed to be reasonable from the point of view of the core of the problem: to provide training examples to the network.

The first simplification step is the use of one dimensional 'surfaces'. This step considerably speeds up the computation of heat diffusion. In these computations we have divided the two dimensional material into boxes. The box size was determined by the stability requirements of the numerical solution of the heat diffusion equation. In this respect one has to take into account the spatial resolution and the penetration depth of the laser. The finite size of the boxes along the surface represents no serious limitation, since it is dictated by the physical constraints. Digitisation normal to the surface, however, does limit the precision of the computation and does not allow one to produce continuous input space for the network. Non-continuous input space, however, can seriously limit network performance. We made the model continuous in a simple and time saving fashion; this will be described later.

Another approximation was made in the modeling of the interaction. We considered only the heat diffusion problem and neglected the coupled acoustic problem. This might be a serious limitation in the picosecond regime. However, the longer the laser pulse, the smaller this problem. Our pulse durations are 100 nanoseconds thereby making this approximation possible for metals – the subject of our studies.

The laser beam energy needed to smooth the surface at any given point depends most on the height of the surface of the given point, then to a smaller extent on the height of its environment. Supposing a sudden heat deposition onto the surface we can approximate how far the heat can spread. From the classical heat diffusion equation the result [9] is

$\delta = (\kappa t_0)^{(3-n)/2}$, where t_0 is the duration of the pulse, κ is the diffusivity of the surface and n is the dimension of the system, in the present case $n = 2$. This implies that if one has a sufficiently short laser pulse the energy needed to reach the required evaporation depth at a given point barely depends on the height of the surface further than the spatial resolution of the laser. Very short pulses, however, are troublesome as we discussed above and we have to compromise with pulses of moderate duration in the 100 nanosecond range, implying that the close environment of a given point has to be taken into consideration.

3 Simulation of evaporation

The heat diffusion equation is

$$\rho c \frac{\partial T(\mathbf{r}, t)}{\partial t} = \kappa \nabla^2 T(\mathbf{r}, t) + H(\mathbf{r}, t), \tag{1}$$

where $T(\mathbf{r}, t)$ is the temperature in the material, ρ is the density, c is the specific heat and κ is the conductivity of the material. The heat source of the laser beam is represented by the term $H(\mathbf{r}, t)$ and has the form

$$H(\mathbf{r}, t) = I_0(\mathbf{r}) h(t) e^{-x/\alpha}, \tag{2}$$

where α is the penetration depth, I_0 the intensity density at the surface, and x is measured from the surface. The time dependent part of $H(\mathbf{r}, t)$ is $h(t)$ and we assume the following

simple form

$$h(t) = \begin{cases} 1 & \text{if } 0 \leq t \leq t_0 \\ 0 & \text{otherwise} \end{cases} \quad ; \tag{3}$$

the pulse begins at $t = 0$ and lasts until $t = t_0$.

In order to solve the differential equation we used a finite difference method on a two dimensional grid. In the boundaries parallel to the surface we used periodic boundary conditions. For the lower boundary we supposed a large heat reservoir, which keeps the temperature on the boundary constant; for the upper boundary we set the heat current to zero.

The above form of heat diffusion equation (Eq. (1)) is valid only if there is no phase transition. For the sake of simplicity we collected latent heat at the boiling point only by assuming amorphous metals as our substrate. That is, at every gridpoint we let the temperature increase up to the temperature of evaporation T_e. After that we let latent heat accumulate until it reached the evaporation heat of the point Q_e, then we considered the box represented by the gridpoint to be evaporated.

In order to decrease the error introduced by the surface digitisation we corrected the final surface by the truncated part of the initial surface and took into consideration the amount of latent heat remaining in the boundary boxes after the laser evaporation was finished. In this way the model was made continuous and allowed us to provide continuous input and error to the network.

The grid size along the surface was 200 nm, the height of a box was 100 nm; typical polishing depths were 2 μm; pulse duration was taken as 100 ns.

Surfaces were randomly generated in Fourier space by allowing wavelengths of 4 to 16 boxes for the Fourier components. A periodic boundary condition was required, then the surface was determined by taking the inverse Fourier transform.

4 The control system

The core of the control system – as mentioned earlier – was a neural network, other parts of the control system were used to transform the input and output of the network. As it was also mentioned it is not necessary to know the surface height at each point to determine the appropriate laser energy at a given point. The surface therefore was cut into few-box-wide elements by the control system; the center of these elements went through the whole surface; i.e. we used a 'moving window' filter. The width of the window was determined by estimating the lateral size of the environment that influences heat diffusion at a given position. The network was then presented the surface elements of the window. The network calculated the necessary laser energy at the center of the element, then its output was stored by the control system and the window moved on. The shot was taken when the appropriate laser energy was designed for each point.

The network consists of two layers. The first is a Kohonen's self organising net [7, 8]. Every neuron possesses an internal representation of part of the external world. In

the training sessions care is taken to conserve the topology of the external world in the internal representation. To every input a 'winner-take-all' procedure selects and activates a single neuron, that most resembles the actual input. In this way the continuous input space is mapped onto the discrete space of possible one-neuron activations. The mapping is optimal in the sense that the network will discover the distribution of input vectors and the larger the density of input vectors, the more neurons it will allocate to that region. Topology conserving means that the closer the input vectors are in the topology of the input space the closer their internal representations will be in the metric of the internal representation.

The first layer of neurons is connected to another layer of neurons, which determines the output of the network. This second layer is a LAM-style (Linear Associative Memory [10]) network. LAM neurons sum their inputs weighed by the stored weights and the output is simply that sum.

Each neuron in the first layer of the network stored a surface element of three boxes, or a three dimensional input representation vector. Each neuron in the second layer stored four weights referred to as the center value (1 weight) and gradient (3 weights). When an input vector is presented to the network that neuron whose input representation vector is the closest to the presented input vector will become active, all the others stay silent. The output of this neuron will be the distance of the three components of its stored input representation vector and the input vector presented, and a fourth output that is constant and is set to one. All the outputs of the other neurons of the first layer are zero.

Each neuron of the second layer is connected to the outputs of one neuron of the first layer. The neurons in the second layer sum their inputs weighed by their stored weights: the constant is multiplied by the weight known as the center value and the others by the components of the gradient. Since only one neuron of the first layer is active, only one neuron of the second layer receives inputs other than zero, thus only one second-layer neuron might have non-zero output. This output will be the energy of the laser beam at the center of the surface element. The architecture of the network is shown in Figure 1.

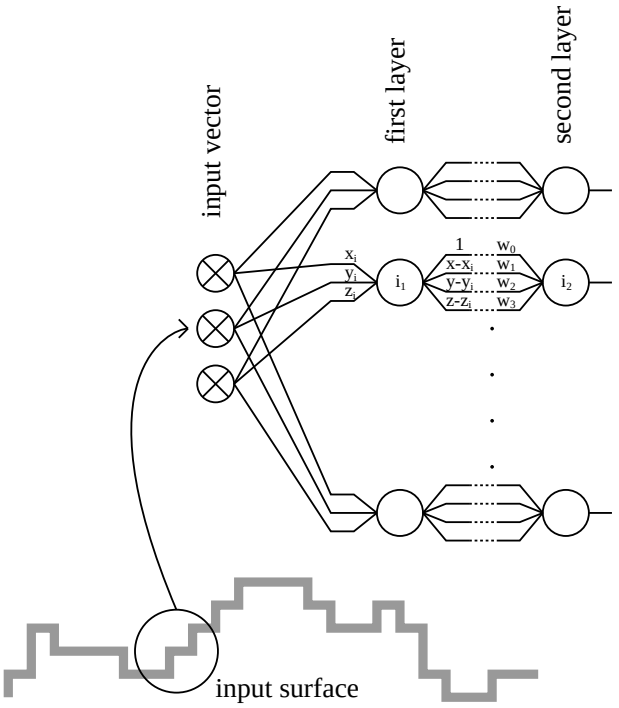

Figure 1: Network architecture. The network consists of two layers, the input layer and the linear approximator. The input layer receives three analogue inputs: the height of the surface at a given point and the heights of the two neighbouring surface points. The first layer performs the Kohonen digitisation of the problem, the second layer approximates the laser energy output between points in a linear fashion.

The names 'center value' and 'gradient' deserve some explanation. The output of the

network at each point can be written as

$$f = f(\mathbf{x}_0) + \mathbf{A}(\mathbf{x} - \mathbf{x}_0), \tag{4}$$

where $\mathbf{x}$ is the three dimensional input vector, $\mathbf{x}_0$ is the input representation vector stored by the winning neuron, and $f(\mathbf{x}_0)$ and $\mathbf{A}$ denote the four weights of the second layer neuron. This form shows that the network linearly approximates the surface height – laser energy function.

Before training, the input vectors of the neurons of the first layer were randomly chosen, the input vectors of the second layer were set to zero. During training, sample surfaces were generated as described earlier. The surface elements were presented to the network, and the network calculated the laser energy distribution. The finite difference method was then used to determine the effect of the laser pulse on the surface.

The input vectors of the first layer neurons were taught in the usual way [7]: the input vector of the winning neuron is changed so as to move towards the input vector, the input vectors of the other neurons are moved in the same direction but to a smaller extent, that depends on their distance from the winning neuron. Mathematically:

$$\mathbf{x}_r \rightarrow \mathbf{x}_r + \epsilon G(\rho(r,s))(\mathbf{x} - \mathbf{x}_s), \tag{5}$$

where $\mathbf{x}$ is the input vector presented to the network, r denotes an arbitrary neuron, s denotes the winner, ϵ is the overall learning rate, ρ is the metric defined among the neurons, and G is the distance dependence of the learning rate. In our examples ρ was a Euclidian metric and G was a Gaussian function:

$$G(x) = \exp\left[-\frac{x^2}{2\sigma^2}\right], \tag{6}$$

where σ characterises the cooperativeness of the network. Both ϵ and σ depend on time.

A modified Delta-rule was used to train the neurons of the second layer: the topology-conserving property of the first layer allowed some cooperativeness to be applied here as well. Another important difference is that the error of the network's output is not known: only the error in the output surface is known. Fortunately, however, the error of the output surface is a monotone function of the error of the network output being zero if the latter is zero. That is, one may use the error in the output surface instead of the error of the network. Now the learning rule may be written as

$$w_{r\alpha} \to w_{r\alpha} + \epsilon G(\rho(r,s)) i_\alpha \Delta, \tag{7}$$

where $w_{r\alpha}$ denotes the αth weight of neuron r: w_{r0} is its output and the other elements are the components of the gradient. s, ϵ and G are the same as above. However, ϵ and σ are not necessarily the same in the input and output layers, and ϵ can differ for the center value and the gradient. The input of the second layer neuron connected to the winner of the first layer is denoted by i_α: 1 for $\alpha = 0$ and $\mathbf{x} - \mathbf{x}_0$ for the rest. The error in the output surface is Δ, and $\Delta > 0$ if the surface is higher than it should be, i.e. if the energy was too low.

Training is divided into two sessions. In the first session only the first layer is trained, and no output is produced; in the second session mainly the second layer is trained, though the first layer remains adaptive as well. There is no theoretical reason for this separation but it very much reduces the CPU time necessary for simulations (in our examples up to 40% CPU time was saved), it could also reduce the cost of training in practice.

As has already been mentioned, both ϵ and σ varied in time. For the input layer they were exponentially decreased until the end of the first session and then this value was kept until the end of the training. For the second layer ϵ and σ were not defined in the first session and were exponentially decreased until the end of the training. At the end of training all the learning parameters had small but non-zero values. After the second session the training was finished, but if we kept the learning parameters at their final value the network would remain adaptive, thus slow changes in the environment will not affect its performance.

5 Simulation results

First, we trained the network on 32-box-wide surfaces containing two Fourier components; this meant a wavelength of 16 boxes. We found that the input vectors spanned a thin, approximately two dimensional region of the three dimensional input space (Fig. 2), so we used a two dimensional Kohonen network. From the heat diffusion equation we estimated that three-box-wide surface elements were necessary to determine the appropriate laser field. For comparison we used a network with three, and an other one (otherwise identical) with one dimensional input vector space. We found that the former could learn to control the evaporation process considerable better. Figure 3 shows a typical surface on which the network was trained and the output surface generated by the trained (3D)

network. We allowed the network to learn the input space for 2000 steps, then learn the energy for another 1000 steps. In Figure 4 we depicted the average error in the output surface as a function of time (learning steps). Figure 5 shows the characteristic behaviour of each neuron. It is interesting to notice how much the cooperation helped the neurons of low winning frequencies: the average error is a weak function of the winning frequency. The values of the parameters we used can be found in Table 1.

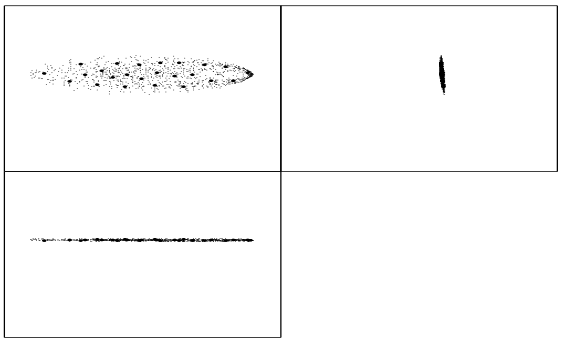

Figure 2: Random samples of three-point-wide surface elements. The domain covered by the three-point-wide elements of the randomly generated surfaces are shown by taking two dimensional projections along three orthogonal planes. The large solid dots correspond to the positions of the input representation vectors. The surfaces have two random frequency components, in other words the wavelength of the shortest surface roughness components is about 8 boxes, i.e. about 5 μm long.

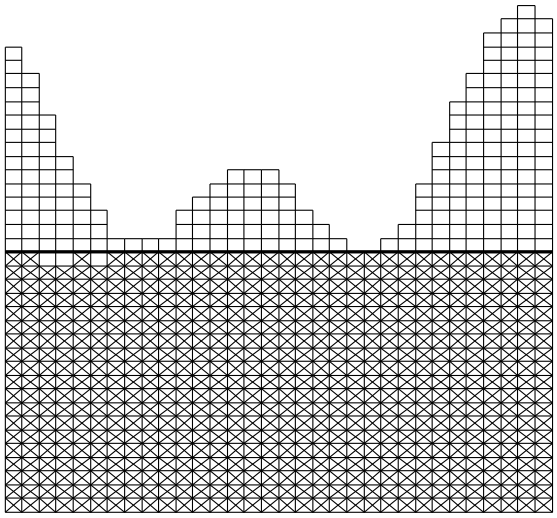

Figure 3: Smoothing surfaces: typical performance of the trained network. Empty boxes correspond to the original surface, filled boxes to surface after the shot. Wide line shows the target surface level. Box sizes are 100nm (vertical) $\times$ 200nm (horizontal).

The form of the region of the three dimensional space spanned by the surface elements depends on the input surface set. If we allow more than 2 Fourier components the input vectors are not restricted to the same thin region but can span a domain of similar dimensions in every direction. In this case, or if we want to use longer pulses, we have to use a three dimensional network.

Initial width of Gaussian for InMap	:	1.0000
Final width of Gaussian for InMap	:	0.0100
Initial value of scaling factor for InMap	:	0.2000
Final value of scaling factor for InMap	:	0.0500
Initial width of Gaussian for Central	:	2.0000
Final width of Gaussian for Central	:	0.1000
Initial value of scaling factor for Central	:	0.0010
Final value of scaling factor for Central	:	0.0010
Initial width of Gaussian for Grad	:	2.0000
Final width of Gaussian for Grad	:	0.1000
Initial value of scaling factor for Grad	:	0.0001
Final value of scaling factor for Grad	:	0.0001
Maximum learning step number (1st session)	:	2000
Maximum learning step number (2nd session)	:	3000
Number of neurons in x direction[1]	:	6 / 24
Number of neurons in y direction[1]	:	4 / 1

Table 1: Parameter values used in the example. In marked items the first value corresponds to the 3-box-wide elements network, the second to 1-box-wide.

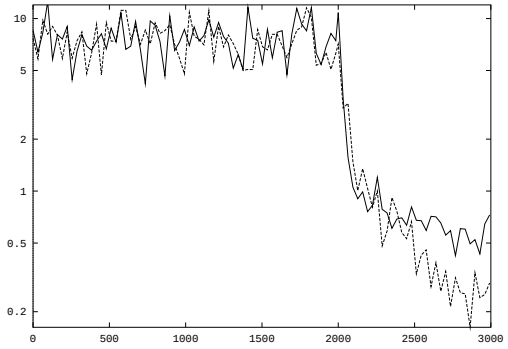

Figure 4: Network errors during training. Training was designed in a way that could reduce training time. In the first 2000 steps the input field was Kohonen digitised and the output was not trained. After the first 2000 steps output training was switched on and output error started to decrease. Output error is given in box units. Box sizes are 100nm (vertical) x 200nm (horizontal). Continuous line corresponds to the network with 1 dimensional input space, broken line to the one with 3 dimensional input surface.

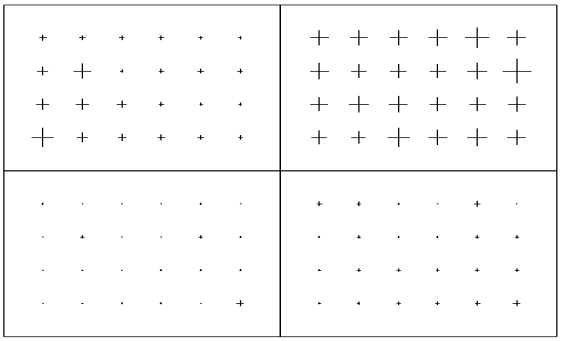

Figure 5: Characteristic behaviour of individual neurons. The size of the crosses is proportional to the magnitude. The position of a cross in the windows corresponds to the position of the neuron in the two-dimensional Kohonen map. The winning frequency is shown in the upper left, the size of the receptive field in the upper right windows. Mean error and mean absolute error are shown in the lower left and lower right windows, respectively.

Our network can also be used to mill some pattern into a smooth surface. This problem

is more difficult to control, however, since heat diffusion based evaporation process tends to smooth surfaces.

We used networks with one and three dimensional input space to control milling. The network parameters were the same as in the previous example. The final error was higher than in the previous case, as expected. According to the simulation results evaporation based one shot laser milling can mill a surface of 1 mm^2 to a depth of a few μm with an error bar of less than 50 nm and resolution of 200 nm. Correcting laser shots are feasible.

The error as a function of time is shown on Fig. 6, a sample surface generated by the network on Fig. 7.

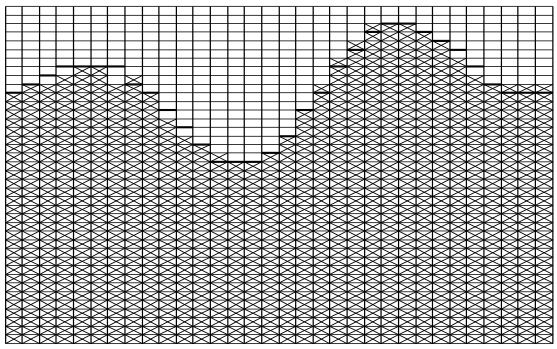

Figure 6: Milling patterns: typical performance of the trained network. Empty boxes correspond to the original surface, filled boxes to surface after the shot. Wide line shows the target surface level. Box sizes are 100nm (vertical) $\times$ 200nm (horizontal).

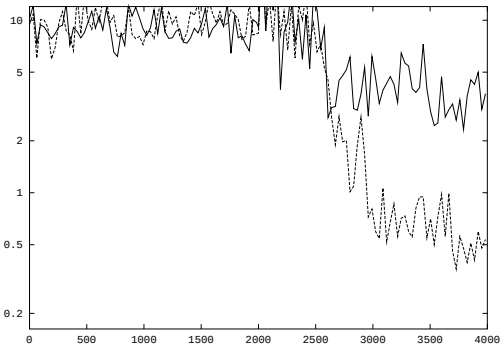

Figure 7: Network errors during training. After the first 2000 steps was output training switched on. Output error is given in box units. Box sizes are 100nm (vertical) $\times$ 200nm (horizontal). Continuous line corresponds to the network with 1 dimensional input space, broken line to the one with 3 dimensional input surface.

6 Conclusion

We have shown that a simple extended Kohonen's network can successfully be applied to control spatially modulated laser pulses for surface manufacturing. It was demonstrated that surface roughness may be decreased by an order of magnitude in one shot over a fairly large surface area. This value is due only to our numerical model that limited the performance through the digitisation error. Digitisation was determined by our computing power.

The adaptive properties of the network are favourable if the environment is likely to change or if the equipment of laser machining has aging components. Same control for evaporative laser milling show attractive properties.

The control with a two-dimensional Kohonen network was satisfactory for surfaces of long wavelength roughness in relation to the spatial resolution of the laser. Shorter wavelength roughness components require networks of higher dimensions.

In the simulations we used a simplified model of the laser-material interaction but it is believed that more realistic models for metals would not substantially change the results in the 100 nanosecond domain.

Acknowledgement

This project was partly supported by the Hungarian Academic Research Fund (Grant # AKA-1-300-2-91-0-750).

References

1 (a) Y. LeCun, B. Bosher, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jacker, Neural Comput **1**, 541 (1989), (b) G. A. Carpenter and S. A. Grossberg, Comput. Vision Graphics and Image Proc. **37**, 54 (1987).

2 (a) J. J. Hopfield, Proc. Nat. Acad. Sci. **87** 3058 (1984), (b) B. Kosko, Appl. Opt. **26**, 4947 (1987).

3 (a) K. S. Narendra and K. Parthasarathy, IEEE Trans. on Neural Networks **1**, 4 (1990), (b) K. S. Narendra and K. Parthasarathy, IEEE Trans. on Neural Networks **2**, 252 (1991).

4 (a) D. Gregoris and V. M. Ristic, Appl. Opt. **26**, 4475 (1987), (b) J. J. J. Dirckx, W. F. Decraemer, and G. Dielis, Appl. Opt. **27**, 1164 (1988), (c) J. S. Lim and M. S. Chung, Appl. Opt. **27**, 2649 (1988), (d) J. S. Lim, J. Kim, and M. S. Chung, Opt. Lett. **14**, 1252 (1989).

5 (a) H. Liu, J. Davis, and R. Lilly, Opt. Lett. **10**, 635 (1985), (b) L. M. Blinov, Electro-Optical and Magneto-Optical Properties of Liquid Crystals (Wiley, New York, 1983), (c) D. Casasent, Appl. Opt. **18**, 2445 (1983).

6 Special Issue on Neural Network Hardware, in IEEE Trans. on Neural Networks **2** (2) (1991)

7 (a) T. Kohonen, in *Proceedings of the* 2^{nd} *Scandinavian Conference on Image Analysis*, edited by E. Oja and O. Simula, Epso, Suomen Hahmontunnistustutkimuksen Seuro (1981), p. 214., (b) T. Kohonen, Self-organization and Associative memory (Springer, Berlin, 1984).

8 H. Ritter, T. Martinez and K. Schulten, Neural Networks **2**, 159 (1988).

9 A. Lőrincz, J. Appl. Phys. **67**, 2567 (1990).

10 J. Anderson, Kybernetik **5**, 113 (1968).