Self-organizing Maps for Optimized Robotic Trajectory Planning Applied to Surface Coating

Maria Tzinava, Konstantinos Delibasis, Spyros Kamnis

Self-organizing maps for optimized robotic trajectory planning applied to surface coating

Maria Tzinava[1], Konstantinos Delibasis[1], Spyros Kamnis[2]

[1] Department of Computer Science and Biomedical Informatics, University of Thessaly, Lamia 35131, Greece
[2] Monitor Coatings, 2 Elm Road, Tyne and Wear NE29 8SE, United Kingdom
tzinavamaria@gmail.com
kdelibasis@gmail.com
spyros@monitorcoatings.com

Abstract. The process of surface coating is widely applied in the manufacturing industry. The accuracy of coating strongly affects the mechanical properties of the coated components. This work suggests the use of Self-Organizing Maps (Kohonen neural networks) for an optimal robotic beam trajectory planning for surface coating applications. The trajectory is defined by the one-dimensional sequence of neurons around a triangulated substrate and the neuron weights are defined as the position, beam vector and node velocity. During the training phase, random triangles are selected according to local curvature and the weights of the neurons whose beam coats the selected triangles are gradually adapted. This is achieved using a complicated coating thickness model as a function of stand-off distance, spray impact angle and beam surface spot speed. Initial results are presented from three objects widely used in manufacturing. The accuracy of this method is validated by comparing the simulated coating resulting from the SOM-planned trajectory to the coating performed for the same objects by an expert.

Keywords: Surface coating, Self-Organizing Maps, Robotic beam trajectory, Triangulated substrate, Coating thickness.

1 Introduction

High value manufacturing sectors are continually seeking new ways to improve the performance and durability of critical components such as those used in aerospace, defense and automotive sectors. Great variations in the application method of coating materials, the deposition kinematic effects, the new complex substrate (component to be coated) designs adopted by the original equipment manufacturers (OEM) and the lack of information related to the mode of operation of the coated components are common problems in the coating sector. As a result, multiple design iterations of thermal or cold spray are required in an effort to converge to a coating plan for any given substrate, which come at high cost for both the OEM's and the supply chain.

Given the industry's transition away from hard chromium plating to more environmentally friendly alternatives, such as spray processes, the need for accurate and flexible coating planning tools comes at an opportune time. Within the context of thermal and cold spraying, a pre-spray optimization strategy has to satisfy several requirements. The central requirement is minimizing the coating thickness variation, achieve uniform or tailored coating properties using a model which cannot be expected to be analytically available. The ability to predict, evaluate and visualize the coating properties on difficult to reach areas of intricate components is of paramount importance and an issue that industrialists come across regularly at the qualification stage of a coating application. A further important requirement is the optimization of the kinematic quality of the robotic spray path, with respect for example to the surface curvature, shadowing and velocity.

Among numerous neural network architectures that could be used to address the robotic gun trajectory planning challenges, one is of particular interest and was introduced by Teuvo Kohonen in the 1980s [1]. Self-organizing map (SOM), sometimes also called Kohonen map, is a single layer neural network with units arranged along an n-dimensional grid. Most applications use two-dimensional, rectangular or hexagonal grids. SOMs use unsupervised, competitive learning to produce low-dimensional projections of high-dimensional data, preserving the similarity and topology relations between the data items. These characteristics are very desirable for our application, since smooth trajectories that follow the object's local curvature without sharp changes of consecutive positions and beam vectors are required, without the need for generating training datasets. SOMs have been used in various applications, despite their simplicity, including visualizations, generation of feature maps, pattern recognition and classification. Some applications focus on control of robotic arm, learning motion maps, collision avoidance for multi-vehicle systems including navigation and robotics [2-4]. Other applications can be found in chemistry [5], disease recognition in medical images, psycholinguistic studies [6], similarity of music recordings [7], maritime applications for the analysis of passive sonar recordings and for planning ship trajectories [8], classification of satellite images [9] and many other.

In this work, we propose the use of a SOM, arranged in 1D with appropriate weight vectors, in order to derive a smooth robotic gun trajectory that performs surface coating within required specifications. The training algorithm has been redesigned to generate optimized coating thickness along the substrate's surface. Results are presented for three different objects, typical in manufacturing, which compare favorably with the ones achieved by an expert.

2 Methodology

2.1 An overview of the proposed algorithm

The proposed SOM is defined as an ordered sequence of 256 nodes (or neurons). The nodes represent the consecutive positions of the spraying gun as it moves along its trajectory with variable speed, pointing at a coating beam direction at each position. Thus, each node i contains its position p_i, the speed magnitude s_i with direction from the current node to the next one and the beam direction $\mathbf{g}_i$. A triangle is considered visible by a node if it lies within a cylindrical beam of a predetermined radius (r_0) and it is not shadowed by another triangle.

For each repetition up to a total number N_{rep}, a nested loop of b_s iterations (referred to as batch size) is performed, in a manner similar to batch stochastic optimization of feedforward neural networks. The triangles visible from at least one node are stored in table **A**. This calculation takes place once for each repetition, in order to reduce the computational complexity.

For each iteration, a random triangle is selected from **A** and the winner node and its neighbors are determined. The correction of the weights δp, δs and $\delta \mathbf{g}$ (position, speed and direction of the beam) of these nodes are appropriately calculated, according to the learning algorithm. During the iterations of the same batch, the corrections δp, δs and $\delta \mathbf{g}$ are accumulated into Δp, Δs and $\Delta \mathbf{g}$ respectively and the node weights (p,s,$\mathbf{g}$) are updated at the end of the batch. These steps are encoded in the pseudocode below:

Initialize neuron weights, calculate barycenter and normal vector for each triangle of the substrate
For rep=1: N_{rep} *// Repetition Loop*
  Calculate table **A**
  Initialize $\Delta p = 0$, $\Delta s = 0$ and $\Delta \mathbf{g} = 0$
  For t=1: b_s *// Iteration Loop*
    Select random triangle r from **A**
    Find winner node w and its neighbors Γ_w
    Calculate $\delta \mathbf{g_w}$, $\delta \mathbf{g_i}$, δs_w, δs_i, δp_w, δp_i where $i \in \Gamma_w$
    $\Delta \mathbf{g_w} = \Delta \mathbf{g_w} + \delta \mathbf{g_w}$, $\Delta \mathbf{g_i} = \Delta \mathbf{g_i} + \delta \mathbf{g_i}$
    $\Delta s_w = \Delta s_w + \delta s_w$, $\Delta s_i = \Delta s_i + \delta s_i$
    $\Delta p_w = \Delta p_w + \delta p_w$, $\Delta p_i = \Delta p_i + \delta p_i$
  Update direction $\mathbf{g_w} = \mathbf{g_w} + \Delta \mathbf{g_w}$, $\mathbf{g_i} = \mathbf{g_i} + \Delta \mathbf{g_i}$
  Update speed $s_w = s_w + \Delta s_w$, $s_i = s_i + \Delta s_i$
  Update position $p_w = p_w + \Delta p_w$, $p_i = p_i + \Delta p_i$

2.2 SOM initialization

A substrate with arbitrary geometry in a triangulated form is inserted in the algorithm, in STL format. The object is placed with its center of mass at the origin of the frame of

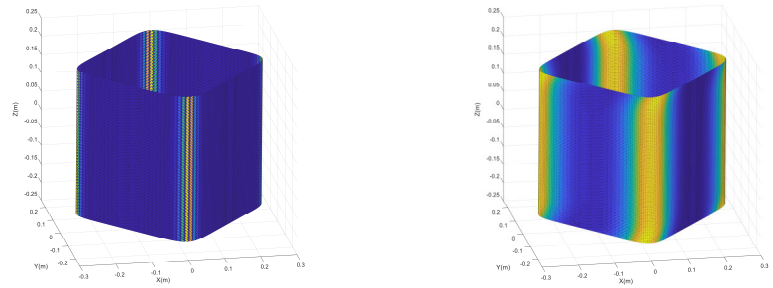
reference (0,0,0). The normal vectors **n** and the position of the barycenter **B** for each triangle of the object is calculated.

For a given substrate, the SOM is initialized with the position of the nodes equally spaced in a full circle, clockwise, around the center of mass of the object on a plane vertical to *z* axis. For each node, the beam's direction is initialized towards the center of the axis system. The initial value of the speed is set to 0.05 m/s for all nodes. The ideal stand-off distance (SoD) and coating thickness are set.

Substrate curvature pre-processing. During the training phase, triangles of the substrate are randomly selected and presented to the SOM. Substrate areas of high curvature require more precise neuron weight adaptation, thus they should be sampled more densely. To this end a quantity *c* that represents the local curvature of the object at each triangle is calculated.

First, a matrix **T** is created, whose i^{th} row $\mathbf{T}_i$, contains the neighbor triangles of triangle *i*. Two triangles are considered to be neighbors if they have at least one common edge. For each triangle *i* the dot product of its normal vector and the normal vectors of all neighbor triangles are calculated and the one with the smallest value is selected for the calculation of the local curvature (see Fig. 1a).

$$c_i^0 = 1 - \min(\mathbf{n}_i \cdot \mathbf{n}_j) \;, j \in \mathbf{T}_i \tag{1}$$



(a) (b)

Fig. 1. a) Initial object curvature in color scale. b) Object curvature in color after the diffusion.

Then an iterative diffusion method is applied to those curvature values (see algorithm below) to achieve a smoother curvature distribution over the substrate (see Fig. 1b). This process takes place once for every new substrate and is not to be confused with the iterative training of the SOM:

```
For each iteration m
  For each triangle i
```

$$k_i = c_i^{m-1} + b\left(\left(\sum_{j=1}^{\beta_i} c_j^{m-1}\right) - \beta_i \cdot c_i^{m-1}\right)$$

$$\mathbf{c}^t = \mathbf{k}$$

where *b* is a constant in [0,1], β is the number of neighbors for each triangle.

2.3 Learning algorithm

In the beginning of each repetition, matrix **A** is constructed. The barycenter of the triangle that is closest to the beam of node (p) and the speed of the beam's projection on this triangle v_{spot} are also calculated [10].

Random triangle selection -Winner determination. For each iteration, a random triangle r from matrix **A** is selected, with probability proportional to substrate's diffused curvature (c), as follows. Triangles with curvature c_k=0 will have zero probability to be randomly selected in the aforementioned method. To alleviate this problem the curvature of each triangle is increased by a constant quantity empirically set equal to 0.1. The curvature value of each triangle is converted to probability by dividing it by the sum of all curvatures and the cumulative curvature is computed.

$$C_i = \frac{1}{\sum_{m=1}^{N} c_m} \sum_{k=1}^{i} c_k \tag{2}$$

A random number ξ is selected according to the uniform distribution between 0 and 1. The triangle r satisfies the following condition:

$$C_r \le \xi < C_{r+1} \tag{3}$$

The triangles that belong to a part of the object with larger curvature are more likely to be selected than the ones that are on a flatter surface. This results in greater object sampling and thus enhanced accuracy in areas with more complex shape.

Subsequently, using the matrix **A**, a winner node is selected, as following. For every node i for which the particular triangle r with barycenter B_r is visible, the parameter b_i is calculated as shown below:

$$b_i = c_r \left(d_{i,r} - d_0 \right) / d_0 + \left(1 - c_r \right) \sin\left(\left| \pi/2 - \theta_{i,r} \right| \right), \tag{4}$$

where $d_{i,r} = \left| B_r - p_i \right|$ is the Euclidean distance between p_i and B_r, referred to as Stand-of-Distance (SoD). Thus, quantity b_i is dominated by the percentage difference of the current and the ideal SoD d_o at object areas with high curvature, whereas at areas with low curvature, the second term that quantifies the deviation of the impact angle from the ideal value of $\pi/2$ becomes dominant. The node with the smallest value of b is declared as the winner node w.

$$w = \arg\min\left(b_i \right) \tag{5}$$

Definition of neighborhood. After the determination of winner node w, a set of neighbor nodes $i \in \Gamma_w$ is defined, with $\left| w - i \right| \le n_b$. Thus, n_b nodes before and after the current winner are considered to be winner's neighbors. In case the geometric setup of nodes

is not a closed loop the number of neighbors affected is smaller, if the winner node is on the edges. The learning rate of each node in the neighborhood is adjusted by a weight calculated using a Gaussian function of node index with respect to the winner node and standard deviation that decreases linearly from σ_0 to σ_1 with the number of epochs n_{ep}:

$$a_w(i) = a_{w,i} = \begin{cases} \exp\left(-(w-i)^2 / \left(2\sigma_n^{\,2}\right)\right), |i-w| \le n_b \\ 0 \qquad\qquad\qquad\qquad\quad , \text{otherwise} \end{cases} \tag{6}$$

The standard deviation is obtained as following:

$$\sigma_n = n_b \left(\frac{\sigma_1 - \sigma_0}{n_{ep}} k + \sigma_0 \right) \tag{7}$$

The variables of these nodes are affected in the same way to the winner node, but to a lesser degree, depending on their distance from the winner (see Fig.2).

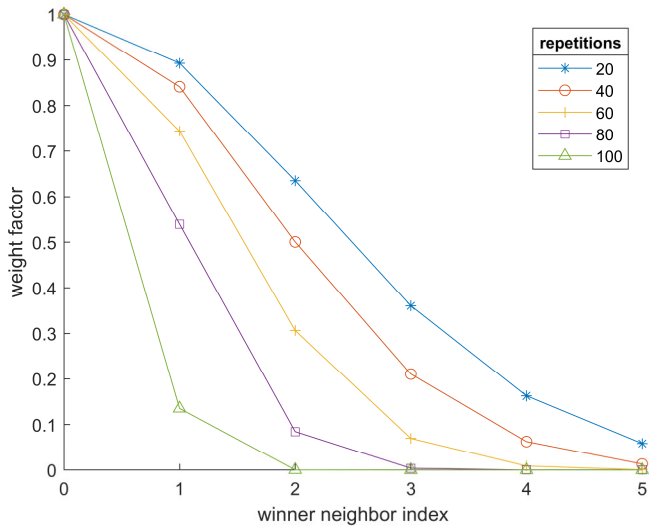


Fig. 2. Change of weight factors with repetitions

Coating thickness calculation. The coating thickness h for a random triangle r that is being coated by neuron i, is calculated as a function of the following variables: stand-off distance (SoD) d, impact angle θ and the speed of beam's spot on the triangle r (spot speed) v_{spot}. For node i positioned at p_i with beam vector $\mathbf{g_i}$, that coats triangle r with normal vector n_r, (visible by node i), the v_{spot} is calculated according to [10] and the SoD and the impact angle θ_i, r are calculated as follows:

$$d_{i,r} = \left\| p_i - B_r \right\| \tag{8}$$

$$\theta_{i,r} = \cos^{-1}\left(\mathbf{g}_i \cdot \mathbf{n}_r\right) \tag{9}$$

For the calculation of coating thickness h, we derive the thickness h_o for the given SoD and impact angle θ_i, r using bilinear interpolation, according to [10] as follows:

$$h_o = a_1 d_{i,r} + a_2 \theta_{i,r} + a_3 d_{i,r} \theta_{i,r} \tag{10}$$

where a_1, a_2 and a_3 are parameters calculated from experimental results [10],[11].
The calculated thickness h_0 corresponds to spot speed of 502 m/s. In order to convert the thickness to the current spot speed we apply the following:

$$h = h_o \frac{f_h\left(v_{spot}\right)}{f_h\left(v_{spot} = 502\right)} \tag{11}$$

where f_h is the hyperbolic interpolation of thickness versus spot speed is performed according to

$$f_h\left(v_{spot}\right) = \frac{1}{b v_{spot} + c} \tag{12}$$

The parameters b, c are calculated using experimental results [11].

Updating the neuron weights. The first variable that is updated during SOM learning is the direction of the beam. Aiming at an impact angle between the winner node and triangle close to π/2, the beam direction of the winner w and the neighbor nodes i, is updated as shown below:

$$\delta \mathbf{g}_i^t = sign(\mathbf{n}_r \cdot \mathbf{g}_w) \lambda_v a_{w,i} \mathbf{n}_r \tag{13}$$

where $\delta \mathbf{g}_i^t$ is the change of the beam node vector during iteration t, $\mathbf{g}_w$ is the beam vector of the winner node, $\mathbf{n}_r$ is the normal vector of random triangle r and $\lambda_v = 0.1$. It follows that the update for the winner becomes $\delta \mathbf{g}_w^t = sign(\mathbf{n}_r \cdot \mathbf{g}_w) \lambda_v \mathbf{n}_r$, since $a_{w,w} = 1$ according to Eq.(6), whereas for neurons outside the winner's neighborhood ($|w - n| \le n_b$), $a_{w,i} = 0$.

Subsequently, the node speed s, which represents the speed of the gun as it passes through the positions of the nodes in reference, is updated. The initial value of the speed is set to 0.05 m/s with direction from the current node to next one clockwisely. The speed changes in order for an ideal thickness to be achieved in the spraying process. The formula for the node speed update δs_i^t during iteration t is:

$$\delta s_i^t = \begin{cases} (h_i - \hat{h}) \lambda_s a_{w,i} {s_w}^2 & , \ i = w \\ \left(s_w - s_i\right) \lambda_s a_{w,i} & , \ i \neq w \end{cases}, \tag{14}$$

where h is the calculated thickness for triangles visible by the winner node beam, if the spraying gun was placed in the position of the winner node, h_i is an ideal thickness, $\lambda_s = 0.1$, s_w is the speed of the winner node and s_i the neighbor's speed. The value of speed for neighbor nodes is updated differently. The change is defined in reference to the speed difference between the winner's and each neighbor's speed.

Then, the position of the nodes is updated. The vector δp_i that defines this change is calculated using the following formula:

$$\delta p_i^t = \frac{D_i - d_o}{\max(D, d_o)} \lambda_p a_{w,i} \mathbf{g}_i, \tag{15}$$

where $D = \left| B_r - \left(p_i + \Delta p_i^{t-1} \right) \right|$ and $\lambda_p = 0.2$. Fig.3 below depicts schematically the update of the position and beam direction of a winner node and a neighbor node for a randomly selected triangle r of the object during one iteration:

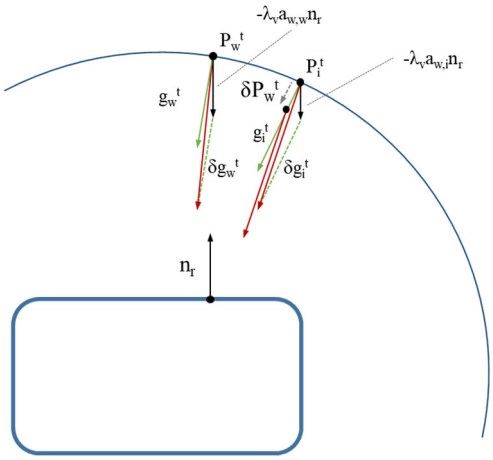


Fig. 3. Position and beam direction correction of a winner and a neighbor node for one iteration

As mentioned in the overview of the proposed system, the evolution of the proposed SOM is performed in N_{rep} repetitions of batches with b_s iterations. Thus, the updates for each neuron are accumulated during each batch:

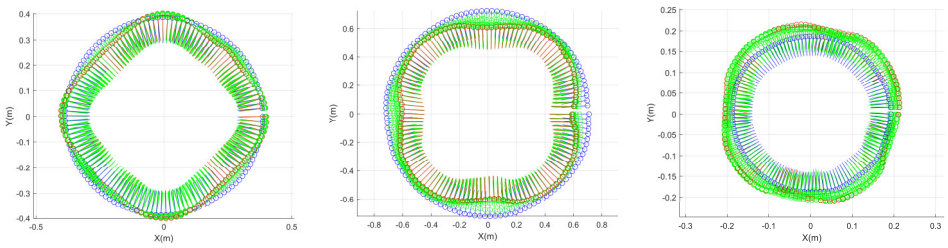
$$\Delta \mathbf{g}_i = \sum_{t=1}^{b_s} \delta \mathbf{g}_i^t, \quad \Delta s_i = \sum_{t=1}^{b_s} \delta s_i^t, \quad \Delta p_i = \sum_{t=1}^{b_s} \delta p_i^t \tag{16}$$

and are used to update the neuron weights at the end of each batch:

$$\mathbf{g}_i^{rep} = \mathbf{g}_i^{rep-1} + \Delta \mathbf{g}_i^t, \quad s_i^{rep} = s_i^{rep-1} + \Delta s_i^t, \quad p_i^{rep} = p_i^{rep-1} + \Delta p_i^t \tag{17}$$

3 Results

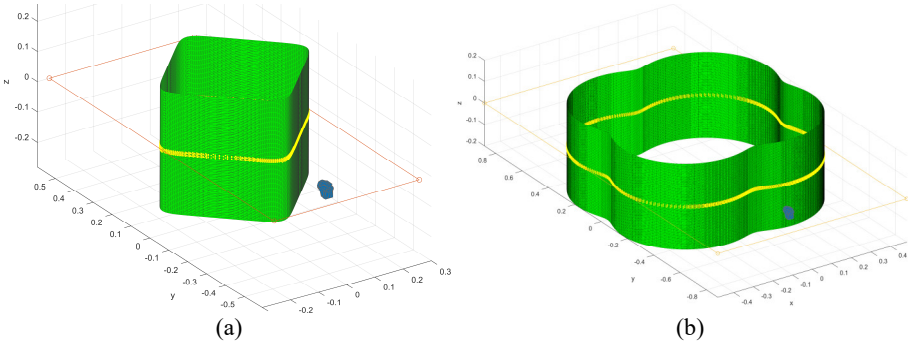
A billet mold, a component with epitrochoid cross section and a mud-rotor are typical objects used as substrates for coating, thus they were selected for our test objects. Fig. 4 summarizes the training of the SOM for 500 repetitions and batch size equal to 10 (thus 5000 iterations in total), for the three test objects. The node position, beam vector and speed are depicted for the initial repetition, every 100 intermediate repetitions and the final one (blue, green and red color respectively). The length of the vectors is analogous to the gun speed.



(a) (b) (c)

Fig. 4. Initial, every 100 intermediate repetitions and final repetition node positions and beam vectors (blue, green and red color respectively) for a) the billet mold, b) epitrochoid and c) the mud-rotor. SOM evolution was performed for 500 repetitions and batch size equal to 10 for all three objects. The length of the vectors is analogous to the gun speed.

The coating achieved by the proposed algorithm is shown in Fig. 5 in comparison to the manual, expert-based coating for the billet mold (left column) and the epitrochoid (right column). The triangulated substrates are shown in the 1st row, with the coating plane indicated in yellow color. The coating thickness at the indicated plane achieved by the SOM-based method and by the expert is shown in the 2nd and 3rd raw respectively.



(a) (b)

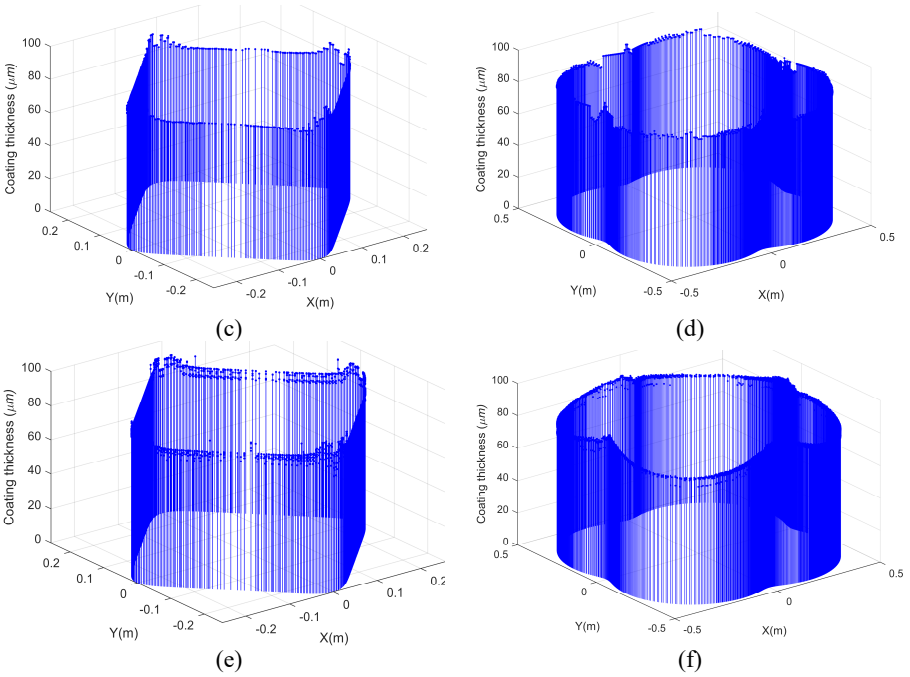


Fig. 5. Coating plane of the triangulated substrates: a) billet mold and b) epitrochoid. Coating thickness of the billet mold using c) proposed method and e) manual method and of the epitrochoid substrate using d) proposed method and f) manual method.

The coating achieved by the proposed algorithm for the mud rotor is shown in Fig. 6 at the plane indicated in Fig.6a. The coating thickness along the object intersection marked in yellow is plotted for the SOM based method and for the expert in Fig.6b and Fig.6c respectively. It can be visually observed that the thickness achieved by the SOM method is more uniform than the one achieved by the expert.

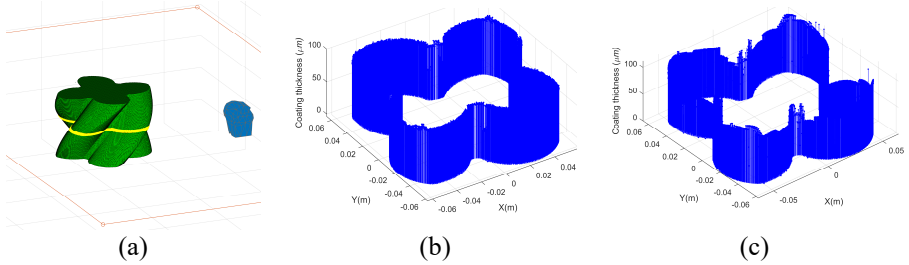


Fig. 6. a) Coating plane of the triangulated mud rotor. Coating thickness of the mud rotor at the selected plane using the b) proposed method and c) manual method

The coating thickness distribution is summarized using boxplots for the three test objects, for the proposed SOM-based and the expert-based coating. It can be observed in Fig.7 that the thickness achieved by the SOM-based coating is better distributed

round the required value, with both the 1st, 3rd quartile and the minimum and maximum thickness being closer to the ideal value, compared to the expert-based result.

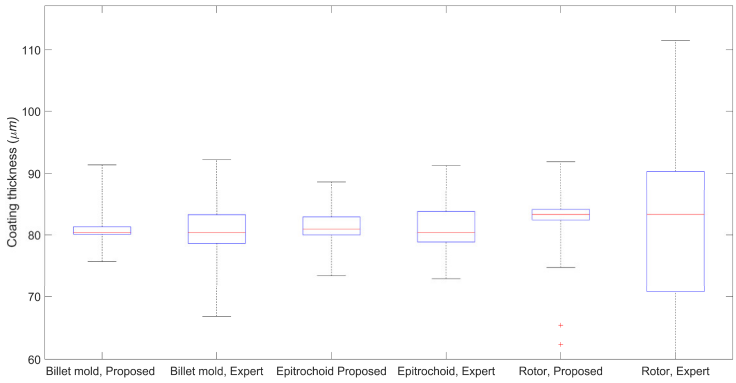
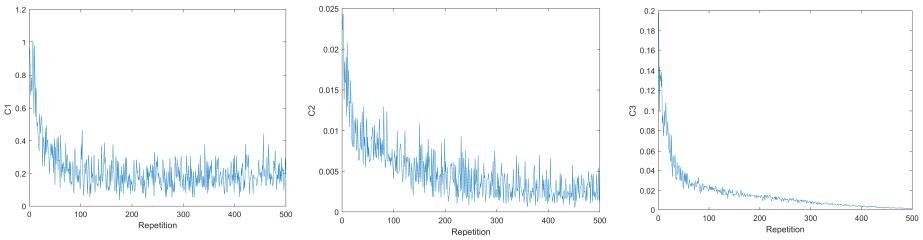


Fig. 7. Distribution of thickness values of proposed method for triangles of the coating plane of the object in comparison with the ones derived from a manual process for the three test objects.

The convergence of the SOM is assessed by calculating the following quantities:

$$C_1(rep) = N - \sum_{i=1}^{N}\left(\mathbf{g}_i^{rep-1} \cdot \overline{\Delta\mathbf{g}_i^{rep}}\right),\ C_2(rep) = \sum_{i=1}^{N}\left|\Delta s_i^{rep}\right|,\ C_3(rep) = \sum_{i=1}^{N}\left\|\Delta p_i^{rep}\right\| \quad (18)$$

Typical results of the evolution of the quantities C_1, C_2 and C_3 for the coating of the billet mold object are shown in Fig. 8.



(a) (b) (c)

Fig. 8. The evolution of quantities C_1, C_2 and C_3 for the SOM-based coating of the billet mold.

4 Conclusions and further work

A SOM-based method for planning robotic arm trajectories for the optimization of surface coating processes has been presented. The proposed approach adapts the robotic gun kinematics utilizing a complicated coating thickness model to generate coating

thickness within strict specifications for several surfaces. Three different objects, typical in the manufacturing industry have been tested, in comparison to an expert. Results show that the distribution of SOM-coating thickness is superior to the one achieved by an expert. The coating planning is performed within few minutes for an object with 10^5 triangles, using Matlab in a MW-Windows laptop (Intel i7@2.6GHz, 16GB RAM).

Further work incudes the extension of SOM on a 2D mesh of neurons that will cover arbitrarily complicated object geometries.

Acknowledgment

This work was partially financially supported by the Interdepartmental Postgraduate Programme "Informatics and Computational Biomedicine", School of Science, University of Thessaly, Greece.

References

1. Kohonen, T.: Self-organized formation of topologically correct feature maps, Biol. Cybern. 43, pp. 59-69, (1982).
2. Kubota, N., Nojima, Y., Kojima, F., Fukuda, T., & Shibata, S.: Intelligent control of self-organizing manufacturing system with local learning mechanism. In: IECON Proceedings (Industrial Electronics Conference) (1999).
3. Razavian, A.A., Sun, J.: Cognitive based adaptive path-planning algorithm for autonomous robotic vehicles. In: Proceedings of the IEEE SoutheastCon, Ft. Lauderdale, FL, USA, 8–10 April 2005.
4. Miljković, D.: Brief review of self-organizing maps. In: 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 1061-1066, Opatija (2017).
5. Maltarollo, V. G., Honório, K. M., & da Silva, A. B. F.: Applications of artificial neural networks in chemical problems. Artificial neural networks-architectures and applications, 203-223, InTech, (2013).
6. Kohonen, T.: MATLAB Implementations and Applications of the Self-Organizing Map, Unigrafia, Helsinki, Finland, 2014.
7. Pampalk, E., Dixon S. and Widmer, G.: Exploring music collections by browsing different views, Computer Musical Journal, Vol. 28, No. 2, pp. 49-62, (2004).
8. Lobo, V. J.: Application of self-organizing maps to the maritime environment. In Information Fusion and Geographic Information Systems (pp. 19-36). Springer, Berlin, Heidelberg, (2009).
9. Richardson, A. J., Risien, C., & Shillington, F. A. (2003). Using self-organizing maps to identify patterns in satellite imagery. *Progress in Oceanography*, *59*(2-3), 223-239.
10. Tzinava, M., Delibasis, K., Allcock, B., & Kamnis, S.: A general-purpose spray coating deposition software simulator. Surface and Coatings Technology, 399, 126148, (2020).
11. Katranidis V., Gu S., Allcock B., Kamnis S.: Experimental study of high velocity oxy-fuel sprayed WC-17Co coatings applied on complex geometries. Part A: influence of kinematic spray parameters on thickness, porosity, residual stresses and microhardness, Surf. Coat. Technol. 311, 206–215, (2017).