



Training AI-Based Feature Extraction Algorithms, for Micro CT Images, Using Synthesized Data

Matthew Konnik¹ · Bahar Ahmadi¹ · Nicholas May¹ · Joseph Favata¹ · Zahra Shahbazi² · Sina Shahbazzmohamadi¹ · Pouya Tavousi¹

Received: 31 August 2020 / Accepted: 3 February 2021 / Published online: 18 February 2021
© The Author(s) 2021

Abstract

X-ray computed tomography (CT) is a powerful technique for non-destructive volumetric inspection of objects and is widely used for studying internal structures of a large variety of sample types. The raw data obtained through an X-ray CT practice is a gray-scale 3D array of voxels. This data must undergo a geometric feature extraction process before it can be used for interpretation purposes. Such feature extraction process is conventionally done manually, but with the ever-increasing trend of image data sizes and the interest in identifying more miniature features, automated feature extraction methods are sought. Given the fact that conventional computer-vision-based methods, which attempt to segment images into partitions using techniques such as thresholding, are often only useful for aiding the manual feature extraction process, machine-learning based algorithms are becoming popular to develop fully automated feature extraction processes. Nevertheless, the machine-learning algorithms require a huge pool of labeled data for proper training, which is often unavailable. We propose to address this shortage, through a data synthesis procedure. We will do so by fabricating miniature features, with known geometry, position and orientation on thin silicon wafer layers using a femtosecond laser machining system, followed by stacking these layers to construct a 3D object with internal features, and finally obtaining the X-ray CT image of the resulting 3D object. Given that the exact geometry, position and orientation of the fabricated features are known, the X-ray CT image is inherently labeled and is ready to be used for training the machine learning algorithms for automated feature extraction. Through several examples, we will showcase: (1) the capability of synthesizing features of arbitrary geometries and their corresponding labeled images; and (2) use of the synthesized data for training machine-learning based shape classifiers and features parameter extractors.

1 Introduction

Nondestructive volumetric analysis of samples, enabled by X-ray computed tomography (CT), has attracted scientists and engineers from a wide spectrum of disciplines that are interested in identification and measurement of miniature internal features of their samples [8]. While obtaining X-ray CT images of arbitrary objects has become a straightforward procedure, which only requires adjustment of a few imaging

parameters (e.g., energy, # of projections), the interpretation of the resulting 3D images is still a challenging task [25]. For proper interpretation of an X-ray CT image, one must be able to extract well-defined geometric features from the raw data, where the raw data is a gray-scale 3D array of voxels [27]. Conventionally this task is performed manually by the subject matter experts (SMEs). This faces several problems: (1) SMEs have a subjective perception of features. This fact impacts the achievable accuracy of measurements in the manual processes; (2) manual feature extraction is subject to error [19] [2]; and (3) with the trending growth of image data size and the interest in identifying more miniature features, the manual feature extraction practice is becoming increasingly tedious, if not impractical. This calls for automated methods that can extract features accurately and with a high-throughput. The most common approach for achieving this goal is use of computer-vision (CV) techniques, to segment the images into distinct partitions [7] [30] [20] [3] [29] [6]

✉ Sina Shahbazzmohamadi
sina.shahbazzmohamadi@uconn.edu

✉ Pouya Tavousi
pouya.tavousi@uconn.edu

¹ UConn Tech Park, University of Connecticut, Storrs, USA

² Mechanical Engineering Department, Manhattan College, New York, USA

[24] [32] [9], which could hopefully be used for extracting meaningful geometric features. For example, in thresholding, a common CV technique, intensity values and a preset thresholding constant will be used to assign a label to each pixel (voxel) in the 2D (3D) image. Such label is shared among all the pixels (voxels) of the same partition and the result of the segmentation process is a 2D (3D) image that is partitioned into several groups of connected pixels (voxels). Although, the CV techniques may offer an automated process in the absence of image noise (i.e., features that are of no interest), their performance drops drastically in dealing with noise which is prevalent in any image obtained from an X-ray CT practice [17] [4] (Dambreville, Rathi and Tannen 2006). More elaborately, efficiencies associated with the scintillator and photodiode panel on the CT detector regularly contribute to the production of random and statistical noise in the final image through discontinuities in the amount of incident energy converted across scintillator and photodiode areas. Noise production is also induced in the final image through analog circuitry and analog-to-digital converters integral to image reconstruction. These modes of noise production can be mitigated, but not completely removed. Therefore, in practice, the CV methods are only used to assist the manual feature extraction process and cannot provide a fully automated feature extraction process. The success of machine learning (ML) algorithms in automating tasks that are not analytically well-defined, promises use of these methods for automated feature extraction, as a superior alternative to CV-based methods. The idea is to train a machine learning algorithm with sufficient ground truth data (Kotsiantis, Zaharakis and Pintelas, Supervised machine learning: A review of classification techniques 2007) (Singh, Thakur and Sharma 2016) [22] and then use it for automated feature extraction. Here, the ground truth data are obtained from labeled X-ray CT images. Each data point consists of: (1) raw data in the form of a gray-scale 3D array of voxels and (2) the corresponding feature. The caveat is that the proper training of a machine learning algorithm demands huge amounts of labeled data. This is a multifold challenge. The necessity of labeling the raw data manually, makes this process extremely tedious, if not impractical. In addition, labeling process will be subjective, with different outcomes expected from different SMEs. Further, such manual process is subject to error. Double-checking and triple-checking practices to eliminate such error would add to the required time and effort for conducting the manual segmentation. Existing attempts to crowdsource the manual labeling task using volunteers [1] [12] [26] face the challenge of volunteer's lack of motivation. Also, a major drawback of the state-of-the-art software methods that enable manual segmentation is that the user, at each point in time, has only a 2D perception of the sample, which acts as a prohibitive factor on the way of conducting a realistic feature

extraction, given than features extend in three dimensions. Moreover, the data that can be naturally found in the sample images might not be sufficient or diverse enough for training the machine learning algorithms. We propose to address these issues, through imaging of synthesized features. In this process, for any application, depending on the expected classes of features, certain geometries will be modeled in silico and in a parametric fashion. For example, for studying the electrical connections of a multi-layer PCB board, the parametric model of wirings and connections will be generated. Next, through a combinatorial algorithm, different sets of random values will be assigned to the feature parameters to enumerate a large set of features of different shapes, sizes, positions and orientations. This is followed by embedding the computer-synthesized features into the physical sample. Two key considerations in doing so are: (1) the features must be internal rather than appearing on the surface of the object to suit the application of X-ray CT; and (2) the geometric size of these features must be comparable to the naturally-found instance of these features. To satisfy the first condition, we adopted a two-step solution: (a) create features on thin layer objects; and (b) stack the layers to form a 3D object. To satisfy the second condition, we employed the femtosecond pulsed laser which enabled us to fabricate distinct features down to tens of micrometers. The resulting 3D object would then undergo an X-ray CT process. The resulting image of each feature will be paired with the known geometric parameters that were used as a recipe for machining that feature. The resulting pair will be one of many training data points in the entire training data set. Taking the proposed approach compensates for the shortage and lack of diversity of naturally found data. Furthermore, since the exact geometry, position and orientation of each feature is known a priori, the proposed approach offers a significant improvement to the manually labeled data in terms of accuracy and robustness. Finally, needless to say, this approach can be fully automated.

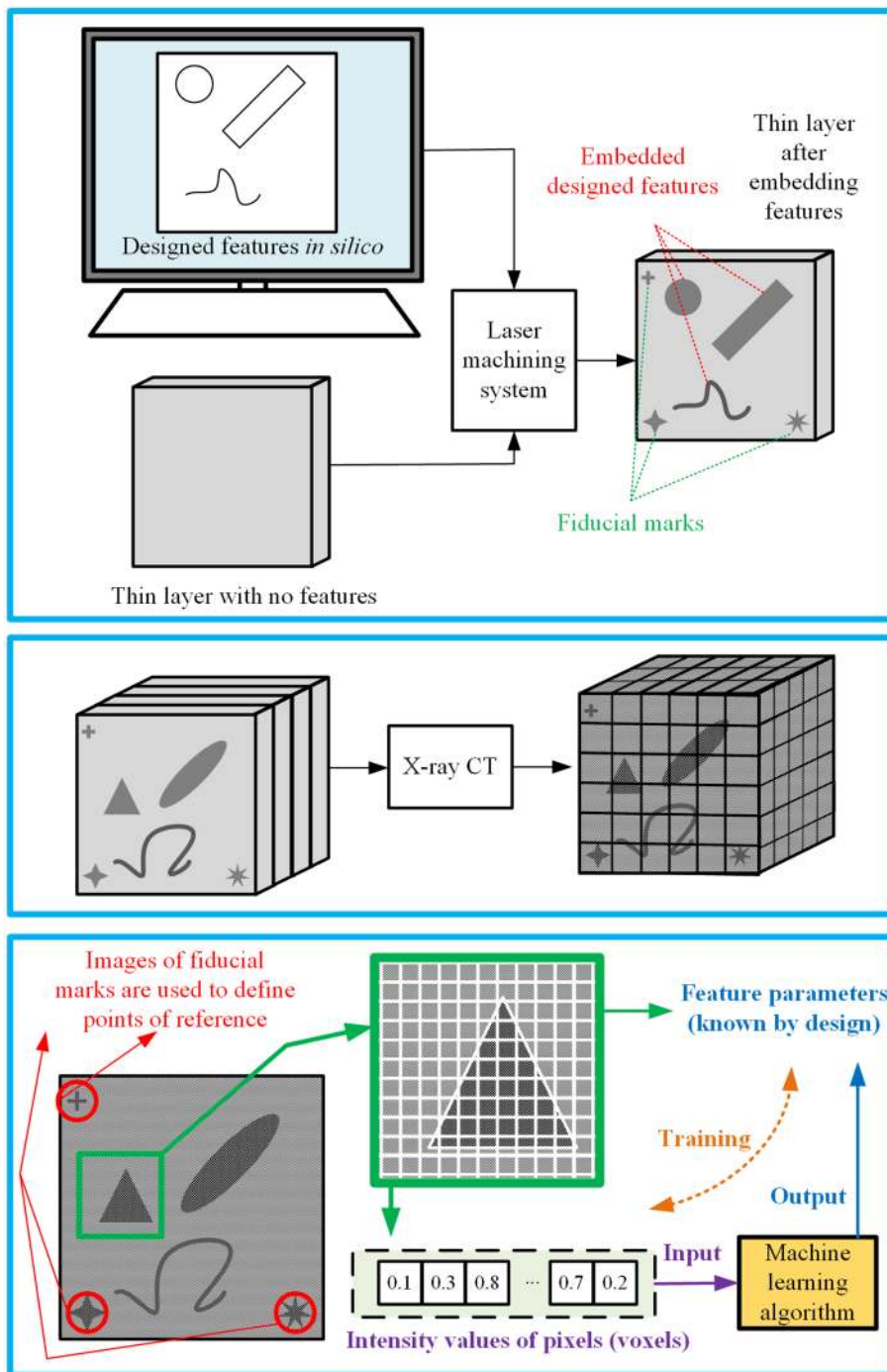
2 Methods

Figure 1 schematically describes the designed workflow for generating synthesized data and using this data for training the machine learning algorithms. A step-wise elaboration of the method is provided in the following.

2.1 Step 1: Feature Design

The layout of shape, size, position, and orientation of features was designed in the Direct Machining Control (DMC). The DMC software, which controls the laser and laser marking system per user's input recipe, allows for the creation of numerous geometries and augmentation of their position,

Fig. 1 Schematic description of the proposed method



size, orientation, and machining parameters on a grid described by the laser scan field. The preliminary features we chose to include in our design included equilateral triangles, rectangles, and ellipses. Although limited, the selected features form a subset of primitive geometries that reflect some key variations of the much large population of possible geometries. Specifically, the ability to identify a regular triangle using the proposed method, would imply the potential of the proposed method to identify any other regular

polygons. Further, rectangles and ellipses were incorporated to investigate the potential of the proposed method in identifying nonregular geometries and various curvatures. As the most immediate application concerns autonomous detection of features in micro-CT imaging environments, it was crucial to not only vary position and geometric size, but also orientation of the features such that any sample with any mounted position can be accommodated without additional processing. Once the parameters of interest were identified,

an appropriate range and step-size for each parameter had to be selected. For this study, we chose to vary length and width of the ellipses and rectangles from 100 to 550 microns, through 9 increments each. Equilateral triangle size was varied by changing the diameter of a circle, by which it can be inscribed. This diameter was varied from 100 to 550 microns through 81 increments to match the number of permutations available for the rectangles and ellipses. The orientation (i.e. angle made with the horizontal axis) of each geometry was also varied from 0 to 90 degrees, throughout 10 increments. Upon selection of feature geometry and variable parameter step-size and range, for-loops were designed in the DMC software to iterate through all possible permutations and generate the appropriate geometries, sizes, and orientations. (Fig. 2).

The feature parameters, including their shape, size, position and orientation are defined relative to a universal coordinate system, defined by the software. In designing the features, a critical consideration was that such universal coordinates system could be retrieved from the lasered sample or its corresponding image. Upon placement of the lasered silicon wafers into the micro-CT, it is probable that a feature of known orientation does not have the expected orientation relative to the detector, or that the locations of lasered areas on different silicon wafers do not line up. Fiducial marks were incorporated into the design layout to absolutize the position and orientation systems which would otherwise be relative. In addition to the two-dimensional considerations,

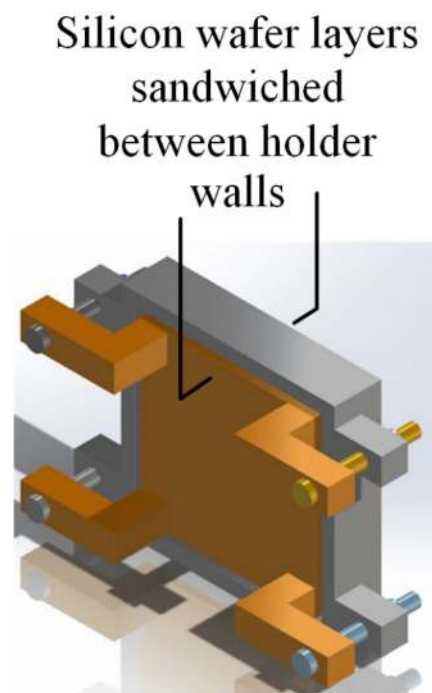


Fig. 2 Silicon wafer layer holding mechanism

the fiducial marks were also required to ensure coplanarity between the image extraction plane and the actual feature plane.

2.2 Embedding Features

As mentioned earlier, polished silicon wafers were chosen as the layers for embedding the features. The reason behind this selection was two-fold: (1) silicon wafers, depending on their grade, are machined to very tight tolerances and maintain very precise geometry and thickness; (2) the authors are experienced in lasing and imaging polished silicon wafers. For this study we used 76.2 mm diameter P-type test grade silicon wafers, single side polished, with a measured thickness of $523 \mu\text{m} \pm 5 \mu\text{m}$.

A mechanism (seen in Fig. 3) was initially designed to hold the layers together during the micro-CT stage of the process and was implemented in the first trial run; however, during the second run, carbon tape was used to hold the layers together (Fig. 4) to reduce data size and increase the overall transmissivity of the subject in the presence of X-rays.

Features, which were used to generate synthesized data, were created using a KM Labs Y-Fi HP ytterbium fiber femtosecond laser machining setup with 20-W average power, a FWHM center wavelength of 1042 nm, 130 fs mean pulse width, an adjustable repetition rate ranging from 0.5 MHz to 15 MHz and an optional burst mode. The scan-head used for rastering the beam was a BasiCube 10, featuring an f-theta lens. For our purposes in lasing features on polished silicon wafers, it was found through a parameterization study that using 3-pulse bursts at a repetition rate of 2 MHz and amplifier power of 60%, yielding an energy per pulse of $2.2 \mu\text{J}$, and rastering for 50 cycles, at a line spacing of $9 \mu\text{m}$ and a marking speed of 0.38 m/s, was sufficient to change the pathlength and absorptivity of the features such that they

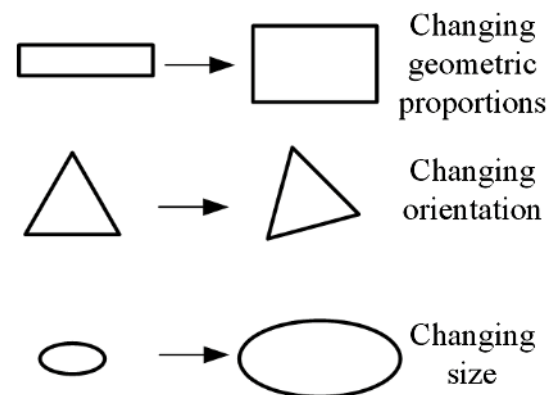


Fig. 3 Changing geometric parameters to produce different shapes

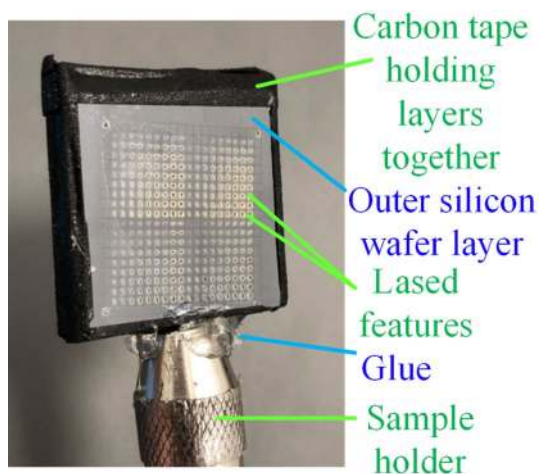


Fig. 4 Second trial silicon wafer stack mounted and prepped for imaging

were easily distinguishable in the reconstructed image. Top part of Fig. 5 illustrates the laser setup.

3 X-ray CT

The micro-CT system used to conduct the imaging processes was a ZEISS Xradia 520 Versa (Fig. 5 bottom). Imaging parameters resulting in acceptable resolution and contrast for the purpose of discerning the geometries were as follows: a 70 kV and 85 μ A source setting, an optical magnification of 0.4X, source and detector distances resulting in a pixel size of 25.34 μ m with an image size of 1004 \times 1024, a mean exposure time of 2.36 s, reconstruction binning of 1, an LE4 source filter, and 1601 projections through 360 degrees rotation.

Upon completion of the micro-CT scan, the resulting reconstructed image files are of type ‘.txm’ and ‘.txrm’. Using the Xradia’s XMController software, the ‘*.txm’ file, which contains a linear z-stack of 1018 images was exported as 1018 individual ‘*.tif’ files, an example of which can be seen in Fig. 6. The ‘*.tif’ files were exported from the XMController software and the resulting images were post-processed in MATLAB (MATLAB 2019b, The MathWorks, Inc., Natick, Massachusetts, United States). The main objective of the post-processing procedure was to prepare the images for data extraction in the context and format of the implemented artificial intelligence (AI) algorithm. The processing procedure began by reading in all 1018 images in MATLAB and creating a 4D voxel value matrix. Of the four dimensions present in the matrix, three were spatial, while one was pigmental. The spatial dimensions were used to identify a pixel location on a specific image plane, each location itself was associated with three intensity values

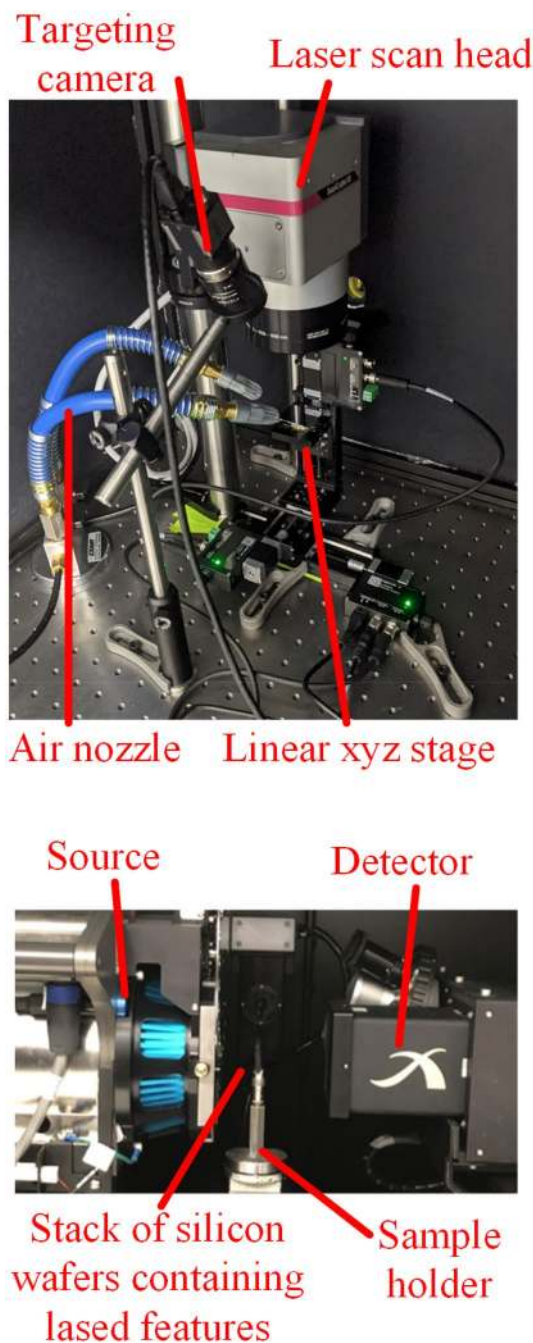


Fig. 5 Top: Laser setup. Bottom: Second trial stack inside the ZEISS Xradia 520 Versa

belonging to the pigmental dimension (red, green, and blue). After matrix formation, it then underwent a rotation transformation such that displaying the first three indices as an image showed the features of interest. Additionally, fine tuning of image rotation in three dimensions was performed with the help of the previously discussed fiducial marks to ensure coplanarity of the feature plane with the image plane. Upon completion of the fine tuning, single images were able

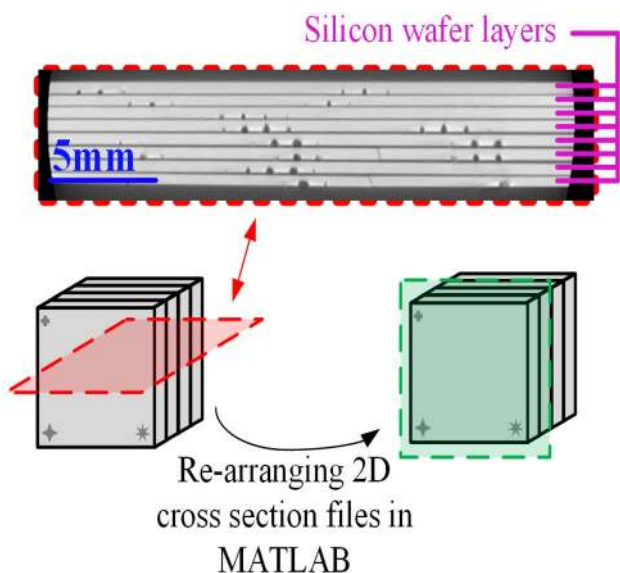


Fig. 6 An example of the ‘*.tif’ files exported from the ‘*.txm’ reconstruction file. The cross section file produced by XMController software (red dotted box) reflects a cross section normal to the silicon wafer layers. By rearranging the voxels in MATLAB, in an automated fashion, new cross sections will be parallel to the silicon wafer layers (green dotted box). See Fig. 7 for an example of the resulting cross sections

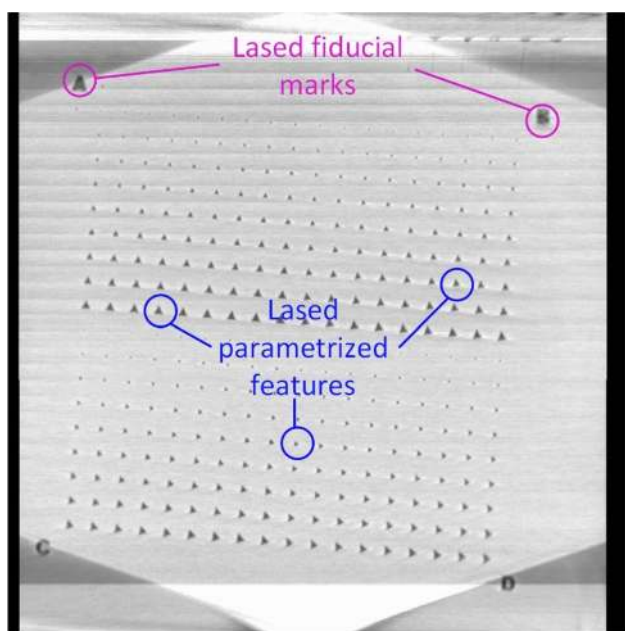


Fig. 7 Image example after CT scan post-processing in MATLAB

to be isolated from the 4D matrix, an example of which is displayed in Fig. 7. Note that more cross sections, containing the features, were produced than needed. Among them, those containing the highest contrast were chosen.

3.1 Preparing Training Data from Resulting Images

After CT scan post-processing, information had to be extracted from the final images and formatted to accommodate training and testing data sets needed for training and testing the neural network algorithm. Knowing that separate information from each individual feature would need to be fed to the algorithm, the features, residing on each image that was previously isolated from the 4D matrix, were segregated mathematically. It is important to note that the coordinate systems between that of the cross section image and that of the recipe (or DMC software) are different. For accurate segregation of features, in an automated fashion, it is key that the two coordinate systems are aligned. To do so, we started by creating a 2D array or pixels (i.e. an image) reflecting the laser machining recipe (which contained the designed features as prescribed by the recipe). This 2D array had identical dimensions to those held by the images isolated from the 4D matrix – 1004 × 1005. The pixel locations of the center of the two fiducial marks were noted on both the 2D array, reflecting the laser machining recipe, and the image, to be segregated. The goal was to find a transformation matrix that would overlay the two coordinate systems. For this, the scaling factor, translation factors, and rotation factors of a transformation matrix of the following form were back calculated using the known pixel locations.

$$\begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} S & 0 & 0 \\ 0 & S & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} J_f \\ I_f \\ 1 \end{bmatrix} = \begin{bmatrix} j_f \\ i_f \\ 1 \end{bmatrix}$$

where t_x and t_y are the translation factors, θ is the rotation factor, S is the scaling factor (only a single scaling factor was required because of image size equality), J_f and I_f are the column and row indices of fiducial mark f belonging to the laser machining recipe image, while j_f and i_f are the column and row indices of fiducial mark f belonging to the isolated image (Fig. 8). After back calculating the required transformation factors, the transformation matrix was used to move and relabel all of the isolated image pixels to their expected location based on the lasing recipe. Then, using the known locations of lasing, the pixels corresponding to each lased feature were extracted, producing images like that displayed in Fig. 9.

Upon creation of the individual feature images, excess information was removed, and the relevant pixel information was collimated to accommodate the input format of the AI algorithm. After collimation of all feature images, the vectors were concatenated such that all features, represented in their own respective row, belonged to the same data matrix. As the inputs to the algorithm could simply be the pixel values of each features respective image, preparation of image data for algorithm training was now complete,

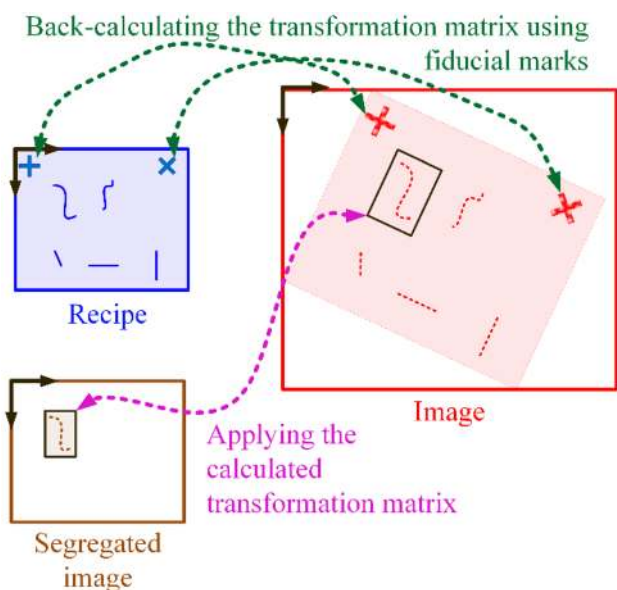


Fig. 8 Segregating feature pixels after finding the coordinate transformation between the recipe and the X-ray image

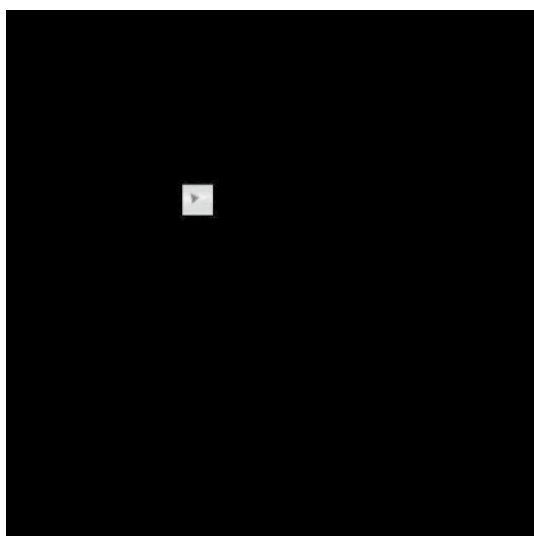


Fig. 9 Single feature image extraction example

with the exception of output selection and inclusion in the data matrix. After preliminary trials, however, it was realized that the success rate of the algorithm would be higher, and training time lower, if an augmented form of the feature images were used (Fig. 10). Rather than having 2,601 pixel value inputs (each feature image was 51 by 51 pixels), it was elected to have 104 data value inputs, of which there were four types. One data value input was equal to the summation of the entire 2,601 pixels belonging to the respective feature image, while another was equal to the standard deviation of those same pixels. The remaining 102 data value inputs were

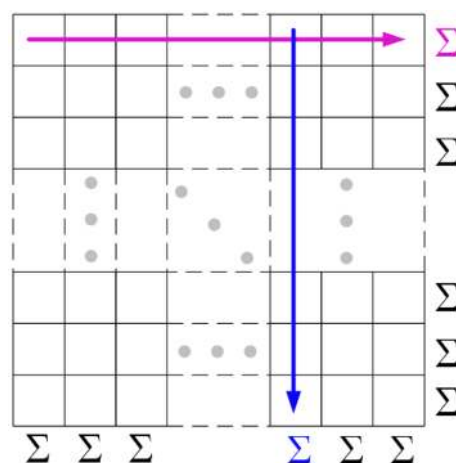


Fig. 10 Using an augmented approach for preparing inputs for the ANN

formed by 51 data values, each described by the summation of a respective row, and 51 data values described by the summation of a respective column.

Having a set geometry, orientation, and size, numerous outputs to the neural network could have been tested, such as aspect ratio, angle, area, equivalent radius, length, etc. For the purposes of showcasing this data synthesis procedure, it was determined that testing angle, area, and various classification circumstances would be sufficient. For training, the output of choice was correlated to its respective feature by being added to the final column in the features' respective row. When preparing the data matrix for angle and area training, radians and square millimeters were used, respectively. For classification, the geometry, or parameter, of interest was labeled with a binary classification variable with '1' signifying a true value, and '0' signifying a false one in the output column of the data matrix. For example, when training for recognizing an elliptical feature, all rows corresponding to an ellipse would have a '1' in their final column, while all others (i.e. rectangles and triangles) would be '0.' The data matrix structure is displayed visually below.

$$\begin{bmatrix}
 x_{1,1} & \cdots & x_{1,j-1} & x_{1,j} & \cdots & x_{1,m} & y_1 \\
 \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
 x_{i-1,1} & \cdots & x_{i-1,j-1} & x_{i-1,j} & \cdots & x_{i-1,m} & y_{i-1} \\
 x_{i,1} & \cdots & x_{i,j-1} & x_{i,j} & \cdots & x_{i,m} & y_i \\
 \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
 x_{n,1} & \cdots & x_{n,j-1} & x_{n,j} & \cdots & x_{n,m} & y_n
 \end{bmatrix}$$

$x_{i,j}$ is the input data value x belonging to the j^{th} input value of the i^{th} image. In this case, the data matrix has a size of n by $m + 1$, where n is the number of features represented in the data matrix, and m is the number of data values for a respective image. $y_{i,m+1}$ is the output of interest

corresponding to the i^{th} image, located in the $m + 1$ column position.

3.2 Selecting and Training the Machine Learning Algorithm

The base AI algorithm selected for evaluation of the synthesized data was a Backpropagation Artificial Neural Network (BP-ANN). The premise for this selection was based on the versatility of this algorithm and successful application to image classification problems (Kotsiantis, Zaharakis and Pintelas, Supervised machine learning: A review of classification techniques 2007) [11] [21] [10].

Several architectures were tried for the ANN. Regardless of that, each ANN consists of an input layer, an output layer and 0, 1 or several hidden layers. Layers are numbered from left to right, with the leftmost layer (input layer) being layer 1. Each layer consists of several nodes. Nodes are numbered from top to bottom, with the topmost node being node 1. Nodes of one layer are connected to the nodes of the before and after layers. Each connection is assigned a weight value, $w^{(l)}_{i,j}$, where l and $l + 1$ are the numbers of the left and right layers, respectively, and i and j indicate the node numbers in the left and right layers, respectively. Each node has an input and an output. The input of the input layer nodes are the 104 data values. Other than that, the input of node j of layer $l + 1$ is a weighted, biased sum of the outputs of nodes of layer l . That is, $p^{(l+1)}_j = (\sum w^{(l)}_{i,j} o^{(l)}_i) + b^{l+1}_j$, where $1 \leq i \leq n^{(l)}$, $n^{(l)}$ is the number of nodes of the l^{th} layer, $p^{(l+1)}_j$ is the input of the j^{th} node of the $(l + 1)^{\text{th}}$ layer, $o^{(l)}_i$ is the output of i^{th} node of the l^{th} layer and b^{l+1}_j is the bias value of the j^{th} node of the $(l + 1)^{\text{th}}$ layer. The output of each node is obtained from passing the node's input to an activation function, Φ , such that $o^{(l)}_i = \Phi(p^{(l)}_i)$. An example is the sigmoid function, $\sigma(x) = 1/(1 + e^{-x})$, which takes a value between 0 and 1, for $-\text{inf} < x < +\text{inf}$. The data values for each extracted image are fed as input to the ANN. Following that, through a feedforward process, a set of final outputs are calculated. The values of these outputs depend on the architecture of the network, the input features and the weight and bias values. The idea is that, throughout a training process, the values of weights and biases can be adjusted such that, the output values can accurately reflect the class as well as parameters of the embedded features. The network training is in fact an optimization process, in which a cost function is minimized through a gradient descent process. This cost function is the cumulative error associated with individual training data. An individual error, in turn, reflects the difference between the value predicted by the network, and the truth.

An adaptive momentum algorithm was attractive for integration into the standard backpropagation algorithm as it is known to increase convergence time through attempted prevention of overbearing momentum influence

in circumstances with smaller gradients (Fig. 11). (Moreira and Fiesler, Neural networks with adaptive learning rate and momentum terms 1995) [31] [28].

An opensource software called Multiple Back-Propagation (Lopes and Ribeiro, GPU Implementation of the Multiple Back-Propagation Algorithm 2009) (Lopes and Ribeiro, An efficient gradient-based learning algorithm applied to neural networks with selective actuation neurons 2003) (Lopes and Ribeiro, Hybrid learning in a multi-neural network architecture 2001) was selected to undertake training and testing procedures for the produced synthesized data, because of its capability of implementing the previously selected algorithm combination. The specific architectures of the backpropagation algorithms designed for the three training scenarios are displayed in Fig. 12.

When referring to the architecture of the neural network (i.e., second column of table of Fig. 12), the following structure has been adopted: “number of inputs”- “number of nodes in hidden layer 1”-...- “number of nodes in hidden layer n”- “number of outputs”. The specific architectures and activation functions used were selected primarily by experimentation.

4 Results and Discussion

A total of 2,430 features were synthesized for training AI-based feature extraction algorithms in the context of X-ray CT images. Of the 2,430 features, there existed three geometric categories: equilateral triangle, rectangle, and ellipse, with 810 features belonging to each category. Each geometry's orientation was varied through 90 degrees in ten steps. Both length and width dimensions describing the rectangles and ellipses were altered from 100 μm to 550 μm in nine steps, while the diameter of the circle, of which each equilateral triangle could be inscribed, was varied from 100 μm to 550 μm in 81 steps to create an identical number of total

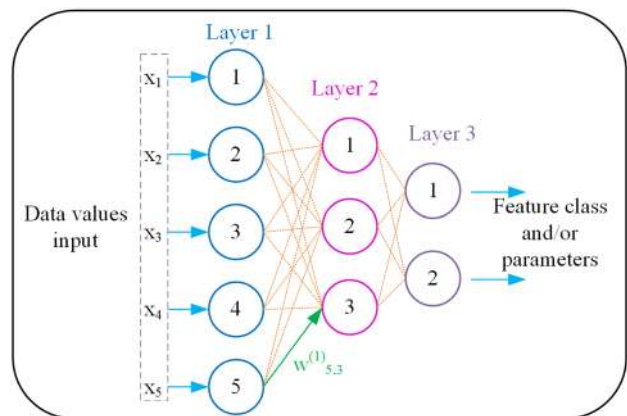


Fig. 11 Generic artificial neural network architecture

Fig. 12 Architectures and activation functions corresponding to specific training scenarios

Training Scenario	Architecture	Activation Function		
		Hidden Layer 1	Hidden Layer 2	Hidden Layer 3
Area	104-5-2-1	Hyperbolic Tangent	Sigmoid	N/A
Angle	104-5-3-2-1	Sigmoid	Sigmoid	Sigmoid
Classification	104-5-2-1	Sigmoid	Sigmoid	N/A

permutations to the other geometries. The features were lased into polished silicon wafers as detailed in Sect. 3.2 and imaged using a Zeiss Xradia 520 Versa using parameters outlined in Sect. 3.3. After imaging, MATLAB was used for image postprocessing, namely applying matrix transformations to ensure proper image view and agreement with the lasing recipe for accurate algorithm training. The image data was then packaged and refined using methods described in Sect. 3.4 prior to algorithm input. Backpropagation neural network algorithms with architectures defined in Sect. 3.5 were then trained and tested by utilizing the Multiple Back-propagation software. Finally, information on network success regarding classification and parameterization of feature orientation and area were obtained.

Three neural networks were trained for shape classifications, classifying rectangle shapes against non-rectangles, ellipse shapes against non-ellipses and triangle shapes against non-triangles, respectively. Out of the entire feature images, 2,049 were used for training and 381 were used for testing the network.

The result of classification on the test data can be seen in Figs. 13a-f. In this representation, data points are indicated by circles. Blue circles indicate an exclusion from a shape class and red circle indicates an inclusion in a shape class. For example, blue circle indicates a non-rectangle shape and red circle indicates a rectangle shape in Figs. 13a and b. The performance of the method for shape classification will be shown in the following. For some cases, it was realized that the classification accuracy of the network was improved when the very smallest features were removed from testing set. The defining parameter used to quantify the smallest features of a dataset was area; that is, for a case where the 100 smallest features were removed, the 100 features with the smallest area were removed.

Per network predications, the circles are classified and placed into two regions of blue (exclusion from a shape class) and red (inclusion in a shape class). The accuracy of the method is indicated by how well the blue circles are placed in the blue region and similarly how well the red circles are placed in red region. A blue circle in a red region will indicate a false positive and a red circle in a blue region will indicate a false negative.

For rectangles, 15 false positive readings and 25 false negatives were resulted, resulting in an overall classification accuracy of 89.5% (Fig. 13a). Figure 13b shows the classification of rectangles vs. non-rectangles when the

100 smallest features were removed from the dataset. Upon removal of the 100 smallest features, the testing set resulted in 22 false positive readings and 9 false negative, resulting in an overall classification accuracy of 86.58%.

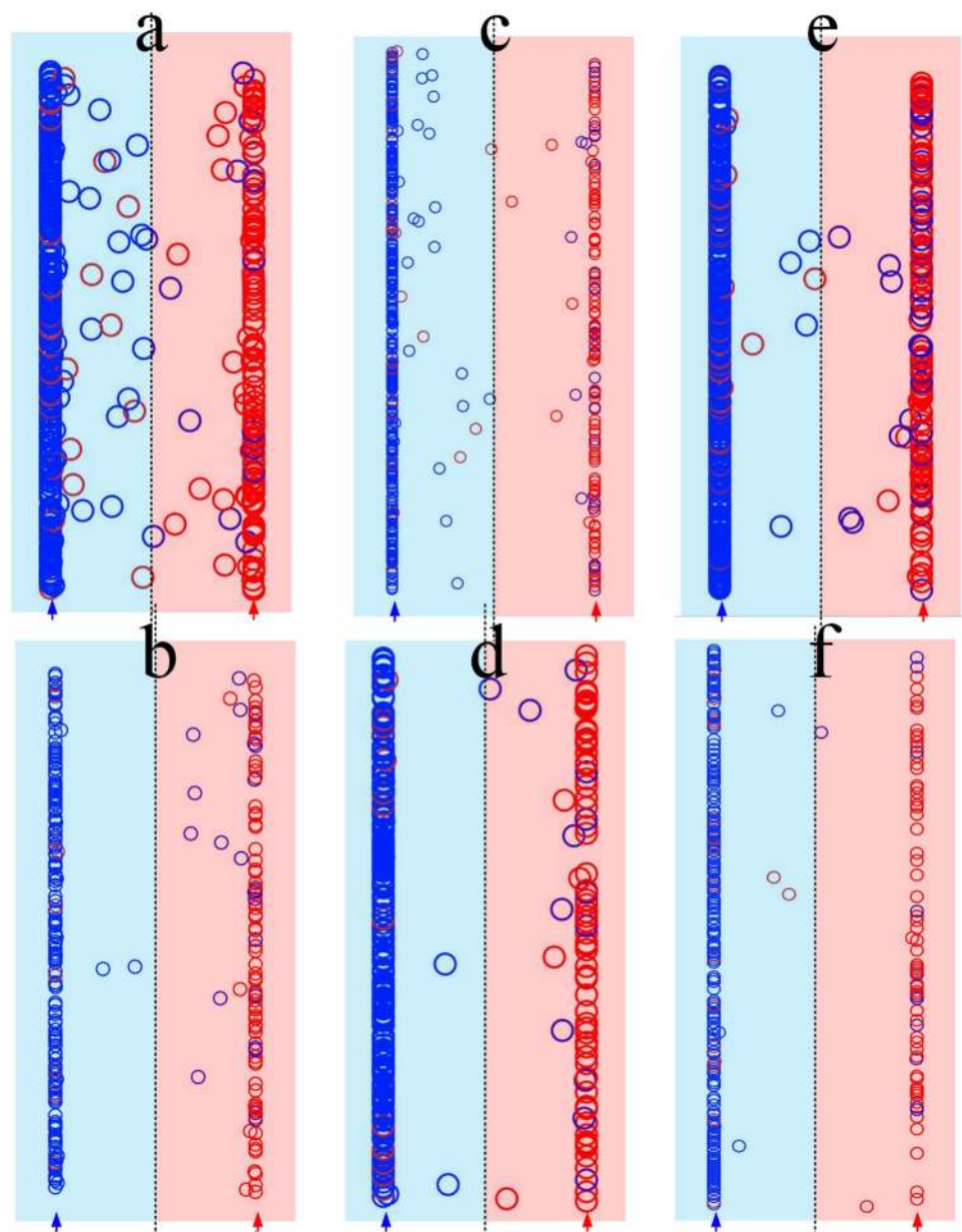
For ellipses, 27 false positive readings and 24 false negatives were resulted, resulting in an overall classification accuracy of 86.6% (Fig. 13c). Figure 13d shows the classification of ellipses vs. non-ellipses when the 100 smallest features were removed from the dataset. Upon removal of the 100 smallest features, the testing set resulted in 13 false positive readings and 12 false negative, resulting in an overall classification accuracy of 89.18%.

For triangles, 37 false positive readings and 28 false negatives were resulted, resulting in an overall classification accuracy of 82.9% (Fig. 13e). Figure 13f shows the classification of triangles vs. non-triangles when the 100 smallest features were removed from the dataset. Upon removal of the 100 smallest features, the testing set resulted in 12 false positive readings and 17 false negative, resulting in an overall classification accuracy of 87.45%.

A neural network was trained to predict the angle at which a feature is oriented relative to the horizontal axis. Out of the entire feature images, 2,126 were used for training and 304 were used for testing the network. The result of such prediction on the test data can be seen in Fig. 14a-i. For each shape, namely, rectangles, ellipses and triangles, this has been conducted three times: starting by including all of the test data, followed by removing the 100 smallest and 200 smallest features from the data to study the effect of image quality on the performance of the AI prediction method. Defined in the same way as for classification, the distinguishing parameter used to quantify the smallest features of a dataset was area. Figures 14a-c belong to rectangles, Figs. 14d-f belong to ellipses and Figs. 14g-i belong to triangles. In these representations, the orientation of dotted blue lines indicates the truth and the orientation of solid red lines indicate the network prediction. Success of the parameterization-based network trainings were classified by the root mean square error (RMSE) of the testing set. Where RSME is defined for this case as follows:

$$RMSE = \sqrt{\frac{\sum_i^N (y_{t,i} - y_{n,i})^2}{N}}$$

Fig. 13 Classification of shapes with ANNs: **a** classification of rectangles vs. non-rectangles; **b** classification of rectangles vs. non-rectangles, excluding the 100 smallest features; **c** classification of ellipses vs. non-ellipses; **d** classification of ellipses vs. non-ellipses, excluding the 100 smallest features; **e** classification of triangles vs. non-triangles; **f** classification of triangles vs. non-triangles, excluding the 100 smallest features



where $y_{i,i}$ is the true output of the i th test feature, $y_{n,i}$ is the network output of the i th feature, and N is the total number of test features for a given training.

The RSME values corresponding to Figs. 14a-c were 0.3462, 0.2842, and 0.1984, respectively. Corresponding to Figs. 14d-f, the RSME values were 0.2610, 0.1422, and 0.1661, respectively. Finally, the RSME values corresponding to Figs. 14g-i were 0.3199, 0.2243, and 0.1751, respectively.

A neural network was trained to predict the feature areas. Out of the entire feature images, 87.5% were used for training and 12.5% were used for testing the network. The result of such prediction on the test data can be seen in Figs. 15a-c for rectangles, ellipses and triangles, respectively. In these charts,

the y-axis reflects the area of different features that are distributed along the x-axis. The solid line indicates truth and the dotted line indicated the network prediction. As previously stated, the root mean square error of the testing set was chosen as the measure of network success for parameterization trainings. In predicting feature areas, the rectangle testing set (Fig. 15a) resulted in an RMSE of 0.0154, whereas the ellipse and triangle testing sets (Figs. 15b and c) resulted in RMSEs of 0.0066 and 0.0041, respectively.

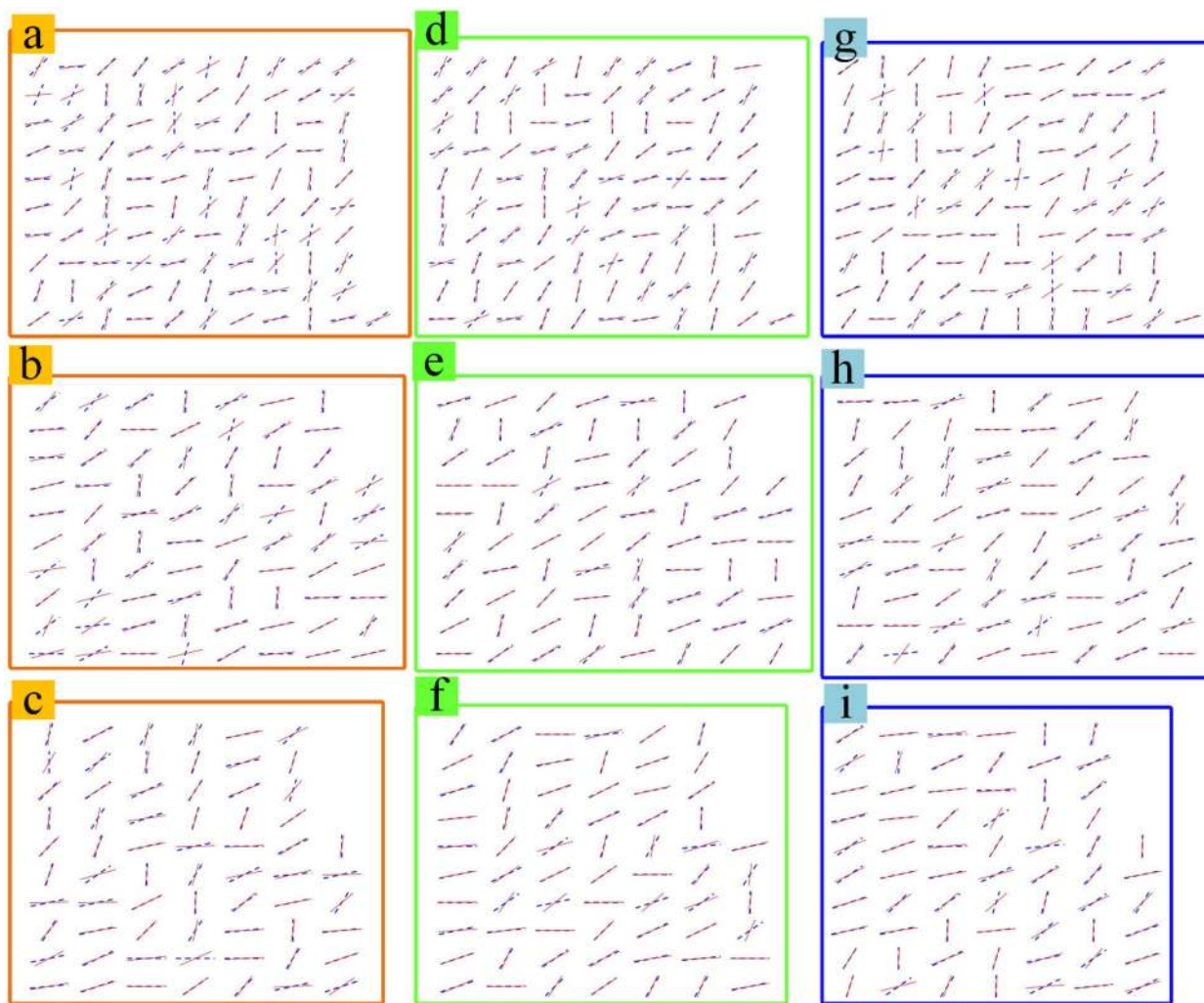


Fig. 14 Angles of features predicted by trained ANNs. Dotted blue signifies truth (actual angle) and solid red signifies the angle predicted by the network: **a** Rectangles; **b** Rectangles, 100 smallest features removed; **c** Rectangles, 200 smallest features removed; **d** Ellip-

ses; **e** Ellipses, 100 smallest features removed; **f** Ellipses, 200 smallest features removed; **g** Triangles; **h** Triangles, 100 smallest features removed; **i** Triangles, 200 smallest features removed

5 Conclusion

X-ray CT has endless potentials in nondestructively providing information about the interior miniature features of objects. Nevertheless, lack of automated yet robust, fast and accurate methods for interpreting X-ray CT images continues to discount the capabilities of this technique. Attempts to address this issue using AI methods face a great challenge, namely lack of labeled ground truth data. In this paper, we proposed a novel technique for synthesizing labeled data and using them for training machine learning algorithms,

towards an automated classification and parametrization tool. We showed that, using a femtosecond laser setup, miniature features could be implemented onto layers of a layered object to mimic the internal features of a generic object. Further, the X-ray images of the embedded features could be correlated with the parameters used for designing those features to obtain labeled training data. Finally, we showed that such training data could be used for training ANNs for undertaking tasks such as classification and parametrization.

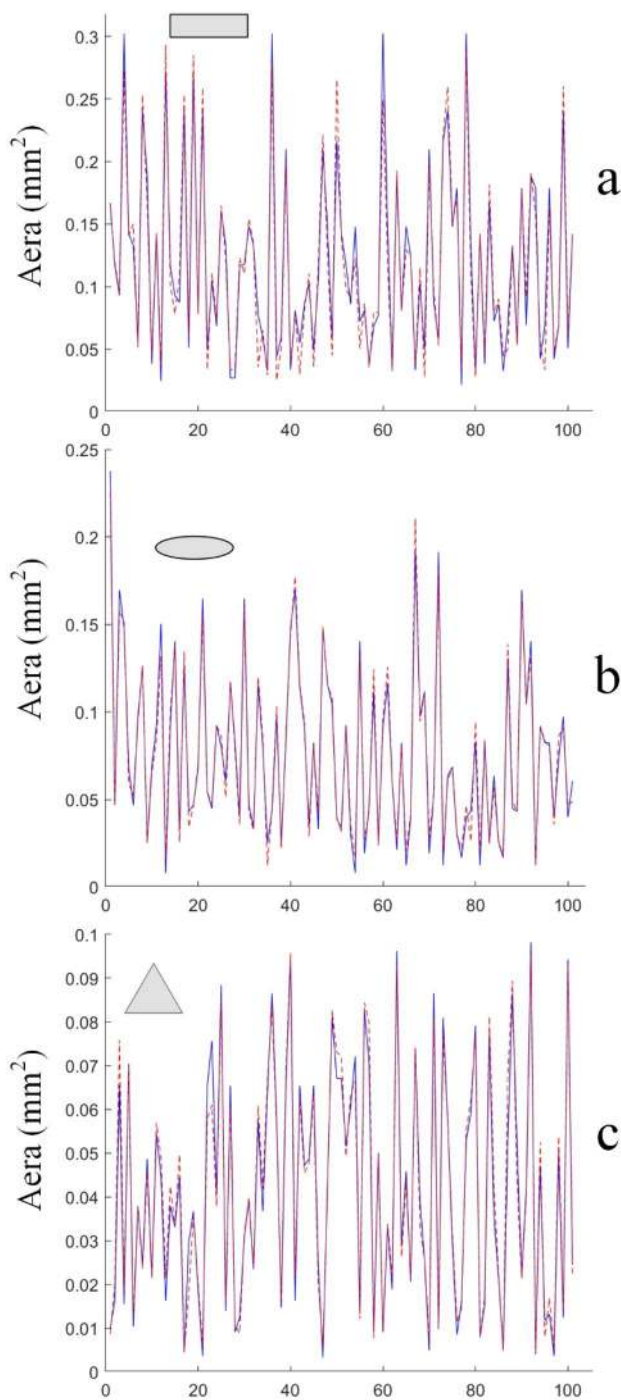


Fig. 15 Areas of features predicted by ANNs. Solid blue indicates truth and dotted red indicates network prediction: **a** rectangles; **b** ellipses; **c** triangles

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated

otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Arganda-Carreras, I., Turaga, S.C., Berger, D.R., Cireşan, D., Giusti, A., Gambardella, L.M., Schmidhuber, J., et al.: Crowdsourcing the creation of image segmentation algorithms for connectomics. *Front. Neuroanat.* **9**, 142 (2015)
- Blaschke, T., Lang, S.: Object based image analysis for automated information extraction: a synthesis. In: *Measuring the Earth II ASPRS Fall Conference*, pp. 6–10 (2006)
- Burger, W., Burge, M.J.: *Principles of Digital Image Processing: Advanced Methods*. Springer, Berlin (2013)
- Chen, S., Zhang, D.: Robust image segmentation using FCM with spatial constraints based on new kernel-induced distance measure. *IEEE Trans. Syst. Man Cybern. Part B* **34**, 1907–1916 (2004)
- Dambreville, S., Rathi, Y., Tannen, A.: Shape-based approach to robust image segmentation using kernel PCA. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pp. 977–984 (2006)
- Gao, H., Siu, W.-C., Hou, C.-H.: Improved techniques for automatic image segmentation. *IEEE Trans. Circuits Syst. Video Technol.* **11**, 1273–1280 (2001)
- Gonzalez, R.C., Woods, R.E., Eddins, S.L.: *Digital image processing using MATLAB*. Pearson Education India, New Delhi (2004)
- Ingman, J.M., Jormanainen, J.P.A., Vulli, A.M., Ingman, J.D., Kari Maula, T.J.: Localization of dielectric breakdown defects in multilayer ceramic capacitors using 3D X-ray imaging. *J. Eur. Ceram. Soc.* **39**, 1178–1185 (2019)
- Kaur, D., Kaur, Y.: Various image segmentation techniques: a review. *Int. J. Comp. Sci. Mob. Comput.* **3**, 809–814 (2014)
- Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014). arXiv preprint <https://arxiv.org/1412.6980>
- Kishore, R., Kaur, T.: Backpropagation algorithm: an artificial neural network approach for pattern recognition. *Int. J. Sci. Eng. Res.* **3**, 6–9 (2012)
- Klein, A.: A game for crowdsourcing the segmentation of Big-Brain data. *Res. Ideas Outcomes* **2**, e8816 (2016)
- Kotsiantis, S.B., Zaharakis, I., Pintelas, P.: Supervised machine learning: a review of classification techniques. *Emerg. Artif. Intell. Appl. Comput. Eng.* **160**, 3–24 (2007)
- Lopes, N., Ribeiro, B.: An efficient gradient-based learning algorithm applied to neural networks with selective activation neurons. *Neural Parallel Sci. Comput.* **11**, 253–272 (2003)
- Lopes, N., Ribeiro, B.: GPU implementation of the multiple backpropagation algorithm. In: Corchado, E., Yin, H. (eds.) *Intelligent Data Engineering and Automated Learning (IDEAL 2009)*, pp. 449–456. Springer, Berlin (2009)
- Lopes, N., Ribeiro, B.: Hybrid learning in a multi-neural network architecture. *{INNS-IEEE} International Joint Conference on Neural Networks ({IJCNN 2001})*, pp. 2788–2793 (2001)
- Meer, P., Mintz, D., Rosenfeld, A., Kim, D.Y.: Robust regression methods for computer vision: a review. *Int. J. Comput. Vision* **6**, 59–70 (1991)
- Moreira, M., Fiesler, E.: *Neural networks with adaptive learning rate and momentum terms*. Tech. rep., Idiap (1995)
- Opitz, D., Blundell, S.: Object recognition and image segmentation: the Feature Analyst® approach. In: *Object-Based Image Analysis*, pp. 153–167. Springer, Berlin (2008)

20. Pal, N.R., Pal, S.K.: A review on image segmentation techniques. *Pattern Recogn.* **26**, 1277–1294 (1993)
21. Poulton, M.M.: Neural networks as an intelligence amplification tool: a review of applications. *Geophysics* **67**, 979–993 (2002)
22. Rawat, W., Wang, Z.: Deep convolutional neural networks for image classification: a comprehensive review. *Neural Comput.* **29**, 2352–2449 (2017)
23. Singh, A., Thakur, N., Sharma, A.: A review of supervised machine learning algorithms. In: 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), pp. 1310–1315 (2016)
24. Wang, B., Gao, X., Tao, D., Li, X.: A unified tensor level set for image segmentation. *IEEE Trans. Syst. Man Cybern. Part B* **40**, 857–867 (2009)
25. Wang, G., Yu, H., De Man, B.: An outlook on X-ray CT research and development. *Med. Phys.* **35**, 1051–1064 (2008)
26. Wang, X., Mudie, L., Brady, C.J.: Crowdsourcing: an overview and applications to ophthalmology. *Curr. Opin. Ophthalmol.* **27**, 256 (2016)
27. Wei, Y., Yu, H., Hsieh, J., Cong, W., Wang, G.: Scheme of computed tomography. *J. X-ray Sci. Technol.* **15**, 235–270 (2007)
28. Yu, C.-C., Liu, B.-D.: A backpropagation algorithm with adaptive learning rate and momentum coefficient. In: Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290), pp. 1218–1223 (2002)
29. Yuheng, S., Hao, Y.: Image segmentation algorithms overview (2017). arXiv preprint <https://arxiv.org/1707.02051>
30. Zaitoun, N.M., Aqel, M.J.: Survey on image segmentation techniques. *Proc. Comput. Sci.* **65**, 797–806 (2015)
31. Zeiler, M.D.: Adadelata: an adaptive learning rate method (2012). arXiv preprint <https://arxiv.org/1212.5701>
32. Zhang, Y.J.: A review of recent evaluation methods for image segmentation. In: Proceedings of the Sixth International Symposium on Signal Processing and its Applications (Cat. No. 01EX467), pp. 148–151 (2001)