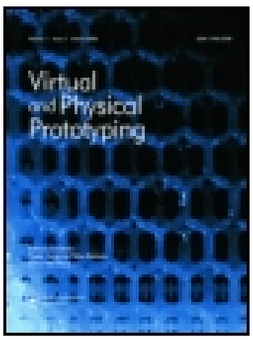

Virtual and Physical Prototyping

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/nvpp20

Rapid surface defect identification for additive manufacturing with in-situ point cloud processing and machine learning

Lequn Chen , Xiling Yao , Peng Xu , Seung Ki Moon & Guijun Bi

To cite this article: Lequn Chen , Xiling Yao , Peng Xu , Seung Ki Moon & Guijun Bi (2020): Rapid surface defect identification for additive manufacturing with in-situ point cloud processing and machine learning, Virtual and Physical Prototyping, DOI: 10.1080/17452759.2020.1832695

To link to this article: https://doi.org/10.1080/17452759.2020.1832695

Published online: 15 Oct 2020.

VIRTUAL AND PHYSICAL PROTOTYPING
https://doi.org/10.1080/17452759.2020.1832695

Taylor & Francis
Taylor & Francis Group

Rapid surface defect identification for additive manufacturing with in-situ point cloud processing and machine learning

Lequn Chen[a,b], Xiling Yao [a], Peng Xu[a], Seung Ki Moon[b] and Guijun Bi[a]

[a]Singapore Institute of Manufacturing Technology, Agency for Science, Technology and Research, Singapore; [b]School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore

ABSTRACT

Surface monitoring is an essential part of quality assurance for additive manufacturing (AM). Surface defects need to be identified early in the AM process to avoid further deterioration of the part quality. In this paper, a rapid surface defect identification method for directed energy deposition (DED) is proposed. The main contribution of this work is the development of an in-situ point cloud processing with machine learning methods that enable automatic surface monitoring without sensor intermittence. An in-house software platform with a multi-nodal architecture is developed. In-situ point cloud processing steps, including filtering, segmentation, surface-to-point distance calculation, point clustering, and machine learning feature extraction, are performed by multiple subprocesses running simultaneously. The combined unsupervised and supervised machine learning techniques are applied to detect and classify surface defects. The proposed method is experimentally validated, and a surface defect identification accuracy of 93.15% is achieved.

ARTICLE HISTORY
Received 15 July 2020
Accepted 1 October 2020

KEYWORDS
Additive manufacturing; directed energy deposition; surface monitoring; defect identification; machine learning

1. Introduction

Additive manufacturing (AM), also known as 3D printing, is a trending technology with great potential in high-mix low-volume manufacturing (Ng, Chua, and Shen 2019; Tan et al. 2019; Choong, Maleksaeedi, et al. 2020; Jiang and Ma 2020; Tan et al. 2020). The versatility of the AM technology makes it an ideal tool to address complex design needs for customised products, especially in crises when conventional processes and supply chains cannot respond quickly enough to meet the rapid-changing demand (Bishop and Leigh 2020; Celik et al. 2020; Choong, Tan, et al. 2020; He et al. 2020). However, unlike the more mature processes such as machining, casting, and forging, AM still lacks robustness and repeatability despite its many advantages. This is particularly true for large-format metal AM processes, such as the directed energy deposition (DED) and powder bed fusion (PBF) that use laser or electron beam to melt metal powders during material deposition (Tan et al. 2015; Jiang et al. 2019; Yu, Sing, Chua, Kuo, et al. 2019; Yu, Sing, Chua, and Tian 2019; Kuo et al. 2020). The rapid heating and cooling cycles generate significant thermal stress in the part that may eventually lead to distortions. Other uncertainties and instabilities in metal AM processes, such as the strong melt-pool dynamics, localised heat accumulation, and speed inconsistency in machine motions, also create defects in AM parts (Collins et al. 2016; Grasso and Colosimo 2017). Surface defects, including local bulges and dents, are important reflections of AM process abnormalities. Therefore, early detection of surface defects is crucial for AM processes to avoid excessive errors leading to unrepairable build failures. The development of effective surface monitoring methods is an important step towards a higher level of industry readiness for AM technologies (Chua, Ahn, and Moon 2017).

In the literature of in-situ surface quality assessment for AM, various non-destructive testing (NDT) methods have been studied (Lu and Wong 2018). One of the most common NDT methods is the vision-based surface topography measurement that captures images using digital cameras and assesses the surface quality by image processing. The popularity of the vision-based method is attributed to the simple setup and inexpensive hardware. Computer vision algorithms for feature extraction have been proven capable of detecting distortions and defects on additively manufactured parts (Okarma and Fastowicz 2020). Stereovision-based methods have also been developed for AM, where two cameras are positioned at different locations with overlapping views of the monitored surface (Li et al. 2018). The dual-camera setup enables the 3D topography reconstruction from

CONTACT Xiling Yao yao_xiling@simtech.a-star.edu.sg, Guijun Bi gjbi@simtech.a-star.edu.sg Singapore Institute of Manufacturing Technology, Agency for Science, Technology and Research, 73 Nanyang Drive 637662, Singapore

the pixels using the digital image correlation (DIC) technique (Holzmond and Li 2017). Another vision-based NDT method is profile scanning by line lasers. Instead of capturing the pixel images of the surface area, the laser profiler captures only the reflected laser line from the surface through a narrow-bandpass optical filter. The distance from the targeted surface to the sensor is then calculated by the triangulation method (García-Díaz et al. 2018). Compared to traditional vision-based methods, laser profile scanning has a higher resolution and can generate accurate height data without the computationally heavy 3D reconstruction step (Tang, Wang, and Zhang 2019).

Different statistical methods have been applied in sensor data processing for surface quality assessment. While the underlying physics of defect formation in AM is complicated, statistical methods can be powerful and yet simple tools for predicting defect occurrence and modelling the characteristics of defects. For example, an adaptive Bayesian methodology has been developed to quantitatively define the geometric shape deviations of additively manufactured parts using only a limited number of sample data (Sabbaghi, Huang, and Dasgupta 2018). Various machine learning models (e.g. random forest, decision tree, and artificial neural network) have been tested for quality control in extrusion-based AM processes such as fused deposition modelling (FDM) (Sohnius et al. 2019; Jiang et al. 2020) and 3D concrete printing (Lao et al. 2020). In the above studies, predictive models have been built to correlate the surface defects with extrusion process parameters such as feed rate, layer height, and infill density, etc. Another research on the FDM applied the random forest regression to predict the printed part's surface roughness based on real-time process conditions (including machine temperatures and vibrations) measured by thermocouples, infra-red cameras, and accelerometers (Wu, Wei, and Terpenny 2018). Machine learning techniques have also been applied in quality control for 3D bioprinting of tissues and organs, where shape deviations were related to the bioprinting process parameters (Yu and Jiang 2020) and the machine learning models were used for process refinement (Ng et al. 2020).

The aforementioned studies on surface quality assessment for AM have common limitations. Raw data output from sensors requires post-processing before they can be further analysed for defect identification. Common post-processing methods include 3D reconstruction, data filtering, and depth image generation, etc. Geometric feature extractions (e.g. line and contour extractions) also need to be carried out using the post-processed sensor data, and the results are then fed into statistical models for defect identification. These data post-processing and feature extraction steps may consume a considerable amount of computing time, and stacking them together may cause significant latency. Therefore, as reported in most literature, surface defect identification is performed only after the sensor has completed the current capturing cycle. And the sensor can start the next capturing cycle only when surface defect identification is completed. This intermittent procedure makes the surface monitoring inefficient, especially when the additively manufactured part is large, which is common in the DED processes.

Therefore, to address the above issue, this work aims to enable on-the-fly surface defect identification during continuous sensor capturing without intermittence. In this research, a rapid surface defect identification method for the robot-based DED system is proposed, featuring a novel in-situ point cloud processing method integrated with machine learning models. A laser profiler is used for surface monitoring. A software platform is developed in-house to establish data transmission among the sensor, robot, and central controller. The multi-nodal architecture of the software platform enables parallel and synchronised computing processes that can perform multiple tasks simultaneously, which makes the point cloud processing and machine learning algorithms to run in-situ during surface scanning. The rapidness of the proposed defect identification method is attributed to the following two novel characters. (1) 3D spatial data in the form of point clouds are processed automatically during, instead of after, sensor capturing. (2) Depth images do not need to be generated, and the computationally expensive geometric feature extraction is not required. Simple point cloud statistics are directly used as input data to the machine learning model that identifies surface defects. Hyperparameters of various machine learning models are optimised via cross-validated grid search, and the model with the best testing accuracy is chosen. The proposed method is validated experimentally and proven to be efficient and accurate.

The rest of this paper is structured as follows: Section 2 explains the details of the proposed method, including the system setup, software architecture, in-situ point cloud processing, and machine learning for surface defect identification. Section 3 presents the experimental results that validate the proposed method and finally, Section 4 concludes the paper with a summary and future work recommendation.

2. Methodology

2.1. System setup

This research is conducted based on the Laser-Aided Additive Manufacturing (LAAM) system developed in-

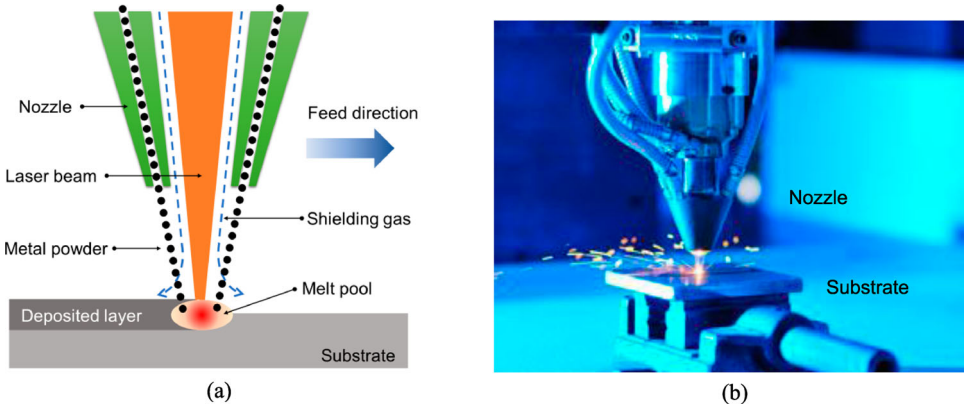

Figure 1. (a) The illustration of the LAAM process. (b) The nozzle used in the LAAM process.

house at the authors' institution. As a member of the DED family, the LAAM process utilises a high-power laser to melt metallic powders as they are deposited. The illustration of the LAAM process is given in Figure 1. A coaxial nozzle is used to aim the laser beam at the substrate and project the powders through the channels. The molten powders enter the melt pool and then solidify rapidly as the nozzle travels in the feed direction. Due to its high build rate, LAAM is particularly suitable for manufacturing large-format components such as those used in aircraft, ships, and offshore structures.

The overall hardware setup of the LAAM system is illustrated in Figure 2. The material flow, energy flow, and signal flow within the LAAM system are indicated by arrows. A Highyag BIMO laser head is installed on an ABB IRB-4400 6-axis robot, next to which an ABB IRBP-A 2-axis positioner is located. Both the robot and the positioner are controlled by an IRC5 controller. An IPG YLS-6000 Ytterbium laser source supplies the 1070 nm continuous laser beam to the laser head via a fibre. The on/off switch and power setting of the laser are also controlled by the IRC5 controller via digital and analogue I/O signals, respectively. A GTV powder feeder delivers metallic powders to the nozzle attached at the end of the laser head. To perform surface monitoring for additively manufactured parts, a laser profiler is

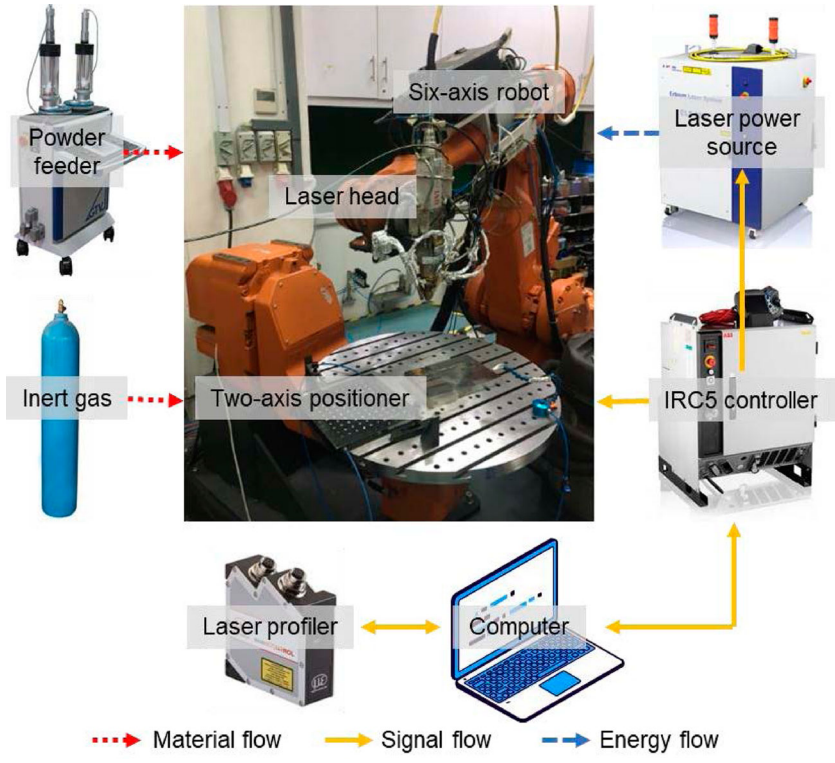

Figure 2. The hardware setup of the LAAM system and the flows of energy, signals, and materials.

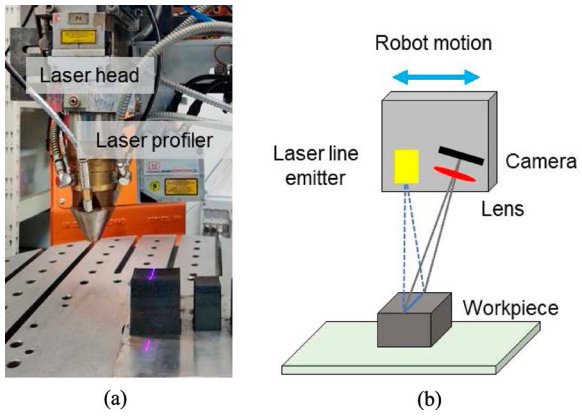

Figure 3. (a) The laser profiler mounted on the robot. (b) The working principle of the laser profiler.

chosen due to its compactness, high accuracy, and high capturing rate. A Micro-Epsilon ScanCONTROL 2950-50/BL laser profiler is integrated into the LAAM system. The sensor is rigidly mounted on the 6th axis of the robot, next to the laser head. This configuration allows the sensor to move along the AM tool path, thus enabling accurate localisation of surface defects in the workpiece's coordinate frame. A detailed view of the sensor-robot coupling is shown in Figure 3(a). The sensor is connected to a computer that communicates with the IRC5 controller, so that the operations of both the sensor and robot can be synchronised. The distance (in Z-direction) between the targeted surface and the sensor is calculated by the 'laser line triangulation' principle (García-Díaz et al. 2018) as illustrated in Figure 3(b). The output of the sensor is a series of (x, z) data representing the 2D surface profile in the sensor's local coordinate frame.

After the sensor is installed, calibration is required to obtain the spatial relationship between the robot tool-centre-point (TCP) and the sensor's local coordinate frame. The standard hand-eye calibration method (Tsai and Lenz 1989) was applied. The detailed implementation of the hand-eye calibration can be found in (Rodríguez-Araújo and Rodríguez-Andina 2015). Once the transformation from the TCP to the sensor frame ($\boldsymbol{T}_S^T$) is obtained, the sensor's pose relative to the workpiece ($\boldsymbol{T}_S^W$) can then be calculated using the following relationship:

$$\boldsymbol{T}_S^W = \boldsymbol{T}_T^W \boldsymbol{T}_S^T \tag{1}$$

where $\boldsymbol{T}_T^W$ is the TCP's pose relative to the workpiece that can be read from the robot controller. The coordinate frames involved in Equation (1) are illustrated in Figure 4. Using the above relationship, 2D profile data captured by the sensor can be transformed into 3D point clouds, which will be explained in detail in Section 2.2.

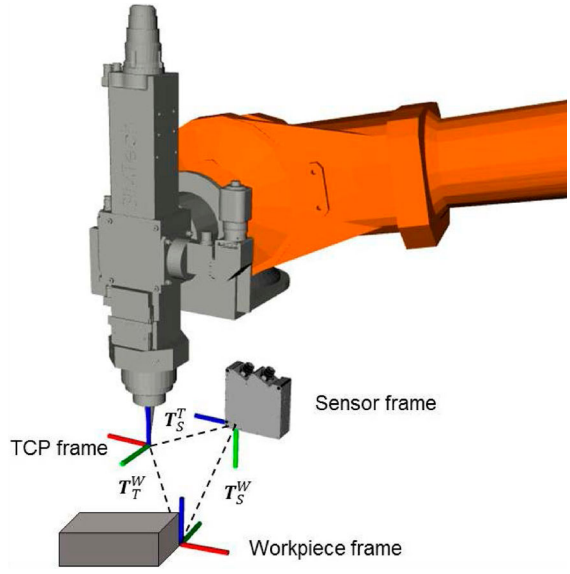

Figure 4. Coordinate transformations among the TCP, sensor, and workpiece frames.

2.2. Multi-nodal software architecture

A software platform dedicated to surface monitoring for LAAM is developed in this research. The proposed multi-nodal architecture of the software platform is introduced in this section.

A computer running a Linux OS (Ubuntu 18.04LS) serves as the central controller for the surface monitoring system, in which the in-house developed software is installed. The ROS library (Quigley et al. 2009) is employed to establish the connection among the sensor, robot, and central controller. An earlier implementation of ROS in AM systems can be found in (García-Díaz et al. 2018). The proposed multi-nodal software architecture is illustrated in Figure 5. The software is started by executing a 'launch' file that automatically starts all the 'node' in the software. Each node is an independent programme that performs a specific task (e.g. reading data and controlling the hardware) or computation (e.g. coordinate transformation and point cloud processing) for surface monitoring in LAAM. These nodes communicate with each other by publishing and subscribing to data on the 'topics' channels. A total of five nodes are launched simultaneously, and the details of each node are described as follows.

The *robot driver node* works as a TCP/IP client that talks to the server running inside the robot controller. Two TCP/IP sockets, i.e. the 'robot state' socket and 'robot command' socket, are created. During surface monitoring, the robot driver node continuously receives

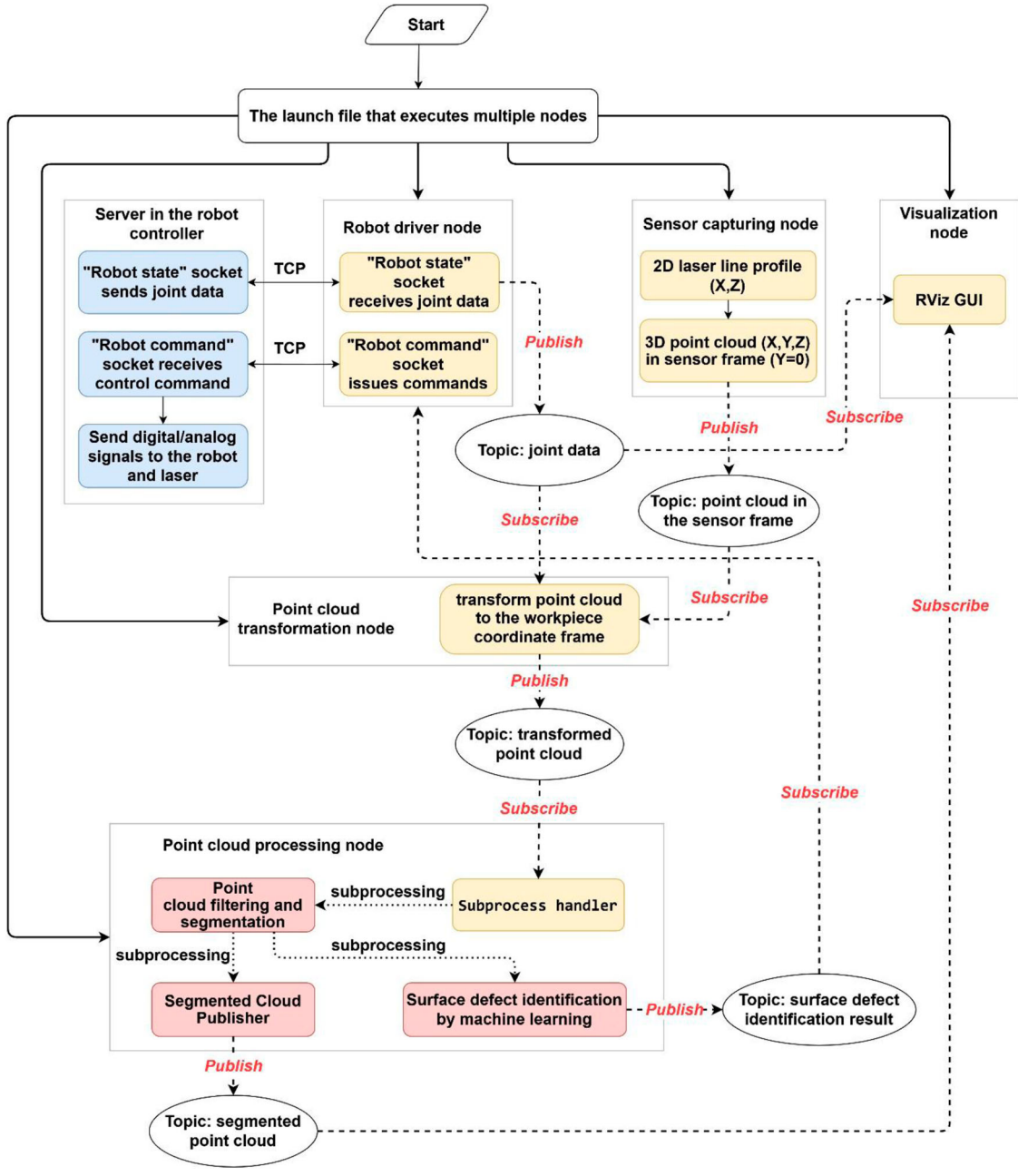

Figure 5. The proposed multi-nodal software architecture for surface monitoring in the LAAM.

the robot joint data from the robot controller through the 'robot state' socket. The joint data contain the TCP pose ($\boldsymbol{T}_T^W$) relative to the workpiece frame, as introduced in Section 2.1, and they are published to the 'joint data' topic. The robot driver node also sends controlling signals through the 'robot command' socket, such as those controlling the robot motion and laser emission.

The *sensor capturing node* operates the laser profiler, switches it on or off, sets its parameters, and receives the output data. The original sensor output as the result of laser line triangulation are (x, z) 2D coordinates. To obtain 3D point clouds in the sensor frame, a zero-valued y_0 coordinate is added to create a list of $\boldsymbol{P} = (x, y_0, z)$ points. These point are then published to the topic called 'point cloud in the sensor frame'.

The *point cloud transformation node* subscribes to both 'joint data' and 'point cloud in the sensor frame' topics that are published by the two nodes described above. The task of this node is to transform the point cloud from the sensor frame to the workpiece frame, using the following relationship:

$$\boldsymbol{T}_p^W = \boldsymbol{T}_S^W \boldsymbol{T}_P^S \tag{2}$$

where $\boldsymbol{T}_P^S$ is the 4×1 vector defined as $\boldsymbol{T}_P^S = [\boldsymbol{P}, 0]^T = [x, y_0, z, 0]^T$, and $\boldsymbol{T}_S^W$ is the 4×4 homogeneous

transformation matrix calculated by Equation (1). After this step, the transform point clouds $\boldsymbol{T}_p^W$, now with non-zero y coordinates and hence representing the 3D topography of the targeted surface, are continuously published to the 'transformed point cloud' topic.

The *point cloud processing node* subscribes to the 'transformed point cloud' topic. It performs in-situ point cloud manipulation and analysis while the sensor capturing process is ongoing, which leads to rapid surface defect identification without sensor or robot intermittence. This node spawns several 'subprocesses' (indicated by dotted arrows in Figure 5) to run parallel programmes for data archiving, point cloud processing, and machine learning. The surface defect identification result is published to the corresponding topic that is subscribed by the robot driver node. The robot driver node will then issue controlling commands (e.g. pausing the robot motion and switching off the laser emission) based on the surface identification result. Further details of the point cloud processing node and its associated subprocesses will be discussed in Sections 2.3 and 2.4.

The *visualisation node* is responsible for real-time graphical rendering of the robot status and surface monitoring result. This node launches the RViz programme that is the native graphical user interface (GUI) for ROS (Kam et al. 2015). The GUI subscribes to both 'joint data' and 'segmented point cloud' topics that are published by the robot driver node and point cloud processing node, respectively. A snapshot of the GUI during surface monitoring is shown in Figure 6, where the white region is the point cloud that represents the targeted surface. Disruptions or errors in the robot motion and sensor capturing can be timely reflected in the GUI, which is useful for human operators to observe the system conditions during the LAAM process.

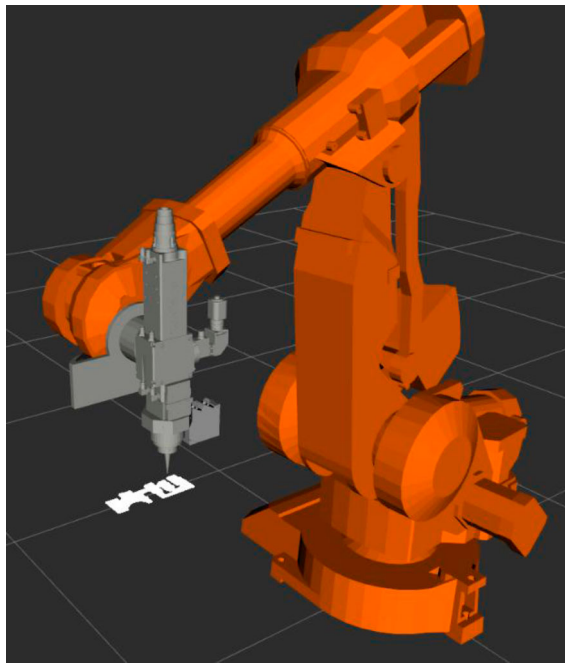

Figure 6. The visualisation node that shows the robot pose and the point cloud captured from the targeted surface.

2.3. In-situ point cloud processing

The proposed in-situ point cloud processing method makes it possible to perform on-the-fly surface defect identification during continuous monitoring without sensor intermittence. At the core of the proposed method is a subprocessing strategy that is used to invoke parallel programmes for in-situ point cloud processing. The subprocessing strategy is illustrated in Figure 7, which is a detailed expansion of the 'point cloud processing node' previously shown in Figure 6. Each subprocess and its associated data processing techniques are explained as follows.

The entry of the point cloud processing node is a subprocess handler with a timer function. The timeout value can be preset by the user (usually between 1 and 3 s). The timeout signal spawns *Subprocess 1* that is responsible for automatic point cloud filtering and segmentation. The timer is reset automatically after each timeout, until the entire surface monitoring programme is shutdown. While Subprocess 1 is running, all the other nodes, including the sensor capturing and point cloud transformation nodes, do not pause or wait. The subprocess handler also stores the data received from the 'transformed point cloud' topic to a temporary memory location called 'buffer'. By doing this, the subprocess handler can maintain a stable pipeline of point cloud while the branched subprocesses are running.

Subprocess 1 retrieves the point cloud from the buffer and feeds it to the filtering and segmentation functions. The raw data captured by the sensor may contain noises, including profiles of surrounding objects or the substrate, which can jeopardise the accuracy in the subsequent calculations for surface defect identification. Therefore, it is critical to remove the noises and isolate only the targeted surface before machine learning and surface defect identification can be performed. As indicated in Figure 7 above, the statistical outlier filter and voxel grid filter are applied consecutively, followed by a Random Sample Consensus (RANSAC) segmentation step. The statistical outlier filter is responsible for removing randomly located sparse points that belong to neither the targeted surface nor the substrate. The voxel grid filter down-samples the points in abnormally high-density regions, resulting in a point cloud with a more homogeneous density. The RANSAC segmentation algorithm removes the substrate from the point cloud and

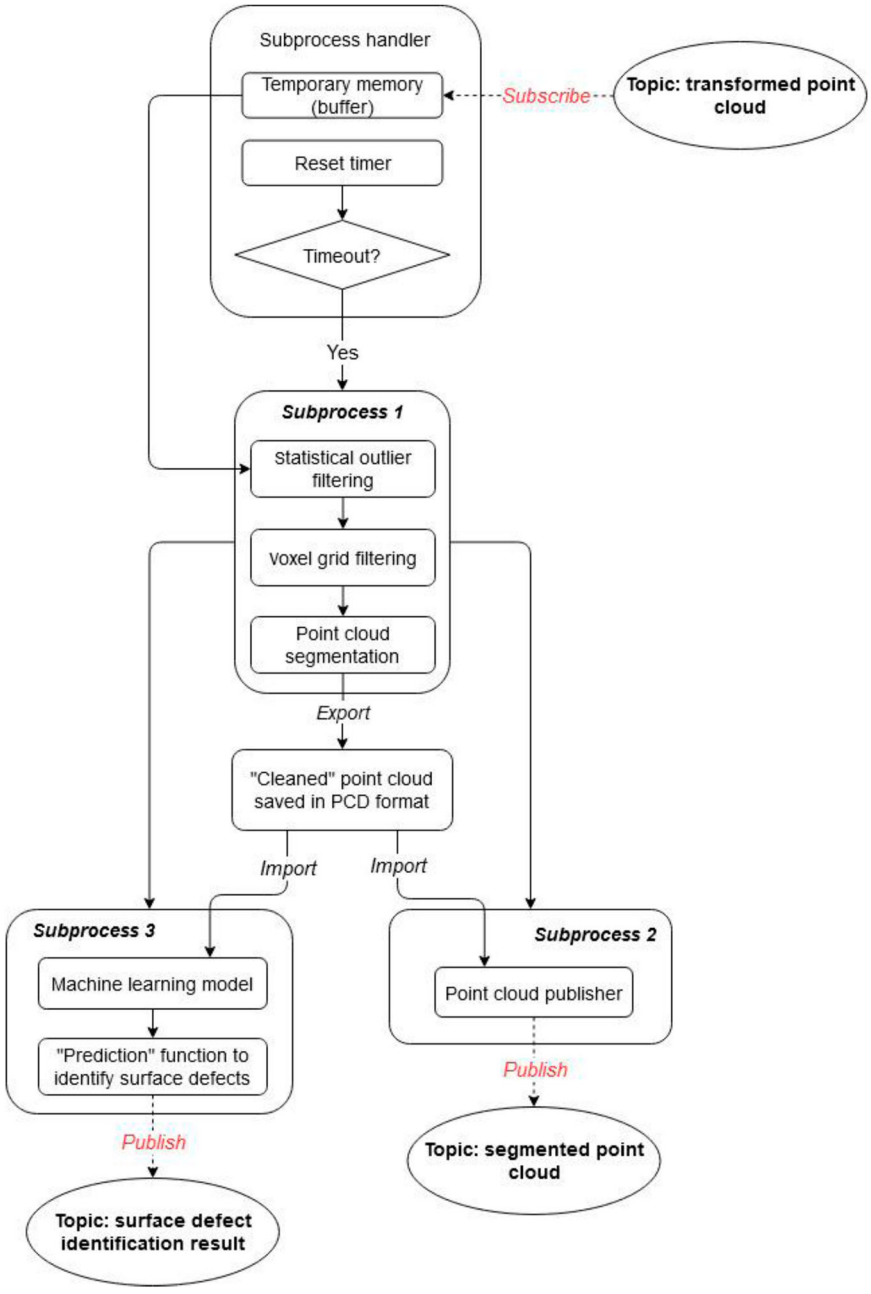

Figure 7. Subprocesses for in-situ point cloud processing.

finds the point cloud regions corresponding to the targeted surface only (Chen et al. 2020). The effect of filtering and segmentation is shown in Figure 8. The green points in Figure 8 belong to the targeted surface identified by the RANSAC method, the white points represent the substrates, and the red points are the noises to be removed by filters.

After the filtering and segmentation are completed, the segmented point cloud is exported to an ASCII file in the PCD format adapted from the Point Cloud Library (Rusu and Cousins 2011). The reason for saving the data into files instead of publishing them to a topic is to ensure that they can be accessed by any other subprocesses or nodes even after Subprocess 1 terminates and its associated memory space is destroyed. Subprocess 2 is spawned from Subprocess 1. It imports the PCD file just saved by Subprocess 1 and then continuously publishes the data to the 'segmented point cloud' topic. The purpose of Subprocess 2 is to communicate with the visualisation node and let the segmented point cloud be plotted in the GUI.

Subprocess 3 is also spawned from Subprocess 1 when the filtering and segmentation steps are completed. Immediately after both Subprocesses 2 and 3 are

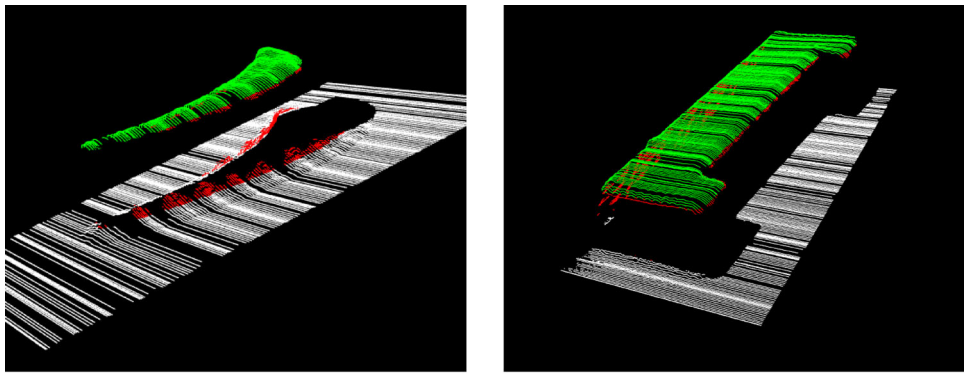

Figure 8. Effects of filtering and segmentation with different surface shapes. (a) The curved targeted surface. (b) The planar targeted surface are both highlighted in green.

launched, Subprocess 1 is terminated until it is restarted by the timer again in the subprocess handler. At the core of Subprocess 3 is a machine learning model that is used to identify surface defects. Subprocess 3 imports the PCD file and then extracts the numerical features from the point cloud data. By performing a prediction function using the numerical features as input arguments, the machine learning model can tell immediately if a particular type of surface defect has occurred. The outcome is then published to the 'surface defect identification result' topic that informs the robot driver node if a response action is needed to pause the LAAM process. The details of the machine learning model will be discussed separately in Section 2.4.

The algorithm of in-situ point cloud processing embedded in a multi-nodal computing framework is a key contribution of this research. Multiple computing steps in the point cloud processing can be conducted automatically without human intervention. Unlike most of the previous research that performs surface defect identification only after the sensor has completed the current capturing cycle, the proposed method allows surface defects to be identified immediately during continuous surface monitoring without sensor intermittence. The subprocessing strategy in the proposed algorithm allows the data manipulation (i.e. point cloud cleaning) and data analysis (i.e. machine learning) to be performed concurrently and continuously. With the aforementioned capability, rapid surface defect identification can be achieved in LAAM.

2.4. Surface defect identification by combined unsupervised and supervised machine learning

In this research, surface defects are identified from the point cloud by a combination of unsupervised and supervised machine learning techniques. As an overview, an unsupervised clustering algorithm is first applied to isolate the point cloud regions that potentially contain surface defects, and then the clustering result is input to a supervised classification algorithm that examines the existence and type of defects in each of the point cloud regions. The details of the algorithms are explained in the rest of this section.

The result of surface defect identification for additively manufactured parts can be classified into four classes. As illustrated in Figure 9 (showing the cross section of the part), these classes include 'no defect' – labelled as 0, 'bulge defect' – labelled as 1, 'dent defect' – labelled as 2, and 'wavy defect' – labelled as 3. A bulge is a protruding region caused by excessive material deposition. A dent is an under-filled region caused by insufficient material deposition. A wavy surface is formed by the coexistence of both bulges and dents, which is commonly found in DED processes. The class label value $Y \in [0, 1, 2, 3]$ is the output of the multiclass classifier, while the input X is a set of numerical features extracted from the segmented point cloud.

Prior to feature extraction, an ideal reference surface needs to be obtained. The reference surface can either be retrieved from the part's CAD model or, if a CAD

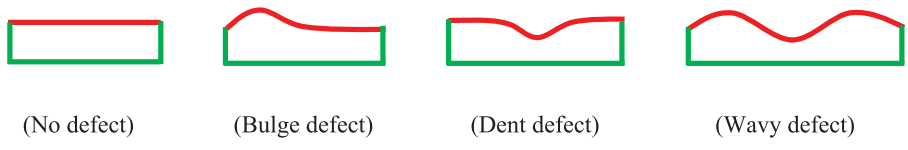

Figure 9. Four classes of surface defect identification result.

model is unavailable, be calculated by fitting a surface to the segmented point cloud. In both cases, the surface is represented by an equation $f(x, y, z) = 0$. Since most additively manufactured parts are printed in parallel layers, the reference surfaces obtained in this research are all planar, written as $f(x, y, z) = ax + by + cz + d = 0$. The coefficients in the surface equation are saved in a file that can be read by Subprocess 3. With the given coefficients, Euclidean distances from the reference surface to the point cloud can be calculated using Equation (3), where (x_i, y_i, z_i) is the coordinate of the ith point. Based on the distance values, the point cloud is splitted into three sets by two thresholds, d^+ and d^-, which are the positive and negative values of ½ of the nominal layer thickness. As expressed in Equation (4), points that are above the reference surface by at least d^+ are denoted as $\boldsymbol{P}_U$, points that are below the reference surface by at least d^- are denoted as $\boldsymbol{P}_L$, and the 'inline' points that are close to the reference surface within the $[d^-, d^+]$ tolerance are denoted as $\boldsymbol{P}_O$.

$$d_i = \frac{ax_i + by_i + cz_i + d}{\sqrt{a^2 + b^2 + c^2}} \tag{3}$$

$$\begin{aligned} \boldsymbol{P}_U &= \{p_i \in \boldsymbol{P} | d_i > d^+\} \\ \boldsymbol{P}_L &= \{p_i \in \boldsymbol{P} | d_i < d^-\} \\ \boldsymbol{P}_O &= \{p_i \in \boldsymbol{P} | d^- \le d_i \le d^+\} \end{aligned} \tag{4}$$

For each of the $\boldsymbol{P}_U$, $\boldsymbol{P}_L$, and $\boldsymbol{P}_O$ sets, unsupervised machine learning is applied first to find clusters of points that can be considered as either defective or defect-free regions. In particular, the DBSCAN clustering algorithm (Ester et al. 1996) is adopted. In this algorithm, points that have a high density of neighbours and occupy a large spaces are grouped to form a cluster. DBSCAN is chosen here due to its capability of finding clusters with arbitrary shapes from a large number of sample points. The pseudocode of DBSCAN for point cloud clustering is summarised in Figure 10, where ε is the radius of neighbourhood searching and *minPt* is the minimum number of neighbours that a point must have for it to be considered as a *core point*. If the core point is never visited in the previous iteration of the *FOR* loop, a new cluster is formed by this core point and all its neighbours. Non-core points that are within the ε distance to an existing cluster are labelled as *border points* and added to that cluster. The search for core and non-core points propagates until all clusters in the point cloud are found. As illustrated in Figure 11, clusters above the reference surface (i.e. formed by points in $\boldsymbol{P}_U$) are considered as bulge regions, and those below (i.e. formed by points in $\boldsymbol{P}_L$) are the dent regions. Clusters with points in the proximity of the reference surface are the defect-free flat regions. The few dispersed points (i.e. not belonging to any cluster) are considered as noise, regardless of their distances to the reference plane.

Numerical features of the point cloud can now be extracted from the clusters. A total of 12 features are extracted, as listed in Table 1, which are the evaluating metrics of the 3D size and shape of the point cloud clusters. Before training the classifier, the correlation between the output class and each of the numerical features needs to be computed using a statistical method. Only the features that are the most correlated with the output class are chosen as the input for the classifier training, while the less significant features can be ignored. The pairwise correlation (r_{ij}) among the features and the output class is computed by Spearman's formulation in Equation (5), where D_{ij} is the difference between the ranks of the ith and jth variables, and N is the number of observations (Hauke and Kossowski 2011). The value of r_{ij} ranges from −1 (i.e. the most negative correlation) to 1 (i.e. the most positive correlation), and 0 represents 'no correlation'.

$$r_{ij} = 1 - \frac{6\sum (D_{ij})^2}{N^3 - N} \in [-1,\ 1] \tag{5}$$

In this research, eight different classification algorithms are trained and compared for their accuracies in surface defect identification. These algorithms include Support Vector Machine (SVM), K-Nearest Neighbours (KNN), Gaussian Process (GP), Decision Tree (DT), Naive Bayes (NB), Artificial Neural Network (ANN), and two ensemble methods, Random Forest (RF) and AdaBoost (AB), that can combine multiple base learners. The open-source Scikit-learn library (Pedregosa et al. 2011) is adopted to implement the above algorithms in the proposed software platform. The performance of each classifier is influenced by its hyperparameters that are passed to the classifier as predetermined arguments. Hyperparameters can either be discrete selections (e.g. the kernel function type in SVM) or continuous numeric (e.g. the kernel coefficient in SVM). In this research, the hyperparameters of each classifier candidate are optimised by grid search, an exhaustive search method that compares all possible combinations of hyperparameter values. Although more time-consuming than the alternative random search method (Bergstra and Bengio 2012), grid search ensures that the global optimal hyperparameters can be found. In each iteration of the grid search, the classifier candidate is trained, and the performance is evaluated by a k-fold cross-validation method to avoid overfitting. In k-fold cross-validation, the training data is evenly split into k sets or 'folds'. The data in k–1 folds are used to train

DBSCAN algorithm (pseudocode) for clustering of point clouds

```
FOR point cloud set P' in {P_u, P_L, or P_O}:
    C = 0  #cluster counter
    FOR point P in P':
        Neighbours of P: Q(P) = {Q ∈ P_u | ‖Q−P‖ ≤ ε}
        Number of Neighbours: N(P) = Count(Q(P))

        IF N(P) ≥ minPt:
            Mark P as core point
            IF P is not in other clusters:
                C = C + 1
                Mark Q(P) as the C^th cluster: Q(P)^C
            ELSE:
                Add Q(P) to the C^th cluster: Q(P)^C = Q(P) ∪ Q(P)^C
            ENDIF
        ELSE IF ‖Q(P)^i−P‖ ≤ ε FOR i ∈ [1,C]:
            Mark P as border point
            Add P to the i^th cluster: Q(P)^i = {P} ∪ Q(P)^i
        ELSE:
            Mark P as noise point
        ENDIF
    ENDFOR
ENDFOR
```

Figure 10. The DBSCAN algorithm applied in point cloud clustering.

the classifier, while the remaining fold is used to validate the classifier's performance. In this research, the performance metric is the classification accuracy defined as the ratio of correct predictions to the total predictions, as expressed in Equation (6). The above validation is looped for each of the *k* folds, and the final 'score' of the k-fold cross-validation is the average accuracy value. At the end of the grid search, the hyperparameter combination that yields the highest cross-validated score is passed to the classifier.

$$\text{Accuracy} = \frac{\&\text{num; Correctly predicted classes}}{\&\text{num; Total predictions}} \quad (6)$$

The dataset obtained from experiments is randomly divided into the training and testing data in the ratio of 8:2. After the classifier is trained, its accuracy is evaluated using only the testing data. The definition of testing accuracy is the same as that in k-fold cross-validation, i.e. in Equation (6). The most accurate classifier can then be chosen to perform surface defect identification in Subprocess 3 of the point cloud processing node. By performing the classification using the extracted numerical features as input, Subprocess 3 can rapidly find surface defects before the next timeout in the subprocess handler.

Embedding the machine learning models within the in-situ point cloud processing algorithm is another key contribution of this research. The combination of unsupervised and supervised learning in surface defect identification allows defect regions to be isolated from the sensor measurement and the defect type to be recognised rapidly. The multi-nodal computing framework ensures a continuous stream of sensor data during the machine learning process, and at the same time it enables the sharing of defect identification result among different nodes.

3. Experimental results and discussion

Experiments were conducted to validate the proposed rapid surface defect identification method. The effectiveness of the in-situ point cloud processing and machine learning in DED surface monitoring was demonstrated successfully.

Figure 12 shows an example of a printed surface being monitoring in the LAAM system. Figure 12(a) is an as-fabricated metal part that has an uneven top surface due to process instability. Before this part was completed, its current layer was scanned by the laser profiler and the result is given in Figure 12(b). The initial point cloud data contained noises and part of the substrate. The in-situ point cloud processing node was executed in the multi-nodal software platform, and the filtering and segmentation functions were performed in Subprocess 1 concurrently as the sensor scanning was ongoing. The resultant cleaned point cloud is shown in Figure 12(c). The reference surface is superimposed with the point cloud in Figure 12(c), and the surface-to-point distances are indicated by the colour of the points. Within the in-situ point cloud processing

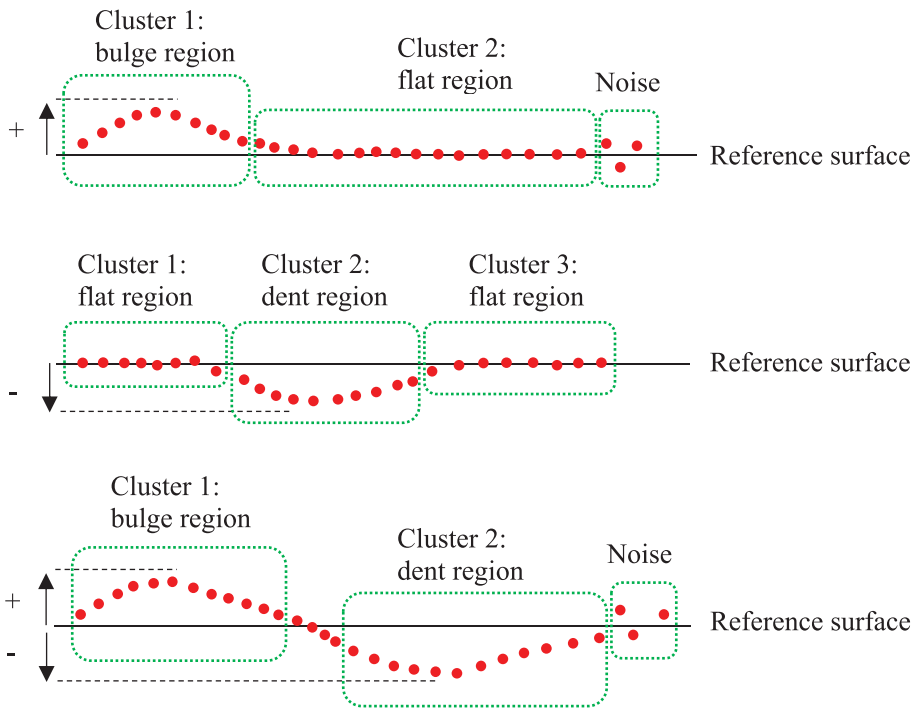

Figure 11. Point cloud clusters of bulge regions, dent regions, flat regions, and noise points.

node, the cleaned point cloud was passed to the machine learning model in Subprocess 3 for defect identification. The DBSCAN clustering was performed to identify point cloud regions that potentially contained surface defects. An example of the DBSCAN clustering result is shown in Figure 13, where three clusters are isolated.

In order to train the classifier for surface defect identification, 73 samples in different sizes and shapes were fabricated by the LAAM. Some of the examples are shown in Figure 14, and their corresponding transformed point clouds are shown in Figure 15. Surface features of the samples were extracted to construct the training dataset, and they included all four surface classes (i.e. 'no defect', 'bulge defect', 'dent defect', and 'wavy defect') as introduced in Section 2.4. The 73 samples, including those shown in Figure 14, were built from stainless steel 316L powder with the diameter of 45–90 µm. The LAAM process window for this material was obtained in preliminary studies before this research, and the ranges of operating parameters are listed in Table 2. In order to randomise the type of surface features on the samples, the operating parameters for each sample was chosen randomly from the process window. Therefore, as shown in Figure 14, even parts in similar shapes (e.g. the rectangular blocks) may have different surface features due to their different combination of operating parameters.

Table 1. Numerical features extracted from the point cloud.

No.	Feature Names	Descriptions
1	MaxBH	Maximum bulge height
2	MeanBH	Mean bulge height
3	StdBH	Standard deviation of bulge height
4	MaxDD	Maximum dent depth
5	MeanDD	Mean dent depth
6	StdDD	Standard deviation of dent depth
7	%B	Percentage of bulge area in the point cloud
8	%D	Percentage of dent area in the point cloud
9	SizeB	Bounding area's size of a bulge
10	SizeD	Bounding area's size of a dent
11	ARB	Bounding area's aspect ratio of a bulge
12	ARD	Bounding area's aspect ratio of a dent

The correlation matrix among the numerical features and the output class was calculated using the training dataset. The result is plotted as a heatmap in Figure 16, where the colour represents the level of correlation. It can be observed from the top row that the numerical features 'MaxDD', 'MeanDD', 'SizeB', 'SizeD', and 'ARD' were less correlated to the output class than the other features. Therefore, the above five features were removed from the classifier's input for both training and testing.

Eight classifiers were trained, and their hyperparameters were optimised by cross-validated grid search. The results are listed in Table 3. Terminologies and notations of the hyperparameters in Table 3 are adopted from the Scikit-learn library. Each of the trained classifiers was tested independently, and their performances were compared in terms of accuracies. The bar chart in Figure 17 shows the accuracies of different classifiers

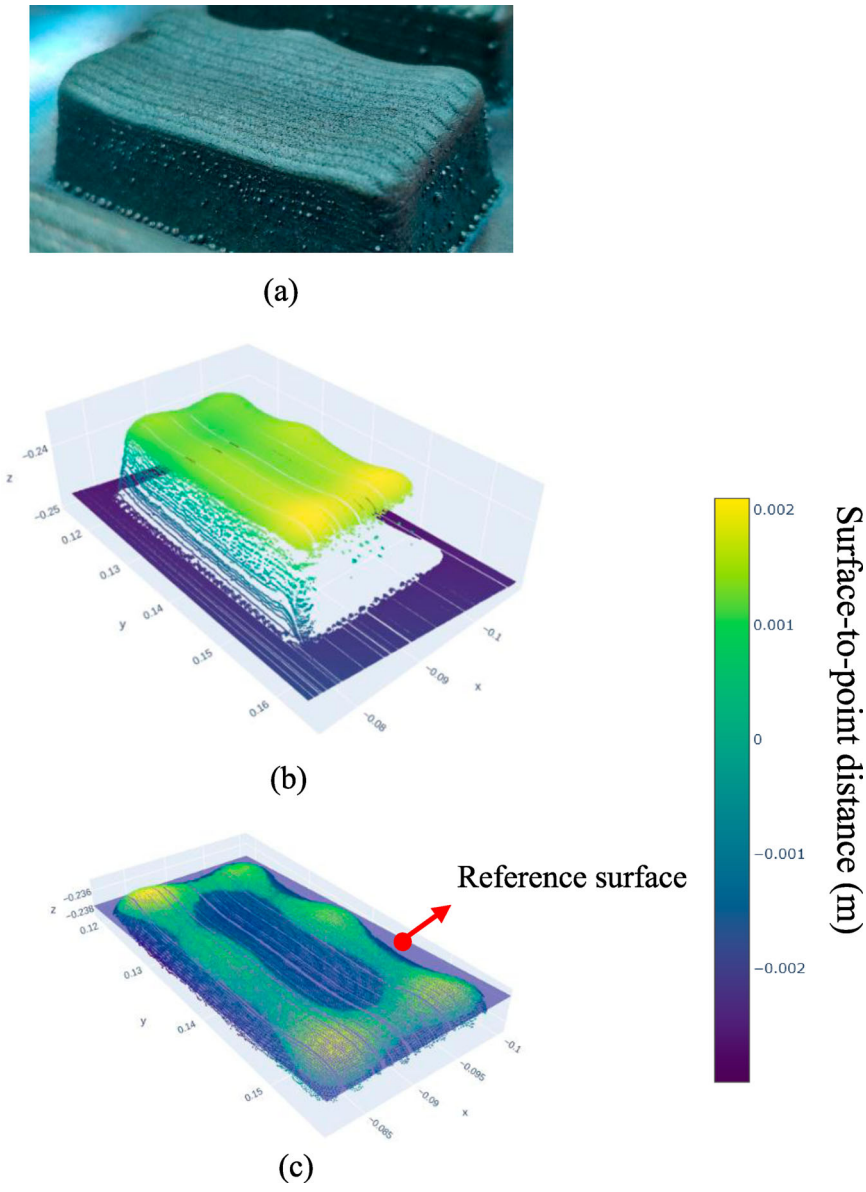

Figure 12. (a) An example of a printed part being monitored. (b) The initial point cloud containing noises. (c) The cleaned point cloud after filtering and segmentation.

in the testing experiments. It was found that while KNN was the simplest algorithm among the eight candidates, it was also the most accurate (93.15%). The other classifiers had accuracies less than 70%. In the KNN classifier, the feature-based distance from the *i*th test data (unlabelled) to the *j*th training data (labelled) is calculated by the Euclidean function (Singh, Yadav, and Rana 2013), as expressed in Equation (7) where L_n is the feature value and N is the total number of features being used in the classification. K training data that have the smallest distances to the test point (i.e. the 'nearest neighbours') are used to decide which class the test data should belong to. The KNN classification uses a majority voting approach, i.e. the test data is given the same label as the majority of its nearest neighbours. According to the optimal KNN hyperparameters shown in Table 3, the number of neighbours (K) is 9, meaning that each test point is voted by 9 neighbouring training points. The weight function is 'distance', meaning that the labels of the nearest neighbours are assigned with weights that are inversely proportional to the distance from the test point. A KD tree algorithm is applied to search the nearest neighbours efficiently (Bentley 1975).

$$\mathrm{Dist}(X_i, X_j) = \sqrt{\sum_{n}^{N} (L_{X_i,n} - L_{X_j,n})} \tag{7}$$

Due to its high accuracy found by the experiments, KNN was chosen as the final classifier that was added to

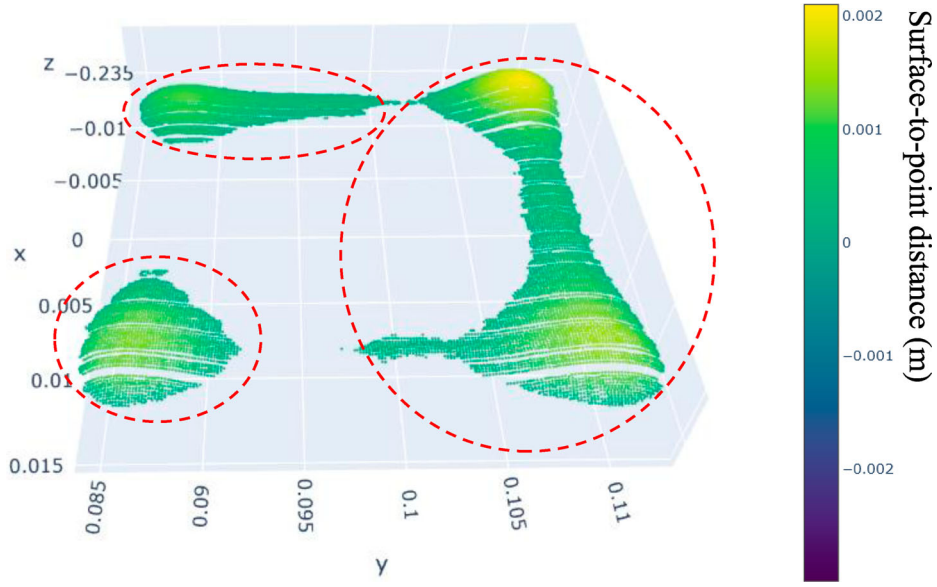

Figure 13. An example of DBSCAN clustering result that shows surface defects in 3D.

Subprocess 3 of the proposed software platform to perform surface defect identification. When any one of the three defect classes was detected, a message was sent by Subprocess 3 to notify both the human operator and the robot controller. As illustrated in Figure 5, in the multi-nodal software architecture developed in this research, the robot driver node received the surface defect identification result published by the point cloud processing node (in which Subprocess 3 was running). Via the TCP channel, the 'robot command' socket within the robot driver node sent an analogue signal that reduced the laser power to zero, followed by two digital signals that switched off both the laser source and the robot motion. By doing this, the LAAM process could be paused immediately after surface defects occurred, thus preventing further quality deterioration leading to unrepairable build failures. Since surface defects in the LAAM are sometimes caused by hardware abnormalities of the laser source, nozzle, powder feeder, and chiller system, the automatic pausing mechanism also plays a vital role in ensuring safety and avoiding serious system fault. Subsequent actions can include re-programming the LAAM tool path to repair the surface, as well as troubleshooting and remediating system errors before the printing process could be resumed. Based on the 3D point cloud of surface defects, it is also possible to generate the repairing tool path automatically, which is part of the authors' future work.

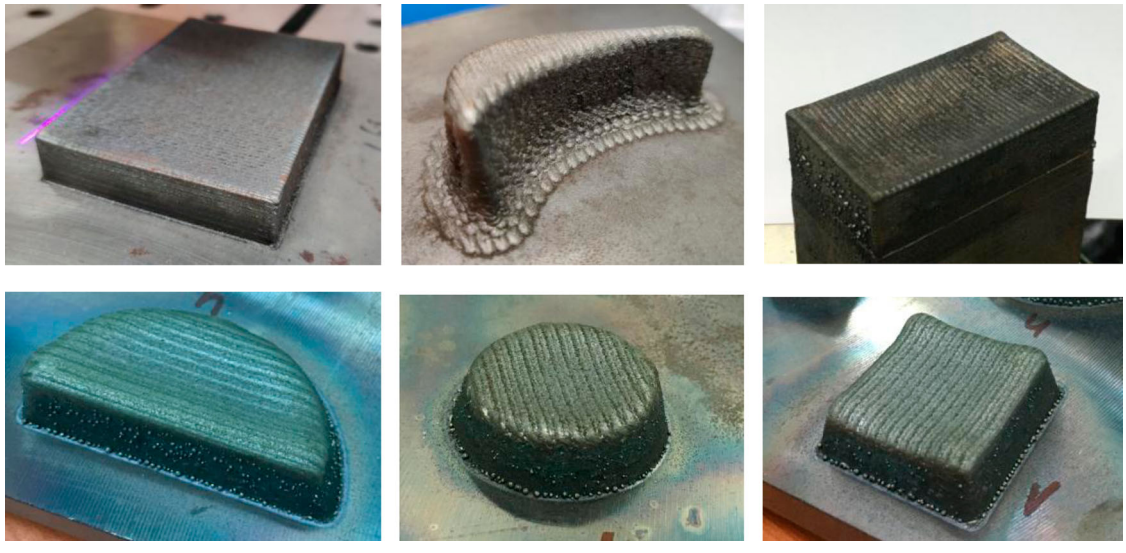

Figure 14. Examples of LAAM-printed samples used in classifier training.

Table 2. The LAAM process window for sample fabrication using stainless steel 316L powder.

Laser power	Laser spot diameter	Printing speed	Layer thickness	Powder feeding rate (on the GTV feeder)
850–1050 W	2–3 mm	20–30 mm/s	0.2–0.3 mm	1.5–1.8 rpm

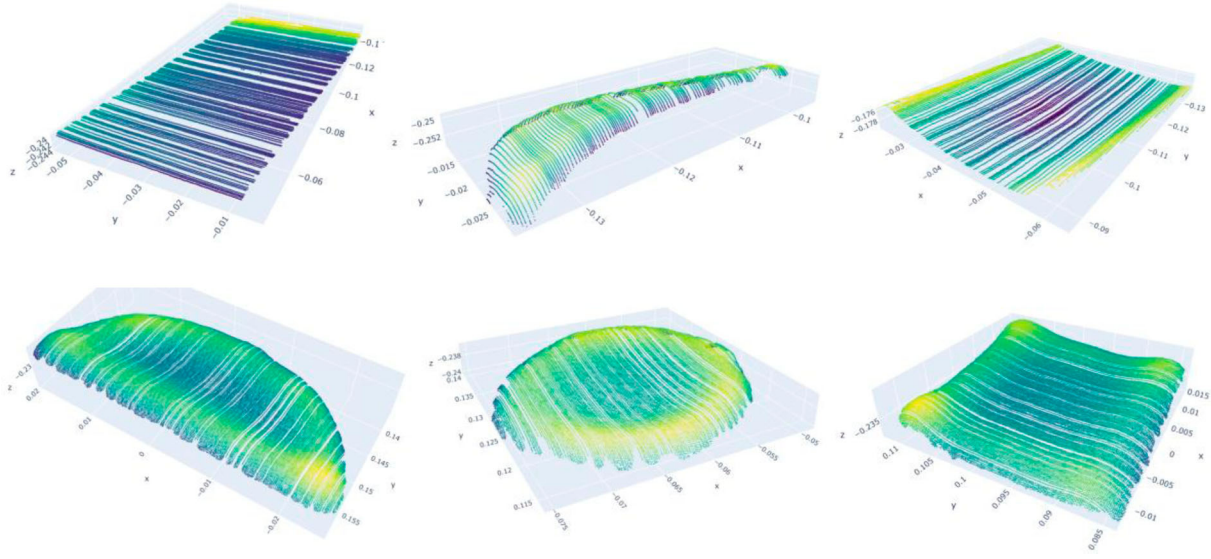

Figure 15. Examples of transformed point clouds used in classifier training.

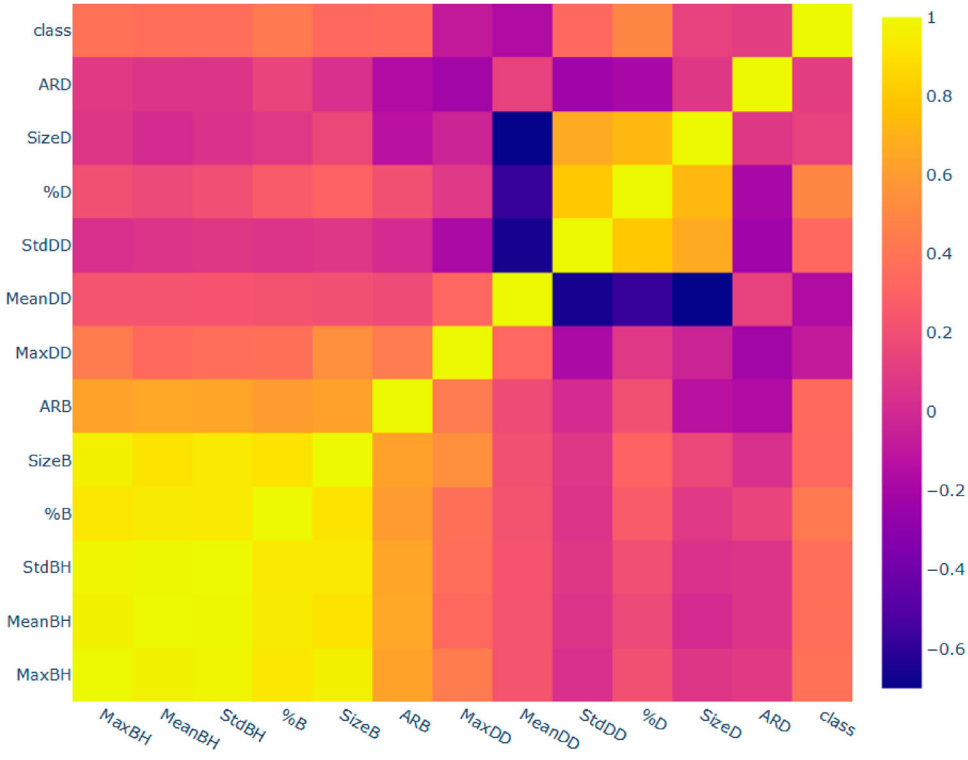

Figure 16. The heatmap that show correlations between numerical features and the output class.

The proposed surface defect identification method is compared with the three-dimensional digital image correlation (DIC), a state-of-the-art surface monitoring technique successfully applied in AM processes such as the FDM and PBF (Holzmond and Li 2017; Bartlett et al. 2018; Bartlett and Li 2019). Both the proposed method

Table 3. The optimal hyperparameters of the trained classifiers.

Classifiers	Hyperparameters	Optimal values
SVM	Kernel	Radial basis function (RBF)
	Regularization parameter (C)	100
	Kernel coefficient (γ)	0.001
KNN	Number of neighbours	9
	Weight function	Distance
	Algorithm to calculate nearest neighbours	KDTree
GP	Kernel	1.0 * RBF(1.0)
	Optimizer	None
DT	Minimum sample split	2
	The splitter strategy	Random
	Criteria for split quality	Gini impurity
	Maximum depth of the tree	4
NB	Variance smoothing	1e−9
ANN	Activation function	'Identity'
	Solver for weight optimization	Stochastic gradient descent (SGD)
	Learning rate	Constant
	Number of neurons in the 1st hidden layer	20
	Number of neurons in the 2nd hidden layer	10
RF	Minimum sample split	4
	Number of tree estimators:4	4
	Criteria for split quality	Gini impurity
	Maximum depth of the tree	3
AB	Number of estimators	10
	Boosting algorithm	SAMME

and the DIC can capture surface defects in 3D. The DIC utilises a dual-camera setup fixed above the build platform, and the images from both cameras are correlated using the intrinsic pattern of the targeted surface, thus converting 2D pixels into 3D coordinates. Unlike the DIC, the proposed method uses a robotised laser profiler that moves around the workpiece instead of being fixed. Therefore, the surface monitoring area is not restricted by either the field-of-view or the focusing range of the camera. The DIC is troubled by occasional obstruction of the camera view by the print head (Holzmond and Li 2017), while the proposed method with a robotised laser profiler is free from this issue, making it suitable for the DED process. In this research, the multi-nodal computing framework integrated with machine learning models can rapidly detect and classify different types of surface defects even when multiple defect regions appear in the same layer, which was not reported in the literature of DIC.

The DIC has been applied in residual stress measurement for AM parts (Croom, Bumgardner, and Li 2016; Bartlett et al. 2018; Bartlett and Li 2019). In principle, the technology developed in this paper can also be applied to measure residual stress if it is used in PBF systems (e.g. SLM and EBM). Just like the DIC, the proposed method can obtain curved surface profiles in 3D, and the mapping from the surface curvature to the residual stress can be done in the same way as reported by the above references. Residual stress is the main cause of part distortion in the PBF, and a deterministic elastic model can be used to convert the warpage to the residual stress (Bartlett et al. 2018). However, this paper focuses on the DED process which is different from the PBF in terms of surface defect formation. In the DED, surface defects are influenced by not only the residual stress but also intrinsic process instabilities, such as the shifting of the powder stream focus relative to the laser spot and the inconsistent robot speed, that can result in wavy layers and collapsed edges instead of only warpages (Shi et al. 2020). Therefore, a deterministic mapping between surface curvature and residual stress is less feasible in DED, and the residual stress measurement using the proposed surface monitoring method is not performed in this research.

The technology developed in this research can also be applied by the general 3D printing community

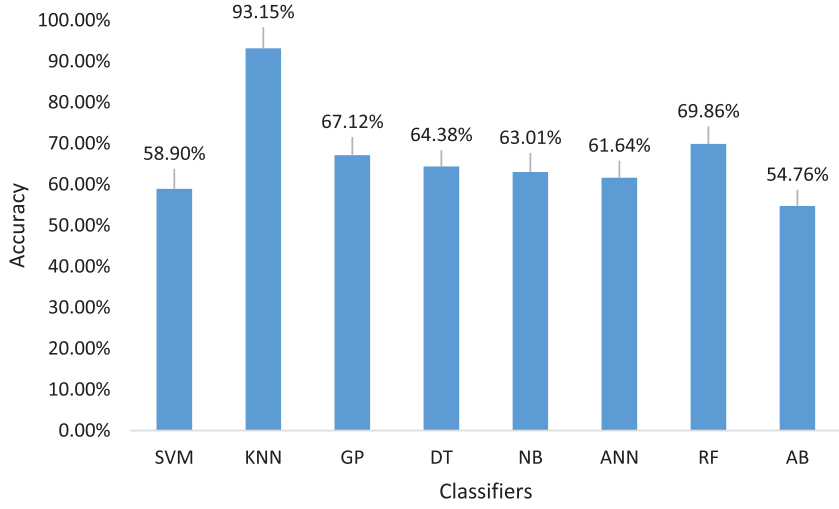

Figure 17. Accuracies of the trained classifiers.

beyond the current focus on the DED. Other processes may find the proposed method useful in monitoring the printed parts' surface conditions. Although different training datasets need to be experimentally generated for classifier training in different AM processes, the framework of the proposed method and the multi-nodal software platform can remain mostly intact, making the technology easy to adopt by different processes.

4. Conclusions

In this paper, a novel rapid surface defect identification method for DED-type additive manufacturing is proposed. The key contribution of this work is the development of an in-situ point cloud processing with machine learning methods that enables automatic and continuous surface monitoring without sensor intermittence or human intervention. A laser profiler was integrated into a robot-based LAAM system that was used to fabricate metallic parts with powders. A software platform was developed with a multi-nodal architecture. Within the software platform, multiple subprocesses were executed in parallel, which enabled the in-situ point cloud processing and analysis. A machine learning model that combines clustering and multiclass classification was embedded in the in-situ point cloud processing node to identify surface defects. Experiments were conducted to train the machine learning model and validate the proposed method. Eight different classifiers were tested and optimised. The KNN model demonstrated the best performance with the highest surface defect identification accuracy of 93.15%. The proposed method has successfully detected and classified surface defects of additively manufactured parts with minimal human input. Based on the technology developed in this work, future research can be conducted to automatically repair the surface defects by generating DED tool paths based on the point cloud data, with the target to further enhance the productivity of DED processes.

Acknowledgements

This work was supported by A*ccelerate under the GAP grant for Machine Learning Platform for Manufacturing Equipment and Solutions (ACCL/19-GAP025-R20A). We also wish to acknowledge the funding support for this project from Nanyang Technological University under the Undergraduate Research Experience on Campus (URECA) programme.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by A*ccelerate [grant number ACCL/19-GAP025-R20A].

Notes on contributors

Lequn Chen is a student of Mechanical Engineering at Nanyang Technological University, Singapore.

Xiling Yao is a research scientist at Singapore Institute of Manufacturing Technology.

Peng Xu is a research scientist at Singapore Institute of Manufacturing Technology.

Seung Ki Moon is an associate professor at Nanyang Technological University, Singapore.

Guijun Bi is a senior research scientist at Singapore Institute of Manufacturing Technology.

ORCID

Xiling Yao http://orcid.org/0000-0002-6478-5842

References

Bartlett, Jamison L., Brendan P. Croom, Jeffrey Burdick, Daniel Henkel, and Xiaodong Li. 2018. "Revealing Mechanisms of Residual Stress Development in Additive Manufacturing via Digital Image Correlation." *Additive Manufacturing* 22 (August): 1–12. doi:10.1016/j.addma.2018.04.025.

Bartlett, Jamison L., and Xiaodong Li. 2019. "An Overview of Residual Stresses in Metal Powder Bed Fusion." *Additive Manufacturing* 27 (May): 131–149. doi:10.1016/j.addma.2019.02.020.

Bentley, Jon Louis. 1975. "Multidimensional Binary Search Trees Used for Associative Searching." *Communications of the ACM* 18 (9): 509–517. doi:10.1145/361002.361007.

Bergstra, James, and Yoshua Bengio. 2012. "Random Search for Hyper-parameter Optimization." *Journal of Machine Learning Research* 13 (10): 281–305.

Bishop, Elizabeth G., and Simon James Leigh. 2020. "Using Large-Scale Additive Manufacturing as a Bridge Manufacturing Process in Response to Shortages in Personal Protective Equipment during the COVID-19 Outbreak." *International Journal of Bioprinting* 6 (4). doi:10.18063/ijb.v6i4.281.

Celik, H. Kursat, Ozkan Kose, Mihaela-Elena Ulmeanu, Allan E. W. Rennie, Thomas N. Abram, and Ibrahim Akinci. 2020. "Design and Additive Manufacturing of Medical Face Shield for Healthcare Workers Battling Coronavirus (COVID-19)." *International Journal of Bioprinting* 6 (4). doi:10.18063/ijb.v6i4.286.

Chen, Lequn, Xiling Yao, Peng Xu, Seung Ki Moon, and Guijun Bi. 2020. "Surface Monitoring for Additive Manufacturing with In-Situ Point Cloud Processing." In *2020 6th International Conference on Control, Automation and Robotics (ICCAR)*, 196–201. doi:10.1109/ICCAR49639.2020.9108092.

Choong, Yu Ying Clarrisa, Saeed Maleksaeedi, Hengky Eng, Suzhu Yu, Jun Wei, and Pei-Chen Su. 2020. "High Speed

4D Printing of Shape Memory Polymers with Nanosilica." *Applied Materials Today* 18 (March). doi:10.1016/j.apmt.2019.100515.

Choong, Yu Ying Clarrisa, Hong Wei Tan, Deven C. Patel, Wan Ting Natalie Choong, Chun-Hsien Chen, Hong Yee Low, Ming Jen Tan, Chandrakant D. Patel, and Chee Kai Chua. 2020. "The Global Rise of 3D Printing during the COVID-19 Pandemic." *Nature Reviews Materials* 5 (9): 637–639. doi:10.1038/s41578-020-00234-3.

Chua, Zhong Yang, Il Hyuk Ahn, and Seung Ki Moon. 2017. "Process Monitoring and Inspection Systems in Metal Additive Manufacturing: Status and Applications." *International Journal of Precision Engineering and Manufacturing-Green Technology* 4 (2): 235–245. doi:10.1007/s40684-017-0029-7.

Collins, P. C., D. A. Brice, P. Samimi, I. Ghamarian, and H. L. Fraser. 2016. "Microstructural Control of Additively Manufactured Metallic Materials." *Annual Review of Materials Research* 46 (1): 63–91. doi:10.1146/annurev-matsci-070115-031816.

Croom, Brendan P., Clifton Bumgardner, and Xiaodong Li. 2016. "Unveiling Residual Stresses in Air Plasma Spray Coatings by Digital Image Correlation." *Extreme Mechanics Letters* 7 (June): 126–135. doi:10.1016/j.eml.2016.02.013.

Ester, Martin, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise." In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 226–231. KDD'96. Portland, Oregon: AAAI Press.

García-Díaz, Antón, Verónica Panadeiro, Baltasar Lodeiro, Jorge Rodríguez-Araújo, John Stavridis, Alexios Papacharalampopoulos, and Panagiotis Stavropoulos. 2018. "OpenLMD, an Open Source Middleware and Toolkit for Laser-Based Additive Manufacturing of Large Metal Parts." *Robotics and Computer-Integrated Manufacturing* 53 (October): 153–161. doi:10.1016/j.rcim.2018.04.006.

Grasso, Marco, and Bianca Maria Colosimo. 2017. "Process Defects and in Situ Monitoring Methods in Metal Powder Bed Fusion: A Review." *Measurement Science and Technology* 28 (4), doi:10.1088/1361-6501/aa5c4f.

Hauke, Jan, and Tomasz Kossowski. 2011. "Comparison of Values of Pearson's and Spearman's Correlation Coefficients on the Same Sets of Data." *Quaestiones Geographicae* 30 (2): 87–93. doi:https://doi.org/10.2478/v10117-011-0021-1.

He, Haijun, Min Gao, Balázs Illés, and Kolos Molnar. 2020. "3D Printed and Electrospun, Transparent, Hierarchical Polylactic Acid Mask Nanoporous Filter." *International Journal of Bioprinting* 6 (4). doi:10.18063/ijb.v6i4.278.

Holzmond, Oliver, and Xiaodong Li. 2017. "In Situ Real Time Defect Detection of 3D Printed Parts." *Additive Manufacturing* 17 (October): 135–142. doi:10.1016/j.addma.2017.08.003.

Jiang, Jingchao, and Yongsheng Ma. 2020. "Path Planning Strategies to Optimize Accuracy, Quality, Build Time and Material Use in Additive Manufacturing: A Review." *Micromachines* 11 (7): 633. doi:10.3390/mi11070633.

Jiang, Jingchao, Fei Weng, Shiming Gao, Jonathan Stringer, Xun Xu, and Ping Guo. 2019. "A Support Interface Method for Easy Part Removal in Directed Energy Deposition." *Manufacturing Letters* 20 (April): 30–33. doi:10.1016/j.mfglet.2019.04.002.

Jiang, Jingchao, Chunling Yu, Xun Xu, Yongsheng Ma, and Jikai Liu. 2020. "Achieving Better Connections between Deposited Lines in Additive Manufacturing via Machine Learning." *Mathematical Biosciences and Engineering* 17 (April): 3382–3394. doi:10.3934/mbe.2020191.

Kam, Hyeong Ryeol, Sung-Ho Lee, Taejung Park, and Chang-Hun Kim. 2015. "RViz: A Toolkit for Real Domain Data Visualization." *Telecommunications Systems* 60 (2): 337–345. doi:10.1007/s11235-015-0034-5.

Kuo, C. N., C. K. Chua, P. C. Peng, Y. W. Chen, S. L. Sing, S. Huang, and Y. L. Su. 2020. "Microstructure Evolution and Mechanical Property Response via 3D Printing Parameter Development of Al–Sc Alloy." *Virtual and Physical Prototyping* 15 (1): 120–129. doi:10.1080/17452759.2019.1698967.

Lao, Wenxin, Mingyang Li, Teck Neng Wong, Ming Jen Tan, and Tegoeh Tjahjowidodo. 2020. "Improving Surface Finish Quality in Extrusion-Based 3D Concrete Printing Using Machine Learning-Based Extrudate Geometry Control." *Virtual and Physical Prototyping* 15 (2): 178–193. doi:10.1080/17452759.2020.1713580.

Li, Zhongwei, Xingjian Liu, Shifeng Wen, Piyao He, Kai Zhong, Qingsong Wei, Yusheng Shi, and Sheng Liu. 2018. "In Situ 3D Monitoring of Geometric Signatures in the Powder-Bed-Fusion Additive Manufacturing Process via Vision Sensing Methods." *Sensors* 18 (4): 1180. doi:10.3390/s18041180.

Lu, Q. Y., and C. H. Wong. 2018. "Additive Manufacturing Process Monitoring and Control by Non-Destructive Testing Techniques: Challenges and in-Process Monitoring." *Virtual and Physical Prototyping* 13 (2): 39–48. doi:10.1080/17452759.2017.1351201.

Ng, Wei Long, Alvin Chan, Yew Soon Ong, and Chee Kai Chua. 2020. "Deep Learning for Fabrication and Maturation of 3D Bioprinted Tissues and Organs." *Virtual and Physical Prototyping* 15 (3): 340–358. doi:10.1080/17452759.2020.1771741.

Ng, Wei Long, Chee Kai Chua, and Yu-Fang Shen. 2019. "Print Me An Organ! Why We Are Not There Yet." *Progress in Polymer Science* 97 (October). doi:10.1016/j.progpolymsci.2019.101145.

Okarma, Krzysztof, and Jarosław Fastowicz. 2020. "Computer Vision Methods for Non-Destructive Quality Assessment in Additive Manufacturing." In *Progress in Computer Recognition Systems, edited by Robert Burduk, Marek Kurzynski, and Michał Wozniak*, 11–20. Advances in Intelligent Systems and Computing. Cham: Springer International Publishing. doi:10.1007/978-3-030-19738-4_2.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, et al. 2011. "Scikit-Learn: Machine Learning in Python." *Journal of Machine Learning Research* 12 (October): 2825–2830.

Quigley, Morgan, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. 2009. "ROS: An Open-Source Robot Operating System." In *ICRA Workshop on Open Source Software*, 3:5. Kobe.

Rodríguez-Araújo, Jorge, and Juan J. Rodríguez-Andina. 2015. "ROS-Based 3D on-Line Monitoring of LMD Robotized Cells." In *2015 IEEE 13th International Conference on*

Industrial Informatics (INDIN), 308–313. doi:10.1109/INDIN.2015.7281752.

Rusu, Radu Bogdan, and Steve Cousins. 2011. "3D Is Here: Point Cloud Library (PCL)." In *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai.

Sabbaghi, Arman, Qiang Huang, and Tirthankar Dasgupta. 2018. "Bayesian Model Building From Small Samples of Disparate Data for Capturing In-Plane Deviation in Additive Manufacturing." *Technometrics* 60 (4): 532–544. doi:10.1080/00401706.2017.1391715.

Shi, Tuo, Jianjun Shi, Zhixin Xia, Bingheng Lu, Shihong Shi, and Geyan Fu. 2020. "Precise Control of Variable-Height Laser Metal Deposition Using a Height Memory Strategy." *Journal of Manufacturing Processes* 57 (September): 222–232. doi:10.1016/j.jmapro.2020.05.026.

Singh, Archana, Avantika Yadav, and Ajay Rana. 2013. "K-Means with Three Different Distance Metrics." *International Journal of Computer Applications* 67 (10): 13–17. doi:10.5120/11430-6785.

Sohnius, Felix, Peter Schlegel, Max Ellerich, and Robert H. Schmitt. 2019. "Data-Driven Prediction of Surface Quality in Fused Deposition Modeling Using Machine Learning." In *Production at the Leading Edge of Technology*, edited by Jens Peter Wulfsberg, Wolfgang Hintze, and Bernd-Arno Behrens, 473–481. Berlin: Springer. doi:10.1007/978-3-662-60417-5_47.

Tan, Hong Wei, Jia An, Chee Kai Chua, and Tuan Tran. 2019. "Metallic Nanoparticle Inks for 3D Printing of Electronics." *Advanced Electronic Materials* 5 (5). doi:10.1002/aelm.201800831.

Tan, Xipeng, Yihong Kok, Yu Jun Tan, Marion Descoins, Dominique Mangelinck, Shu Beng Tor, Kah Fai Leong, and Chee Kai Chua. 2015. "Graded Microstructure and Mechanical Properties of Additive Manufactured Ti–6Al–4 V via Electron Beam Melting." *Acta Materialia* 97 (September): 1–16. doi:10.1016/j.actamat.2015.06.036.

Tan, Hong Wei, Nitipon Saengchairat, Guo Liang Goh, Jia An, Chee Kai Chua, and Tuan Tran. 2020. "Induction Sintering of Silver Nanoparticle Inks on Polyimide Substrates." *Advanced Materials Technologies* 5 (1). doi:10.1002/admt.201900897.

Tang, Shangyong, Guilan Wang, and Haiou Zhang. 2019. "In Situ 3D Monitoring and Control of Geometric Signatures in Wire and Arc Additive Manufacturing." *Surface Topography: Metrology and Properties* 7 (2). doi:10.1088/2051-672X/ab1c98.

Tsai, R. Y., and R. K. Lenz. 1989. "A New Technique for Fully Autonomous and Efficient 3D Robotics Hand/Eye Calibration." *IEEE Transactions on Robotics and Automation* 5 (3): 345–358. doi:10.1109/70.34770.

Wu, Dazhong, Yupeng Wei, and Janis Terpenny. 2018. "Surface Roughness Prediction in Additive Manufacturing Using Machine Learning." In *MSEC2018*. Volume 3: Manufacturing Equipment and Systems. doi:10.1115/MSEC2018-6501.

Yu, Chunling, and Jingchao Jiang. 2020. "A Perspective on Using Machine Learning in 3D Bioprinting." *International Journal of Bioprinting* 6 (1). doi:10.18063/ijb.v6i1.253.

Yu, W. H., S. L. Sing, C. K. Chua, C. N. Kuo, and X. L. Tian. 2019. "Particle-Reinforced Metal Matrix Nanocomposites Fabricated by Selective Laser Melting: A State of the Art Review." *Progress in Materials Science* 104 (July): 330–379. doi:10.1016/j.pmatsci.2019.04.006.

Yu, Wenhui, Swee Leong Sing, Chee Kai Chua, and Xuelei Tian. 2019. "Influence of Re-Melting on Surface Roughness and Porosity of AlSi10Mg Parts Fabricated by Selective Laser Melting." *Journal of Alloys and Compounds* 792 (July): 574–581. doi:10.1016/j.jallcom.2019.04.017.