Adaptive Toolpath Planning for Hybrid Manufacturing Based on Raw 3D Scanning Data

Panagiotis Stavropoulos(✉), Lydia Athanasopoulou, Thanassis Souflas, and Konstantinos Tzimanis

Laboratory for Manufacturing Systems and Automation (LMS), Department of Mechanical Engineering and Aeronautics, University of Patras, Patras, Greece
pstavr@lms.mech.upatras.gr

Abstract. The wide industrial adoption of metal Additive Manufacturing (AM) technologies has shown that AM can rarely thrive on its own but should be integrated in a wider manufacturing chain. The integration of AM with milling, to form the so-called hybrid manufacturing process is one of the most popular approaches. A key bottleneck of hybrid manufacturing is toolpath planning. In a hybrid manufacturing workflow, 3D scanning is often integrated between milling and AM, in an inspection step, which is highly important for the correct planning of the next process, since AM parts can have a significant divergence from their original CAD. In order to plan the toolpath of the successive AM or milling process, one must consider the 3D scan as an input, posing a new challenge compared to traditional toolpath planning technologies. The surface reconstruction of the part, based on the point cloud can be very computationally heavy. To this end, new techniques for toolpath planning directly from the raw 3D scanning data are necessary. This study presents such a technique to provide an automated tool for toolpath planning of both AM and machining processes, based on point cloud data. Edge extraction is performed through eigenvalue analysis, followed by segmentation of the part based on popular clustering algorithms. Next, curve fitting on the external and internal parts of the segmented point cloud is performed using alpha shapes, followed by polyline offsetting for the toolpath generation. The methodology is validated through a case study on a real component.

Keywords: Hybrid Manufacturing · Additive Manufacturing · Milling · Toolpath Planning · Point Cloud · 3D scanning

1 Introduction

The industrial adoption of Additive Manufacturing (AM) due to its numerous benefits [1, 2] has also highlighted the bottlenecks of AM that are inherent to its process mechanism, namely the poor surface quality and dimensional accuracy. To address this, the integration of AM in a wider manufacturing process chain is proposed [3]. Hybrid manufacturing, incorporating AM and subtractive manufacturing, usually in the form of milling, is a popular approach, with commercial machine tools already available. Apart from the physical integration of the processes, their digital integration is also crucial to achieve

F. J. G. Silva et al. (Eds.): FAIM 2023, LNME, pp. 273–282, 2024.
https://doi.org/10.1007/978-3-031-38241-3_31

an efficient hybrid manufacturing workflow. For this purpose, 3D scanning, integrated between AM and milling as an intermediate inspection step, is a key enabler. 3D scanning can provide an accurate representation of the actual status of the manufactured part and quantify the divergence of the actual part, manufactured by AM, from its nominal 3D CAD model. This facilitates the appropriate toolpath planning for a successive AM or milling process, under the hybrid manufacturing context [4]. This adaptive toolpath planning approach, using the 3D scanning results, in the form of a point cloud, as an input enables the seamless digital integration of the two processes. Nevertheless, it introduces a new complexity level, compared to traditional toolpath planning processes that are followed from commercial CAM systems. This is linked to the representation of the 3D model of the part in the form of the point cloud, instead of a standardized surface-based representation (e.g., BREP). The surface reconstruction from the point cloud to enable the direct use of existing toolpath planning approaches is a computationally heavy process, requiring significant user expertise. On the other hand, the use of raw point clouds introduces additional challenges, due to the lack of robustness in the point cloud, particularly when it contains unorganized data. To this end, there is a need for approaches that address these challenges and enable direct toolpath planning from raw point cloud data.

Although the published literature on the topic of toolpath generation from raw point cloud data is still limited, there have been various approaches proposed. Masood et al. [5] have developed an algorithm for toolpath generation based on point clouds that were divided into segments. Each segment was fitted with a B-Spline, which represented the machining toolpath. Zou and Zhao [6] utilized conformal point cloud parametrization to generate machining toolpaths for freeform surfaces. Dhanda et al. [7] used a curvature-based segmentation approach for the point cloud to partition the point cloud into several regions. Then, a grid-based adaptive planar toolpath strategy was employed to machine each region independently. Chui et al. [8] have proposed an algorithm for toolpath generation in 5-axis machining directly from point cloud data. They performed the triangulation of the point cloud and used the mesh points as tool contact locations. Then, a 3D biarc fitting technique was used to generate the 5-axis toolpath. Ghogare et al. [9] proposed a method for direct toolpath planning from point clouds, using the boundary of the point cloud as a master cutter path and calculating adjacent side toolpaths, using and iso-scallop strategy. Popescu et al. [10] have proposed an approach based on the graph theory for toolpath generation in point clouds for roughing milling operations. The cutting areas are identified using a binary map and the toolpath is generated using the Dijkstra algorithm inside the graph.

2 Approach

This approach takes as input the scanned 3D model of the manufactured part, in the form of a point cloud and the CAD model of the original geometry of the part aiming to guide the supervised algorithms that will be used for the intermediate steps until the tool path generation. The CAD of the part is digitized in a point cloud format, before it is used as an input for the proposed algorithm. The proposed approach is based on four key building blocks. First, the scanned part is registered in the CAD coordinate system and aligned with the original model. Next, the deviations of the scanned part are calculated (disparity map) and the volume to be processed is extracted. This is performed in an open-source software (CloudCompare [11]) in this work. In general, it is considered that the volume to be processed will be used as an input to the proposed algorithm, based on any commercial software, therefore the disparity mapping is out of the scope of this paper.

Next, the volume to be processed is sliced according to the selected process parameters (axial depth of cut for milling or layer height for AM). After that, the preparation of the data for the extraction of the inner and outer contours is performed. The edges (inner and outer) of the volume are identified, to detect its limits in the cartesian space, as well as the existence of islands (pockets, etc.) in the part. For this step, the original CAD of the part is digitized, and its point cloud is generated, where the edge detection is performed (Sect. 3.1).

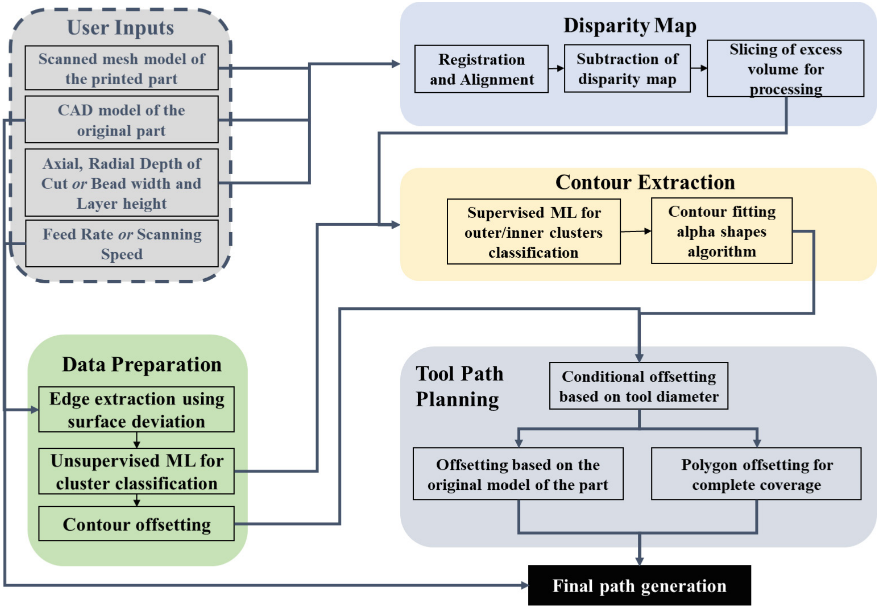
User Inputs
Scanned mesh model of the printed part
CAD model of the original part
Axial, Radial Depth of Cut or Bead width and Layer height
Feed Rate or Scanning Speed
Disparity Map
Registration and Alignment
Subtraction of disparity map
Slicing of excess volume for processing
Contour Extraction
Supervised ML for outer/inner clusters classification
Contour fitting alpha shapes algorithm
Data Preparation
Edge extraction using surface deviation
Unsupervised ML for cluster classification
Contour offsetting
Tool Path Planning
Conditional offsetting based on tool diameter
Offsetting based on the original model of the part
Polygon offsetting for complete coverage
Final path generation

Fig. 1. Toolpath planning workflow

When the data preparation is finished, the outer and inner contours are extracted and segmented, using Machine Learning (ML) algorithms (Sect. 3.2). Finally, the toolpath is generated, and the part program is written (Sect. 3.3). The following paragraphs will clarify the way that the present work enables the automation regarding the transition from the initial scanned model to the tool path generation either for AM or for machining without the need for user input during the several steps of the methodology. The approach is summarized in Fig. 1.

2.1 Eigenvalue Analysis and Edge Detection

For the edge detection, the work from Bazazian et al. [12] has been adopted. The point clouds that are generated from 3D scanning do not contain any information regarding the normal vectors of the points which means that it is not known if the points belong to flat surfaces or edges. The knowledge of normal vectors at each point can facilitate the identification of an edge point. However, given the fact that point clouds are unstructured, a normal vector for a single point cannot be identified, thus requiring the examination of its neighboring points, to form a local surface. Principal Component Analysis (PCA) is applied to neighborhoods of the point clouds to estimate the local normal vectors. For every point of the cloud, a least squares local plane is fitted to its k nearest neighbors. The normal of each point is the eigenvector corresponding to the smallest eigenvalue of the covariance matrix. Covariance is an indicator showing how much each of the dimensions varies from the mean with respect to each other. The covariance matrix for a sample point of a 3-dimensional dataset is given below:

$$C = \begin{bmatrix} Cov(x,x) & Cov(x,y) & Cov(z,x) \\ Cov(y,x) & Cov(y,y) & Cov(z,y) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{bmatrix} \tag{1}$$

The covariance $Cov(x,y)$ can be computed as it can be seen below while the same procedure applies for all the other covariance values.

$$Cov(x,y) = \frac{\sum_{i=1}^{k}(x_i - \overline{x})(y_i - \overline{y})}{n-1} \tag{2}$$

Since the estimation of normal vectors by PCA is based on the calculation of the eigenvalues of the covariance matrix, it is possible to identify edge features solely based on the variation of these eigenvalues for each point. With all the values of the matrix known, the eigenvalues of the covariance matrix (λ0, λ1, λ2) are utilized. The concept of the surface variation is introduced. The surface variation, $\sigma_k(p)$, , for each sample point with k neighbors enables to distinguish whether the point belongs to a flat plane or a salient point (edge).

$$\sigma_k(p) = \frac{\lambda_0}{\lambda_1 + \lambda_2 + \lambda_3} \tag{3}$$

Since the smallest eigenvalue of the covariance matrix as it regards the flat surfaces equals zero, then the value of the surface variation, σ_k, for the flat surfaces is zero.

2.2 Segmentation of Volumes to be Processed

Having determined which of the points represent edges, the next step is the clustering of the volumes that need to be processed along the z-axis. This enables to identify the geometrical changes of the part along the z-axis and then fix the number of machining/additive operations that need to be performed, in order to process the part as intended. This clustering operation is performed through unsupervised machine learning, using the DBSCAN algorithm that has been selected among the connectivity-based, centroid-based and grid-based options due to the high density of the point that represent edges and different geometries of the part. DBSCAN is a density-based, non-parametric algorithm. A DBSCAN cluster incorporates two very important properties. At first all the points that are included on the cluster are mutually density-connected and the second mentions that if a point is density-reachable from some point of the cluster, it is part of the cluster as well. When the clusters are identified, it is possible to group geometries which do not change in the XY plane, as we move along the z-axis. The clustering on ZY plane follows the edge extraction of the part.

The clustering of the geometries on the XY plane takes place after the end of clustering on ZY plane. This is a crucial operation that needs to be performed, in order to identify islands (e.g., pockets) that exist in the geometry. With this knowledge, it is possible to program the toolpath appropriately and include z-hops among the different clusters, in order to create collision-free and gouge-free toolpaths. For this purpose, several different clustering approaches, both unsupervised (e.g., spectral clustering, k-means) and supervised (e.g. SVM, KNN). K-nearest-neighbours (KNN) has proven to be the most effective algorithm for the specific application, enabling significant clustering performance. However, although the significant positive aspects, KNN has a major drawback, which is the fact that it is a supervised clustering algorithm, therefore it requires a training dataset. It is necessary to find a method to tackle this drawback, otherwise the path planning tool would be unusable for a potential end-user. The DBSCAN algorithm did not perform very well in the clustering of the point cloud data on the XY plane; however, it performed very well for the clustering of the original CAD on the XY plane. So, an innovative approach was followed to address the issue of the training dataset and automate the entire process:

- The point cloud of the scanned part, as well as the digitized point cloud of the original model are sliced along the z-axis according to the layer height (for AM operations) or the depth of cut (for subtractive operations)
- For each slice, DBSCAN is used on the digitized point cloud of the original CAD model to cluster the geometries (e.g., identify islands). These clusters are labelled sequentially from the innermost to the outermost cluster and exported. This is the training dataset for KNN.
- KNN is employed on the corresponding slice of the point cloud of the scanned part, to cluster it in the XY plane. Through this process, it is possible to effectively cluster the scanned part, even for very complex or sparse geometries

2.3 Tool Path Generation

The last step of the workflow is to generate the toolpaths for AM or machining for inner and outer contours. After the clusters of each layer of the point cloud have been identified, it is necessary to extract their inner and outer contours, which will act as the limits of the path, during the toolpath planning stage. This process constitutes a curve fitting problem. There are several techniques that can be employed, such as fitting parametric curves (e.g., B-Splines or NURBS), which give a good control on the curve fitting process and the handling of the curves afterwards. However, this requires the development of complex algorithms. On the other hand, when dense point clouds are handled (which is always the case in the context of 3D scanned parts) polylines can generate sufficiently satisfactory results in terms of curve fitting. To this end, alpha shapes are used to extract the inner and outer contours at each layer of the point cloud. Alpha shapes are families of piecewise linear curves that are associated with a set of points. After the contours of the different clusters are extracted, polyline offsetting is used to generate the offset curves accordingly. The offset curves are generated to compensate for the tool radius (subtractive operations) and the track width (AM operations). Finally, polyline offsetting is used again to generate the toolpath that covers the whole volume to be processed. The offset values are based on the radial engagement (subtractive operations) and the track width/overlap (AM operations). In the future, other toolpath patterns (one-way, zigzag, raster, etc.) will be also examined, especially for the case of AM. When machining operations are considered, there are some cases, where the programmed radial engagement is larger than the available material to be removed. Such cases need to be handled to prevent gouging, utilizing the digitized point cloud of the original model. Alpha shapes are used to generate the polylines that comprise its outer geometry and polyline offsetting is used to compensate for the tool radius. The machining pass is performed on the outer surface of the part, cutting only the small parts of excess material that exist. After the above steps have been completed, the toolpath for the whole part can be generated.

3 Case Study and Results

This case study focuses on the post processing with conventional machining of a part that has been made with AM process, particularly Material Extrusion (MEX-AM) process. The test piece was manufactured by Polylactic Acid (PLA) in a Sindoh 3DWOX1 printer, using process parameters for PLA that ensure good dimensional accuracy and surface quality, based on previous works of the authors on MEX-AM process development [13]. The test piece has been selected is the NAS 979 artefact from NIST [14] (also called the circle-diamond-square artefact) that is often used to assess the performance of AM processes (Fig. 2). Distinctive features can be detected aiming to test the methodology on several cases; holes of various sizes across the height, diamond-shaped geometry, and circle-shaped geometry etc.

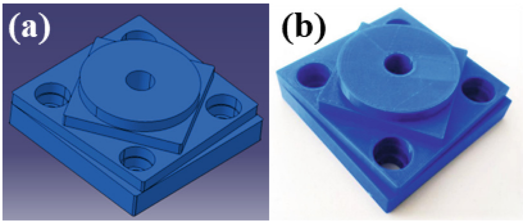

Fig. 2. (a) Test part design; (b) manufactured part

The nominal CAD model of the part has been digitized into, 1.000.012 points scattered in the 3D space (Fig. 3a) so as to provide a point cloud that enables the accurate and thorough inspection of the excess region, also providing enough points for the detection of disparity between the initial and the scanned model, while asking for minimum computational and processing power. On the other hand, the point cloud for the areas where the excess material is found contains 1.000.416 points (Fig. 3b).

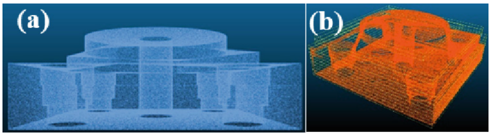

Fig. 3. (a) point cloud of the original part; (b) point cloud of the disparity map

The developed point cloud is imported to the algorithm that has been deployed in Python related to edge extraction using eigenvalue analysis. For the KNN algorithm and the clustering of edges, the threshold for the σ_k has been set to 0.03 while the k-nearest neighbors used for local normal vector calculation through PCA have been set to 40. Regarding the threshold for σ_k, a sensitivity analysis has been performed by modifying its value and identifying the edge extraction quality.

In general, the value of σ_k should be close to zero but selecting the absolute zero as a target value leads to numerical instabilities. Such instabilities are related to the extraction of vertical edges (i.e., parallel to the build direction) that would then compromise the clustering along the build axis in the next step, as well as the inclusion of outlying points. The results of such a sensitivity analysis are presented in Fig. 4.

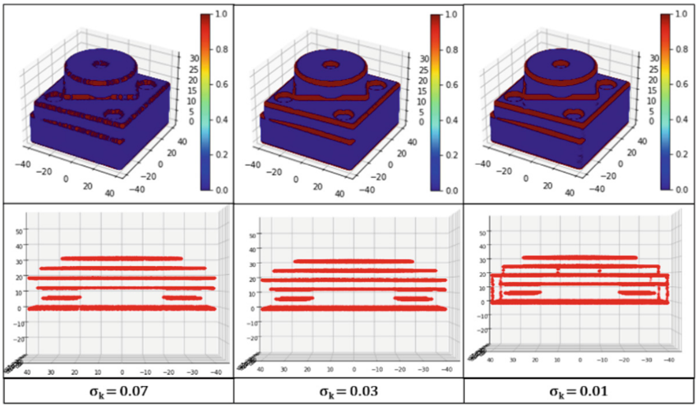

Fig. 4. Sensitivity analysis for σ_k

For the neighborhood size for the normal vector calculation, one needs to select such a value that would provide an accurate calculation, while maintaining a low computational cost. In general, for neighborhoods including more than 40 points, the benefits in calculation accuracy are negligible, so this has been the value used in the proposed algorithm [15]. Once the edges have been extracted, their x, y, z coordinates are grouped in a dataset. The next step is the implementation of DBSCAN algorithm on the edges that have been extracted from the original CAD. For the DBSCAN algorithm the maximum distance parameter has been set to 1 and the number of samples in a neighborhood for a point that is considered as a core point has been set to 5. In order to proceed to the next step, slicing of the point cloud that represents the disparity map has been performed so as to generate the layers that will be analyzed so as to obtain the corresponding paths. The slicing has been conducted with Python so as to match the slicing height with the extracted edges, using as slicing value the cutting depth of the conventional process, which has been selected to be equal to 2 mm, along the cutting direction, which in this case is the z axis. 15 layers have been generated based on the scattering of the points in the 3D space. Considering that at first the edge extraction and the classification takes place at ZY plane and then to XY plane, Fig. 5 and Fig. 6 are developed.

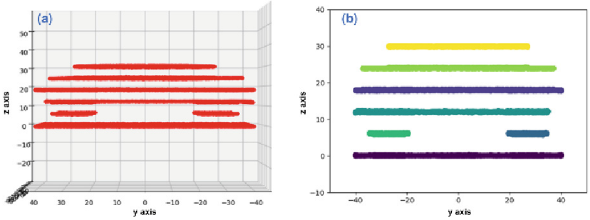

Fig. 5. (a) Edge extraction on the ZY plane; (b) DBSCAN clustering

After identifying the disparity map across the height comparing the initial CAD and the scanned model, the contour generation for each slice follows. The slicing value equals the axial depth of cut that has been chosen at 2 mm. The procedure for extracting the alpha shapes of each region is identical to the clustered internal and external geometries. The last phase of the proposed methodology focuses on the generation of the tool path. Each contour is examined with respect to the conditional offsetting principle. If the contour contains enough region with respect to the tool radius where internal offset polylines can be created, polygon offsetting is being performed. The offset value is defined equal to the tool radius. For each contour per layer inner offset paths are created until complete coverage of the layer surface. On the other hand, if the tool size exceeds the region that allows the generation of inner offset paths, the offset paths that have been generated using the original point cloud are selected. In total, 64 paths were created using the first condition and 8 paths using the second condition. This ratio is highly affected by the part geometry. During the final steps of the path generation a z hop value is defined, which represents the distance the tool has to raise above the part for the transition to different contours without collision with the existing bodies.

The z hop value has been selected to be 10 mm above the part height. A point is added at the end of each path and at the beginning of the next path so as to ensure the cutting tool motion above the part. By combining the different steps of the presented methodology, the final path is generated and illustrated in Fig. 7.

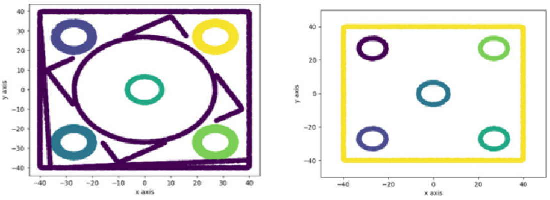

Fig. 6. Edge extraction on the XY plane for different Z levels

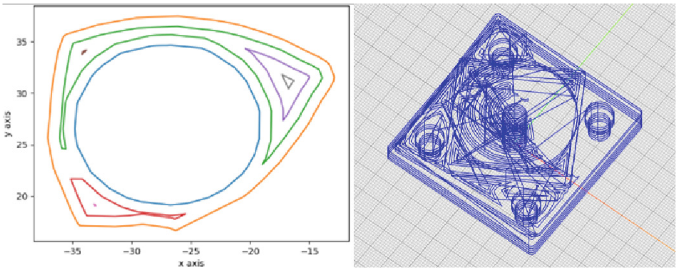

Fig. 7. Toolpath generated for finish machining of the manufactured part

4 Conclusions

This work proposes a methodology for adaptive toolpath planning directly from point cloud data in the context of hybrid manufacturing. Through the use of advance computational approaches and machine learning, it was possible to provide an algorithm of reduced complexity of implementation, based on discrete steps, contrary to existing approaches that lie solely on computational geometry, but are not easy to implement. Moreover, the minimized need of user input for the proposed algorithm provides an approach that reduces the skills barrier and can support the seamless integration of 3D scanning in the hybrid manufacturing workflow, thus leading to increased industrial acceptance of hybrid manufacturing technologies. As a future work, validation of the algorithms in a hybrid manufacturing scenario, using metal AM and milling will be pursued.

Acknowledgement. This work has been co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE - INNOVATE (project code: T2EDK-03896).

European Union
European Regional
Development Fund
EPAnEK 2014-2020
OPERATIONAL PROGRAMME
COMPETITIVENESS
ENTREPRENEURSHIP
INNOVATION
ΕΣΠΑ
2014-2020
Partnership
Agreement
2014 - 2020

References

1. Wohlers Associates Inc., Wohlers Report 2021: 3D Printing and Additive Manufacturing Global State of the Industry (2021)
2. Stavropoulos, P., Tzimanis, K., Souflas, T., et al.: Knowledge-based manufacturability assessment for optimization of additive manufacturing processes based on automated feature recognition from CAD models. Int. J. Adv. Manuf. Technol. **122**, 993–1007 (2022)
3. Souflas, T., Bikas, H., Ghassempouri, M., et al.: A comparative study of dry and cryogenic milling for directed energy deposited IN718 components: effect on process and part quality. Int. J. Advanced Manufacturing Technol. **119**, 745–758 (2022).
4. Stavropoulos, P., Bikas, H., Avram, O., Valente, A., Chryssolouris, G.: Hybrid subtractive–additive manufacturing processes for high value-added metal components. The Int. J. Advanced Manufacturing Technol. **111**(3–4), 645–655 (2020)
5. Masood, A., Siddiqui, R., Pinto, M.: Tool path generation, for complex surface machining. Using Point Cloud Data, Procedia CIRP **26**, 397–402 (2015)
6. Zou, Q., Zhao, J.: Iso-parametric toolpath planning for point clouds. Comput. Aided Des. **45**(11), 1459–1468 (2013)
7. Dhanda, M., Kukreja, A., Pande, S.S.: Region-based efficient computer numerical control machining using point cloud data. ASME. J. Comput. Inf. Science in Eng. **21**(4), 041005 (2021)
8. Chui, K.L., Chiu, W.K., Yu, K.M.: Direct 5-axis tool-path generation from point cloud input using 3D biarc fitting. Robotics and Comput.-Integrated Manufacturing **24**(2), 270–286 (2008)
9. Ghogare, S., Pande, S.S.: Efficient CNC tool path planning using point cloud. In: Proceedings of the ASME 2018 13th International Manufacturing Science and Engineering Conference. Volume 4: Processes. College Station, Texas, USA. June 18–22 (2018)
10. Popescu, D., Popister, F., Popescu, S.: Direct toolpath generation based on graph theory for milling roughing. Procedia CIRP **25**, 75–80 (2014)
11. CloudCompare[GPL software]. http://www.cloudcompare.org/. Accessed 21 Dec 2022
12. Bazazian, D., Casas J.R., Ruiz-Hidalgo J.: Fast and robust edge extraction in unorganized point clouds. International Conference on Digital Image Computing: Techniques and Applications (DICTA), pp. 1–8 (2015)
13. Stavropoulos, P., Papacharalampopoulos, A., Tzimanis, K.: Design and implementation of a digital twin platform for AM processes. Procedia CIRP **25**, 1722–1727 (2021)
14. Moylan, S., Slotwinski, J., Cooke, A.: An additive manufacturing test artifact. J. Res. Nat. Inst. Stand. Technol. **119**, 429–459 (2014)
15. Sanchez, J., Denis, F., Coeurjolly, D.: Robust normal vector estimation in 3D point clouds through iterative principal component analysis. ISPRS J. Photogramm. Remote. Sens. **163**, 18–35 (2020)