Reverse Engineering for Aeronautics: Study on Parts Semantic Segmentation

Philippe Williatte[1], Alexandre Durupt[1], Sebastien Remy[2] and Matthieu Bricogne[1]

[1]Université de Technologie de Compiègne
[2]Université de Technologie de Troyes

Corresponding author: Philippe Williatte, philippe.williatte@utc.fr

Abstract. Reverse Engineering (RE) is an activity which consists in digitizing a real part in order to create a numerical or virtual model of it. It is conducted on components that does not have any Computer Aided Design (CAD) model, or only a semantically poor 3D representation. Industrial applications for RE are: (A) manufactured parts inspection and dimensional control, (B) CAD model re-design and modifications; and (C) Product Data Management (PDM) system and Data-base search for 3D models and relative heterogeneous data retrieval. Aeronautical components present several challenges for RE activities, such as complex structures and shapes, large volume of data and a high need of precision in CAD models rebuilt. Moreover, development of local freeform shapes descriptors remains an area of research for applications such as complex surface labeling and retrieval in raw data, and for global mesh semantic segmentation (i.e., decomposition of the mesh into meaningful regions that can be associated with definition features).

This paper presents a study on semantic segmentation techniques for complex aeronautical components. Machine Learning and Deep-learning model-based methods for geometry processing will be evaluated on a set of real aircraft engine parts.

Keywords: Reverse Engineering, Segmentation, CAO, Shape Descriptors.
DOI: https://doi.org/10.14733/cadaps.2023.557-573

1 INTRODUCTION

Reverse Engineering (RE) is usually considered as a set of activities carried out with the intention of recreating a 3D numerical model from digitized data of a physical part [11]. It is conducted on components that do not have any Computer Aided Design (CAD) model, or only a semantically poor 3D representation, such as a mesh or a "freeze" resulting body. More generally, industrial applications for RE are: (A) manufactured parts inspection and dimensional control, (B) CAD model re-design for new product development or downward application as Computer Aided Manufacturing

or simulation; (C) Product Data Management (PDM) system and database search for 3D models and relative heterogeneous data retrieval.

Knowledge-Based Reverse Engineering (KBRE) methods are interested in coherent data (saved in databases) research and reuse during RE activities [7], and use the concept of data "signatures" as a means of information extraction and storage in a specific formalism (ie. Knowledge capitalization). Not restricted to 3D mesh remodeling, RE activities also include geometry processing and Product knowledge management in databases as PDM systems, making RE a reliable solution within a company's strategy for Product data integrity and sustainability.

The aeronautic industry context presents several challenges for RE activities, starting with complex product structures and freeform shapes with aerodynamic properties. When Reverse-Engineering digitized models, one must deal with large data volumes and a high need of precision in CAD model rebuilt. Moreover, the development of freeform geometric shape descriptor for complex shapes recognition remains an open question. These limitations have an impact on classical RE methods and tools when considering aeronautical Parts, especially for Part mesh semantic segmentation (i.e. mesh decomposition into labelled meaningful regions).

This paper presents a study on semantic segmentation for complex aeronautical Parts. Model-based segmentation methods relying on Machine Learning and Deep-Learning tools for geometry processing will be evaluated on a set of real aircraft engine parts. A capitalization (learning) phase on CAD definition models is used for computing performant geometric shape descriptors, which are saved within feature's signatures for reuse during the semantic segmentation of the mesh. Developed tools are compared both in terms of results accuracy and efficiency. Method enforcement consistency will be discussed considering technical feasibility as well as application genericity.

Section 2 presents a quick overview of state-of-the-art methods for RE applications in the scientific literature. In section 3, domain specific challenges are introduced and technical aspects are covered. Section 4 presents the study and the evaluation of each tested method.

2 STATE OF THE ART

2.1 General RE Process

Classical RE processes can generally be separated into four major steps: (a) Digitizing; (b) Pre-Processing; (c) Segmentation; (d) Modelling [7].

Digitizing a real object consists in creating a 3D virtual representation of it (usually a point cloud or a 3D mesh). This can be achieved through various methods (see [2] for an exhaustive review). It should be noted that, for some RE needs, 3D data come directly from tessellated CAD models and not from real world components.

Pre-Processing steps -like decimation of the number of acquired points, mesh generation, surface smoothing, etc.- are operations available in most acquisition and RE software. Mesh "simplification" is usually essential for subsequent steps. When considering sensible aeronautic parts, minimum preprocessing should be used to avoid over-modifying the acquired shape.

Segmentation is the process of sub-dividing a 3D mesh into distinct regions called segments. Based on geometry processing, segmentation quality is crucial, since it represents essential - and sometimes the only - information for the modeling step and influences the overall reconstruction process. Segmentation is called semantic when each identified region can be easily associated with a construction operation or a specific surface [7]. See [13], [18] and [23] for exhaustive surveys.

Modelling operations are used to generate 3D models compatible with CAD-CAM applications. *Freeform* approaches are commonly distinguished from *feature-based* methods. The former is an explicit modelling approach, which consist in describing a solid by its surfaces, and usually result in a "frozen" or "dead" solid. Parametric patches are fitted on the mesh by NURBS and b-Splines surfaces interpolation. With a fine tuning, freeform fitting allows for the reconstruction of complex CAD surfaces with high precision and details [3, 10]. Meanwhile, the term *feature* is widely used in

the literature to describe implicit modelling, which seeks to recover a parametric and semantic model (using a procedural approach with sequences of constructions operations with parenting relations). Main methods are primitives fitting [20], 2D sections [14], CAD or CSG template fitting [6] [21]. KBRE is a set of methods that consist in capitalizing heterogeneous information about the product during an upstream stage for reuse optimization. By using data "signatures", which saves knowledge in a specific formalism, the user is able to retrieve useful information and heterogeneous data during the RE application. PHENIX[1] [11] project developed a methodology for knowledge capitalization in the form of parametric geometric references that are adjusted to the 3D mesh. METIS[2] project [5] integrates the use of heterogeneous data (drawings, images, instructions, templates) to the RE process. Shape descriptors integrated in the signature are used to recognize and label specific features which are linked to support databases, providing knowledge that would not be available with a geometric analysis of the digitized model only. In this paper, *features* will be considered as generic shapes with which designers can associate certain attributes and knowledge useful for reasoning about the product [23].

The ability to retrieve knowledge on local features is closely linked with the segmentation activity, which extract image regions of features in the global mesh. Next sub-section is a review on mesh segmentation methods, with a focus on methods performing identification and labeling of local features.

2.2 3D Mesh Segmentation

The finality of the 3D mesh segmentation process is to cluster points (vertices) with similar characteristics into homogeneous meaningful regions [23]. Several state-of-the-art methods are based on shape analysis for geometric intrinsic information extraction (low order differential properties such as surface curvature and principle directions for example) [18], [22]. Edge-based methods detect points which have rapid changes in intensity to identify boundaries between segments. Region-based methods use neighborhood information to combine points in groups sharing similar properties, either by growing segments with neighboring points (bottom-up method), or by dividing large segments according to user driver criteria (top-down method). Attribute-based methods first compute specific attributes (distance, density, point distribution) to create group of points sharing common attributes. Neither of those technics could really be considered as sufficiently semantic, in the sense that segments computing is only based on geometric analysis of the mesh.

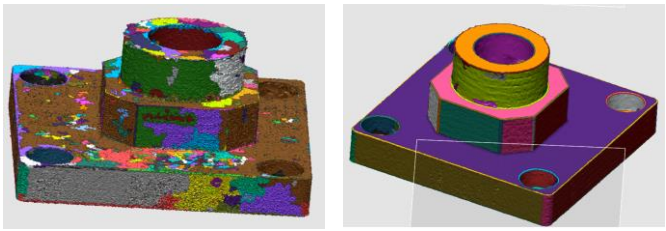
More precisely, semantic of models is all information in addition to its geometry. Local semantics are information on the individual parts which composed the model, as dimensions and driving parameters, functional specifications, modeling methods, general Product Manufacturing Information (PMI), and design intents. Segmentation is considered as semantic if computed segments can be associated with local features of the definition model, and their attributes. It is usually based on shapes analysis for general or capitalized knowledge extraction. Model-based methods use the analytical definition of geometric primitives (general canonical surface equation) to group points together as part of a quadrics. Methods usually detect the type of primitive, and then optimize equation's parameters with a least square fitting algorithm [1], [4], [20]. Because 85% of engineering CAD models can be represented by associations of primitive shapes [8], methods for the semantic segmentation of more complex surfaces have not been intensively studied in the literature. Methods considering "freeform" surfaces segmentation usually consist in predefined surfaces (mesh or b-rep) fitting to local regions of the data with a deviation minimization [22], [24].

[1] ANR PHENIX : Product History-Based Reverse Engineering, Programme Cosinus, 2008-2011, DeltaCAD, Université de Technologie de Troyes, Université de Technologie de Compiègne

[2] ANR METIS : ModElisation Tridimensionnelle de Maquettes Numériques Par l'Intégration de Données Géométriques et de Connaissances HétérogèneS, Programme Modèle Numérique 2012 - 2015, DeltaCAD, IFP Energies Nouvelles, École Centrale de Nantes, Université de Technologie de Troyes, Université de Technologie de Compiègne

According to [24], non-regular shape identification in raw data would make shape reuse possibilities wider, but freeform features should not depend on a predefined library.

Machine/deep learning technologies have been applied to geometric shape analysis for several applications such as global mesh labeling [9], [12], [26], primitives fitting improvement [16] or parametric curve and surface approximation [15]. As opposed to purely analytic model fitting and geometric reasoning, deeper methods extract feature descriptors during a supervised learning phase to perform shape classification, labeling and semantic segmentation [19]. IA based techniques are usually presented as outperforming geometric reasoning in term of completeness, noise robustness, and generalization [25].



a) b)

Figure 1: segmentation results with a commercial software: a) low-quality segmentation resulting in over-segmented mesh; b) valid quality semantic segmentation where segments can be labelled and associated with features.

The literature survey reveals potential methods limitations for aeronautical meshes segmentation: apart from being very sensitive to data noise and user-adjusted parameters, methods based on intrinsic geometric properties variations often result in unlabeled regions, and thus cannot isolate specific features-associated segments. Model-based methods are more prompt to identify and fit features to mesh data, but most are limited to geometric primitives with canonical equations, making them unsuitable with the shape typologies of our applications. Machine/deep learning methods constitute a good research direction to fill the gap, with more generalizable applications. However, technologies such as Artificial Neural Network are known to be computationally expensive, while shape descriptors should meet certain requirements in terms of simplicity, easiness of calculation and processing speed [17]. The following section presents the study on semantic segmentation methods and tools for aeronautic parts.

3 STUDY ON SEMANTIC SEGMENTATION FOR AERONAUTIC PARTS

3.1 Context and Challenges

Following study is part of researches carried out in the context of a PhD thesis in partnership with industrials from the aeronautic sector. RE application for aircraft engine parts is characterized by heavy meshes (up to 10M vertices) that are to be partially re-modelled with high precision. In this work, we will only consider the segmentation step of the overall RE process, considering that digitized mesh quality and conformity is ensured by the metrology department providing the digitized data. No preprocessing steps shall be applied, in order to keep the mesh the most representative of the physical Part. Modelling steps consists of b-rep surfaces fitting and/or deformable templates fitting to the mesh, which can be completed with sufficient results by commercial RE software [7], as Geomagic design X[3] or CATIA V5 RE workbench[4].

[3] https://fr.3dsystems.com/software/geomagic-design-x
[4] https://www.3ds.com/products-services/catia/products/v5/portfolio/domain/Shape_Design_Styling/

Figure 2 illustrates the topology of the example Part. It is composed of inner and outer frames supporting multi-instantiated blades, each decomposed in 6 features. When RE a 3D mesh of this product, regions corresponding to labelled specific features (label 0 to 6) need to be extracted and processed separately.

Because of the complex feature's shapes, most traditional methods like [20] are ineffective. As

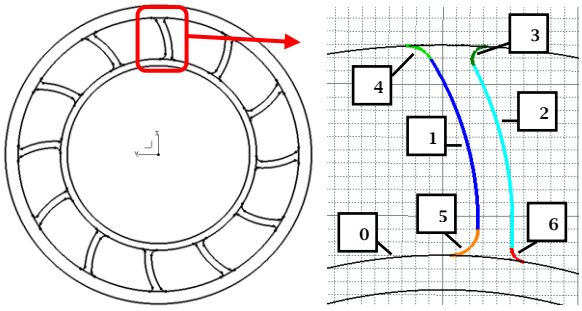
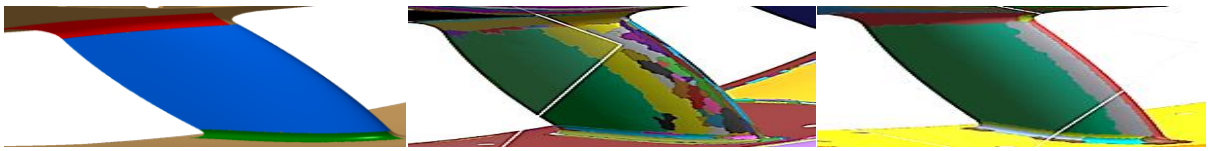


Figure 2: simplified sketch of an aircraft engine exhaust casing and its blade's topology.

an example, Figure 3 illustrates a blade segmentation results using Geomagic Design X: Figure 3.a) is the result we would like to obtain after a perfect semantic segmentation. Features 2, 3, 6 are identified, labelled, and segments are correctly delimited. Segmentation on the raw mesh (figure 2.b) results in a highly over-segmented mesh. After automatic preprocessing, results could be usable (figure 2.c), only requiring manual merging of segments. However, no preprocessing was allowed in our applications to maintain the accuracy of the mesh. Achieving better results with commercial tools would be possible by gradually adjusting segmentations parameters, which would require skilled user and long processing times. Therefore, methods for the semantic segmentation of aircraft engine parts (and more generally any part with complex shapes) still needs to be studied.

In the continuity of researches on KBRE carried out by research teams at the Universities of



a) b) c)

Figure 3: a) targeted segmentation result of a blade; b) low quality segmentation result (no preprocessing); c) correct quality segmentation result.

Technology Compiegne and Troyes, we make the assessment that capitalizing knowledge about proprietary components in an upstream stage allows for better knowledge management and reuse optimization. More precisely, complex shape signatures can be computed on definitions 3D CAD models for in-situ reuse within specialized semantic segmentation tool.

Three methods for semantic mesh segmentation will be tested and compared on real aircraft engine parts. First method is based on a point-to-point distance criteria between a scan and its 3D tessellated definition model. Then, model-based methods using Ordinary-Least-Squares nonlinear regression[5] and Gaussian Process regression[6] for complex surfaces approximation will be tested as features shape descriptors for automatic freeform surfaces recognition and labelling. The third studied method consists in using the neural network Pointnet++ [19] on a private dataset to evaluate Deep-learning capacities for semantic segmentations in industrial area.

[5] https://towardsdatascience.com/introduction-to-linear-regression-and-polynomial-regression-f8adc96f31cb
[6] https://towardsdatascience.com/gaussian-process-regression-from-first-principles-833f4aa5f842

3.2 Knowledge Capitalization on 3D Data and Segmentation Method

$Part_{def}$ refers to a definition CAD model, and S_{def_i} ($i \in [1,n]$) to n local surfaces (ie. B-rep features). $Part_{scan}$ is the digitized model (mesh) of one physical component. $Part_{mesh}$ and S_{mesh_i} refer to tessellated 3D models of $Part_{def}$ and S_{def_i}, that will be used either directly for segmentation, or for shape descriptors computation and/or learning. Finally, $segm_i$ are sub-meshes of $Part_{scan}$. If the semantic segmentation perform well, $segm_i$ should be the image of S_{def_i} in $Part_{scan}$.

Methods in this study are based on a capitalization (learning) phase that computes "features

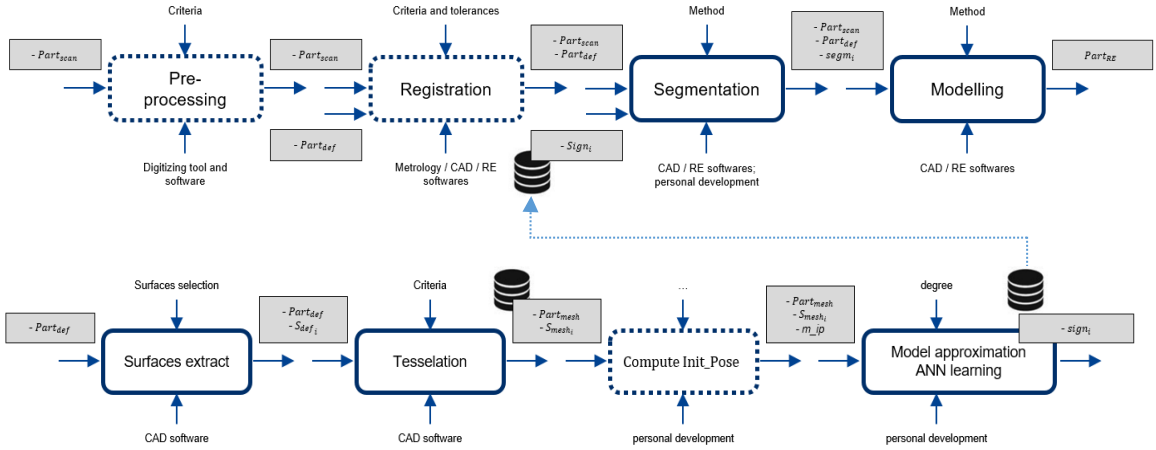


Figure 4: General RE process.

signatures", called $sign_i$. Those are composed of specific shape descriptors and associated metadata. Tools are developed in Python programming language. Basic operations on STL format meshes are made using Trimesh API[7].

Next sub-sections present methods technical aspects for knowledge capitalization and mesh segmentation. Experimentations, methods discussions and comparison will be studied in section 4.

3.2.1 Point-to-Point distance criteria

This method is based on the assumption that every point p_j of $Part_{scan}$ can be considered as part of $segm_i$ if close enough to S_{def_i}, or equivalently to S_{mesh_i}. In other words:

$$p_j(x_j, y_j, z_j) \in segm_i \quad if\ and\ only\ if \quad mindist(p_j, S_{mesh_i}) < \varepsilon \tag{1}$$

with $mindist(p_j, S_{mesh_i})$ being the minimum Euclidean distance between p_i and every point of the tessellated surface S_{mesh_i}, and ε a user-defined threshold. We denote that we use point-to-point distance rather than point-to-mesh faces (or even parametric surface) distance for computation efficiency. In here, no global segmentation is applied to $Part_{scan}$. The user must query a feature (S_{def_i}) to be recognized in the mesh, and submesh corresponding to that feature only will be extracted. Point-to-point distance criteria segmentation is done as follow:

(a) Capitalization: $Part_{mesh}$ is generated by tessellation of $Part_{def}$ using a CAD software. The user selects and tessellate local features of interest (S_{mesh_i}). Capitalization can be made in an upstream stage, or during the RE activity

[7] https://trimsh.org/

(b) $Part_{scan}$ is registered (align) with $Part_{mesh}$ (if not already done), using Iterative Closest Point algorithm.

(c) The user inputs S_{def_i} name (query) and the distance criteria threshold ε.

(d) $segm_i$ is generated (cf. equation (1)). Result appreciation is left to the user, threshold can be adjusted for better results.

(e) Previous steps (c) and (d) are done for each feature that are to be recovered in $Part_{scan}$.

3.2.2 Surface model approximation

- Polynomial implicit function of surface

This model-based segmentation technique is based on implicit model of surfaces fitting into the mesh vertices. Just like the well-known RANSAC method ([1], [20]) iteratively tries to fit primitives (quadrics) to regions of points in a Cartesian space, our method uses more complex surfaces models defined by high order polynomial implicit functions.

An implicit function f_d (with degree d) of a surface S_{def} is defined as follow:

$$S_{def} = \{p(x,y,z) \mid f_d(p) = 1\}, \quad f: \mathbb{R}^3 \to \mathbb{R}, \; f_d(x,y,z) = \sum_{d_x=1}^{d} \sum_{d_y=1}^{d} \sum_{d_z=1}^{d} \alpha_{d_x,d_y,d_z} \, x^{d_x} y^{d_y} z^{d_z} \quad (2.1)$$

with $(d_x + d_y + d_z \leq d)$. This mean, for any point p_j of art_{scan} , and for an admitted threshold ε_1:

$$p_j \in segm_i \quad \text{if and only if} \quad \left(f_{d,i}(p_j) - 1\right)^2 < \varepsilon_1 \quad (2.2)$$

$f_{d,i}$ parameters are approximated with Ordinary Least Square polynomial regression[8] on S_{mesh_i} vertices, and represent the core of $sign_i$. When approximated, $f_{d,i}$ is relative to $S_{def,i}$ Cartesian coordinate system. To fit this implicit function of surface (which we call the OLS model of $S_{def,i}$) to random data points, we need to optimize a rigid Euclidean transformation that minimize the squared sum of predictions errors. On a set of m points, our program tries to optimize parameters $(\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)$ that minimize:

$$cost_i = \sum_{j=0}^{m} \left(f_{d,i}(x_{t,j}, y_{t,j}, z_{t,j}) - 1\right)^2 \quad with \quad \begin{pmatrix} x_t \\ y_t \\ z_t \\ 1 \end{pmatrix} = M.p = \begin{pmatrix} \theta_{xx} & \theta_{xy} & \theta_{xz} & t_x \\ \theta_{yx} & \theta_{yy} & \theta_{yz} & t_y \\ \theta_{zx} & \theta_{zy} & \theta_{zz} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} . \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \quad (2.3)$$

M is the transformation matrix computed with parameters $(\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)$. Whether the minimized $cost_i$ is under an admitted threshold ε_2 or not, we can determine if randomly selected data points can be labeled as S_{def_i}. Neighboring points can iteratively be added to the segment if their transformed coordinates (with optimized M matrix) verify equation (2.2).

The method for $Part_{scan}$ semantic segmentation, once features signatures have been capitalized, proceeds as follow:

(a) A random region of the mesh is selected (automatically or by the user).

(b) Candidate surfaces models ($f_{d,i}$) are fitted in data points (cf. equation (2.3)).

(c) If under ε_2, best fitting cost gives the label of the region and the fitting matrix M.

(d) The segment is extended to each neighboring point that meet the surface model (eq. (2.2)).

Main goal of this method is to define a shape descriptor that is precise enough to differentiate geometrically close features, able to label local regions of features, and that is invariant to

[8] https://www.statsmodels.org/stable/generated/statsmodels.formula.api.ols.html

transformations (no need of preliminary registration of the scan, and able to recognize every multi-instantiated feature). This shape descriptor will then be used in a semi-automated semantic segmentation tool for complex aeronautics meshes.

- Gaussian Process Regression

Another way to approximate surfaces descriptor doesn't use a specific model as (2.1), but rather uses Gaussian Process Regression (GPR) to approximate a surface model. Gaussian process is a stochastic supervised learning tool that defines a distribution over a function. Advantages in using GPR instead of implicit polynomial functions is that they are less parametric and no parameters choices (like the degree in (2.1)) are needed.

As with OLS model, by approximating a GPR model[9] $gpr(x,y,z) = 1$ over S_{mesh_i}, a specific features signature $sign_i$ is capitalized and can be fitted to unknown regions of a mesh (equation (2.3)).

- Methodology for model-based methods evaluation

The methodology for testing and evaluating OLS and GPR models is as follow:

(a) Signatures capitalization: As in 3.2.1, $Part_{mesh}$ and features S_{mesh_i} are extracted and tessellated from the definition CAD model.

(b) Model analysis consists in surface models approximations and results analysis to determine if OLS and GPR models can be used as implicit representation for complex surfaces. For each S_{mesh_i}, regression models are evaluated with Mean Squared Error (MSE), Standard Deviation (SD) and the Range of squared predictions errors[10]. This step allows to tune specific parameters (as the degree of $f_{d,i}$, or regressions algorithm parameters)

(c) Model evaluation: Both models are intrinsically unbounded, and therefore could present a lack of precision to differentiate neighboring features points. To evaluate if models are restrictive enough, we compute $segm_i$ on $Part_{mesh}$ (with respect to 2.2). Comparison between $segm_i$ and S_{mesh_i} shows the model's segregations capabilities.

(d) Optimization algorithm choice: the fitting of a model in random data points depends on the optimization algorithm method that minimizes the $cost$ value (eq. 2.3). Several minimization methods[11] are evaluated and compared by fitting approximated models on randomly transformed S_{mesh_i}. The best algorithm in term of result and performance is then selected.

(e) Method evaluation: after proving the representativeness of the surface model, and the optimization program ability to fit OLS and GPR models in data points, it is time to evaluate the actual segmentation method. For each tessellated feature S_{mesh_i} of $Part_{def}$, a local region of S_{mesh_i} is randomly selected and transformed (M' being the rigid transformation matrix). Concurrent feature models are iteratively fitted to the data points, and the best $cost_i$ result determines the label and the optimized transform matrix M. If the optimized matrix M is the inverse of M', this means we can extend the segment to neighboring points.

(f) Method testing on real data: For now, method and tools were evaluated on tessellated data without noise and defect. For testing the effective usability of the solution, local regions of $Part_{scan}$ are manually selected, automatically labelled, and extended as a segment.

3.2.3 Deep-Learning

Nowadays, researches have shown the potential of deep-learning in 3D geometry processing [25]. Tools such as Artificial Neural Network (ANN) are used to learn deep shape descriptors and apply mesh classification and semantic segmentation without shape typology-related limitations. On 3D

[9] https://scikit-learn.org/stable/modules/gaussian_process.html
[10] https://towardsdatascience.com/ways-to-evaluate-regression-models-77a3ff45ba70
[11] https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html

point clouds, Pointnet++ [19] can achieve great results for simple point-clouds segmentation by recognizing local features and assigning labels to each point. However, to our knowledge, very few works have been applied for real industrial needs. Most studies on Deep learning uses public datasets as Shapenet[12], which doesn't fully represent complex shapes typologies of some industrials areas.

With this consideration, we tried to use Pointnet++[13] on a private dataset. First, we reproduced results of the research paper [19] on the public dataset Shapenet. Then, we developed a program for private dataset generation, processing as follow:

(a) Labels are assigned to every vertex of $Part_{mesh}$ (for example, label 1 if the vertex is part of S_{mesh_1}). Every others vertex (not part of any feature of the blades) were set to label 0. An original text file with the list of points is extracted as in Figure 5.

$$\begin{pmatrix} \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} & \boldsymbol{N_x} & \boldsymbol{N_y} & \boldsymbol{N_z} & \boldsymbol{label} \\ x_1 & y_1 & z_1 & n_{x_1} & n_{y_1} & n_{z_1} & 2 \\ x_2 & y_2 & z_2 & n_{x_2} & n_{y_2} & n_{z_2} & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ x_m & y_m & z_m & n_{x_m} & n_{y_m} & n_{z_m} & 1 \end{pmatrix}$$

Figure 5: data example

(b) Thousands of training files are generated by applying random rigid transform to data points, adding Gaussian noise to coordinates (+/- 2%), making random permutation between lines, and randomly sampling the data by removing lines.

(c) Pointnet++ Part segmentation ANN is trained on 80% of the dataset.

(d) Trained Pointnet++ is evaluated on the 20% remaining data (metrics are computed by labels): True positives is the percentage of points correctly labelled i. False positives is the percentage of points that have been incorrectly labelled i. Not Recognized is the percentage of points that should have been labelled i.

In the next section, results of the study are presented, and methods and tools are discussed.

4 RESULTS AND DISCUSSIONS

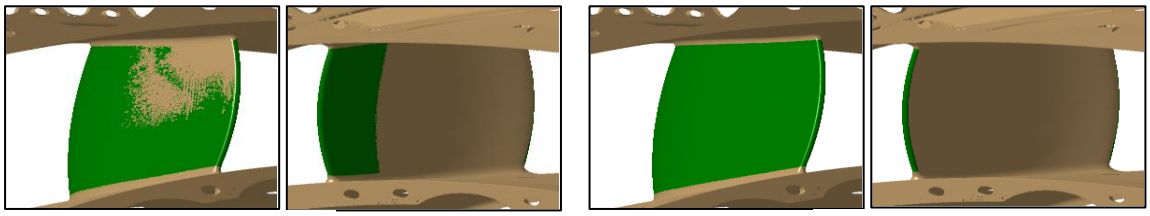
This section presents methods evaluations and segmentations results. The part presented figure 2 is taken here as an example. $Part_{scan}$ is composed of 8089420 vertices and 15923525 faces. The six features $\mathrm{S_{mesh_i}}$ (local surfaces composing the blades) are composed with respectively 9810, 10595, 1874, 1295, 4241 and 2467 vertices. Meshing density was empirically and arbitrary defined to approximately one vertex per squared millimeter. Computing works were made on a DELL Precision m7520 with an Intel Core i5-11500H, 2.9 GHz, 16 Go RAM. Neural Network training was carried out on a supercomputer using NVidia Ampère A100 GPU.

4.1 Point-to-point Distance Criteria

The method presented in 3.2.1 is applied to generate the six segments representing the features composing one blade of $Part_{scan}$. The user must query the feature and specify the distance threshold (iterations with adjusted threshold may produce better results). Each segment processing took an average computation time of less than 20 seconds.

[12] https://shapenet.org/
[13] https://github.com/yanx27/Pointnet_Pointnet2_pytorch

Contrary to model-based methods and Deep-Learning based methods, point-to-point distance criteria segmentation method does not need other capitalization step than local features tessellation. It has the advantage of being simple, fast, and not restrictive to any king of shape. CAD definition model must be available at the time of the RE activity and a registration step is necessary to align $Part_{scan}$ with $Part_{def}$, which can be computationally expensive (see [21] for works on CAD model fitting to data points). Moreover, registration algorithm as the Iterative Closest Point (ICP) can lead to quite important deviation at a local scale. For example, registration took approximately 1 hour in our example (with Geomagic Design X), and led to a mean deviation of 1.7 mm with maximum deviation up to 2mm for some blades surfaces. High local deviations imply increasing the value of ε during the segmentation, which can cause larger segments overlapping each other.



a) b)

Figure 6: $\boldsymbol{segm_2}$ results on $\boldsymbol{Part_{scan}}$ with point-to-point distance criteria method (segment in green).

As an example, figure 6.a) shows $segm_2$ overlapping on what should be $segm_1$, while still missing to capture some vertices of the mesh. For improvement, ICP algorithm has been implemented for local registration of $\mathrm{S_{mesh_i}}$ with $Part_{scan}$, which greatly improves the results (figure 6.b).

During the capitalization activity, every S_{def_i} must be "manually" extracted and tessellated, which makes the solution intrinsically non-automatic. Indeed, for components with multi-instantiated features, this method does not allow the complete semantic segmentation of the scan by just capitalizing on one instance of S_{def_i}. Either a large quantity of features must be stored in mesh format for later use (which would result in a large database), or straightforward segmentations (unitary feature tessellation and mesh segmentation at the time of the RE activity) can be applied for each instance. Using meshed definition features for mesh segmentation is not efficient for long term archival and reuse, either for RE activities automation. However, we consider this method as a great choice for unitary RE applications, as it gives good results in short time, but requires high user assistance. After computing segments, freeform patch fitting results in high detailed CAD surfaces which can be used in downwards applications.

4.2 Surface Model Approximation

This section presents the complete study for surface models approximation using OLS and GPR models. Followed methodology is presented in 3.2.2.3.

4.2.1 Model analysis

Models are approximated on S_{mesh_i} vertices. Analysis is performed using MSE, SD and Range (minimum and maximum value) of squared errors of predictions. $OLS\_fit()$ and $GPR\_fit()$ refers to the regression programs.

Feature	$OLS\_fit()$: *best degree*	$OLS\_fit()$: *MSE*	$OLS\_fit()$: *regression time*	$OLS\_fit()$: *prediction time*	$GPR\_fit()$: *MSE*	$GPR\_fit()$: *regression time*	$GPR\_fit()$: *prediction time*
1	6	4,33E-09	3.390	0.019	4,48E-18	28.041	28.431
2	6	1,04E-09	13.450	0.019	1,85E-18	38.071	35.642
3	7	9,02E-08	43.602	0.005	8,71E-21	0.305	0.226
4	7	1,06E-08	18.865	0.005	1,37E-20	0.182	0.117
5	7	8,56E-07	15.323	0.015	2,52E-15	2.485	2.261
6	7	9,46E-07	5.147	0.008	1,49E-20	0.674	0.524

***Table 1**:* Models approximations analysis on S_{mesh_i} features.

Both models can approximate studied surfaces with a high precision. GPR models are better representations, but are computationally more expensive in predictions than OLS models (Gaussian Processes are known to lose efficiency in high dimensional spaces). If computation time for approximation is not crucial (because capitalized in upper stage), prediction time must be the lowest possible to be usable in our method. GPR model prediction time depends on the number of training data (here, vertices) on which it has been approximated. Figure 8 shows that sampling training data before GPR model approximation makes the prediction time (on the original point set) faster, but increases the MSE. Therefore, sampling is not a valid option for reducing computation time.

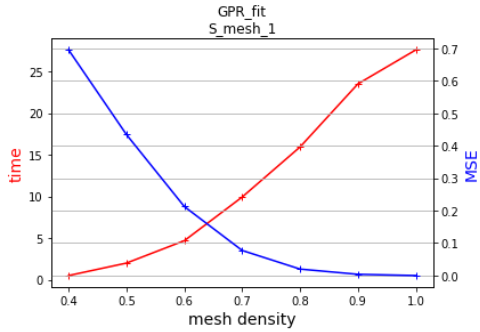


Figure 7: S_{mesh_1} sampling impact on GPR model prediction time (red) and MSE (blue).

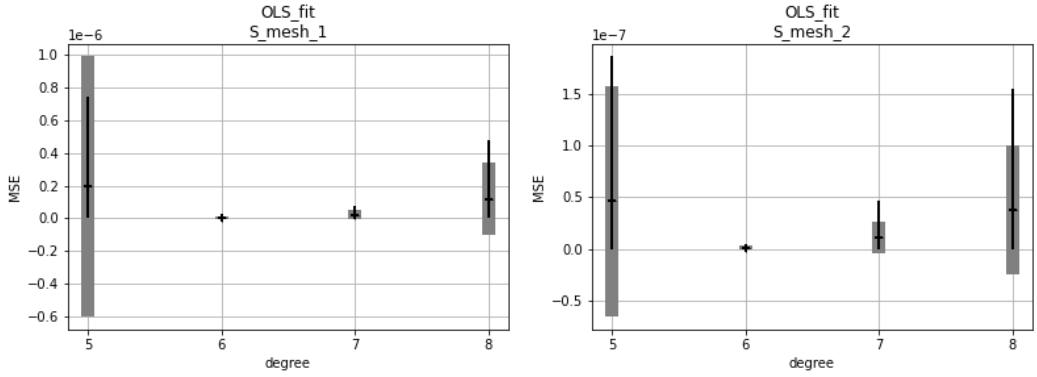


Figure 8: OLS models approximations analysis on S_{mesh_1} and S_{mesh_2}.

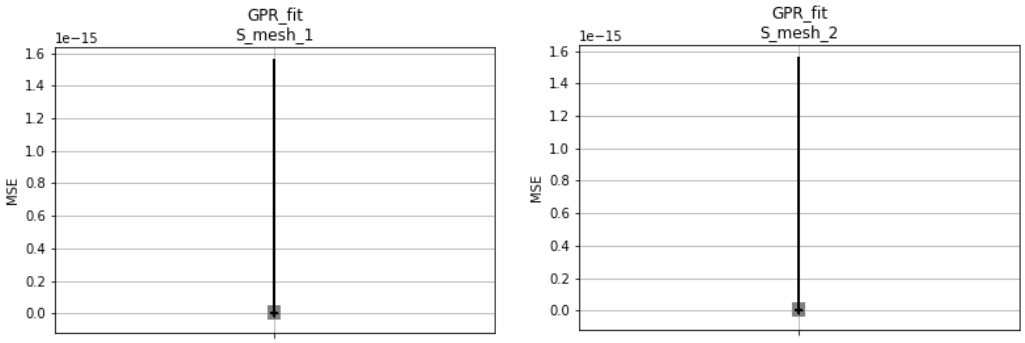


Figure 7: GPR models approximations analysis on S_{mesh_1} and S_{mesh_2}.

4.2.2 Model evaluation:

$Part_{mesh}$ segmentations results show the segregation capabilities of the models: OLS models have high rates of false positive. For example, those represent up to 20% for $segm_4$ and $segm_5$. On the other hand, GPR models have very low false positive rates, underlying their high segregation capabilities. Model evaluations proved that they could be used as shaped descriptors for specific aeronautic features. Next stage consist in testing if those models can identify and precisely label features in unknown data.

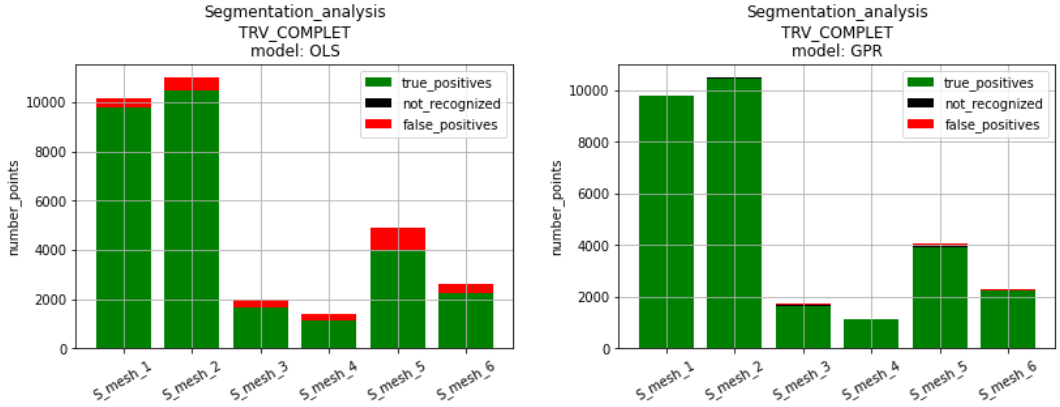


Figure 8: $Part_{mesh}$ segmentation results with OLS and GPR models.

4.2.3 Optimization algorithm choice

Because of the nature of the equation, minimization method must be gradient free or approximate it numerically. We first choose to compare "Sequential Least Squares Programming" (SLSQP)[14]and Powell[15] algorithm for model fitting on 50 randomly transformed S_{mesh_i}. (see Table 2).

Either with OLS or GPR model, Powell algorithm gives a 100% rate of optimization success on every try. SLSQP algorithm gives some wrong optimization results, meaning that S_{mesh_i} would not be correctly labelled. However, processing time of the SLSQP optimization algorithm is much faster, making it the most suitable choice for this method. If GPR models fitting gives slightly better results than OLS ones, processing time makes its use unsuitable for the application.

14 https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.fmin_slsqp.html
15 https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.fmin_powell.html

Feature	*Model*	*Method*	*Result*	*Average time*
1	OLS	POWELL	100%	30
		SLSQP	90%	12
2	OLS	POWELL	100%	32
		SLSQP	95%	13
3	OLS	POWELL	100%	63
		SLSQP	70%	4
4	GPR	POWELL	100%	1327
		SLSQP	80%	448
5	GPR	POWELL	100%	1870
		SLSQP	85%	757
6	GPR	POWELL	100%	618
		SLSQP	100%	265

Table 2: Model fitting on randomly transformed features.

4.2.4 Method evaluation

Random regions of S_{mesh_i} (of radius 5 to 100) are generated and transformed. OLS models of S_{mesh_1} to S_{mesh_6} are fitted to the data. Labeling is considered as a success if the minimum $cost_i$ correspond to the fitting score of the right surface model, and if the optimized matrix M is the inverse of M' (otherwise, the region could be correctly labeled, but extending the segment to neighboring points is not possible). Figure 10 shows the success rate for 50 tests with each surface and radius.

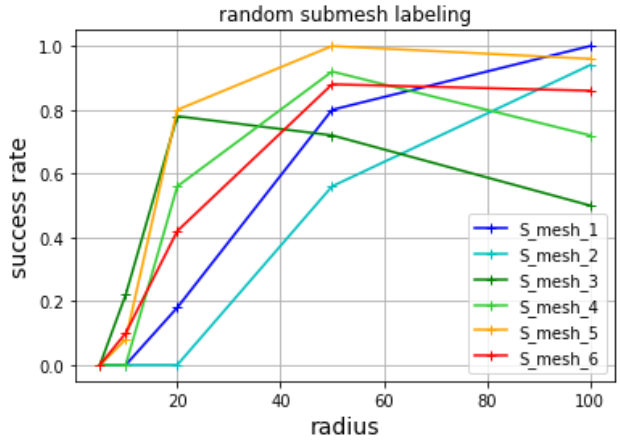


Figure 9: S_{mesh_i} local regions labeling with OLS model fitting method.

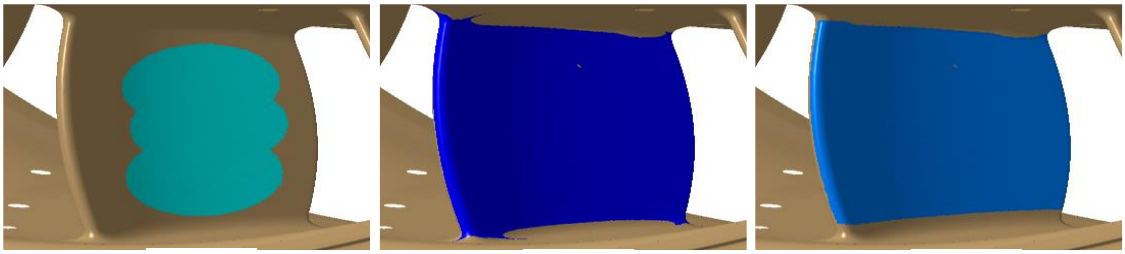
As we can see, small regions (radius under 50mm) are wrongly labelled by model fitting. Because the function we want to minimize is highly non-convex, the optimization algorithm converges to a local minima, producing a badly optimized transformation matrix M. Labeling results are much better when random regions cover bigger surface area of S_{mesh_i} (radius > 50mm). Labeling of features with labels 1, 5 and 2 have success rates of more than 90%. S_{mesh_3} and S_{mesh_4} have the worst results with respectively 46% and 76% of success on the 50 tries. We believe that a more extensive testing and tuning of the optimization algorithm would help achieve better results.

4.2.5 Method testing on real data:

Method evaluation shows quite good results on tessellated data. We tested the method on a real $Part_{scan}$. First, a skilled user manually selects a region of points in the mesh (on any blade of the

model). As for method evaluation, label and optimized transformation matrix is automatically given by the best fitted model. The transform is applied to $Part_{scan}$ which is then segmented.

No quantitative evaluation can be given in here. Most model fitting succeeds to find the right label when the user inputs a large region. In most tests, the segmentation threshold must be iteratively increased to compensate for noise in the data, which ultimately results in large segments overlapping on neighbor features with OLS segmentation. However, while GPR model fitting cannot be used for labeling (because of fitting time), we can use segmentation with GPR model once OLS model fitting succeeds to optimize the mesh transformation. Figure 11.a) shows the local region manually selected. This region is automatically labelled as S_{def_1} by OLS models fitting. Figure 11.b) is the segmentation result by growing the segment to neighboring points. This segment overlaps on points that belongs to other features. Figure 11.c) shows the computed segment with GPR model. For both segmentation steps, iterations may be needed with user assistance for threshold



a) b) c)

Figure 10: automatic labeling and segmentation of S_{mesh_1} in real scan mesh.

Capitalization	$\boldsymbol{Part_{scan}}$ *region selection*	*Labeling (OLS models fitting)*	*Segmentation (OLS model)*	*Segmentation (GPR model)*
manual	manual	automatic	automatic	automatic
~350s	60s	11.5s	22s / iteration	320s / iteration

adjustment.

Table 3: Computation times for S_{mesh_1} segmentation with surfaces models approximation method.

This method proved to be valid for a semantic segmentation of aeronautical Parts meshes, but still requires improvement for more reliability. If labeling is a fully automated process, user assistance is needed during the initial region selection, and iterations with adjusted threshold are necessary during the segment generation.

Some features are wrongly labelled when considering noisy data, or if the originally selected region is too small. However, the method succeeds to identify, recognize, and segment local complex features in noisy and heavy meshes. Finally, surface models are easy to store in knowledge base within features signatures for KBRE applications

4.3 Deep Learning with Pointnet ++

The last studied method for semantic segmentation of aeronautical meshes is the use of the Artificial Neural Network Pointnet++. Among latest methods presented in the scientific literatures, the use of AI to replace classical geometry processing methods is very promising. ANN presents great generalization capacities, as they are not restricted to any typologies of shapes. We chose Pointnet++ over others tools because it used Point Clouds as 3D data representations and can achieve good results. However, researches on ANN for shapes classifications and/or segmentations mostly consider public dataset composed of "light" 3D models, while our applications deal with a minimum of 5M points. This section presents the first results of our study, which should be

considered as a first approach in the use of ANN on heavy industrial dataset rather than a complete extended evaluation. Because using Pointnet ++ on large datasets requires extremely large computational capacities, training is performed on a supercomputer. First, Pointnet++ was trained on the dataset Shapenet and results were very similar with [19]. On this basis, we can evaluate this ANN on a private dataset.

Used dataset (cf. 3.2.3 for dataset generation description) is generated from $Part_{mesh}$ and is composed of 3000 files with shape (910000, 7). Each blade of the model is decomposed in 4 features (surfaces with label 3 and 4 were grouped under label 3, and surfaces 5 and 6 were set to label 4). Every point belonging to other features of the model were set to label 0. We stopped the training after 36 hours, and 6 epochs.

epoch	*Train accuracy*	*mIOU*
1	0.85326	0.16683
2	0.88324	0.166834
3	0.89263	0.164247
4	0.89735	0.166835
5	0.90013	0.166844
6	0.90366	0.166835

Table 4: Pointnet++ training and testing result.

Part segmentations results on a test data with trained pointnet++ took 5 minutes and is illustrated with figure 12. For labels 1 to 4, true positives rates are respectively 5.8%, 6.03%, 3.47% and 0%. 83.4% of point which have been labelled 0 (brown points) are true positives. Segmentations results of the blades features is still poor, but most false positive are located in the neighborhood of the actual features.

Several limitations of our experimentation method must be considered: first, the limited number of epochs; then the use of Pointnet++ witout any parameters change (as mlp tuning, or labels weights); and finally, the fact that training data are highly unbalanced (label 0 represents approximately 83% of each model). To conclude, this first experiment of using the ANN Pointnet++ for aeronautical components meshes segmentation shows promising results, as it can approximately detect the presence of specific features with complex shapes in the right area. Moreover, as previous method, the method is robust to Euclidean transformations and works well for multi-instantiated features. Further experimentations and Neural Network parametrization will hopefully give a better accuracy. Nevertheless, the use of such technology for industrial needs is questionable, as it requires great computations capacities and time.

CONCLUSIONS

In this article, we presented a study on semantic segmentation for Reverse engineering activities on aeronautical components. We compared different technologies for subdividing 3D meshes into semantically meaningful sub-meshes that represent features of the definition CAD model. Our major proposition is the use of surface models using high degree analytic functions approximation and Gaussian process regression to extract and recover geometric information on complex shapes. By capitalizing surfaces models on definitions CAD 3D models, those can be later used as shapes descriptors for sub-mesh labeling and global mesh segmentations, allowing semi-automatic processing of complex aeronautic surfaces in heavy digitized data. Experimentations on real scan data demonstrate the ability of surface models to recognize and label local regions of the mesh, and to extract sub-meshes corresponding to local complex features. However, this method remains prone to errors when used to identify small features, or features with similar shapes.

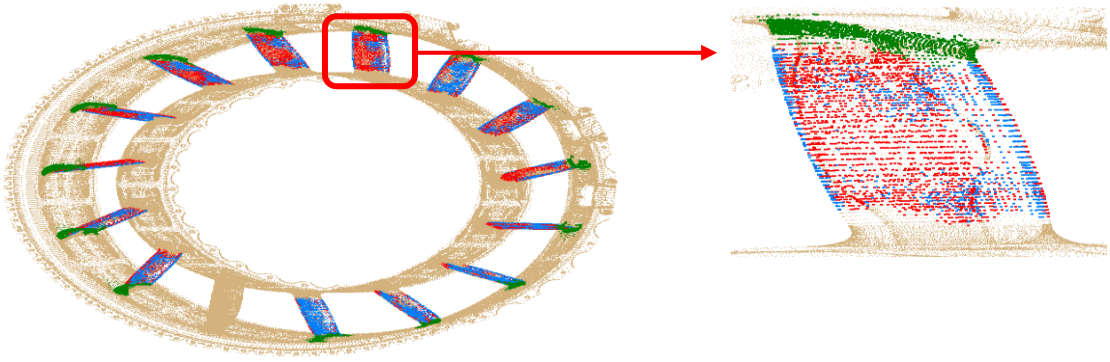


Figure 11: Part segmentation results with Pointnet++.

Although incomplete, we introduced a study on Deep learning applications in industrial context by testing a state-of-the-art Artificial Neural Network called Pointnet++ for point clouds segmentation on a private dataset. Results are very promising, but real use of such technologies would require optimizations for faster computation.

REFERENCES

[1] Attene, M.; Falcidieno, B.; Spagnuolo, M.: Hierarchical mesh segmentation based on fitting primitives, Visual Computer, 22(3), 2006, 181–193. https://doi.org/10.1007/s00371-006-0375-x

[2] Barbero, B. R.; Ureta, E. S.: Comparative study of different digitization techniques and their accuracy, CAD Computer Aided Design, 43(2), 2011, 188–206. https://doi.org/10.1016/j.cad.2010.11.005

[3] Ben Makhlouf, A.; Louhichi, B.; Mahjoub, M. A.; Deneux, D.: Reconstruction of a CAD model from the deformed mesh using B-spline surfaces. International Journal of Computer Integrated Manufacturing, 32(7), 669–681. https://doi.org/10.1080/0951192X.2019.1599442

[4] Bénière, R.; Subsol, G.; Gesquière, G.; le Breton, F.; Puech, W.: A comprehensive process of reverse engineering from 3D meshes to CAD models, Computer-Aided Design, 45(11), 2013, 1382–1393. https://doi.org/10.1016/j.cad.2013.06.004

[5] Bruneau, M.; Durupt, A.; Roucoules, L.; Pernot, J.-P.; Benoît, E.: Towards new processes to reverse engineering digital mock-ups from a set of heterogeneous data, Proceedings of the Ingegraph- ADM- AIP Primeca Conference, 2015.

[6] Buonamici, F.; Carfagni, M.; Furferi, R.; Volpe, Y.; Governi, L.: Reverse engineering by CAD template fitting: study of a fast and robust template-fitting strategy, Engineering with Computers, 2020. https://doi.org/10.1007/s00366-020-00966-4

[7] Buonamici, F.; Carfagni, M.; Furferi, R.; Governi, L.; Lapini, A.; Volpe, Y.: Reverse engineering modeling methods and tools: a survey, Computer-Aided Design and Applications, 15(3), 2018, 443–464. https://doi.org/10.1080/16864360.2017.1397894

[8] Chivate, P. N.; Jablokow, A. G.: Solid-model generation from measured point data, Computer-Aided Design, 25(9), 1993, 587–600. https://doi.org/10.1016/0010-4485(93)90074-X

[9] Dekhtiar, J.; Durupt, A.; Bricogne, M.; Eynard, B.; Rowson, H.; Kiritsis, D.: Deep learning for big data applications in CAD and PLM – Research review, opportunities and case study, Computers in Industry, 100, 2018, 227–243. https://doi.org/10.1016/j.compind.2018.04.005

[10] Dimitrov, A.; Gu, R.; Golparvar-Fard, M.: Non-Uniform B-Spline Surface Fitting from Unordered 3D Point Clouds for As-Built Modeling, Computer-Aided Civil and Infrastructure Engineering, 31(7), 2016, 483–498. https://doi.org/10.1111/mice.12192
[11] Durupt, A.; Bricogne, M.; Remy, S.; Troussier, N.; Rowson, H.; Belkadi, F.: An extended framework for knowledge modelling and reuse in reverse engineering projects, Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 233(5), 2019, 1377–1389. https://doi.org/10.1177/0954405418789973
[12] Guo, K.; Zou, D.; Chen, X.: 3D Mesh Labeling via Deep Convolutional Neural Networks, ACM Transactions on Graphics, 35(1), 2015, 1-12. https://doi.org/10.1145/2835487
[13] Kaiser, A.; Ybanez Zepeda, J. A.; Boubekeur, T.: A Survey of Simple Geometric Primitives Detection Methods for Captured 3D Data, Computer Graphics Forum, 38(1), 2019, 167–196. https://doi.org/10.1111/cgf.13451
[14] Ke, Y.; Fan, S.; Zhu, W.; Li, A.; Liu, F.; Shi, X.: Feature-based reverse modeling strategies, Computer-Aided Design, 38(5), 2006, 485–506. https://doi.org/10.1016/j.cad.2005.12.002
[15] Laube, P.: Machine Learning Methods for Reverse Engineering of Defective Structured Surfaces, 2020. http://www.springer.com/series/16265
[16] Li, L.; Sung, M.; Dubrovina, A.; Yi, L., Guibas, L.: Supervised Fitting of Geometric Primitives to 3D Point Clouds, ACM SIGGRAPH 2011 Papers on - SIGGRAPH '11, 2019. https://doi.org/10.1145/1964921.1964947
[17] Mousa, M. H.: Matching 3D objects using principle curvatures descriptors, Proceedings of 2011 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, 2011, 447–452. https://doi.org/10.1109/PACRIM.2011.6032935
[18] Nguyen, A.; Le, B.: 3D point cloud segmentation: A survey, 6th IEEE Conference on Robotics, Automation and Mechatronics (RAM), 2013, 225–230. https://doi.org/10.1109/RAM.2013.6758588
[19] Qi, C. R.; Yi, L.; Su, H.; Guibas, L. J.: PointNet++: Deep hierarchical feature learning on point sets in a metric space, Advances in Neural Information Processing Systems, 2017, 5100-5109.
[20] Schnabel, R.; Wahl, R.; Klein, R.: Efficient RANSAC for Point-Cloud Shape Detection, Computer Graphics Forum, 26(2), 2007, 214–226. https://doi.org/10.1111/j.1467-8659.2007.01016.x
[21] Shah G. A., Polette A., Pernot J-P., Giannini F., Monti M., « Simulated annealing-based fitting of CAD models to point clouds of mechanical parts' assemblies », Engineering with Computers, vol. 37(4), pp. 2891-2909, 2021. https://doi.org/10.1007/s00366-020-00970-8
[22] Song, Y.; Vergeest, J. S. M.; Bronsvoort, W. F.: Fitting and Manipulating Freeform Shapes Using Templates, Journal of Computing and Information Science in Engineering, 5(2), 2005, 86–94. https://doi.org/10.1115/1.1875592
[23] Theologou, P.; Pratikakis, I.; Theoharis, T.: A comprehensive overview of methodologies and performance evaluation frameworks in 3D mesh segmentation, Computer Vision and Image Understanding, 135, 2015, 49–82. https://doi.org/10.1016/j.cviu.2014.12.008
[24] Vergeest, J. S. M.; Spanjaard, S.; Horva´th, I.; Jelier, J. J. O.: Fitting Freeform Shape Patterns to Scanned 3D Objects, Journal of Computing and Information Science in Engineering, 1(3), 2001, 218–224. https://doi.org/10.1115/1.1419197
[25] Xiao, Y.-P.; Lai, Y.-K.; Zhang, F.-L.; Li, C.; Gao, L.: A survey on deep geometry learning: From a representation perspective. Computational Visual Media, 6(2), 2020, 113-133. https://doi.org/10.1007/s41095-020-0174-8
[26] Zhiron, W.; Song, S.; Khosla, A.; Fisher Y.; Linguang, Z.; Xiaoo, T.; Xiao, J.: 3D ShapeNets: A deep representation for volumetric shapes. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, 1912-1920. https://doi.org/10.1109/CVPR.2015.7298801