Journal of Manufacturing Systems 58 (2021) 108–117

Contents lists available at ScienceDirect

Journal of Manufacturing Systems

journal homepage: www.elsevier.com/locate/jmansys

Technical Paper

A digital twin-based approach for the management of geometrical deviations during assembly processes

Jean-Loup Grégorio [a,b,*], Claire Lartigue [a], François Thiébaut [a], Régis Lebrun [b]

[a] LURPA, ENS Paris-Saclay, Université Paris-Saclay, 94235 Cachan, France
[b] Airbus Central R&T, 92130 Issy-les-Moulineaux, France

ARTICLE INFO

Keywords:
Digital twin
Assembly
Geometry assurance
3D acquisition

ABSTRACT

The recent transformation in the aeronautical industry gives new prospects in the field of product geometry assurance. These include, in particular the creation of sophisticated virtual models, or digital twins, which can reflect the as-built geometry of physical products and optimize the assembly operations consequently. One of the current obstacles to the implementation of such digital twins is linked to the difficult transition from a conceptual model to a usable virtual representation. In this article, we present the hybrid representation of a product which is capable of integrating the different states of the components at each step of the assembly process. We propose a method to update the virtual representation of already assembled components, in order to include the position and orientation deviations of their surfaces. The B-Rep model of each component is updated from data acquired during the assembly of the product. The various steps of this update, and its associated tools are discussed in the article. Based on the knowledge of the as-built component geometry, the geometry of the yet-to-be-assembled components is adapted so that the final product complies with the functional requirements. To this end, we also discuss a formalism to model the product's functional information and to translate it at a geometrical level thanks to an assembly skeleton.

1. Introduction

The increasing integration of data processing systems leads to new prospects in the organization of production systems [1]. These prospects include the possibility to optimize manufacturing and assembly operations in real time, using digital twins of the manufactured products. A promising application is the geometrical quality of assembled products [2]. The virtual model of the product is supplemented with inspection data in order to detect and correct non-compliances resulting from geometrical deviations. Geometrical deviations found at component level tend to propagate along the assembly process. Those deviations cause assembly issues and non-compliance with functional requirements, which further leads to an increase in production costs.

Nevertheless, a recent literature review [3,4] shows that the current developments regarding digital twins in the field of production are mainly conceptual. Thus, key enabling technologies and tools for the implementation of digital twins have become a major research topic [5]. One of the obstacles to this implementation is the transition to a usable virtual representation [6]. One of the major challenges lies in the capacity of this virtual representation to reflect the actual product while being a valid model for simulation and optimization of manufacturing and assembly operations.

In order to answer this problem, some authors [6] suggest to use skin model shapes [7] as virtual representation of digital twins. The skin model shapes, mainly used in the field of tolerancing, are discrete representations of a physical object which can simulate or integrate position, orientation or shape deviations of its surfaces [8]. Simulating the assembly of components thanks to skin model shapes is also possible [9]. The possibility to optimize manufacturing and assembly operations from the understanding of the components deviations (through observation or simulation) is currently limited to the field of tolerancing [10].

In this article we propose a hybrid representation of a product, which can be integrated to the digital twin approach in order to manage geometrical deviations during the assembly process (Fig. 1). Geometrical data about the physical product is transferred to the virtual product using 3D sensors during the observation phase. These data are used to update the as-built components so as to mirror the actual geometry of the product. Based on the knowledge of the as-built components' geometry, the assembly of remaining components can be simulated and optimized. The resulting information is then transferred back during the

* Corresponding author at: LURPA, ENS Paris-Saclay, Université Paris-Saclay, 94235 Cachan, France.
E-mail addresses: jean-loup.gregorio@ens-paris-saclay.fr (J.-L. Grégorio), claire.lartigue@ens-paris-saclay.fr (C. Lartigue).

https://doi.org/10.1016/j.jmsy.2020.04.020
Received 29 October 2019; Received in revised form 26 April 2020; Accepted 30 April 2020
Available online 19 May 2020

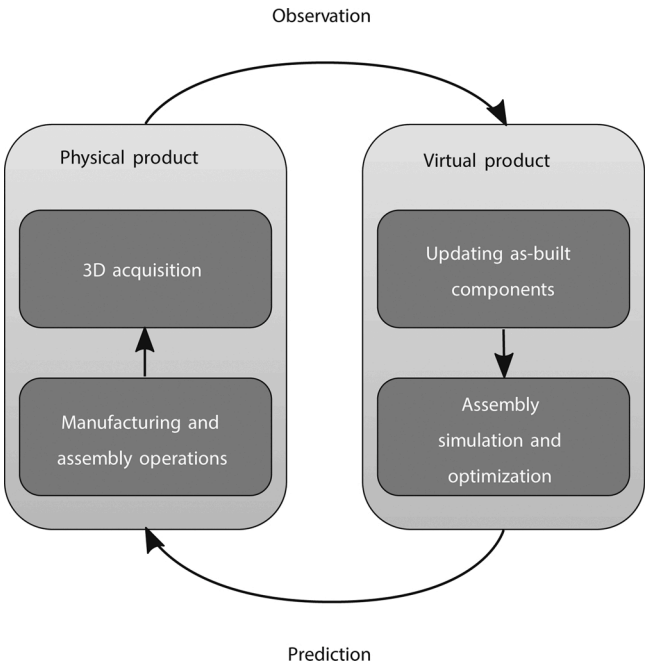

Fig. 1. Digital twin based approach for the management of geometrical deviations during assembly.

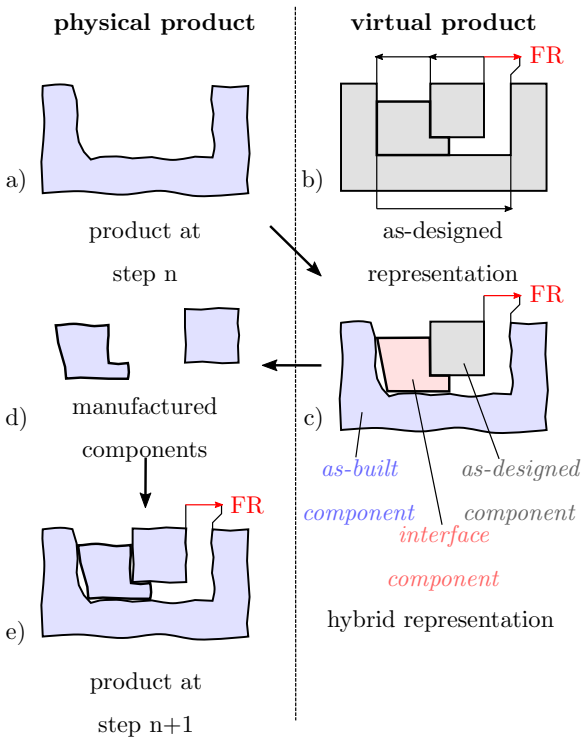

Fig. 2. The proposed hybrid virtual representation (c) is created from the initial as-designed representation of the product (DMU) (b) and the physical product (a) so that the extracted manufacturing and assembly operations (d) result in a final product which fulfills its functional requirements (FR) (e).

prediction stage so that the changes performed on the virtual product can be applied to the physical product.

While current research works are exploring the assembly of components with geometrical deviations, virtual products capable of integrating both as-built components and as-designed components have seldom been studied in the literature. One major difficulty is interfacing as-built components and as-designed components in the virtual product while building an assembly model capable of taking geometrical deviations into account. Once such an assembly model is built, a second difficulty is the optimization of upcoming manufacturing and assembly operations so that the functional requirements of the final product are eventually met. A third and more practical difficulty is finally preserving the continuity of the information linked to the virtual product's geometry during the update phase so that manufacturing and assembly operations can be carried out.

Let us consider the initial virtual product as a Digital Mock-up (DMU). DMUs are extensively used nowadays to support manufacturing and assembly operations. A DMU consists of the as-designed representation of a product, along with some data associated with manufacturing and assembly operations. These data often include geometrical tolerances and functional requirements (Fig. 2b) but also text instructions and machine trajectories. A DMU can be regarded as the static representation of a product. At the step *n* of the assembly process, geometrical deviations are observed on the physical product (Fig. 2a). One originality of the hybrid representation we propose is that the virtual representation of the product is updated according to the measured deviations of the physical product (Fig. 2c). This includes, on the one hand, as-built components reflecting the actual geometry of the physical product, and on the other hand, to-be-built components. These to-be-built components include as-designed and *interface* components. The notion of interface components is derived from the field of aeronautics [11] and designates custom-made components whose designs are updated in order to mitigate the effect of geometrical deviations on the functional requirements of the product (Fig. 2d, e). The choice of the components to be used as interface components is based on feasibility and cost criteria [12]. Other to-be-built components remain in their as-designed states and are assumed to be manufactured within their specified tolerance intervals (which may even be enlarged, as the as-built geometrical deviations of the components cannot be considered as random variables anymore). The initial virtual representation of the product, i.e. Digital Mock-up (Fig. 2b), therefore becomes a hybrid representation (Fig. 2c) composed of as-built, interface and as-designed components [13].

In Section 2 of this article, we describe the proposed hybrid representation for the use of digital twins in assembly processes. This description includes both a component representation and a structuro-functional model of the product. We demonstrate that the B-Rep representation, which is widely used to model as-designed component geometry, can be used to model the position, orientation and size deviation of as-built component surfaces as well as interface component geometry. In Section 3, we will explain how the observation stage (Fig. 1) is performed. During this stage, the virtual model is updated according to the measured data, in order to reflect the geometry of the physical product during assembly. In Section 4, we show that the as-built component can be used to simulate and optimize manufacturing and assembly operations through the update of the interface component. The prediction stage (Fig. 1) is performed as well.

2. Proposition of a virtual product for the digital twin

The proposed approach consists in updating of the original product's DMU to a hybrid virtual representation, comprising as-built, interface and as-designed components. This section explains how the geometry of such components is modeled and how they interact within the updated virtual product.

2.1. Modeling component's geometry

Manufactured products are mainly made of primitive geometrical shapes such as planes, cylinders, spheres, cones or tori. Such geometrical representation of the product can be found in its DMU using the B-Rep formalism. The virtual geometry of the components of the product can

be described as a set of faces delimitating its volume, these faces being described as parametric surfaces expressed in a Euclidean space and are bounded by edges. The intersections of the various surfaces create these edges. The B-Rep formalism is used nowadays in most CAD environments, either in native mode or through the standard for product data exchange (STEP or ISO 10303).

This parametric representation of surfaces is used to update the geometry of the virtual product. During the update, the initial designation of faces and edges is conserved to preserve the continuity of information. We use a direct modeling approach [14,15] in order to manipulate the geometry and to effect changes in the as-designed component models. Unlike the widely used history-based parametric modeling approach [16], direct modeling allows more freedom to perform local geometry changes. It is not necessary to identify specific feature parameters in the component's history tree in order to make a change to the geometry.

In the case of as-built components, the parameters of each surface are used to model position, orientation and size deviation of the faces, as shown in Table 1. This method is directly inspired from the work of [17], which exploit the similarity between vectorial tolerances [18] and the B-Rep description. During the update, the face of each component is modified so that the final component reflects its physical counterpart. To that end, surface parameters are first identified using a 3D geometrical acquisition of the physical product that is being assembled. The component B-Rep is then updated so that its surface parameters match the parameters identified from the physical product (Fig. 3). With this method, only a small number of parameters is necessary in order to model position, orientation and size deviation.

In the case of interface components, surface parameters are used in the same manner in order to effect geometrical changes, so that the functional requirements of the product are virtually met. To that end, surface parameters are determined according to the geometry of as-built components, and the structuro-functional model of the product.

The presented method is based on the following hypotheses:

- Surface topology is preserved: planar surfaces remain planar, cylindrical surfaces remain cylindrical, etc. In the case of as-built components, shape deviations are supposed to be negligible.
- Edges can still be computed: surfaces still intersect after their parameters have been updating. The underlying assumption is that geometrical changes/deviations are generally small compared to face dimensions.

Once surface parameters have been modified, to ensure that the resulting solid is perfectly closed, edges are computed again by intersecting the modified surface and surfaces coming from adjacent faces. Determining neighboring faces is quite straightforward, as the adjacency relations between faces do not change during the update. Edge computation is also eased by the fact that surface topology is preserved during the update and also that geometrical deviations are generally small compared to face dimensions.

The aforementioned steps are implemented in a prototype software, called D3MO for *Deviations 3D MOdeler*, which will be detailed in Section 5. While major commercial CAD software already provides advanced

Table 1
Parameters modeling each surface position, orientation and size changes/deviations.

Surface type	Parameters		
	Position	Orientation	Size
Plane	$\Delta x\ \Delta y\ \Delta z$	$\Delta nx\ \Delta ny\ \Delta nz$	–
Sphere	$\Delta x\ \Delta y\ \Delta z$	–	Δr
Cylinder	$\Delta x\ \Delta y\ \Delta z$	$\Delta nx\ \Delta ny\ \Delta nz$	Δr
Cone	$\Delta x\ \Delta y\ \Delta z$	$\Delta nx\ \Delta ny\ \Delta nz$	Δr
Torus	$\Delta x\ \Delta y\ \Delta z$	$\Delta nx\ \Delta ny\ \Delta nz$	$\Delta r1\ \Delta r2$

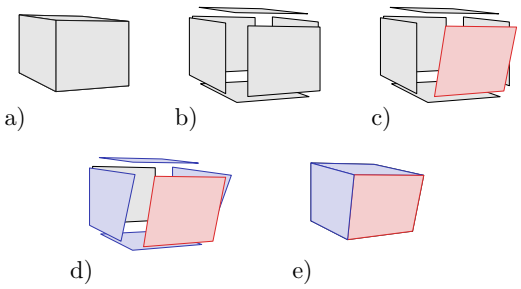

Fig. 3. (a) As-designed component, (b) face decomposition, (c) integration of the new parameters, (d) calculation of the new edges, (e) updated (as-built or interface) component.

direct modeling functionalities [19–21], we chose to develop a custom solution. The main reasons are that the D3MO software is specifically tailored to our needs and allows a seamless integration of the different algorithms developed as part of our work. The STEP neutral format is adopted in order to ensure data exchange between our prototype software and commercial CAD environments. Once updated, as-built and interface components are integrated to the virtual product together or with other as-designed components.

2.2. Structuro-functional model of the product

Geometrical deviations, which are present on as-built components, propagate during the assembly process as components are positioned with each other. Relative positioning between components is usually defined thanks to assembly joints. Each joint involves two surfaces, each one belonging to one of the components. The number, nature and realization order of these joints have an impact on geometrical deviation propagation during assembly and therefore on functional requirements.

Components, joints and their influence on the product's *key characteristics* (KC) may be represented using an Oriented Contact Graph [22]. A KC is one property of a product required to satisfy a function. KCs are generally expressed as geometrical condition between two surfaces. The graph proposed in Fig. 4 represents the structuro-functional model of the product.

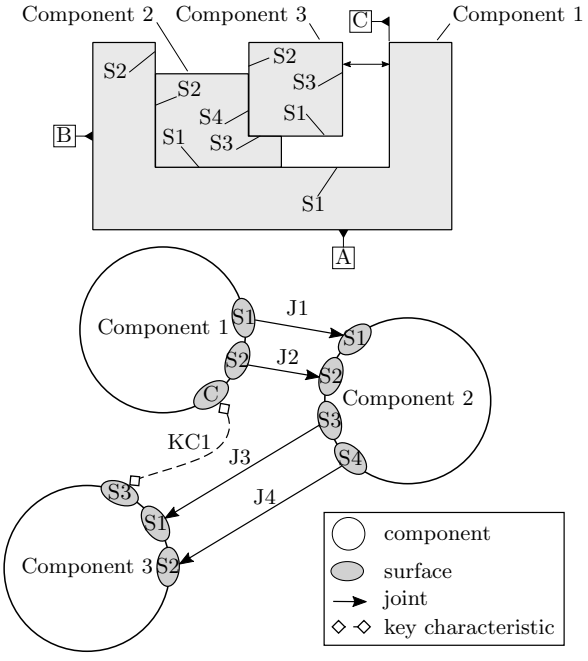

Fig. 4. Example of Oriented Contact Graph.

In this article, the Oriented Contact Graph is slightly modified in order to represent the impact of as-built component update on the product (Fig. 5). The notions of global and local references and specified elements are thus introduced.

Global references are a set of surfaces belonging to the original product's DMU thanks to which as-built components are positioned. The use of such global references allows us to express geometrical deviations using the initial reference geometry of the product. Global references constitute intermediary datum used to link the virtual product's datum to the acquisition datum. Local references are surfaces used in the composition of joints between as-built components and as-designed components. Due to geometrical deviations, the situation of local references regarding global references varies. In the same manner, the situation of specified elements also changes as the representation of components is updated from as-designed to as-built.

Using an Oriented Contact Graph allows one to represent structuro-functional information in a simple and structured manner. It also serves to list the Ji joints and KCi *key characteristics* of the product. Surfaces impacted by the as-built component update are also highlighted thanks to global and local references and specified elements.

In order to link joints and *key characteristics* to the product's geometrical view, a geometrical skeleton is then deduced from the previous graph. Geometrical skeletons have been found useful to include assembly knowledge in the product development process [23]. Geometrical skeletons are associated with a top-down design methodology where functional requirements are cascaded to component levels in order to define their geometry.

The geometrical skeleton proposed in this paper is composed of five types of features: point, line, plane, curve and datum. These features are used to define joints and functional requirements in the product's geometrical view.

First the skeleton datum is built using global references, as defined earlier. Skeleton features are expressed thanks to this datum. Elements corresponding to local references mainly define the assembly joints. The type of assembly joints must be defined according to geometrical deviations, so that the system is statically determined. For example, defining the relative positioning between two components with a set of planar joints is admissible when considering as-designed components but results in an over-constrained system when geometrical deviations are introduced in the virtual product (Fig. 6). The choice of the joint type is up to the product designer and is driven by the kinematical and technological views of the assembly. Skeleton features corresponding to each assembly joint, noted Ji, are then defined and instantiated (Table 2) Only joints involving surface pairs are considered. Skeleton features corresponding to specified elements, noted SEi, are built according to toleranced features. Toleranced features are defined by ISO 1101.

Finally, the product skeleton is then built (Fig. 6). Parameters associated to the skeleton's features are given in Table 3. Curve feature is not considered here and may be modeled using any parametric curve such as

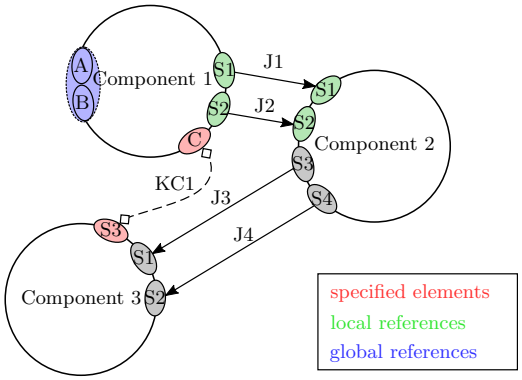

Fig. 5. Oriented Contact Graph modified.

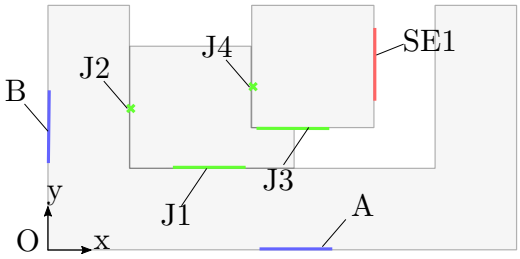

Fig. 6. Example of geometrical skeleton.

Table 2
Skeleton features definition based on joint type

Joint type	Feature	Degree of freedom	
		Translation	Rotation
Cylindrical	Line	1	1
Spherical	Point	0	3
Planar	Plane	2	1
Line-contact	Line	2	2
Curve-contact	Curve	1	3
Point-contact	Point	2	3

Table 3
Parameters modeling skeleton entities.

Feature	Parameters	
	Position	Orientation
Plane	$\Delta x \Delta y \Delta z$	$\Delta nx \Delta ny \Delta nz$
Line	$\Delta x \Delta y \Delta z$	$\Delta nx \Delta ny \Delta nz$
Point	$\Delta x \Delta y \Delta z$	–

B-Spline or NURBS. The initial as-designed representation of the product is first used to define the skeleton's feature parameters.

Parameters of each feature composing the skeleton are updated simultaneously as components composing the virtual product are themselves updated.

3. Updating as-built components

In the previous section, we defined a direct modeling approach which enables us to model the position and orientation deviations of the as-built components of our digital twin. Updating the virtual model of as-built components is done by updating the surface parameters of its B-Rep model. In this section, we define a method which enables us to relate these parameters to the geometry of the physical product under assembly.

While reconstructing the B-Rep model of a component from one of its physical instance is a widely addressed problem [24–26], only a few authors take preexisting data about the component into account during the process. The authors of [27,28] use an *a priori* CAD model as an initial estimate of the model to be reconstructed. Surfaces are adjusted to a 3D acquisition result via an *Iterative Closest Point* (ICP) [29,30] algorithm, which is likely to converge if the initial estimate is "close enough" to the 3D data. The authors of [31] propose to overcome this limitation by stating the reconstruction problem as a search for a model that maximizes probabilities taking several constraints into account, such as the relevance to the *a priori* CAD model.

The proposed method to update the as-built component from its as-designed representation consists of six steps:

(1) Acquisition of the as-built component geometry;

(2) Preprocessing of acquired data in order to produce an exploitable mesh;
(3) Registration of the mesh on the as-designed geometry;
(4) Classification of 3D data;
(5) Surface fitting using the classified data and extraction of the actual parameters;
(6) 3D modeling of the as-built components based on these actual parameters.

First, the as-built components' geometry is acquired using an optical sensor, resulting in a triangular mesh. In our work, the totality of the physical components' visible surfaces is acquired although more optimized strategies [32,33] may be used in order to leverage inspection time and reduce data volume. Then the preprocessing of the acquired data is performed. The acquired mesh is cleaned in order to remove ambiguous or unwanted samples that correspond to acquisition errors or belong to acquisition artifacts. Normal vectors and discrete curvatures are then computed for each vertex of the mesh. This piece of information is needed in the following steps.

The third step consists in registering the mesh on the as-designed component (whose geometry has been tessellated in order to perform this particular step). The goal of this third step is to achieve an approximate matching between the acquisition datum and the CAD datum. This registration is performed in two sub-steps, involving a first coarse registration based on a *Principal Component Analysis* (PCA) algorithm and a second fine registration based on an ICP algorithm.

The fourth step consists in classifying mesh vertices. We address the problem of retrieving the surfaces of the component from 3D acquisition data using a supervised classification framework (Fig. 7). Our work is mostly inspired by the work of [34], which focuses on the classification of point-clouds from urban scenes. In our case, we want to classify the acquired mesh so that a label y is attributed to each vertex based on the surface that it samples. Additionally to the vertex coordinates $v_i \in \mathbb{R}^3$, we use normal vectors $n_i \in \mathbb{R}^3$ and discrete curvatures $k_{1,i} \in \mathbb{R}$ and $k_{2,i} \in \mathbb{R}$ as features of our data vector X. Indeed, these features allow the description of the position, orientation and local variation of a 3D surface.

Before we can classify the acquired mesh, the training data has to be gathered in order to train a classifier which can then apply to new data. The training data may be gathered from the 3D acquisition of other manufactured instances of the same component or generated by simulation. While the first option guarantees the relevance of the gathered data it also requires a tedious phase of collecting and manually labeling the data. When historical data about the considered component cannot be collected, we choose the second option. We use the direct modeling approach described in Section 2 in order to generate virtual instances of the studied component and its geometrical deviations. First, random geometrical deviations are generated and applied to the as-designed representation of the component. Then the simulated as-built components are sampled in the form of a triangular mesh, which is similar to the acquired data. As the parameters of the simulated components' surfaces are known, the generated meshes can be automatically labeled. In practice, for each surface S_j, we select the vertices whose distances and angles are under a given threshold (Eq. (1)), which depends on mesh quality. An extra class is dedicated to the component's edges (i.e. vertices that are not associated to any surface).

$$V_j = \{v | v \in V \wedge d(v, \mathcal{S}_j) < \varepsilon \wedge \theta(v, \mathcal{S}_j) < \alpha\} \tag{1}$$

Finally, a random displacement is introduced for each vertex, in the direction of its normal vector, in order to simulate the noise induced by the acquisition using an optical sensor (Eq. (2)).

$$v_i^{'} = v_i + e_i.n_i \tag{2}$$

Finally, each mesh is registered on the as-designed geometry and normal vectors and discrete curvatures are computed. The simulated components are then used to train the classifier. We chose to use a *Random Forest* classifier, as it is a good compromise between computational speed and prediction performance [34].

Once the mesh is classified, the fifth step is to adjust surfaces to the data. To that end, we first make sure that the measurement datum perfectly matches the CAD datum, so that the parameters drawn from the actual surfaces are actually expressed in the assembly coordinate system. A final registration is performed, based on the global references of the assembly and on the corresponding subsets of the mesh. Surface parameters are then computed by least-square fitting of the primitive shape corresponding to each subset. As the least-square fitting result is very sensitive to the presence of outliers (i.e. wrongly classified data), statistically aberrant vertices are removed from the subsets. To do so, a non-parametric model, based on a *kernel smoothing* algorithm, is built from vertex residuals. Vertices with residuals corresponding to the highest percentiles of this distribution are then removed.

The final step consists in updating each surface parameters of the as-designed components by the previously computed values.

4. Updating interface components

The previous section detailed the possibility to update as-built components in order to reflect the geometrical deviations of the physical product during its assembly process. Based on the geometry of the as-built components, following assembly operations can be simulated and then optimized in order to ensure that the functional requirements of the final product are eventually met. To do so, the geometry of interface components is also updated.

Once as-built components have been updated, the situation of as-designed components in the virtual product is updated as well. For that, the parameters of the skeleton features, as defined in Section 2, are updated according to the as-built component geometry. This includes on the one hand local references and, on the other hand specified elements.

More precisely, the skeleton features which correspond to local references are updated by identifying or computing elements that result

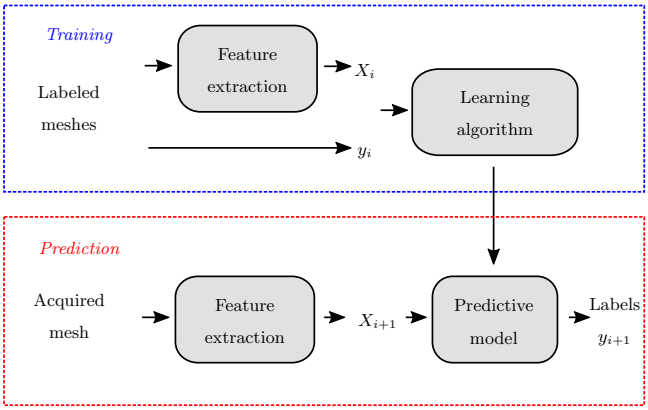

Fig. 7. Supervised classification framework.

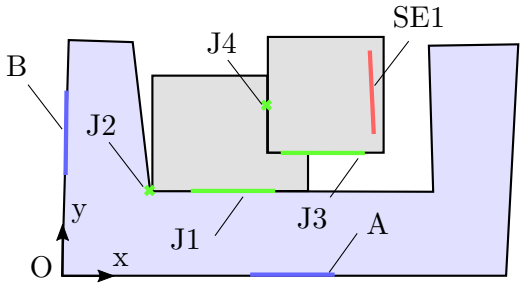

Fig. 8. The situation of as-designed components is updated.

from the contact between as-built and as-design components (Fig. 8). When these elements correspond to surfaces of the as-built components, such as *J*1, they can be directly identified. Otherwise, they have to be evaluated by computing the contact between the component's faces.

The skeleton features which correspond to specified elements are updated as well. Only specified elements that materialize *key characteristics* involving surfaces of the as-built components need to be updated. *Key characteristics* are identified using the Contact Oriented Graph. The new parameters of these specified elements are then deduced from the toleranced features.

After component positioning is updated by considering as-built component geometry, the interface component is updated so that the functional requirements of the final product are met. First, surfaces to update are identified. These surfaces correspond to the surfaces involved in the variation of the product KCs. Such surfaces can be easily identified in the Oriented Contact Graph.

Surface parameters of the interface component are then determined so that the functional requirements are met in the end of the assembly process. In some cases, and as in the example of Fig. 9, the problem of the surfaces update admits several solutions. Several combinations of surface update can lead to a compliant assembled product. The choice of the surfaces to update is up to the product designer. It involves expert knowledge, which should be translated into design rules within the virtual product.

In the example presented in Fig. 9, surfaces *S*2 and *S*3 of the interface component are updated.

After the geometry of the interface component is updated, the situation of the remaining as-designed components can be updated in the virtual product. Manufacturing and assembly operations are then performed according to the information contained in the updated virtual product.

5. Use case and results

The digital twin approach and the hybrid product representation discussed in the previous sections is illustrated through a simple case study.

5.1. Presentation

The considered product is composed of three components: one *base* component (yellow), one *intermediary* component (green) and one *axis* component (orange). The components are assembled in this order to form the final product. A partial representation of the functional requirements is shown in Fig. 10. The CAD environment used is CATIA V5® and the component geometry is described using STEP files.

An instance of the *base* component has been manufactured via 3D printing (Fig. 11a). The geometry of the manufactured component differs from the geometry of the virtual component. Surface position, orientation and size deviations have been introduced on purpose. The magnitude of these deviations (∼1 mm for position and size and ∼0.1 rad for orientation) is such that the assembly operations extracted from the initial DMU would eventually result in a non-compliant final product (Fig. 11b). From then, possible scenarios involve carrying unplanned rework operations on the product or even simply rejecting the *base* component. Both options result in dramatically increasing assembly costs and delay.

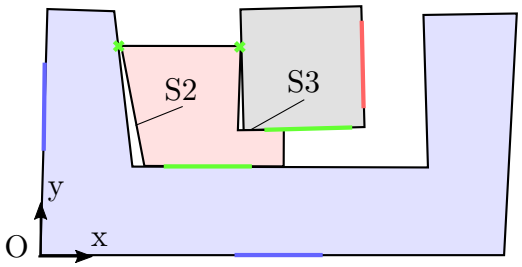

Fig. 9. Update of the interface component.

As an alternative, we propose to use a dynamic representation of the product, i.e. a digital twin, which is able to reflect its geometrical deviations and to optimize manufacturing and assembly operations accordingly. The desired hybrid representation of the product is given in Fig. 12. This hybrid representation contains an as-built component (the *base* component in blue), an interface component (the *intermediary* component in purple) and an as-designed component (the *axis* component in orange). While extra efforts have to be made in order to obtain such a virtual representation of the product, extracted assembly operations lead to a first-time-right physical product.

The structuro-functional model of the product is defined in the next subsection. In the two following subsections, we respectively detail how the as-built component is updated from a 3D acquisition result and how the interface component is updated based on as-built component geometry.

5.2. Structuro-functional model of the product

Fig. 13 presents the Oriented Contact Graph of the product from our case study. The product has three *key characteristics*: the assemblability of the *axis* component on the *base* component (KC1) and the location of the surfaces of the *intermediary* component (KC2 and KC*3*). Joints linking the different component surfaces are defined in Table 4.

Global references are defined by surfaces *A*, *B* and *C*. Global references, local references and specified elements are displayed in Fig. 13.

After defining the Oriented Contact Graph, the geometrical skeleton of our product is built. First, features corresponding to local references are defined. As shown in Fig. 14, defining *J*1, *J*2, *J*3 and *J*4 as planar joints and *J*5 as a cylindrical joint is admissible when considering as-designed components, but it leads to an over-determined system when geometrical deviations are introduced in the virtual product.

Joint types are therefore modified so that the system becomes statically determined. In the presented study case, *J*1 remains a planar joint while *J*2 and *J*3 are respectively defined as line-contact and point-contact joints. *J*4 is a cylindrical joint and *J*5 a point-contact joint (Table 4). The choice of joint types is arbitrary and modifications are performed manually.

Skeleton features corresponding to local references are then instantiated in the geometrical skeleton and positioned in relation to global references (Fig. 15).

Skeleton features corresponding to specified elements are finally instantiated. In our example, the specified element SE1, which is related to *KC*1, is a line feature materializing the axis of the surface *S*2 on the *axis* component. SE2 and SE3 are plane features that materialize the location of planar surfaces *S*4 and *S*5 of the intermediary component.

Parameters of the skeleton features initially correspond to the situation of joints and surfaces of the as-designed product. The skeleton is added to the initial DMU and related to the component's faces via geometrical constraints. Parameters of the skeleton features are driven using a design table, i.e. a simple text file that can be imported into the CAD environment.

5.3. Updating the as-built component

Once the structuro-model of the product is built, the *base* component is updated from its as-designed representation to its as-built representation within the virtual product. The skeleton features are then updated according to the geometry of the manufactured *base* component.

First, the as-built component geometry is acquired using an optical

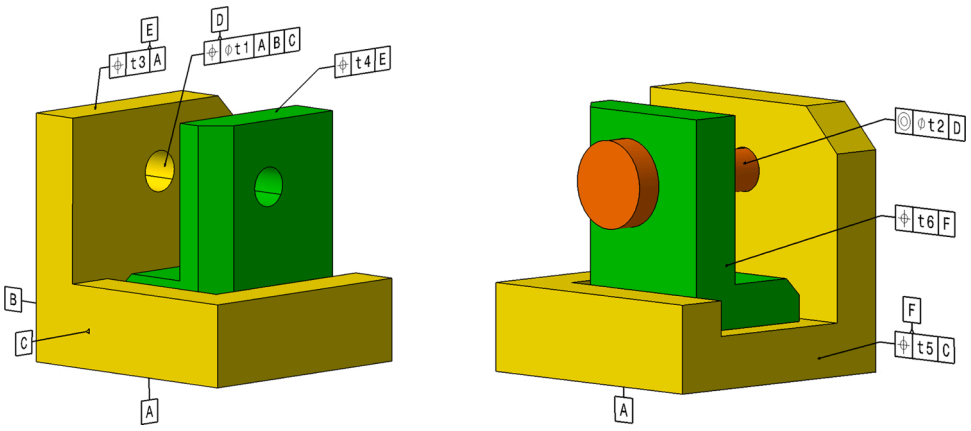

Fig. 10. Initial product DMU showing the partial representation for our case study functional requirements.

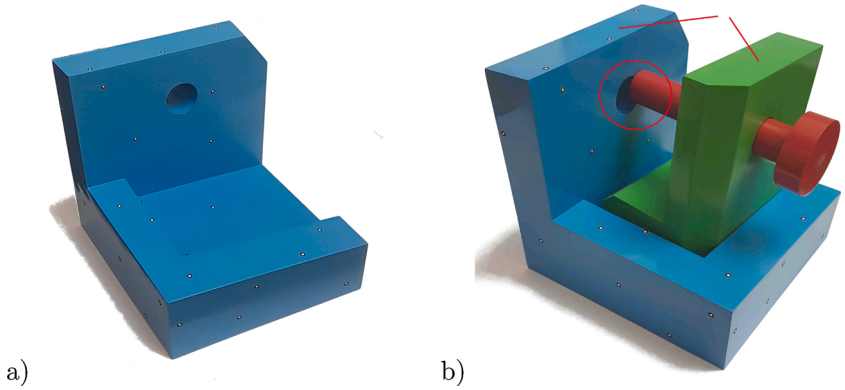

Fig. 11. (a) Manufactured component, (b) non-compliant final product.

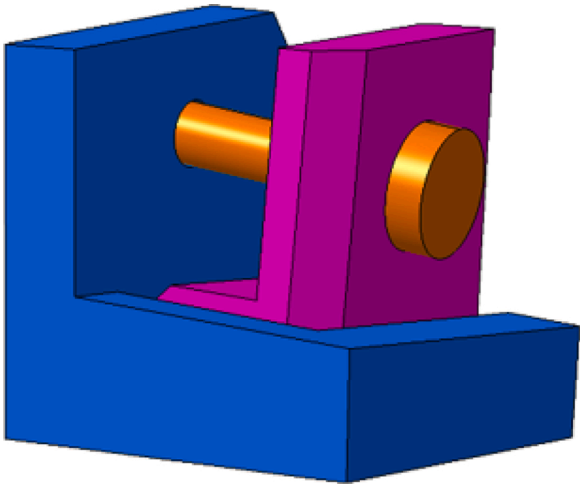

Fig. 12. Updated virtual product.

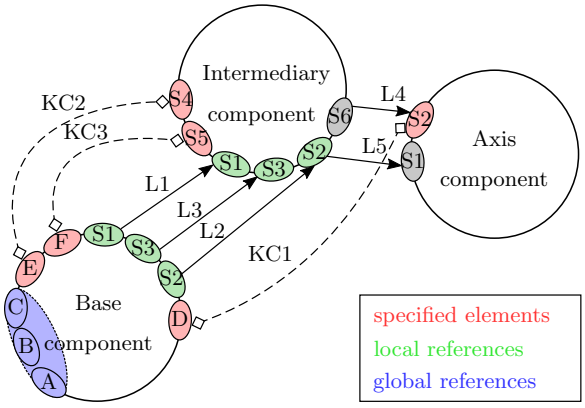

Fig. 13. Oriented Contact Graph of our case study product.

sensor[1] then exported as an STL mesh with the help of a dedicated software (Fig. 17a). The first acquisition step gives a mesh consisting of approximately 620k vertices and 1.2M faces. All the surfaces of the component are digitized.

Then the different steps of the method proposed in Section 3 are performed. After a preprocessing step, the acquired mesh is registered

Table 4
Joints before and after update.

Designation	Before update	After update
J1	Planar	Planar
J2	Planar	Line-contact
J3	Planar	Point-contact
J4	Planar	Point-contact
J5	Cylindrical	Cylindrical

[1] ATOS Core 300. See www.gom.com/metrology-systems.

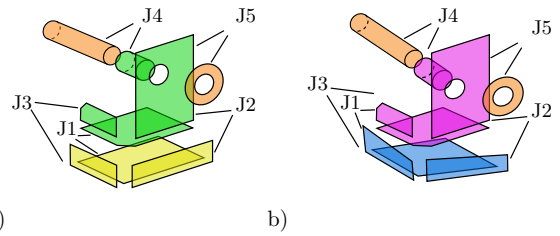

Fig. 14. Surfaces defining the assembly joints (a) initial product representation, (b) hybrid product representation.

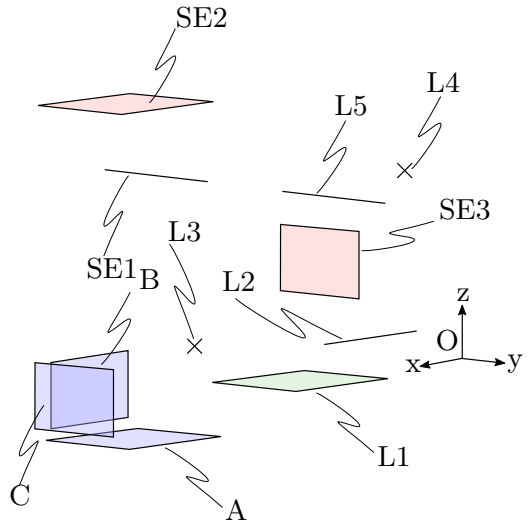

Fig. 15. Geometrical skeleton of our case study product.

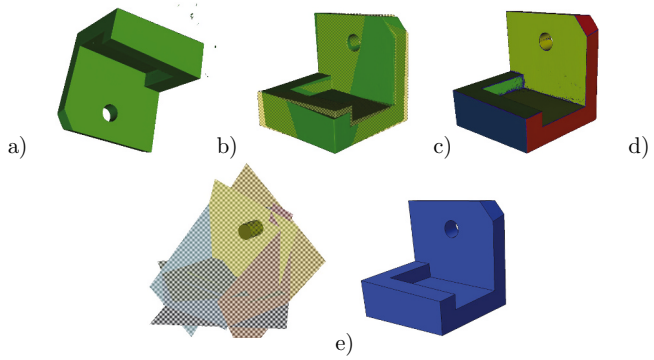

Fig. 17. (a) Acquired mesh, (b) registered mesh (as-designed component in transparent yellow), (c) classified mesh, (d) fitted surfaces, (e) as-built CAD component.

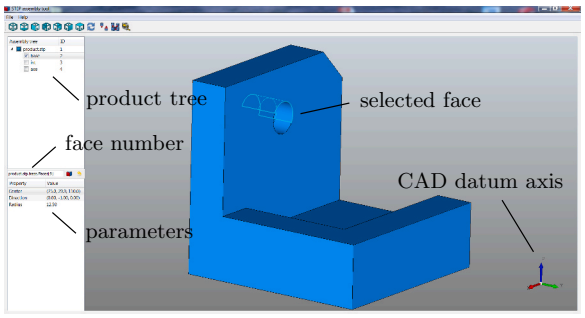

Fig. 18. D3MO software GUI.

on the as-designed component, whose geometry has previously been tessellated (Fig. 17b). Vertices of the acquired mesh are then classified according to the surface they sample (Fig. 17c).

The classifier is first trained with five different meshes resulting from the simulation of geometrical deviations using the as-designed CAD model of the *base* component (Fig. 16). Geometrical deviations are assumed to be independent between surfaces. A uniform statistical distribution is considered in order to generate these geometrical deviations. Parameters are chosen so that the order of magnitude of these deviations is approximately the same that of the manufactured component.

The vertices are thus well classified, both visually and according to the classification metric (∼97% of macro-precision – discarding the edge class – when compared to the hand-labeled ground-truth mesh). Once the acquired mesh is classified, surface parameters are computed by a least-square fitting of the primitive shape, which corresponds to each vertex subset (Fig. 17d). Beforehand, a final registration is performed, based on the global references of the assembly (planes A, B and C presented in Fig. 10) and the corresponding subsets of the mesh.

The final step consists in updating each surface parameters of the as-designed component using the previously computed values. In order to do so, a prototype software, called D3MO, based on the *pythonOCC* library [35], has been developed (Fig. 18). The STEP CAD model of the as-designed component is first imported. The parameters of each face are accessed and modified thanks to the D3MO software. The geometry of the component is then automatically updated according to the new parameters. The updated as-built component can finally be exported in a STEP format (Fig. 17.e).

All the steps of our method have been implemented in the *Python* programming language using well-known libraries for scientific computing [36], uncertainties management [37] and machine learning [38]. We also developed a graphical interface in order to assist the user in the 3D modeling step. Carrying out the aforementioned steps can be done in reduced time (∼5 min) on a laptop computer (2.3 GHz Intel® i5 CPU and 8 GB of RAM).

5.4. Updating the interface component

Based on the geometry of the as-built component, the geometry of the interface component is also updated. To do so, the situation of the interface component is first updated in the virtual product. This is done by updating the parameters of the local references in the product skeleton.

In the proposed case study, the parameters of the skeleton feature $J1$ are directly given by the parameters of the planar surface $S1$ of the *base* component. Other features have to be determined by simulating assembly operations. Thus, the assembly sequence has to be taken into

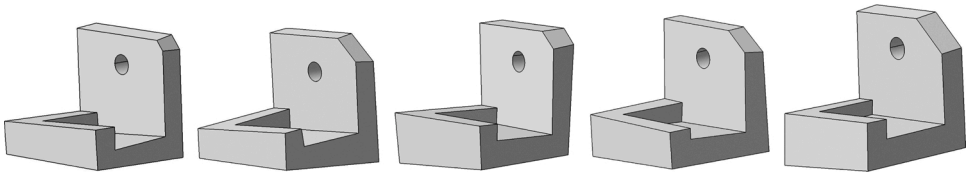

Fig. 16. Simulated instances of the *base* component composing our training set.

account.

In the proposed example, the *intermediary* component is first moved to the *base* component by realizing the planar joint *J*1. Afterwards, the line corresponding to *J*2 is computed by testing all edge-plane combinations between the two surfaces involved in the contact. The valid combination is chosen according to a non-penetration criterion. This calculation is made possible as contact surfaces are fairly simple. Cases involving more complex surfaces would require a more thorough investigation.

*J*3 is determined more easily, as it consists in a point-contact joint between planar faces. Contact points are computed by projecting the face belonging to the moving component onto the planar surface that belongs to the fixed component in the direction given by the last degree of freedom (Fig. 19).

After that, the situation of the interface component is updated in the virtual product, and its geometry is updated so that the functional requirements for the assembled product are met. To do so, surfaces of the interface component that are linked to the specified elements of the skeleton are updated in order to satisfy the condition expressed by the product's *key characteristics* (Fig. 20). In particular, the parameters of theses surfaces are updated according to the parameters of the skeleton specified features.

In the proposed case study, *SE*1 parameters correspond to the parameters of the surface axis *D* of the *base* component. *SE*2 and SE3 respectively correspond to the parameters of the planar surfaces *E* and *F*. The update for the *interface* component is carried out using the D3MO software. The application is therefore quite straightforward.

Finally, the situation of the *axis* component is updated as well, resulting in the hybrid representation presented earlier in Fig. 12. The interface component and the axis component are then manufactured based on the updated virtual product, guaranteeing that the functional requirements on the physical product are met eventually (Fig. 21).

6. Conclusion

In this article, we have proposed a hybrid virtual representation to support digital twin implementation. This representation ensures the respect of geometrical functional requirements during assembly processes. The suggested model is based, on the one hand, on a geometrical representation of the components as a set of configurable surfaces and, on the other hand, on a geometrical skeleton. The proposed hybrid representation reflects the product during its assembly process thanks to the update of the component's geometry and the situation of the skeleton features.

Component geometry is represented using the B-Rep formalism, enabling us to make full use of the capacities of current CAD environments. Local changes in the initial as-designed component geometry are performed by using a direct modeling approach. Component topology is preserved during the update from as-designed to as-built or interface state. Therefore the continuity of information linked to the component's initial as-designed representation is also preserved. As-built and interface components are also updated thanks to a restricted number of parameters. This constitutes an advantage over others virtual representations of components in the literature, such as discrete representations.

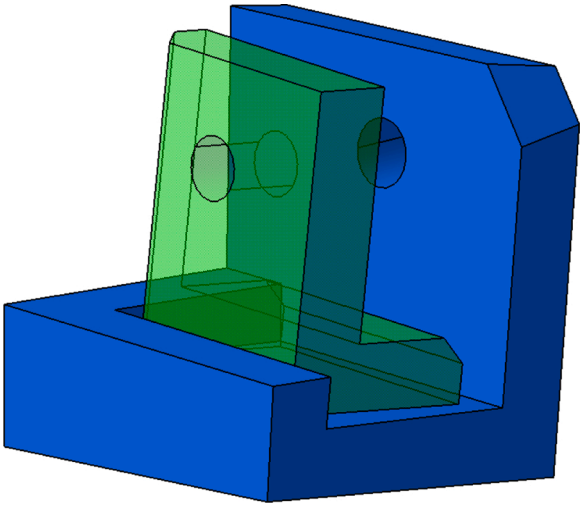

Fig. 19. The situation of the interface component is updated.

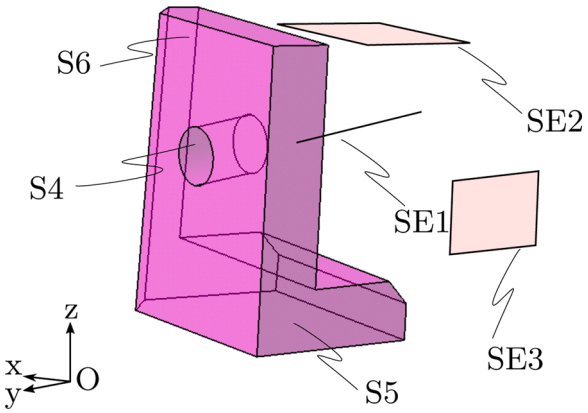

Fig. 20. The geometry of the interface component is updated.

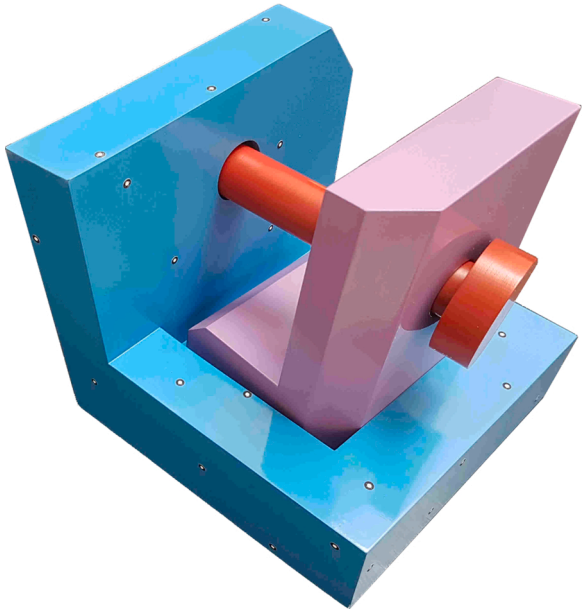

Fig. 21. Final product with updated interface component.

This process is nevertheless completed to the detriment of superior order deviations, especially shape deviations, which cannot be modeled. Shape deviations may not always be considered as negligible. In this case, their influence on the component's relative positioning has to be taken into account. Changing surfaces from geometric primitives to higher order surfaces (such as NURBS) may be a solution to overcome this shortcoming. However, this would also increase the complexity of edge computation as well as the computation of the situation of the different components within the product.

Product information is modeled using a geometrical skeleton. This skeleton helps us to translate the assembly information in terms of geometrical features. Thus, it becomes simple to simulate the impact of geometrical deviations on the assembly and to carry on various updates

in order to ensure the compliance with functional requirements of the final product.

We propose a detailed method in order to update as-built components. This method can be carried out in a relatively short time-span and requires minimal user interaction. Various improvements should nevertheless be planned in order to implement a digital twin which could be truly autonomous and dynamic (i.e. which could be updated in almost real-time). Future works include the possibility to optimize the acquisition step in order to streamline both the collection and processing of 3D data. Indeed, only a few surfaces influence the overall product behavior. One of the first questions to answer would be the ability of the proposed method to perform with sparse or missing data (i.e. non-digitized surfaces).

Eventually, the simulation of the assembly with as-built components was carried out by taking into account a statically determinate assembly. Solving the assembly constraints can be done directly and does not require to use iterative methods, as is often the case in discrete representations. However, additional developments, such as the use of Finite Element Analysis, would be necessary in order to take the component's flexible behavior into account during the assembly simulation.

Last but not least, two products were introduced in order to illustrate the proposed digital-twin-based approach. For the sake of simplicity, these products only consisted of one as-built, one as-designed and one interface component. Thus, the choice of the interface component was quite straightforward. More complexity is expected when dealing with larger products. Especially, the choice of interface components may vary from one product to another according to the nature of geometrical deviations and their impact on the functional requirements. A perspective of our work includes the possibility to integrate these aspects along with cost and feasibility criteria into a decision-based system which would allow an optimal selection of the interface components.

Conflict of interest

The authors declare that there is no conflict of interest.

Acknowledgments

The presented work was carried out thanks to a partnership between Airbus and the Automated Production Research Laboratory (LURPA) from ENS Paris-Saclay.

The authors would like to thank Arthur Alglave for his work in the development of the D3MO software.

References

[1] Tao F, Cheng J, Qi Q, Zhang M, Zhang H, Sui F. Digital twin-driven product design, manufacturing and service with big data. Int J Adv Manuf Technol 2018;94(9–12):3563–76.
[2] S“oderberg R, W”armefjord K, Carlson JS, Lindkvist L. Toward a digital twin for real-time geometry assurance in individualized production. CIRP Ann 2017;66(1):137–40.
[3] Negri E, Fumagalli L, Macchi M. A review of the roles of digital twin in cps-based production systems. Procedia Manuf 2017;11:939–48.
[4] Kritzinger W, Karner M, Traar G, Henjes J, Sihn W. Digital twin in manufacturing: a categorical literature review and classification. IFAC-PapersOnLine 2018;51(11):1016–22.
[5] Qi Q, Tao F, Hu T, Anwer N, Liu A, Wei Y, et al. Enabling technologies and tools for digital twin. J Manuf Syst 2019:1–19.
[6] Schleich B, Anwer N, Mathieu L, Wartzack S. Shaping the digital twin for design and production engineering. CIRP Ann 2017;66(1):141–4.
[7] Schleich B, Anwer N, Mathieu L, Wartzack S. Skin model shapes: a new paradigm shift for geometric variations modelling in mechanical engineering. Comput Aided Des 2014;50:1–15.
[8] Yan X, Ballu A. Toward an automatic generation of part models with form error. Procedia CIRP 2016;43:23–8.
[9] Schleich B, Anwer N, Mathieu L, Wartzack S. Contact and mobility simulation for mechanical assemblies based on skin model shapes. J Comput Inf Sci Eng 2015;15(2):021009.
[10] Yan X, Ballu A. Tolerance analysis using skin model shapes and linear complementarity conditions. J Manuf Syst 2018;48:140–56.
[11] Gómez A, Olmos V, Racero J, Ríos J, Arista R, Mas F. Development based on reverse engineering to manufacture aircraft custom-made parts. Int J Mechatron Manuf Syst 2017;10(1):40–58.
[12] Muelaner JE, Maropoulos PG. Design for measurement assisted determinate assembly (MADA) of large composite structures. Journal of the Coordinate Metrology Systems Conference 2010.
[13] Grégorio J-L, Lartigue C, Thiébaut F, Falgarone H. A reverse-engineering approach for the management of product geometrical variations during assembly. Advances on mechanics, design engineering and manufacturing II 2019:194–202.
[14] Tornincasa S, Di Monaco F. The future and the evolution of CAD. Proceedings of the 14th international research/expert conference: trends in the development of machinery and associated technology 2010:11–8.
[15] Ault HK, Phillips A. Direct modeling: easy changes in CAD? Proceedings of the 70th ASEE Engineering Design Graphics Division Midyear Conference 2016:99–106.
[16] Shah JJ, M“antyl”a M. Parametric and feature-based CAD/CAM: concepts, techniques, and applications. John Wiley & Sons; 1995.
[17] Geis A, Husung S, Oberänder A, Weber C, Adam J. Use of vectorial tolerances for direct representation and analysis in CAD-systems. Procedia CIRP 2015;27:230–40.
[18] Wirtz A. Vectorial tolerancing a basic element for quality control. Proc. of 3rd CIRP Seminars on Computer Aided Tolerancing 1993:115–28.
[19] Ansys spaceclaim. http://www.spaceclaim.com/.
[20] PTC CREO elements/direct modeling. https://www.ptc.com/en/products/cad/elements-direct/modeling.
[21] Dassault systèmes solidworks direct editing. https://blogs.solidworks.com/solidworksblog/2012/05/direct-editing-move-face.html.
[22] Marguet B. Contribution à l'analyse des variations géométriques dans les ensembles structuraux en aéronautiques: démarches et outils [Ph.D. thesis]. École Normale Supérieure de Cachan; 2001.
[23] Demoly F, Toussaint L, Eynard B, Kiritsis D, Gomes S. Geometric skeleton computation enabling concurrent product engineering and assembly sequence planning. Comput Aided Des 2011;43(12):1654–73.
[24] Varady T, Martin RR, Cox J. Reverse engineering of geometric models – an introduction. Comput Aided Des 1997;29(4):255–68.
[25] Benkő P, Martin RR, Várady T. Algorithms for reverse engineering boundary representation models. Comput Aided Des 2001;33(11):839–51.
[26] Buonamici F, Carfagni M, Furferi R, Governi L, Lapini A, Volpe Y. Reverse engineering modeling methods and tools: a survey. Comput Aided Des Appl 2018;15(3):443–64.
[27] Bosché F. Automated recognition of 3D CAD model objects in laser scans and calculation of as-built dimensions for dimensional compliance control in construction. Adv Eng Inform 2010;24(1):107–18.
[28] Erdős G, Nakano T, Váncza J. Adapting CAD models of complex engineering objects to measured point cloud data. CIRP Ann 2014;63(1):157–60.
[29] Chen Y, Medioni G. Object modelling by registration of multiple range images. Image Vis Comput 1992;10(3):145–55.
[30] Besl PJ, McKay ND. Method for registration of 3-D shapes. In: Sensor fusion IV: control paradigms and data structures, vol. 1611; 1992. p. 586–606.
[31] Bey A, Chaine R, Marc R, Thibault G, Akkouche S. Reconstruction of consistent 3D CAD models from point cloud data using a priori CAD models. ISPRS workshop on laser scanning, vol. 1 2011.
[32] Manohar K, Hogan T, Buttrick J, Banerjee AG, Kutz JN, Brunton SL. Predicting shim gaps in aircraft assembly with machine learning and sparse sensing. J Manuf Syst 2018;48:87–95.
[33] Babu M, Franciosa P, Ceglarek D. Spatio-temporal adaptive sampling for effective coverage measurement planning during quality inspection of free form surfaces using robotic 3D optical scanner. J Manuf Syst 2019;53:93–108.
[34] Weinmann M, Jutzi B, Hinz S, Mallet C. Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers. ISPRS J Photogramm Remote Sens 2015;105:286–304.
[35] Paviot T. Pythonocc, 3D CAD/CAE/PLM development framework for the python programming language. 2018. http://www.pythonocc.com/.
[36] Oliphant TE. Python for scientific computing. Comput Sci Eng 2007;9(3):10–20.
[37] Baudin M, Dutfoy A, Iooss B, Popelin A-L. Openturns: an industrial software for uncertainty quantification in simulation. 2015. http://www.openturns.org/.
[38] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: machine learning in python. J Mach Learn Res 2011;12:2825–30.