Simulated annealing-based fitting of CAD models to point clouds of mechanical parts' assemblies

Ghazanfar Ali Shah[1,2,3] · **Arnaud Polette**[1] · **Jean-Philippe Pernot**[1] · **Franca Giannini**[2] · **Marina Monti**[2]

Abstract
This paper introduces a new fitting approach to allow an efficient part-by-part reconstruction or update of editable CAD models fitting the point cloud of a digitized mechanical parts′ assembly. The idea is to make use of parameterized CAD models whose dimensional parameters are to be optimized to match the acquired point cloud. Parameters may also be related to assembly constraints, e.g. the distance between two parts. The optimization kernel relies on a simulated annealing algorithm to find out the best values of the parameters so as to minimize the deviations between the point cloud and the CAD models to be fitted. Both global and local fitting are possible. During the optimization process, the orientation and positioning of the CAD parts are driven by an ICP algorithm. The modifications are ensured by the batch calls to a CAD modeler which updates the models as the fitting process goes on. The modeler also handles the assembly constraints. Both single and multiple parts can be fitted, either sequentially or simultaneously. The evaluation of the proposed approach is performed using both real scanned point clouds and as-scanned virtually generated point clouds which incorporate several artifacts that could appear with a real scanner. Results cover several Industry 4.0 related application scenarios, ranging from the global fitting of a single part to the update of a complete Digital Mock-Up embedding assembly constraints. The proposed approach demonstrates good capacities to help maintaining the coherence between a product/system and its digital twin.

Keywords CAD assembly models · Digital twin · Constrained fitting · Registration · Simulated annealing · ICP · As-scanned point clouds

1 Introduction

Today, the needs to reconstruct or update 3D information related to real-world objects, products, systems, buildings, environments, terrestrial surfaces and even human beings have become mainstream. This topic turns out to be of primary interest in the scope of the fourth industrial revolution, commonly known as Industry 4.0, and for which the demand for access to and use of up-to-date digitized information related to the digital twin of a system has raised [1]. The increased interest in 3D digitization, combined to the emergence of low-cost devices have certainly speeded up the development and spreading of new techniques in many different application domains, ranging from facial reconstruction and comparison on smartphones to complex reconstructions for mechanical engineering or architectural applications. Depending on the application domain, a wide variety of scenarios can be foreseen, and the needs range from the acquisition and treatment of incomplete point clouds, to the full reconstruction or update of CAD assembly models potentially composed of several components and parts. In product development, the objective is to reconstruct or to update CAD parts, CAD assembly models and even full Digital Mock-Up (DMU) which can then be exploited at different stages of the Product Development Process (PDP). For instance, the reconstructed or updated 3D models can then be exploited to simulate products' and systems' behavior, so

✉ Ghazanfar Ali Shah
ghazanfar_ali.shah@ensam.eu

Jean-Philippe Pernot
jean-philippe.pernot@ensam.eu

[1] Arts et Métiers Institute of Technology, LISPEN, HESAM Université, F-13617 Aix-en-Provence, France

[2] Istituto di Matematica Applicata e Tecnologie Informatiche "Enrico Magenes", CNR Via De Marini 6, 16149 Genoa, Italy

[3] DIME-Dipartimento di Ingegneria meccanica, energetica, gestionale e dei trasporti, Università degli Studi di Genova, Genoa, Italy

as to understand the origin of some failures, to define mitigation plans and come up with enhanced performances. It can also be used to check assembly specifications of products and systems which would not have been disassembled before scanning, as well as many more scenarios. Actually, such an approach can greatly enhance the capacity of companies to develop more competitive products, while reducing the development times and costs [2]. In the scope of the Industry 4.0, the idea is to be able to update an as-is DMU so as to maintain the coherence between a physical system and its digital twin. This could be very helpful to get access to real-time information regarding for instance the state of a machine or of a production line. At the end, this would allow more accurate and faster simulations and analyses as well as more proactiveness and agility for decision-making.

Unfortunately, current approaches do not fully meet the above-mentioned requirements for high-level reconstruction and update tools able to consider the information at the level of the parts, assemblies and underlying semantics. Today, the reconstruction of CAD models from point clouds follows a more or less manual, cumbersome, and time-consuming patch-by-patch fitting strategy during which engineers have to face many issues: pre-processing of the data, segmentation of point clouds, decomposition in patches, fitting of primitives, trimming and stitching of the resulting surfaces. At the end, manifold B-Rep models are obtained but cannot be easily modified as they are considered as dead models which do not rely on real building trees [3]. This is an important limitation of the existing techniques, which prevents efficient reconstructions or updates of consistent and editable CAD models. Actually, despite this effervescence around the development of new reconstruction algorithms, very few address the challenging problem of reconstructing or updating CAD models that could then be edited and more successfully exploited in the downstream stages of the PDP. This is true when considering single parts and it is even truer for the assemblies of several parts potentially subjected to assembly constraints. Again, being able to check as-built and as-is models and tolerances as well as to analyze accessibility issues for maintenance planning without disassembling the components can drastically speed up the reverse engineering process and thus improve the performance of the PDP. More generally, being able to keep track of the evolutions of a product/system through its digital twin is of foremost importance to support decision-making and to evolve towards more optimized and autonomous products and systems. This requires a paradigm shift.

This paper introduces a new part-by-part fitting approach to circumvent those issues and allow a more global and efficient reconstruction or update of editable CAD models fitting the point cloud of a digitized mechanical parts' assembly. Depending on the adopted scenario, assembly constraints can also be specified and maintained during the fitting process. The idea is to directly make use of parameterized CAD models whose parameters are to be optimized. Parameters may also be related to assembly constraints, e.g. the distance between two parts. The proposed fitting approach relies on an optimization kernel which makes use of a simulated annealing (SA) algorithm to find out the best values of the parameters so as to minimize the deviations between the point cloud and the CAD models to be fitted. Both global and local fitting are possible. During the optimization process, the orientation and positioning of the CAD parts are driven by an ICP algorithm. The consistency of the CAD models is ensured by calling a modeler, which updates the CAD models as the fitting process goes on. This reduces the risk to get for instance badly fitted surfaces, gaps, overlapping surfaces, self-intersections. The modeler also handles the assembly constraints when specified between the CAD parts. Both single and multiple parts can be fitted, either sequentially or simultaneously.

The contribution is threefold: (1) a new part-by-part fitting framework allows the fitting of parameterized and editable CAD models to point clouds of digitized mechanical parts assemblies; (2) both local and global fitting can be performed thanks to a segmentation strategy controlled by a simulated annealing optimization algorithm; (3) the method can fit single parts as well as multiple parts optionally constrained together with assembly constraints and fitted simultaneously. The proposed approach demonstrates good capacities to help maintaining the coherence between a product or system and its digital twin.

The paper is organized as follows. Section 2 reviews the related works and positions the fitting technique with respect to state-of-the-art approaches. The proposed framework is then introduced in Sect. 3 and the details of the building blocks are given in Sect. 4. The approach is then tested and validated on several test cases, including both global and local fitting configurations, on single parts as well as on assemblies of parts subjected to assembly constraints (Sect. 5). Section 6 ends this paper with conclusions and perspectives.

2 Related works

The literature is plenty of more or less sophisticated reconstruction techniques from point clouds [4]. Here, a focus is put on fitting methods of interest to reconstruct or update CAD models. To classify the existing approaches, it is important to stress that efficient reconstruction algorithms use priors, i.e. assumptions made by algorithms to combat imperfections in the point cloud and to eventually focus what information about the shape is reconstructed. Without prior assumptions, the reconstruction problem is ill-posed, i.e. an infinite number of solutions can satisfy the fitting

problem. Of course, when addressing update issues, prior assumption is satisfied straightforwardly since it is assumed that the model to be updated exists. More precisely, the techniques which can help reconstructing or updating CAD models from point clouds can make use of three main priors: the geometric primitive prior, the global regularity prior and the data driven prior.

The *geometric primitive* prior is adopted in several techniques which try to fit basic geometric primitives in point clouds. Among the existing techniques, Fischler et al. have introduced the well-known random sampling consensus (RANSAC) paradigm to extract shapes by randomly identifying minimal sets from the point cloud and constructing corresponding shape primitives [5]. Schnabel et al. have proposed an automatic algorithm to detect basic shapes in unorganized point clouds [6]. Their method is based on RANSAC paradigm and performs a random sampling to detect planes, spheres, cylinders, cones and torii. For models with surfaces composed of these basic shapes only, e.g. CAD models, the resulting representation solely consists of shape proxies. This method has been extended not only to fill in the gaps between the detected primitives but also to synthesize plausible edges and corners generated from the intersections of basic primitives [7]. The method can be applied on point clouds of mechanical assemblies but it is unable to identify the parts properly saying. Bey et al. [8] have proposed a method to reconstruct CAD models from 3D point clouds, assuming that an a priori CAD model, roughly similar to the scene to reconstruct, is given. Their work is actually limited to cylindrical shapes and does not work on complete CAD models. Lari et al. have introduced an approach for the identification, parameterization, and segmentation of planar and linear/cylindrical features from laser scanning data, while considering the internal characteristics of the input point cloud, i.e. local point density variation and noise level in the dataset [9]. Their method is limited to very simple primitives and the resulting model is not watertight. Clustering approaches can also be used to detect geometric primitives in point clouds. To this aim, Attene et al. have proposed a hierarchical face clustering algorithm for triangle meshes based on fitting primitives belonging to an arbitrary set [10]. Their method is particularly efficient and is completely automatic which can be interesting in the reverse engineering context. It generates a binary tree of clusters, each of which fitted by one of the primitives employed. However, their method does not perform well on incomplete scanned data and there is no CAD model reconstruction.

The *global regularity* prior deals with high-level properties such as symmetries, structural repetitions, and canonical relationship [11]. This is particularly interesting when considering the fitting of CAD models for which basic primitives can follow some specific rules. Among the existing techniques, Li et al. [12] have developed the so-called GlobFit method that simultaneously recovers a set of locally fitted primitives along with their global mutual relations. Starting with a set of initial RANSAC-based locally fitted primitives [6], relations across the primitives such as orientation, placement, and equality are progressively learned and conformed to. This algorithm operates under the assumption that the data correspond to a man-made engineering object consisting of basic primitives, possibly repeated and globally aligned under common relations. Monszpart et al. have proposed the so-called RAPter algorithm to abstract raw scans by regular arrangements of primitive planes by simultaneously extracting a set of primitives along with their inter-primitive relations [13]. However, those methods can hardly deal with CAD models made of more complex features, e.g. blends, draft features. Furthermore, they act at the level of the parts and not at the level of an assembly of parts. Finally, as they were initially designed to reconstruct basic primitives, this category of methods does not allow for a proper update of neither existing CAD models nor DMU. Such an ability would, however, be of great interest to update digital twins, and thus, to answer some Industry 4.0-related needs.

The *data driven* prior makes use of existing objects or object parts to be fitted to the point clouds in a rigid or non-rigid manner. This is much more studied in the context of scene understanding than for the reconstruction or update of mechanical parts assemblies. For instance, Nan et al. [14] have proposed an interesting approach for indoor scene understanding. The method is based on initial random selection and iterative region growing with increasing classification likelihood. The scene is then reconstructed by deforming template models to fit the classified points. However, this technique allows for global fitting only and the adopted templates are far more simple than real industrial CAD models which integrate much more features. Assembly constraints are also not considered. Anyway, in the CAD domain, Liu Ip et al. have proposed a new approach to retrieve a CAD model in a point cloud and to align it using a transformation based on principal components analysis [15]. Here, the parameters of the CAD model are not modified so as to best fit the point cloud. The rigid transformation could also be obtained using an efficient variant of the ICP (Iterative Closest Point) algorithm [16]. Rabbani et al. have compared several techniques to directly fit CSG objects to point clouds [17]. Their ICT (Iterative Closest Triangle-point) algorithm is able to identify the values of some free parameters as well as to maintain the relationships between some other parameters. The algorithm can perform global fitting which limits its use when considering mechanical parts' assemblies. Buonamici et al. [18] have introduced a template-based technique for the reverse engineering of mechanical parts. Here, a CAD template is fitted upon the mesh generated from the point cloud, optimizing its dimensional parameters and position/

orientation by means of particle swarm optimization algorithm. In their approach, the reference mesh is subdivided into N meshes (i.e. N single features) through a segmentation process with the help of a commercial RE software, and afterwards, each segmented region is then registered to the corresponding surface of the CAD model. This method allows for a global fitting of the CAD template to the reference mesh. Unfortunately, the method focuses on global fitting of parts, and neither the reconstruction nor update of CAD assembly models is considered. Wang et al. [19] have designed a framework to create 3D models from the boundary surface meshes of industrial parts. It exploits a divide-and-conquer strategy to construct all primitive features of parts. According to the geometric and topological relationships among features, some modeling operations are performed to obtain the final model. Here again, the result is a dead B-Rep model which can be hardly modified in the downstream stages of the PDP. Another semi-automated approach is introduced by Stark et al. [20] for the reconstruction of 3D assembly models through the process of segmentation, part identification and structure identification. Point cloud data obtained from scanning are used for finding parts from database and retrieved parts are used for reconstruction of assembly model. Bénire et al. [21] have introduced an automatic process for the reverse engineering of models from 3D meshes. However, the method works on meshes obtained by mechanical object discretization. Finally, Xu et al. [22] have developed a modeling strategy to reconstruct a mechanism from multi-view images. It uses an interactive part modeling step to draft the parts which are then fitted to the mechanism point cloud. Parts are then aligned using an optimization step and the motion parameters can be estimated using a pre-recorded video clip of the mechanism motion. This method is interesting as it does not need to use a parameterized model as input; however, the segmentation properly speaking is left to the user who specifies the correspondences between the parts and the point cloud.

As conclusion, even if some of the previously discussed methods can be of interest to support the fitting process of mechanical parts, they still suffer from several limitations. First, mostly single CAD parts are reconstructed, and very few attention is paid to the direct reconstruction or update of CAD assembly models from digitized part assemblies. Indeed, existing methods mostly work at the level of the parts, and they require the assembly to be disassembled. This is incompatible when considering the needs for checking and processing as-is DMUs, and especially in the context of the Industry 4.0 where the digital twins should follow as much as possible the evolution of the physical products/systems, without disassembling them. Second, the reconstructed CAD models cannot be easily modified in the downstream stages of the PDP as they often correspond to dead B-Rep models. These are the limitations to overcome with the proposed fitting framework. In line with the Industry 4.0-related scenarios identified and discussed in the introduction, the proposed method exploits the assumption that parameterized CAD models to be fitted are a priori identified.

3 Overall framework

This section introduces the overall framework of the simulated annealing-based fitting technique (Fig. 1). The details of the different modules are provided in Sect. 4. Starting from a real-life mechanical parts' assembly, a point cloud PC_0 is first acquired using for instance a laser scanner. One or several CAD models to be fitted in the point cloud are then selected, being $\mathcal{M}^\kappa$ the κ-th one (with $\kappa \in \{1, \ldots, N_m\}$, and N_m the number of models to be fitted). CAD models can optionally be part of an assembly structure $\mathcal{A}$, potentially composed of subassemblies, and their fitting can be performed either sequentially or simultaneously. Here, it is assumed that the CAD models are parameterized at the level of the parts (e.g. lengths, diameters and angles parameterizing the features) and/or at the level of the assemblies (e.g. lengths and angles parameterizing the relative positions and orientation of the parts the ones with the others). Both types of parameters can be considered as numerical variables x_k of the fitting process. Each CAD model may also contain additional internal constraints (e.g. equations linking parameters of the part, constraints between geometric

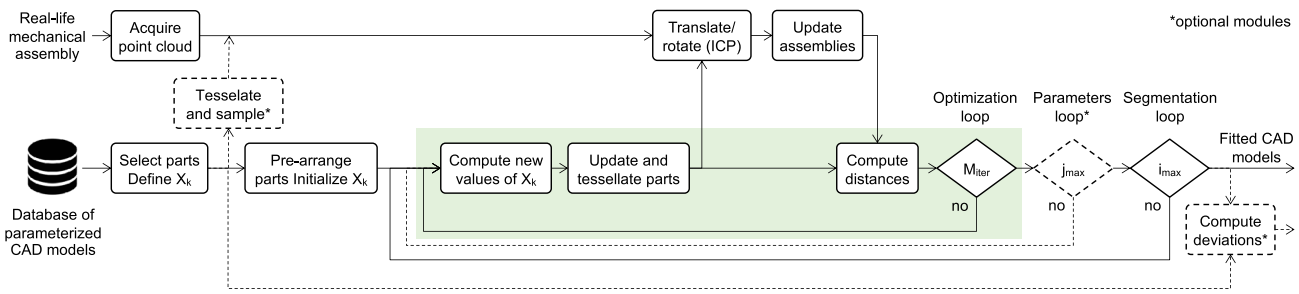


Fig. 1 Overall framework to fit parameterized CAD assembly models to point clouds of digitized mechanical part assemblies

entities as for instance perpendicularity and parallelism), and so does each assembly (e.g. coaxiality of two axes, contact between two parts, equations between the parts′ parameters). These internal constraints do not take part to the optimization process, and are directly handled by the CAD modeler in charge of updating the assemblies and CAD models each time their numerical parameters change. The numerical parameters can optionally be gathered in several groups $\mathcal{G}_j$, with $j \in \{1, \dots, j_{\max}\}$, corresponding to the level of details they are related to. The parts are pre-arranged within the point cloud; this initializes the values x_k^0 of all the numerical variables. The iterative fitting process then starts following three nested loops (Fig. 1):

- the *segmentation loop* aims at segmenting the initial point cloud PC_0 so as to consider only the points in the surrounding of the N_m models to be fitted, and thus to allow for local fitting. This iterative process is performed $i_{\max}$ times. During the i-th loop, with $i \in \{1, \dots, i_{\max}\}$, a threshold $\mathcal{E}_i^\kappa$ is computed for each CAD model ($\kappa \in \{1, \dots, N_\mathrm{m}\}$). It is used to get the surrounding point cloud PC_i^κ, while cropping PC_{i-1}^κ obtained during the previous segmentation loop. Thus, the number of points to be considered in the optimization loop progressively reduces step after step. For the first loop ($i = 1$), each $\mathcal{E}_1^\kappa$ is initialized from the oriented bounding box of the κ-th pre-arranged CAD model. These thresholds are used to initialize the segmentation and to generate the various PC_1^κ from the initial point cloud PC_0 and from the position of the pre-arranged CAD models. In case of global fitting, the segmentation loop is not used and the point cloud PC_0 is considered as a whole all along the fitting process;
- the *parameters loop* is optional and it is executed only if more than one group of parameters ($j_{\max} > 1$) is defined. For each step of the segmentation loop, the parameter loop treats the $j_{\max}$ groups one after the other. Each group $\mathcal{G}_j$ contains one or more parameters $x_{j,k}$ to be optimized in the optimization loop;
- the *optimization loop* is the core of the proposed approach. It is based on a simulated annealing algorithm which iteratively modifies the numerical parameters of the CAD models until they perfectly fit their respective cropped point clouds PC_i^κ according to a stop criterion. Here, the algorithm works on a given group $\mathcal{G}_j$ containing a restricted set of parameters $x_{j,k}$ to be optimized. During this loop, several steps are to be performed. First, the SA algorithm identifies new parameters' values $x_{j,k}(t)$, and the CAD models are then updated accordingly. The update is left to the CAD modeler which ensures the consistency of the updated B-Rep models and manages the internal constraints defined in the building trees of the parts. The updated CAD models are then tessellated on one hand, and used as input of an ICP algorithm on the other hand. The ICP algorithm is used to compute a homogeneous transformation matrix so that the updated CAD models align with their respective PC_i^κ. The CAD modeler then updates the assemblies so as to satisfy the possibly defined assembly constraints between the parts. Thus, the position and orientation of the parts are not considered directly as variables of the optimization process. The distances between the updated CAD models and their correspondent cropped point clouds PC_i^κ can now be computed and the result serves as the objective function to be minimized during the optimization. The optimization loop stops when a max number of iteration M_{iter} without any change (up to a given accuracy ϵ_{iter}) is reached, otherwise the SA algorithm goes on with the computation of new values $x_{j,k}(t+1)$. Once the optimal values of the parameters have been found, the algorithm goes back to the parameters loop if $j < j_{\max}$, otherwise it goes back to the segmentation loop so as to start a new segmentation if $i < i_{\max}$.

At the end of the loops, one or several fitted CAD models are obtained. Depending on the adopted scenario and needs, this fitting process can be repeated for other parts of the digitized assembly. Finally, since the fitted CAD models are editable, the user can still modify them while rounding for instance the values of the optimized parameters. Moreover, the final cropped point clouds $\mathrm{PC}_{i_{\max}}^\kappa$ correspond to partial segmentations of PC_0 reflecting the fitted parts.

For experimentation and validation purposes, the scanning of a real-life mechanical assembly can be optionally bypassed (dashed lines of Fig. 1) by the automatic generation of an as-scanned point cloud using the virtual acquisition technique of Montlahuc et al. [23]. In this case, the initial CAD models to be fitted are known, and the values of their parameters can be compared with the ones resulting from the proposed fitting strategy. The virtual scanning technique is briefly introduced in Sect. 5 as it is used to benchmark and validate the fitting algorithm.

4 Fitting framework modules

This section introduces the technical aspects underlying the fitting of one or several CAD models $\mathcal{M}_\mathrm{m}$ in the point cloud PC_0 of a digitized parts' assembly. CAD models can optionally be part of an assembly structure $\mathcal{A}$, potentially composed of subassemblies, and constrained with assembly constraints. For sake of clarity, the successive steps are illustrated on the fitting of a single flange in the point cloud of a digitized valve (Fig. 2). The point cloud is composed of 1204k points. Section 5 presents and discusses more results

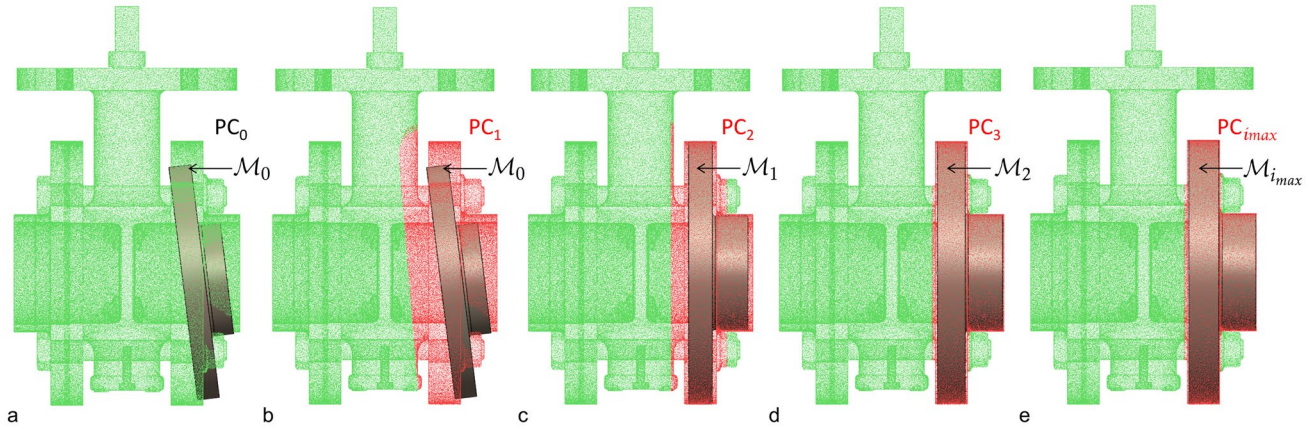


Fig. 2 Fitting the parameterized CAD model of a flange in the point cloud of a digitized valve assembly ($N_m = 1$): **a** coarse pre-arrangement in the initial point cloud PC_0; **b–d** evolution of the cropped point clouds PC_i during the segmentation loops; **e** final fitted flange $\mathcal{M}_{i_{max}}$ and associated segmentation $PC_{i_{max}}$

to cover various fitting scenarios involving several parts and assembly structures.

4.1 CAD models' selection and pre-arrangement

Once the parts' assembly has been digitized, a point cloud PC_0 is available; the first step is to select the CAD models to be fitted. The way this is performed depends on the adopted fitting scenario. The models can come from an existing database available in the company, when considering for instance the need to either update the DMU of a system/product or control its assembly. They can also be roughly and rapidly sketched and parameterized starting from scratch using the CAD modeler. Designers can also make use of other existing databases which could be browsed using ad hoc assembly retrieval approaches [24].

Each of the N_m selected CAD model, in the following indicated as $\mathcal{M}^\kappa$, is defined by a set of control parameters $p^\kappa_{k'}$, $k' \in \{1, \dots, N^\kappa_p\}$, which correspond to either dimensions or angles. Additional numerical parameters $a_{k'}$, $k' \in \{1, \dots, N_a\}$ may also be considered if the CAD models are part of an assembly structure $\mathcal{A}$ involving assembly constraints also parameterized by dimensions or angles. Both types of parameters are considered as numerical variables x_k to be optimized during the optimization loop. The assembly structure and the CAD models may also include built-in constraints (e.g. contact between parts, coaxiality, symmetry, parallelism, relationships between parameters) to be maintained during the modifications. The updates of the CAD models and assembly structure are left to a CAD modeler which ensures the consistency of the B-Rep model all along the optimization process. Thus, both the assembly structure $\mathcal{A}$ and its associated CAD models $\mathcal{M}^\kappa$ can be seen as the result of several generation functions g^* described in the assembly and building trees and so that:

$$\begin{cases} \mathcal{A} = g^a(a_1, \dots, a_{N_a}) \\ \mathcal{M}^\kappa = g^\kappa(p^\kappa_1, \dots, p^\kappa_{N^\kappa_p}), \quad \forall \kappa \in \{1, \dots, N_m\} \end{cases} \tag{1}$$

Optionally, the control parameters can be split in several groups $\mathcal{G}_j$, $j \in \{1, \dots, j_{max}\}$, according to the level of details to which they correspond. The parameters of the group $\mathcal{G}_j$ are denoted $p_{j,k}$, $k \in \{1, \dots, N_{pj}\}$, and the numerical variables for the optimization loops are the $x_{j,k} = p_{j,k}$. Actually, using a three-level decomposition turns out to be a good trade-off considering commonly adopted CAD modeling strategies ($j_{max} = 3$): $\mathcal{G}_1$ groups the parameters of the assembly structure as well as the parameters driving the structural features (e.g. pads, pockets, revolutions), $\mathcal{G}_2$ gathers together the parameters of the detail features (e.g. holes, ribs), and $\mathcal{G}_3$ is concerned by the parameters used to finalize the CAD model (e.g. fillets, chamfers). When considering the fitting of several CAD models, all their parameters are mixed up within those three groups. Thus, when referring to those groups, the upper indices κ is no more used. Moreover, to simplify the writing, this upper indices will also not be used in case a single part is to be fitted ($N_m = 1$).

For example, Fig. 3 shows the parameters of the single flange ($N_m = 1$) to be locally fitted in PC_0 as shown on Fig. 2. It is defined by eight control parameters ($N_p = 8$): radii r_i with $i \in \{1, \dots, 6\}$, heights h_1 and h_2. It also contains symmetry constraints to be maintained by the CAD modeler. As only one single flange is to be fitted, there are no assembly constraints ($N_a = 0$). The three groups can then be defined as follows:

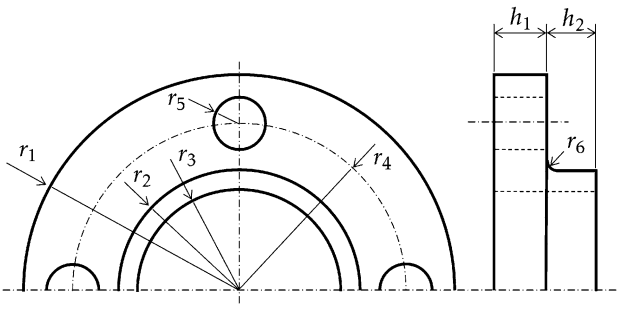


Fig. 3 Definition of the eight parameters controlling the shape of the half-flange that is fitted in the point cloud of a digitized assembly as depicted on Fig. 2

$$\begin{aligned}\mathcal{G}_1 =&\{r_1, r_2, r_3, h_1, h_2\} \,:\, p_{1,1} = r_1,\ p_{1,2} = r_2,\\ p_{1,3} =&r_3, p_{1,4} = h_1 \ \text{ and } \ p_{1,5} = h_2,\\ \mathcal{G}_2 =&\{r_4, r_5\} \,:\, p_{2,1} = r_4 \ \text{ and } \ p_{2,2} = r_5,\\ \mathcal{G}_3 =&\{r_6\} \,:\, p_{3,1} = r_6.\end{aligned} \tag{2}$$

The CAD models and the assembly structure are then pre-arranged in the point cloud PC_0; this initializes the values x_k^0 of all the parameters, and thus it initializes the pre-arranged CAD models $\mathcal{M}_0^\kappa$ and assembly structure $\mathcal{A}_0$. This step depends on the adopted fitting scenario. For instance, it can be performed using pre-identified primitives to align certain axes of the CAD models with axes obtained using for instance RANSAC paradigm [6]. The initial values of the control parameters x_k^0 can also be tuned in a coarse manner or using global scaling factors. Of course, as for any numerical method, the idea is to start as close as possible to the final solution to avoid local minimums. This step is illustrated on Fig. 2a where a coarse pre-arrangement of the flange has been performed.

4.2 Segmentation loop

From this pre-arrangement, if a local fitting is to be considered, the algorithm crops the initial point cloud PC_0 to get the subsets PC_1^κ associated to the N_m models to be fitted. For the κ-th model, the cropping is driven by an initial threshold $\mathcal{E}_1^\kappa$. All the points of PC_0 which have a distance to $\mathcal{M}_0^\kappa$ greater than $\mathcal{E}_1^\kappa$ are cropped (Fig. 2b). By default, $\mathcal{E}_1^\kappa$ is set up to 10% of the diagonal of the pre-arranged part oriented bounding box. This percentage corresponds somehow to the level of confidence in the pre-arrangement, thus the default value can be adjusted. The cropping is done in the same way for each model $\mathcal{M}^\kappa$, but with different $\mathcal{E}_1^\kappa$ values. The cropped PC_1^κ are then used to optimize the parameters of all the groups $\mathcal{G}_j$ in the parameters and optimization loops. Those two loops end up with the definition of the $\mathcal{M}_1^\kappa$ fitting the PC_1^κ. The process goes on this way with successive values $\mathcal{E}_i^\kappa$, $i \in \{2, \dots, i_\mathrm{max}\}$ used to crop all the points of the PC_{i-1}^κ which have a distance to the $\mathcal{M}_{i-1}^\kappa$ greater than $\mathcal{E}_i^\kappa$, and thus define the PC_i^κ used to generate the $\mathcal{M}_i^\kappa$. Here, the $\mathcal{E}_i^\kappa$ are defined according to the max distance between the PC_{i-1}^κ and the $\mathcal{M}_{i-1}^\kappa$ such that

$$\mathcal{E}_i^\kappa = \omega_s^\kappa \times \max\left\{ \mathrm{d}\left(\mathcal{M}_{i-1}^\kappa,\ \mathrm{PC}_{i-1}^{\ \kappa}[\tau]\right) : \tau \in [1..n_{i-1}^\kappa] \right\}, \tag{3}$$

wherein n_{i-1}^κ is the number of points in PC_{i-1}^κ and ω_s^κ is a segmentation weight empirically set up to 10%.

The segmentation loops are illustrated on the example of Fig. 2. Figure 2c shows $\mathcal{M}_1$ fitting PC_1 and used to compute $\mathcal{E}_2$ and PC_2. Similarly, Fig. 2d depicts $\mathcal{M}_2$ fitting PC_2 and used to get $\mathcal{E}_3$ and PC_3. Figure 2e shows the final fitted CAD model $\mathcal{M}_{i_\mathrm{max}}$ fitting $\mathrm{PC}_{i_\mathrm{max}}$, with $i_\mathrm{max} = 3$.

4.3 Parameters loop

As introduced in Sect. 4.1, the parameters $x_{j,k}$ of the j-th group can refer to either parameters of the assembly structure or parameters of one or several CAD models. The parameters grouping is not performed according to the models but considering the level of details to which they correspond. Thus, the number of groups j_max is limited and is not affected by the number N_m of models to be fitted.

If groups of parameters are considered ($j_\mathrm{max} > 1$), then this nested loop treats each group one by one, otherwise the parameters are treated all together in a single step. The treatment makes use of the cropped point clouds PC_i^κ generated in the i-th segmentation loop. For each group $\mathcal{G}_j$, $j \in \{1, \dots, j_\mathrm{max}\}$, it consists in running the optimization loop to get the optimized parameters $x_{j,k}$, with $k \in \{1, \dots, N_{pj}\}$.

4.4 Optimization loop

The objective of this loop is to find out the optimal values of the parameters $x_{j,k}$ of a parameters group $\mathcal{G}_j$ (parameters loop) so that the CAD models $\mathcal{M}_i^\kappa$ best fit their respective cropped point clouds PC_i^κ (segmentation loop). Actually, this can be formulated as a minimization problem:

$$\min_{\substack{x_{j,k} \in \mathcal{D}_{j,k} \\ k \in [1..N_{pj}]}} E_{i,j}(x_{j,1}, \dots, x_{j,N_{pj}}) = \sum_{\kappa=1}^{N_\mathrm{m}} \mathrm{d}\left(\mathrm{PC}_i^{\ \kappa}, \mathcal{M}_i^\kappa\right), \tag{4}$$

where $x_{j,k}$ are the variables, $\mathcal{D}_{j,k}$ their definition domains, $\mathcal{M}_i^\kappa$ the CAD models whose deviations to the cropped point clouds PC_i^κ are to be minimized, and $E_{i,j}$ the energy function characterizing this overall deviation. During the successive optimization loops, the $\mathcal{M}_i^\kappa$ are updated by the CAD modeler using the generation functions g^* of Eq. (1) where only the subset of parameters $x_{j,k}$ is modified. As previously said,

the functions g^* incorporate built-in constraints (e.g. symmetries, relationships) which are not directly accessible in the optimization loop but can be satisfied from the building and assembly trees. The way the distances between the points of the PC_i^κ and the CAD models $\mathcal{M}_i^\kappa$ are computed is explained in Sect. 4.4.4.

4.4.1 Resolution using simulated annealing algorithm

The minimization of the energy function $E_{i,j}$ is obtained using a metaheuristic algorithm able to solve the optimization problem in a large solution space. Simulated annealing (SA) algorithm has demonstrated good proficiency to find efficiently the solution from a pre-arranged configuration. It performs a global stochastic search that evolves towards local searches as the time goes on. Thus, it is particularly interesting in the present case for which the initial positions have been pre-arranged in the close vicinity of the final fitted configurations. The algorithm handles a very limited set of constraints, mainly the lower and upper bounds of the parameters values, and both the internal and assembly constraints (e.g. coincident, parallelism, coaxiality, contact) are left to the CAD modeler in charge of the updates. The use of such an algorithm is particularly interesting in the present case as the energy function to be minimized, and the geometric constraints to be satisfied, are not defined by equations but by means of black boxes combining calls to several procedures of the CAD modeler [25, 26]. Other metaheuristics have been tested but have demonstrated a lower efficiency than SA. For instance, particle swarm optimization (PSO) generates candidate solutions which can be significantly different from the initial position. This optimization strategy may result in configurations that are hard for the CAD modeler to update, and may even cause the software to crash.

Starting from an initial configuration, potentially composed of an assembly structure $\mathcal{A}(0)$ and one or several CAD models $\mathcal{M}_i^\kappa(0)$ defined by the variables $x_{j,k}(0)$ of the j-th group, the algorithm iterates on the parameters $x_{j,k}(t)$ thus creating an evolution of the current assembly structure $\mathcal{A}(t)$ and models $\mathcal{M}_i^\kappa(t)$. The optimization loop stops when a max number of iteration M_{iter} without any change of $E_{i,j}$ (up to a given accuracy ϵ_{iter}) is reached, otherwise the SA algorithm goes on with the computation of new values $x_{j,k}(t+1)$ defining an updated assembly structure $\mathcal{A}(t+1)$ and updated models $\mathcal{M}_i^\kappa(t+1)$. In this nested loop, the updates are performed while only considering the parameters of the j-th group. The way the SA control parameters are tuned is discussed in Sect. 5.2.

4.4.2 CAD models update and tessellation

The SA algorithm evolves thanks to the variations of the energy function to be minimized. This energy is based on the distance between the current CAD models $\mathcal{M}_i^\kappa(t)$ and their cropped point clouds PC_i^κ. Thus, at each step of the optimization loop, the CAD modeler updates the $\mathcal{M}_i^\kappa(t)$ according to the values $x_{j,k}(t)$. However, the possible assembly structure is not updated at this step, but after the ICP registration as discussed in Sect. 4.4.3. Once updated, each CAD model is then tessellated to get the triangle meshes $\mathcal{M}_{i\triangleright}^\kappa(t)$. Actually, in the proposed implementation, the tessellated models are directly the ones generated by the modeler for the visualization purposes, thus they generally contain few triangles with widely varying shapes and dimensions. Then, it is possible to compute the deviation between each $\mathcal{M}_{i\triangleright}^\kappa(t)$ and its PC_i^κ by means of computing the distances between a point cloud and a mesh (Sect. 4.4.4).

4.4.3 Registration with ICP and assemblies update

The parameters $x_{j,k}(t)$ controlling the evolution of the CAD models $\mathcal{M}_i^\kappa(t)$ during the optimization loop do not affect the position and orientation of the parts with respect to the cropped point clouds PC_i^κ to be fitted. Indeed, adding six additional parameters to control the position and orientation of each part (i.e. $6 \times N_{\mathrm{m}}$ additional control parameters overall) would clearly reduce the performances of the SA algorithm. Thus, once the $\mathcal{M}_i^\kappa(t)$ have been updated, their position and orientation are modified using an ICP algorithm [27] that finds a best fit rigid body transformation between each $\mathcal{M}_{i\triangleright}^\kappa(t)$ and its PC_i^κ. Then, if it exists, the assembly structure $\mathcal{A}(t)$ is updated by the CAD modeler and the assembly constraints are satisfied.

4.4.4 Distance computation

Once the possible assembly structure $\mathcal{A}(t)$ has been updated, and once the $\mathcal{M}_i^\kappa(t)$ have been updated, tessellated and registered, the deviation between PC_i^κ and $\mathcal{M}_i^\kappa(t)$ can be computed as follows:

$$d(\mathrm{PC}_i^\kappa, \mathcal{M}_i^\kappa(t)) = \sum_{\tau=1}^{n_i^\kappa} d^2(\mathrm{PC}_i^\kappa[\tau], \mathcal{M}_{i\triangleright}^\kappa(t)), \tag{5}$$

where n_i^κ is the number of points in PC_i^κ, and d(point, mesh) is the distance function that returns the closest distance between a point and a mesh. In the proposed implementation, this evaluation is performed efficiently by CloudCompare called in batch mode. When considering several CAD models to be fitted simultaneously, the deviations are computed for each model $\mathcal{M}^\kappa$ with $\kappa \in \{1, \ldots, N_{\mathrm{m}}\}$, and each

contribution is added up in the overall energy function to be minimized (Eq. 4).

4.5 End of the loops

The three loops operate in a nested manner. Once the optimal parameters values have been found in the related optimization loop, the algorithm goes back to the parameters loop if $j < j_{\max}$, otherwise it goes back to the segmentation loop and if $i < i_{\max}$ it starts a new segmentation. When all the loops are finished, the $\mathcal{M}^{\kappa}_{i_{\max}}$ correspond to the fitted CAD models, and the $\mathrm{PC}^{\kappa}_{i_{\max}}$ to the by-part segmentations of the original point cloud.

This is illustrated on Fig. 2 which shows the evolution of the cropped point clouds PC_i during the segmentation loops, as well as the final fitted flange $\mathcal{M}_{i_{\max}}$ and associated segmentation $\mathrm{PC}_{i_{\max}}$. The numerical results associated to this example are discussed in Sect. 5.6.

5 Results and discussion

This section aims at presenting some additional results to validate various fitting scenarios. It briefly introduces the technique used to generate as-scanned point clouds, as well as the procedure to tune the initial temperature of the SA algorithm. Four experimentations are then presented and discussed. The first example illustrates the global fitting of a L-like shape in an as-scanned point cloud. It allows the validation and benchmark of the proposed approach when considering more or less noisy data. The second example is used to validate the proposed fitting strategy on a real scanned point cloud. The third example illustrates how the fitting approach can be used to track the position and orientation of robot arms, and thus update the digital twin of a physical system in the scope of the Industry 4.0. The fourth example is more complex as it demonstrates how to update a CAD assembly model following a part-by-part fitting strategy, while also considering the simultaneous fitting of several parts as well as the possibility to handle assembly constraints during the optimization loops.

The core of the fitting algorithm has been implemented in MATLAB®, which is able to call the built-in functions of SolidWorks® to perform the successive updates and ensure the consistency of the resulting B-Rep models during the optimization loops. The ICP algorithm is run in CloudCompare also called in batch mode.

5.1 As-scanned point cloud generation

To experiment and validate the proposed fitting strategy, it is mandatory to rely on ground truth data. Thus, the overall fitting framework has been customized to allow bypassing the scanning of a real-life mechanical assembly (dashed lines of Fig. 1) and to enable the use of as-scanned virtually generated point clouds. As-scanned point clouds are automatically generated from CAD models. The resulting point clouds incorporate various realistic artifacts (e.g. non-uniform sampling, missing data, noise and outliers) that would appear if the corresponding real objects were digitalized with a real acquisition device. The details of the virtual scanning algorithm can be found in the paper of Montlahuc et al. [23]. Overall, the generation of an as-scanned point cloud follows a six-step sequential process: wrapping of the assembly model to produce a watertight triangle mesh, resampling to control the density, removal of hidden points to simulate the occlusion phenomenon, generation of misalignments to simulate multiple poses, insertion of noise and outliers. Each step is optional. The removal of points follows the efficient HPR algorithm of Katz et al. [28], and the insertion of noise is obtained, while moving points along the line of sight using a Gaussian random distribution. This is illustrated on Fig. 4.

Following this virtual scanning process, the final values p_k^F of all the parameters controlling the fitted CAD models and assembly structure can be compared to the original values p_k^D of the ground truth models as they appear in the DMU. As a consequence, for each parameter, the relative deviation δp_k and the absolute deviation Δp_k can be computed as follows:

$$\Delta p_k = \frac{|\delta p_k|}{\left|p_k^D\right|} = \frac{\left|p_k^D - p_k^F\right|}{\left|p_k^D\right|} \; \forall k \in [1 \dots N_{\mathrm{p}}] \quad \text{with } N_{\mathrm{p}} = \sum_{j=1}^{j_{\max}} N_{pj}, \tag{6}$$

where N_{p} stands for the overall number of parameters possibly distributed in the parameters groups. The absolute deviation is computed at the end of each segmentation loop ($\forall i \in \{1, \dots, i_{\max}\}$) so as to appreciate the convergence step after step, whereas the relative deviation is computed

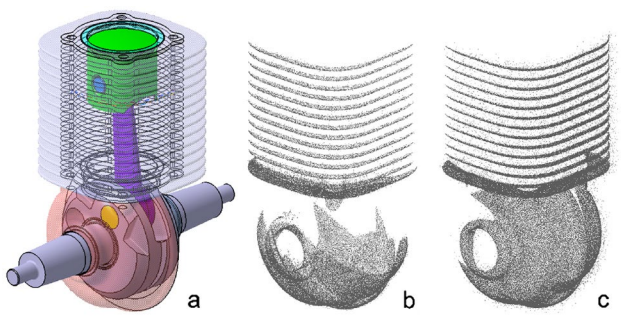


Fig. 4 Generation of as-scanned point clouds [23]: **a** CAD assembly model to be virtually scanned, **b** point cloud without noise, and with noise, **c** using different view points

on the final configuration only ($i = i_{\max}$) so as to assess the accuracy of the overall fitting process.

5.2 Tuning of the SA algorithm initial temperature

For an efficient search of the x_k optimal numerical values, the SA algorithm initial temperature T_0 needs to be tuned. In the proposed implementation, T_0 is initialized before entering the three nested loops and following a procedure similar to the one of Ben-Ameur [29]. More precisely, the initialization procedure looks for the optimal temperature T_0 that maximizes the decrease of the energy function in a measuring window of width W_{iter}, i.e. between iteration 1 and W_{iter}. This is performed while launching several times the SA algorithm with different T_0 . For each T_0, the value of the energy function is tracked during the W_{iter} very first iterations of the SA algorithm, and the temperature which gives rise to the most important decrease of the energy function is selected. Actually, to smooth the effect of the stochastic behavior, the energy function is averaged while considering its values during the L_{iter} last iterations of the measuring window, i.e. between iterations ($W_{\text{iter}} - L_{\text{iter}}$) and W_{iter}. As a consequence, the initial temperature T_0 of the SA algorithm varies depending on the fitting scenarios.

In the following examples, selecting a measuring width of $W_{\text{iter}} = 25$ first iterations and considering the $L_{\text{iter}} = 5$ last iterations of this window to smooth the energy values have proved to be good trade-off between waiting a sufficient stabilization of the stochastic behavior, and avoiding too much iterations. Actually, these values must be compared to the 150 iterations usually required for the SA algorithm to converge, i.e. to reach a max number of iteration M_{iter} without any change of the energy function (up to a given accuracy ϵ_{iter}). Moreover, in the proposed implementation, the values of T_0 to be tested evolve between 5 and 50 with an increment of 5. Thus, ten simulations (stopped after W_{iter} iterations) are necessary to identify the best initial temperature for a given fitting example.

5.3 Global fitting of a symmetric L-like shape

The first example deals with the global fitting of a symmetric L-like shape controlled by five parameters ($N_a = 0$, $N_m = 1$ and $N_p = 5$): the lengths L and ℓ, the thickness e, the holes radius R and the size of the fillet r. This part has been rapidly sketched and parameterized in a pre-processing step. Three groups of parameters are defined according to the level of detail they correspond to (Fig. 5 and Table 1). This decomposition conforms to the classical CAD modeling strategy, which starts from structural and detail features, and ends up with skinning features. From the original CAD model, an as-scanned point cloud has been created using the HPR algorithm from 6 viewpoints. The point cloud is composed of about 500k points. At first, no noise has been added. As it is a global fitting, no segmentation is performed and the optimization loop is run on the three parameter groups. At each iteration, the ICP algorithm is run to control the position and orientation of the updated part with respect to the point cloud.

Following the initialization procedure detailed in Sect. 5.2, the initial temperature T_0 of the SA algorithm is set up to 10 for the L-like shape fitting example. This is the value of T_0 that maximizes the decrease of the energy function in the very first iterations. The algorithm stops when a max number of iteration ($M_{\text{iter}} = 50$) without any change (up to an accuracy $\epsilon_{\text{iter}} = 10^{-1}$) is reached. The algorithm runs for the successive segmentations (segmentation loop), and for the three groups of parameters (parameters loop). The graphics of Fig. 6 show the SA evolution curves for the three parameters groups $\mathcal{G}_j$ of the last segmentation loop ($i = 3$). The group $\mathcal{G}_1$ is optimized first, then $\mathcal{G}_2$, and $\mathcal{G}_3$ at last. The stochastic nature of the SA algorithm is clearly visible. Such an algorithm is interesting to solve this global optimization problem in a large search space. Of course, in scenarios where the final parameters values are to be rounded, the minimization process could stop earlier while acting on the value of ϵ_{iter}.

Figure 5 shows the fitted L-like shape and the numerical results are gathered together in Table 1. The average

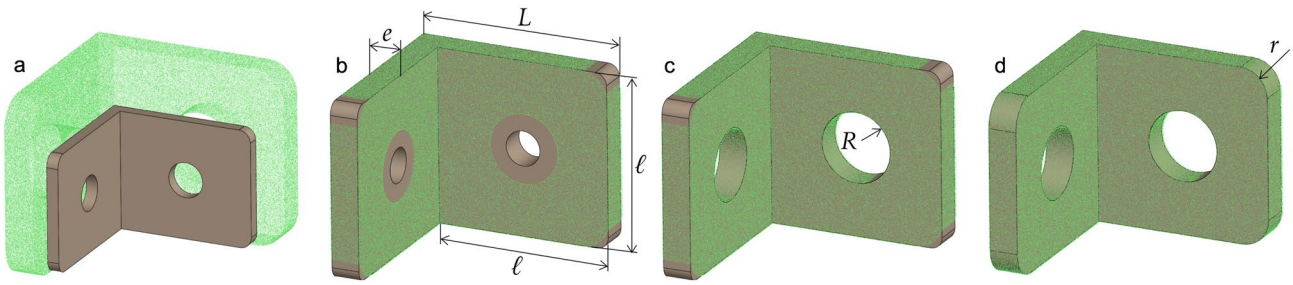


Fig. 5 Global fitting of a L-like shape following three optimization loops: **a** coarse pre-arrangement in the initial point cloud; **b** loop on $\mathcal{G}_1$; **c** loop on $\mathcal{G}_2$; **d** final fitted part after a loop on $\mathcal{G}_3$

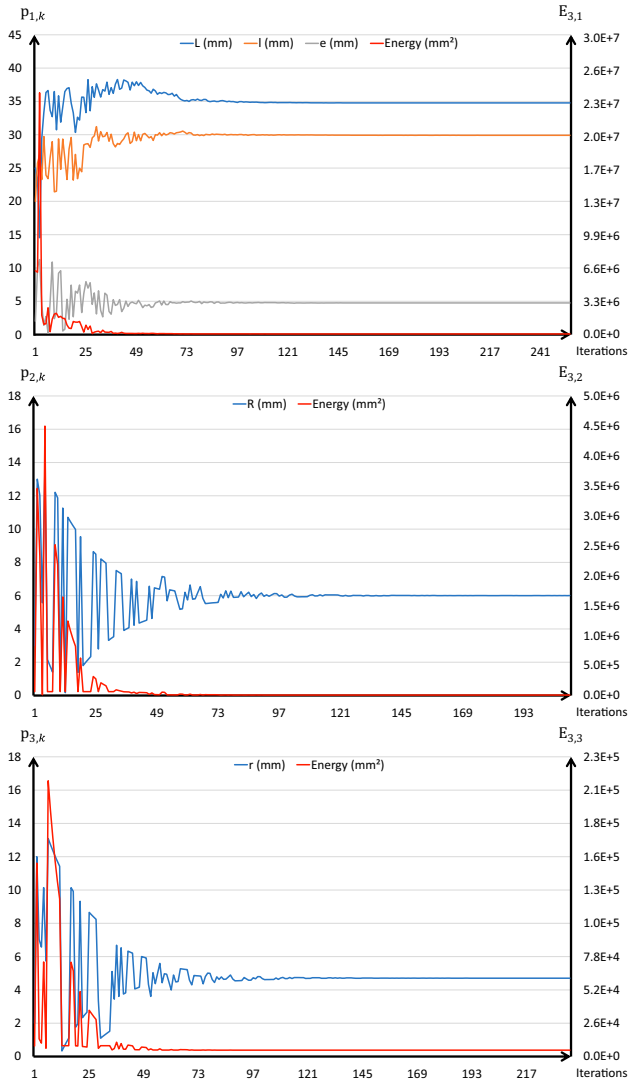


Fig. 6 Simulated annealing evolution curves for the three parameters loops ($j \in \{1, \ldots, 3\}$) of the last segmentation step ($i = 3$) during the global fitting of a L-like shape: parameters values evolution (left) and energy evolution (right)

absolute deviation between the point cloud and the CAD model is about 0.1133 mm (min = 0.0 mm, max = 0.3907 mm, std = 0.0495 mm) and the average relative deviation is about 0.0034 (when using the average of the three dimensions of the oriented bounding box as a reference distance), which is quite low. The relative and absolute deviations between the parameters final values p_k^F and the ones of the original part p_k^D are low. Actually, the largest deviations are for parameters e and r. This is due to the fact that those parameters are controlling relatively small features whose size variations have a low impact on the energy function to be minimized. This effect is more deeply analyzed in Sect. 5.4 where alternative solutions are sketched. Using this part-by-part fitting strategy, the final L-like shape better fits the point cloud than if a traditional patch-by-patch reconstruction process would have been followed. Indeed, the L-like shape is fitted globally and the final solution does not depend on a reference surface which would have been fitted individually at the beginning of the manual reconstruction process. Moreover, internal constraints (e.g. symmetry, perpendicularity, holes centered on the faces) are directly managed by the CAD modeler, which handles the successive updates and maintains the CAD model consistency step after step. Finally, if the parameters would not have been distributed in the three groups, the results would have been less good. Indeed, when considering the five parameters at the same level, the average absolute deviation is about 0.2636 mm (min = 0.0 mm, max = 2.2474 mm, std = 0.4264 mm) and the average relative deviation is about 0.0080, and both values are higher than when considering the parameters within the three groups. This validates the proposed decomposition strategy.

Another experimentation has been performed to study the impact of the presence of noise on the fitting results. To this aim, a noise has been added to the whole point cloud using a Gaussian noise random distribution controlled by an amplitude factor. Results of the fitting process with two levels of noise (50 µm and 100 µm) are listed in Table 2. One can clearly observe that as the noise increases, the relative deviation also increases coherently. This effect is more significant for small features. Obviously, the levels of noise which have been applied are far greater than what can appear on a well-calibrated acquisition device used in normal conditions. Thus, the proposed approach is relatively stable to noise, except for small detail features which can hardly be well fitted in the presence of too much noise. But in this case, following a more conventional manual reconstruction process

Table 1 Results for the global fitting of a L-like shape (without noise and with an initial temperature of the SA algorithm $T_0 = 10$)

Groups	p_k	p_k^0 (mm)	p_k^D (mm)	p_k^F (mm)	δp_k (mm)	Δp_k
$\mathcal{G}_1$	L	25	35	34.7800	0.2200	0.0063
	ℓ	20	30	29.9210	0.0790	0.0026
	e	2	5	4.7514	0.2486	0.0497
$\mathcal{G}_2$	R	3	6	6.0045	− 0.0045	0.0008
$\mathcal{G}_3$	r	2	5	4.7061	0.2939	0.0588

would lead to the same difficulties. Anyhow, depending on the adopted fitting scenario, the obtained parameter values may also be rounded at the end.

5.4 Global fitting of a sonotrode to a real scanned point cloud

This second example studies the behavior of the fitting algorithm when dealing with real scanned data. This time, a ROMER Absolute Arm 7520 SI (7 axis, 2m acquisition volume, absolute encoders, RSI laser sensor 30000pts/s) has been used to scan a Sonotrode (Fig. 7a) and get a point cloud of about 257k points (Fig. 7b). The resulting point cloud incorporates noise and artifacts. The CAD model of the Sonotrode to be fitted is controlled by ten parameters split in three groups ($N_a = 0$, $N_m = 1$ and $N_p = 10$): structural features ($\mathcal{G}_1$), geometric features ($\mathcal{G}_2$) and skinning features ($\mathcal{G}_3$). Following the initialization procedure of Sect. 5.2, the initial temperature T_0 of the SA algorithm is set up to ten for the Sonotrode example, and the other control parameters are the same as for the previous example ($M_{\text{iter}} = 50$ and $\epsilon_{\text{iter}} = 10^{-1}$). The fitting process starts by the pre-arrangement of the CAD model with respect to the reference point cloud (Fig. 7b). This initializes the fitting algorithm, which then starts by optimization the structural parameters of $\mathcal{G}_1$ (Fig. 7c). For this loop, the holes D_3 and D_4 have been deactivated from the building tree. Doing this way, the fitting of D_1 is not affected by the internal points related to the holes, and the corresponding areas of the point cloud can be cropped through the segmentation loop. The parameters of $\mathcal{G}_2$ are then optimized (Fig. 7d). The axes of the holes are constrained using built-in constraints (i.e. perpendicularity, coaxiality and circular repetition) of the CAD modeler in charge of the updates. Due to the occlusion phenomenon, there are too few points to accurately identify the depth of the holes, which, therefore, has not been considered as a parameter to be optimized. As part of the last optimization loop, the parameters of $\mathcal{G}_3$ are then optimized to get the final fitted CAD model of the Sonotrode (Fig. 7e). The numerical results are synthesized in Table 3.

This fitting problem is more complex and the difficulties are threefold: (i) the number of variables is greater than in the previous case, resulting in more iterations; (ii) some features are quite small when compared to the size of the object, and their contribution within the energy function becomes negligible, thus causing greater deviations of the corresponding parameters

Table 2 Evolution of the relative deviation Δp_k with respect to the amplitude of the inserted noise (0 µm, 50 µm, 100 µm) for the global fitting of the L-like shape

Groups	p_k	Relative deviation Δp_k		
		0 µm	50 µm	100 µm
$\mathcal{G}_1$	L	0.0063	0.0068	0.0187
	ℓ	0.0026	0.0000	0.0099
	e	0.0497	0.0553	0.1315
$\mathcal{G}_2$	R	0.0008	0.0018	0.0192
$\mathcal{G}_3$	r	0.0588	0.0327	0.1332

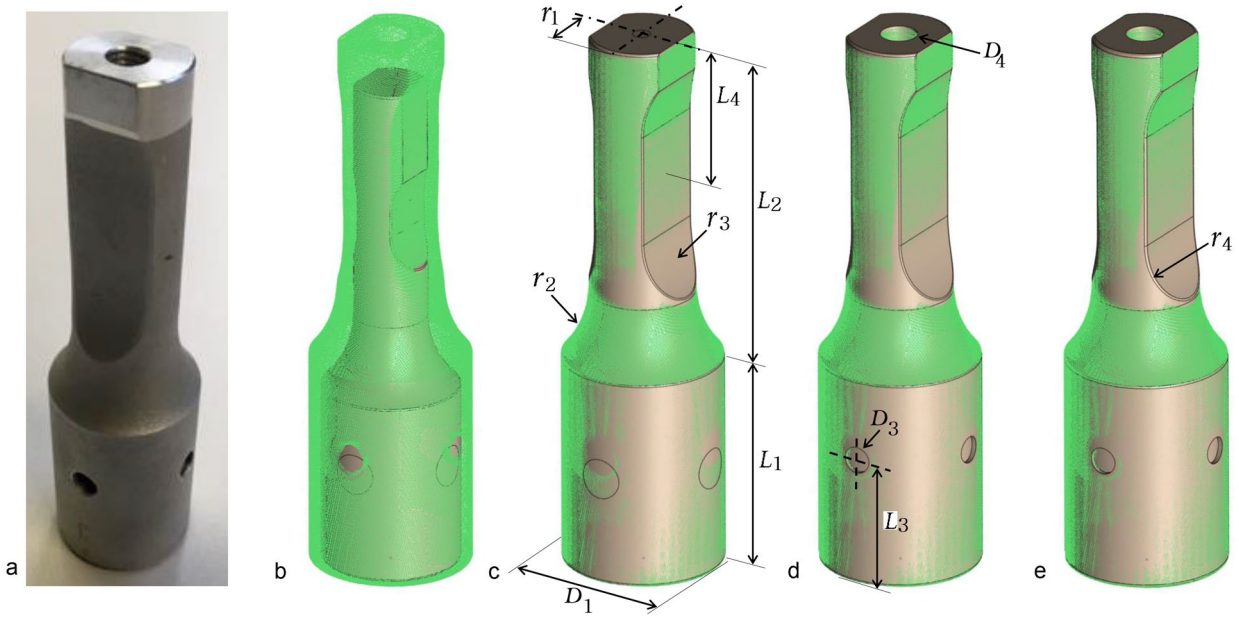


Fig. 7 Global fitting of a Sonotrode to a real scanned point cloud following 3 optimization loops: **a** scanned Sonotrode; **b** coarse pre-arrangement in the initial point cloud; **c** loop on $\mathcal{G}_1$; **d** loop on $\mathcal{G}_2$; **e** final fitted part after a loop on $\mathcal{G}_3$

Table 3 Results for the global fitting of a Sonotrode to a real point cloud obtained by a laser scanner

Groups	p_k	p_k^0 (mm)	p_k^D (mm)	p_k^F (mm)	δp_k (mm)	Δp_k
$\mathcal{G}_1$	D_1	30	35	34.9872	0.0128	0.0004
	L_1	40	47	47.6183	− 0.6183	0.0132
	L_2	70	75	74.0385	0.9615	0.0128
	r_1	7	11.5	11.4656	0.0344	0.0030
	r_2	22	24	21.1645	2.8355	0.1181
	r_3	12	20	20.0028	− 0.0028	0.0001
$\mathcal{G}_2$	D_3	9	6	6.6348	− 0.6348	0.1058
	D_4	4	8	7.9101	0.0899	0.0112
	L_3	27	30	29.7042	0.2958	0.0099
$\mathcal{G}_3$	r_4	0.5	1.5	0.8376	0.6624	0.4416

values (e.g. parameter r_4 in $\mathcal{G}_3$); (iii) due to the occlusion phenomenon, the internal holes cannot be well captured by the acquisition device, resulting in difficulties to get a good fitting of the internal holes' parameters (i.e. parameters in $\mathcal{G}_2$ and depth of the holes). Despite those difficulties, the average absolute deviation between the point cloud and the CAD model of the Sonotrode is about 0.0622 mm (min = 0.0 mm, max = 5.8095 mm, std = 0.3536 mm) and the average relative deviation is about 0.0010, which is quite low. The larger values of min and max can directly be ascribed to the issue (iii).

This example shows that the size of the features clearly influences the quality of the fitting. This is directly linked to the contribution of those features within the overall energy function to be minimized. This issue could be overcome while considering a particular weighting strategy for each feature of the CAD model to be fitted. As a consequence, in a pre-processing step, the parameterized CAD model could be used to compute the sensitivity of each parameter p_k to shape variations, and then infer the proper distribution of the weights. This is further discussed in the conclusion.

5.5 Local fitting of robot arms

The third example illustrates how the fitting approach can be used to track successive moves of robot arms, and thus update the digital twin of a physical system (Fig. 8). The robot is supposed to be stopped during the update of its digital twin. Such a possibility is particularly interesting to maintain the coherence between the physical system and its digital twin in the scope of the Industry 4.0 [1].

The DMU of the considered robot is composed of three arms and one fixed support ($N_\mathrm{m} = 3$). The orientation of the arms with respect to the others is parameterized by three angles α_i (with $i \in [1 \dots 3]$) gathered in a single group $\mathcal{G}_1$ (Fig. 8a). The shape of the arms is not considered in this test case ($N_\mathrm{p} = 0$), which focuses on the global fitting of the assembly structure ($N_\mathrm{a} = 3$). Here, the objective is to be able to retrieve the values of the arms' rotation parameters after three moves of the robot. To this aim, three as-scanned point clouds have been created using the HPR algorithm from six viewpoints, thus generating a bunch of 614 k/520 k/410 k points for, respectively, the first, second and third move. Following the initialization procedure of Sect. 5.2, the initial temperature T_0 of the SA algorithm is set up to 25 for the robot arms example, and the other control parameters

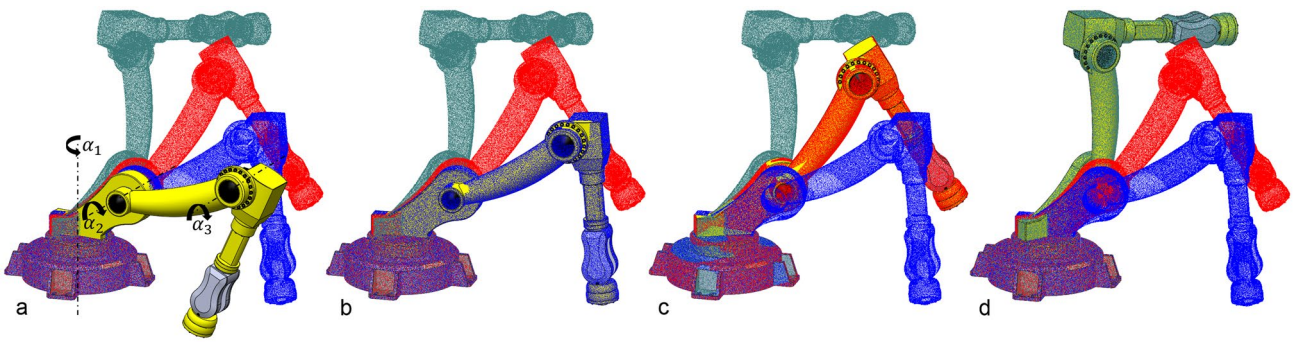


Fig. 8 Fitting of robot arms constrained with assembly constraints and parameterized by 3 rotation angles: **a** initial configuration; **b**–**d** final configurations fitting respectively the first, second and third robot moves

are the same as for the previous example ($M_{\text{iter}} = 50$ and $\epsilon_{\text{iter}} = 10^{-1}$). Starting from an initial configuration, the fitting algorithm is sequentially called three times so that the robot arms successively fit the three virtually generated point clouds (Fig. 8b–d).

The numerical results are provided in Table 4. It clearly shows that the parameters' values obtained after a fitting serves as new initial values for the next fitting. As in the previous test cases, the relative deviations of the control parameters are very low. In the worst case, i.e. for the move that gives the largest deviation, the average absolute deviation between the point cloud and the CAD models is about 0.0068 mm (min = 0.0 mm, max = 5.5210 mm, std = 0.1071 mm) and the average relative deviation is about 0.00001, which is also quite low. Actually, when only dealing with assembly parameters only, the fitting process is very efficient as it does not require CAD models updates. This demonstrates the pertinence of the proposed technique to tackle such a fitting scenario in the scope of the Industry 4.0. This is a first step towards the accurate tracking of robots evolving in complex industrial environments.

5.6 Local fitting of multiple parts in a valve assembly

The last example illustrates how the fitting strategy can be used to update the CAD assembly model of a valve. Here, it is assumed that the CAD models already exist and need to be updated. To be able to validate the approach, from the original DMU made of 40 assembled parts, an as-scanned point cloud has been created using the HPR algorithm from 10 viewpoints. The resulting point cloud is composed of 1204 k points. No noise has been added. The reconstruction process follows a part-by-part fitting strategy where several parts can be fitted simultaneously, while also satisfying assembly constraints. Each part is defined by several control parameters, whose values are to be optimized (Table 5). In this scenario, the parts with the biggest extent are fitted first so as to get a good initial fitting, which will then serve as a reference for the fitting of the other parts. However, as it will be explained, during the successive fittings, parameters of previously fitted parts can still be reconsidered as variables for the upcoming fitting steps. Indeed, considering more points step after step can improve the fitting accuracy of previously fitted parts. Of course, if the final objective is to control possible misalignments between assembled parts, assembly constraints should not be specified so as to let the algorithm capturing the as-is assembly configuration, but this is not the considered scenario here. The SA control parameters are tuned as for the previous examples ($M_{\text{iter}} = 50$ and $\epsilon_{\text{iter}} = 10^{-1}$), except for the initial temperature T_0 that is tuned differently for each of the parts to be fitted.

At first, after a pre-arrangement of the two flanges in the point cloud, the initial temperature T_0 of the SA algorithm is set up to 15 while following the initialization procedure of Sect. 5.2. The two flanges are fitted simultaneously using eight parameters to control the shape of the flanges and one parameter to control the distance between the two flanges ($N_{\text{m}} = 2$, $N_{\text{p}} = 8$ and $N_{\text{a}} = 1$). Additional assembly constraints are also used and directly handled by the CAD modeler in charge of the successive updates (Fig. 9.a_1 and a_2): the axes of the cylindrical faces of radius r_1 have to be coincident (orange color), so do the axes of the through holes of radius r_5. Overall, the deviations between the final parameters values and the ones of the original part are low. Actually, the largest deviations are for parameters r_5 and r_6. This is due to the fact that those radii are rather small and very few points can contribute to the fitting. Anyhow, a more conventional reverse engineering process would lead to the same issue. Since the two flanges are fitted simultaneously, the resulting deviations are smaller than if the flanges would have been fitted sequentially. This is due to the fact that the number of points and their distribution in the 3D space make this configuration more stable than when fitting the flanges one after the other.

Once the two flanges fitted, the update process keeps on going with the simultaneous fitting of two identical screws defined by four control parameters (Fig. 9.b_1 and b_2). For this fitting step, the parameter r_4 of the flanges is reconsidered as a variable that can be further optimized during the fitting of

Table 4 Results obtained when fitting three successive moves of robot arms parameterized by three rotation angles

Groups	p_k	p_k^0 (mm)	p_k^D (mm)	p_k^F (mm)	δp_k (mm)	Δp_k
$\mathcal{G}_1$ Move-1	α_1	75	90	89.9990	0.0010	0.0000
	α_2	15	30	29.9946	0.0054	0.0002
	α_3	65	95	94.9968	0.0032	0.0000
$\mathcal{G}_1$ Move-2	α_1	89.9990	115	114.9987	0.0013	0.0000
	α_2	29.9946	55	54.9954	0.0046	0.0001
	α_3	94.9968	120	119.9976	0.0024	0.0000
$\mathcal{G}_1$ Move-3	α_1	114.9987	140	139.9985	0.0015	0.0000
	α_2	54.9954	90	89.9935	0.0065	0.0001
	α_3	119.9976	170	170.0019	− 0.0019	0.0000

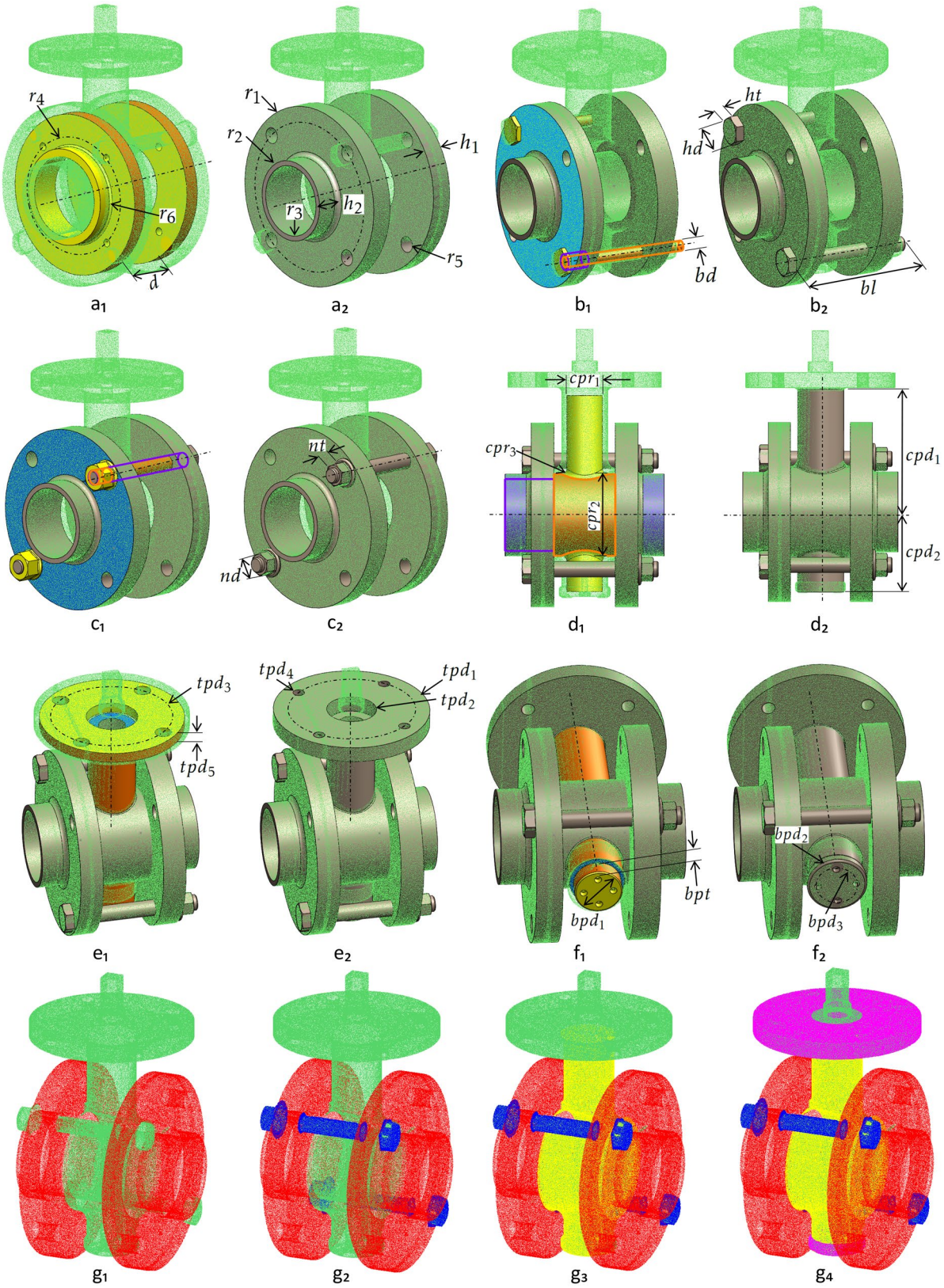
the screws. Thus, the SA algorithm tries to find the optimal values of five parameters (four parameters of the screws, and one parameter of the flanges). The pre-arrangement is performed while using the axes of the previously fitted flanges. Each screw is constrained to have its axis coincident with the axis of the flange's through hole (orange and purple colors), and the front planar face of the flange should be coincident with the contact face of the screw head (blue color). Following the initialization procedure of Sect. 5.2, the initial temperature T_0 of the SA algorithm is set up to 10 for the screws fitting. Here again, as the two screws are fitted simultaneously, the final parameters deviations are better controlled. Similarly, the two nuts are then fitted based on the previously fitted screws and flanges (Fig. 9.c_1 and c_2). Each nut is parameterized by two control parameters and it is constrained to have its axis coincident with the axis of the screw (orange and purple), and the bottom face of the nut should be coincident with the planar face of the flange (blue color). For this fitting step, the parameter r_4 of the two flanges and the diameter *bd* of the screws are reconsidered as variables that can be further optimized during the fitting of the nuts. Thus, the SA algorithm tries to find the optimal values of four parameters (two parameters of the nuts, one parameter of the flange and 1 parameter of the screw). In addition, the diameter *bd* of both the screws and nuts are linked to the radius r_5 of the fitted flanges. Here, the initial temperature T_0 of the SA algorithm is set up to 5 for the nuts fitting.

Following the part-by-part fitting strategy, the central part is then fitted while optimizing the values of five control parameters (Fig. 9.d_1 and d_2). Using the same coloring strategy, the axes of the two parts are constrained to be coincident, and the lateral contact faces of the central part are constrained to be coincident with the back faces of the two flanges. Here, the initial temperature is set up to 5. Then, the top plate is fitted to the point cloud. It is parameterized by five control parameters and aligned with the central part using coincidence constraints on both the axes and contact faces (Fig. 9.e_1 and e_2). The initial temperature is set up to 10 for the top plate. For this fitting step, the parameter cpd_1 of the central part is reconsidered as a variable to be further optimized. Thus, the SA algorithm tries to find the values of six parameters (five parameters of the top plate, and one parameter of the central part). Finally, the bottom plate is fitted to the point cloud. It is parameterized by four control parameters and aligned with the central part with which it is also in contact (Fig. 9.f_1 and f_2). For this fitting step, the parameter cpd_2 of the central part is reconsidered as a variable to be further optimized. Thus, the SA algorithm tries to find the values of five control parameters (4 parameters of the bottom plate, and one parameter of the central part). Here, the initial temperature is set up to 5 for the bottom plate.

This example clear shows the potential of the proposed part-by-part fitting strategy to update an entire CAD assembly model so that it fits the point cloud of a digitized assembly. The method is also able to segment the point cloud of the digitized valve so as to highlight the different parts (Fig. 9.g_1 to g_4). The relative deviations of the control parameters are quite low (Table 5), except for the radius of some fillets and for small features for which larger deviations can be ascribed to a lack of data in the corresponding areas of the point cloud. This has been discussed for the previous experiments. Here again, depending on the adopted scenario, the obtained parameter values may be rounded at the end. Of course, being the screws and nuts standard elements, the obtained parameters values can serve as a reference to select in a catalog the closest available ones. Finally, Table 6 gathered together the absolute and relative errors measured between the point cloud and each fitted part. Those errors are rather small when compared to the size of the parts.

6 Conclusion and future works

This paper has introduced a new framework able to fit simultaneously several parameterized CAD models in the point cloud of a digitized assembly. CAD models to be fitted can also be constrained with assembly constraints. Both the consistency of the CAD models and the constraints between them are managed by the CAD modeler which acts every time the parameters are modified. The resulting CAD models can directly be used and edited in the stages of the PDP. When considering local fitting, the initial point cloud is also segmented at the end of the process. The proposed approach is very promising when considering the need to maintain the coherence between a physical system and its digital twin in the scope of the Industry 4.0. It bypasses the traditional tedious and time-consuming patch-by-patch reverse engineering process. It makes use of a simulated annealing algorithm to find the optimal values in a large search space. The framework has been designed around three nested loops which act at different levels: the segmentation loop that isolates step after step the subset of the original point cloud to which the CAD models have to be fitted; the parameters loop which splits the optimization problem according to three categories of parameters, similar to what is traditionally used when modeling CAD parts and assemblies in a CAD environment; and the optimization loop which runs the simulated annealing algorithm on the subproblems. The method has proved its efficiency for global and local fitting. It has been tested on several configurations, for which as-scanned point clouds have been generated following an ad hoc virtual scanning approach. Thus, it has been possible to measure the deviations between the parameters of the fitted parts, and the ones of the original parts as they appear in the DMU. The


r4
r6
d
a1
r1
r2
h1
r3
h2
r5
a2
bd
b1
ht
hd
bl
b2
nt
nd
c1
c2
cpr1
cpr3
cpr2
d1
cpd1
cpd2
d2
tpd3
tpd5
e1
tpd4
tpd1
tpd2
e2
bpt
bpd1
f1
bpd2
bpd3
f2
g1
g2
g3
g4

◂**Fig. 9** Successive fittings of multiple CAD models in the point cloud of a digitized valve assembly: (a_1) coarse pre-arrangement of two flanges in the initial point cloud; (a_2) final fitted flanges; (b_1) pre-arrangement of two screws; (b_2) final fitted screws; (c_1) pre-arrangement of two nuts; (c_2) final fitted nuts; (d_1) pre-arrangement of the central part; (d_2) final fitted central part; (e_1) pre-arrangement of the top plate; (e_2) final fitted top plate; (f_1) pre-arrangement of the bottom plate; (f_2) final fitted bottom plate; (g_1 to g_4) successive segmentations of PC_0

influence of the noise and of the parameters grouping on the final results have also been analyzed. The experimentations have been performed on mechanical parts assemblies, but the method could be extended to allow the more general local fitting of parameterized objects in virtual environments. A first attempt to define a tracking approach able to maintain the coherence between a physical system and its digital twin has proved to be promising, and it already gives good results on a simple 3-axes robot.

The various experimentations allowed to demonstrate several interesting features of the approach, as well as ways of improvement. Among others, it is clearly demonstrated that for the local fitting, a sufficient amount of information has to be accessible. This is of course very similar to the traditional manual reconstruction process, and the proposed approach cannot capture the invisible. This is notably true to fit small features as well as to capture interfaces between parts. Thus, few disassembly steps might be required before scanning. The exploitation of RANSAC-based, or even manually, extracted primitives to be used as constraints of the fitting problem could help solving this issue. The integration of this fitting technique as part of a reverse engineering framework might also help speeding the entire reconstruction process in case no parameterized model exists. However, this requires some adaption so as to be able to work with parameters of 2D sketches and 3D features, and thus to allow reconstructing a full CAD model step after step

Table 5 Results for the local fitting of multiple parts in a valve assembly

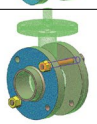
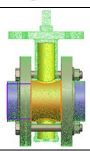
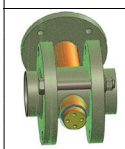
Pre-arranged parts(s)	Fitted part(s)	Groups	p_k	Initial values p_k^0 (mm)	Target values p_k^D (mm)	Final values p_k^F(mm)			Absolute dev. δp_k(mm)			Relative dev. Δp_k
						i=1	*i=2*	*i=3*	*i=1*	*i=2*	*i=3*	*i=3*
		Flanges										
		$\mathcal{G}_1$	r_1	80	90	90.0697	90.0697	90.171	-0.0697	-0.0697	-0.171	0.0019
			r_2	38	40	39.8267	39.8267	40.1883	0.1733	0.1733	-0.1883	0.0047
			r_3	30	35	34.4396	34.4396	34.9131	0.5604	0.5604	0.0869	0.0025
			h_1	15	22	23.8543	23.8543	22.2166	-1.8543	-1.8543	-0.2166	0.0098
			h_2	15	25	24.934	24.934	24.9712	0.066	0.066	0.0288	0.0012
			d	70	60	57.3323	57.3323	59.7876	2.6677	2.6677	0.2124	0.0035
		$\mathcal{G}_2$	r_4	60	75	74.9759	74.9759	74.9759	0.0241	0.0241	0.0241	0.0003
			r_5	5	8	7.4503	7.4503	7.4503	0.5497	0.5497	0.5497	0.0687
		$\mathcal{G}_3$	r_6	2	5	0.3446	3.7187	4.5439	4.6554	1.2813	0.4561	0.0912
		Screws										
		$\mathcal{G}_1$	hd	21	24	24.7793	24.7793	24.3001	-0.7793	-0.3001	-0.3001	0.0125
			ht	5	8	8.9433	8.9433	8.895	-0.9433	-0.9433	-0.895	0.1119
			bl	130	121	139.959	139.959	120.1898	-18.959	-18.959	0.8102	0.0067
			bd	10	14	14.3687	14.3687	14.2536	-0.3687	-0.3687	-0.2536	0.0181
			r_4	74.9759	75	74.9225	74.9225	74.9684	0.0775	0.0775	0.0316	0.0004
		Nuts										
		$\mathcal{G}_1$	r_4	74.9684	75	74.9868	74.9807	75.0393	0.0132	0.0193	-0.0393	0.0005
			bd	14.2536	14	14.4136	14.4408	14.7502	-0.4136	-0.4408	-0.7502	0.0536
			nd	25	21	21.052	20.7961	20.8564	-0.052	0.2039	0.1436	0.0068
			nt	15	12	14.0638	14.5033	14.807	-2.0638	-2.5033	-2.807	0.2339
		Central part										
		$\mathcal{G}_1$	cpd_1	117	125	118.052	123.99	123.8636	6.948	1.01	1.1364	0.0091
			cpd_2	75	70	81.1333	79.7596	80.1172	-11.1333	-9.7596	-10.1172	0.1445
			cpr_1	35	45	45.0812	46.0554	45.8565	-0.0812	-1.0554	-0.8565	0.0190
			cpr_2	80	90	87.2774	89.4893	89.7125	2.7226	0.5107	0.2875	0.0032
			cpr_3	5	10	0.5925	0.0008	5.3119	9.4075	9.9992	4.6881	0.4688
		Top plate										
		$\mathcal{G}_1$	tpd_1	140	150	145.1966	150.2271	150.0571	4.8034	-0.2271	-0.0571	0.00038
			tpd_2	50	45	40.8289	38.9789	40.4789	4.1711	6.0211	4.5211	0.10047
			tpd_3	115	120	115.4108	115.746	118.9028	4.5892	4.254	1.0972	0.00914
			tpd_4	16	12	6.4896	9.3327	11.4995	5.5104	2.6673	0.5005	0.04171
			tpd_5	10	15	15.0677	15.1752	15.1411	-0.0677	-0.1752	-0.1411	0.00941
			cpd_1	123.8636	125	124.6546	124.7564	124.807	0.3454	0.2436	0.193	0.00154
		Bottom plate										
		$\mathcal{G}_1$	bpd_1	40	50	47.9315	48.3134	49.0784	2.0685	1.6866	0.9216	0.0184
			bpd_2	1	6	0.046	0.1388	4.1624	5.954	5.8612	1.8376	0.3063
			bpd_3	25	35	28.0237	30.7956	33.1279	6.9763	4.2044	1.8721	0.0535
			bpt	15	10	17.2641	17.0953	18.8609	-7.2641	-7.0953	-8.8609	0.8861
			cpd_2	67	70	62.9326	63.1342	61.35	7.0674	6.8658	8.65	0.1236

Table 6 Absolute and relative errors measured between the point cloud and each locally fitted part of the valve assembly

Parts	Absolute error (mm)					Relative error				
	Min	Max	Median	Mean	Dev.Std.	Min	Max	Median	Mean	Dev.Std.
Flanges	− 1.9720	1.9065	− 0.0067	− 0.0118	0.2056	0.0115	0.0111	0.0000	0.0001	0.0012
Screws	− 1.0178	1.2258	0.0781	0.0290	0.4098	0.0078	0.0093	0.0006	0.0002	0.0031
Nuts	− 2.0938	2.1350	0.0741	0.0582	0.4286	0.0156	0.0159	0.0006	0.0004	0.0032
Central Part	− 2.1930	2.1742	0.2658	0.3320	0.5730	0.0186	0.0185	0.0023	0.0028	0.0049
Top plate	− 1.0940	2.0927	− 0.0030	0.0099	0.2063	0.0104	0.0199	0.0000	0.0001	0.0020
Bottom plate	− 2.2872	3.7552	− 0.0526	0.0868	0.7390	0.0586	0.0963	0.0013	0.0022	0.0189

following a feature-by-feature reconstruction strategy. This is part of future works.

The proposed approach is modular and each module can still be improved separately. As already explained, the energy function could be weighed to better reflect the influence of the different parameters on the shape variations. The weights could rely on a sensitivity analysis run at the beginning of the fitting process. Moreover, the current implementation is yet at the level of the prototype and it does not fully allow for real-time fitting. This is mainly due to the modularity of the implementation that calls external modules that need to exchange with the optimization kernel. Thus, a better integration of the different modules can drastically reduce the time spent to read/write/update the data in the various data structures of the different modules. In addition, in the current version, the thresholds used to tune the accuracy of the final results have been set up to quite low values so as to get very good fitting results. Of course, depending on the scenarios and parts to be fitted, a good trade-off between accuracy and time has to be found. Thus, the thresholds could be increased and the final results rounded. This is particularly true for instance for the standard parts that will anyhow be selected in a catalog and defined with predefined parameters values (e.g. screws, nuts). Parallelization can also be a good mean to speed up the overall process, while considering for instance that the parameters loops are executed step after step while letting the user free to interact and continue his/her work on the already fitted parts.

Finally, the use of parameterized CAD models as priors is a promising idea to avoid learning the parameters from the data as there are explicitly encoded in the CAD models. More generally, global regularity priors (e.g. symmetries, repetitions, relationships) are directly encoded in the CAD models and can thus be exploited when solving the fitting problem. Clearly, more domain-dependent knowledge still remain to be exploited to better support advanced treatments of geometric models. This could also help better solving the tracking of systems evolving in complex environments.

References

1. Lu Y (2017) Industry 4.0: a survey on technologies, applications and open research issues. J Ind Inf Integr 6:1–10
2. Bagci E (2009) Reverse engineering applications for recovery of broken or worn parts and re-manufacturing. Adv Eng Softw 40(6):407–418
3. Falcidieno B, Giannini F, Léon J-C, Pernot J-P (2014) Processing free form objects within a product development process framework. In: Michopoulos JG, Paredis CJJ, Rosen DW, and Vance JM (eds) Advances in Computers and Information in Engineering Research, Vol 1. in ASME-Press, pp 317–344
4. Berger M, Tagliasacchi A, Seversky LM, Alliez P, Guennebaud G, Levine JA, Sharf A, Silva CT (2016) A survey of surface reconstruction from point clouds. Comput Gr Forum 36(1):301–329
5. Fischler MA, Bolles RC (1981) Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun ACM 24(6):381–395
6. Schnabel R, Wahl R, Klein R (2007) Efficient ransac for point-cloud shape detection. Comput Gr Forum 26(2):214–226
7. Schnabel R, Degener P, Klein R (2009) Completion and reconstruction with primitive shapes. Comput Gr Forum 28(2):503–512
8. Bey A, Chaine R, Marc R, Thibault G (2012) Effective shapes generation for bayesian cad model reconstruction. In: Proceedings of the 5th eurographics conference on 3D object retrieval, pp 63–66
9. Lari Z, Habib A (2014) An adaptive approach for the segmentation and extraction of planar and linear/cylindrical features from laser scanning data. ISPRS J Photogramm Remote Sens 93:192–212
10. Attene M, Falcidieno B, Spagnuolo M (2006) Hierarchical mesh segmentation based on fitting primitives. Vis Comput 22(3):181–193
11. Mitra N, Wand M, Zhang HR, Cohen-Or D, Kim V, Huang Q-X (2013) Structure-aware shape processing, In: SIGGRAPH Asia 2013 courses, pp 1:1–1:20
12. Li Y, Wu X, Chrysathou Y, Sharf A, Cohen-Or D, Mitra NJ (2011) Globfit: consistently fitting primitives by discovering global relations. ACM Trans Gr 30(4):52:1–52:12
13. Monszpart A, Mellado N, Brostow GJ, Mitra NJ (2015) Rapter: rebuilding man-made scenes with regular arrangements of planes. ACM Trans Gr 34(4):103:1–103:12
14. Nan L, Xie K, Sharf A (2012) A search-classify approach for cluttered indoor scene understanding. ACM Trans Gr 31(6):137:1–137:10
15. Ip CY, Gupta SK (2007) Retrieving matching cad models by using partial 3d point clouds. Comput Aid Des Appl 4(5):629–638
16. Gelfand N, Mitra NJ, Guibas LJ, Pottmann H (2005) Robust global registration. Symp Geom Process 2(3):197–206
17. Rabbani T, Van Den Heuvel F (2004) Methods for fitting csg models to point clouds and their comparison. In: Proceedings of

the 7th IASTED international conference on computer graphics and imaging, Kauai, HI, USA 1719, pp 279–284
18. Buonamici F, Carfagni M, Furferi R, Governi L, Lapini A, Volpe Y (2018) Reverse engineering of mechanical parts: a template-based approach. J Comput Des Eng 5(2):145–159
19. Wang J, Gu D, Yu Z, Tan C, Zhou L (2012) A framework for 3d model reconstruction in reverse engineering. Comput Ind Eng 63(4):1189–1200
20. Stark R, Grosser H, Müller P (2013) Product analysis automation for digital mro based on intelligent 3d data acquisition. CIRP Ann Manuf Technol 62(1):123–126
21. Bénière R, Subsol G, Gesquière G, Le Breton F, Puech W (2013) A comprehensive process of reverse engineering from 3d meshes to cad models. Comput Aid Des 45(11):1382–1393
22. Xu M, Li M, Xu W, Deng Z, Yang Y, Zhou K (2016) Interactive mechanism modeling from multi-view images. ACM Trans Gr 35(6):236:1–236:13
23. Montlahuc J, Shah GA, Polette A, Pernot J-P (2019) As-scanned point clouds generation for virtual reverse engineering of cad assembly models. Comput Aid Des Appl 16(6):1171–1182
24. Lupinetti K, Pernot J-P, Monti M, Giannini F (2019) Content-based cad assembly model retrieval: survey and future challenges. Comput Aid Des 113:62–81
25. Gouaty G, Fang L, Michelucci D, Daniel M, Pernot J-P, Raffin R, Lanquetin S, Neveu M (2016) Variational geometric modeling with black box constraints and dags. Comput Aid Des 75:1–12
26. Pernot J-P, Michelucci D, Daniel M, Foufou S (2019) Towards a better integration of modelers and black box constraint solvers within the product design process. Ann Math Artif Intell 85(2):147–173
27. Besl PJ, McKay ND (1992) A method for registration of 3-d shapes. IEEE Trans Pattern Anal Mach Intell 14(2):239–256
28. Katz S, Tal A, Basri R (2007) Direct visibility of point sets. ACM Trans Gr 26(3):24:1–24:11
29. Ben-Ameur W (2004) Computing the initial temperature of simulated annealing. Comput Optim Appl 29:369–385