Contents lists available at SciVerse ScienceDirect

CIRP Annals Manufacturing Technology

Journal homepage: www.elsevier.com/locate/cirp

Mixed-initiative assembly planning combining geometric reasoning and constrained optimization

Csaba Kardos[a,b], József Váncza (1)[a,b]

[a] *Centre of Excellence in Production Informatics and Control, Institute for Computer Science and Control, Hungarian Academy of Sciences, Budapest, Hungary*
[b] *Department of Manufacturing Science and Engineering, Budapest University of Technology and Economics, Budapest, Hungary*

The paper presents a generic workflow of semi-automated and optimized process planning in mechanical assembly. It supports the production engineer throughout the entire planning process, departing from part analysis via task sequencing up to the generation of detailed work instructions. Main stages such as part analysis, macro planning and various aspects of micro planning are presented along with a feedback mechanism which warrants executability of plans using the available technological and human resources. Emphasis is set on the essential role of geometric reasoning and its combined use with constrained optimization. The workflow is demonstrated on industrial case studies.

Assembly, Planning; Optimization

1. Introduction

Assembly planning is facing the problem of arranging *objects in space* and *actions in time* so that products specified by design can be realized in production. The space is densely populated, not only by parts of the product, but also by the applied technological resources, whereas key objectives of production require the execution of actions within as short a time frame as possible. Objects and actions involved in mechanical assembly are strongly related and constrain each other in many ways, due to technology, product structure and geometry, after all [1]. In any domain, the solution of this "puzzle" of production engineering provides essential input for designing the structure and planning the operation of assembly cells and lines [2], capacity- and production planning, scheduling and controlling the operation of assembly systems [3], selecting and designing fixtures and grasping devices [4], detailed path and motion planning [5][6], generation of robot programs and work instructions [3], training [7], product personalization [8], and feedback to product design. In many cases, an assembly by disassembly approach is taken [5], opening thus important application potentials towards de- and remanufacturing [9].

The main, generally accepted *requirements* towards assembly planning are the following: planning should depart from a generic CAD model of the product and capture and comply with all relevant constraints of the product, the actual assembly technology, and the resource base (typically fixtures, tools, human and robot operators) [2][5]. In the cramped world of assembly, the origin of most of these constraints is *geometric* by nature: parts, fixtures and tools should not only fit but also be movable along appropriate paths without collision. While the generated plans should meet the requirements of all stakeholders responsible for different aspects of plan execution, not only feasibility – in this case executability – but also optimality is to be warranted. However, a key epistemological prerequisite comes from admitting that both the geometric representation of the objects involved and the domain knowledge formalized may be *imperfect* and *incomplete*. Hence, any workflow should make possible the participation of engineers in problem solving and keep the time complexity of the planning process in bay.

The key *engineering principles* of resolving the above issues stood the test of time: *decomposition* exploiting *locality* leads almost unanimously to the use of *features* which specify tasks of assembling specific components with every possible technological detail [2][10]. *Hierarchical decomposition* concentrates the strongly interrelated combinatorial problems of setup planning, task sequencing and resource assignment into macro planning, and refers the handling of all other aspects of assembly like collision avoidance, tool trajectory, fixture design, etc., to micro planning [2]. However, any feature-based model is only a single interpretation of the design, where features are taken out of the context of the global planning problem. When put together again, local pieces of the plan can get easily into conflict [11]. Implications of the findings of micro-level planning (like the results of collision tests or path planning) should be enforced in macro planning, too [5]. In all the main decisions, the human planner should still have a say, calling for *mixed-initiative* solution approaches and advanced *visualization* [7].

The precursor of this work presented a decomposition approach for feature-based assembly planning, along with a mixed-integer linear programming (MILP) model for solving the integrated setup planning, task sequencing and resource assignment problem [11]. Technologies of placing, insertion, and screwing had been covered. The goal of this paper is to present the extension of the earlier model (while keeping its key concepts and notation) with new developments related to *geometric reasoning,* which is employed in the specification of the planning problem, in the validation of plans and in path planning. An overall *workflow* will also be presented which seamlessly integrates constrained optimization with geometric reasoning, and whenever needed, with production engineers' involvement.

2. Basic model and planning workflow

As for key modelling *assumptions*, the target product, even if its representation contains conflicts (either due to its approximate nature or to the existence of elastic parts) is taken realizable. A monotonous, two-handed process is considered, where each task performs the assembly of two subassemblies or parts. There is no prior assumption on the base and moved parts. The *liaison graph* of the product [2] is assumed to have a tree-structure, however,

there are no pre-determined subassemblies. Any task may use alternative fixtures and tools, with fixed execution times. While fixtures have an approximate geometric model (which should be refined later by fixture design), models of tools like grasping devices, wrenches, screw drivers are taken from catalogues. A human hand can also serve as a tool. Movement of objects–typically parts, subassemblies, tools, but also auxiliary inspection devices–is performed in two phases: along an arbitrary path to the so-called *near position*, and from here a fine movement puts the object into its *goal position*. *Collision avoidance* throughout the whole movement is required.

The planning *workflow* has four main stages (see also Fig. 1):

1. *Analysis and problem definition*: interpretation of the CAD model, identification of features along with the specification of resource alternatives, generation of initial planning constraints.
2. *Macro planning*: solution of task sequencing and resource assignment, optimized for minimal changeovers.
3. *Micro planning*: validation of macro-level plans, collision checking of intermediate configurations, fixtures and tools. Path planning to near positions, validation of paths. In case of infeasibility, constraints feedback to macro planning.
4. *Postprocessing*: generation of robot codes, manipulator control programs as well as operator work instructions.

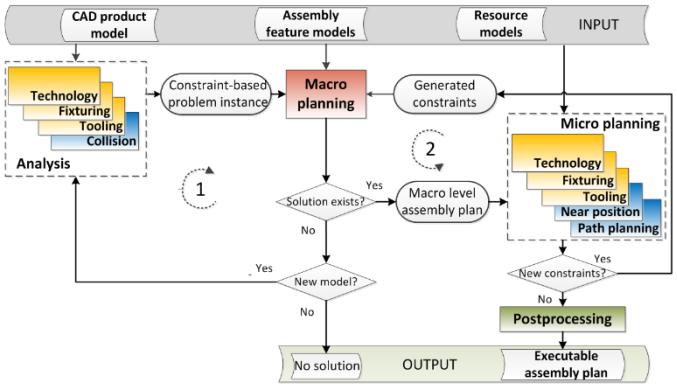

Fig. 1. The overall assembly planning workflow.

Following the *least commitment* principle, macro planning starts only with constraints which must be met by *any* assembly plan: ordering of tasks together with the assignment of applicable fixtures and tools is arranged so that connectivity relations between parts can be physically realized and maintained with having as few changeovers as possible. The so-called *Benders decomposition scheme* provides a formal connection between macro and micro planning by way of augmenting the master problem of macro planning, whenever validity checks require, with constraints generated by micro planners [11]. Again, the new constrains on task sequencing and/or resource assignment must be met in *any* solution, hence the planning process is sound. There are almost necessarily issues which cannot be formalized, calling for a room for engineering criticism and decisions. Hence, this cautious attitude requires iterations. There are two *iteration* cycles in the workflow (see Fig. 1): (1) Whenever macro planning fails to find a solution, there is a way back to resume planning with a new technological (i.e., feature-based) interpretation of the product. (2) Macro plans are validated from a number of aspects, by specialized micro planners. In case of infeasibility, micro planners generate new constraints as input for macro planning.

3. Representation and analysis

3.1 Geometric modelling and basic procedures

For representing all objects involved in the assembly, triangle *mesh models* are used. This provides only an *approximate* representation for the product and its parts, deviating from their exact geometries and without their explicit functions and relations. Hence, e.g., a screw, with its connector function (often exploited in assembly models) is not explicitly distinguished. On the other hand, having this generic representation as a kind of skin model [12] for any (and all) objects concerned during the planning process, one can employ generic and very efficient collision and proximity test methods for identifying part relations and assembly features, generating constraints as for task precedences and the use of resources, as well as for path planning [13]. This model serves visualization as well, which is a crucial point in supporting engineering interaction. However, geometric reasoning must be robust towards the resolution of the mesh model.

Given a set of mesh objects O, for two meshes $o_i, o_j \in O$ and a homogenous transformation matrix Θ applied to o_j, a *collision detection procedure* $C(o_i, o_j, \Theta) = c_{i,j}^{\Theta}$ returns the vertices of those triangles from both meshes which are in contact. The returned vertices forming a *Collision Point Cloud* (CPC) can be assigned either to o_i or o_j, making the sets $c_{i,j,1}^{\Theta}$ and $c_{i,j,2}^{\Theta}$ respectively. Furthermore, a *proximity query procedure* is defined as follows: $Q(o_i, o_j, \Theta) = q_{i,j}^{\Theta}$, returning the Euclidean distance between the two closest points of the meshes and 0 if they collide.

3.2 Building the connectivity and liaison graphs

Given a set of mesh objects $o \in O$ representing the parts, their *connectivity graph* is defined as $G_c(V, E)$ $V(G_c) = \{o\} \in O$ where edges E are given by the following adjacency matrix:

$$A_{G_c} = \left[a_{G_c i,j}\right] = \begin{cases} 0, \ if \ q_{i,j}^{I} \geq r \\ 1 \ , if \ q_{i,j}^{I} < r \end{cases}$$, where I is a 4x4 identity matrix and $r \geq 0$ is a threshold for the proximity. Ideally, r should be zero, which would mean collision detection and thus the connectivity matrix showing only actual contact between two meshes. However, so as to handle imperfect mesh models, a distance threshold is applied for detecting connectivity.

The connectivity graph represents the relationship between parts in their assembled state; with its connected pairs of nodes denoting potential pairs of components to be assembled. Therefore assembly features can be assigned to the edges of G_c. The feature-based approach allows for a rich representation of the assembly tasks, taking into account parts, assembly technology, (dis)assembly directions, physical parameters, etc. Each feature model contains information on the applicable set of fixtures and tools as candidate resources, and also positioning information for each candidate equipment. Extending the model in [11] a feature for task t is formulated as $F_t : \langle \rho_t, a_t, b_t, \Theta_t, P_t, k_t \rangle$ where ρ_t is the feature type, a_t and b_t are two parts (base and moved, yet unspecified) concerned with Θ_t defining a homogenous transformation matrix describing the joining. Movements in the micro-world of the feature are linear which can be further extended with interpolation of a spiral movement (e.g., in case of a screw) defined by the additional parameters in P_t. The candidate resources together with their positional transformation matrices are denoted by k_t.

Taking a connectivity graph G_c and the features, a liaison graph representation G_L can be defined as a spanning tree of G_c, where each edge stands for an assembly task. Note that a single task can realize multiple connections at the same time. The feature specification transforms the connectivity graph to a liaison graph, but this is non-trivial as there are many possible technological interpretations of the same design. Hence, assembly-specific expertise is needed here to warrant that the feature-based interpretation and the resulting liaison graph define a feasible problem. Pattern-based heuristics and best-practices are applied (e.g., assembling similarly sized parts), though human interaction is still vital in this step. A working example of a ball valve assembly is shown in Fig. 2 with a 3D exploded view along with its connectivity graph and a possible design interpretation shown by the liaison graph representation (denoted by bold lines).

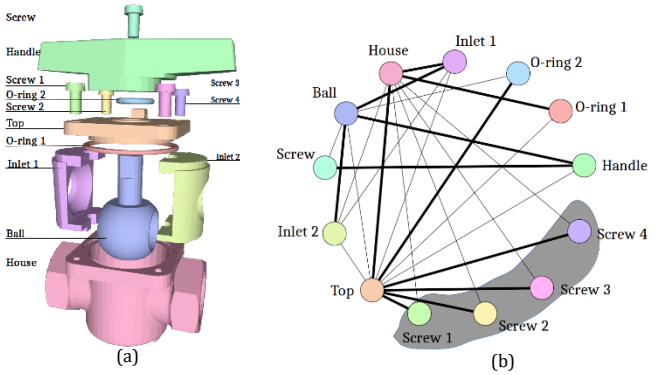

Fig. 2. Mesh model (a) and the connectivity and liaison graph (b) of the working example including a possible composite feature for screws.

3.3 Geometric reasoning for disassembly directions

Having the features and the liaison graph defined, identification of the feature parameters is a tedious task, which greatly affects the efficiency of any feature-based planning model. Therefore, efforts on supporting parameter extraction can be of much use during the phase of analysis. A heuristic approach introduced for extracting linear disassembly directions in [14] was further developed and is detailed below. It applies assembly by disassembly for triangle meshes. Given two meshes $(o_i, o_j) \in O$ a suitable direction is to be found based on their contacts. In order to discover these vertices collision detection is applied. The set of contacting vertices between the two meshes are found as follows: $c_{i,j,1} = \{c_{i,j,1}^{\Theta_1} \cup \ldots \cup c_{i,j,1}^{\Theta_k}\}$ $c_{i,j,2} = \{c_{i,j,2}^{\Theta_1} \cup \ldots \cup c_{i,j,2}^{\Theta_k}\}$, where Θ_k denotes linear translations in k different directions sampled from a unit sphere.

The method finds the most suitable translation for disassembling o_j from o_i by evaluating each direction defined by the transformations Θ_k. Since the CPCs were acquired by moving o_j therefore $c_{i,j,1}$ and $c_{i,j,2}$ are referred to as *static* and *moved point cloud*, respectively. The theoretical goal here is to find a clear line of sight for every point in the moved point cloud along a given direction. The CPCs, however, are just as good of a representation as the initial meshes they are acquired from. Therefore aggregation is used to cancel out the noise (especially intersections between the meshes), where the moved point cloud is represented by a given number (l) of different center points $V_{i,j,l}$. These are obtained using the cluster centers of a k-means clustering performed on $c_{i,j,2}$. When evaluating a direction Θ_k for a centre point $V_{i,j,l}$, the radius of the largest cylinder is taken that is centered around $V_{i,j,l}$ with an axis defined by Θ_k and does not contain any points from $c_{i,j,1}$. This radius is used as an evaluation metric $M_{i,j,l}^{\Theta}$. The overall evaluation for k different directions Θ_k and l different center points $V_{i,j,l}$ is called the *Disassembly Direction Metric* (DDM) and is calculated as follows:

$$M_{i,j}^{\Theta_k} = \min_l M_{i,j,l}^{\Theta_k}$$

and the most suitable (dis)assembly direction is hence:

$$\Theta_f = \underset{\Theta_k}{\operatorname{argmin}}\, M_{i,j}^{\Theta_k}.$$

Θ_f can be applied also for determining the *insertion depth* between the meshes by projecting the points in the two CPCs along the selected direction and subtracting the minimal value in $c_{i,j,2}$ from the maximal value in $c_{i,j,1}$. The example in Fig. 3 shows also the link between DDM and the mesh model resolution.

The above parameter extraction method only returns translational parameters and, in order to utilize the details of the feature-based representation, additional parameters, such as the thread angle or depth of a screw, or the safety distance for inserting parts have to be manually included. Moreover, in the case when there is a repeating pattern of assembling identical parts such as a set of bolts or screws, the definition of composite features is helpful in reducing the search space, even though such groups are now defined by human interaction (see Fig. 2.). Nevertheless, supporting mixed-initiative problem solving is one of the key goals of the introduced workflow as it can be hardly expected that analysis will cover every domain-specific aspect of process planning by means of automated methods.

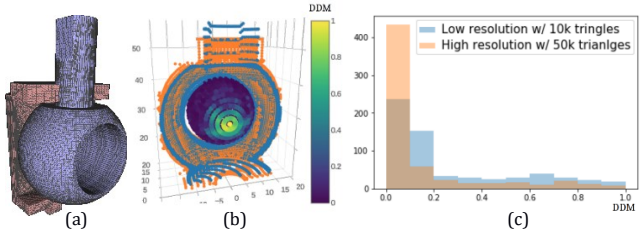

Fig. 3. Mesh model of *Ball* and *Inlet_1* (a), their CPCs and disassembly direction metrics, displaying the selected direction as the brightest (b), and the distribution of DDMs at two different mesh resolutions (c).

4. Micro-level planning

The goal of micro-level evaluation is to provide detailed feasibility analysis of the macro plan with regard to technology, tooling, fixturing and movements, and in case of collisions to generate additional feasibility cuts in form of constrains for macro planning (see Fig. 1). The constraints are disjunctive and must be satisfied by any valid assembly plan. This is done by using the e*xtended liaison graph* G_{ELG} which captures the assembly configuration at the time of performing a specific task t. Besides the part-to-part relations of G_L, G_{ELG} contains also the tool-to-part and fixture-to-part relations [11]. The collision-based evaluation has two phases: (1) *near positioning*, which consists of realizing the movement defined by the assembly feature, and (2) *path planning*, when the part and its tool are moved from an external pick-up place to the near position. Since near positioning is more constrained and implies no search because its parameters are completely specified by a given feature, it is performed first.

4.1. Near positioning

Evaluation of near positioning is applied feature by feature for a plan which specifies already the order of tasks, and for each feature the fixed components together with fixture and the moved component together with its tool. These two sets of objects $O_A, O_B \subset O$ are tested by a method $e_n(O_A, O_B, \Theta_N, \Theta_G, P)$ where Θ_N and Θ_G specify the homogenous transformation matrices of the near and goal positions of the moved part in O_B, and P gives additional parameters of the feature to be realized. E.g., in case of screwing the additional parameters of *thread depth* and *lead* are taken into account to handle the change in rotation into Θ_G. The evaluation is carried out by using continuous collision check for each pair $(o_i, o_j) \in O_A \times O_B$ moving o_j along the path defined by Θ_N, Θ_G, P, thus e_n returns the combined result of these checks. The task cannot be realized if the collision detection fails for any pair of $(o_i, o_j) \in O_A \times O_B$. By identifying the colliding objects and using the G_{ELG} the required feasibility cuts can be generated and fed back to macro planning.

The above evaluation cannot be applied for parts which are directly connected in G_c because they are (intentionally or due to the mesh model representation) in collision in their goal positions. A typical example is an object inside a box with a top (see Fig. 2 *Inlet_1-House-Top*). Similarly to the approach of Local Translational Freedom (LTF) [15], these cases are simplified to translational motions defined by the transformation of the feature. Thus a method is required for evaluating a translational movement defined by a homogenous transformation between two triangle meshes. For every pair of directly connected nodes in G_c, DDM is applied and provides this evaluation, assuming that it was performed in position Θ_G.

4.2. Path planning

Micro-level evaluation also checks if there exists a collision-free path from the pick-up location Θ_s for moved objects (parts, subassemblies, tools) $o_j \in O_B$ to the near position Θ_N. This is a path planning problem in the virtual space of the workcell including the already present objects $o_i \in O_A$. Due to changing assembly configurations and relatively low number of macro-level iterations, a single-query algorithm was used. The *Rapidly-exploring Random Trees* (RRT) algorithm is well suited for being combined with distance- and collision-query algorithms over triangle mesh models of objects [16]. For each task, path planning is applied as a method $e_p(O_A, O_B, \Theta_s, \Theta_e)$ which returns a feasible path if it exists. When there is no feasible path the generated feasibility cut is usually weaker than those for near positioning since a failed evaluation in this case implies a complete set of failed paths with possibly various collisions.

5. Implementation and case studies

The planning workflow was implemented in Python, using FICO Xpress 8.0 in the macro planning phase and FCL for collision and proximity queries [13]. Table 1. shows key execution statistics for the product family of the working example, run on an average PC. For a small-sized problem, initialization (including the creation of collision models and calculation of DDMs) has the largest time consumption and it also scales up with the number of triangles. Fig. 4(a) presents an intermediate assembly plan with a single changeover which proved to contain several conflicts during the micro planning phase, while Fig. 4(b) shows an optimal and executable plan (albeit with two changeovers), after complying with eight new feasibility cuts generated as a result of geometric reasoning. Note that in these experiments macro planning started with an almost empty, under-constrained model which was though well prepared for geometric calculations.

Table 1. Results of experiments on a 4-element ball valve product family.

	# parts	# triangles	# constraints (init)	Running time (init) [s]
Ball valve #1	9	199,886	11(2)	10.85 (8,62)
Ball valve #2	11	279,850	11(2)	13.80 (10,14)
Ball valve #3	13	303,880	11(2)	23.98 (20.42)
Ball valve #4	13	545,478	11(2)	35.57 (30.86)

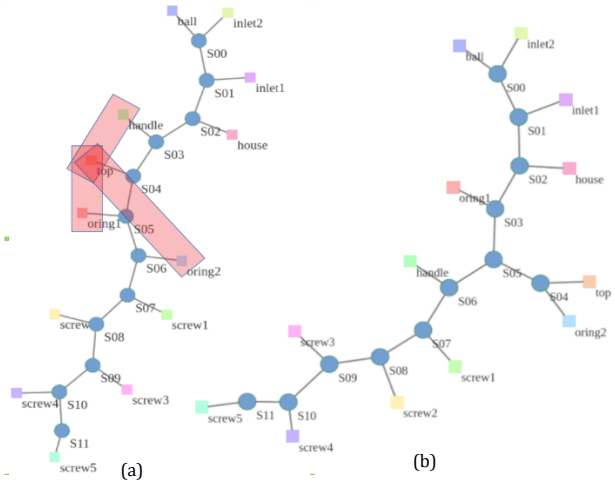

Fig. 4. A shorter but infeasible plan with conflicts highlighted (a) and a feasible optimal plan (b) generated for the working example.

Experiments with more complex assemblies have also confirmed the viability of the approach. The most intricate case study was performed on an assembly from the automotive industry consisting of 29 parts, represented by over 3 million tringles. Solution time took 331 s including 57 s for initializing the model. Micro planning generated 8 new disjunctive constraints on the fly. The resulting assembly plan is shown in Fig. 5.

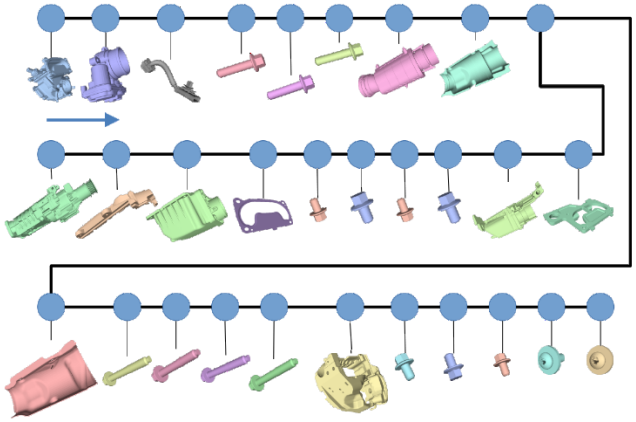

Fig. 5. Assembly plan generated for complex automotive component.

6. Conclusions

The paper presented the extension of a decomposition-based assembly planning model, focusing on geometric reasoning which interleaves with constrained optimization and human interaction in the workflow. The approach is cautious: it assumes neither perfect models nor complete knowledge at the outset of planning. In the end, against all the above difficulties and eventual human involvement, the workflow warrants an executable and optimized assembly plan or, alternatively, proves that the actual problem has no solution. This remains so even if some modules used in the phases of analysis and micro planning are changed or augmented, as it is planned in the future to cover the recognition of composite features and specific feature parameters, to support detailed fixture design, and to generate robot codes and work instructions.

Acknowledgement

This research has been supported by the EU H2020 SYMBIO-TIC No. 637107 and the GINOP-2.3.2-15-2016-00002 grants. The authors thank András Kovács for his advice.

References

[1] Wilson RH, Latombe JC (1994) Geometric reasoning about mechanical assembly. *Artificial Intelligence* 71:371–396.

[2] Hu SJ, Ko J, Weyand L, ElMaraghy HA, Lien TK, Koren Y et al. (2011) Assembly system design and operations for product variety. *CIRP Annals - Manufacturing Technology* 60(2):715-733.

[3] Krüger J, Wang L, Verl A, et al. (2017) Innovative control of assembly systems and lines. *CIRP Annals - Manufacturing Technology* 66(2):707-730.

[4] Fantoni G, Santochi M, Dini G, Tracht K, Scholz-Reiter B, Fleischer J, Lien TK, Seliger G, Reinhart G, Franke J, Hansen HN, Verl A (2014) Grasping devices and methods in automated production processes. *CIRP Annals - Manufacturing Technology* 63(2):679-701.

[5] Ghandi S, Masehian E (2015) Review and taxonomies of assembly and disassembly path planning problems and approaches. *Computer-Aided Design* 67-68:58-86.

[6] Pellegrinelli S, Orlandini A, Pedrocchi N, Umbrico A, Tolio T (2017) Motion planning and scheduling for human and industrial-robot collaboration. *CIRP Annals - Manufacturing Technology* 66(1):1-4.

[7] Leu MC, ElMaraghy HA, Nee AYC, Ong SK, Lanzetta M, Putz M, Zhu W, Bernard A (2013) CAD model based virtual assembly simulation, planning and training. *CIRP Annals - Manufacturing Technology* 62(2):799-822.

[8] Tan C, Hu SJ, Chung H, Barton K, Piya C, Ramani K, Banu M (2017) Product personalization enabled by assembly architecture and cyber physical systems. *CIRP Annals - Manufacturing Technology* 66(1):33-36.

[9] Tolio T, Bernard A, Colledani M, Kara S, Seliger G, Duflou J, Battaia O, Takata S (2017) Design, management and control of demanufacturing and remanufacturing systems. *CIRP Annals - Manufacturing Technology* 66(2):585-609.

[10] van Holland W, Bronsvoort WF (2000) Assembly features in modeling and planning. *Robotics and Computer Integrated Manufacturing* 16(4):277-294.

[11] Kardos Cs, Kovács A, Váncza J (2017) Decomposition approach to optimal feature-based assembly planning. *CIRP Annals - Manufacturing Technology* 66(1):417-420.

[12] Schleich B, Anwer N, Mathieu L, Wartzack S (2014) Skin model shapes: A new paradigm shift for geometric variations modelling in mechanical engineering. *Computer-Aided Design* 50:1–15.

[13] Pan J, Chitta S, Manocha D (2012) FCL: A general purpose library for collision and proximity queries. *IEEE Int. Conf. Robotics and Automation*, pp. 3859-3866.

[14] Kardos Cs, Váncza J (2018) Application of generic CAD models for supporting feature-based assembly process planning. *Procedia CIRP* 67:446-451.

[15] Romney B, Godard C, Goldwasser M, Ramkuma G (1995) An efficient system for geometric assembly sequence generation and evaluation. *ASME Conference on Computers in Engineering*, pp. 699–712.

[16] LaValle SM, Kuffner JJ (2001) Randomized kinodynamic planning. *Int. J. of Robotics Research* 20(5):378–400.